Securing Routing Protocols through Information Corroboration

by

Tao Wan

A thesis submitted to

the Faculty of Graduate Studies and Research

in partial fulfillment of

the requirements for the degree of

Doctor of Philosophy

Ottawa-Carleton Institute for Computer Science

School of Computer Science

Carleton University

Ottawa, Ontario

January 2006

The undersigned recommend to

The Faculty of Graduate Studies and Research

acceptance of the thesis,

Securing Routing Protocols through Information Corroboration

submitted by

Tao Wan

Dr. Douglas Howe

(Director, School of Computer Science)

Dr. Evangelos Kranakis

(Thesis Co-Supervisor)

Dr. Paul Van Oorschot

(Thesis Co-Supervisor)

Dr. John Ioannidis

(External Examiner)

Carleton University

January 2006

Abstract

The continuous functioning of the Internet has become so vital to the normal operation of today's electronic communication activities that its disruption can cause catastrophic consequences. However, it is well-known that the modern Internet is not secure; both Internet application software and the underlying Internet infrastructure (such as routing) are vulnerable to a variety of attacks.

This thesis studies the vulnerabilities of Internet routing protocols and examines practical mechanisms for improving their security. Specifically, we propose to verify the factual correctness of routing updates in a vectoring routing protocol by corroborating information from multiple sources. Based on this method, two proposals, S-RIP and psBGP, are developed for respectively improving the security of Routing Information Protocol (RIP) and the Border Gateway Protocol (BGP), both of which are based on vectoring approaches and widely used on the Internet. Advantages of our proposals include: *simplicity* – cryptographic mechanisms used are manageable; *effectiveness* – they can successfully defend against threats from uncoordinated malicious parties; and *incremental deployability* – they can be incrementally deployed with some incremental benefits.

Acknowledgements

I wish to thank my co-supervisors, Dr. Evangelos Kranakis and Dr. Paul Van Oorschot, for guiding me through this long journey. Special thanks to Dr. Kranakis for his continuous encouragement in my frustration, and to Dr. Van Oorschot for his generous financial support, being an outstanding model of excellence, and for his extensive comments on my research. I must also thank my thesis external examiner, Dr. John Ioannidis, my thesis committee members, Dr. Micheal Barbeau, Dr. Ashraf Matrawy, and Dr. Jiying Zhao for their comments on this thesis, and Dr. Anil Somayaji for his proof reading of this thesis.

I am grateful to Mike Just for his collaboration with related research. I also wish to thank Steve Bellovin, Stephen Chou, John Ioannidis, Angelos Keromytis, Stefan Mink, Jean-Marc Robert, and anonymous reviewers for their constructive comments on our early papers that form the basis of this thesis.

It was a great pleasure to work at Carleton's Digital Security Group, with Jen Hall, Mohammad Mannan, Hong Shen, Julie Thorpe, Miguel Vargas Martin, Dave Whyte, and Glenn Wurster. My sincere thanks also go to the administrative staff at the School of Computer Science, Sandy Dare, Linda Pfeiffer, and Claire Ryan, and to Andrew Miles, Gerardo Reynaga for their technical support.

This research was supported in part by Alcatel, CSE (Communication Security Establishment, Government of Canada), MITACS (Mathematics of Information Technology and Complex Systems), NCIT (National Capital Institute of Telecommunications), NSERC (the Natural Sciences and Engineering Research Council), and OCIPEP (Office of Critical Infrastructure Protection and Emergency Preparedness, Government of Canada).

Last but not the least, special thanks to my wife, Jing Chen, my parents, my sisters and brothers for their continuous support in my life.

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

Today's Internet, a critical communication infrastructure which we are increasingly reliant upon, is insecure due to both design and implementation vulnerabilities of Internet protocols across all protocol layers. Of particular importance is the security of Internet routing protocols, which provide network layer routing services for interconnecting a large number of distributed networks. Unlike network services in other layers where one instance of a failure usually has only local effect, failure of a single Internet router may have global effect and can cause catastrophic consequences (as explained later). Thus, it is crucial to protect Internet routing protocols from potential failures to ensure the continuous functioning of the Internet, especially in the era when the Internet is on the way to consolidating other communication networks such as the telephony network.

This thesis focuses on the security design for two existing Internet routing protocols, namely the Routing Information Protocol (RIP) [83] and the Border Gateway Protocol (BGP) [109]. RIP is a popular *intra-domain* routing protocol used in an enterprise environment, and BGP is the only IETF standard *inter-domain* routing protocol used on the Internet. Both are based on the *vectoring* approach. Our goal is to improve the Internet routing security by designing practical and incrementally deployable security extensions for both RIP and BGP.

In this chapter, we give an overview of Internet routing, define the scope of the problems studied by this thesis, and outline our approaches and contributions.

1.1 Overview of Internet Routing

The Internet routing infrastructure consists of a large number of routers each of which runs distributed routing algorithm(s) and exchanges reachability information with others for establishing communication paths across the Internet. In addition, each router needs to forward packets not destined to itself toward their ultimate destinations to facilitate communication among non-directly connected networks. To this end, two fundamental functions are required for a router:

Control Plane – exchanging reachability information with other routers for establishing correct routing tables. Existing routing protocols (e.g., RIP and BGP) can ensure the correctness of a routing table provided that all routers participating in the protocol properly follow the routing protocol specifications, and the routing table is not tampered with.

Data Plane – forwarding packets toward their ultimate destinations. Each packet carries a destination IP address which is used by a router to search its routing table and select a next hop router which is closer to the packet's ultimate destination. If every Internet router builds a correct routing table and follows the forwarding process correctly, packets originated from anywhere on the Internet will eventually be able to reach their ultimate destinations.

The Internet routing infrastructure can be viewed as a collection of Autonomous Systems (ASes), each of which consists of a number of routers under a single technical administrative authority (e.g., sharing common routing policies). Internet routing protocols can be classified as *intra-domain* (used within an AS) and *inter-domain* (used between ASes). RIP [83], Open Shortest Path First (OSPF) [91], and Intermediate Systems to Intermediate Systems (IS-IS) [98] are three of the most popular intra-domain routing protocols. BGP [109] is the only inter-domain routing protocol used on the Internet.

There are two popular approaches used by Internet routing protocols: *vectoring* and *link state* (LS), based on Bellman-Ford [10, 38] and Dijkstra algorithms [35] respectively. In a vectoring routing protocol, each node maintains a routing table consisting of the best route and the associated costs for each destination in a network. The routing table of one node is computed based on the routing tables of its direct neighbors. Thus, each vectoring node has only partial view of the network topology, i.e., the best route to each destination in the network. Examples of vectoring routing protocols include RIP and BGP. RIP is referred to as a *distance vector* routing protocol due to the fact that it uses distance to represent the cost of a route. BGP is often referred to as a *path vector* routing protocol since it records the path through which a routing update has traversed.

In an LS routing protocol, each node floods to the whole network the state of each link attached to itself, and a link state database is then constructed from the link state advertisements received from every other node in the network. A shortest path from the constructing node to every other node in the network can be computed using Dijkstra algorithm [35]. Thus, all nodes in the network maintain the same link state database representing a complete view of the network topology.

1.2 Routing Security Problems

It is well-known that today's Internet routing infrastructure is insecure in both control and data planes. In the routing control plane, Internet routing protocols could fail and result in incorrect routing tables when some routers in the network have byzantine failures (i.e., functioning but not correctly) [103]. Particularly, a well-behaved router (i.e., correctly following the routing protocol specifications) may end up with an incorrect routing table due to failures of other routers in the network. In the routing data plane, Internet routers with correct routing tables may fail to forward packets. Both problems could be caused by many factors, e.g., hardware faults, software faults, misconfigurations, or malicious attacks [103]. We next discuss these two types of routing security problems, followed by a discussion of software vulnerabilities that can be exploited by an adversary to gain the control of an Internet router and launch attacks on both control and data planes.

1.2.1 Routing Protocol Vulnerability

Existing Internet routing protocols lack strong built-in security services. Firstly, many routing protocols provide none or weak mechanisms (e.g., plain-text password or system-wide shared keys) for authenticating neighbors, allowing an adversary to easily join the operation of a routing protocol. For example, an adversary compromising a personal computer (PC) inside a network may be able to capture the plain-text password used by a router for authenticating neighbors and then turn the compromised PC into a router by running a routing protocol configured with the captured password. In this way, an unauthorized adversary can successfully participate in routing operations and spread false routing information to cause service disruption.

Secondly, most routing protocols assume a trustworthy environment. In the case where there is no data origin authentication, routing updates are accepted only with rudimentary validation – for example, RIP [83] only checks that a routing update is from an IP address of a direct neighbor and that the source UDP port number is 520. When data origin authentication is implemented, routing updates are verified for the correctness of data origin and integrity only. However, after a route update is verified to be "authentic", the routing information conveyed in the update is trusted and used to update the recipient's routing table. This is risky since data origin authentication, which by our definition [87, p.361] includes data integrity, cannot guarantee the factual correctness of a message. A malicious entity or a compromised legitimate entity can send false information in a correctly signed message. A recipient can detect unauthorized alteration of the message, but cannot tell if the information conveyed in the message is factually correct unless it has the perfect knowledge of what it expects to receive. For example, a malicious legitimate *BGP speaker* (a router running BGP) can originate routes for an address space which it is not authorized to announce, resulting in service disruption of the legitimate owner of the hijacked address space. These malicious announcements cannot be detected with data origin authentication since the announcing router has legitimate keying materials for generating cryptographically correct messages (e.g., with valid digital signatures).

1.2.2 Forwarding Misbehavior

A misbehaving router (e.g., under the control of an adversary) may participate in routing protocols correctly (i.e., advertising correct routing information) but manipulate packets passing through the router. For example, an adversary can misconfigure a router (e.g., by installing static routes in the routing table) to fail to forward packets properly (e.g., selectively or completely dropping packets). An adversary can also manipulate packets to cause loss of confidentiality and integrity by eavesdropping and modifying packets.

1.2.3 Software Vulnerability

Internet routers can be misconfigured and usually run software with exploitable vulnerabilities allowing an adversary to obtain unauthorized access to the routers for manipulating routing protocols and data packets. For example, heap and buffer overflow vulnerabilities in Cisco IOS were discovered in April 2005 and reported in July 2005 [81], which would allow an attacker to completely take control of an affected router. As another example, numerous vulnerabilities in multiple implementations of Simple Network Management Protocol (SNMP) were discovered in March 2003 [27], which would allow unauthorized users to gain privileged access to an affected router. For a third example, a survey [63] on 471 Internet routers shows that a majority of them run services (e.g., Telnet and HTTP) with known flaws, and 17% of them accept connections from arbitrary IP addresses, allowing exploitations of these vulnerable services from anywhere on the Internet. To summarize, software vulnerabilities are ubiquitous and present in almost all software. Internet routers are composed of both hardware and software, and unavoidably bear software vulnerabilities.

1.3 Statement of the Problem

This thesis addresses security problems of routing protocols, particularly on the routing control plane. We study how to improve routing protocol security to tolerate certain byzantine failures. More specifically, we focus on how to effectively verify the factual correctness of routing updates to detect and contain fraudulent routing updates. Software vulnerabilities can be reduced by following security guidelines of software development [128]. However, it is unrealistic to expect perfect software since software engineering is an art more than a science and can hardly be perfect in a real world. Forwarding misbehavior appears unavoidable unless a router system is perfectly secured and cannot be compromised.

Nonetheless, forwarding misbehavior has only local effect, i.e., only the packets passing through a misbehaving router are subject to manipulation. We suggest that proactive monitoring approaches [65, 125] can be used to mitigate the risk.

We choose to focus on security designs for routing protocols based on vectoring approaches. Securing link state routing protocols has been well studied by Perlman [103] and others [94, 29, 141]. However, the security of vectoring routing protocols has not received a similar level of treatment partially due to the difficulty of validating DV routing updates, which are aggregated results of routing updates from other nodes [103, 141]. We further choose to study two vectoring routing protocols widely used on the Internet, namely RIP and BGP. They respectively represent intra-domain and inter-domain vectoring routing protocols, and account for a large portion of user traffic on the Internet. Although RIP has certain limitations and is being replaced by OSPF and IS-IS, it still has a large user base. If both RIP and BGP are secured, the security of the Internet routing infrastructure will be substantially improved. Our work on routing security in emerging mobile ad-hoc networks is presented in [65, 137], but not included in this thesis because forwarding misbehavior studied in [65] is not the focus of this thesis, and the techniques presented in [137] are similar to S-RIP.

1.4 Contributions of this Thesis

Many solutions based on cryptographic mechanisms have been proposed for securing routing protocols [103, 94, 123, 124]. However, cryptographic mechanisms alone cannot guarantee the factual correctness of routing updates. Thus, it is questionable if any router should deserve the full trust of any other router even it possesses all cryptographic credentials given that a legitimate router may be compromised or the owner of a router may be malicious. We propose to use information corroboration for improving confidence in the correctness of a routing update. Based on this idea, we propose S-RIP and psBGP for improving the security of RIP and BGP respectively.

1.4.1 Information Corroboration

To verify the correctness of a routing update, ideally, there would be a trusted routing authority which a router could check with. Unfortunately, neither does such an authority currently exist, nor is it practical to build one. It is also impractical to solely rely upon replication techniques, which are commonly used for building fault-tolerant systems (see §2.3), to build a robust Internet routing infrastructure, since potential costs might be prohibitive.

We propose to use *consistency* as an approximation of *correctness*. An advertised route is cross-checked for its consistency with information obtained from other sources. If there is a sufficient amount of consistency (to be defined in a particular routing protocol), the route is accepted as proper. This approach is inspired by the referral model widely used in social society for increasing confidence in the truth of information in the absence of an authoritative source of truth about that information. Information used for the corroboration and the nodes involved vary among routing protocols. Chapters 5 and 6 discuss the details of how to perform consistency checks for S-RIP and psBGP respectively.

We make use of a rating mechanism to measure trustworthiness of each information source, i.e., confidence in the correctness of the information obtained from a particular source. A simple and efficient method is developed for computing corroborated confidence in information that is consistent within a corroborating group. Although developed independently based on our intuition, it turns out that this method is consistent with Dempster-Shafer theory (DST) of evidence reasoning [33, 121] when information sources involved are independent.

1.4.2 Secure RIP (S-RIP)

Using information corroboration, we develop a secure distance vector routing protocol based on RIP, namely S-RIP. In S-RIP, each router rates every other router in a network with a numeric value, indicating trust in the correctness of the routing information provided by that node. A routing update received from one router is verified by cross-checking its consistency with the routing information of another router from which that update is derived. A routing update is verified as proper if the corroborated confidence in that update is no less than a configurable threshold. In this way, trust can be established in an advertised route using information corroboration without relying upon a trusted authority.

Our security analysis of S-RIP shows that it can successfully counter selected uncoordinated threats with high probability, including *neighbor spoofing*, *prefix hijacking*, and *distance frauds*. Our simulation results indicate that S-RIP routing overhead is relatively low, and can be effectively controlled with S-RIP thresholds.

1.4.3 Pretty Secure BGP (psBGP)

We also design a security extension for BGP, namely psBGP. Highlights of psBGP include:

1) psBGP makes use of a *centralized trust model* for AS number authentication. Each AS obtains a public key certificate from one of several trusted certificate authorities, e.g., Regional Internet Registries (RIRs), binding an AS number to a public key. We believe that such a trust model provides best possible authorization of AS number allocation and best possible authenticity of AS public keys. Authentication is usually the first step towards authorization. Without such a guarantee, an attacker may be able to impersonate another AS and thus be able to announce prefixes assigned to the impersonated AS.

2) psBGP makes use of a *decentralized trust model* for verifying the propriety of IP prefix assignment. Each AS creates a digitally signed *Prefix Assertion List (PAL)* consisting of a number of bindings of an AS number and prefixes, one for itself and one for each of its neighbors. An assertion made by an AS regarding its own prefixes (*prefix assertion*) lists all prefixes assigned to itself. An assertion made by an AS for a neighboring AS (*prefix endorsement*) may list all or a subset of the prefixes assigned to that neighbor. An *AS prefix graph* (see §6.6.3) is built by each AS based on the $PALs$ it has received from other ASes, and its ratings of those ASes. An AS prefix graph is then used for evaluating the trustworthiness and preference of a prefix origin by an AS. A prefix assertion is verified as proper if it is endorsed by a sufficient number of neighbors or the combined trust in the asserting and endorsing ASes is sufficient. In this way, trust in prefix assertions can be established using information corroboration in the absence of a trusted authority.

As discussed in Chapter 6 and 7, advantages of psBGP include: 1) *simplicity* – it uses a PKI which has a small number of certificate types, short certificate chains, and is of manageable size; 2) *effectiveness* – it can successfully defend against threats from uncoordinated, misconfigured or malicious BGP speakers; 3) *incremental deployability* – it can be incrementally deployed with some incremental benefits.

1.5 Overview of this Thesis

The rest of this thesis is organized as follows. In Chapter 2, we provide background information for later use in this thesis. In Chapter 3, we review existing approaches for improving routing security and two systems which make use of information corroboration for improving security. In Chapter 4, we present a framework for securing routing protocols, including a threat model, security goals, fundamentals of information corroboration, and a rating mechanism. In Chapter 5, we propose S-RIP for countering selected RIP threats, analyze the security effectiveness of S-RIP, and present simulation results. In Chapter 6, we begin by discussing BGP security threats, and outline five BGP security goals. We then present psBGP for achieving these goals. The novelties of psBGP are the use of a decentralized approach for verifying the propriety of IP prefix assignment, and the use of a stepwise approach for verifying AS_PATH integrity. In Chapter 7, we analyze psBGP, and compare it with S-BGP and soBGP – two other leading proposals for securing BGP. We conclude in Chapter 8.

Chapter 2

Background

In this chapter, we provide background information, for later use in the thesis, on Internet routing protocols (§2.1), public key cryptography (§2.2), fault-tolerant systems (§2.3), and Dempster-Shafer theory (§2.4).

2.1 Internet Routing Protocols

Conceptually, a routing network can be abstracted as a graph, where a vertex is a router and an edge is a network link. If a network consists of a small (e.g., several) or medium (e.g., tens or hundreds) number of routers, a single routing protocol is capable of exchanging and maintaining routing information in that network. Since there is a large number of routers (e.g., hundreds of thousands or more) on the Internet, any single routing protocol currently available cannot scale to that size. As a result, a hierarchical routing approach has been used for the Internet. The first level of the routing hierarchy is *inter-domain* routing protocols, and the second level is *intra-domain* routing protocols. BGP is the only inter-domain routing protocol used on the Internet. Commonly used intra-domain routing protocols include RIP, OSPF, and IS-IS.

In this section, we give an overview of RIP, OSPF, and BGP. RIP and BGP are based on vectoring approaches, and are the main focus of this thesis. OSPF is based on a link-state approach and its security is not studied in this thesis. A short description of OSPF is given here for the use in later discussions.

2.1.1 Routing Information Protocol (RIP)

RIP (we mean RIPv2 [83]) is widely used by many small and medium size organizations, despite the fact that it has certain limitations. For example, the maximum distance between two nodes in RIP is limited to 15 hops. A RIP routing table consists of a number of entries, with one for each destination in the network. Each route entry contains at least the following information:

- *destination*: the IP address and the subnet mask of a destination;
- *distance*: number of hops from this router to the destination;
- *next hop*: the IP address of the next router along the path to the destination;
- *timers*: several timers are associated with each route entry. One timer is set to 180 seconds. If no routing update about this route is received within three minutes, the distance of this route is set to 16, which designates infinity in RIP. Once a routing update is received about this destination, this timer is reset to 180 seconds.

A RIP router periodically (every 30 seconds) advertises routing updates to direct neighbors. A RIP routing update message consists of up to 25 routes. Each route contains a destination (IP and subnet mask), a distance, and a next hop. A next hop is only useful if it is directly reachable from a recipient. *Triggered Updates* are used to speed network convergence. Whenever the cost of a route changes, a routing update is triggered immediately without waiting for a normal periodic routing update.

A router or a host can also solicit routing information from another RIP router by sending a routing update request. For example, after a router reboots, it sends a routing update request to direct neighbors to collect routing information for initializing its routing table. A routing update request can also be used for diagnosis purpose. Routing update messages (request or response) are transmitted over UDP. In most cases, both source and destination port numbers are 520. A routing update request may originate from a port number other than 520. In that case, a routing update response will be returned to the originating port. RIP requires that a routing update response must be from a direct neighbor for it to be used for updating the recipient's routing table. However, RIP does not mandate that a routing update request must be from a direct neighbor. Therefore, a route update response message may be sent to a remote node (i.e., two or more hops away).

An advertised route may lead to the change of a recipient's routing table if it is: 1) a new route; 2) better than the existing route; or 3) from the originator of the existing route. In case 3), the existing route will always be updated regardless of the cost of the new route. If the newly received route has the same cost as the existing one, only the time-out timer associated with that route is reinitialized. Although it is understood that a routing update response message should be carefully validated before being used to update a routing table, RIP only performs rudimentary checks (e.g., the source IP and port number). Thus, it is vulnerable to a variety of attacks (see §5.2 for RIP vulnerability analysis).

Since RIP does not keep the complete path information to a destination, it is possible that a router advertises a route back to the router from which it learned that route. This can lead to the *counting to infinity* problem. RIP adopts the *Split Horizon* approach for solving the problem. In this approach, a router will not advertise a route back to the router from which it learned the route. An extension of the split horizon approach, which is commonly referred to as *Split Horizon with Poisoned Reverse*, requires a router to advertise a route back to the router from which it learned the route, but with a distance of infinity (16-hop). Split horizon approaches can break a loop involving two nodes, but not three nodes or more. The routing loop problem can be solved by associating each route with a complete path consisting of all nodes that have propagated this route in order. AS_PATH in BGP serves exactly this purpose.

