ECPV: EFFICIENT CERTIFICATE PATH VALIDATION IN PUBLIC-KEY INFRASTRUCTURE

M. Halappanavar and R. Mukkamala
Department of Computer Science, Old Dominion University, Norfolk, VA, USA

Abstract In the current public-key infrastructure (PKI) schemes based on X.509, a relying party must validate a user's certificate as well as the existence of a path from its trust points to the CA of the certificate. The latter part is referred to as certificate path validation. In this paper, we suggest an efficient certificate path validation scheme (ECPV) that employs delegation with efficient computing at relying parties. In particular, in our scheme, a relying party is provided with certificate path validation trees (CPVTs) depending on its trust points and applicable trust policies. This information should enable a relying party to perform certificate path validation locally. The CPVAs can be deployed either as autonomous entities or in a federated mode. We discuss the two major components of ECPV: the data harvester and the data analyzer. Some of the concerns of security, trust, and performance are also discussed.

Keywords: Certificate Path Validation Authority (CPVA), certificate path validation trees (CPVT), delegated path discovery (DPD), delegated path validation (DPV), public-key Infrastructure (PKI), security and trust

1. INTRODUCTION

Digital certificates have emerged as a popular tool to provide authentication, privacy, integrity, non-repudiation and other security requirements in modern day transactions [1]. PKI provides a comprehensive infrastructure for issuing and managing public keys in the form of digital certificates to a set of users [1,10]. When two users want to communicate with one another in a secure manner, they will have to authenticate each other prior to the actual communication. This stranger-to-stranger communication is made possible by having a trusted third-party (TTP) in the form of Certification Authority (CA) whom both the parties trust. When a user or subject submits a certificate to a relying party (RP) for a service, the relying party would like to validate the user certificate prior to offering a service.

Typically, the validation of a certificate is done in three steps. In the first step, a relying party (RP) builds a chain of trust, also known as certificate chain, starting from the CA that has issued the user's certificate to the CA that RP trusts (also known as a trust point). This step is known as *certificate path discovery.* Generally, trust between CAs is established by issuing CA-CA cross-certificates. These certificates can also be revoked from time to time. Thus there is a need to confirm the validity of the links in the certificate chain built in step one. The second step conducts the validation, and is referred to as *certificate path validation.* (The logic for the validation process described above is described in RFC 2459 [9,10] and is based on various parameters specified in certificate policies.) In the third step, the validity of the issued certificate is itself verified with information provided by its CA. In this paper, we are interested in the first two steps of the validation process.

Recent research trends in public-key infrastructure favor delegation of authority for certificate path validation to trusted-severs and for path discovery to untrusted-servers. DPV (Delegated Path Validation) and DPD (Delegated Path Discovery) are examples of such protocols [9]. These protocols are still evolving and are described in Internet drafts [8,9]. Simple Certificate Validation Protocol (SCVP) is another delegation protocol that is conceptually similar to DPV and DPD [6].

In this paper, we propose an efficient certificate path validation (ECPV) scheme to overcome the weaknesses in the current path validation protocols. Our scheme is based on one or more certificate path validation authorities (CPVA) that upload RPs with a certificate path validation trees (CPVT). An RP, in turn, can use this locally available tree to validate certificate paths of user certificates. The trees shall be periodically updated by the CPVA. The CPVAs can be implemented either as an autonomous entity working in isolation or organized as a federation to share information.

The paper is organized as follows. In section 2, we discuss some of the issues in certificate path validation and current solutions. In section 3, we describe the proposed ECPV scheme for certificate path validation. A more detailed description of the CPVA module is provided in section 4. Two possible options for deployment are discussed in section 5. Section 6 reviews the security, trust, and performance aspects of the proposed scheme. Finally, section 7 summarizes the contributions of the paper and discusses plans for future work.

2. ISSUES IN CERTIFICATE PATH VALIDATION

As explained in section 1, certificate path validation involves the task of discovering a path and then verifying its validity. The path discovery is both computation and communication intensive, since it involves discovering a network of CAs that trust each other. Several factors contribute to the complexity. First, there are many possible trust topologies [5,8,9,11]. Strict hierarchical, multi-rooted hierarchical, mesh, bilateral cross-certification, and bridge topologies are a few example topologies. However, many of the present RP implementations make simple assumptions about the trust structure and are not capable of handling these complex trust relationships [11,14]. Changes in the trust hierarchies over time also necessitate changes in the relying party software, an impractical proposition.

