X.509 证书解析程序设计和实现——实验报告

一、X.509 证书结构描述

*这部分的内容保存在“X509 证书结构解析.txt”文件中

①X.509 证书的总体结构

```
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.
}
```

X.509 证书（Certificate）结构主要分成了三大部分：证书主体、证书签名算法标识、证书签名值三个部分。其中证书主体和证书签名算法标识的存储结构为结构体，证书签名值的存储格式为 BitString。

②证书主体（TBSCertificate）结构

```
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
}
```

证书主体（TBSCertificate）结构分成了 10 个部分，包括了证书版本号、证书序列号、证书签名算法标识、证书发行者名称、证书有效期、证书主体名称、证书公钥、证书发行者 ID、证书主体 ID、证书扩展部分。各部分的存储结构如下图所示。

1. 证书版本号

```
Version ::= INTEGER { v1(0), v2(1), v3(2) }                   -- 版本号（v1,v2,v3）
```

证书版本号的存储结构为整型数。

2. 证书序列号

```
CertificateSerialNumber ::= INTEGER                           -- 证书序列号存储结构
```

证书序列号的存储结构为整型数。

3. 证书签名算法标识

```
AlgorithmIdentifier ::= SEQUENCE {                            -- 签名算法标识存储结构
    algorithm               OBJECT IDENTIFIER,                -- 签名算法名称
    parameters              ANY DEFINED BY algorithm OPTIONAL -- 签名算法参数
}

parameters:                                                   -- DSA(DSS)算法时的参数,RSA算法没有此参数
    Dss-Parms ::= SEQUENCE {
        p             INTEGER,
        q             INTEGER,
        g             INTEGER
    }
```

证书签名算法标识的存储结构为 AlgorithmIdentifier，里面包含了算法名称和算法参数。算法名称使用 OID 结构保存，算法参数的存储结构为整型数。

4. 证书发行者名称、证书主体名称

```
Name ::= CHOICE {                                          -- 证书发行者名称存储结构
    RDNSequence
}

RDNSequence ::= SEQUENCE OF RelativeDistinguishedName

RelativeDistinguishedName ::= SET OF AttributeTypeAndValue

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

AttributeType ::= OBJECT IDENTIFIER

AttributeValue ::= ANY DEFINED BY AttributeType
```

证书发行者名称、证书主体名称均使用 Name 结构体进行存储，包含了 OID 结构和其他自定义结构。

5. 证书有效期

```
Validity ::= SEQUENCE {                                    -- 证书有效期存储结构
    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
}
```

证书有效期使用 Time 结构体进行存储。

6. 证书公钥

```
SubjectPublicKeyInfo ::= SEQUENCE {                        -- 证书公钥存储结构
    algorithm           AlgorithmIdentifier,               -- 公钥算法
    subjectPublicKey    BIT STRING                         -- 公钥值
}

subjectPublicKey:                                          -- RSA算法时的公钥值
    RSAPublicKey ::= SEQUENCE {
        modulus           INTEGER,
        publicExponent    INTEGER
    }
```

证书公钥使用 AlgorithmIdentifier 和 BitString 组成的结构体进行存储。

7. 证书发行者 ID、证书主体 ID

```
UniqueIdentifier ::= BIT STRING                            -- 证书唯一标识存储结构
```

证书发行者 ID、证书主体 ID 使用 BitString 进行存储。

8. 证书扩展部分

```
Extensions ::= SEQUENCE SIZE (1..MAX) OF Extension            -- 拓展部分存储结构
    Extension ::= SEQUENCE {
        extnID      OBJECT IDENTIFIER,
        critical    BOOLEAN DEFAULT FALSE,
        extnValue   OCTET STRING
    }
```

证书扩展部分使用 Extension 结构体进行存储。

③证书签名算法标识（signatureAlgorithm）结构

```
AlgorithmIdentifier ::= SEQUENCE {                          -- 签名算法标识存储结构
    algorithm               OBJECT IDENTIFIER,              -- 签名算法名称
    parameters              ANY DEFINED BY algorithm OPTIONAL -- 签名算法参数
}

parameters:                                                 -- DSA(DSS)算法时的参数,RSA算法没有此参数
    Dss-Parms ::= SEQUENCE {
        p             INTEGER,
        q             INTEGER,
        g             INTEGER
    }
```

