Modeling and Evaluation of Certification Path Discovery in the Emerging Global PKI

Meiyuan Zhao[1] and Sean W. Smith[2]

[1] Communications Technology Lab
Intel Corporation
Hillsboro, OR 97124
meiyuan.zhao@intel.com

[2] Department of Computer Science
Dartmouth College
Hanover, NH 03755
sws@cs.dartmouth.edu

Abstract. Establishing trust on certificates across multiple domains requires an efficient certification path discovery algorithm. Previously, small exmaples are used to analyze the performance of certification path discovery. In this work, we propose and implement a simulation framework and a probability search tree model for systematic performance evaluation. Built from measurement data collected from current PKI systems in development and deployment over more than 10 countries, our model is (to the best of our knowledge) the largest simulated PKI architecture to-date.

1 Introduction

Public key infrastructure (PKI) is a powerful tool for protecting information. Current development and deployment of PKI systems shows a trend toward an emerging global PKI, where individual PKI domains by governments, institutions, and enterprise establish trust relationships via cross-certification technology. However, as a PKI becomes more complicated, so does the work required for validating an individual certificate. The first step is *certification path discovery*: constructing a "chain of certificates" that connects the certificate in question to a trust anchor. It is challenging to locate appropriate resources to establish a candidate path and to maximize its chance of being valid.

The global PKI spans many countries and consists of many domains, CAs, repositories, and users. PKI protocols need to be robust in such a complex network environment. By establishing trust relationships between domains, cross-certification confronts us with a complex "certificate topology". Moreover, users in different PKI domains may display completely different behaviors that may impact the effectiveness of PKI protocols.

Previous analyses of certification path discovery focused mostly on using small examples to understand algorithm options. In this study, we evaluate its performance in the context of the emerging global PKI. The power of *simulation* allows us to model such complex certificate topologies and to simulate realistic situations. It also enables us to explore a wide range of algorithm options and different network environments, and to examine the effect of user activities as well. We make the following contributions:

- We design and implement a PKI simulation framework for general-purpose PKI performance study. This framework implements classical X.509 PKI services and is flexible to allow new types of models and performance studies.
- We design and implement a *PathBuilder* module for this framework. This module uses novel probability search tree models to simulate a variety of algorithm behaviors for certification path discovery.
- We model a global PKI architecture using measurement data collected from current PKI system deployment over more than 10 countries. To the best of our knowledge, this is the largest simulated PKI architecture to-date.
- Using these tools, we evaluate performance of certification path discovery using a range algorithm options. We show that the performance is sensitive to algorithm options, PKI architectures, and user activities.

We hope to make our tools publicly available, as open source.

In the rest of this paper, Sect, 2 discusses the background of PKI system and certification path discovery. Sect, 3 presents previous research. Sect, 4 discusses our simulation framework for general purpose PKI systems. Sect, 5 discusses details of our work on modeling certification path discovery and performance analysis. Finally, we conclude this work with discussions in Sect, 6 and 7.

2 PKI and Certification Path Discovery

PKI was first proposed [14] for securely distributing public keys. It has now evolved to architectures providing comprehensive services for public key *certificates*; these services include storing and retrieving certificates, maintaining and updating certificate status, and validating certificates. In a traditional X.509 [10] PKI system, the certificate storage service is provided by a repository that supports protocols for users to store and retrieve directory information; the protocol used most commonly here is the *Lightweight Directory Access Protocol (LDAP)* [23]. The *certificate status information (CSI)* service communicates the validity status of certificates. A certificate is typically considered as “valid”, “revoked”, or “unknown”. Classical approaches to CSI includes periodically updated data structures such as a *certificate revocation list (CRL)* [10], and online protocols such as *online certificate status protocol (OCSP)* [17].

2.1 Certification Path Discovery

The user who tries to validate a certificate is referred to as *relying party*. A certificate validation service handles *certification paths*, sequences of certificates representing a trust path to the certificate of interest. In such a sequence, the issuer of the first certificate is called a *trust anchor*; a trust anchor is an entity the relying party trusts by default. The last certificate in the sequence is called the *target*; the target certificate is the one that the relying party is trying to validate. In a path, consecutive certificates are linked together by having the *subject* of the previous certificate match the *issuer* of the next certificate.

A certificate validation service is composed of two stages: certification path *discovery* and certification path *validation.* The latter stage is well-established. RFC3280 defines an algorithm to validate a certification path. Basically, the algorithm examines each certificate in the path to decide if they satisfy all required conditions. Unfortunately, the algorithm for actual construction of candidate certification paths is not well defined. Several issues affect the practibility and efficiency of the certification path discovery process; we now consider some.

PKI Architecture. One critical issue is the increasing complexity of *PKI architectures*, a term we use to describe the organization of CAs and their trust relationships. A typical *PKI domain* defines a set of certification policies to manage certificates for its local users. There could be several *certification authorities(CAs)* in the system issuing certificates. These CAs may form a hierarchy having a root CA issuing *CA certificates* for subordinate CAs who in turn issue *end entity certificates* for normal users. The root CA is the common trust node for all subordinate CAs and users in this domain.

