数字证书格式

1　概数字证书格式

1.1 概述

数字证书就是互联网通讯中标志通讯各方身份信息的一串数字，提供了一种在 Internet 上验证通信实体身份的方式，数字证书不是数字身份证，而是身份认证机构盖在数字身份证上的一个章或印（或者说加在数字身份证上的一个签名）。它是由权威机构——CA 机构，又称为证书授权（Certificate Authority）中心发行的，人们可以在网上用它来识别对方的身份。

2　数字证书常见标准

- 符合 PKI ITU-T X509 标准，传统标准（.DER .PEM .CER .CRT）
- 符合 PKCS#7 加密消息语法标准(.P7B .P7C .SPC .P7R)
- 符合 PKCS#10 证书请求标准(.p10)
- 符合 PKCS#12 个人信息交换标准（.pfx *.p12）

PKCS 全称是 Public-Key Cryptography Standards ，是由 RSA 实验室与其它安全系统开发商为促进公钥密码的发展而制订的一系列标准，PKCS 目前共发布过 15 个标准。 常用的有：

PKCS#7 Cryptographic Message Syntax Standard

PKCS#10 Certification Request Standard

PKCS#12 Personal Information Exchange Syntax Standard

X.509 是常见通用的证书格式。所有的证书都符合为 Public Key Infrastructure (PKI) 制定的 ITU-T X509 国际标准。基本的证书格式，只包含公钥。

x509 证书由用户公共密钥和用户标识符组成。此外还包括版本号、证书序列号、CA 标识符、签名算法标识、签发者名称、证书有效期等信息。

3　证书格式

3.1 证书格式分类

分为 2 大类：密钥库（KeyStore 含私钥，也可能有公钥）和公钥证书（仅含公钥）

3.2 密钥库文件格式（KeyStore）

格式　：JKS
扩展名　：.jks/.ks

描述　：【Java Keystore】密钥库的 Java 实现版本，provider 为 SUN
特点　：密钥库和私钥用不同的密码进行保护

格式　：JCEKS
扩展名：.jce
描述　：【JCE Keystore】密钥库的 JCE 实现版本，provider 为 SUN JCE
特点　：相对于 JKS 安全级别更高，保护 Keystore 私钥时采用 TripleDES

格式　：PKCS12
扩展名：.p12/.pfx
描述　：【PKCS #12】个人信息交换语法标准
特点　：1、包含私钥、公钥及其证书
　　2、密钥库和私钥用相同密码进行保护

格式　：BKS
扩展名：.bks
描述　：【Bouncycastle Keystore】密钥库的 BC 实现版本，provider 为 BC
特点　：基于 JCE 实现

格式　: UBER
扩展名：.ubr
描述　：【Bouncycastle UBER Keystore】密钥库的 BC 更安全实现版本，provider 为 BC

3.3 证书文件格式（Certificate）

- 格式　：DER
  扩展名　：.cer/.crt/.rsa
- 描述　：【ASN .1 DER】用于存放证书
  特点　：不含私钥、二进制

- 格式　：PKCS7
  扩展名　: .p7b/.p7r
  描述　：【PKCS #7】加密信息语法标准
- 特点　: 1、p7b 以树状展示证书链，不含私钥
  2、p7r 为 CA 对证书请求签名的回复，只能用于导入

- 格式　：CMS
  扩展名　：.p7c/.p7m/.p7s
  描述　：【Cryptographic Message Syntax】
  特点　: 1、p7c 只保存证书
  2、p7m：signature with enveloped data
  3、p7s：时间戳签名文件

格式　　：PEM
扩展名　：.pem
描述　　：【Printable Encoded Message】
特点　　：1、该编码格式在 RFC1421 中定义，其实 PEM 是【Privacy-Enhanced Mail】的简写，但他也同样广泛运用于密钥管理
2、ASCII 文件
3、一般基于 base 64 编码

- 格式　　：PKCS10
扩展名　：.p10/.csr
描述　　：【PKCS #10】公钥加密标准【Certificate Signing Request】
特点　　：1、证书签名请求文件
2、ASCII 文件
3、CA 签名后以 p7r 文件回复

- 格式　　：SPC
扩展名　：.pvk/.spc
描述　　：【Software Publishing Certificate】
特点　　：微软公司特有的双证书文件格式，经常用于代码签名，其中
1、pvk 用于保存私钥
2、spc 用于保存公钥

4　数字证书文件格式（cer 和 pfx）的区别

作为文件形式存在的证书一般有这几种格式：

1.带有私钥的证书由 Public Key Cryptography Standards #12，PKCS#12 标准定义，包含了公钥和私钥的二进制格式的证书形式，以 pfx 作为证书文件后缀名。

2.二进制编码的证书 证书中没有私钥，DER 编码二进制格式的证书文件，以 cer 作为证书文件后缀名。

3.Base64 编码的证书证书中没有私钥，BASE64 编码格式的证书文件，也是以 cer 作为证书文件后缀名。

由定义可以看出，只有 pfx 格式的数字证书是包含有私钥的，cer 格式的数字证书里面只有公钥没有私钥。

5　读取证书密钥

其中，我介绍如何从 p12/pfx 文件中提取密钥对及其长度:

1，首先，读取 pfx/p12 文件（需要提供保护密码）；

2，通过别名(Alias,注意，所有证书中的信息项都是通过 Alias 来提取的)提取你想要分析的证书链；

3，再将其转换为一个以 X509 证书结构体；

4，提取里面的项，如果你的证书项放在第一位（单一证书），直接读取 x509Certs[0]（见下面的代码）这个 X509Certificate 对象；

5，X509Certificate 对象有很多方法，

```
X509Certificate keyPairCert = x509Certs[0];
int iKeySize = X509CertUtil.getCertificateKeyLength(keyPairCert);
System.out.println("证书密钥算法="+keyPairCert.getPublicKey().getAlgorithm());
System.out.println("证书密钥长度="+iKeySize);
```

提取了他所需要的信息。

附：

X.509 数字证书结构和实例

一、 X.509 数字证书的编码

X.509 证书的结构是用 ASN1(Abstract Syntax Notation One)进行描述数据结构，并使用 ASN1 语法进行编码。

ASN1 采用一个个的数据块来描述整个数据结构，每个数据块都有四个部分组成：

1、数据块数据类型标识（一个字节）

数据类型包括简单类型和结构类型。

- 简单类型是不能再分解类型，如整型(INTERGER)、比特串(BIT STRING)、字节串(OCTET STRING)、对象标示符(OBJECT IDENTIFIER)、日期型(UTCTime)等。
- 结构类型是由简单类型和结构类型组合而成的，如顺序类型(SEQUENCE, SEQUENCE OF)、选择类型（CHOICE）、集合类型（SET）等。
  - 顺序类型的数据块值由按给定顺序成员成员数据块值按照顺序组成；
  - 选择类型的数据块值由多个成员数据数据块类型中选择一个的数据块值；
  - 集合数据块类型由成员数据块类型的一个或多个值构成。

这个标识字节的结构如下：

1.1. Bit8-bit7 用来标示 TAG 类型，共有四种，分别是 universal(00)、application(01)、context-specific(10)和 private(11)。

1.2. Bit6 表示是否为结构类型(1 位结构类型)；0 则表明编码类型是简单类型。

1.3. Bit5-bit1 是类型的 TAG 值。根据 bit8-bit7 的不同值有不同的含义，具体含义见下表。

当 Bit8-bit7 为 universal（00）时，bit5-bit1 的值表示不同的 universal 的值：

标记（TAG）	对应类型	备注
[UNIVERSAL 1]	BOOLEAN	[有两个值:false 或 true]
[UNIVERSAL 2]	INTEGER	[整型值]
[UNIVERSAL 3]	BIT STRING	[0 位或多位]
[UNIVERSAL 4]	OCTET STRING	[0 字节或多字节]
[UNIVERSAL 5]	NULL	
[UNIVERSAL 6]	OBJECT IDENTIFIER	[相应于一个对象的独特标识数字]
[UNIVERSAL 7]	OBJECT DESCRIPTOR	[一个对象的简称]
[UNIVERSAL 8]	EXTERNAL, INSTANCE OF	[ASN.1 没有定义的数据类型]
[UNIVERSAL	REAL	[实数值]

9]		
[UNIVERSAL 10]	ENUMERATED	[数值列表，这些数据每个都有独特的标识符，作为 ASN.1 定义数据类型的一部分]
[UNIVERSAL 12]	UTF8String	
[UNIVERSAL 13]	RELATIVE-OID	
[UNIVERSAL 16]	SEQUENCE, SEQUENCE OF	[有序数列，SEQUENCE 里面的每个数值都可以是不同类型的，而 SEQUENCE OF 里是 0 个或多个类型相同的数据]
[UNIVERSAL 17]	SET, SET OF	[无序数列，SET 里面的每个数值都可以是不同类型的，而 SET OF 里是 0 个或多个类型相同的数据]
[UNIVERSAL 18]	Numeric String	[0－9 以及空格]
[UNIVERSAL 19]	Printable String	[A-Z、a-z、0-9、空格以及符号 '()+,-./:=?]
[UNIVERSAL 20]	TeletexString, T61String	
[UNIVERSAL 21]	VideotexString	
[UNIVERSAL 22]	IA5String	
[UNIVERSAL 23]	UTCTime	[统一全球时间格式]
[UNIVERSAL 24]	GeneralizedTime	
[UNIVERSAL 25]	GraphicString	
[UNIVERSAL 26]	VisibleString, ISO646String	
[UNIVERSAL 27]	GeneralString	
[UNIVERSAL 28]	UniversalString	
[UNIVERSAL 29]	CHARACTER STRING	
[UNIVERSAL 30]	BMPString	
[UNIVERSAL 31]		reserved for future use

当 Bit8-bit7 为 context-specific（10）时，bit5-bit1 的值表示特殊内容：

[0] –- 表示证书的版本

[1] –- issuerUniqueID,表示证书发行者的唯一 id

[2] –- subjectUniqueID,表示证书主体的唯一 id

[3] –- 表示证书的扩展字段

如 SEQUENCE 类型数据块，其 TAG 类型位 UNIVERSAL（00）,属于结构类型（1），TAG 值为 16（10000）所以其类型标示字段值为（00110000），即为 0x30。再如，证书扩展字段类型的数据块，TAG 类型为（10），属结构类型（1），TAG 的值为 3（00011），所以其类型标示字段值为（10100011），即为 0xA3。

2、数据块长度（1-128 个字节）

长度字段，有两种编码格式。

- 若长度值小于等于 127，则用一个字节表示，bit8 = 0, bit7-bit1 存放长度值；
- 若长度值大于 127，则用多个字节表示，可以有 2 到 127 个字节。第一个字节的第 8 位为 1，其它低 7 位给出后面该域使用的字节的数量，从该域第二个字节开始给出数据的长度，高位优先。
- 还有一种特殊情况，这个字节为 0x80，表示数据块长度不定，由数据块结束标识结束数据块。

3、数据块的值

存放数据块的值，具体编码随数据块类型不同而不同。

4、数据块结束标识（可选）

结束标示字段，两个字节（0x0000）,只有在长度值为不定时才会出现。

二、 X.509 证书的结构

1、X.509 证书基本部分

1.1. 版本号.