2.1.2 Open Shortest Path First (OSPF)

OSPF [91] is an intra-domain routing protocol based on link state approach, and supports hierarchical routing. An AS running OSPF can be divided into a number of areas. Each node within an area advertises the states of its links in Link State Advertisements (LSAs) to every other node in the same area by flooding. An LSA usually consists of a link identifier (e.g., a subnet attached to the link), state of the link (up or down), cost of the link, and neighbors of the link. Every node receives the LSAs from every other node in the area, and builds the same link state (or topological) database (which is a weighted graph as each edge is associated with a cost). Using Dijkstra's algorithm [35], each node can compute a shortest path from itself to every other node within the same area.

An OSPF node with multiple network interfaces may connect to multiple areas, in which case, it will maintain a separate topological database for each area. Such node is often referred to as an Area Border Router (ABR). All ABRs (and some other routers) form an OSPF backbone area, which is responsible for distributing routing information between areas. In this way, routing flooding is limited to a smaller OSPF area rather than the whole AS. Packets originated from one OSPF area and destined to another area will be first forwarded to the ABR of the originating area, then to the ABR of the destination area, and finally to the ultimate destination.

2.1.3 The Border Gateway Protocol (BGP)

In this section, we give a brief overview of BGP and its operational practice, including IP address allocation, AS business relationship, and AS route exporting policies.

Overview of BGP

BGP is a vectoring inter-domain routing protocol. A BGP speaker establishes BGP sessions over TCP with its direct neighbors, exchanges routing information with them, and updates its own routing table based on the information received from them. Unlike a simple distance vector routing protocol (e.g., RIP [49]) in which a route usually has a simple metric (e.g., number of hops), a BGP route is associated with a number of attributes and the best route is selected among multiple routes to the same destination based on local policy. One notable route attribute is AS_PATH, which consists of a sequence of ASes traversed by this route. Thus, BGP is often referred to as a *path vector* routing protocol. Other details of BGP are discussed in §6.2.2.

IP Address Allocation

Internet Assigned Number Authority (IANA) [57] is the central authority of the whole IP address space. In the early stage of the Internet when it was small, any organization could directly apply to IANA for IP address space. As the Internet grew, it became apparent that a single authority was not capable of handling the large number of address space requests. As a result, a hierarchical structure has been developed for IP address space allocation, and IANA is the root of the IP address space allocation hierarchy.

On the second level of the hierarchy are five Regional Internet Registries (RIRs), each of which is responsible for the IP address space allocation in a particular geographic area. RIRs include: American Registry for Internet Numbers (ARIN) [5], Réseaux IP Européans (RIPE) [112], Asia Pacific Network Information Centre (APNIC) [4], Latin American and Caribbean Internet Addresses Registry (LACNIC) [76], and African Network Information Centre (AfriNIC) [2]. On the next level are large Internet Service Providers (ISPs), which can directly obtain IP address space from one of the four RIRs. A large ISP may further allocate a portion of its address space to a downstream service provider, or to an end user organization. In this thesis, we use *address allocation* to refer to the activity that one organization distributes IP address space to another organization, which may further distribute to a third organization or use for its own. Once an address space is allocated from one organization to another, the first organization no longer has right to use it unless the allocation relationship ends. *Address delegation* refers to the activity that an organization authorizes another to announce its address space in BGP. For example, an organization not running BGP may authorize a service provider to announce its address space allocated by another service provider. We use *address assignment* to refer to both address allocation and delegation when distinction is not relevant.

AS Business Relationship

ASes on the Internet can be roughly classified into three categories: a *stub-AS* has only one connection to other ASes; a *multi-homed AS* has more than one connection to other ASes, but is not designed to carry traffic for other ASes (e.g., for the purpose of load balance or redundancy); and a *transit-AS* has more than one connection to other ASes, and is designed to carry traffic for others.

While a stub-AS may have only one BGP speaker, a multi-homed or transit-AS often has more. A BGP session between two BGP speakers located within two different ASes is often referred to as *external-BGP* (eBGP), and a BGP session between two BGP speakers within a common AS is often referred to as *internal-BGP* (iBGP). An eBGP speaker actively exchanges routing information with an external neighbor by importing and exporting BGP routes. An iBGP speaker helps propagate routing updates to other BGP speakers within a common AS, and it usually does not make changes to a routing update.

Two ASes usually have one of the following four types of business relationship [56, 41]: *customer-to-provider*, *provider-to-customer*, *peer-to-peer*, and *sibling-to-sibling*. A customer AS usually pays a provider AS for accessing the rest of the Internet. Two peering ASes usually find it mutually beneficial to allow each other to access their respective customers. Two sibling ASes are usually owned by a common organization and allow each other to have access to the rest of the Internet (not only their respective customers). For example in Figure 2.1, ASes A, B, C, and D may attach to a common Network Access Point (NAP) and establish BGP sessions with each other. Among these four ASes, each pair forms a peer-to-peer relationship. ASes A, B, C, and D have their own customer ASes E, F, G, and H respectively. G has a customer I, and H has a customer J. ASes F and H may be owned by the same ISP but are located in different geographic locations, and they form a sibling-to-sibling relationship.

BGP Route Exporting Policy

An AS usually defines its route exporting policy for another AS based on their business relationship [41].

- *customer-to-provider*: a customer AS exports to a provider AS its own routes and the routes it has learned from its customers (including its direct customers and their customers).

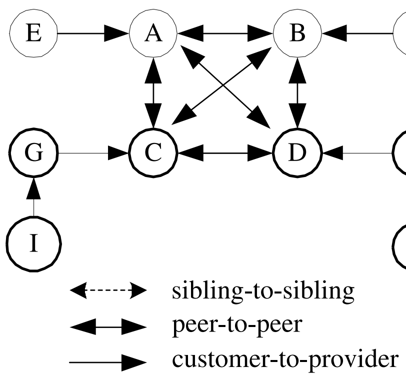


Figure 2.1: A simple AS topology with different types of AS relationships

- *provider-to-customer*: a provider AS exports to a customer AS its full routing table, including its own routes, the routes it has learned from its customers, providers, peers, and siblings.

- *peer-to-peer*: two peering ASes export to each other their own routes and the routes they have learned from their respective customers.

- *sibling-to-sibling*: two sibling ASes export to each other their full routing tables, including their own routes, the routes they have learned from their customers, providers, peers, and siblings.

2.2 Public Key Cryptography

Information security is a broad area and associated with diverse objectives [87, p.3] that can be more specifically defined in a given problem domain. Many techniques can accomplish information security objectives; cryptography provides a set of those techniques for achieving *confidentiality*, *data integrity*, *authentication*, and *non-repudiation*, among other security goals.

This thesis employs both symmetric and public key based cryptographic techniques for achieving certain security goals (to be defined later in the thesis, e.g., in §4.3). Here we give a brief overview of public key based cryptographic techniques, which are extensively used in this thesis (see [87] for a rigorous treatment of the subject).

2.2.1 Public Key Encryption

Public key based cryptography was first introduced by Diffie and Hellman in 1976 with their seminal paper titled "New Directions in Modern Cryptography" [34]. A practical public key encryption algorithm, now commonly referred to as RSA, was discovered in 1978 by Rivest, Shamir, and Adleman [114].

Unlike symmetric key based encryption where a common secret key is used for both encryption and decryption, public key based encryption uses two different keys for encryption and decryption. An encryption key is made public, and commonly referred to as *public key*. The corresponding decryption key is kept private, and commonly referred to as the *private key*. Consider a two-party communication between Alice and Bob, each of whom holds a public and private key pair, i.e., (e_a, d_a) by Alice and (e_b, d_b) by Bob. For Alice to send a message m privately to Bob, Alice first obtains Bob's public key e_b, and encrypts m to the cipher text c using an encryption transformation function with e_b. Alice sends Bob the transformed cipher text c. Bob decrypts c using a decryption transformation function with his private key d_b. Mathematical properties of public key based encryption are designed with the intention that a cipher text encrypted using Bob's public key e_b can only be decrypted with the corresponding private key d_b. Since d_b is only known to Bob, a third party will not be able to decrypt c, thus the privacy of m is protected.

Two main advantages of public key encryption are: 1) it provides a foundation for scalable key management; and 2) it can be used for constructing advanced cryptographic primitives (e.g., digital signatures) that offer new security properties (e.g., non-repudiation).

2.2.2 Digital Signature

Reconsider the above two-party communication between Alice and Bob. The public key encryption alone does not offer data origin authentication or data integrity. For example, a man-in-the-middle can replace c with c' which can be decrypted to m' by Bob. Bob cannot tell if m' is indeed the original message from Alice or the original message has been tampered with. To achieve data origin authentication which includes data integrity, digital signatures can be used.

A digital signature mechanism is a cryptographic primitive that allows for binding an identity to a piece of information such as a message m. A digital signature of m is a bit string that is dependent on both m and some secrets known only to the signer. In a digital signature mechanism based on public key cryptography, the secret is the private key of the signer. A digital signature of m can be an appendix to m, or contain m. The former is referred to as *digital signature with appendix*, and the latter *digital signature with message recovery* [87, p.427]. Hereafter, we only consider public key based digital signature with appendix, abbreviated as digital signature. For simplicity, we assume that a single public and private pair is used for both encryption and decryption, as well as digital signature generation and verification. However, a different key pair can be used for each purpose in both theory and practice, and is often recommended.

A digital signature mechanism consists of a signing algorithm and a verification algorithm. We next briefly review RSA based digital signature mechanisms (see [87, ch.11] fore more generic discussions on the subject). To sign a message m, Alice first computes a hash of m, denoted by h_m, using a well-known one-way hash function. Alice then signs over h_m using her private key d_a to compute a digital signature s_m. Both m and s_m are made available to a verifier Bob. To verify s_m, Bob first obtains Alice's public key e_a reliably. By reliably, we mean that Bob is assured that e_a is indeed the public key corresponding to the signing key d_a of Alice. Bob computes the hash of m, h_m, using the same one-way hash function used in the signing process. Bob next decrypts h'_m from s_m using Alice's public key e_a. The signature verifies successfully if $h_m{=}h'_m$, and fails otherwise.

2.2.3 Public Key Certificates

In order for public key based cryptographic mechanisms to be successful, the authenticity and integrity of a public key must be guaranteed. In other words, it must be verifiable that a public key indeed belongs to a claimant and has not been tampered with. Such assurance is usually achieved with public key certificates. A public key certificate is a signed data structure which binds a public key to an entity and other attributes. A typical public key certificate (e.g., X.509v3 [26]) usually consists of, but is not limited to, the following fields:

- *subject identifier*, the identifier of the entity to which this certificate is issued. It is required that a subject identifier is unique with the system in which the certificate is used. For example, it can be an X.509 distinguished name (DN), or email address.

- *public key*, the public key of the entity identified by the subject identifier.

- *expiration date*, usually consists of two dates (a *from* and a *to* date) specifying the time window during which this certificate is considered valid.

- *issuer identifier*, the identifier of the entity that issues this certificate, for instance, the DN of a certification authority.

- *digital signature*, the digital signature over all date fields in the certificate, obtained using the private key of the issuer of this certificate.

To issue a public key certificate to an entity, the issuing entity, which is usually a certification authority (CA), should have a policy in place for verifying that the entity to which the certificate is issued is indeed the authorized holder of the identifier to be bound with a public key. It is desirable that the issuing entity has jurisdiction over the name space which the certificate subject identifier belongs to. Otherwise, the authority of the public key certificate may be questionable. In addition, it is desirable that the issuing entity asks the party to which the certificate is issued to prove the possession of the private key corresponding to the public key carried in the certificate.

A public key certificate must be revoked if the public key carried in the certificate is no longer considered valid before expiration date. Examples of such circumstances include: the corresponding private key has been compromised; the party to whom the certificate is issued is no longer with the issuing organization, among others. Two approaches can be used for checking the revocation status of a public key certificate: Certificate Revocation List (CRL) and Online Certificate Status Protocol (OCSP) [95]. CRL is a signed list of the certificates (more precisely, their serial numbers) that have been revoked by the issuer of these certificates. To verify the status of a certificate, a verifier searches it in the CRL; a match indicates that the certificate has been revoked. In OCSP, a verifier sends a request to a server which returns the status of the certificate in question. Each approach has its own advantages and disadvantages; further discussion is beyond the scope of this thesis.

2.3 Fault-Tolerant Systems

In this section, we briefly review existing approaches to the design of *fault-tolerant* systems, particularly on distributed systems. A distributed system is composed of multiple physically separated processors which are connected together by networks, and communicate with each other via sending and receiving messages over the network. For example, a routing infrastructure is a distributed system that consists of physically separate routers which communicate with one another via routing updates. A fault-tolerant system can often be described using a state machine approach [78, 118, 119].

2.3.1 Introduction

A system is fault-tolerant if it continues to function properly or remain in a legitimate state, possibly with degraded service, when individual processors in the system have faults or failures [77]. Failures can be *simple* or *byzantine* [77, 103, 119]. A processor is said to have simple failures, which are also referred to as *fail-stop* failures, if it completely stops functioning. A processor is said to have byzantine failures if it continues functioning but with arbitrary or unpredictable behaviors. Properties of a fault-tolerant system include, but not limited to, *failure detection*, *self-stabilization*, and *failure masking*.

A system with *failure detection* can detect and/or report failures, but does not take action to tolerate them. Thus, manual intervention is often required to remove failures in order for the system to return to a legitimate state. Strictly speaking, such a system is not fault-tolerant by itself. However, failure detection is usually inexpensive to implement, and cost-effective for non-mission critical systems.

A *self-stabilizing* system can eventually reach a legitimate state regardless of initial configurations, and can remain in a legitimate state despite the occurrence of failures [35]. Such system can automatically detect and react to some simple failures. Most routing protocols, such as RIP, have the property of self-stabilization and are tolerant to, albeit with degraded service, simple transient failures such as link-down. However, it is considered difficult to design a self-stabilizing practical system that can tolerate byzantine failures.

A *failure masking* system remains in a legitimate state without degrading service even if some components within the system have failures. In other words, failures within the system are masked and invisible from outside of the system. For example, an ensemble implementing $t+1$ replicas of a processor can mask t simple failures [119] since the only functioning processor in the ensemble continues to produce correct output. Masking t byzantine failures requires $2t+1$ replicas using a majority voting scheme [119], provided that data origin authentication is guaranteed.

Design of fault-tolerant systems usually employs replication or redundancy techniques to withstand failures, including, but not limited to, *hardware* redundancy, *software* redundancy, and *information* redundancy. A real-world fault-tolerant system usually involves combinations of these techniques.

2.3.2 Hardware Redundancy

Hardware redundancy can be used to tolerate hardware failures. A typical example is Triple Modular Redundancy (TMR), which is composed of three identical modules and one voter. Given the same input, each module produces the same output. The voter takes the input from each module, and produces an output based on certain rules, for instance by the rule of majority voting. While the voter is a single point of failure in TMR, it is usually simple and can be designed with sufficient robustness to reduce the probability of failure. Moreover, the voter itself can also be replicated to remove such single point of failure. Hardware redundancy is usually used in conjunction with other types of redundancy. One disadvantage is that it always incurs additional costs. In a large distributed system, it might be prohibitively expensive to employ hardware redundancy to remove every single point of potential hardware failure. Thus, it is desirable to explore techniques that do not require hardware replications.

2.3.3 Software Redundancy

If a system is implemented in software, it can be replicated by running multiple instances on a common processor or multiple physically separate processors. The former is immune to certain software failures (e.g., a run-time failure), but cannot tolerate hardware failures. The latter can tolerate both software and hardware failures, but is more costly. Nevertheless, both approaches are vulnerable to software design or implementation failures, e.g., software bugs. This is exactly the vulnerability that frequently gets exploited by computer worms, which spread among a large number of computers across the Internet. To reduce the probability that an implementation failure occurs in all software replicas, different software implementations accomplishing the same functionality can be used. Such mechanism is often referred to as software *diversity* [39]. For example, the Internet root Domain Name System (DNS) is deployed using different software implementations running on different platforms. In this way, a software failure (e.g., a buffer overflow) in one implementation is unlikely to affect others, thus significantly lowering the risk that a single failure can bring down the whole system. However, it is still vulnerable to software design failures.

2.3.4 Information Redundancy

Information redundancy is commonly used in data transmission and data storage for error detection and correction. Trivial examples using information redundancy include Cyclic Redundancy Checks (CRC), Huffman coding [55], Automatic Repeat reQuest (ARQ) [18], and Spread Spectrum [130], among others. Digital signatures can also be considered as such technique since they add additional information (i.e., a digital signature) onto primary data to allow for data integrity verification. Other examples of using information redundancy for error detection include massive publishing [88] and routing flooding [103].

While information redundancy has been widely used in detecting data corruption or unauthorized data alternation, it cannot ensure the factual correctness of the information conveyed in the data. For example, it appears impossible that a story published in a newspaper could have been tampered with since one would have to modify many copies, if not all, to avoid being detected. However, the truth of the story cannot be guaranteed by massive publishing since the story may be based on a fraudulent data source or the author of the story may intend to mislead. One way to improve confidence is to corroborate the story with another one of the same concern published in a different newspaper by a different author. If they are consistent, one's confidence in the truth of the story significantly increases, assuming that each author has done his or her own due diligence and the two are not colluding. If two stories are inconsistent, one must be partially false. This thesis applies the same idea to improving confidence in the factual correctness of routing updates.

2.4 Dempster-Shafer Theory (DST)

Dempster-Shafer theory (DST) [121] is a mathematical theory of belief reasoning. It was developed by Shafer based on the earlier work by Dempster [33], primarily Dempster's belief combination rule. DST has been applied to security related areas including reputation systems in electronic commerce [146] and intrusion detection [28]. We first introduce some basic concepts of DST, followed by an example for illustrating these basic concepts. We then present the evidence combination rules in DST.

2.4.1 Basic Concepts

In DST, a *Universe of Discourse*, denoted by U, is a finite set of mutually exclusive and exhaustive propositions about a domain. U is often referred to as the *frame of discernment*. Let $U=\{s_1, s_2, .., s_n\}$, and 2^U be the power-set of U, i.e., all subsets of U. Each subset of U represents a general proposition about the domain. Three functions are associated with U: *basic probability assignment*, *belief function*, and *plausible belief function*.

A *basic probability assignment* is a function $m : 2^U \rightarrow [0, 1]$, where

$$m(\emptyset)=0 \quad and \quad \sum_{S\subseteq U} m(S)=1.$$

For a given proposition $S\subseteq U$, $m(S)$ represents the strength of the evidence supporting S. The basic probability assignment in DST is different from the probability assignment in the classical probability theory in that its domain is the power-set of U instead of individual elements of U. Thus, $m(S)$ is not the probability that proposition S is true, as defined in the classical probability theory. It represents the strength of evidence that will contribute to the final belief in S.

If S contains a single element, $m(S)$ represents the strength of the evidence that directly supports that element. If it contains two or more, $m(S)$ represents the sum of the strengths of the evidence that will contribute to the final belief in each of the elements in S. In other words, $m(S)$ characterizes the uncertainty of the evidence since it is not clear how much the evidence will contribute to the final belief in each of the elements in S. When new evidence comes from independent sources, such uncertainty can be reduced.

A *belief function* $Bel : 2^U \rightarrow [0,1]$ can be defined using the basic probability assignment. For a given subset S of U, $Bel(S)$ is the sum of the basic probability assignments of all subsets of S, i.e.,

$$Bel(S) = \sum_{R \subseteq S} m(R) \quad for\ S \subseteq U.$$

$Bel(S)$ can be interpreted as the belief in S drawn from all the known evidence that supports S. If S contains a single element s_1, $Bel(S)$ is equivalent to the probability that s_1 is true based on all known evidence. If S contains two or more elements, $Bel(S)$ represents the combined belief that any of the elements in S are true. It does not represent the probability that the elements in S are all true.

A *plausible belief function* $Pl : 2^U \rightarrow [0,1]$ can be defined using the belief function. For a given subset S of U, let $\overline{S}$ denote the complement of S.

$$Pl(S) = 1 - Bel(\overline{S}) \quad for\ S \subseteq U.$$

$Pl(S)$ can be interpreted as the belief in S if all unknown evidence turns out to be supportive of S or against $\overline{S}$. Since there is always unknown evidence at any moment, the true belief in S, denoted by $Tb(S)$, is always in between the belief $Bel(S)$ and the plausible belief $Pl(S)$. If all unknown evidence supports S, then $Bel(S) < Tb(S) = Pl(S)$. If all the unknown evidence supports $\overline{S}$ or is against S, then $Bel(S) = Tb(S) < Pl(S)$.

2.4.2 An Example

Here we use an example to illustrate the three functions in DST as described above. Let $U{=}\{B,R,W\}$ be the set of representations of possible colors of a ball, which could be *black, red,* or *white*. B is the proposition that the ball is black, R is the proposition that the ball is red, and W is the proposition that the ball is white. The basic probability assignments of U derived from some evidence are given below[1]:

$$m(B){=}0.3 \quad m(R){=}m(W){=}0 \quad m(B,R){=}0.2 \quad m(B,W){=}m(R,W){=}0 \quad m(B,R,W){=}0.5$$

$m(B){=}0.3$ represents that the strength of the evidence that can directly contribute to the belief in B is 0.3. $m(B,R){=}0.2$ represents that the strength of the evidence that can contribute to either B or R is 0.2. $m(B,R,W){=}0.5$ represents that the strength of the evidence that can contribute to B, R, or W is 0.5. Both $m(B,R)$ and $m(B,R,W)$ represent the uncertainty of the evidence. Based on the basic probability assignments, we can derive the following belief functions and plausible belief functions:

$$\begin{gathered}
Bel(B){=}\textstyle\sum_{X\subseteq B} m(X){=}m(B){=}0.3 \quad Bel(R){=}0 \quad Bel(W){=}0\\
Bel(B,R){=}m(B){+}m(R){+}m(B,R) = 0.3{+}0{+}0.2{=}0.5\\
Bel(B,W){=}m(B){+}m(W){+}m(B,W) = 0.3{+}0{+}0{=}0.3\\
Bel(R,W){=}m(R){+}m(W){+}m(R,W) = 0{+}0{+}0{=}0\\
Bel(B,R,W){=}m(B){+}m(R){+}m(W){+}m(B,R){+}m(B,W){+}m(R,W){+}m(B,R,W){=}1
\end{gathered}$$

Similarly, we can also derive the following plausible belief functions:

$$\begin{gathered}
Pl(B){=}1{-}Bel(\overline{B}){=}1{-}Bel(R,W){=}1{-}0 = 1\\
Pl(R){=}1{-}Bel(\overline{R}){=}1{-}Bel(B,W){=}1{-}0.3 = 0.7\\
Pl(W){=}1{-}Bel(\overline{W}){=}1{-}Bel(B,R){=}1{-}0.5 = 0.5\\
Pl(B,R){=}1{-}Bel(\overline{B,R}){=}1{-}Bel(W){=}1{-}0 = 1\\
Pl(B,W){=}1{-}Bel(\overline{B,W}){=}1{-}Bel(R){=}1{-}0 = 1\\
Pl(R,W){=}1{-}Bel(\overline{R,W}){=}1{-}Bel(B){=}1{-}0.3 = 0.7\\
Pl(B,R,W){=}1{-}Bel(\overline{B,R,W}){=}1{-}Bel(\phi){=}1{-}0 = 1
\end{gathered}$$

[1]By abuse of notation, we use the abbreviation $m(s)$ to denote $m(\{s\})$.