The introduction of *cross-certification* enables isolated PKI domains to efficiently establish trust with each other. In cross-certification, CAs from different PKI domains certify to each other, so that relying parties are able to establish trust paths for certificates in remote PKI domains without changing their trust anchor configuration. Furthermore, *bridge CAs* are introduced to bring structure and efficiency to cross-certification. Bridges ease the job for ordinary CAs by handling PKI policies and other constraints of cross-certification. Bridge CAs also help reduce the number of required certificates. Without a bridge CA, N domains need up to $N(N-1)/2$ cross-certificate pairs to establish trust with each other. A bridge CA reduces this number to N, where every CA cross-certifies only with the bridge CA.

Currently, there are several bridge CAs in operation or in development. In the US, the *Federal Bridge CA (FBCA)* [8] cross-certifies with more than eight Federal agency PKIs. The *Higher Education Bridge CA (HEBCA)* [9] facilitates electronic communications within and between educational institutions and Federal and state governments. The *SAFE* bridge [20] sets up trust between members of the BioPharma Association and other enterprise and government PKIs. *CertiPath* [3] is a commercially-managed bridge CA connecting to enterprise PKIs of several aerospace companies.

The trends toward bridging and cross-certification hasten the emergence of a global PKI architecture. However, this architecture creates new challenges for certification path discovery; algorithms must construct a path by traversing different PKI domains, dealing with different PKI policies and handling different protocols.

An algorithm to build certification paths within a PKI architecture can choose one of two directions: the *forward direction* (from the target to trust anchor) and the *reverse direction* (from the trust anchor to the target). The field has seen some debate on which direction is the best for certification path discovery. It appears that the forward direction is mostly appropriate for hierarchical PKIs. We assert that the choice not only depends on the topology of the PKI architecture, but also on other issues, such as the availability of resources that allow the algorithm to locate the appropriate certificates.

Directories store certificates using tuples of the form `(name, attribute)`, where `name` refers to the identity and `attribute` describes the type of object related to this identity. There are several types of attributes useful for certificate retrieval. The directory uses *cACertificate* attribute to store all certificates issued to the CA by the CAs in the same domain and *userCertificate* attribute to store all certificates issued to the end entity. The *crossCertificatePair* attribute has two elements. Its *issuedToThisCA* element stores all certificates issued to this CA including the ones by the CAs in remote domains. Its *issuedByThisCA* element may contain a subset of certificates issued by this CA to other CAs. All objects in the directory are indexed by the name and the attribute. The response to the retrieval request will return a list of objects that satisfy the criteria.

Several private certificate extensions can be used to indicate how to access services related to the certificate. The *Authority Information Access (AIA)* indicates how to access services by the issuer of the certificate. We can use AIA to specify the address of the directory where users can retrieve directory entries for the issuer. The AIA can also specify a list of CAs that have issued certificates to this issuer. Similarly, *Subject Information Access (SIA)* extension indicates how to access services by the subject of the certificate. Although properly defined, these directory attributes and certificate extensions are not fully populated in practice. This makes it difficult for the discovery algorithm to locate appropriate certificates for the path building procedure.

Optimizations. Often, the discovery algorithm faces choice of branches when building a candidate path in the certificate topology. Several optimization techniques have been proposed to help reduce wrong choices in order to speed up the process. For instance, checking signatures and revocation status early can help eliminate bad certificates early, rather than after we have used them to build a candidate path. However, trade-off exists, since the algorithm spends extra time and resource for these operations. Another approach is to prioritize branches to maximize the chance of sucessful discovery. For instance, the *Certificate Path Library (CPL)* [4] used by the *Certificate Arbitrator Module (CAM)* [21] defines a list of criteria to set priorities for branches.

We realize that many of the optimizations deserve more careful evaluation. Recall that in X.509 certificates, the issuer and subject are uniquely identified by their *distinguished names (DNs)*. DNs are an ordered list of naming attributes. Each attribute is called a *Relative Distinguished Name (RDN)*. The usage of RDNs tend to be meaningful to the local PKI system. One may declare that certificates that match more RDNs between the subject DN and the issuer DN should have priority. In other words, the algorithm expects that the issuer and the subject of a certificate in the local PKI domain have similar distinguished names, and the algorithm prefers to stay in the local PKI domain. It is unclear how effective this optimization is in practice. This is yet another reason why we need a systematic way to evaluate it as well as other proposed optimizations.

3 Related Work

Prior research has analyzed certification path discovery using small examples. Elley et al. [6] stated that optimizations in path construction are valuable. They presented a comparison of two directions for path building (forward vs. reverse), analyzed the advantages and disadvantages of each approach, and concluded that building in the reverse direction is often more effective than building in the forwarding direction. Lloyd published a white paper [15] that discussed options for effective and efficient certification path construction algorithm. He specifically pointed out that the forward direction is best suited for hierarchical trust models and the reverse direction is best suited for distributed trust models; he also suggested that building in both directions and meeting in the middle might be a good approach. Russell et al. analyzed the performance issues for constructing and validating long certification paths in cross-domain PKI systems, and proposed the concept of virtual certificates and synthetic certificates to avoid reconstructing and re-verifying certification paths [19]. Unlike these studies, we quantify the performance of the algorithm and evaluate different building options using simulation. (Our work also has the side-effect of producing a simulation tool that can be used for subsequent analyses as well.)

