PKIX Working Group
INTERNET DRAFT

A. Arsenault
DOD
S. Turner
IECA

Expires in six months from September 8, 1999

Internet X.509 Public Key Infrastructure
PKIX Roadmap
<draft-ietf-pkix-roadmap-03.txt>

Status of this Memo

Abstract

This document provides an overview or 'roadmap' of the work done by the IETF PKIX working group. It describes some of the terminology used in the working group's documents, and the theory behind an X.509-based PKI. It identifies each document developed by the PKIX working group, and describes the relationships among the various documents. It also provides advice to would-be PKIX implementors about some of the issues discussed at length during PKIX development, in hopes of making it easier to build implementations that will actually interoperate.

1 Introduction

1.1 This Document

This document is an informational Internet draft that provides a "roadmap" to the documents produced by the PKIX working group. It is intended to provide information; there are no requirements or specifications in this document.

Section 2 of this document defines key terms used in this document. Section 3 covers "PKIX theory"; it explains what the PKIX working group's basic assumptions were. Section 4 provides an overview of the various PKIX documents. It identifies which documents address which areas, and describes the relationships among the various documents. Section 5 contains "Advice to implementors". Its primary purpose is to capture some of the major issues discussed by the PKIX working group, as a way of explaining WHY some of the requirements and specifications say what they say. This should cut down on the number of misinterpretations of the documents, and help developers build interoperable implementations. Section 6 contains a list of references. Section 7 discusses security considerations, and Section 8 provides contact information for the editors.

2 Terminology

There are a number of terms used and misused throughout PKI-related literature. To limit confusion caused by some of those terms, throughout this document, we will use the following terms in the following ways:

- Certification Authority (CA) - an authority trusted by one or more users to create and assign certificates. Optionally the certification authority may create the user's keys. It is important to note that the CA is responsible for the certificates during their whole lifetime, not just for issuing them.

- Certificate Policy (CP) - a named set of rules that indicates the applicability of a certificate to a particular community and/or class of application with common security requirements. For example, a particular certificate policy might indicate applicability of a type of certificate to the authentication

of electronic data interchange transactions for the trading of goods within a given price range.

- Certification Practice Statement (CPS) - A statement of the practices which a certification authority employs in issuing

certificates.

- Root CA - a CA that is directly trusted by an end entity; that is, securely acquiring the value of a root CA public key requires some out-of-band step(s). This term is not meant to imply that a root CA is necessarily at the top of any hierarchy, simply that the CA in question is trusted directly.

- Top CA - a CA that is at the top of a PKI hierarchy.

Note: This is often also called a "root CA," from since in data structures terms and in graph theory, the node at the top of a tree is the "root." However, to minimize confusion in this document, we elect to call this node a "Top CA," and reserve "Root CA" for the CA directly trusted by the user. Readers new to PKIX should be aware that these terms are not used consistently throughout the PKIX documents, as [RFC2459] uses "Root CA" to refer to what this and other documents call a "Top CA," and "most-trusted CA" to refer to what this and other documents call a "Root CA."

- Subordinate CA - A "subordinate CA" is one that is not a root CA for the end entity in question. Often, a subordinate CA will not be a Root CA for any entity but this is not mandatory.

- Registration Authority (RA) - an optional entity given responsibility for performing some of the administrative tasks necessary in the registration of subjects, such as: confirming the subject's identity; validating that the subject is entitled to have the attributes requested in a certificate; and verifying that the subject has possession of the private key associated with the public key requested for a certificate.

- End-entity - a subject of a certificate who is not a CA.

- Relying party - a user or agent (e.g., a client or server) who

relies on the data in a certificate in making decisions.

- Subject - a subject is the entity (CA or end-entity) named in a certificate. Subjects can be human users, computers (as represented by DNS names or IP addresses), or even software agents.

3 PKIX Theory

3.1 Certificate-using Systems and PKIs

At the heart of recent efforts to improve Internet security are a group of security protocols such as S/MIME, TLS, and IPSec. All of these protocols rely on public-key cryptography to provide services such as confidentiality, data integrity, data origin authentication, and non-repudiation. The purpose of a PKI is to provide trusted and efficient key and certificate management, thus enabling the use of authentication, non-repudiation, and confidentiality.

Users of public key-based systems must be confident that, any time they rely on a public key, the associated private key is owned by the subject with which they are communicating. (This applies whether an encryption or digital signature mechanism is used.) This confidence is obtained through the use of public key certificates, which are data structures that bind public key values to subjects. The binding is achieved by having a trusted CA verify the subject's identity and 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.

Certificates are used in the process of validating signed data. Specifics vary according to which algorithm is used, but the general process works as follows:

Note: there is no specific order in which the checks listed below must be made; implementers are free to implement them in the most

efficient way for their systems.

- The recipient of signed data verifies that the claimed identity of the user is in accordance with the identity contained in the certificate;

- The recipient validates that no certificate in the path is revoked (e.g., by retrieving a suitably-current Certificate Revocation List (CRL) or querying an on-line certificate status responder), and that all certificates are within their validity periods at the time the data was signed;

- The recipient verifies that the data are not claimed to have any attributes for which the certificate indicates that the signer is not authorized;

- The recipient verifies that the data have not been altered

since signing, by using the public key in the certificate.

If all of these checks pass, the recipient can accept that the data was signed by the purported signer. The process for keys used for encryption is similar.

Note: It is of course possible that the data was signed by someone very different from the signer, if for example the purported signer's private key was compromised. Security depends on all parts of the certificate-using system, including but not limited to: physical security of the place the computer resides; personnel security (i.e., the trustworthiness of the people who actually develop, install, run, and maintain the system); the security provided by the operating system on which the private key is used; and the security provided the CA. A failure in any one of these areas can cause the entire system security to fail. PKIX is limited in scope, however, and only directly addresses issues related to the operation of the PKI subsystem. For guidance in many of the other areas, see [POLPROC].

A collection of certificates, with their issuing CA's, subjects,

relying parties, RA's, and repositories, is referred to as a Public Key Infrastructure, or PKI.

3.2 PKIX History

[This still needs more work.]

In the beginning there was ITU-T Recommendation X.509. It defines a widely accepted basis for a public-key infrastructure, including data formats and procedures related to distribution of public keys via certificates digitally signed by CAs. X.509 does not however include a profile to specify the support requirements for many of the certificate data structure's sub-fields, for any of the extensions, nor for certain data values. PKIX was formed in October 1995 to deliver a profile for the Internet PKI of X.509 version 3 certificates and version 2 CRLs. The Internet PKI profile [FORMAT] went through eleven draft versions before becoming an RFC. Other profiles have been developed in PKIX for particular algorithms to make use of [FORMAT]. There has been no sense of conflict between the groups that developed these profiles as they are seen as complimentary.

The development of the management protocols has not been so straightforward. [CMP] was developed to define a message protocol that is used between entities in a PKI. The demand for an enrollment protocol and the desire to use PKCS-10 message format as the

certificate request syntax lead to the development of two different documents in two different groups. The Certificate Request Syntax [CRS] draft was developed in the SMIME WG which used PKCS10 [PKCS10] as the certification request message format. Certificate Request Message Format [CRMF] draft was also developed but in the PKIX WG. It was to define a simple enrollment protocol that would subsume both the [CMP] and [CRS] enrollment protocols, but it did not use PKCS10 as the certificate request message format. Then, [CMMF] was developed to define an extended set of management messages that flow between the components of the Internet PKI. CMMF over CMS [CMC] was developed to allow the use of an existing protocol (S/MIME) as a PKI management protocol, without requiring the development of an entirely new protocol such as CMP [CMP]. It also included [PKCS10] as the certificate request syntax, which caused work on [CRS] to stop. Information from [CMMF] has been moved into [CMP] and [CMC] so [CMMF]

is being discontinued.

Development of the operational protocols has been slightly more straightforward. Two documents for LDAPv2 have been developed one for defining LDAPv2 as an access protocol to repositories [PKI-LDAPv2]; and one for storing PKI information in an LDAP directory [SCHEMA]. Using FTP and HTTP to retrieve certificates and CRL from PKI repositories was documented without a fight in [FTPHTTP]. Likewise, methods, headers, and content-types ancillary to HTTP/1.1 to publish and retrieve X.509 certificates and CRLs was documented in [WEB] without much argument.

[Need to add text about OpenCDP vs DistributionPoints, Why DCP was started, information on TSP, and OCSP, and caching OCSP.]

3.3 Overview of the PKIX Approach

PKIX is an effort to develop specifications for a Public Key Infrastructure for the Internet using X.509 certificates. The PKIX working group was initially chartered in 1995. A Public Key Infrastructure, or PKI, is defined as:

The set of hardware, software, people, policies and procedures needed to create, manage, store, distribute, and revoke certificates based on public-key cryptography.

A PKI consists of five types of components [MISPC]:

- Certification Authorities (CAs) that issue and revoke certificates;
- Organizational Registration Authorities (ORAs) that vouch for

the binding between public keys and certificate holder identities and other attributes;

- Certificate holders that are issued certificates and can sign digital documents and encrypt documents;
- Clients that validate digital signatures and their certification paths from a known public key of a trusted CA;

- Repositories that store and make available certificates and Certificate Revocation Lists (CRLs).

Figure 1 is a simplified view of the architectural model assumed by the PKIX Working Group.

