Performance Evaluation of Distributed Security Protocols Using Discrete Event Simulation

Dartmouth Computer Science Technical Report TR2005-559

A Thesis

Submitted to the Faculty

in partial fulfillment of the requirements for the

degree of

Doctor of Philosophy

in

Computer Science

by

Meiyuan Zhao

DARTMOUTH COLLEGE

Hanover, New Hampshire

October, 2005

Examining Committee:

(chair) Sean W. Smith

David M. Nicol

David F. Kotz

M. Douglas McIlroy

Carlisle Adams

Charles K. Barlowe, Ph.D.
Dean of Graduate Studies

Abstract

The Border Gateway Protocol (BGP) that manages inter-domain routing on the Internet lacks security. Protective measures using public key cryptography introduce complexities and costs.

To support authentication and other security functionality in large networks, we need public key infrastructures (PKIs). Protocols that distribute and validate certificates introduce additional complexities and costs. The certification path building algorithm that helps users establish trust on certificates in the distributed network environment is particularly complicated.

Neither routing security nor PKI come for free. Prior to this work, the research study on performance issues of these large-scale distributed security systems was minimal. In this thesis, we evaluate the performance of BGP security protocols and PKI systems. We answer the questions about how the performance affects protocol behaviors and how we can improve the efficiency of these distributed protocols to bring them one step closer to reality.

The complexity of the Internet makes an analytical approach difficult; and the scale of Internet makes empirical approaches also unworkable. Consequently, we take the approach of simulation. We have built the simulation frameworks to model a number of BGP security protocols and the PKI system. We have identified performance problems of Secure BGP (S-BGP), a primary BGP security protocol, and proposed and evaluated Signature Amortization (S-A) and Aggregated Path Authentication (APA) schemes that significantly improve efficiency of S-BGP without compromising security. We have also built a simulation framework for general PKI systems and evaluated certification path building algorithms, a critical part of establishing trust in Internet-scale PKI, and used this framework to improve algorithm performance.

Preface

This dissertation is the final work of my Ph.D. study at the Department of Computer Science, Dartmouth College. It serves as documentation of my research work during the study, which has been made from fall 2000 until summer 2005.

The dissertation consists of twelve chapters. Chapter 1 introduces research problems, motivations, research questions, and the approach we choose to conduct this research. Chapters 2 to 6 introduce performance problems, our evaluation framework, and evaluation results. Chapter 7 presents our new proposals for efficient BGP security and discusses their performance using the same evaluation framework. Chapters 8 to 10 present the performance problem of distributed PKI systems, our evaluation framework, and evaluation results. Chapter 11 discusses the performance of PKI on BGP security. Finally, Chapter 12 concludes this research with a summary of contributions and discussions on future research work.

This research has been supported in part by Sun, Cisco, the Mellon Foundation, NSF (CCR-0209144, EIA-9802068), AT&T/Internet2 and the Office for Domestic Preparedness. This research program is also a part of the Institute for Security Technology Studies, supported under Award number 2000-DT-CX-K001 from the U.S. Department of Homeland Security, Science and Technology Directorate. Points of view in this document are those of the author and do not necessarily represent the views of the sponsors.

This dissertation adopts materials and texts from our research papers [126, 166, 167, 168].

Readers can contact me via email `Meiyuan.Zhao.Adv05@Alum.Dartmouth.ORG`.

Acknowledgment

I would like to express my deepest gratitude to Sean Smith, my advisor. I thank you for your continuing guidance and support during my five years of research. Your sharp sense of research direction, great enthusiasm, and strong belief in the potential of this research has been a tremendous force for the completion of this work. I have learned so many things from you, including the research process and excitement, writing papers, giving talks, and reviewing papers, and many more. Most importantly, I thank you for encouraging me in each step of my growing path. Your strong belief in me and continuous encouragement have made this research such an exciting experience that our collaboration finally produces something that we are both proud of.

This dissertation would not have been possible without the assistance of many people. I would like to express my great gratitude to David Nicol, my former advisor, who has made invaluable suggestions in many aspects of my work. You are the one who brought me into the secure routing and simulation fields. Your broad knowledge and sharp insight have assisted me identify the research directions, find the appropriate approaches, and finally make achievements.

I thank David Kotz, for taking the time reading my thesis and giving me valuable suggestions. Doug McIlroy, I thank you for sharing the enthusiasm for my research. Carlisle Adams, I feel honored to have you on my thesis committee; thank you for reading so carefully through my thesis and sharing your comments.

I would like to thank my colleagues and friends in former Cybersecurity Research Group at Institute of Security and Technology Studies. My discussions with Brian Premore, Michael Liljenstam, Jason Liu, Guanhua Yan, and Yougu Yuan gave me many inspirations. I feel so fortunate to work with many wonderful people in PKILab at Dartmouth College. I thank them all. Scott Rea, thank you for sharing your experience and knowledge with me on building real public key infrastructure systems. Eileen Zishuang Ye, John Marchesini, Alex Iliev, Sarah Sinclair, John Baek, Chris Masone, Anna Shubina, Mike Fromberger, Bob Brentrup, and Mark Franklin, thank you for research suggestions and the fun times.

There are many other people whose names are not mentioned here. It does not mean that I have forgotten you or your help. It is a privilege for me to work and share life with so many bright and energetic people. Your talent and friendship have made Dartmouth such a great place to live.

I would never get this far without the support of my parents. Thank you for always believing in me and supporting me. Your love and encouragement have been and will always be a great source of inspiration in my life.

Qing, my dear husband, you are always my strength. I owe my deepest gratitude to you for your infinite patience that accompanied me along this long journey. Your love pulled me through many difficult times. I thank you for the most wonderful ten years of my life. I look forward to the journey ahead of us with you and our incoming baby. I love you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The Internet consists of an enormous number of heterogeneous, independently managed computer networks. The Border Gateway Protocol (BGP) is the standard routing protocol that *autonomous systems (ASes)* use to establish and maintain routing information between their domains. As the only standard and ubiquitous way of managing routing across the entire Internet, BGP plays an important role in Internet efficiency, reliability, and security. Lack of strong security for BGP is a serious concern that has been recognized for some time. As routers maintain their routing tables by hearsay, a security solution should protect the route announcements exchanged by routers. A natural approach that provides strong security is to use *public key cryptography*: a router could verify the authenticity of each component of an announced route it receives, if each router along the way digitally signs the data it adds. To work in practice, such an approach usually requires a *public key infrastructure (PKI)*.

However, neither public key cryptography nor public key infrastructure come for free. Costs are involved. When designing and analyzing these large information-distribution systems, it is easy to overlook these implementation details, and the performance impact they can have on the overall protocol. Furthermore, evaluating how much these costs affect overall performance is challenging. The complexity of the Internet makes an analytical approach difficult; and the scale of Internet makes empirical approaches also unworkable. In this research, we use discrete event simulation to evaluate performance of primary BGP security protocols and PKI systems in distributed internetwork environments.

1.1 Inefficient BGP Security

There have been a number of security protocols proposed to secure inter-domain routing. However, they are far from being deployed in the real world. There are many factors that affect the practicability of a security protocol. For instance, the security might not be strong enough, or the design may not be flexible or scalable. Performance is also a critical issue. It is difficult for an expensive security protocol to be considered sufficiently practical to be deployable. Performance is easily ignored during the protocol design process. In this research, we are interested in specific performance problems, and we ask (and answer) research questions about how to evaluate and improve performance.

1.1.1 Performance Problems

BGP suffers from performance problems. The Internet has become increasingly dynamic, which affects network stability. Increases in the number of routing changes causes higher CPU usage in routers, which increases the end-to-end response times observed by users. Many studies have observed that the more work BGP needs to do, the less stable the network is. The Internet suffers when the inter-domain routing system is overloaded or inefficient.

Securing BGP unfortunately introduces more performance overhead on routers. In addition to ordinary BGP operations, routers spend more CPU time and space on handling various security-related data structures. Here we enumerate several specific problems associated with BGP security.

Problem 1: Prolonged Update processing

The instability of BGP has attracted a lot of attention. Despite many efforts on speeding up the convergence time, it is still an unsolved problem. Security protocols for BGP exacerbate the problem. Additional security operations cost routers even longer time to converge to stable routes. We ask questions about how much more CPU time is spent by BGP routers and how much impact such CPU overhead has on convergence time.

Problem 2: Increased network traffic

Security-related data structures, such as digital signatures, public key certificates, etc., are relatively large compared to ordinary BGP messages. Once the security protocols require such data to be transmitted over the network, network traffic is increased. We ask questions about how much longer the Update messages can be for the necessary security operations.

Problem 3: Increased local memory costs

Inter-domain routing results in large routing tables that contain an enormous number of routes to every corner of the Internet. Beyond these routing tables, now routers must also store additional security related data structures. We ask questions about how much more space routers need to cache the data and whether routers can store them within current hardware configurations.

Problem 4: Additional management complexity

Some security protocols need additional management processes or protocols to manage security information besides ordinary BGP routing. Additional actions besides ordinary routing activities introduce complexity. We ask questions about how much complexity routers can handle.

1.2 Research Plan on Evaluating BGP security

We have presented the general problem that a BGP security protocol results in degraded Internet performance. Now, we ask specific research questions about how to evaluate the performance degradation that enforcing security adds to BGP.

In this research, we evaluate the following BGP security protocols:

1. Secure BGP (S-BGP) [84]
2. Origin Authentication (OA) [5]
3. Signature Amortization (S-A) [126]
4. Aggregated Path Authentication (APA) [168]

S-BGP is an important subject protocol in our study for three reasons. First, it is possibly the most concrete security proposal for BGP to date. Second, it provides comprehensive solutions to various parts of BGP routing. Last, compared with other protocols it offers relatively strong security. The rest of three protocols try to improve the efficiency of S-BGP from different perspectives. All these security protocols will be compared with S-BGP. As part of this research, we invented Signature Amortization (S-A) and Aggregated Path Authentication (APA) to improve the performance of S-BGP. We quantify their performance and compare it with S-BGP.

1.2.1 Research Questions

In evaluating S-BGP and other security protocols, we wish to answer the following particular questions.

Question 1: How to evaluate BGP security performance?

The first challenge we are facing is to find right approaches to performance evaluation. We need a method that is able to present performance problems and to summarize the behaviors of security protocols. We also should design a set of metrics that can compare security protocols qualitatively and quantitatively. Furthermore, the evaluation system takes into account the protocol design as well as the network environment where the protocol is operating. In brief, the research question is to design and develop an evaluation framework capable of evaluating BGP security protocols in large-scale networks.

Question 2: How much performance overhead is introduced by S-BGP?

S-BGP introduces computational and space costs to ordinary BGP operations. We need to identify the security-related data types involved, discover their corresponding time and space overheads, display or translate these overheads into proper measurement metrics, and finally present good performance evaluation results compared with ordinary BGP without any security extension. It is also challenging to evaluate and compare S-BGP variants that use some of the proposed performance optimizations.

Question 3: How to evaluate S-BGP PKIs?

The PKI systems of S-BGP are very important not only to the design but also to the overall system performance. We should carefully evaluate their performance. Compared with normal S-BGP operations on route information, the details of operations for S-BGP PKI systems are not so clearly specified by authors. Thus our evaluation needs to consider possible options. Another challenge comes from the scale of the system. S-BGP PKIs express authorization of all participating entities over the Internet. It is very difficult to use a single mathematical model or equation to represent these authorization relationships and corresponding performance overheads. Furthermore, as these PKI systems do not exist yet, it is challenging to collect data for evaluation purposes.

Question 4: Can we improve the efficiency?

Finally, our research does not stop with just evaluating the performance of existing protocols. Rather, the goal is to diagnose performance problems that hamper real-world deployment, and to suggest more efficient protocols to improve the practicability of BGP security. Therefore, with in-depth understanding of the performance, the next challenge is to propose new efficient protocols that solve the performance problems in existing protocols without compromising security.

1.3 Inefficient PKI Systems

A PKI system contains heterogeneous components that interact with each other. There are a number of different protocols operating in a typical PKI system. Previous studies have mostly focused on how to make those entities operate correctly and securely. Yet protocol performance is one of the important factors for a PKI system. Performance plays a critical role especially in a large-scale networks. In this research, we are interested in specific performance problems of PKI systems. Our focus is on detailed evaluation and potential improvement of performance.

1.3.1 Performance Problems

The PKI systems have performance problems. Two components are especially problematic: certificate validation service and certificate revocation service. These services are complex with many different types of involved parties. They introduce complex data structures and operations that are expensive in terms of time and space. Moreover, some of the protocols have many options. There has not been a detailed performance evaluation system that compares the benefits and costs of these options.

Next, we enumerate several specific problems.

Problem 1: Heterogeneous systems

Many different types of parties have roles in a PKI systems. Various authorities issue certificates and revocation information. Other servers serve certificates and related information. Relying parties use certificates for their PKI-enabled applications. Not to mention that there are not only computers and servers, but also human activities are involved. The protocols for certificate revocation (such as CRLs and OCSP) and certificate validation (such as certification path building and validation) are particularly complex. The scale of the system exacerbates the situation. We ask questions about what types of parties affect system performance, in what types of protocols, and to what degree.

Problem 2: Network latency

A PKI system provides services for users. One user request may need interactions with multiple PKI protocols, each causes several network operations. We ask the question about how fast users can receive responses from the networked PKI system, with such complexity of network operations.

Problem 3: Local space costs

As user population increases, a PKI system may process more requests and more certificates. Consequently, the protocols will handle larger amount of data. Now different protocol designs could cause different space costs on the local entities and on the network. We ask questions about how much space the protocols need locally and how the space costs affect other entities on the network.

Problem 4: Complexity of Certification Path Discovery

A PKI system may deal with multiple PKI domains over the network. Each domain has servers and authorities that work under their own policies and configurations. Once these domains cross-certify with each other, the PKI system is much more complicated. Certification path building algorithm is the particular procedure that faces such complexity. We ask questions about how much performance overhead the algorithm needs to pay to traverse PKI domains to retrieve the necessary information to build a valid certification path and if there are ways to improve its performance.

1.4 Research Plan for Evaluating PKI systems

We have identified several performance problems with PKI systems. Because of the scale and complexity of the PKI systems, it is challenging to evaluate related protocols. In this research, we seek to build a general evaluation framework to apply to the most problematic protocols, certificate revocation and validation services. Searching through the literature, we found there was not much attention to certificate validation services. We focus on certificate validation in this research. There are two steps in validating a certificate. First, the user collects the necessary information to build a certification path for the target certificate. Then the user uses a standard algorithm to validate the certification path. The certification path building algorithm is not standardized. In practice, implementations of this algorithm are ad-hoc. Furthermore, there are very few studies of the algorithm performance issues. Hence, we choose certification path discovery as one focus in this research. The problems related to certification path validation algorithm are part of the future research.

In evaluating PKI systems, we identify two primary subjects:

1. Evaluation framework for general PKI systems
2. Performance of certification path building algorithms

Next we enumerate the main research questions related to these two subjects.

1.4.1 Research Questions

Question 1: How can we evaluate performance of PKI systems?

It is challenging to design and develop an evaluation framework for PKI systems, because this framework should cover a number of data structures, a variety of involved entities, and multiple protocols and services. Moreover, we should be able to use this framework to study protocols as they are operating in the network. The framework should have two important properties: flexibility and scalability. It should be flexible so that we can easily add new protocols and configurations. It also should be scalable so that we will be able to use it to study very large PKI systems.

Question 2: How efficiently can certification path discovery perform?

Given the evaluation framework, the next challenge is to use it to evaluate certification path algorithms. We need to find an approach to express the algorithm behavior and performance issues within the evaluation framework. The research question is to understand the efficiency of the algorithms in the network given large user populations, dynamic user behaviors, and large-scale networks. We should understand how quickly the algorithm can discover a certification path and how much local space the user needs to store relevant data.

Question 3: How do the path building options help performance?

Certification path building algorithms face many choices during the building procedure. Many algorithm options help make decisions to improve the efficiency. Yet, there is no quantitative evaluation of these options. Here the research question for us is to use the evaluation framework to evaluate and compare the algorithm options. We hope to make concrete suggestions on the most effective algorithm options.

Question 4: How does a PKI architecture affect the algorithm efficiency?

There is an important observation that the performance of certification path building algorithm is sensitive to the network environment and types of PKI architecture. We ask questions about how to use the evaluation framework to express PKI architectures so that we can examine the PKI architecture issue for certification path building algorithm. Moreover, we need to develop a PKI architecture that is realistic, scalable, and easy to express in the evaluation framework. We hope to make suggestions to PKI implementers on optimal approaches to deploy their PKI systems to achieve good performance.

Question 5: Can we find new approaches to improve algorithm efficiency?

Finally, with an in-depth understanding of the performance issues with certification path building, we ask questions about how to improve the algorithm performance. Besides evaluating existing algorithm options, we also try to attack the performance problem from the protocol design itself. The challenge is to propose new efficient protocols that solve the performance problems in existing protocols without compromising security.

1.5 Approach

Discrete-event simulation [8, 94] is the methodology we are going to apply for our performance evaluation. The particular simulation tool we propose to use is a Java-based discrete-event network simulator—SSFNet [127]. Besides simulation, there are other options of methodology for evaluating properties and/or performance of distributed security protocols, such as testbeds/prototypes, analytical models, and formal methods. Simulation is the best approach for our studies, given the characteristics of the protocols.

Although protocols securing BGP and protocols in PKI systems are quite different, they do share common natures. First, they both are *autonomous* in that no entity should simply trust any other entity in the system. In both BGP and PKI systems, organizations have their intra-domain policies as well as inter-domain policies. Second, they are *distributed* systems in that no entity can possess a complete view of the system. BGP routers do not have the a correct graph for the complete network topology of the Internet. Similarly, entities in PKI can typically obtain only partial information on certificates issued by authorities to other authorities or other entities. Moreover, since it demands time for newly updated information to be propagated through the system, the graph that the entity has does not represent a current snapshot of the system. Such a situation further increases difficulties for operations that use these graphs. Third, both systems require *large-scale* analysis, in that some global effects are not predictable when we examine the systems in a small scale. Convergence behavior of BGP and incorrect decision of trusting certificates in the above discussion are examples of such effects. Finally, both systems demonstrate *highly dynamic behavior*. Various dynamic factors have impacts on protocol behavior. Such factors include policies, network connectivity, network traffic, client reactions, protocol real-time decisions, etc.

Theoretical and measurement-based approaches are two types of primary approaches that previous research studies have used to study BGP routing behavior and PKI systems. Theoretical studies focus on the fundamental properties of the systems, such as whether the routing system can converge or can the relying party conclude trust on a certificate given all evidence. While capable of proving some important properties of the protocols, theoretical approaches have limitations. Both the BGP routing system and the PKI system display numerous detailed behaviors of the system that are too complex for theoretical analysis to handle.

Measurement-based approaches, such as testbeds, prototypes, and Internet measurement, are better for studying protocols in practical environments. For BGP routing behavior, the analytical method is commonly used by researchers. However it requires a fairly large-scale deployment of measurement facility and software over the Internet. Furthermore, a pure measurement-based approach can not easily control the conditions where the routing behavior is observed. For PKI systems, previous research has tried to use testbeds. As the scale of PKI systems grows on the network, a relatively small testbed is not able to examine unpredictable global effects.

The simulation approach stands in the middle between a theoretical approach and a measurement-based approach. As in the empirical measurement studies, we can use simulation to observe the network's behavior in practical scenarios. And it is easy to modify the system in any way we like. However, simulation has drawbacks. We can not expect the simulation to model every detail in the system. It is important that we extract the important properties.

Using simulation, we are going to explore primary BGP routing properties, security properties, test new approaches, and evaluate corresponding BGP behavior on a large scale. Similarly, simulation enables us to explore a number of protocol settings, different network conditions, and various algorithm options for evaluating certification path building.

We use SSFNet for studying BGP security protocols since it is the only scalable and detailed simulation model for BGP routing. BGP implementation in SSFNet implements the BGP specification and provides more than 50 switches to configure BGP behaviors, recording events, and reporting routing behaviors. Then our plan is to add more functions on top of the existing BGP module to simulate security protocols.

When this research began, there was no comprehensive simulation model for networked PKI systems. We built our own, again, using SSFNet. It is a Java-based discrete-event simulation package. It is capable of modeling large, complex networks. SSFNet simulates hosts, network links, and network protocols in detail. In modeling PKI systems, we simulate a number of entities, such as relying parties, servers, CAs, and other related entities. We also model certificates, CRLs, and any other necessary data structures. Most importantly, we simulate PKI protocols, observe their behaviors, and evaluate and analyze their performance.

1.6 Thesis Roadmap

This thesis covers two primary subjects: BGP security and PKI systems. We expand each subject into multiple topics. Thus, in order for the reader to better keep track of the contents in this thesis, we provide a roadmap.

We start by introducing BGP background in Chapter 2. Chapter 3 presents the simulation framework we designed for evaluating BGP security protocols. Chapter 4 introduces the design of Secure BGP (S-BGP) in detail. We evaluate the performance of three primary components of S-BGP in Chapter 5, 6, and 11 respectively. Based on the performance analysis, we designed and evaluated new efficient path authentication schemes for BGP. Chapter 7 presents details.

We use a separate thread to present performance evaluation on PKI systems. Chapter 8 introduces background on PKI systems and certification path building algorithm. Chapter 9 presents the evaluation framework. Chapter 10 discusses the spanning tree model for simulating certification path building and presents performance results.

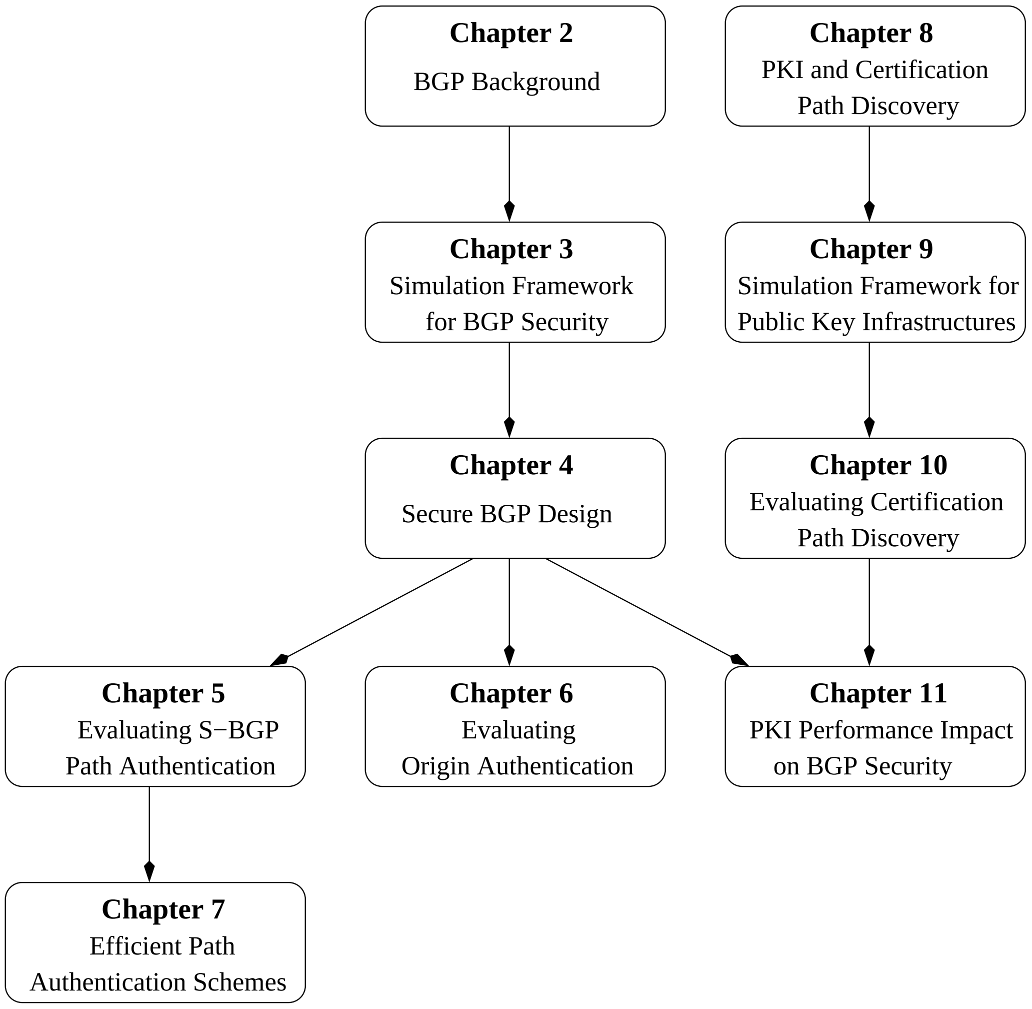

Chapter 2
BGP Background
Chapter 8
PKI and Certification Path Discovery
Chapter 3
Simulation Framework for BGP Security
Chapter 9
Simulation Framework for Public Key Infrastructures
Chapter 4
Secure BGP Design
Chapter 10
Evaluating Certification Path Discovery
Chapter 5
Evaluating S−BGP Path Authentication
Chapter 6
Evaluating Origin Authentication
Chapter 11
PKI Performance Impact on BGP Security
Chapter 7
Efficient Path Authentication Schemes

Figure 1.1: Thesis roadmap

Chapter 2

BGP Background

The *Border Gateway Protocol (BGP)* manages inter-domain routing on the Internet. Its wellness is critical to Internet availability, efficiency, and security. Previous research on BGP falls into two primary categories: routing properties and security. The studies on routing properties have focused on theoretical properties about routing protocols (such as whether or not they are loop-free) or how protocols behave in practical scenarios. Among these routing properties, convergence has been intensively studied, since it is the property of BGP that implementers and administrators are most concerned about. Fast convergence is a desirable performance property that people would like for BGP. Previous security studies have focused on how to apply cryptographic techniques, systematic designs, and/or traffic engineering to identify BGP security threats, to enhance the routing authentication and authorization, and to protect BGP from malicious attacks on routing information. This chapter briefly reviews the BGP design, problems, and prior research into performance and security.

2.1 Inter-Domain Routing

Routing in the Internet consists of two levels—*inter-domain routing* and *intra-domain routing*. The boundary of these two levels is defined by autonomous systems. The classic definition of *autonomous system (AS)* is a set of routers under a single technical administration, using an interior gateway protocol and common metrics to route packets within the AS, and using an exterior gateway protocol to route packets to other ASes [141]. For instance, the Dartmouth campus network forms an autonomous system under the Dartmouth network management policy. Nowadays, there can be multiple interior routing protocols and metrics with in one AS. The use of the term "autonomous system" stresses the fact that the administration of an AS appears to other ASes to have a single coherent interior routing plan and presents a consistent picture of what destinations are reachable through it. We commonly consider an AS a *domain*. Each AS has its own unique identifier, referred to as *AS number*.

The Border Gateway Protocol (BGP) is an inter-domain routing protocol. BGP is a variation on the distance vector routing paradigm. A BGP *speaker*—a router executing BGP—maintains connections, commonly referred as *BGP sessions*, with neighboring speakers. An autonomous system contains one or more BGP speakers. A BGP speaker uses BGP sessions to distribute network reachability information to those neighbors and to select its own best or preferred paths to destinations based on the reachability information learned from them. Such a decision is based on the distributed shortest-path computations using the Bellman-Ford algorithm. BGP is categorized as a *path vector protocol*, because BGP speakers can use a number of path attributes to select preferred paths. BGP is more flexible than *distance vector protocols*. Routers executing a distance vector protocol use the path length as the only criterion to select preferred paths.

2.1.1 BGP

In this section, I briefly introduce the details of BGP operations that are relevant to my research study in this thesis.

Prefixes and forwarding tables

An autonomous system manages subnetworks, each one described by an IP *prefix*—a fixed pattern of the n highest order bits shared by all devices in the subnetwork. The value of n is variable to different prefixes. Packet forwarding is based on prefixes: given a packet's destination IP address, a router searches its *forwarding tables* for the longest prefix that the destination belong to, and forwards through the router's network port the table associated with that prefix.

BGP routes

The construction of forwarding tables is based on the BGP *routes* that the speaker manages. A BGP route specifies a path to reach the destination. We can use a tuple `(NLRI, AS path)` to express the function. *Network Level Reachability Information (NLRI)* is the route field that defines the route destination. NLRI contains a list of prefixes. In some cases, NLRI is as simple as one prefix to express the IP address range of a subnetwork. More generally, a BGP speaker uses multiple prefixes in a single route to announce the reachability of multiple subnetworks in the autonomous system. The *AS path* is a sequence of ASes through which one can traverse the network. The last AS in this sequence is the *originator* of this route that manages the subnetworks that contains the prefixes. For instance, if the autonomous system AS_k owns IP prefix p, the autonomous system AS_0 might send out $(p, \{AS_0, AS_1, \ldots AS_k\})$ to announce the route that AS_0 prefers to use for reaching p. Besides NLRI and AS path, there are other path attributes. All path attributes are helpful to BGP speakers to make routing decisions.

RIBs and route selection

BGP speakers keep routes in the *Routing Information Bases* (RIBs). There are three types of RIBs—*Loc-RIB*, *Adj-RIBs-In*, and *Adj-RIBs-Out*—serving different purposes. Loc-RIB records all preferred routes for each prefix. It's also the base for constructing forwarding tables. The speaker uses Adj-RIBs to record all routes received from or sent to its neighboring speakers, known as its *peers*. For each peer, one Adj-RIBs-In keeps received routes from that peer and one Adj-RIBs-Out stores all sent routes to that peer. Depending on implementation, Adj-RIBs and Loc-RIB can either be physically separated or centralized. We consider the Adj-RIBs-In and Loc-RIB together as the routing table of the BGP speaker. Typically, the BGP speaker keeps all received routes to the same prefix and marks one of them as the preferred route that belongs to Loc-RIB.

The way that a speaker handles incoming routes, chooses preferred routes, and sends out corresponding new routes is controlled by *policies*. Policies can be different for each autonomous system. The most commonly used route selection policy is the path length. In other words, if multiple routes to the same destination exist in the routing table, the BGP speaker prefers the one with the shortest AS path. There are a set of rules that specify what kinds of routes the speaker is allowed to accept in Adj-RIBs-Ins, a set of rules that define what it means to be "preferred" routes in this autonomous system, and yet another set of rules to control what and to where the speaker should send out to peers.

Updates and route announcements

Typically, a speaker's Loc-RIB changes when it adds a new route, deletes a preferred route, or replaces a previously preferred route with a new one. BGP speakers incrementally send *Update* messages to announce such changes to peers. If the speaker learns a new preferred route to a prefix, it updates its Loc-RIB and informs all its peers (except the peer who sent this new route earlier) about the change. In doing so, it extends the existing AS path by prepending its AS number to it and sends out Update messages. The Update messages that are used to announce new routes are also called *advertisements*. When a speaker announces a route to prefix p, it implicitly *withdraws* the last route it announced to p. The recipient, understanding this implicit route withdrawal, decides whether it prefers the new route. A withdrawal can also occur as an explicit announcement, with no mention of an alternative preferred route. The recipient may examine the previously received routes from Adj-RIBs-Ins to the same prefix and decide whether there is an alternative that can be the new preferred route. If no such route found at hand, it simply withdraws the route and announces to its peers in turn.

Besides normal route exchange, Update messages can occur when BGP sessions change. One way Update messages occur is when an AS announces prefixes it *originates* to its peers. This typically occurs when it has just set up new BGP sessions with peers, or when it has just learned that a subnetwork is reachable via the interior routing system within the AS. In this case, the AS path of the announced route in the Update message contains a single AS number. Another way Update messages get generated is related to the change of connection state between two peers. As mentioned above, peers maintain BGP sessions with each other; periodical KeepAlive messages are responsible for maintaining the liveness of the session. Sessions do go down, for a variety of reasons. When a session is established (or re-established) the endpoints share their entire routing tables with each other, in the form of a large number of route announcements in Update messages.

Rate Limiting

BGP rate-limits the sending of Update messages with a parameter called the *Minimum Route Advertisement Interval (MRAI)*, which is basically the minimum amount of time that must elapse between successive batches of Updates sent to a peer. Speakers use output buffers to keep waiting Update messages and send them in batches when reaching the MRAI. There can be different MRAI for each peer; or one MRAI can control all peers. In practice, throughout the Internet, the default value of the MRAI is 30 seconds.

2.1.2 BGP Convergence

One of the primary metrics used to evaluate the effectiveness of a routing protocol is convergence time. Generally, the *convergence time* of a distributed system is the time required for the state of the system to become stable after a change occurs within the system. The BGP is considered to be in a stable state when no Update messages are actively generated or sent and all speakers have stable RIBs. A variety of changes that can trigger the routing activities. Such changes include a speaker announcing a change in its routing information via an advertisement or a withdrawal, a physical change in the network such as a down BGP session, a router crash, or a link failure. Speakers will then propagate such a change via Update messages through the network, like a wave. The convergence time measures the length of time for such a wave of announcements to die out completely and the system to return to a stable state. Before speakers reach a stable state, the continual changing of preferred routes degrades the effectiveness of packet forwarding. Traffic to any destination for which the route to that destination is changing has a greatly reduced chance of being transferred successfully. Longer convergence times thus reflect increased network instability and decreased network reliability, and may cause severe network performance problems, such as delayed packet forwarding, prolonged latencies, increase of packet loss rate, and increase of network congestions.

2.1.3 Studies on BGP Properties and Performance

Theoretical Approach

Early research on distributed routing algorithms focused on theoretical analysis to prove fundamental properties about routing algorithms. Tajibnapis gave a proof that distance vector routing in a dynamic network can converge in finite time [154]. However, his algorithm failed to solve the *counting to infinity* problem. In certain network circumstances, affected by link failures, the algorithm tries to count the shortest paths to infinity. The popular interior gateway protocol RIP [100] solves the problem by redefining “infinity”.

Cheng, Riley, Kumar, and Garcia-Luna-Aceves first described the *delayed convergence* problem. They discovered that, in certain circumstances, the routers running the distributed Bellman-Ford algorithm may repeatedly announce information corresponding to paths with loops [23]. Such a phenomenon is referred to as “route oscillation”. The authors then proposed the idea of adding path information in the routing algorithm to solve the delayed convergence problem. It turned out to be a precursor to BGP.

In early 1990’s, people tried to design an inter-domain routing protocol for the Internet. There were several options. Breslau and Estrin proposed three design criteria [19]: location of the routing decision (at the source or a hop-by-hop basis), type of the algorithm (link-state or distance vector), and routing policies (explicit or embedded). At first, the favorite was a source routing, link state algorithm, and with explicit policy for long-term inter-domain routing. However, this choice faced severe scalability problems. Given the size of the Internet, it is extremely difficult to keep the link state database synchronized on all routers.

In 1994, BGP began to be the approved inter-domain routing protocol for the Internet. In 2000, Varadhan, Govindan, and Estrin observed that with certain BGP policy rules, it is possible the some BGP speakers would never converge [157]. They called the phenomenon "persistent route oscillation". This study stimulated more research on identifying route oscillations and designing safe policies to prevent oscillations. Griffin and Wilfong tried to use a theoretical modeling approach to address the BGP divergence problem. They designed the Simple Path Vector Protocol (SPVP) as an abstract model of BGP [61, 62]. They used SPVP to check safety conditions of policy combinations. It turned out that, in the worst case, computing such combinations is NP-hard. They further used SPVP to show a set of sufficient conditions for safety of BGP policies [60].

In order to understand slow convergence, Labovitz et al. computed loose bounds for BGP convergence [90]. In a fully connected topology, there could be totally $O(n!)$ Update messages generated after a prefix becomes unreachable, where n is the number of nodes.

Empirical Approaches

Gao et al. noted in [51] that "proving properties about a dynamic, distributed protocol is very difficult in practice". Hence much BGP convergence research is conducted by empirical approaches. The most thorough study of BGP instability was by Labovitz, Malan, and Jahanian [92]. The authors used data collected from BGP routing traces to analyze routing instabilities and pathological behaviors. They came to the conclusion that there were several orders of magnitude more BGP Updates than expected; and about 99% of them are redundant or pathological. Later, they were able to identify some of the causes of the redundant and pathological information, most of which were from manufacturer's bugs and poor implementation choices [93]. In their follow-up studies, Labovitz et al. collected more BGP data for their analysis and concluded that although the Internet backbone infrastructure fails quite often, the Internet is surprisingly robust, since most of the outages are short-lived [90]. The authors injected various BGP messages and observed changes in BGP routing tables at various locations. They found that the convergence was considerably longer than expected and there existed strong correlation between delayed BGP convergence and increased packet loss rate and latency. Later, Labovitz et al. found that routing policy and topology had an impact on slow convergence as well [91].

Simulation

The third approach used for BGP study is simulation. Compared with intensive work in theoretical study and empirical study, there are only a few efforts that used simulation. Premore [138] built a detailed simulation model of BGP using SSFNet [127], a discrete-event network simulator that is used for large-scale network protocol modeling and study. The detailed implementation was compliant with the official specification, and it also included several extensions and vendor-specific features. This simulator has a large number of options for configuring BGP behavior. Premore further used his simulation to develop analytic models to quantify BGP advertisement, withdrawal, and convergence behavior under a wide variety of conditions. His study suggested that the configurable rate-limit timer in BGP, whose value is rarely changed from the default by network operators, may be the key to a more stable Internet [59]. For any given combination of parameters, there exists an optimal setting for the rate-limiting timer above which convergence time increases linearly with timer setting. We are going to revisit this issue in Section 6.2.2. Premore's BGP simulation had been used in a number of other BGP studies. Mao et al. used it to study the impact of route flap damping—a technique to reduce route oscillation—on BGP instability, and concluded that route flap damping exacerbates Internet routing convergence [102]. Pei et al. used the simulation experiments to show that by adding consistency assertions in Update messages, BGP convergence could be improved [131]. Zhao et al. experimented with a new BGP extension for managing multiple-origin AS conflicts [169].

2.2 BGP Security

Lacking security is not a unique problem for BGP. For most routing protocol designs, security is only considered as an add-on feature. In 1988, Perlman first identified the routing security problem in her thesis [132]. According to her, there are two types of network failures, similar to classical system failures. A *simple failure* occurs when a node or a link in a network becomes inoperative, and ceases to function at all. A *Byzantine failure* occurs when nodes or links continue to operate, but incorrectly. A node with Byzantine failure may corrupt, delay, or forge messages, or may send conflicting messages to different nodes. Most of the threats or vulnerabilities for routing protocols are related to this type of failure. Since then, there had been many research efforts on identifying vulnerabilities of individual routing protocols and corresponding security improvements. Recently, the *IETF Routing Protocol Security Requirements working group (rpsec)* observed that the lack of a common set of security requirements and methods for routing protocols has resulted in a wide variety of security mechanisms for individual routing protocols. The working group has documented the generic threats to routing protocols [9] and generic security requirements for routing protocols [139]. Most of security vulnerabilities of generic routing protocols also exist as BGP vulnerabilities.

It has been widely recognized that the lack of security in BGP is a critical problem to the Internet [84, 123]. One of the main security issues is that speakers completely believe the information told to them by peers. Each speaker constructs its routing table and forwarding table via hearsay: in general, the only knowledge AS_0 has of the particular path $AS_1, AS_2, ...AS_k$ to a prefix p is the fact the AS_k claims to originate p, and that along the way, each AS_i forwards to AS_{i-1} the commitment of $AS_{i+1}...AS_k$ to participate in this path. Consequently, nothing prevents a malicious speaker Q from simply fabricating a claim to originate a prefix, or from fabricating a route to a prefix and announcing that to its peers. If the fabrication is sufficiently attractive, this misinformation will propagate throughout the network.

A comprehensive analysis of the security vulnerabilities in BGP was developed by Murphy in [123]. Recently, the BGP Security Requirements by RPSEC group also documented BGP vulnerabilities [36]. We briefly summarize the key points in these analyses. The nature of the BGP protocol is that the messages a speaker sends are extensions of messages that the speaker earlier received. A compromised speaker can insert false information into the message it sends. If such misinformation is accepted and propagated, network performance may suffer:

- Data may be delayed in its delivery, or even prohibited from being delivered.
- Views of network connectivity may be incorrect.
- Data may be falsely routed through a portion of the network designed to eavesdrop, or even modify the data.
- Fed by false information, the BGP protocol itself may begin to misbehave in such a way that stable routes to networks are not constructed.

There are three primary areas where BGP may be secured that represent different levels of protection.

- The speakers cryptographically secure the BGP data transmitted via BGP sessions, to allow other speakers to verify that the data has been transmitted by indicated speakers indeed.
- The speakers authenticate route information, prefix and AS path in particular, to allow other speakers to cryptographically verify that the route was originated and propagated correctly.
- The originator and the propagators of route information may provide approaches to help other speakers to verify that their policies are correctly preserved.

To ensure the correct behavior of BGP speakers, we have to make sure that speakers learn authentic route information from peers. There are several questions we can ask about the information contained within a received Update [36, 84, 123]. These questions suggest approaches to secure BGP routing.

- *Is the originating AS authorized to propagate the prefix we have received?.*
  This question suggests the need for an authorization mechanism to prove that a certain AS has the authority to originate routes to certain prefixes. Typically, the AS can send some proof that the prefixes are assigned to it by some authority.
- *Is the AS path received via an Update valid?*
  This question indicates several aspects in the route propagation. We divide it into several smaller individual questions:
  - At the lower level, is the protection on BGP sessions strong enough to prevent deliberate change of information by malicious outsiders?
  - At a higher level, can we prevent malicious outsiders from impersonating legitimate speakers? In other words, can we make sure the speakers are authorized to represent certain autonomous systems?
  - In terms of BGP operation, has the AS path been transmitted and extended by transient ASes correctly? A slightly weaker question related to it is: has the Update actually traveled the path?

  - Does the AS path specified actually exist and, based on the AS path, is it possible to traverse that path to reach a given prefix? This last question may suggest some cryptographic mechanisms as well as network engineering approaches.

When it comes to deployment of any BGP security systems, there are a few important issues we need to take care of. One aspect is the performance. Security systems would require time and space overhead above normal BGP operations. Increasing convergence times can adversely affect the usability of the network. Ideally, we expect the BGP's convergence speed with security extension to be equivalent to BGP running without the security system in operation. If a BGP security system is efficient and effective, we might also expect even faster convergence speed, since fraudulent operations are successfully eliminated. However, security adds additional operations that require more time for execution. To minimize the impact on convergence, Christian [139] encourages implementations that implement security to utilize at least one of the following methods for validating routing information:

- The contents of the Update is authenticated at the same time while the Update is processed.
- The RIBs may be authenticated periodically by scanning the data and verifying the route information. This action can be triggered by some pre-defined events.

Security mechanisms may also produce additional data structures. Speakers may spend more space locally to store security related data. They also spend more network resources to transmit the data. The additional amount may exceed local space capacity of speakers. If it is necessary for vendors to add more storage devices to routers, the amount should not be so large that it exceeds the hardware capability.

The security system should also allow incremental deployment, since we cannot expect to deploy a newly secured BGP protocol instantaneously. Thus, with a security system in operation, BGP should allow transmitting, receiving, and acting on both secured and unsecured routing information. Furthermore, the BGP session should be maintained regardless of whether the security system is in place.

2.2.1 Related Research

We briefly discuss the history of routing security research and BGP security mechanisms.

There were several security mechanisms proposed for each individual routing protocol. Since there are two major types of routing algorithms—link-state routing and distance vector routing—security mechanisms for routing protocols naturally fall into two categories. In 1993, Kumar analyzed the security requirements of network routing protocols, and discussed the general measures needed to secure the link-state routing and distance-vector routing protocol classes [89]. He identified two sources of threats to the secure operation of routing protocols in networks. The first source of threats are subverted routers that legitimately participate in a routing protocol. The second source of threats are intruders who may illegally attempt to interfere in routing protocols by masquerading as routers. Since attacks by subverted routers were considered difficult to detect, he focused his attention on securing against intruders, and focused mainly on the links between legitimate routers. For distance vector protocols, he proposed three major countermeasures: neighbor-to-neighbor digital signature of routing updates, the addition of sequence numbers and timestamps to the updates, and the addition of acknowledgments and retransmissions of routing updates. Smith et al. [150] took a similar approach to secure distance vector protocols. In addition to adding sequence information to updates, they also proposed to add predecessor information. The digital signatures on updates covered the destination, predecessor, and sequence information fields. Such a design required one digital signature for the entire routing update. These are one of the first efforts of using public-key cryptography, namely digital signatures, to secure routing operations. However, the protections on distance-vector protocols are less effective for BGP. Earlier in their work, Smith et al. [149] tried a similar approach to secure BGP. Since there are a lot more path information included in routing updates, the protection that Smith could provided was limited.

In 1998, an MD5 signature option was added to TCP that carries an MD5 keyed digest in a TCP segment to enhance security for BGP [70]. Nowadays, MD5 has become the standard option in commercial routers. This technique only provides the point-to-point authentication between peers and the data integrity. It does not protect the routers from being compromised and injecting invalid routing information. BGP updates are still subject to more advanced attacks.

Murphy wrote a series of Internet drafts on BGP vulnerability analysis that provided comprehensive discussion on BGP vulnerabilities from different levels and different types of messages [123]. We echo that the risks in BGP rise from three fundamental vulnerabilities:

1. BGP has no internal mechanisms that provides strong protection of the integrity, freshness and peer entity authenticity of the messages in peer-to-peer communication.

2. No mechanism has been specified within BGP to validate the authority of an AS to announce NLRI information.

3. No mechanism has been specified within BGP to ensure the authentication of the path attributes announced by an AS.

The first fundamental vulnerability motivated the mandated support of the MD5 signature option in the BGP specification, assuming that the MD5 algorithm is secure and that the shared secret is protected and chosen to be difficult to guess. Murphy further proposed a set of protections for BGP security [122]. The idea is to use signatures to authenticate the route announcement at each hop while it is transmitted by en-route speakers. She pointed out that this solution could protect the entire AS path and other path attributes at the cost that it required the validation of a number nested signatures equal to the AS path length in the route. It caused substantial costs on Update message size and validation computation time. The overhead grows as the network size grows.

Kent et al. [84] analyzed vulnerabilities and security requirements associated with BGP and proposed *Secure BGP (S-BGP)* for an authorization and authentication system that addressed most of the security problems associated with BGP. The S-BGP solution consists of three major components. First, there are two Public Key Infrastructures based on X.509v3 certificates. One PKI is designed to enable BGP speakers to validate the identities and authorization of BGP speakers. The second PKI is for allocating IP address space to organizations and ASes. Second, S-BGP uses *attestations* to attest that AS path and NLRI in routes are authenticated. Third, S-BGP uses IPsec [34] to provide protection on peer-to-peer communication. S-BGP is probably the most comprehensive and dominant proposed security system for BGP. It is also an important subject in our performance evaluation study. We present more detailed discussions of S-BGP design and performance issues in Chapter 4. In their later studies [83, 85], Kent et al. discussed the issues on deploying S-BGP in real world. They collected a variety of data sources to analyze S-BGP's performance impacts on BGP processing, transmission bandwidth, and routing table sizes. These studies concluded that the memory requirements of holding route information, necessary public key certificates, and related cryptographic data were a major obstacle for S-BGP deployment.

soBGP [37, 38, 161] is another major security mechanism proposed mainly by Cisco Systems. soBGP started from the beginning by asking the question "Who are you?". soBGP addressed the problem by using an *EntityCert* that tied an AS number to a public key. The EntityCert is signed by a third party. There are a small number of "root keys" are used to validate a set of advertised EntityCerts. Then these certificates are used in turn to build up the database of known good AS number and key bindings in the system. EntityCerts form a web of trust among entities in BGP processing. Next, *AuthCerts* are used to express the authorization of IP address allocation. To authenticate the AS path, soBGP defined an *ASPolicyCert* for an AS to authenticate all its peer relationships. With a database of ASPolicyCerts, a BGP speaker can build a topology map of the paths of the entire Internet. Compared with S-BGP, soBGP's authorization mechanism is more flexible in authenticating AS numbers. However, it has a serious scaling problem because speakers should have a picture of the entire Internet. Moreover, the topology map is against the distributed nature of BGP.

Aiello et al. proposed *Origin Authentication (OA)* [5] to address the BGP origin authentication problem. Instead of standard X.509 public key certificates, OA uses lightweight attestations to express the authorization of ASes to originate route announcements to prefixes. Aiello et al. concluded that since the attestations are much smaller than certificates, OA could allow in-band transmission of proofs of prefix allocation and authorization. We will present OA's detailed design and performance evaluation in Chapter 11.

Realizing that strong public key cryptography and public key infrastructures may face problems of performance and difficulty in deployment, researchers started to consider other approaches. Goodell et al. proposed an *Internet Route Validation (IRV)* service that acts as companion to BGP [55]. IRV defines a service that protects against rogue, subverted, or grossly misconfigured ASes, and is used to identify and diagnose routing configuration problems. IRV was a separate protocol to BGP, so that it can be incrementally deployed. Subramanian et al. [151] proposed the *Listen* and *Whisper* protocols to address the BGP security problem. The Listen protocol helps data forwarding by detecting "incomplete" TCP connections; the Whisper protocol uncovers invalid route announcements by detecting any inconsistency among multiple Update messages originating from a common AS. The Listen and Whisper approach dispenses with the requirement of PKI or a trusted centralized database, and aims for "significantly improved security" rather than "perfect security". Recently, *pretty secure BGP (psBGP)* [159] was proposed to address the practicality issue. Essentially, the solution was to combine the centralized model of authenticating AS numbers with a de-centralized model of authenticating IP prefix ownership and AS paths.

Besides public key cryptography, there are efforts on securing BGP using symmetric key algorithms [56, 74, 165]. These proposals are more efficient on the operational latency, but require more storage, loose time synchronization, and/or complex key-pair pre-distribution.

In this chapter, we reviewed the prior research on the BGP convergence problem and security problem. There were a lot of efforts on understanding the BGP convergence problem, measuring convergence speed, and seeking for new protocol designs and options to improve convergence speed. On the other hand, the focus of designing a security protocol for BGP was not so much on convergence and other performance properties that people are concerned about. In our research, we are interested in understanding the performance impact of security protocols for BGP and looking for more efficient security solutions.

Chapter 3

Simulation Framework for BGP Security

We use simulation to model BGP security for performance evaluation. The model contains an abstraction of security protocols, a network topology model, and a protocol behavior monitoring facility. We present details of the simulation model in this chapter. Some of the text is adopted from our research papers [126, 166].

3.1 Choice of Simulation

Previous empirical studies have shown that BGP converges considerably more slowly than expected [90]. Studies of S-BGP also have indicated that adding security to may also put on unexpected additional burden to BGP processing, network bandwidth utilization, and routing table size [83, 85]. However, the empirical approach has considerable limitations. Empirical studies must work in the Internet as it is now. The Internet cannot be easily modified to accommodate a study with a variety of security systems with varying parameters. As noted in [92], experimenting with even a single smaller parameter may have significant adverse effects.

A traditional approach for performance evaluation of security protocols is the ***testbed*** or prototype, typically a closed network environment with a real implementation of the protocol and associated monitoring and measurement facilities. Experimenting with a *small* testbed may produce useful results. However, in our study, we are interested in the connection of BGP security and slow convergence, which would not be shown unless the scale of the network is large enough. We may need a large testbed for our study. Recently, there are efforts to construct global scale testbeds for network protocol analysis, such as the PlanetLab testbed [136]. PlanetLab is an open-environment testbed, where it connects to the real Internet. However, in our evaluation work, some of the BGP experimentation may cause severe routing instability. We would prefer a closed environment such that such effects would be controlled and would not affect normal routing activities on the Internet.

