Trust Models and Management in Public-Key Infrastructures

John Linn

RSA Laboratories
20 Crosby Drive
Bedford, MA 01730 USA

jlinn@rsasecurity.com

6 November 2000

ABSTRACT

This paper presents and compares several trust models currently being considered and applied for use with public-key certificate infrastructures based on the X.509 specification, including subordinated hierarchies, cross-certified meshes, hybrids, bridge CAs, and trust lists. Approaches and issues concerning constraints on path validity are also considered, as are aspects of path construction.

1. INTRODUCTION AND GOALS

Fundamentally, Public-Key Infrastructures (PKIs) are facilities for representing and managing trust relationships. Relying parties depend on chains of PKI-provided certificates in order to determine whether or not to accept a given key as representing a particular principal. The chains extend from "trust anchor" points, which verifiers directly accept as reliable, reaching certified principals through zero or more intermediary CAs. In all cases, verifiers' trust is based on some set of public keys provided out-of-band, constituting points of entry into the infrastructure.

A range of trust models have emerged as PKI designs and products have grown beyond local domains to satisfy needs of larger and more diverse communities. Standards now provide tools to constrain various aspects of certification paths, but these tools require validation in operational deployment. Path construction raises complex issues of its own, more so in some trust models than in others, but has received narrower attention than the validation of paths once constructed (e.g, as considered in [6], [4], and [8]). Each of these areas contributes to the operation and management of a large-scale PKI. This paper characterizes and evaluates a variety of alternatives and associated issues, assessing current and likely directions as PKIs continue evolving to larger scale deployments. Since standards and current industry directions emphasize development of PKIs based on the X.509 specification [6] and profiled for Internet usage in [4], this paper's treatment emphasizes aspects related to X.509 environments.

2. ALTERNATIVE TRUST MODELS

This section characterizes several different trust models that have been applied or proposed for use in X.509-based PKIs:

- Subordinated hierarchy
- Cross-certified mesh

- Hybrid
- Bridge CA
- Trust lists

In addition to individual "pure" models, combinations of these models can be integrated, with varying capacities and implications for interoperability. A verifier can be "multi-headed" in policy terms, able to apply the rules of multiple models in the course of constructing and verifying certification paths. Until and unless PKI usage practice converges to a single common model, such flexibility will be important to achieve cross-vendor interoperability.

Within all models, cross-certification (the certification of one CA by another) is the fundamental technique used to compose certification paths. Some, but not all, models constrain the subsets of CAs which are eligible to cross-certify one another. Cross-certification can, but need not always, be reciprocal; it is often valid for a CA A to certify B independent of whether or not B certifies (or is eligible to certify) A.

All models require that participants obtain trusted knowledge of one or more public keys to enable them to validate certification paths. This information may be obtained through configuration, by out-of-band management coordination, and/or by users' acceptance of keys offered within their communications. These keys are commonly, though not necessarily, provided to verifiers in the form of self-signed certificates; although this approach is a convenient implementation strategy and allows convenient integrity checking, it does not add fundamental enhancement of trust relative to other representations.

Generally, the discussions are framed in terms of path validation by relying parties; notably, however, the models can also be deployed in a delegated mode, within which a relying party delegates validation processing to be performed by a separate trusted server. The delegated model can simplify client-side processing, can reduce network traffic for certificate retrieval, and can centralize policy management for the subscribers within an administrative domain. Relying parties can indicate their accepted policies and/or trust anchors, or can defer their selection to the delegated validator. If desired, path construction can be delegated, with validation of a constructed path performed directly by the relying party. Facilities for delegated processing at the path level are current topics of discussion in the Internet standards community, but have not stabilized at this writing.

2.1 Subordinated Hierarchy

This model corresponds to a hierarchic structure, with a well-known root CA and inferior descendants that are strictly limited in terms of the name subtrees that they can certify. The root CA certifies its immediate descendants, which in turn certify their descendants, and so on. Subordinate CAs need not certify their superiors for operation within a subordinated hierarchy. Within this model, each participant must have knowledge of the root CA's public key, which forms the fundamental trust anchor for all participants. Compromise of the corresponding private key is disastrous for the security of its hierarchy. In a strict hierarchy, however, a root CA normally issues or revokes certificates only on an occasional basis. As a result, it is likely that such a CA will be able to remain off-line, contributing to reduced compromise potential for its private key.