+---+

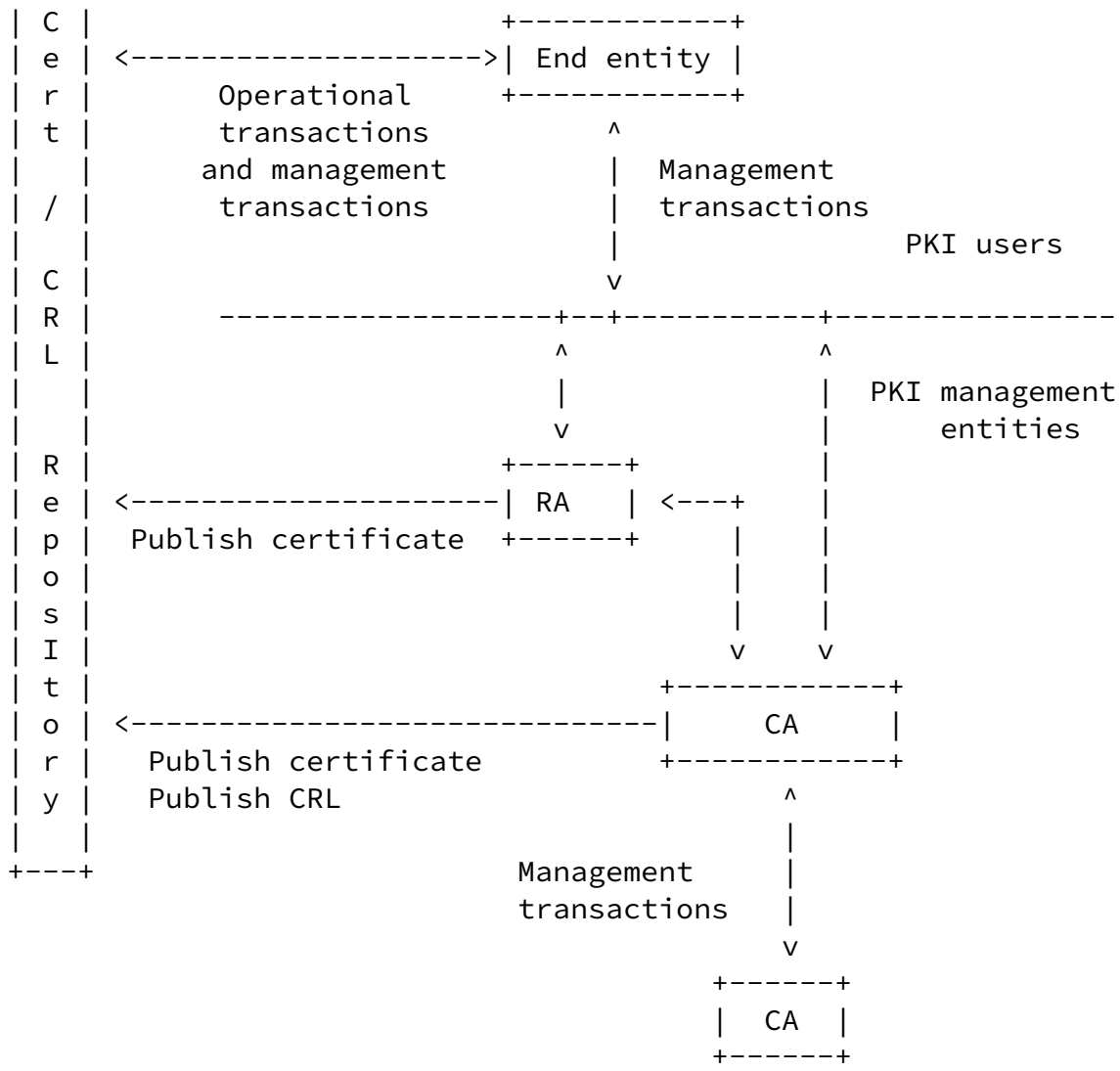

Figure 1 - PKI Entities

3.4 X.509 certificates

ITU-T X.509 (formerly CCITT X.509) or ISO/IEC/ITU 9594-8, which was first published in 1988 as part of the X.500 Directory recommendations, defines a standard certificate format [X.509]. 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 may be used to support directory access control.

The Internet Privacy Enhanced Mail (PEM) RFCs, published in 1993, include specifications for a public key infrastructure based on X.509v1 certificates [PEM]. The experience gained in attempts to deploy [PEM] 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/ITU 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].

ISO/IEC/ITU and ANSI X9 have also developed standard extensions for use in the v3 extensions field [X.509][X9.55]. These extensions can convey such data as additional subject identification information, key attribute information, policy information, and certification path constraints. However, the ISO/IEC/ITU and ANSI X9 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 PKIX to specify a profile for Internet, electronic mail, and IPsec applications. Environments with additional requirements may build on this profile or may replace it.

3.5 Functions of a PKI

This section describes the major functions of a PKI. In some cases, PKIs may provide extra functions.

3.5.1 Registration

This is the process whereby a subject first makes itself known to a CA (directly, or through an RA), prior to that CA issuing a

certificate or certificates for that subject. Registration involves the subject providing its name (e.g., common name, fully-qualified domain name, IP address), and other attributes to be put in the certificate, followed by the CA (possibly with help from the RA) verifying in accordance with its Certfication Practice Statement (CPS) that the name and other attributes are correct.

3.5.2 Initialization

Initialization is when the subject (e.g., the user or client system) gets the values needed to begin communicating with the PKI. For example, initialization can involve providing the client system with the public key and/or certificate of a CA, or generating the client system's own public/private key pair.

3.5.3 Certification

This is the process in which a CA issues a certificate for a subject's public key, and returns that certificate to the subject and/or posts that certificate in a repository.

3.5.4 Key Pair Recovery

In some implementations, key exchange or encryption keys will be required by local policy to be "backed up", or recoverable in case the key is lost and access to previously-encrypted information is needed. Such implementations can include those where the private key exchange key is stored on a hardware token which can be lost or broken, or when a private key file is protected by a password which can be forgotten. Often, a company is concerned about being able to read mail encrypted by or for a particular employee when that employee is no longer available because she is ill or no longer works for the company.

In these cases, the user's private key can be backed up by a CA or by a separate key backup system. If a user or her employer needs to recover these backed up key materials, the PKI must provide a system that permits the recovery without providing an unacceptable risk of compromise of the private key.

3.5.5 Key Generation

Depending on the CA's policy, the private/public key pair can either be generated by the user in his local environment, or generated by the CA. In the latter case, the key material may be distributed to the user in an encrypted file or on a physical token - e.g., a smart card or PCMCIA card.

3.5.6 Key Update

All key pairs need to be updated regularly (i.e., replaced with a new key pair) and new certificates issued. This will happen in two cases: normally, when a key has passed its maximum usable lifetime; and exceptionally, when a key has been compromised and must be replaced.

3.5.6.1 Key Expiry

In the normal case, a PKI needs to provide a facility to gracefully transition from a certificate with an existing key to a new certificate with a new key. This is particularly true when the key to be updated is that of a CA. Users will know in advance that the key will expire on a certain date; the PKI, working together with certificate-using applications, should allow for appropriate keys to work before and after the transition. There are a number of ways to do this; see [insert appropriate reference here] for an example of one.

3.5.6.2 Key Compromise

In the case of a key compromise, the transition will not be "graceful" in that there will be an unplanned switch of certificates and keys; users will not have known in advance what was about to happen. Still, the PKI must support the ability to declare that the previous certificate is now invalid and shall not be used, and to announce the validity and availability of the new certificate.

> Note: compromise of a private key associated with a Root CA is catastrophic for users relying on that Root CA. If a Root CA's private key is compromised, that CA's certificate must be revoked and all certificates subordinate to it must also be revoked. Until

such time as the Root CA has been issued a new certificate and the Root CA issues certificates to users relying upon it, users relying on that Root CA are cut off from the rest of the system, as there is no way to develop a valid certification path back to a trusted node.

Further, users will likely have to be notified by out-of-band mechanisms about the change in CA keys. If the old key is compromised, any "update" message telling subordinates to switch to a new key could have come from an attacker in possession of the old key, and could point to a new public key for which the attacker already has the private key. It is possible to have anticipated this event, and "pre-placed" replacement Root CA keys with all relying parties, but some secure, out-of-band mechanism will have to be used to tell users to make the switch, and this will only help if the replacement key has not been compromised.

Additionally, once the Root CA is brought back up with a new key, it will likely be necessary to re-issue certificates, signed with the new key, to all subordinate users, since their current certificate would be signed with a now-revoked key.

3.5.7 Cross-certification

A cross-certificate is a certificate issued by one CA to another CA which contains a public CA key associated with the private CA signature key used for issuing certificates. Typically, a cross-certificate is used to allow client systems/end entities in one administrative domain to communicate security with client systems/end users in another administrative domain. Use of a cross-certificate issued from CA_1 to CA_2 allows user Alice, who trusts CA_1, to accept a certificate used by Bob, which was issued by CA_2. Cross-certificates can also be issued from one CA to another CA in the same administrative domain, if required.

Cross-certificates can be issued in only one direction, or in both directions, between two CA's. That is, just because CA_1 issues a cross-certificate for CA_2, CA_2 does not have to issue a cross-certificate for CA_1.