A theoretical approach to analyze security systems is *formal methods.* According to the definition in [162] formal methods involve a formal language, formulas specifying the system, and reasoning about the formulas. One popular method that falls into this category is the *state exploration method.* It models the security protocol using a finite state system. Then we can apply model checking technology to search all reachable states exhaustively. The goal is prove that all reachable states are safe. Although it is a good tool for automated verification of correct properties of many critical systems, keeping the state space small requires drastically simplified assumptions. Because of the complexity of networks such as the Internet, theoretical models are inevitably highly simplified in order to make the mathematics tractable.

Simulation stands in the middle ground between these approaches. We use simulation to implement experiments with fewer simplifications than theoretical model checking would require, more flexibility than is feasible in empirical studies, and larger scale than is provided by testbeds. It is true that a simulation will inevitably be a simplified model of a network in some respects, and simulation may not be able to produce as neat results as those by theoretical studies. Simulation does provide us a field in which we may experiment with a rich set of network behaviors. We kept in mind, however, that simplification decisions need to be made carefully and intelligently.

3.2 BGP Simulation on SSFNet

The simulator used for this study was SSFNet [29, 127]. It contains the BGP module with detailed BGP implementation [137]. It was a natural choice since SSFNet was the only network simulator with BGP implementation[1]. More importantly, our study was focused on large-scale networks. One of the primary goals of SSFNet is scalability. Simulated protocols can function efficiently even in very large simulated networks. SSFNet has been used in a number of other BGP studies [102, 59, 131, 170, 118]. We are convinced that SSFNet is the right choice.

SSFNet is a Java-based discrete-event simulation [8, 94] package. It is capable of modeling large complex networks. SSFNet performs simulation at the IP packet level. Hosts and network links are simulated. Each host may contain multiple protocols, organized as a protocol graph. The supported protocols include IP, TCP, UDP, BGP, OCSP, HTTP, etc. *Domain Modeling Language (DML)* specifications are used to configure the network topology and protocol attributes.

SSFNet's BGP module not only implements BGP specifications in detail, it also provides more than 50 switches in DML for configuring BGP behaviors, recording events, and reporting routing tables. Among the rich set of options for BGP behavior, we used the defaults as defined in specifications and used in practice, such as

- the policy for selecting a path is shortest-number-of-hops,
- no router is a reflector,
- no route aggregation is performed, and
- default timer values are used.

[1]PDNS [142] recently started to include BGP in the ns2 simulator, which was after we started our study.

3.3 Models for Security Mechanisms

We added modules on top of BGP to let the simulator understand security mechanisms. Since our major focus was on performance, mainly on time and space overhead, we decided to simplify the security mechanisms in the simulation while preserving their characteristics in terms of performance. From this perspective, the detailed security related data are not important. Immediately, we can discard some details in the data structures, such as the public keys and private keys, certificates, signatures, hash values, etc. However, we preserve three major properties for each security mechanism:

- the types of security data structures associated with route announcements,
- the data size of each data structure type, and
- the execution delays for each security-related operation, such as signature generation, verification, hashing, and table look-ups.

Message Size

We model the necessary increase in data size associated with route announcements. For instance, if one route announcement requires k signatures being transmitted with the route, we model the Update message size to increase by $k \cdot \sigma$ bytes, where σ is the size of one signature. We model it as the virtual message size. That is, the system does not allocate the data with its modeled size. The "size" attribute of the data is used by SSFNet to compute the time overhead to transmit the data over the simulated network. On the lower level, the transmission of an IP packet is done by passing the reference to another object in Java. At the same time, certain simulation time will be charged to reflect the network delay of sending this packet. The actual time depends on the data size, simulated network link delay, and link quality.

Space Module

The original BGP module in SSFNet does not contain a module to model the local space on BGP routers. We added one for the purpose of our study. Previous studies have shown that memory and storage requirements became the major hurdle for deploying S-BGP in the Internet [83, 85]. In fact, the problem is not unique to S-BGP. Given the scale of the Internet, the ever-increasing routing table size is already a big burden on the BGP routers. Any additional security data would exacerbate the situation. Most of the BGP security mechanisms proposed to use local disk or memory as the cache to store generated and/or received security data to avoid redundant operations. None of them actually took efforts to measure how many redundant operations are reduced or how much storage was needed for this purpose.

In the space module, routers cache two types of information:

- data related to route announcements in Update messages sent or received, such as signatures, hash values, attestations, and any other information accompanying route announcements, and

- data related to key distribution, such as secret keys, or certificates being distributed to routers out-of-band.

Depending on the design of the individual security mechanism, we record the increase of cache usage when the routers take action on caching a new piece of information. The BGP module in SSFNet considers the Loc-RIB together with all Adj-RIBs-Ins as the routing table. Routing table size is measured as the total size of routes stored in all these RIBs. With a security mechanism in operation, we consider the resulting routing table size to be the size of the original table size plus any additional security related data size to routes in the routing table.

Processing Delays

In a typical BGP simulation configuration, there is only one advertised destination by a simulated speaker. To compensate for this simplicity, the BGP module has a facility to estimate and impose additional synthetic router workload. The workload of a router is a measure of the amount of processing demands being imposed upon it. Common contributors to router workload are inter-domain and intra-domain routing protocol tasks, traffic forwarding, and SNMP message request handling. If the router workload is high, one might expect it to take longer to process a simple BGP message than it would take to process the same message if the workload was low. The router workload facility defines a *workload-induced delay* parameter. This delay is imposed on a per-update message basis, and is intended to capture the amount of time required to process that Update plus some additional time to account for other processing taking place in the router that we are not explicitly modeling.

The workload facility defines the BGP processing flows using a *message sender* and a number of buffers, including several *incoming buffers*, an *event handler*, and an *outgoing message buffer*. Any incoming Update message will go through this control follow. If the processing of this Update message results in new Update messages to be sent out, new messages are placed in outgoing buffers. Finally, when triggered by the MRAI timer, the message sender goes through outgoing message buffers and sends out Update messages. This design provides several places where we can impose delays related to BGP workload. First, the BGP speaker waits for some time before putting an incoming message into an incoming buffer. Next, the event handler spends some time on processing a message. Lastly, it costs some time for the message sender to send out a message.

In modeling additional delays introduced by a security mechanism, we follow the same philosophy. Figure 3.1 demonstrates our addition to the original workload facility. We borrow the diagram of the facility structure from [138]. We use dash-bounded boxes to illustrate our additions. They represent appropriate operations and their latencies introduced by a security mechanism. The implementation of these boxes may be different when it comes to modeling a specific security mechanism. We will discuss more details in later chapters.

Depending on the security mechanism, there could be two places to impose the delay of verifying an Update message. Either the router can verify every route announcement from incoming Update messages, or the router can delay the verification until it decides to insert the new route into Loc-RIB. We call the latter approach *lazy verification.* The advantage of lazy verification is that routers may save time on verification, since only a small fraction

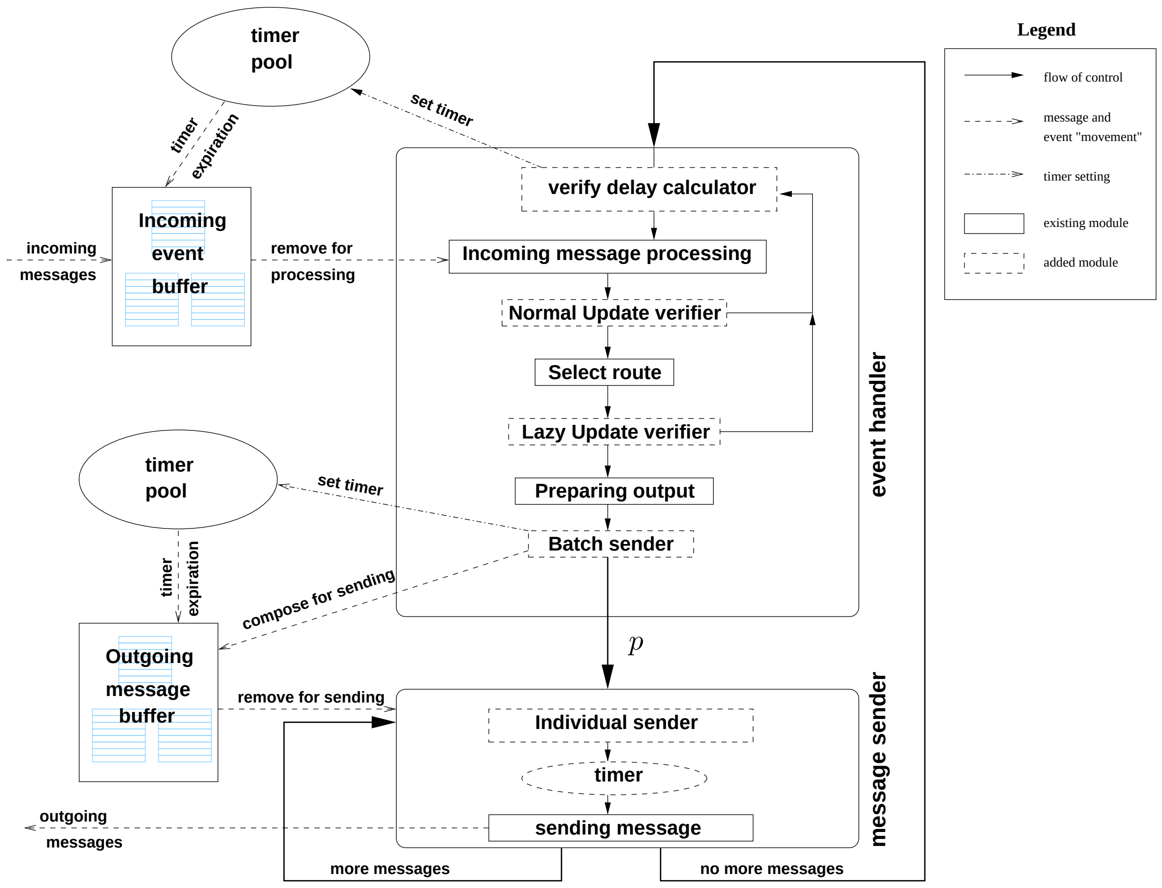


Figure 3.1: Simplified flow control model for BGP Update processing with a security system. In a simulation, processing delay p is imposed after an event is handled, but before outgoing messages are sent. We borrow the diagram of the facility structure from [138] and illustrate our additions using the dash-bounded boxes.

of received routes are selected as preferred routes and installed in Loc-RIB. Nonetheless, to reflect the choices, we put two verifiers in the event handler and a configuration switch to choose between them.

Similarly, there are two places to impose the delay of sending an Update message. Some security mechanisms process individual Update messages in the sending process and add related cryptographic data to the outgoing route announcements. We use *individual sender* to capture the necessary delays by cryptographic operations, such as hashing, encryption, and signing. Other security mechanisms may choose to cryptographically process outgoing Update messages in a batch. We design *Batch sender* to model this function. The batch sender models the time latencies by cryptographic operations while messages are still in the output buffers. With batch sender in effect, individual sender will not charge additional latencies related to signature generation.

3.4 Model Configuration

3.4.1 Benchmarks for Cryptographic Overhead

There are operational details involved in each type of cryptographic operations for a BGP security system. We simplified the simulation model by only taking into account their performance overhead instead of actual implementations. This is the task of *benchmarking.* In this section, we only present the basic methodology used to obtain benchmarks. The actual numbers will be presented when it comes to modeling and experimenting with an actual security mechanism.

There are two types of overhead introduced by cryptographic operations or security operations—space and time.

Benchmarking the space overhead is relatively easy. For hashing functions and signature algorithms, the resulting data size can be determined by definition. In other words, given the key size or the length of parameters, we immediately can derive the resulting size of hash values and signatures. Notice that security systems define new data structures to hold cryptographic data, which includes not only the hash value or signature itself, but also necessary data header. Unfortunately, not all of the security mechanisms in our evaluation study have a real implementation or prototype. Hence, for fair comparison, we only focus on the space used by cryptographic data.

For other types of data, such as attestations and certificates, we sought help from their designers. Typically, the designers of a security mechanism provided the measurement of related data structure in their studies.

To learn the running times, we combine the efforts from four sources. For most of the commonly used operations, we use public crypto libraries, such as OpenSSL [129] to understand their running times. Using the same crypto library allows us to provide fair comparison of the implementations.

For newly proposed algorithms, popular crypto libraries may not contain an implementation. For these, we have tried two approaches. One choice is to decompose the operations in the algorithm, benchmark the running time of each step, and sum them up. This strategy works well for some of the algorithms, such as DSA signing with pre-computation. We tried to stay with the same crypto library as the common basis for comparison. However, when some of the primitives in the algorithm are not supported by the same library, we had no choice but to switch to other specialized cryptographic library or the designers' own prototypes.

Essentially, what we care about is the necessary CPU cycles involved to finish an algorithm. We are careful to make sure that all the benchmarks of running time in terms of seconds are normalized to the same processor speed. The speed we chose is 200MHz, a typical processor speed for average BGP edge routers.

Lastly, we take care of the time overheads by a protocol or by network latency. We cannot break such latencies down into pieces and measure the time of each piece precisely. Rather, we treat the protocol as a black box. By measuring real implementations, we estimate the range of time for one operation. For example, we measured the Update message processing latency on the BGP router on campus at Dartmouth College. We concluded that, typically, it cost 65 to 115 milliseconds to finish processing one Update message.

3.4.2 Topology Configuration

Throughout this study, we evaluate security mechanisms in the same network topology setting. We use a 110-AS topology, with one operating BGP speaker per AS. For modeling simplicity, each BGP speaker announces two prefixes. In our model, each AS also possesses *virtual BGP speakers* that don't actually run a simulated BGP protocol. We use the number of such BGP speakers to represent the size of an AS; its size affects the time it takes for one Update message to be propagated through an AS.

Topology

We use the public data provided by RouteViews project [145] to generate a graph of AS connectivity of the Internet, then reduce the size to 110 ASes using a collapsing procedure. We used a heuristic procedure to create the topology. The resulting topology preserves the distribution of connectivity in the interior of the Internet. We start with a large AS

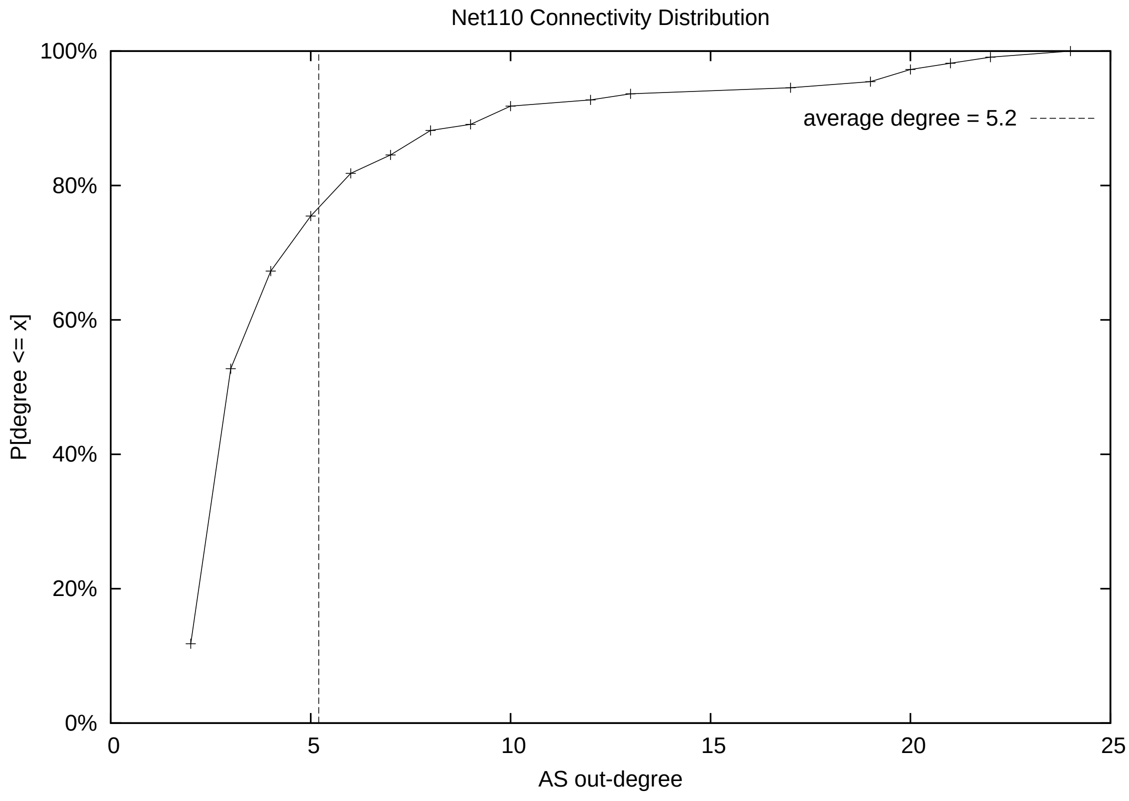


Figure 3.2: Degree Distribution of 110 AS system

topology built from Internet measurements. Then we heuristically merge nodes to reduce the graph, reduce the graph further by randomly removing edges and retaining the largest connected component, and then merging low degree nodes some more. This reduced graph still preserves certain macroscopic properties [44] seen on the entire Internet. The resulting 110-AS topology exhibits the heavy-tailed distribution of connectivity seen in the real Internet. The cumulative distribution function of the 110-AS model is illustrated in Figure 3.2. While the average degree is 5.2, nearly 80% of ASes have degree smaller than the average, while 10% have degree larger than 10. The core of the Internet is a highly connected place.

Policy Filtering

Business practices in the Internet today dictate certain types of relationships between the administrations of different autonomous systems. Whenever two ASes decide to exchange traffic by connecting their BGP routers together, they must establish policy for the peering session. We briefly discuss three basic types of peering relationships and present how we reflect such relationships in the topology configuration.

In a *customer-provider* relationship, one AS, acting as the customer, is provided with access to the rest of the Internet by another AS (the provider). In order to make the Internet reachable by the customer, the provider will advertise to the customer any routes it learns from any other AS, as well as any routes originated by the provider itself. On the other hand, the customer only needs to advertise to the provider any routes that the customer itself originates. To make the customer reachable from anywhere else in the Internet, the provider should advertise those routes, learned from the customer, to all other peer ASes.

In a *peer-to-peer* relationship, two ASes agree to share traffic between their customers, where an AS must advertise to the peer all of its customers' routes as well as routes originated by itself. However, the AS will not advertise to the peer any routes learned from other peers or from providers or siblings. A peer-to-peer relationship is useful to shorten the lengths of the routing paths between each others' customers.

In a *sibling-to-sibling* relationship, each AS provides the other with access to all of the routes known to it. This type of relationship is most common in the backbone of the Internet, where the Tier 1 providers form an densely interconnected network.

Gao and Rexford have done some work [50, 52] to estimate peering relationships on the Internet, using only the connectivity topology at the AS level. Their analysis results largely matched the privately collected data. Their methodology provides nice tips for us to model more realistic AS-level topologies for our BGP study. Here, we use a similar approach to configure the peering relationships for the 110-AS topology.

- Find ASes with largest out-degrees that form a densely interconnected core, such as a complete graph (or close to complete). Edges in the core are assigned a sibling-to-sibling relationship. There should be no more than 15 ASes in this set, which is roughly the backbone size of the current Internet.

- All ASes that are connected to the core are considered the second tier. We assign a customer-provider relationship on the edges that connect the core and the second tier. All edges within this tier are assigned a peer-to-peer relationship.

- This process continues as a new tier gets a customer-provider relationship assigned to its upper tier. The procedure stops when all edges in the topology are assigned.

3.4.3 Routing Activities

In the Internet, several types of activities can trigger a wave of Update messages. For instance, a router can originate a new route announcement and inject it into its Loc-RIB, or withdraw a prefix that it announced earlier. In our performance study, we are more interested in one scenario: *router rebooting from a crash.* The volume of workload is much higher, because a rebooting router will get table dumps—an announcement for each prefix, from each of its peers, after re-establishing sessions with neighbors. This router will announce its own preferences for prefixes as it is processing these incoming Updates.

Previous studies have shown that global routing stability is very sensitive to workload on the routers. Cowie et al. [30] studied the correlation of BGP Update spikes with CodeRed II and Nimda worm propagation, and concluded that worm attacks are responsible for the sudden dramatic increase of Update messages by routers. It is suspected that extra Update messages are caused by link failures. Excessive network traffic by worms create these link failures. Wang et al. [160] further pointed out that excessive session resets at the monitoring point contributed most to the Update spikes and a substantial amount of BGP Updates do not reflect AS-level topology changes. BGP behaves quite differently when under stress.

Our experimental scenario design reflects their observations. The goal is to understand the BGP behavior in the worst case with highest workload and the performance impacts by a security mechanism. To maximize the effect, we choose to reboot the router with largest out-degree.

3.5 Measurements

Given one execution of the SSFNet model with a set of configurations, we record several measurements. Table 3.1 summarizes the performance metrics in time and space. We ran each experiment 20 times, and computed the mean value of a variety of measures. It turned out that the ratio of standard deviation to mean is less than 5% throughout, and so for our purpose it suffices to report the means.

Metrics	**Meaning**
#Anns	Total number of Route Announcements that occurred
#Updates	Total number of Update messages that occurred
#verif.	Total number of signature verification operations
#sign	Total number of signature generation operations
basic CPU	Total CPU time on basic BGP operations (in seconds)
crypto CPU	Total CPU time on security operations (in seconds)
convergence	System convergence time (in seconds)
Update message size	includes basic Update message and security data (in bytes)
Cache size	cache for routes and related security data (in bytes)

Table 3.1: Measurement metrics in time and space.

The total number of Updates is the sum of number of advertisements and withdrawals by all routers. Similarly, other numbers are defined as total counts by all routers. The counts of cryptographic operations may not be directly proportional to the execution time, caused by different signature algorithms. Hence, we also measure CPU time as the absolute time spent by all routers on basic BGP operations as well as on security related operations. Crypto CPU compared with basic CPU may give some idea of relative efficiency of a security mechanism.

Ultimately, the most important metric we use to measure the efficiency of a security mechanism is convergence time. We define the convergence time for a given experiment phase to be the total amount of simulation time that elapsed from the time that the crashed router started to reboot to the earliest time after which no router spends any more time processing Update messages that resulted from the originate rebooting event. This definition indicates that the router has not converged until it processes all the incoming Update messages even though these messages do not cause new route selection.

For space, we measure both the message size and the storage cost in memory. For simulation, we assume a simple form of Update messages. In other words, we measure the bytes for basic Update message fields and bytes for additional data structures, such as signatures and hash values. Note that the current MTU limitation on an Update message is 4096 bytes. In experiments, we relax this limitation. The experiments report both average and maximum message sizes for us to understand the efficacy of different options.

In simulation, we assume that routers store data in memory as much as possible. Since we do not set memory capability in simulation, we just simply assume that all caches are held in memory. Thus, the memory cost is equal to the cache usage size.

Chapter 4

Design of Secure BGP

We use the simulation framework of BGP security to evaluate the performance of Secure BGP (S-BGP). As we have introduced in Chapter 2, S-BGP is a secure, scalable, deployable architecture for an authorization and authentication system that addresses most of the security problems associated with BGP. It is currently one of the primary security proposals for BGP. S-BGP proposes countermeasures for securing BGP route distribution involving two Public Key Infrastructures, a new path attribute containing "attestations", and the use of IPsec. Our evaluation study focuses on the performance impacts by PKIs and the attestations. In this chapter, we discuss the design details of S-BGP. We will present performance evaluation on each S-BGP component in later chapters. Chapter 5 and Chapter 6 present our studies on evaluating performance of attestations and Chapter 11 discusses PKI impact on BGP performance.

4.1 Overview

Secure BGP (S-BGP) [84] focuses on authorization and authentication of route distribution. As we have introduced in Chapter 2, the S-BGP countermeasures consists of three components: PKIs, attestations, and IPsec. The first two components are our focus. First, S-BGP sets up public key infrastructures to help establish the authenticity of the involved parties. S-BGP uses X.509 [71] public key certificates and puts BGP-related information into certificate extensions. Second, speakers digitally sign the routes they announce to peers; with these X.509 certificates, recipients can verify the signatures to authenticate the received routes. Generated signatures and other information for route authentication are encapsulated in *attestations*. More specifically, each speaker uses *address attestations (AAs)* for authenticating prefixes in the route announcements, and *route attestations (RAs)* for authenticating the AS path. Overall, with the countermeasures, the BGP speakers can achieve two basic goals:

- validating the authenticity and data integrity of BGP Updates that it receives, and
- verifying the identity and authorization of the senders.

According to S-BGP designers, there are eight features needed for correctness of BGP operations [84]. We quote these features as the following:

- *Each Update received by a speaker from a peer was sent by the indicated peer, was not modified en route from the peer, and contains routing information no less recent than the routing information previously received for the indicated prefixes from that peer.*
- *The Update was intended for receipt by the peer that received it.*
- *The peer that sent the Update was authorized to act on behalf of its AS to advertise the routing information contained within the Update to BGP peers in the recipient AS.*
- *The owner of an address space corresponding to a reachable prefix advertised in an Update was authorized by its parent organization to own that address space.*
- *The first AS in the route was authorized, by the owners of the address space corresponding to the set of reachable prefixes, to advertise those prefixes.*
- *If the Update indicates a withdrawn route, then the peer withdrawing the route was a legitimate advertiser for that route, prior to its withdrawal.*
- *The peer that sent the Update correctly applied the BGP rules and its AS's routing policies in modifying, storing, and distributing the Update, in selecting the route, and in deriving forwarding information it.*
- *The BGP speaker that received the Update correctly applied the BGP rules and its AS's routing policies in determining whether to accept the Update.*

The S-BGP security architecture meets six major criteria for primary correct operation features of BGP. The local policies by BGP speakers and ASes are not typically visible outside an AS. The countermeasures cannot meet the last two criteria since the speakers cannot confirm how other speakers follow their local policies on how to process Update messages.

4.2 IPsec and Router Authentication

The BGP MD5 option was defined to protect data integrity between peer-to-peer communications. However, the protection is very limited and there is no prescribed key management scheme. S-BGP proposed to use the *Encapsulating Security Payload (ESP)* protocol from the IP security protocol suite (IPsec) to provide authentication, data integrity, and anti-replay on a point-to-point basis. The *Internet Key Exchange (IKE)* protocol together with S-BGP's certificates (we will discuss them shortly) is used for key management service in support of ESP.

4.3 S-BGP PKIs

S-BGP uses two PKIs to enable BGP speakers to validate the identities and authorization of BGP speakers and of owners of ASes and of portions of the IP address space. These PKIs mirror existing infrastructure. The first PKI contains certificates to authenticate the *owners*

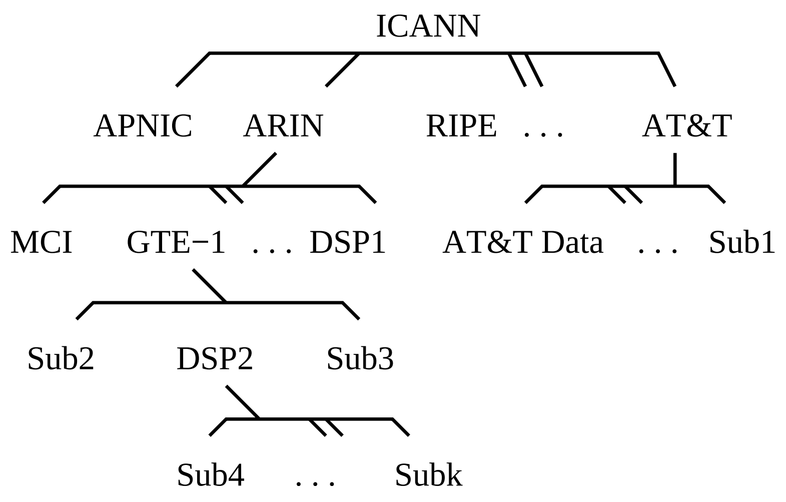


Figure 4.1: Sketch of the S-BGP PKI for IP address allocation. The root of the hierarchy is ICANN [80]. The next tier are the large organizations to whom ICANN assigns a large portion of IP addresses. Subsets of their IP addresses are further assigned to smaller organizations.

of portions of the IP address space. The second PKI is to authenticate BGP speakers, ASes, and the owners of ASes. Figures 4.1 and 4.2 illustrate the structures of these PKIs.

Both PKIs are hierarchies rooted at ICANN [80]. ICANN issues itself self-signed certificates and further issues certificates to the the first tier of organizations, typically *Regional Internet Registries (RIRs)* such as ARIN, RIPE, APNIC, and LACNIC. Note that the organizations that assign addresses and the organizations that obtain autonomous system numbers from a Registry may be different. An organization could receive its AS number from a registry and its address block from an ISP.

A PKI for Address Allocation

IP address allocation certificates specify the IP address assignment by organizations. An issuer can allocate a list of prefixes to the same organization using a single certificate. We can express the address allocation using the following form:

$$Org_x \rightarrow (Org_y, p_1, p_2, \ldots, p_k) \ .$$

In the address allocation PKI, ICANN issues itself a certificate claiming the ownership of the entire IP address space on the Internet. Consequently, it issues certificates to RIRs as it assigns IP address blocks to them. The certificate contains an extension that specifies the set of address blocks ICANN is allocating to that RIR. Each RIR further assigns portions of its address blocks and issues corresponding certificates to the third tier organizations of the hierarchy. The process continues until it reaches end subscribers. A typical certification path for an address block is similar to the following:

“ICANN→Registry→ISP/DSP. . .→ISP/DSP→Subscriber”.

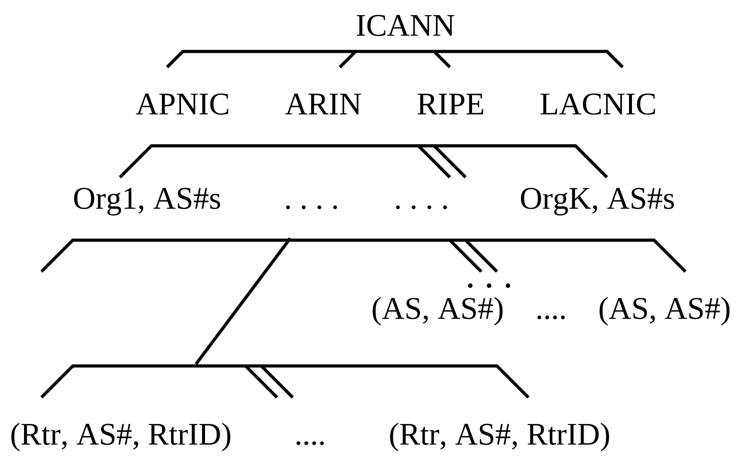


Figure 4.2: Sketch of the S-BGP PKI for AS numbers and routers. A certificate for AS number binds an AS with its public key and AS number. A certificate for router binds the router ID with the router's public key and the AS that the router is authorized to speak for.

In some cases, the system does not require that address assignments be certified all the way to the subscriber. If a subscriber's address is allocated from that of a DSP or an ISP with which it is currently affiliated, then the certification process need only be effective as far as the ISP/DSP.

A PKI for Assignment of ASes and Router Associations

The second PKI contains certificates for AS number assignments, as well as identity certificates of organizations, ASes, and BGP speakers. The AS number authentication is similar to address allocation authentication. At the top, ICANN has another self-signed certificate. It assigns AS numbers to RIRs. Then, each RIR assigns some of its AS numbers and issues certificates to the third tier organizations (also called AS owners). These AS owners, in turn, issue certificates for authenticated ASes. At each level, authorities agree to comply the certification policies ensure that all AS number assignments are distinct. AS owners also issue certificates for BGP speakers; each such certificate binds the router name to an AS number and router ID, testifying that the speaker is authorized to represent a certain AS. At the fourth tier of the hierarchy, there are typically several router certificates, each specifying the same AS number with a different router ID. Typical certification paths in AS number and BGP speaker identification PKI are as follows:

"ICANN$\rightarrow$Registry$\rightarrow$AS owners$\rightarrow$AS numbers"
"ICANN$\rightarrow$Registry$\rightarrow$ISP/DSP… $\rightarrow$BGP speakers".

S-BGP calls for out-of-band approaches to distribute certificates. Three reasons caused this decision. First, BGP Updates are limited in length of 4096 bytes. Certificates are relatively large. Second, certificates with each Update would often be redundant and wasteful of bandwidth. Third, certificates are often relatively stable; they can stay unchanged for

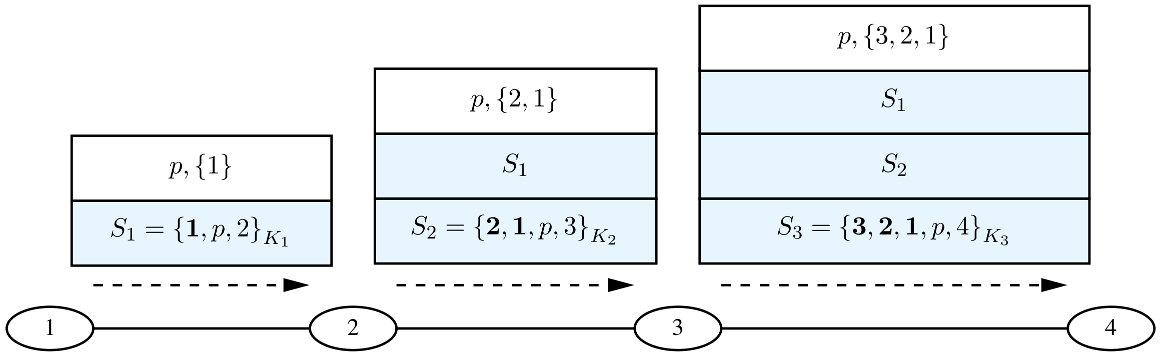

$p, \{1\}$
$S_1 = \{\mathbf{1}, p, 2\}_{K_1}$
$p, \{2, 1\}$
S_1
$S_2 = \{\mathbf{2}, \mathbf{1}, p, 3\}_{K_2}$
$p, \{3, 2, 1\}$
S_1
S_2
$S_3 = \{\mathbf{3}, \mathbf{2}, \mathbf{1}, p, 4\}_{K_3}$
1
2
3
4

Figure 4.3: This figure sketches the process of sending route announcements and their route attestations. We have four ASes numbered as `1`, `2`, `3`, and `4`. The figure shows the AS path components in bold.

months or maybe a year. The out-of-band distribution relies on ISPs/DSPs to provide local access to certificates for the speakers within their respective ASes. ISPs/DSPs and subscriber organizations are responsible for validating certificates, e.g., by an administrator for an AS. Thus routers can save the time and effort of doing this work. Then, validated certificates are "pushed" to routers to help them validate attestations on routes.

4.4 Address Attestations

An attestation establishes that the subject of the attestation (an AS) is authorized by the issuer to advertise a path to the specified blocks of address space. An *Address attestation (AA)* expresses that an AS is authorized by an organization Org_x (the signer of the AA) to announce certain blocks of IP address space. BGP speakers use AAs together with corresponding address allocation certificates to ensure that the origin AS in the route announcement is authorized to originate routes to the IP prefixes.

Similar to certificates, AAs are typically distributed out-of-band for the same reasons. If an organization has more than one AS, there are separate attestations for each AS.

4.5 Route Attestations

A *route attestation (RA)* is signed by a BGP speaker to authenticate the existence and position of an AS number in an AS path [84]. Figure 4.3 demonstrates the structure of RAs. Such attestation is nested: each BGP speaker signs the AS path in sequence, as it joins. That is, first the origin BGP speaker signs the AS number of the origin autonomous system and the intended receiver (in the form of AS number). The next signer is the receiver of this RA; it computes and signs the concatenation of the new AS path, the prefix, and intended receiver. The process goes on until the entire AS path is signed.

Figure 4.3 gives an example of creating and distributing RAs. `AS 1` initiates the process by sending a route announcement with AS path {`1`} to prefix `p`. We simplify the form of the route announcement as the tuple `(p, {1})`. This route states that `AS 1` owns prefix p and it

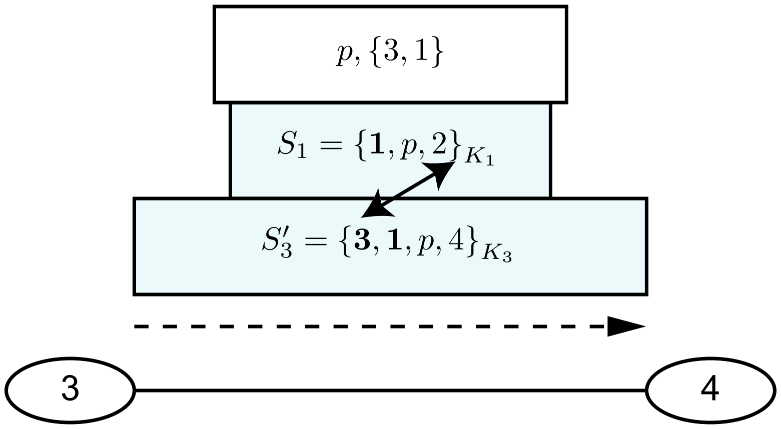

$p, \{3, 1\}$
$S_1 = \{\mathbf{1}, p, 2\}_{K_1}$
$S'_3 = \{\mathbf{3}, \mathbf{1}, p, 4\}_{K_3}$
3
4

Figure 4.4: This figure sketches how S-BGP would stop an attempt by `AS 3` to forge a route announcement.

is reachable via this speaker. The corresponding speaker generates the RA by signing {`1, p, 2`} using its private key K_1. It puts its AS number as the AS path first, then the prefix, then the intended recipient. The other ASes continue this process by signing the new AS path, the prefix, and the intended recipient. Each new RA is sent together with all previous ones, so that they are effectively chained together.

The inclusion of the intended recipient and the prefix in each signature is necessary to prevent "cut-and-paste" attacks. To continue the earlier example, consider Figure 4.4. `AS 3` is not able use the attestations it has received to forge an attestation for route `(p, {1,3})` that `AS 4` will accept. In other words, `AS 3` is trying to strip away `2` and fool `AS 4` into believing a fraudulent, more attractive 2-hop route. To do so, `AS 3` would need a signed statement from `AS 1` offering to route information to p directly from `AS 3`. However, since the signed link that `AS 3` has from `AS 1` explicitly specifies that `AS 1` links to `AS 2`, not `AS 3`, `AS 4` will detect the forgery.

The validation of RAs is also facilitated by corresponding authentication certificates of AS numbers and BGP speakers as well.

4.6 Performance Optimization

Route Validation

Route validation needs both attestations and certificates. As discussed in [84], to validate a route $(p, \{AS_n, AS_{n-1}, \ldots, AS_1\})$ received from AS_n, AS_{n+1} needs

- 1 AA from each organization owning an address block or blocks in the NLRI,
- 1 address allocation certificate from each organization owning an address block or blocks in the NLRI,
- 1 AA from every speaker along the AS path (AS_n to AS_1) where the RA generated and signed by router$_x$ specifies the NLRI and the AS path from AS_{x+1} through AS_1,

and

- 1 certificate for each speaker along the path to check the signatures on the RAs.

Optimizations

The primary performance issues of S-BGP countermeasures come from route attestations.

First, RAs consume *space*. The requirement that a route announcement with AS path of length k should carry k RAs in the message significantly increases the *message size*. This increase can threaten the Update MTU; this increase also increases network bandwidth usage.

Second, RAs consume extra CPU cycles for processing. This is a two-fold effect. On generating RAs, because of the inclusion of the intended recipient in RAs, the speaker signs the same route n times if it is sent to n peers. The signing operations explode easily as more Update messages get generated. On validating RAs, a route with an AS path of length k requires k signature verification operations.

To improve the performance, S-BGP calls for several optimizations:

DSA S-BGP uses DSA as the digital signature algorithm to create attestations. The resulting signature size can be potentially much shorter than those by other popular digital signature algorithms with equivalent levels of security. For instance, 1024-bit key DSA produces 40-byte signatures, while 1024-bit RSA generates 128-byte signatures. This optimization is intended to mitigate the space problem.

DSA with pre-computation DSA signing with pre-computation may be applied to speed up the signing process. In DSA signing algorithm, several steps of operations are not related to the message to be signed. Hence intermediate data can be computed in advance, and be stored securely for later usage [107]. Since most CPU intensive computations are done with pre-computation steps, signing with pre-computed data is much faster than the original signing operation. The purpose of this optimization is to speed up the signing process. We use "pDSA" as the notation for DSA with pre-computation in our discussion. Quickly searching through the literature, I found that most standard crypto libraries do not provide pDSA.

Lazy verification S-BGP uses a lazy strategy when validating routes. A speaker verifies attestations only when it is about to install the route into Loc-RIB. In other words, signatures only get verified when the corresponding route is selected as a preferred route by a speaker. This strategy can potentially reduce the number of necessary signature verification operations.

Caching To avoid redundant operations, S-BGP calls for local caches to store validated and sent routes. The purpose is still to reduce the necessary number of cryptographic operations. In exchange, speakers use more space in local cache to store routes, which may become another performance problem. To allow efficient operations with caches, the routers try to hold as much data as possible in memory. Thus, additional caches increase memory cost.

Non-volatile storage In some circumstances, the local caching in memory may not be enough to solve the redundant route validation problem. For instance, when a speaker crashes, all information in memory is gone. The speaker still needs to validate all routes from table dumps after it reboots. An optimization is to apply non-volatile storage, so that validated routes are preserved in cache across router reboots. During the reboot process, the speaker receives a huge amount of Update messages from its peers. Non-volatile storage can help reduce the large number of verification operations performed by the rebooting speaker.

Chapter 5

Performance of S-BGP Route Attestations

Path authentication schemes refer to general approaches to BGP route authentication that meet the following three criteria:

- Each Update received by a speaker from a peer was sent by the indicated peer, was not modified en route from the peer, and contains routing information no less recent than the routing information previously received for the indicated prefixes from that peer.
- The Update was intended for receipt by the peer that received it.
- The peer that sent the Update was authorized to act on behalf of its AS to advertise the routing information contained within the Update to BGP peers in the recipient AS.

In other words, path authentication focuses on the AS path of the route announcement.

S-BGP route attestations are one example of a path authentication scheme. We evaluate performance impact by S-BGP route attestations on BGP behavior in this chapter. We use the general security model we have set up in Chapter 3 and further configure it to model S-BGP RAs. This chapter presents our simulation configurations and experimental results. The performance of PKIs will be presented in Chapter 11 and performance by address attestation will be discussed and compared with another security mechanism, Origin Authentication (OA), in Chapter 6.

5.1 Configuration and Benchmarks

Using the simulation model presented in Chapter 3, we model the complete process of generation and processing of attestations. In the configurations, there are several parameters as illustrated in Table 5.1. Some of them are set as default, while others vary for each set of experiments. Throughout all experiments, S-BGP uses SHA-1 as hash function and always applies lazy verification to reduce number of verifications. If a speaker uses caching optimization, the contents in the cache persist across router reboots.

parameter	**values**	**default**
signature algorithm	DSA/pDSA	DSA
hashing algorithm	SHA-1	SHA-1*
lazy verification	true/false	true*
caching	true/false	false
non-volatile storage	true/false	true*

Table 5.1: Configuration parameters for evaluation experiments on S-BGP.

parameter	**value**
DSA verify	31.0ms
DSA sign	25.5ms
pDSA verify	31.0ms
pDSA sign	0.015ms
Signature length	40 bytes
Hash length	20 bytes

Data Size (bytes)	1-56	57-64	65-120	121-128
SHA-1 Time (μs)	7.87	13.71	12.90	18.82

Table 5.2: Constants and benchmarks used for simulation. DSA and pDSA are based on 1024-bit keys with 160-bit modulus. The time needed for hashing is proportional to the length of the data size, stepping up linearly in accordance with SHA-1 construction.

Another set of configurations to the model are the benchmarks. The operation delays are obtained using OpenSSL library. Numbers are normalized to a 200MHz processor, a typical configuration for ordinary BGP routers. As discussed in Section 3.4.1, since OpenSSL does not support pre-computation of DSA signing we apply the decomposition for benchmark. Table 5.2 demonstrates the running times by DSA operations with or without pre-computation.

The running time of hashing data using SHA-1 depends on the data size. Table 5.2 also illustrates that the time needed for hashing is proportional to the length of data size, stepping up linearly in accordance with the SHA-1 construction.

It is easy to determine the unit data length once the signature and hashing algorithms are chosen. Table 5.2 presents the signature length and hash length.

5.2 Performance of Route Attestations

We evaluate performance by comparison. In this section, S-BGP's performance is compared with BGP without a security system in operation. We defer our presentation of performance of address attestations to the next chapter when we compare them with Origin Authentication scheme.

Table 5.3 illustrates the performance of BGP with no security mechanisms. During the rebooting process, speakers exchange a large number of advertisements, as we expected. The rebooting router re-establishes 24 BGP sessions with its peers. The table dumps produce

metrics	**value**
#Updates	20334.7
#Ads	19571.8
basic CPU	1310.6 seconds
convergence	153.7 seconds
Average message size	36.1 bytes
memory	9.0 KB

Table 5.3: BGP reboot performance summary.

20334 Update messages in total, the majority of which are route advertisements. The entire rebooting process converges in 153.7 seconds, more than two and a half minutes.

We evaluate S-BGP with default configurations as well as that with proposed optimizations. The following is some notation. S-BGP(p) stands for S-BGP using DSA precomputation. S-BGP(c) is S-BGP using caching optimization. And S-BGP(pc) is S-BGP with both optimizations.

5.2.1 Cryptographic Operations

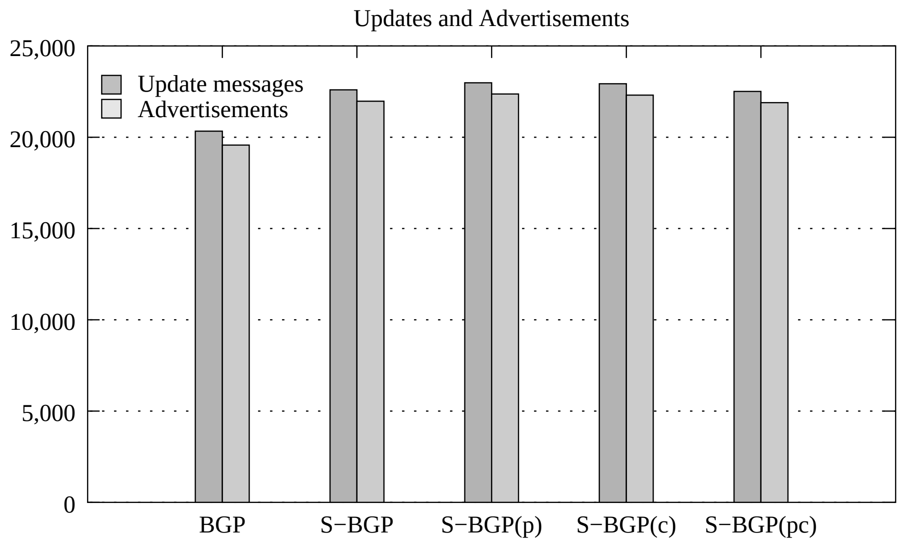


Figure 5.1: Number of Update messages and advertisements. S-BGP only slightly increases the numbers.

As shown in Figure 5.1, S-BGP slightly increases the total number of Update messages and advertisements, because S-BGP needs a longer time to "think"; speakers spend longer time to process every Update message. The rebooting speaker tries to select the preferred routes to all prefixes among table dumps. In our experimental setting, the speaker will establish the same Loc-RIB no matter what BGP version it is running, with or without a security system. A less expensive protocol allows the speaker to process Update messages faster, and thus settle down with preferred routes earlier. Other routes that come later than the preferred one no longer trigger additional Update messages. In this case, S-BGP and its optimized versions are much more expensive than original BGP with a security protocol.

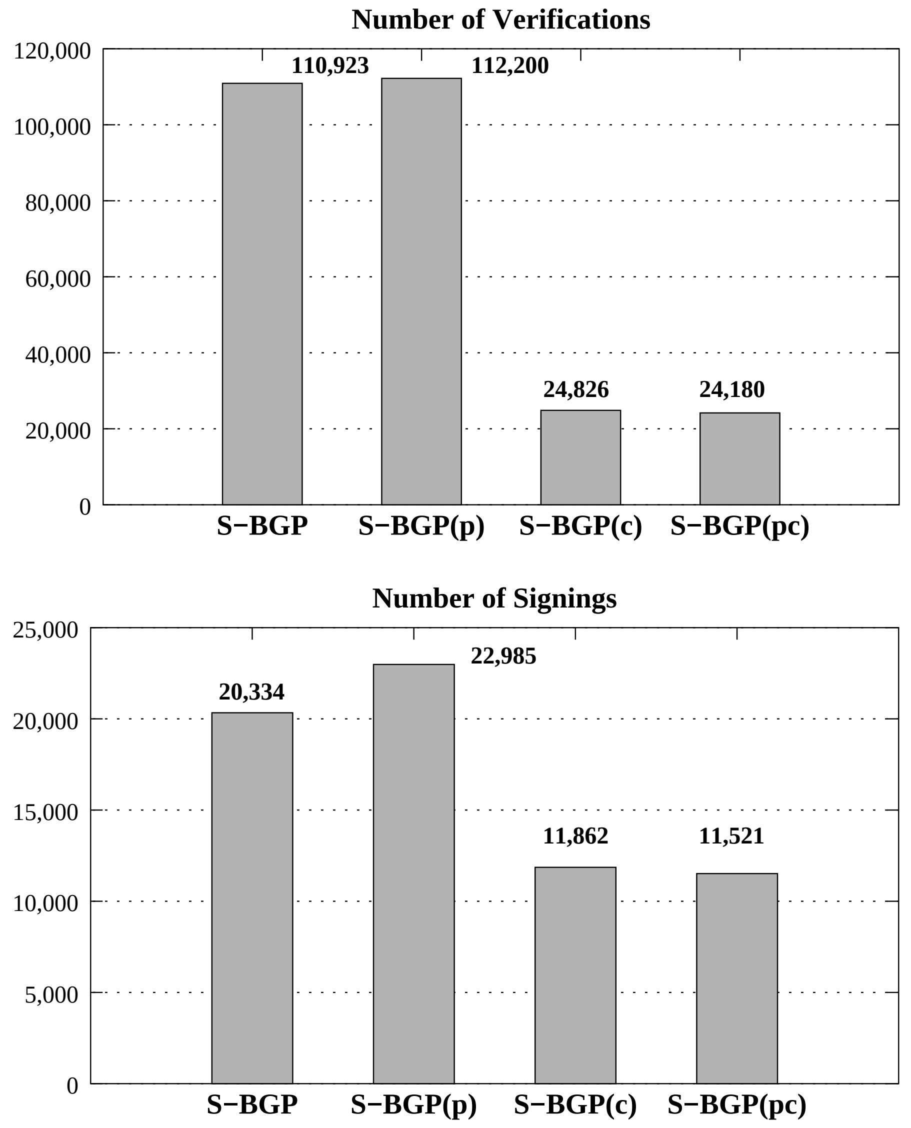


Figure 5.2: Number of cryptographic operations by S-BGP and its variants.

Figure 5.2 illustrates the cryptographic operations for S-BGP and optimized versions. We mainly compare verify and signing operations. On average, every route announcement requires 1 signing and 5.4 verifications. The number echos the average AS path length of all route announcements. The majority of the cryptographic operations are verifications.

