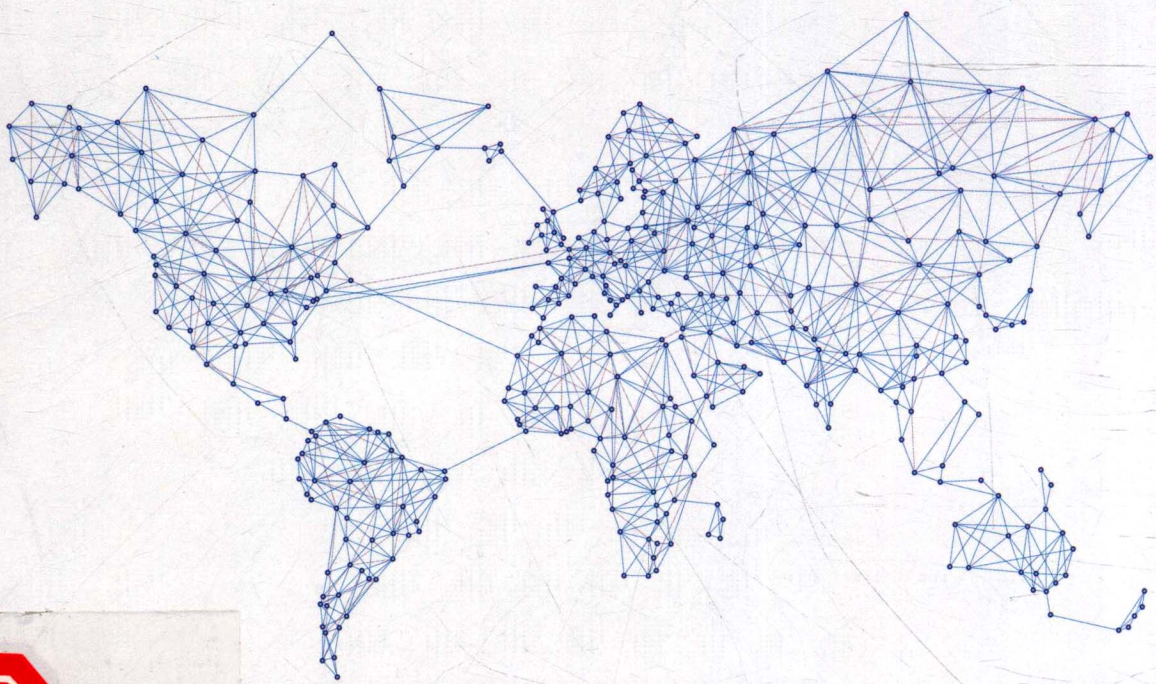
# 传媒信息安全技术

## Media Information Security

冯柳平　刘华群　编著



科学出版社

# 传媒信息安全技术

冯柳平　刘华群　编著

科学出版社

北京

## 内 容 简 介

本书介绍传媒信息安全技术，全书共分为 6 章。第 1 章 XML 安全技术，主要介绍 W3C、OASIS 和 IETF 等推出的一系列 XML 安全标准和规范。第 2 章数字版权管理技术，主要介绍 DRM 系统模型、关键技术，以及 DRM 标准。第 3 章云计算与云安全，分析了云计算所面临的安全问题，重点介绍了云计算的数据安全、身份认证及访问控制等技术。第 4 章云计算安全标准及其测评体系，介绍了国内外云计算安全标准建设情况，重点讨论了云计算安全框架和评测框架。第 5 章物联网技术及安全，介绍了物联网的基本概念和体系结构，分析了物联网安全威胁，并介绍了物联网安全模型。第 6 章移动互联网安全，介绍了移动互联网的基本概念和架构，对移动终端安全、位置隐私保护以及数据传播模式和安全管控的相关研究进行了介绍。