Second is the trust policy issue. Each RP has an acceptable trust policy for the chain of CAs that it is trying to validate for a given certificate [3]. The policy would describe the minimum level of CA practices that it expects from each of the CAs on the path. Similarly, each CA also specifies, in its CA-CA certificate, the practices (policies) that it trusts other CAs to follow [1,2]. As a consequence of a wide variety of such policies, the task of RP becomes more complex during path discovery since it should only select those links that are compatible with its (RP's) own policy. Checking the policy compatibility is not a simple process [2,7,9].

Third, response time is also one of issues for certificate path validation. For example, an RP is very much interested in responding to its users as soon as possible with the service they requested. However, it cannot do so until the path validation and the subsequent certificate validation are complete.

Fourth, CA and repository availability is also an important concern to an RP. All of the CAs and the relevant certificate and revocation information repositories need to be available for an RP to discover and validate a path. If a trust path is long, with several CAs, there could be a problem in data collection as some entities along the path may not be accessible during validation [9,11,14]. In other words, even though a CA-CA certificate is valid, due to the inability to access a CA's data at the validation time, an RP may not be able to validate a certificate.

Finally, RP is concerned about the cost of validation. It would like to minimize this cost. When each RP undertakes the task of discovery and validation (i.e., if it is not delegated to a server), it is an expensive process. Substantial cost savings may not be realized if the entire process of validation is repeated for each certificate that the RP receives, even when delegated to a trusted validation server.

From the above discussion, it is clear that certificate validation is a complex process involving considerable amount of communication and computational resources.

3. ECPV: PROPOSED SCHEME FOR CERTIFICATE PATH VALIDATION

ECPV is based on the idea that it is much more efficient to provide a relying party with path validation information in the form of trees (CPVT) rather than to require the RP to request an external server for each certificate validation. Section 3.1 describes the CPVTs in greater detail.

3.1 Certificate Path Validation Trees (CPVT)

To explain the concept of CPVT, let us consider the example in Figure 1. Figure 1 (a) describes a set of trust relationships between a set of CAs (CA1 - CA9). Here, an arrow between one CA and the other (e.g., from CA1 to CA5) indicates that one CA has issued a cross-CA certificate to the other (e.g., CA1 issued a certificate to CA5). The arrow type indicates the similarity between the trust parameters, which could be established by policy mappings (e.g., CA1 policy OID_x maps to policy OID_y for CA5).
Suppose a relying party RP1 trusts CA1. In addition, it accepts input parameter set 1 for some type of requests and input parameter set 3 for other types of user requests. In this case, the intent is to build two trees for RP1 with CA1 as the root, one with IP set 1 and the other with IP set 3. Figures (1b) and (1c) are the required trees derived from Figure 1a. These are what we refer to as certificate path validation trees or CPVTs. For RP1 to accept a certificate issued by CAx under IP set 1, for example, it expects a path from CA1 to that CAx in the CPVT in Figure (1b). In this case, for example, only CA1, CA3, CA7, and CA9 meet the criteria. In fact, each directional edge in Figure 1 is associated with an expiration time (not shown in the figure), derived from the expiration time of the cross-CA certificate and the CRL (or ARL) issued by the parent CA. Thus, an edge in the network or CPVT is valid only if it has not expired at the time of its usage. The advantages of building the path from trust point, commonly referred to as building in *reverse direction,* are very well discussed in [7].

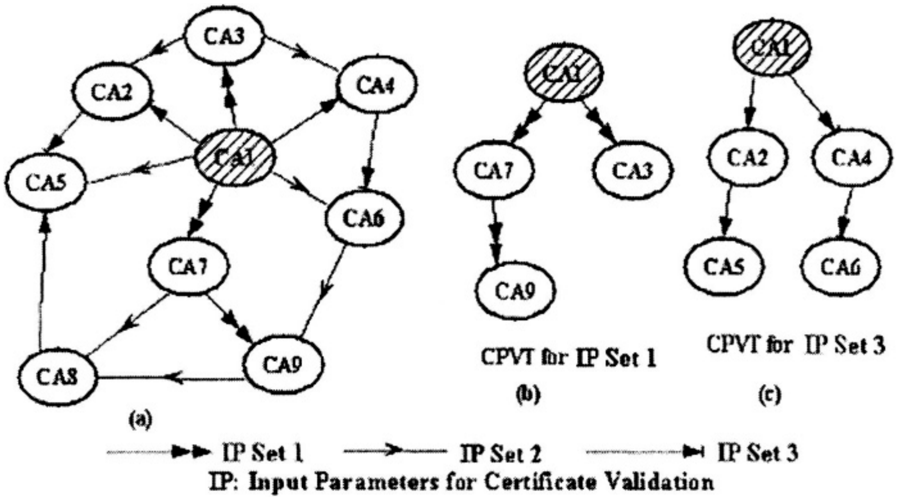

Figure 1. Illustration of Certificate Path Validation Trees (CPVT)

