XML 数字签名技术在远程考试中的应用研究

Application of the XML Signature to The Long – Distance Examination

黄　寅　（宁波广播电视大学 信息工程与技术系 浙江宁波 315016）

摘要：实施基于网络的远程考试是远程教育的快速发展的必然趋势，本文探讨数字签名技术在保证考试数据的安全性方面的研究。首先阐释数字签名的原理，分析 XML 数字签名的特点，最后设计了一个 XML 数字签名在远程考试中的应用实例。

关键词：数字签名 XML 信息安全 RSA PKI

1　远程考试数据安全性需求概述

从长远的角度看，基于 Web 的远程考试今后将成为现代远程教育的一个重要组成部分。在开放环境下进行考试，文档的明文传输是非常危险的，特别是对诸如试卷、答卷等敏感数据。如何保证考试的有效性、机密性、完整性和可靠性是远程考试的关键所在。远程考试数据安全性需求如下：

（1）有效性。确保试卷和答卷数据都是有效的，试卷必须是考试机构发布的，答卷必须是确定的考试对象提交的。

（2）机密性。考试数据无论是试卷还是答卷都属机密文档，在网络传输中需要采取相应的加密措施以防非法用户截取。

（3）完整性。防止用户对网络中传输的数据进行无意或恶意的修改、复制和删除，防止数据丢失。保证整个考试环节中电子试卷和答卷数据都是完整的，且考试用户在提交答卷后不可以篡改数据。

（4）可靠性。对数据源进行验证，以确保数据由合法用户发出，防止数据发送方在数据发送后否认其行为，同时也要防止接收方在收到数据后否认曾收到过此数据及篡改数据。

以上这些远程考试中对数据安全性需求，可以考虑采用数字签名作为远程考试安全性提供有力的支撑。本文根据远程考试基于网络环境的特点，从技术角度出发，提出以数字签名技术结合 XML 技术，即 XML 数字签名技术来解决远程考试中安全性的关键问题。

2　相关技术介绍

2.1　数字签名技术

目前数字签名采用较多的是基于 PKI 的公钥密码技术，如基于 RSA Date Security 的 PKCS（Public Key Cryptography Standards）、Digital Signature Algorithm、x.509、PGP（Pretty Good Privacy）。建立在公共密钥体制基础上的数字签名在工作时，首先发送方对信息施以数学变换，所得的信息与原信息唯一对应；在接收方进行逆变换，得到原始信息。只要数学变换方法优良，变换后的信息在传输中就具有很强的安全性，很难被破译、篡改。这一个过程称为加密，对应的反变换过程称为解密。

以应用最广泛的 RSA 算法为例说明公钥算法的数学原理。为了产生 2 个密钥，选取 2 个大素数 p 和 q，计算乘积 n = pq。然后随机选取加密密钥 e，使 e 和 (p −1)(q −1) 互素，最后用欧几里德扩展算法[1]计算解密密钥 d，以满足 ed = 1 mod ((p −1)(q −1)) 可以证明 d 和 n 也是互素的，则 e 和 n 是公开密钥，d 是私人密钥。2 个素数 p 和 q 不再需要，但绝不可泄漏。RSA 算法可以简化为表 1，其中 m 为待加密的消息，c 为用 RSA 算法加密消息 m 后得到的密文。

由于 e 和 n 必须公开，如果通过分解 n 得到 p 和 q，就可以利用欧几里德扩展算法从公开密钥 e 计算出私人密钥 d，所以 RSA 算法的安全强度等价于对 n 进行质因数分解的难度。目前在数论中，对大数进行质因数分解尚没有有效的算法，因此只要选择足够大的 p 和 q，RSA 算法的安全性是可信的。但在数学上从未证明过必须分解 n 才能从 c 和 e 计算出 m，也可以通过猜测 (p −1)(q −1) 的值来攻击 RSA，但这种攻击没有分解 n 容易[2]。密码分析者还可能尝试采用其他的方法绕过大

数质因数分解的困难去破解 RSA 算法，然而目前在密码学领域，RSA 算法被公认为是非常安全的。

表 1　RSA 算法

公开密钥	n：2 个大素数 p 和 q 的乘积（p 和 q 必须保密） e：与（p－1）（q－1）互素
私人密钥 加密过程 解密过程	$d: e^{-1} \bmod ((p-1)(q-1))$ $c = m^e \bmod n$ $m = c^d \bmod n$