本书可作为高等学校计算机科学与技术、信号与信息处理相关专业高年级学生及硕士研究生信息安全技术课程的教材，也可作为相关领域研究人员的参考书。

**图书在版编目（CIP）数据**

传媒信息安全技术/冯柳平，刘华群编著. —北京：科学出版社，2016.6

ISBN 978-7-03-049211-1

I. ①传… II. ①冯…②刘… III. ①传播媒介－信息安全－安全技术 ①IV. ①G206.2

中国版本图书馆 CIP 数据核字（2016）第 147040 号

责任编辑：赵丽欣 / 责任校对：马英菊

责任印制：吕春珉 / 封面设计：东方人华平面设计部

科学出版社 出版

北京东黄城根北街 16 号

邮政编码：100717

http://www.sciencep.com

厚诚则铭 印刷

科学出版社发行　　各地新华书店经销

*

2016 年 6 月第　一　版　　开本：787×1092　1/16

2016 年 6 月第一次印刷　　印张：9 3/4

字数：236 800

**定价：36.00 元**

（如有印装质量问题，我社负责调换〈厚诚则铭〉）

销售部电话 010-62136230　编辑部电话 010-62134021

# 前　言

近年来，传媒技术飞速发展，传媒产业形成了平面媒体、网络媒体、移动媒体等全方位覆盖的全媒体传播格局。传媒信息安全已经成为传媒产业信息化发展中必须面对的问题，加强传媒信息安全技术研究，对推进传媒产业的发展有着重要的作用。

本书介绍传媒信息安全技术，全书共分为 6 章。第 1 章 XML 安全技术，主要介绍 W3C、OASIS 和 IETF 等推出的一系列 XML 安全标准和规范。第 2 章数字版权管理技术，主要介绍 DRM 系统模型、关键技术，以及 DRM 标准。第 3 章云计算与云安全，分析了云计算所面临的安全问题，重点介绍了云计算的数据安全、身份认证及访问控制等技术。第 4 章云计算安全标准及其测评体系，介绍了国内外云计算安全标准建设情况，重点讨论了云计算安全框架和评测框架。第 5 章物联网技术及安全，介绍了物联网的基本概念和体系结构，分析了物联网安全威胁，并介绍了物联网安全模型。第 6 章移动互联网安全，介绍了移动互联网的基本概念和架构，对移动终端安全、位置隐私保护以及数据传播模式和安全管控的相关研究进行了介绍。

本书可作为高等学校计算机科学与技术、信号与信息处理相关专业高年级学生及硕士研究生信息安全技术课程的教材，也可作为相关领域研究人员的参考书。本书约需 40~60 学时讲授。

在本书的编写过程中，参考了国内外传媒信息安全技术相关领域的众多文献，在此对文献的作者表示衷心感谢。

本书得到了高端印刷装备信号与信息处理北京市重点实验室师生的大力支持，得到了国家自然科学基金项目（No. 61370140）、协同创新中心绿色印刷与出版技术项目（No.PXM2016_014223_000025）和北京印刷学院学科与研究生教育项目的资助，在此特表感谢。

# 目　录

第1章

# XML 安全技术

XML（Extensible Markup Language）是 W3C（World Wide Web Consortium，万维网联盟）制定的一种数据交换标准。通过 XML，开发者可以在不同平台、不同操作系统之间进行数据交换。XML 以其结构化、互操作性、易于交换和可扩展性的特点在诸多领域得到了应用，如电子商务、电子政务、数字出版等，尤其是随着 Web 服务（Web Service）的蓬勃发展，XML 越来越多地活跃在数据交换和存储领域。

随着越来越多的企业利用 XML 来传输结构化数据，XML 文档的安全问题也变得越来越重要。XML 的优势来自于它的语义和结构的灵活性和可扩展性，也正是这些优点带来了一些重要的安全问题。传统安全技术对于信息的加密、数字签名及密钥管理方面的研究已经很深入，但现有的安全技术和标准并不是为 XML 专门设计的。如 SSL（Secure Sockets Layer，安全套接字层）、TLS（Transport Layer Security，安全传输层协议）、IPSec（Internet Protocol Security，IP 安全协议）等在一定程度上能满足 XML 的安全需求。但是，它们只能保护客户端和服务器之间数据传输的安全，而无法保护存储在计算机系统中的数据内容的安全，一旦服务器（或客户端）接收了数据，这些数据在服务器（或客户端）上就不受保护。不但如此，对于复杂实体链中的端到端安全，如每个实体只能查看或修改局部的数据，传统的信息安全技术是无法做到的。

