

Extended Validation Models in PKI: Alternatives and Implications

Marc Branchaud
RSA Security
Vancouver, BC, Canada
marcnarc@rsasecurity.com

John Linn
RSA Laboratories
Bedford, MA, USA
jlinn@rsasecurity.com

1. Introduction

The fundamental goal of PKIs is to provide a means for participating entities to establish and manage trust in other entities, either within or across domain boundaries. As PKIs have evolved, so has the set of alternate methods supporting validation of entities, their certificates, and their keys. Validation processing determines whether or not the acceptance of a certificate or key represents a suitable risk to a relying party. As such, it is a central and necessary basis to support reliance on PKI-based authentication.

Increasingly, PKI designers seek to distribute validation-related information and processing among cooperating components, reducing the complexity at individual relying parties. These techniques afford the potential for great power, but also imply fundamental shifts in the trust relationships among the entities involved within a PKI. In this paper, we examine traditional technology practice in the field, consider newly emerging alternatives and their characteristics, and look ahead to candidate future directions and their implications.

2. Existing PKI Practice

Early work in PKI definition culminated in Version 1 of CCITT Recommendation X.509 [CCIT88]. This work, initially instigated as a supporting mechanism to authenticate directory queries and responses, assumed the availability and use of a directory as a repository for certificates and Certificate Revocation Lists (CRLs). Use of certificates and CRLs, signed by responsible Certification Authorities (CAs), made it unnecessary for the directory to be strongly trusted for security purposes; no compromised directory could forge an entry that a verifier would accept. As a central premise, no CA or other security-relevant component operating as a CA's agent needed to be accessible on-line in order to support authentication with certificates once issued.

In the late 1980s and early 1990s, related work in the Internet community concerned usage of X.509 certificates to support e-mail protection, a project known as Privacy-Enhanced Mail (PEM) [Linn93] [Kent93]. PEM was designed to operate in environments common at that time, where communications facilities were often costly and confined to store-and-forward messaging. For these reasons, even fixed-site users often accessed their messages in an off-line mode. As a result of these factors, the PEM design placed a high priority on making messages self-contained for processing purposes. Facilities were designed so that certificate chains could be attached to messages and so that CRLs could be obtained via e-mail. Further, PEM contemplated a hierarchic certification model, within which it was straightforward for a sender to predetermine the path elements that would be acceptable to message recipients.

CRLs are the "traditional" and most widely standardized revocation facility for certificate-based infrastructures, and have an existing and growing installed base. Their evolution has continued as additional extensions have been defined, particularly to allow partitioning of CRL data across multiple objects in large-scale environments. They provide a means for propagating revocation information in a signed fashion, allowing its storage in directories and other stores that need not be trusted for security purposes. Nonetheless, CRLs have long been one of the most controversial components of PKI systems. Their disadvantages include limited timeliness, the need for widespread propagation of potentially large objects (within which most individual verifiers are interested in only a small portion of the content), and the fact that concise proofs of validity are not directly available for presentation along with certificates.

The IETF PKIX (Public Key Infrastructure using X.509) working group was established in 1995 and became the central forum for defining PKI facilities for use by Internet applications. In addition to profiling X.509 certificate and CRL constructs for Internet usage [Hous99], it also undertook the definition of the Online

Certificate Status Protocol (OCSP) [Myer99] as an alternative to CRLs, enabling responder servers to provide revocation information for certificates in the form of signed responses to per-certificate queries. CAs can explicitly delegate their revocation reporting to separate responders acting as their agents, representing this delegation with an extension in the certificates that they issue to the responders. Through this delegation, the CA private keys used for certificate signing can remain in off-line systems, though the responders' private keys must be accessible to sign responses to on-line queries.

The OCSP semantics were designed to enable migration from CRLs; OCSP responders can use CRLs as a source of revocation information or can be more directly coupled into CA databases. Like CRLs, OCSP responses are signed and can be saved along with validated messages to enable revalidation at a future point in time. OCSP queries determine whether one or more individual certificates have been revoked or are in suspension. It explicitly excludes validation of the certificate's signature, timeliness, or path-level validation, though extensibility for additional validation services is possible within the protocol framework. OCSP requestors must perform path-level processing, as the public key of a certificate issuer is required in order to construct a status query for a certificate. Variant and alternative protocols have been proposed with broader functionality, and will be discussed next.

