Internet Public Key Infrastructure

Part I: X.509 Certificate and CRL Profile

<draft-ietf-pkix-ipki-part1-02.txt>

Status of this Memo

This document is an Internet-Draft. 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."

To learn the current status of any Internet-Draft, please check the
"1id-abstracts.txt" listing contained in the Internet-Drafts Shadow
Directories on ftp.is.co.za (Africa), nic.nordu.net (Europe),
munnari.oz.au (Pacific Rim), ds.internic.net (US East Coast), or
ftp.isi.edu (US West Coast).

Abstract

This is the second draft of the Internet Public Key Infrastructure
X.509 Certificate and CRL Profile. Since the first version was
distributed, ISO has completed work on X.509 Version 3 Certificates
and X.509 Version 2 Certificate Revocation Lists (CRLs). Many of the
Internet community requirements that were in the previous version of
this document have been included in the final ISO document. As a
result, this document has gotten simpler. Please send comments on
this document to the ietf-pkix@tandem.com mail list.

1 Executive Summary

<< Write this last. >>
2 Requirements and Assumptions

Goal is to develop a profile and associated management structure to facilitate the adoption/use of X.509 certificates within Internet applications for those communities wishing to make use of X.509 technology. Such applications may include WWW, electronic mail, user authentication, electronic payment systems, IPSEC, as well as others. In order to relieve some of the obstacles to using X.509 certificates, this document defines a profile to promote the development of certificate management systems; development of application tools; and interoperability determined by policy, as opposed to syntax.

Some communities will need to supplement, or possibly replace, this profile in order to meet the requirements of specialized application domains or environments with additional authorization, assurance, or operational requirements. However, for basic applications, common representations of frequently used attributes are defined so that application developers can obtain necessary information without regard to the issuer of a particular certificate or certificate revocation list (CRL).

As supplemental authorization and attribute management tools emerge, such as attribute certificates, it may be appropriate to limit the authenticated attributes that are included in a certificate. These other management tools may be more appropriate method of conveying many authenticated attributes.

2.1 Communication and Topology

The users of certificates will operate in a wide range of environments with respect to their communication topology, especially users of secure electronic mail. This profile supports users without high bandwidth, real-time IP connectivity, or high connection availability. In addition, the profile allows for the presence of firewall or other filtered communication.

2.2 Acceptability Criteria

The goal of the Internet Public Key Infrastructure (PKI) is to meet the needs of deterministic, automated identification, authentication, access control, and authorization functions. Support for these services determines the attributes contained in the certificate as well as the ancillary control information in the certificate such as policy data and certification path constraints.
2.3 User Expectations

Users of the Internet PKI are people and processes who use client software and are the subjects named in certificates. These uses include readers and writers of electronic mail, the clients for WWW browsers, and the key manager for IPSEC within a router. This profile recognizes the limitations of the platforms these users employ and the sophistication/attentiveness of the users themselves. This manifests itself in minimal user configuration responsibility (e.g., root keys, rules), explicit platform usage constraints within the certificate, certification path constraints which shield the user from many malicious actions, and applications which sensibly automate validation functions.

2.4 Administrator Expectations

As with users, the Internet PKI profile is structured to support the individuals who generally operate Certification Authorities (CAs). Providing administrators with unbounded choices increases the chances that a subtle CA administrator mistake will result in broad compromise. Also, unbounded choices greatly complicates the software that must process and validate the certificates created by the CA.
Overview of Approach

Following is a simplified view of the architectural model assumed by the PKIX specifications.

The components in this model are:

- **end entity**: user of PKI certificates and/or end user system that the PKI certifies;
- **CA**: certification authority;
- **RA**: registration authority, i.e., an optional system to which a CA delegates certain management functions;
- **repository**: a system or collection of distributed systems that store certificates and CRLs and serves as a means of distributing these certificates and CRLs to end entities.
entities.

3.1 X.509 Version 3 Certificate

Application of public key technology requires the user of a public key to be confident that the public key belongs to the correct remote subject (person or system) with which an encryption or digital signature mechanism will be used. This confidence is obtained through the use of public key certificates, which are data structures that bind public key values to subject identities. The binding is achieved by having a trusted certification authority (CA) digitally sign each certificate. A certificate has a limited valid lifetime which is indicated in its signed contents. Because a certificate’s signature and timeliness can be independently checked by a certificate-using client, certificates can be distributed via untrusted communications and server systems, and can be cached in unsecured storage in certificate-using systems.

The standard known as ITU-T X.509 (formerly CCITT X.509) or ISO/IEC 9594-8, which was first published in 1988 as part of the X.500 Directory recommendations, defines a standard certificate format. The certificate format in the 1988 standard is called the version 1 (v1) format. When X.500 was revised in 1993, two more fields were added, resulting in the version 2 (v2) format. These two fields are used to support directory access control.

The Internet Privacy Enhanced Mail (PEM) proposals, published in 1993, include specifications for a public key infrastructure based on X.509 v1 certificates [RFC 1422]. The experience gained in attempts to deploy RFC 1422 made it clear that the v1 and v2 certificate formats are deficient in several respects. Most importantly, more fields were needed to carry information which PEM design and implementation experience has proven necessary. In response to these new requirements, ISO/IEC and ANSI X9 developed the X.509 version 3 (v3) certificate format. The v3 format extends the v2 format by adding provision for additional extension fields. Particular extension field types may be specified in standards or may be defined and registered by any organization or community. In June 1996, standardization of the basic v3 format was completed [X.509-AM].

ISO/IEC and ANSI X9 have also developed a set of standard extensions for use in the v3 extensions field [X.509-AM][X.9.55]. These extensions can convey such data as additional subject identification information, key attribute information, policy information, and certification path constraints.