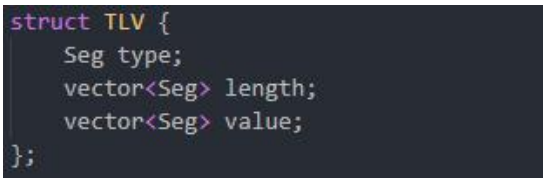
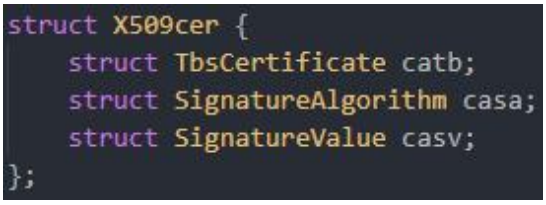
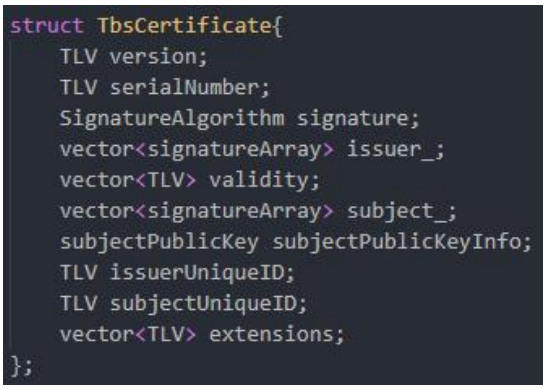
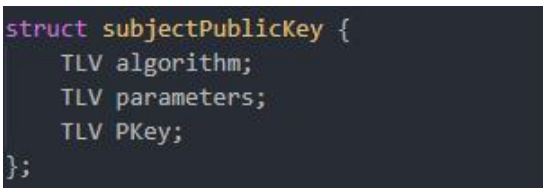
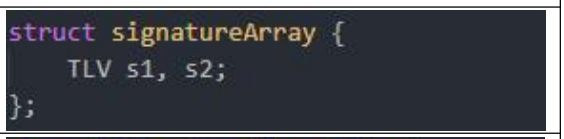
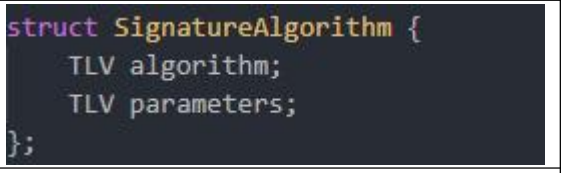
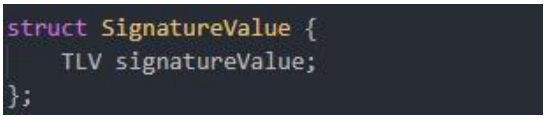
证书签名算法标识的存储结构为 AlgorithmIdentifier，里面包含了算法名称和算法参数。算法名称使用 OID 结构保存，算法参数的存储结构为整型数。

④证书签名值（signatureValue）结构

```
    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.
```

证书签名值的存储格式为 BitString。

二、数据结构

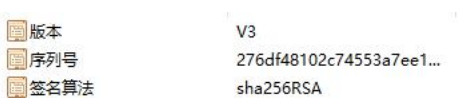
说明	代码截图
X.509 证书的编码格式使用（键[T]，长度[L]，值[V]）的格式，因此使用一个结构体进行证书内容的分割和保存。Seg 结构体用于保存字段的值，数据类型为整型数。	
X.509 证书的整体结构	
X.509 证书的证书主体的保存结构	
X.509 证书的公钥算法的保存结构	
X.509 证书的发行者和证书拥有主体的保存结构	
X.509 证书的签名算法的保存结构	
X.509 证书的签名值的保存结构	