Some researchers have tried systematic approaches to evaluate PKI systems. Iliadis et al. presented a mechanism-neutral framework for the evaluation of CSI mechanisms [11, 12]. The authors proposed a complete evaluation framework that consists of management, performance, and security criteria. This general purpose framework can be used to evaluate many different types of CSI systems. Unfortunately, this system fails to provide quantitative analysis.

Simulation was used for CSI system evaluation too. Årnes implemented a simulation to evaluate certificate revocation performance [1]. His simulation model contains a set of simulation input and output variables, and the models used these variables to compute intermediate variables. However, the simulation models are strictly controlled by formulas. The network environment and user activities are not included.

Muñoz et al. implemented CERVANTES, a testbed for certificate validation [16]. This is a Java platform that allows researchers to develop and test their own “real” revocation systems and to analyze the temporal behaviors. The model makes a few assumptions about configurations, including population size, latency, and connectivity. The testbed is configured with a CERVANTES server and a few clients generating status checking requests. This testbed approach is more realistic than the simulation model by Årnes in that it has real implementations and it takes into account the network environment. However, it is limited by the scale of experiments.

4 PKI Simulation Framework

We design and implement a simulation framework that is capable of modeling PKI protocols and services in network environments. We focus on realizing several important features for this simulation framework—power, flexibility, and scalability.

The framework should be **powerful** to model various PKI protocols such as certificate issuance, revocation, and validation. It should handle different types of network topologies and environments. It should also include different user activities, since PKI systems involve both computer systems and users.

The framework needs to be **flexible** to allow users to add new simulation models of protocols and configurations easily. For this purpose, we design the simulation framework using modules to provide flexible interface for model users to model their own protocols. Several basic modules serve as the building blocks. These modules provide flexibility to improve the functionality easily.

We also need the simulation framework to be **scalable** to handle large-scale network environments, large relying party populations, large number of certificates, and complicated certificate topologies. For this purpose, we design the simulation framework to allow modeling using different levels of abstratctions.

Given these modeling requirements, we use *SSFNet* [18], the Java-based network simulator, to build the framework. SSFNet is good at modeling large-scale networks. It is able to model protocols at packet level as well as at the higher levels. The *Domain Modeling Language (DML)* [5] provided by SSFNet is a powerful tool to configure a variety of protocol behaviors. Furthermore, the modular design of SSFNet allows us to add more modules for PKI protocols.

The simulation framework models major services by a general-purpose X.509 PKI system. There are five primary components: certificates, storage and retrieval services, CSI services, PKI architectures, and certificate validation services. Each of these components is implemented as an independent module with a flexible interface. The simulation implements basic functionalities. At the high level, the PKI simulation framework consists of four major components—PKI Data, PKI Entities, PKI Protocols, and Network Topology. Each component is a module that provides a set of configurations that allow users to specify the behaviors and parameters.

The ***PKI Data*** module specifies data forms used in a PKI system. We implement certificates and CRLs. The PKI Data module provides a flexible level of abstraction for modeling. It can be as detailed as the certificate/CRL fields defined by X.509 profile. At the other extreme (when model users do not care about the contents in a certificate/CRL at all), the "length" parameter can be used to model the entire data structure.

The ***PKI Entities*** module manipulates PKI data. The module has built-in support for three basic PKI entities—relying parties, CAs, directories.

The *Relying Party* submodule fulfills any task by end entities in a PKI system, including requesting certificate issuance, requesting certificate revocation, retrieving data from a directory, and validating certificates. Relying parties may have a local cache to store the retrieved certificates and CRLs. Furthermore, one relying party submodule can model common features as well as differences of entire relying party population in a PKI domain. Thus one relying party module may represent many relying parties at a time.

The *Certification Authority* submodule models basic functionality by a CA, such as issuing certificates, manipulating data in directories, and validating certificates.

The *Directory* submodule models the database of certificates and CRLs; it supports data access protocols such as LDAP. The database grants read-only privilege to relying parties and full privilege to CAs on their own data. The directory model supports several popular directory attributes: *cACertificate*, *userCertificate*, *crossCertificatePair*, and *CertificateRevocationList*.