标识证书的版本（版本 1、版本 2 或是版本 3）。

1.2. 序列号

标识证书的唯一整数，由证书颁发者分配的本证书的唯一标识符。

1.3. 签名

用于签证书的算法标识，由对象标识符加上相关的参数组成，用于说明本证书所用的数字签名算法。例如，SHA-1 和 RSA 的对象标识符就用来说明该数字签名是利用 RSA 对 SHA-1 杂凑加密。

1.4. 颁发者

证书颁发者的可识别名（DN）。

1.5. 有效期

证书有效期的时间段。本字段由”Not Before”和”Not After”两项组成，它们分别由 UTC 时间或一般的时间表示（在 RFC2459 中有详细的时间表示规则）。

1.6. 主体

证书拥有者的可识别名，这个字段必须是非空的，除非你在证书扩展中有别名。

1.7. 主体公钥信息

主体的公钥（以及算法标识符）。

1.8. 颁发者唯一标识符

标识符—证书颁发者的唯一标识符，仅在版本 2 和版本 3 中有要求，属于可选项。

1.9. 主体唯一标识符

证书拥有者的唯一标识符，仅在版本 2 和版本 3 中有要求，属于可选项。

2、X.509 证书扩展部分

可选的标准和专用的扩展（仅在版本 2 和版本 3 中使用），扩展部分的元素都有这样的结构：

```
Extension ::= SEQUENCE {
    extnID      OBJECT IDENTIFIER,
    critical    BOOLEAN DEFAULT FALSE,
    extnValue   OCTET STRING }
```

extnID：表示一个扩展元素的 OID

critical：表示这个扩展元素是否极重要

extnValue：表示这个扩展元素的值，字符串类型。

扩展部分包括：

2.1. 发行者密钥标识符

证书所含密钥的唯一标识符，用来区分同一证书拥有者的多对密钥。

2.2. 密钥使用

一个比特串，指明（限定）证书的公钥可以完成的功能或服务，如：证书签名、数据加密等。如果某一证书将 KeyUsage 扩展标记为“极重要”，而且设置为“keyCertSign”，则在 SSL 通信期间该证书出现时将被拒绝，因为该证书扩展表示相关私钥应只用于签写证书，而不应该用于 SSL。

2.3. CRL 分布点

指明 CRL 的分布地点。

2.4. 私钥的使用期

指明证书中与公钥相联系的私钥的使用期限，它也有 Not Before 和 Not After 组成。若此项不存在时，公私钥的使用期是一样的。

2.5. 证书策略

由对象标识符和限定符组成，这些对象标识符说明证书的颁发和使用策略有关。

2.6. 策略映射

表明两个 CA 域之间的一个或多个策略对象标识符的等价关系，仅在 CA 证书里存在。

2.7. 主体别名

指出证书拥有者的别名，如电子邮件地址、IP 地址等，别名是和 DN 绑定在一起的。

2.8. 颁发者别名

指出证书颁发者的别名，如电子邮件地址、IP 地址等，但颁发者的 DN 必须出现在证书的颁发者字段。

2.9. 主体目录属性

指出证书拥有者的一系列属性。可以使用这一项来传递访问控制信息。

三、 X.509 证书详细描述

Certificate ::= SEQUENCE {

tbsCertificate TBSCertificate, -- 证书主体

signatureAlgorithm AlgorithmIdentifier, -- 证书签名算法标识

signatureValue BIT STRING -- 证书签名值，是使用 signatureAlgorithm 部分指定的签名算法对 tbsCertificate 证书主题部分签名后的值.

}

TBSCertificate ::= SEQUENCE {

version [0] EXPLICIT Version DEFAULT v1, -- 证书版本号

serialNumber CertificateSerialNumber, -- 证书序列号，对同一 CA 所颁发的证书，序列号唯一标识证书

signature AlgorithmIdentifier, --证书签名算法标识

issuer Name, --证书发行者名称

validity Validity, --证书有效期

subject Name, --证书主体名称

subjectPublicKeyInfo SubjectPublicKeyInfo,--证书公钥

issuerUniqueID [1] IMPLICIT UniqueIdentifier OPTIONAL,

-- 证书发行者 ID(可选)，只在证书版本 2、3 中才有

subjectUniqueID [2] IMPLICIT UniqueIdentifier OPTIONAL,

-- 证书主体 ID(可选)，只在证书版本 2、3 中才有

extensions [3] EXPLICIT Extensions OPTIONAL

-- 证书扩展段（可选），只在证书版本 3 中才有

}

Version ::= INTEGER { v1(0), v2(1), v3(2) }

CertificateSerialNumber ::= INTEGER

AlgorithmIdentifier ::= SEQUENCE {

algorithm OBJECT IDENTIFIER,

parameters ANY DEFINED BY algorithm OPTIONAL }

Parameters:

Dss-Parms ::= SEQUENCE { -- parameters ，DSA(DSS)算法时的 parameters, RSA 算法没有此参数

p INTEGER,

q INTEGER,

```
            g           INTEGER }

SignatureValue:
    Dss-Sig-Value ::= SEQUENCE { -- sha1DSA 签名算法时,签名值
            r       INTEGER,
            s       INTEGER }

    Name ::= CHOICE {
            RDNSequence }

    RDNSequence ::= SEQUENCE OF RelativeDistinguishedName

    RelativeDistinguishedName ::=   SET OF AttributeTypeAndValue

    AttributeTypeAndValue ::= SEQUENCE {
            type     AttributeType,
            value    AttributeValue }

    AttributeType ::= OBJECT IDENTIFIER
    AttributeValue ::= ANY DEFINED BY AttributeType

    Validity ::= SEQUENCE {
            notBefore      Time,   --  证书有效期起始时间
            notAfter       Time  --  证书有效期终止时间  }

    Time ::= CHOICE {
            utcTime        UTCTime,
            generalTime    GeneralizedTime }

    UniqueIdentifier ::= BIT STRING

    SubjectPublicKeyInfo ::= SEQUENCE {
            algorithm           AlgorithmIdentifier, --  公钥算法
            subjectPublicKey    BIT STRING           --  公钥值
            }

SubjectPublicKey:
    RSAPublicKey ::= SEQUENCE { -- RSA 算法时的公钥值
            modulus           INTEGER, -- n
            publicExponent    INTEGER -- e -- }

    Extensions ::= SEQUENCE SIZE (1..MAX) OF Extension

    Extension ::= SEQUENCE {
```

```
extnID      OBJECT IDENTIFIER,
critical    BOOLEAN DEFAULT FALSE,
extnValue   OCTET STRING }
```

四、 X.509 数字证书实例

这是从 RFC 2459 Internet X.509 Public Key Infrastructure 标准文档中摘取的两个证书例子。本文在例子的原来基础上加了些注释。

1、DSA 证书，CA 证书

证书包含 699 字节，证书版本号为 3。

该证书包含以下内容：

(a) 证书序列号是 17 (0x11);
(b) 证书使用 DSA 和 SHA-1 哈希算法签名;
(c) 证书发行者的名字是 OU=nist; O=gov; C=US
(d) 证书主体的名字是 OU=nist; O=gov; C=US
(e) 证书的有效期从 1997-6-30 到 1997-12-31;
(f) 证书包含一个 1024 bit DSA 公钥及其参数（三个整数 p、q、g）;
(g) 证书包含一个使用者密钥标识符(subjectKeyIdentifier)扩展项
(h) 证书是一个 CA 证书(通过 basicConstraints 基本扩展项标识)

地址	内容	意义
0000	30 82 02 b7	SEQUENCE Certificate:: SEQUENCE 类型(30)，数据块长度字节为 2（82），长度为 695（02 b7）
0004	30 82 02 77	SEQUENCE tbsCertificate:: SEQUENCE 类型,长度 631
0008	a0 03	Version::特殊内容-证书版本(a0),长度 3
0010	02 01 02	INTEGER 2 version::整数类型(02),长度 1,版本 3(2)
0013	02 01 11	INTEGER 17 serialNumber::整数类型(02),长度 1,证书序列号 17
0016	30 09	SEQUENCE signature:: SEQUENCE 类型(30),长度 9
0018	06 07	signature:: OBJECT IDENTIFIER 类型,长度 7,dsa-with-sha 算法 OID 1.2.840.10040.4.3: dsa-with-sha
	2a 86 48 ce 38 04 03	
0027	30 2a	SEQUENCE 以下的数据块表示 issuer 信息,长度为 42
0029	31 0b	SET 开始一个集合，长度为 11
0031	30 09	SEQUENCE 开始一个序列，长度为 9
0033	06 03	OBJECT IDENTIFIER 类型,长度 3 OID 2.5.4.6 C
	55 04 06	

0038	13 02	PrintableString 'US'
	55 53	
0042	31 0c	SET 开始一个集合，长度为 12
0044	30 0a	SEQUENCE 开始一个序列，长度为 10
0046	06 03	OBJECT IDENTIFIER 类型,长度 3 OID 2.5.4.10 O
	55 04 0a	
0051	13 03	PrintableString 'gov'
	67 6f 76	
0056	31 0d	SET 开始一个集合，长度为 13
0058	30 0b	SEQUENCE 开始一个序列，长度为 11
0060	06 03	OBJECT IDENTIFIER 类型,长度 3 OID 2.5.4.11: OU
	55 04 0b	
0065	13 04	PrintableString 'nist'
	6e 69 73 74	
0071	30 1e	SEQUENCE validity:: SEQUENCE 类型(30),长度 30
0073	17 0d	notBefore:: UTCTime 类型(23),长度 13 UTCTime '970630000000Z'
	39 37 30 36 33 30 30 30 30 30 30 5a	
0088	17 0d	notBefore:: UTCTime 类型(23),长度 13 UTCTime '971231000000Z'
	39 37 31 32 33 31 30 30 30 30 30 5a	
0103	30 2a	SEQUENCE 以下数据块表示 subject 信息,长度 42
0105	31 0b	SET,长度 11
0107	30 09	SEQUENCE 长度 9
0109	06 03	OBJECT IDENTIFIER 类型,长度 3 OID 2.5.4.6: C
	55 04 06	
0114	13 02	PrintableString 'US'
	55 53	
0118	31 0c	SET,长度 12
0120	30 0a	SEQUENCE 长度 10
0122	06 03	OBJECT IDENTIFIER 类型,长度 3 OID 2.5.4.10: O
	55 04 0a	
0127	13 03	PrintableString 'gov'
	67 6f 76	
0132	31 0d	SET,长度 13
0134	30 0b	SEQUENCE 长度 11
0136	06 03	OBJECT IDENTIFIER 类型,长度 3 OID 2.5.4.11: OU
	55 04 0b	
0141	13 04	PrintableString 'nist'