三、编译运行结果

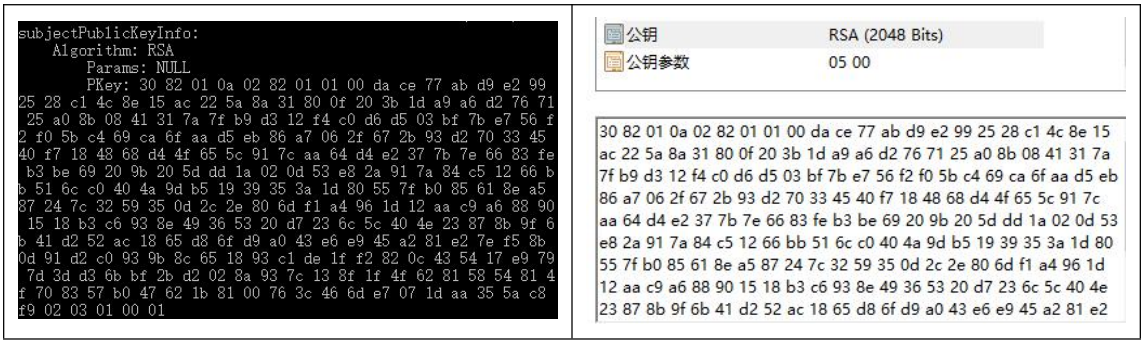
解析 DER_x509.cer 得到的结果

```
C:\Users\asus\Desktop\web安全\X.509>Reader.exe
Version: V3
SerialNumber: 276df48102c74553a7ee1258
SignatureAlgorithm:
    Algorithm: sha256RSA
    Params: NULL
Issuer:
    C = BE
    O = GlobalSign nv-sa
    CN = GlobalSign Organization Validation CA - SHA256 - G2
Validity:
    notBefore: 2018/09/18 09:32:07 GMT
    notAfter: 2020/09/18 09:21:04 GMT
Subject:
    C = CN
    S = e4 b8 8a e6 b5 b7 (UTF-8)
    L = e4 b8 8a e6 b5 b7 (UTF-8)
    O = e4 b8 8a e6 b5 b7 e5 b9 bb e7 94 b5 e4 bf a1 e6 81 af e7 a7 91 e6 8a 80 e6 9c 89 e9 99 90 e5 85 ac e5 8f b8 (UTF-8)
    CN = 2a 2e 62 69 6c 69 62 69 6c 69 2e 63 6f 6d (UTF-8)
subjectPublicKeyInfo:
    Algorithm: RSA
        Params: NULL
        PKey: 30 82 01 0a 02 82 01 01 00 da ce 77 ab d9 e2 99 25 28 c1 4c 8e 15 ac 22 5a 8a 31 80 0f 20 3b 1d a9 a6 d2 76 71 2
5 a0 8b 08 41 31 7a 7f b9 d3 12 f4 c0 d6 d5 03 bf 7b e7 56 f2 f0 5b c4 69 ca 6f aa d5 eb 86 a7 06 2f 67 2b 93 d2 70 33 45 40 f
7 18 48 68 d4 4f 65 5c 91 7c aa 64 d4 e2 37 7b 7e 66 83 fe b3 be 69 20 9b 20 5d dd 1a 02 0d 53 e8 2a 91 7a 84 c5 12 66 bb 51 6
c c0 40 4a 9d b5 19 39 35 3a 1d 80 55 7f b0 85 61 8e a5 87 24 7c 32 59 35 0d 2c 2e 80 6d f1 a4 96 1d 12 aa c9 a6 88 90 15 18 b
3 c6 93 8e 49 36 53 20 d7 23 6c 5c 40 4e 23 87 8b 9f 6b 41 d2 52 ac 18 65 d8 6f d9 a0 43 e6 e9 45 a2 81 e2 7e f5 8b 0d 91 d2 c
0 93 9b 8c 65 18 93 c1 de 1f f2 82 0c 43 54 17 e9 79 7d 3d d3 6b bf 2b d2 02 8a 93 7c 13 8f 1f 4f 62 81 58 54 81 4f 70 83 57 b
0 47 62 1b 81 00 76 3c 46 6d e7 07 1d aa 35 5a c8 f9 02 03 01 00 01
issuerUniqueID: NULL
subjectUniqueID: NULL
Extensions:
    Other: ellipsis
SignatureAlgorithm:
    Algorithm: sha256RSA
    Params: NULL
SignatureValue: 35 13 01 e0 20 2c 84 ce 76 c9 91 9f 8e 74 f1 5e 49 0d b9 2d 25 96 ae e6 87 02 52 ce 0e d7 64 71 81 8f 30 90 85
 24 e1 2c 17 9c 78 31 97 c7 e8 c2 b2 3d fd b7 b1 41 25 94 1b 45 79 d4 33 8c c0 1b 0c 0d 85 3b 8d 41 eb 0c 34 51 54 26 80 e6 a0
 d4 ac 75 b3 c9 e9 16 8b ae 9d bd 2f 9a 2c a2 29 49 20 aa 53 88 c7 70 64 ea d6 67 a3 e7 c4 48 f1 16 64 a5 7a 6b 93 b0 af 00 ee
 1c 5f 8d d2 07 b7 ec b7 da 1a d8 e2 07 01 37 e0 78 6a 1c d7 0d 9b 91 f0 7c 36 c6 8e f2 59 d0 0a f0 54 a8 db a3 f5 c3 1a 24 03
 38 86 b0 37 da 89 c1 70 35 c0 1e 02 a2 65 2a 95 68 b1 0e 40 56 0c 82 00 5d 8a 9f f1 50 d9 ed 4b 43 d9 59 c8 70 75 ab 85 37 13
 89 09 07 08 81 ca b2 0a bd b9 57 52 d0 8d 4e 9c 64 06 4a 87 e3 71 3d b5 47 91 a1 2d 0f 75 46 55 81 ea a1 31 64 ce 27 c5 59 2e
 bf b5 2c 82 07 a2 32 b9 91
```