Subordinated hierarchies are probably the earliest PKI trust models to have been considered in detail, dating back at least to the early evolution of X.509 and their contemplated use in Internet Privacy-Enhanced Mail (PEM) [7]. They allow tight controls to be placed on the namespace elements that different CAs may certify, consistent with the principle of least privilege. Given only a validated name, it is easy to delimit the set of CAs contributing to its validation. Path construction is simple, as naming-based traversal from the root to any leaf is simple and authoritative, providing the only valid path to that leaf. A message originator can enclose with a message a full certification path, and can be assured that the path will be acceptable to all recipients within the hierarchy; this property is particularly convenient for store-and-forward environments such as E-mail.

For a relying party to validate a message's originator within a subordinated hierarchy, it need only validate the path from the root to that originator; reconstruction of the already-known path component from the root to the relying party is not required. As a variant approach, cross-certification may be permitted within the hierarchy, as to support high-usage paths or direct inter-organizational relationships; if this is applied, a path between any two leaves can always be constructed by traversing the root, but short-cut alternatives can be constructed for some cases.

This model is directly applicable within isolated, hierarchic enterprises, but has generally proven more difficult to apply subordinated hierarchies which cross organizational boundaries. It is frequently hard to identify a single entity that all communicating parties are willing to trust as a root, and to establish common policies acceptable to all participants. Participants are sometimes reluctant to rely on other organizations to preserve the integrity of their subordinated namespaces. It is important, therefore, to be able to adapt the subordinated hierarchy model in a manner that will allow multiple, independently managed hierarchies and which will permit communication across hierarchy boundaries. The basic prerequisite for this adaptation is to provide verifiers with access to the root keys of other hierarchies. Trust lists, considered in Section 2.5 of this paper, provide one means to manage dissemination of multiple root keys for such an environment. Meshes, Hybrid Trust Models, and Bridge CAs, discussed respectively in Sections 2.2, 2.3, and 2.4, represent approaches that avoid the need for preconfiguration of distant root keys within endpoints.

2.2 Cross-Certified Mesh

This model is very broad in its generality; any CA may certify any other CA, except if and as naming constraints are applied. (Subordinated hierarchies can be considered as special cases of meshes, with particularly tight naming constraints, but do not possess many of meshes' differentiating characteristics.) As such, the mesh model serves well to represent dynamically changing organizational structures, but its generality can make it more difficult to determine whether a particular CA is or is not an appropriate certifier for another CA. Long, non-hierarchic paths spanning many organizations may not be uniformly trustworthy. In contrast to subordinated hierarchies, a validated name within a general mesh is insufficient to identify the CAs involved in its path. Access decisions based on authentication within a mesh model may, therefore, require inspection not only of an authenticated name but also of the path that contributed to its authentication.

This model corresponds particularly well to environments where communicating organizations are not hierarchically related to one another, but can also be applied to model hierarchical structures. Interoperable paths are enabled incrementally, as CA pairs cross-certify; if cross-certification is initiated across anticipated high-usage paths, the lengths of those paths can be reduced. General traversal within a mesh requires that cross-certificates be issued reciprocally, for both directions; if

this model is applied to a hierarchy, a given CA certifies its direct superiors as well as its direct subordinates. For operation within this model, each participant must be pre-configured with the public key of its local CA to use as a trust anchor.

Path construction in a mesh environment is significantly more complicated than within a subordinated hierarchy, requiring the ability to iteratively obtain and combine sets of cross-certificates issued by various CAs. To verify a message's originator in a mesh environment, a full path beginning at the originator and proceeding to the verifier's domain must be obtained. Each verifier constructs paths relative to its own set of one or more trust anchors. Loops must be detected and rejected, as must dead-end candidate paths encountered in the course of path construction. Multiple paths may exist between a pair of points in a mesh; their selection may need to be filtered (e.g., on a policy screening basis) and prioritized.

Unconstrained meshes afford broad privilege to any CA that has obtained a cross-certified relationship with another. Certain X.509 extensions (BasicConstraints' PathLenConstraint, PolicyConstraints, PolicyMappings) were introduced in X.509 V3, allowing mesh paths to be constrained in various dimensions. Their suitability and sufficiency will benefit from usage experience and warrants further analysis; these issues are considered in Section 3 of this paper.