	6e 69 73 74	
0147	30 82 01 b4	SEQUENCE subjectPublicKeyInfo:: SEQUENCE 类型(30), 长度 436
0151	30 82 01 29	SEQUENCE 类型(30), 长度 297
0155	06 07	IDENTIFIER 类型,长度 7 OID 1.2.840.10040.4.1
	2a 86 48 ce 38 04 01	
0164	30 82 01 1c	SEQUENCE 类型(30), 长度 284 DSA 算法的 parameters,三个整数 p、q、g
0168	02 81 80	INTEGER p 参数，长度 128
	d4 38 02 c5 35 7b d5 0b a1 7e 5d 72 59 63 55 d3 45 56 ea e2 25 1a 6b c5 a4 ab aa 0b d4 62 b4 d2 21 b1 95 a2 c6 01 c9 c3 fa 01 6f 79 86 83 3d 03 61 e1 f1 92 ac bc 03 4e 89 a3 c9 53 4a f7 e2 a6 48 cf 42 1e 21 b1 5c 2b 3a 7f ba be 6b 5a f7 0a 26 d8 8e 1b eb ec bf 1e 5a 3f 45 c0 bd 31 23 be 69 71 a7 c2 90 fe a5 d6 80 b5 24 dc 44 9c eb 4d f9 da f0 c8 e8 a2 4c 99 07 5c 8e 35 2b 7d 57 8d	
0299	02 14	INTEGER q 参数，长度 20
	a7 83 9b f3 bd 2c 20 07 fc 4c e7 e8 9f f3 39 83 51 0d dc dd	
0321	02 81 80	INTEGER g 参数，长度 128
	0e 3b 46 31 8a 0a 58 86 40 84 e3 a1 22 0d 88 ca 90 88 57 64 9f 01 21 e0 15 05 94 24 82 e2 10 90 d9 e1 4e 10 5c e7 54 6b d4 0c 2b 1b 59 0a a0 b5 a1 7d b5 07 e3 65 7c ea 90 d8 8e 30 42 e4 85 bb ac fa 4e 76 4b 78 0e df 6c e5 a6 e1 bd 59 77 7d a6 97 59 c5 29 a7 b3 3f 95 3e 9d f1 59 2d f7 42 87 62 3f f1 b8 6f c7 3d 4b b8 8d 74 c4 ca 44 90 cf 67 db de 14 60 97 4a d1 f7 6d 9e 09 94 c4 0d	
0452	03 81 84	BIT STRING (0 unused bits) subjectPublicKey :: 公钥值， BIT

		STRING 类型，长度 132 字节(好像应该是 131 字节)
0455	02 81 80	INTEGER 公钥值，表现为 integer 类型,128 字节,1024 位
	aa 98 ea 13 94 a2 db f1 5b 7f 98 2f 78 e7 d8 e3 b9 71 86 f6 80 2f 40 39 c3 da 3b 4b 13 46 26 ee 0d 56 c5 a3 3a 39 b7 7d 33 c2 6b 5c 77 92 f2 55 65 90 39 cd 1a 3c 86 e1 32 eb 25 bc 91 c4 ff 80 4f 36 61 bd cc e2 61 04 e0 7e 60 13 ca c0 9c dd e0 ea 41 de 33 c1 f1 44 a9 bc 71 de cf 59 d4 6e da 44 99 3c 21 64 e4 78 54 9d d0 7b ba 4e f5 18 4d 5e 39 30 bf e0 d1 f6 f4 83 25 4f 14 aa 71 e1	
0587	a3 32	extensions:: 特殊内容-证书扩展部分(a3) ，长度 50
0589	30 30	SEQUENCE, 长度 48
0591	30 0f	SEQUENCE 扩展 basicConstraints,长度 9
0593	06 03	OID 2.5.29.19: basicConstraints
	55 1d 13	
0598	01 01	BOOLEAN true，表示为 CA 证书
	ff	
0601	04 05	OCTET STRING, 长度 5
	30 03 01 01 ff	
0608	30 1d	SEQUENCE 扩展 subjectKeyIdentifier,长度 29
0610	06 03	OID 2.5.29.14: subjectKeyIdentifier
	55 1d 0e	
0615	04 16	OCTET STRING 扩展 subjectKeyIdentifier 的值, 长度 22
	04 14 e7 26 c5 54 cd 5b a3 6f 35 68 95 aa d5 ff 1c 21 e4 22 75 d6	
0639	30 09	SEQUENCE signatureAlgorithm:: = AlgorithmIdentifier, 长度 9
0641	06 07	OID 1.2.840.10040.4.3: dsa-with-sha
	2a 86 48 ce 38 04 03	
0650	03 2f	BIT STRING (0 unused bits) bit 串，证书签名值，47 字节
0652	30 2c	SEQUENCE, 长度 44

0654	02 14	INTEGER 签名值,20 字节,160bit
	a0 66 c1 76 33 99 13 51 8d 93 64 2f ca 13 73 de 79 1a 7d 33	
0674	02 14	INTEGER 签名值,20 字节,160bit
	5d 90 f6 ce 92 4a bf 29 11 24 80 28 a6 5a 8e 73 b6 76 02 68	

2、RSA 证书，非 CA 证书

证书包含 675 字节，证书版本号为 3。

该证书包含以下内容：

(a) 证书序列号是 256 (0x100);