比对结果

程序输出	Windows 显示
Version: V3 SerialNumber: 276df48102c74553a7ee1258 SignatureAlgorithm: Algorithm: sha256RSA Params: NULL	
Issuer: C = BE O = GlobalSign nv-sa CN = GlobalSign Organization Validation CA - SHA256 - G2	
Validity: notBefore: 2018/09/18 09:32:07 GMT notAfter: 2020/09/18 09:21:04 GMT 注：GMT 时间与北京时间有时差（+8）	
Subject: C = CN S = e4 b8 8a e6 b5 b7 (UTF-8) L = e4 b8 8a e6 b5 b7 (UTF-8) O = e4 b8 8a e6 b5 b7 e5 b9 bb e7 94 b5 e4 bf a1 e6 81 af e7 a7 91 e6 8a 80 e6 9c 89 e9 99 90 e5 85 ac e5 8f b8 (UTF-8) CN = 2a 2e 62 69 6c 69 62 69 6c 69 2e 63 6f 6d (UTF-8) 注：C++无法输出 UTF8 编码，下图为转码后得到的结果	

```
> decodeURI("%2a%2e%62%69%6c%69%62%69%6c%69%2e%63%6f%6d")
"*.bilibili.com"
> decodeURI("%e4%b8%8a%e6%b5%b7")
"上海"
> decodeURI("%e4%b8%8a%e6%b5%b7%e5%b9%bb%e7%94%b5%e4%bf%a1%e6%81%af%e7%a7%91%e6%8a%80%e6%9c%89%e9%99%90%e5%85%ac%e5%8f%b8")
"上海幻电信息科技有限公司"
```



得出结论

程序输出结果与 Windows 内置解析程序输出的结果一致，证明解析 X.509 证书成功，实验结果符合预期目标。

*证书的扩展部分由于不清楚其内容的实际含义，因此在本次实验中只对其进行了 TLV（键、长度、值）编码格式的解析，而没有对其内容进行翻译。