3. Delegated Validation

The IETF PKIX working group is currently developing requirements and examining proposals for delegating certification path construction and validation from a relying party's client to a server. This work is at least partially motivated by the success of OCSP, which has seen wide adoption because it allows applications to offload the task of determining a certificate's status. Similarly, delegated path processing is seen as a way to offload the tricky and onerous work of discovering and validating certification paths.

PKIX is exploring two aspects of server-side path processing. In the first, Delegated Path Discovery (DPD), a server is given a target end-entity certificate and one or more trust anchors. The server then sets about discovering candidate paths between the trust anchor(s) and the target certificate. These certification paths are returned to the client, which will then set about validating the chains using the regular X.509/PKIX rules, including checking the status of each certificate, using CRLs or OCSP. In this model, the client need not trust the DPD server, as the client performs all cryptographic, status and path validation itself.

The second aspect, Delegated Path Validation (DPV), is more interesting from a trust management perspective. With DPV, the client expects the server to not only discover candidate certification paths but to also completely validate and status-check these paths. The client is essentially looking for a Boolean response as to whether or not it can accept the target end-entity certificate. The client must completely trust its DPV server, as it abdicates to the server all responsibility for determining acceptable paths. Even where the client provides the server with its own policy inputs or trust anchors, such data are primarily advisory as the client can neither control nor examine the server's internal processing.

It is this complete delegation of responsibility that fundamentally distinguishes DPV from DPD and OCSP. The delegation model of the latter two protocols expects their clients to intelligently participate in the PKI, and so their servers can only return copies of CA-sourced information. OCSP and DPD servers must operate within the confines of that information, and are essentially mechanistic publishing and data-gathering tools. The clients will verify that their results come from an authoritative source. On the other hand, DPV servers are free to draw on other sources of information in order to synthesize appropriate responses. DPV servers can act with much more autonomy, as their clients have no way to verify the "correctness" of their behavior.

A number of advantages are gained when clients are willing to abdicate their responsibility and control to follow the DPV model. Not the least of these is a radical simplification of the client applications. Clients can do away with path validation and discovery code, which may include support for multiple protocols and algorithms. In the limit, clients need not even be able to parse a certificate's contents in order to use some DPV services.

Another advantage of the DPV delegation model is that it provides domain administrators with a convenient central point to manage inter-organizational trust on behalf of the domain's relying parties. Trust relationships can be switched off and on for all or part of a domain according to whatever criteria the domain's policies dictate, and without the need for any reconfiguration or even notification of the domain's clients. This property alone may motivate enough organizations to deploy DPV, enabling rapid, widespread adoption.

It is necessary to recognize, however, that reliance on DPV implies a dependency on on-line service availability in order to perform validation. Further, aggregation of queries within a DPV server may present a convenient point for privacy-relevant data about client requests to be collected. Nonetheless, DPV's operational

characteristics relative to off-line validation are motivating growing interest, and broad adoption appears likely.

The DPV delegation model turns each DPV server into a trust anchor for its clients. This distributes the trust anchors throughout the PKI, which in turn reduces the number of clients that must be reconfigured following the compromise of any particular trust anchor. This effect can also be achieved without DPV, through the deployment of general CA hierarchies where (as opposed to top-down hierarchies), each CA certifies its parent as well as its children. In a general hierarchy any CA can act as a trust anchor that provides connectivity with the entire PKI, and the compromise of any CA will only require reconfiguration of its trusting clients as well as its parent and child CAs. We note, however, that such general CA hierarchies have proven to be rare in practice, whereas the DPV model expresses this property naturally.