2.4.3 The Combination Rule

Evidence from two independent sources, represented by the basic probability assignments m_1 and m_2 respectively, can be combined to yield a new basic probability assignment, m_3, using Equation 2.1.

$$m_3(S) = \frac{\sum_{X\cap Y=S} m_1(X)\cdot m_2(Y)}{1-\sum_{X\cap Y=\emptyset} m_1(X)\cdot m_2(Y)} \qquad (X,Y\subseteq U) \tag{2.1}$$

The numerator $\sum_{X\cap Y=S} m_1(X)\cdot m_2(Y)$ represents the aggregated evidence that supports S, and $K{=}\sum_{X\cap Y=\emptyset} m_1(X)\cdot m_2(Y)$ represents the aggregated conflicting evidence. The denominator, $1{-}K$, represents the maximal non-conflicting evidence and serves as a normalization factor. More specifically, the denominator ensures $\sum_{S\subseteq U} m_3(S){=}1$, and has the effect of ignoring conflicting evidence. To illustrate how the combination rule works, We reuse the above example and assume the following two basic probability assignments

$$m_1(B){=}0.6 \quad m_1(R){=}0.1 \quad m_1(W){=}0.3; \quad m_2(B){=}0.2 \quad m_2(R){=}0.7 \quad m_2(W){=}0.1$$

Using Equation (2.1), we can obtain from m_1 and m_2 a new basic probability assignment m_3. We first compute the aggregated conflicting evidence K, then compute m_3.

$$\begin{aligned}K &= \sum_{X\cap Y=\emptyset} m_1(X)\cdot m_2(Y) = m_1(B)\cdot m_2(R) + m_1(B)\cdot m_2(W)\\ &+ m_1(R)\cdot m_2(B) + m_1(R)\cdot m_2(W) + m_1(W)\cdot m_2(B) + m_1(W)\cdot m_2(R)\\ &= 0.42 + 0.06 + 0.02 + 0.01 + 0.06 + 0.21 = 0.78\end{aligned}$$

$$\begin{aligned}m_3(B) &= \frac{\sum_{X\cap Y=B} m_1(X)\cdot m_2(Y)}{1-K} = \frac{m_1(B)\cdot m_2(B)}{1-K} = \frac{0.12}{1-0.78} = \frac{12}{22}\\ m_3(R) &= \frac{\sum_{X\cap Y=R} m_1(X)\cdot m_2(Y)}{1-K} = \frac{m_1(R)\cdot m_2(R)}{1-K} = \frac{0.07}{1-0.78} = \frac{7}{22}\\ m_3(W) &= \frac{\sum_{X\cap Y=W} m_1(X)\cdot m_2(Y)}{1-K} = \frac{m_1(W)\cdot m_2(W)}{1-K} = \frac{0.03}{1-0.78} = \frac{3}{22}\end{aligned}$$

Chapter 3

Literature Review

In this chapter, we review literature related to this thesis, including routing security, and systems using information corroboration for improving security.

3.1 Routing Security

We classify routing security mechanisms into two categories: securing the control plane (i.e., securing routing protocols) and securing the data plane (i.e., securing data forwarding). We next briefly review routing security mechanisms in each category.

3.1.1 Securing the Control Plane

Significant work has been done in securing routing protocols. Perlman [103] is among the first to recognize and study the problem of routing security. Perlman classified router failures into two categories: *simple failures* and *byzantine failures*. A router with simple failure stops functioning completely. A router with byzantine failure may continue functioning, but not properly. A byzantine failure could be caused by hardware faults, software bugs, misconfiguration, or malicious attacks. Perlman proposed using digital signatures, resource reservation, hop by hop acknowledgments, and source routing, among other mechanisms, to achieve robust flooding and robust routing. The proposed solutions guarantee to find a non-faulty path (i.e., all intermediate links and routers on the path are non-faulty) from a non-faulty source to a non-faulty destination provided that such a path exists in the network. This thesis differs from Perlman's work in three aspects: 1) We focus on securing existing routing infrastructures with incremental deployability, while Perlman's solution is to build a new robust routing protocol. 2) We focus on securing distance vector routing protocols (RIP and BGP), while Perlman's work is based on a link state approach. 3) We focus on securing the control plane only, while Perlman's work also addresses the security of the data plane.

Bellovin [11] discussed security vulnerabilities of Internet routing protocols as early as 1989 (see also [15]). More recently, Bellovin and Gansner [14] discussed potential link-cutting attacks against Internet routing.

Kumar [75] proposed the use of digital signatures and sequence numbers for protecting subverted network links. By gaining the control of a network link, an intruder can manipulate routing updates, e.g., modifying or replaying routing updates. Digital signatures can prevent unauthorized modification of routing updates. Sequentially numbering routing updates can prevent replay attacks. Kumar also proposed to use retransmission and acknowledgments for improving reliability and security.

Smith et al. [124] also made use of digital signatures and sequence numbers, and proposed a loop-free path finding algorithm for securing distance vector routing protocols. By including a new field, next-to-last-hop, in routing updates, a recipient node can validate, based on its local routing information, if a routing loop has been formed. This approach can prevent unauthorized modification and replay of routing updates, as well as fraudulent routing updates that can lead to routing loops. However, it cannot prevent fraud on route metrics, e.g., maliciously making a route distance longer or shorter.

Zhang [148] suggested that public-key based digital signatures are computationally inefficient for signing routing updates since both signature generation and verification must be done online using computationally inefficient algorithms. Zhang proposed the use of one-time digital signatures combined with one-way hash chains for signing routing updates. While this approach is theoretically more efficient than public key based digital signatures, it has practical disadvantages and incurs significant message overhead.

Goodrich [45, 46] proposed a method, called leap-frog, for securing distance vector routing protocols. The proposed method ensures that a node always chooses the best route among all the routes for a common destination it has received from direct neighbors. This approach assumes that a node will receive a route for a particular destination from each of its neighbors in the network, which may not be true.

Mittal and Vigna [90] proposed the use of intrusion detection for securing distance vector routing protocols. A precomputed master routing database contains the paths and associated costs from every node to every other node in a network. Sensors are installed on selected subnets of the network. Routing information related to a subnet is extracted automatically from the master routing database, and is distributed to the sensor installed on that subnet. A sensor uses its portion of the master routing information to validate routing updates transmitted over its subnet. One advantage of this approach is that it does not require modifications to the routing protocol to be secured, thus it is incrementally deployable. However, it also shares the disadvantage of other intrusion detection systems that it can detect but cannot prevent fraudulent routing updates from spreading across the network.

Hu et al. [52, 53, 54] proposed several efficient mechanisms using one-way hash chains and authentication trees as construction primitives for securing distance vector routing protocols. Their approach can prevent false routing updates with certain frauds, such as shortening the distance of a route (referred to as *shorter distance fraud*). However, some other frauds are not addressed, for instance making the distance of a route longer (referred to as *longer distance fraud*). Another disadvantage is that it significantly increases message overhead since each route must include one or more hash values to authenticate itself, for instance, 128 bits per hash if using hash function MD5 [113].

Pei et al. [101] proposed a triangle theorem for detecting questionable RIP advertisements. Probing messages based on UDP [105] and ICMP [106] are used to further determine the validity of a questionable route. One disadvantage of this approach is that a probing message can be manipulated. For example, a node advertising an invalid route can convince a receiver that the route is valid by: 1) manipulating the TTL value in a probing message; or 2) sending back an ICMP message (port unreachable) on behalf of the destination of the probing message.

Considerable research has been published on improving BGP security. For a thorough analysis of BGP vulnerabilities and protections, see Murphy [93, 92].

Goodell et al. [44] proposed a protocol, namely Inter-domain Routing Validator (IRV), for improving the security and accuracy of BGP. In this approach, each AS builds an IRV server which is authoritative of the inter-domain routing information of that AS. An IRV server maintains a routing database, which is separated from, but needs to be synchronized with, the routing tables of BGP speakers. An IRV server can query another IRV server to verify BGP update messages received by its hosting AS. Improper prefix origination and AS_PATH could be detected by uncovering the inconsistency among responses from other IRV servers. One advantage of IRV is that it supports incremental deployment since it does not require changes to the existing routing infrastructure. However, it is considered as a second line defense mechanism since it detects incorrect BGP routing information, but does not prevent it from propagating. Another challenge of this approach is to synchronize two separate routing databases respectively maintained by an IRV server and a BGP speaker. There is some similarity between the IRV method and the information corroboration proposed in this thesis, i.e., they both consider validating routing updates by corroborating them with data from different, albeit possibly dependent, sources. However, the detailed validation mechanisms are completely different. Moreover, the mechanisms proposed in this thesis are intended to be integrated with existing routing protocols and validate routing updates dynamically, while IRV runs in parallel with the existing BGP routing infrastructure.

S-BGP [72, 71, 120] is the most complete and concrete security proposal to date for addressing BGP vulnerabilities. S-BGP makes the use of centralized PKIs for authenticating AS numbers and IP prefix ownership. S-BGP PKIs are rooted at RIRs, and run parallel to the existing hierarchical system of AS number assignment and IP address allocation (see discussion on IP address allocation in §2.1.3). AS_PATH integrity is protected using nested digital signatures. More discussion on S-BGP is given in §7.3.1.

soBGP [144] proposes the use of a web-of-trust model for AS public key authentication, and a centralized hierarchical model for IP prefix ownership verification. AS_PATH is verified for plausibility by checking against an AS topology graph. Each AS issues a certificate listing its neighboring ASes. A global AS graph can be constructed from those certificates, and is used for verifying the existence of an AS_PATH. See §7.3.2 for more discussion on soBGP.

Kruegel et al. [74] proposed a model of AS topology augmented with physical Internet connectivity to detect and stop anomalous route announcements. Their approach passively monitors BGP control traffic, and does not require modification to the existing routing infrastructure. Therefore, it would appear easy to deploy. Disadvantages of this approach include: it requires information about physical Internet topology (e.g., distance between two ASes), which might not be practical, and it cannot stop fraudulent routing updates from propagating on the Internet.

In a rigorous study of prefix origination authentication, Aiello et al. [3] formalized the IP prefix delegation system, presented a proof system, and proposed efficient constructions for authenticating prefix origination. Real world routing information is analyzed and used to reconstruct the IP prefix delegation graph over the Internet. They discover that the current prefix delegation on the Internet is relatively static and dense, but also note that it is extremely difficult, if not impossible, to determine such delegation structure.

Listen and Whisper are mechanisms proposed by Subramanian et al. [125] for protecting the BGP data plane and control plane respectively; they are best used together. The first approach (Listen) detects invalid data forwarding by detecting "incomplete" (as defined in [125]) TCP connections. Whisper uncovers invalid routing announcements by detecting inconsistency among *path signatures* of multiple update messages, which originate from a common AS but traverse different paths.

Hu et al. [51] proposed a Secure Path Vector (SPV) protocol for securing BGP. SPV makes use of efficient cryptographic primitives, e.g., authentication trees and one-way hash chains for protecting AS_PATH, and it is argued that it is more efficient than S-BGP.

Many researchers have explored the security of link state routing protocols (e.g., OSPF) [103, 94, 30, 29, 21, 141, 104]. Securing wireless ad hoc networks has also attracted extensive interest [149, 84, 149, 22, 52, 53, 147, 54, 137]. Reputation-based systems have been proposed for facilitating trust establishment in electronic commerce [110, 146].

3.1.2 Securing the Data Plane

Traditionally, securing the data plane has received less attention than securing routing protocols, primarily because forwarding failures do not have the same cascading effect as routing protocol failures. For example, a misbehaving router can misforward packets passing through the router, but this does not affect other routers' forwarding behaviors. Recently, the advancement of mobile ad hoc networks (MANETs) has generated significant interests in this area due to the fact that mobile nodes are often resource constrained and forwarding misbehaviors in a MANET can cause more serious consequences than on the Internet. We next briefly review some proposed approaches for securing the data plane.

Perlman [103] proposed the use of hop-by-hop packet acknowledgment for detecting packet loss, and source routing for retransmitting packets. In this scheme, a router receiving a packet is required to send an acknowledgment back to the source of the packet. If an acknowledgment is not received by the source within a time window, the packet is considered lost, and is retransmitted using a different path. In this way, failures such as link-down or malicious packets dropping will be detected. Disadvantages of this approach include: 1) hop-by-hop packet acknowledgment generates significant network overhead; and 2) source routing is considered impractical on the Internet.

Cheung et al. [30] proposed a probing method for mitigating denial of service attacks in a fixed routing infrastructure using neighborhood probing. In their method, a testing router sends a probe message to a tested router along a path which starts from the testing router, goes through the tested router, (or with one or more well-behaved intermediate routers sitting in front of the tested router) and ends at the testing router. If the probing message can successfully arrive at the testing router, the tested router proved being well-behaved. This approach requires a testing router to have a private address which allows it to generate a packet destined to itself but which goes through the tested router, which may not be practical.

A distributed monitoring approach is proposed by Bradley et al. [21] for detecting disruptive routers. This method is based on the principle that any received packets not destined for the recipient router should leave that router. Each router maintains several counters for each of its neighbors, and each counter stores the packet number of a certain type. All routers within a common AS exchange their counters periodically (or by request); any routers dropping packets can be detected and further isolated. This is an interesting approach, but some of its assumptions are not practical.

Padmanabhan and Simon [99] proposed a secure traceroute to locate faulty routers on the Internet. In their approach, end hosts monitor network performance. If the performance of an end-to-end communication to a particular destination degrades, a *complaint* bit is set in all subsequent packets to that destination. The complaining host or its closest router can initiate a troubleshooting process if sufficient packets with the complaint bit set are observed. The troubleshooting node first sends a secure traceroute packet to the next hop, which can be derived from its routing table. The router receiving the secure traceroute packet should send a response back which also includes a next hop address. This process repeats until a faulty router is located (no traceroute response received from that router) or every router on the path to the ultimate destination proves healthy. Shared secret keys are negotiated between the diagnosing node and the diagnosed one. Cryptographic mechanisms are then used to provide various security services (e.g., authentication).

Marti et al. [84] proposed and implemented two protocols for detecting and mitigating misbehaving nodes in MANETs based on Dynamic Source Routing (DSR) [62] by *overhearing* neighborhood transmissions. Let node *A* forward to *B* a packet originated from *S* and destined to *D*. With this approach, *A* will overhear *B*'s transmission to confirm that *B* indeed forwards the packet along the routing path to the next hop, say node *C*. If node *A* does not overhear such a transmission, the tally of node *B*'s misbehavior increases. If the tally exceeds a preconfigured threshold, *B* is classified as misbehaving and the originating node *S* is informed. After learning that *B* is misbehaving, *S* will rate *B* with a small negative value. Since *S* uses the paths with the highest metrics taking into account of the node ratings, the result is that *B* is excluded from the network for a certain period of time if not permanently. This method is effective in detecting misbehaving nodes that are one-hop away. To monitor the behavior of nodes two or more hops away, one has to rely on the information from other nodes, which introduces the vulnerability that well-behaved nodes could be excluded from the network by malicious or incorrect accusation. Such malice is often referred to as *blackmail* attacks.

Buchegger and Le Boudec [22] developed the CONFIDANT protocol for encouraging node cooperation in dynamic ad-hoc networks. Each node monitors the behavior and maintains the reputation of its neighbors. The reputation information may be shared among friends. A trust management approach similar to PGP is used to validate received reputation information. Nodes with low reputation may be isolated from the network. Thus, nodes are encouraged to be cooperative for their own interest.

Awerbuch et al. [7] proposed a secure routing protocol for resisting byzantine failures in MANETs. This method requires an ultimate destination to send an acknowledgment back to the originator for each of its successfully received packets. If the loss rate of acknowledgment packets exceeds the predefined threshold, which is set slightly above the normal packet loss rate, the route used for sending packets from the source to the destination is detected as faulty and a binary search probing technique is deployed to locate the faulty link. The disadvantages of this protocol include: 1) it introduces significant routing overhead, especially when communication patterns are asymmetric; and 2) a data packet with an inserted probe list can be distinguished from those without probe lists, despite the fact that the probe list is encrypted by each forwarding router, and it cannot be tampered with. Thus, a malicious node can defeat this method by treating probing packets differently than other packets. For example, a malicious node can avoid detection by dropping data packets without probe lists but forwarding other packets.

We proposed a proactive distributed *probing* technique [65] for detecting and mitigating the malicious packet dropping attack. In this approach, every node proactively monitors the forwarding behavior of other nodes. Suppose node A wants to monitor the forwarding behavior of node B. A will send a probe message to a node one hop away from B, say to node C. C is required to respond to the probe message by sending back an acknowledgment to A. If A receives the acknowledgment within a time window, it acts as a confirmation that node B indeed forwarded the probe message to C. With the assumption that a probe message is indistinguishable from a normal data packet, A confirms that B also forwards other packets properly.

Malicious nodes silently dropping packets exhibit the same behavior as selfish nodes, which may choose to drop packets for the sake of saving their own constrained resources, such as battery or CPU cycle. Selfishness and its threats to network performance have been well studied by Roughgarden [115]. Incentive mechanisms have been proposed to encourage selfish nodes to be cooperative and to forward packets for others. The probing scheme proposed in [65] can also be used to detect and mitigate the selfishness problem.

3.2 Information Corroboration

In this section, we review some areas in which information corroboration has been used for increasing confidence in truth of information, and improving system security.

The main idea behind information corroboration is to take into account multiple sources to reduce the likehood of accepting false information from some faulty sources. Assume there are n entities, and the probability that each entity $1 \leq i \leq n$ lies is $0 \leq p_i \leq 1$. $p_i = 0$ and 1 respectively represents that entity i never and always lies. The probability that all entities lie at the same time is $\prod_1^n p_i$, which will be less than p_i except in the case that $\forall\ 1 \leq i \leq n$ $p_i = 0$ or 1. In these cases, all entities always tell the truth or always lie, thus corroborating many of those entities does not yield additional benefit.

This simple idea is behind many mechanisms used for improving robustness, fault tolerance, or availability, all of which mitigate risks associated with single points of failures. Here we present two such examples. A brief survey on methods for achieving compromising tolerance using independent corroboration is given by Kahn [66]. More recently, van Oorschot proposed the use of information corroboration for achieving authentication [127], for example, email source authentication.

3.2.1 Web-of-Trust

As discussed in §2.2.3, it is critical to ensure the authenticity and integrity of a public key in order for a public key based mechanisms to be effective. Within an administrative domain where a common trust can be established among all entities, it is straightforward to have public key certificates used with the domain signed by the commonly trusted certification authority to protect and establish trust in public keys. However, it continues to be viewed as impractical to assume a common trust anchor over the Internet for various reasons such as political, economical, cultural, and religious reasons, among others. A web-of-trust is a non-hierarchical model for establishing trust in a public key and entity binding, which usually employs information corroboration in the absence of a commonly trusted authority. In a web-of-trust model, the trust in a public key and entity binding is obtained based on the number of parties signing that binding and a verifier's trust in the signing parties.

We use Pretty Good Privacy (PGP) [153] to illustrate how a web-of-trust model works. In PGP, trust relationships can be represented by a directed graph $G{=}(V, E)$, where vertices $u, v{\in}V$ are bindings of a public key and an entity (abbreviated key-entity bindings), and a directed edge $u{\rightarrow}v$ represents that key-entity binding v is signed using the private key corresponding to the public key in the key-entity binding u. In other words, key-entity binding v is endorsed by the entity in the key-entity binding u. A vertex v may have multiple incoming edges, indicating that v is endorsed by multiple entities.

A trust value is manually assigned by one entity to another, representing a personal opinion of the trustworthiness of that entity. In PGP, one of the following trust categories can be assigned to an entity: *fully trusted, marginally trusted, untrusted*, and *unknown*. A key-entity binding is trusted if it is endorsed by a fully trusted entity or by two or more marginally trusted entities.

In the former case, trust in the authenticity of a public key is exclusively obtained from the full trust in the signing entity. In the later case, trust is obtained collectively from the marginal trust in the multiple signing entities. The idea behind such trust is that a single marginally trusted entity may misbehave and erroneously sign a key-entity binding in which the public key actually does not belong to the bound entity. However, the likehood that multiple signing entities misbehave altogether is likely low. Thus, confidence increases in the authenticity of a key-entity binding if it is signed by multiple albeit marginally trusted entities. However, it has been pointed out by Reiter and Stubblebine [108] that this model has some problems, which arise from the fact the one user in the system can have multiple identities and hold multiple key-entity bindings. In other worlds, the multiple signing parties of a key-entity binding could be the same user. As a result, one user can create multiple key pairs, and use them to "manufacture" the equivalence of endorsement from several parties, thus defeating the purpose of information corroboration. This problem illustrates the importance of identity authentication in a system.

3.2.2 Email Authentication

We start with a brief overview of the Simple Mail Transfer Protocol (SMTP) [107, 61], then show how a sender address can be spoofed, and end with a description of an email authentication proposal which makes use of information corroboration.