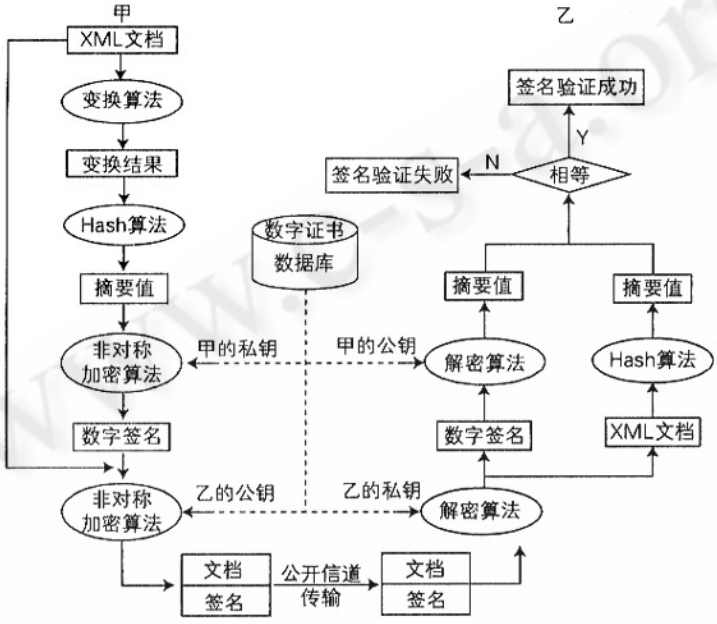

图 1　数据签名的生成与验证

数字签名的实施主要分为两个阶段，即生成签名和签名验证，如图 1 所示。甲方首先将要签名的文档经过一系列的变换算法生成一个文档摘要值，并对此摘要值运用 RSA 算法以甲的私钥作参数生成数字签名，若被签署的文档是试卷等机密文件，还需对文档用乙的公钥进行 RSA 加密后再传输。加密后的文档和数字签名一起在公开信道传输，到达乙方后，乙首先用自己的私钥解密文档，并计算文档的摘要值，然后用甲的公钥解密数字签名，还原出摘要值，将两个摘要值进行比较，若值相等则签名验证成功，否则验证失败。

2.2 XML 技术

2.2.1 XML 的基础标准

互联网国际标准组织 W3C 针对 XML 应用中的公用特征、方法或规则，制定了一些 XML 的基础标准。XML 标准体系的框架主要有以下三个要素组成：

（1）模式，用来描述 XML 文档合法结构、内容和限制。模式提供创建 XML 文档必要的框架，详细说明 XML 文档中不同元素及其属性的有效结构、限制和数据类型。XML Schema 是 XML 体系中正式的类型语言，同 XML 规范、Namespace 规范一起成为 XML 体系的坚实基础。

（2）可扩展样式语言 XSL，由两部分组成，一部分是 XSLT 和 XPath，定义了如何将一个 XML 文件转换为其它的可供显示或打印的文件格式；另一部分是 FO，为转换后文件中的各个对象定义语义和显示方式。XSL 偏向于动态，可以利用 XSL 实现排序、过滤等较为复杂的功能。

（3）可扩展链接语言 XLL，用于处理 XML 文档间的链接。它不只是单个目标的连接，而可以是非常复杂的链接，包括双向链接以及间接链接。XML 作为一种元标语言，提供的是描述具体应用语言的基本方法。针对具体的应用领域还有相应的应用标准。例如，具体的置标表示的语义，附加的语法约束等。

2.2.2 XML 数字签名规范

XML 数字签名充分利用了 XML 语言本身及其相关的标准，譬如 Xpath、Xpoint、XSLT 强大的表达能力和扩展能力，能够在元素级这样较细的粒度上实现对文档的特定部分进行签名，并且签名后的文档仍然是一个 XML 文档。2001 年 8 月由 IETF 和 W3C 共同组建的 XML Signature 工作组公布了 XML 数字签名规范。这一规范定义了与 XML 语法兼容的数字签名语法，可描述数字签名本身和签名的生成与验证过程。作为一个安全有效的数字签名方案，该规范提供了数字签名的完整性、签名确认、不可抵赖性。

3　XML 签名在远程考试中的应用

3.1 XML 签名方案的设计

下面以远程考试中学生对答卷数据进行 XML 数字签名为例，来探讨 XML 数字签名在远程教育中的应用。基于 PKI/CA 中心来探讨远程考试数据传递流程的设计与实现，首先高校需要构建自己的 PKI/CA 认证