针对 XML 所出现的安全问题，W3C、OASIS（Organization for the Advancement of Structured Information Standards，结构化信息标准促进组织）、IETF（Internet Engineering Task Force，互联网工程任务组）和其他几个团体共同制定了 XML 安全标准和规范，为以 XML 作为数据交换载体的应用提供安全性保障。

## 1.1 XML 安全标准

XML 安全技术是一种新兴的网络安全技术，它将 XML 技术融合到数据安全中的数据加密、数字签名、公钥管理、权限管理、接入控制、认证鉴别等技术中，用以保护 XML 数据乃至网络中各种数据的安全。

W3C、OASIS 和 IETF 等推出了一系列 XML 安全标准和规范，XML 安全技术的应用也不断扩展，一些大公司如 Microsoft、IBM 等都在其产品中加入 XML 安全技术，XML

安全技术正在受到越来越多的关注。

XML 安全标准和规范包括 XML 加密（XML Encryption）、XML 签名（XML Signature）、XML 密钥管理规范（Key Management Specification，XKMS）、安全断言标记语言（Security Assertion Markup Language，SAML）、可扩展访问控制标记语言（eXtensible Access Control Markup Language，XACML）、可扩展权利标记语言（eXtensible right Markup Language，XrML）及 Web 服务安全规范（Web Services Security，WS- Security）等。

W3C 协同 IETF 在 1999 年 6 月开始制定规范，用来支持 XML 语法的加密和数字签名的创建，2001 年 8 月 20 日公布了 XML 数字签名规范，2002 年 9 月公布了 XML 加密规范的推荐标准。

在处理数字签名和加密文档时要用到 PKI（Public Key Infrastructure，公钥基础设施），为了方便 PKI 与 XML 应用程序以及使用这些程序的 Web 服务进行集成，2001 年 Microsoft、Verisign 和 WebMethods 共同开发了开放的 XKMS 规范。随后，该规范被提交到 W3C，W3C 组建了一个 XML 密钥工作组（XML Key Management Working Group），以协同其他感兴趣的参与者对其做进一步的开发，2003 年 4 月 18 日发布 2.0 版本。XKMS 与 XML 签名、XML 加密结合，对密钥、证书进行管理，包括注册、分发、撤销等，它还允许客户通过 Web 服务取得密钥信息。

SAML 是交换验证和授权信息的 XML 框架，用于在不同的安全域（security domain）之间交换认证和授权数据。SAML 标准定义了身份提供者（identity provider）和服务提供者（service provider），这两者构成了不同的安全域。它定义了对验证、属性和授权信息进行 XML 编码的语法和语义以及这些安全信息的传输协议。SAML 是结构化信息标准促进组织 OASIS 安全服务技术委员会（Security Services Technical Committee）制定发布的标准。该组织分别于 2002 年 11 月、2003 年 8 月和 2005 年 3 月制定了 SAML 的三个版本——1.0、1.1 和 2.0 版本。

XACML 是 OASIS 访问控制标记语言技术委员会（Access Control Markup Language Technical Committee）制定的 XML 规范，是一种可扩展的访问控制策略语言，用于以 XML 表示访问对象的授权策略。2003 年 2 月 18 日发布了 1.0 版本。

2004 年 4 月 19 日，OASIS 组织发布了 WS-Security（Web 服务安全）标准的 1.0 版本。2006 年 2 月 17 日，发布了 1.1 版本。WS-Security 是最初 IBM、Microsoft、VeriSign 和 Forum Systems 开发的，现在协议由 Oasis-Open 下的一个委员会开发，官方名称为 WSS（Web Service Security）。协议包含了关于如何在 Web 服务消息上保证完整性和机密性的规约。

表 1-1 显示了 XML 相关安全标准[1]。

表 1-1 XML 相关安全标准

| 安全标准和规范 | 开发机构 | 说　明 |
| --- | --- | --- |
| XML Encryption | W3C XML Encryption Working Group | XML 加密规范，实现 XML 数据加密及解密 |
| XML Signature | IETF/W3C XML Signature Working Group | XML 数字签名规范，实现身份验证、保证数据完整性和不可否认性等 |

