Functional Requirements for Path Validation Systems

Path Validation Working Group

Draft Version 0.8

March 30, 2004

1 Introduction

This document specifies requirements for PKI clients used in the Federal PKI. Requirements are specified for path validation, path discovery, and auditing. This document considers two basic scenarios for implementing these requirements: PKI client functionality may be performed locally or delegated entirely to a trusted server. Supplemental requirements are specified for clients and servers for the special case of delegated PKI processing.

In most cases, a relying party (e.g., an e-mail application) will perform its own path discovery and validation locally. Auditing may be handled by the same component that performs path discovery and validation, or may be left to each PKI-enabled application.

However, a relying party may offload these responsibilities to a trusted server. In this case, a Delegated Path Validation (DPV) server performs the path validation, path discovery, and most audit functions. The relying party sends path validation requests to the DPV server and verifies and processes the DPV server's responses; all PKI-specific processing is performed on the server.

Sections 2 and 3 of this document specify the requirements for path validation and path discovery. In the case of a "fat" client that performs its own path discovery and validation, these requirements apply to the client itself. In the case of a "thin" client that offloads path discovery and validation to a DPV server, these requirements apply to the DPV server. Section 4 specifies the auditing requirements. In a fat client environment, all auditing requirements are implemented by the PKI client and the PKI-enabled applications. In a thin client environment, the server handles generic auditing requirements, while the PKI-enabled client must fulfill application-specific requirements. Section 5 specifies requirements for the communication between a DPV client and a DPV server. The requirements in section 5 only apply to thin client implementations.

2 Path Validation

PKI clients used in the Federal government must be able to validate the certification paths that will be found in a multi-domain PKI architecture. In such environments, PKI clients will encounter, and must be able to process, several constraint extensions in addition to the basic X.509 fields. The draft *NIST Recommendation on X.509 Path Validation* [10] specifies varying levels of functionality for path validation modules. Path validation modules that are to be used in the Federal government, which must be suitable for use within a multi-domain PKI architecture, must meet the requirements for a *Bridge-enabled PVM* as specified in [10].

3 Path Discovery

This document does not attempt to prescribe the method that must be used to perform path discovery, but merely specifies the information sources that an implementation must be able to utilize when searching for the certificates and CRLs needed to construct a

certification path. *Certification Path Building* [2] and *Understanding Certification Path Construction* [5] may contain useful information on the implementation of path building functionality.

The Federal PKI allows for two different approaches to certificate and CRL distribution. The first approach assumes a well-connected directory system. The second approach assumes that directories are disjoint and that certificate contents themselves direct retrieval strategies. Section 3.1 describes path discovery strategies that apply in the case of a well-connected directory system. Section 3.2 describes path discovery strategies that apply in the case of a disjoint directories. Section 3.3 summarizes the requirements for PKI path discovery clients.

3.1 Directory Based Certificate and CRL Discovery

In the classic PKI model, certificates and CRLs are stored in a system of directories. PKI clients include the IP address or DNS name of their local directory as part of their configuration and all certificates and CRLs needed for path discovery and validation may be located by querying this directory using LDAP. If the requested information is not in the directory that the PKI client queries then the directory may either obtain the information on behalf of the PKI client (e.g., using DSP) or return an LDAP referral to the PKI client indicating where the information can be found. The certificates and CRLs needed to perform path validation are stored in the directory according to the rules specified in X.509 and RFC 2587 [1].

When performing path discovery, the PKI client may either begin with the end entity certificate and build towards the trust anchor or begin with the trust anchor and build towards the end entity certificate. When building from the end entity to the trust anchor, the PKI client will need to find, for each certificate in the path, the intermediate certificate that precedes that certificate in the path. These certificates may be obtained by querying the issuing CA's directory entry (i.e., by obtaining the contents of the cACertificate and/or crossCertificatePair attributes of the directory entry that corresponds to the name in the issuer field of the certificate). The cACertificate attribute of the CA's directory entry will contain any certificates that the CA has issued to itself (e.g., key rollover certificates) and may also contain some of the cross-certificates issued to the CA by other CAs. The crossCertificatePair attribute of the CA's directory entry will contain all cross-certificates issued to the CA by other CAs.