Two candidate protocols have so far emerged to support DPV-style services. These are the Simple Certificate Validation Protocol (SCVP) [Malp01] and the XML Key Management Specification (XKMS) [W3C01].¹ XKMS in fact provides more than just delegated validation services; however, this section will focus on the DPV-style services defined in “Tier 2” of the XKMS XML Key Information Service Specification (X-KISS). We note in passing that extensions to OCSP to support a DPV service have been proposed. As these are not materially different from the services proposed in SCVP, we do not include the OCSP extensions in our analysis.

SCVP and Tier 2 of XKMS (hereafter referred to simply as XKMS) have significant syntactical differences, both in terms of each protocol’s messages and in terms of how a client specifies the “certificate” to be validated. SCVP is an ASN.1 protocol that exclusively supports X.509 certificates, while XKMS is an XML protocol that supports X.509, PGP and SPKI certificates. XKMS can also identify keys by reference (URI) or even a simple name. (The relative merits of ASN.1 or XML are beyond the scope of this paper, and are irrelevant to delegated validation.)

Fundamentally, both protocols enable a client to delegate validation operations to a server. Each allows the client to request various types of assertions from the server regarding the certificate in the query. SCVP’s focus on X.509 limits these to certification paths and assertions of certificate revocation status, while XKMS

allows clients to also request raw public key values (on the assumption that the server has validated the keys according to the appropriate criteria). Both protocols allow the client to also request various forms of evidence to support the assertion in the response, including certification paths and revocation status information.

In delegating validation operations, the client in either protocol trusts that the server will act appropriately on its behalf. Neither protocol explicitly defines server behavior, although both imply that the server should perform the operations that a non-DPV client would in order to validate the certificate (e.g. perform PKIX-style certificate path discovery and validation when queried about an X.509 certificate). As with client-based validation, extensions such as nameConstraints can be incorporated in cross-certified paths in order to limit the transitivity of trust. SCVP allows the client to provide some input into the server’s processing, such as policy identifiers, trusted CAs and certificate revocation information. SCVP clients can also specify a “configuration identifier” to, for example, inform the server of the context of the client’s query.

XKMS does not allow its client to provide similar information. This omission may not be as glaring as it appears, for we note that in a delegated validation system clients will not themselves validate the evidence included in a response (though they will authenticate the response itself). Thus the server is free to ignore the client’s ancillary inputs, or it can tailor the evidence in its response to match those inputs, and the client is not in any position to detect such manipulation. (If the client were able or willing to verify the evidence itself, it would not really need a delegated *validation* service.) This is what we mean when we say that a DPV client must trust its server. Any ancillary input in a request can be ignored, and any evidence in a response is mainly useful to third parties auditing the response. XKMS, as it lacks facilities for clients to provide ancillary input, merely makes these facts more explicit.

This need to completely trust DPV servers does not mean that delegated validation cannot be a good and useful technology. It does, however, change the nature of PKIs in ways that will be explored in the rest of this paper. For now, we close this section with the notion that some ancillary input can still play an important role in a DPV system. Although information such as trusted CAs and revocation information might help a server, we expect that servers will typically have their own resources for obtaining such information, and will be configured with the trusted authorities of the clients they serve. What will be truly useful is an indication of the client context. This would allow the server to perform different levels of validation depending on whether, for example, the client is merely reading e-

¹ The specifications for both SCVP and XKMS are still in draft form and subject to change. The analysis presented here considers the protocols as specified in January 2002.

mail or is performing an expensive purchase. Such a context indicator could in fact instruct the server to employ different trusted authorities and/or different path discovery and processing rules. All this could be triggered by a single identifier communicated from the client to the server, the values of which could be standardized or determined by private agreement.