3.2 System architecture

The system architecture for ECPV is shown in Figure 2. The various components of the system are as follows:

(1) *CA Module:* The certification authority (CA), its registration authority (RA), the repository (REP), and OCSP responder together represent the CA module.

(2) *CPVA module:* The CPVA (Certificate Path Validation Authority) is the critical module that will perform the process of path discovery and verification. This module consists of harvester (HRV) and analyzer. CPVA is responsible for building the CPVTs and distributing them to the RP modules.

(3) *RP module:* The relying parties (RP) service the requests from subjects for certificate validation. The relying parties provide requisite information, including trust points and validation policies, to the CPVA module, which builds relevant CPVTs based on the input and download CPVTS to the RPs. RPs are the primary users of the infrastructure for validation of certificates.

(4) *Subject module:* Subjects or clients request service from the RPs and offer their certificates for validation. The subject's influence the system design and performance from their diversity and rate of requests.

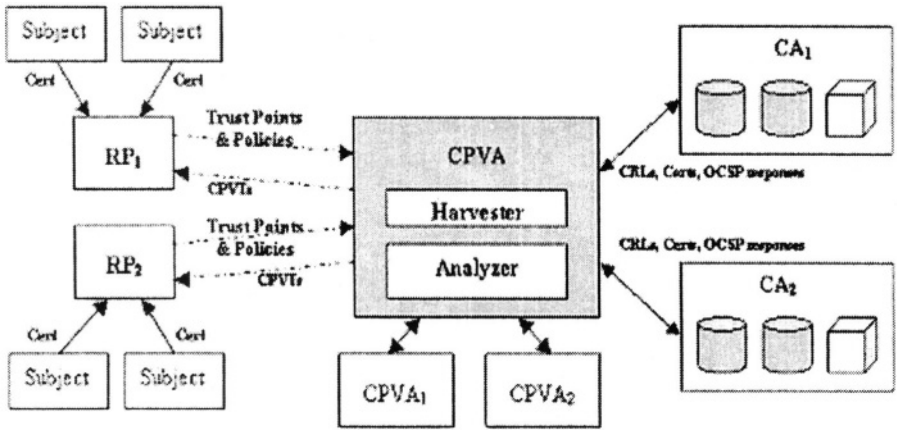

Figure 2. System Architecture

Another important aspect of our architecture is the interactions among the modules. Among all interactions in our system, the following three are relevant with respect to this paper: (i) the RP-CPVA interactions, (ii) CPVA-CA interactions, and (iii) CPVA-CPVA interactions. We summarize these three types of interactions in the form of scenario-like description of events starting from an RP submitting a request to the uploading of CPVTs to it.

1. Each relying party (RP) submits a set of input parameters with vital information like trust point CAs, policy constraints, name constraints, maximum path length, etc.
2. The CPVA consolidates the RP information and requests its harvester component to collect data a set of <trust point, input parameter set> pairs.
3. Harvester collects all the requisite information including CA certificates, CRLs, OCSP responses, etc. (In the federated scheme, a harvester may also collect CPVTs from other CPVAs).
4. The analyzer analyzes the collected data, and builds CPVTs based on the RPs' requests initially received. The analysis includes looking into the CRLs (also referred to as ARLs [1,10]) and the OCSP responses for validity of CA certificates. In addition, it needs to take a note of the validity of each response---how long is a published CRL applicable. Using this information, it builds CPVTs with trust points as the roots.
5. The CPVA uploads the appropriate trees to the respective RPs (i.e., each RP gets only the relevant CPVTs).
6. Since some edges of a CPVT may expire after the initial upload, the trees automatically get pruned at the RP while being used for certificate

verification. In addition, the CPVA also keeps track of the expirations and updates its trees with periodic data. The updated trees may be either automatically uploaded (in a push scheme) or requested by a RP (in a pull scheme).