3.5.8 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 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, 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). CA's may also issue CRLs aperiodically; e.g., if an important key is deemed compromised, the CA may issue a new CRL to

expedite notification of that fact, even if the next CRL does not have to be issued for some time. (A problem of aperiodic CRL issuance is that end-entities may not know that a new CRL has been issued, and thus may not retrieve it from a repository.)

An entry is added to the CRL as part of the next update following notification of revocation. An entry may be removed from the CRL after appearing on one regularly scheduled CRL issued beyond the revoked certificate's validity period. [Say why here]

An advantage of the CRL 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 CRL is issued - this may be up to one hour, one day, or one week depending on the frequency that the CA issues CRLs.

As with the X.509 v3 certificate format, in order to facilitate interoperable implementations from multiple vendors, the X.509 v2 CRL format needed to be profiled for Internet use. This was done as part of [FORMAT]. However, PKIX does not require CAs to issue CRLs. On-line methods of revocation notification may be applicable in some environments as an alternative to the X.509 CRL. PKIX defines a protocol known as OCSP [OCSP] to facilitate on-line checking of the status of certificates.

On-line revocation checking may significantly reduce the latency between a revocation report and the distribution of the information to relying parties. Once the CA accepts the report as authentic and valid, any query to the on-line service will correctly reflect the certificate validation impacts of the revocation. However, these methods impose new security requirements; the certificate validator must trust the on-line validation service while the repository does not need to be trusted.

3.5.9 Certificate and Revocation Notice Distribution/Publication

As alluded to in sections 3.1 and 3.5.8 above, the PKI is responsible for the distribution of certificates and certificate revocation notices (whether in CRL form or in some other form) in the system. "Distribution" of certificates includes transmission of the certificate to its owner, and may also include publication of the certificate in a repository. "Distribution" of revocation notices may

involve posting CRLs in a repository, transmitting them to end-entities, and/or forwarding them to on-line responders.

3.6 Parts of PKIX

This section identifies the five different areas in which the PKIX working group has developed documents. The first area involves profiles of the X.509 v3 certificate standards and the X.509v2 CRL standards for the Internet. The second area involves operational protocols, in which relying parties can obtain information such as certificates or certificate status. The third area covers management

protocols, in which different entities in the system exchange information needed for proper management of the PKI. The fourth area provides information about certificate policies and certificate practice statements, covering the areas of PKI security not directly addressed in the rest of PKIX. The fifth area deals with providing time stamping and data certification services, which can be used to build such services as non-repudiation.

3.6.1 Profile

An X.509v3 certificate is a very complex data structure. It consists of basic information fields, plus a number of optional certificate extensions. Many of the fields and numerous extensions can take on a wide range of options. This provides an enormous degree of flexibility, which allows the X.509v3 certificate format to be used with a wide range of applications in a wide range of environments. Unfortunately, this same flexibility makes it extremely difficult to produce independent implementations that will actually interoperate with one another. In order to build an Internet PKI based on X.509v3 certificates, the PKIX working group had to develop a profile of the X.509v3 specification.

A profile of the X.509v3 specification is a description of the contents of the certificate and which certificate extensions must be supported, which extensions may be supported, and which extensions may not be supported. [FORMAT] provides such a profile of X.509v3 for the Internet PKI. In addition, [FORMAT] suggests ranges of values for many of the extensions.

[FORMAT] also provides a profile for Version 2 CRLs for use in the Internet PKI. CRLs, like certificates, have a number of optional extensions. In order to promote interoperability, it is necessary to constrain the choices an implementor supports.

In addition to profiling the certificate and CRL formats, it is necessary to define particular Object Identifiers (OIDs) for certain encryption algorithms, because there are a variety of OIDs registered

for some algorithm suites. Many of the OIDs are defined in [FORMAT] to promote interoperability. Also, PKIX has produced two documents ([ECDSA] and [KEA]) which provide guidance on the proper implementation of specific algorithms.

Certain countries are in a process of updating their legal frameworks in order to regulate and incorporate recognition of signatures in electronic form. Many of these frameworks introduce certain basic requirements on certificates, often termed Qualified Certificates, supporting these types of "legal" signatures. Partly as a result of this there is a need for a specific certificate profile providing standardized support for certain related issues such as a common structure for expressing unambiguous identities of certified subjects (unmistakable identity). In December 1998, PKIX adopted as a work item the development of a refinement of [RFC2459] that further profiles PKIX certificates into qualified certificates. This work is reflected in [QC].

3.6.2 Operational Protocols

Operational protocols are required to deliver certificates and CRLs (or other certificate status information) to certificate using systems. Provision is needed for a variety of different means of certificate and CRL delivery, including distribution procedures based on LDAP, HTTP, FTP, and X.500. Operational protocols supporting these functions are defined in [FTPHTTP], [OCSP], and [PKI-LDAPv2].

3.6.3 Management Protocols

Management protocols are required to support on-line interactions between PKI user and management entities. For example, a 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. A management protocol can be used to carry user or client system registration information, or a request for revocation of a certificate.

There are two parts to a "management protocol". The first is the format of the messages that will be sent, and the second is the actual protocol that governs the transmission of those messages. Originally, the PKIX working group developed two documents, [CRMF] and [CMMF], that together described the necessary set of message formats, and two other documents, [CMP] and [CMC], that described protocols for exchanging those messages. However, the message formats defined in [CMMF] were inserted into both [CMP] and [CMC], and thus [CMMF] has been dropped as a PKIX document.

3.6.4 Policy Outline

As mentioned before, profiling certificates and specifying operational and management protocols only addresses a part of the problem of actually developing and implementing a secure PKI. What is also needed is the development of a certificate policy (CP) and certification practice statement (CPS), and then following those documents. The CP and CPS should address physical and personnel security, subject identification requirements, revocation policy, and a number of other topics. [POLPROC] provides a framework for certification practice statements.

3.6.5 Time-Stamp and Data Certification Services

In late 1998, the PKIX working group began two efforts that were not in the original working group charter, but were deemed to be appropriate because they described infrastructure services that could be used to provide desired security services. The first of these is time stamping, described in [TSP]. Time stamping is a service in which a trusted third party - a Time Stamp Authority, or TSA - signs a message, in order to provide evidence that it existed prior to a given time. Time stamping provides some support for non-repudiation, in that a user cannot claim that a transaction was later forged after compromise of a private key, because the existence of the signed time stamp indicates that the transaction in question could not have been created after the indicated time.

[TSP] also defines the role of a Temporal Data Authority, or TDA. A TDA is a Trusted Third Party (TTP) that creates a temporal data token. This temporal data token associates a message with a particular event and provides supplementary evidence for the time included in the time stamp token. For example, a TDA could associate the message with the most recent closing value of the Dow Jones Average. The temporal data with which the message is associated should be unpredictable in order to prevent forward dating of tokens.

At the Minneapolis IETF meeting, it was disclosed that the materials covered in the Timestamp Internet draft may be covered by patent(s). Use of the material covered by the patent(s) in question has not be granted by the patentholder. Thus, anyone interested in implementing the PKIX Timestamp draft must be aware of this intellectual property issue.

The second new effort is the definition of a Data Certification Server, or DCS, protocol [DCS]. A DCS is a Trusted Third Party that verifies the correctness of specific data submitted to it.

This is different from the TSP service in that a TSA will not attempt to parse and/or verify a message sent to it for certification;

instead, it will merely append a reliable indication of the current

time, and sign the resulting string-of-bits. This offers an indication that the given string-of-bits existed at a specified time; it does not offer any indication of the correctness or relevance of that string of bits. By contrast, the DCS certifies possession of data or the validity of another entity's signature. As part of this, the DCS verifies the mathematical correctness of the actual signature value contained in the request and also checks the full certification path from the signing entity to a trusted point (e.g., the DCS's CA, or the Root CA in a hierarchy).

The DCS supports non-repudiation in two ways. First, it provides evidence that a signature or public key certificate was valid at the time indicated in the token. The token can be used even after the corresponding public key certificate expires and its revocation information is no longer available on CRLs (for example). Second, the production of a data certification token in response to a signed request for certification of another signature or public key certificate also provides evidence that due diligence was performed by the requester in validating the signature or public key certificate.

4 PKIX Documents

This section describes each of the documents written by the PKIX working group. As PKIX progresses, this section will need to be continually updated to reflect the status of each document (e.g., Proposed Standard, Draft Standard, Standard, Informational Draft, Informational RFC, something-that-was-just-thrown-out-for-consideration, etc.)

4.1 Profile

DOCUMENT TITLE: Internet X.509 Public Key Infrastructure Certificate and CRL Profile (RFC 2459)

DESCRIPTION: This document describes the profiles to be used for X.509v3 certificates and version 2 CRLs by Internet PKI participants. The profiles include the identification of ISO/IEC/ITU and ANSI extensions which may be useful in the Internet PKI. The profiles are presented in the 1988 Abstract Syntax Notation One (ASN.1) rather

than the 1994 syntax used in the ISO/IEC/ITU standards. Would-be PKIX implementors and developers of certificate-using applications should start with [FORMAT] to ensure that their systems will be able to interoperate with other users of the PKI.

[FORMAT] also includes path validation procedures. The procedures presented are based upon the ISO/IEC/ITU definition, but the presentation assumes one or more self-signed trusted CA certificates.

The procedures are provided as examples only. Implementations are not required to use the procedures provided; they may implement whichever procedures are efficient for their situation. However, implementations are required to derive the same results as the example procedures.

STATUS: Proposed Standard.