To construct paths within a mesh environment, potentially many directories or other certificate databases (though not necessarily their associated CAs) must be accessible when a path is to be constructed. If the directory entry for one CA on a path is unavailable, it may not be possible to construct that path. If and as longer paths become commonplace, this will multiply the dependency on directory availability. For some cases, this impact may be moderated by availability of alternate paths through other CAs. Tradeoffs between accessibility and desired confidentiality for published directory information may further impact the level of interoperability that mesh models can achieve in practice.

Meshes do not enable general path construction to be accomplished unless the necessary cross-certificates have been pre-established between one or more pairs of CAs positioned along the path. Further, a message's originator cannot generally construct and present a single path for his or her certificate that will be acceptable to all possible verifiers. Meshes do, however, have the important advantage of being deployable in a "bottom-up" fashion without dependence on the prior availability of a top-level root CA. End users can, instead, initiate their involvement in a distributed PKI in a decentralized manner, beginning with their local CAs. When organizations identify mutual needs for secure communication, this can be supported on a pairwise fashion through cross-certification between the organizations' CAs.

2.3 Hybrid Model

When independent enterprises establish separate subordinated hierarchies, and then develop a need to communicate, some form of cross-certification must be applied to link the hierarchies. Pairwise cross-certification between subordinated hierarchies has the attractive characteristic that it can preserve the enterprises' ability to enforce name constraints within their hierarchies. In its informative Annex G, ISO's [5] discusses this case specifically, presenting a hybrid trust model with the following properties:

- Multiple root CAs exist

- All non-root CAs are certified within a root CA's hierarchy, with paths certified both "downward" from the root and "upward" towards it

- Root CAs establish a cross-certified mesh among themselves, so each hierarchy can reach every other hierarchy via a single cross-certificate at the root level

- Selective cross-certification between non-root CAs is permitted

The ISO Hybrid model distinguishes three types of cross-certificates, as follows:

- *Hierarchical cross-certificates* parallel the paths of subordinated hierarchies, but extend upward towards the root CAs. These allow relying parties to use their local CA as a trust anchor, yet discover certification paths extending to other arbitrary points within the PKI.

- *General cross-certificates* interconnect CAs, either at the root level or between lower points within the connected hierarchies.

- *Special cross-certificates* are intended to allow selective establishment of certification paths that may not conform to the restrictions ordinarily imposed hierarchically along the paths from the root CAs. In particular, the specification notes that special cross-certificates may be created between 'leaf' CAs, the holders of certificates that bear a path length constraint of '0'. If a CA's certificate contains a BasicConstraints extension with a pathLenConstraint of 0, and that CA issues a special cross-certificate to another CA, the cross-certificate would not form an acceptable path component without a selective override to BasicConstraints processing. This approach implies a question: how are verifiers to determine whether or not a particular cross-certification should be accepted? Potentially, such paths could be accepted only by those verifiers that directly trust the CAs issuing the special cross-certificates, but further definition is required in this area.

The hybrid model provides important value by combining two factors:

- the ability to construct arbitrary paths simply, assuming that the inter-root mesh is easy to traverse, and

- the ability to introduce direct "short-cut" cross-certifications for high-usage paths joining lower points within the hierarchies.

The inter-root mesh is available as a universal resource for certified connectivity, but need not be used for all paths. Participants in this model may choose whether to employ the keys of one or more roots as pre-configured trust anchors, or to use the key of a trusted local CA; interoperability is possible even if different choices are made by two communicating endpoints. The ICE-TEL web of hierarchies trust model, described in [2], represents a form of hybrid model; among other elements, it incorporates user-directed depth-based control of policies for acceptance of cross-certifications.

2.4 Bridge CA

Pairwise inter-hierarchy cross-certification, as in the hybrid model, serves well to link a small number of hierarchies. It does not scale well to larger numbers, however, since the necessary number

of cross-certifications grows as *n(n-1)*, where *n* is the number of hierarchies to be fully connected. Recognizing this limitation, the Bridge CA model was developed for the U.S. Federal PKI [1].

