Structure Analysis and Generation of X.509 Digital Certificate Based on National Secret

Hua Jiang¹, Gang Zhang¹, Jinpo Fan¹
¹Communication Engineering Department, Beijing Electronics Science and Technology Institute, No. 7, Fu Feng Road, Beijing, China

Corresponding Author: email: 18810968002@163.com

Abstract: The X.509 format certificate is a widely used digital certificate standard and is a series of data used to identify the identity information of parties to a communication. The design is based on the national secret algorithm SM2, SM3, replacing the ECDSA algorithm and the SHA-256 algorithm in the strongswan source code with the SM2 and SM3 algorithms respectively, Replace the ECDSA-SHA256 certificate with the SM2-SM3 certificate and use the X.509 certificate format for the encryption product.

1. Preface
At present, the RSA signature algorithm has always occupied the dominant position of digital certificate authentication, and the SM2 algorithm researched independently in China is a new domestic asymmetric algorithm, which is more advanced than the RSA algorithm. Based on the national commercial password security and other reasons, the national password management department is being widely promoted nationwide, and the digital certificate based on the national secret algorithm should be further studied and promoted.

2. Basic knowledge
2.1 X.509 digital certificate
X.509 is the format standard for public key certificates in cryptography. X.509 certificate have been used in many Internet protocols including TLS/SSL (the cornerstone of WWW World Wide Web Safe Browsing). They are also used in many non-online applications, such as electronic signature services. The X.509 certificate contains the public key, identity information (such as the network host name, organization name or individual name, etc.) and signature information (which can be the signature of the certificate issuing authority CA or self-signed). For a certificate signed by a trusted certificate issuing authority or otherwise verifiable, the certificate owner can use the certificate and the corresponding private key to create a secure communication and digitally sign the document. X.509 is a set of certificate standards defined by the ITU-T standardization department based on their previous ASN.1 definition. The digital certificate based on SM2-SM3 is an X.509 digital certificate.

2.2 National secret algorithm
Like the RSA algorithm, the SM2 algorithm belongs to the asymmetric algorithm system and belongs to the Elliptic Curve Cryptography (ECC) algorithm. However, unlike the RSA algorithm, the RSA algorithm is based on the mathematical problem of large integer decomposition. The SM2 algorithm is
3. Analysis of X.509 digital certificate structure

![Figure 1. X.509 certificate diagram](image)

3.1 the basic part of the X.509 certificate

3.1.1 Version number
Identifies the version of the certificate (version 1, version 2, or version 3).

3.1.2 Serial number
A unique integer identifying the certificate. A unique identifier for this certificate assigned by the certificate issuer.

3.1.3 Signature
The algorithm identifier used for the visa book, consisting of the object identifier plus the relevant parameters, is used to describe the digital signature algorithm used in this certificate. For example, the object identifiers for SHA-1 and RSA are used to indicate that the digital signature uses SHA-1 hashes.

3.1.4 Issuer
The distinguished name (DN) of the certificate issuer.

3.1.5 Validity period
The period of time during which the certificate is valid. This field consists of "Not Before" and "Not After", which are represented by UTC time or normal time, respectively.

3.1.6 Subject
The distinguished name of the certificate owner. This field must be non-null unless you have an alias in the certificate extension.

3.1.7 Subject public key information
The public key of the principal (and the algorithm identifier).

3.1.8 Issuer unique identifier
Identifier—The unique identifier of the issuer of the certificate, which is optional.

3.1.9 Subject unique identifier
The unique identifier of the certificate owner, which is optional.

3.1.10 X.509 certificate extension
Optional standards and specialized extensions, the elements of the extension have this structure:
Extension ::= SEQUENCE {
  extnID OBJECT IDENTIFIER,
  Critical BOOLEAN DEFAULT FALSE,
  extnValue OCTET STRING
}
extnID: indicates the OID of an extension element
Critical: indicates whether this extension element is extremely important
extnValue: indicates the value of this extension element, the string type.