(b) 证书使用 RSA 和 MD2 哈希算法签名;

(c) 证书发行者的名字是 OU=Dept. Arquitectura de Computadors; O=Universitat Politecnica de Catalunya; C=ES

(d) 证书主体的名字是 CN=Francisco Jordan;OU=Dept. Arquitectura de Computadors; O=Universitat Politecnica de Catalunya; C=ES

(e) 证书的有效期从 1996-5-21 到 1997-5-21;

(f) 证书包含一个 768 bit RSA 公钥;

(g) 证书是一个非 CA 证书(通过一个基本扩展项标识)

(h) 证书包含证书主体别名、证书发行者别名–都是 URLs

(i) 证书包含一个发行者密钥标识符和证书策略扩展，和

(j) 证书包含一个密钥用法的扩展，指定用于数字签名

```
0000 30 80          : SEQUENCE   (size undefined) // Certificate:: SEQUENCE 类型(30)，数据块长度不定，由 00、00 作为结束符
0002 30 82 02 40 576: . SEQUENCE        // tbsCertificate:: SEQUENCE 类型,长度 576
0006 a0 03          3: . . [0]          // Version:: 特殊内容-证书版本(a0),长度 3
0008 02 01          1: . . . INTEGER 2   //整数类型(02),长度 1
                : 02                //  版本 3(2)
0011 02 02          2: . . INTEGER 256   //serialNumber:: 整数类型(02),长度 2
                : 01 00             //  证书序列号 256
0015 30 0d         13: . . SEQUENCE      // signature:: SEQUENCE 类型(30),长度 13
0017 06 09          9: . . . OID 1.2.840.113549.1.1.2: MD2WithRSAEncryption
                                // signature:: OBJECT IDENTIFIER 类型,长度 9
                : 2a 86 48 86 f7 0d 01 01 02 //MD2WithRSAEncryption 算法(见注 1)
0028 05 00          0: . . . NULL
0030 30 68         88: . . SEQUENCE             // 以下红色的数据块表示 issuer 信息
0032 31 0b         11: . . . SET
0034 30 09          9: . . . . SEQUENCE
0036 06 03          3: . . . . . OID 2.5.4.6: C
                : 55 04 06
```

```
0041 13 02        2: . . . . . PrintableString 'ES'
                : 45 53
0045 31 2d       45: . . . SET
0047 30 2b       43: . . . . SEQUENCE
0049 06 03        3: . . . . . OID 2.5.4.10: O
                : 55 04 0a
0054 13 24       36: . . . . . PrintableString
                'Universitat Politecnica de Catalunya'
                : 55 6e 69 76 65 72 73 69 74 61 74 20 50 6f 6c 69
                : 74 65 63 6e 69 63 61 20 64 65 20 43 61 74 61 6c
                : 75 6e 79 61
0092 31 2a       42: . . . SET
0094 30 28       40: . . . . SEQUENCE
0096 06 03        3: . . . . . OID 2.5.4.11: OU
                : 55 04 0b
0101 13 21       33: . . . . . PrintableString
                'OU=Dept. Arquitectura de Computadors'
                : 44 65 70 74 2e 20 41 72 71 75 69 74 65 63 74 75
                : 72 61 20 64 65 20 43 6f 6d 70 75 74 61 64 6f 72
                : 73
0136 30 1e       30: . . SEQUENCE       // validity:: SEQUENCE 类型(30),长度 30
0138 17 0d       13: . . . UTCTime '960521095826Z' // notBefore:: UTCTime 类型(23)
长度 13
                : 39 36 30 37 32 32 31 37 33 38 30 32 5a
0153 17 0d       13: . . . UTCTime '979521095826Z' // notBefore:: UTCTime 类型(23)
长度 13
                : 39 37 30 37 32 32 31 37 33 38 30 32 5a
0168 30 81 83     112: . . SEQUENCE             //  以下红色的数据块表示 subject 信
息
0171 31 0b       11: . . . SET
0173 30 09        9: . . . . SEQUENCE
0175 06 03        3: . . . . . OID 2.5.4.6: C
                : 55 04 06
0180 13 02        2: . . . . . PrintableString 'ES'
                : 45 53
0184 31 2d       12: . . . SET
0186 30 2b       16: . . . . SEQUENCE
0188 06 03        3: . . . . . OID 2.5.4.10: O
                : 55 04 0a
0193 13 24       36: . . . . . PrintableString
                'Universitat Politecnica de Catalunya'
                : 55 6e 69 76 65 72 73 69 74 61 74 20 50 6f 6c 69
                : 74 65 63 6e 69 63 61 20 64 65 20 43 61 74 61 6c
                : 75 6e 79 61
```

```
0231 31 2a        42: . . . SET
0233 30 28        40: . . . . SEQUENCE
0235 06 03         3: . . . . . OID 2.5.4.11: OU
                 : 55 04 0b
0240 13 21        33: . . . . . PrintableString
                 'Dept. Arquitectura de Computadors'
                 : 44 65 70 74 2e 20 41 72 71 75 69 74 65 63 74 75
                 : 72 61 20 64 65 20 43 6f 6d 70 75 74 61 64 6f 72
                : 73
0275 31 19        22: . . . SET
0277 30 17        20: . . . . SEQUENCE
0279 06 03         3: . . . . . OID 2.5.4.3: CN
                 : 55 04 03
0284 13 10        16: . . . . . PrintableString 'Francisco Jordan'
                 : 46 72 61 6e 63 69 73 63 6f 20 4a 6f 72 64 61 6e
0302 30 7c         2: . . SEQUENCE // subjectPublicKeyInfo:: SEQUENCE 类型(30),
长度不定
0304 30 0d        13: . . . SEQUENCE
0306 06 09         9: . . . . OID 1.2.840.113549.1.1.1: RSAEncryption //algorithm::
OBJECT IDENTIFIER 类型,长度 9
                 : 2a 86 48 86 f7 0d 01 01 01 //  表示 RSA 算法(见注 1)
0317 05 00         0: . . . . NULL
0319 03 6b       107: . . . BIT STRING   (0 unused bits) // subjectPublicKey::
公钥值，BIT STRING 类型，长度 107 字节
                 : 00    (0 unused bits)
0321 03 68       104: . . . . BIT STRING   (0 unused bits)
0323 02 61        97: . . . . . INTEGER (0 unused bits) //  公钥值，96 字节，768 位
                 : 00    (0 unused bits)
: be aa 8b 77 54 a3 af ca 77 9f 2f b0 cf 43 88 ff
                 : a6 6d 79 55 5b 61 8c 68 ec 48 1e 8a 86 38 a4 fe
                 : 19 b8 62 17 1d 9d 0f 47 2c ff 63 8f 29 91 04 d1
                 : 52 bc 7f 67 b6 b2 8f 74 55 c1 33 21 6c 8f ab 01
                 : 95 24 c8 b2 73 93 9d 22 61 50 a9 35 fb 9d 57 50
                 : 32 ef 56 52 50 93 ab b1 88 94 78 56 15 c6 1c 8b
0423 02 03         3: . . . . . INTEGER    // RSA 加密算法的 exponent 值
                 : 01 00 01
0428 a3 81 97      151: . . [3]         // extensions::  特殊内容-证书扩展部分(a3),
长度 151
0431 30 3c         60: . . . SEQUENCE
0433 30 1f         31: . . . . SEQUENCE        //  扩展发行者密钥标识符
authorityKeyIdentifier
0435 06 03         3: . . . . . OID 2.5.29.35: authorityKeyIdentifier
                 : 55 1d 23
0440 04 14         22: . . . . . OCTET STRING
```