中心，每个学生和教务管理部门都有自己数字证书，并且这些证书可以通过网络方便地获取。

签名方案的体系结构如图 2 所示，客户端使用 XML 浏览器访问 Web 服务器，通过 Web 服务器访问 CA 服务器或信息服务器。Web 服务器提供文档中转，发布和访问 CA 服务器与信息服务器；CA 服务器提供安全注册，安全信息发布服务（发布证书和撤销列表），信息服务器中存储各种格式的数据，譬如：数据库、XML 资源和各种应用等信息。客户通过客户端计算机向 CA 服务器进行注册获取自己的证书，或者通过 Web 服务器查询其他客户的证书和撤销列表来获取它们的证书。客户对生成的 XML 文档进行签名后，通过 Web 服务器提交到信息服务器分类存储；其他被授权的用户通过 Web 服务器查询到相应的信息，从签名者的证书获取他的公钥，就可以对已签名的 XML 文档进行验证。

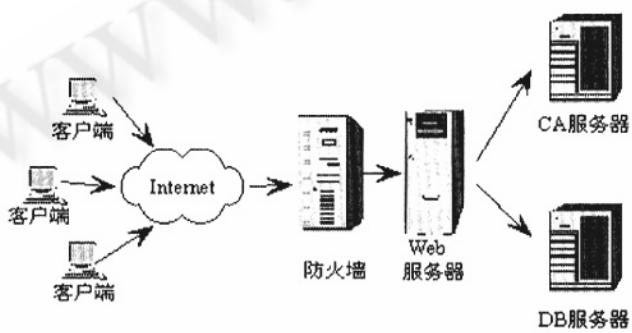

图 2　签名方案的体系结构

XML 支持嵌入式签名（Enveloped Signature）、装式签名（Enveloping Signature）和分离式签名（Detached Signature），三种方式各有所长，嵌入式签名支持对同一文档的多重签名，封装式签名和封装式签名可实现对多个文档同时签名。嵌入式签名和封装式签名比较直观，易理解，分离式签名结构比较灵活。远程考试系统中的电子试卷和学生答卷的文件格式可能多样性，灵活度与可扩展性是我们设计 XML 数据签名时需要考虑的因素，因此我们选择分离式签名，电子试卷文件与签名文件彼此独立，通过签名元素 <Reference> 中的 URI 属性关联。

3.2 xml 签名的生成与验证

XML 数字签名过程如下：首先将需进行数字签名的 XML 转换为 DOM 对象；获得同时包括公钥和私钥的密钥对。用私钥对内容进行数字签名，公钥信息可嵌入 XML 数字签名的 <KeyInfo> 元素内发送给接收方，以便用来验证其中的数字签名。根据需要签名的 DOM 文档、用来签名的私钥以及用于验证的公钥，创建一个 Signer 对象。Signer 是数字签名过程中的主要对象，Signer 对象根据 W3C XML Signature 规范对 XML 文档签名；指定 XML 中的哪一部分需要签名。通过增加到 Signer 对象的引用，可以指定数字签名的 XML 位置。增加引用的方法是调用 Signer 对象中的 addReference 方法。需要签名的元素的 XPath 作为一个参数提供给 addReference；最后一步是对 XML 签名，通过调用 Signer 对象中的 sign 方法来完成。当调用 sign 方法时若不传递任何参数将生成的封外签名，即被签数据与 XML 签名相互独立，调用带有 XPath 参数的 sign 方法，并且数字签名放置在文档中，则生成封内签名，被签数据将嵌入到 XML 签名。

XML 数字签名的验证包括签名校验（Signature Validation）和引用检验（Reference Validation）两个部分。签名校验就是对 <SignedInfo> 元素的校验。它利用 <CanonicalizationMethod> 元素中指定的规范化方法，对 <SignedInfo> 进行规范化处理，然后用 <SignatureMethod> 指定的算法重新计算其摘要值，并用 <KeyInfo> 中或已知的签名者公钥来计算最后的签名值。如果该值与 <SignatureValue> 中的数值吻合，则这一步骤的校验通过。引用检验的目的是确保被签署对象没有被做任何的修改，而签名确认的目的则是保证签署人身份的真实性。引用检验的过程是先对 <Reference> URI 属性中的签名对象实施 <Tranform> 中指定的变换算法（如果有 <Tranforms> 元素的话），然后利用 <DigestMethod> 指定的单向哈希算法计算每个参考的摘要值，并与 <DigestValue> 中的数值进行比较，如果对所有的参考对象该值都是吻合的，则参考检验成功。当上述两步都成功时，该 XML 数字签名的校验才算通过。