Optimizing by caching the verified and signed routes performs well; it can significantly reduce cryptographic operations. Verifications are reduced by 78.4%; signing operations are reduced by 50%. These results reflect that during rebooting, about half the route announcements received by speakers are redundant. Caching works better for verifications, since when speakers cache routes, they verify the same suffix of AS paths only once. In

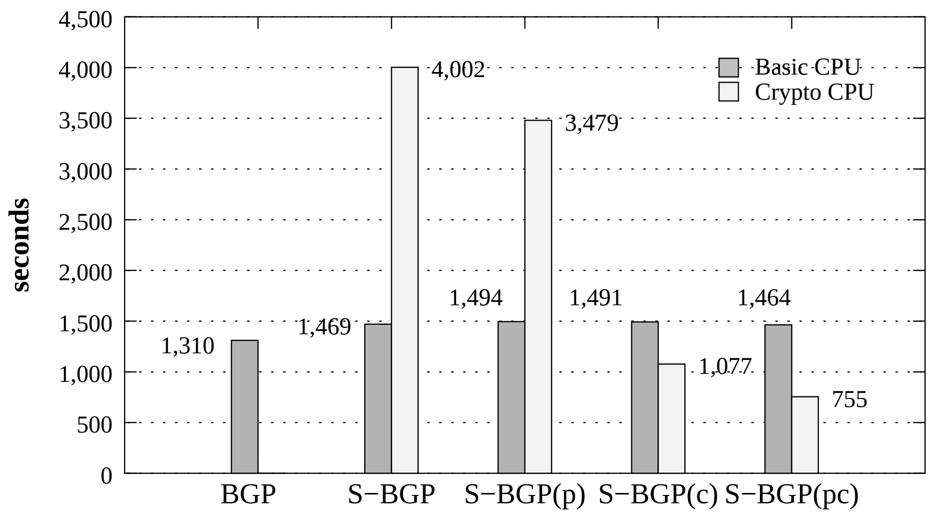


Figure 5.3: Total CPU time spent by BGP speakers. Comparison of S-BGP and its variants.

this set of experiments, we do not limit the cache size used by BGP speakers. Experiment results show that 256KB cache is sufficient for current experimental settings.

5.2.2 Time Delays

Figure 5.3 shows that speakers spend up to 73% of the total CPU cycles on S-BGP cryptographic operations, including signing, verification, and hashing.

S-BGP with DSA pre-computation can help to reduce the crypto CPU usage by about 520 seconds. Such saving is limited since DSA pre-computation only speeds up signing, which contributes to only 15% of the total number of cryptographic operations.

Caching optimization performs very well for reducing CPU time. BGP speakers spend less time on cryptographic operations than on normal Update processing. Overall, the combination of pre-computation and caching performs the best in this category.

The convergence time comparison illustrated in Figure 5.4 shows us two interesting phenomena.

- S-BGP without optimization significantly prolongs the convergence time. The increase ratio is more than 330%.
- Optimizations are effective to decrease the convergence time. pDSA does a better job than the caching optimization. Overall, with both optimizations, BGP convergence time increases by 46%.

Next, we suggest a method to understand convergence time and these phenomena. Recall that the topology contains 110 ASes and 220 prefixes, two managed by each AS. The rebooting router receives 218 prefixes that originated in other ASes. Among 24 peers, this highly connected router maintains five sibling-to-sibling relationships as a part of the routing backbone, and acts as the provider for the other 19 peers. This means that it receives

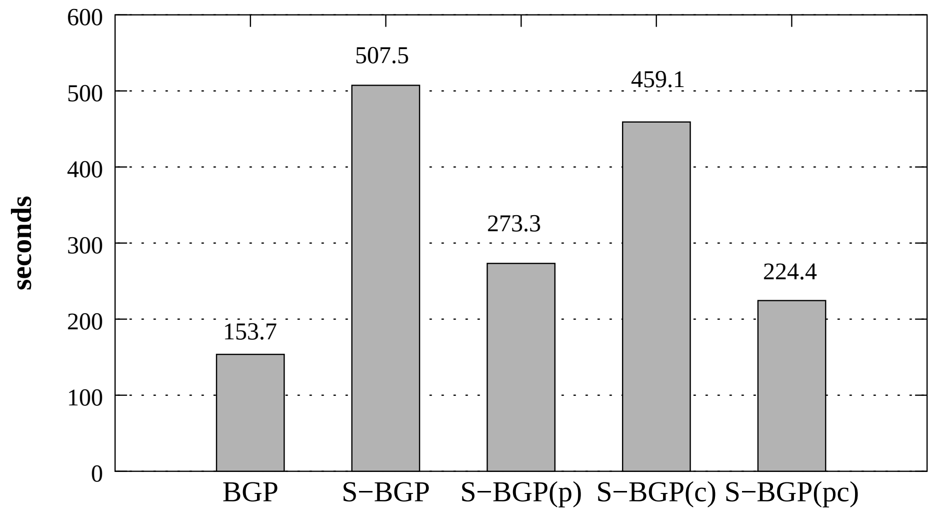


Figure 5.4: Convergence time for router rebooting. Comparison of S-BGP and its variants.

$5 \times 218 + 19 \times 2 = 1128$ Updates; with an average of 65 milliseconds nominal processing per Update, it takes 73.3 seconds (of the 153.7 second convergence time) to work through the table dumps. The table dump processing is an important component of convergence time, but is not the only component.

Here, we use a simple model of the cost of processing a received Update message to understand convergence time:

$$F + \texttt{Pr}\{\texttt{routepreferred}\} \times (L \times C_v + N_p \times C_s) \tag{5.1}$$

where F is a fixed cost, L is length of the AS path, C_v is the cost of verifying a signature, $\texttt{Pr}\{\texttt{routepreferred}\}$ is the probability that the Update reports an AS path that the recipient prefers, N_p is the number of peers receiving the resulting Update, and C_s is the cost of signing that Update. According to benchmarks, DSA signing and verification costs are comparable, while pDSA signing is 1700 times faster than normal DSA signing. In the experiment, L is about 5.4, and $N_p = 24$. For S-BGP using pDSA, the cost can be approximated as follows:

$$F + \texttt{Pr}\{\texttt{routepreferred}\} \times (5.4 \times C_v + 0.014 \times C_s) \tag{5.2}$$

Thus, the term $\texttt{Pr}\{\texttt{routepreferred}\} \times L \times C_v$ is the critical added cost.

To understand the effectiveness of the optimization that caches verified and signed routes, we need to consider savings on crypto operations. Let $\texttt{Pr}\{\texttt{sign}\}$ be the probability that the route should be signed, and $\texttt{Pr}\{\texttt{verify}\}$ be the probability that the route should be verified. The cost of processing one Update message is the following:

$$F + \texttt{Pr}\{\texttt{routepreferred}\} \times (L \times C_v \times \texttt{Pr}\{\texttt{verify}\} + N_p \times C_s \times \texttt{Pr}\{\texttt{sign}\}) \tag{5.3}$$

Recall that previous analysis shows that L is about 5.4, $\texttt{Pr}\{\texttt{verify}\}$ is about 22%, $N_p = 24$, and $\texttt{Pr}\{\texttt{sign}\}$ is about 50%. Here, the cost of processing a normal Update message is still

	BGP	S-BGP(c)	S-BGP(cp)
average memory cost (KB)	9.0	112.3	109.7
maximum memory cost (KB)	20.4	217.4	211.1

Table 5.4: Average and maximum memory on routers.

estimated as F, because additional cost of operating caches routes is negligible. The above model can be approximated as

$$F + \texttt{Pr}\left\{\texttt{routepreferred}\right\} \times (1.19 \times C_v + 12 \times C_s) \ . \qquad (5.4)$$

Given that C_v and C_s are comparable according to DSA benchmarks, we conclude that pDSA is more effective than the caching optimization. Experimental results confirm our analysis. S-BGP with pDSA converges much faster. The factor of 1700 reflects the difference.

5.2.3 Memory

BGP speakers store all cached routes in memory (e.g., RAM). Table 5.4 illustrates the average and maximum memory cost for individual BGP speakers. In measuring routes in memory, we assume that BGP uses default minimum settings for Update messages. Thus we do not model any additional path attributes. Mainly, we measure the length of the headers, NLRI, and AS path. In addition to those fields, we count signatures, key lengths, and hash values for S-BGP. We admit that such a measurement is not accurate, since we ignore other S-BGP-specific fields. Nonetheless, such an approach provides us a good common base to compare S-BGP with other security systems that do not have an implementation or prototype.

Table 5.4 shows that maximum memory cost on a router is approximately double the size of average. S-BGP caching optimization performs roughly the same on memory with or without pDSA.

Looking at the memory measurement more closely, we found that routers closer to the core of the topology tend to use less memory. Memory cost on edge routers varies greatly. Nonetheless, the maximum memory consumption is by an edge router.

Figure 5.5 illustrates memory cost on each router. Tier 1 ASes are the core of the topology. Tier 2 ASes are connected to the core, while tier 3 ASes are two hops away from the core. There are two clearly divided areas in Figure 5.5. On the right hand side, all routers in the core consume fairly low memory, most of which are lower than average. On the left side area, routers in tier 2 and tier 3 ASes show variation on memory cost. Such variation is created by the randomness in the experiments. During the experiment phase, the rebooting router needs to establish a new routing table for 218 prefixes, while other routers eventually insert two more routes to the prefixes originated by the rebooting router. Since table dumps are random, some of the edge routers receive the ultimately preferred routes to the prefixes fairly early. At this point, they stop caching any additional routes. Some other routers keep changing their mind on selecting preferred routes; they end up caching more routes.

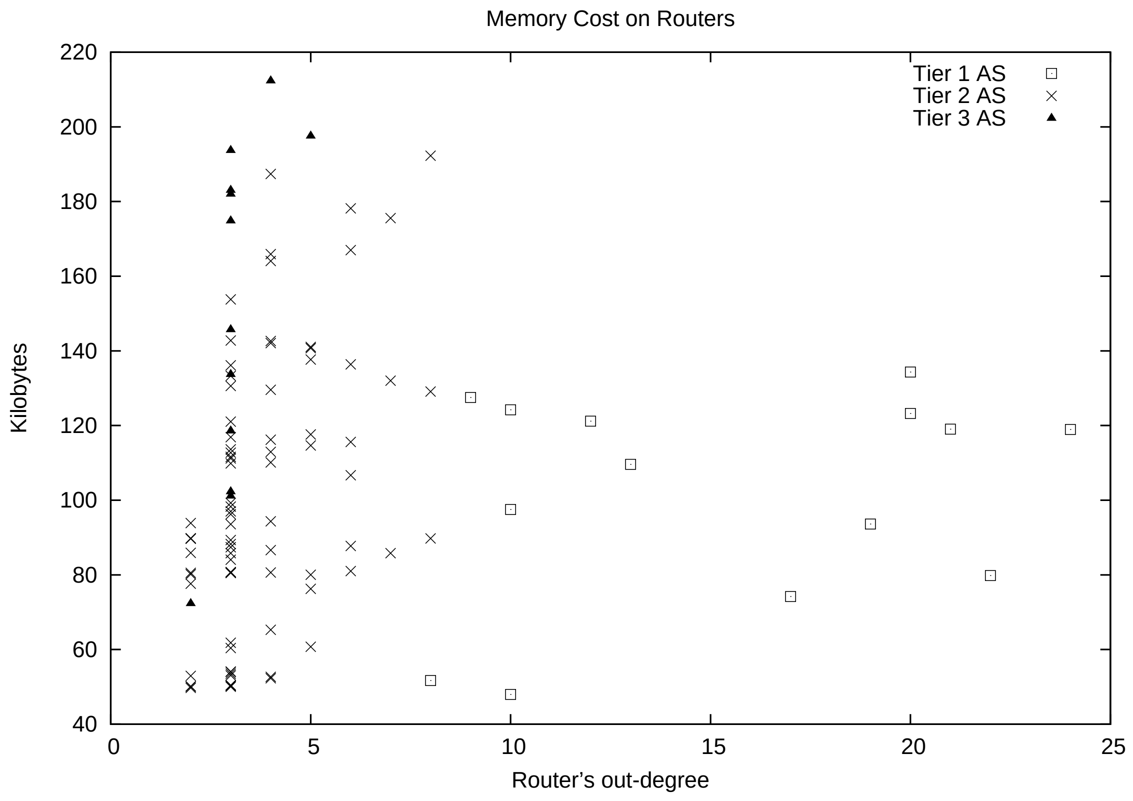


Figure 5.5: Memory costs correlated with AS out-degree and levels of AS in topology.

	BGP	S-BGP	S-BGP(p)	S-BGP(c)	S-BGP(cp)
Average message size (B)	36.1	338.8	336.9	336.0	336.0
Maximum message size (B)	42.6	527.7	584.0	596.4	605.7

Table 5.5: Average and maximum message size

Statistically, it seems that edge routers consume more memory for routes. We posit two reasons. First, as a pure customer in the network, an edge router may receive more route announcements than the ones in the core. Second, the AS paths recorded by edge routers are longer. Edge routers cache more signatures for a route announcement.

5.2.4 Update Message Size

Lastly, the measurements on message size are straightforward. Obviously, S-BGP increases message size, and all of its optimizations do not help in this category. Shown in Table 5.5, the increase is even higher on maximum message size since number of signatures grows as the length of AS path increases. In our experiments, there are only 220 prefixes being processed, so we do not expect that these longer Update messages would saturate the network bandwidth.

Performance study results in [84] are consistent with our measurements. Based on 3.6

	S-BGP	S-BGP(p)	S-BGP(c)	S-BGP(cp)
Prolonged Convergence	330%	178%	299%	146%
Increase Message Size	939%	933%	933%	933%
Memory burden	–	–	1248%	1219%

Table 5.6: Summary of S-BGP performance.

route attestations per path, Kent et al. measured that message overhead increase is over 700%. The authors concluded that since Update transmission represented a very, very small amount of data relative to a subscriber traffic, the network will not be congested by Update messages. Of course, they were looking at BGP activities in normal conditions (at most one Update per second). Here, we are experimenting with table dumps within short amount of time. Our discussion would be more complete if we take into account the network congestion. One area of future work is to model a larger network topology and more prefixes, to understand the impacts that increased message length has on bandwidth utilization and network congestion.

5.3 Discussion

From the above performance results we conclude that S-BGP faces serious performance issues. Table 5.6 summarize the performance degradation. As we have discussed, prolonged convergence time is a critical performance issue by S-BGP. The proposed optimizations can help relieve long convergence problem, but they have their own issues.

pDSA (signing with pre-computed values) speeds up DSA signing considerably, and thereby shortens the convergence time. However, to keep pDSA secure, implementations need to use secure storage to store all pre-computed values. Furthermore, the storage should be large enough to hold these values. The DSA signature is a pair of values (r, s), where r does not depend on the message. We can create a sequence of random k values, then pre-compute r values for each of them. We can also pre-compute k^{-1} for each of those k values. The parameter k is subject to configuration. We should not generate too many pre-computed values, which may increases the risk of breaking the secure storage. Then when a message comes along, we can compute s for a given r and k^{-1}. For security, each signature requires a new value of k, and that value must be chosen randomly. If Eve ever recovers a k that Alice used to sign a message, she can recover Alice's private keys, x. If Eve ever obtains two messages signed using the same k, even if she does not know what it is, she can recover x. And with x, Eve can generate undetectable forgeries of Alice's signature [107, 147]. In any implementation of the DSA, a good random number generator is essential to the system's security. Even with a strong random number generator, pDSA opens up a window for Eve. Instead of exploiting some properties of the random number generator, Eve now can launch an attack on memory locations where the implementation stores pre-generated k values. These locations should be protected by a strong security mechanism. Such a task is not straightforward or easy by any means.

Caching, on the other hand, generates incredibly high demands on routers' memory. Once again, we assume that the speakers hold all cached data in memory. Our simulation results have shown that memory cost by S-BGP is 12 times the memory of plain BGP. We used a BGP routing table dump from the RouteViews project to gain a better understanding of memory demands. This table dump, collected from the AS6447 archive on May 04, 2005, takes 209MB in MRT[116] format. It contains 162,237 unique IP prefixes and 2,011,005 unique (AS path, prefix) tuples. To cache all received S-BGP route attestations, the router should record 8,284,042 DSA signatures, which requires about 316MB in cache in addition to the original routing table. This estimated space increase is much smaller than the simulation results. In fact, we underestimate the amount by looking at signature length only. As discussed in [85], the average size would be larger since other information is involved in each attestation. Meanwhile in simulation, we underestimate the space for routes as we ignore several path attributes. Routes in practice contain a lot more fields than just the prefix and AS path. Kent et al. noted that, for S-BGP in 2003, the Adj-RIBs space required for RAs is about 30–35MB per peer, and the total requirement for a speaker with tens of peers may be gigabytes [85]. However, currently deployed BGP speakers cannot be configured with more than 128M or 256M of RAM. It is possible that the speakers use hard disk to hold cached data. However, according to the design of S-BGP, the speakers will try to keep it in memory so that operations on the cache are efficient enough to keep up with the BGP Update processing.

To speed up convergence, S-BGP optimizations introduce additional problems as we have discussed above. Nonetheless, the resulting convergence time is not even close to optimal. Such results made us seek for other cryptographic techniques to address the performance problem. Chapter 7 will introduce our proposed more efficient mechanism and analyze the performance.

5.3.1 Switching to ECDSA

Elliptic Curve Cryptography (ECC) is emerging as an attractive type of public-key cryptosystem. Unlike traditional public-key cryptosystems, such as RSA and DSA, ECC is based on mathematics of *elliptic curves*—non-singular plane curves that are defined by an equation of the form $y^2 = x^3 + ax + b$. The main benefit of ECC is that under certain situations it uses smaller keys than other methods while providing an equivalent or higher level of security. In some cases, ECC is able to offer equivalent security with smaller key sizes, faster computations, lower power consumption, as well as memory and bandwidth savings.

ECDSA [164] is the elliptic curve counterpart of the traditional DSA algorithm. It has attracted a lot of attention recently. Popular cryptographic libraries support ECDSA. Some hardware architecture, such as Sizzle [63], even allow efficient SSL handshakes on small devices using ECDSA. S-BGP can easily switch to use ECDSA as the digital signature algorithm. We use the latest ECDSA implementation in OpenSSL 0.9.8 beta version to understand its performance. It is unclear if it outperforms DSA, except on the key size criterion. Table 5.7 demonstrates the performance data of different signature algorithms with equivalent security. Compared to DSA, ECDSA signing may perform faster. Recalling that majority of cryptographic operations for S-BGP RAs are verifications, we expect no significant improvement on processing latencies. Moreover, both DSA and ECDSA using 160-bit curves produce 40-byte signatures. Hence, ECDSA does not help S-BGP reduce the

	DSA	ECDSA		
	(1024-bit)	secp192r1	sect163k1	sect163r2
Key Size (bytes)	408	180	139	155
Sign (*ms*)	3.5	1.0	3.1	3.1
Verify (*ms*)	4.5	4.4	8.2	8.7

Table 5.7: Key size and running times of signature algorithms with equivalent security. The benchmark numbers are obtained from OpenSSL 0.9.8 beta version with ECDSA support on a 1GHz processor. Elliptic curves for ECDSA are recommended by NIST for Federal government use [164]. Key size accounts for public key and domain parameters.

size of attestations or signatures. Certainly, more developments on speeding up ECDSA in the future may lead us to re-examine this issue. Besides ECDSA, there are many newly proposed cryptographic algorithms that take advantage of ECC. We are going to revisit this issue in Chapter 7.

Chapter 6

Evaluating Origin Authentication

Origin authentication schemes refer to general approaches to BGP route authentication that meet the following two criteria:

- The organization that claims to own an address space corresponding to a reachable prefix advertised in an Update indeed has the address ownership authorized by its parent organization.
- The first AS in the route was authorized, by the owners of the address space corresponding to the set of reachable prefixes, to advertise those prefixes.

In other words, origin authentication focuses on prefixes defined in the NLRI field of the route announcement and the originating AS. S-BGP address allocation certificates and address attestations are one example of an origin authentication scheme. In this chapter we evaluate S-BGP address attestations by comparing them with another origin authentication scheme—*Origin Authentication (OA)* proposed by Aiello et al. [5]. Both schemes serve the same purpose for authenticating prefixes in routes with slightly different approaches. We review the design architecture of OA in Section 6.1 followed by detailed performance discussion in Section 6.2.

6.1 Origin Authentication (OA) Scheme

Aiello et al. considered the semantics, design, and costs of origin authentication in inter-domain routing. Their work makes four main contributions.

- They formalized the semantics of address delegation and use on the Internet.
- They developed and characterized broad classes of origin authentication proof systems,
- They estimated the *address delegation graph* representing the current use of IPv4 address space using available routing data, and
- They used a traced based simulation to evaluate the proposed services.

	S-BGP	OA
Architecture	Hierarchy	
types of address allocation	four	one
Data structure	certificates and AAs	attestations
Distribution	out-of-band	in-line

Table 6.1: Comparison of S-BGP and OA. They authenticate origin authentication using the same IP address allocation hierarchy. The details of schemes are different. We explain each of the differences in later sections.

OA defines the assignment of IP addresses as *delegation*. The ownership of individual prefixes may be delegated from one organization to another several times. Similar to S-BGP, the delegations form a hierarchy reflecting current IP address allocation infrastructure. The root of the hierarchy is IANA [75], the Internet Assigned Number Authority that controls the Internet IP address allocation. A delegation path's validity is checked by the following three steps: 1) the ownership source is IANA, 2) the path is acyclic, and 3) the last assignment edge is AS-respecting; the assignment is made from an organization to an AS.

OA address delegations and S-BGP address allocation architecture have several common features in terms of expressing and authenticating IP address allocations. However, they are different in a number of ways. Table 6.1 summarizes the features of these two origin authentication schemes. They express very similar hierarchies of IP address allocations. In processing the address allocations, the forms, the data structures, and the ways that organizations distribute data are different. S-BGP expresses the address allocation in the following form: $Org_x \rightarrow Org_y, p_1, p_2, \ldots, p_k$. On the other hand, OA designs more forms to express the IP address allocation semantics in order to find the most efficient form. In terms of the data structures, S-BGP uses certificates and address attestations, and OA uses attestations only. The attestation size is much smaller than certificate size. Finally, unlike S-BGP, speakers implementing OA send attestations for origin authentication in Update messages.

Delegation Attestations

Besides the delegation semantics, the delegated paths are authenticated by *Origin Authentication Tags (OATs)*. OATs consist of a delegation path, a set of *delegation attestations*, one for each edge in the path, and an *ASN Ownership Proof*. In order for an OAT to be positively verified, each delegation attestation must be positively verified and the validity of the path must be verified. A delegation attestation is a signed statement by the delegator organization attesting a delegation of IP prefixes to other organizations (the delegatee). The ASN ownership proof is a statement signed by ICANN attesting to the fact that one or more AS numbers are among those granted to a particular organization. S-BGP's PKI for AS number certificate can be one mechanism to be used for such proof. The design goal of delegation attestations is to make them lightweight to be appropriate for in-band transmission. BGP speakers could send the delegation path with delegation attestations together with the route announcement, while letting speakers retrieve the ASN Ownership Proof out-of-band.

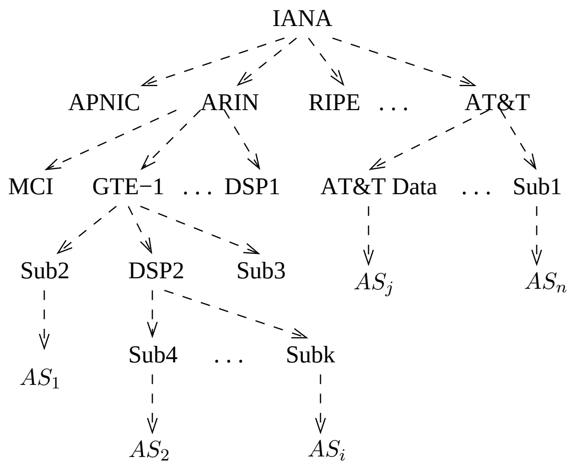


Figure 6.1: Design of OA address delegation attestations that forms a hierarchy similar to that of S-BGP.

Figure 6.1 sketches the design OA for origin authentication. The hierarchy shows the difference between S-BGP and OA. BGP uses address allocation certificates for most address allocation authorization, and lets end subscribers create address attestations to attest that a certain AS is authorized to originate routes to the prefixes. All of the related data structures are distributed out-of-band and periodically. Consequently, a separate revocation mechanism is needed on certificates and AAs. Instead of two types of data structure, OA uses delegation attestations from top to bottom for authenticating address delegations. This data structure is much smaller than certificates, thus allowing in-band distribution with Update messages. Revocation of previous delegation is handled by creating a new attestation and distributing it via Update.

Constructions

The structure of address allocation certificates in S-BGP is fixed—a single organization issues a certificate to single *subject* organization, binding its public key with a list of prefixes allocated to this organization. Rather than this fixed form, OA calls for three basic types of delegation attestations.

Simple Delegation Attestation Simple delegation attestation is a signed statement by Org_x binding one prefix p to an organization Org_y. The form is $Org_x \rightarrow (p, Org_y)$, where Org_y can be an organization identifier or an AS number. We denote this construction as *OA-Simple*. It is easy to construct, maintain, and distribute. However, because associations must be created, signed, and validated individually, they can create a significant resource burden.

Authentication Delegation List To reduce the cost of OA-Simple, an organization can create a single list of all of its delegations and sign that list. The delegation could be written as $Org_x \rightarrow [(p_1, Org_{y_1}), \ldots, (p_k, Org_{y_k})]$, where k is the total number of delegations Org_x

has created. We denote this construction as *OA-List*. To verify a delegation path, the router should be provided with one OA-list for every organization on the path. In practice, some organizations may generate quite large lists. While OA-List can significantly reduce the number of signing and verifications performed by organizations and speakers, the verifiers must commit significant bandwidth and storage for these lists.

OA-Simple and OA-List are two extremes of construction. A natural means of compromise is to let Org_x sign the statement of delegations made to the same organization or AS. This variant is called *AS authentication delegation list*, which we denote as *OA-AS-List*. The statement could be written as $Org_x \rightarrow [(p_1, \ldots, p_j), Org_y]$. This design most closely resembles the address allocation certificates of S-BGP. The advantage of this approach is that AS can collect proofs for all addresses that it originates.

Authentication Delegation Tree Rather than a plain list, all delegations by an organization can be organized using a Merkle hash tree [108, 109]. The leaves are the hash values of delegations in the form (p, Org_y). The internal node is a hash of the children of the node. Org_x signs the root of the tree, R. The attestation for a delegation (p, Org_y) consists of the signature on R as well as the hash path in the tree to facilitate verifiers to reconstruct the root R from the hash of (p, Org_y). We denote this construction as *OA-Tree*.

6.2 Performance Evaluation

The above constructions provide equivalent level of security with their own strengths and weaknesses in terms of performance. We next use our simulation model to evaluate them. We target the three constructions OA-Simple, OA-List, and OA-Tree, and the variant OA-AS-List. In the comparison, we keep in mind that OA-AS-List mirrors the design of address allocation certificates and AAs of S-BGP. In our evaluation, we configure the OA-AS-List exactly the same way as the S-BGP origin authentication scheme, except that OA-AS-List uses attestations to express address allocations. The performance results on cryptographic operations and time latencies are applicable to S-BGP origin authentication. Furthermore, we consider OA-AS-List as the variant of S-BGP origin authentication that distributes all origin authentication related data in-line.

6.2.1 Model Configurations

In modeling OATs for a route announcement, the entire address delegation path to the prefix and corresponding delegation attestations are enclosed. Since there is no implementation or prototype for OA, we cannot model the complete data structures. To model the space cost, we only take into account necessary fields for OATs, including minimum space for the (p, Org_x) pair expressing delegation and the signatures. We assume that OATs (except ASN ownership proofs) are distributed in-band. Thus, the model of space affects the calculation of message size and storage costs.

For timing, we only model the validation of OATs on BGP speakers. Although speakers send OATs with route announcements, delegation attestations are prepared by organizations off-line. Lacking knowledge of these organizations—such data is typically private—we are not able to model these organizations in simulation. As the result, we measure only signature

parameter	**value**
signature algorithm	DSA
hashing algorithm	SHA-1
caching	true/false
Signature length	40 bytes
Hash length	20 bytes
Length of Prefix	5 bytes
Organization Identifier	4 bytes

Table 6.2: Configuration parameters for evaluating origin authentication.

verifications and consequent convergence time. Table 6.2 presents some of the parameters and constants we use in experiments.

Address Delegation Graph

Simulated address delegation paths for each prefix are necessary in the experimental configuration. Thus, we add one more module in the simulation—the *approximated address delegation graph*. In the model, each AS has one operating router originating two prefixes. We should configure one delegation path for each of these prefixes.

Similar to the approach we used to create AS-level topology, we start by the AS-level topology the Internet derived from real routing tables. Admitting that it is difficult to determine the exact delegation structure, Aiello et al. devised a method to estimate the address delegation graph of the IPv4 address space on the Internet. Their approximation took advantage of several publicly available data sources. At the top level, IANA make assignments of large blocks of addresses to organizations. The assignment details are available in [76]. Next, they use the RouteViews project repository. By examining the route announcements, one can infer the delegation relationship from announced prefixes. In particular, when a prefix p announced by an AS, say AS_1, is a superset of p' announced by another AS, AS_2, we can infer that the organization that AS_1 belongs to delegates p' to AS_2 or to the organization that AS_2 belongs to. They eventually generated an approximation with a few nice properties. The graph conforms to a Zipf distribution. Most delegations are performed by a relatively small number of organizations. Despite a few changes, the delegation graph is relatively stable over months.

The authors kindly shared their graph with us. We heuristically reduced it to the size of our model. In practice, ASes typically announce many prefixes, each of which is assigned via different delegation paths. In our model, only two prefixes are originated by each AS. We add randomness in the model to capture the diversity of the real world. To match the approximated delegation graph with our model, we first take the subgraph that covers all 110 ASes. Since there are typically many prefixes owned by an AS, we let BGP speakers randomly choose a path for each prefix based on the origin AS. We limit the path length to seven (since address delegation paths are reported to be no longer than 4-5, in practice). This randomly chosen path determines what address delegation attestations are involved.

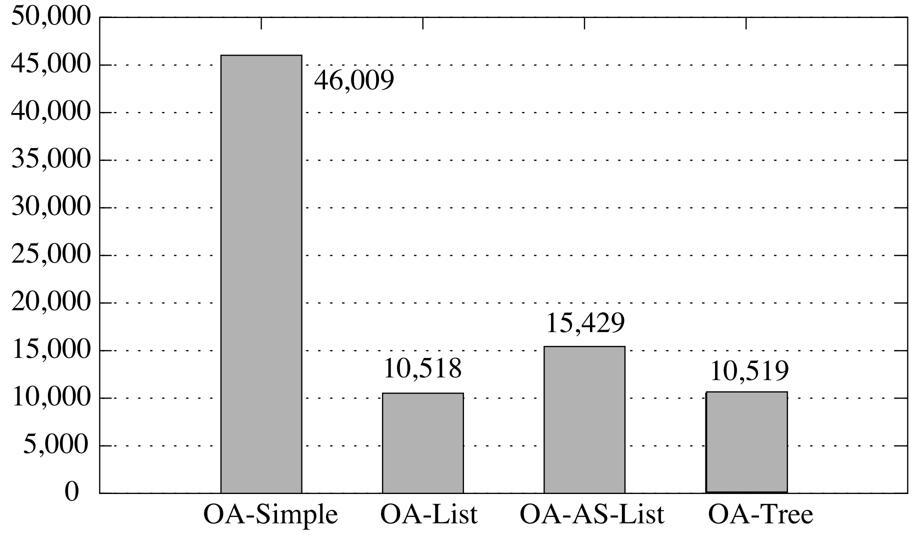


Figure 6.2: Number of verification operations by OA address delegation attestation constructions.

6.2.2 Performance Results

Signatures and Verifications

Since organizations prepare delegation attestations off-line, we only compare signature verifications on attestations. Figure 6.2 presents the measured results. During the experiment, there are about 20,000 route announcements. This indicates the average length of address delegation path is about 2.3. As expected, OA-List requires the least number of verifications. A more important conclusion we can draw from Figure 6.2 is that both the OA-AS-List and OA-Tree constructions are fairly effective in reducing verification operations. OA-Tree requires about the same number of verifications as OA-List. Extra hashing operations by OA-Tree are fairly fast according to the benchmarks.

Time Delays

Figure 6.3 compares the CPU time by OA constructions with S-BGP RAs. All OA constructions except OA-Simple spend less CPU cycles for crypto operations than S-BGP(cp). Interestingly, Figure 6.4 shows that OA-Simple converges much faster than S-BGP(cp). In fact, all OA address delegation attestation constructions can quickly converge, only a few seconds longer than plain BGP. Why? To explain the phenomenon, we examine our simple Update processing model again. According to the definition in Section 5.2, the approximate time to process an Update message for S-BGP(cp) can be written as follows:

$$F + \texttt{Pr}\left\{\texttt{routepreferred}\right\} \times (1.19 \times C_v + 12 \times C_s/1700) \ . \tag{6.1}$$

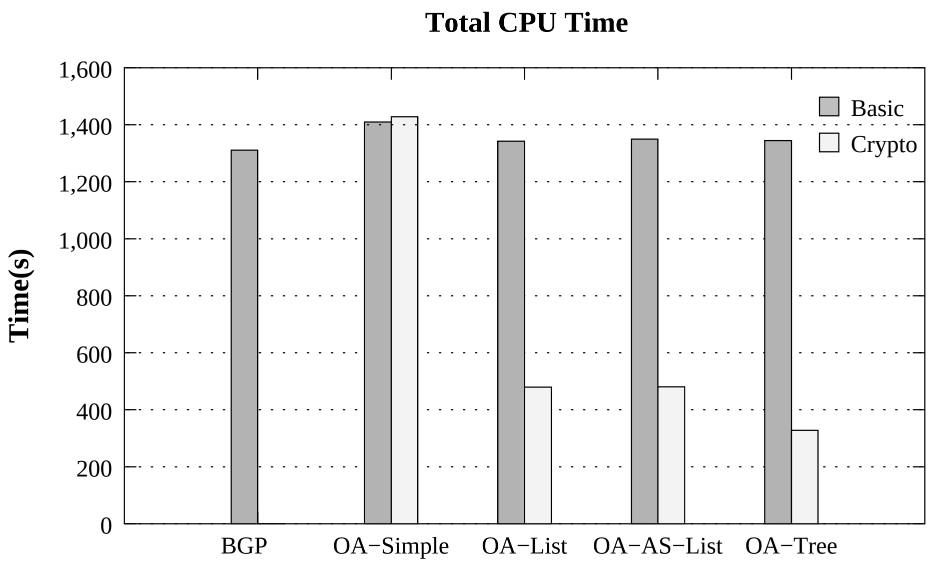


Figure 6.3: Total CPU time by OA address delegation attestation constructions.

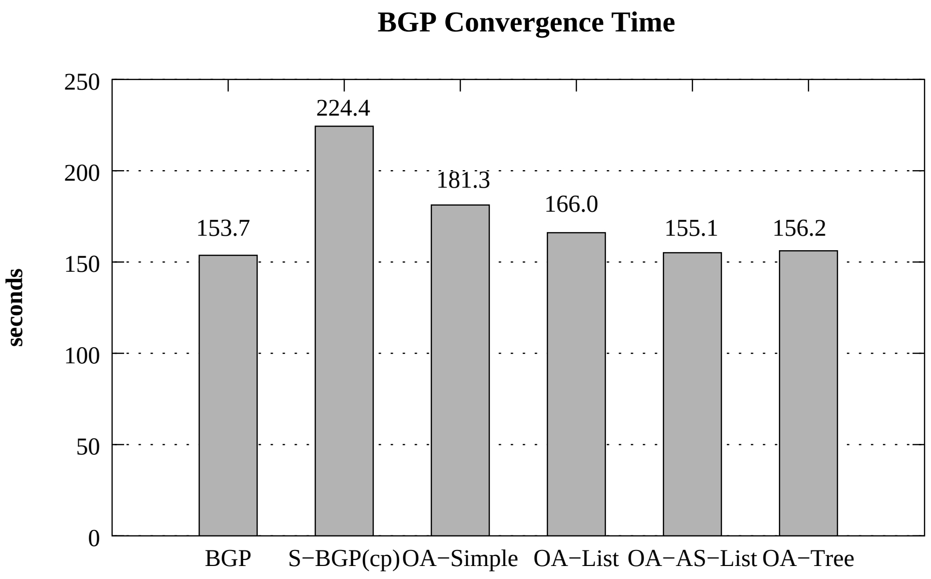


Figure 6.4: Convergence time by OA address delegation attestation constructions. S-BGP(cp) shows the performance of route attestations by S-BGP with all optimizations. It is here only for comparison purpose.

For OA address delegation attestations, there is no cost of signing attestation. So, for OA, the cost of processing an Update can be expressed as the following:

$$F + \texttt{Pr}\left\{\texttt{routepreferred}\right\} \times N \times C_v \ , \tag{6.2}$$

where N is the length of address delegation path. We have learned that the average of N is 2.3. Thus, the approximated time for OA-Simple is

$$F + \texttt{Pr}\left\{\texttt{routepreferred}\right\} \times 2.3 \times C_v \ . \tag{6.3}$$

Comparing Equations 6.1 and 6.3, OA-Simple is clearly more expensive. However, the term $12 \times C_s/1700$ in Equation 6.1 is an important factor for convergence time. Despite its small value, the signing latency can significantly slow down BGP convergence. This result indicates an observation.

> **Observation:** Convergence time is more sensitive than to signing cost to verification cost.

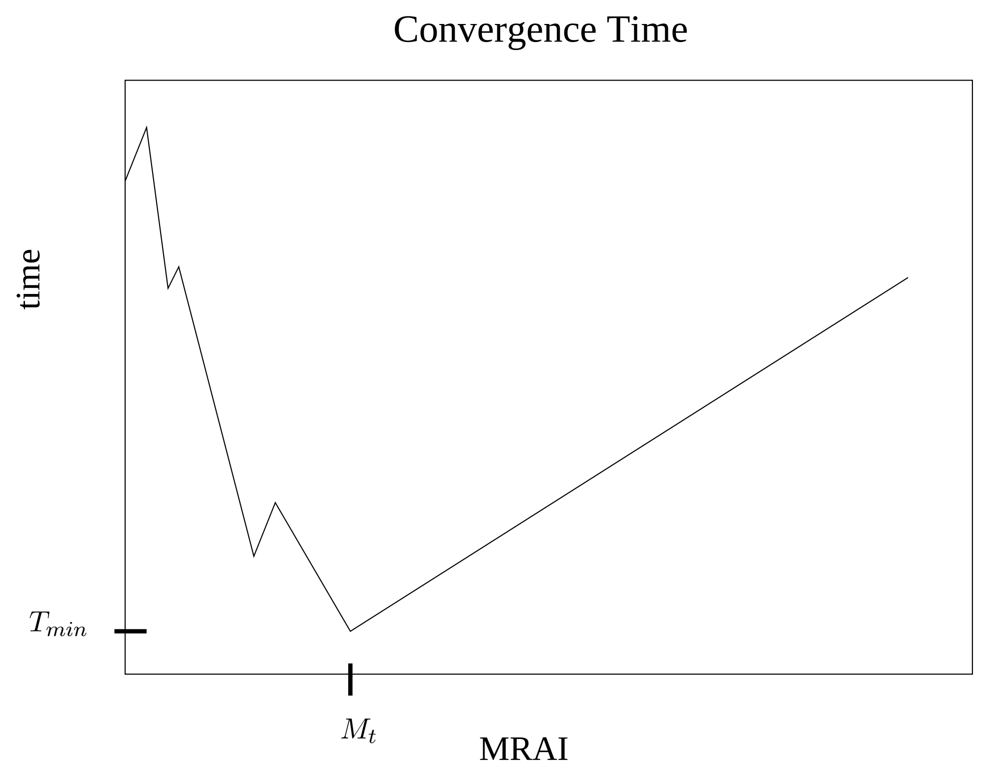


Figure 6.5: Optimal MRAI affects convergence time. We borrow this figure from the BGP experimental study by Griffin and Premore [59].

A useful aid for reasoning about what is happening is the graph shown in Figure 6.5. It captures the key observations Premore has made [138] on how convergence behaves as a function of the value of the MRAI timer. Namely, there is an optimal value for MRAI,

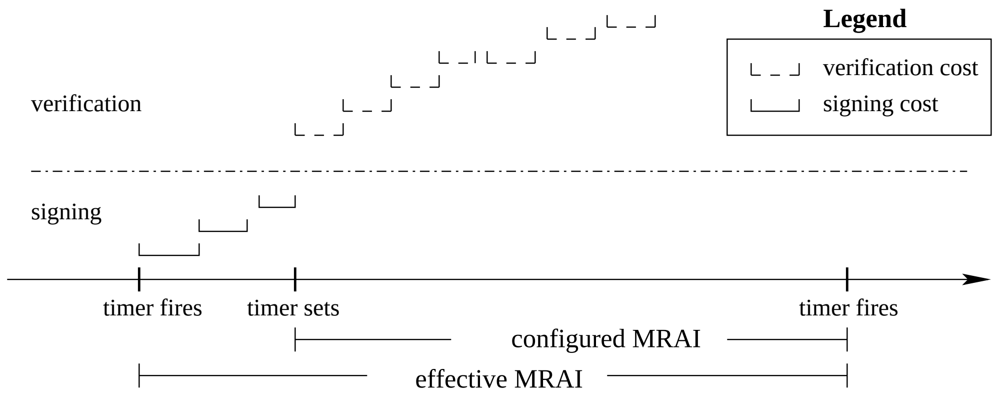


Figure 6.6: An example of how the timing of signing and verification operations affect the MRAI timer. Signing latencies associated with message sending effectively prolong the actual MRAI timer value.

denoted as M_t, where convergence time is minimized to T_{min}. Beyond M_t, convergence time increases linearly. However, the MRAI optimal value varies from network to network, and may be difficult to determine in practice.

Since the BGP message sending process is governed by the MRAI timer, Update messages arrive in "waves". In the second part of the curve, the MRAI timer is so large that all the Update messages are processed before MRAI fires again. Thus the convergence time grows linearly with the MRAI value.

This curve leads us to revisit the way we model verification and signing latencies. Figure 6.6 illustrates the timing that a speaker handles verification and signing operations correlated with the MRAI timer. Once the timer fires, the speaker starts to send out Update messages waiting in the output buffers [1]. These messages require signing operations. After finishing with the sending process, the speaker sets the MRAI timer again and processes the received messages if the incoming buffers are not empty. Received messages may need verification operations. The MRAI timer is set at 30 seconds by default. This value typically is long enough to allow the speaker to handle all received messages in this batch. After handling these messages, there can be more outgoing messages in the buffers. They are going to be sent out when the MRAI timer fires next time. Figure 6.6 shows that the verification latencies overlap with the period when the speaker is waiting for the MRAI timer to expire. In contrast, signing costs do not overlap the MRAI timer. As the result, the speaker has a longer *effective* MRAI value than the *configured* value. Based on the conclusion from Griffin and Premore shown in Figure 6.5, we conclude that the signing costs increase the MRAI value, thus they directly prolong the convergence time, given that the configured MRAI value is long enough to handle incoming messages. A small amount of time for signing operations has larger impact on convergence than the same amount of time for verifications. The impact by verifications will show when the verification latencies in a batch are too much for the speaker to handle within one period of MRAI value, as shown in the first part of curve in Figure 6.5.

[1]This is one of the approaches to charge signing latencies. This approach is used by S-BGP. We'll discuss the other approach for more efficient security mechanisms in Chapter 7.

Attestation Constructions	OA-Simple	OA-List	OA-AS-List	OA-Tree
Storage for Attests. (KB)	42.80	666.27	13.23	30.22
Message Size (Bytes)	496.97	36293.37	575.35	1029.24

Table 6.3: Average memory cost and message size by OA address delegation attestation constructions.

Memory

We let BGP speakers cache verified attestations and associated prefixes; we then measure the average memory cost and message size. Table 6.3 shows that the OA-List scheme is more costly than other schemes, mainly because the list construction produces extremely long delegation attestations. In the approximated address delegation graph, the average number of delegations made by organizations is about 56.96. Moreover, about 16 organizations make 80% of the address delegations. Obviously, this graph has high connectivity and the delegations are concentrated on very small portion of organizations. These features are the reason why the AS-List approach can produce long lists of prefixes in address delegation attestations. According to Figure 6.2, the AS-Tree approach handles the smallest number of signatures; however, its memory cost and message size are worse than OA-AS-List, mainly because the AS-Tree approach involves hash values, which are much longer than organization identifiers.

Update Message Size

Similarly, because of the large number of delegations by a few organizations, the resulting Update messages by OA-List are incredibly long. From Table 6.3 we observe that OA-AS-List is, again, performing fairly well. The Update message size is close to that of OA-Simple.

6.2.3 Discussion

To summarize, we have made several important observations in evaluating origin authentication schemes.

OA-AS-List is the best Combining time latency, memory cost, and message size, overall, OA-AS-List stands out as the best. OA provides us a chance to examine different approaches to construct address delegation attestations. The winning design is very similar to S-BGP certificates. Our experiments thus confirm the design of S-BGP certificate for address allocation.

In-band distribution is possible Although Update messages produced by OA address delegation attestations are more than 10 times longer than ordinary Update messages, their lengths tend to stay stable. This is different from the messages by S-BGP RAs, whose size is greatly affected by AS path length. Moreover, the message length growth by OA-AS-List is still way below the BGP Update MTU. All these observations indicate that if attestations (rather than certificates) are used to authenticate NLRI, it is possible to allow attestations to be distributed in-band.

Signing latency is a major factor in convergence time Our experiments have shown that signing latencies seem to affect convergence time greatly, compared with verification latencies. Although interesting, this hypothesis might not hold for other network settings or other security systems. We have given some preliminary explanation of it. We will revisit this issue in Chapter 7.

Chapter 7

Efficient Path Authentication Schemes

We have defined the term *path authentication* earlier. Path authentication focuses on the AS path field of the route announcement. S-BGP route attestation, with the help of certificates for AS numbers and routers, is one example of a path authentication scheme. We have evaluated the performance of S-BGP RAs in Chapter 5 and have shown that the design has serious performance problems. In this Chapter, we propose new path authentication schemes that can significantly improve BGP performance in all criteria: time delays, message size, and memory demands. We also use simulation to evaluate their performance in comparison with S-BGP.

We propose two schemes—*Signature Amortization (S-A)* and *Aggregated Path Authentication*. Signature Amortization focuses on reducing the time delays for the signature verification and signing. In exchange, it increases the cost for space. Based on S-A, we further design the Aggregated Path Authentication scheme that can achieve significant performance improvements in both time and space. These new path authentication schemes maintain the strong security that S-BGP provides, while providing more a efficient security solution to ease practical deployment on the Internet.

7.1 Signature Amortization

Signature Amortization (S-A) is designed to speed up the processing of S-BGP RAs. Questions related to reducing the cost of cryptography in the routing context have been raised before, e.g. [56, 74, 68, 165]. Such methods typically work to reduce the cost by reducing the dependence on public-key cryptography. As useful as these may be, there are real difficulties in applying those methods in BGP. The approach we explore is to do expensive private-key operations less often, amortizing that cost over multiple messages.

Speeding Up Validation

As we have seen, the most significant drawback of DSA is its high verification cost. If we eschew caching, then we cannot escape validating every path suffix when we validate an AS path. From this point of view RSA is more attractive, because its verification cost is (by our measurements) 12.5 times faster. Where RSA fails us in this context is the high cost of signing every Update. From our previous experiments, we have learned that there are a lot more verifications than signing operations. During router rebooting, each route announcement needs 1 signing and 5.4 verifications on S-BGP RAs. Choosing RSA might thus speed up AS path validation considerably.

Speeding Up Signing

We have developed two steps to amortize the cost of expensive private-key operations. The idea is to do these operations less frequently. Instead of generating one signature for each outgoing Update message, we seek ways to reduce the number of signing operations. We found two ways to aggregate messages in groups, and create one signature for each group. To help verification, the signer should also include auxiliary information in each message so that the verifiers can prove the membership of this message in the group, and thus validate the RA. Here are the message aggregation techniques.

- *Amortization across peers.* We use bit vectors to reduce the signing operations on messages sent to peers. This method is also referred as *S-A-vector*.

- *Amortization across output buffers.* We use Merkle hash trees to organize all messages in a router's output buffers as a single group. We do so incrementally, based on the vector technique in the first step. The complete S-A solution uses both techniques, which we refer to as *S-A-tree*.

Next, we discuss design details of each technique.

7.1.1 Amortization Across Peers

Recall the reason for the signature explosion : when a speaker makes an announcement, it makes it to multiple peers (typically). In BGP the messages to peers are identical; in S-BGP they are not. Security requires that the recipient be named and be part of the message that is signed. Thus, at first glance, every message must be signed individually.

There is a disarmingly simple solution. Suppose that a speaker logically enumerates its peers with indexes ranging from 0 up to the maximum number of peers, N. In practice $N < 64$. A speaker can thus use a bit vector, N bits long, to describe any subset of its peers. The idea then is to have a speaker create a bit vector that describes the full set of peers filtered to receive a route announcement, put the bit vector in the message rather than the recipient's identity, and sign the message. *This one message can be sent to all the peers, and only one new signature is involved.* If a speaker knows its logical position in a peer's enumeration, it can validate that an Update was intended for it by simply checking whether its bit is set.

The only issue left is determining how a speaker learns its logical identity in each of its peer's enumerations, and how it can prove this identity to its relying parties. A direct solution involves PKI certificates. Recall that in the S-BGP framework a speaker acquires the certificate of each peer, principally to obtain its public key. Certificate formats are general enough that we can require that a speaker's certificate name each of the speaker's peers (e.g., in an extension). We simply require that this naming include the enumeration.

This solution does require generation of new certificates when peers change, but that is a relatively infrequent event. Later in [85], Kent et al. suggested a similar approach to amortize the signing cost, by simply attaching the list of AS numbers to indicate recipients of the Update message. This AS number list is a functionally equivalent data structure to bit-vectors. The AS number list is flexible, but it does require more space in messages.

By the BGP default policy, new route announcements are sent to all peers except the one who sent the route to the speaker. Depending on the route filtering policy, the recipient set may be smaller. The filtering rules are configured statically. Thus, with high probability, a given repeated route announcement will be sent to the same set of recipients. Thus, S-A-vector is still able to exploit the caching optimization to reduce redundant cryptographic operations.

7.1.2 Amortization Across Output Buffers

While S-A-vector can be most helpful to routers with many peers, most routers only maintain a few peering sessions. Thus, we need to find another way to further increase the degree of message aggregation. The target, then, is the output buffers that routers use to keep outgoing Update messages. The message-sending process is controlled by one or multiple MRAI timers. We use the hash tree techniques of [108, 109] to amortize the cost of signing messages in output buffers.

Operationally what we do is to tag Update messages that are going into an output buffer as being "unsigned". These messages will contain bit vectors, reflecting a cross-peer aggregation that we continue to exploit. We delay the actual signature until the message is free to be transmitted—either immediately, or (if it must wait for its MRAI timer to fire) when *any one* of the MRAI timers fires. At that point the speaker tags all of the messages in all of the buffers that are tagged as "unsigned" to be "signed". Then the speaker computes the hash values of each message with its bit vector, constructs the hash tree of these hash values, and signs the root. The messages that are released by the MRAI timer are sent (possibly leaving some signed messages in other buffers, but they will not be signed again), with the signature of the tree root. The advantage to the hash-tree method is that we can sign all of these messages using just one expensive private-key operation. The use of a vector-based technique first helps us reduce the hash tree size.

More formally, suppose $M = \{\langle m_i, v_i\rangle \mid 1 \leq i \leq n\}$ is the set of n groups of unsigned Update messages collected from output buffers, where v_i is the bit vector of recipients for m_i. The goal is to generate one signature σ, such that verifiers are able to not only verify the group signature itself, but also validate that $\langle m_i, v_i\rangle$ is a member message that contributes to the group signature σ.