However, the ISO/IEC and ANSI standard extensions are very broad in their applicability. In order to develop interoperable
implementations of X.509 v3 systems for Internet use, it is necessary to specify a profile for use of the X.509 v3 extensions tailored for the Internet. It is one goal of this document to specify a profile for Internet WWW, electronic mail, and IPSEC applications. Environments with additional requirements may build on this profile or may replace it.

3.2 Certification Paths and Trust

A user of a security service requiring knowledge of a public key generally needs to obtain and validate a certificate containing the required public key. If the public-key user does not already hold an assured copy of the public key of the CA that signed the certificate, then it might need an additional certificate to obtain that public key. In general, a chain of multiple certificates may be needed, comprising a certificate of the public key owner (the end entity) signed by one CA, and zero or more additional certificates of CAs signed by other CAs. Such chains, called certification paths, are required because a public key user is only initialized with a limited number (often one) of assured CA public keys.

There are different ways in which CAs might be configured in order for public key users to be able to find certification paths. For PEM, RFC 1422 defined a rigid hierarchical structure of CAs. There are three types of PEM certification authority:

(a) Internet Policy Registration Authority (IPRA): This authority, operated under the auspices of the Internet Society, acts as the root of the PEM certification hierarchy at level 1. It issues certificates only for the next level of authorities, PCAs. All certification paths start with the IPRA.

(b) Policy Certification Authorities (PCAs): PCAs are at level 2 of the hierarchy, each PCA being certified by the IPRA. A PCA must establish and publish a statement of its policy with respect to certifying users or subordinate certification authorities. Distinct PCAs aim to satisfy different user needs. For example, one PCA (an organizational PCA) might support the general electronic mail needs of commercial organizations, and another PCA (a high-assurance PCA) might have a more stringent policy designed for satisfying legally binding signature requirements.

(c) Certification Authorities (CAs): CAs are at level 3 of the hierarchy and can also be at lower levels. Those at level 3 are certified by PCAs. CAs represent, for example, particular organizations, particular organizational units (e.g., departments, groups, sections), or particular geographical areas.
RFC 1422 furthermore has a name subordination rule which requires that a CA can only issue certificates for entities whose names are subordinate (in the X.500 naming tree) to the name of the CA itself. The trust associated with a PEM certification path is implied by the PCA name. The name subordination rule ensures that CAs below the PCA are sensibly constrained as to the set of subordinate entities they can certify (e.g., a CA for an organization can only certify entities in that organization’s name tree). Certificate user systems are able to mechanically check that the name subordination rule has been followed.

The RFC 1422 CA hierarchical model has been found to have several deficiencies, including:

(a) The pure top-down hierarchy, with all certification paths starting from the root, is too restrictive for many purposes. For some applications, verification of certification paths should start with a public key of a CA in a user’s own domain, rather than mandating that verification commence at the top of a hierarchy. In many environments, the local domain is often the most trusted. Also, initialization and key-pair-update operations can be more effectively conducted between an end entity and a local management system.

(b) The name subordination rule introduces undesirable constraints upon the X.500 naming system an organization may use.

(c) Use of the PCA concept requires knowledge of individual PCAs to be built into certificate chain verification logic. In the particular case of Internet mail, this is not a major problem -- the PCA name can always be displayed to the human user who can make a decision as to what trust to imply from a particular chain. However, in many commercial applications, such as electronic commerce or EDI, operator intervention to make policy decisions is impractical. The process needs to be automated to a much higher degree. In fact, the full process of certificate chain processing needs to be implementable in trusted software.

Because of the above shortcomings, it is proposed that more flexible CA structures than the RFC 1422 hierarchy be supported by the PKIX specifications. In fact, the main reason for the structural restrictions imposed by RFC 1422 was the restricted certificate format provided with X.509 v1. With X.509 v3, most of the requirements addressed by RFC 1422 can be addressed using certificate extensions, without a need to restrict the CA structures used. In particular, the certificate extensions relating to certificate policies obviate the need for PCAs and the constraint extensions obviate the need for the name subordination rule.
3.3 Revocation

When a certificate is issued, it is expected to be in use for its entire validity period. However, various circumstances may cause a certificate to become invalid prior to the expiration of the validity period. Such circumstances might include change of name, change of association between subject and CA (e.g., an employee terminates employment with an organization), and compromise or suspected compromise of the corresponding private key. Under such circumstances, the CA needs to revoke the certificate.

X.509 defines one method of certificate revocation. This method involves each CA periodically issuing a signed data structure called a certificate revocation list (CRL). A CRL is a time stamped list identifying revoked certificates which is signed by a CA and made freely available in a public repository. Each revoked certificate is identified in a CRL by its certificate serial number. When a certificate-using system uses a certificate (e.g., for verifying a remote user’s digital signature), that system not only checks the certificate signature and validity but also acquires a suitably-recent CRL and checks that the certificate serial number is not on that CRL. The meaning of "suitably-recent" may vary with local policy, but it usually means the most recently-issued CRL. A CA issues a new CRL on a regular periodic basis (e.g., hourly, daily, or weekly). Entries are added to CRLs as revocations occur, and an entry may be removed when the certificate expiration date is reached.

An advantage of this revocation method is that CRLs may be distributed by exactly the same means as certificates themselves, namely, via untrusted communications and server systems.

One limitation of the CRL revocation method, using untrusted communications and servers, is that the time granularity of revocation is limited to the CRL issue period. For example, if a revocation is reported now, that revocation will not be reliably notified to certificate-using systems until the next periodic CRL is issued -- this may be up to one hour, one day, or one week depending on the frequency that the CA issues CRLs.