续表

| 安全标准和规范 | 开发机构 | 说　明 |
| --- | --- | --- |
| XKMS 2.0 | W3C XKMS Working Group (WG) | XML 分发和注册公钥信息的规范，适合与已被提议的标准 XML Signature 和 XML Encryption 联合使用 |
| SAML 2.0 | OAIS Security Services TC(Technical Committee) | 提供标准的方法在 XML 文档中定义用户的认证信息、授权信息等，在可信任域之间交互认证和授权信息 |
| XACML 2.0 | OASIS eXtensible Access Control Markup Language TC | XML 访问控制标记语言，基于 XML 规范来表达在网上进行信息访问的策略 |
| XrML | ContentCuard eXtensible Right Markup Language TC | XrML 用于指定使用某资源时相应的权限和条件，其语法用 XML Schema 描述 |
| WS-Security 1.1 | OASIS Web Service Security (WSS) TC | 以 XML 为基础的 Web 服务安全规范 |

# 1.2 XML 加密

## 1.2.1 XML 加密和传统加密的区别

加密技术是指利用技术手段把重要的数据变为乱码（加密）传送，到达目的地后再用相同或不同的手段还原（解密）。加密包括两个元素：算法和密钥。一个加密算法是将普通的文本明文与一串数字密钥的结合，产生不可理解的密文的步骤。密钥由位序列数字组成。

当前，安全传输层协议 TLS 是因特网上安全通信的事实标准。TLS 是继著名的安全套接字层 SSL 之后的端到端安全性协议，提供了通信双方之间的端到端安全性会话。但 SSL 或 TLS 未涉及两个重要领域：加密交换数据的一部分和多方之间的安全会话。而 XML 加密可以有效地解决这些问题。XML 加密为需要安全地交换结构化数据的应用程序提供了端到端的安全服务。XML 本身就是结构化数据最流行的技术，因此基于 XML 的加密技术自然成为满足数据交换应用程序复杂的安全性能要求的技术。

首先，XML 加密与 SSL、TLS 最大不同是引入了加密粒度的概念。XML 文档作为一种结构化数据，能对加密的粒度进行控制，即加密粒度可选。XML 文档可以和任何其他文档一样作为一个整体进行加密，然后安全地发送给一个或多个接收者。这是 SSL 或 TLS 的一个普通功能。但是，更为有趣的是，如何来解决这种情形：加密 XML 文档的某些部分，而让那些不包含敏感信息的部分以明文的形式存在。XML 加密的一个重要特点就是它支持对 XML 文档的一些特定部分进行加密。加密的数据可以是任意的数据（包括 XML 文档、二进制数据等）、XML 元素或者 XML 元素内容。加密的结果是一个 XML 元素，它包含或引用密文数据。另外，还可以对 XML 文档进行超级加密，即对加密过的元素或内容进行再加密。

通过对 XML 加密粒度的选择，我们就能选择对 XML 文档的不同信息进行不同的加密处理，以控制对不同元素的授权查看。例如，对于某个客户的购买信息，商场只需

要知道客户的名称和地址，无需知道信用卡的信息，而银行则无需知道购买货物的详细信息；对住院看病的患者来说，医院的研究人员无需知道个人住院治疗的详细信息，但管理人员可能正需要这些信息。

其次，XML 加密状态持久，能保证数据内容的安全。XML 文档一经加密，在解密之前，不论是存储于磁盘空间中，还是在网络的传输过程中，或是在某个网络节点停留时，都处于加密状态，未经授权无法访问到密码信息，能确保数据内容的安全性。

再次，XML 加密可实现多方安全会话。使用 XML 加密，每一方都可以保持与任何通信方的安全或非安全状态，可以在同一文档中交换安全的和非安全的数据。XML 加密可实现多方安全通话。例如，一个包含许多聊天室的安全聊天应用程序，其中每个聊天室都有几个人。可以在聊天伙伴之间交换 XML 加密的文件，这样一个聊天室中的数据对其他聊天室而言是不可见的。

最后，XML 加密能满足各种应用环境的需求。XML 加密是以 XML 形式表现被加密的数据的，加密过程不会改变文档的格式，加密后的 XML 文档仍是一个格式良好的 XML 文档。由于 XML 文档在加密的过程中数据格式一致，可以方便地被基于 XML 的应用系统直接处理，无需格式转换。XML 能满足各种应用环境的共性需求，XML 既可以应用于消息传输，也可以应用于文档数据的存储，并支持特定应用的特殊需求，具有扩展能力。