DOCUMENT TITLE: Internet X.509 Public Key Infrastructure Representation of Elliptic Curve Digital Signature Algorithm (ECDSA) Keys and Signatures in Internet X.509 Public Key Infrastructure Certificates <draft-ietf-pkix-ipki-ecdsa-01.txt>

DESCRIPTION: This document provides Object Identifiers (OIDs) and other guidance for IPKI users who use the Elliptic Curve Digital Signature Algorithm (ECDSA). It profiles the format and semantics of the subjectPublicKeyInfo field and the keyUsage extension in X.509 V3 certificates containing ECDSA keys. This document should be used by anyone wishing to support ECDSA; others who do not support ECDSA are not required to comply with it.

STATUS: WG Last Call.

DOCUMENT TITLE: Internet X.509 Public Key Infrastructure Representation of Key Exchange Algorithm (KEA) Keys in Internet X.509 Public Key Infrastructure Certificates (RFC 2528)

DESCRIPTION: This document provides Object Identifiers (OIDs) and other guidance for IPKI users who use the Key Exchange Algorithm (KEA). It profiles the format and semantics of the subjectPublicKeyInfo field and the keyUsage extension in X.509 V3 certificates containing KEA keys. This document should be used by anyone wishing to support KEA; others who do not support ECDSA are

not required to comply with it.

STATUS: Informational RFC.

DOCUMENT TITLE: Internet X.509 Public Key Infrastructure Enhanced CRL Distribution Options (OpenCDP) <draft-ietf-pkix-ocdp-01.txt>

DESCRIPTION: This document proposes an alternative to the CRL Distribution Point (CDP) approach documented in [FORMAT]. OCDP separates the CRL location function from the process of certificate and CRL validation, and thus claims some benefits over the CDP approach.

STATUS: Under WG review.

DOCUMENT TITLE: Internet X.509 Public Key Infrastructure Qualified Certificates <draft-ietf-pkix-qc-00.txt>

DESCRIPTION: This document profiles the format for and defines requirements on information content in a specific type of certificates called Qualified Certificates. A "Qualified Certificate" is a certificate that is issued to a natural person (i.e., a living human being); contains an unmistakable identity based on a real name or a pseudonym of the subject; exclusively indicates non-repudiation as the key usage for the certificate's public key; and meets a number of requirements.

STATUS: Under WG review.

DOCUMENT TITLE: An Internet AttributeCertificate Profile for Authorizations <draft-ietf-pkix-acx509prof-00.txt>

DESCRIPTION: This document profiles the format for an defines requirements on X.509 Attribute Certificates to support authorization services required by various Internet protocols (TLS, CMS, and the consumers of CMS, etc.). Two profiles are defined on that supports basic authorizations and on the supports proxiable services.

STATUS: Under WG review.

DOCUMENT TITLE: Diffie-Hellman Proof-of-Possesion Algorithms <draft-

ietf-pkix-dhpop-00.txt>

DESCRIPTION: This documents describes two signing algorithms using the Diffie-Hellman key agreement process to provide a shared secret as the basis of the signature. It allows Diffie-Hellman a key agreement algorithm to be used instead of requiring that the public key being requested for certification correspond to an algorithm that is capable of signing and/or encrypting. The first algorithm generates a signature for a specific verifier where the signer and recipient have the same public key parameters. The second algorithm generates a signature for arbitrary verifiers where the signer and recipient do not have the same public key parameters.

STATUS: Under WG review.

4.2 Operational Protocols

DOCUMENT TITLE: Internet X.509 Public Key Infrastructure Operational Protocols - LDAPv2 (RFC 2559)

DESCRIPTION: This document describes the use of LDAPv2 as a protocol for PKI elements to publish and retrieve certificates and CRLs from a

certificate repository. [LDAPv2] is a protocol that allows publishing and retrieving of information.

STATUS: Proposed Standard.

DOCUMENT TITLE: Internet X.509 Public Key Infrastructure LDAPv2 Schema (RFC 2587)

DESCRIPTION: This document defines a minimal schema necessary to support the use of LDAPv2 for certificate and CRL retrieval and related functions for PKIX. This document supplements [LDAPv2] by identifying the PKIX-related attributes that must be present.

STATUS: Proposed Standard.

DOCUMENT TITLE: X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP <draft-ietf-pkix-ocsp-08.txt>

DESCRIPTION: This document specifies a protocol useful in determining

the current status of a certificate without the use of CRLs. A major complaint about certificate-based systems is the need for a relying party to retrieve a current CRL as part of the certificate validation process. Depending on the size of the CRL, this can cause severe problems for bandwidth-challenged devices. Depending on the frequency of CRL issuance, this can also cause timeliness problems. (E.g., if CRLs are only published weekly, with no interim releases, a certificate could actually have been revoked for just short of one week without it being on the current CRL, and thus improper use of that certificate could still be occurring.)

OCSP attempts to address those problems. It provides a mechanism whereby a relying party identifies one or more certificates to an approved OCSP "responder", and the responder sends back the current status of the certificate(s) - e.g., "revoked", "notRevoked", "unknown". This can dramatically reduce the bandwidth required to transmit revocation status - a relying party does not have to retrieve a CRL of many entries to check the status of one certificate. It can (although it is not guaranteed to) improve the timeliness of revocation notification, and thus reduce the window of opportunity for someone trying to use a revoked certificate.

STATUS: Approved as Proposed Standard.

DOCUMENT TITLE: Internet Public Key Infrastructure: Caching the Online Certificate Status Protocol

DESCRIPTION: To improve the degree to which it can scale, OCSP allows caching of responses - e.g., at intermediary servers, or even at the

relying party's end system. This document describes how to support OCSP caching at intermediary servers.

STATUS: Has been discontinued.

DOCUMENT TITLE: Internet X.509 Public Key Infrastructure Operational Protocols: FTP and HTTP (RFC 2585)

DESCRIPTION: This document describes the use of the File Transfer Protocol (FTP) and the Hyper-text Transfer Protocol (HTTP) to obtain certificates and CRLs from PKI repositories.

STATUS: Proposed Standard.

DOCUMENT TITLE: WEB based Certificate Access Protocol-- WebCAP/1.0

DESCRIPTION: This document specifies a set of methods, headers, and content-types ancillary to HTTP/1.1 to publish, retrieve X.509 certificates and Certificate Revocation Lists. This protocol also facilitates determining current status of a digital certificate without the use of CRLs. This protocol defines new methods, request and response bodies, error codes to HTTP/1.1 protocol for securely publishing, retrieving, and validating certificates across a firewall.

STATUS: Has been discontinued.

4.3 Management Protocols

DOCUMENT TITLE: Certificate Management Messages over CMS <draft-ietf-pkix-cmc-04.txt>

DESCRIPTION: This document defines the means by which PKI clients and servers may exchange PKI messages when using S/MIME's Cryptographic Message Syntax [CMS]as a transaction envelope. CMC supports the certificate request message body specified in the Certificate Request Message Format [CRMF] documents, as well as a variety of other certificate management messages. The primary purpose of this specification is to allow the use of an existing protocol (S/MIME)as a PKI management protocol, without requiring the development of an entirely new protocol such as CMP. A secondary purpose is to codify in IETF standards the current industry practice of using PKCS 10 messages [PKCS10] for certificate requests.

STATUS: Under WG review.

DOCUMENT TITLE: Internet X.509 Public Key Infrastructure Certificate Management Message Formats

DESCRIPTION: This document contains the formats for a series of messages important in certificate/PKI management. These messages let CA's, RA's, and relying parties communicate with each other. Note that this document only specifies message formats; it does not specify a protocol for transferring messages. That protocol can be

either CMP or CMC, or perhaps another custom protocol.

STATUS: Has been discontinued, as all useful information from it has been moved into [CMP] and [CMC].

DOCUMENT TITLE: Internet X.509 Certificate Request Message Format (RFC 2511)

DESCRIPTION: CRMF specifies a format recommended for use whenever a relying party is requesting a certificate from a CA or requesting that an RA help it get a certificate. The request message format was needed before many of the other message formats had to be finalized, and so it was put into a separate document. This document only specifies the format of a message. Specification of a protocol to transport that message is beyond the scope of CRMF.

STATUS: Proposed Standard.

DOCUMENT TITLE: Internet X.509 Public Key Infrastructure Certificate Management Protocols (RFC 2510)

DESCRIPTION: This document specifies a new protocol specifically developed for the purpose of transporting messages like those specified in CRMF among PKI elements. In general, CMP will be used in conjunction with CRMF, and will then be run over a transfer service (e.g., S/MIME, HTTP) to provide a complete PKI management service.

STATUS: Proposed Standard.

4.4 Policy Outline

DOCUMENT TITLE: Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework (RFC 2527)

DESCRIPTION: As noted before, the specification and implementation of certificate profiles, operational protocols, and management protocols is only part of building a PKI. Equally as important - if not more important - is the development and enforcement of a certificate security policy, and a Certification Practice Statement (CPS). The purpose of this document (PKIX-4) is to establish a clear relationship between certificate policies and CPSs, and to present a framework to assist the writers of certificate policies or CPSs with their tasks. In particular, the framework identifies the elements

that may need to be considered in formulating a certificate policy or a CPS. The purpose is not to define particular certificate policies or CPSs, per se.

STATUS: Informational RFC.