4. Recursive and Chained Validation

Current standardization initiatives [Pink01] emphasize use of DPV between clients and their associated DPV servers, but it is possible to envision extending the paradigm so that DPV servers themselves consult other DPV servers to perform validation. A generalized PKI structure could incorporate four tiers of elements generating or processing validation data: issuing CAs, their OCSP responders, DPV servers trusted by relying parties (RPs), and the RPs themselves. If and as use of DPV becomes common, DPV-based queries might even be extended back to DPV services associated with CAs. In this fashion, DPV could be applied as a means to publish CA status information, eventually eliminating the OCSP tier. Potentially, use of CA-provided DPV could make non-DPV status checking facilities moot and could reduce the number of protocols required within an overall PKI. It remains likely, however, for DPV to remain under RPs' jurisdiction, querying OCSP responders maintained on behalf of issuing CAs. Independent of whether CAs export their status information via CRLs, OCSP, or DPV, it is possible for multiple layers of DPV interaction to be interposed between an RP and the information that an issuing CA provides for one of its certificates. This section discusses alternatives and issues arising in such *inter-server delegated* models.

When large-scale PKIs combine OCSP and DPV, trust may be delegated in two directions: from CAs, via their OCSP responders, and from relying parties, via hierarchies of DPV servers, each invoking the other's services. Borrowing the metaphor of a weather chart, a path extending from a relying party to a remote issuer CA can be thought of as crossing a "trust front", delimiting zones of responsibility affiliated with the relying party from zones affiliated with the issuer. In the general case, path validation requires information provided both from the issuing CA domain (i.e., certificates and their associated revocation information) and from the RP domain (to reflect its trust relationships and policies). When delegations to responders or validation servers extend from either domain, the delegating domain must trust its delegates to represent its interests, accurately process data obtained from other domains, and reflect the data that it's authoritative to provide.

In a DPV environment, each RP will normally focus its validation requests on a single trusted DPV server (although it may be replicated to ensure availability), which will be responsible for validating any and all certificates that its RPs receive. A given DPV server may select to delegate among a set of other DPV servers, depending on the particular certificate being queried and on the policies under which validation is being performed. If different clients use different DPV server paths to validate a particular certificate, it is possible that the information they receive via different sources (and their associated paths) may yield different results. This prospect arises whenever an RP obtains status information indirectly via active intermediaries, rather than by accessing the information directly from its source.

We distinguish three forms of interaction that can take place among delegated validation servers:

- *Chained* queries, within which a specific server is recognized as authoritative to respond to a query about a particular certificate. All queries about that certificate that are received by other servers are forwarded to the authoritative server, and the information obtained by a requesting server can remain in a form that is traceable to the authoritative domain.
- *Referred* queries, which resemble chained queries except that the requesting server is redirected to the authoritative server rather than having the query mediated on the requestor's behalf by the server that initially received it. As with chained queries, the information obtained by a requesting server can remain in a form that is traceable to the authoritative domain.
- *Recursive* queries, where each server aggregates information obtained from other servers in order to respond to the queries it receives, but where a requestor must directly and fully trust the delegated validation server that it contacts as the provider for information about a set of domains. In this form of interaction, intermediaries distill the information issued by authoritative sources and actively integrate it with data maintained in their own databases, rather than transferring and processing it in a fashion that keeps it independently verifiable by relying parties.

Different DPV servers may employ different query models, satisfying different goals and constraints; further, a given DPV server may employ different models depending on the domain associated with a particular certificate that is to be validated. Of the models, recur-

sive queries imply the greatest level of trust delegation from the requesting server to the target, as they move the critical operations of aggregation and synthesis of validation data to the target. This reduces the amount of data that needs to be propagated to requestors, typically making protocol exchanges simpler and more compact. Analogous to end-entity client use of DPV, recursive inter-server DPV interactions imply that requesting DPV servers place fundamental trust in the target DPV servers to which they direct their queries. In a recursive DPV environment, multiple DPV services, operated by different domains, may be involved in determining the validity of a certificate. The domains responsible for operating DPV services may not directly correspond to the domains responsible for individual certification path elements, though validation policies could be applied to enforce such a constraint.