The Bridge CA model embodies a central cross-certification authority, whose purpose is to provide cross-certificates rather than acting as the root of certification paths. Effectively, its services offer the inter-root mesh of the hybrid trust model discussed in Section 2.3, operating as a peer to the roots of the hierarchies it connects rather than at a level superior to them. When an enterprise cross-certifies with a Bridge CA, it obtains the ability to construct and validate paths to other enterprises that have also cross-certified with that Bridge CA. Name constraints may be applied on the Bridge's cross-certification of an enterprise, and/or on the enterprise's cross-certification of the Bridge. Within the Bridge CA model, as in the mesh model, a participant is preconfigured with the public key of its local CA to act as a trust anchor; configured knowledge of the Bridge CA's own public key is not required.

The Bridge CA approach appears to be a prime candidate to extend PKIs across large numbers of organizations, particularly those sharing compatible policies. The U.S. Federal Bridge represents an important initial example of this architecture, but additional deployments of Bridge CAs may, e.g., occur within industry associations.

2.5 Trust Lists

The concept of trust lists appears in various contexts, such as the sets of embedded root keys that are familiar within web browsers. In this model, client systems (or their delegated verifiers) are provided with the public keys of a set of trusted roots. To be validated successfully, a certificate must chain to one of these trusted roots, sometimes directly and without intervening CAs. For interoperability purposes, individual CAs operating within other trust models may be certified within a verifier's trust list.

The trust list model differs fundamentally from the other models in that the necessary actions to make remote entities resolvable through trust paths are driven by prospective verifiers (and their administrators) rather than being made on behalf of the entities that are to be resolved. As a result, different verifiers within a trust list environment, even if they share a common CA, may experience different results when seeking to validate a particular entity. Relative to cross-certified approaches, use of trust lists moves management overhead away from intermediary CAs and towards end systems and their associated administrators. Consistent management is required if a common policy is to be achieved across an organization.

Unlike the other models, the trust list model can require that endpoints or their delegates be configured with large sets of public keys, if communication with peers under the jurisdiction of large numbers of CAs is to be supported. In the web environment, where huge numbers of browsers are preloaded with a small number of trusted root keys, the operational franchises of root key holders have become commercially valuable assets. Trust lists are conveniently useful among a relatively small number of globally well-known CAs (e.g., signers of SSL server certificates), for direct use within enterprises, and/or to support interactions across a modest and predetermined set of enterprise boundaries. In larger contexts, fine-grained configuration of trust lists to represent appropriate sets of local authorities for use at particular clients could require extensive management.

2.6 Comparison Among Models

The following table summarizes some primary characteristics of the alternative trust models.

	Subordinated Hierarchy	Cross-Cert. Mesh	Hybrid	Bridge CA	Trust Lists
Trust Anchor Public Key(s)	Hierarchy Root	Local CA	Hierarchy Root or Local CA	Local CA	Listed CAs that are to be accepted as signers
Inter-Enterprise Support	Weak beyond common root	Good through moderate numbers of enterprises	Good through moderate numbers of enterprises	Very good through large scale	Fair: may require intensive management
Path Construction	Simple within local hierarchy: down from root only	Hard: may be multiple routes to source, requiring iteration	Medium: multiple routes may exist, but simple path known	Simple: all non-local paths traverse bridge	Simple but limited: all available paths begin within local trust list
Directory Dependency	Low	High	Medium	Medium	Low
Growth Model	Top-down	Pairwise between CAs	Top-down or pairwise	Pairwise with bridge	Recognition by verifiers

3. PATH CONSTRAINTS: APPROACHES AND ISSUES

Within the various PKI trust models, it may be possible to establish certification paths to and through large numbers of entities, whose identities and operational policies may or may not be recognized by, or acceptable to, verifiers. X.509 V3's policy management extensions allow administrators to configure a range of tradeoffs between interoperability and tightly constrained policy control. Detailed processing procedures have been specified and recently refined within the standards community, but support and usage practice for these extensions is still maturing.

3.1 Current X.509 Policy Extensions

This section summarizes the path processing constraints currently defined in X.509 certificate extensions.

3.1.1 BasicConstraints' PathLenConstraint

The PathLenConstraint component within a certificate's BasicConstraints extension allows a limit to be placed on the number of CA certificates which may follow that certificate within a valid path. This responds to an intuitive sense that longer paths involving more entities at greater distance from a verifier may be less trustworthy; in practice, however, it has often proven difficult to set a threshold value which provides useful assurance without rejecting paths which could be usefully accepted. Since this constraint is applied at a CA level, potentially distant from some or all of the end users and systems which may wish to make use of a certification path, it may be difficult to assign values that are suitable for all of the path's consumers. Addition of a new hierarchic level within a remote organization, e.g., could play havoc with a validation structure closely tied to PathLenConstraint values.