4.5 Time-Stamp and Data Certification Services

DOCUMENT TITLE: Internet X.509 Public Key Infrastructure Time Stamp Protocols <draft-ietf-pkix-time-stamp-01.txt>

DESCRIPTION: This document defines the specification for a time stamp service. It defines a Time Stamp Authority, or TSA, a trusted third party who maintains a reliable time service. When the TSA receives a time stamp request, it appends the current time to the request and signs it into a token to certify that the original request existed prior to the indicated time. This helps provide non-repudiation by preventing someone (either a legitimate user or an attacker who has successfully compromised a key) from "back-dating" a transaction. It also makes it more difficult to challenge a transaction by asserting that it has been back-dated. Note that the TSA does not provide any data parsing service; that is, the TSA does not attempt to validate that which it signs; it merely regards it as a string of bits whose meaning is unimportant, but existence is vital.

STATUS: Under WG review.

DOCUMENT TITLE: Internet X.509 Public Key Infrastructure Data Certification Server Protocols <draft-ietf-pkix-dcs-00.txt>

DESCRIPTION: This document defines a data certification service, or DCS, which can be used to certify both the existence and correctness of a message or signature. In contrast to the time stamp service described above, the DCS certifies possession of data or the validity of another entity's signature. As part of this, the DCS verifies the mathematical correctness of the actual signature value contained in the request and also checks the full certification path from the signing entity to a trusted point (e.g., the DCS's CA, or the Root CA in a hierarchy).

The DCS supports non-repudiation in two ways. First, it provides evidence that a signature or public key certificate was valid at the time indicated in the token. The token can be used even after the corresponding public key certificate expires and its revocation information is no longer available on CRLs (for example). Second, the production of a data certification token in response to a signed request for certification of another signature or public key certificate also provides evidence that due diligence was performed

by the requester in validating the signature or public key certificate.

STATUS: Under WG review.

DOCUMENT TITLE: Basic Event Representation Token <draft-ietf-pkix-bert1-01.txt>

DESCRIPTION: This document defines a finite method of representing a discrete instant in time as a referable event. The Basic Event Representation Token (BERT) is a light-weight binary token designed for use in large numbers over short periods of time. The tokens contain only a single instance of an event stamp and the trust process is referenced against an external reference.

STATUS: Under WG review.

DOCUMENT TITLE: Internet X.509 Public Key Infrastructure Extending trust in non repudiation tokens in time <draft-ietf-pkix-extend-trust-non-repudiation-token-00.txt>

DESCRIPTION: This document describes a method to maintain the trust in a token issued by a non-repudiation Trusted Third Party (NR TTP) (DCS/TSA/TDA) after the key initially used to establish trust in the token expires. The document describes a general format for storage of DCS/TS/TDA tokens for this purpose, which establishes a chain of custody for the data.

STATUS: Under WG review.

5 Advice to Implementors

This section provides guidance to those who would implement various parts of the PKIX suite of documents. The topics discussed in this section engendered significant discussion in the working group, and there was at times either widespread disagreement or widespread misunderstanding of them. Thus, this discussion is provided to help readers of the PKIX document set understand these issues, in the hope of fostering greater interoperability among eventual PKIX implementations.

5.1 Names

PKIX has been referred to as a "name-centric" PKI because the certificates associate public keys with names of entities. Each certificate contains at least one name for the owner of a particular public key. The name can be an X.500 distinguished name, contained in the subjectDN field of the certificate. There can also be names such

as RFC822 e-mail addresses, DNS domain names, and URIs associated with the key; these attributes are kept in the subjectAltName extension of the certificate. A certificate must contain at least one of these name forms, it may contain multiple forms if deemed appropriate by the CA based on the intended usage of the certificate.

5.1.1 Name Forms

There are two possible places to put a name in an X.509v3 certificate. One is the subject field in the base certificate (often called the "Distinguished Name" or "DN" field), and the other is in the subjectAltName extension.

5.1.1.1 Distinguished Names

According to [FORMAT], a PKIX certificate must have a non-null value in the Subject field, except for an end-entity certificate, which is permitted to have an empty subject field. Furthermore, if a certificate has a non-null Subject field, it MUST contain an X.500 Distinguished Name.

5.1.1.2 SubjectAltName Forms

In addition to the DN, a PKIX certificate may have one or more values in the subjectAltName extension.

The subjectAltName extension allows additional identities to be bound to the subject of the certificate - e.g., it allows "umbc.edu" and "130.85.1.3" to be associated with a particular subject, as well as "C=US, O=University of Maryland, L=Baltimore, CN=UMBC". X.509-defined options for this extension include: Internet electronic mail addresses; DNS names; IP addresses; and uniform resource indentifiers (URIs). Other options can exist, including locally-defined name forms.

A single subjectAltName extension can include multiple name forms, and multiple instances of each name form.

Whenever such Alternate Name forms are to be bound into a certificate, the subject alternative name (or issuer alternative name) extension must be used. It is technically possible to embed an Alternate Name Form in the subject field. For example, one could make a DN contain an IP address via a kludge such as "C=US, L=Baltimore, CN=130.85.1.3". However, this usage is deprecated; the alternative name extension is the preferred location for finding such information. As another example, some legacy implementations exist where an RFC822 name is embedded in the subject distinguished name as an EmailAddress attribute. Per [FORMAT], PKIX-compliant

implementations generating new certificates with electronic mail addresses MUST use the rfc822Name in the subject alternative name field to describe such entities. Simultaneous inclusion of the EmailAddress attribute in the subject distinguished name to support legacy implementation is deprecated but permitted.

In line with this, if the only subject identity included in a certificate is an alternative name form, then the subject distinguished name must be empty (technically, an empty sequence), and the subjectAltName extension must be present. If the subject field contains an empty sequence, the subjectAltName extension must be marked critical.

If the subjectAltName extension is present, the sequence must contain at least one entry. Unlike the subject field, conforming CAs shall not issue certificates with subjectAltNames containing empty GeneralName fields. For example, an rfc822Name is represented as an IA5String. While an empty string is a valid IA5String, such an rfc822Name is not permitted by PKIX. The behavior of clients that encounter such a certificate when processing a certification path is not defined by this working group.

Because the subject alternative name is considered to be definitively bound to the public key, all parts of the subject alternative name must be verified by the CA.

5.1.1.2.1 Internet e-mail addresses

When the subjectAltName extension contains an Internet mail address, the adress is included as an rfc822Name. The format of an rfc822Name is an "addr-spec" as defined in [RFC-822]. An addr-spec has the form local-part@domain; it does not have a phrase (such as a common name) before it, or a comment (text surrounded in parentheses) after it, and it is not surrounded by "<" and ">".

5.1.1.2.2 DNS Names

When the subjectAltName extension contains a domain name service label, the domain name is stored in the dNSName attribute(an IA5String). The string shall be in the "preferred name syntax," as specified by [DNS]. Note that while upper and lower case letters are allowed in domain names, no signifigance is attached to the case. In addition, while the string " " is a legal domain name, subjectAltName extensions with a dNSName " " are not permitted. Finally, the use of the DNS representation for Internet mail addresses (wpolk.nist.gov instead of wpolk@nist.gov) is not permitted; such identities are to be encoded as rfc822Name.

5.1.1.2.3 IP addresses

When the subjectAltName extension contains an iPAddress, the address shall be stored in the octet string in "network byte order," as specified in [IP]. The least significant bit (LSB) of each octet is the LSB of the corresponding byte in the network address. For IP Version 4, as specified in [IP], the octet string must contain exactly four octets. For IP Version 6, as specified in [IPv6], the octet string must contain exactly sixteen octets [RFC1883].

5.1.1.2.4 URIs

[FORMAT] notes "When the subjectAltName extension contains a URI, the name MUST be stored in the uniformResourceIdentifier (an IA5String). The name MUST be a non-relative URL, and MUST follow the URL syntax and encoding rules specified in [RFC 1738]. The name must include both a scheme (e.g., "http" or "ftp") and a scheme-specific-part. The scheme-specific-part must include a fully qualified domain name or IP address as the host. As specified in [RFC 1738], the scheme name is not case-sensitive (e.g., "http" is equivalent to "HTTP"). The host part is also not case-sensitive, but other components of the scheme-

specific-part may be case-sensitive. When comparing URIs, conforming implementations MUST compare the scheme and host without regard to case, but assume the remainder of the scheme-specific-part is case sensitive."

5.1.2 Scope of Names

The original X.500 work presumed that every subject in the world would have a globally-unique distinguished name. Thus, every subject could be easily located, and there would never be a conflict. All that would be needed is a sufficiently-large name space, and rules for constructing names based on subordination and location.

Obviously, that is not practical in the real world, for a variety of reasons. There is no single entity in the world trusted by everybody to reside at the top of the name space, and there is no way to enforce uniqueness on names for all entities. (These were among the reasons for the failure of PEM to be widely implemented.)

This does not mean, however, that a name-based PKI cannot work. It is important to recognize that the scope of names in PKIX is local; they need to be defined and unique only within their own domain.

Suppose for example that a Top CA is established with DN "O=IETF, OU=PKIX, CN=PKIX_CA". That CA will then issue certificates for users subordinate to it. The only requirement - and this can be enforced procedurally - is that no two distinct entities beneath this Top CA

have the same name. We can't prevent somebody else in the world from creating her own CA, called "O=IETF, OU=PKIX, CN=PKIX_CA", and it is not necessary to do so. Within the domain of the original Top CA, there will be no conflict, and the fact that there is another CA of the same name in some other domain is irrelevant.