When building from the trust anchor to the end entity certificate, the PKI client will need to find, for each intermediate certificate in the path, the certificate that follows that certificate in the path. These certificates, other than the end entity certificate itself, may be obtained by querying the subject CA's directory entry (i.e., by obtaining the contents of the cACertificate and/or crossCertificatePair attributes of the directory entry that corresponds to the name in the subject field of the certificate). The cACertificate attribute of the subject CA's directory entry will contain any certificates that the CA has issued to itself (e.g., key rollover certificates). The crossCertificatePair attribute of the

subject CA's directory entry may contain cross-certificates that the subject CA has issued to other CAs.

It should be noted that, according to X.509 and RFC 2587, the inclusion of cross-certificates issued by a CA in the CA's directory entry is optional. So, PKI clients should be prepared to perform path validation from the end entity to the trust anchor in cases where the cross-certificates necessary to perform path validation in the other direction have not been included in the directory. In a mesh architecture, it will usually be the case that a CA will include all of the CA certificates that it has issued in the crossCertificatePair attribute of its directory entry. In a hierarchical architecture, however, it will frequently be the case that CAs will not include certificates issued to subordinate CAs in their directory entries.

If a certificate includes a cRLDistributionPoints extension, then the PKI client should use the contents of that extension to determine where the CRL that covers the certificate may be found. If the cRLDistributionPoints extension includes a directoryName, then the PKI client should obtain the CRL from the authorityRevocationList or certificateRevocationList attribute of the directory entry indicated by that directoryName. If the certificate does not include a cRLDistributionPoints extension, then the PKI client should obtain the CRL from the authorityRevocationList or certificateRevocationList attribute of the certificate issuer's directory entry (i.e., the directory entry that corresponds to the name in the issuer field of the certificate).

3.2 Certificate Content Based Certificate and CRL Discovery

In some cases, certificates will include extensions that explicitly state where other certificates needed to construct a certification path may be found. Certificates may also include an extensions that explicitly states where certificate status information may be found.

In order to aid in the construction of certification paths from an end entity certificate to a trust anchor, certificates may include an authorityInfoAccess extension [4] with one or more instances of the id-ad-caIssuers access method. Each instance of the id-ad-caIssuers access method specifies an access location where certificates that were issued to the issuer of the certificate that includes the extension may be found. While the access location may take many forms, path discovery implementations should be prepared to handle cases in which the access location is either an HTTP or LDAP URI. When the access location is an HTTP URI, it either points to a file containing a single certificate or to a certs-only CMS message [13] that includes a collection of certificates that were issued of the issuer of this certificate. When the access location is an LDAP URI, it specifies the location (e.g., IP address or DNS name of server, directory entry, and possibly attributes) where certificates issued to the issuer of this certificate may be found. If the LDAP URI does not specify attributes, then one should look in the cACertificate and crossCertificatePair attributes of the specified directory entry.

In order to aid in the construction of certification paths from a trust anchor to an end

entity certificate, certificates may include a subjectInfoAccess extension [4] with one or more instances of the id-ad-caRepository access method. Each instance of the id-ad-caRepository access method specifies an access location where CA certificates that were issued by the subject of the certificate that includes the extension may be found. While the access location may take many forms, path discovery implementations should be prepared to handle cases in which the access location is either an HTTP or LDAP URI. When the access location is an HTTP URI, it either points to a file containing a single certificate or to a certs-only CMS message [13] that includes a collection of certificates that were issued by the subject of the certificate. When the access location is an LDAP URI, it specifies the location (IP address or DNS name of server, directory entry, and possibly attributes) where CA certificates issued by the subject of this certificate may be found. If the LDAP URI does not specify attributes, then one should look in the cACertificate and crossCertificatePair attributes of the specified directory entry.

While many certificates include an authorityInfoAccess extension, the subjectInfoAccess extension is not commonly included in certificates at the present time. In general, it should be assumed that the subjectInfoAccess extension will not be included in certificates that are issued to hierarchically subordinate CAs. When a certificate includes an authorityInfoAccess extension with an id-ad-caIssuers access method or a subjectInfoAccess extension with an id-ad-caRepository access method, path discovery implementations should use the information in these extensions to locate certificates rather than querying their local directory.

A certificate may also include a cRLDistributionPoints extension that includes a URI to specify the location of a CRL that covers this certificate. Path discovery implementations should be prepared to handle cases in which the certificate includes a cRLDistributionPoints extension with either an HTTP or LDAP URI. When the cRLDistributionPoints extension includes an HTTP URI, the URI specifies the location of a file that contains the CRL. When the cRLDistributionPoints extension includes an LDAP URI, the URI specifies the location (IP address or DNS name of server, directory entry, and attribute) where the CRL is located.