Let's start from a simple case. Suppose $M = \{\langle m_1, v_1\rangle, \langle m_2, v_2\rangle\}$; then we could apply the private key signing to generate

$$\sigma = \ \mathit{pk\_sign}(H(H(m_1 \parallel v_1) \parallel H(m_2 \parallel v_2)))\ ,$$

where H is a sufficiently strong cryptographic hash function, $\parallel$ denotes concatenation, and *pk_sign* is the private-key operation in a standard public-key cryptosystem. Let L, R be a simple encoding of "left" and "right", we could then use $\sigma \parallel H(m_2 \parallel v_2) \parallel R$ as the signature on $< m_1, v_1 >$. With the indicated ordering of concatenation, the verifier is

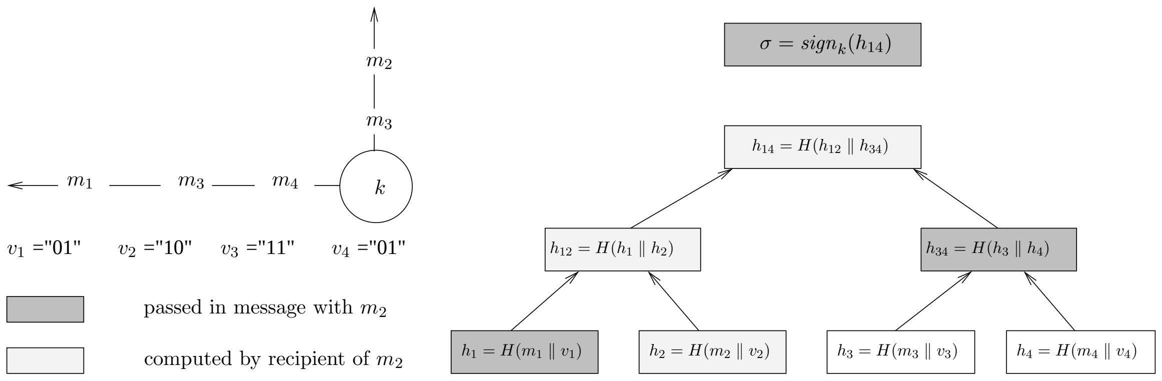

m_2
m_3
m_1
m_3
m_4
k
v_1 ="01"
v_2 ="10"
v_3 ="11"
v_4 ="01"
passed in message with m_2
computed by recipient of m_2
$\sigma = sign_k(h_{14})$
$h_{14} = H(h_{12} \parallel h_{34})$
$h_{12} = H(h_1 \parallel h_2)$
$h_{34} = H(h_3 \parallel h_4)$
$h_1 = H(m_1 \parallel v_1)$
$h_2 = H(m_2 \parallel v_2)$
$h_3 = H(m_3 \parallel v_3)$
$h_4 = H(m_4 \parallel v_4)$

Figure 7.1: BGP speaker in AS k sends four messages to two peers. The speaker computes five hash values in total to build a hash tree and sends messages with associated fields.

able to reconstruct the hash value for σ, then verify the signature. Furthermore, the hash value reconstruction process convinces the verifier that $< m_1, v_1 >$ belongs to the group of messages associated with σ.

In general, to sign a set of n messages, e.g., $n = 2^k$, we could build a binary tree of depth k and do a private-key operation on the root; the "signature" of any given message consists of the private-key signature of the root, along with the path from that message to the root, specified via k pairs of hashes and L, R values.

Let $m_1, \ldots, m_K$ be the K messages with their bit vectors $v_1, \ldots, m_K$, then a leaf of the tree T is the hash value of a message and its bit vector, $h_i = H(m_i \parallel v_i)$. Any interior node h in T is the hash value of its two child hash values. We define the root of the constructed hash tree T as $h_{1K} = Root(h_1, \ldots, h_K)$. Let $Route(h, r)$ be the path in T that allow reconstruction of r from the node h in the tree. Essentially, it contains a sequence of pairs $(h, R/L)$ that indicate the value and ordering for computing the hash value to a higher level at each step. The signer signs h_{1K}

$$\sigma = pk\_sign(r) \ ,$$

and sends $(\sigma, route(h_i, r))$ as the signature for the recipient of m_i.

Figure 7.1 illustrates these ideas with a simple example. At the time an MRAI timer fires, a router has unsigned messages for two peers, in two different buffers. A hash tree is built, the leaves of which are hashes on the individual messages. The value assigned to an interior node is the hash of the concatenation of the values of its children. The figure illustrates what accompanies m_2 when it is sent and the values that are recomputed by the receiver. Clearly, the tree path accompanied with m_2 is

$$Route(h2, r) = \{(h_1, L), (h_{34}, R)\} \ .$$

The amount of information comprising a signature grows logarithmically in the size of the tree, as does the work of verifying the message.

The default value of MRAI timer is 30 seconds. It is likely that the speaker would have several messages when MRAI fires. Thus, in most circumstances, S-A-tree renders a higher degree of aggregation than S-A-vector.

However, S-A-tree does have two drawbacks. The additional information for each signature extends the message size. The amount of information grows logarithmically in the size of each tree, and number of signatures grows linearly in the length of AS path.

Furthermore, S-A-tree eliminates the chance of further exploiting caching optimization. Although a speaker may have sent the same (m, v) pair many times, each time it is accompanied with a different set of outgoing messages, with high probability. Thus the resulting signature σ is different each time. The verifier has to perform the hash value reconstruction and public key operation anyway. In our performance evaluation, we will not explore the caching for S-A-tree.

7.2 Aggregated Path Authentication

We designed S-A for BGP path authentication in order to speed up the process of validating AS paths. The major drawback of S-A is its high cost on space, mainly on longer Update messages. Even with S-A-vector only, the resulting message size is longer than S-BGP, since extra bit vectors are needed in the message. Looking closely at the basic design of S-BGP RAs, we realize the root of the problem comes from the requirement that K attestations are needed for validating an AS path of length K. This extra information occupies a large part of the message space.

However, this requirement is not inevitable. We have found that a new type of signature algorithm—*aggregate signature*—can be used to reduce the number of signatures for validating an AS path. This new technique requires much less data for a route announcement, so that we can dramatically reduce message size, as well as the memory cost.

Consequently, we designed a new path authentication scheme: *Aggregated Path Authentication (APA)* that combines S-A, the time efficient scheme, with aggregate signature, the space efficient scheme [168]. What we have achieved is a new efficient BGP path authentication scheme that out-performs S-BGP RAs in both time and space criteria.

7.2.1 Aggregate Signatures

An *aggregate signature* is a digital signature that supports aggregation: given n signatures on n distinct messages from n distinct users using an aggregate signature algorithm, it is possible to aggregate all these signatures into a single short signature. This single signature (and the n original messages) will convince the verifier that the n users did indeed sign the n original messages.

There are two main proposals of constructing aggregate signatures [15], *general* aggregate signature schemes and *sequential* aggregate signature schemes.

General Aggregate Signature

The first approach is based on a co-GDH signature scheme, which can be based on any gap group. The short signature scheme by Boneh, Lynn, and Shacham *(BLS)* [17] is such a co-GDH signature scheme that makes use of elliptic curves. It is referred to as a "general" aggregate signature, since the aggregation algorithm is public—given n signatures

$\sigma_1, \ldots, \sigma_n$, anyone can use a public aggregation algorithm to take all n signatures and calculate the aggregate signature σ. The aggregation algorithm uses a bilinear map between two (multiplicative) cyclic groups of prime order p, G_1 and G_2. Given an additional group G_T, such that $|G_1| = |G_2| = |G_T|$, a bilinear map is a map $e : G_1 \times G_2 \rightarrow G_T$ with the following properties:

- Bilinear: for all $u \in G_1, v \in G_2$ and $a, b \in \mathbb{Z}, e(u^a, v^b) = e(u, v)^{ab}$.
- Non-degenerate: $e(g_1, g_2) \neq 1$, where g_1 is a generator of G_1 and g_2 is a generator of G_2.

We briefly summarize the *Sign*, *Verify*, *Aggregate*, and *Aggregate Verify* algorithms as the following. Let H be a hash function.

- **Sign** For a particular user, the algorithm works as a normal co-GDH signature scheme. Given the private key x and a message M, compute $h \leftarrow H(M)$, where $h \in G_2$, and $\sigma \leftarrow h^x$. $\sigma \in G_2$ is the signature.
- **Verify** Given a user's public key $v = g_1^x$, the message M, and the signature σ, compute $h \leftarrow H(M)$; accept if $e(g_1, \sigma) = e(v, h)$ holds.
- **Aggregate** For a set of users U, where $|U| = k$, given signatures $\{\sigma_i \in G_2 \mid 1 \leq i \leq k\}$ on messages $\{M_i \mid 1 \leq i \leq k\}$, compute $\sigma \leftarrow \prod_{i=1}^{k} \sigma_i$. The aggregate signature is $\sigma \in G_2$.
- **Aggregate Verify** Given the aggregate signature σ, the message set $\{M_i \mid 1 \leq i \leq k\}$ on which it's based, and public keys $v_i \in G_1$ for all users $u_i \in U$, verify the aggregate signature σ in two steps:
  1. ensure that the messages M_i are all distinct, otherwise reject; and
  2. compute $h_i \leftarrow H(M_i)$ for $1 \leq i \leq k$, and accept if $e(g_1, \sigma) = \prod_{i=1}^{k} e(v_i, h_i)$ holds.

Like a co-GDH signature, a bilinear aggregate signature is a single element of G_2. Note that the aggregation can be performed incrementally. This way, the aggregation is as fast as a modular multiplication. The calculation involved in the verify and aggregate verify algorithms is mostly the mapping e, which can be implemented using pairing calculations. We discuss more details on how to compute pairing efficiently later.

Sequential Aggregate Signatures

A sequential aggregate signature scheme [99] is based on homomorphic trapdoor permutations, such as RSA. Each signer incrementally signs the new message and incorporates it into the aggregate signature σ. A party with knowledge of n messages, public keys of the n ordered signers, and σ is able to verify that each signer s_i has correctly signed his message M_i and σ is a valid sequential aggregate signature. The designers also showed how to instantiate the construction with the RSA trapdoor permutation. Briefly, we review the resulting RSA aggregate signature scheme for n users with moduli of length l in the following. Let $H : \{0,1\}^* \rightarrow \{0,1\}^{l-1}$ be a hash function. Note that the following version requires that the moduli must be ordered. However, the moduli can be unrestricted with a few additional steps in the algorithm.

- **Key generation** Each user i generates an RSA public key (N_i, e_i) and private key (N_i, d_i).
- **Aggregate Sign** `AggrSign`(σ', M_i): As the base case, let $i = 0$, and the initial aggregate signature $\sigma' \leftarrow 0$ on an empty message set. For the ith signer, given a valid aggregate signature σ' on previous $i-1$ messages $\{M_1, \dots, M_{i-1}\}$ and public keys, she computes
$h_i \leftarrow H((M_1, \dots, M_i), ((N_1, e_1), \dots, (N_i, e_i)))$, $y \leftarrow h_i + \sigma'$, and outputs $\sigma \leftarrow y^{d_i} \mod N_i$.
- **Aggregate Verify** Given aggregate signature σ on i messages $\{M_1, \dots, M_i\}$, and public keys, the verifier does the following checks:
1. public keys satisfy requirements;
2. check that $0 \leq \sigma \leq N_i$;
3. if $\gcd(\sigma, N_i) = 1$, let $y \leftarrow \sigma^{e_i} \mod N_i$, else let $y \leftarrow \sigma$;
4. compute $h_i \leftarrow H((M_1, \dots, M_i), ((N_1, e_1), \dots, (N_i, e_i)))$ and $\sigma' \leftarrow (y - h_i) \mod N_i$;
5. verify σ' recursively;
6. accept if $\sigma = 0$ holds when $i = 0$, reject otherwise.

7.2.2 Aggregated Path Authentication

Now, we describe the aggregated path authentication schemes that apply S-A together with aggregate signature schemes. Again, S-A is a good candidate to speed up processing Updates, while aggregate signatures can shorten the message size and release the memory burden. The number of signatures in a route announcement is no longer linear in the length of the AS path. One aggregate signature is enough to show the authenticity of the entire AS path.

Using S-A, we have two choices. Recall that S-A amortizes signing cost in two steps. We may choose to use S-A-vector or S-A-tree. Although S-A-vector may not result in a high degree of signing amortization, it allows us to apply two additional optimizations. First, since speakers no longer enclose hash paths in Update messages to help signature verification, we can certainly further reduce the storage consumption. Second, S-A-vector can use caching optimization.

As discussed in Section 7.2.1, we also have two choices on aggregate signature schemes. General aggregate signatures are based on the short signature scheme (BLS). For standard security parameters, the signature length is about half that of a DSA signature with a similar level of security. Sequential aggregate signatures are implemented using the RSA trapdoor permutation, thus the signature length is the same that of a RSA signature with the same level of security. Certainly, general aggregate signatures outperform in terms of space. However, the aggregate verify operation provided by the sequential aggregate signature scheme may be substantially faster than the one provided by general aggregate signature scheme. Since the majority of the cryptographic operations performed by BGP speakers are verifications, the sequential aggregate signature scheme potentially can be the winner in terms of speed.

To achieve the most efficient aggregate path authentication scheme, we design four constructions by combining the choices we have for S-A and aggregate signatures. We then use network simulation to evaluate their performance and to identify the most efficient scheme.

First, we define some notation. Let (pa, p) be the current route announcement, where p is the announced prefix and pa is the associated AS path to be sent. Let v be the bit vector indicating the recipients of the route. Let σ' be the aggregate signature on the previous route announcement and σ be th newly generated aggregate signature on the updated route announcement. Let $a \parallel b$ stand for concatenating a with b. We now consider our four constructions.

GAS-V BGP speakers organize outgoing route announcements using a bit vector v, generate BLS signature $s \leftarrow sign(pa \parallel p \parallel v)$, and compute the aggregate signature $\sigma \leftarrow \sigma' \cdot s$. The outgoing route announcement contains the route, the bit vector, and the aggregate signature, $\{(pa, p, v), \sigma\}$. For caching, speakers can cache the route (pa, p, v) and associated aggregate signature σ to avoid duplicated signing or verification. Furthermore, we can use either software or hardware implementation of the pairing calculation for verify and aggregate verify operations, we denote these two variants as *GAS-V(SW)* and *GAS-V(HW)*.

GAS-T BGP speakers organize outgoing route announcements using a bit vectors, construct a hash tree (the resulting root of the tree is R), generate BLS signature $s \leftarrow sign(R)$, and compute the aggregate signature $\sigma \leftarrow \sigma' \cdot s$. The outgoing route announcement contains the route (pa, p), the bit vector v, the hash path in the tree, and the aggregate signature σ. BGP speakers do not cache any signed or verified routes, their aggregate signatures, or hash paths. Similar to GAS-V, there are two variants for GAS-T—*GAS-T(SW)* and *GAS-T(HW)*.

SAS-V BGP speakers organize outgoing route announcements using a bit vector v, and generate aggregate signature σ using the function $\texttt{AggrSign}(\sigma', pa \parallel p \parallel v)$. The outgoing route announcement contains the route, the bit vector, and the aggregate signature, $\{(pa, p, v), \sigma\}$. For caching, speakers can cache the route (pa, p, v) and associated aggregate signature σ to avoid duplicated signing or verification.

SAS-T BGP speakers organize outgoing the route announcements using bit vectors, construct a hash tree (the root of the tree is R), and generate aggregate signature $\sigma \leftarrow \texttt{AggrSign}(\sigma', R)$. The outgoing route announcement contains the route (pa, p), the bit vector v, the hash path in the tree, and the aggregate signature σ. BGP speakers do not cache any signed or verified routes, their aggregate signatures, or hash paths.

7.3 Sequential Aggregate Signature for S-BGP RAs

During the process when we explore the aggregated path authentication schemes, we did try a simple step. That is, we applied the sequential aggregate signature scheme directly to S-BGP route attestations.

Similar to APA schemes, a single aggregated signature is enough to for the recipients to validate the entire AS path. Different from APA schemes, the data aggregated by the algorithm is only the signatures for S-BGP route attestations. In our study, we simply denote this scheme as *SAS*. Like S-BGP route attestations, SAS can use utilize caching optimization to avoid redundant cryptographic operations.

Lysyanskaya et al. suggested that S-BGP path authentication is a potential application of sequential aggregate signature scheme[99]. Our study is the first report that evaluates the performance of SAS for S-BGP path authentication.

7.4 Performance Evaluation

In evaluating the performance of aggregated path authentication, our main goal is to use quantitative results to identify the most efficient techniques for BGP path authentication. We evaluate S-A, SAS, and all APA constructions. We compare the experimental results with the performance of S-BGP route attestations.

7.4.1 Configurations and Benchmarks

In the configuration, all vector-based constructions, such as S-A-vector, GAS-V, and SAS-V, apply caching optimization. Previous experiments have shown the effectiveness of caching, hence we treat caching as a default configuration this time. S-A-vector is slightly different. We use it only to demonstrate potential memory cost by S-A if caching is applied. Other than that, we use S-A-tree in comparison with other aggregated path authentication constructions.

It's straightforward to decide the unit length of data structures involved in path authentication. For similar levels of security, S-BGP uses the DSA algorithm with SHA-1, which results in 40-byte signatures and 20-byte hash values. A BLS short signature is 20 bytes long, and thus so is the aggregate signature by the general aggregate signature scheme. An RSA signature is 128 bytes long, which is also the length of a sequential aggregate signature. In building the hash trees, S-A uses SHA-1. Thus hash values are 20 bytes long.

We obtained the running times for standard signature algorithms, such as RSA and DSA, by benchmarking the OpenSSL crypto library. However, OpenSSL does not have implementations of aggregate signatures. Fortunately, we can decompose for calculation of each algorithm, obtain the running time of each step, and combine to estimate the total running time. We consulted the community and the literature from other crypto libraries or implementations [10, 16, 114].

The implementation of sequential aggregate signatures just requires minor modification of the RSA algorithm. SAS aggregate sign is the same as the ordinary RSA signing algorithm plus a few additional big number calculations, whose running times are negligible compared with the signing operation. The recursive aggregate verification algorithm works through layers of the aggregation until the base case. Thus the running time can be estimated as the time of k RSA verifications plus a bit of time for extra calculation.

To understand the running times by general aggregate signatures based on elliptic curves, we turn to the literature on pairing-based cryptosystems. From [10], we learn that it takes about 2.2 ms to sign a message using BLS. Aggregation on two BLS signatures needs another modular multiplication on 157-bit numbers, whose running time is negligible compared with signing.

Algorithms	Running Time (ms)
Miller's Algorithm on $\mathbb{F}_{3^{97}}$	24.0
BKLS on $\mathbb{F}_{3^{97}}$	23.6
Refined Duursam-Lee on $\mathbb{F}_{3^{97}}$ [58]	16.8
Modified Duursam-Lee on $\mathbb{F}_{3^{97}}$ [11]	8.6
Hardware implementation [86]	1.3

Table 7.1: Running times of Tate pairing calculation. Running times by software implementations are normalized to a 1 GHz processor. The hardware implementation assumes a conservative 10 MHz clock frequency on the target technology.

	RSA 1024-bit	DSA 1024-bit	SAS 1024-bit	GAS based on $\mathbb{F}_{3^{97}}$
Sign (ms)	50.0	25.5	50.0	11.0
Verification (ms)	2.5	31.0	2.5	43.0×2
SW Aggregate Verification (ms)	–	–	$2.5 \times k$	$43.0 \times (k+1)$
HW Aggregate Verification (ms)	–	–	–	$1.3 \times (k+1)$
Aggregate Sign (ms)	–	–	50.0	11.0
Signature length (bytes)	128	40	128	20

Table 7.2: Benchmarks of signature algorithms with the same level of security. Running times are normalized to a 200 MHz CPU, except for the hardware implementation of aggregate verification. We assume that aggregate verification handles k distinct messages. Signature length for the general aggregate signature is based on BLS on $\mathbb{F}_{3^{97}}$. For the same level of security, BLS renders 157-bit signatures.

To estimate verification and aggregation verification performance, we need to understand the pairing calculation. The pairing calculation in the verification and aggregate verification operations is relatively slow compared with scalar point multiplication in signing operations. In the general aggregate signature scheme, one verification is composed of two pairing calculations, and an aggregate verification on k distinct messages requires $k + 1$ pairing calculations. In recent years, an ever-increasing number of pairing-based cryptosystems have appeared in the literature [140]. In turn, this has driven research into efficient algorithms for the implementation of bilinear pairing on elliptic curves. To date, the Tate pairing [48] has attracted attention as the most efficiently computable bilinear pairing on elliptic curves. In particular, the Tate pairing over supersingular elliptic curves achieves its maximum security in characteristic three. The classic algorithm for Tate pairing computation on elliptic curves is Miller's algorithm [112, 113]. Later, BKLS/GHS algorithms furthered this development so that the Tate pairing became easier to compute in practice [10, 49]. Duursma and Lee [35] further improved the Tate pairing calculation and extended it to more general hyperelliptic curves. Yet even more enhancements to the Duursam-Lee Algorithm have appeared for supersingular elliptic curves over fields of characteristic three [11, 58, 148].

Accompanied with huge improvements in software implementations of Tate pairing, there are a few efforts on developing hardware to calculate pairings efficiently [57, 86, 130]. The main observation lies in the fact that arithmetic architectures in the extension field

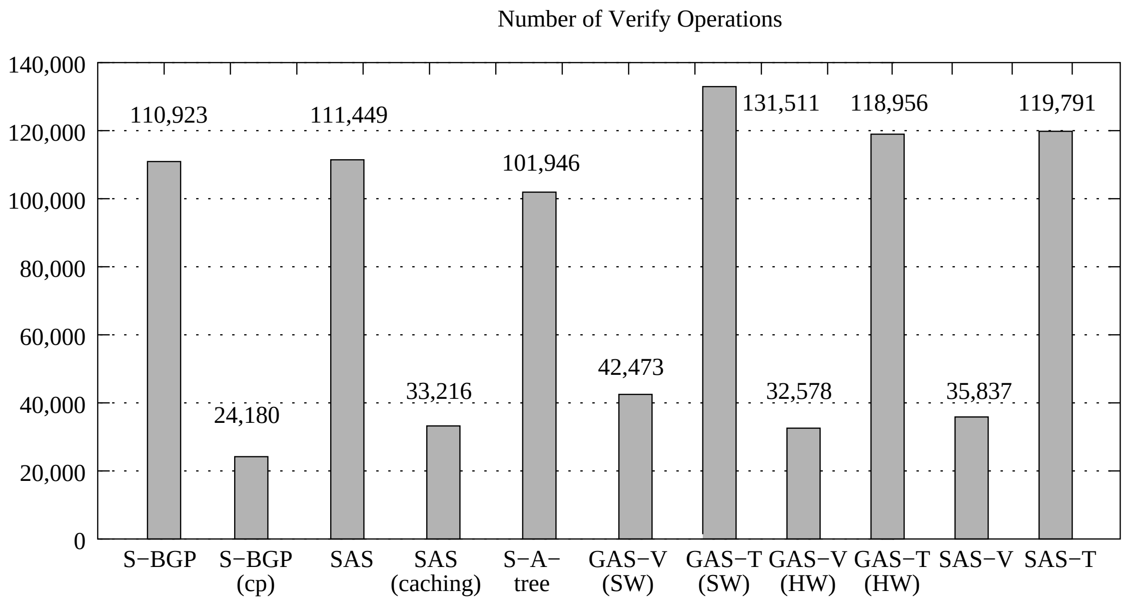


Figure 7.2: Counts for verify operations. For one aggregated verification on k messages, we count it as k verifications. We also treat the pairing calculation the same way as for normal signature verifications.

$GF(3^{6l})$ are good candidates for parallelization, leading to a similar calculation time in hardware as for operations over the base field $GF(3^m)$ [86]. Table 7.1 summarizes the running times of pairing calculations. We chose the running times of the most efficient software optimization and hardware acceleration for our simulation experiments. Table 7.2 illustrates our estimation of running time and signature length. We use these numbers as parameters in the simulation experiments.

7.4.2 Performance Results

Signatures and Verifications

To ease the comparison, we transform the aggregate verifications as several normal verification operations. For GAS-based constructions, we count one aggregate verification operation to validate an AS path of length k as $k + 1$ single verifications, each of which mainly involving a pairing calculation. Similarly for SAS-based constructions, including SAS, we count one aggregate verification operation as k single ones, each mainly involves a normal RSA verification. For singing numbers, normal signing and aggregate signing are treated identically.

Figures 7.2 and 7.3 show the number of verification and signing operations during router rebooting. SAS with or without caching optimizations needs roughly the same number of verification and signing operations as the S-BGP with or without caching optimization. S-A and aggregated path authentication schemes do not help in reducing verifications. Caching optimization works effectively for all of them. The reducing rate is 3–4. On the other hand, S-A and aggregate path authentication dramatically reduce signing operations. From the

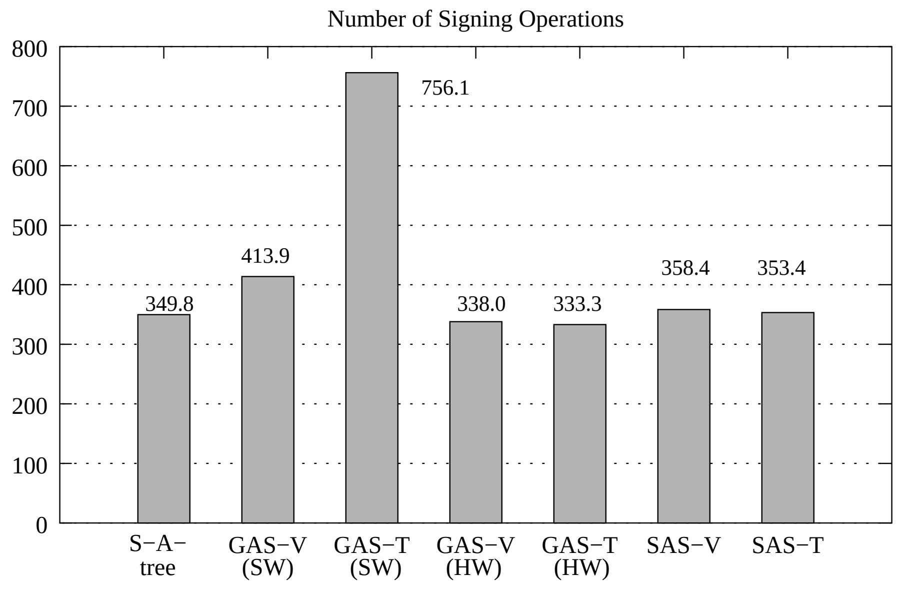


Figure 7.3: Counts for signing operations. Numbers for S-BGP, S-BGP(cp), SAS, and SAS with caching are 22,072, 11,521, 22,303, and 12,035 respectively. They are too large to be shown in this figure.

experimental data, we have made an important observation:

> **Observation:** Signature amortization techniques help to reduce up to 98% of signings by S-BGP.

The number of signing operations for S-BGP and S-BGP (cp) are 22,072.3 and 11,521.9 respectively, which are too large to be shown in Figure 7.3. For S-A-tree, the average message aggregation rate is about 60. In other words, using bit vectors and hash trees, S-A-tree manages to collapse 60 expensive private-key operations into one. Aggregate rates by vector-based constructions are smaller. With the help of caching optimization, they can achieve similar level savings.

Surprisingly, GAS-T (SW) is less efficient; it almost doubles the number of signings compared with other aggregated path authentication constructions. The cause is its slow signature verifications. Recall that messages arrive and are processed in "waves", as discussed in Section 6.2.2. The pairing calculation using software implementation is so slow, speakers spend a much longer time processing received routes. Hence, the number of Update messages that are able to make it into the output buffers before the next MRAI timer fires is significantly reduced. The aggregate rate completely depends on the number of unsigned messages that GAS-T can use to construct a hash tree. GAS-T (SW) is only able to achieve a 28.2 aggregate rate.

Time Delays

Figure 7.4 illustrates that CPU time spent for path authentication schemes varies dramatically. Most constructions of aggregated path authentication as well as S-A-tree benefit from

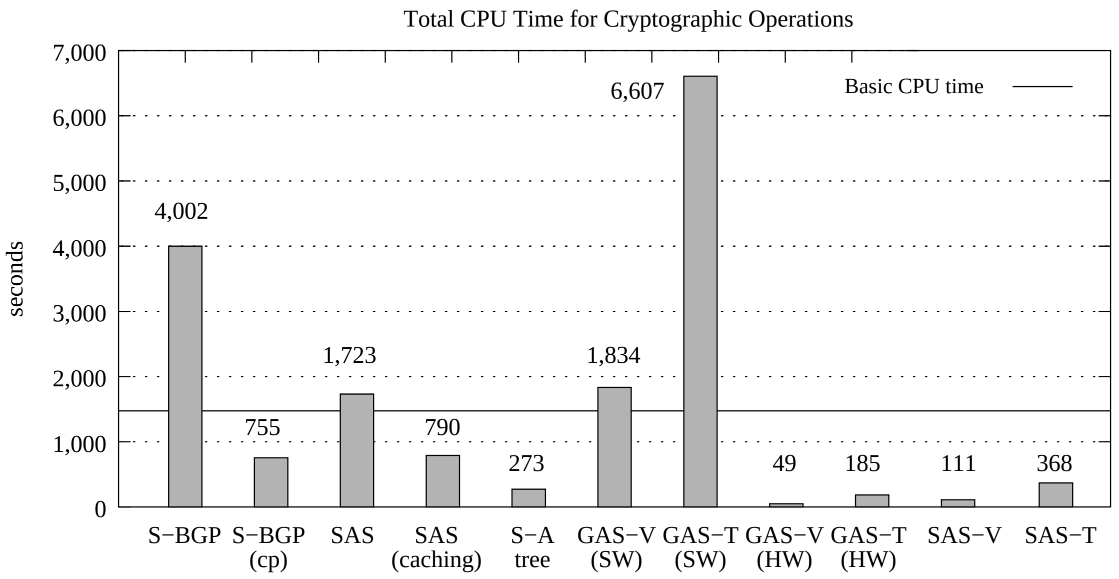


Figure 7.4: CPU time spent for cryptographic operations. The horizontal line shows the CPU time spent for normal Update processing.

huge savings on signing operations. They spend little time on cryptographic operations. Schemes that use software implementation of Tate pairing are an exception. GAS-V (SW) is expensive because of slow pairing calculations. GAS-T (SW) is even more expensive than S-BGP, since the saving rate for each signing operation is not good enough either.

Figure 7.5 compares convergence time for the various schemes. SAS schemes, our first step in saving space, significantly prolong the convergence process. According to Figures 7.2 and 7.3, SAS schemes require a similar number of cryptographic operations as S-BGP. The timing of these operations is quite different. Although a verify operation by sequential aggregate signature algorithm is more than 10 times faster than a DSA verification, the signing operation is 2 times slower. The slowing down of signing has a more significant impact on convergence time than the faster verification. This result confirms our observation made in Chapter 6.

In contrast, aggregated path authentication constructions are able to speed up the convergence up to 69%. Instead of almost six minutes more, we are able to achieve a fast convergence that is only a few seconds longer than normal BGP processing. There are three primary cases of correlations between cryptographic delays and convergence time. GAS-V (HW), S-A-tree, and GAS-T (SW) represents them respectively.

Let's examine our simple processing model again. As a reminder, let's first repeat Equation 5.3 we defined for analysis.

$$F + \mathtt{Pr}\left\{\mathtt{routepreferred}\right\} \times \left(L \times C_v \times \mathtt{Pr}\{\mathtt{verify}\} + N_p \times C_s \times \mathtt{Pr}\{\mathtt{sign}\}\right) . \quad (7.1)$$

The processing latency for an Update by S-BGP is estimated as the following:

$$Cost_{S-BGP} = F + \mathtt{Pr}\left\{\mathtt{routepreferred}\right\} \times \left(5.4 \times C_v + 24 \times C_s\right) , \quad (7.2)$$

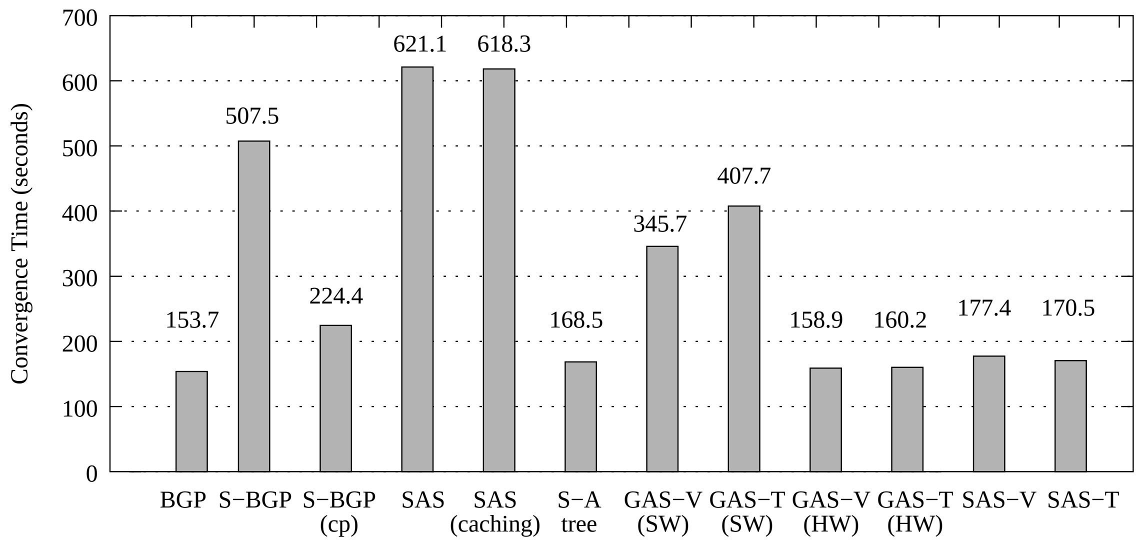


Figure 7.5: Convergence time comparison for S-BGP, S-A, and aggregated path authentication constructions.

where C_v and C_s are the verification cost and signing cost by standard DSA. The operation costs by other algorithms are normalized to C_v and C_s. Taking into account all saving rates on verification and signing, the processing delays for an Update message are approximately

$$Cost_{GAS-V\ (HW)} = F + \texttt{Pr}\{\texttt{routepreferred}\} \times (0.08 \times C_v + 0.16 \times C_s) \tag{7.3}$$

$$Cost_{S-A-tree} = F + \texttt{Pr}\{\texttt{routepreferred}\} \times (0.44 \times C_v + 0.8 \times C_s) \tag{7.4}$$

$$Cost_{GAS-T\ (SW)} = F + \texttt{Pr}\{\texttt{routepreferred}\} \times (8.88 \times C_v + 0.35 \times C_s)\ . \tag{7.5}$$

Now, we analyze these three cases by comparing them with S-BGP.

GAS-V (HW) converges the fastest among all constructions. Compared with S-BGP, it costs less for verification and signing operations since all cryptographic operations are faster than DSA used by S-BGP.

Equation 7.4 shows that the cost of processing one Update by S-A-tree is strictly less expensive than S-BGP. However, it comes from a significant saving on number of signing operations. RSA signing is about 1.96 times more expensive than DSA signing.

GAS-T (SW) represents the third case. Figure 7.4 has shown that GAS-T (SW) spends many more CPU cycles on cryptographic operations. However, it converges about 100 seconds faster. Equation 7.5 suggests that the rebooting router spends more time on verifications, but less time on signing. Once again, we have come to the same observation we have discussed in Section 6.2.2: convergence time is more sensitive to signing cost.

Message Size

Our experiment results presented in Figure 7.6 further conclude that GAS-V (HW) is not only the fastest in convergence time, but is also the most economic in space. Note that GAS-V (or GAS-T) with either software or hardware pairing calculations presents a similar

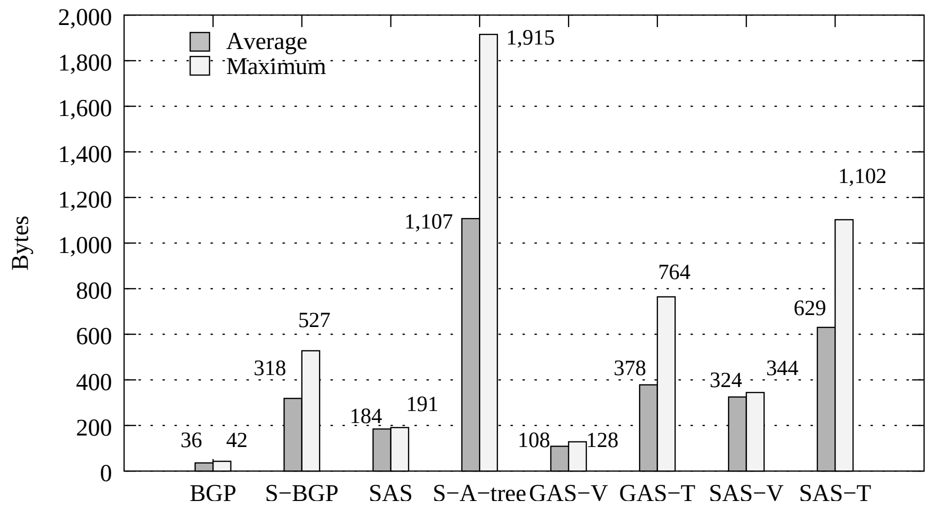


Figure 7.6: Message size by S-A and aggregated path authentication constructions.

performance on space. Hence, we simply illustrate experimental results for hardware pairing calculation in Figure 7.6. Among all aggregated path authentication constructions, GAS-V produces the shortest Update messages. Using GAS-V, we have successfully shortened S-BGP messages by 66%.

On the other hand, although S-A-tree converges as quickly as GAS-V (HW), it produces extremely long Update messages. The extra cost comes from longer signatures, bit vectors, and paths in hash trees for each AS number in AS path.

As mentioned above, the aggregated path authentication schemes are all capable of fast convergence. For the resulting message size, however, tree-based schemes generate longer Update messages, because of the extra hash values carried in the messages. Moreover, GAS-based schemes outperform SAS-based schemes on message size, because of much shorter aggregate signatures (20 bytes vs. 128 bytes). Vector-based schemes (GAS-V and SAS-V) have another nice feature—the maximum message size is close to the average size. This feature gives us confidence that vector-based schemes will have no difficulty complying with the Update message MTU limit, in the simulated network.

Memory Cost

We also compare the signature memory overhead for caching schemes with the plain BGP. The numbers shown in Figure 7.7 are the average memory overheads for caching signatures on one BGP speaker. This includes signatures for AAs as well as for RAs. For completeness of experiments, we include the memory overheads on AAs for all of the path authentication schemes. GAS-V, again, achieves the best performance. The resulting overhead is only about 28% the amount spent by S-BGP. Note that the current criticism on S-BGP practicality is mainly on extra memory burden [72]. Our experiments indicate that GAS-V is not just efficient—it can be a practical security solution for BGP path authentication.

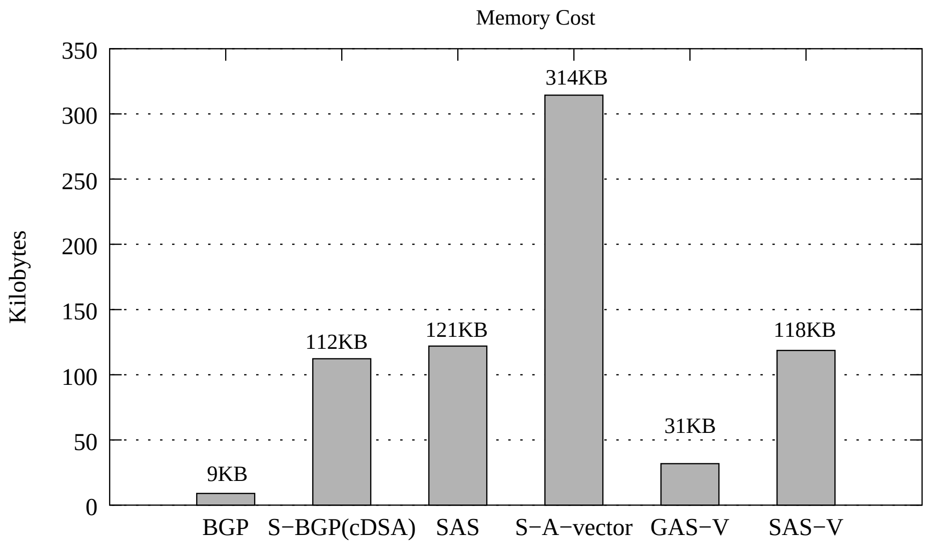


Figure 7.7: Memory cost by S-A and aggregated path authentication constructions.

7.5 Discussion

We design aggregated path authentication based on our thorough understanding of performance issues in BGP processing and S-BGP. We have discovered efficient path authentication schemes that use faster cryptography, perform operations less frequently, and render shorter signatures. Aggregated path authentication is a more practical path authentication scheme without compromising security. Overall, GAS-V (HW) becomes the winner of the performance competition. Table 7.3 summarizes its achievements on performance improvement over S-BGP and S-BGP(cp).

Criteria	**Improvement over S-BGP**	**Improvement over S-BGP(cp)**
Convergence	88.4%	29.2%
Message Size	65.9%	–
Memory cost	–	72.3%

Table 7.3: Performance improvements by GAS-V (HW).

7.5.1 The Real World

Limited by the scale of the simulation, we are not able to model the entire Internet to study the performance impacts. Using the publicly available BGP data, we could get a sense of what it looks like if an aggregated path authentication scheme is deployed in the Internet and how much it can improve based on S-BGP.

To make our discussion complete, we use the BGP routing table dump from Route-Views to understand the memory cost on a real router. This table dump was collected from the AS6447 archive on May 04, 2005, which takes 209MB in MRT[116] format. It contains 162,237 unique IP prefixes and 2,011,005 unique (AS path, prefix) tuples. To cache

	S-BGP	GAS-V	Saving Rate
Routing Table	209MB	209MB	–
Signature memory for AAs	77MB	38MB	51%
Signature memory for RAs	316MB	70MB	65%
Signature memory increase	188%	52%	73%
Memory for Certificates	75–85MB	75–85MB	–
Overall memory increase	229%	92%	60%
Adjusted memory increase	204%	68%	67%

Table 7.4: Memory cost comparison of S-BGP and GAS-V in the real world. We assume that the optimization on certificates saves 60% of the memory. The adjusted memory increase reflects this optimization.

all received S-BGP route attestations, the router should record 8,284,042 DSA signatures, which requires about 316MB in cache. Moreover, if the router also caches signed address attestations, it should consume another 76.8MB at least. In total, we are looking at 393MB memory cost, adding more than 180% to the BGP routing table size. In fact, we underestimate the amount by looking at signature length only. As discussed in [85], the average size would be larger since other information is involved in each attestation. We specifically chose to examine signature length only, so that we could have appropriate comparison with aggregated path authentication schemes.

Now, let us calculate the space requirement for the GAS-V scheme, the most efficient scheme shown in the simulation. We assume that the bit vectors take 4 bytes on average. Since there are 2,011,005 unique (AS path, prefix) tuples, GAS-V will take 70MB of memory to cache all received and sent aggregate signatures and bit vectors, and 38MB to cache all address attestations (108MB in total). This number is substantially smaller than the one for S-BGP. The actual memory cost for signatures only is reduced to less than 29%, increasing the routing table size by only 52%. We believe GAS-V is much more manageable and feasible for real-world deployment.

To deploy S-BGP, we must consider the memory overhead for all necessary data structures, one of them is storage for certificates. In the simulation, we just assume every path authentication scheme uses the same amount of memory for certificates as the common base for comparison. Now, we pay closer attention to this amount. As discussed in [85], the scale of the Internet in 2003 required 75–85MB memory on a BGP speaker to store all necessary public key certificates. Taking this amount into account, we conclude that the overall savings by GAS-V against all memory overheads by S-BGP is 60%.

Kent et al. proposed to let ISPs extract only necessary information from certificates and AAs, and push the data to their routers [83]. Routers do not need to store signatures for certificates and AAs. This optimization may save 50%–60% of the corresponding memory. We adjust the above calculation accordingly. The resulting overall memory saving is estimated as 67%. Moreover, if we consider the extracted information from certificates, the space impact of different signature algorithms comes from their key sizes. Shown in Table 5.7, the 1024-bit DSA with 160-bit exponent requires about 408 bytes to store the public key and domain parameters. This number is higher than the key size for BLS or RSA, which suggests that we can expect the overall memory saving to be slightly higher than 67%.

As noted in [85], for S-BGP in 2003, the Adj-RIBs space required for RAs is about 30–35MB per peer, and the total requirement for a speaker with tens of peers may be gigabytes. However, currently deployed BGP speakers cannot be configured with more than 128M or 256M of RAM. With aggregated path authentication, we still call for additional RAM on routers, unfortunately. The overall 67% memory savings certainly reduce the gap between the memory demands and reality[1].

7.5.2 Hardware Acceleration

Another interesting issue brought up by aggregated path authentication schemes is the hardware acceleration for pairing calculation. Recent rapid developments in improving pairing calculation are driven by applicability to many new EC-based cryptosystems and protocols. To give a few examples, pairing-based systems can support identity-based encryption systems, efficient key agreement protocols, credentials and secret handshakes, provable secure signatures, short signatures, group signatures, blind signatures, proxy signatures, multi-signatures, threshold signatures, intrusion-resilient encryption systems, etc. Many studies have designed and prototyped significantly improved efficient software/hardware implementations of pairing calculation that make these cryptosystems practical. For instance, in 2001 when short signature was first invented, the authors reported 2.9 seconds for verification [16]. Then in 2004, the reported running time for BLS verification was reduced to 45.2 *ms* [10]. Now, hardware parallelization allows the pairing calculation to be accomplished within 1.8 *ms* [86]. Our experiments confirm that such hardware implementation allow us to achieve efficient BGP security in both speed and space.

In addition, the aggregate verification algorithm in the general aggregate signature scheme provides us even more opportunities for potential hardware parallelization. As discussed in Section 7.2.1, the core calculation for aggregate verification is to test whether the following equation holds: $e(g_1, \sigma) = \prod_{i=1}^{k} e(v_i, h_i)$. The pairing calculations on the right hand side are all independent. For a typical small value of k, it should be possible to design a hardware implementation that further parallelizes the calculations. In fact, k is fairly small in the real world. Current BGP CIDR report shows that AS paths are of length 3.7 on average and 11 maximum [25].

Moreover, we envision wide deployment of such hardware accelerators for cryptographic calculations not only for its efficiency, but also for its practicality and usability in the real world.

[1]We do not exclude the possibility of adding more memory on routers. If one cannot add memory to a router because the vendor provided no additional slots for more DIMMs, then any scheme that exceeds the memory limitations of deployed routers will be decried as not deployable.

Chapter 8

Public Key Infrastructure Background

Nowadays, everyone is concerned about protecting information. Everyone agrees that security is necessary. The question is how to *implement* it. The public key infrastructure (PKI) is one of the powerful tools that protect information for services and systems [73]. However, implementing functions of PKI involves complexities. A challenging problem is to help users use certificates and render trust in other parties. The certification path discovery problem is our focus here. In this chapter, we review the basics of public key infrastructures and certification path discovery. We also discuss their performance issues in terms of implementation and deployment.

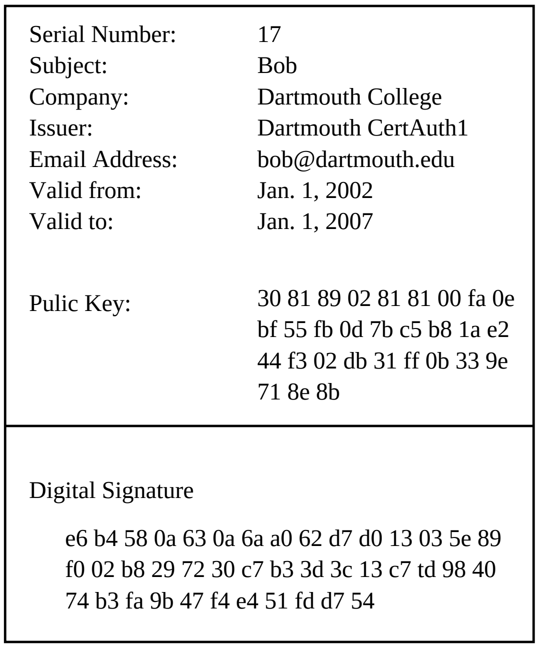
8.1 Public Key Infrastructure

Public key cryptography is a revolution. Compared with symmetric key cryptography, people no longer need to share a secret beforehand whenever they want to initiate secure communication with other people. Public key cryptography is much more flexible and scalable. Public key infrastructure (PKI) was first created [88] for securely distributing public keys. Not only that, PKI has evolved to be an architecture that provide comprehensive services for public keys—to store and distribute public keys securely, to maintain and change status information related to public keys reliably, and to help users establish trust in public keys.

8.1.1 PKI Basics

A *public key certificate* (or simply certificate) is the most basic element of a PKI. Each certificate contains a public key and identifies the user with the corresponding private key. The complete PKI system consists of a number of service components that handles certificates. For convenience of discussion, we have divided PKI into five primary components.

- Certificates
- Storage and retrieval services



Serial Number:	17
Subject:	Bob
Company:	Dartmouth College
Issuer:	Dartmouth CertAuth1
Email Address:	bob@dartmouth.edu
Valid from:	Jan. 1, 2002
Valid to:	Jan. 1, 2007
Pulic Key:	30 81 89 02 81 81 00 fa 0e bf 55 fb 0d 7b c5 b8 1a e2 44 f3 02 db 31 ff 0b 33 9e 71 8e 8b

Digital Signature

e6 b4 58 0a 63 0a 6a a0 62 d7 d0 13 03 5e 89
f0 02 b8 29 72 30 c7 b3 3d 3c 13 c7 td 98 40
74 b3 fa 9b 47 f4 e4 51 fd d7 54

Figure 8.1: Bob's certificate.

- Certificate status information services
- PKI Architecture
- Validation services

We discuss each of these components in the following sections.

8.1.2 Certificates

A public key certificate is a data structure that contains the public key and further information associated with it. Typically, a certificate serves as a *binding* between the public key and the principal, which identifies the user with the corresponding private key. A principal can be human, server, client machine, software, privilege, or anything involved in public-key based communication. The entire contents of the certificate are digitally signed by the issuer.

The *issuer* creates and signs the certificate. The issuer may be a trusted authority, such as *Certification Authority (CA)* or *Attribute Authority (AA)*. Semi-trusted entities or untrusted entities may also be issuers of certificates.

The *subject* is actually the principal in the certificate. The certificate represents the binding of the subject and its public key. The subject may stand for an entity, such as an authority or an end user. In traditional PKI systems, end users are referred to as *end entities (EEs)*. Typically they do not issue certificates to other entities. In more flexible PKI design, EEs can issue special-purpose certificates to other EEs.

Figure 8.1 demonstrate a simple sample certificate. It binds Bob's public key with his identity as a member at Dartmouth College. As discussed by Housley and Polk [73], a public key certificate is an approximation of the concept of *ideal certificates.* They define an ideal certificate with the following nine features:

1. It is a pure digital object that allows distribution and processing automatically.
2. It contains the identity of the user who holds the private key. The identity refers to the name, the organization, and contact information of the user.
3. It is easy to verify the freshness of the certificate. In other words, the certificate is time-stamped by an issuance date, and it should be easy to compare the current time with the issuance date.
4. The issuer of the certificate would be a trusted party.
5. Th certificate should be uniquely and easily distinguished among a large number of certificates.
6. It should be easy to protect the certificate from forgery. In other words, we can easily determine the forged certificates.
7. After the certificate is created, it should stay untampered. Any unauthorized change on the certificate should be easily identified.
8. The certificate should have an expiry date. Once current time has passed the expiry date, we can determine that the information in the certificate is no longer current.
9. The certificate should indicate the application purpose, so that we can determine if the certificate is appropriate for an application.

