Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt. The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html. For potential updates to the above required-text see: http://www.ietf.org/ietf/1id-guidelines.txt

Abstract

A traditional X.509 v3 certificate format contains only one public key. Nowadays, many Internet applications like SET, S/MIME and IPSEC etc. need more than one public key. This document describes a Multiple-Public-Key (MPK) certificate format to include multiple public keys in a digital certificate. The MPK certificate format is not only compatible with X.509 v3 certificate format; but also can reduce the length of multiple certificates and speed up the verification time needed for multiple public keys.

Table of Contents
1. Introduction...................................................1
2. Why needs a Multiple-Public Key (MPK) certificate............2 3.
3. Definitions....................................................2
5. Multiple-Public-Key (MPK) certificate syntax..................6 6.
6. References......................................................8
7. Contact........................................................8

1. Introduction

Public-key cryptography and certificates now are
regarded as the preferred enablers of strong security for many applications, for example, Web-service, e-mail, VPN, etc. The technique of public-key cryptography uses pairs of keys: a public key that is broadly available, and a corresponding private key known only to the user.

Chen-Chi

Multiple-Public-Key (MPK) Certificate Format
September, 2001

(i.e. also can be either service or application) that owns the key. The public key certificate is used to vouch the authenticity of this public key. Public-key certificates, such as X.509 certificates [1], are regarded as the base for the skeletal structure of trust within a distributed public-key cryptosystem. By signing a certificate, a certificate issuer, such as CA (Certificate Authority), binds a public key to an entity, for example like a person, service, or application, together with entity's other information. By verifying the signature of the certificate can establish trust in the entity's public key.

2. Why needs a Multiple-Public Key (MPK) certificate

Most major security standards, such as Secure Multipurpose Internet Mail Extensions (S/MIME) [2], IP Security (IPSEC) [3], Secure Sockets Layer (SSL)/Transport Layer Security (TLS) [4] and Secure Electronic Transactions (SET) [5] rely on PKI. In some applications, each user needs two or more public keys. For instance, in SET standard, except the cardholder, there are two pairs of public/private keys associated with each merchant and payment gateway. One pair of keys is used for signing messages and the other pair of keys is used for key exchange. Using current X.509 v3 certificate format, there are two digital certificates for each merchant and payment gateway. In the processes of merchant registration, purchase request, payment authorization and payment capture, these two digital certificates need to be sent and verified frequently. In S/MIME, each user kept at least two keys for signing certificate and key exchange and, sometimes, has one more extra encryption public key.

Our motivation of this proposal comes from why a certificate cannot contain more than one public key if applications need more than one public key. So, we propose a Multiple-Public-Key (MPK) certificate format to include multiple public keys in one certificate. The MPK certificate format is not only compatible with X.509 v3 [1] certificate format; but also can reduce the length of multiple certificates and speed up the verification time needed for multiple public keys.

3. Definitions

The basic definitions apply for the document.
ASN.1: Abstract Syntax Notation One, as defined in X.208 [6].

BER: Basic Encoding Rules, as defined in X.209 [7].

DER: Distinguished Encoding Rules for ASN.1, as defined in X.509.

Certificate: A type that binds an entity’s distinguished name to a public key with a digital signature of a certificate issuer. This type is defined in X.509 v3 [1]. This type also contains an issuer-specific serial number, the distinguished name of the certificate issuer (the signer), the issuer’s signature algorithm identifier, a validity period and some optional parameters. Session 4 gives more information.

4. Review of X.509 v3 public-key certificates

The most universally supported PKI standard is the X.509. The X.509 has a primary purpose that is to define a standard digital certificate format because certificates are the most important part of an interoperable public-key standard.

This session describes the syntax of X.509 v3 certificates, for reference purposes. To be precise, it describes the newest syntax given in RFC 2459 [9].

An X.509 certificate consists three parts: certificate information, a signature algorithm identifier, and a digital signature on the certificate information. The certificate information consists of an issuer-specific serial number, a signature algorithm identifier, the distinguished name of the certificate issuer, a validity period, an entity’s distinguished name, the entity’s public key and some optional and extension attributes.

The process by which the certificate is constructed involves the following steps:

The certificate issuer constructs a CertificateInfo value containing certificate information.

The CertificateInfo value is signed with the certificate issuer’s private key.

The CertificateInfo value, a signature algorithm identifier, and the certificate issuer’s signature are collected together into a Certificate value.