The activities or behaviors of PKI entities are configured and controlled by the ***PKI Protocols*** module. We have identified four categories of protocols—issuing certificates, revoking certificates, storing and retrieving certificates, and validating certificates. We have implemented their basic functions In SSFNet, each host contains a *protocol graph*

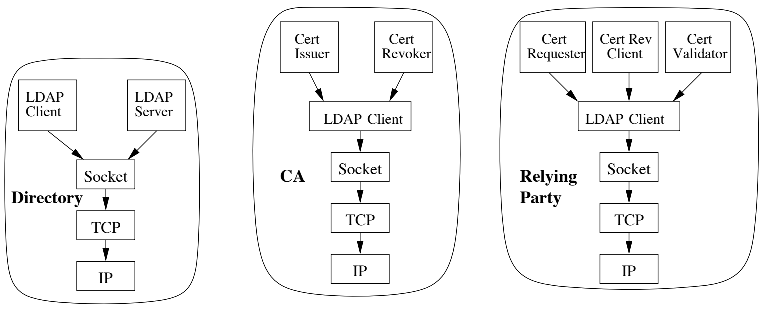

Fig. 1. The demonstration of example protocol graphs for each type of PKI entity. The basic communication protocol is LDAP.

representing the network protocols that are supported by the host. Fig. 1 illustrates typical types of hosts in this framework and typical supported protocols. Advanced protoocls for issuing, revoking, and validating certificates rely on the LDAP client protocol.

The PKI protocol module models protocol behaviors and produces the resulting performance overhead. In our implementation of LDAP, for instance, the LDAP client can send LDAP requests to corresponding LDAP server to request data. This procedure also produces related performance data such as network latency and the amount of transmitted data.

Finally, all PKI protocols operate with the help of the ***Network Topology*** module. Model users can use DML to configure any type of network topology. We suggest a star-shaped network topology that can be easily scaled to a large number of PKI domains. The network is centered around a number fully connected routers running inter-domain routing protocol, BGP. They establish the "routing core". The routing core connects all the PKI domains in. Each PKI domain forms a subnetwork with its own administration policies.

Within one PKI domain, users may configure any type of network topology with the choice of PKI-related entities and protocols. For demonstration purpose, we use a simple configuration. In each PKI domain, one directory serves the entire PKI domain. Multiple CAs share this directory. One relying party represents the relying party population in the PKI domain. All PKI entities are directly connected with the border router.

Monitoring and Measurement. In order to measure performance of PKI protocols and activities in the simulation framework, we design a set of monitoring options for monitoring a simulation run. Model users can turn on a subset of the options to observe the desired types of behavior. Current implemented options support five types of events: (1) LDAP states; (2) LDAP data sending and receiving; (3) timer setting and expiration; (4) directory data changes; and (5) message sending and receiving.

Limited by space, we omit[3] the detailed list of monitoring operations in this report. Basically, model users can print output in ASCII form or store it as a binary record. We design the records to cover as much information as possible. Model users can use

[3] Full details can be found in [24].

such measurement data to produce meaningful results, such as the number of requests, the timing of requests, the data size for each request, and the network delay for each request.

5 Evaluating Certification Path Discovery

PathBuilder is a special model for evaluating certification path discovery. PathBuilder models the behavior of the algorithm and relies on the PKI simulation framework to perform network activities. This section discusses the design of PathBuilder and presents the performance results.

5.1 PathBuilder Model

In designing the PathBuilder model, we need to take into account several important issues. PathBuilder should be able to model the trials and errors that occur during certification path discovery. Furthermore, PathBuilder should handle large-scale models, a variety of building optimizations, and user activities.

The PathBuilder module is part of the `Cert Validator` protocol, a new protocol model that handles the certificate validation process. In the protocol graph of a host, it resides on top of the LDAP Client. There are four primary modules in the PathBuilder: the Certificate Topology module, the Search Tree module, the Build Options module, and the Monitoring module. The Search Tree module is the central component. The Certificate Topology and Building Options modules configure the behavior of the Search Tree module. The Monitoring module handles the experimental output produced by the Search Tree module.

Certificate Topology. The *Certificate Topology* module is shared by all PathBuilder instances. It configures the complete certificate topology.A PathBuilder instance may configure its own partial view of the certificate topology, which is decided by the local certificate cache of the host.

Search Tree. The *Search Tree* module is the central focus in our design. As we have discussed in Sect, 2, the certification path discovery process is similar to exploring a graph. In fact, we can use a search tree to represent all choices that the algorithm has when traversing the certificate topology. The root is the start point of path building. Each branch in the tree represents a certificate. A candidate certification path (if it exists) is a path in the tree that connects the root with a leaf. The certification path building is the procedure that the algorithm walks in the tree to find this path. On reaching a node in the search tree, the algorithm retrieves certificate information either from the local cache or from the remote directories. The latter case involves LDAP requests and responses, which thus introduce network latency and data transmission overhead.

Following this logic, we model the procedure of building a certification path in four phases: constructing a search tree, assigning probabilities on branches, tree walking with probability, and generating LDAP requests.

In phase one, the model generates a search tree based on the configuration parameters: trust anchors, target, and the building direction. The algorithm may construct a search using a *forward search tree* rooted at the issuer of the target or a *reverse search tree* rooted at the relying party's trust anchors.

In phase two, the model assigns probabilities to each branch in the tree; the probability on a branch represents the likelihood that that certificate is chosen as the next step in the tree walk. Unless we are considering prioritizing the branches, each child branch from an internal node has equal probability to be chosen.