: 30 12 80 10 0e 6b 3a bf 04 ea 04 c3 0e 6b 3a bf
: 04 ea 04 c3
0464 30 19 25: . . . . SEQUENCE // 扩展 keyUsage
0466 06 03 3: . . . . . OID 2.5.29.15: keyUsage
: 55 1d 0f
0471 01 01 1: . . . . . TRUE
: ff
0474 04 04 4: . . . . . OCTET STRING
: 03 02 07 80
0480 30 19 25: . . . . SEQUENCE //扩展 certificatePolicies
0482 06 03 3: . . . . . OID 2.5.29.32: certificatePolicies
: 55 1d 20
0487 04 21 33: . . . . . OCTET STRING
: 30 1f 30 1d 06 04 2a 84 80 00 30 15 30 07 06 05
: 2a 84 80 00 01 30 0a 06 05 2a 84 80 00 02 02 01
: 0a
0522 30 1c 28: . . . . SEQUENCE //扩展 subjectAltName
0524 06 03 3: . . . . . OID 2.5.29.17: subjectAltName
: 55 1d 11
0529 04 15 21: . . . . . OCTET STRING
: 30 13 86 11 68 74 74 70 3a 2f 2f 61 63 2e 75 70
: 63 2e 65 73 2f
0552 30 19 25: . . . . SEQUENCE //扩展 issuerAltName
0554 06 03 3: . . . . . OID 2.5.29.18: issuerAltName
: 55 1d 12
0559 04 12 18: . . . . . OCTET STRING
: 30 14 86 12 68 74 74 70 3a 2f 2f 77 77 77 2e 75
: 70 63 2e 65
0579 30 80 : . SEQUENCE (indefinite length) // signatureAlgorithm
不知为何这里的前面算法为空
0581 06 07 7: . . OID
0583 05 00 0: . . NULL
0585 00 00 0: . . end of contents marker
0587 03 81 81 47: . BIT STRING // 签名值
: 00 (0 unused bits)
: 5c 01 bd b5 41 88 87 7a 0e d3 0e 6b 3a bf 04 ea
: 04 cb 5f 61 72 3c a3 bd 78 f5 66 17 fe 37 3a ab
: eb 67 bf b7 da a8 38 f6 33 15 71 75 2f b9 8c 91
: a0 e4 87 ba 4b 43 a0 22 8f d3 a9 86 43 89 e6 50
: 5c 01 bd b5 41 88 87 7a 0e d3 0e 6b 3a bf 04 ea
: 04 cb 5f 61 72 3c a3 bd 78 f5 66 17 fe 37 3a ab
: eb 67 bf b7 da a8 38 f6 33 15 71 75 2f b9 8c 91
: a0 e4 87 ba 4b 43 a0 22 8f d3 a9 86 43 89 e6 50
0637 00 00 0: . . end of contents marker

注 1: OID 表示的算法
DSA -- 1.2.840.10040.4.1
sha1DSA -- 1.2.840.10040.4.3
RSA -- 1.2.840.113549.1.1.1
md2RSA -- 1.2.840.113549.1.1.2
md4RSA -- 1.2.840.113549.1.1.3
md5RSA -- 1.2.840.113549.1.1.4
sha1RSA -- 1.2.840.113549.1.1.5

证书生成及转换

1 生成证书

方法 1

1. 生成私钥 Generate the private key 请使用以下命令来生成私钥:

openssl genrsa -des3 -out my.key 1024

查看 KEY 信息

openssl rsa -noout -text -in my.key

如上所示，此命令将生成 1024 位的 RSA 私钥，私钥文件名为： my.key，会提示您设定私钥密码，请设置密码，并牢记!