### 1.2.2 XML 加密规范

XML 加密规范[2]是加密 XML 数据、以标准 XML 格式表示加密结果以及解密器处理过程的一套标准方法。XML 加密支持目前流行的一系列加密算法，如对称加密（包括分组加密、流加密）、不对称加密、消息摘要等。通常对称加密算法用于 XML 数据的批量加密，非对称加密算法则用于安全地交换对称密钥。

在进行 XML 加密时，采用标准的 XML 标记语法来表示相关信息、算法以及实际加密的数据。加密结果被表示为一个 XML 加密文档。这个 XML 加密文档可直接含有加密的数据，也可以间接地从外部引用加密的数据，而且应用程序能方便地访问和处理被加密的数据。

XML 加密规范定义了如下所示的名称空间：

xmlns:xenc="http://www.w3.org/2001/04/xmlenc#"

XML 名称空间除了提供语法的作用域外，还被用做规范中引用的算法标识符的前缀，能很快识别出被标识的算法（如 RSA、3DES、SHA-1 等）。下面列出一些在 XML 加密规范中定义的算法标识符示例：

http://www.w3c.org/2001/04/xmlene#rsa-1_5

http://www.w3c.org/2001/04/xmlene#tripledes-cbc

http://www.w3c.org/2001/04/xmldsig#shal

当加密任意数据时，其加密的结果就是 EncryptedData 元素。该元素作为 XML 加密的根元素，包含 XML 有关数据的所有信息。它包含三个主要的子元素：EncryptionMethod 元

素指定用来加密数据和解密数据的算法，这些算法由 URI 指定；KeyInfo 元素提供有关使用哪个密钥来解密数据的信息，也由 URI 指定；CipherData 元素则包含实际的加密信息。

XML 加密生成的文件是格式良好的 XML 文件，EncryptedData 元素的结构如下：

```
<EncryptedData Id? Type? MimeType? Encoding?>
    <EncryptionMethod/>?
    <ds:KeyInfo>
        <ds:EncryptedKey>?
        <ds:AgreementMethod>?
        <ds:KeyName>?
        <ds:RetrievalMethod>?
        <ds:*>?
    </ds:KeyInfo>?
    <CipherData>
      <CipherValue> | <CipherReference URI?>
    </CipherData>
    <EncryptionProperties>?
</EncryptedData>
```

这里?表示出现 0 次或 1 次；+表示出现 1 次或多次；*表示出现 0 次或多次；|表示选择；空元素标记意味着元素必须是空的。

（1）EncryptedData 元素

EncryptedData 元素是封装加密或解密所需相关信息的最外层元素。使用 EncryptedData 元素构建加密数据，该元素包含与加密和解密信息相关的数据，如加密密钥的信息、算法信息、加密数据以及加密数据的引用等。

加密后的数据保留了 XML 的语法特征。在生成的加密数据文件中，EncryptedData 元素用来替代加密的数据，若加密数据是某个 XML 文件本身，EncryptedData 元素则成为文件的根元素；若加密数据是 XML 文件的内部元素，则该元素和它的内容一起被删除并用 EncryptedData 元素替代。

XML 加密也能对其他非 XML 文件进行加密，如 HTML 文件、JPG 文件等，其加密方式与加密整个 XML 文件一样，但要先转换为 Base64 格式。

（2）EncryptionMethod 元素

EncryptionMethod 元素是 EncryptedData 元素的子元素，指定加密使用的算法。如果没有提供这个元素，那么参与 XML 加密的应用程序一定以某种方法共享或者隐含地知道所使用的加密算法。

（3）KeyInfo 元素

KeyInfo 元素是 EncryptedData 元素的子元素，该元素提供用于加密和解密数据的对称会话密钥。如果没有提供这个元素，那么参与 XML 加密的应用程序一定以某种方法共享或者隐含地知道所使用的加密算法。

（4）EncryptedKey 元素

EncryptedKey 元素是 KeyInfo 元素的子元素，该元素用于交换对称会话密钥。

（5）AgreementMethod 元素

AgreementMethod 元素是 KeyInfo 元素的子元素，该元素用于建立一个应用程序定义的、共享会话密钥的方法。如果没有提供这个元素，那么参与 XML 加密的应用程序必须以某种方式来处理密钥协议。