4. THE CERTIFICATE PATH VALIDATION AUTHORITY (CPVA)

As shown in Figure 2, CPVA has two components: harvester and analyzer. While the claimed simplicity of RP software comes from the harvester's functionality, the benefits in performance are due to both the components. We now describe these components.

4.1 The harvester

The concept of harvester is adopted from the concept of data harvesting in digital libraries [14]. A harvester traverses the CA repositories and OCSP responders and collects all the requisite information like certificates, CRLs, and OCSP responses.

The repositories use multiple directory protocols like DAP, LDAP, FTP and HTTP, and also have directory chaining, directory referrals. This necessitates the harvester software to be complex and capable of communicating with different protocols. The components of the harvester, as follows:

- *Bulk Harvester:* The bulk harvester is activated for gathering of information at initialization as well as occasionally to refresh data.
- *Incremental Harvester:* The incremental harvester is executed frequently to keep the data current. It can be activated on events like certificate expiry, CRL expiry, addition of new CAs, etc.
- *The Scheduler:* The Scheduler components triggers the incremental harvester and the bulk harvester at appropriate times to keep the data up-to-date.
- *Data Normalizer:* Since the collected data during the harvesting can be in different formats, the data normalizer reformats the data into single format for easy processing by the data analyzer module.

4.2 The analyzer

Once the harvester has obtained all the relevant information, it is the responsibility of the analyzer to construct the CPVTs to upload to the relying parties. In addition, the analyzer is responsible for detecting any changes in the current CPVTs that it may have previously uploaded to RPs. For example, in a push model where the CPVA is responsible for updating the CPVTs at the RPs, the data analyzer should be able to upload the changes (or the new CPVTs) to the affected RPs. Since data analyzer is mainly responsible for constructing the CPVTs for the relying parties, the next task is to generate CPVTs from the mesh. There are several options in building the trees.

The analyzer has the following four components.

- *The Client Processor:* The Client processor consolidates the data received from the various RPs that the CPVA serves.
- *The CPVT Builder:* The builder performs the primary action of building the CPVTs based on the information obtained from the Harvester and other CPVAs. The Builder builds the trees for different trust points according to particular input parameter set.
- *Harvester Coordinator:* This component interacts with the harvester to specify the information that the harvester needs to collect and feed the Scheduler with appropriate times for re-harvesting the data.
- *Federation Coordinator:* As shall be discussed in section 5, the CPVA scheme can be deployed in autonomous mode or in a federated mode. When the federated model is used, the data analyzer will collect CPVTs from other CPVAs and use them to build the CPVTs.

5. DEPLOYMENT OPTIONS WITH ECPV

The ECPV scheme with CPVAs as primary components can be deployed in at least two ways: autonomous and federated. In the autonomous mode, the CPVA shall work in isolation to collect the data, to build the CPVTs, and serve only its clients. Certainly, this scheme is not highly scalable and does not offer high performance benefits in information processing. These disadvantages can be overcome by the federated mode of deployment where CPVAs would coordinate with another to build and to maintain the CPVTs in a highly scalable and distributed manner.

5.1 Autonomous harvesting

When RPs handle certificates from a few domains, and the number of CAs are limited, autonomous harvesting works very well. In this method, each CPVA module is independent (of other CPVAs). Its harvester is responsible for contacting the individual CAs/OCSP responders/directory repositories to obtain the needed certificates, revocation lists, and certificate status responses. CPVA module is responsible for several RP's (say in its domain). It is responsible for harvesting data from CAs related to requests from its RPs.

5.2 Federated harvesting

Once PKI technology becomes popular among the internet users, there would be thousands of CAs issuing certificates for their own specific domains. In other words, unlike a handful of CAs we currently have (e.g., Verisign, Entrust, etc.), there would be several autonomous ones issuing and managing certificates. In such an environment, it would not be possible for a single harvester to obtain information about all CAs, their CRLs, and the trust paths connecting them.

6. SECURITY, TRUST, AND PERFORMANCE ISSUES

As stated already, the primary impetus for our work is two-fold: to simplify the task of relying parties by delegating some of the data collection and processing work to CPVAs; and to arrive at a scheme that is efficient and scalable. These objectives are to be achieved without compromising the security and trust offered by the traditional certificate path validation schemes. In this section, we briefly discuss our scheme in the context of security, trust and performance.