Another potential problem with CRLs is the risk of a CRL growing to an entirely unacceptable size. In the 1988 and 1993 versions of X.509, the CRL for the end-user certificates needed to cover the entire population of end-users for one CA. It is desirable to allow such populations to be in the range of thousands, tens of thousands, or possibly even hundreds of thousands of users. The end-user CRL is therefore at risk of growing to such sizes, which present major communication and storage overhead problems. With the version 2 CRL format, introduced along with the v3 certificate format, it becomes
possible to arbitrarily divide the population of certificates for one CA into a number of partitions, each partition being associated with one CRL distribution point (e.g., directory entry or URL) from which CRLs are distributed. Therefore, the maximum CRL size can be controlled by a CA. Separate CRL distribution points can also exist for different revocation reasons. For example, routine revocations (e.g., name change) may be placed on a different CRL to revocations resulting from suspected key compromises, and policy may specify that the latter CRL be updated and issued more frequently than the former.

As with the X.509 v3 certificate format, in order to facilitate interoperable implementations from multiple vendors, the X.509 v2 CRL format needs to be profiled for Internet use. It is one goal of this document to specify such profiles.

Furthermore, it is recognized that on-line methods of revocation notification may be applicable in some environments as an alternative to the X.509 CRL. On-line revocation checking eliminates the latency between a revocation report and CRL the next issue. Once the revocation is reported, any query to the on-line service will correctly reflect the certificate validation impacts of the revocation. Therefore, this document will also consider standard approaches to on-line revocation notification.

### 3.4 Operational Protocols

Operational protocols are required to deliver certificates and CRLs to certificate using client systems. Provision is needed for a variety of different means of certificate and CRL delivery, including request/delivery procedures based on E-mail, http, X.500, and WHOIS++. These specifications include definitions of, and/or references to, message formats and procedures for supporting all of the above operational environments, including definitions of or references to appropriate MIME content types.

### 3.5 Management Protocols

Management protocols are required to support on-line interactions between Public Key Infrastructure (PKI) components. For example, management protocol might be used between a CA and a client system with which a key pair is associated, or between two CAs which cross-certify each other. The set of functions which potentially need to be supported by management protocols include:

(a) **registration**: This is the process whereby a user first makes itself known to a CA, prior to that CA issuing a certificate or certificates for that user.
(b) initialization: Before a client system can operate securely it is necessary to install in it necessary key materials which have the appropriate relationship with keys stored elsewhere in the infrastructure. For example, the client needs to be securely initialized with the public key of a CA, to be used in validating certificate paths. Furthermore, a client typically needs to be initialized with its own key pair(s).

(c) certification: This is the process in which a CA issues a certificate for a user’s public key, and returns that certificate to the user’s client system and/or posts that certificate in a public repository.

(d) key pair recovery: As an option, user client key materials (e.g., a user’s private key used for encryption purposes) may be backed up by a CA or a key backup system associated with a CA. If a user needs to recover these backed up key materials (e.g., as a result of a forgotten password or a lost key chain file), an on-line protocol exchange may be needed to support such recovery.

(e) key pair update: All key pairs need to be updated regularly, i.e., replaced with a new key pair, and new certificates issued.

(f) revocation request: An authorized person advises a CA of an abnormal situation requiring certificate revocation.

(g) cross-certification: Two CAs exchange the information necessary to establish cross-certificates between those CAs.

Note that on-line protocols are not the only way of implementing the above functions. For all functions there are off-line methods of achieving the same result, and this specification does not mandate use of on-line protocols. For example, when hardware tokens are used, many of the functions may be achieved through as part of the physical token delivery. Furthermore, some of the above functions may be combined into one protocol exchange. In particular, two or more of the registration, initialization, and certification functions can be combined into one protocol exchange.

Part 3 of the PKIX series of specifications defines a set of standard message formats supporting the above functions. The protocols for conveying these messages in different environments (on-line, e-mail, and WWW) are also specified.

4 Certificate and Certificate Extensions Profile

The goal of this document is to create a profile for X.509 v3 certificates that will foster interoperability and a reusable public
key infrastructure. Since the publication of earlier versions of this draft, substantial changes have been made to the Amendment \[X.509-AM\] to X.509 defining version 3 and certificate extensions. Those changes have brought the base document into close alignment with the recommendations of earlier versions of this draft. As a result, this document provides a concise profile rather than attempting to recreate the Amendment as a standalone document.

Certificates may be used in a wide range of applications and environments covering a broad spectrum of interoperability goals and a broader spectrum of operational and assurance requirements. The goal of this document is to establish a common baseline for generic applications requiring broad interoperability and limited special purpose requirements. In particular, the emphasis will be on supporting the use of X.509 v3 certificates for informal internet electronic mail, IPSEC, and WWW applications. Other efforts are looking at certificate profiles for payment systems.

4.1 Basic Certificate Fields

The X.509 v3 certificate basic syntax is as follows. For signature calculation, the certificate is encoded using the ASN.1 distinguished encoding rules (DER) \[X.208\]. ASN.1 DER encoding is a tag, length, value encoding system for each element.

Certificate ::= SEQUENCE {
    tbsCertificate TBSCertificate,
    signatureAlgorithm AlgorithmIdentifier,
    signature BIT STRING }

TBSCertificate ::= SEQUENCE {
    version [0] Version DEFAULT v1,
    serialNumber CertificateSerialNumber,
    signature AlgorithmIdentifier,
    issuer Name,
    validity Validity,
    subject Name,
    subjectPublicKeyInfo SubjectPublicKeyInfo,
    issuerUniqueID [1] IMPLICIT UniqueIdentifier OPTIONAL,
        -- If present, version must be v2 or v3
    subjectUniqueID [2] IMPLICIT UniqueIdentifier OPTIONAL,
        -- If present, version must be v2 or v3
    extensions [3] Extensions OPTIONAL
        -- If present, version must be v3
}

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

Validity ::= SEQUENCE {
    notBefore UTCTime,
    notAfter  UTCTime }

UniqueIdentifier ::= BIT STRING

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

Extensions ::= SEQUENCE OF Extension

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

The following items describe a proposed use of the X.509 v3 certificate for the Internet.