3.1.2 CertificatePolicies

The CertificatePolicies extension lists policies that are recognized by an issuing CA as applicable to a particular certificate, associating a certificate with corresponding Certificate Policy and Certification Practices documents. Commonly, identified policies are associated with particular organizations or domains of usage, particular sets of operational procedures, and/or particular levels of technical assurance. To avoid combinatorial explosion as multiple policy dimensions are represented for certified entities, it may become appropriate to separately identify different policy dimensions applicable to an entity. For example, one OID might indicate an entity's procedural assurance level, with a separate OID indicating its technical assurance level, rather than a single OID indicating a specific combination of procedural and technical assurance.

Qualifiers may accompany the OIDs used to identify policies, but can be ignored at a relying party's option. The qualifiers can reference human-readable information (e.g., a pointer to a certification practices statement) or algorithmically processable data; the latter are notably more useful as elements for automated path validation.

3.1.3 PolicyConstraints

The PolicyConstraints extension allows an issuing CA to specify that certificate policies must be explicitly identified (with the CertificatePolicies extension) for a path to be accepted as valid, and/or that policy mapping (with the PolicyMappings extension) be inhibited. These constraints may be applied to all certificates occurring in a path beyond the certificate containing the PolicyConstraints extension, or may be applied (using the SkipCerts element) only following a specified number of intermediate certificates. As with BasicConstraints' PathLenConstraint, it may prove difficult in practice to set reliably useful non-zero values for SkipCerts.

3.1.4 PolicyMappings

The PolicyMappings extension allows an issuing CA to specify that one of its certificate policies can be considered equivalent to a different policy associated with the domain of the subject being certified. Use of this extension is convenient from a mechanistic perspective, but practical factors may limit its adoption. Policies of different organizations, even if functionally comparable in the large, are likely to diverge in subtle or detailed aspects. The U.S. Federal Bridge CA is intended to

make extensive use of PolicyMappings, and may provide valuable operational experience in this area.

3.2 Aspects of Constraints

The defined path constraint extensions emphasize matching of OID-identified policies and allow a distance metric to be applied to certification paths. These facilities can be mechanized in a straightforward fashion, and are being applied as tools to manage some aspects of trust, but give an intuitive sense of addressing only a subset of a large problem space. The broad goal at hand is to provide assurance that the set of certificates and CAs making up a certification path comprises a suitable basis for accepting the binding between a principal's certified name and public key. Here, as elsewhere, assurance can never be absolute; if measurable, its metrics may differ from the viewpoints of different verifiers.

Policy identifiers represent a CA's assertion of conformance to a particular policy, and their appearance in successive certificates within a path can be used to establish an accountable chain of affiliation with a particular policy. On the other hand, the identifiers do not themselves provide independent verifiers with a means to assess that conformance. If and as third-party accreditation of CAs becomes common, the fact of such accreditation might be made visible to verifiers in the form of certificates or some other machine-queriable representation.

The X.509 certification path processing procedure supports algorithmic validation that a particular policy is indicated on all certificates within a chain or subchain. If the initial certificate's assertion is valid, this can demonstrate a chain of accountability, but a chain extending from an initial invalid assertion can't be automatically distinguished from a valid case without other information. To counter this threat, relying parties must obtain knowledge of one or more "policy roots" for the policies they recognize, and validate that paths asserting those policies chain to one of the known policy roots. For some cases, this can be accomplished by extending relying parties' trust in initial public keys to span the policy identifiers in the self-signed certificates where those keys are represented. This would not, however, serve to distinguish valid from invalid cases when particular policy identifiers first appear at a lower point within a chain; validation of such cases may require additional management of policy trust information.

The defined facilities, based on matching comparisons among policy identifiers and associated qualifiers, operate at coarse granularity. A certifier either asserts conformance with a policy in its entirety, or not at all. Given that policies are lengthy specifications in their own right, will naturally evolve over time, and that different organizations may share a core of common policy elements but wish to incorporate their own extensions or variations, a tension emerges. The interpretations of object identifiers are intended to be immutable, but the cost of changing a policy identifier would be high, for interoperability purposes and for the management of certificates whose embedded OIDs and qualifiers would be changed. These factors create a strong, perhaps irresistible, temptation to define many policy elements within Certification Practices Statements (CPSs) whose contents may change without changing the policy OIDs and qualifiers that reference them. This practice dilutes the effectiveness of algorithmic comparisons for the purpose of determining whether certifiers' procedures match.