SMTP Basics

SMTP is the IETF standard protocol for transferring electronic mail on the Internet. It defines a set of commands that can be exchanged between two parties for mail transfer, along with a set of command status codes. Figure 3.1 illustrates the SMTP message flow between an originating SMTP server representing "*alice.com*" and a receiving SMTP server representing "*bob.com*" for delivering an email message from "*x1@alice.com*" to "*y1@bob.com*". Note the sender address specified by the SMTP command "HELO" and "MAIL FROM" can be forged to an arbitrary address if the mail server representing "*bob.com*" does not employ any authentication mechanism. This is exactly the vulnerability exploited by spammers.

Sender Address Spoofing

A spammer usually sends to a large number of recipients unsolicited emails with spoofed sender addresses. Since SMTP does not verify the authenticity of an originating party's domain name, a spammer can use a single SMTP engine (e.g., running on a compromised PC) to send out a large number of spammed emails with arbitrary purported sender addresses. Figure 3.2 shows how an attacker sends out a spoofed message from "*attack.com*" to "*bob.com*" using "*alice.com*" as the sender domain.

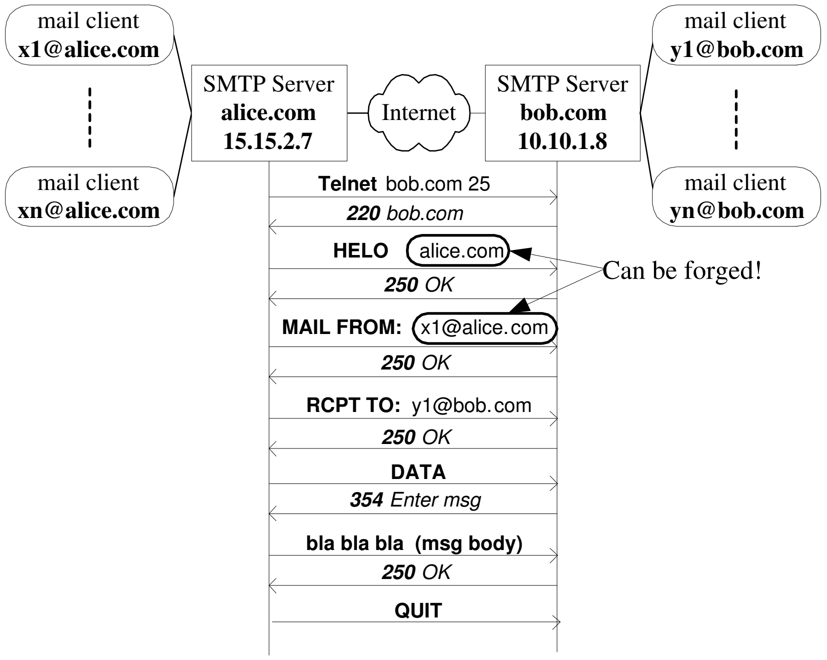


Figure 3.1: SMTP message flow

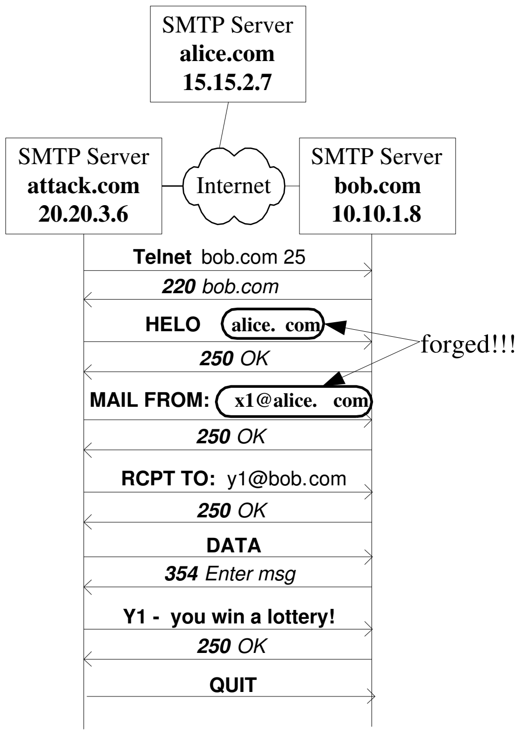


Figure 3.2: Spamming - sender address spoofing

Email Authentication

A number of mechanisms have been proposed for fighting spam by authenticating sender addresses. Sender Policy Framework (SPF) [79] is a popular proposal which has been adopted by a number of organizations. While SPF has been combined with Microsoft's Caller Identifier proposal [31] into a new proposal, namely Sender ID [82], it is reviewed here independently for the sake of completeness. SPF requires a domain running SMTP servers to publish in its DNS records the identities (e.g., IP addresses) of its authorized outgoing SMTP servers. An SMTP server implementing SPF can verify the authenticity of a sender address (i.e., the domain name in the MAIL FROM field) by checking the consistency between the IP address of an originating SMTP server and the IP addresses of the authorized SMTP servers published by the sender domain.

For example in Figure 3.2, "*alice.com*" publishes in its DNS $15.15.2.7$ as the IP address of its authorized outgoing email server. To publish the IP addresses of authorized email servers, a domain needs to add new records, namely SPF records, into its DNS records. A verifier can then look up in DNS SPF records the IP addresses of the authorized email servers for a particular domain. Upon receiving from "*attack.com*" the SMTP commands "HELO *alice.com*" (which can be omitted by a sender) or "MAIL FROM: *alice.com*", the SMTP server in "*bob.com*" verifies the sender IP address "$20.20.3.6$" against the IP address of the authorized SMTP server published by "*alice.com*" which is "$15.15.2.7$". Since they are inconsistent, the SMTP server in "$bob.com$" detects that this email has originated by an unauthorized party (or with a spoofed sender address), and thus can reject it. If every domain adopts this mechanism, a significant amount of spam might be detected and dropped.

The trust in the authenticity of a sender's domain name is obtained due to the fact that it is unlikely, albeit possible, that a spammer can modify DNS records which are separate from email services.

Chapter 4

A Framework for Securing Routing Protocols

4.1 Introduction

All existing Internet routing protocols assume a trustworthy environment, where routers are trusted to follow routing protocol specifications and be cooperative. Such an assumption was reasonable during the early stages of the Internet when the number of devices connected to the Internet was relatively small and the use of the Internet was limited to a small non-hostile population (e.g., research communities). Today's Internet connects hundreds of millions of computers and spans almost the entire world. While most Internet users are well behaved, there are many who are constantly trying to exploit Internet vulnerabilities for a variety of reasons including information warfare, financial gains, and personal glory, among others. Thus, the trustworthy Internet does not exist anymore and cannot be safely assumed by routing protocols.

Instead, we need to assume a hostile environment for Internet routing protocols, in which legitimate routers can be compromised and unauthorized devices can join the operation of routing protocols, which are usually designed with null or weak authentication mechanisms. In other words, a routing protocol should be designed to withstand inside misbehaving routers or byzantine failures. Routing protocols with this property are commonly referred to as tolerating byzantine failures or having byzantine robustness [103]. In contrast, traditional routing protocols can tolerate only transient failures, e.g., temporary link failures, and do not have byzantine robustness.

In this chapter, we propose the use of information corroboration for enhancing the security of Internet routing protocols. While corroboration has been used in many other areas for improving confidence in the truth of information and for resisting failures, to the best of our knowledge, it has not been systematically applied to routing protocol security.

The rest of the chapter is organized as follows. In §4.2, we present a threat model for routing protocols. In §4.3, we summarize a number of generic security goals for routing protocols, which will be instantiated by a particular routing security proposal, e.g., by S-RIP or psBGP. In §4.4, we present security mechanisms for achieving the above routing protocol security goals. More specifically, we review cryptographic mechanisms, and propose the use of information corroboration for verifying the factual correctness of routing updates. §4.5 presents a rating mechanism for measuring the trustworthiness of nodes (e.g., routers or ASes) since node behaviors are usually unpredictable in a hostile environment and they cannot be fully trusted or distrusted forever. A simple method is proposed for computing combined confidence in an assertion that is consistent among a corroborating group. We show that this method is consistent with the Dempster-Shafer theory.

4.2 A Threat Model for Routing Protocols

Routing protocols face many threats. In this section, we present a threat model that identifies the sources of threats, and a selective threats which should be addressed to make a routing protocol safe.

4.2.1 Sources of Threats

An interconnected network consists of *routers* (i.e., nodes running routing protocols and actively exchanging routing information with others), *hosts* (i.e., nodes not running any routing protocol or not actively exchanging routing information with any other), and network *links* which connect routers and hosts. Threats against a routing protocol can be from routers, hosts, and network links (see Figure 4.1). Threats from a legitimate router are commonly referred to as *inside* threats, and from a host or network link are referred to as *outside* threats. Accordingly, an attacker in control of a legitimate router is referred to as an *insider*, and one in control of a host or network link is referred to as an *outsider*.

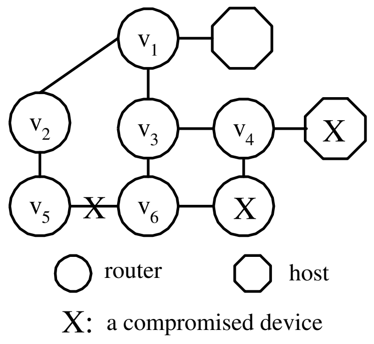


Figure 4.1: Threats can be from a router, a host, or a network link.

Inside and outside attackers usually have different capabilities in terms of the threats they can initiate against a routing protocol, especially when cryptographic mechanisms are implemented for protecting routing protocols. An insider is considered to have full capability, but an outsider is not. For example, an insider has all cryptographic keying materials, and is indistinguishable from a legitimate router. An outsider has no access to legitimate cryptographic keying materials, thus its capability is limited.

When no cryptographic mechanism is implemented on a router, there is still a slight difference between the capabilities of an insider and an outsider. For example, an outsider may not be able to inject routing updates into the network as easily as an insider. However, such a difference is considered insignificant, and thus we ignore it here. In other words, an outsider is treated equivalent to an insider when no cryptographic mechanism is used.

An outsider in control of a host in a network can turn the host into a router by running proper routing software on the host, and may then become an insider by exploiting vulnerabilities of *neighbor authentication* mechanisms used by a routing protocol. For example, a BGP session is based on TCP. Without proper neighbor authentication, any host in the network can establish a BGP session over TCP with a legitimate BGP router and become a legitimate BGP speaker. In other words, an outsider can readily become an insider by exploiting the neighbor authentication vulnerability of a routing protocol. Such an exploitation is referred to as *neighbor spoofing*.

An outsider in control of a network link can also become an insider by tapping a host onto the link and launching neighbor spoofing. A compromised network link may or may not directly connect two legitimate routers. An attacker has more capability in the former case than in the latter since it has direct access to, thus can manipulate, routing updates transmitted between two legitimate routers over the compromised link. In the latter case, an insider has no such capability. However, such an advantage diminishes if cryptographic mechanisms are implemented for protecting routing updates. For simplicity, we treat threats from a network link the same way as from a host. For example, in Figure 4.2, if the direct link between nodes v_5 and v_6, denoted by $e(v_5, v_6)$, is in the control of an adversary v_m, $e(v_5, v_6)$ becomes a path from v_5 to v_6 via v_m that is controlled by an adversary. In other words, threats that can be launched from link $e(v_5, v_6)$ can now be launched from host v_m.

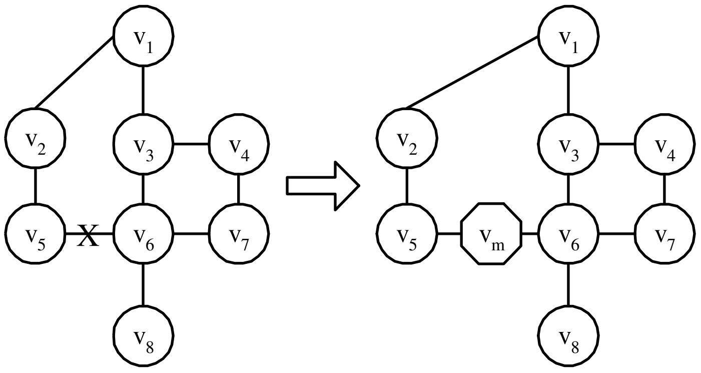


Figure 4.2: A compromised network link changes the network topology.

To summarize, an interconnected network can be modeled as a graph $G{=}(V_g, V_b, E)$, where V_g is a set of *well-behaved* or good nodes, V_b is a set of *misbehaving* or bad nodes, and E is a set of edges connecting nodes. V_b can be further divided into two subsets: one containing insiders and the other containing outsiders.

4.2.2 Direct Threats

An attack against a routing protocol can be modeled in three phases: *pre-attack*, *during-attack*, and *post-attack*. Prior to an attack, a malicious party will try to gain control of as many legitimate routers as possible, and to introduce as many outside misbehaving routers as possible. In other words, an attacker may try to increase the number of both inside and outside misbehaving routers, and use them to launch attacks. To gain control of a legitimate router, an attacker can exploit software vulnerabilities and operating system weaknesses, or influence an authorized owner of that router (e.g., by social engineering), among other techniques. These threats are not considered direct threats against a routing protocol since they do not directly exploit any routing protocol vulnerability. Besides, other protocols or applications running on the router can also be affected by these threats. Introducing outside misbehaving routers into a routing infrastructure usually involves exploiting routing protocol neighbor authentication weakness, thus is considered to be a direct threat.

After an attacker controls one or more misbehaving routers (inside or outside), he may start to launch attacks against a routing protocol to cause a router to misbehave. Recall the operation of a DV routing protocol. A router receives inputs (i.e., routing updates) from direct neighbors, processes them along with local configurations based on routing protocol specifications, and produces outputs (e.g., routing updates) to be further propagated to other routers. Thus, for a router to operate properly, the following conditions must be met:

1. the routing protocol implementation running on the router must conform to the routing protocol specifications;
2. local configurations must be correct – in other words, the router must not be misconfigured; and
3. inputs received from other routers must be correct.

An event resulting in the violation of any of the above conditions can cause a router to misbehave or to operate improperly. Thus, an attacker can perform the following malicious actions. First, it can manipulate a routing protocol implementation to cause deviation from its specifications. Second, it can misconfigure a router with erroneous data. Third, it can directly manipulate outputs of the router, which are to be further propagated. For example, it can modify, delete, insert, or replay routing updates. All of the above attacks will result in erroneous routing updates to be propagated to other routers.

A routing protocol may use the same database for both packet forwarding and routing update advertisements, or use a separate one for each purpose. In the former case, all three types of attacks have the same consequences, i.e., a misbehaving router will misforward packets based on incorrect routing tables, and advertise erroneous routing updates to other nodes. In the latter case, manipulation of routing updates to be advertised to other nodes does not affect the local routing table since they are separate from each other.

After the above attacks, erroneous routing updates will propagate from a misbehaving router to other misbehaving or well-behaved routers. A well-behaved router using an erroneous routing update as input to its routing protocol operation will end up with an incorrect routing table, and thus will misroute user packets. In addition, it will also propagate incorrect routing updates to other nodes. Example consequences include denial-of-service, i.e., user packets cannot reach their ultimate destinations, and loss of confidentiality and data integrity, i.e., user packets are misrouted to a location in the control of an adversary. If the objective of an attacker is to cause denial-of-service or disrupting routing operations, such an objective has been achieved. If there are other objectives, e.g., traffic interception, additional actions need to be taken. Further discussion is beyond the scope of this thesis.

4.2.3 Indirect Threats

A routing protocol is also vulnerable to threats against its underlying protocols. For example, BGP is based on TCP, and thus vulnerable to all threats against TCP (e.g., TCP reset attacks [142]). These threats can be better addressed in the corresponding protocols, and are not considered in this thesis. However, some of these threats can be mitigated by the proposed cryptographic mechanisms for countering direct threats against routing protocols.

4.3 Security Goals for Routing Protocols

Based on the above threat model, we propose a set of generic security goals that should be achieved by a serious proposal for securing routing protocols, particularly based on a vectoring approach. These security goals address direct threats to routing protocols, but leave most indirect threats not addressed. Thus, they should not be considered sufficient.

1. *Data Origin Authentication*. A router should be able to authenticate that the sender of a routing update is the entity whom it claims to be. This is a fundamental security goal for most distributed systems, and it holds for routing protocols as well. It aims to prevent entity spoofing. In addition, it provides data integrity.

2. *Data Integrity*. A router should be able to authenticate that a received routing update has not been tampered with. This is also a fundamental security goal for most distributed systems including routing protocols. It aims to prevent unauthorized modification of routing updates. While data origin authentication provides data integrity, it is presented here as a separate routing security goal to emphasize its importance.

3. *Data Truthfulness*. A router should be able to verify the truthfulness of the information carried in a routing update. More specifically, a routing update contains a number of fields, each of which or in combination with others can be interpreted as an assertion of a particular aspect of a route for reaching a destination. A router should be able to verify the truthfulness of such assertions. This goal aims to prevent a router from accepting false routing information, ensure a router's continuous proper functioning, and stop fraudulent routing updates from spreading.

4. *Neighbor Authentication*. It is also desirable that a router is able to verify that another router requesting to establish a neighbor relationship or sending it routing updates is authorized to do so. This goal aims to prevent an unauthorized entity from participating in routing protocol operations, thus reducing the total number of misbehaving routers that can be controlled by an adversary.

4.4 Security Mechanisms for Routing Protocols

Different techniques are used in this thesis for achieving each of the above security goals (cf. §4.3). We begin with a brief summary of how to achieve the first three goals, followed by a discussion on how to use information corroboration for achieving the fourth goal, i.e., data truthfulness verification.

4.4.1 Cryptographic Mechanisms

Cryptographic mechanisms can be used to achieve data origin authentication and data integrity. For example, if two parties share a secret key, Message Authentication Codes (MAC) [87, p.111] can achieve data origin authentication which includes data integrity. If each party holds a public and private key pair (and can verify the authenticity of the public keys of others), digital signatures can achieve both. The remaining challenge is to choose a proper cryptographic mechanism which is suitable to a particular routing protocol with its practical constraints.

To achieve neighbor authentication, it appears sufficient to apply suitable cryptographic mechanisms for data origin authentication in conjunction with certain prior knowledge of the topology of the network which is to be secured. For example, an intra-domain routing protocol runs on a network within an administrative domain. The personnel responsible for the network management usually design, deploy, and maintain the routing infrastructure. Thus, they have prior knowledge of which router connects to which. Such knowledge can be distributed to routers to prevent them from establishing neighbor relationships with unauthorized devices. In an inter-domain routing protocol, connections between ASes are primarily determined by business agreements. In other words, an AS is allowed to directly connect to another AS only when there is a business agreement between them (see §2.1.3 for AS business relationships). Thus, each AS has sufficient knowledge of which other ASes are neighbors. Other techniques (e.g., [42]) can also be used to detect neighbor spoofing. For example, two BGP speakers with a direct BGP session are usually located within one or two hops. By restricting the Time To Live (TTL) field (which has a maximum value of 255) in received packets to certain values (e.g., 253 or higher), a BGP speaker is able to detect and drop malicious packets (e.g., requests for establishing a neighbor relationship) originated from hosts three or more hops away. Such technique can significantly reduce the risk for accepting malicious packets, albeit cannot remove them completely.

4.4.2 Information Corroboration

While cryptographic mechanisms are effective in data origin authentication and data integrity, they cannot guarantee data truthfulness. We believe that corroboration plays an important role in improving confidence in information when the trusted authority of that information is not available. Assume that Alice sends Bob a message. After authenticating the origin of the message, Bob next needs to assess its truthfulness. The assessment process can be modeled as a function, which takes a number of inputs, and produces an output, i.e., the message is true or false. (Figure 4.3).

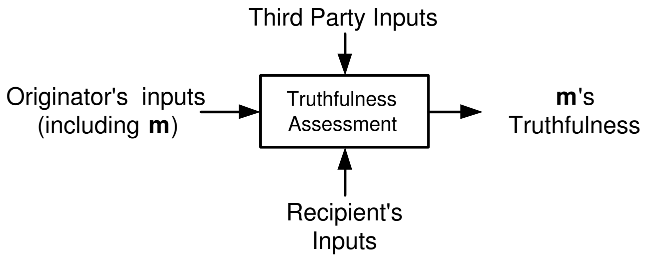


Figure 4.3: A trust model for assessing the truthfulness of a message m.

Parties involved in the assessment process include Alice, Bob, and third parties. Assume only one of them is authoritative of the message. In the case that the message contains multiple pieces of information each of which has its own authority, the message can be divided into smaller units each of which is assessed separately for its truthfulness. Many factors need to be considered in evaluating the truthfulness of a message. We classify them into three categories:

1. A set of inputs from the originator of the message, including the message itself and other supplementary information.

2. A set of inputs from the recipient of the message, including the recipient's knowledge of the information conveyed in the message, previous experience with the message originator, trust in the message originator, among others.

3. A set of inputs from third parties related to the message, including direct opinions of a third party in the message, or relevant information from a third party that is useful to the assessment (e.g., for cross consistency checks).

With these inputs, the truthfulness of the message can be determined as follows:

1. If the originator of the message is authoritative, the recipient trusts the message immediately, provided that the message integrity is guaranteed, i.e., it was not modified in transit after being originated by the actual originator. However, there remains some risk in fully trusting an authority since it is possible that the authority being fully trusted misbehaves.

2. If the recipient is authoritative on the message, an immediate decision can also be made regarding its truthfulness. This is ideal since the assessment can be done internally without the involvement of any third party. While each entity can try to acquire as much information as possible to become more authoritative, it is usually the role played by an entity that determines its authority.

3. If a third party is authoritative on the message, the recipient should consult with the third-party authority regarding the truthfulness of the message. Both *polling* and *pushing* modes can be used for obtaining the opinion of the authority. In the polling mode, the recipient solicits the authority's opinion when required (i.e., on-demand). In the pushing mode, the authority voluntarily makes its opinion available to the recipient and others in a manner guaranteeing data origin authentication.