2. 生成 CSR 文件 Generate the CSR(证书签名请求文件) 请使用以下命令来生成 CSR

openssl req -new -key my.key -out my.csr #查看 CSR 信息 openssl req -noout -text -in my.csr

此命令将提示您输入 X.509 证书所要求的字段信息，包括国家(中国添 CN)、省份、所在城市、单位名称、单位部门名称 (可以不填直接回车)。 请注意：除国家缩写必须填 CN 外，其余都可以是英文或中文。 请输入您要申请 SSL 证书的域名，如果您需要为申请 SSL 证书就不能只输入 domain.com。SSL 证书是严格绑定域名 的。 请不要输入 Email、口令(challenge password)和可选的公司名称，直接打回车即可。

3. 用私钥给证书签名

openssl x509 -req -days 365 -in my.csr -signkey my.key -out my.crt

4. 查看证书信息 openssl x509 -noout -text -in my.crt

方法 2

1. 用 openssl 创建 CA 证书的 RSA 密钥(PEM 格式):

openssl genrsa -des3 -out ca.key 1024

2. 用 openssl 创建 CA 证书(PEM 格式,假如有效期为一年)：

openssl req -new -x509 -days 365 -key ca.key -out ca.crt

openssl 是可以生成 DER 格式的 CA 证书的，好用 IE 将 PEM 格式的 CA 证书转

换成 DER 格式的 CA 证书。 查看证书:

openssl x509 -in ca.crt -noout -text

2 证书类型转换

1. x509 到 pfx

openssl pkcs12 -export -out server.pfx -inkey server.key -in server.crt

2. PEM 格式的 ca.key 转换为 Microso 可以识别的 pvk 格式。

pkv -in ca.key -out ca.pvk -nocrypt -topvk

#没有安装 pvk 的同学可以用下面命令代替

openssl rsa -in ca.key -outform PVK -pvk-strong -out ca.pvk

3. pem 转 PKCS12

openssl pkcs12 -export -inkey my.key -in my.crt -out my.p12

4. PKCS#12 到 PEM 的转换

openssl pkcs12 -nocerts -nodes -in cert.p12 -out private.pem #验证 openssl pkcs12 -clcerts -nokeys -in cert.p12 -out cert.pem

5. 从 PFX 格式文件中提取私钥格式文件 (.key)

openssl pkcs12 -in mycert.pfx -nocerts -nodes -out mycert.key

6. 转换 pem 到到 spc

openssl cr12pkcs7 -nocrl -certfile venus.pem -outform DER -out venus.spc

用 -ouorm -inform 指定 DER 还是 PAM 格式。例如：

openssl x509 -in Cert.pem -inform PEM -out cert.

7. PEM 到 PKCS#12 的转换

openssl pkcs12 -export -in Cert.pem -out Cert.p12 -inkey key.pem openssl pkcd12 -export -inkey server.key -in server.crt -out server.pfx

(server.key 和 server.crt 文件是 Apache 的证书文件，生成的 server.pfx 用于导入 IIS)

数字信封生成、加密与解密

1、进行数字签名

1）包含证书和原文信息

openssl smime -sign -inkey prikey.pem -signer certself.pem -in install.log -out install_sign.msg

2）不包含证书信息

openssl smime -sign -inkey prikey.pem -signer certself.pem -passin pass:"123456" –nocerts -in install.log -out install_sign.msg

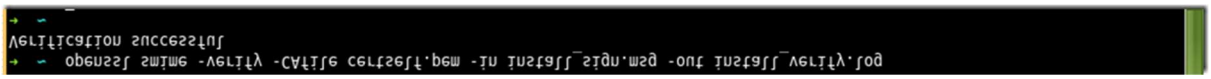
3）不包含原文

openssl smime -sign -inkey prikey.pem -signer certself.pem -passin pass:"123456" -nodetach -in install.log -out install_sign.msg

2、进行签名验证

1）包含证书和原文信息

openssl smime -verify -CAfile certself.pem -in install_sign.msg -out install_verify.log



2）不验证签名者证书信息

openssl smime -verify -noverify -CAfile certself.pem -signer certself.pem -in text_sign.msg -out text_verify.log

3）不包含原文

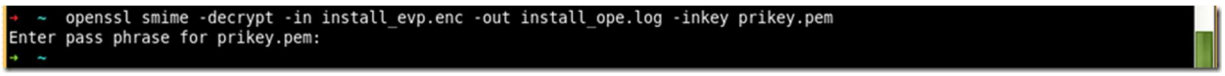
openssl smime -verify -nodetach -CAfile certself.pem -signer certself.pem -in text_sign.msg -out text_verify.log

3、进行数字信封加密

openssl smime -encrypt -in install.log -out install_evp.enc certself.pem

4、进行数字信封解密

openssl smime -decrypt -in install_evp.enc -out install_ope.log -inkey prikey.pem



5、smime 格式与 pkcs#7 格式的互转

openssl smime -in text_sign.msg -pk7out -out test_pkcs.pem

openssl pkcs7 -in test_pkcs.pem -text

6、对一个现存的消息添加一个签名者

openssl smime -resign -in mail.msg -signer newsign.pem -out mail2.msg