6.1 Security and Trust Aspects of ECPV schemes

One of the primary concerns of a relying party in adopting the ECPV for path discovery and validation is of trust. In other words, why should a relying party trust that the information provided by CPVA is correct? This concern exists even with the current delegated path discovery and validation protocols [8,9]. But let us look at our schemes.

Under autonomous deployment, where the CPVA collects all the needed information (e.g., CRLs, expiration times, OCSP responses, etc.) by itself and prepares customized CPVTs for each relying party, the solution is simple. A CPVA signs the provided CPVT. Thus, there is a guarantee that an impostor process has not supplied incorrect results. This also ensures non-repudiation. In fact, to encourage more RPs to use its service, a CPVA may offer some kind of a guarantee as is done by Digital Signature Trust (http://www.digsigtrust.com) in the case of public-key certificates.

In addition, optionally, it provides all the information used to arrive at the CPVT. Since the information is itself signed by the providing entities (e.g., a CA digitally signs the CRL that it issues, and an OCSP responder signs its response), the evidence is self-explanatory and is trusted.

Under federated deployment, a CPVA that provides the CPVT to a relying party is not necessarily the one that directly contacted the corresponding CAs. Instead, it may have received either the raw information or part of the trees from other CPVAs. So it appears that the trust is somewhat weakened. However, if each CPVA signs its response (e.g., partial trees) or simply forwards the CA signed raw information to its requester, then the concern is mitigated. For example, if each CPVA, when it receives a signed partial CPVT from another CPVA, checks the signature, prepares a new CPVT (if it merged several partial CPVTs into one), and digitally signs it, then there would be no problem of trust. The idea of providing guarantees (as in Digital Signature Trust discussed above) by each CPVA further enhances the trust aspects of using the CPVA servers. If the RP does not trust the CPVA, it can receive all the supporting data along with the CPVT and verify, this process is considerably faster than RP having to perform the path discovery process all by itself.

6.2 Performance aspects of CPVA schemes

Let us consider CPVA and its impact on the relying parties. Clearly, a relying party is simplified due to the delegation of data collection and path construction to CPVA. This is similar to SCVP and DPD/DPV [8,9]. But CPVA significantly differs from other schemes in the information it provides to the RPs. While other schemes only provide the status information (yes/no/error/don't know) to an RP's request, CPVA empowers a relying party with validation trees that could be used by the RP to locally verify users' certificates. In other words, the response time, or latency, of an RP is greatly improved due to CPVA. In addition, by providing policy-specific CPVTs, an RP is able handle requests with varied values. For example, if a request is for a low-value service, it could use a CPVT of a weaker trust assurance. On the other hand, for a high valued transaction it could use a CPVT with a strong trust assurance. In summary, by caching the CPVTs, the performance of the RPs is greatly improved. In addition, the CPVA can handle much larger number of RPs since it receives requests from RPs less frequently.

Now let us look at the comparison between autonomous and federated schemes. Clearly, the load on CPVA is much larger in the autonomous scheme since its harvester is responsible for directly collecting data from all the required CAs. On the other hand, in federated mode, the load is distributed among several CPVAs. However, in this scheme, since the same request (and hence data) flows through several CPVAs, there is higher communication and processing overhead on the network and the intermediate CPVAs. But this scheme is highly scalable and has better performance than autonomous mode.

Both schemes can greatly benefit from caching at the RPs and at the CPVAs. Since CA's certificates often have longer life and are less likely to be revoked, the CPVTs tend to have much longer validity periods. This means that a CPVT once uploaded to an RP can be used for quite some time to validate subject's certificate paths. In addition, the incremental approach of updating the trees along with the push/pull methods used to distribute the changes greatly enhances the performance of the RPs as well as the overall system.

6.2.1 Costs and benefits

The performance of CPVA can best be quantified in terms of its costs and benefits when compared to traditional validation schemes.

The following benefits are claimed performance *advantages* of CPVA over traditional schemes:

- **Improved response time at RPs due to reduced time for certificate path validation.** This implies that RP's response to user's service requests is improved.
- **Improved system availability at RPs due to localized certificate path validation.** This implies that the user requests are much more likely to be validated and serviced.
- **Reduction in overall communication cost in the system due to harvesting at CPVA.** This is due to the harvesting of CA and cross-CA certificate data ahead of time and using this information to supply validation trees to the RPs.

However, these benefits are accrued at the following *costs.*

- **Cost of harvesting.** The harvester may collect more (bulk) information than needed. In addition, it needs to collect data to keep CPVA's validation trees up-to-date.