4. software design
The research results are mainly used in the design of strongswan-based IPSec VPN gateway, written in Python language, using ECDSA-SHA-256 certificate as a template to modify and replace and generate a new SM2-SM3 certificate.

4.1 SM2 algorithm replacement
In strongswan, the ECDSA (Elliptic Curve Digital Signature Algorithm) in openssl is called by default to implement the signature verification function. Therefore, the key pair, signature, and checksum of SM2 are implemented by modifying the source code in ECDSA. Mainly change the parameters of the SECP256k1 curve in the ECDSA source code to the parameters of SM2, and add sm2verify(), sm2sign(), sm2verify_digest(), sm2sign_digest(), sm2sign_number() and other functions to the source code. Finally, through the national standard algorithm test V1.3.3 software for SM2 to generate key pairs, signatures, check and comparison, fully compatible with each other, fully interoperable, indicating correct rewriting.

4.2 SM3 algorithm replacement
The hash algorithm that is loaded by default in strongswan is sha-1, the digest value of sha-1 is 96 bits or 160 bits, and the length of SM3 algorithm is 256 bits. The digest value of the sha-256 algorithm is 256 bits long, which corresponds to SM3. Write SM3.py with python3.5, the specific code will not go into details.

4.3 Interface UI Design
This interface is developed using eric6 software. Eric6 is an IDE program for the Python programming language. It is powerful and does not lose any IDE program under the Python platform. It takes up less memory and runs faster. The most important thing is that it is seamless with PyQt5. PyQt5 is based on the Python programming language. The external GUI development language, its solid underlying foundation and powerful visual interface design make PyQt5 a leader in Python language GUI development. The update speed is fast, and the speed of developing GUI programs is fast. It can be said that other GUI development languages are beyond the reach.

The Qt designer provided by eric6 software is used to design the overall interface framework, which is designed with the public key, the serial number of the certificate, the certificate issuer, the effective start and end date of the certificate, and the user of the certificate.

The power of eric6 software lies in the separation of interface and logic. The .py file here is
compiled from .ui file, so when the .ui file changes, the corresponding .py file will also change. The .py file compiled from the .ui file is called the interface file. Since the interface file is initialized each time it is compiled, you need to create a new .py file to call the interface file. This new file is called a logical file or a business file. In the logical file, write the code, extract the certificate body in the digital certificate, replace the public key in the main body with the public key provided by the external to obtain the new subject, SM3 hashes the new subject, and then uses SM2 to sign. Get a 64-bit new signature value, and finally convert the decimal data entered in the interface box into a hexadecimal certificate format by Ascill encoding. Finally, the data is integrated according to the X.509 digital certificate encoding method and displayed in the box of the new certificate. Finally, the certificate data can be saved in the desired format. Figure 2 below shows the interface for generating a CA certificate.