4.1.1 Version

This field describes the version of the encoded certificate. When extensions are used, as expected in this profile, use X.509 version 3 (value is 2). If no extensions are present, but a UniqueIdentifier is present, use version 2 (value is 1). If only basic fields are present, use version 1 (the value is omitted from the certificate as the default value).

Implementations should be prepared to accept any version certificate. In particular, at a minimum, implementations should recognize version 3 certificates; determine whether any critical extensions are present; and accept certificates without critical extensions even if they don’t recognize any extensions. A certificate with an unrecognized critical extension must always be rejected.

Generation of version 2 certificates is not expected by implementations based on this profile.

4.1.2 Serial number

The serial number is an integer assigned by the certification authority to each certificate. It must be unique for each certificate issued by a given CA (i.e., the issuer name and serial number identify a unique certificate).
4.1.3  Signature

This field contains the algorithm identifier for the algorithm used by the CA to sign the certificate. Section 7.2 lists the supported signature algorithms.

4.1.4  Issuer Name

The issuer name identifies the entity who has signed (and issued the certificate). The issuer identity may be carried in the issuer name field and/or the issuerAltName extension. If identity information is present only in the issuerAltName extension, then the issuer name may be an empty sequence and the issuerAltName extension must be critical.

<< Should we say anything about name constraints here? >>

4.1.5  Validity

This field indicates the dates on which the certificate becomes valid (notBefore) and on which the certificate ceases to be valid (notAfter). It is strongly recommended that UTCTime values be expressed Greenwich Mean Time (Zulu) and not use seconds (i.e., times are YYMMDDHMMZ). If seconds are used, a value of 00 seconds should never be encoded.

4.1.6  Subject Name

The subject name identifies the entity associated with the public key stored in the subject public key field. The subject identity may be carried in the subject field and/or the subjectAltName extension. If identity information is present only in the subjectAltName extension (e.g., a key bound only to an email address or URI), then the subject name may be an empty sequence and the subjectAltName extension must be critical.

<< Should we say anything about name constraints here? >>

4.1.7  Subject Public Key Info

This field is used to carry the public key and identify the algorithm with which the key is used.

4.1.8  Unique Identifiers

The subject and issuer unique identifier are present in the certificate to handle the possibility of reuse of subject and/or issuer names over time. This profile recommends that names not be
reused and that Internet certificates not make use of unique identifiers. CA’s conforming to this profile should not generate certificates with unique identifiers. Applications conforming to this profile should be capable of parsing unique identifiers and making comparisons.

4.2 Certificate Extensions

The extensions defined for X.509 v3 certificates provide methods for associating additional attributes with users or public keys, for managing the certification hierarchy, and for managing CRL distribution. The X.509 v3 certificate format also allows communities to define private extensions to carry information unique to those communities. Each extension in a certificate may be designated as critical or non-critical. A certificate using system (an application validating a certificate) must reject the certificate if it encounters a critical extension it does not recognize. A non-critical extension may be ignored if it is not recognized. The following presents recommended extensions used within Internet certificates and standard locations for information. Communities may elect to use additional extensions; however, caution should be exercised in adopting any critical extensions in certificates which might be used in a general context.

The extensions referenced below are defined in detail in the X.509 Amendment along with the specification of syntax and object identifiers. Because the intent is to align exactly with those definitions, the material is not reproduced here. The following sections describe Internet profiling decisions and Internet private extensions.

<< Once finalized, the certificate ASN.1 definition will be included in an appendix to this document. >>

4.2.1 Standard Extensions

4.2.1.1 Authority Key Identifier

The authority key identifier extension provides a means of identifying the particular public key used to sign a certificate. The identification can be based on either the key identifier (the subject key identifier in the issuer’s certificate) or on the issuer name and serial number. The key identifier method is recommended in this profile. This extension would be used where an issuer has multiple signing keys (either due to multiple concurrent key pairs or due to changeover). In general, this extension should be included in certificates.
4.2.1.2 Subject Key Identifier

The subject key identifier extension provides a means of identifying the particular public key used in an application. Where a reference to a public key identifier is needed (as with an Authority Key Identifier) and one is not included in the associated certificate, a SHA-1 hash of the subject public key shall be used. The hash shall be calculated over the value (excluding tag and length) of the subject public key field in the certificate. This extension should be marked non-critical.

4.2.1.3 Key Usage

The key usage extension defines the purpose (e.g., encipherment, signature, certificate signing) of the key contained in the certificate. The usage restriction might be employed when a multipurpose key is to be restricted (e.g., when an RSA key should be used only for signing or only for key encipherment). The profile recommends that when used, this be marked as a critical extension.

4.2.1.4 Private Key Usage Period

This profile recommends against the use of this extension. CA’s conforming to this profile should not generate certificates with private key usage period extensions.

4.2.1.5 Certificate Policies

The certificate policies extension contains a sequence of policy information terms, each of which consists of an object identifier (OID) and optional qualifiers. These policy information terms indicate the policy under which the certificate has been issued and the purposes for which the certificate may be used. This profile strongly recommends that a simple OID be present in this field. Optional qualifiers which may be present are expected to provide information about obtaining CA rules, not change the definition of the policy. Applications are expected to have a list of those policies which they will accept and to compare the policy OIDs in the certificate to that list.