At the cost of additional complexity, finer-grained specification could be achieved by defining syntactic representations for policy components. It is unlikely, however, that a single globally defined syntax could represent all of the policy elements which diverse communities might wish to employ.

This implies a need for extensible syntax. Unfortunately, broad and independent use of extensibility features could conflict with the ability to compare policy indicators algorithmically.

As the issuers of certificates, CAs are the entities that must impose any path constraints that are applied using certificate extensions. A tradeoff arises. CAs closer to the endpoints of a path are more likely to apply constraints consistent with the endpoints' needs, as to designate policies recognized or required by the endpoints. On the other hand, CAs which are closer to the midpoints of paths are likely to be responsible for establishing the inter-organizational cross-certifications needed for interoperability between arbitrary peers. These CAs comprise the backbone of the network which certification paths define. Where long certification paths arise, it is unlikely that these essential intermediaries will be cognizant of, or will be able to directly serve, the policy requirements of all endpoints that may depend on the paths they enable. In the limit, some trust criteria may necessarily be resolved within endpoints themselves, based on policies that are neither mechanizable nor necessarily visible by intermediary CAs.

4. PATH CONSTRUCTION: APPROACHES AND ISSUES

4.1 Linkage Techniques

To construct a certification path, a sequence of valid linked certificates must be collected. Within a path, each adjacent pair of certificates is linked, by a combination of matching names and/or key identifiers. For a certificate pair to be suitable for use, it must also satisfy a number of constraints, such as appropriate validity periods, verifiable signatures, authorization of a certifier to act as a CA, use of acceptable algorithms, indication of acceptable policies, and so forth. Path construction and validation, therefore, requires comparisons among many data elements, contained within certificates and/or known to participants through other means.

X.509 refers to a CA *n*'s "forward certificates" as certificates which have been issued to *n* by other CAs, with its "reverse certificates" those issued by *n* and certifying others. These terms suggest that forward progress through a path necessarily proceeds towards a trust anchor. In fact, however, path construction may be undertaken in either direction:

- In the "forward" direction (also known as "bottom-up"): from a specified certificate towards a trust anchor (at individual steps, proceeding from certificates' issuers and/or associated key identifiers and seeking other certificates that bear matching elements corresponding to their subjects), or

- In the "reverse" direction (also known as "top-down"): from a trust anchor towards a specified certificate (at individual steps, proceeding from certificates' subjects and/or associated key identifiers and seeking other certificates that bear matching elements corresponding to their issuers)

In general, and particularly in non-hierarchic trust models, it may be possible to construct multiple paths between a given pair of certificates, and these must be prioritized. Loops must be detected and avoided. Desirably, use of unnecessarily long paths should be minimized. The problems of efficiently constructing effective certification paths in complex environments have received some consideration in available materials (e.g., [3]), but will benefit from continuing analysis and discussion.

4.2 Retrieval Topics

4.2.1 Certificate Storage

Within X.500 and Lightweight Directory Access Protocol (LDAP) directories, certificates are stored within elements of multi-valued attributes, contained within the directory entries of related entities. Several directory attributes are defined for certificate storage purposes:

- The userCertificate attribute type contains the public-key certificates for a user, as issued by one or more CAs.

- The cACertificate attribute of a CA's directory entry stores any self-issued certificates, and certificates that have been issued to the CA by other CAs in the same realm as the CA.

- The forward elements of the crossCertificatePair attribute of a CA's directory entry store all certificates issued to a CA, except self-issued certificates. Optionally, the reverse elements of the crossCertificatePair attribute in a particular CA's directory entry may contain a subset of certificates issued by that CA to other CAs. Individual attribute values within crossCertificatePair may contain either or both forward and reverse elements. When both forward and reverse elements appear within a single attribute value, the issuer name in one certificate is expected to match the subject name in the other and vice versa. Further, the subject public key in one certificate is to be capable of verifying the digital signature on the other certificate and vice versa. It is not required, however, that all matching pairs of elements must necessarily be stored within the same attribute values.