Referred and chained queries, in contrast, collect validation data for integration and processing by the requesting entity, whether an RP or a DPV server; as such, they transfer a larger volume of information for requestor processing. These approaches can enable independent auditability, can contribute to post-facto revalidation for non-repudiation purposes, and can help DPV servers to partition the impact of compromised data originating from particular sources. If signatures are applied and retained on the received data, requestors can preserve records of the validation data they obtain in a form that is provably traceable to authoritative issuers. It appears unlikely that many RPs will commonly revalidate determinations made by their DPV servers, unless on an exception or post-facto basis, as RPs prepared to perform this processing would gain relatively little benefit from using DPV. They would need to obtain (either through DPV itself or via out-of-band means) trusted keys with which to verify the signatures of remote servers, and to validate that those servers were authoritative to report status on behalf of specific domains. Even within a model where individual RPs elect to delegate trust to DPV servers, it may still be desired for those DPV servers to maintain validation data records on behalf of their domains.

One fundamental design choice for PKI designs and deployments is as follows:

- Is the set of certificates that a relying party can validate intentionally confined to those domains with which it (and/or its DPV server) has established direct working relationships; or
- Is broader validation sought as a goal?

If the scope of certificates to be validated can be constrained in advance, directly established relationships between individual DPV servers may suffice to support the RP community of interest. If universal vali-

dation is desired, however, DPV servers must be able to take recourse to a general certification hierarchy or mesh as a means to obtain trust connectivity among one another.

Recursive DPV distributes knowledge of suitable paths and sources for validation data among the set of DPV servers rather than collecting it within each domain that requests that certificates be validated. In a recursive model, individual entities need not perform end-to-end path discovery. This decentralization may prove valuable in enabling interconnected PKIs to scale to very large sizes. The distributed nature of recursive DPV echoes and accentuates a basic DPV characteristic; not only are its RPs unable to independently revalidate processing performed within their own DPV servers, they also lack independent assurance of the quality of information on which their servers depend. Overall, recursive DPV offers significant power and flexibility, but also implies significant growth in the set of on-line components comprising a distributed trusted computing base for certificate validation. Chained and referred DPV models distribute more data and processing complexity among participating components, but allow the participants to operate with a higher level of mutual suspicion among one another.

5. Implications and Future Directions

There are profound implications that arise when using inter-server delegated validation models such as those described in the previous section. Ostensibly, DPV services perform two distinct functions: certification path discovery and verification, and certificate status confirmation. The latter is necessary because of the nature of certificates. Specifically, a certificate conveys a binding that its CA believed to be true when the certificate was issued. However, typical certificate users often need to know if the CA still believes the binding to be appropriate when the certificate is used, rather than it was issued. Hence the desire to check a certificate's status.

An obvious effect of inter-server delegated DPV is that it eliminates the need for CRLs. This is because such a system constructs paths between domains online, through the involvement of each intermediate domain's servers. Since each domain is making an active assertion about which pathways are valid, and each includes certificate status as one of its validity criteria, separate mechanisms for obtaining certificate status are no longer necessary.

Active domain participation can be leveraged even further. For example, it can change the way that inter-domain trust is established and managed. This is currently achieved with cross-certification, but if a domain

is going to make an active statement about the status of its cross-certificates, it may as well instead make a statement about the relationships the cross-certificates represent. With active participation, domains need no longer rely on cross-certificates as a source of policy information, but can instead manage this data locally within their DPV servers. This allows for much finer control of the inter-domain relationship. At the coarsest level, the relationship can be present for some clients but not for others, or only at particular times. A more sophisticated application would enable the relationship under certain contexts, but not others. This would be the equivalent of having multiple cross-certificates between two domains that are issued under different policies, something that is theoretically possible but has yet to be seen in practice. The central administration benefits of DPV servers make practical the application of multiple policies to a relationship with another domain.

The effects of active domain participation can be felt even further. Consider that in a fully delegated DPV environment queries about a particular certificate eventually reach the issuer of that certificate, or an entity designated by that issuer to speak authoritatively on its behalf. This is necessary to obtain the status of the certificate. However, if the issuer of the certificate (or its designate) is making an active statement about the certificate's status, it could just as easily take advantage of the situation to make statements about the certificate's actual contents. In the limit, the certificate's issuer could, in response to a query, return certificate contents as separate elements rather than in the form of a certificate. So, for example, if the name of the certificate's subject has changed since the certificate was issued, the issuer could return the new name in its DPV response. The issuer could also return the subject's current public key, which would altogether eliminate the need to publish revocation information.