4. If the message authority is not available, the recipient needs to consult with entities who might have certain albeit incomplete knowledge regarding the message. Since none of them is authoritative on the message, their opinions should be corroborated to form a conclusion on the truthfulness of the message. Information corroboration is an important technique here due to the fact that in many cases, a trusted authority either does not exist or it is impractical to contact the authority in real-time.

5. If no entity is authoritative or has partial knowledge on the message, the recipient may resort to relying on its local knowledge to determine the trustworthiness of the message.

The output of the assessment procedure is usually a binary decision that the message is either true or false. To achieve more flexibility, we use a numeric value to represent the trust degree or the *belief* in a message (see §4.5). The higher the value, the stronger the perceived trustworthiness of the message. Without losing generality, we normalize the value representing the belief in a message to $[0, 1]$. In conjunction with local parameters (e.g., trust thresholds), a belief in a message can be used to determine whether or not a message is trustworthy.

4.5 A Rating Mechanism

We propose to use a rating mechanism for measuring the trustworthiness of nodes in a network. Each node i assigns a numerical value in the range $[0, 1]$ to every other node j in a routing domain, denoted by $r_i(j)$, representing i's confidence in j's trustworthiness, e.g., in the truthfulness of a routing update from j. The higher $r_i(j)$ is, the greater confidence i has in j's trustworthiness. $r_i(j)$=0 and 1 represents that i fully distrusts and trusts j respectively. While any numeric value can be used for node rating, without loss of generality, we normalize it to $[0, 1]$.

Node i's rating of j can be static or dynamic. In the former case, i is preconfigured with an array of ratings for other nodes in the network, and the rating array remains unchanged unless updated manually. In the latter case, a node is also preconfigured with a rating array, but will dynamically update the array based on certain rules and local parameters. The main advantage of dynamic ratings is that they can automatically reflect node behavior changes in the rating array. For example, a node continuously providing correct routing updates will be gradually rated higher, and one providing fraudulent information (which is detected as such) will be rated lower over time (but never reach 0 unless rated 0 initially). One disadvantage of this dynamic approach is that it introduces a new vulnerability that the rating of a well-behaved node might be decreased due to miscategorizing its routing information (e.g., caused by wrong accusation from malicious nodes). In contrast, static ratings do not have this vulnerability. However, they have the drawback that the nodes' recent behavior cannot be automatically reflected in the rating array.

4.5.1 Combined Confidence

To support corroboration, we next present a method [136] for computing the confidence value in a statement which is consistent among a set of assertions (see §5.3.4 and §6.5.1) made by a group of nodes (e.g., routers or ASes) (a *corroborating* group) based on one's ratings of those nodes.

Let $v_1, .., v_n$ be a group of nodes which independently produce a set of consistent assertions $a_{v_1}, .., a_{v_n}$. Let $\lambda_{v_1,..,v_n}$, abbreviated by $\lambda_{[1..n]}$, denote a common subset that can be derived from each of the above n consistent assertions. The precise meaning of $\lambda_{[1..n]}$ depends on the type of consistency in question (see §5.3.4 and §6.5.1). We next show how node v_i computes a confidence value or a belief in $\lambda_{[1..n]}$, denoted $b_i(\lambda_{[1..n]})$, based on v_i's ratings of $v_1, .., v_n$ in the corroborating group. By definition, v_i's rating of $v_j, 1 \leq j \leq n$, represents v_i's confidence in the assertion a_{v_j} made by v_j or a subset λ_{v_j} derived from a_{v_j}, i.e., $b_i(\lambda_{v_j}) = b_i(a_{v_j}) \triangleq r_i(v_j)$. Where there is no ambiguity, we omit the subscript on r in $r_i(v_j)$, and on b in $b(\lambda_{[1..n]})$. We define $\oplus$ as the operator for combining confidence in two consistent assertions. $b(\lambda_{[1..n]})$ is defined as:

$$b(\lambda_{[1..n]}) = b(\lambda_{v_1}) \oplus .. \oplus b(\lambda_{v_n}) = \begin{cases} r(v_1) & \text{if } n{=}1 \\ r(v_2) + \big[1 - r(v_2)\big] \cdot b(\lambda_{v_1}) & \text{if } n{=}2 \\ r(v_n) + \big[1 - r(v_n)\big] \cdot b(\lambda_{[1..(n-1)]}) & \text{if } n{\geq}3 \end{cases} \tag{4.1}$$

The rationale behind equation (4.1) is that one's confidence in the correctness of an assertion increases when the number of nodes endorsing that assertion increases. The amount by which the confidence increases depends on the remaining confidence that can be increased and the trustworthiness of an endorsing node. For a given corroborating group, equation (4.1) has the following properties:

1. endorsement from a fully distrusted node (i.e., one rated 0) does not increase one's confidence;

2. endorsement from a fully trusted node (i.e., one rated 1) increases one's confidence to a maximum value (i.e., 1);

3. if there are multiple nodes in the corroborating group, none of which is fully trusted but two or more are marginally trusted (i.e., rated with a value in $(0, 1)$), one's confidence increases but never reaches maximum.

4. the order of the nodes in the corroborating group has no effect in the final confidence. In other words, equation (4.1) is commutative and associative.

These properties are intuitive, and serve well the purpose of information corroboration. For example, the property of commutativity and associativity ensures that a corroborated confidence is independent of the order of information gathering. Although developed independently based on our intuition, equation (4.1) is consistent with Dempster-Shafer theory of evidence reasoning [33, 121] provided that for each i ($1 \le i \le n$), v_i's assertion is independent of each other (see §4.5.2 for a sketched proof). The advantage of equation (4.1) is that it is intuitive and computationally efficient. Although Dempster-Shafer theory is more general (recall §2.4), e.g., it can handle conflicting information, it is computationally less efficient since it involves set operations. Other methods (e.g., [108, 64]) might also be applicable to information corroboration. However, they are usually proposed for other applications (e.g., trust in public key certificates) which have their own particular requirements, and can bring additional complexities which in turn reduce efficiency.

4.5.2 Consistency of Equation (4.1) and DST

Here we present a proof sketch that equation (4.1) is consistent with DST. Let λ be an assertion, $\bar{\lambda}$ be the complement of λ (i.e., we assume λ is an assertion which has a negator), and $U = \{\lambda, \bar{\lambda}\}$. For a corroborating group $(v_1, .., v_n)$ which agree upon λ, we assume that each assertion made by $v_1, .., v_n$ is independent of one another. Thus, each such assertion can be considered as a piece of independent evidence from the asserting node for supporting λ. Node i's rating of v_j, $1 \le j \le n$, represents the strength of v_j's assertion in supporting λ. In DST, $r_i(v_j)$ represents the basic probability assignment of λ based on the evidence from v_j, denoted by $m_j(\lambda)$. Since there is no evidence from v_j supporting $\bar{\lambda}$, $m_j(\bar{\lambda}) = 0$. According to DST (recall §2.4.1), $m_j(\lambda, \bar{\lambda}) = 1 - r_i(v_j)$. Thus, node i has n basic probability assignments for U, which are listed in Table 4.1. For simplicity, we omit the subscript i in

$r_i(v_j)$, and the set brackets {} in the basic probability assignments and belief functions of $\{\lambda\}, \{\bar{\lambda}\}$, and $\{\lambda, \bar{\lambda}\}$. Other notation used in the rest of this section is given in Table 4.2.

basic probability assignment	ϕ	$\{\lambda\}$	$\{\bar{\lambda}\}$	$\{\lambda, \bar{\lambda}\}$
m_1	0	$r(v_1)$	0	$1 - r(v_1)$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
m_n	0	$r(v_n)$	0	$1 - r(v_n)$

Table 4.1: Basic probability assignments for U=$\{\lambda, \bar{\lambda}\}$

$m_{[1..k]}$	the basic probability assignment combined from $m_1, .., m_k$
$Bel_{[1..k]}$	the belief function based on the basic probability assignment $m_{[1..k]}$

Table 4.2: Notation for Dempster-Shafer theory

We next show the combined confidence $b(\lambda_{[1..n]})$ obtained from the corroborating group $\{v_1, .., v_n\}$ by equation (4.1) is consistent with the combined belief $Bel_{[1..n]}(\lambda)$ computed from n independent basic probability assignments by DST. Since the subsets of $\{\lambda\}$ include itself and the empty set, we can obtain

$$Bel_{[1..n]}(\lambda) = \sum_{X \subseteq \{\lambda\}} m_{[1..n]}(X) = m_{[1..n]}(\lambda) + m_{[1..n]}(\phi) = m_{[1..n]}(\lambda)$$

we next show that $m_{[1..n]}(\lambda)$=$b(\lambda_{[1..n]})$ using induction.

1. When n=1, we know from Table 4.1 that $m_1(\lambda)$=$r(v_1)$=$b(\lambda_1)$.

2. Assume when n=k, $m_{[1..k]}(\lambda)$=$b(\lambda_{[1..k]})$. Since no one in the corroborating group $\{v_1, .., v_k\}$ supports $\bar{\lambda}$, we obtain $m_{[1..k]}(\bar{\lambda})$=0, and $m_{[1..k]}(\lambda, \bar{\lambda})$=1$-b(\lambda_{[1..k]})$.

3. When n=k+1, $m_{[1..k+1]}(\lambda)$ can be computed from the basic probability assignments $m_{[1..k]}$ and m_{k+1} using equation (2.1) as follows:

$$\begin{aligned}
m_{[1..k+1]}(\lambda) &= \frac{\sum_{X\cap Y=\lambda} m_{[1..k]}(X)\cdot m_{k+1}(Y)}{1-\sum_{X\cap Y=\phi} m_{[1..k]}(X)\cdot m_{k+1}(Y)} \\
&= \frac{m_{[1..k]}(\lambda)\cdot m_{k+1}(\lambda) + m_{[1..k]}(\lambda)\cdot m_{k+1}(\lambda,\bar{\lambda}) + m_{[1..k]}(\lambda,\bar{\lambda})\cdot m_{k+1}(\lambda)}{1-\left[m_{[1..k]}(\lambda)\cdot m_{k+1}(\bar{\lambda}) + m_{[1..k]}(\bar{\lambda})\cdot m_{k+1}(\lambda)\right]} \\
&= \frac{b(\lambda_{[1..k]})\cdot r(v_{k+1}) + b(\lambda_{[1..k]})\cdot\left[1-r(v_{k+1})\right] + \left[1-b(\lambda_{[1..k]})\right]\cdot r(v_{k+1})}{1-\left[b(\lambda_{[1..k]})\cdot 0 + 0\cdot r(v_{k+1})\right]} \\
&= b(\lambda_{[1..k]}) + \left[1-b(\lambda_{[1..k]})\right] r(v_{k+1}) = b(\lambda_{[1..k+1]})
\end{aligned}$$

The induction proof establishes that $b(\lambda_{[1..n]})$ calculated using equation (4.1) based on the ratings of $\{v_1, .., v_n\}$ in the collaborating group is consistent with the combined belief in λ computed using DST from n independent basic probability assignments.

4.6 Summary

In this chapter, we presented a framework for securing Internet routing protocols, which consists of a threat model, several security goals for routing protocols, and a number of security mechanisms based on cryptography and information corroboration. While cryptographic techniques are effective in data origin authentication and data integrity, they cannot guarantee data truthfulness. Information corroboration is commonly utilized by human beings for acquiring trust in information. We believe it can be adopted into routing protocols for detecting fraudulent routing updates, and improving routing security. To facilitate information corroboration in routing protocols, we proposed a rating mechanism in which each node rates every other node within the routing domain with a numeric value in $[0, 1]$, representing one's confidence in the correctness of a routing update from that node. A simple and intuitive method is proposed for computing combined confidence in a routing update that is endorsed by one or more other nodes.

The framework presented here is generic, and will be instantiated by a particular routing security proposal. In Chapters 5 and 6, we respectively present S-RIP for securing RIP, and psBGP for securing BGP, both of which are based on this framework. However, S-RIP and psBGP instantiated the framework differently due to the protocol and operational difference between RIP and BGP.

Chapter 5

S-RIP: A Secure Distance Vector Routing Protocol

5.1 Introduction

In this chapter, we propose a secure distance vector routing protocol, namely S-RIP, based on RIP [83], which can prevent router spoofing, prefix hijacking, and distance fraud (see §5.2). In S-RIP, a received route advertisement is verified for its factual correctness before being used to update a routing table. In the absence of an online trusted authority that has perfect knowledge of network topology and dynamics, we propose to use *consistency* as an approximation of *correctness* by corroborating information from multiple sources. An advertised route is treated as correct if it is consistent among those nodes that have propagated that route. Unless all nodes in the corroborating group, i.e., involved in a consistency check, are in collusion, a consistent route is likely correct with a high degree of confidence (see §5.6.1). By this approach, nodes surrounding a misbehaving node will likely uncover the inconsistency of misinformation and prevent it from further spreading.

We apply the rating mechanism presented in §4.5 to determine how many nodes to involve in a consistency check, providing the flexibility for balancing security and efficiency. Firstly, the notion of fully trusting or distrusting a node is replaced by node *rating* or *reputation* measured by a numeric value. Although in an intra-domain routing protocol such as RIP, routers are under a single administrative domain and tend not to be mutually suspicious, they could be compromised due to software flaws. Malicious nodes can also manage to join the routing operation by exploiting routing protocol vulnerabilities. Therefore, fully trusting any individual node even in an intra-domain routing protocol may introduce the vulnerability that a malicious node can poison the routing tables of many other well-behaved nodes. Node rating provides the flexibility to relax this notion, and can be interpreted as an estimation that a node will provide correct information in the near future. Secondly, we make use of the method for computing combined confidence (equation (4.1)) in the correctness of a consistent route from the ratings of those nodes involved in the consistency check. Combined with confidence thresholds, this method effectively creates a *sized window* for determining how many nodes to involve in a consistency check.

The rest of the chapter is organized as follows. §5.2 analyzes RIP security vulnerabilities and outlines five security goals for RIP. §5.3 summarizes security mechanisms of S-RIP. §5.4 presents a rating mechanism for S-RIP. Details of S-RIP are presented in §5.5. Security and efficiency of S-RIP are analyzed in §5.6. S-RIP simulation results are presented in §5.7. We conclude the chapter in §5.8.

5.2 RIP Vulnerability and Security Goals

In this section, we discuss RIP vulnerability and outline five security goals for improving RIP security.

5.2.1 RIP Vulnerability

RIP has several known security vulnerabilities. For example, RIP has a weak neighbor authentication mechanism, and it does not have any mechanism for preventing a *questionable node* (an unauthorized node, or a compromised or malicious legitimate node) from advertising fraudulent routing information about route distance and next hop. We next summarize them into five categories.

1) [**Neighbor Spoofing**] An unauthorized node can easily participate in RIP operation by exploiting RIP neighbor authentication vulnerability. We refer to such exploitation as *neighbor spoofing*. RIPv1 [49] recommends that each router is configured with a list of

neighbors which are authorized to send routing advertisements to this router. Routing advertisements from a host not on the neighbor list are ignored by a router. However, RIPv1 does not have any data origin authentication mechanism. Thus a malicious host can spoof the IP address of an authorized neighbor, and thus can successfully advertise routes to a legitimate RIP router. RIPv2 [83] only uses a clear-text password for authenticating routers. Since a clear-text password can be easily captured, it provides only marginal additional security in practice. Keyed MD5 has been proposed [8] to replace the password-based authentication mechanism. However, it uses a system-wide shared secret key, and thus is still vulnerable in that one compromised router discloses keying materials of every other router in the network.

2) [**Prefix Hijacking**] A questionable node can claim a zero distance to a network which either does not exist or is not directly connected to the claimant. In other words, a questionable node can originate a direct route for a network or a block of IP addresses which it is not authorized to originate (for example, for a block of IP addresses assigned to another router, not the claimant). This is often referred to as *prefix hijacking*. The proposed MD5 authentication [8] with system-wide shared secret key(s) makes neighbor spoofing difficult, but cannot prevent prefix hijacking. Although a bigger issue in inter-domain routing protocol (e.g., BGP), prefix hijacking can also cause serious problems in an intra-domain routing protocol such as RIP. Figure 5.1 shows that a malicious node can easily cause service disruption by prefix hijacking. A similar incident (referred to as a *blackhole*) occurred in 1978 in the ARPANET [86]. With the control of another malicious node, an attacker can mount more sophisticated attacks (e.g., man-in-the-middle or session hijacking) without being detected.

3) [**Short Distance Fraud**] A questionable node may claim a distance shorter than the actual distance to a destination, which is often referred to as *shorter distance fraud*. This fraud can be used to attract traffic allowing a variety of attacks (e.g., eavesdropping, session hijacking).

4) [**Longer Distance Fraud**] A questionable node can claim a distance longer than the actual distance to a destination, which is referred to as *longer distance fraud*. This can be used to avoid carrying traffic, which may lead to unfair utilization of network bandwidth,

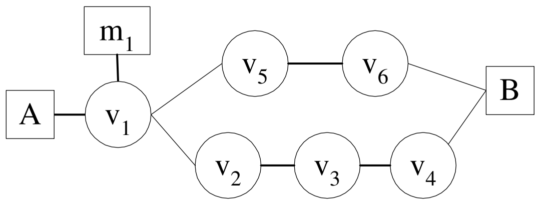


Figure 5.1: m_1 advertises a zero distance route for B. As a result, v_1's routing table is poisoned by an incorrect route for B. Traffic from A to B will be forwarded by v_1 to m_1, which causes service disruption against A since m_1 does not have a route to B other than the one via v_1.

cause network congestion, and result in denial of service. This fraud is different from malicious packet dropping attacks [65]. While they both result in packet dropping, the latter can be detected by known techniques (e.g., secure traceroute [99]), while the former is more stealthy.

5) [**Next Hop Fraud**] A questionable node may provide false information on next hops of advertised routes, to cause routing loops or unreachable routes installed in a recipient's routing table.

5.2.2 RIP Security Goals

To counter the above security vulnerabilities, we outline five security goals for RIP based on the generic goals for a routing protocol discussed in §4.3. These security goals are *necessary* in the sense that failure of any of them has serious security consequences. Thus, we believe that a serious RIP security proposal should meet all of them. However, they should *not* be considered *sufficient* since other threats against RIP remain. For example, security vulnerabilities of the protocols underlying RIP are not addressed by these security goals, nor is misbehavior in the data plane (e.g., packet dropping). We seek to design security mechanisms that can significantly improve RIP security, but do not attempt to pursue a perfect RIP security solution. In fact, we believe that perfect security solutions for real world complex systems either do not exist or are impractical.

G1. *(Router Authentication)* It should be verifiable that an entity claiming the identity of a RIP router (i.e., an IP address) is indeed assigned that identity by the authority of the RIP domain.

G2. *(Data Integrity)* It should be verifiable that a RIP control message (update request or response) has not been subject to unauthorized modification en route.

G3. *(Destination Authentication)* It should be verifiable that a router advertising a zero-hop route for a destination indeed has direct connectivity to that destination.

G4. *(Distance Authentication)* It should be verifiable that a router advertising a non-direct route for a destination that is reachable from a recipient router (i.e., a route with a distance of $1 \leq n \leq 14$ hops, hereafter referred to as a reachable route) is indeed n hops away from that destination.

G5. *(Next-hop Authentication)* It should be verifiable that a router u advertising a reachable route with a next hop v is indeed a direct neighbor of v.

G1 and G2 relate to data origin authentication and data integrity, and G3, G4 and G5 to the truthfulness of RIP routing updates. Threats against RIP can be committed by individual or colluding nodes. In this thesis, we mainly focus on threats by uncoordinated individuals. Collusion usually implies intent, which appears difficult to define and analyze.

5.3 Security Mechanisms of S-RIP

Here we introduce and summarize security mechanisms employed by S-RIP, including pairwise shared secret keys for router authentication, an authoritative router-prefix mapping for destination authentication, and consistency checks for route distance authentication.

5.3.1 Router Authentication in S-RIP (G1)

We require Assumption 1 (see next page) for achieving *router authentication*. To verify that an entity claiming to be a RIP router v_i is indeed v_i, a router v_j can ask the claimant to demonstrate that it holds the unique secret key v_j shares with v_i. For example, v_j can verify the message authentication code (MAC) of a routing update message from v_i using an authentication algorithm (e.g., keyed MD5).

Assumption 1 (A1). *Every router shares a different key with every other router in a RIP domain.*

An advantage of pairwise shared keys is that compromising one router does not disclose the full keying materials of another. One major issue with pairwise shared keys is that they increase the complexity of key management. For example, whenever a router is added into a network, it must be preconfigured with a number of secret keys, one for each of the existing routers. In addition, a different secret key must also be distributed to each of the existing routers. This key configuration process introduces significant overhead and may result in configuration errors. However, a Key Distribution Center (KDC) [87, p.] based approach for key management is practical for intra-domain routing protocols, and automatic and secure configuration tools (e.g., for updating SNMP community strings) may also be useful for reducing the complexity of key management. Overall, we believe the assumption of pairwise shared keys is practical for intra-domain routing protocols, and is consistent with the strong recommendation of a group of security experts (see [32]).

Public key based methods (e.g., digital signatures) offer an attractive alternative in some regards, but also typically require a public key infrastructure (PKI) which has its own setup costs and comes at some additional cost in performance (which we expect will become less important over time). Taking into account that RIP is mainly used by small to medium size organizations, which may lack sufficient skill sets in managing a PKI, we do not recommend the use of PKI in S-RIP. At the current time, we believe the use of pairwise shared keys is realistic in practice, acknowledging that some additional price must be paid to make an insecure routing protocol secure.

5.3.2 Data Integrity in S-RIP (G2)

Pairwise shared secret keys and a message authentication algorithm such as Keyed-MD5 can offer data origin authentication which includes data integrity.

5.3.3 Destination Authentication in S-RIP (G3)

To provide destination authentication, we require Assumption 2 (see below). A router v_i is allowed to originate a zero distance route for a destination f only if v_i is directly connected to f as specified by the router-prefix mapping (see Algorithm 1). Note f represents a subnet if it specifies a block of IP addresses, or a host if it specifies a single IP address.