The following discussion is divided into two parts. The first part describes the certificate information type CertificateInfo, and the second part describes the top-level type Certificate.
4.1 CertificateInfo

Certificate information has ASN.1 type CertificateInfo:

CertificateInfo ::= SEQUENCE { version
[0]                  EXPLICIT Version DEFAULT v1,
serialNumber                 CertificateSerialNumber,
issuer                       Name,
validity                     Validity,
subject                      Name,

Chen-Chi Category -
Expiration 3

Multiple-Public-Key(MPK) Certificate Format
September,2001

subjectPublicKeyInfo SubjectPublicKeyInfo, *
issuerUniqueID [1] IMPLICIT UniqueIdentifier
OPTIONAL
-- If present, version
shall be v2 or v3
* subjectUniqueID[2] IMPLICIT
UniqueIdentifier OPTIONAL
-- If present, version
shall be v2 or v3
* extensions [3] EXPLICIT Extensions
OPTIONAL
-- If present, version
shall be v3

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

CertificateSerialNumber ::= INTEGER
Validity ::= SEQUENCE {
  NotBefore UTCTime,
  NotAfter UTCTime }

SubjectPublicKeyInfo ::= SEQUENCE {
  algorithm AlgorithmIdentifier,
  subjectPublicKey BIT STRING }

AlgorithmIdentifier ::= SEQUENCE {
  Algorithm OBJECT IDENTIFIER,
  Parameters ANY DEFINED ALGORITHM OPTIONAL}

UniqueIdentifier ::= BIT STRING

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

Extension ::= SEQUENCE {
  ExtnID OBJECT IDENTIFIER,
  Critical BOOLEAN DEFAULT FALSE,
  ExtnValue OCTET STRING }

The fields of type CertificateInfo have the following meanings:

version: is the version number, for compatibility
with future revisions of X.509. Its default value is
v1, to which the Version type assigns the integer 0.
The [0] tag on version is an explicit tag.

serialNumber: is the issuer-specific serial number of
the certificate. Every certificate for a particular
issuer must have a different serial number.

signature: identifies the issuer’s signature algorithm (and any associated parameters).

issuer: is the distinguished name of the certificate issuer.

validity: is the validity period for the certificate. The validity period specifies the points in time between which the certificate is considered valid.

subject: is the distinguished name of the certificate subject (the entity whose public key is certified).

subjectPublicKeyInfo: contains information about the public key being certified. The information identifies the entity’s public-key algorithm (and any associated parameters).

issuerUniqueID and subjectUniqueID: may only appear if the version 2 or 3.

extensions may only appear if the version is 3. If present, this field is a SEQUENCE of one or more certificate extensions.

Note. issuerUniqueID, subjectUniqueID and extensions fields marked asterisk are optional.

4.2 Certificate

An X.509 certificate has ASN.1 type Certificate:

Certificate ::= SEQUENCE {
  certificateInfo CertificateInfo,
  signatureAlgorithm AlgorithmIdentifier,
  signature BIT STRING }

The fields of type Certificate have the following meanings:

certificateInfo: is the certificate information. It is the value being signed.

signatureAlgorithm: identifies the signature algorithm (and any associated parameters) under which the certificate information is signed. Examples include PKCS #1’s md2WithRSAEncryption and md5WithRSAEncryption. The value of this field should be the same as the value of the signature field of the certificate information.

signature: is the result of signing the certificate information with the certificate issuer’s private key.

The signature process consists of two steps:
a. The value of the certificateInfo field is DER encoded, yielding an octet string.

b. The result of step a is signed with the certificate issuer’s private key under the specified signature algorithm, yielding a bit string, the signature.

Note. The syntax for Certificate is usually written with the X.509 SIGNED macro.

Certificate ::= SIGNED SEQUENCE {
  version [0] Version DEFAULT v1988, serialNumber CertificateSerialNumber,

Chen-Chi Category -
Expiration 5

Multiple-Public-Key (MPK) Certificate Format
September, 2001

signature AlgorithmIdentifier, issuer Name, validity Validity, subject Name, subjectPublicKeyInfo SubjectPublicKeyInfo, *
issuerUniqueID [1] IMPLICIT UniqueIdentifier OPTIONAL
  -- If present, version shall be v2 or v3
* subjectUniqueID [2] IMPLICIT UniqueIdentifier OPTIONAL
  -- If present, version shall be v2 or v3
* extensions [3] EXPLICIT Extensions OPTIONAL
  -- If present, version shall be v3
}

The SIGNED macro has been expanded in this discussion for simplicity.

5. Multiple-Public-Key (MPK) certificate syntax

This document exports one type, MPKCertificate. This session gives the syntax for MPK certificate: MPKCertificate.