The third phase is the actual tree walking process. This process is the depth-first-search that chooses branches according to their probabilities. At each step, the model randomly chooses a branch based on the assigned probabilities.Available branches have positive probability assignments. Once one branch is chosen, its probability is set to zero so that it won't be considered in the future; consequently, the model needs to adjust the probabilities of the remaining branches to maintain their priority relation. This process ends when a candidate certification path is found or when the entire tree is explored. In the case where multiple candidate certification paths exist, any one of them satisfies the termination condition.

In the last phase, the log of tree walking is sent to the Monitoring module. The model also translates this log into a sequence of LDAP requests, either for CA certificates or for cross-certificate pairs. For the certificates that do not exist in local cache of the relying party, the model passes a request list to the LDAP client protocol, which then executes these requests and produce corresponding performance measurement.

Build Options. Our *Build Options* module handles build options: criteria to distinguish branches and change the way the probabilities get assigned. There are a variety of build options, each of which has its own properties and features. Our analysis indicates that we need to model them case by case. For one example, the CAM implementation requires that certificates matching more RDNs within the issuer DN and the subject DN have priority. We denote this option as the *RDN matching option*. Using the RDN matching option, the model assigns positive probabilities to the branches with the highest matching number. The rest of the branches all have zero probability.

Monitoring. The *Monitoring* module outputs any types of events related to PathBuilder. There are mainly three types of events: (1) search tree statistics, such as tree size, tree height, etc.; (2) LDAP retrieval activities; and)3) performance, such as network latency and amount of data transmission.

5.2 Experiment Configurations

In this section, we present simulation experiments that use the simulation framework and PathBuilder module to evaluate the certification path discovery algorithm. The experiment settings contain a set of configurations of the simulation model and a new protocol module that invokes the certificate path building processes.

Certificate Topology We design a certificate topology based on both current state and future directions of PKI deployment. To the best of our knowledge, this certificate topology is the first systematic attempt to model the emerging global PKI architecture. It is also the largest simulated PKI architecture model, and expresses the current major efforts in building a bridge-to-bridge environment for PKI systems. The configured

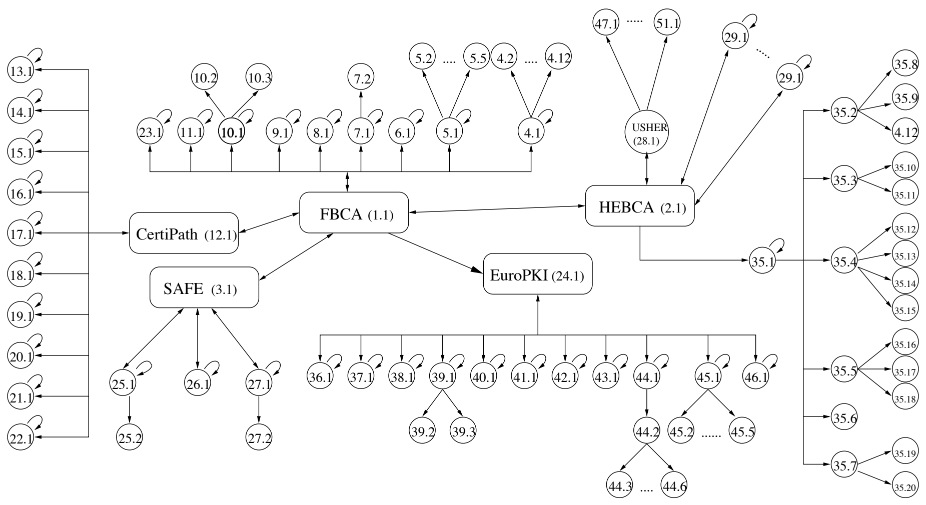

Fig. 2. The configured certificate topology for experiments. The topology is a combination of current deployment and future plans. Each CA with a self-issued certificate can be treated as a trust anchor. The unique ID for each CA is the tuple: (domainID, caID). We assign an index number to each CA in the model for simple implementation.

PKI architecture models 5 bridge CAs, 51 PKI domains with 103 ordinary CAs, and 30 million certificate users over 13 countries.

The certificate topology for our experiment is illustrated in Fig. 2. We use the four principal bridge CAs (FBCA, HEBCA, SAFE, and CertiPath) as the central piece in our experimental certificate topology. We configure the implemented or prototyped cross-certification relationships between them. We also added the *U.S. Higher Education Root (USHER)* [22], a large sector CA in development. These bridge CAs have cross-certified with many PKI domains for government agencies, institutions, and enterprises. Our configuration is mostly based on the current deployment situation. We have obtained complete data about FBCA and government agency PKI systems that cross-certify with FBCA. For systems that we could not get information for, we approximate each as a simple hierarchy that has one root CA. We use the same strategy to configure architectures of other PKI domains.