<< Do we want to say anything about criticality? >>

4.2.1.6 Policy Mappings

This extension may be supported by CAs and/or applications, and it is always non-critical.
4.2.1.7 Subject Alternative Name

The subject alternative names extension allows additional identities to be bound to the subject of the certificate. Defined options include an rfc822 name (electronic mail address), a DNS name, an IP address, and a URI. Other options exist, including completely local definitions. Multiple instances of a name and multiple name forms may be included. Whenever such identities are to be bound into a certificate, the subject alternative name (or issuer alternative name) extension shall be used. (Note: a form of such an identifier may also be present in the subject distinguished name; however, the alternative name extension is the preferred location for finding such information.) Further, if the only subject identity included in the certificate is an alternative name form (e.g., an electronic mail address), then the subject distinguished name should be empty (an empty sequence), the subjectAltName extension should be used, and the subjectAltName extension must be marked critical.

<< In the previous version we said: The use of otherName should not be used in conjunction with this profile. Should we put this restriction back? >>

Alternative names may be constrained in the same manner as subject distinguished names using the name constraints extension as described in section 4.2.1.11.

4.2.1.8 Issuer Alternative Name

As with 4.2.1.7, this extension is used to associate Internet style identities with the certificate issuer. If the only issuer identity included in the certificate is an alternative name form (e.g., an electronic mail address), then the issuer distinguished name should be empty (an empty sequence), the issuerAltName extension should be used, and the issuerAltName extension must be marked critical.

4.2.1.9 Subject Directory Attributes

The subject directory attributes extension is not recommended as an essential part of this profile, but it may be used in local environments. This extension is always non-critical.

4.2.1.10 Basic Constraints

The basic constraints extension identifies whether the subject of the certificate is a CA and how deep a certification path may exist through that CA. This profile requires the use of this extension, and it shall critical for all certificates issued to CAs.
4.2.1.11 Name Constraints

The name constraints extension provides permitted and excluded subtrees that place restrictions on names that may be included within a certificate issued by a given CA. Restrictions may apply to the subject distinguished name or subject alternative names. Any name matching a restriction in the excluded subtrees field is invalid regardless of information appearing in the permitted subtrees. Restrictions for the rfc822, dNSName, and uri name forms are all expressed in terms of strings with wild card matching. An "*" is the wildcard character. The minimum and maximum fields in general subtree are not used for these name forms. For uris and rfc822 names, the restriction applies to the host part of the name. Examples would be foo.bar.com; www*.bar.com; *.xyz.com.

<< X.208 (1988) defines is IA5String as PrintableString plus NumericString plus SPACE and DELETE. Thus, it does not allow the for the * character. What should we use instead? >>

4.2.1.12 Policy Constraints

The policy constraints extension may be used by CAs.

4.2.1.13 CRL Distribution Points

The CRL distribution points extension identifies how CRL information is obtained. The profile recommends support for this extension by CAs and applications. Further discussion of CRL management is contained in section 5.

4.2.2 Private Internet Extensions

4.2.2.1 Subject Information Access

The name information in the certificate identifies the entity to which the public key is bound. In some instances, it may also be necessary to know where to find additional information about the named entity. In the case of X.500 names, this relationship is automatic. The subject information access extension provides a means of identifying where and how to find information about the subject. The extension specifies a method of obtaining information and a general name form indicating where. This extension should always be non-critical.

subjectInfoAccess EXTENSION ::= {
  SYNTAX SubjectInfoAccessSyntax
  IDENTIFIED BY { TBD-OID-1 } }

Housley, Ford, & Solo                                          [Page 17]
SubjectInfoAccessSyntax ::= SEQUENCE SIZE (1..MAX) OF AccessDescription

AccessDescription ::= SEQUENCE {
  accessMethod OBJECT IDENTIFIER,
  accessLocation GeneralName
}

<< What upper bound should be assigned for MAX? >>
<< Where is the list of access method OIDs going to be specified? >>

4.2.2.2 Authority Information Access

The authority information access extension indicates how to access CA information and services for the issuer of the certificate in which the extension appears. Information and services include certificate status or on-line validation services, certificate retrieval, CA policy data, and CA certificates (certificates certifying the target CA to aid in certification path navigation). This extension may be included in subject or CA certificates and may be critical or non-critical.

<< What does it mean to mark this extension critical? >>

authorityInfoAccess EXTENSION ::= {
  SYNTAX AuthorityInfoAccessSyntax
  IDENTIFIED BY { TBD-OID-2 } }

AuthorityInfoAccessSyntax ::= SEQUENCE {
  certStatus [0] SEQUENCE OF AccessDescription,
  certRetrieval [1] SEQUENCE OF AccessDescription,
  caPolicy [2] SEQUENCE OF AccessDescription,
  caCerts [3] SEQUENCE OF AccessDescription
}

4.2.2.3 CA Information Access

The CA information access extension indicates how to access CA information and services for the subject of the certificate in which the extension appears. Information and services include certificate status or on-line validation services, certificate retrieval, CA policy data, and CA certificates (certificates certifying the target CA to aid in cert path navigation). This extension may be included only in CA certificates and may be critical or non-critical. CA certificates may include both an authority and a caInfoAccess extension to describe access methods for both the CA and its issuer.

caInfoAccess EXTENSION ::= {
  SYNTAX CAInfoAccessSyntax
  IDENTIFIED BY { TBD-OID-2 } }
CAInfoAccessSyntax ::= SEQUENCE {
    certStatus        [0] SEQUENCE OF AccessDescription,
    certRetrieval     [1] SEQUENCE OF AccessDescription,
    caPolicy          [2] SEQUENCE OF AccessDescription,
    caCerts           [3] SEQUENCE OF AccessDescription  }

4.3 Examples

<< Certificate samples including descriptive text and ASN.1 encoded blobs will be inserted. >>

5 CRL and CRL Extensions Profile

As described above, one goal of this X.509 v2 CRL profile is to foster the creation of an interoperable and reusable Internet PKI. To achieve this goal, guidelines for the use of extensions are specified, and some assumptions are made about the nature of information included in the CRL.