Besides these nice features, we also need to establish additional services to handle certificates. Other components of PKI come into play. Our discussion in this chapter as well as in this thesis will be more focused on X.509 PKI. We also review other types of PKIs. They provide similar services with similar components.

Some PKI Systems

There are various types of PKI that are widely deployed or have been proposed [40, 71, 143]. They differ in the configuration information required, trust rules, and flexibility. The pre-history of PKI goes back to Diffie and Hellman's seminal paper on public-key cryptography [32]. They proposed a key directory called a Public File that users could consult to find other users' public keys. Kohnfelder proposed the concept of a certificate in his thesis [88] two years later. The certificate separates the signing and lookup functions by allowing a CA to bind a name to a key through a digital signature and to store the resulting certificate in a repository.

X.509

Some years after Kohnfelder's thesis, X.500, a global directory of named entities, was used for handling certificates. The X.509 certificate is named after the document in which it was originally specified: CCITT Recommendation X.509 [21]. This document specifies the authentication framework for the X.500 Directory. Nowadays, the focus is no longer on supporting the Directory, but more on developing a general purpose PKI. The commonly used version 3 of the X.509 certificate introduces a number of certificate extensions. Then the fourth edition of the X.509 standard [163] came out. It defines a few more extensions without modifying the certificate format. The Public-Key Infrastructure (X.509) (PKIX) working group in the Internet Engineering Task Force (IETF) profiled X.509 certificates for the Internet [71]. Besides authentication, the PKIX working group has proposed profiles of Attribute Certificates [45] and Proxy Certificates [155] to support authorization and privilege delegation.

PGP

Zimmermann first created *Pretty Good Privacy (PGP)* (e.g., [20, 53]) in 1991 to protect the messages and files used in bulletin systems. PGP rapidly acquired considerable attention around the world. Today, PGP becomes one of the most used email security systems. PGP uses both public-key cryptography and symmetric key cryptography. Public/private key pairs are used to distribute the newly generated session keys securely. The session key then is used to encrypt the plain text of a message. PGP users' public keys are available from many PGP key servers around the world (e.g. [115]).

Unlike the traditional PKI systems, the PGP system that links public keys to identities is known as the "web of trust". In PGP's scheme, there is no fully trusted authority that attests to the bindings between public keys and users. PGP leaves the decision of trusting a public key to the users. In OpenPGP [20, 128], a *trust signature* supports the creation of certificate authorities. These certificate authorities are trusted to sign other public keys. OpenPGP allows the owner of the key to make other keys certificate authorities.

SPKI/SDSI

Lampson and Rivest [143] proposed another form of public key infrastructure, called *Simple Distributed Security Infrastructure* (SDSI). An interesting feature is its decentralized name space. Under such a naming mechanism, the owner of each public key can create a local name space relative to that public key. The local name spaces can be linked together in a flexible and powerful manner to enable chains of authorization and to define groups of authorized principles [1]. At the same time, Carl Ellison et al. [41, 40] developed a *Simple Public Key Infrastructure* (SPKI). It emphasized simplicity of design and a flexible means of specifying authorizations. Both efforts were merged later, called SPKI/SDSI. It is motivated by addressing the complexity of X.509 PKIs and bringing power and flexibility to new PKIs.

As discussed by Adams and Just [2], despite different types of PKIs and their evolution over time, the essential definition and characteristics are very similar. The ways they are used to validate certificates share many common features.

8.1.3 Storage and retrieval services

One of the responsibilities of CAs is to publish certificates and their status information. A *repository* accepts certificates and status information from one or more CAs and makes them available to parties that need them to implement security services.

A repository is accessible by its network address and access protocol. It provides certificate information upon request. Requests could be based on the name of a user or CA or other information. Repositories are not trusted entities; the user accepts the certificates because the CA signed them. The source of the certificates does not affect its trustworthiness. Since the data itself establishes its integrity, a repository may be designed to maximize availability and performance.

One widely deployed directory standard for repository is X.500, along with languages for querying it such as the *Lightweight Directory Access Protocol (LDAP)* [158]. Such a repository is also called an *LDAP directory* or simply *Directory*. The information in the Directory is indexed by `(name, attribute)`. The *name* field specifies the name string of the entity, such as a user or a CA. The *attribute* field describes the type of the object related to this entity that the requester is interested in retrieving. For instance, `(A, userCertificate)` specifies the user certificate of user A; and `(B, cACertificate)` describes the certificates issued to CA B. For CAs and end entities, their *distinguished names* (DNs) are used as the search key for the name field. To help users locate the appropriate repository for retrieving certificates, the address of the repository should be indicated to users. We will examine details of a Directory later together with certification path discovery issues.

Originally, the repository was referred as the X.500 registry. In today's practice, it is anything that carries the certificate. For instance, the repository can be a file, a relational database, the Berkeley DB, the Windows registry. The certificate can even be included with the data it authenticates.

One of most important issues involved in storing and retrieving certificates is the *naming* mechanism. X.509 certificates use ASN.1 types to define certificate structure and certificate extensions. *Abstract Syntax Notation One (ASN.1)* is a formal language for describing messages exchanged among an extensive range of applications. The issuer and subject are specified using the DNs. The DN is a structured type that supports a hierarchical naming system. Directories typically use the subject DNs as the key to store corresponding certificates and other data structure. DNs are an ordered list of naming attributes. Each attribute is called a *Relative Distinguished Name (RDN)*. Essentially, a RDN has a type and a value. It is conventionally written as "type x = value y". As an example, Table 8.1 illustrates a few of the attribute types frequently used in RDNs. For instance, the subject DN of my Dartmouth certificate can be the following:

```
DC=edu,DC=dartmouth,C=US,O=Dartmouth College,CN=Meiyuan Zhao
```

Now we know how to locate an object in the repository. Another question comes out: how to locate the appropriate repository to search for objects. The subject DN might be able to give some clue. However, it does not necessarily help identify the Internet address or DNS name of the repository. X.509 certificate structure has two optional extensions—*Authority Information Access (AIA)* and *Subject Information Access (SIA)* that tell how to access CA/subject information and services. The extension contains a list of access descriptors, each of which is comprised of the access method field and the access location field. If the *uniform resource identifier (URI)* field is populated, then a URL allowing the information to be fetched with HTTP, FTP, or LDAP is provided.

String	X.500 Attribute Type
CN	commonName
L	localityName
O	organizationName
OU	orgnaizationUnitName
C	countryName
DC	domainComponent

Table 8.1: Frequently used attribute types for RDNs.

The naming issue of X.509 PKI is related to two famous problems: the "which Directory?" problem and the "which John Smith" problem. These problems have been identified, remain to be concerns, and cause complexities in deployment [2, 3, 42, 65, 82]. Some studies provide detailed discussions and possible solutions. We recommend [42, 65, 82] for further discussion.

8.1.4 Certificate Status Information (CSI) Services

If someone realizes his key has been stolen, or if someone gets fired from an organization, it is important to be able to revoke their certificate. Certificates typically have a relatively long period of validity, since it is a lot of trouble to issue a certificate. If anything happens before expiration such that people decide not to trust the certificate anymore, the CA that issued this certificate may revoke it and publish the certificate status information efficiently. CSI Services communicate the validity status of single certificates. A certificate is typically considered as "valid", "revoked", or "unknown". A user will consider a certificate to be valid if she is confident that the certificate is not revoked according to the evidence she has collected. She may conclude that the certificate is revoked if she learns and validates the information from authorities that the certificate is revoked. If the user cannot obtain enough information about the certificate status, she considers its status as unknown.

CRLs There are many revocation mechanisms. The most commonly used one is the *certificate revocation list (CRL)*. It is defined in the X.509 profile [71]. The basic idea of a CRL is that the CA periodically issues a signed list of all the revoked certificates and publishes it via some access method, such as HTTP, FTP, or LDAP. This list must be issued periodically, even if no certificates have been revoked since the last CRL. Each CRL must be properly signed and dated. If the *CRL Distribution Point (CDP)* extension in X.509 certificate is populated, users get to know where and how to obtain the CRLs needed to determine if the certificate is revoked.

CRLs have several problems. First, since CDP is an optional extension, it is not guaranteed to be populated. Users may not be able to locate the place to fetch the latest CRLs. In 2001, Verisign had the problem of effectively distributing a special CRL for three bogus certificates that Verisign falsely issued to entities in Microsoft [4]. Since the CDP was not populated in the certificates issued by Microsoft, users were not able fetch the latest CRL and learn the of the revocation. Microsoft finally had to release software patches to manually install the CRL on their client software. Second, users unnecessarily download large

CRLs to check against a few certificates. It wastes time, network bandwidth, and local storage space. If a large number of CRL downloads happen in a short period of time, it can even cause network congestion or create single-point-failure. Caching the CRLs may cause such a problem. Although it is true that caching of CRLs locally reduces the amount of network communication by CRL downloads, the local copies tend to expire at the same time. If these users send requests for a fresh CRL right after the expiry time, the directory is under stress. It is indicated that caching CRLs is one of the factors that contributes to the extremely long delays of the response from `crl.verisign.com` in January 2004 [66]. Third, certificate status information provided by CRLs is not so timely. Normally, if anything happened since the last CRL update, the user will definitely miss it. Even if the last CRL reflects the current situation, the users may not be able to know it. The main reason that causes this time gap is the mismatch of CRL validity time and update time. The common requirement of CRLs is that CRLs are updated and published every 6 or 12 hours. However, they are typically valid for one week. This means that once the user has a copy of a CRL that has not expired yet, he/she might not bother to download a newer copy. In the worst case, he/she misses revoked certificates in the last seven days.

Researchers have proposed several variants of CRLs, such as over-issued CRLs, segmented CRLs [71], delta CRLs [27], and windowed CRLs [105]. These proposals address the CRL size problem. Instead of a large list, they use smaller sublists. They use either time or certificate type to break the list into groups. Kocher proposed *Certificate Revocation Tree (CRT)* to decrease the amount of information downloaded by users [87]. CRTs organize information using Merkle's hash tree. They out-perform other CRL variants when the client population is large and the CRL is long.

Cooper proposed another variation that assigns different expiry dates to different CRL types [26]. His design breaks the synchronization of CRL's expiration and prevents single-point-failure caused by sudden burst of CRL downloading requests at the same time.

OCSP The *Online Certificate Status Protocol (OCSP)* [124] was designed to solve the timeliness problem of CRLs. In the OCSP, an *OCSP responder* answers a query about a certificate by returning a signed statement of that certificate's status at the current time. The OCSP responder can be the CA itself or a dedicated entity with its own certificate issued by the CA. With such a certificate, the dedicated OCSP responder is authorized to answer the queries about certificate status. OCSP responders typically rely on a real-time database of certificate status (which may not necessarily be the Directory managed by the CA). Some may depend on other types of databases, such as CRLs [97].

OCSP has strengths and drawbacks. Compared to CRLs, OCSP provides timely certificate status information. Furthermore, it transmits a much smaller amount of data for each query. However, it is problematic in several ways. First, the signed OCSP responses consume non-trivial bandwidth. Second, in large systems, expensive private-key operations are not able to keep up with millions of requests in a short time. Third, if only a single OCSP responder is serving, inevitably it becomes a performance bottleneck and is subject to denial-of-service attacks. Fourth, for services availability, OCSP responders have to remain online all the time, which makes them attractive to attackers. Lastly, if distributed OCSP responders are providing the service [97] to solve the performance bottleneck problem, we then face the information synchronization problem. Furthermore, since these OCSP responders have their own key pairs, the adversary may compromise any of the responders to compromise the entire system.

An interesting design related to CRLs, is called NOVOMODO [111]. It is perhaps the most efficient and scalable CSI mechanism to date. The design is based on one-way hash chains. When it issues the certificate, the CA also computes a "validity hash chain" and a "revocation target", Y_1, for the certificate. The validity hash chain contains a sequence of hash values indicating that the certificate will be valid in the future within its validity period, one for each day. The last value of the chain, called "validity target" and the "revocation target" are included in the certificate. Once a day, the corresponding hash value in the validity hash chain is published, so that users can verify that the certificate is still valid on the ith day. Once it is revoked, the CA publishes Y_0 instead. After verifying that $Y_1 = H(Y_0)$, users can conclude that the certificate is revoked.

Why bother with revocation anyway? As revocation causes so much trouble in terms of security and performance, Rivest et al. raised the question "can we eliminate revocation?" [144]. The argument is that if CAs issue short-lived certificates, they need not bother with revocation afterward. However, since certification itself is a lot of work, short-lived certificates dramatically increase workload on authorities. Such certificates might only be suitable for certain type of applications that accept lightweight, low assurance certificates. Furthermore, short-lived certificates increase the risk of key compromise. If an application cares about key compromise within a short amount of time, CAs still cannot avoid the work of revocation. To reduce the risk, we can shorten the validity of these certificates, which adds even more workload on certifications. It is a problem of trade-offs. Therefore, in some circumstances, revocation is still inevitable [106].

8.1.5 PKI Architecture

It is easy for Alice to accept and validate Bob's certificate when they both use the same CA. As the system scales, there can be hundreds of domains and authorities. We are faced with a new problem: how can Alice determine if a trustworthy CA issued Bob's certificate? There is a straightforward solution. Alice can maintain a list of CAs that she decides to trust (a "CA trust list"). While it may be reasonable for a small number of CAs, this solution is not scalable. However, if CAs establish trust among themselves, Alice then determines the trust of a CA by establishing a trust path via CAs that she trusts directly. This is potentially a scalable solution.

The term "PKI architecture" describes the organization of CAs and their trust relationships. In some studies, such architecture is referred as the *trust model* of PKI [82, 133].

For large domains, there is more than one CA. More complicated architectures are required. Essentially, there are two types: *hierarchical* PKI and *mesh* PKI.

Hierarchical PKI Traditional PKI architecture is the hierarchical PKI. CAs are related through superior-subordinate relationships. All the users trust the same *root CA*. CAs may issue certificates to their subordinate CAs or to end users. Any user certificate has a deterministic path from itself to the trusted root. The compromise of the root CA has severe impact on the entire enterprise PKI.

Mesh PKI Mesh PKI architecture is referred to as *web of trust*. This architecture is similar to the PGP model where there is no universally trusted authority. In this architecture, CAs are related in peer-to-peer relationships. Users may trust their own CA, but there is no common root CA to be trusted by all users. Compared with hierarchical PKI, mesh PKI is very resilient to CA compromise, except for the users certified by that compromised CA.

Cross-Enterprise PKIs

Many applications will cross the boundaries between two domains. PKIs must provide users with the tools they need to establish secure communications between users of different enterprise PKIs. We can combine the CA trust list, the hierarchical PKI, and the mesh PKI architectures to create a hybrid PKI.

Extended Trust List Similar to a CA trust list, the extended trust list contains multiple trust points, each of which identifies a PKI. However, the extended trust list architecture does not resolve the problems of trust list management and CA compromise.

Cross-Certified PKIs Driven by the needs of secure communications, PKI domains may wish to establish peer-to-peer trust relationships. Instead of making changes to the extended trust lists of users, CAs in different PKI domains use certificates to establish trust in each other's public keys. This relationship is typically bi-directional. One CA in a PKI domain issues a certificate to a CA in another PKI domain, and vice versa. These two certificates are referred to as a *cross-certificate* pair. The issuance of cross-certificate pairs is carefully controlled by the PKI policies of each involved PKI domain. Notice that the cross-certifying relationship is different from the peer-to-peer trust in mesh PKI domain, where the latter contains certificates issued by the CAs in the same domain. Furthermore, the certificates are not necessarily in a pair.

Now, if users trust the local CA, they can derive trust in the remote CA via the cross-certificate pair. In fact, users can use a chain of certificates to derive trust. This certificate chain is defined as a *certification path*. The user directly trusts the issuer of the first certificate. In the chain, the subject of the current certificate is the issuer of the successive certificate. And the subject of the last certificate is the target in which the user intends to establish trust. Certification paths in a cross-certified PKI environment may be quite complex. Cross-certified PKIs may have different architectures. Algorithms that discover the certification path for users should deal with the complexity. Furthermore, the method of arbitrary cross-certification does not scale well. If many PKIs wish to cross-certify with each other, many new certificates must be issued. For instance, cross-certifying n PKIs requires $(n^2 - n)/2$ relationships and $n^2 - 2$ certificates. Then a newly joined PKI invokes $2n$ more certificates to establish peer-to-peer relationships with these n PKIs.

Bridge CA A bridge CA addresses the drawbacks of the extended trust list and cross-certified PKIs. It acts like a trust arbitrator. The bridge CA establishes peer-to-peer relationships with different PKIs. Unlike root CAs, the bridge CA is not considered as a trust point by any user. Its sole role is to help PKIs establish trust relationships. Now, it is easy to add new CAs to a bridge-connected PKI. Only a pair of cross certificates is

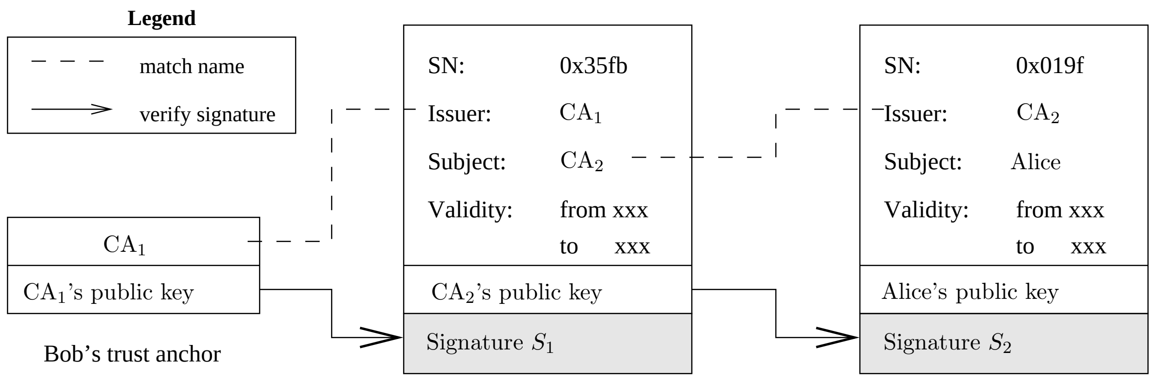


Figure 8.2: An example certification path that links CA$_1$ to Alice's end entity certificate
.

needed. However, a bridge CA does not solve the problem of certification path discovery. The algorithm is as complicated as in mesh PKI.

Overall, none of these architectures is perfect for all situations. People who deploy PKI should choose the best architecture that is suited to their enterprise situation. A single CA is appropriate for a small community. A hierarchical PKI is best suited to organizations with a well-defined structure. For organizations with no well-defined structure, a mesh PKI may not be a bad idea. For a small number of enterprise PKIs, cross-certification is an appropriate approach. Once the population increases, a bridge CA may be called for. The extra effort of setting up dedicated bridge CAs may pay off.

8.1.6 Validation services

In complex PKI environments, validating a certificate requires more than simply checking its validity status. In this section, we introduce issues related to certificate validation services in cross-domain environments, where multiple CAs are involved.

There are several entities involved in certificate validation. The *relying party* (RP) is the user who wishes to validate a certificate in question. The certificate to be validated is referred to as the *target*. The relying party maintains a list of public keys of CAs that he/she determines to trust directly. Typically, these trustworthy CAs have self-issued certificates. They are referred to as *trust anchors* (TAs). The process of certificate validation consists of two primary steps:

1. Discovering certification paths (certificate chains) that start from a TA of the relying party and end at the target, and

2. Validating that each certificate in the a certification path is valid and they chain together correctly.

These two steps are defined as *certification path discovery* and *certification path validation* respectively. Figure 8.2 illustrates an example certification path for Alice's certificate. Bob is the relying party in this example. CA$_1$ is one of his trust anchors. The certification path is a chain of certificates. For the certification path to be valid, it must satisfy the following conditions:

- The issuer and subject IDs should be correctly linked together.
- The signature of each certificate can be verified using the public key in the previous certificate or by the trust anchor.
- The intersection of validity periods of certificates is non-empty, and the current time falls into the resulting validity period.
- Every certificate is not revoked at the current time.
- Certificates satisfy any additional requirements specified by the system or by Bob.

The detailed algorithm for validating a certification path is described in the X.509 profile [71].

Currently, there is no standard for certification path discovery. However, there have been some efforts to improve the performance of certification path building algorithms. The PKIX working group published a draft [28] that analyzes the performance issues and complexity of the algorithms and provides some recommendations for efficient path discovery. A few other studies provided similar performance discussion of the algorithms, and provided different sets of recommendations [39, 96]. We are going to discuss the problems of certification path building algorithms in detail in Section 8.2.

Having established the certification path, relying parties can take advantage of several services to validate the path to validate the target. One of the choices is to use *Delegated Path Discovery (DPD)/Delegated Path Validation (DPV)* [134]. This specification contains the definition of messages exchanged by the servers and relying parties. The purpose of separating these two services is to satisfy relying parties' requirements separately. Some relying parties are only concerned about whether the certificate and the corresponding certification path are valid. They do not require additional information on the validation process itself. Other relying parties may need more than simple "yes/no" answers. A DPV server is for the first type of relying parties. It accepts a set of parameters by the relying party, constructs the certification path, and validates the path for the relying party. The DPD server, on the other hand, returns to the relying party the resulting certification path together with any other additional information, such as full CRLs, OCSP responses, etc. The requesters uses a set of rules, called the *path discovery policy*, to determine which information to return.

Another flavor of the certificate validation protocol is *Simple Certificate Validation Protocol (SCVP)* [101]. Similar to DPD/DPV, SCVP allows relying parties to offload the task of certificate validation to a server. An SCVP server can provide a wider range of information about the target certificate. It can provide validity information of a single certificate, as well as the entire certification path. The purpose of SCVP is to simplify client implementations and allow enterprises to centralize trust and policy management. We can consider SCVP as the protocol that combines the functionality of OCSP and DPV/DPD.

8.2 Certification Path Discovery

Certification path discovery plays a critical role in cross-domain PKI systems. Despite its paramount importance, very few studies have paid attention to it. As noted by Lloyd [96]:

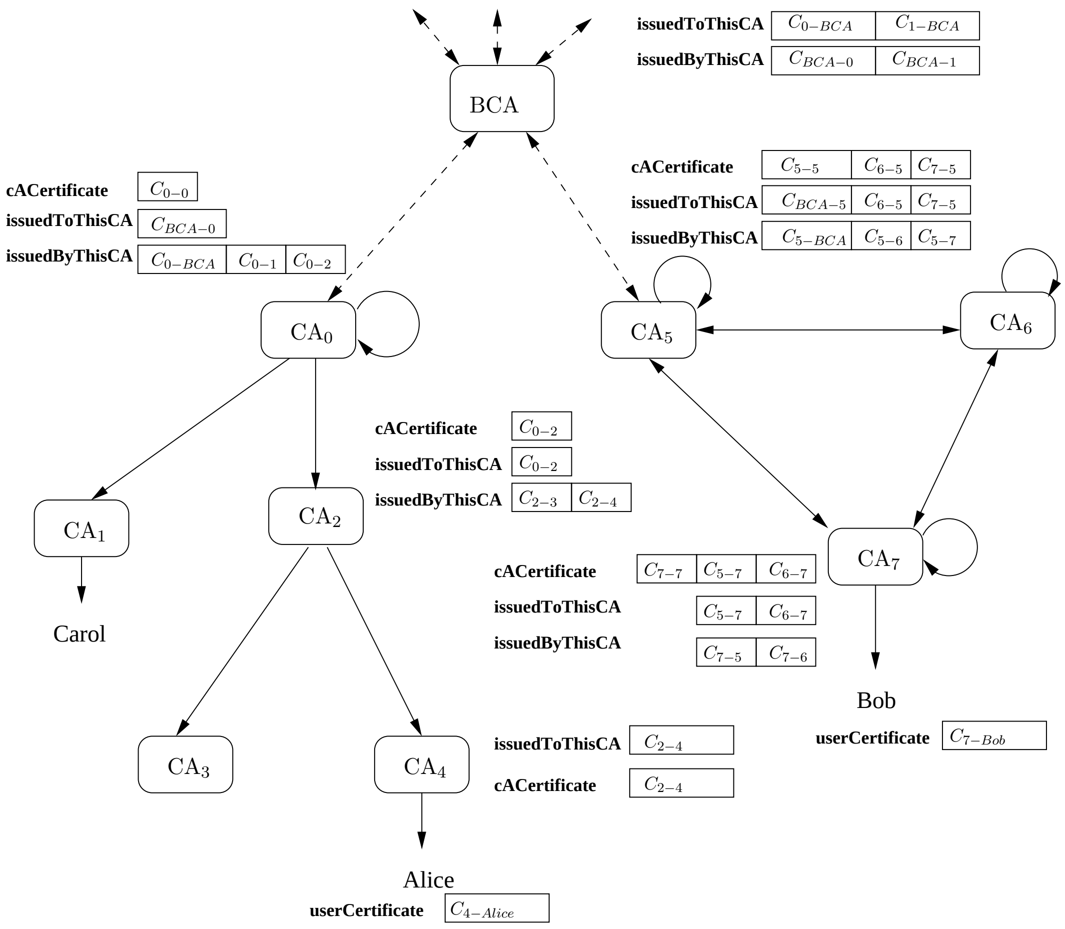

issuedToThisCA C_{0-BCA} C_{1-BCA}
issuedByThisCA C_{BCA-0} C_{BCA-1}
BCA
cACertificate C_{5-5} C_{6-5} C_{7-5}
issuedToThisCA C_{BCA-5} C_{6-5} C_{7-5}
issuedByThisCA C_{5-BCA} C_{5-6} C_{5-7}
cACertificate C_{0-0}
issuedToThisCA C_{BCA-0}
issuedByThisCA C_{0-BCA} C_{0-1} C_{0-2}
CA$_0$
CA$_5$
CA$_6$
cACertificate C_{0-2}
issuedToThisCA C_{0-2}
issuedByThisCA C_{2-3} C_{2-4}
CA$_1$
CA$_2$
CA$_7$
cACertificate C_{7-7} C_{5-7} C_{6-7}
issuedToThisCA C_{5-7} C_{6-7}
issuedByThisCA C_{7-5} C_{7-6}
Carol
Bob
userCertificate C_{7-Bob}
CA$_3$
CA$_4$
issuedToThisCA C_{2-4}
cACertificate C_{2-4}
Alice
userCertificate $C_{4-Alice}$

Figure 8.3: Alice-Bob example. Rectangles with round corners represent CAs, unidirectional arrows represent certificate issuance, and bi-directional arrows (including the dashed arrows between a CA and BCA) represent mutual cross-certification. C_{i-j} represents the certificate that CA$_i$ issues to CA$_j$. This figure also shows some auxiliary information (Directory attributes and certificate extensions) that might help Bob build the certification path to validate Alice's certificate $C_{4-Alice}$.

> *Certification path construction is a complex process and is subject to some degree of trial and error. The certification path construction process has not been standardized, and there is very little published information available to help implementers and product evaluators understand the complexities involved.*

In this thesis work, we set certification path building algorithms as one of the targets for performance evaluation. In this section, we discuss the problems of the current algorithm design and deployment.

Figure 8.3 sketches the example cross-domain PKI system we are going to use to explain certification path discovery problems. In this example, we have two enterprise PKIs. Alice belongs to the hierarchical PKI on the left. Bob belongs to the mesh PKI on the right. His trust anchor is CA$_7$. CA$_4$ issued Alice $C_{4-Alice}$ for signing purposes. Alice signed an email and sent it to Bob. To verify the signature, Bob should first validate $C_{4-Alice}$. The two enterprise PKIs are interconnected via a bridge CA, BCA. Carol is a potential adversary who may try to attack the transaction between Alice and Bob. Clearly, Bob's goal is to discover a valid certification path between CA$_7$ and Alice. Bob will encounter several problems. He has to decide how to handle the name strings so that the certificates are effectively and correctly linked together. He needs to determine appropriate approaches to identify and retrieve certificates for certification path building. Lastly, Bob faces the problem of constructing a valid certification path efficiently with all useful information in hand. Next, we discuss these three problems.

8.2.1 Name Chaining and Key Identifier Chaining

As the requirement of a certification path dictates, names in certificates must be chained together correctly. We have shown the concept in Figure 8.2. In most cases, name chaining is enough for a certification path. In the situation that entities commonly have multiple key pairs and multiple public key certificates for different purposes, we need to be extra careful to choose the correct certificate for path building purposes. Naming itself is not enough for an effective chaining. Indeed what should be chained together are public keys. The *Authority Key Identifier (AKID)* and *Subject Key Identifier (SKID)* are X.509 certificate extensions that can be used to facilitate the chaining process. AKIDs are used to distinguish one public key from another when a given CA has multiple signing keys. SKIDs provide a way to identify certificates that contain a specific public key.

Now besides name chaining, in order to have certificates chained together correctly, we need to have the SKID of the previous certificate match the AKID of the current certificate. SKID and AKID can be represented using a **keyIdentifier**. According to RFC3280, a CA conforming to the Internet and CRL profile must populate the AKID of all certificates that it issues with the same **keyIdentifier** that is populated in the SKID of the certificate used to verify the digital signature on those issued certificates.

There could be multiple ways of calculating the **keyIdentifier**. For instance, we can use the 20-byte SHA-1 value of the public key or a monotonically increasing sequence of integers. RFC3280 does not mandate a particular calculation method.

8.2.2 Identifying Appropriate Certificates

One of the biggest problem with the certification path discovery process is locating the appropriate certificates.

In our example, we assume CAs use LDAP servers that support LDAP access to maintain Directories for certificates and CRLs. It turns out that some X.509 certificate attributes and Directory attributes can help. The latest edition of X.509 [163] mandates certain requirements on the syntax and expected use of these attributes.

Directory Attributes

There are many attributes defined in [163]. The most relevant ones for certification path building are **cACertificate**, **crossCertificatePair** and **userCertificate**.

cACertificate "The **cACertificate** attribute of a CA's directory entry shall be used to store self-issued certificates (if any) and certificates issued to this CA by the CAs in the same realm as this CA [163]." For instance, the directory entry of CA$_5$ should have C_{5-5}, C_{6-5}, and C_{7-5} in this attribute; but C_{BCA-5} should not be there.

crossCertificatePair This attribute contains two elements: **issuedToThisCA** and **issuedByThisCA**.

> *The **issuedToThisCA** element of the **crossCertificatePair** attribute of a CA's directory entry shall be used to store all, except self-issued certificates issued to this CA. Optionally, the **issuedByThisCA** elements of the **crossCertificatePair** attribute of a CA's directory entry may contain a subset of certificates issued by this CA to other CAs. If a CA issues a certificate to another CA, and the subject CA is not subordinate to the issuer CA in a hierarchy, then the issuer CA shall place that certificate in the **issuedByThisCA** element of **crossCertificatePair** attribute of its own directory entry [163].*

Immediately, we notice an optional behavior by CAs that we should not rely on: **issuedByThisCA** is not necessarily fully populated. Although, all certificates issued by a CA to a non-subordinate or peer CA must be stored in the **issuedByThisCA** element of the cross-certificate pair attribute of the issuing CA's directory entry, it is not reasonable to assume that it is populated in a hierarchical PKI.

In Figure 8.3, CAs have all directory attributes fully populated. Thus, the **issuedByThisCA** element of the cross-certificate pair attribute of CA$_0$'s directory entry contains C_{0-BCA}, C_{0-1}, and C_{0-2}. In practice, the last two certificates may not be there. This optional behavior creates problems for certification path discovery. Suppose Bob has constructed a partial path CA$_5$ $\rightarrow$ CA$_0$. With an only partially populated **issuedByThisCA** element, Bob cannot go anywhere further.

Useful Certificate Extensions

Directory entry attributes does not solve the problem of locating appropriate certificates completely. Certificates are distributed over several Directories. Each CA may maintain its own Directory that contains the entries for itself as well as for all its users. Alternatively, several CAs in the same enterprise PKI may share a single Directory that populates entries for all CAs and their users. In practice, the latter approach is more common for hierarchical PKIs. It eases the management task for the CA administrators. Nonetheless, we still need a method to help users search for appropriate certificates by jumping from one Directory to another.

The X.509 Certificate and CRL profile defines a private certificate extension: *Authority Information Access (AIA)*. AIA indicates how to access CA information and services for the issuer of the certificate in which the extension appears [71]. One possible usage of AIA is to point to a list of CAs that have issued certificates to the issuer of this certificate. Or, we may see the URI of the LDAP server that users can access to fetch the directory entry of the issuer. These example usages of AIA are very helpful to set up the linkage between Directories that the user needs to hop for constructing a certification path. For example, the newly setup SAFE Bridge Certification Authority TEST Environment makes full use of AIA in the prototype certificates [146]. Each test certificate includes two types of information for the issuer CA. It has the URI for downloading the issuer's certificate in PKCS#7 format. There is also the URI of the issuer CA's Directory LDAP address. However, there is not mandatory behavior on how to use AIA. According to the data we have collected, not all enterprise PKIs use the AIA extension in certificates.

Besides AIA, the X.509 Certificate and CRL profile also defines *Subject Information Access (SIA)* extension that indicates how to access information and services for the subject of the certificate. The SIA extension is new. We haven't seen actual usage of it. CA administrators may start to exploit it in the future. One possible usage is to indicate the URI of the Directory for the subject CA's information. Again, it can be very useful to help users continue hopping between directories.

Given the current practical situation, we can let users fully rely on AIA and SIA to find out the next directory to be explored during certification path discovery. The best possible practical solution that we can see to solve this problem is to configure the algorithm implementation properly so that it has prior knowledge of where to locate proper directories. It is not the best solution for the cross-enterprise PKI environment, since there could be dynamic changes of directories in the system. We stress that there is a need to fully exploit AIA and SIA extensions.

8.2.3 Building Certification Paths

Now, we have everything set up. It is time for Bob to build a certification path to Alice's certificate. Immediately, Bob is faced with an important question: Where to start? In other words, he has to decide whether to start the building process from Alice's certificate or from his trust anchors.

The *forward* direction is from the target to a trust anchor. The opposite direction is referred to as the *reverse* direction. The choice of building direction is affected by the PKI architecture. In a strictly hierarchical PKI, the forward direction is probably faster. In our example, we assume that Bob trusts CA_0 directly. He then starts building from Alice's certificate. He can quickly reach CA_0 by retrieving certificates using the **issuedToThisCA** element in cross-certificate pairs, since each time there is only one certificate to choose. This nice feature has been intensively discussed by Elley et al. [39] and Lloyd [96]. On the other hand, building in the reverse direction in a hierarchy is not as efficient. Suppose Bob has reached CA_0 from his trust anchor CA_7 so far. He has to make a choice in the next step. If Bob decides to go with CA_1, he will not be able to find Alice's certificate. Hence, Bob experiences a failure. In general, when faced with multiple choices, what a relying party can do is nothing more than building trails one by one.

One may suggest building certification paths using the forward direction even when in fully distributed architectures. It might make sense to do that if there are only a few choices at each step. In our example, there could be tens or even hundreds of cross-certificate pairs at the bridge CA, BCA. Merely relying on **issuedToThisCA** elements will result in many building failures and rollbacks. In this situation, building in the reverse direction makes more sense. While there are still some errors and trails when building in this way, with the help of other optimizations, we may end up with fewer branches at each step [39].

In fact, if the building algorithm can intelligently foresee the potential architecture it is going to encounter, it can work more intelligently by switching building directions during path building. In our example, Bob can start with the forward direction. Once he arrives at CA_0, he can switch to the reverse direction to build the other half of the certification path. This way, he can significantly reduce the number of errors and trails at BCA.

Optimizations

Elley et al. gave detailed discussions on how optimizations, such as name constraints and policy processing, can be used to improve building efficiency considerably [39]. Here, we discuss these two together with other optimizations.

Name Constraints This is a certificate extension used by CAs to limit the name space within which all subject names in subsequent certificates in the certification path should be located [71]. In validating a certification path, name constraints are processed in the reverse direction. If the algorithm builds certification path in the reverse direction too, then the branch that does not satisfy the name constraints can be filtered out immediately. On the other hand, if building in the forward direction, although some of the branches are suspicious with unmatched names at a given point, the algorithm cannot eliminate them. In general, building in the reverse direction is more effective than building forward with regard to processing name constraints [39]. We realize that the conclusion is drawn from the fact that name constraints are designed to be effective to only the *subsequent* certificate in the certification path reverse direction.

Policy Processing For an end entity certificate, the certificate policy extension is used to limit for which purpose the certificate will be used. For a CA certificate, the policy extension is used by the issuer to limit which policies are considered acceptable in a certification path. The analysis is similar to the previous discussion on name constraints. Since the policy extension is designed to validate a certification path in the reverse direction, it is more effective in helping build certification paths in the reverse direction as well.

Priority on Branches Another interesting approach to reduce the errors and trials is to find the ordering of branches in order to maximize the chance that the candidate path will turn out to be an acceptable path in the end. The *Certificate Path Library (CPL)* [31] used by the *Certificate Arbitrator Module (CAM)* [152] to prioritize the branches uses a list of criteria. The suggested rules are the following:

1. Certificates from the **cACertificate** directory attribute have priority over certificates from the **crossCertificatePair** attribute. In other words, the algorithm tends to stay within the enterprise PKI as much as possible before jumping out to other PKI domains.

2. Certificates in which the issuer's algorithm *object identifier (OID)* equals to subject's algorithm OID should have priority. In other words, the algorithm tries to handle as few signature algorithms as possible to reduce processing complexity.

3. Certificates with a non-empty policies extension have priority over certificates with an empty policies extension. In other words, certificates with additional policy constraints are preferred.

4. Certificates with fewer RDN elements in the Issuer DN should have priority. The algorithm favors shorter issuer distinguished names.

5. Certificates that match more RDNs between the issuer DN and the relying party's trust anchor DN should have priority. The algorithm expects that such certificates should be closer to the trust anchor in the certificate chain. We expect the algorithm to use this option when building the path in the forward direction.

6. 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 certification in the local PKI domain have similar distinguished names. The algorithm prefers to stay in the local PKI domain as much as possible.

7. Certificates with longer validity periods should have priority. The resulting certification path may have longer overall validity with such certificates.

Other vendors may have adopted variants of this list for their needs. We cannot judge a priori which criteria work better in a real deployed PKI environment. This is one of the most important goals in our modeling research—to produce more concrete suggestions on exploiting certain optimizations using quantitative analysis rather than qualitative discussions only.

Eliminate Branches Besides above optimizations, there are other tricks that the relying party can use to filter out branches immediately. For instance, the algorithm can choose to verify the signature on a certificate during path building (instead of afterwards). Certificates that failed the signature verification may be eliminated. However, there are two problems with this approach. First, the algorithm spends additional computation in path building. Second, and most importantly, the algorithm may not be able to filter out the branch completely. The algorithm may not be able to verify the signature either because the signature is broken or because the public key is not correct. The algorithm cannot distinguish these two cases if it does not trust the public key first. Therefore, elimination only makes sense for building in the reverse direction.

Another way of elimination is to rely on CSI provided either by CRLs or by OCSP. If a certificate is revoked already, the entire certification path should be invalid. At first glance, it makes sense to find it out as early as possible. The algorithm can then roll back immediately to try out a new path, instead of wasting time on constructing the rest of the current path. However, checking CSI introduces a lot of performance overhead. It is unclear if it really helps speed up the certificate validation process. What relying parties really care about is the overall time latencies for validating a single certificate. It is a trade-off. Either it takes a longer time to discover a certification path for the path validation algorithm, or the path discovery algorithm quickly returns a candidate certification path and leaves the caller with the possibility that the path can turn out to be invalid later on because of a revoked certificate in the path.

Furthermore, checking CSI for certification path discovery introduces additional complexity. As discussed in [28], to be able to trust the CSI returned from either CRLs or OCSP responses, we have to establish trust in signers of this information. This requirement indicates that we only eliminate branches by revocation information when we build in the reverse direction. In this case, we have already validated the partial path from the trust anchor up to the CA that signed the CRL. For OCSP responses, the situation is even more complicated. In practice, some OCSP responders are separated from their owner CAs. They have their own certificates. Now, the algorithm should construct a valid certification path for the OCSP responder first in order to continue constructing the actual certification path for the target. To solve one problem, we have created two equally complicated problems. The PKIF toolkit newly developed by Orion Security Solutions [135] is an implementation dedicated to solve the certification path discovery problem for OCSP responders.

After all, because there are complicated issues involved, signature verification and checking certificate revocation are not recommended to be used for optimizing the certification path discovery process [28].

8.3 Related Work

PKI technology has been around for many years. During its development there have been many arguments on all aspects, such as its practicability, available, scalability, robustness, interoperability, etc. As X.509 becomes the major form of PKI, continual arguments focus on whether we should stay with it or switch to alternatives that may be potentially more elegant. Despite these arguments, researchers continue to work on improving functionality and performance of individual PKI components. There is one component that attracts the most attention—CSI systems. Researchers have proposed many alternative approaches to traditional CRLs or OCSP to improve performance, availability, and robustness. Besides qualitative analysis, there are also a few efforts on quantitative approaches to study performance and trust management issues. This includes analytical approaches, algebraic approach, and simulation. We will review the related work on implementing certification path discovery algorithm and studies on its optimization issues.

8.3.1 Current Deployment

In real-world practice, X.509 PKI systems have been deployed. The systems have evolved from simple single CA PKIs, to complex enterprise PKIs, to cross-certified PKIs, and now to bridge technology. To date, there have been several bridge CAs in operation or prototype to improve interoperability between enterprise PKIs.

The *Federal Bridge Certification Authority (FBCA)* [46] is one of the first bridge CAs in operation. The FBCA supports interoperability among Federal Agency PKI domains in a peer-to-peer fashion. The FBCA has issued cross-certificate pairs with more than eight federal agencies PKIs, including the Department of Defense, State Department, US Treasury, NASA, USDA, Department of Energy, the Federal Access Certificates for Electronic Services (ACES), and the State of Illinois. FBCA will continue to cross-certify other federal agency, such as the principal CA for Department of Homeland Security, and other bridge certification authorities.

One of the bridge certificate authorities with which the FBCA will cross-certify is HEBCA—*Higher Education Bridge Certification Authority* [69] operated at Dartmouth College. HEBCA facilitates electronic communications within and between institutions of higher education as well with federal and state governments. HEBCA is under development. The current prototype has issued cross-certificates with PKIs of several institutions, including Dartmouth College, University of Wisconsin, University of Texas, University of Virginia, Duke University, and University of California Office of the President.

Another just-initiated CA is called USHER [156]. It will cross-certify with HEBCA. The purpose of USHER is to allow institutions that do not have their own enterprise PKI system to be able to communicate with other institution PKIs securely.

CertiPath [22] is the first commercially managed PKI bridge. It connects to enterprise PKIs of several aerospace companies, including Boeing, Rolls Royce, Northrop Grumman, Lockheed Martin, and Airbus/EADS. Moreover, the CertiPath PKI bridge will cross-certify with FBCA.

SAFE [146] is going to be the bridge certification authority the sets up secure communication between PKIs in BioPharma Association and the other enterprise and government PKIs via FBCA. Recently, SAFE launched a test environment that contains a testing BCA and two sample enterprise PKIs. In the future, the SAFE bridge will cross-certify with Johnson & Johnson and other enterprise PKIs in the pharmaceutical industry.

We are also aware of other efforts all over the world that try to improve PKI interoperability. For instance, *Trans-European Research and Education Networking Association (TERENA)* have set up an academic CA repository, TACAR, that lists all participating PKIs root CA certificates [153].

8.3.2 Debates

Several years ago, people had doubt regarding whether PKI technology will ever blossom. The doubt came from the PKI development situation at the time. While people were cheering about PKI technology, there had not been a real "infrastructure" in use at the time.

Ellison and Schneier's paper [42] enumerated ten risks of PKI that people were not being told about. Their argument is that security is a chain; it's only as strong as the weakest link. The security of any CA-based system is based on many links and they are not all cryptographic. People are involved in many processes. It is very hard to maintain the security of PKI; it is full of risks.

On the other hand, Gutmann argued that although there were some false starts, X.509 has slowly evolved into a flexible public key infrastructure model [65]. However, like other flexible objects, PKI sacrifices some utility in trying to be all things to all people. The use of innovative public key infrastructure models can make the technology meet today's requirements. Gutmann analyzed several popular PKI problems—the identity problem, the revocation problem, and the certificate chain problem—and provided PKI design recommendations. For identity, he suggested to choose a locally meaningful identifier instead of a globally distinguished name. Gutmann considered revocation to be problematic. So, he suggested to avoid it if possible. If people cannot avoid revocation, his advice is to try use an online status query mechanism. He also preferred PKIs specifically designed to address a particular problem. This kind of PKI system is much easier to work with. We realize that in some situations, general purpose PKI is necessary, such as in large government PKIs. We suggest that the design should be focused on several popular applications first and we need to make it as flexible as possible.

Adams and Just examined the evolution of PKI in the past ten years and showed how the understanding of PKI technology has matured over the years [2]. The essential characteristics have remained unchanged, while the definition of PKI became broader and more flexible. And PKI is developing from an evaluation and comparison of several quite different forms of PKI as well as a consideration of PKI criticisms over the years.

8.3.3 Studies of CSI Systems

One of the goals of this thesis research is to understand PKI performance. Previous PKI performance studies are mainly focused on performance of CSI Systems. People realize that revocation is a hard problem [2, 6, 42, 64, 65, 77, 78, 95, 119, 121, 125, 144]. Researchers are trying to find a balance between design complexity, security, and performance overhead.

Research on CRLs mainly focuses on reducing the data size [27, 105, 106], changing the data structure [87, 110, 111, 120], and modifying the CRL publishing schemes [26, 54]. There are also a few efforts on improving online certificate status systems. CoreStreet proposed distributed OCSP to relieve the burden on a single OCSP responder [97]. Boneh et al. proposed the concept of an online *semi-trusted mediator (SEM)* to allow fast revocation [13, 14].

There have been two types of approaches used by researchers to understand CSI performance. In the context of our research, we are interested in finding the most effective methodology to conduct an evaluation study on PKI system in general and certificate path discovery in particular. We will spend some time next to examine these representative approaches.

8.3.4 Analytical Studies

Cooper presented a model for the distribution of revocation information using CRLs [26]. The model was used to highlight inefficiencies in the traditional method of distributing CSI using CRLs. Cooper used an exponential inter-arrival probability density to model the certificate validation attempts by relying parties. This model revealed that if CRLs cached by every relying party expire at the same time, there will be huge a increase of request rate followed by an exponential decline in the request rate. Consequently, Cooper proposed over-issued CRLs and segmented CRLs that significantly reduce the request rate by relying parties. Although his model is extremely simple, it does provide a useful approach to analyze certificate usage behavior quantitatively.

An analytical method does have limitations. Cooper's model only works with over-simplified situations. For instance, the high expected request rate does not mean it will happen in practice. In fact, most of relying parties do not bother with expired CRLs in the local cache until they really need them to validate certificates. The actual request rate does not only rely on CRL expiry periods, but also on the usage of certificates. Moreover, the size of CRLs varies, which may also affect the final impact of CRL downloading on the network. Besides, it is too artificial to assume that all requests arrive at the same time.

Iliadis et al. presented a mechanism-neutral framework for the evaluation of CSI mechanisms [77, 78]. Instead of focusing on only one criterion for the comparison, the authors proposed a complete evaluation framework that consists of management, performance, and security criteria. They used the framework to demonstrate evaluation of CRLs. This general purpose framework can be used to evaluate many different types of CSI systems. Unfortunately, it does not provide quantitative analysis.

8.3.5 Simulation and Testbed Studies

Årnes implemented a simulation to evaluate certificate revocation performance [7]. His simulation model contains a set of simulation input and output variables, and the models used to these compute intermediate variables. Using this simulation model, Årnes et al. conducted their research on evaluating CRLs, delta-CRLs, and OCSP [6]. This simulation model is much more powerful than Cooper's model. It has a rich set of configurations, parameters, and constants. It is able to evaluate more than just CRLs. The limitation is that the simulation models are strictly controlled by formulas. The network environment and user activities are not included.

Muñoz et al. implemented a testbed for certificate validation—CERVANTES [121]. It is a Java platform that allows researchers to develop and test their own "real" revocation systems. It includes the authors' own implementation of a the main standards (CRLs and OCSP). 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. CERVANTES is able to analyze temporal behaviors of CSI systems. 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 the experiments.

8.3.6 Certification Path Discovery

In contrast to the heated discussions on CSI systems, there are very few studies on evaluating algorithm for certification path discovery. Elley et al. [39] 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 [96] that discussed options for effective and efficient certification path construction algorithm. He came to a similar conclusion about building direction. 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, and building in both directions and meeting in the middle is a good approach.

The simulation study by Mukkamamla et al. [117] quantitatively examines the performance issues for certification path processing protocols and path construction algorithms. The authors studied the performance specifically for a protocol they proposed—*efficient certificate path validation mechanism* (ECPV). In their performance analysis, they identified several key factors that affect the performance. These factors include validity period, revocation of CA-CA certificates, rate of validation requests, maximum path length, and trust models. In their discussion, the number of CAs accessed by the mechanism is a function of the maximum path length, the average number of cross-CA certificates, and the average number of trust anchors specified by a relying party. The simulation results showed that the number of visits to repositories is sensitive to values of maximum path length and number of trust anchors. This experimental study is a good effort that tries to examine potential trade-offs in algorithm options. However, the study is unconvincing in that they constructed the simulation model based on many unrealistic assumptions and the evaluation is only based on a few algorithm options and measurement metrics. Moreover, it would be more convincing if they compared their ECPV protocol with other standard protocols, such as DPD, DPV [134], and SCVP [101].

Chapter 9

Simulation Framework for PKI Systems

In this thesis research, we designed and implemented simulation models for networked general-purpose PKI systems, which we believe are the first open source simulation models for large-scale cross-enterprise PKI systems.

There were several challenges that we encountered in modeling large-scale general PKI systems. First, the scale of the target PKI systems makes us examine the model carefully, so that we are able to model large number of entities with reasonable detail. Second, unlike the BGP routing system, PKI systems contain a lot of human activities. It is challenging to combine the protocol specification and human activities. Even with the protocols, not all details are specified clearly in the standard. It is our job to make reasonable assumptions and create practical models. Third, we want our framework to be an effective foundation for future analysis of this important security area. Therefore, it is very important to design a flexible interface for the model so that additional modules can be easily added and configured.

We have fulfilled this challenging task. We designed and implemented a detailed ***operational model*** that specifies certificates, entities, and PKI activities in detail. The operational model provides a flexible interface that allows us to add additional modules to further model certificate related protocols, such as protocols for certificate revocation and validation.

In the simulation, we provide modeling functionality at multiple levels of modeling details. At a higher level, we designed a ***certification path discovery model*** to demonstrate how we use the operational model to study its performance. We use SSFNet to model major components in a general purpose X.509 PKI system, including

- issuing certificates,
- storing and retrieving certificates,
- revoking certificates, and
- validating certificates.

In this chapter, we introduce our methodology for the operational model in detail, including the model design, configuration, and measurement metrics. Chapter 10 presents the conceptual model of certification path discovery algorithm and performance study.

9.1 Choice of Simulation

Large-scale PKI systems share many common features with the BGP system that makes us choose to use simulation for our performance study.

Although the current scale of deployed PKI systems hasn't reached the size of the Internet, it is the direction of development. More and more security applications call for a PKI system at the global scale. Gaining concrete understanding of performance of a large-scale PKI system before it is actually realized would be valuable to help designers and developers to make prudent choices. Given the target scale of PKI systems in our study, a testbed, such as CERVANTES [121] is not appropriate.