（6）KeyName 元素

KeyName 元素是 KeyInfo 元素的子元素，可以用该元素选择使用易读的名字来访问会话密钥。

（7）RetrievalMethod 元素

RetrievalMethod 元素是 KeyInfo 元素的子元素，该元素能够提供到另一个含有私有会话密钥的 EncryptedKey 元素的 URI 链接。

（8）CipherData 元素

CipherData 元素是 EncryptedData 元素必选的子元素，该元素包含或引用实际的加密数据。如果这个元素包含加密的数据，就使用 CipherValue 子元素；如果这个元素引用加密的数据，就使用 CipherReference 子元素。

CipherData 元素是 EncryptedData 元素的唯一必选子元素，该元素是有意义的，这是因为 EncrypedData 元素一定会提供加密的数据。

（9）CipherValue 元素

CipherValue 元素封装了实际的加密数据。

（10）CipherReferenee 元素

CipherReferenee 元素封装了对外部加密数据的引用。

（11）EncryptionProperties 元素

EncryptionProperties 元素提供了应用程序专用的附加信息，如加密操作的起源、日期和时间，这些信息可能非常有用。

### 1.2.3 XML 加密粒度的选择

下面给出一个图书购买的 XML 文档 BookOrder-1.xml，根元素 BookOrder 包含商品交易的信息，它有两个子元素：Book 和 Payment。Book 元素包含图书的具体信息，如书名 Name、价格 Price、数量 Quantity；Payment 元素包含付款相关的信息，如顾客姓名 CustomerName、信用卡号 CardId、信用卡有效期 Expiration。

```
<?xml version="1.0"  encoding="UTF-8"?>
<BookOrder>
    <Book>
        <Name>XML Technology</Name>
        <Price>35.0</Price>
        <Quantity>12</Quantity>
```

```
    </Book>
    <Payment>
        <CustomerName>Zhang Shan</CustomerName>
        <CardID>6222 0002 0012 3769 489</CardID>
        <Expiration>01/2018</Expiration>
    </Payment>
</BookOrder >
```

对绝大部分的应用来说，都会认为 Payment 元素是敏感信息；但是对 Book 元素的看法可能会不一样，有些公司可能会认为是敏感信息，而有些则不这么认为，因此会使用不同的加密方案，采用不同的加密粒度。

1. 对整个 XML 文档加密

首先将整个 XML 文档转换成字节流，再进行加密，然后将加密的字节流进行 Base64 编码。用 Base64 对加密后的字节流编码是因为在基于 ASCII 字符的 XML 文档中是不能直接载有二进制数据的。

经过 Base64 编码后得到了可以在 XML 文档中进行传输的字符流。但这时的字符流不能直接进行传输，因为传输的节点不能根据加密后的数据解释传输目标在哪里，也不知道怎样对文档进行解析，简单地说，就是加密后的字符流已经不再是一个可以被识别的 XML 文档了。因此，要将加密后的 XML 文档进行封装，封装好的 XML 文档存放在 DocumentEncrypt.xml 中：

```
<EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
Type='http://www.isi.edu/in-notes/iana/assignments/media-types/text/xml'>
    <CipherData>
        <CipherValue>
            v3A3ZjnuVTxiHK0YLotIvUgRXDr3NwSxWOtR6y……
        </CipherValue>
    </CipherData>
</EncryptedData>
```

在 DocumentEncrypt.xml 中，EncryptedData 元素包含一个名称空间属性 xmlns，其值为固定值 http://www.w3.org/2001/04/xmlenc#，是不能改变的；包含一个类型属性 Type，它能告诉解析器该元素里的数据在加密前是什么类型，IANA(Internet Assigned Numbers Authority，因特网编号管理局) 定义了 XML 的各种常用数据类型，如果有特殊的应用程序数据类型，可以在 EncryptedData 元素的 Type 属性里指定它们。例如，加密前的原始数据是 XML 文档，则 Type 的属性值为 http://www.isi.edu/in-notes/iana/assignments/media-types/text/xml。

CipherData 元素是保存加密后的数据的地方，它包含一个子元素 CipherValue，CipherValue 元素的内容就是原来的 XML 文档加密后的数据。