Assumption 2 (A2). *Each router within a RIP domain is configured with a router-prefix mapping (rpm):* $\{(F_1, v_1), .., (F_n, v_n)\}$*, where for* $1 \le i \le n$*,* F_i *denotes a set of IP prefixes each of which represents either a subnet or a host, and* v_i *denotes a router.* (F_i, v_i) *specifies that* v_i *is directly connected to each IP prefix* $f \in F_i$*.*

Algorithm 1 prefix_is_owned(f, v)

```
1: GLOBAL: the router-prefix mapping $rpm$
2: INPUT: an IP prefix $f$; a node $v$
3: OUTPUT: TRUE or FALSE
4: for each entry $(f_i, v_i) \in rpm$ do
5:     if $v_i = v$ and $f_i = f$ then
6:         Return(TRUE)
7: Return(FALSE)
```

A *router-prefix mapping* is realistic for an intra-domain routing protocol such as RIP since network configurations are administratively controlled by a single authority (e.g., a network administrator), which usually has perfect knowledge of the network configuration. Similar assumption is also required by some other approaches for improving RIP security. For example, Mittal and Vigna proposed a method [90] for detecting abnormal RIP advertisements, which requires a complete network topology including both router-prefix and router-router connections. Our router-prefix mapping consists of only router-prefix connections, and thus is a subset of their requirement.

The router-prefix mapping can be securely distributed to each router during network initialization, e.g., it can be pre-configured on each router. Ongoing updates (e.g., additions of subnets or routers) can then be done through a secure channel (e.g., SSH) between the central authority and each router. Although a network topology may be dynamic (e.g., caused by link failures), we expect a router-prefix mapping to be relatively static since addition and deletion of subnets usually occur far less frequently than link failures. Other alternatives can also be used to authenticate if a router is authorized to originate a direct route for a destination, for instance, *address attestation* in S-BGP [72], and *authorization certificates* in soBGP [144]. However, they may require a public key infrastructure, which has its own drawbacks and thus is not recommended for S-RIP.

5.3.4 Distance Authentication in S-RIP (G4)

Here we define the notation used in the rest of the chapter. We use $[f, dist(v_1, f), nh(v_1, f)]_{v_1}$ to denote a route advertised by v_1 for a destination f, where $dist(v_1, f)$ represents the distance from v_1 to f, and $nh(v_1, f)$ represents the next hop from v_1 to f. When there is no ambiguity, we omit the subscript v_1 in $[f, dist(v_1, f), nh(v_1, f)]_{v_1}$.

In a distance vector routing protocol such as RIP, the distance of one route is based on the distance of another. Unless a node has perfect knowledge of network topology and dynamics, it appears difficult, if not impossible, to verify the factual correctness of route distance which is the aggregated result of some other routes [103, 141].

We propose to use *consistency* as an approximation of correctness by corroborating information from multiple nodes. The distance of an advertised route is validated by cross checking its consistency with the distances of some other routes from which this route is derived. If the distance of one route is consistent with the distance of a sufficient number of other such routes, either directly (per Definition 1 below) or transitively (per Definition 2 below), it is treated as correct. Otherwise, it is incorrect. For simplicity, we say two routes are consistent if their distances are directly or transitively consistent (per Definitions 1 and 2).

Definition 1. *(***Direct Consistency***) Consider two routes* $p_1{=}[f, dist(v_1, f), nh(v_1, f)]_{v_1}$, *and* $p_2{=}[f, dist(v_2, f), nh(v_2, f)]_{v_2}$. *We say that* p_1 *is* **directly consistent** *with* p_2 *if* p_1 *is computed directly based on* p_2 *by following RIP specification, i.e.,* $nh(v_1, f){=}v_2$*; and* $dist(v_1, f){=}dist(v_2, f){+}1$.

An example of our corroboration technique is as follows. In Figure 5.2, when node v_2 advertises to v_1 a 2-hop route for $192.168/16$ with v_3 as the next hop, v_1 queries v_3's

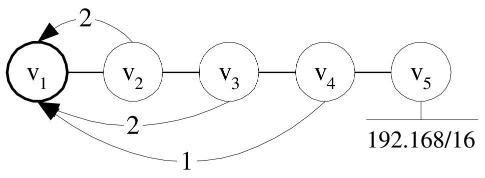


Figure 5.2: An example of consistency checks

route for $192.168/16$, which is 2 hops away. Since v_2's route for $192.168/16$ is supposed to be one hop longer than v_3's route for $192.168/16$ (this is specifically based on RIP, but can be easily generalized), an inconsistency is detected. Although v_1 does not know which node (v_2 or v_3) provides invalid information, it is clear that something is abnormal with this route. Therefore, it will not be used by v_1 to update its routing table. If v_2 advertises a 3-hop route for $192.168/16$, it is consistent with v_3's 2-hop route. Thus, it may be accepted. However, it is possible that v_2 and v_3 are in collusion, or v_3 fails to detect the fraudulent route from v_4 (e.g., v_3 fully trusts v_4). Therefore, v_1 may need to consult with additional nodes (see Definition 2 for consistency among multiple routes). §5.5 presents the algorithm details for S-RIP including consistency checks.

Definition 2. *(***Transitive Consistency** *) Consider $k{\geq}3$ routes $p_1, p_2, .., p_k$. We say that p_1 is* **transitively consistent** *with p_k if for $1{\leq}i{\leq}k{-}1$, p_i is directly consistent with p_{i+1}.*

To support consistency checks, we require Assumption 3 (see below). For example, in Figure 5.2, v_2 should inform v_1 that v_3 is the next hop of its route for $192.168/16$. Upon request, v_3 should also inform v_1 that v_4 is the next hop for $192.168/16$. Otherwise, its behavior is called into question.

Assumption 3 (A3). *Each router indicates the next hop of each route in its routing table, either voluntarily for direct neighbors or upon request to others.*

One property of a DV routing protocol is that a node only communicates with its direct neighbors and does not need to maintain the network topology beyond its direct neighbors. In an LS routing protocol, a node advertises the states of its attached network links to every other node in the network by flooding, and each node maintains a complete view of the network topology. A3 allows a node to query non-direct neighbors, which expands to a dynamic area the neighbor-to-neighbor communication boundary in a DV routing protocol (by the rating mechanism in §5.4).

Thus, we note that our approach falls in between the DV and LS approaches. Pictorially, the communication range of an LS node covers the whole network (flooding), while the communication range of a traditional DV node only covers its direct neighbors (neighbor-to-neighbor). In S-RIP, the communication range of a node is dynamic. Although it is certainly beyond direct neighborhood and could reach the whole network, most likely, it will only cover a nearby neighborhood (e.g., within 2 or 3 hops) dependent on window size (§5.4.3). Therefore, additional routing overhead generated by non-neighbor querying is limited, as confirmed by our simulation results in §5.7. Requirement of storage space is also increased in S-RIP, but very slightly since an S-RIP node only needs to maintain the information of remote nodes when they are either being or will be consulted for a consistency check.

Another question which arises is: how does a node query a remote node if it does not have a known route for that node? For example, in Figure 5.2, for v_1 to validate a route for v_3, v_1 may need to query v_3. However, v_1 cannot communicate with v_3 if it does not have a route for v_3. This is a known problem that a secure routing protocol relies upon a routing protocol for node reachability. In S-RIP, a temporary routing table is maintained, which contains all received routes to be validated. In other words, it is maintained based on received RIP routing updates. However, the temporary routing table is only used for route validation (not for forwarding data traffic). When a route passes a validation, it is moved to the regular routing table and can be used for forwarding data traffic. In the above example, v_1 first installs in its temporary routing table the route for v_3 (with v_2 as the next hop – v_1 and v_2 are directly connected), based on the route it received from v_2 (with v_3 as the next hop from v_2 to $192.168/16$). v_1 sends to v_2 a routing request destined for v_3. v_2 should have a direct route for v_3 since it advertises to v_1 that v_3 is the next hop to $192.168/16$. Otherwise, it is misbehaving. When v_3 receives a route request from v_1, it sends back to v_1 a route response via a route either in its temporary routing table or the regular one. This route request and response process incurs additional routing overhead, but adds another level of assurance that intermediate nodes are actually forwarding packets. If we can make a route request or response message indistinguishable from a normal data packet (e.g., by IPsec ESP [70]), this process may detect misbehavior in the data plane (i.e., a router advertising correct routes but not forwarding data packets).

To implement A3 in RIP, the next hop field in a RIP routing update message can be utilized. In RIP, the next hop field is currently used only for route optimization (avoiding an extra hop). For example, v_2 will not include v_3 in the next hop field (i.e., will set it to 0) unless it believes that v_1 should forward traffic destined for $192.168/16$ directly to v_3. With A3, v_2 voluntarily includes v_3 in the next hop. This changes the meaning of a next hop from *this is your next hop* to *this is my next hop*. Thus, A3 allows a receiving node, instead of an advertising node, to decide which node should be the next hop. Despite the change of the meaning, A3 is still compatible with RIP since a receiving node will ignore the next hop field (treating it as null) if it is not directly reachable. To interoperate with an existing implementation of RIP, an S-RIP node may get next hop information from a RIP node by external mechanisms, e.g., SNMP MIB query [25].

Besides route optimization, A3 allows a router to construct a complete path to a destination if it chooses to do so, i.e., by repeatedly sending a route request to each next hop on the route until the ultimate destination (if it is a RIP router) or the node directly connected to the ultimate destination (if it is a subnet) has been queried. This is useful for diagnosis and detection of misconfiguration, e.g., routing loops or malicious packet dropping [99].

5.3.5 Next-hop Authentication in S-RIP (G5)

To authenticate the next hop, say v_2, of a received route $p_1{=}[f, dist(v_1, f), nh(v_1, f)]_{v_1}$ in S-RIP, v_0 performs the following verifications: 1) v_2 is a legitimate RIP router, i.e., v_0 shares a secret key with v_2; 2) v_2 reports to v_0 that v_1 is the next hop of the route from v_2 to v_1, i.e., $nh(v_2, v_1){=}v_1$; and 3) the distance from v_2 to v_1 is 1 hop, i.e., $dist(v_2, v_1){=}1$.

5.3.6 Interacting with Non-RIP Domain in S-RIP

Let v_0 receive a route $p_1{=}[f, dist(v_1, f), nh(v_1, f)]_{v_1}$, and $v_2{=}nh(v_1, f)$. If v_0 does not share a secret key with v_2 but there is an entry (f, v_2) in the router-prefix mapping (i.e., Algorithm_1 – prefix_is_owned(f, v_2) returns TRUE), v_2 is considered as a legitimate non-RIP router. Such a router does not speak RIP, but usually runs another routing protocol (e.g., OSPF), and can function as a gateway from a RIP domain to a non-RIP domain. To authenticate the next hop which is a legitimate non-RIP router, we require the router-prefix mapping to include additional information of which RIP routers are direct neighbors of a non-RIP router, i.e., RIP-router to non-RIP router connections. Such information is further used to verify if a RIP router is allowed to advertise a route with a non-RIP router as the next hop. For the convenience of later reference, we define Algorithm 2, which is essentially equivalent to Algorithm 1.

Algorithm 2 router_is_neighbor(v, w)

```
1: GLOBAL: the router-prefix mapping $rpm$
2: INPUT: a RIP router $v$; a non-RIP router $w$
3: OUTPUT: TRUE or FALSE
4: for each entry $(v_i, w_i) \in rpm$ do
5:     if $v_i = v$ and $w_i = w$ then
6:         Return(TRUE)
7: Return(FALSE)
```

5.4 A Rating Mechanism for S-RIP

In this section we apply the rating mechanism proposed in §4.5 to S-RIP. We first present a simple method for automatically updating node ratings, then define localized rules for processing routing updates, followed by a sized window method for balancing security and efficiency in S-RIP.

5.4.1 Rating Update Function

Recall that we use the rating of a node as an estimation of the confidence that this node will provide correct routing information in the near future. In an intra-domain routing protocol such as RIP, there should be no policy bias regarding the trustworthiness of routers within the domain since they are under the same administration. In other words, all routers are equally trusted in principle. However, routers may behave differently over time for various reasons. We propose that every router rates every other router in its network domain equally at initialization, and then dynamically updates node ratings based on historical behavior using equation (5.1); here $r_i(j,t{+}1)$ represents i's rating of j at time $t{+}1$, and $c_i(j,t{+}1)$ represents a weight assigned by i to j at time $t{+}1$, representing i's evaluation of j's "trustworthiness" based on additional information gained in the most recent time period. Many possibilities exist for $c_i(j,t{+}1)$. We propose equation (5.2) for its simplicity.

$$r_i(j,t{+}1) = \frac{r_i(j,t)}{2} + c_i(j,t{+}1) \tag{5.1}$$

$$c_i(j,t) = \begin{cases} 0.5 & \text{if } j \text{ involved in a successful consistency check at time } t \\ 0 & \text{otherwise} \end{cases} \tag{5.2}$$

Properties of Equation (5.1)

One property of equation (5.1) is that if $r_i(j,t){\neq}1$, $r_i(j,t{+}1)$ will always be less than 1. Thus, if node i does not rate j by 1 initially, $r_i(j)$ will always be in the range $[0,1)$. Another property is that the rating of a node decreases dramatically if it provides incorrect information and is detected as such. The rating of a node increases gradually if it is detected as providing correct information. This property is intuitive and consistent with the way in which human beings rate one another.

5.4.2 Verification Rules

We propose using two thresholds (θ_1, θ_2) to divide the S-RIP rating domain into three categories: *low, medium,* and *high* (Figure 5.3). When applying the rating mechanism to S-RIP, rules are required to determine how to interact with nodes with different ratings. As an example, we develop the following rules for S-RIP for governing how to process routing advertisements based on node ratings.

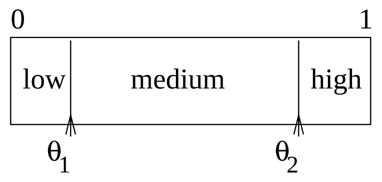


Figure 5.3: Rating is represented by a value in the range $[0, 1]$. The rating domain is divided by two thresholds into three categories.

Rule 1. *(***Low Rating***). If node i's rating of node j is in the low range ($0 \leq r_i(j) < \theta_1$), node i will ignore routing updates from j without cross-checking its consistency with any other node for a specified time period P_1.*

This rule can effectively mitigate potential denial of service attacks by a malicious node which may try to engage another node into a long period of validation by advertising large volumes of useless routing information. One disadvantage is that an untrusted node does not have chances to raise its reputation since all of its routing advertisements are dropped without validation. To overcome this shortcoming, we propose to timeout a node's low reputation and reassign it a medium rating value. This allows a node to raise its reputation after a specified time period P_1, which can be set to an appropriate value based on local information. For example, if it usually takes a half hour for an administrator to complete troubleshooting a problematic router, P_1 can be set to 30 minutes.

Rule 2. *(***Medium Rating***). If node i's rating of node j is in the medium range ($\theta_1 \leq r_i(j) < \theta_2$), node j is on probation, and node i will cross check the consistency of a routing update from j with other node(s).*

If every node initially assigns a medium rating to every other node, Rule 2 provides opportunities for a well-behaved node to raise its reputation into the high range by providing consistent routing information. The reputation of a misbehaving node can decrease into the low range if it provides misinformation and causes consistency check failures.

Rule 3. *(***High Rating***). If node i's rating of node j is in the high range ($\theta_2 \leq r_i(j) \leq 1$), node j is trusted by i for a specified time period P_2 without cross checking the consistency of a routing update from j.*

One disadvantage is that there is a potential risk that fraudulent information from a trusted node may be propagated since its consistency is not cross checked. To minimize such risk, we propose that a node can only maintain a high reputation for a specified period of time P_2, e.g., 5 minutes. After that period, its rating is reset to a medium value. The risk window of accepting malicious information from a misbehaving trusted node can be reduced by decreasing P_2. Another way to minimize this risk is to increase θ_2. The higher θ_2 is, the longer a node will take to raise its rating into the high level. In the extreme case, θ_2 can be set to 1 so that no node will be trusted during the course of operation unless it is rated 1 during initialization.

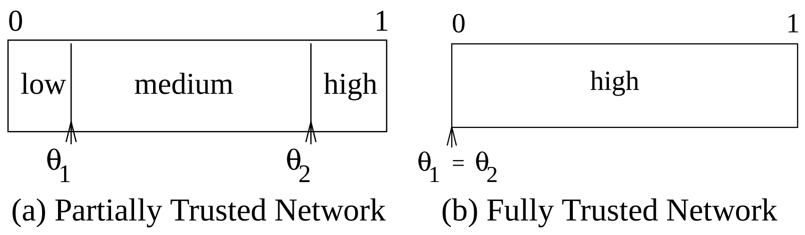


Figure 5.4: Moving θ_1 and θ_2 close to each other increases trust degree and decreases network overhead. The extreme case, where $\theta_1 = \theta_2 = 0$, emulates a network in which all routing information is fully trusted, which is in fact the assumption (at some risk) made for today's Internet.

We can emulate a trustworthy environment by setting both θ_1 and θ_2 to 0 (Figure 5.4(b)), where every node is trusted by every other node and no routing advertisements are verified. Verification overhead can be managed by adjusting the rating thresholds (Figure 5.4(a)). §5.7.3 shows by simulation that S-RIP verification overhead is relatively low in a partially trusted network, and can be balanced against security, e.g., by moving two thresholds toward each other.

5.4.3 Sized Windows

Since multiple nodes might have propagated a route in question, a mechanism is required to decide the number of nodes to involve in S-RIP route corroboration. The more nodes involved (which agree with the advertised route), the higher the confidence acquired in the correctness of that route; but the network overhead will also be higher since more messages will be transmitted over the network, and more processing overhead will result at network routing nodes.

S-RIP makes use of a *sized window* mechanism for balancing the trade-off between security and efficiency. The maximum size of the window is the total number of the nodes on the route to be corroborated. The window size starts from 1. In other words, there is only one node in the window before the consistency check starts, which is the advertiser of that route. The window size keeps growing until:

- an inconsistency occurs, i.e., a node reports conflicting information, in which case the route fails the consistency check; or
- the current corroborative confidence is at least θ_2, in which case, the route succeeds the consistency check; or
- all nodes on that route have been corroborated and no one reports conflicting information, in which case the route succeeds the consistency check regardless of the final corroborative confidence.

5.5 S-RIP Details

We now present the details of S-RIP. We respectively use v_0, v_1, and f to represent the recipient, the advertiser, and the ultimate destination of a RIP route.

When router v_0 receives from v_1 an advertised route $p_1{=}[f, dist(v_1, f), nh(v_1, f)]_{v_1}$, v_0 processes the route as required by RIP (see §2.1.1). If the route will be used to update v_0's routing table, S-RIP is triggered to perform additional validations. S-RIP will not be triggered if p_1 does not trigger a route change (in normal RIP). Although some timer (e.g., 180-second route expiration timer) associated with p_1 will be re-initialized, there is no need for S-RIP to verify p_1 at this point since such verification should have been done by S-RIP when v_0 received p_1 the first time. In the sequel we elaborate how v_0 verifies $p_1{=}[f, dist(v_1, f), nh(v_1, f)]_{v_1}$ in S-RIP, including router legitimacy checks, destination authentication, consistency checks, and handling of infinite route. The full algorithm is given in Algorithm 4.

Initial Checks (Algorithm 4:lines 4–7). v_0 verifies the legitimacy of the advertiser v_1. If v_0 does not share a secret key with v_1, then v_1 is not a legitimate RIP router. In this case, p_1 is rejected. v_0 next verifies the trustworthiness of v_1. If its reputation is in low range, i.e., $r(v_1){<}\theta_1$, p_1 is rejected. If v_1 has a high reputation, i.e., $r(v_1){\geq}\theta_2$, p_1 is accepted.

Destination Authentication (Algorithm 4:lines 11–16). If $dist(v_1, f){=}0$, p_1 represents a route for a subnet which is directly attached to v_1 (or owned by v_1). The router-prefix mapping (see §5.3.3) is used to verify that v_1 is indeed directly connected to that subnet (see Algorithm 1). If the verification succeeds, p_1 is accepted. Otherwise, it is rejected.

Authentication of Next Hop (Non-RIP Router) (Algorithm 4:lines 20–25). If $v_2{=}nh(v_1, f)$ is a legitimate non-RIP router, then v_1 must directly connect with v_2. If v_1 is indeed a direct neighbor of v_2, and f is directly connected to v_2 (according to the router-prefix mapping), p_1 will be accepted. Otherwise, it is rejected. Router v_2 is verified as a legitimate non-RIP router if v_0 does not share a secret key with v_2, but the router-prefix mapping consists of an entry for v_2. If v_2 is a legitimate RIP router, the consistency check of p_1 follows.

Consistency Checks (Algorithm 4:lines 26–37). If $1{\leq}dist(v_1, f){\leq}14$, p_1 represents a route for f that is *reachable* from v_0. In this case, v_0 will check the consistency of p_1 with the route for f from $v_2{=}nh(v_1, f)$. To do so, v_0 will request from v_2 its routes for f (denoted by p_2) and for v_1 (denoted by p_2'). The message flows are given in Table 5.1, where * denotes an information field to be provided. v_0 first checks if v_1 is a direct neighbor of v_2. In RIP, the distance between two direct neighbors is 1 hop. Thus, if $dist(v_2, v_1){=}1$ and $nh(v_2, v_1){=}v_1$, the verification succeeds. Otherwise, it fails. v_0 next checks the distance consistency of p_1 and p_2. According to Definition 1 (§5.3.4), p_1 is consistent with p_2 if $dist(v_1, f){=}dist(v_2, f){+}1$. Otherwise, they are inconsistent.

$v_0 \rightarrow v_2$	$[f, *, *]$ $[v_1, *, *]$
$v_0 \leftarrow v_2$	$[f, dist(v_2, f), nh(v_2, f)]$ $[v_1, dist(v_2, v_1), nh(v_2, v_1)]$

Table 5.1: Route request and response in S-RIP

If p_1 is consistent with p_2, v_0 uses equation (4.1) to compute a combined confidence $b(\lambda_{v_1,v_2})$. If it is at least θ_2, v_0 accepts the received route from v_1. Otherwise, the consistency check continues until one of the following conditions holds:

(a) Node $v_k\,(2{\le}k{\le}n)$ had been involved in this consistency check before, in which case, a routing loop is detected, and thus v_0 rejects p_1 (Algorithm 4: lines 18–19).