Similar to the BGP routing system, a PKI system consists of many distributed entities running protocols and making decisions independently. As a result, the features of such complicated distributed behaviors are very hard for a simple theoretical model to capture. As we have discussed in Section 8.3.4, Cooper designed a very simple analytical model for CRLs. This model is limited in that it ignores other important factors that may affect the performance of CRLs. For instance, it does not take into account the data size. A white paper by CoreStreet Ltd. in 2004 [98] reported that the CRL approach by the Department of Defense takes a considerable amount of time to download the large (5–6 MB) CRL files. Such large CRLs create the pressure on network links and cause single-point-failures or denial-of-service from just ordinary use. In addition, the downloading latency is also affected by the replying parties' physical locations in the network.

Simulation needs some statistical models to configure network activities and related parameters. For instance, some network protocols in practice produce traffic patterns that can be characterized by statistical models using numerical analysis. It has been revealed that the HTTP request pattern is similar to a traditional queuing system. The inter-arrival time of HTTP requests follows an exponential probability density [47]. In our modeling work, we also use some analytical models for PKI systems to capture important factors for performance.

Some researchers designed calculus models for PKI systems. For instance, Maurer proposed an approach to modeling and reason about a PKI from a user Alice's point of view [103, 104]. His model consists of a set of calculi to reasoning about the authenticity of a certificate for Alice. Such models focus on the security features of a PKI system. The properties are expressed using logic or algebra. It is very difficult to use such techniques to express performance characteristics, such as time latency and bandwidth utilization.

In a word, our best choice for evaluating PKI systems is to use simulation. We also take into account useful features from other alternative approaches, such as the statistical models of network activities and traffic patterns. They are used as parameters for our simulation model configurations.

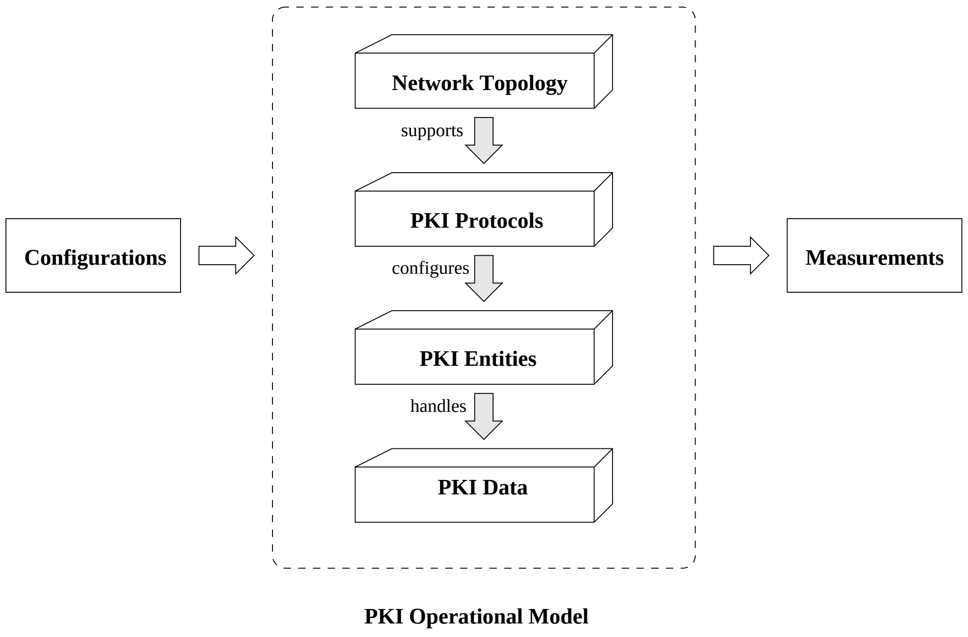


Figure 9.1: Design of operational model in simulation.

9.2 The Operational Model

The operational model simulates basic PKI components and their activities in the network environment. As we have discussed in Chapter 8, there are five primary components: certificates, storage and retrieval services, CSI services, PKI architectures, and certificate validation services. In the simulation, each of these components is implemented as an independent module with a flexible interface. The simulation implements basic functionalities. Additional features are added to the model as additional modules.

Figure 9.1 demonstrates the design concept. The model contains four major modules. The basic module is the *PKI Data* module that specifies data forms used in a PKI system. In this module, we implement certificates and CRLs. Users can easily add more data forms using the interface provided by the PKI data module. Then, the *PKI Entities* module manipulates PKI data. The module has built-in support for some basic PKI entities—relying parties, CAs, directories. Depending on the protocol models in the simulation, more types of entities may be required. 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. In our implementation, we have implemented basic functions of all these four types of protocols. Finally, all PKI protocols are operating with the help of the *Network Topology* module.

Overall, the entire operational model can be treated as a single box. Each module in the model provides a set of configurations that allow users to specify the behaviors and parameters. Meanwhile, the PKI protocol module also produces measurement data as the simulation runs. It can be related to any metrics that users are interested in measuring.

Field	**Type**	**Example**	**Primary?**
Serial Number	integer	1234	yes
Issuer	String	"DoD Root"	yes
Subject	String	"DoD CA3"	yes
Valid from	date	2000-01-01_00:00:00 GMT	yes
Valid to	date	2010-01-01_00:00:00 GMT	yes
Policy	String	"high"	no
Name constraint	String	"dartmouth.edu"	no
Revoked	boolean	true/false	–
Size	integer	1024 bytes	–

Table 9.1: Modeled fields and properties for certificates.

The simulation experiments are as easy as configuring the model, running the simulation, and processing the measurement data from simulation runs.

Next, we present each module in detail.

9.2.1 PKI Data

Certificates

The public key certificate is absolutely the basic data form in any PKI system. We model a subset of all fields defined for X.509 public key certificates. For the model, we selected only those fields that had a significant impact on performance. Table 9.1 summarizes the fields we selected as primary elements in a modeled certificate. These fields uniquely identify a simulated certificate. Of course, our judgment of whether a field should be present in the model can be changed in the future if the requirement of a specific performance evaluation changes. Other fields can be easily added into a certificate. We use a tuple "(type, value)" to define a field, which is a similar style used for defining real fields for X.509 certificates.

Besides fields, certificates also have additional properties. Two of them are important. We defined a boolean field to specify if a certificate is revoked. It helps the model to quickly print out the certificate status for the purpose of measurement. The entities in the model still need to go through the standard methods to verify certificate status, such as fetching CRLs or sending a request to an OCSP responder. The other property is the *size* of the certificate. Since we discard many fields for a certificate in the model, we need a property to summarize all other details we ignore. Data length is the property related to performance. The model specifies the length of certificate fields. In the extreme case, where we do not care about the contents in a certificate at all, we can still model it using a certificate length parameter. SSFNet provides the facility to transmit data of a certain size even if there is no actual contents in the data. We call it "virtual data size". The simulation is able to charge network latency correctly when transmitting a message with specified virtual size.

For issuer/subject, we use the corresponding entity's name string. In most cases, it is the network address of the entity. If the entity does not exist in the model, we use a sequence number as the name string. The model contains a facility to keep the largest sequence number assigned to an entity so far. We also assume that (by default) certificates are not revoked.

Field	**Type**	**Example**	**Primary?**
CRL number	integer	12	yes
Issuer	String	"DoD Root"	yes
This Update	date	2000-01-01_00:00:00 GMT	yes
Next Update	data	2000-01-01_06:00:00 GMT	yes
Revoked Certificates	list	134, 958	no
Number of Revoke Certificates	integer	100	yes

Table 9.2: Fields and properties for CRLs.

The validity fields and certificate size are configurable. The "valid from" field is picked randomly from a date range. Users can assume any date range. Similarly, the length of certificate validity is also configurable. In our experiments, we either use real data to configure the validity, or we assume that certificates are issued between 2000-01-01_00:00:00 GMT and 2003-01-01_00:00:00 GMT and they are valid for 5 years. It turns out that most of our experiments last for only a few hours. Thus these settings are good enough. Similarly, we assume certificates have random size. In our simulation experiments, we choose 600–1500 bytes by default. We obtained this range by examining real certificates from our data collection.

CRLs

We also implemented CRLs. Table 9.2 describes the fields and properties of a CRL in our model. The choice and assignment of these fields are similar to the design of certificates. The "This Update" and "Next Update" fields are configurable. The values depend on the revocation services that control the CRLs. We configure these values so that the CRLs are valid for 7 days. This validity length is commonly adopted in many real PKI domains, such as the Department of Defense PKI [33]. We are also interested in the size of CRLs. Intuitively, we know that CRLs are defined using several fixed fields, such as the header and the list of revoked certificates and corresponding revocation reason code. The number of revoked certificates is enough to decide the resulting CRL data size. We examined the CRLs published by CAs from the Department of Defense PKI [33] and the Johnson & Johnson PKI [81]. The data suggest that the file size of CRLs is linear to the number of revoked certificates. Figure 9.2 shows that the fitting line is good enough for modeling.

9.2.2 Entities

We model three types of primary entities—relying parties, CAs, and directories. *BasicPlayer* is the interface we designed for all types of entities. Any new type of entity can be implemented using the BasicPlayer interface. Figure 9.3 demonstrates the concept.

Relying Party (RP) This is the player in the model that 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 data retrieved so far. In the configuration, relying parties send requests

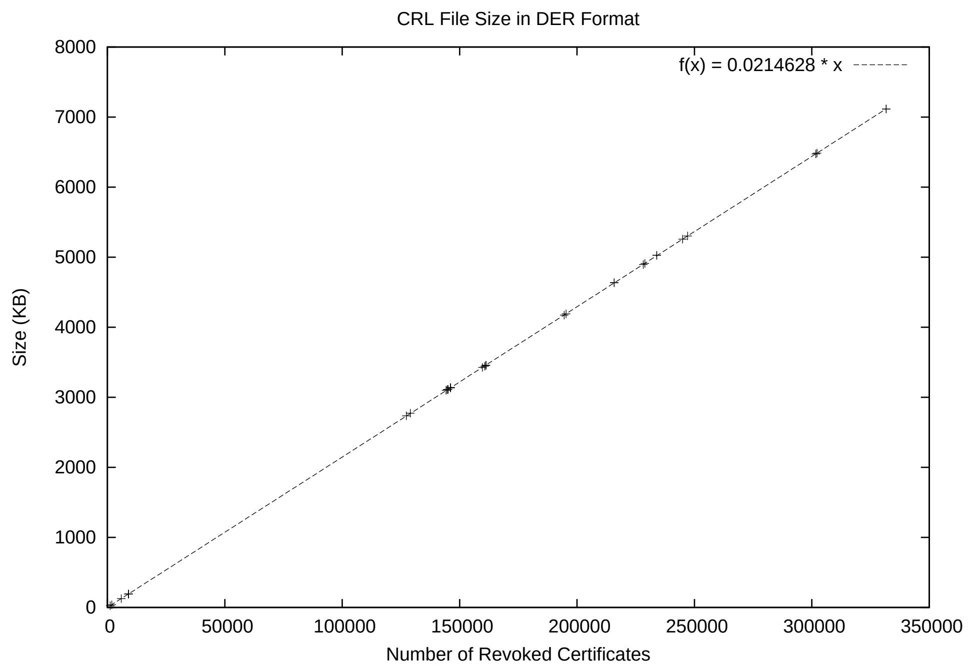


Figure 9.2: CRL file size.

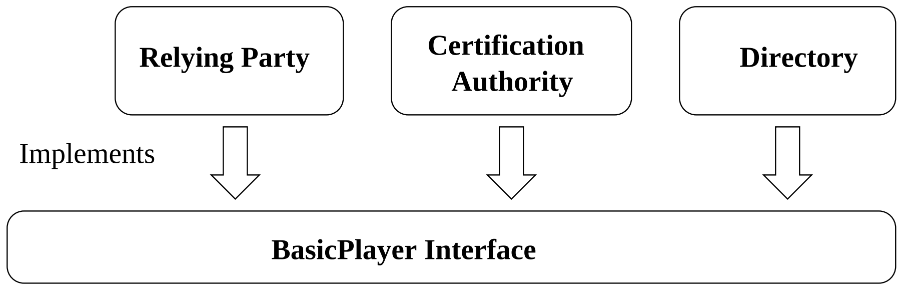


Figure 9.3: Implemented entities in the operational model

to CAs and directories. The model leaves an interface for adding additional protocols that the relying party may use for communication, such as the protocols for PKI-enabled applications. In some circumstances, we use a single relying party in the model to represent many users, even the entire relying party population in an enterprise PKI. Thus, it is very important that our relying party model can express the common features as well as the differences in the population. The diversity is mostly related to user behavior, which we are going to model using protocols.

Certification Authority (CA) A certification authority shares some functionality with the relying party, such as retrieving data from directory and validating certificates, and using a local cache. Its main function, however, is to manipulate data in its directory. The CA inserts newly issued certificates and deletes expired certificates. Furthermore, if the CA supports CRLs, it needs to update the CRLs for newly revoked certificates as well as for

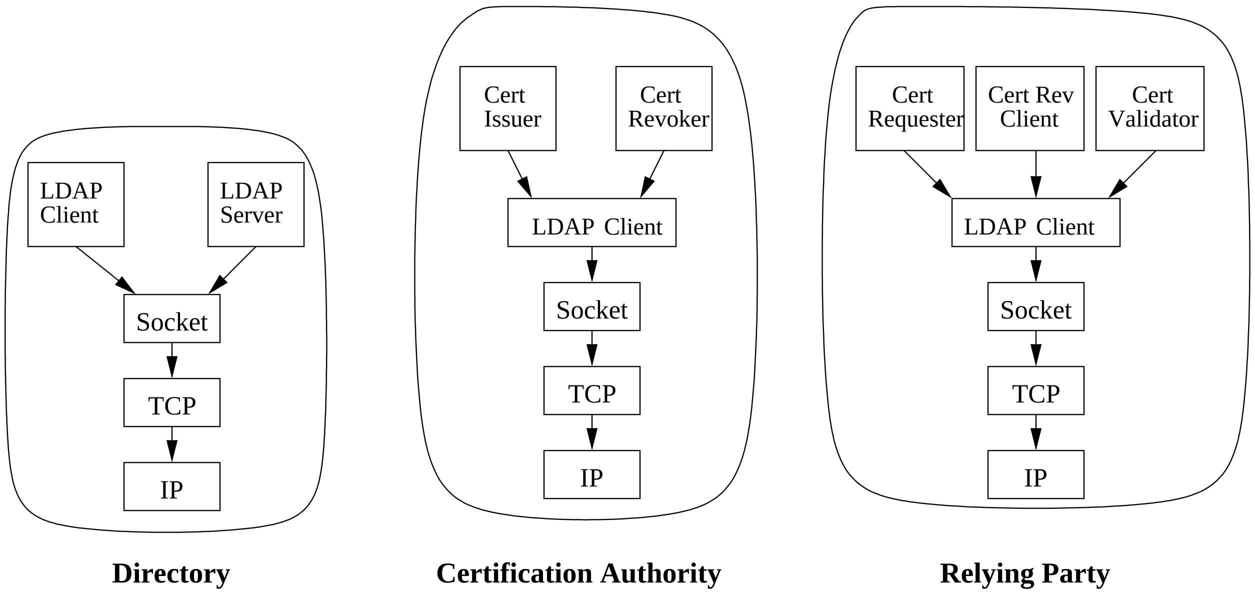


Figure 9.4: The demonstration of example protocol graphs for each type of PKI entity. The basic communication protocol is LDAP. The certificate issuing and revoking protocols are modeled as well.

periodical updates according to the configured CRL update interval. CAs also maintain the database for serial number usage. There are two methods typically used to assign serial numbers to certificates—sequence numbers or randomly generated serial numbers. We use the first approach in the model by default. Thus, CAs remember the maximum value that has been assigned so far. If needed, the model can also adopt the second approach. Visiting randomly generated serial numbers avoids the need to remember any value and therefore eliminates the risk associated with a crash of the database.

Directory The directory maintains the database of certificates and CRLs. It supports LDAP (the standard directory protocol) for data retrieval and manipulation. Relying parties are granted read-only privilege, while CAs have full privilege. According to LDAP, the information is accessible using the (namestring, attribute) pair. The directory model supports several popular directory attributes: *cACertificate*, *userCertificate*, *crossCertificatePair*, and *CertificateRevocationList*. Again, it is easy to extend the model to support other attributes.

9.2.3 Protocols

Protocols configure the entities and control their behaviors. The operational model provides four protocols:

- the certificate issuance protocol,
- the certificate revocation protocol,

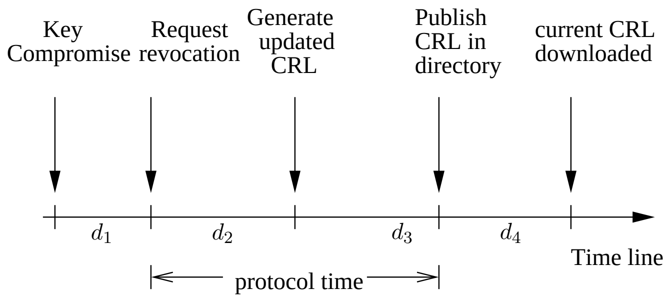


Figure 9.5: Protocol latencies for revoking certificates.

- LDAP, and
- the certificate validation protocol.

We model all PKI protocols at the application layer in the TCP/IP stack. In SSFNet, each host contains a *protocol graph* representing the network protocols that are supported by the host. Figure 9.4 illustrates typical types of hosts in the PKI operational model and their supported protocols. All services rely on socket connections. For CAs and relying parties, the LDAP client protocol is the basic application protocol. Advanced protocols for issuing, revoking, and validating certificates rely on the LDAP client protocol. In this chapter, we discuss details of certificate issuance protocol, certificate revocation protocol, and LDAP. We will present the certificate validation protocol and performance study in Chapter 10

Parameter	**Unit**
`issue_cert_req_latency`	seconds
`issue_cert_gen_latency`	seconds
`issue_cert_pub_latency`	seconds
`issue_cert_total_latency`	seconds
`cert_size`	bytes

Table 9.3: Parameters for modeling certificate issuance.

Certificate Issuance Protocol In practice, there are many ways for certification authorities to issue certificates for users. The procedure is typically defined in *Certificate Practice Statement (CPS)* of the enterprise PKI. In general, there are several steps involved, such as registering users, verifying user identity, verifying the private key, generating the certificate, and distributing the certificate. The user or an authority may generate a key pair. We disregard the details by modeling the process using time latency models. Shown in Figure 9.6, there are three possible latencies injected between phases. The actual time delay values and their probability distributions are subject to model configuration. If the simulation users are not interested in modeling details in the certificate issuance protocol, they can also use a single time delay value d to summarize the total latency. Besides temporal behavior, the issuing CA also determines the size of the certificate. As we have discussed before, it is a

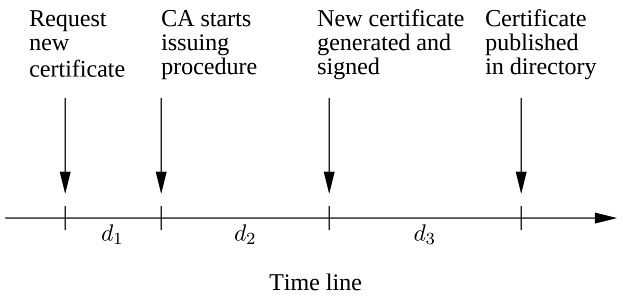

Request new certificate
CA starts issuing procedure
New certificate generated and signed
Certificate published in directory
d_1
d_2
d_3
Time line

Figure 9.6: Protocol latencies for issuing certificates. The process starts when a user generates a certificate issuance request and ends when the certificate is available in the directory, or is returned to the user, or both.

configurable parameter. In our experiments, the certificate size is 600–1500 bytes. Table 9.3 summarize the configurable parameters for certificate issuance protocol.

Certificate Revocation Protocol Similar to the certificate issuance protocol, the procedure for revoking a certificate consists of several phases. It is invoked by an event that causes the certificate revocation. In Figure 9.5, the example reason is "key compromise". This is one of the common reasons. After some time, the CA or the user notices the event and starts the protocol. Typically, a well-maintained enterprise PKI also provides a website for users to send requests to revoke their certificates. This interface is for users who still control their private key securely. It is not workable for key compromise, unless the user has multiple key pairs and only a subset have been compromised. The protocol starts right after the website receives a request. Otherwise, the CA administrator collects requests and revokes certificates in a batch. Nonetheless, it is commonly believed that the certificate revocation protocol starts when a revocation request is received. The CA finishes the protocol by generating and signing a new CRL and publishing it in the directory. Some time after publication, the CRL copy is retrieved by a relying party. The last time delay is not controlled by the certificate revocation protocol. Instead, the value is calculated together with certificate validation protocol. However, users may wish to examine the certificate revocation protocol alone. Thus, we still make this parameter configurable in the model.

Parameter	**Unit**
`revoke_cert_req_latency`	seconds
`revoke_cert_CRL_gen_latency`	seconds
`revoke_cert_CRL_pub_latency`	seconds
`revoke_cert_CRL_dwld_latency`	seconds
`CRL_update_interval`	hours
`CRL_validity`	days
`certificate_population`	–
`revoke_cert_rate`	percent

Table 9.4: Parameters for modeling certificate revocation.

We divide the entire procedure into such detailed phases so that users can configure

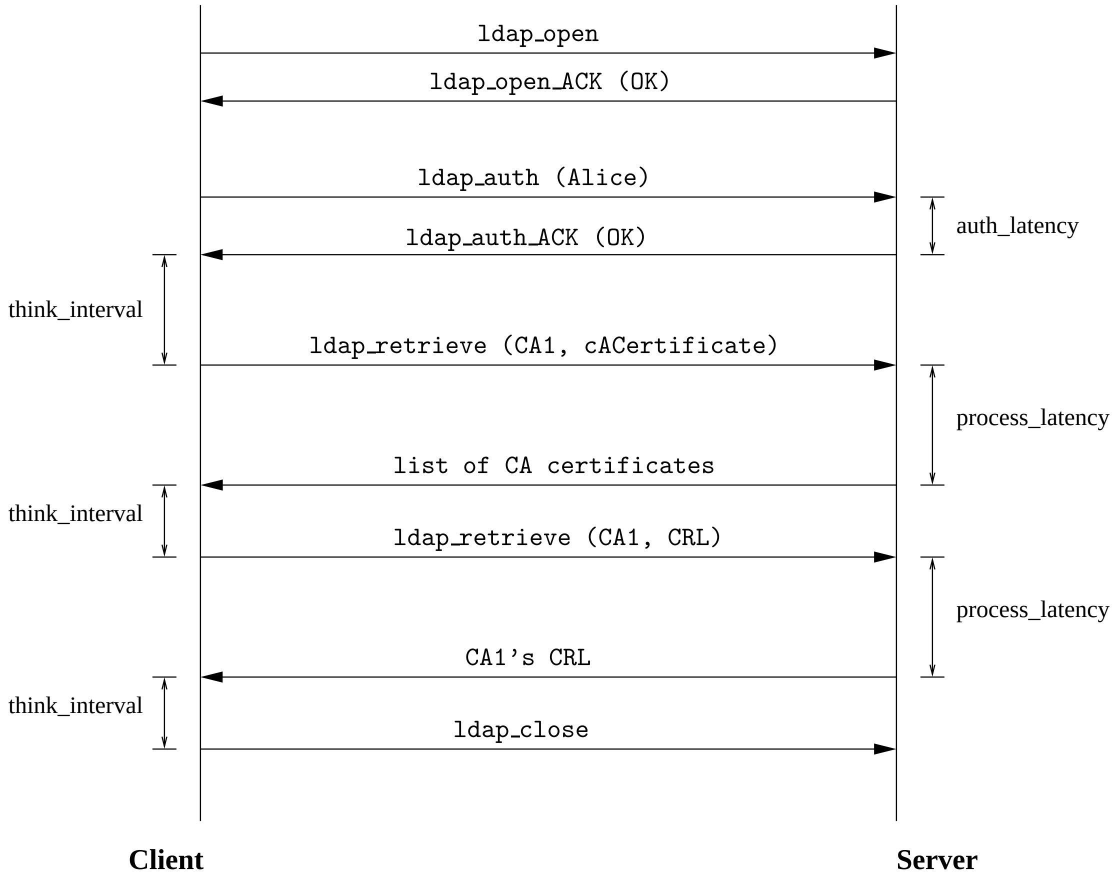


Figure 9.7: Example communication between LDAP client and server.

the model with different timing delays to conduct close examinations. Besides the actual certificate revocation action, the CAs should also update the CRL copies in the directory regularly. Periodically, the CA takes the latest CRL copy, deletes any expired certificates, generates a new CRL with an updated certificate list, and publishes it in the directory. The time interval is configurable. Moreover, if the CA has collected some revocation requests, it will process them at this point as well. The revoked certificates are appended to the list accordingly.

Once the new CRL is generated, the CA should also specify the size of the resulting data file. As we have discussed, the size is determined by the number of revoked certificates in the list. In turn, this number is calculated directly or from the total certificate population issued by this CA and the certificate revocation rate. The model allows users to configure these parameters. Table 9.4 summarizes all configurable parameters for the certificate revocation protocol.

LDAP The LDAP module controls the behaviors of communication between the LDAP server and its LDAP clients. Directories are typical entities that operate LDAP Servers. CAs and relying parties are LDAP clients. Figure 9.7 sketches the protocol diagram

Message	
`ldap_open`	`ldap_retreive`
`ldap_open_ACK`	`ldap_data`
`ldap_auth`	`ldap_insert`
`ldap_auth_ACK`	`ldap_insert_ACK`
`ldap_close`	`ldap_update`
`ldap_close_ACK`	`ldap_update_ACK`

Parameter	**Unit**
`persist_connection`	true/false
`inter_request_think_latency`	seconds
`auth_process_latency`	seconds
`request_process_latency`	seconds

Table 9.5: Messages and parameters for LDAP model.

using an example. There are two types of communications. The first set of messages controls the network connection between server and client. Such messages include `ldap_open`, `ldap_close`, `ldap_authen`, and their corresponding acknowledgment messages. The client authenticates itself to the server. This step is necessary for CAs to gain full privilege to the directory. Certain processing latency can be injected at this point. The time delay value is configurable. Furthermore, the client can choose to use the `persist_connection` option to keep the connection open until it finishes all data requests. Otherwise, the LDAP directory will close the connection after serving the one data request.

The other type of communication involves data exchange between client and server. We have modeled three types of requests. The `dir_retrieve` request asks for data from the server. The `dir_insert` request requires new data to be inserted into the LDAP server. For each request, only one directory attribute will be processed. In other words, if the client needs to download all CA certificates to the CA and the CRL it has issued, the client should send two separate requests. This design is consistent with the LDAP protocol specification. We have also designed a special type of request, `dir_update`, to ask the LDAP server to delete all certificates and CRLs that are expired already. All of these operations take time. Hence, we also designed configurable parameters to model operational latencies. Table 9.5 summarizes all LDAP messages and configurable parameters in our model.

9.2.4 Network Topology

The network topology module supports entities and their protocols. Here, we concentrate on the layout of subnetworks and their connectivities. Using DML, provided by SSFNet, users can configure any type of network topology. We suggest a type shown in Figure 9.8.

It is a star-shaped network topology that can be easily scaled to a large number of enterprise PKI domains. The topology consists of a number of PKI domains and a "routing core". There are six autonomous systems with one BGP router each. They are fully connected. The sole function of this routing core is to help connection with PKI domains.

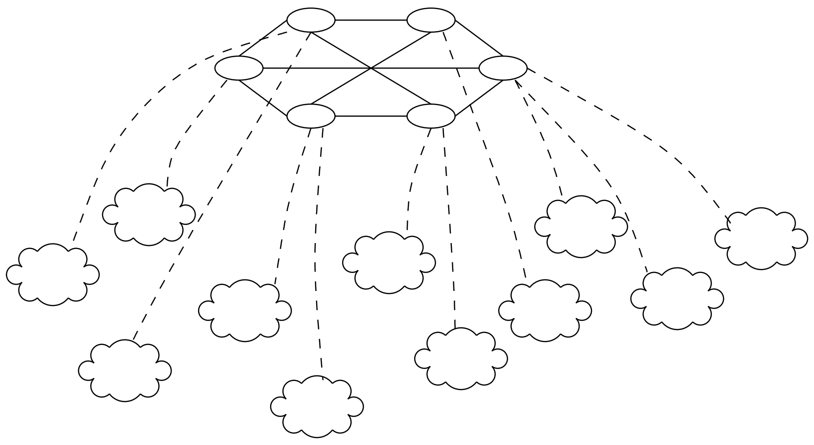


Figure 9.8: High-level network topology.

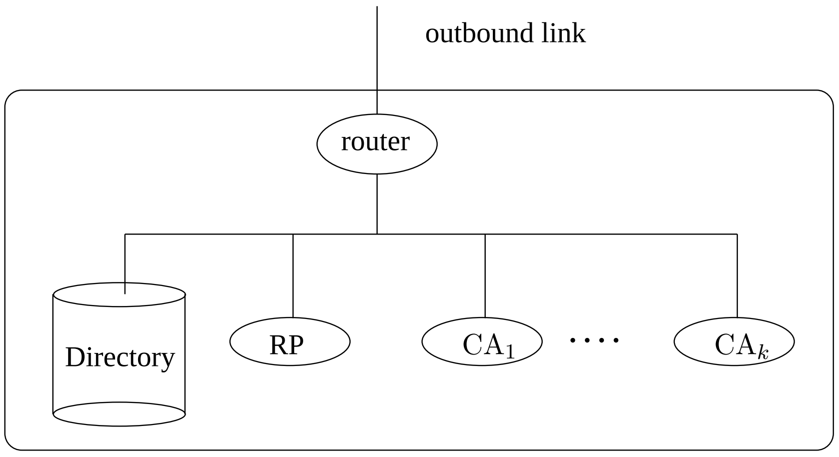


Figure 9.9: Detailed network topology within one PKI system.

There are a number of PKI domains connected to the routing core. Each PKI domain is an autonomous system with one BGP router. Other entities are directly connected to this router. Such a design matches practical deployment in the real world. Typically, an enterprise or an institution has its own subnetwork with its own administration policies. Such subnetworks can be considered as an autonomous system. To let PKI domains directly connect to the routing core, we try to minimize the number of routers involved in the communication, thus minimizing the performance impacts from non-PKI related protocols. Such a topology can be easily scaled to a large number PKI domains.

Within one PKI system, users can configure any type of entities in any number. In our design, a directory can serve the entire PKI domain. It is able to store certificates and CRLs from all CAs within the domain. Hence, we suggest to configure one directory for a PKI domain. Moreover, one replying party is able to represent the entire user population in a PKI system. The requests by one user may be modeled by a request inter-arrival distribution. Then, requests by multiple users can be represented by a combined distribution. Hence, we suggest to configure one relying party for the entire PKI domain. This approach can save the computational resource consumed by simulation. Finally, since there could be a large number of such PKI domains, we cannot assume a complicated network topology for each of them. We try to keep it the simplest architecture, unless complex topologies are specially required. Figure 9.9 illustrates an example network topology for one PKI system.

9.3 Monitoring and Measurement

Option	Description
`show_ldap_connection_open`	Show when LDAP connection open occurs
`show_ldap_connection_close`	Show when LDAP connection close occurs
`show_ldap_authentication`	Show when LDAP server processes an authn request
`show_ldap_data_request`	Show when LDAP data request occurs
`show_ldap_session_summary`	Show the summary of a LDAP session
`show_ldap_delay_timer_set`	Show when LDAP server process latency timers are set
`show_ldap_delay_timer_exp`	Show when LDAP server process latency timers expire
`show_ldap_think_timer_set`	Show when LDAP client think latency timers are set
`show_ldap_think_timer_exp`	Show when LDAP client think latency timers expire
`show_CRL_timer_set`	Show when CRL update timer is set
`show_CRL_timer_exp`	Show when CRL update timer expires
`show_dir_data_insert`	Show when new data is added into the directory
`show_dir_data_delete`	Show when data is deleted from the directory
`show_dir_summary`	Show summary of the directory at the end of simulation
`show_cert_issue_msg`	Show when messages for issuing certificate occur
`show_handle_certissuing`	Show various steps during certificate issuing
`show_cert_revoke_msg`	Show when messages for revoking certificate occur
`show_handle_certrevoke`	Show various steps during certificate revocation
`show_cert_validate_msg`	Show when messages for validating certificate occur
`show_handle_certvalidate`	Show various steps during certificate validation

Table 9.6: Monitoring options for PKI operational model.

We designed a set of monitoring options for monitoring a simulation run. Each option is a boolean variable. Users can turn on a subset of the options to observe the desired types of behavior. The monitoring facility is built upon the built-in monitoring structure provided by SSFNet. The options are configured in a model's DML file. Users can print out output in ASCII form or store it as a binary record. Current implemented options support the following types of events:

- LDAP event type: connection establishment, authentication, and connection maintenance.
- LDAP data sending and receiving.
- Timer setting and expiration.
- Directory data changes.
- Message sending and receiving.

A full list of monitoring options is given in Table 9.6. When printing out monitoring information, we try to cover as much information as possible. For data-related events, the print-outs will include the temporal information, as well as the data size involved. With the help of scripts, users can easily process the output of the monitoring options and produce desired measurement data. The following list gives some example measurement metrics that we can produce from measurement data.

- Number of requests
- Timing distribution of requests
- Data size for each request
- Network delay for each request

The actual choices depend on the purpose of the performance study using the simulation. The requests generally refer to any types of request related to PKI services, such as issuing request, revocation request, validation request, or individual LDAP request.

Chapter 10

Evaluating Certification Path Discovery

The certificate validation service is one of the primary functions in a PKI system. As we have discussed before, this service consists of two steps: building a candidate certification path for the target certificate and validating this path. RFC3280 defines the standard algorithm for validating certification paths. Yet, RFC3280 does not define the actual first step of certificate validation: constructing a potential valid certification path. There is no standard algorithm for certification path discovery. There are many options an implementer may encounter. By modeling certification path discovery, we are able to understand its performance impact and to compare different building options.

There are challenges in modeling and simulating certification path discovery. First, the algorithm is complex. In particular, the simulation should be able to model trials. The algorithm picks one certificate out of a set of candidates and tries out the path to see if it can reach the target or the trust anchor. If not, the algorithm should be able to quickly roll back to the choice point and try another candidate. Modeling this procedure can be tricky. Second, the scale of the system is an issue. We need to model the algorithm efficiently, so that many entities can use the module in the simulation without significantly hurting simulation performance. Third, we need to address the fact that a great many options exist for the algorithm. It is our job to identify critical ones, implement them efficiently in the model, and compare their performance. The challenge is not only the selection of the options, but also modeling and data analysis.

We have designed a model of the certification path discovery algorithm: *PathBuilder*. We build it upon the operational model. The PathBuilder builds certification paths upon request, experiments with different building options, and produces performance measurement results. In the rest of this chapter, we present the PathBuilder model and discuss the performance study using PathBuilder with the operational model introduced in Chapter 9.

10.1 PathBuilder Model

The PathBuilder model should be efficient, should be scalable to serve a large numbers of requests, and should be flexible to handle a variety of building options. Figure 10.1 sketches

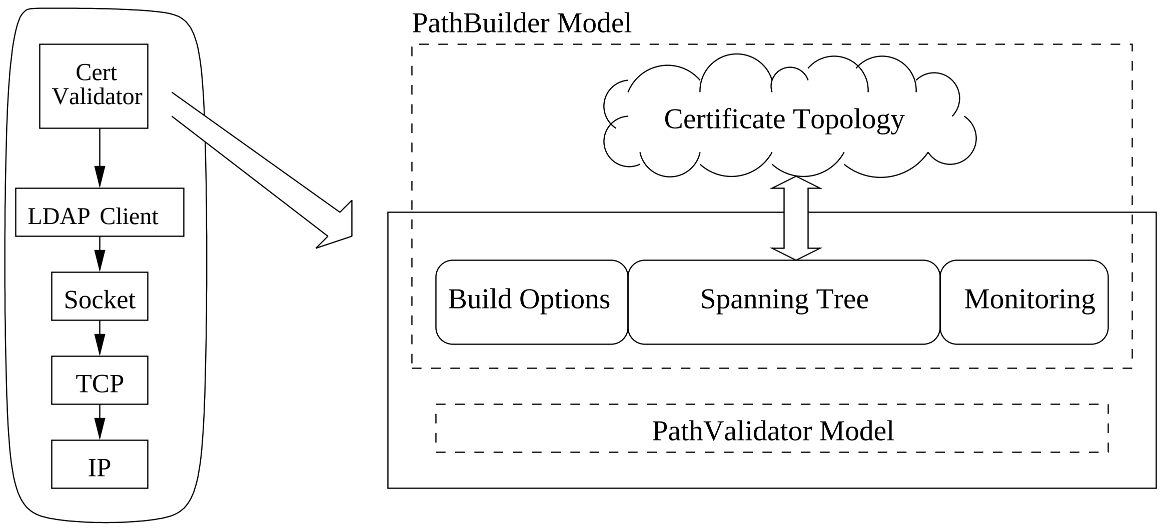


Figure 10.1: Structure of a host that supports the PathBuilder model. In the protocol graph, the cert validation protocol is running above the LDAP client. The protocol contains two models, for certification path building and validation respectively. (PathValidator Model is part of the future work.) The PathBuilder model consists of four modules. If there are multiple hosts running their own cert validation protocols, the certificate topology module is shared among them.

the design of the PathBuilder model. The *cert validation protocol* uses PathBuilder and PathValidator. We configure the protocol to be run on top of the LDAP client.

There are four primary modules in the PathBuilder:

- The search tree module
- The certificate topology module
- The build options module
- The monitoring module

The search tree module is the central focus in our design. It models the actual algorithm that constructs certification paths. It relies on the information from the certificate topology module. We assume that PathBuilder has a complete and consistent image of the certificate topology. In practice, path builders have incomplete image of certificate topology. Hence, the PathBuilder model will combine the image with further configurations to model the actual view of the image by a particular relying party. With this assumption, it makes sense to let all path builders share one image of certificate topology. Therefore, it is outside the protocol graph. Besides the search tree module, there is a module to handle building options for the algorithm. Finally, the monitoring module produces output data that describes the specific behavior of PathBuilder.

Field	Description
`node.name`	Node id that uniquely identifies the node
`node.DN`	Distinguished name for the node (optional)
`node.size`	Size of user population by the CA (optional)
`edge.from`	Node id of the certificate issuer
`edge.to`	Node id of the certificate subject
`edge.validity`	Validity of the certificate (optional)
`edge.revoke`	Boolean indicating if it is revoked (optional)

Table 10.1: Properties in the certificate topology model. Nodes and edges are the basic elements in the topology. Fields of the nodes and edges express these properties.

10.1.1 Certificate Topology

The certificate topology module captures the PKI architecture used in the simulation. We use a new representation for certificates in the certificate topology module. Here, the certificate topology is a graph with nodes and edges. CAs and bridge CAs are the nodes, and certificates are the directional edges pointing from the issuer to the subject. An edge only has attributes of the certificate related to certification path building. Table 10.1 summarizes the information defined for a node and an edge. Besides basic information that constructs the graph, other fields are the resource used for experimenting with building operations.

The simulation maintains the certificate topology in two ways. First, the majority of the information comes from the static configuration in the DML file. The simulation then constructs the topology at the beginning of a simulation run. Second, once the simulation is started, the certificate topology changes when new certificates are issued, when some certificates are revoked, or when some expire. The entities that control such activities will inform the certificate topology model. Hence, the certificate topology always maintains the up-to-date image of the PKI architecture in the simulation.

10.1.2 Search Tree

Before we present the model formally, let's use an example to explain the intuition behind the design of search tree model. Assume that CA `E` issued a user certificate to `EE`, (`E` $\rightarrow$ `EE`). Alice is the entity who is about to build the certification path for (`E` $\rightarrow$ `EE`). She trusts `TA` as her trust anchor. Figure 10.2 shows the example PKI architecture that Alice is dealing with. We also assume that Alice does not have any *priori* knowledge of the PKI architecture. The building procedure could be the following. Alice decides to build a path in the forward direction. She starts from the target. She immediately reaches `E`. Now Alice retrieves all certificates issued to `E`. There is only one choice that leads Alice to `C`. Next, Alice retrieves all certificates issued to `C`. There are two choices, Alice chooses `A`. Finally, Alice reaches `TA` when retrieving certificates issued to `A`.

As we can see, Alice encounters choices during the building procedure. In fact, all possible choices can be expressed using a search tree. The *forward search tree* expresses all choices of path construction in the forward direction. The example forward search tree is rooted at `E`, the issuer of the target certificate.

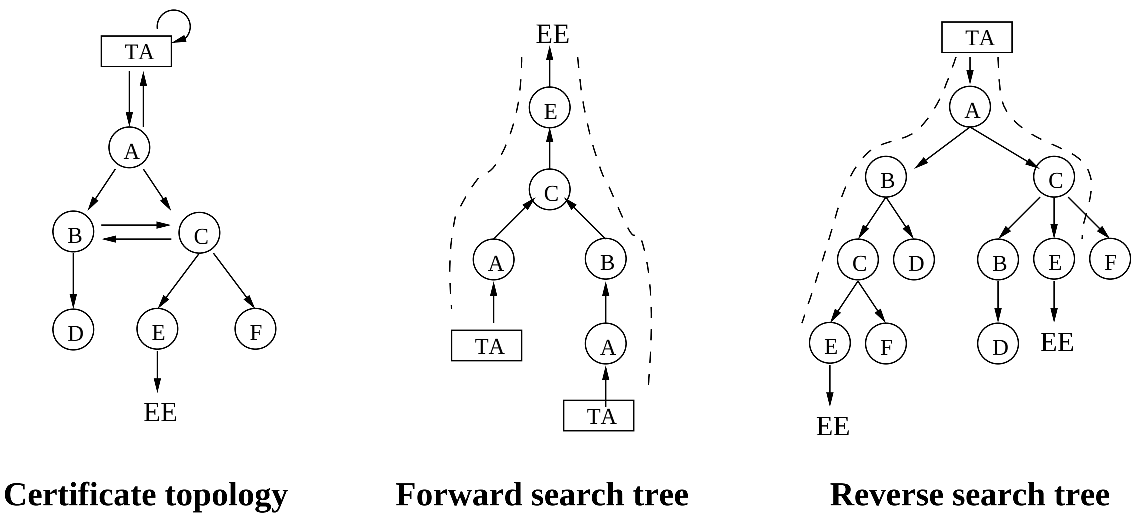


Figure 10.2: Search trees for example certificate topology. The trust anchor in the example is `TA`. The target is the certificate `(E` $\rightarrow$ `EE)`. Since the algorithm knows the target, it can use the issuer of the target as the end point of certification during the construction. The "forward search tree" is rooted at `EE`. The leaves are either `TA` or dead ends. The "reverse search tree" is rooted at `TA`. The leaves are either `EE` or dead ends. Dotted lines indicate valid certification paths that could be discovered and are potentially valid.

Similarly, if Alice builds a certification path in the reverse direction, the *reverse search tree* expresses her choices. The tree is rooted at her trust anchor `TA`. Figure 10.2 shows that there are two candidate certification paths. If Alice is lucky, she discovers the shorter one first. Typically, relying parties prefer shorter certification paths. This is because short paths are easier for validation. There are fewer signature verifications and other validation operations on the path. Furthermore, a shorter path has a lower probability to be invalid due to revoked certificates. However, it is possible that Alice discovers the longer path first. It is also possible that Alice reaches `D` at some point, and is forced to back up one step and try out alternative certificates. Alice's choices of certificates at each step affect the final building results. The number of LDAP requests and contents of retrieved data could be different as well. A search tree can help us understand the factors that affect Alice's choices and how to build a path efficiently while reducing the number of LDAP requests and/or the amount of data retrieved.

Without assumptions of how to prioritize the branches, the choice of a branch out of a set of candidates is completely random. Thus, we use a probability function to model Alice's tree walk. At each step, the model generates a random value from some defined probability function and chose the corresponding branch according to the value. We call this entire process as a *probability tree walk*. Now, with this model, we can easily reflect Alice's preference on some branches by assigning higher probability on them. The basic requirement at each internal node in the search tree is that the sum of probability values of its branches is equal to 1.

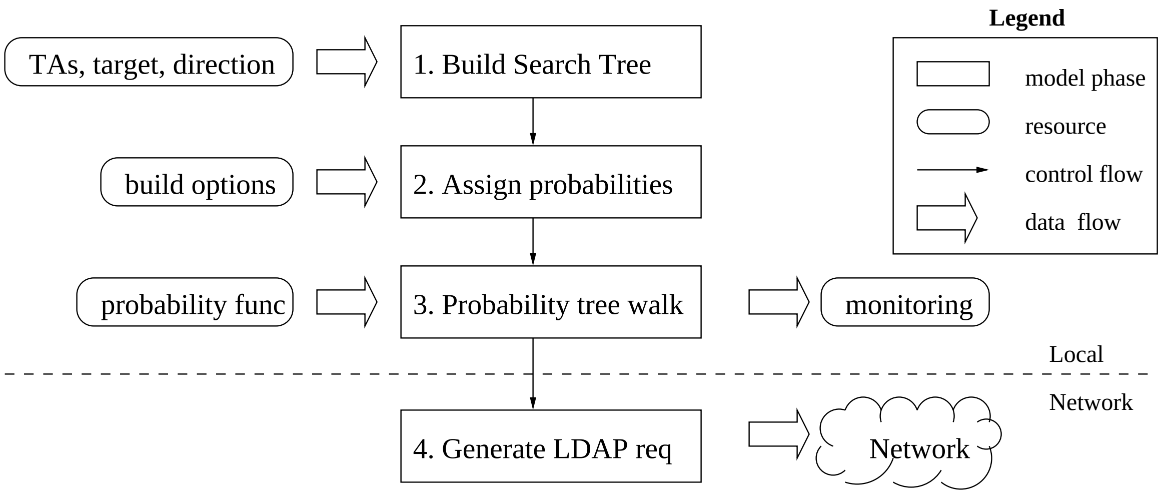


Figure 10.3: The structure of the search tree module. It has four phases in sequence. In each phase, the model takes the parameters (shown to the left of phases), fulfills the tasks, and produces output. The monitoring facility is responsible for printing performance output data from the simulation. The network is the environment where the actual resulting LDAP requests are executed by the LDAP client protocol session.

The module

As shown in Figure 10.3, the search tree model contains four phases. The first three phases happen locally within the PathBuilder, while the last phase is the network operations with the LDAP client and directories.

Build search tree In the first phase, the simulation constructs the search tree, $ST = T(V, E)$ by the request of a relying party. The simulation uses the global certificate topology to build ST. The request from the relying party has three parameters: trust anchors, target certificate, and building direction. The relying party can specify one or more trust anchors to build a certification path. The simulation treats the set of trust anchors as a single node in the search tree.

Assign probability In the second phase, the simulation assigns a probability on each $e \in E$. The parameter is a set of building options $\mathcal{O}$, to be discussed later. For now, let's assume that $\mathcal{O} = \emptyset$. Each internal node $u \in ST$ has a set of children nodes children$(u) = \{v|(u, v) \in E\}$. Let $N_u = |\text{children}(u)|$ be the number of children of u. We assign equal probability to each branch:

$$p = \Pr((u, v)) = \frac{1}{N_u}, \quad \text{for each } v \in \text{children}(u) \ .$$

Probability tree walk In the third phase, the simulation starts a tree walk until it discovers a candidate certification path or finishes exploring the entire tree. Then we send the resulting tree walk logs to the monitoring facility. Randomly generated values control the choices of branches at each step of the tree walk. In above example, the index of the chosen branch is simply calculated as

$$i = \lfloor R \cdot N_u \rfloor + 1 \ ,$$

where $R \in [0, 1)$ is the randomly generated value. After the ith branch is chosen, the simulation sets the probability of the branch to 0 to mark that it has been explored, $Pr((u, v_i)) = 0$. Now, we set $N_u \leftarrow N_u - 1$, and label the remaining branches from 1 to N_u. The probabilities of the rest of the branches are adjusted as $p = 1/N_u$ accordingly. In any case, $\sum_{j=1}^{N_u} \Pr((u, v_j)) = 1$ holds.

Besides adjusting probabilities at each tree walk step, the simulation also records a log $\mathcal{L}$ of the sequence of actions that the tree walking has taken. Each element is a tuple `(CA name, data type)` indicating how the PathBuilder should fetch data for the next step in the tree walk. There are three choices for the data type: CA certificate, cross-certificate pair, or CRL. Among the criteria used by CAM to prioritize branches (see Section 8.2.3), we assume that, by default, PathBuilder applies the first criterion. That is certificates from **cACertificate** have priority over certificates from **crossCertificatePair**. This is a natural choice. Retrieving CA certificates typically results in fewer certificates and a smaller amount of data. In hierarchical PKIs, there is usually only one CA certificate issued to a CA. In some circumstances, the simulation needs to retrieve both the CA certificate and a cross-certificate pair before it can advance to the next step in the tree walk. This happens because the CA certificate retrieval may fail to produce any useful certificates for the next step.

In the end of the third phase, the simulation calls the monitoring facility to report measurement data and passes $\mathcal{L}$ to phase four to produce a list of necessary LDAP requests.

Generate LDAP request In the last step, the simulation takes the resulting log $\mathcal{L}$ and translates it into a sequence of LDAP requests. If the local cache of retrieved certificates and CRLs is configured, the simulation will use the cache to reduce the number of necessary LDAP requests. Finally, the PathBuilder hands over the list of LDAP requests to the LDAP client to fulfill the rest of the operations in the network environment. The PathBuilder will watch for the execution of LDAP retrievals. In the end, it reports to the monitoring facility about the time delays and amount of data being transmitted over the network.

10.1.3 Build Options

Build options are criteria that can distinguish child branches and change the way the probabilities get assigned. In Section 8.2.3 we discussed a variety of choices the certification path algorithm may use to potentially speed up the building process. As these criteria are expressed in words mostly, it is challenging to translate them into quantitative probability values. Here, we pick a couple of build options to explain how we construct the probability model.

Let's start with an easy option—certificate revocation using CRLs. This option work helps the algorithm if building in the reverse direction. If the algorithm is building in the reverse direction, there are two additional tasks for the search tree model. The tree walk log $\mathcal{L}$ should contain actions of retrieving CRLs at each step. Upon discovering a revoked certificate, the search tree model should reset the probability on the corresponding branch to zero, so that the probability tree walk will not visit the entire subtree following that branch.

Next, let's examine a more complicated build option, from CAM implementation (as we discussed in Section 8.2.3)—certificates that match more RDNs within the issuer DN and the subject DN have priority. We denote this option as the *RDN matching option.* The assumption in using this option to prioritize branches is that the CAs in the same enterprise PKI tend to share a significant number of RDNs in their DNs. Thus, choosing certificates with similar DNs means that local CAs have priority.

We use the following strategy for assigning probabilities. Let u be an internal node in the search tree T, children(u) be the set of children nodes of u, C be the set of branches from u, $C = \{c = (u, v) \mid v \in \text{children}(u)\}$, and match$(u, v)$ be the number of RDN matches between the DNs of u and v. We assign the probabilities in three steps.

1. Calculate match(u, v) for each $v \in \text{children}(u)$. There could be multiple values for different children. Let M be the set of all possible values of matches computed from C. The maximum element in M is defined as $maxm$.

2. Divide C into subsets, each with the same match value. The subset of branches with match value $maxm$ is defined as $maxC$.

3. Assign probability 0 to all branches $c \in C - maxC$. The branches $c \in maxC$ get equal positive probabilities, $\Pr(c) = 1/|maxC|$. The assignment satisfies $\sum_{c \in maxC} \Pr(c) = 1$ and $\sum_{c \in C} \Pr(c) = 1$.

This strategy ensures that the probability tree walk only explores branches with maximum match values.