CRLs may be used in a wide range of applications and environments covering a broad spectrum of interoperability goals and an even broader spectrum of operational and assurance requirements. This profile establishes a common baseline for generic applications requiring broad interoperability. Emphasis is placed on support for X.509 v2 CRLs. The profile defines a baseline set of information that can be expected in every CRL. Also, the profile defines common locations within the CRL for frequently used attributes, and common representations for these attributes.

This profile does not define any private Internet CRL extensions or CRL entry extensions.

Environments with additional or special purpose requirements may build on this profile or may replace it.

5.1 CRL Fields

The X.509 v2 CRL syntax is as follows. For signature calculation, the data that is to be signed is ASN.1 DER encoded. ASN.1 DER encoding is a tag, length, value encoding system for each element.

CertificateList ::= SEQUENCE {
    tbsCertList         TBSCertList,
    signatureAlgorithm AlgorithmIdentifier,
    signature           BIT STRING  }

TBSCertList ::= SEQUENCE {
    version            Version OPTIONAL,
The following items describe the proposed use of the X.509 v2 CRL for in Internet PKI.

5.1.1 Version

This field describes the version of the encoded CRL. When extensions are used, as expected in this profile, use version 2 (the integer value is 1). If neither CRL extensions nor CRL entry extensions are present, use version 1 (the integer value must be omitted).

5.1.2 Signature

This field contains the algorithm identifier for the algorithm used to sign the CRL. Section 7.2 lists the signature algorithms used in the Internet PKI.
5.1.3 Issuer Name

The issuer name identifies the entity who has signed (and issued the CRL). The issuer identity may be carried in the issuer name field and/or the issuerAltName extension. If identity information is present only in the issuerAltName extension, then the issuer name may be an empty sequence and the issuerAltName extension must be critical.

5.1.4 This Update

This field indicates the issue date of this CRL. It is strongly recommended that UTCTime values be expressed Greenwich Mean Time (Zulu) and not use seconds (i.e., times are YYMMDDHHMMZ). If seconds are used, a value of 00 seconds should never be encoded.

5.1.5 Next Update

This field indicates the date by which the next CRL will be issued. The next CRL could be issued before the indicated date, but it will not be issued any later than the indicated date.

As described in the previous section, the UTCTime for Next Update should be expressed Greenwich Mean Time (Zulu) and not use seconds.

5.1.6 Revoked Certificates

Revoked certificates are listed. The revoked certificates are named by their serial numbers. Certificates are uniquely identified by the combination of the issuer name or issuer alternative name along with the user certificate serial number. The date on which the revocation occurred is specified. As described in section 5.1.4, the UTCTime for revocationDate should be expressed Greenwich Mean Time (Zulu) and not use seconds. CRL entry extensions are discussed in section 5.3.

When a CA wishes to revoke a certificate that it issued to another CA, the revocation shall appear on the CRL. The revocation should also appear on the authority revocation list (ARL). The CA is revoking a certificate that it issued.

5.2 CRL Extensions

The extensions defined by ANSI X9 and ISO for X.509 v2 CRLs [X.509-AM] [X9.55] provide methods for associating additional attributes with CRLs. The X.509 v2 CRL format also allows communities to define private extensions to carry information unique to those communities. Each extension in a CRL may be designated as critical or non-critical. A CRL validation must fail if it encounters an critical
extension which it does not know how to process. However, an unrecognized non-critical extension may be ignored. The following presents those extensions used within Internet CRLs. Communities may elect to use additional extensions; however, caution should be exercised in adopting any critical extensions in CRLs which might be used in a general context.

5.2.1 Authority Key Identifier

The authority key identifier extension provides a means of identifying the particular public key used to sign a CRL. The identification can be based on either the key identifier (the subject key identifier in the CRL signer’s certificate) or on the issuer name and serial number. The key identifier method is recommended in this profile. This extension would be used where an issuer has multiple signing keys, either due to multiple concurrent key pairs or due to changeover. In general, this non-critical extension should be included in certificates.

5.2.2 Issuer Alternative Name

The issuer alternative names extension allows additional identities to be associated with the issuer of the CRL. Defined options include an rfc822 name (electronic mail address), a DNS name, an IP address, and a URI. Multiple instances of a name and multiple name forms may be included. Whenever such identities are used, the issuer alternative name extension shall be used. Further, if the only issuer identity included in the CRL is an alternative name form (e.g., an electronic mail address), then the issuer distinguished name should be empty (an empty sequence), the issuerAltName extension should be used, and the issuerAltName extension must be marked critical.

5.2.3 CRL Number

The CRL number is a non-critical CRL extension which conveys a monotonically increasing sequence number for each CRL issued by a given CA through a specific CA X.500 Directory entry or CRL distribution point. This extension allows users to easily determine when a particular CRL supercedes another CRL. CAs conforming to this profile shall include this CRL.

5.2.4 Issuing Distribution Point

The issuing distribution point is a critical CRL extension that identifies the CRL distribution point for a particular CRL, and it indicates whether the CRL covers revocation for end entity certificates only, CA certificates only, or a limited set of reason codes. Since this extension is critical, all certificate users must
be prepared to receive CRLs with this extension.

The CRL is signed using the CA’s private key. CRL Distribution Points do not have their own key pairs. If the CRL is stored in the X.500 Directory, it is stored in the Directory entry corresponding to the CRL distribution point, which may be different that the Directory entry of the CA.

CRL distribution points, if used by a CA, should be partition the CRL on the basis of compromise and routine revocation. That is, the revocations with reason code keyCompromise (1) shall appear in one distribution point, and the revocations with other reason codes shall appear in another distribution point.

5.2.5 Delta CRL Indicator

The delta CRL indicator is a critical CRL extension that identifies a delta-CRL. The use of delta-CRLs can significantly improve processing time for applications which store revocation information in a format other than the CRL structure. This allows changes to be added to the local database while ignoring unchanged information that is already in the local database.