- A pkiPath attribute, recently defined within X.509, may be used to store certification paths, each consisting of a sequence of cross-certificates. This attribute, if used, would contain some certification paths from that CA to other CAs, potentially enabling efficient retrieval of selected high-usage paths. Its usage, however, is not required, and its contents may not represent the complete set of a CA's forward certification paths. Given sufficient adoption, pkiPath may become a useful place to publish frequently used paths, allowing them to be accessed even when the directory entry of one of a path's constituent CAs becomes unavailable. This could enhance overall path availability, particularly within mesh environments.

Particularly in large mesh models, CAs will issue and hold large numbers of cross-certificates. Path construction depends on the ability to obtain these certificates, and efficient retrieval is important. If the optional reverse elements of crossCertificatePair attributes are used at all, they may contain only a proper subset of the cross-certificates that a CA has issued. This complicates the task of constructing reverse direction paths, beginning at a trust anchor and extending to an end entity. It has been suggested that population of the reverse elements be mandated, but this question remains pending as of this writing.

4.2.2 Selective Retrieval and Matching Rules

Efficient directory-based path construction in large environments would benefit from the ability to retrieve certificates selectively from an entity's directory entry, based on criteria specified by the requestor. Current LDAP directories are limited in their ability to support selective retrieval, instead requiring clients to retrieve all of an attribute's values. If a directory entry contains a large number of

certificates, it is inefficient for the requestor to obtain all of them, then to sift through the returned values to pinpoint a single certificate of interest. Extensions to LDAP schema or protocol elements to support selective retrieval have been proposed but have not been finalized for standardization purposes at this writing.

The current version of X.509 defines a range of matching rules for certificates, cross-certificate pairs, lists of certificates, certificate policies, and supported algorithms within certificates. The simplest, CertificateExactMatch, is most widely supported in current LDAP technology, but allows selection only by the pair of certificate issuer and serial number. An alternative X.509 matching rule, CertificateMatch, allows matching on a broader range of certificate elements; its support would enable more flexible selection among certificates.

5. CONCLUSIONS

As PKIs grow beyond enterprise boundaries to span larger and more heterogeneous environments, trust management becomes more complex. Several approaches have been defined to link certification structures, differing in their security properties, their scalability, their management requirements, their implications on path construction and validation, and their dependencies on directory services. It appears likely that usage of multiple models will proliferate. As a result, verifiers' abilities to flexibly accommodate multiple approaches will be important for interoperability. Delegated path validation through domain-level services may be helpful in providing such flexibility. Operational experience will be important in informing selection among alternative models, and in validating the usability and sufficiency of policy extensions within certificates.

6. ACKNOWLEDGMENT

The author would like to thank all contributors to the Internet Engineering Task Force (IETF) Public-Key Infrastructure (X.509) (PKIX) working group, which has presented a valuable forum for discussion of many of the topics considered in this paper.

7. REFERENCES

[1] W. E. Burr, "Public Key Infrastructure (PKI) Technical Specifications: Part A – Technical Concept of Operations", Working Draft, 4 September 1998, available at http://csrc.nist.gov/pki/twg/baseline/pkicon20b.PDF.

[2] D. W. Chadwick, A. J. Young, N. Kapidzic Cicovic, "Merging and Extending the PGP and PEM Trust Models – The ICE-TEL Trust Model", IEEE Network Magazine, May 1997.

[3] S. Chokhani, P. Hesse, "Path Development in a PKI Network Environment", presentation to U.S. Federal PKI Technical Working Group, July 1999.

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

[5] ISO/TC68/SC2, "Banking – Certificate Management Part 1: Public Key Certificates", ISO/DIS 15782-1, 14 December 1999.

[6], ITU-T, Draft Revised Recommendation X.509, “Draft Revised ITU-T Recommendation X.509 | ISO/IEC 9594-8: Information Technology – Open Systems Interconnection – The Directory: Public-Key and Attribute Certificate Frameworks”, Version 4, June 2000.

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

[8] U. Maurer, “Modelling a Public-Key Infrastructure”, Proceedings, 1996 European Symposium on Research in Computer Security (ESORICS ’96), E. Bertino (Ed.), Lecture Notes in Computer Science, Berlin: Springer-Verlag, vol. 1146, pp. 325-350, 1996.