We also modify the probability adjusting algorithm to reflect this build option. The tree walk uses the same strategy that we used for the complete random probability model—we continue to explore branches and adjust the probabilities on other branches, until the set of available branches becomes empty. The difference is that for the RDN matching option, the set of branches is $maxC$, instead of C. Once $maxC$ is empty, the tree walk replaces $maxC$ with the branch subset with the second maximum match value, assigns the probabilities on branches in new $maxC$, and continues the tree walking process. We say that the tree walk visits all branches in C if all subsets have been visited.

To support the RDN matching option, we need to know the DNs of CAs. We create a new DN module in the certificate topology to record all distinguished names configured for CAs. We specify the configuration in a DML file.

Besides CRL checking and RDN matching, 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. Nonetheless, the modular design of the search tree module limits the modification of probability assignment and adjustment modules.

If the simulation enables multiple build options, the model needs to handle combined probabilities. The branch prioritizing and elimination maintain a nice property: categorizing branches into subsets, such that only the branches in the preferred subset have positive probabilities. Now, if we combine two build options, it is natural that we assign positive probabilities on branches that belong to both preferred subsets. Let's examine the two

Monitor option	Description
`show_tree_size`	Show the search tree size
`show_leaf_count`	Show the number of leaves in the tree
`show_pathlen_max`	Show the maximum path length or the tree height
`show_pathlen_min`	Show the minimum path length
`show_pathlen_avg`	Show the average path length
`show_num_valid_path`	Show the number of candidate paths in the tree
`show_num_ldap_req`	Show the number of LDAP requests sent out
`show_num_cacerts`	Show the number of CA certificates retrieved
`show_num_xcerts`	Show the number of cross certificate pairs retrieved
`show_num_CRLs`	Show the number of CRLs retrieved
`show_build_delay`	Show the time delay of the entire path building procedure
`show_data_amount`	Show the total amount of data transmitted over the network
`show_treewalk_log`	Show the entire log of the probability tree walk
`show_ldap_requests`	Show the details of each ldap request

Table 10.2: The monitoring options for PathBuilder. Users can turn on a subset of the options for measurement purposes.

example options again. Suppose for node u, we have identified that $\{C_{11}, C_{12}\}$ are the two subsets by CRL option, and that $\{C_{21}, C_{22}, C_{23}, C_{24}\}$ are the subsets by RDN matching option. We place the subsets in descending order of preference values. Then the new positive probabilities are assigned to $c \in C_{11} \cap C_{21}$.

The case is more complicated when $C_{11} \cap C_{21} = \emptyset$. The complexity comes from the ordering of these two options. If the PathBuilder prefers to use the CRL option first, the next candidate subset should be $C_{11} \cap C_{22}$. The idea is that C_{11} should always be the subset that contributes to the intersection. The reason is that the CRL option is used for branch elimination. A branch in C_{12} should never be considered a valid candidate. If the RDN matching option is preferred, the situation is slightly different. The sequence of candidate subsets is $C_{12} \cap C_{21}$, $C_{11} \cap C_{22}, C_{12} \cap C_{22}, \ldots$ This time, we should consider the subsets with highest possible matching value first. In general, if the simulation enables multiple build options, it should configure the ordering of them properly. The probability assignment and adjustment process should follow the same fashion we have shown in CRL and RDN matching options.

10.1.4 Monitoring

Like the monitoring facility for the PKI operational model, we also design a set of options for the monitoring behavior of PathBuilder. The model keeps track of the all related events and measurement data. Table 10.2 presents all monitoring options. For two options, `show_build_delay` and `show_data_amount`, the output information comes from the LDAP protocol traffic measurement by the operational model.

10.2 Performance Study

So far, we have introduced the operational PKI model and the PathBuilder model. In this section, we present simulation experiments that use these models to evaluate the performance of 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.

10.2.1 Requesting Protocol

The requesting protocol is the new protocol we designed to activate the PathBuilder during simulation. The basic function is simple—generating parameters for certification path building and passing them to PathBuilder. The requesting protocol keeps sending out requests to PathBuilder. The inter-arrival time of requests follows an exponential density function with the mean value $1/\lambda = 10$ seconds. A relying party can have a requesting protocol attached to its protocol graph. Moreover, we examine a number of settings on the contents of the requests. The configurations depend on the location of the relying parties, the target that the relying party chooses, and the building directions.

Location Our experiment setting has either one relying party or multiple ones, each running a requesting protocol. In either case, we assume that the requesting protocol will always use the root CA of the local PKI domain of the relying party as the trust anchor in the requests.

Choosing target Generally, the requesting protocol can select any CA in the certificate topology as the issuer of the target certificate. Since we use one relying party module to represent the entire user population in its local PKI domain, it makes sense to use the requesting protocol to model the requesting behavior of the large number of users. One of the interesting behaviors we can model is the requesting locality. Generally, ordinary relying parties use certificates by local users more often than the ones by remote users. For instance, as a Dartmouth user, Alice mostly validates certificates from other Dartmouth users. We use a *preference* rate to reflect how much the user prefers to validate the local certificate. The value varies from 0.2 to 0.9.

Hence, the process of selecting a target CA is the following. First, the requesting protocol uses a randomly generated value to decide whether it should select the target CA from the local domain or remote domains. Second, in either group of CAs, the protocol randomly chooses one CA. A virtual user certificate issued by this CA is the target in the next request.

Direction The direction is the third parameter in the request. We are going to examine both directions in the experiments.

10.2.2 Certificate Topology

We have two choices in configuring certificate topology. Either we use a certificate topology based on real data from real PKIs, or we can generate virtual topologies for experiment purposes. As the purpose of the performance study is to understand the performance impact on large-scale PKI systems, we tend to build large certificate topologies. We realize that bridge-to-bridge development is the direction people are pushing to for future PKI systems. Thus, we follow this trend and examine the current deployment situation.

As we have discussed in Section 8.3.1, currently there are four principal bridge CAs deployed, under construction, or in planning. We use these four 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 large sector CA, USHER, in the model. USHER issues certificates to CAs in some universities who do not have their own independent PKI management policies.

These bridge CAs have cross-certified with many PKI domains for government agencies, institutions, and enterprises. The 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. We configure the architecture directly into the certificate topology model. In addition, we have learned that the Canadian government PKI will cross certify with FBCA. Without knowing more details about the Canadian government PKI, we assume it is a simple hierarchy that has one root CA. We use the same strategy to configure architectures of other PKI domains. For instance, all PKI systems in aerospace industry have a simple one-CA architecture. They cross-certify with the CertiPath bridge CA. Many PKI systems by universities also have such an architecture. The only CA on campus may cross-certify with HEBCA, or become a subordinate CA of USHER.

Besides PKI systems in the United States, we also try to model the connections to the PKI systems in other countries. There is no bridge CA in Europe currently, yet we learn that the bridge technology is in plan. The future bridge CA may cross-certify with FBCA. Among all individual PKI systems in Europe, EuroPKI [43] has a large hierarchy. It contains 21 CAs from 11 different countries. Hence, we collected certificate information from the EuroPKI system, turned the root into a bridge CA, and configured it into the experimental certificate topology. Similarly, the Brazilian government has set up a restricted hierarchical PKI for all government agencies and enterprises [18]. The future efforts could be on expanding this architecture and cross-certifying the root CA with HEBCA. We configured this portion into the experimental certificate topology as well. Figure 10.4 demonstrates the entire certificate topology used for out performance study.

Configured experimental certificate topology consists of 51 PKI domains, 6 bridge CAs, and 103 ordinary CAs. It is the common practice that individual PKI systems have hierarchical architectures, while bridge CAs cross-certify with each other according to their needs.

Besides the architecture, we also configured the user population for each PKI system. Only a few PKI systems, such as the Johnson&Johnson PKI and the DoD PKI, provide us user certificate population size. For those whose sizes we do not know, we use a random number for configuration. The random number is at most 100,000.

Furthermore, to be able to study the RDN matching option, we need the distinguished names of CAs. We have data for FBCA, US government agencies, Johnson&Johnson PKI, EuroPKI, Brazilian PKI, and a few universities that cross-certify with HEBCA. The real data is directly configured into the certificate topology. Overall, the average number of matches between issuer DN and subject DN is 1.48. Table 10.3 summarizes the statistics

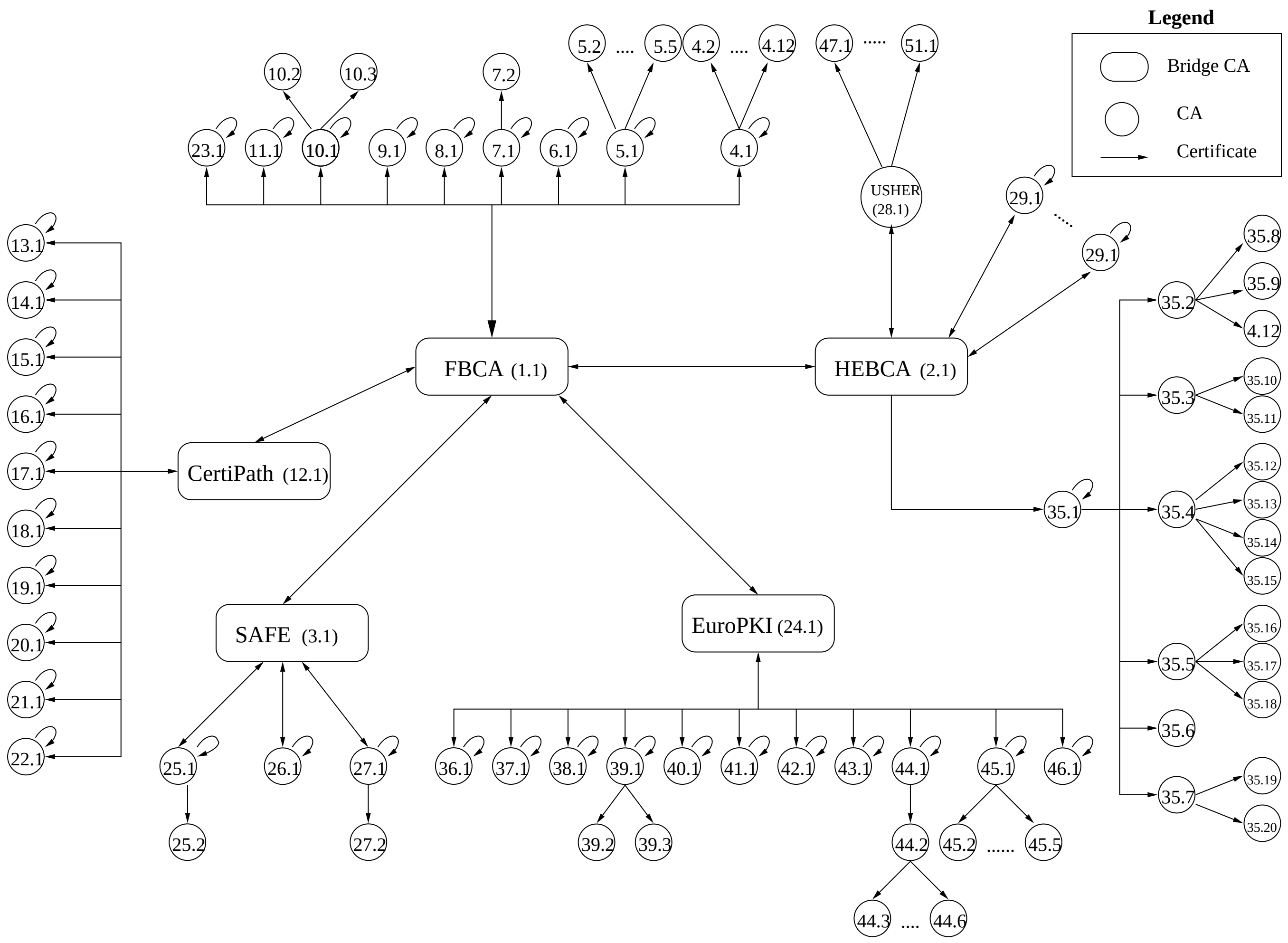


Figure 10.4: The configured certificate topology for experiments. The topology is a combination of current deployment and future plans. Each CA with a self-issued certificate can be treated as a trust anchor. The unique ID for each CA is the tuple: (domainID, caID). We assign an index number to each CA in the model for simple implementation.

Domains	Average match rate
All	1.48
DoD PKI	3.91
EuroPKI	0.97
Brazilian PKI	1.80

Table 10.3: RDN matching statistics. This table shows average match rate from configured DNs and certificates. DoD PKI, EuroPKI, and Brazilian PKI are three large local PKI domains. DoD PKI gives much higher matching rate.

of configured distinguished names. Notice that certificates in the DoD PKI match much more RDNs than the rest of the certificates. For the rest of the CAs whose DNs we do not know, we assume that the match number is zero or a random integer within a range. We'll use experiments to compare different assignment strategies.

To the best of our knowledge, our certificate topology is the first systematic attempt to model the emerging global PKI architecture. It expresses the current major efforts in building a bridge-to-bridge environment for PKI systems. It is a very large PKI architecture with 30 million users over 13 countries.

10.2.3 Operational Model Configurations

There are a few parameters in the operational model that we should configure.

Since the focus is the protocols for PathBuilder, we try to minimize the impact from other factors. We configure other network protocols using default and basic parameters, so that they do not display any abnormal behavior that may potentially affect the performance of PathBuilder. In each PKI domain, we model only three network entities: a BGP router connecting to the routing core, a directory that stores local certificates and CRLs, and a relying party that may support the PathBuilder requesting protocol. In our experiments, the certification path building process is much faster than the CRL update interval. So, we assume that CRL updating traffic will not bother PathBuilder. As a result, we do not really need a real CA in a PKI domain. There is no certificate to be issued or revoked during the experiment time period. So, all certificates are configured statically.

The only network protocol used by PathBuilder is LDAP. We configure the process latencies by a client and server. If the client has a list of LDAP requests to be sent to LDAP servers, it needs to "think" for a period of time before sending each request. This thinking time models any activity that the PathBuilder has done for preparing a LDAP request. In the experiments, `think_interval` is about 3–7 μs. On the other hand, the process latency for each request by the LDAP server is much larger. Like traditional queuing systems, the `process_latency` follows an exponential density function with mean time of 32 ms.

10.3 Results

In evaluating performance, we examine most types of measurement data provided by PathBuilder model. The comparison is mostly between two sets of experiments—forward direction and reverse direction. We also evaluate one build option—the RDN matching option. We compare the experimental results with the ones without any build options. The purpose is to examine whether the RDN matching option can improve the efficiency of the certification path building algorithm. In this section, we are going to present results demonstrating the performance comparison by different building directions, followed by more detailed analysis on user behavior and the building option. All numbers presented in this section are the average of multiple simulation runs.

Property	Forward	Reverse
`tree_size`	31.3	69.1
`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

Table 10.4: Properties of the forward search tree vs. reverse search tree. All numbers are the average of measurements from multiple simulation runs and by multiple certification path building requests.

10.3.1 Forward vs. Reverse

In this set of experiments, there are 45 relying parties, one for each PKI domain that are sending requests to build certification paths between their trust anchors and randomly selected targets. At this point, the local preference of selecting a target is fixed at 0.2.

Tree Properties

Table 10.4 compares the properties of the forward search tree and the reverse search tree on average. Reverse search trees are significantly larger than forward search trees. Experiments show that the tree size is doubled. Other properties indicate that reverse direction can produce flat search trees. The resulting average path length is about the same for both directions. Our experimental certificate topology is mostly hierarchical except in the center where six bridge CAs are cross-certified with each other. Thus, the certificate topology is naturally divided into two areas, the "mesh area" that consists of bridge CAs and the "edge areas" where individual hierarchies cross-certify with the mesh area.

Building forward in hierarchical PKI systems is much more efficient than building reverse. If building in the forward direction, the PathBuilder always handles only one certificate until it reaches the root CA in the hierarchy. Then, the PathBuilder has several choices while exploring in the mesh area. The path building process ends immediately when the PathBuilder crosses the mesh and reaches the trust anchor. In the experiment setting, relying parties treat the root CAs as trust anchors.

On the other hand, if the PathBuilder constructs certification paths in the reverse direction, it is faced with a lot more branches. Starting from the trust anchor, a root CA, the PathBuilder enters the mesh area immediately. Now, the PathBuilder is constantly dealing with a set of certificates when it is working in the mesh area and further goes into a hierarchy to find the target.

Network Performance

Table 10.5 illustrates the network performance for building an average certification path. The forward direction out-performs the reverse direction, as we expect. The PathBuilder ignores CRLs; it only sends out LDAP requests for retrieving CA certificates and cross certificate pairs. Although the forward search tree is much smaller, the numbers of LDAP requests are close for both directions. Recall that the PathBuilder tries CA certificates

	Forward	**Reverse**
# LDAP requests	36.2	40.0
# retrieved CA certificates	18.2	0
# retrieved cross certificate pairs	81.5	152.8
building delay	7.7 seconds	9.1 seconds
data size	89.8KB	122.19KB

Table 10.5: Comparison of operational performance in the network by simple certification path building. The performance data is from the random tree walk with no build operation.

first when building in the forward direction. In some circumstances, the PathBuilder fails to retrieve more useful information from CA certificates and then switches to a cross-certificate pair. In particular, it looks for the issuedToThisCA element. Thus, to be able to go forward one step in the tree walk, the PathBuilder may need to perform two LDAP requests. Nonetheless, the forward direction can help build certification paths faster and requires less data transfer.

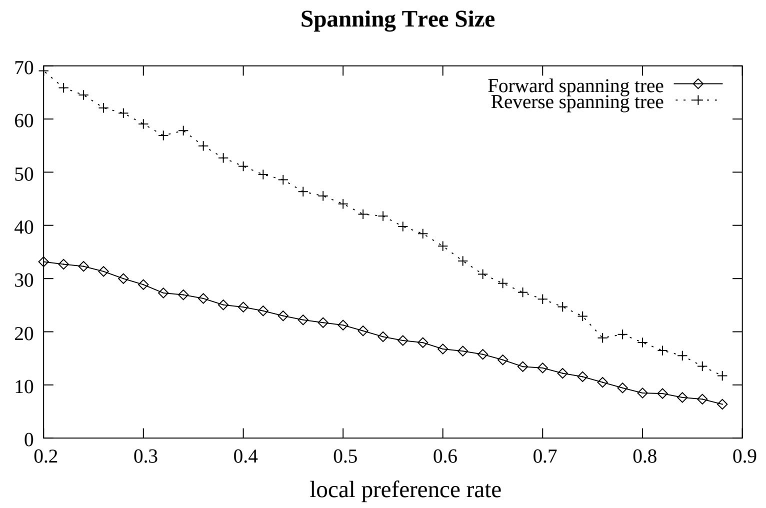
Since the reverse search tree is relatively flat, we will further explore build options that may help speed up tree walk on a reverse search tree.

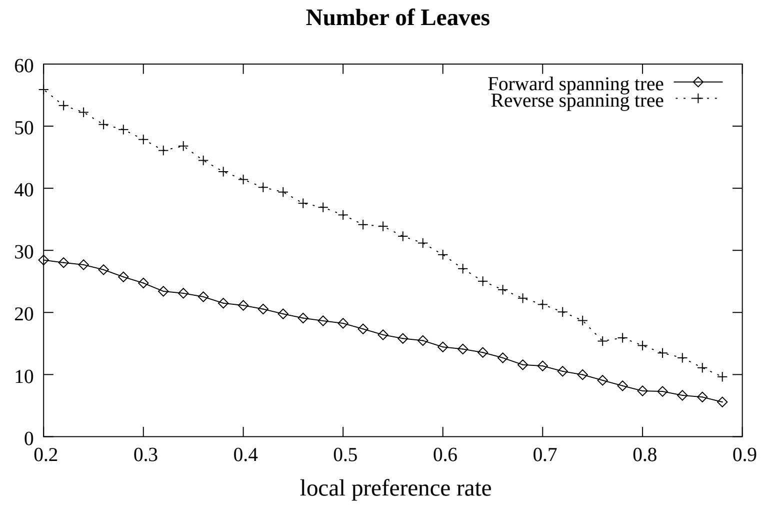
10.3.2 Local Preference

Now, we vary the local preference in the experiments to examine the performance impact by user behaviors. We explore local preference in the range of 0.2 to 0.9. Typically, building certification paths for a local target is much faster than building for targets in remote PKI systems. In an extreme case, if the local PKI domain has only one CA, the issuer of the target certificate is the trust anchor and the entire building process is not necessary. In the experimental certificate topology, many PKI domains are in this situation.

Figures 10.5 and Figure 10.6 illustrate the performance results for selected tree properties and for network operations. All resulting search trees have the similar shape as we have discussed above—reverse search trees are relatively flat. All the performance overheads decrease linearly as the preference rate increases. The reverse search tree demonstrates faster overhead decrease than the forward search tree, which indicates that building in the reverse direction benefits more from larger local preference rate.

Based on most of the measurement metrics, the forward direction still out-performs the reverse direction. We notice that building in either direction requires approximately the same number of LDAP requests. Consequently, the slightly longer building time delay is decided by the amount of data transmitted over the network. Again, there are three types of LDAP requests produced by constructing a certification path in the forward direction: the successful CA certificate retrieval, the failed CA certificate retrieval, and the successful cross-certificate pair retrieval. The reverse direction only needs cross-certificate pair retrievals. The LDAP retrieval requests for CA certificates are relatively lightweight. The responses carry less data, which need less transmission time. Such differences explain why the same number of LDAP requests by the reverse direction causes slightly longer network delay and more transmitted data. On average, the forward direction requires about 16% to 24% less data transmission.





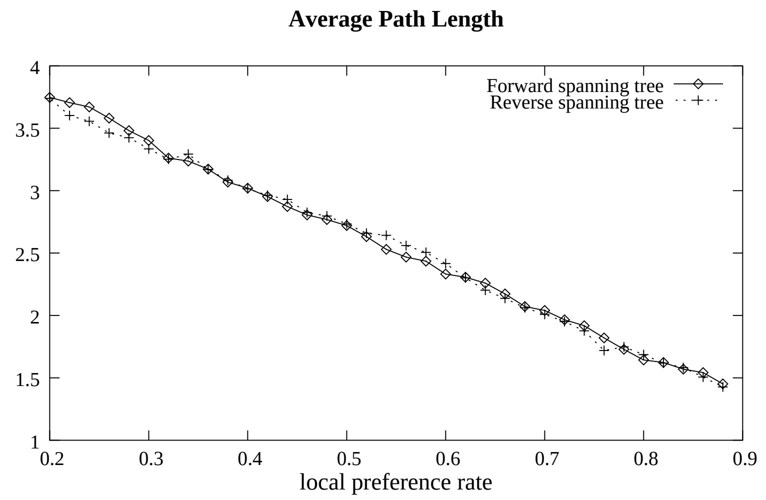
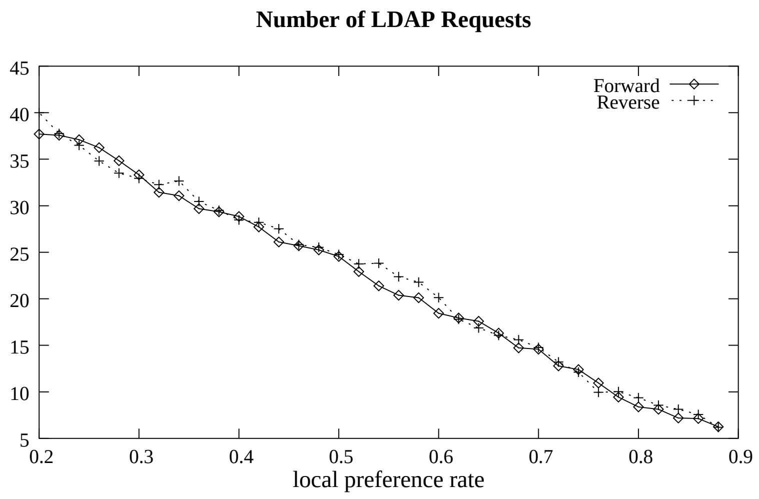
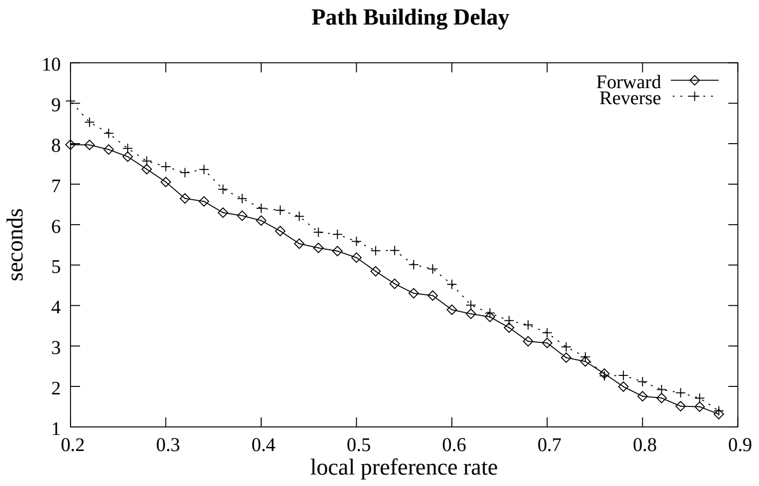


Figure 10.5: The selected properties for search trees given various local preference rates. Larger preference rate helps reduce tree size.





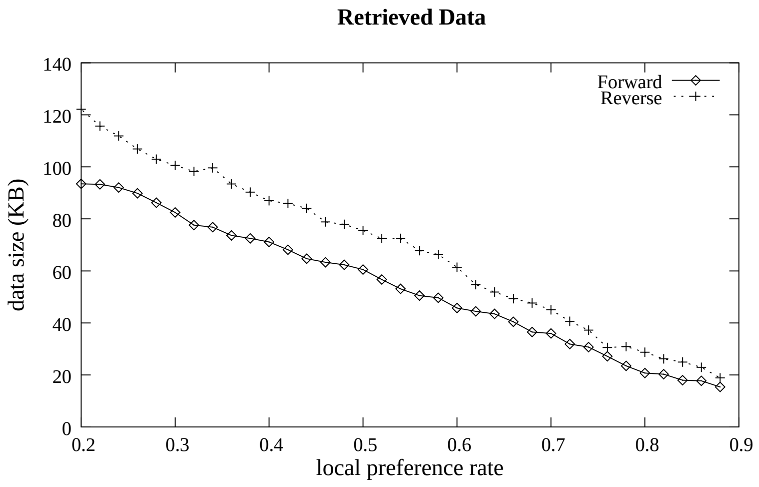


Figure 10.6: Network performance by each certification path building process. Large preference rates can significantly improve the efficiency.

10.3.3 RDN Matching Option

So far, we have examined the certification path building performance without any build option. We can see that the forward direction is generally more efficient than the reverse direction. However, there are still chances that we may be able to improve the performance when building paths in the reverse direction. As experiments have shown, reverse search trees are relatively flat. If a build option can help speed up the tree walk procedure, we may be able to achieve faster certification path construction.

We examine the RDN matching option. Previous experiments assumed that the RDN match value of any two certificates is zero. Now, we have possible positive match values. Let's first compare the performance of complete random tree walk with the performance of the RDN matching option that uses a DN-configured certificate topology. Since not all CAs have configured DNs, we simply assume that the match value is zero if there is no DN for either the issuer or the subject.

Figure 10.7 compares the number of LDAP requests and number of cross certificate pairs retrieved from LDAP servers. The RDN matching option does not have a noticeable impact on path building in the forward direction, which matches our expectation. The forward direction faces most of its choices when exploring in the mesh area of the certificate topology. In configuring RDNs in this area, the RDN matching values between bridge CAs are mostly very low. Thus, this optimization does not help the PathBuilder to narrow down choices. Furthermore, this result matches the real-world situation. Typically the bridge CAs belong to completely different organizations and even countries. Their DNs have the fewest elements in common. The results in Figure 10.8 present the consistent comparison. The RDN matching option does not help to improve the network performance by tree walk on forward search trees.

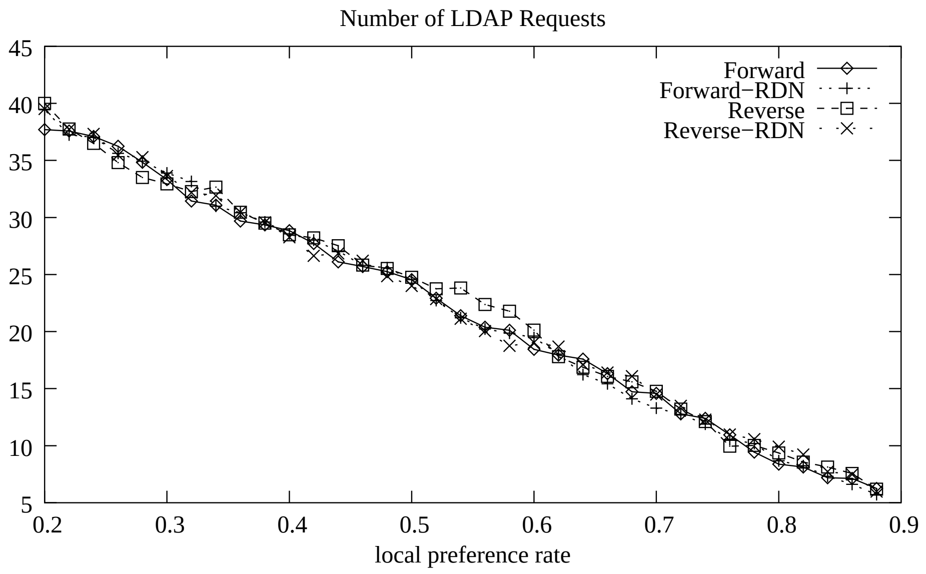
Interestingly, we observe that RDN matching option does help in reducing the number of retrieved cross certificate pairs for the reverse direction. The average number of such pairs by each LDAP request has dropped from 3.8 to 3.4, with approximately the same number of total LDAP requests. This 11% improvement indicates that the RDN matching option can help the PathBuilder to avoid some CAs with a large number of branches.

However, as we see in Figure 10.8, the comparison of the amount of data transmission does not show a similar improvement. The results by a random tree walk and by tree walk with the RDN matching option are fairly close. We believe that this inconsistency comes from the actual contents in each retrieved certificate pair. For the tree walk with RDN matching, the PathBuilder requests fewer cross certificate pairs that only have the issuedByThisCA element. In other words, RDN matching helps the PathBuilder spend less time on exploring hierarchies from top to bottom.

In short, 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.

10.4 Observations

This analysis of the RDN matching option demonstrates how we use the operational model of PKI and the PathBuilder certification path building model to understand performance



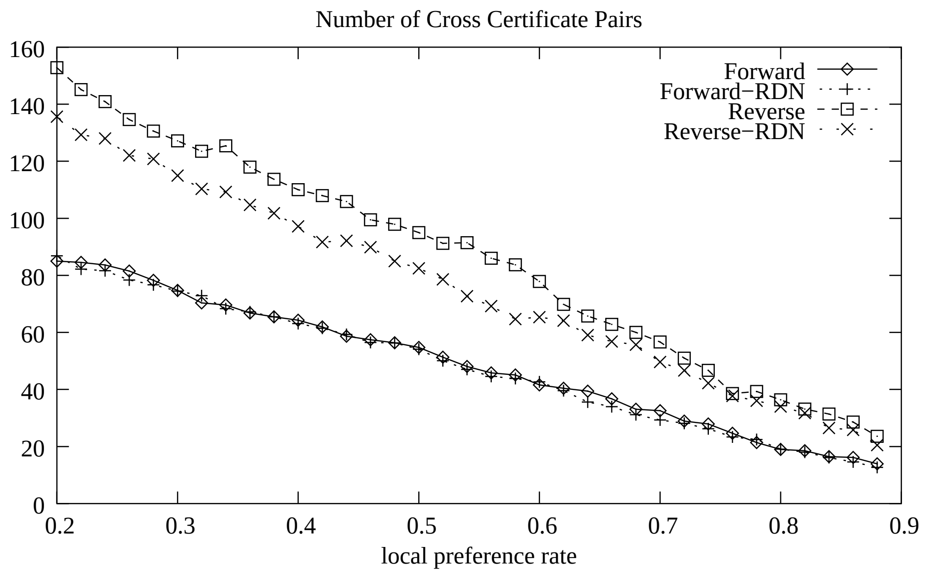


Figure 10.7: Tree walks in forward and reverse search tree. “Forward” and “Reverse” denote tree walks without optimization. “*-RDN” denotes the performance by RDN matching optimization.

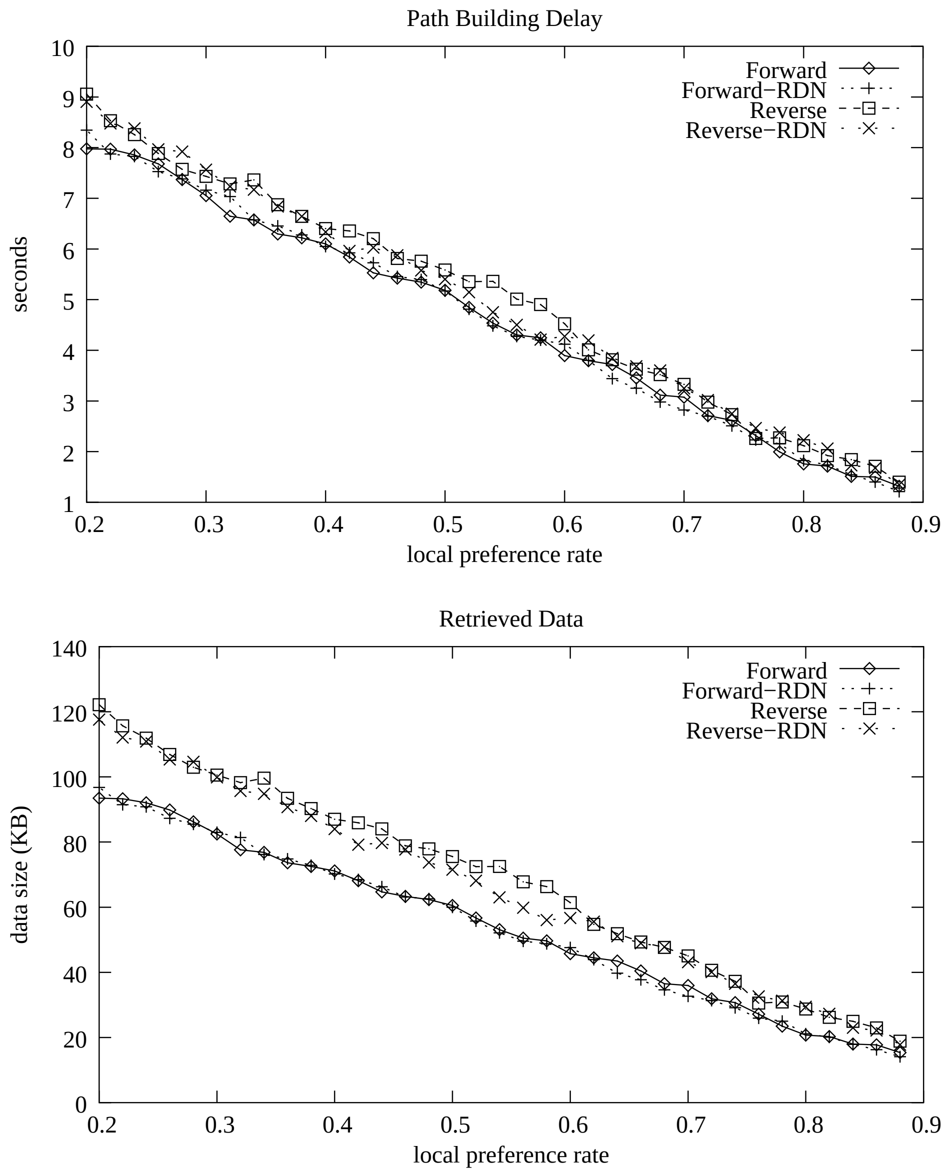


Figure 10.8: Network performance of tree walks in forward and reverse search tree. "Forward" and "Reverse" denote tree walks without optimization. "*-RDN" denotes the performance by the RDN matching optimization.

impact. We have just scratched the surface in this field. Yet, we have made some interesting observations from the experiments.

Observation 1: The performance difference by building directions heavily depends on the architecture of certificate topology. In our experiments, we use a close-to-reality configuration for experimental certificate topology containing a number of local hierarchical PKIs and a few bridge CAs that interconnect them. This architecture favors the forward direction.

The performance of tree walks on a forward search tree is close to optimal, with just a few choices to make in the "mesh" area of the bridge CAs. On the other hand, reverse search trees require a lot more overhead on tree walks. Our experiments have shown that the forward direction out-performs the reverse direction on almost all measurement metrics consistently.

Observation 2: Users' behavior on using certificates significantly affects the resulting certification path building performance. We have used a simple criterion to model the user behavior: the choice of target certificate. Experiments have shown that building certification paths for local target certificates is much more efficient. More investigation is needed on understanding how users use certificates in the real world.

Observation 3: A building optimization as simple as the RDN matching option can help improve performance if building in the reverse direction. Even though there are about the same number of LDAP requests, this option slightly reduces the network delay and the amount of data transmitted over the network. Most of the savings come from the reduced number of cross certificate pairs retrieved from local hierarchical PKIs.

10.5 Implementation Suggestions

Our modeling work helped us obtain a deeper understanding of the certification path discovery algorithm. We have discovered a few approaches that can improve the performance most effectively. We consider them as implementation suggestions. We were not able to explore all of them in our experiments, which certainly leads to future work.

10.5.1 Forward, forward, forward

As we have seen, traversing a search tree from a leaf to the root is much more efficient than doing it in the opposite direction. To avoid inefficient tree walks, the algorithm should use the forward direction as much as possible. Furthermore, relying parties should set their trust anchors as the root CA of the local PKI domain. If such a setting is not available, relying parties should at least try to set their trust anchors close to the edge of the local domain.

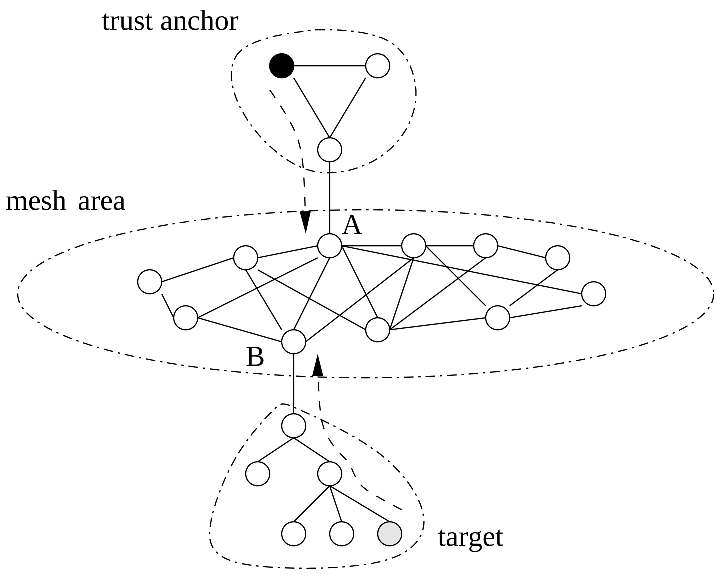


Figure 10.9: Example of the "meet-in-the-middle" approach for certification path discovery. Two example PKI domains cross-certify with their bridge CA in the mesh area.

10.5.2 Meet in the Middle

Our experiment has shown that the forward direction is favorable. The resulting performance is close to optimal. The remaining problem is the mesh area, where the PathBuilder still needs to make decisions among a set of certificates. Therefore, we need an approach to speed up the process when PathBuilder is walking in the mesh area. We suggest that, if possible, the algorithm should try to build certification paths in both forward and reverse directions and meet in the middle. Figure 10.9 demonstrates the benefit of the "meet-in-the-middle" approach using an example. Suppose Alice in the top PKI domain tries to construct a certification path for the target in the bottom domain. Building in both directions, she can quickly enter the mesh area in the middle. Typically, the connectivity in the mesh area is high, but the paths are fairly short, within one or two steps. Instead of trying each branch one by one when building in a single direction, Alice can simply compare the neighbors of bridge CA `A` and bridge CA `B` to discovers the direct link between them quickly.

There are several approaches for Alice to judge that she has crossed the boundary of a PKI domain. For instance, the sudden change of similarity in DNs indicates the cross domain action. Or Alice can look for self-issued certificates. Typically, only the root of a hierarchical PKI has a self-issued certificate. Once reaching such certificate, Alice knows that she is about to leave the domain. There could be other ways too. In general Alice has fair good sense on when she crosses the boundary, and when she should stop to wait for a potential meeting point.

10.5.3 Caching

It is a common practice that the certification path building algorithm exploit caching to avoid re-retrieving certificates from remote LDAP servers. Our experiments show that without this caching optimization, the number of LDAP requests can be as high as 40 in order to build one path. Such a process costs 8-9 seconds in total. Most of the relying parties would not tolerate such long latency. Therefore, we consider certificate caching as a must for PathBuilder.

There are two problems related to caching certificates. First, the scale of the certificate topology may be very large. Thus, it is a high cost for each relying party to maintain their own local cache. Second due to certificate reissuing and revocation, the information in the local cache may be stale. Therefore, the relying parties need to find a trade-off. At some point, they should try to go to the remote LDAP severs to fetch fresh copies of certificates.

There is a simple solution to these two problems—using a dedicated server. One server can serve the entire PKI domain. In fact, PKI architects have worked on setting up such dedicated servers. CoreStreet designed and developed a certificate validation service [79]. The design is very similar to our PathBuilder model. On a dedicated server, the algorithm uses the local database of certificates to build the search trees for PKI systems of US government agencies (PKI domains number 4 to 11 in Figure 10.4). The certificate database on the server is large enough to store all necessary certificates, and it is properly maintained to store up-to-date information. The clients need to spend time on only one request to the server to gain information on the resulting certification path. DoD is planning to use this validation service in the near future. We suggest that PKI developers implement a similar server to serve relying parties in the local domain.

10.5.4 Other Optimizations

Besides the RDN matching option, there are many other possible optimizations, to which our model can extend We certainly will extend our model to explore more optimizations and quantify their performance overheads. Elley et al. [39] suggested that name constraints and policy processing are two important optimizations that could potentially help improve performance, especially when the algorithm is building a certification path in the reverse direction. These optimizations may be effective at reducing a number of branches to only a few choices. Our simulation framework is able to handle the addition of new models for these options.

Chapter 11

PKI Performance Impact on BGP Security

In this thesis research, we have used simulation models to explore BGP security protocols and PKI systems. In particular, we use simulation to study the performance of a number protocols that use public-key cryptography to provide strong security to BGP route announcements. Previous experiments (by us and by everyone else) assumed that BGP speakers have established the trustworthiness of all related public key certificates before the simulation starts. However, in practice, there are many complicated issues related to validating a certificate in a PKI system. Now we revisit the certificate validation problem for BGP security and use simulation to understand its performance impact on BGP security and on BGP behavior.

As discussed in Chapter 4, S-BGP designs two hierarchical PKIs to support route authentication. The first hierarchy is mainly for authenticating AS number and BGP speaker assignments. The validity of AS number certificates and router certificates affects the overall security of S-BGP path authentication. Validating these certificates in order to verify route attestations introduces additional performance overhead. The second hierarchy is for IP address allocation, which is directly connected to the security of S-BGP address attestations. Similarly, additional performance overhead is involved. In this chapter, we discuss our simulation model of evaluating performance of validating related certificates for S-BGP path authentication and origin authentication.

11.1 Simulation Model

We use our simulation model to study the performance of PKIs for S-BGP. The major function of PKI in this context is certificate validation. In particular, we simplify the certificate validation into checking certificate status. As long as the involved certificates are not revoked, the BGP speakers can use the corresponding public keys to verify the attestations in Update messages.

BGP Speaker Behavior

We focus on measuring additional latency due to checking certificate status during normal BGP activities. The measurement metric for experiments is convergence time.

Thus, we model the BGP speakers to handle certificates in real time. That is, once the BGP speaker needs to verify a route announcement with attestations, it identifies the necessary public key certificates and checks their status one by one. The model has two assumptions. First, BGP speakers have their own local database which contain public key certificates for all possible IP address allocations, AS numbers, and BGP speakers. Second, BGP speakers also have databases to hold all relevant address attestations needed for origin authentication.

Certificate Validity Mechanisms

The simulation model examines two standard mechanisms for providing certificate status information: CRLs and OCSP.

In relying on CRLs, BGP speakers check the certificate status with fresh copies of the relevant CRLs. The model assumes that a BGP speaker may not have all necessary fresh copies. Thus, for any missing CRLs, the speaker contacts the corresponding LDAP servers to download CRLs and stores them in the local database. The model also assumes that newly downloaded CRLs will not expire during the experiment. In other words, once installed in the database, they are within the validity period at least for the next few minutes.

A speaker that relies on OCSP responses has two choices of how to handle OCSP requests if there are multiple certificates it needs to validate. A straightforward approach is to let the speaker send OCSP requests one at a time and release the next request once receiving the response for the previous one. As the requests go to different OCSP responders over the network, it can try to send out requests in parallel. The model compares performance for these two approaches. They are denoted as *sequential OCSP* and *parallel OCSP*, respectively.

We benchmark the latencies introduced by downloading CRLs or requesting OCSP responses as 0.5–1.0 seconds, most of which is from network latency.

AS-level PKI topology

In addition to the AS-level network topology we have in the model, the PKI structure attached to it is necessary. The following is the list of assumptions for the modeled PKI structure:

- There are 110 PKI domains, each specifying one autonomous system.
- Each PKI has one CA, one LDAP server, and an online OCSP responder.
- BGP speakers directly trust the public keys of all CAs and OCSP responders.

With the last assumption, we minimize the workload of constructing and validating certification paths. Now validating a certificate mainly involves the work of checking certificate status information.

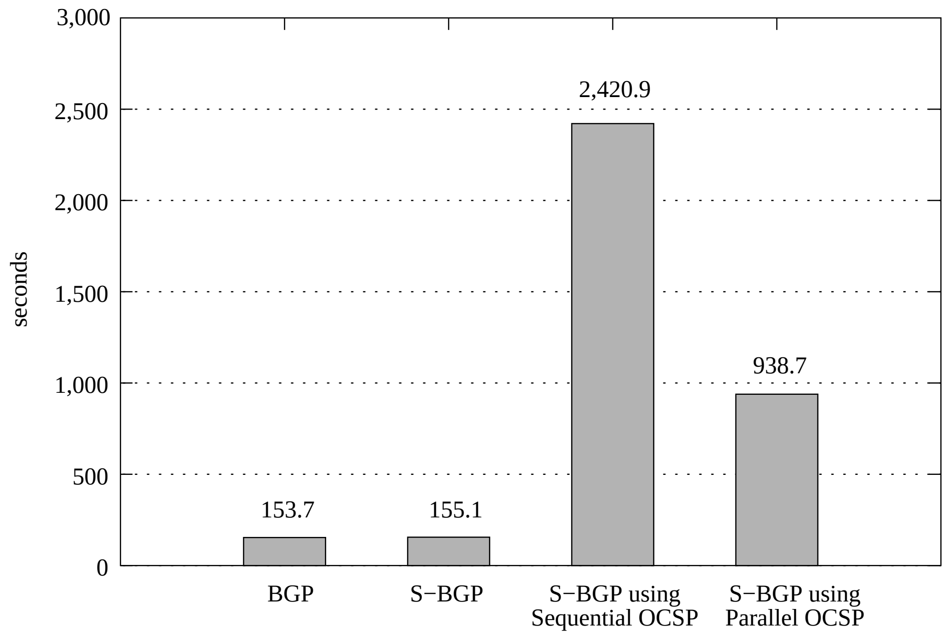


Figure 11.1: Convergence time of S-BGP with origin authentication, using OCSP for certificate validity.

11.2 PKI for Origin Authentication

We first simulate the PKI processing latency for origin authentication. In modeling OA, we have designed a PKI hierarchy model for IP address allocation. We use the same model here, except the purpose is for measuring PKI performance. Each prefix in the model is associated with a chain of IP address allocation certificates. The speaker validates each certificate to validate the address attestation.

OCSP

During router rebooting, there are about 67,000 to 69,000 OCSP requests sent out by BGP speakers in total, which introduces significant overhead that affects convergence time. Figure 11.1 compares the increased convergence time for S-BGP with and without PKI overhead. S-BGP origin authentication is very efficient since only signature verifications are involved in validating address attestations. On the contrary, convergence times with OCSP requests are unacceptably long. Although OCSP offers real-time certificate status, its performance make it infeasible in practice.

CRLs

Figure 11.2 presents the increased convergence time with the increased percent of missing local copies of CRLs. If the BGP speakers do not have any CRLs to start with, the convergence time is still manageable. The relative increase is at most 34.4%.

Clearly, for BGP speakers, CRLs are more efficient in terms of handling certificate status.

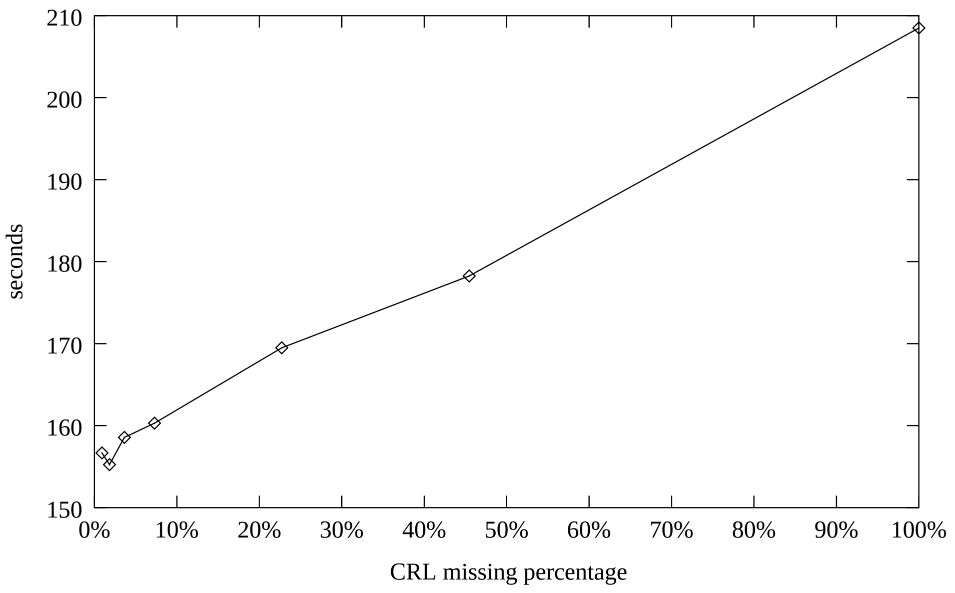


Figure 11.2: Convergence time of S-BGP origin authentication, with support of CRLs for certificate validity.

At the same time, because CRLs are updated periodically, BGP speakers should tolerate the possibility of missing some revoked certificates. Thus, OCSP and CRLs demonstrate trade-offs between security and efficiency.

11.3 Path Authentication

The PKI performance impact on path authentication is similar to the impact on origin authentication. Here, BGP speakers handle certificates of AS numbers and BGP speakers. One AS number certificate and one router certificate serve for the validity for each route attestation. Thus, for an AS path of length k, the receiver speaker should validate $2k$ certificates. In total, the experimental AS-level topology is configured with 220 related certificates.

OCSP

As for origin authentication, use of OCSP for path authentication dramatically increases convergence time. Parallel requests are much more efficient than sequential requests. Yet, they still prolong convergence by 53.3%.

CRLs

To download CRLs, BGP speakers need to contact at most 110 organizations. For one route attestation, the speaker uses one CRL to validate the status of the corresponding AS number certificate and router certificate. The performance is again similar to origin authentication.