When a delta-CRL is issued, the CAs shall also issue a complete CRL.

The value of BaseCRLNumber identifies the CRL number of the base CRL that was used as the starting point in the generation of this delta-CRL. The delta-CRL contains the changes between the base CRL and the current CRL issued along with the delta-CRL. It is the decision of a CA as to whether to provide delta-CRLs. Again, a delta-CRL shall not be issued without a corresponding CRL. The value of CRLNumber for both the delta-CRL and the corresponding CRL shall be identical.

A CRL user constructing a locally held CRL from delta-CRLs shall consider the constructed CRL incomplete and unusable if the CRLNumber of the received delta-CRL is more that one greater that the CRLNumber of the delta-CRL last processed.

5.3 CRL Entry Extensions

The CRL entry extensions already defined by ANSI X9 and ISO for X.509 v2 CRLs [X.509-AM] [X9.55] provide methods for associating additional attributes with CRL entries. The X.509 v2 CRL format also allows communities to define private CRL entry extensions to carry information unique to those communities. Each extension in a CRL entry may be designated as critical or non-critical. A CRL validation must fail if it encounters an critical CRL entry extension which it does not know how to process. However, an unrecognized non-critical CRL entry extension may be ignored. The following
presents recommended extensions used within Internet CRL entries and standard locations for information. Communities may elect to use additional CRL entry extensions; however, caution should be exercised in adopting any critical extensions in CRL entries which might be used in a general context.

5.3.1 Reason Code

The reasonCode is a non-critical CRL entry extension that identifies the reason for the certificate revocation. As are strongly encouraged to include reason codes in CRL entries; however, the reason code CRL entry extension should be absent instead of using the unspecified (0) reasonCode value.

5.3.2 Hold Instruction Code

The hold instruction code is a non-critical CRL entry extension that provides a registered instruction identifier which indicates the action to be taken after encountering a certificate that has been placed on hold.

<< The Directory Specification does not define any standard hold instruction codes. Where will they be defined? Are any specific to the Internet environment? >>

5.3.3 Invalidity Date

The invalidity date is a non-critical CRL entry extension that provides the date on which it is known or suspected that the private key was compromised or that the certificate otherwise became invalid. This date may be earlier than the revocation date in the CRL entry, but it must be later than the issue date of the previously issued CRL. Remember that the revocation date in the CRL entry specifies the date that the CA revoked the certificate. Whenever this information is available, CAs are strongly encouraged to share it with CRL users.

The GeneralizedTime values included in this field shall be expressed in Greenwich Mean Time (Zulu) and omit trailing zeros in fractional seconds. Normally, GeneralizedTime will be expressed as YYYYMMDDHHMMSSZ.

5.4 Examples

<< CRL samples including descriptive text and ASN.1 encoded blobs will be inserted. >>
6 Certificate Path Validation

<< There used to be long section here that contained the certification path validation procedures. The major deviations from the procedures outlined in [X.509-AM] were due to certificate hold processing. These procedures are no longer onerous, so there is no reason to prohibit the use of the hold facility. Does anyone see a reason to retain this section? >>

7 Algorithm Support

7.1 One-way Hash Functions

One-way hash functions are also called message digest algorithms. SHA-1 is the most popular one-way hash function used in the Internet PKI. However, PEM uses MD2 for certificates [RFC 1422] [RFC 1423]. For this reason, MD2 is included in this profile.

7.1.1 MD2 One-way Hash Function

MD2 was also developed by Ron Rivest, but RSA Data Security has not placed the MD2 algorithm in the public domain. Rather, RSA Data Security has granted license to use MD2 for non-commercial Internet Privacy-Enhanced Mail. For this reason, MD2 may continue to be used with PEM certificates, but SHA-1 is preferred. MD2 is fully described in RFC 1319 [RFC 1319].

At the Selected Areas in Cryptography ’95 conference in May 1995, Rogier and Chauvaud presented an attack on MD2 that can nearly find collisions [RC95]. Collisions occur when two different messages generate the same message digest. A checksum operation in MD2 is the only remaining obstacle to the success of the attack. For this reason, the use of MD2 for new applications is discouraged. It is still reasonable to use MD2 to verify existing signatures, as the ability to find collisions in MD2 does not enable an attacker to find new messages having a previously computed hash value.

<< More information on the attack and its implications can be obtained from a RSA Laboratories security bulletin. These bulletins are available from <http://www.rsa.com/>. >>

7.1.2 SHA-1 One-way Hash Function

SHA-1 was developed by the U.S. Government. SHA-1 is fully described in FIPS 180-1 [FIPS 180-1].

SHA-1 is the one-way hash function of choice for use with both RSA the DSA signature algorithms.
7.2 Signature Algorithms

RSA and DSA are the most popular signature algorithms used in the Internet.

There is some ambiguity in 1988 X.509 document with respect to the definition of the SIGNED macro and the representation of a signature in a certificate or a CRL. The interpretation selected for the Internet requires that the data to be signed (e.g., the one-way hash function output value) is first ASN.1 encoded as an OCTET STRING and the result is encrypted (e.g., using RSA Encryption) to form the signed quantity. As part of the SIGNED macro, this signature value is then ASN.1 encoded as a BIT STRING.

7.2.1 RSA Signature Algorithm

A patent statement regarding the RSA algorithm can be found at the end of this profile.