Besides PKI systems in the United States, we also try to model the connections to the PKI systems in other countries. EuroPKI [7] is currently a root CA in Europe that connects many PKI systems from several countries. We model it as a bridge CA in our certificate topology to further expand the scale and to predict that PKI systems in Europe may cross-certify with FBCA in the future. We also expand the topology to cover PKI development in South America. The PKI domain number 35 shown in Fig. 2 is the current Brazilian PKI system for all government agencies and enterprises [2]. This PKI system may cross-certify with HEBCA in the future. Finally, the certificate topology is configured with DNs of CAs. They are partially configured using the collected data. The configuration of user population size is based on the combination of measurement data and random assignment.

Configuring PKI Simulation Framework We use the simplest network configuration to minimize the impact of network protocols on the certification path discovering process. Each PKI domain has one router, one directory, and one relying party sending out certificate validation requests. For these path-building experiments, all certificates are configured statically. As one Relying Party module models the entire relying party population in a PKI domain, we use the configured local preference rate to generate random target certificates for the experiments. Each relying party has one trust anchor—its root CA in local PKI domain.

5.3 Performance Results

In this section, we evaluate performance of certification path discovery by comparing building directions and building options. We conduct the simulation experiments with 10 runs. The standard deviation of experiment results is less than 5%. Thus, the mean value is sufficient for presentation.

Forward vs. Reverse. Table 1 compares the performance by building directions. In terms of search tree properties, the reverse search trees are significantly larger. Experiments show that the average tree size is doubled. And the reverse search trees are flatter according to the path length measurements. Overall, forward search trees are more efficient than reverse search trees. This result is reasonable given that the experimental certificate topology is mostly hierarchical except in the center where bridge CAs are cross-certified with each other. The forward direction encounters only one choice when exploring a hierarchy from a leaf to the root. On the other hand, the reverse direction needs to handle many branches going from the root to a leaf.

Both directions generate similar number of LDAP requests for one target certificate. In some cases, the forward direction fails to retrieve certificates from the cACertificate attribute, then tries to search for issuedToThisCA element of a cross-certificate pair. Thus, one tree walk step may need two LDAP requests. Nonetheless, the forward direction still out-performs the reverse direction. The network latency and the amount of data transmission is smaller for the forward direction.

Property	Forward	Reverse
`avg_tree_size`	31.3	69.1
`avg_num_leaf`	26.9	55.9
`max_path_len`	3.9	4.9
`min_path_len`	2.8	2.3
`avg_path_len`	3.6	3.7

Property	Forward	Reverse
# LDAP requests	36.2	40.0
# retrieved CA certs	18.2	0
# retrieved x-cert pairs	81.5	152.8
building delay	7.7 s	9.1 s
data size	89.8KB	122.19KB

Table 1. Properties and network performance of the forward search tree vs. reverse search tree.

Local Preference. In this set of experiments, we vary the local preference rate in the range of 0.2 to 0.9. We found that for both building directions, the performance overheads decrease linearly as the local preference rate increases. This makes sense. Local targets require shorter certification paths. If there is only one CA in the PKI domain, the issuer of the target is the same as the relying party's trust anchor.

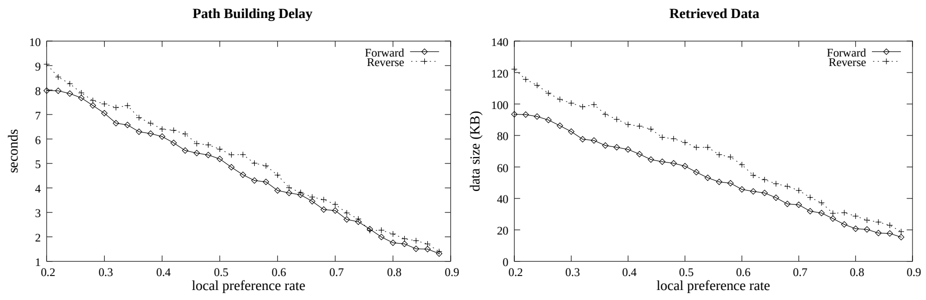

Fig. 3. Network performance by each certification path building process.

Fig. 3 illustrates the performance results for network operations. We notice that the reverse direction leads to slightly more data transmission and longer network latency, although the resulting number of LDAP requests is similar to the forward direction. On average, the reverse direction requires about 16% to 24% more data transmission. The reverse direction relies on retrieving cross-certificate pairs. LDAP server will respond with all certificates issued to and issued by a CA. In general, the returned amount is larger than retrieving data using only cACertificate attribute.

RDN Matching Option. Next, we examine how the RDN matching option helps to improve performance, especially for the reverse direction. Limited by our collected data, the certificate topology does not have a configured distinguished name for every entity. We thus assume that the RDN match value is zero if there is no DN for either the issuer or the subject.