This is analogous to the current DNS or IP address situations. On the Internet, there is only one node called www.ietf.org. The fact that there might be 10 different intranets, each with a host given the DNS name www.ieft.org, is irrelevant and causes no interoperability problems - those are different domains. However, if there were to be another node on the Internet with domain name www.ietf.org, then there would be a problem due to name duplication.

The same applies for IP addresses. As long as only one node on the Internet responds to the IP address 130.85.1.3, there is no problem, despite the fact that there are 100 different corporate Intranets, each using that same IP address. However, if two different nodes on the Internet each responded to the IP address 130.85.1.3, there would be a problem.

5.1.3 Certificate Path Construction

Certificate path construction has been the topic of many discussions in the WG. The issue centered around how best to get a certificate when the connection between the subject's name and the name of their CA, as well as the CA's repository (or directory), may not be obvious. Many proposals were put forth, including implementing a global X.500 Directory Service, using DNS SRV records, and using an attribute to point to the directory provider. At the end of the discussion the group decided to use the authority information access extension defined in [FORMAT], which allows the CA to indicate the access method and location of CA information and services. The extension would be included in subject's certificates and could be used to associate an Internet style identity for the location of repository to retrieve the issuer's certificate in cases where such a location is not related to the issuer's name.

Another discussion related to certificate path construction was where to store the CA and end-entity certificates in the directory (specifically LDAPv2 directories). Two camps emerged with different views on where to store CA and cross-certificates. In the CA's directory entry, one camp wanted self-issued certificates stored in the cACertificate attribute, certificates issued to this CA stored in the forward element of the crossCertificatePair, and certificates issued from this CA for other CAs in the reverse element of the crossCertificatePair attribute. The other camp wanted all CA certificates stored in the cACertificate attribute, and certificates

issued to/from another domain stored in the crossCertificatePair attribute. There was a short discussion that the second was more efficient than first, and that one solution or the other was more widely deployed. The end result was to indicate that self-issued certificates and certificates issued to the CA by CAs in the same domain as the CA are stored in the cACertificate attribute. The crossCertificatePair attribute's forward element will include all but

self-issued certificates issued to the CA. The reverse element may include a subset of certificates issued by the CA to other CAs. With this resolution both camp's implementations are supported and are free to chose the location of CA certificates to best support their implementation.

5.1.4 Name Constraints

A question that has arisen a number of times is "how does one enforce Name constraints when there is more than one name form in a certificate?" That is, [FORMAT] states that:

> Subject 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.

What does this mean? Suppose that there is a CA with DN "O=IETF, OU=PKIX, CN=PKIX_CA", with the subjectAltName extension having dNSName "PKIX_CA.IETF.ORG". Suppose that that CA has issued a certificate with an empty DN, with subjectAltName extension having dNSName set to "PKIX_CA.IETF.ORG", and rfc822Name set to Steve@PKIX_CA.IETF.ORG. The question is, are name constraints enforced on these two certificates - is the name of the end-entity certificate considered to be properly subordinate to the name of the CA?

The answer is "yes". The general rules for deciding whether a certificate meets name constraints are:

- If a certificate complies with name constraints in any one of its name forms, then the certificate is deemed to comply with name constraints.

- If a certificate contains a name form that its issuer does not, the certificate is deemed to comply with name constraints for that name form.

In deciding whether a name form meets name constraints, the following rules apply (in all cases Name B is the name in the name constraints

extension):

5.1.4.1 rfc822Names

Three variations are allowed:

- If the name constraint is specified as "larry@mail.mit.edu", then Name A is subordinate to Name B (in B's subtree) if Name A contains all of Name B's name (specifies a particular mailbox). For example, larry@mail.mit.edu is subordinate, but larry_sanders@mail.mit.edu is not.

- If the name constraint is specified as "mail.mit.edu", then Name A is subordinate to Name B (in B's subtree) if Name A contains all of Name B's name (specified all mailboxes on host mail.mit.edu). For example, curly@mail.mit.edu and mo@mail.mit.edu are subordinate, but mo@mail6.mit.edu and curly@smtp.mail.mit.edu are not.

- If the name constraint is specified as ".mit.edu", then Name A is subordinate to Name B (in B's subtree) if Name A contains all of Name B's name, with the addition of zero or more additional host or domain names to the left of Name B's name. That is, smtp.mit.edu is subordinate to .mit.edu, as is pop.mit.edu. However, mit.edu is not subordinate to .mit.edu. When the constraint begins with a "." it specifies any address within a domain. In the previous example, "mit.edu" is not in the domain of ".mit.edu".

Note: If rfc822 names are constrained, but the certificate does not contain a subject alternative name, the EmailAddress attribute will be used to constrain the name in the subject distinguished name. For example (c is country, o is organization, ou is organizational unit, and em is EmailAddress), Name A ("c=US, o=MIT, ou=CS, em=curly@mail.mit.edu") is subordinate to Name B ("c=US, o=MIT, ou=CS") (in B's subtree) if Name A contains all of Name B's names.

5.1.4.2 dNSNames

Name A is subordinate to Name B (in B's subtree) if Name A contains all of Name B's name, with the addition of zero or more additional domain names to the left of Name B's name. That is, www.mit.edu is subordinate to mit.edu, as is larry.cs.mit.edu. However, www.mit.edu is not subordinate to web.mit.edu.

5.1.4.3 x.400 addresses

Name A is subordinate to Name B (in B's subtree) if Name A contains

all of Name B's name. For example (c is country-name, admd is administrative-domain-name, and o is orgnaization-name, ou is organizational-unit-name, pn is personal-name, sn=surname, and gn is given-name in both Name A and B), the mneumonic X.400 address (using PrintableString choices for c and admd) "c=US, admd=AT&T, o=MIT, ou=cs, pn='sn=Doe,gn=John'" is subordinate to "c=US, admd=AT&T, o=MIT, ou=cs" and "c=US, admd=AT&T, o=MIT, pn='sn=DOE,gn=JOHN'" (pn is a SET that includes, among other things, sn and gn).

5.1.4.5 DNs

Name A is subordinate to Name B (in B's subtree), if Name A contains all of Name B's name, treating attribute values encoded in different types as different strings, ignoring case in PrintableString (in all other types case is not ignored), removing leading and trailing white space in PrintableString, and converting internal substrings of one or more consecutive white space characters to a single space. For example, (c is country, o is organization, ou is organizational unit, and cn is common name):

- Assuming PrinatString types for all attribute values in Name A and B, "c=US, o=MIT, ou=CS" is subordinate to "c=us, o=MIT, ou=cs", as is "c=US, o=MIT, ou=CS, ou=adminstration". Another example, "c=US, o=MIT, ou=CS, cn= John Doe" (note the white spaces) is subordinate to "c=US, o=MIT, ou=CS, cn=John Doe".

- Assuming UTF8String types for all attribute values in Name A and B, "c=US, o=MIT, ou=CS, ou=administration" is subordinate to "c=US, o=MIT, ou=CS", but "c=US, o=MIT, ou=cs, ou=Adminstration". "c=US, o=MIT, ou=CS, cn= John Smith" (note the white spaces) is not subordinate to "c=US, o=MIT, ou=CS, cn= John Smith".

- Assuming UTF8String types for all attribute values in Name A and PrintableString types for all attribute values in Name B, "c=US, o=MIT, ou=cs" is subordinate to "c=US, o=MIT, ou=CS" if the verification software supports the full comparison algorithm in the X.500 series. "c=US, o=MIT, ou=cs" is NOT subordinate to "c=US, o=MIT, ou=CS" if the verification software supports the comparison algorithm in [FORMAT].

5.1.4.6 URIs

The constraints apply only to the host part of the name. Two variations are allowed:

- If the name constraint is specified as ".mit.edu", then Name A is subordinate to Name B (in B's subtree) if Name A contains

all of Name B's name, with the addition of zero or more additional host or domain names to the left of Name B's name. That is, www.mit.edu is subordinate to .mit.edu, as is curly.cs.mit.edu. However, mit.edu is not subordinate to .mit.edu. When the constraint begins with a "." it specifies a host.

- If the name constraint is specified as "abc.mit.edu", then Name A is subordinate to Name B (in B's subtree) if Name A conatins all of Name B's name. No leftward expansion of the host or domain name is allowed.

5.1.4.7 iPaddresses

Two variations are allowed depending on the IP version:

- For IPv4 addresses: Name A (128.32.1.1 encoded as 80 20 01 01) is subordinate to Name B (128.32.1.0/255.255.255.0 encoded as 80 20 00 00 FF FF FF 00) (in B's subtree) if Name A falls within the net denoted in Name B's CIDR notation.

- For IPv6 addresses: Name A (4224.0.0.0.8.2048.8204.16762 encoded as 10 80 00 00 00 00 00 00 00 08 08 00 20 0C 41 7A) is subordinate to Name B (4224.0.0.0.8.2048.8204.0/ 65535.65535.65535.65535.65535.65535.65535.0 encoded as 10 80 00 00 00 00 00 00 00 08 08 00 20 0C 00 00 FF FF FF FF FF FF FF FF FF FF FF FF FF FF 00 00) (in B's subtree) if Name A falls within the net denoted in Name B's CIDR notation.

5.1.4.8 Others

As [FORMAT] does not define requirements for the name forms otherName, ediPartyName, or registeredID there are no corresponding name constraints requirements.