XML 数字签名的校验过程如下：(1)根据 <CanonicalizationMethod> 对 <SignedInfo> 做规范化处理，并对于在 <SignedInfo> 中的每一个 <Reference> 元素，通过 URI 等方式从中获得需要被计算摘要的数，应用 <Transforms> 规则进行相应的转换；再根据 <Di-

gestMethod > 计算出摘要值并与 < DigestValue > 比较,如果不同,则认证失败,从而签名确认失败,这说明被签署的对象已经被修改过了;否则进行下一步。(2)对 < SignedInfo > 按照 < CanonicalizationMethod > 做计算。(3)从 < KeyInfo > 中或者外部源得到密钥(Key)的相关信息。(4)通过 < SignatureMethod >,使用密钥信息和已经在第二步规范化的 < SignedInfo > 元素进行计算,并将计算结果与 < SignatureValue > 比较。如果相同,则校验成功,数字签名有效;否则签名认证失败。

3.3 编程实现

本文利用 IBM 提供的 XML Security Suite 安全组件 Java 库中的 XML 数字 API: XSS4J 和 XPath 处理器 Apache Xalan – J_2_5_1 来实现 XML 数字签名。

3.3.1 生成签名

```
//从密钥库中获取私钥
Key key = keystore. getKey( alias,keypass) ;
//用 Java 的 r 类生成一个待填充的签名 DOM 树,并配置摘要//算法、规范化方法和签名算法。
TemplateGenerator siggen  =new TemplateGenerator
( doc, XSignature. SHA1, Canonicalizer. W3C2, SignatureMethod. RSA) ;
//对生成的签名对象 DOM 树添加参考元素,其中变量 URI 中//保存着文件的路径及文件名 siggen. addReference( siggen. createReference( URI) ) ;
// 对被签文档进行适当的数据变换 addXPathTransform( String expression) ;
Element signaturElement = signatureGen
. getSignatureElement( ) ;
KeyInfo. insertTo( signaturElement) ; //插入签名元素
//生成签名上下文实例
SignatureContext  sigContext  =  new  SignatureContext
( ) ;
// 用私钥为摘要生成数字签名
SigContext. setDResovler( new AdHocDResolver( doc) ) ;
SigConrext. sign( sigElement,key) ;
```

3.3.2 签名验证

```
    //创建 Java Key Store 对象
KeyStore keystore = KeyStore. getInstance( "JKS" ) ;
//密钥存储路径及访问密码
Keystore. load( new FileInputStream( keystorepath) ,
storepass) ;
X509Certificate cert = ( X509Certificate)
keystore. getCertofocate( alias) ;
//从密钥识中获取公钥
Key key = cert. getPublicKey( ) ;
//用公钥重新成被签文档的摘要
SignatureContext  sigContext  =  new  SignatureContext
( ) ;
//签名验证
Validity validity = sigContext. verify( sigElement,
key) ;
```

服务器端调用签名引擎完成数字签名后,将签名信息(包括摘要、数字签名本身和公钥参数)嵌入 XML 响应消息中发往客户端,客户机接收 XML 文档并将摘要、数字签名和公钥参数解析成应用程序数据,然后客户机使用嵌入的密钥参数重新构造公钥,并采用同样的散列算法计算接收消息的摘要并与来自明文的摘要进行比较,以"验证"方式调用签名引擎,并传递摘要、签名和公钥来验证签名。

4 总结

数字签名的应用范围十分广泛,除了上述在远程考试服务中有应用外,还可用于远程金融交易、远程医疗、电子商务、自动模式处理等等,XML 数字签名有着十分广泛地应用前景。本文分析了 XML 与数字签名原理并讨论了其在远程教育领域中的一个应用,目前 XML 数字签名仍是一个较新的研究热点。

参考文献

1 Merkle RC. A Certified Digital Signature[Z]. Advances in Cryptology – Crypto,2001. 218 – 238.

2 Wu CK Wang X M. Determination of the True Value of the Euler Totient Function in the RSA Cryp tosystem from a Set of Possibilities. Electronics Letters,1993, 29 (1):84 ~ 85.

3 Didier Martin. XML 高级编程[M]。机械工业出版社,2001 年 1 月出版。

4 Charles F. Goldfarb. XML 实用技术[M]。清华大学出版社,1999 年 9 月出版。