2. 对 XML 文档元素加密

对 XML 文档元素加密与对整个 XML 文档加密过程相似。先将待加密元素转换成字节流，然后再对字节流进行加密、Base64 编码等操作，最后得到该元素的密文。对 Payment 元素加密后的 XML 文档存放在 ElementEncrypt.xml 中：

```
<?xml version="1.0"  encoding="UTF-8"?>
<BookOrder>
    <Book>
        <Name>XML Technology </Name>
        <Price>35.0</Price>
        <Quantity>12</Quantity>
    </Book>
    <EncryptedData xmlns="http://www.w3.org/2001/04/xmlenc#"
        Type="http://www.w3.org/2001/04/xmlenc#Element">
        <CipherData>
            <CipherValue>
              v3A3ZjnuVTxiHK0YLotIvUgRXDr3NwSxWOt......
            </CipherValue>
        </CipherData>
    </EncryptedData>
</BookOrder>
```

将 ElementEncrypt.xml 与 DocumentEncrypt.xml 比较会发现，由于只是选择了 Payment 元素进行加密，加密后的 XML 文档保留了绝大部分的原文档信息，只是原来 Payment 元素被换成了 EncryptedData 元素，结构与整体加密 XML 文档类似。

ElementEncrypt.xml 的 EncryptedData 元素也有一个类型属性 Type，但它的值是 http://www.w3.org/2001/04/xmlenc#Element，不再使用 IANA 类型，而是使用 XML 加密指定的类型，末尾的片段 #Element 表示一个元素。

3. 对 XML 文档元素内容加密

对 XML 文档元素加密后，为什么还要提出对元素内容加密呢？因为如果存在透露元素名称或其属性的特殊需要，而又要保持其内容安全时，元素内容加密就很方便。例如，如果图书销售商只想保密信用卡号，则加密 CardId 元素的文本内容即可。对 CardId 元素内容加密后的 XML 文档存放在 ContentEncrypt.xml 中：

```
<?xml version="1.0"  encoding="UTF-8"?>
<BookOrder>
    <Book>
        <Name>XML Technology</Name>
```

```
        <Price>35.0</Price>
        <Quantity>12</Quantity>
    </Book>
    <Payment>
        <CustomerName>Zhang Shan</CustomerName>
        <CardId>
            <EncryptedData  xmlns="http://www.w3.org/2001/04/xmlenc#"
                Type="http://www.w3.org/2001/04/xmlenc#Content">
                <CipherData>
                    <CipherValue>
                        7lomEVOALG3vFZHSJstlbNSxnDP1jla……
                    </CipherValue>
                </CipherData>
            </EncryptedData>
        </CardId>
        <Expiration>01/2018</Expiration>
    </Payment>
</BookOrder >
```

从 ContentEncrypt.xml 可以看出，CardId 元素内容被替换成了 EncryptedData 元素，并嵌入到原 XML 文档中。这里使用 http://www.w3.org /2001/04/xmlenc#Content 作为 EncryptedData 元素的 Type 属性值。当 XML 解析器遇到该元素，便知道这里面包含有加密信息，需要解密才能进行正常使用。

4. 超级加密

XML 加密除了能对 XML 文档的部分内容进行加密外，还能通过超级加密，对加密后的数据进行再加密，这种加密方式能够控制文件选定部分内容的查阅权限，将数据资料传递给不同的当事人并能保证数据的保密性。例如，若传递一份订单资料给某公司，需经过公司的销售部门及财务部门，可以先用财务部门的密钥去对付款的元素部分加密，形成一个包含元素加密的 XML 文档，然后再利用销售部门的密钥将这份 XML 文档整体加密，形成一个超级加密的 XML 文档。当订单传递给销售部门时，销售部门对加密的 XML 文档进行解密，也不会看到付款部分的信息。直到该文档被传到财务部门后，才能看到整份 XML 文档的内容。

一份 XML 文档可以包含多个 EncryptedData 元素，但是 EncryptedData 元素不能成为另一个 EncryptedData 元素的父元素或子元素，因此超级加密必须要加密 EncryptedData 元素包含的所有内容，不能只加密 EncryptedData 元素的子元素或元素的内容。

XML 加密有一个共同的特征，所有需要加密的数据在加密后都会被 EncryptedData 元素替换掉，而其他元素保持不变。