5.1.5 Wildcards in Name Forms

A "wildcard" in a name form is a placeholder for a set of names; e.g. "*.mit.edu" meaning "any domain name ending in .mit.edu", and *@aol.com meaning "email address that uses aol.com". There are many people who believe that allowing wildcards in name forms in PKIX certificates would be a useful thing to do, because it would allow a single certificate to be used by all members of a group. These members would presumably have attributes in common; e.g., access rights to some set of resources, and so issuance of a certificate with a wildcard for the group would simplify management.

After much discussion, the PKIX working group decided to permit the

use of wildcards in certificates. That is, it is permissible for a PKIX-conformant CA to issue a certificate with a wildcard. However, the semantics of subject alternative names that include wildcard characters are not addressed by PKIX. Applications with specific requirements may use such names but must define the semantics.

5.1.6 Name Encoding

A very important topic that consumed much of the WG's time was the implementation of the directory string choices. While the long term goal of the IETF was clear, use UTF8String, the short term goals were not so clear. Many implementations only use PrintableString, others use BMPString, and still others use Latin1String (ISO 8859-1) and tag it as TeletexString (there are others still). To ensure that there is consistency with encodings [FORMAT] defines a set of rules for the string choices. PrintableString was kept as the first choice because of it's widespread support by vendors. BMPString was the second choice, also for it's widespread vendor support. Failing support by PrintableString and BMPString, UTF8String must be used. In keeping with the IETF mandate, UTF8String can be used at any time if the CA supports it. Also, processing implementations that wish to support TeletexString should handle the entire ISO 8859-1 character set and not just the Latin1String subset.

5.2 POP

Proof of Possession, or POP, means that the CA is adequately convinced that the entity requesting a certificate containing a public key Y has access to the private key X corresponding to that

public key.

POP is important because it provides an appropriate level of assurance in the correct operation of the PKI as a whole. At its lowest level, POP counters the "self-inflicted denial of service"; that is, an entity voluntarily getting a certificate that cannot be used to sign or encrypt/decrypt information. However, as the following two examples demonstrate, POP also counters less direct, but more severe, threats.

5.2.1 POP for Signing Keys

It is important to provide POP for keys used to sign material, in order to provide non-repudiation of transactions. For example, suppose Alice legitimately has private key X and its corresponding public key Y. Alice has a certificate from Charlie, a CA, containing Y. Alice uses X to sign a transaction T. Without POP, Mal could also get a certificate from Charlie containing the same public key, Y. Now, there are two possible threats: Mal could claim to have been the

real signer of T; or Alice can falsely deny signing T, claiming that it was instead Mal. Since no one can reliably prove that Mal did or did not ever possess X, neither of these claims can be refuted, and thus the service provided by and the confidence in the PKI has been defeated. (Of course, if Mal really did possess X, Alice's private key, then no POP mechanism in the world will help, but that is a different problem.)

One level of protection can be gained by having Alice, as the true signer of the transaction, include in the signed information her certificate or an identifier of her certificate (e.g., a hash of her certificate). This might make it more difficult for Mal to claim authorship - he would have to assert that he incorrectly included Alice's certificate, rather than his own. However, it would not stop Alice from falsely repudiating her actions. Since the certificate itself is a public item, Mal indeed could have inserted Alice's certificate into the signed transaction, and thus its presence does not indicate that Alice was the one who participated in the now-repudiated transaction. The only reliable way to stop this attack is to require that Mal prove he possesses X before his certificate is issued.

For signing keys used only for authentication, and not for non-repudiation, the threat is lower because users may not care about Alice's after-the-fact repudiation, and thus POP becomes less important. However, POP SHOULD still be done wherever feasible in this environment, by either off-line or on-line means.

5.2.2 POP for Key Management Keys

Similarly, POP for key management keys (that is, keys used for either key agreement or key exchange) can help to prevent undermining confidence in the PKI. Suppose that Al is a new instructor in the Computer Science Department of a local University. Al has created a draft final exam for the Introduction to Networking course he is teaching. He wants to send a copy of the draft final to Dorothy, the Department Head, for her review prior to giving the exam. This exam will of course be encrypted, as several students have access to the computer system. However, a quick search of the certificate repository (e.g., search the repository for all records with subjectPublicKey=Dorothy's-value) turns up the fact that several students have certificates containing the same public key management key as Dorothy. At this point, if no POP has been done by the CA, Al has no way of knowing whether all of the students have simply created these certificates without knowing the corresponding private key (and thus it is safe to send the encrypted exam to Dorothy), or whether the students have somehow acquired Dorothy's private key (and thus it is certainly not safe to send the exam). Thus, the service to be

provided by the PKI - allowing users to communicate with one another, with confidence in who they are communicating with - has been totally defeated. If the CA is providing POP, then either no students will have such certificates, or Al can know with certainty that the students do indeed know Dorothy's private key, and act accordingly.

CMP requires that POP be provided for all keys, either through on-line or out-of-band means. There are any number of ways to provide POP, and the choice of which to use is a local matter. Additionally, a certificate requester can provide POP to either a CA or to an RA, if the RA can adequately assure the CA that POP has been performed. Some of the acceptable ways to provide POP include:

- Out-of-band means:

- For keys generated by the CA or an RA (e.g., on a token such as a smart card or PCMCIA card), possession of the token can provide adequate proof of possession of the private key.

- For user-generated keys, another approach can be used in environments where "key recovery" requirements force the requester to provide a copy of the private key to the CA or an RA. In this case, the CA will not issue the requested certificate until such time as the requester has provided the private key. This approach is in general not recommended, as it is extremely risky (e.g., the system designers need to figure out a way to protect the private keys from compromise while they are being sent to the CA/RA/other authority, and how to protect them there), but it can be used.

- On-line means:

  - For signature keys that are generated by an end-entity, the requester of a certificate can be required to sign some piece of data (typically, the certificate request itself) using the private key. The CA will then use the requested public key to verify the signature. If the signature verification works, the CA can safely conclude that the requester had access to the private key. If the signature verification process fails, the CA can conclude that the requester did not have access to the correct private key, and reject the request.

  - For key management keys that are generated by the requester, the CA can send the user the requested public key, along with the CA's own private key, to encrypt some piece of

data, and send it to the requester to be decrypted. For example, the CA can generate some random challenge, and require some action to be taken after decryption (e.g., "decrypt this encrypted random number and send it back to me"). If the requester does not take the requested action, the CA concludes that the requester did not possess the private key, and the certificate is not issued.

Another method of providing POP for key management keys is for the CA to generate the requested certificate, and then send it to the requester in encrypted form. If the requester does not have access to the appropriate private key, the requester cannot decrypt the certificate, and thus cannot use it. After some period of time in which the certificate is not used, the CA will revoke the certificate. (This only works if the certificate is not made available to any untrusted entities until after the requester has successfully decrypted it.)

5.3 Key Usage Bits

The key usage extension defines the purpose (e.g., encipherment, signature, certificate signing) of the key contained in the certificate. This extension is used when a key that could be used for more than one operation is to be restricted. For example, when an RSA key should be used only for signing, the digitalSignature and/or nonRepudiation bits would be asserted. Likewise, when an RSA key should be used only for key management, the keyEncipherment bit would be asserted. When used, this extension should be marked critical.

The eight bits defined for this extension identify seven mechanisms and one service, namely:

- digitalSignature
- nonRepudiation
- keyEncipherment
- dataEncipherment
- keyAgreement
- keyCertSign
- cRLSign
- encipherOnly
- decipherOnly

According to [FORMAT], bits in the KeyUsage type are used as follows:

- The digitalSignature bit is asserted when the subject public key is used to verify digital signatures that have purposes other than non-repudiation, certificate signature, and CRL signature. For example, the digitalSignature bit is asserted when the subject

public key is used to provide authentication via the signing of

ephemeral data.

- The nonRepudiation bit is asserted when the subject public key is used to verify digital signatures used to provide a non-repudiation service which protects against the signing entity falsely denying some action, excluding certificate or CRL signing.

- The keyEncipherment bit is asserted when the subject public key is used for key transport. For example, when an RSA key is to be used for key management, this bit must asserted.

- The dataEncipherment bit is asserted when the subject public key is used for enciphering user data, other than cryptographic keys.

- The keyAgreement bit is asserted when the subject public key is used for key agreement. For example, when a Diffie-Hellman key is to be used for key management, this bit must asserted.

- The keyCertSign bit is asserted when the subject public key is used for verifying a signature on certificates. This bit may only be asserted in CA certificates.

- The cRLSign bit is asserted when the subject public key is used for verifying a signature on revocation information (e.g., a CRL).

- The meaning of the encipherOnly bit is undefined in the absence of the keyAgreement bit. When the encipherOnly bit is asserted and the keyAgreement bit is also set, the subject public key may be used only for enciphering data while performing key agreement.

- The meaning of the decipherOnly bit is undefined in the absence of the keyAgreement bit. When the decipherOnly bit is asserted and the keyAgreement bit is also set, the subject public key may be used only for deciphering data while performing key agreement.

PKIX does not restrict the combinations of bits that may be set in an instantiation of the keyUsage extension.

The discussion on the PKIX mailing list has centered on the digitalSignature bit and the nonRepudiation bit. The question has come down to something like: When support for the service of non-repudiation is desired, should both the digitalSignature and nonRepudiation bits be set, or just the nonRepudiation bit?