(b) The consistency check with $v_k\,(2{\le}k{\le}n)$ fails, in which case, v_0 rejects p_1 (Algorithm 4: lines 29–32).

(c) The combined confidence in p_1 is at least θ_2, i.e., $b(\lambda_{[1..k]}){\ge}\theta_2$, in which case, v_0 accepts p_1 (Algorithm 4: lines 33–35).

(d) $b(\lambda_{[1..k]}){<}\theta_2$, and $k{=}n$. In this case, v_0 performs destination authentication for f using the router-prefix mapping. If f succeeds the authentication, p_1 is accepted regardless of $b(\lambda_{[1..n]})$. Otherwise, the route is dropped. (Algorithm 4: lines 11–16).

After a consistency check, the ratings of all nodes involved in the consistency check will be updated using Algorithm 3, based on the status of the consistency check. If it was successful, the rating of every node involved in the consistency check is increased. Otherwise, either the rating of the node failing a destination authentication is decreased or the rating of every node involved in this failed consistency check is decreased.

Algorithm 3 update_ratings(V, s) – by node x

```
1: GLOBAL: node $x$'s ratings $r(v_i)$ of other nodes
2: INPUT: a set of nodes $V$; an ACCEPT or REJECT indicator $s$
3: OUTPUT: updated $r(v_i)$
4: if $s$ = ACCEPT then
5:    for each $v_i \in V$ do
6:       $r(v_i) \leftarrow r(v_i)/2 + 0.5$
7: else
8:    for each $v_i \in V$ do
9:       $r(v_i) \leftarrow r(v_i)/2$
```

Infinite Route (Algorithm 4:lines 6–7). If $dist(v_1, f){\ge}15$, p_1 represents a route for f which is *unreachable* from v_0. Such a route is not be verified in S-RIP since it appears difficult to verify an unreachable route, and is trivial for a misbehaving v_1 to make a valid route unreachable. For example, v_1 can simply disable a network interface or drop packets destined to f. The consequence of such misbehavior is that v_0 drops the route to f via

v_1. As a result, v_0 will not forward packets destined to f through v_1. If this is the only route in the network from v_0 to f, v_0 will not be able to forward packets destined to f. Such a result seems equivalent to the result of a misbehaving node on this route dropping packets destined to f. In a network designed with redundancy to accommodate single point of failures, there would be an alternative route from v_0 to f not via v_1. Such a path will be discovered by v_0 in normal RIP operation, and verified by S-RIP. If it passes S-RIP verification, it can be used by v_0 to communicate with a device whose IP address within the address range specified by f, bypass misbehaving nodes on the route it received from v_1.

5.6 S-RIP Analysis

In this section, we present an analysis of S-RIP, including its security properties, efficiency, and deployment.

5.6.1 Security Analysis of S-RIP

Here we analyze security properties of S-RIP against the five security goals for RIP, which are outlined in §5.2.2.

Proposition 1 (Router Authentication – G1). *S-RIP achieves router authentication.*

Proof Outline. S-RIP uses pairwise shared keys with a Message Authentication Code (MAC) to protect the integrity of routing updates. Since every node shares a different key with every other node, a correct MAC of a message also indicates that the message is originated from the only other party which the recipient shares a secret key with. Thus, router authentication is achieved.

Proposition 2 (Data Integrity – G2). *S-RIP achieves data integrity.*

Proof Outline. S-RIP uses pairwise shared keys with a Message Authentication Code (MAC) to protect the integrity of routing updates. A routing update message with an invalid MAC can thus be detected.

Algorithm 4 accept_route(p, u) – S-RIP Algorithm (by node x)

```
1: GLOBAL: thresholds $\theta_1, \theta_2$; $x$'s ratings $r(v_i)$ of other nodes; $x$'s secret key table $K$
2: INPUT: route $p$=[$f$, $dist(u, f), nh(u, f)$]; node $u$, which advertising $p$
3: OUTPUT: ACCEPT or REJECT $p$
4: if $K[u] = \phi$ or $r(u) < \theta_1$ then
5:    return(REJECT)
6: if $r(u) \geq \theta_2$ or $dist(u, f) = 15$ then
7:    return(ACCEPT)
8: $s \leftarrow$ REJECT; $b \leftarrow r(u)$    /* b: corroborative confidence in $p$ */
9: $v \leftarrow u$; $V \leftarrow u$    /* V: set of nodes involved in the consistency check */
10: while $dist(v, f) \geq 0$ do
11:    if $dist(v, f) = 0$ then
12:       if prefix_is_owned($f, v$) = TRUE    /* Algorithm 1 */ then
13:          $s \leftarrow$ ACCEPT
14:       else
15:          $s \leftarrow$ REJECT; $V \leftarrow v$    /* V reset to $v$ */
16:       break    /* to line 39 */
17:    $w \leftarrow nh(v, f)$
18:    if $w \in V$    /* a routing loop is detected */ then
19:       $s \leftarrow$ REJECT; break    /* to line 39 */
20:    if $K[w] = \phi$ and prefix_is_owned($f, w$) = TRUE    /* $w$ is a non-RIP router */ then
21:       if router_is_neighbor($v, w$) = TRUE    /* Algorithm 2 */ then
22:          $s \leftarrow$ ACCEPT
23:       else
24:          $s \leftarrow$ REJECT; $V \leftarrow v$
25:       break    /* to line 39 */
26:    $V \leftarrow V + w$ /* $w$ is a RIP router; continue consistency check*/
27:    Request routes (* to be filled) from $w$: $[f, *, *]$ and $[v, *, *]$    /* see Table 5.1 */
28:    /* denote the * values: $[f, dist(w, f), nh(w, f)]$ or $[v, dist(w, v), nh(w, v)]$ */
29:    if request not met then
30:       $s \leftarrow$ REJECT; break    /* to line 39 */
31:    if $nh(w, v) \neq v$ or $dist(w, v) \neq 1$ or $dist(v, f) \neq dist(w, f) + 1$ then
32:       $s \leftarrow$ REJECT; break    /* to line 39 */
33:    $b \leftarrow b + (1 - b) \cdot r(w)$    /* see equation (4.1) */
34:    if $b \geq \theta_2$ then
35:       $s \leftarrow$ ACCEPT; break    /* to line 39 */
36:    else
37:       $v \leftarrow w$
38: end while
39: update_ratings($V, s$)    /* Algorithm 3 */
40: Return(s)
```

We next analyze how S-RIP achieves G3 and G4, which relate to the truthfulness of routing updates. We consider the case that S-RIP is configured for maximum security, i.e., all routers sitting on a route will be involved in the consistency check for that route.

Given a route update $p{=}[dest, dist, nh]$ in S-RIP, an adversary can manipulate p in the following ways: (**T1**) falsifying the destination $dest$; (**T2**) falsifying the distance $dist$; and (**T3**) falsifying the next hop nh. The propositions below show that S-RIP can resist these threats.

A misbehaving node might also try to mislead a node performing the consistency check by: (**T4**) providing false route responses; (**T5**) not responding to route requests; or (**T6**) not forwarding a route request or response. These frauds (namely *disruption fraud*) will lead to consistency check failures, thus even a valid route advertised by a well-behaved node can be dropped. We view this as a good trade-off between security and effectiveness since it might be desirable not to use a route involving a misbehaving node even if it is not known exactly which node on the route is misbehaving. For simplicity, we do not consider disruption fraud in the following security analysis since it will result in consistency check failures and thus can be detected.

Proposition 3 (Destination Authentication – G3). *In S-RIP, a route with a falsified destination will be detected if there is at most one misbehaving node in the network.*

Proof Outline. S-RIP assumes an authoritative router-prefix mapping, which specifies which router directly connects to which subnet. If a misbehaving router advertises a direct route for a destination which does not exist or it is not directly connected to, this route with falsified destination can be detected using the router-prefix mapping. If a misbehaving router falsifies the destination of a route it learned from another router, which must be well-behaved as there is only one misbehaving router in the network, such falsified destination will be detected by a consistency check with the well-behaved router. Thus, Proposition 3 is established.

Proposition 4 (Distance Authentication – G4). *In S-RIP, an advertised route with a falsified distance will be detected if there is at most one misbehaving node in the network.*

Proof Outline. A misbehaving router which falsifies the distance of a route to zero can be detected using the router-prefix mapping (equivalent to Destination Authentication; see Proposition 3). A misbehaving router advertising a route with a falsified distance of one or more hops will be detected by the consistency check with a well-behaved next hop router of that route (as there is only one misbehaving router in the network and it is the advertising router of this route). Thus, Proposition 4 is established.

Proposition 5 (Next Hop Authentication – G5). *In S-RIP, an advertised route with a falsified next hop will be detected if there is at most one misbehaving node in the network.*

Proof Outline. Let v_1 be the misbehaving node in the network, which advertises $p_1{=}[f, dist(v_1, f), nh(v_1, f)]$. We say $v_2{=}nh(v_1, f)$ is falsified if one of the following condition is met: 1) v_2 is not a legitimate RIP router or non-RIP router. In this case, it will be detected since a legitimate verifying RIP router does not share a secret key with v_2, nor is v_2 specified in the router-prefix mapping. 2) v_2 is a legitimate RIP router but not a direct neighbor of v_1. Since there is only one misbehaving node in the network, which is v_1, the well-behaved node v_2 will report a node other than v_1 as the next hop from v_2 to v_1 during the consistency check. Thus, this case will be detected. 3) v_2 is a legitimate non-RIP router, but not a direct neighbor of v_1. This case will be detected using the router-prefix mapping. Therefore, Proposition 5 is established.

Theorem 1 (Routing Update Authentication). *In S-RIP, a falsified routing update will be detected provided there is at most one misbehaving node in the network.*

Proof Outline. A routing update P consists of a number of routes p. Based on Propositions 3, 4, and 5, we know that $\forall p{\in}P$, any falsified field in p will be detected if there is at most one misbehaving node in the network. It follows that any falsified field in any route in P will be detected. Thus, Theorem 1 is established.

Definition 3 (Collusion). *Let v_1 be a router advertising to v_0 a falsified route p_1 for f, and let p_2 be the route for f provided by v_2 during a consistency check of p_1 by v_0. Let $p_1{\Leftrightarrow}p_2$ denote p_1 and p_2 are consistent, and $p_1{\not\Leftrightarrow}p_2$ denote p_1 and p_2 are not consistent. v_1 and v_2 are in* **collusion** *if v_2 intentionally provides a falsified p_2 such that $p_2 \Leftrightarrow p_1$.*

Theorem 2 (Authentication in Presence of Multiple Misbehaving Nodes). *Let k_m be the maximum network diameter supported by RIP. Suppose there are multiple misbehaving nodes in a network, no two of which are in collusion. We assume that a misbehaving node falsifies the distance of a route to a value in $[0, k_m{-}1]$ (a falsified route of distance $k{\geq}k_m$ is not verified). Then, S-RIP will detect a falsified route with probability at least $1{-}\frac{1}{k_m-1}$.*

Proof Outline. Let v_1 advertise route $p_1{=}[f, dist(v_1, f), nh(v_1, f)]$, and $v_2{=}nh(v_1, f)$. Let $p_2{=}[f, dist(v_2, f), nh(v_2, f)]$ and $p_2'{=}[v_1, dist(v_2, v_1), nh(v_2, v_1)]$ be the routes provided by v_2 during the consistency check of p_1. If only one of v_1 and v_2 is misbehaving, then a falsified route always causes inconsistency with the correct one. Thus, it is always detected, i.e., with probability 1. If both v_1 and v_2 are misbehaving but not in collusion, the probability that a falsified route is not detected is equal to the probability that $p_1{\Leftrightarrow}p_2$. Assume that a misbehaving v_2 intends to help v_1 pass the consistency check by always setting $nh(v_2, v_1){=}v_1$, but has no knowledge of the actual falsified distance of p_1 except that p_1 is reachable route for v_0, i.e., $dist(v_0, v_1){+}dist(v_1, f){\leq}k_m$. Since $dist(v_0, v_1){=}1$, $dist(v_1, f){\leq}k_m{-}1$. With this information and the intention of colluding, v_2 will always report $dist(v_2, v_1){=}1$, and report $dist(v_2, f)$ in a such way that $dist(v_1, v_2){+}dist(v_2, f){=}dist(v_1, f)$, i.e., $dist(v_2, f){\leq}k_m{-}2$. $p_1{\Leftrightarrow}p_2$ requires that $dist(v_1, f){=}dist(v_1, v_2){+}dist(v_2, f){=}1{+}dist(v_2, f)$. Since v_2 does not know $dist(v_1, f)$, we assume that $dist(v_2, f)$ is randomly chosen from $[0, k_m{-}2]$. Then the probability that $p_1{\Leftrightarrow}p_2$ is $\frac{1}{k_m-1}$, and the probability that $p_1{\not\Leftrightarrow}p_2$ is $1{-}\frac{1}{k_m-1}$. Thus, Theorem 2 is established.

In RIP $k_m{=}15$, the probability that $p_1{\not\Leftrightarrow}p_2$ is $1{-}\frac{1}{14}{=}92.9\%$ if both v_1 and v_2 are misbehaving. Otherwise, the probability that $p_1{\not\Leftrightarrow}p_2$ is 1. Therefore, S-RIP can detect a falsified route with the probability of at least 92.9% in the presence of multiple non-colluding misbehaving nodes in a network.

5.6.2 Analysis of S-RIP Network Overhead

We analyze network overhead generated by S-RIP in the worst case that a consistence check involves all nodes on a route in question (see Algorithm 4: lines 10,27,36). S-RIP network overhead in average cases is studied using simulation in §5.7.

Consider a network with n routers and m subnets. Assume the average length of a route is $l{+}1$ hops, where l is dependent of the network topology. For maximal security, each router would verify every route it receives with every other router on that route. For a single route with a length of $l{+}1$ hops, the number of messages required for a consistency check, including requests and responses, is $2{\cdot}l$. Each message travels a number of hops. The first request message is sent to the node two hops away, and travels 2 hops. The last request message is sent to the node $l{+}1$ hops away, and travels $l{+}1$ hops. A response message travels as many hops as the corresponding request message assuming they traverse the same route. Therefore, the total number of hops (message transmissions) traveled by all request and response messages resulting from the consistency check of a single route is $2{\cdot}[2{+}3{+}..{+}(l{+}1)]{=}(1{+}l){\cdot}l$. Assume every router keeps a route for every subnet in the network. Each router would need $(1{+}l){\cdot}l{\cdot}m$ message transmissions for verifying every route. Over the whole network, the total number of message transmissions in the most secure case is $(1{+}l){\cdot}l{\cdot}m{\cdot}n$.

We use RIP messages for route request and response. Each S-RIP route request or response message has two route entries (see Table 5.1), one for the route from the recipient to the ultimate destination, and one from the recipient to its predecessor node on that route. The RIP message header [83] is 24 bytes including authentication data, and each route entry is 20 bytes. Since one S-RIP route request or response message consists of a RIP header and two route entries, it is in total 64 bytes. Including the UDP header (8 bytes) and IP header (20 bytes, without options), a packet carrying an S-RIP route request or response message is 92 bytes. The total overhead of routing validation, in addition to the overhead of regular routing updates, in the most secure case, is $92{\cdot}(1{+}l){\cdot}l{\cdot}m{\cdot}n$ bytes.

As confirmed by our simulation (§5.7), the validation overhead by S-RIP is relatively high in the maximally secured case, especially during network initialization (e.g., after a router reboots). However, S-RIP provides the flexibility for balancing security and efficiency via two configurable thresholds θ_1 and θ_2. As showed in §5.4.2, S-RIP overhead is relatively low in a partially secured network.

S-RIP validation overhead can also be reduced by optimized implementation (e.g., transmitting several route requests or responses in a single message). For example, if v_1

advertises to v_0 three routes with a same next hop v_2, v_0 can send a single message with 4 route entries to v_2, one for each of three advertised destinations and one for v_1. The size of the packet carrying this message is 132 bytes, which consists of a 20-byte IP header, a 8-byte UDP header, a 24-byte RIP header, and four 20-byte route entries. This size is considerably smaller than the total size (276 bytes) of three standard packets, each of which is 92 bytes.

After a network converges (i.e., all routes have been discovered by all routers in the network), consistency checks in S-RIP will only be triggered by a topology change, such as a link failure. After monitoring a production network within a Canadian government department which consists of 61 routers and 142 subnets for 3 months, we observed 95 link failures, which is about one failure per day. While this network cannot represent every other network, and the period of time we monitored the network does not represent all the conditions of the network, it nonetheless serves as an example of showing the stability of some networks. We expect that the routing overhead generated by S-RIP will not be significant even in a maximally secured network once it reaches the convergence state.

5.6.3 Comments on Deployability of S-RIP

A practical challenge of securing routing protocols is how to make the secured version interoperative with the existing infrastructure. Despite their technical merits, many proposed mechanisms for securing routing protocols are not widely deployed due to the fact that they require significant modifications to existing implementations and/or do not provide backward interoperability. Since it is unrealistic to expect that an existing routing infrastructure can be replaced by a secured version in a very short period of time, ideally a secured version should be compatible with the insecure protocols. It is also desirable that security can be increased progressively as more routers are deployed with the secured protocol.

To this end, S-RIP supports incremental deployment. We propose that messages exchanged in S-RIP conform to the message format defined in RIP [83]. S-RIP can be implemented as a compatible upgrade to the existing RIP; an S-RIP router performs routing functions the same way as a RIP router. Therefore, deploying S-RIP on a router only requires a down time for the period of installation and rebooting of RIP processes. Since RIP router responds to a routing request from a non-direct neighbor (a remote node), an S-RIP router can successfully get information (albeit not authenticated) from a non-secured router for a consistency check. In other words, a RIP router can participate in a consistency check, but not initiate a consistency check. Thus, even before S-RIP is deployed on all routers, the routing table of an S-RIP router is partially protected as it is built from validated routing updates. The more routers deployed with S-RIP, the more reliable the routing tables in the network become. Thus, security in RIP can be increased incrementally with the deployment of S-RIP.

5.7 Simulation of S-RIP

We implemented S-RIP in the network simulator NS2 as an extension to the distance vector routing protocol provided by NS2 [36]. S-RIP is triggered when a received route is used to update a recipient's routing table. In this section, we present our simulation results on how security and routing overhead are affected by different threshold settings and number of misbehaving nodes in S-RIP.

5.7.1 Simulation Environment

Network Topology. We simulated S-RIP with four different network topologies, which are generated randomly using the Waxman model [143] with $20, 30, 40$, and 50 nodes respectively. Given a certain number of nodes and a plane with a dimension $s \times s$, the Waxman model first randomly distributes nodes in the plane. It then defines the probability of a link (u, v) by $P(u, v) = \alpha e^{\frac{-d}{\beta L}}$, where $0<\alpha, \beta\leq 1$, d is the Euclidean distance between u and v, and $L=\sqrt{2}s$ is the maximum distance between any two nodes. In our simulation, we configure $s=10$, $\alpha=0.2$, $\beta=0.2$. Setting $s=10$ ensures that the maximum distance between any two nodes does not exceed 15, which is the maximum hop number allowed in RIP [49]. α controls the total number of links in the graph, and β controls the ratio of long links relative to shorter ones.

Fraud. For each network topology, we randomly select $10\%, 20\%, 30\%, 40\%$ and 50% nodes to commit fraud for each of the S-RIP threshold configurations as described next. A misbehaving node can commit shorter or longer distance fraud (§5.2.1) or both. More specifically, a misbehaving node periodically (every 2.5 seconds) and randomly selects a route from its routing table and makes its distance shorter or longer.

Maximally Secured	$\theta_1 = 0$	$\theta_2 = 1$
Partially Secured-1	$\theta_1 = 0.1$	$\theta_2 = 0.9$
Partially Secured-2	$\theta_1 = 0.2$	$\theta_2 = 0.8$
Partially Secured-3	$\theta_1 = 0.3$	$\theta_2 = 0.7$
Not Secured	$\theta_1 = 0$	$\theta_2 = 0$

Table 5.2: Simulated S-RIP threshold configurations

S-RIP Threshold Configurations. We simulated 5 different configurations of S-RIP thresholds θ_1 and θ_2 (see Table 5.2). Each node initially rates every other node with 0.5. Based on equation (4.1), a successful consistency check of a route p involving two, three and four nodes (each is rated 0.5) respectively brings one's confidence in p to $0.75, 0.875$, and 0.94. Thus, the three partially secured cases (partially secured-3, secured-2, and secured-1) respectively require the involvement of one, two, and three additional nodes (rated with 0.5) in order to succeed a consistency check. Node ratings are dynamically updated. More specifically, the rating of a node increases after it was involved in a successful consistency check, and drops after involved in a failed one. A node rated lower than θ_1 or higher than θ_2 is re-rated with 0.5 after 2 seconds.

Simulation Scenarios. Combining four network topologies, five fraud scenarios, and five configurations of S-RIP thresholds, we created 100 simulation scenarios in total. Each simulation lasts 180 seconds (in simulation time). We run each simulation five times, and present the average result of the five runs.

5.7.2 Simulation Metrics

We use two metrics for evaluating S-RIP, namely, *risk window* for accepting advertised routes for which consistency checks are not performed, and *routing overhead* generated by S-RIP.

A Risk Window

We counted the total number of advertised routes (n_1) which are accepted by all routers without any consistency check due to the fact that the advertisers have a reputation at least θ_2 (i.e., they are trusted). We also counted the total number of times (n_2) advertised routes are checked for their consistency. $\frac{n_1}{n_1+n_2}$ represents the lower bound of the probability that a malicious route may be accepted. It is used to measure the risk that a routing table may be poisoned by malicious updates.