3.3 Requirements for Path Discovery

Path discovery implementations may build certification paths starting with the end entity certificate and building towards the trust anchor, starting with the trust anchor and building towards the end entity certificate, or using a combination of the two. However, path discovery implementation must be prepared to build paths from the end entity towards the trust anchor within hierarchies in order to accommodate the case where a CA has issued a certificate to a subordinate CA but does not include that certificate in its directory entry or include a subjectInfoAccess extension in the certificate.

To support directory based certificate and CRL discovery, path discovery implementations must be able to:

1. Configure one or more LDAP directories to query;

2. Process and follow LDAP V3 referrals.

When building from the end entity toward the trust anchor, path discovery implementations must be able to utilize all of the following methods to find the necessary CA certificates to construct the path:

1. Obtain CA certificates located in a certs-only CMS message that is pointed to by an HTTP URI in an authorityInfoAccess extension. If the implementation examines the file extension, it must accept and process both .p7c and .p7b files.
2. Obtain CA certificates located in an LDAP accessible directory that is pointed to by an LDAP URI in an authorityInfoAccess extension that specifies the LDAP server's name (IP address or DNS name), the directory entry in which the certificates are located, and the attributes (cACertificate and/or crossCertificatePair) within which the certificates may be found. Where the LDAP URI does not specify attributes, the implementation should request both the cACertificate and crossCertificatePair.
3. When a certificate does not include an authorityInfoAccess extension with an id-ad-caIssuers access method, obtain certificates from the cACertificate and crossCertificatePair attributes of the certificate issuer's directory entry by querying a locally configured directory.

Path discovery implementations that build from the trust anchor towards the end entity certificate in non-hierarchical environments must be able to utilize all of the following methods to find the necessary CA certificates to construct the path:

1. Obtain CA certificates located in a certs-only CMS message that is pointed to by an HTTP URI in a subjectInfoAccess extension. If the implementation examines the file extension, it must accept and process both .p7c and .p7b files.
2. Obtain CA certificates located in an LDAP accessible directory that is pointed to by an LDAP URI in a subjectInfoAccess extension that specifies the LDAP server's name (IP address or DNS name), the directory entry in which the certificates are located, and the attributes (cACertificate and/or crossCertificatePair) within which the certificates may be found. Where the LDAP URI does not specify attributes, the implementation should request both the cACertificate and crossCertificatePair.
3. When a CA certificate does not include a subjectInfoAccess extension with an id-ad-caRepository access method, obtain certificates from the cACertificate and crossCertificatePair attributes of the certificate subject's directory entry by querying a locally configured directory.

Path discovery implementations must be able utilize all of the following methods to obtain the CRLs needed to validate certificates:

1. Obtain the CRL from the file specified in an HTTP URI in a cRLDistributionPoints extension.

2. Obtain the CRL from an LDAP directory when the location of the CRL is specified in an LDAP URI in a cRLDistributionPoints extension where the LDAP URI specifies the LDAP server's name (IP address or DNS name), the directory entry in which the CRL is located, and the attribute (certificateRevocationList or authorityRevocationList) that holds the CRL.
3. Obtain the CRL from the certificateRevocationList or authorityRevocationList attribute of the directory entry specified using the directoryName name form in the cRLDistributionPoints extension by querying a locally configured directory.
4. When a certificate does not include a cRLDistributionPoints extension, obtain the CRL from the certificateRevocationList or authorityRevocationList attribute of the certificate issuer's directory entry by querying a locally configured directory.

4 Auditing

Auditing requirements for PKI-enabled applications depend upon the context in which certificates are used. PKI-enabled applications that verify digital signatures for financial disbursement will have different auditing requirements from a system that uses PKI for authentication. However, general purpose PKI implementations need to create audit records that satisfy more stringent applications, but may leave the decision to maintain audit records to each application. Details for auditing requirements may be found in [8] and [9].

Path validation systems must maintain sufficient information to support re-validation of the end certificate, or provide that information to the application. Information required to support re-validation includes:

- All certificates in the certification path; and
- Status information (e.g., CRLs) for all certificates in the path.

Where path validation mechanisms maintain the audit records, the application may be provided with a reference to these records rather than the information itself.