(It is noted that provision of the service of non-repudiation requires more than a single bit set in a certificate. It requires an entire infrastructure of components to preserve for some period of

time the keys, certificates, revocation status, signed material, etc., as well as a trusted source of time. However, the nonRepudiation key usage bit is provided as an indicator that such keys should not be used as a component of a system providing a non-repudiation service.)

According to [SIMONETTI], the intent is that the digitalSignature bit should be set when what is desired is the ability to sign ephemeral transactions; e.g., for a single session authentication. These transactions are "ephemeral" in the sense that they are important only while they are in existence; after the session is terminated, there is no long-term record of the digital signature and its properties kept. When something is intended to be kept for some period of time, the nonRepudiation bit should be set. This generally implies that an application will digitally sign something; thus, some implementors turn on the digitalSignature bit as well. Other implementors, however, keep the two bits mutually exclusive, to prevent a single key from being used for both ephemeral and long-term signing.

While [FORMAT] is silent on this specific issue, the working group's general conclusion is that a certificate may have either or both bits set. If only the nonRepudiation bit is set, the key should not be used for ephemeral transactions. If only the digitalSignature bit is set, the key should not be used for long-term signing. If both bits are set, the key may be used for either purpose.

To actually enforce this requires that an application understands whether it is signing ephemeral transactions or for the long-term. The application developers will have to understand the difference, and set up their checks on the key usage bits field accordingly. For example, TLS implementors should check only the digitalSignature bit, and ignore the nonRepudiation bit. S/MIME implementors, though, will have a difficult choice to make, since their application could be used for either purpose, and they will generally accept signing using keys associated with certificates having either or both bits being turned on. Certification Authorities should know what applications they are providing certificates for, and provide certificates according to the requirements of those applications. If CA's are tied into non-repudiation systems, they may treat certificates differently when the nonRepudiation bit is turned on (e.g., store information associated with the certificate - like the user's identification provided during certificate registration, or certificate generation date/time stamps - for longer periods of time, in more secure environments).

The bottom line is that this is an area where PKI implementors are somewhat limited in what they can do. The onus is on developers of

certificate-using systems to understand their requirements and request certificates with the appropriate bits set.

5.4 Trust Models

(This section will describe the various trust models that PKIX can support. It is important to note that PKIX is bound to neither a pure hierarchical model a la PEM, nor a web of trust model a la PGP. PKIX can support either of those models, or any flavor in between. The implications of different trust models should be described:

- efficiency of revocation - certification path building - etc.)

6 Acknowledgements

A lot of the information in this document was taken from the PKIX source documents; the authors of those deserve the credit for their own words. Other good material was taken from mail posted to the PKIX working group mail list (ietf-pkix@imc.org). Among those with good things to say were (in alphabetical order, with apologies to anybody we've missed): Sharon Boeyen, Santosh Chokhani, Warwick Ford, Russ Housley, Steve Kent, Ambarish Malpani, Matt Fite, Michael Myers, Tim Polk, Stefan Santesson, Dave Simonetti, and.

7 References

[BERT1] McNeil, M., and Glassey, T., "Basic Event Representation Token," <draft-ietf-pkix-bert1-01.txt>, May 1999.

[CACHE] "Internet Public Key Infrastructure: Caching the Online Certificate Status Protocol," <draft-ieft-pkix-ocsp-caching-00.txt>, April 1998.

[CMC] Myers, M., Liu, X., Fox, B., and Weinstein, J., "Certificate Management Messages over CMS," <draft-ieft-pkix-cmc-04.txt>, May 1999.

[CMP] Adams, C., Farrell, S., "Internet X.509 Public Key Infrastructure Certificate Management Protocols", RFC 2510, March

1999.

[CMS] R. Housley, "Cryptographic Message Syntax," <draft-ietf-smime-cms-13.txt>, April 1999.

[CRMF] Myers, M., Adams, C., Solo, D., and Kemp, D., "Internet X.509 Certificate Request Message Format," RFC 2511, March 1999.

[DCS] Adams, C., and Zuccherato, R., "Internet X.509 Public Key

Infrastructure Data Certification Server Protocols", <draft-ietf-pkix-dcs-00.txt>, 23 September 1998.

[DHPOP] Prafullchandra, H., and Schaad, J., "Diffie-Hellman Proof-of-Possession Algorithms," <draft-ietf-pkix-dhpop-00.txt>, February 1999.

[DNS] Mockapetris, P.V., "Domain names - concepts and facilities," RFC 1034, November 1987.

[ECDSA] Bassham, L., Johnson, D., and Polk, W., "Internet x.509 Public Key Infrastructure: Representation of Elliptic Curve Digital Signature Algorithm (ECDSA) Keys and Signatures in Internet X.509 Public Key Infrastructure Certificates," <draft-ietf-pkix-ipki-ecdsa-01.txt>, November 1997.

[ETNPT] Namjoshi, P., "Internet X.509 Public Key Infrastructure Extending trust in non repudiation tokens in time," <draft-ietf-pkix-extend-trust-non-repudiation-token-00.txt>, May 1999.

[IP] Postel, J., "Internet Protocol," RFC 791, September 1981.

[IPv6] Deering, S., and Hinden, R., "Internet Protocol, Version 6 [IPv6] Specification," RFC 1883, December 1995.

[FORMAT] Housley, R., Ford, W., Polk, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and CRL Profile," RFC 2459, January 1999.

[FTPHTTP] Housley, R., and Hoffman, P., "Internet X.509 Public Key Infrastructure Operational Protocols: FTP and HTTP," RFC 2585, July 1998.

[KEA] Housley, R., and Polk, W., "Internet X.509 Public Key Infrastructure Representation of Key Exchange Algorithm (KEA) Keys in Internet X.509 Public Key Infrastructure Certificates," RFC 2528, March 1999.

[LDAPv2] Yeong, Y., Howes, T., and Kille, S., "Lightweight Directory Access Protocol", RFC 1777, March 1995.

[MISPC] Burr, W., Dodson, D., Nazario, N., and Polk, W., "MISPC Minimum Interoperability Specification for PKI Components, Version 1", <http://csrc.nist.gov/pki/mispc/welcome.html> September 3, 1997.

[OCDP] Hallam-Baker, P., and Ford, W., "Internet X.509 Public Key Infrastructure Enhanced CRL Distribution Options (OpenCDP)," <draft-ietf-pkix-ocdp-01.txt>, August 7, 1998.

[OCSP] Myers, M., Ankney, R., Malpani, A., Galperin, S., and Adams, C., "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP," <draft-ietf-pkix-ocsp-08.txt>, March 1999.

[PEM] Kent, S., "Privacy Enhancement for Internet Electronic Mail: Part II: Certificate-Based Key Management," RFC 1422, February 1993.

[PKCS10] RSA Laboratories, "The Public-Key Cryptography Standards(PKCS)," RSA Data Security Inc., Redwood City, California, November 1993 Release.

[PKI-LDAPv2] Boeyen, S., Howes, T., and Richard, P., "Internet X.509 Public Key Infrastructure Operational Protocols - LDAPv2," RFC 2559, April 1999.

[POLPRAC] Chokhani, S., and Ford, W., "Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework," RFC 2527, March 1999.

[QC] Santesson, S., Polk, W., and Gloeckner, P., "Internet X.509 Public Key Infrastructure Qualified Certificates", <draft-ietf-pkix-qc-00.txt>, 3 February 1999.

[RFC-822] Crocker, D., "Standard for the Format of ARPA Internet Text Messages," RFC 822, August 1982.

[SCHEMA] Boeyen, S., Howes, T., and Richard, P., "Internet X.509 Public Key Infrastructure LDAPv2 Schema," RFC 2587, June 1999.

[SIMONETTI] Simonetti, D., "Re: German Key Usage", posting to ietf-pkix@imc.org mailing list, 12 August 1998.

[TSP] Adams, C., Cain, P., Pinkas, D., and Zuccherato, R., "Internet X.509 Public Key Infrastructure Time Stamp Protocols", <draft-ietf-pkix-time-stamp-02.txt>, May 1999.

[WEB] Reddy, S., "WEB based Certificate Access Protocol-- WebCAP/1.0," <draft-ietf-pkix-webcap-00.txt>, April 19, 1998.

[X.509] ITU-T Recommendation X.509 (1997 E): Information Technology - Open Systems Interconnection - The Directory: Authentication Framework, June 1997.

[X9.42] ANSI X9.42-199x, Public Key Cryptography for The Financial Services Industry: Agreement of Symmetric Algorithm Keys Using Diffie-Hellman (Working Draft), December 1997.

[X9.55] ANSI X9.55-1995, Public Key Cryptography For The Financial

Services Industry: Extensions To Public Key Certificates And Certificate Revocation Lists, 8 December, 1995.

[X9.57] ANSI X9.57-199x, Public Key Cryptography For The Financial Services Industry: Certificate Management (Working Draft), 21 June, 1996.

8 Security Considerations

TBSL

9 Editor's Address

Alfred Arsenault U. S. Department of Defense 9800 Savage Road Suite 6734 Fort George G. Meade, MD 20755-6734 (410) 684-7114 awarsen@missi.ncsc.mil

Sean Turner IECA, Inc. 9010 Edgepark Road Vienna, VA 22182 (703)

628-3180 turners@ieca.com

10 Disclaimer

This work constitutes the opinion of the editors only, and may not reflect the opinions or policies of their employers.