Figure 2 interface framework

Figure 3 generates a certificate

5. Analysis of certificate structure examples
Certificate analysis:

| address | content | significance |
|---------|---------|--------------|
| 0000    | 30 82 01 97 | Certificate header, SEQUENCE type (30), data block length byte is 2 (82), length is 407 (01 97) |
| 0004    | 30 82 01 3C | The certificate body header, SEQUENCE type (30), data block length byte is 2 (82), length is 316 (01 3C) |
| 0008    | A0 03 02 01 02 | Special Content - Certificate Version (A3), Length 3, Integer Type (02), Length 1, Version 3 (2) |
| 0013    | 02 08 6D 50 2E BC 5E F2 BD 98 | Integer type (02), length 8 (08), certificate serial number (6D 50 2E BC 5E F2 BD 98) |
| 0023    | 30 0A 06 08 2A 86 48 CE 3D 04 03 02 | SEQUENCE type (30), length 10 (0A), OBJECT IDENTIFIER type (06), length 8, algorithm (2A 86 48 CE 3D 04 03 02) |
| 0035    | 30 2F 31 0B 30 09 06 03 55 04 06 13 02 43 48 31 30 11 06 03 55 04 0A 13 0A 73 74 72 6F 6E 67 53 77 61 6E 31 0B 30 09 06 03 55 04 03 13 02 43 41 | Certificate issuer: SEQUENCE type (30), length 47 (2F) SET starts a collection with a length of 11 (0B), id-at-countryName (55 04 06), corresponding to CH (43 48) SET starts a collection with a length of 19 (13), Id-at-organizationUnitName (05 04 0A), corresponding to strongSwan (73 74 72 6F 6E 67 53 77 61 6E) SET starts a collection with a length of 11 (0B), Id-at-commonName (55 04 03), corresponding to CA (43 41) |
| 0084    | 30 1E 17 0D 31 38 30 33 32 31 30 34 35 35 33 5A 17 0D 32 31 30 33 32 30 31 30 34 35 35 33 5A | SEQUENCE type (30), length 30 (2F) Notbefore: UTCTime type (23), length 13, UTCTime '180321104553Z' Notafter: UTCTime type (23), length 13, UTCTime '210320104553Z' |
| 0116    | 30 2F 31 0B 30 09 06 03 55 04 06 13 02 43 48 31 13 30 11 06 03 55 04 0A 13 0A 73 74 72 6F 6E 67 53 77 61 6E 31 0B 30 09 06 03 55 04 03 13 02 43 41 | The following data block represents the body information, length 47 (2F), SET starts a collection with a length of 11 (0B), id-at-countryName (55 04 06), corresponding to CH (43 48) SET starts a collection with a length of 19 (13), Id-at-organizationUnitName (05 04 0A), corresponding to strongSwan (73 74 72 6F 6E 67 53 77 61 6E) SET starts a collection with a length of 11 (0B), Id-at-commonName (55 04 03), corresponding to CA (43 41) |
| 0165    | 30 59 30 13 06 07 2A 86 48 CE 3D 02 01 06 08 2A 86 48 CE 3D 03 01 07 03 | SEQUENCE type (30), length 89 (59)Key parameter (06 08 2A 86 48 CE 3D 03 01 07) public key value, expressed as integer |
6. Conclusion
In this paper, the generation of digital certificates based on national secret algorithm has been realized. The python source code library for ECDSA be amended as SM2 algorithm. Support national secret algorithms to meet the needs of national security and economic development. This paper also analyzes the format of the X.509 digital certificate and elaborates on the meaning of each part of the digital certificate. You can freely set the contents of the certificate to generate and save. However, there are also some problems that need to be solved. For example, the generated certificate is transmitted to the encryption board of the IPSec VPN to facilitate the reading of the encryption card.

References
[1] Mukkamala R, Balusani S. Active Certificates: A New Paradigm in Digital Certificate Management[C]// International Conference on Parallel Processing Workshops. IEEE Computer Society, 2002:30.
[2] Zhang J, Hu N, Raja M K. Digital certificate management: Optimal pricing and CRL releasing strategies[J]. Decision Support Systems, 2014, 58(58):74-78.
[3] Maes S H, Sedivy J. Portable information and transaction processing system and method utilizing biometric authorization and digital certificate security[J]. Lancet, 1999, 1(7793):32.
[4] Maes S H, Sedivy J. Portable information and transaction processing system and method utilizing biometric authorization and digital certificate security[J]. Lancet, 1999, 1(7793):32.
[5] Han S L, Min M A, Wang T. Design and Implementation of Digital Certificate Application System[J]. Netinfo Security, 2012.
[6] Haiwen O U, Wang Y, Ouyang C, et al. SM2-based digital certificate parsing and validity verification[J]. Journal of Computer Applications, 2016.
[7] Harwani B M. Introduction to Python Programming and Developing GUI Applications with PyQT[J]. Chromatographia, 2011, 32(12):1088.
[8] Bouda, Peter, Bouda, et al. Pyqt Und Pyside: Gui- Und Anwendungsentwicklung Mit Python Und Qt[J]. Open Source Press, 2015.