Fig. 4 shows the impact of the RDN matching option. The RDN matching option helps in reducing the number of retrieved cross-certificate pairs for the reverse direction. The average 11% improvement suggests that the RDN matching option helps the reverse direction by avoiding some CAs with a large number of branches. Thus, the algorithm spends less time in exploring hierarchies from the root to the target. However, the RDN matching option does not reduce the amount of data transmission significantly. The amount of data in each cross-certificate pair retrieval response is still a leading factor. On the other hand, the RDN matching option has no noticeable impact on path building in the forward direction. The forward direction only encounter branch choices

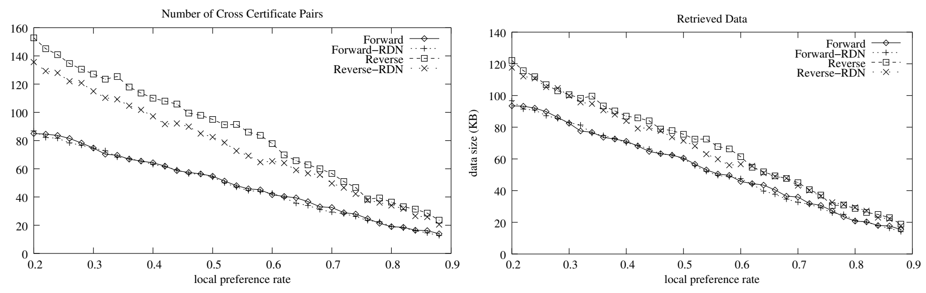

Fig. 4. Performance with RDN matching optimization. “Forward” and “Reverse” denote tree walks without optimization. “*-RDN” denotes the performance by RDN matching optimization.

when dealing with bridge CAs. These CAs typically have completely different DNs and RDN elements. The RDN matching option cannot reduce the number of choices.

Overall, simulation experiments with the RDN matching option have shown that it can help speed up the certification path building process in the reverse direction. The improvement is limited, however; the forward direction is still more efficient.

6 Discussions

Using our simulation models, we have just scratched the surface in understanding the performance of certification path discovery. Yet, we have made some important observations. In this section, we discuss these observations and further make suggestions on efficient certification path discovery.

First, the performance difference from building direction heavily depends on the architecture of certificate topology. The emerging global PKI contains a few bridge CAs and a number of hierarchical PKI systems. This architecture favors the forward direction. In practice, we suggest that the algorithm should use the forward direction as much as possible. To further make the tree walk process more effective, we suggest that relying parties set their trust anchors close to the edge of their local domains.

The hierarchical structure of local PKI domains favors any approach if it explores the local PKI domains bottom to top. We suggest that building the certification path in both directions and meeting in the middle may be the best choice. This approach not only maximally takes advantage of the hierarchies, but also significantly reduces the number of branches to explore when the algorithm is working in the center area of the certificate topology where multiple bridge CAs cross-certify with each other. Starting from both the target and the trust anchor, the algorithm quickly reaches the center area from both directions. At this point, the algorithm has dicovered two neighbor sets that may possibly contain several bridge CAs. By comparing these two neighbor sets, the algorithm may be able to discover the common node or the direct link between them quickly.

How does the algorithm decide when to pause for meeting? There are several approaches. One approach is to examine the DNs. The sudden change of similarity in DNs indicates that the algorithm may have just crossed the boundary of a PKI domain. Or, the algorithm looks for self-issued certificates, typically issued by the root CA of a hierarhical PKI domain. In general, the algorithm has a fairly good sense on when it crosses the boundary.

Second, we observe that a building optimization as simple as the RDN matching option can help improve performance if buiding in the reverse direction. The savings come from the reduced number of cross-certificate pairs retrieved from directories. Besides the RDN matching option, there are many other possible optimizations. In general, if the reverse direction is necessary, any build option that helps reduce the number of choices when exploring the certificate topology can significantly improve the performance. For instance, Elley et al. [6] suggested that name constraints and policy processing are two important optimizations. We expect these optimizations may reduce the network latency as well as the amount of transmitted data.

Lastly, the relying parties' certificate usage patterns significantly affect the performance. The simple criterion of local preference rate shows this difference. We suggest that deployer of the algorithm obtain a good understanding of the certificate usage pattern. If relying parties make frequent requests regarding validating certificates in remote domains, the deployer may need to explore approaches to minimize the performance impacts. For instance, one can choose to carefully deploy certificate caches to store certificates and revocation information as much as possible. We should also try to maintain the maximal availability of these caches to relying parties. Smart organization of the information in the cache can help too. For instance, CoreStreet implemented an online certificate validator that is able to return a sequence of certificates that may lead to the most efficient certification path in Federal PKI systems [13].

7 Conclusions

In conclusion, we use simulation to evaluate performance of certification path discovery. We have implemented a simulation framework suitable for performance studies of general-purpose PKI systems. It provides facilities to model data structures, entities, protocols, and large-scale network environments. Classical X.509 PKI services are implemented in the framework. The flexible interface of this framework enables researchers to evaluate new protocols or services in the simulated environment. We design a novel search tree model to simulate certification path discovery. Probabilistic tree walking is an effective technique to model a variety of algorithm options.

In our performance study, we examined several example algorithm options and their impact on performance. Given the current situation of PKI deployment and our experimental results, we suggest that building certification path in both the forward and reverse directions is the best choice. We also have shown that choosing certificates smartly can help improve algorithm efficiency significantly.