Taking this notion to its extreme conclusion leads us to consider the elimination of certificates entirely. Rather than obtaining a certificate, entities enrolling in a PKI could register their public key with an authority, which would give them an identifier of some sort. This identifier could then be presented, as is possible in XKMS, instead of a certificate in any challenge-response, key establishment, or digital signature verification protocol. The receiving party would submit the identifier to its local DPV server, which would resolve it (by eventually querying the authority that issued the identifier) into the registered public key that could be used to complete the protocol.

It is the online, active nature of inter-server delegated DPV that makes this possible. The ultimate scenario described above may or may not be practical, or even desirable, but it does highlight the fundamental

shift that DPV services create in PKI. The original framers of X.509 did not contemplate domains making active statements about their certificates. Indeed, any online components (i.e. the X.500 directory) were specifically not trusted. DPV heralds a departure from that philosophy, and it will have profound effects on the very infrastructure itself.

6. Conclusions

On-line validation methods for PKI certificates are attracting increasing interest and adoption. Current standards directions emphasize the simplification of client-side processing. Two aspects are fundamental:

- Reduced volume of validation support data propagated to clients; and
- Reduced complexity of validation processing within clients.

In order to satisfy these goals, clients must delegate their trust to new services, relying on those services to perform validation for them. Rather than being eliminated, validation complexity will move to a new set of distributed components, operating as active intermediaries. When this delegation is performed, many "traditional" PKI assumptions will no longer hold, as new trusted points become active peer participants within distributed PKI-based architectures.

All on-line validation strategies can improve upon CRLs' schedule-driven revocation responsiveness, if their underlying information sources permit; DPV introduces the possibility of a different sort of latency as data is aggregated at intermediaries. Approaches differ significantly, however, in the scope of active components that must be trusted for validation purposes, and in the trusted properties that they must provide. As adoption of delegated validation proceeds, facilities to constrain these trust characteristics, preserving appropriate levels of mutual suspicion, are likely to be important.

Finally, delegated validation services represent a fundamental change in the assumptions that underlie a PKI. Most significantly, domain authorities can become active participants in the PKI, interacting dynamically with relying parties rather than merely making assertions at particular points in time. This has profound effects on the nature of PKI technology, leading to questions about the explicit need for revocation, and even about the nature of certification itself.

Acknowledgments

The authors thank the workshop's anonymous reviewers and Russ Housley of RSA Laboratories for their comments on this paper.

References

- [CCIT88] CCITT Recommendation X.509 (1988), "The Directory - Authentication Framework".
- [Hous99] R. Housley, W. Ford, W. Polk, and D. Solo, "Internet X.509 Public Key Infrastructure: Certificate and CRL Profile", Internet RFC-2459, January 1999, <http://www.ietf.org/rfc/rfc2459.txt>.
- [Kent93] S. Kent, "Privacy Enhancement for Internet Electronic Mail: Part II: Certificate-Based Key Management", Internet RFC-1422, February 1993, <http://www.ietf.org/rfc/rfc1422.txt>.
- [Linn93] J. Linn, "Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures", Internet RFC-1421, February 1993, <http://www.ietf.org/rfc/rfc1421.txt>.
- [Malp01] A. Malpani, P. Hoffman, R. Housley, T. Freeman, "Simple Certificate Validation Protocol (SCVP)", Internet draft work in progress, IETF PKIX working group, July 2001.
- [Myer99] M. Myers, R. Ankney, A. Malpani, S. Galperin, C. Adams, "X.509 Public Key Infrastructure: Online Certificate Status Protocol - OCSP", Internet RFC-2560, June 1999, <http://www.ietf.org/rfc/rfc2560.txt>.
- [Pink01] D. Pinkas, "Delegated Path Validation and Delegated Path Discovery Protocol Requirements", Internet draft work in progress, IETF PKIX working group, November 2001.
- [W3C01] P. Hallam-Baker, editor, "XML Key Management Specification (XKMS)", W3C Note, March 2001, <http://www.w3.org/TR/xkms/>.