A MPK certificate consists of three parts: MPK-certificate information, a signature algorithm identifier, and a digital signature on the MPK-certificate information.

The following discussion is divided into two parts. The first part describes the certificate information type MPKCertificateInfo, and the second part describes the top-level type MPKCertificate.

5.1. MPKCertificateInfo

MPK-certificate information shall have ANS.1-type MPKCertificateInfo:

MPKCertificateInfo ::= SEQUENCE OF ( SEQUENCE {
  ecertificateInfo ECertificateInfo ))
EcertificateInfo ::= SEQUENCE {
  version [0] EXPLICIT Version DEFAULT v1 OPTIONAL,
  serialNumber CertificateSerialNumber OPTIONAL,
  issuer Name OPTIONAL,
  validity Validity OPTIONAL,
  subject Name OPTIONAL,
  subjectPublicKeyInfo SubjectPublicKeyInfo, *
  issuerUniqueID [1] IMPLICIT UniqueIdentifier OPTIONAL
    -- If present, version shall be v2 or v3
  * subjectUniqueID [2] IMPLICIT UniqueIdentifier OPTIONAL
    -- If present, version shall be v2 or v3
  * extensions [3] EXPLICIT Extensions OPTIONAL
    -- If present, version shall be v3
}

Chen-Chi
Expiration 6

Multiple-Public-Key(MPK) Certificate Format
September,2001

ANS.1-type MPKCertificateInfo consists of a SEQUENCE of one or more certificate information, ECertificateInfo. It is the only difference between CertificateInfo and ECertificateInfo that each field of type ECertificateInfo is marked optional except subjectPublicKeyInfo and issuerUniqueID, subjectUniqueID and extensions fields marked asterisk still remain the same definition as in session 4.1. In MPK certificate, ASN.1-type ECertificateInfo can be divided into two styles:

The first ECertificateInfo of MPKCertificateInfo is called the base certificate information unit. In base certificate information unit, each field of type ECertificateInfo cannot be optional.

Every other ECertificateInfo of MPKCertificateInfo is called the reference certificate information unit. In reference certificate information unit, each field of type ECertificateInfo can be optional or not according to base certificate information unit and the subjectPublicKeyInfo field is mandatory.

5.2. MPKCertificate

A MPK certificate shall have ASN.1-type MPKCertificate:

MPKCertificate ::= SEQUENCE {
  mpkCertificateInfo MPKCertificateInfo,
  signatureAlgorithm SignatureAlgorithmIdentifier,
  signature Signature
}

SignatureAlgorithmIdentifier ::= AlgorithmIdentifier

Signature ::= BIT STRING

The fields of type MPKCertificate have the following
meanings:

mpkCertificateInfo is the MPK-certificate information. It is the value being signed.

signatureAlgorithm identifies the signature algorithm (and any associated parameters) under which the MPK-certificate information is signed. Examples include PKCS #1’s md2WithRSAEncryption and md5WithRSAEncryption.

signature is the result of signing the MPK-certificate information with the certificate issuer’s private key.

The signature process consists of two steps:

1. The value of the mpkCertificateInfo field is DER encoded in accordance with the inheritance rules, yielding an octet string. The inheritance rules are summarized as follows two rules:

   a. To save space, the value of the field of the reference certificate information unit needs to omit and inherits the value of the field of the base certificate information unit if it is duplicate to the value of the field of the base certificate information unit.

   b. To keep base certificate information unit and reference certificate information unit independently, issuerUniqueID, subjectUniqueID and extensions fields masked on asterisk of the reference certificate unit dont inherit from the base certificate unit.

   Therefore, the value of mpkCertificateInfo field consists of the value of one base certificate information unit and the value of zero or more reference certificate information units. The MPK certificate is built by one or more X.509 v3 certificate so it can be compa

tible with X.509 v3.

2. The result of step 1 is signed with the certificate issuer’s private key under the specified signature algorithm, yielding a bit string, the signature.

Note. The syntax for MPKCertificate could equivalently be written with the X.509 SIGNED macro:

MPKCertificate ::= SIGNED MPKCertificateInfo

6. References

International Standard 9594-8 | ITU-T

7. Contact:

Chen-Chi Lin National Cheng Kung University
lchenchi@chtti.com.tw

Chen-Chi Category - Expiration

Lein Harn University of Missouri-Kansas City
harn@cstp.umkc.edu

Chi-Sung Laih
National Cheng Kung University
laihcs@eembox.ee.ncku.edu.tw

Chen-Chi Category - Expiration