The RSA algorithm is named for it’s inventors: Rivest, Shamir, and Adleman. The RSA signature algorithm is defined in PKCS #1 [PKCS#1]. It combines either the MD2 or the SHA-1 one-way hash function with the RSA asymmetric encryption algorithm. As defined in PKCS #1, the ASN.1 object identifiers used to identify these signature algorithms are:

```
md2WithRSAEncryption OBJECT IDENTIFIER ::= { 
    iso(1) member-body(2) US(840) rsadsi(113549) pkcs(1) 
    pkcs-1(1) 2  }

sha-1WithRSAEncryption OBJECT IDENTIFIER ::= { 
    iso(1) identified-organization(3) oiw(14) secsig(3) 
    algorithm(2) 29  }
```

When either of these object identifiers is used within the ASN.1 type AlgorithmIdentifier, the parameters component of that type shall the ASN.1 type NULL.

7.2.2 DSA Signature Algorithm

A patent statement regarding the DSA can be found at the end of this profile.

The Digital Signature Algorithm (DSA) is also called the Digital Signature Standard (DSS). DSA was developed by the U.S. Government, and DSA is used in conjunction with the the SHA-1 one-way hash function. DSA is fully described in FIPS 186 [FIPS 186]. The ASN.1 object identifiers used to identify this signature algorithm is:
The DSA algorithm syntax includes optional parameters. These parameters are commonly referred to as p, q, and g. The AlgorithmIdentifier within subjectPublicKeyInfo is the only place within a certificate where these parameters shall be present. If the DSA algorithm parameters are absent from the subjectPublicKeyInfo AlgorithmIdentifier and the CA signed the subject certificate using DSA, then the certificate issuer’s DSA parameters apply to the subject’s DSA key. If the DSA algorithm parameters are absent from the subjectPublicKeyInfo AlgorithmIdentifier and the CA signed the subject certificate using a signature algorithm other than DSA, then the subject’s DSA parameters are distributed by other means. The parameters are included using the following ASN.1 structure:

Dss-Parms ::= SEQUENCE {
    p             INTEGER,
    q             INTEGER,
    g             INTEGER  }

When signing, DSA algorithm generates two values. These values are commonly referred to as r and s. To easily transfer these two values as one signature, they are ASN.1 encoded using the following ASN.1 structure:

Dss-Sig-Value ::= SEQUENCE {
    r             INTEGER,
    s             INTEGER  }

7.3 Subject Public Key Algorithms

<< Add a section that lists the public key algorithms that are supported by this profile. Obviously, RSA, DSA, Diffie-Hellman, and KEA will be included. Are there others? >>

<< Should a different algorithm identifier be assigned to RSA signature keys and RSA key management keys? If so, there will be one subsection for each within this section. >>

References


Patent Statements

The Internet PKI relies on the use of patented public key technology. The Internet Standards Process as defined in RFC 1310 requires a written statement from the Patent holder that a license will be made available to applicants under reasonable terms and conditions prior to approving a specification as a Proposed, Draft or Internet Standard.

Patent statements for DSA, RSA, and Diffie-Hellman follow. These statements have been supplied by the patent holders, not the authors of this profile.

Digital Signature Algorithm (DSA)

The U.S. Government holds patent 5,231,668 on the Digital Signature Algorithm (DSA), which has been incorporated into Federal Information Processing Standard (FIPS) 186. The patent was issued on July 27, 1993.

The National Institute of Standards and Technology (NIST) has a long tradition of supplying U.S. Government-developed techniques
to committees and working groups for inclusion into standards on a royalty-free basis. NIST has made the DSA patent available royalty-free to users worldwide.

Regarding patent infringement, FIPS 186 summarizes our position; the Department of Commerce is not aware of any patents that would be infringed by the DSA. Questions regarding this matter may be directed to the Deputy Chief Counsel for NIST.

RSA Signature and Encryption

<< Now that PKP has dissolved, a revised patent statement for RSA from RSADSI is needed. >>

Diffie-Hellman Key Agreement

<< Now that PKP has dissolved, a revised patent statement for Diffie-Hellman from Cylink is needed. >>

Obsolete PKP Patent Statement

<< This statement is included here until a replacement from RSADSI and Cylink can be obtained. >>

The Massachusetts Institute of Technology and the Board of Trustees of the Leland Stanford Junior University have granted Public Key Partners (PKP) exclusive sub-licensing rights to the following patents issued in the United States, and all of their corresponding foreign patents:

Cryptographic Apparatus and Method
("Diffie-Hellman")......................... No. 4,200,770

Public Key Cryptographic Apparatus
and Method ("Hellman-Merkle")............. No. 4,218,582

Cryptographic Communications System and
Method ("RSA")............................. No. 4,405,829

Exponential Cryptographic Apparatus
and Method ("Hellman-Pohlig").............. No. 4,424,414

These patents are stated by PKP to cover all known methods of practicing the art of Public Key encryption, including the variations collectively known as El Gamal.

Public Key Partners has provided written assurance to the Internet Society that parties will be able to obtain, under reasonable,
nondiscriminatory terms, the right to use the technology covered by these patents. This assurance is documented in RFC 1170 titled "Public Key Standards and Licenses". A copy of the written assurance dated April 20, 1990, may be obtained from the Internet Assigned Number Authority (IANA).

The Internet Society, Internet Architecture Board, Internet Engineering Steering Group and the Corporation for National Research Initiatives take no position on the validity or scope of the patents and patent applications, nor on the appropriateness of the terms of the assurance. The Internet Society and other groups mentioned above have not made any determination as to any other intellectual property rights which may apply to the practice of this standard. Any further consideration of these matters is the user’s own responsibility.

Security Considerations

This entire memo is about security mechanisms.

Author Addresses:

Russell Housley
SPYRUS
PO Box 1198
Herndon, VA 22070
USA
housley@spyrus.com

Warwick Ford
Nortel Secure Networks
PO Box 3511, Station C
Ottawa, Ontario
Canada KY 4H7
wford@bnr.ca

David Solo
BBN
150 CambridgePark Drive
Cambridge, MA 02140
USA
solo@bbn.com