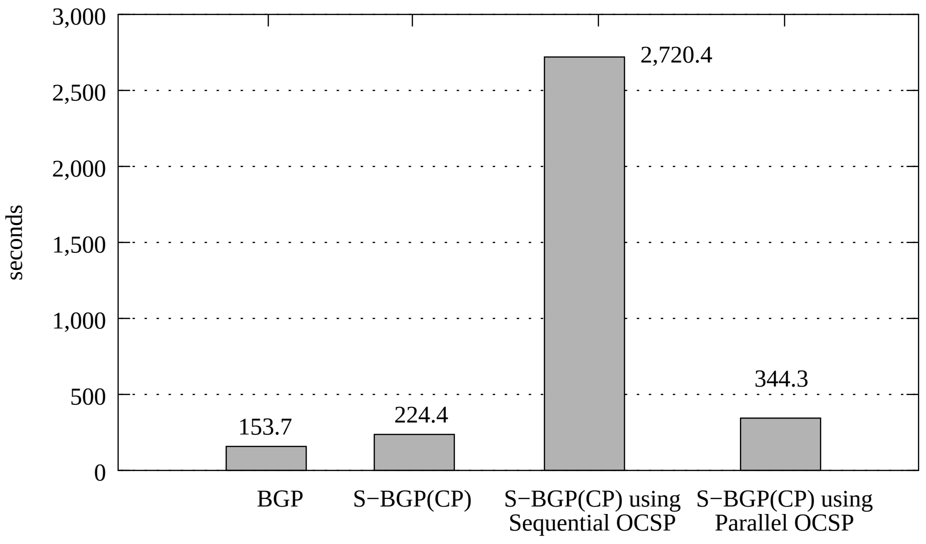


Figure 11.3: Convergence time of S-BGP path authentication, with support of OCSP for certificate validity. The comparison is based on S-BGP with all optimizations.

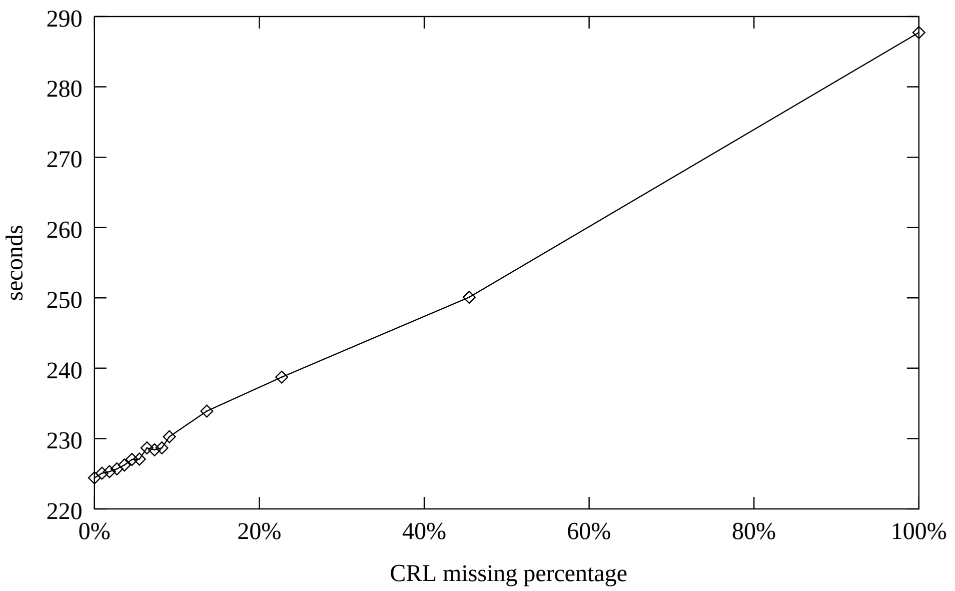


Figure 11.4: Convergence time of S-BGP path authentication, with support of CRLs for certificate validity.

Downloading CRLs is much more efficient than OCSP requests. Figure 11.4 presents the experiment results. CRL downloading increases the convergence time by at most 28%.

11.4 Space Issue

In addition to processing latency and convergence time, space cost on BGP speakers is an important concern. Typically, certificates are relatively large data structures. Carrying certificates in Update messages can easily exceed the message MTU, 4096 bytes. In fact, BGP security protocols have proposed various ways of distributing certificates to BGP speakers. In S-BGP, network administrators can push data to BGP speakers' local database periodically. soBGP proposed to define a new type of BGP message—SECURE message—to be used only for transmitting certificates. BGP speakers may have local database to store all related certificates.

Our simulation shows that certificates consume about 6KB storage on each BGP speaker, on average. The relatively small scale of the simulated network prevents us from directly inferring potential storage issues in the real world. IP address allocations, AS number assignments, and router assignments for the Internet produce many more certificates. The CIDR BGP report from AS1221 (Telstra) [24] shows that there are 181,031 active BGP entries in a routing table. To validate ownerships of these prefixes, we need roughly the same number of address allocation certificates. Besides, this report also concludes that there are about 18,233 unique ASes and 50,000 organizations. Considering both PKIs by S-BGP, each BGP speaker needs about 190MB in total to store all certificates, assuming certificate length is about 2KB on average.

Chapter 12

Conclusions and Future Research

12.1 Conclusions

During the course of our research we have used simulation intensively to study performance of large-scale security systems. Secure inter-domain routing and the certification path discovery algorithm in public key infrastructures are two primary objects in our study. With detailed quantitative analysis of protocols and their options, we discovered several ways to improve their efficiency. Solutions range from protocol implementation changes to simple configuration changes.

12.1.1 Efficient BGP Security

We set up a framework using network simulation to evaluate the costs of S-BGP. The more we looked at the performance, the more new potential approaches we discovered to improve performance [126, 166, 168].

S-BGP is inefficient S-BGP, one of the primary security proposals for BGP, introduces performance overheads on timing and space metrics. Optimizations do speed up the convergence time, but not enough. Moreover, S-BGP optimizations have side-effects. DSA pre-computation requires secure storage on each BGP speaker. Caching creates serious space problems. Too high space cost is a crucial issue for S-BGP real-world deployment.

Fast signature verification for path authentication Simulation results showed that verifications contribute a majority of signature operations. Efficient verifications naturally speed up the path authentication, as shown by faster convergence time. All of our proposed efficient path authentication schemes use signature algorithms that verify signatures much faster than DSA used by S-BGP.

Reducing signing operations Despite their relatively small number, the efficiency of signing operations is also a factor that affects convergence time. We applied RSA directly to S-BGP route attestations and evaluated the performance impacts. The experimental results confirmed that fast verifications alone do not help speed up convergence. Expensive signing operations dramatically prolong convergence.

Consequently, we designed techniques to reduce the number of expensive signing operations. We revisited the signing process carefully, and found that we can let the speakers sign the same announcement only once, rather than once for each recipient. In doing so, the speakers indicates a set of recipients of the route. We found that bit vectors are very effective at implementing this signature reduction. We further reduce the signing operations by signing all messages in the output buffers only once, rather than once for each message. The Merkle hash tree is one way we found that can realize this signature reduction.

We designed a Signature Amortization (S-A) scheme that combines bit vector and hash tree techniques to save as much as 98% of the total signing operations by S-BGP. Convergence time by S-A is just a few seconds longer than normal BGP protocol without any security system in operation. Furthermore, S-A does not require local memory space for caching routes. However, S-A generates even longer Update messages.

Aggregate signature to save space We identified that the primary cause of the S-BGP space problem is the requirement that an AS path of length k needs k signatures. We applied aggregate signature algorithms to produce one signature for the entire AS path. Simulation experiments with sequential aggregate signatures showed that the new signature algorithm immediately reduces the Update message size by 42%.

Aggregated path authentication for efficiency in time and space Finally, our exploration of time efficient and space efficient techniques for S-BGP led us to a set of Aggregated Path Authentication schemes. APA schemes combine the efforts of S-A and aggregate signature schemes to reduce both computational overhead and space costs. Choosing various options from each technique, we tried out six different constructions for APA—GAS-V(SW), GAS-T(SW), GAS-V(HW), GAS-T(HW), SAS-V, and SAS-T. Simulation experiments have shown their dramatic performance improvements compared to S-BGP: GAS-V(HW) has minimum impact on BGP convergence and can substantially reduce 66% of the message size and $60 - 70\%$ of the memory cost.

Avoid OCSP for certificate validity Many of the recently proposed BGP security protocols rely on secure and efficient PKI systems. We used simulation to evaluate the standard certificate revocation mechanisms that support these PKI systems. Experiments showed that OCSP introduces so much computational overhead that it is not workable to use it to support certificate validity service for BGP security. Compared with OCSP, CRLs are more acceptable. Our work is the first effort that quantitatively evaluates the performance of certificate validity service for BGP security.

12.1.2 Efficient Certification Path Discovery

Using SSFNet, we built a simulation framework to study general purpose large-scale PKI systems. The framework contains an operational model of PKI activities in the network environment. We used this framework to model the certification path discovery algorithm and quantified its performance. We found that there are a few simple building options we can use the speed up the certification path construction process.

Use forward direction Our study on current PKI system deployment showed that the hierarchical PKI is a very common PKI architecture design for organizations with various sizes. Constructing certification paths in the forward direction is much more efficient than the reverse direction.

Use "Meet-in-the-middle" option Besides hierarchical architecture in local domains, more and more PKI systems cross-certify with each other via bridge technology. Current and future PKI deployments tend to construct an emerging global PKI system with a cloud of bridge CAs in the center and many hierarchical PKI systems on the edge. Such an architecture suggests that constructing certification paths from both ends via edge area and meeting in the central area is the most efficient approach. We suggest that the certification path building algorithm pause when it enters the bridge CA cloud while building the path in the forward direction, then try to construct the other half of the path in the reverse direction, and quickly meet in the middle.

Use reasonably good distinguished names Simulation experiments evaluated the effectiveness of the RDN matching option for the certification path building algorithm. This option can help reduce the amount of data retrieved from repositories, if building in the reverse direction. Our experiments suggest that PKI architects should design well-matched distinguished names for local CAs and end entities.

User behaviors affect performance We modeled limited types of user behavior patterns. A simple experiment on certificate local preference clearly showed a performance difference for different local preferences. If users tend to validate more local certificates, the algorithm performs much more efficiently.

12.2 Summary of Contributions

Our research has made significant contributions for the security and network community. Now, we summarize our accomplishments.

Implementation of a simulation framework for BGP security protocols

Our simulation of BGP security protocols is by far the most thorough use to-date of simulation to evaluate the performance of BGP security protocols. The simulation is based on the BGP implementation in SSFNet, the first and only detailed BGP simulator. We extend the simulator to model memory overhead on routers. The simulator provides a flexible interface that allows the simulation to model a broad range of protocols. It provides a scalable framework for performance evaluation of any BGP security protocol. The simulator also extends the existing integrated monitoring and debugging system for reporting on aspects of the security protocols during execution.

Development of simulation models for BGP security protocols

We use the simulation framework to model several currently proposed security protocols that provide strong security to BGP, including several current primary proposals for origin authentication and path authentication. The models are compliant with the designs of these protocols. We also include corresponding additional features for performance optimization. We have modeled their computational overhead and space costs, and have produced concrete quantitative analysis of their performance impact on BGP routing behavior.

The simulation is able to model a variety of security related data structures, such as public key certificates, digital signatures, attestations, hash values, etc., as well as the operations that handle these data structures. Furthermore, the models explore different caching strategies of the security data using the memory model provided by the simulation framework.

Detailed performance evaluation of Secure BGP

With the models, we conducted detailed simulation experiments to understand the performance impact of S-BGP. We monitored and reported performance during router rebooting, a typical scenario that puts the routers under stress. The high load on routers helps disclose the maximum overhead of a security protocol. We not only evaluate the common signature processing, but also the underlying public key infrastructures. We compared the measurements with the plain BGP protocol without a security system in operation.

We are the first to report detailed performance analysis of S-BGP. We found that S-BGP is very expensive in terms of computational overhead and space costs. Furthermore, we used simulation to compare a number of unimplemented optimizations and options. In short, we conducted thorough performance study on S-BGP and obtained an in-depth understanding of its performance problems.

New protocols for efficient BGP path authentication

Based on our thorough understanding of S-BGP and its performance problems, we design Signature Amortization and Aggregated Path Authentication schemes for efficient path authentication. We combined time efficient techniques, such as bit vectors and Merkle hash trees, with space efficient techniques, aggregate signature schemes. These new efficient schemes dramatically improve performance without degrading security.

Implementation of the first detailed PKI network system simulator

Though several network simulators existed when this research began, none of them supported networked operations in a PKI system. We built a full implementation for protocols and environments that supports performance evaluation of PKI. We simulated a variety of related data structures, involved parties, and protocols. We implemented the certificate issuing, revocation, and validation services. The simulator supports a number of standard protocols, such as LDAP and CRLs. It also provides a flexible interface that allows researchers to add more simulated protocols and activities. Similar to the BGP simulator, our PKI simulator contains a fully integrated monitoring and debugging system for reporting on dozens of aspects of the protocol during simulation executions. To the best of our knowledge, it is the first detailed simulator for PKI systems in network environments.

Investigation of a large-scale PKI architecture

We configure a detailed PKI architecture that represents the current PKI system deployment across the world. The resulting certificate topology expresses the current major efforts on building bridge-to-bridge environment for PKI systems. It is a massive experimental PKI architecture, and contains more than 30 million users over 13 countries. To the best of our knowledge, it is the first systematic attempt to model the emerging global PKI architecture.

Development of probability model of certification path building

We design and develop a probability model of a certification path building algorithm. We simulate the behavior of constructing certification paths by a number of users as probability random walks on certificate search trees built upon certificate topologies. The search tree model is able to integrate path building options to assign probabilities on tree branches by different strategies. It also contains a monitoring and debugging system for reporting on algorithm behaviors during execution.

Exposition of simple methods for improving the performance of certification path building.

Previous studies have discussed the choice of building directions and made suggestions that the forward direction is most suitable for hierarchical PKI systems. We quantify the beneficial effects of the forward direction, and compare it with the reverse direction. We examine the RDN matching option for certification path construction. We show that good assignment of distinguished names helps improve algorithm performance if building in the reverse direction. Besides the options, we also show that building efficiency is affected by user behavior. Remote targets require much longer latency and more data transmission.

In addition to these results, we also make suggestions about certification path building options that may potentially improve algorithm efficiency. Certain options, such as meet-in-the-middle and caching, appear to be the most effective candidate options.

12.3 Suggestions for Future Research

Our research has made significant progress in evaluating large-scale distributed security protocols. Yet this is still a very wide-open field. There are several research directions to be explored to complement our efforts.

12.3.1 Improvements on BGP Security Simulation

Extending Current Simulation We used simulation to evaluate most functions defined for S-BGP, including address attestations, route attestations, and underlying PKI architectures. Yet, the simulation model has limitations. Next, we discuss further improvements of the current simulation model as future research.

- **More detailed topology** In our experimental 110-AS topology, we made many simplifying assumptions. For instance, we assumed that each AS has only one BGP speaker. One extension of the simulation is to drop this assumption. Introducing multiple speakers in an AS not only makes the model more realistic, but also allows us to model several important routing behaviors ignored by the single speaker topology. By assuming that each AS has only one BGP speaker, we eliminated the possible model for security on IBGP sessions. It is common that there are several speakers for one AS and the internal sessions need to be secured as well. Consequently, security on IBGP sessions also have a performance impact. Furthermore, with multiple speakers, we are able to model multi-homing, a common configuration of AS-level relationships, and to explore what a security protocol can do to authenticate such peering relationships.

- **BGP options** Another assumption of the simulation is that ASes use default routing policies and no additional options. In fact, the routing policies and BGP options, such as route aggregation, may affect the design of security protocols. Currently, most security protocols for BGP do not protect the routing policies, because it is a very difficult problem. It is a future direction to provide stronger BGP security by taking into account these additional issues. In the research of these new BGP security protocols, our simulation model can provide detailed performance report to help researchers to find efficient approaches.

- **PKI certificate dynamics** The current PKI model for S-BGP simply assumes that all certificates are static. That is, during execution no certificate is revoked or newly issued. In reality, the PKI system is dynamic. Changes may affect the BGP behavior. It would very interesting to extend the current model to examine certificate dynamics. Several approaches may be useful for this purpose. We can design the certificate issuing model based on real data. For instance, Wan et al. [159] reported that the number of ASes grew by an average of 190 per month during January to August, 2004, with an average of 347 ASes added and 157 ASes removed. IP address allocations are more complicated, which requires careful analysis of BGP routing tables. In modeling such dynamics, we need to extend the simulation to model certificate issuing and revocation. The goal is to model more realistic PKI systems.

- **Impact of certificate revocation** The latest S-BGP proposal [85] suggests that routes that had been accepted, but whose signing certificates have now expired or revoked, be considered withdrawn. Such a design could have massive impact on the Internet routing. A speaker can contribute to signatures on a large number of routes in a routing table. Withdrawing all of them has an effect similar to a router crashing. It would be very interesting to evaluate the benefit and overhead of this design.

Model More Security Protocols Currently, our model is focused mostly on S-BGP with various signature schemes. Beside signature-based solutions, there are a number of protocols that rely on symmetric cryptography [74], database management [38, 55, 159, 161], and system monitoring [55, 151]. To provide comprehensive performance evaluation of BGP security protocols, the simulation model certainly needs new modules to model new features other than signature processing and certificate processing.

Among these techniques, database management is particularly interesting. Besides S-BGP, soBGP is another primary proposal for securing BGP. Essentially, the design consists of adding various certificates and giving servers a database to distribute and validate these certificates. These servers use BGP speakers to help certificate distribution, for instance via the newly defined SECURE message. Although servers help reduce the memory burden on speakers, they introduce additional overhead on speakers to handle certificate distribution. In addition, the timeliness of certificate distribution directly decides the route security. To understand these issues, the simulation should add a database model that simulates the contents as well as the operations on the database.

12.3.2 Improvements on PKI System Simulation

Our research on modeling and simulation of PKI systems leaves a lot of space for future research. We have identified many potential extensions of the current model.

Extensions of the Search Tree Model Currently, we have modeled only the building directions and RDN matching option to demonstrate how to use the search tree model to evaluate the performance of a certification path building algorithm. There are many other options that may be beneficial to the algorithm's performance.

- **Exploring more build options** Name constraints and policy mapping are two popular options. Modeling them requires additional implementation of the probability assignment on tree branches. Moreover, measurement data are also necessary for model configurations. As discussed by Elley et al. [39] these options can help speed up the algorithm if building in the reverse direction. Simulation experiments with these new options can quantify their benefits in the experimental certificate topology.
- **Caching** Among all designed measurement metrics, the number of LDAP accesses is the most important. The algorithm processing overhead mostly comes from the network latencies by these requests and responses. Our experiments have indicated that caching can be a very important optimization. And it is common to have a local cache for path builders in practice. A new simulation model of local cache can quantify its performance improvements, space overhead, and other side effects.
- **Meet-in-the-middle** Our simulation experiment results led us to the hypothesis that meet-in-the-middle is the best choice for the algorithm. To verify this hypothesis, we need to modify the search tree model. Currently the building direction is expressed exclusively by a search tree. The new search tree should be able to consider both directions in the same tree or graph.
- **Revocation** Although not recommended for certification path building [28], we think that certificate revocation offers opportunities to examine interesting instances in the building algorithm. And certificate status information is needed anyway for validating a certification path. First, checking revocation information introduces additional

latencies and space costs. Second, not checking revocation information or relying on out-of-date information may causes construction of invalid certification paths. In this case, relying parties have to repeat their efforts to try to discover a valid path. A simulation that models these situations can help users evaluate the trade-offs between performance overhead and successful rate.

Extending the operational model The operational model of PKI provides the network environment we used to evaluate network performance of the certification path building algorithm. The purpose of developing the operational model is not only for the certification path building algorithm but also to implement an evaluation framework for general large-scale PKI systems with a flexible model interface. We can extend the current operational model in a number of ways.

- **More CSI mechanisms** Besides CRLs, there are a number of different approaches that provide certificate status information. CRL variants offer more efficient data structures. OCSP provides more timely information. For the purpose of performance comparison, these approaches should be included in our operational model.

- **Certificate validation protocols** The current model covers most of the services by a general PKI system, except the certificate validation service. Once certification paths are constructed, relying parties rely on a certification path validation algorithm to eventually validate the target certificate. There are complicated operations involved, such as verifying signatures, checking name constraints and policy mappings, and checking CSI for each certificate. Complete implementation of this algorithm helps the simulation to evaluate CSI systems, as well as some certification path building options.

PKI Simulation for Risk Analysis Although we built the PKI simulation model for performance study, it is actually powerful to help other studies on PKI systems. The essential purpose of PKI systems is to support information security. According to Blakely et al. [12], information security is in fact “information risk management”. In the future, we can use the current PKI simulation framework to build a systematic solution for risk analysis of large-scale PKI systems. The risk analysis framework may contain a currently modeled PKI services and network environment, as well as a new model for attack scenarios. Using such frameworks, we can evaluate potential risks encountered by large-scale PKI systems and provide guidelines for designing and deploying more secure, more robust, and more efficient PKI systems.

For the modeling purpose, we suggest two additions to the current model—the model of attack scenarios and a risk analysis system. The attack model may cover several adversary activities, such as inserting forged certificates, compromising keys, inserting false certificate revocation information, and carrying out denial-of-service attacks on servers. The risk analysis system may contain three phases—risk identification, risk quantification and measurement, and risk evaluation [67]. The most challenging part of risk analysis is to quantify the probabilities and magnitude of adverse effects and the consequences of attacks.

Above are just some suggestions. As risk analysis for information security is still a wide open area, we expect that our simulation framework can substantially help future research in this field.

Bibliography

[1] Martín Abadi. On SDSI's Linked Local Name Spaces. In *10th Computer Security Foundations Workshop (CSFW '97)*, pages 98–108, Rockport, MA, June 1997. IEEE Computer Society.

[2] Carlisle Adams and Mike Just. PKI: Ten Years Later. In *3rd Annual PKI R&D Workshop*, pages 69–84, April 2004.

[3] Carlisle Adams and Steve Lloyd. *Understanding PKI: Concepts, Standards, and Deployment Considerations.* Addison Wesley Professional, second edition, 2003.

[4] CERT Advisory. CA-2001-04 Unauthentic "Microsoft Corporation" Certificates. `http://www.cert.org/advisories/CA-2001-04.html`, March 2001.

[5] William Aiello, John Ioannidis, and Patrick McDaniel. Origin Authentication in Interdomain Routing. In *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS 2003)*, pages 165–178. ACM Press, October 2003.

[6] André Årnes, Mike Just, Svein J. Knapskog, Steve Lloyd, and Henk Meijer. Selecting Revocation Solutions for PKI. In *Proceedings of The Fifth Nordic Workshop on Secure IT Systems (NORDSEC 2000)*, October 2000.

[7] André Årnes, Mike Just, Steve Lloyd, and Henk Meijer. Certificate Revocation Performance Simulations. project paper, June 2000.

[8] Jerry Banks, John S. Carson II, Barry L. Nelson, and David M. Nicol. *Discrete-Event System Simulation.* Prentice-Hall, 3rd edition, 2000.

[9] A. Barbir, S. Murphy, and Y. Yang. Generic Threats to Routing Protocols. IETF Internet Draft `http://www.ietf.org/internet-drafts/draft-ietf-rpsec-routing-threats-07.txt`, October 2004.

[10] Paulo S. L. M. Barreto, Ben Lynn, and Michael Scott. Efficient Implementation of Pairing-Based Cryptosytems. *Journal of Cryptology*, 2004.

[11] Paulo S.L.M. Berreto. A note on efficient computation of cube roots in characteristic 3. Cryptology ePrint Archive, Report 2004/305. `http://eprint.iacr.org/2004/305`, 2004.

[12] Bob Blakley, Ellen McDermott, and Dan Geer. Information Security is Information Risk Management. In *New Security Paradigms Workshop*, Cloudcroft, New Mexico, September 2001.

[13] Dan Boneh, Xuhua Ding, and Gene Tsudik. Fine-Grained Control of Security Capabilities. *ACM Transactions on Internet Technology (TOIT)*, 4(1):60–82, February 2004.

[14] Dan Boneh, Xuhua Ding, Gene Tsudik, and Chi Ming Wong. A Method for Fast Revocation of Public Key Certificates and Security Capabilities. In *10th Usenix Security Symposium*, pages 297–308, 2001.

[15] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. A Survey of Two Signature Aggregation Techniques. *RSA CryptoBytes*, 6(2):1–10, 2003.

[16] Dan Boneh, Ben Lynn, and Hovav Shacham. Short Signatures from the Weil Pairing. In *Proceedings of Asiacrypt 2001*, number 2248 in LNCS, pages 514–532. Springer-Verlag, 2001.

[17] Dan Boneh, Ben Lynn, and Hovav Shacham. Short Signature from the Weil Pairing. *Journal of Cryptology*, 17(4):297–319, 2004.

[18] Brazilian Government PKI System. `http://www.icpbrasil.gov.br/`.

[19] Lee Breslau and Deborah Estrin. Design of Inter-Administrative Domain Routing Protocols. In *Proceedings of SIGCOMM 1990*, pages 231–241, September 1009.

[20] J. Callas, L. Donnerhacke, H. Finney, and R. Thayer. OpenPGP Message Format. RFC2440, `http://www.ietf.org/rfc/rfc2440.txt`, November 1998.

[21] CCITT. The Directory—Authentication Framework. Recommendation X.509, 1988.

[22] CertiPath: Enabling Trusted Communication. `www.certipath.com`.

[23] Chunhsiang Cheng, Ralph Riley, Srikanta P.R. Kumar, and J.J. Garcia-Luna-Aceves. A Loop-Free Extended Bellman-Fort Routing Protocol without Bouncing Effect. In *Proceedings of SIGCOMM 1989*, pages 224–236, September 1989.

[24] CIDR BGP Reports from AS1221 (Telstra), October 2004. `http://www.cidr-report.org/as1221/`.

[25] CIDR BGP Reports from AS1221 (Telstra), May 2005. `http://www.cidr-report.org/as1221/`.

[26] David A. Cooper. A Model of Certificate Revocation. In *15th Annual Computer Security Applications Conference (ACSAC'99)*, pages 256–264, Phoenix, Arizona, USA, December 1999. IEEE Computer Society.

[27] David A. Cooper. A More Efficient Use of Delta-CRLs. In *Proceedings of the 2000 IEEE Symposium on Security and Privacy*, pages 190–202, May 2000.

[28] M. Cooper, Y. Dzambasow, P. Hesse, S. Joseph, and R. Nicholas. Internet x.509 public key infrastructure: Certification path building. Internet Draft, draft-ietf-pkix-certpathbuild-04.txt, June 2004.

[29] James Cowie, David Nicol, and Andy Ogielski. Modeling the Global Internet. *IEEE Computing in Science and Engineering*, 1(1):42–50, Jan.–Feb. 1999.

[30] James Cowie, Andy Ogielski, Brian Premore, and Yougu Yuan. Global Routing Instabilities during Code Red II and Nimda Worm Propagation. Preliminary Report, `http://www.renesys.com/projects/bgp_instability`, September 2001.

[31] Certification Path Library (CPL). Cygnacom Solutions. `http://www.cygnacom.com/products/index.html#cpl`.

[32] Whitfield Diffie and Martin E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[33] Crls by cas in the department of defense pki. `ldap://crl.chamb.disa.mil`.

[34] Naganand Doraswamy and Dan Harkins. *IPsec : The New Security Standard for the Internet, Intranets, and Virtual Private Networks.* Prentice Hall, 1999.

[35] I. Duursma and H.-S. Lee. Tate pairing implementation for hyperelliptic curves $y^2 = x^p - x + d$. In *Advances in Cryptology - Asiacrypt 2003*, number 2894 in LNCS, pages 111–123. Springer-Verlag, 2003.

[36] B. Christian Ed. BGP Security Requirement. IETF Internet Draft `http://www.ietf.org/internet-drafts/draft-ietf-rpsec-bgpsecrec-01.txt`, February 2004.

[37] Brian Weis (editor). Secure Origin BGP (soBGP) Certificates. IETF Internet Draft `http://www.ietf.org/internet-drafts/draft-weis-sobgp-certificates-03.txt`, February 2005.

[38] Russ White (editor). Architecture and Deployment Considerations for Secure Origin BGP (soBGP). IETF Internet Draft `http://www.ietf.org/internet-drafts/draft-white-sobgparchitecture-01.txt`, February 2005.

[39] 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.

[40] C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen. SPKI Certificate Theory. RFC3280, `http://www.ietf.org/rfc/rfc2693.txt`, September 1999.

[41] Carl Ellison. SPKI Requirements. RFC2692, `http://www.ietf.org/rfc/rfc2692.txt`, September 1999.

[42] Carl Ellison and Bruce Schneier. Ten Risks of PKI: What You're not Being Told about Public Key Infrastructure. *Computer Science Journal*, XVI(1):1–7, 2000.

[43] EuroPKI Top Level Certification Authority. `http://www.europki.org/ca/root/en_index.html`.

[44] Michalis Faloutsos, Petrof Faloutsos, and Christos Faloutsos. On Power-Law Relationships of the Internet Topology. In *Proceedings of ACM SIGCOMM'99*, pages 251–262. ACM Press, 1999.

[45] S. Farrell and R. Housley. An Internet Attribute Certificate Profile for Authorization. RFC3281, `http://www.ietf.org/rfc/rfc3281.txt`, April 2002.

[46] Federal Bridge Certification Authority. `http://csrc.nist.gov/pki/fbca`.

[47] Anja Feldmann, Anna C. Gilbert, Polly Huang, and Walter Willinger. Dynamics of IP traffic: A study of the role of variability and the impact of control. In *Proceedings of ACM SIGCOMM'99*, pages 301–313. ACM Press, 1999.

[48] G. Frey and H. Ruck. A remark considering m-divisibility in the divisor class group of curves. *Mathematics of Computation*, 62:865–874, 1994.

[49] S. Galbraith, K. Harrison, and D. Soldera. Implementing the Tate pairing. In *Algorithm Number Theory Symposium - ANTS V*, volume 2369 of *LNCS*, pages 324–337. Springer-Verlag, 2002.

[50] Lixin Gao. On Inferring Autonomous System Relationships in the Internet. *IEEE/ACM Transactions on Networking (TON)*, 9(6):733–745, December 2001.

[51] Lixin Gao, Timothy G. Griffin, and Jennifer Rexford. Inherently Safe Backup Routing with BGP. In *IEEE INFOCOM 2001)*, April 2001.

[52] Lixin Gao and Jennifer Rexford. Stable Internet Routing Without Global Coordination. In *Proceedings of ACM SIGMETRICS 2000*, pages 307–317, June 2000.

[53] Simson Garfinkel. *PGP: Pretty Good Privacy.* O'Reilly, 1994.

[54] Irene Gassko, Peter S. Gemmell, and Philip MacKenzie. Efficient and Fresh Certification. In *Public Key Cryptography*, volume 1751 of *LNCS*, pages 342–353. Springer-Verlag, January 2000.

[55] Goeffrey Goodell, William Aiello, Timothy Griffin, John Ioannidis, Patrick McDaniel, and Aviel Rubin. Working around BGP: An Incremental Approach to Improving Security and Accuracy in Interdomain Routing. In *The 10th Annual Network and Distributed System Security Symposium*, San Diego, California, February 2003.

[56] Michael Goodrich. Efficient and Secure Network Routing Algorithms. provisional patent filing, `http://www.cs.jhu.edu/~goodrich/cgc/pubs/rout-ing.pdf`, January 2001.

[57] P. Grabher and D. Page. Hardware Acceleration of the Tate Pairing in Characteristic Three. In *Workshop on Cryptographic Hardware and Embedded Systems 2005 (CHES 2005)*, Edinburgh, Scotland, August 2005.

[58] R. Granger, D. Page, and M. Stam. On Small Characteristic Algebraic Tori in Pairing-Based Cryptography. Cryptology ePrint Archive, Report 2004/132. `http://eprint.iacr.org/2004/132`, 2004.

[59] Timothy G. Griffin and Brian Premore. An Experimental Analysis of BGP Convergence Time. In *The 9th International Conference on Network Protocols (ICNP'01)*, pages 53–61, November 2001.

[60] Timothy G. Griffin, F. Bruce Shepherd, and Gordon Wilfong. Policy Disputes in Path-Vector Protocols. In *Proceedings of the 7th International Conference on Network Protocols (ICNP'99)*, pages 21–30, Toronto, Canada, Oct–Nov 1999.

[61] Timothy G. Griffin and Gordon Wilfong. An Analysis of BGP Convergence Properties. In *Proceedings of SIGCOMM 1999*, pages 277–288, August 1999.

[62] Timothy G. Griffin and Gordon Wilfong. A Safe Path Vector Protocol. In *IEEE INFOCOM 2000)*, pages 490–499, March 2000.

[63] Vipul Gupta, Matthew Millard, Stephen Fung, Yu Zhu, Nils Gura, Hans Eberle, and Sheueling Chang Shantz. Sizzle: A Standards-based end-to-end Security Architecture for the Embedded Internet. In *Third IEEE International Conference on Pervasive Computing and Communications*, March 2005.

[64] Peter Gutmann. Everything you Never Wanted to Know about PKI but were Forced to Find Out. `http://www.cs.auckland.ac.nz/pgnut001/pubs/pkitutorial.pdf`, 2001.

[65] Peter Gutmann. PKI: It's Not Dead, Just Resting. *IEEE Computer*, 35(8):41–49, August 2002.

[66] Peter Gutmann. Re:VeriSign CRL single point failure. posting to `cryptography@metzdowd.com` mailing list, 10 January 2004.

[67] Yacov Y. Haimes. *Risk Modeling, Assessment, and Management.* John Wiley & Sons, Inc., 1998.

[68] Ralf C. Hauser, Tony Przygienda, and Gene Tsudik. Lowering security overhead in link state routing. *Computer Networks*, 31(8):885–894, 1999.

[69] Higher Education Bridge Certification Authority (HEBCA)—Transforming Education Through Information Technologies. `http://www.educause.edu/hebca/`.

[70] A. Heffernan. Protecting of BGP Sessions via the TCP MD5 signature option. RFC 2385, `http://www.ietf.org/rfc2385.txt`, 1998.

[71] 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.

[72] Russ Housley. S-BGP memory issues are the obstacle for real-world deployment. Personal communication, April 2005.

[73] Russ Housley and Tim Polk. *Planning for PKI*. John Wiley & Sons, Inc., 2001.

[74] Yih-Chun Hu, Adrian Perrig, and Marvin Sirbu. SPV: Secure Path Vector Routing for Securing BGP. In *Proceedings of SIGCOMM 2004*, pages 179–192. ACM Press, August 2004.

[75] IANA: Internet Assigned Numbers Authority. `http://www.iana.org`.

[76] IANA Internet Protocol v4 Address Space. `http://www.iana.org/assignments/ipv4-address-space`.

[77] 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.

[78] 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.

[79] CoreStreet Inc. Distributed Path Validation—Massive Scalability for Federated PKIs. Presentation st FBCA Path Discovery & Validation Working Group, August 2004.

[80] Internet corporation for assigned names and numbers. `http://www.icann.org`.

[81] CRL by POLCA in the Johnson & Johnson PKI. `http://jjedscrl.jnj.com/polca.crl`.

[82] Charlie Kaufman, Radia Perlman, and Mike Speciner. *Network Security—Private Communication in a Public World*. Prentice Hall, second edition, 2002.

[83] Stephen Kent, Charles Lynn, Joanne Mikkelson, and Karen Seo. Secure Border Gateway Protocol (S-BGP) – Real World Performance and Deployment Issues. In *The 7th Annual Network and Distributed System Security Symposium (NDSS'00)*, San Diego, California, February 2000.

[84] Stephen Kent, Charles Lynn, and Karen Seo. Secure Border Gateway Protocol. *IEEE Journal of Selected Areas in Communications*, 18(4):582–592, April 2000.

[85] Steve Kent. Securing the Border Gateway Protocol: A Status Update. In *Seventh IFIP TC-6 TC-11 Conference on Communications and Multimedia Security*, October 2003.

[86] T. Kerins, W. P. Marnane, E. M. Popovici, and P.S.L.M. Barreto. Efficient hardware for the Tate pairing calculation in characteristic three. In *Workshop on Cryptographic Hardware and Embedded Systems 2005 (CHES 2005)*, Edinburgh, Scotland, August 2005.

[87] Paul C. Kocher. On Certificate Revocation and Validation. In *Proceedings of the Second International Conference on Financial Cryptography (FC'98)*, volume 1465 of *LNCS*, pages 172–177. Springer-Verlag, 1998.

[88] Loren M. Kohnfelder. Toward a Practical Public-Key Cryptosystem. bachelor's thesis, Dept. Electrical Engineering, MIT, Cambridge, Mass., 1978.

[89] Brijesh Kumar. Integration of Security in Network Routing Protocols. *ACM SIGSAC Review*, 11(2):18–25, 1993.

[90] Craig Labovitz, Abha Ahuja, Abhijit Bose, and Farnam Jahanian. Delayed Internet Routing Convergence. In *Proceedings of SIGCOMM 2000*, pages 175–187, August 2000.

[91] Craig Labovitz, Abha Ahuja, Roger Wattenhofer, and Srinivasan Venkatachary. The Impact of Internet Policy and Topology on Delayed Routing Convergence. In *Proceedings of INFOCOM 2001*, pages 537–546, April 2001.

[92] Craig Labovitz, G. Robert Malan, and Farnam. Jahanian. Internet Routing Instability. In *The Proceedings of SIGCOMM 97*, pages 115–126, September 1997. Version also appeared in Transactions on Networking. SIGCOMM 1997 Best Student Paper Award.

[93] Craig Labovitz, G. Robert Malan, and Farnam Jahanian. Origins of Internet Routing Instability. In *IEEE INFOCOM 1999*, pages 218–226, New York, NY, June 1999.

[94] Averill M. Law and W. David Kelton. *Simulation Modeling and Analysis*. McGraw-Hill, Inc., 3rd edition, 2000.

[95] Ninghui Li and Joan Feigenbaum. Nonmonotonicity, User Interfaces, and Risk Assessment in Certificate Revocation (position paper). In *Proceedings of the 5th International Conference on Financial Cryptography (FC'01)*, volume 2339 of *LNCS*, pages 166–177. Springer-Verlag, 2001.

[96] Steve Lloyd. Understanding Certification Path Construction. PKI Forum White Paper, September 2002.

[97] CoreStreet Ltd. Distributed OCSP - Security, Scalability, and Availability for Certificate Validation. White paper. `http://www.corestreet.com/whitepapers/distributed-ocsp.pdf`, 2002.

[98] CoreStreet Ltd. Vulnerability Analysis of Certificate Validation Systems. White paper. `http://www.corestreet.com/whitepapers/w04_01v2_vulnerability_analysis_v%alidation_systems.pdf`, 2004.

[99] Anna Lysyanskaya, Silvio Micali, Leonid Reyzin, and Hovav Shacham. Sequential Aggregate Signatures from Trapdoor Permutations. In *Eurocrypt 2004*, volume 3027 of *LNCS*, pages 74–90. Springer-Verlag, 2004.

[100] G. Malkin. RIP Version 2. RFC2453, `http://www.ietf.org/rfc2453.txt`, November 1998.

[101] A. Malpani, R. Housley, and T. Freeman. Simple Certificate Validation Protocol (SCVP). Internet Draft, draft-ietf-pkix-scvp-14.txt, April 2004 (work in progress).

[102] Zhuoqing Morley Mao, Ramesh Govindan, George Varghese, and Randy H. Katz. Route Flap Damping Exacerbates Internet Routing Convergence. In *Proceedings of SIGCOMM 2002*, August 2002.

[103] John Marchesini and Sean Smith. Modeling Public Key Infrastructure in the Real World. In *EuroPKI 2005*, Springer-Verlag LNCS, to appear, July 2005.

[104] Ueli Maurer. Modelling a Public-Key Infrastructure. In *Proceedings of the Fourth European Symposium on Research in Computer Security (ESORICS'96)*, volume 1146 of *LNCS*, pages 325–350. Springer-Verlag, 1996.

[105] Patrick McDaniel and Sugih Jamin. Windowed Certificate Revocation. In *Proceedings of INFOCOM 2000*, pages 1406–1414, March 2000.

[106] Patrick McDaniel and Aviel Rubin. A Response to 'Can We Eliminate Certificate Revocation Lists?'. In *Proceedings of the Fourth International Conference on Financial Cryptography (FC'00)*, volume 1962 of *LNCS*, pages 245–258. Springer-Verlag, 2000.

[107] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*, chapter 11, page 451. CRC Press, 2001.

[108] R. Merkle. Protocols for Public Key Cryptosystems. In *Proc 1980 Symposium on Security and Privacy, IEEE Computer Society*, pages 122–133, April 1980.

[109] R. Merkle. A Certified Digital Signature. In G. Brassard, editor, *Advances in Cryptology – CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 218–238. Springer-Verlag, 1990.

[110] Silvio Micali. Efficient Certificate Revocation. Technical Memo MIT/LCS/TM-5426, March 1996.

[111] Silvio Micali. NOVOMODO: Scalable Certificate Validation and Simplified PKI Management. In *1st Annual PKI Research Workshop*, pages 15–25, April 2002.

[112] Victor S. Miller. Short Programs for Functions on Curves. `http://crypto.standford.edu/miller`, 2002.

[113] Victor S. Miller. The Weil Pairing, and Its Efficient Calculation. *Journal of Cryptology*, 17(4):235–261, September 2004.

[114] Multiprecision Integer and Rational Arithmetic C/C++ Library (MIRACL). Shamus Software Ltd. `http://indigo.ie/~mscott`.

[115] MIT PGP Public Key Server. `http://pgp.mit.edu`.

[116] Multi-threaded Routing Toolkit. `http://www.mrtd.net`.

[117] Ravi Mukkamamla and Mahantesh Halappanavar. ECPV: Efficient Certificate Path Validation in Public-key Infrastructure. In *Proceedings of 17th IFIP WG11.3 Working Conference on Database and Application Security*, Estes Park, Colorado, August 2003.

[118] Jennifer Joyce Mulligan. Detection and Recovery from the Oblivious Engineer Attack. Master's thesis, Massachusetts Institute of Technology, September 2002.

[119] Jose Muñoz and Jordi Forné. Evaluation of Certificate Revocation Policies: OCSP vs. Overissued CRL. In *DEXA Workshops 2002. Workshop on Trust and Privacy in Digital Business (TrustBus02)*, pages 511–515. IEEE Computer Society, September 2002.

[120] Jose Muñoz, Jordi Forné, Oscar Esparza, and Miguel Soriano. Implementation of an Efficient Authenticated Dictionary for Certificate Revocation. In *Eighth IEEE International Symposium on Computers and Communications (ISCC'2003)*, pages 238–243. IEEE Computer Society, June 2003.

[121] 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.

[122] S. Murphy. BGP Security Protections. IETF Internet Draft `http://www.ietf.org/internet-drafts/draft-murphy-bgp-protect-02.txt`, October 2002.

[123] Sandra Murphy. BGP Security Vulnerabilities Analysis. Internet-Draft `http://www.ietf.org/internet-drafts/draft-murphy-bgp-vuln-01.txt`, October 2004.

[124] 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.

[125] Moni Naor and Kobbi Nissim. Certificate Revocation and Certificate Update. *IEEE Journal on Selected Areas in Communications*, 18(4):561–570, April 2000.

[126] David M. Nicol, Sean W. Smith, and Meiyuan Zhao. Evaluation of Efficient Security for BGP Route Announcements using Parallel Simulation. *Simulation Practice and Theory Journal, special issue on Modeling and Simulation of Distributed Systems and Networks*, 12(3–4):187–216, July 2004. Preliminary version released as Dartmouth Technical Report TR2003-440R1, February 2003.

[127] 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.

[128] OpenPGP. `http://www.openpgp.org`.

[129] OpenSSL: The Open Source toolkit for SSL/TLS. `http://www.openssl.org`.

[130] D. Page and N. P. Smart. Hardware implemenation of Finite Fields of Characteristic Three. In *Workshop on Cryptographic Hardware and Embedded Systems 2002 (CHES 2002)*, number 2523 in LNCS, pages 529–539, Edinburgh, Scotland, August 2002. Springer-Verlag.

[131] Dan Pei, Xiaoliang Zhao, Lan Wang, Dan Massey, Allison Mankin, S. Felix Wu, and Lixia Zhang. Improving BGP Convergence through Consistency Assertions. In *Proceedings of INFOCOM 2002*, June 2002.

[132] R. Perlman. *Network Layer Protocol with Byzantine Robustness*. PhD thesis, MIT, October 1988. The MIT Press, LCS TR-429.

[133] Radia Perlman. An Overview of PKI Trust Model. *IEEE Network*, 13(6):38–43, Nov–Dec 1999.

[134] D. Pinkas and R. Housley. Delegated Path Validation and Delegated Path Discovery - Protocol Requirements. RFC3379, `http://www.ietf.org/rfc/rfc3379.txt`, September 2002.

[135] PKIF. Orion Security Solutions.

[136] PlanetLab—An open platform for developing, deploying, and accessing planetary-scale network services. `http://www.planet-lab.org`.

[137] BJ Premore. SSFNet BGP User's Guide. `http://www.ssfnet.org/bgp/user-guide=ps.zip`.

[138] Brian Premore. *An Analysis of Convergence Properties of the Border Gateway Protocol Using Discrete Event Simulation*. PhD thesis, Dartmouth College, June 2003.

[139] JJ. Puig, M. Achemlal, E. Jones, and D. McPherson. Generic Security Requirements for Routing Protocols. IETF Internet Draft `http://www.ietf.org/internet-drafts/draft-ietf-rpsec-generic-requirements-01.txt`, January 2005.

[140] Rana Barua Ratuna Dutta and Palash Sarkar. Pairing-Based Cryptographic Protocols: A Survey. Cryptology ePrint Archive, Report 064/2004. `http://eprint.iacr.org/2004/064`, 2004.

[141] Y. Rekhter and T. Li. A Border Gateway Protocol 4 (BGP-4). RFC1771, `http://www.ietf.org/rfc1771.txt`, March 1995.

[142] George Riley. Parallel/Distributed NS. `http://www.cc.gatech.edu/computing/compass/pdns/`.

[143] Ronald L. Rivest. SDSI - A Simple Distributed Security Infrastructure. `http://theory.lcs.mit.edu/~rivest/sdsi11.html`, October 1996.

[144] Ronald L. Rivest. Can We Eliminate Certificate Revocations Lists? In *Proceedings of the Second International Conference on Financial Cryptography (FC'98)*, volume 1465 of *LNCS*, pages 178–183. Springer-Verlag, 1998.

[145] The Route Views Project. `http://www.antc.uoregon.edu/route-views/`.

[146] SAFE Bridge Certification Authority TEST Environment. SAFE-BioPharma Association, `http://www.safe-biopharma.org/`.

[147] Bruce Schneier. *Applied Cryptography.* John Wiley & Sons, second edition, 1996.

[148] M. Scott and Paulo S.L.M. Barreto. Compressed Pairings. In *Advances in Cryptology - CRYPTO'2004*, number 3152 in LNCS, pages 140–156. Springer-Verlag, 2004. Updated version: Cryptology ePrint Archive, Report 2004/032. `http://eprint.iacr.org/2004/032`.

[149] Bradley Smith and J.J. Garcia-Luna-Aceves. Securing the Border Gateway Routing Protocol. In *Proceedings of Global Internet'96*, London, UK, November 1996.

[150] Bradley Smith, Shree Murthy, and J.J. Garcia-Luna-Aceves. Securing Distance Vector Routing Protocols. In *The 6th Annual Network and Distributed Systems Security Symposium (NDSS'97)*, San Diego, California, February 1997.

[151] Lakshminarayanan Subramanian, Volker Roth, Ion Stoica, Scott Shenker, and Randy H. Katz. Listen and Whisper: Security Mechanisms for BGP. In *Proceedings of First Symposium on Networked Systems Design and Implementation (NSDI 2004)*, March 2004.

[152] MitreTek Systems. Certificate Arbitrator Module. `http://cam.mitretek.org/cam/`.

[153] TACAR—TERENA Academic CA Repository. `http://www.terena.nl/tech/task-forces/tf-aace/tacar/certs.html`.

[154] William D. Tajibnapis. A Correctness Proof of a Topology Information Maintenance Protocol for a Distributed Computer Network. *Communications of the ACM*, 20(7):477–485, July 1977.

[155] S. Tuecke, V. Welch, D. Engert, L. Pearlman, and M. Thompson. Internet X.509 Public Key Infrastructure Proxy Certificate Profile. `http://www.ietf.org/internet-drafts/draft-ietf-pkix-proxy-09.txt`, November 2003.

[156] USHER: The Root Certificate Authority for Trust in Higher Education Research and Education. `http://usher.internet2.edu`.

[157] Kannan Varadhan, Ramesh Govindan, and Deborah Estrin. Persistent Route Oscillations in Inter-Domain Routing. *Computer Networks*, 32(1):1–16, January 2000.

[158] M. Wahl, T. Howes, and S. Kille. Lightweight Directory Access Protocol (v3). RFC2551, `http://www.ietf.org/rfc/rfc2551.txt`, March 1997.

[159] Tao Wan, Evangelos Kranakis, and P.C. van Oorschot. Pretty Secure BGP (psBGP). In *The 12th Annual Network and Distributed System Security Symposium (NDSS'05)*, San Diego, California, February 2005.

[160] Lan Wang, Xiaoliang Zhao, Dan Pei, Randy Bush, Daniel Massey, Allison Manchin, S. Felix Wu, and Lixia Zhang. Observation and Analysis of BGP Behavior under Stress. In *Internet Measurement Workshop*, 2002.

[161] Russ White. Securing BGP Through Secure Origin BGP. *The Internet Protocol Journal*, 6(3):15–22, September 2003.

[162] Jeannette M. Wing. A Specifier's Introduction to Formal Methods. *IEEE Computer*, 23(9):8–23, September 1990.

[163] The directory: Public-key and attribute certificate frameworks. ISO/IEC International Standard 9594-8/ITU-T Recommendation X.509(03/2000), March 2000.

[164] ANSI X9.63. The Elliptic Curve Digital Signature Algorithm (ECDSA). American Bankers Association, 1999.

[165] K. Zhang. Efficient Protocols for Signing Routing Messages. In *The 5th Annual Network and Distributed Systems Security Symposium (NDSS'98)*, San Diego, California, March 1998.

[166] Meiyuan Zhao, Sean W. Smith, and David M. Nicol. Evaluating the Performance Impact of PKI on BGP Security. In *4th Annual PKI R&D Workshop*, April 2005.

[167] Meiyuan Zhao, Sean W. Smith, and David M. Nicol. The Performance Impact of BGP Security. *IEEE Network Magazine, special issue on Interdomain Routing*, November/December. 2005. In press.

[168] Meiyuan Zhao, Sean W. Smith, and David M. Nicol. Aggregated Path Authentication for Efficient BGP Security. In *Proceedings of the 12th ACM Conference on Computer and Communications Security (CCS 2005)*, November 2005. To appear.

[169] Xiaoliang Zhao, Dan Pei, Lan Wang, Dan Massey, Allison Mankin, S. Felix Wu, and Lixia Zhang. Detection of Invalid Routing Announcement in the Internet. In *Proceedings of DNS 2002*, pages 59–68, June 2002.

[170] Xiaoliang Zhao, Dan Pei, Lan Wang, Dan Massey, Allison Mankin, S. Felix Wu, and Lixia Zhang. Validation of the MOAS Conflicts through Assertions. In *Proceedings of The International Conference on Dependable Systems and Networks*, June 2002.