- **Cost of analysis.** The analyzer needs to analyze harvested data and build trees some of which may never be needed.

In summary, while the RPs benefit by receiving the validation trees for local certificate path validations, the system of CPVAs place additional communication and computing overhead on the system. In this section, we provide a back-of-the-envelope analysis to quantify the costs and benefits.

6.2.2 Key Parameters influencing CPVA Performance

While several parameters influence the performance of the proposed scheme, the following three *parameters* are the key factors.

- *Validity period and revocation of CA-CA certificates:* Each CA-cross certificate is valid for certain period of time. Unless its issuing CA revokes it, a cross certificate is valid until the specified expiration time. Whenever such a certificate expires or has been revoked, it results in the following costs for the CPVA:
    - Communication cost for the harvester to collect information to update the validation data.
    - Computation cost to select and rebuild the affected validation trees (if any).
    - Communication cost to send updated trees to the affected RPs

- *Rate of validation requests:* Rate of validation requests of RPs determines how effectively the validation trees are being used by the RPs. For example, when the rate of requests is high, it implies that the savings due to local validation trees at RPs is high, and, hence, it may offset the cost of building, maintaining, and uploading of these trees to RPs.

- *Maximum (trust) path length and topology:* Wider adoption of PKI will necessitate greater number of CAs, probably with complex CA-CA trust relationships. For medium to low security applications, larger (5-10) values for the maximum path length can be adopted to facilitate wider acceptance of certificates.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed an efficient certificate path validation scheme. In our scheme, a relying party receives a certificate path validation tree (CPVT) from the delegated server (CPVA). The tree is stored locally at the relying party. A relying party can now validate the path locally. The delegated server consists of a data harvester and a data analyzer. We proposed some implementation schemes for both these components. The local caching of CPVTs greatly reduces the overhead due to delegated server. At the same time, since a relying party is not responsible for collecting data from different CAs and OCSP responders on the internet, its software can still be simple. We have suggested several options for implementing the data harvester and the data analyzer. We have also shown that our scheme does not weaken the security and trust properties of delegation.

We are currently looking into further design options for implementing the CPVA. One of our objectives is to improve the efficiency of harvester and analyzer. We are also looking into extending the representation structures from trees (CPVT) to other data structures.

REFERENCES

[1] C. Adams and S. Lloyd, Understanding Public-key Infrastructure: Concepts, Standards, and Deployment Considerations. Macmillan Technical Publishing, 1999.

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

[3] S. Lloyd (Editor), CA-CA Interoperability, White paper, PKI Forum, March 2001. Available at (http://www.pkiforum.org/pdfs/ca-ca_interop.pdf)

[4] R. Moskowitz, "An Unlikely PKI Cavalry," Network Computing, Security Watch Column, Nov. 1999.

[5] A. Malpani, Bridge Validation Authority, White paper, Valicert, December 2001, Available at http://www.valicert.com/corporate/library/pdfs/Bridge_VA_Whitepaper.pdf

[6] Malpani and R. Housley, "Simple Certificate Validation Protocol (SCVP)," Internet Draft, March 2002.

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

[8] D. Pinkas and R. Housley, "Delegated Path Validation and Delegated Path Discovery Protocol Requirements," Internet Draft, May 2002.

[9] D. Pinkas, "Delegated Path Validation and Delegated Path Discovery Protocols," Internet Draft, July 2001.

[10] P. Hesse, *Certification Path Development Software, Briefing to FPKITWG,* April 2000.

[11] M. Branchaud and J. Linn, *Extended Validation Models in PKI: Alternatives and Implications, Proc. 1st Annual PKI Research Workshop,* Bethesda, Maryland, USA, April 16, 2002, pages: 30-36.

[12] T. Cormen, C. Leiserson, and R. Rivest, *Introduction to Algorithms,* Prentice-Hall, December 2000.

[13] G. Coulouris, J. Dollimore, and T. Kindberg, *Distributed Systems: Concepts and Design,* Third Edition, Addison-Wesley, 2001.

[14] C. Bowman, P. Danzig and D. Hardy, *Harvest: A Scalable. Customizable Discovery and Access System,* Technical Report CU-CS-732-94, University of Colorado – Boulder.

[15] G. Coulouris, J. Dollimore and T. Kindberg, *Distributed Systems Concepts and Design,* Addison-Wesley Publishers Limited, Third Edition 2001.