5 Delegated Path Validation Protocol Requirements

Delegated path validation (DPV) presents an alternative to the classic PKI model. In this model, a PKI client requests a trusted server perform certification path development and validation for a given certificate. The PKI client does not need any additional knowledge about the PKI, the directory structure, or the details of path validation. The DPV server must, of course, implement the functionality identified in Sections 2 and 3.

Additional requirements are imposed on both PKI client and server by the DPV protocols themselves. The PKI client expresses its requirements for certificate validation, transmits the request to the trusted server, and waits for a response. The DPV server must be able

to interpret the client requirements for path validation. The DPV server must retrieve the necessary certificates, CRLs, and status information to build the path to the client's designated trust anchor. The DPV server must then perform path validation with respect to the client's security requirements (e.g., acceptable certificate policies). Finally, the server must be able to generate, sign and transmit the response to the client. The client must validate the server's digital signature and process the response.

DPV clients must support the following requirements:

- Support policy selection through at least one of the following methods:
  - Specify requirements for validation, including trust anchor and acceptable certificate policies, in the request.
  - Configure distinct DPV servers to handle validation with respect to specific policies.
- Include the end certificate in the request.
- Verify the DPV server's digital signature on the response.
- Process the server's response to determine if a valid path was constructed, or the failure code.

DPV servers must support the following requirements:

- Enforce client requirements for path validation (e.g., trust anchor and acceptable certificate policies) where specified in the request.
- Where client requirements for path validation are omitted, perform validation with respect to default policy requirements.
- Obtain all certificates, CRLs, and status information to build one or more paths to the designated trust anchor.
- Return complete certification path (i.e., all certificates, CRLs, and status information) or a transaction id that uniquely identifies the path.
- Perform path validation with respect to the client's security requirements (e.g., acceptable certificate policies).
- Generate and sign a response message.
- Maintain centralized auditing logs, including the following information for each transaction:
  - Identity of requester;
  - certificate of interest;
  - validation result (verification or denial)
  - certificate status information used in validation
  - where validation failed, the reason for denial
  - server system time/date.

6 Acronyms

CA	Certification Authority
CRL	Certificate Revocation List
DNS	Domain Name System
DPV	Delegated Path Validation
DSP	Directory System Protocol
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
LDAP	Lightweight Directory Access Protocol
NIST	National Institute of Standards and Technology
PKI	Public Key Infrastructure
PVM	Path Validation Module
URI	Uniform Resource Identifier

7 References

[1] Sharon Boeyen, Tim Howes, and Patrick Richard. Internet X.509 Public Key Infrastructure: *LDAPv2 Schema*, RFC 2587, June 1999.

[2] Matt Cooper, Yuriy Dzambasow, Peter Hesse, Susan Joseph, and Richard Nicholas. Internet X.509 Public Key Infrastructure: *Certification Path Building*, June 2004.

[3] Russell Housley and Paul Hoffman. Internet X.509 Public Key Infrastructure: *Operational Protocols: FTP and HTTP*, RFC 2585, May 1999.

[4] Russell Housley, Tim Polk, Warwick Ford, and David Solo. Internet Public Key Infrastructure: *X.509 Certificate and Certificate Revocation List (CRL) Profile*, RFC 3280, April 2002.

[5] Steve Lloyd. *Understanding Certification Path Construction*, September 2002.

[6] Tim Howes and Mark Smith. *The LDAP URL Format*, RFC 2255, December 1997.

[7] Tim Berners-Lee, Larry Masinter, and Mark McCahill. *Uniform Resource Locators (URL)*, RFC 1738, December 1994.

[8] National Archives and Records Administration. Records Management Guidance for Agencies Implementing Electronic Signature Technologies, October 2000.

[9] National Archives and Records Administration. Records Management Guidance for PKI-Unique Administrative Records, March 2003.

[10] National Institute of Standards and Technology. NIST Recommendation on X.509 Path Validation, Draft version 0.5, May 2004.

[11] Denis Pinkas and Russell Housley. *Delegated Path Validation and Delegated Path Discovery Protocol Requirements*, RFC 3379, September 2002.

[12] Tim Polk, Russell Housley, and Larry Bassham. Internet Public Key Infrastructure: *Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and CRL Profile*, RFC 3279, April 2002.

[13] Blake Ramsdell. S/MIME Version 3 Message Specification, RFC 2633, June 1999.