We are just getting started on understanding the performance of PKI services in complicated systems. In the future, we plan to extend the experiments to quantify the performance of the "meet-in-the-middle" approach and to explore more algorithm options, such as name constraints and policy mappings. We will also examine the algorithm in more realistic environments, e.g., varying number of LDAP servers for each domain, allowing the certificates to be issued or revoked dynamically, configuring more trust anchors for each relying party, or allowing relying parties cache some certificates and certificate status information. In the long run, we can use the simulation framework not only for performance evaluation, but also for other purposes, such as risk analysis. The framework can be used to model attacking scenarios and risk management system to help us understand the security of current PKI design.

References

1. André Årnes, Mike Just, Steve Lloyd, and Henk Meijer. Certificate Revocation Performance Simulations. project paper, June 2000.
2. Brazilian Government PKI System. `http://www.icpbrasil.gov.br/`.
3. CertiPath: Enabling Trusted Communication. `www.certipath.com`.

4. Certification Path Library (CPL). Cygnacom Solutions. `http://www.cygnacom.com/products/index.html#cpl`.
5. Domain Modeling Language (DML) Reference Manual. `http://www.ssfnet.org/SSFdocs/dmlReference.html`.
6. Yassir Elley, Anne Anderson, Steve Hanna, Sean Mullan, Radia Perlman, and Seth Proctor. Building Certification Paths: Forward vs. Reverse. In *The 10th Annual Network and Distributed Systems Security Symposium (NDSS'01)*, February 2001.
7. EuroPKI Top Level Certification Authority. `http://www.europki.org/ca/root/en_index.html`.
8. Federal Bridge Certification Authority. `http://csrc.nist.gov/pki/fbca`.
9. Higher Education Bridge Certification Authority (HEBCA)—Transforming Education Through Information Technologies. `http://www.educause.edu/hebca/`.
10. R. Housley, W. Polk, W. Ford, and D. Solo. Internet X.509 Public Key Infrastructure Certificate and CRL Profile. RFC3280, `http://www.ietf.org/rfc3280.txt`, April 2002.
11. John Iliadis, Stefanos Gritzalis, Diomidis Spinellis, Danny De Cock, Bart Preneel, and Dimitris Gritzalis. Towards a Framework for Evaluating Certificate Status Information Mechanisms. *Computer Communications*, 26(16):1839–1850, October 2003.
12. John Iliadis, Diomidis Spinellis, Dimitris Gritzalis, Bart Preneel, and Kokratis Katsikas. Evaluating Certificate Status Information Mechanisms. In *Proceedings of the 7th ACM conference on Computer and Communications Security (CCS'00)*, pages 1–8. ACM Press, 2000.
13. CoreStreet Inc. Distributed Path Validation—Massive Scalability for Federated PKIs. Presentation st FBCA Path Discovery & Validation Working Group, August 2004.
14. Loren M. Kohnfelder. Toward a Practical Public-Key Cryptosystem. bachelor's thesis, Dept. Electrical Engineering, MIT, Cambridge, Mass., 1978.
15. Steve Lloyd. Understanding Certification Path Construction. PKI Forum White Paper, September 2002.
16. Jose L. Muñoz, Jordi Forné, Oscar Esparza, and Miguel Soriano. CERVANTES—A Certificate Validation Test-Bed. In *First European PKI Workshop: Research and Applications (EuroPKI 2004)*, volume 3093 of *LNCS*, pages 28–42, Samos Island, Greece, June 2004. Springer-Verlag.
17. M. Myers, R Ankney, A. Malpani, S. Galperin, and C. Adams. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol. RFC2560, `http://www.ietf.org/rfc/rfc2560.txt`, June 1999.
18. Andy T. Ogielski and James H. Cowie. SSFNet: Scalable Simulation Framework - Network Models. `http://www.ssfnet.org`. See `http://www.ssfnet.org/publications.html` for links to related publications.
19. Selwyn Russell, Ed Dawson, Eiji Okamoto, and Javier Lopez. Virtual Certificates and Synthetic Certificates: New Paradigms for Improving Public Key Validation. *Elsevier Computer Communications*, 26:1826–1838, 2003.
20. SAFE Bridge Certification Authority TEST Environment. SAFE-BioPharma Association, `http://www.safe-biopharma.org/`.
21. MitreTek Systems. Certificate Arbitrator Module. `http://cam.mitretek.org/cam/`.
22. USHER: The Root Certificate Authority for Trust in Higher Education Research and Education. `http://usher.internet2.edu`.
23. M. Wahl, T. Howes, and S. Kille. Lightweight Directory Access Protocol (v3). RFC2551, `http://www.ietf.org/rfc/rfc2551.txt`, March 1997.
24. Meiyuan Zhao. *Performance Evaluation of Distributed Security Protocols Using Discrete Event Simulation*. PhD thesis, Dartmouth College, Hanover, NH, October 2005. TR2005-559.