Routing Overhead

To determine how much network overhead is generated by S-RIP, we compared the S-RIP overhead with the total routing overhead, which is calculated as the sum of S-RIP overhead and regular routing update overhead in RIP. Since the distance vector routing protocol provided by NS2 is not a strict implementation of the RIP RFC [83], we could not obtain network overhead directly from the NS2 trace file. We use $\frac{92x}{92x+zy}$ to calculate the ratio of S-RIP overhead relative to the total routing overhead, where 92 bytes is the size of the packet carrying an S-RIP message (see §5.5), x is the total number of S-RIP message transmissions, y is the total number of rounds of regular routing updates, and z is total number of bytes of overhead generated by one router in one round of regular routing updates. x and y are derived from simulation outputs, which are used to generate Figure 5.6, and z is calculated as follows.

In our non-optimized implementation, each S-RIP message contains only two route entries. The size of each message is 92 bytes, including the headers of RIP (24 bytes with authentication data), UDP (8 bytes), and IP (20 bytes).

$$2 * RIP\_entry(20) + RIP\_hdr(24) + UDP\_hdr(8) + IP\_hdr(20) = 92\, bytes$$

where, RIP_hdr includes authentication data, and IP_hdr excludes optional fields (e.g., source routing). A full RIP message with authentication data is 512 bytes, and can contain up to 24 entries. An IP packet carries a full RIP message is 540 bytes with both UDP and IP headers. For a network with n nodes, each round of regular routing updates by one node requires $\frac{n}{24}$ full RIP messages, and a partial RIP message containing $n \bmod 24$ route entries. A partial RIP message with m route entries is $20m{+}52$ bytes with all headers. Therefore, the routing overhead z generated by a single router in a network with n nodes in each round of regular routing update is $540\frac{n}{24} + 20(n \bmod 24) + 52$ bytes.

5.7.3 Simulation Results

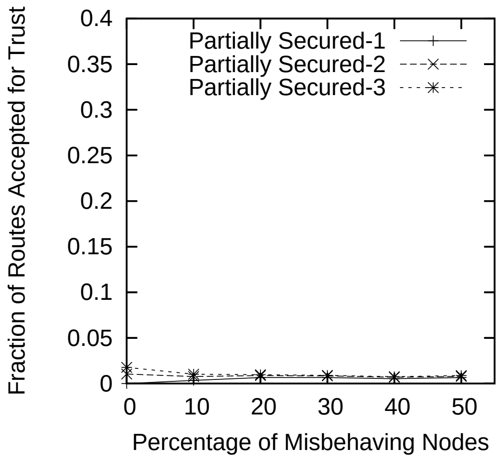
Here we present our simulations results for the four random networks of different size. Simulation results for the non secured scenario ($\theta_1{=}0, \theta_2{=}0$) are not presented since S-RIP is never triggered. Thus in this case, the risk of accepting malicious routing updates is 100%, and there is no S-RIP routing overhead.

A Risk Window

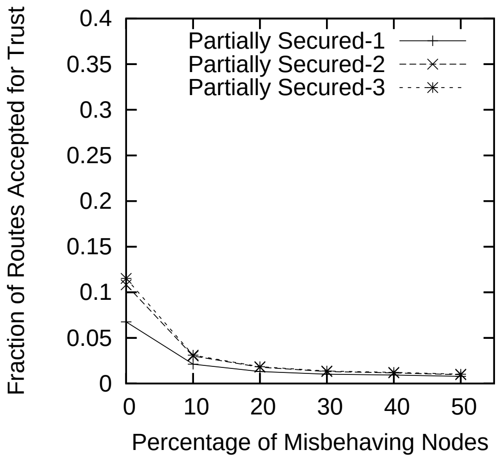
Figure 5.5 shows (in conjunction with Table 5.2) that: 1) The lower the second threshold (θ_2), the higher the risk of accepting a malicious route. This is because the higher θ_2 is, the longer it takes for a node to become trusted (i.e., its rating is at least θ_2). 2) The less misbehaving nodes there are, the higher the risk of accepting a malicious route. The reason is that when there are fewer misbehaving nodes in the network, more nodes will become trustworthy. This is similar to the social phenomena that people from a friendly or non-hostile neighborhood are less vigilant and thus more susceptible to lies. 3) When there are 20% or more misbehaving nodes, the risk of accepting a malicious route becomes very low (below 5%). In addition, the difference among the three partially secured scenarios becomes insignificant since most nodes have a rating of less than 0.7, i.e., they are untrusted in all three scenarios. Although the risk of routing tables being poisoned becomes low, the risk that correct routes are not in routing tables becomes high. Therefore, data traffic may be dropped, though not routed to a misbehaving node.

S-RIP Routing Overhead

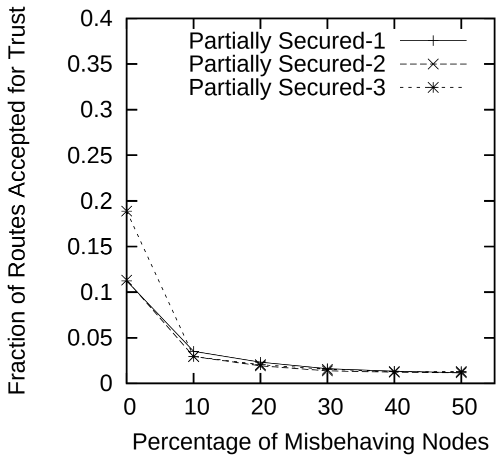
Figure 5.6 compares the S-RIP network overhead in different scenarios. 1) In a maximally secured network, S-RIP overhead is high (about 10% to 25% of the total routing over-



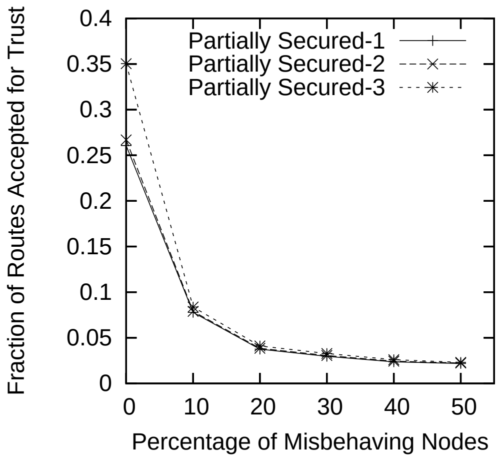
(a) Network with 20 nodes



(b) Network with 30 nodes

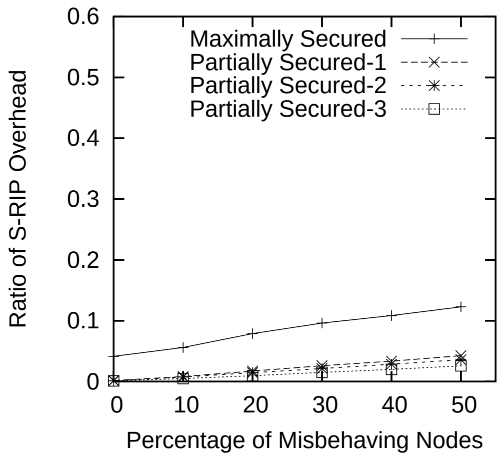


(c) Network with 40 nodes

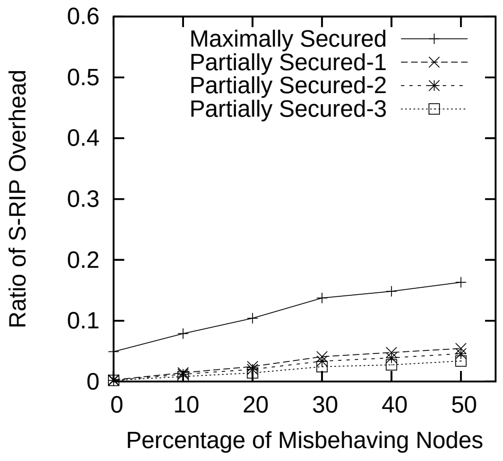


(d) Network with 50 nodes

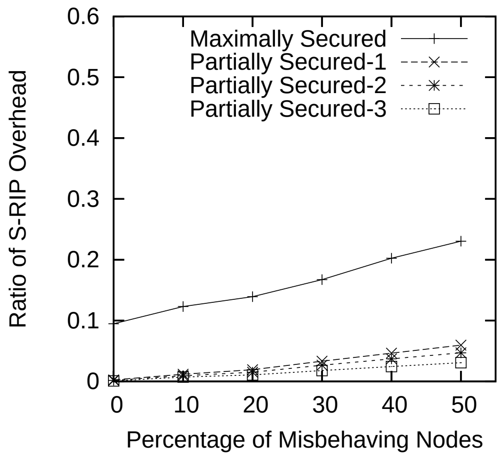
Figure 5.5: Fraction of routes accepted for which consistency checks are not performed.



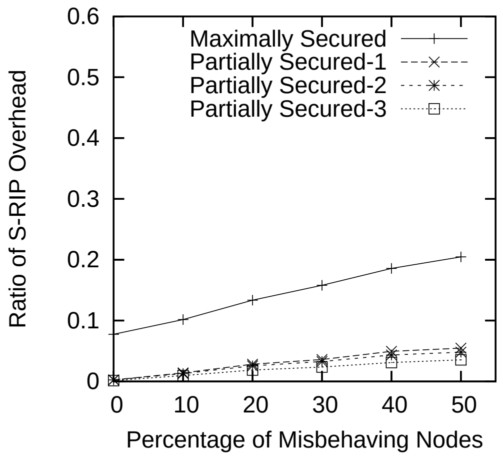
(a) Network with 20 nodes



(b) Network with 30 nodes



(c) Network with 40 nodes



(d) Network with 50 nodes

Figure 5.6: Ratio of S-RIP routing overhead.

head). 2) In the three partially secured scenarios, S-RIP overhead is relatively low (less than 5.5%). S-RIP overhead increases when the number of misbehaving nodes increases, but only slightly. On the one hand, more misbehaving nodes trigger more malicious updates, which in turn trigger more consistency checks, and result in more S-RIP overhead. On the other hand, more misbehaving nodes result in more consistency check failures, which in turn decreases the ratings of more nodes into the low range, and thus result in fewer consistency checks for malicious updates. Therefore, more misbehaving nodes result in only slight increase in S-RIP overhead. To summarize, S-RIP performs consistently in four random networks of different size.

5.8 Summary

In this chapter, we first analyzed security vulnerabilities of RIP, and then proposed S-RIP for improving RIP security. Our security analysis of S-RIP shows that, with high probability, S-RIP can detect a fraudulent route that has a falsified destination, distance, or a next hop. Our simulation results demonstrate that security and network overhead in S-RIP can be balanced by adjusting two thresholds θ_1 and θ_2.

S-RIP makes use of symmetric cryptographic mechanisms for data origin authentication and data integrity. For organizations which have skill sets and resources for managing a public key infrastructure, public key based cryptographic mechanisms, e.g., digital signatures, can also be used. In that case, each router can be issued a public key certificate signed by a certification authority which is trusted by all routers within the administrative domain. A public key certificate issued to a router can encode policy information indicating that the owner of the private key corresponding to the public key in the certificate is authorized to participate in the routing protocol. Therefore, an unauthorized entity will not be able to join the routing protocol and spread fraudulent routing information since it does not have an authorized public key certificate. This effectively prevents router spoofing. In addition, routing request and response messages can also be digitally signed if computing power is not a problem, thus removing the need of pairwise shared secret keys among routers. The downside is that digital signature generation and verification is computationally expensive, which could result in the delay of the route verification process and network convergence.

S-RIP treats all nodes involved in a consistency check equally in updating their ratings without making sufficient attempt to identify nodes which are more likely to be misbehaving during the consistency check. In other worlds, if a consistency check fails unless it is being caused by a destination fraud in which case a misbehaving node can be clearly identified, the ratings of all nodes involved in the consistency check are reduced. This might open an opportunity for an attacker to manipulate the automatic rating update mechanism with T4, T5, and T6 (see §5.6.1) so that a well-behaved node is rated low by some other nodes, and thus appears untrustworthy to them. For example, an attacker can cause the consistency check of a valid route to fail by not responding properly to a route request during the consistency check. As a result, the rating of the well-behaved node advertising a valid route can be lowered by a verifying node. However, an attacker cannot manipulate the rating of another node at its will, and a well-behaved node has opportunity to raise its reputation by participating in consistency checks that involve only well-behaved nodes. For future work, additional simulation can be performed to study the impact of disruption fraud (T4, T5, and T6) on node ratings in different networks.

S-RIP makes use of two timers P_1 and P_2 (see §5.4.2) for expiring low and high ratings respectively. In our simulation, we arbitrarily set both P_1 and P_2 to 2 seconds in simulation time (recall the total simulation time is 180 seconds). As a future work, P_1 and P_2 can be adjusted for a particular simulation scenario to study their impact on the risk of accepting a fraudulent route, and S-RIP network overhead.

Chapter 6

Securing Inter-domain Routing and psBGP

6.1 Introduction and Motivation

The Internet routing infrastructure consists of a number of Autonomous Systems (ASes), each of which consists of a number of routers under a single technical administration (e.g., sharing the same routing policy). The Border Gateway Protocol (BGP) [109] is the IETF standard inter-domain routing protocol for exchanging reachability information between ASes on the Internet. Each network layer destination is identified by an IP prefix representing a range of IP addresses. An AS announces its IP prefixes via BGP to its direct neighbors, which may further propagate the prefix announcement to their neighbors. A remote AS receiving such announcements may build routes for forwarding traffic destined to the addresses within the address range specified by the announced prefixes.

One critical question with BGP is the following: which AS has a right to announce a given IP prefix? The current version of BGP does not have any mechanism to verify the propriety of IP prefix announcements. This opens a serious security hole which allows one AS to announce IP prefixes allocated or delegated (hereafter *assigned* - see §2.1.3) to other ASes. This is commonly referred to as *prefix hijacking*. Examples of consequences include denial of service (i.e., legitimate user traffic cannot get to its ultimate destination) and man-in-the-middle attacks (i.e., legitimate user traffic is forwarded through a router under the control of an adversary). Warnings about attacks exploiting routing vulnerabilities were given as early as 1988 by Perlman [103], and 1989 by Bellovin [11]; and such attacks have recently reportedly been carried out by spammers [15].

Many proposals [72, 44, 144, 3] have been made for improving BGP security, and in particular, for verifying if an AS has the right to announce a given IP prefix. There are two main approaches: 1) building centralized routing registries storing information about address space assignments, e.g., Internet Routing Registry (IRR) [59]; and 2) building a strict hierarchical public key infrastructure (PKI) in parallel to the existing IP address assignment structure (e.g., S-BGP [120, 80]). While these two approaches may differ in many ways, e.g., protecting a database itself vs. protecting individual objects in the database, they both typically require a large scale PKI to provide strong security or to meet some operational requirements (e.g., multi-homing). Such a PKI continues to be viewed as impractical by many experts [6].

IRR needs to perform identity authentication to verify if an entity requesting to make changes to the routing database is authorized to do so. Currently in IRR, PGP [153] is used for public key authentication. However, this authentication is done using a sender's email address when an object is first created, and thus is vulnerable to email spoofing [154]. As a result, a global PKI or something equivalent, appears to be required to provide stronger guarantees. S-BGP makes use of a hierarchical tree structure for address assignment, rooted at Regional Internet Registries (RIRs). For each consecutive pair of nodes on the address assignment chain, the first node (an organization) on the chain assigns a subset of its own address space to the second one. While an organization obtaining its address space from its Internet Service Providers (ISPs) may not need to appear on an address delegation chain (i.e., need not be issued relevant certificates), it will need those certificates (e.g., a public key certificate and an address assignment certificate) to do multi-homing (i.e., connecting to two independent ISPs). Multi-homing has been considered as a common operational practice which must be supported [129]. This implies that many multi-homed organizations not running BGP may also need to be involved in the S-BGP PKI, resulting in a large scale global PKI.

In addition, it appears difficult to build a centralized PKI for verifying IP address assignment given the complexity, if not impossibility, of tracing how the existing IP address space is assigned, and tracing all changes of IP address assignments. This is in part due to the large number of prefixes in use and organizations involved, and frequent organization changes (e.g., corporations splitting, merging, bankruptcy, etc.). As pointed by Aiello et al. [3], it is exceptionally difficult to even approximate an IP address delegation graph for the Internet. Therefore, it may well be impossible to build a centralized PKI mirroring such a complex and unknown delegation structure.

Aside from the challenges of requiring a global PKI, many IP addresses were given out before the existing hierarchical address allocation structures were in place. Thus, address assignment chains might not be applicable to them. Fundamentally, all these approaches assume a trusted source of authoritative routing information which allows for detecting false prefix announcements. We suggest that such an assumption may not be realistic, or at least it would be very difficult to build an infrastructure to realize it. As noted by Atkinson and Floyd [6] on behalf of the Internet Architecture Board (IAB): “*a recurring challenge with any form of inter-domain routing authentication is that there is no single completely accurate source of truth about which organizations have the authority to advertise which address blocks*".

6.1.1 psBGP Highlights

In this chapter, we present a new BGP security proposal – Pretty Secure BGP (psBGP), fleshing out a preliminary overview [138]. psBGP includes defenses against falsification of BGP UPDATE messages, and a new approach for verifying the propriety of prefix origin by cross checking information from multiple, ideally independent, sources. Specific psBGP security goals are outlined in §6.4. psBGP is based on the following concepts: 1) there is no universally trusted authority which has full knowledge (i.e., all aspects of the factual reality) of prefix assignments on the Internet; 2) some entities may know part of such truth; and 3) corroboration of information from different sources can increase confidence in the assessment of that information. In particular, RIRs are the trusted authority of initial prefix allocations, and some ASes might have partial knowledge of prefix assignments of their direct neighbors. The major architectural highlights of psBGP are as follows.

1) psBGP makes use of a *centralized trust model* for AS number authentication. Each AS obtains a public key certificate from one of several trusted certificate authorities (i.e., RIRs), binding an AS number to a public key. We suggest that such a trust model provides best possible authorization of AS number allocation and best possible authenticity of AS public keys. Authentication is usually the first step towards authorization. Without such a guarantee, an attacker may be able to impersonate another AS and thus be able to announce prefixes assigned to the impersonated AS.

2) psBGP makes use of a rating mechanism for flexibility in balancing security and practicality in prefix origin and AS_PATH verification.

3) psBGP makes use of a *decentralized trust model* for verifying the propriety of IP prefix assignment. Each AS periodically issues a digitally signed *Prefix Assertion List (PAL)* consisting of a number of bindings of an AS number and (zero or more) IP prefixes, one such binding for itself and one for each of its neighbors. An assertion made by an AS s_i regarding its own prefixes (*prefix assertion*) lists all prefixes assigned to s_i. An assertion made by s_i for a neighboring AS s_j (*prefix endorsement*) may list all or a subset of the prefixes assigned to s_j. An *AS prefix graph* (see §6.6.3) is built independently by each AS s_i based on the $PALs$ which s_i has received from other ASes and s_i's ratings of those ASes. An AS prefix graph is then used for evaluating the trustworthiness and preference of a prefix origin by an AS, in conjunction with its local configurable parameters (e.g., its trust in those ASes involved in a prefix assertion, and trust thresholds). In this way, the difficult task of tracing IP address assignments is distributed across ASes on the Internet.

4) psBGP modifies the S-BGP digital signature approach with a rating mechanism and a stepwise approach for verifying AS_PATH integrity. Each AS computes a weight for an AS_PATH based on ratings of the ASes digitally signing the path, and determines whether or not to accept the path based on local parameters. This approach allows an upgrading path to countering increased threats, as recommended in [17].

Our design is inspired by the referral model widely used in social society for increasing confidence in the truth of information when an authoritative source of truth regarding that information is not available. For example, a job applicant is usually required to provide reference letters to allow cross checking the applicant statements on his quality and background. A reference letter should be from an individual who has closely worked with the applicant, e.g., a former supervisor. Similarly in psBGP, each AS should obtain endorsement for its prefix assertions from some ASes which are likely to have, or likely to be reliable sources for, knowledge of its prefix assignment, e.g., a direct neighbor with which it has a business relationship. An AS choosing to endorse a prefix assertion made by a neighboring AS should carry out some form of due diligence (or other means to increase accountability) to increase confidence in the correctness of that assertion, i.e., to increase its own confidence that the asserted prefix is indeed assigned to the asserting AS. The security assurances of this aspect of psBGP are directly related to the quality of such due diligence, which will impose extra work on BGP operators; this is the price to pay for increased security.

As discussed in what follows, advantages of psBGP include: 1) *simplicity* – it uses a PKI which has a simple structure, a small number of certificate types, and is of manageable size; 2) *effectiveness* – it is designed to successfully defend against selected threats from uncoordinated, misconfigured or malicious BGP speakers; and 3) *incremental deployability* – it can be incrementally deployed with some incremental benefits.

6.1.2 Organization

The rest of the chapter is organized as follows. §6.2 defines notation, and discusses BGP threats. §6.3 outlines our observations of BGP prefix origin during Google's May-2005 outage. §6.4 summarizes BGP security goals. psBGP is presented in §6.5 and §6.6.

6.2 BGP Security Threats

In this section, we define notation, discuss BGP security threats in general, and describe several attacks by exploiting BGP security vulnerabilities.

6.2.1 Notation

A and B denote entities (e.g., an organization, an AS, or a BGP speaker). X or Y denotes an assertion which is any statement. An assertion may be *proper* or *improper*. We avoid use of the term *true* or *false* since in BGP, it is not always clear that a statement is 100% factual or not. An assertion is proper if it conforms to the rules (e.g., psBGP rules) governing the related entity making that assertion. Table 6.1 defines some of the notation used in this thesis.

$\mathbb{S}, s_i$	$\mathbb{S}$ is the set of all AS numbers; currently $\mathbb{S} = \{1, \ldots, 2^{16}\}$. $s_i \in \mathbb{S}$ is an AS number.
$\mathbb{P}, f_i$	$\mathbb{P}$ is the set of all IP addresses. $f_i \subseteq \mathbb{P}$ is an IP prefix specifying a range of IP addresses. $f_i = f_j \cup f_k$ if the IP addresses specified by f_i equal those by f_j and f_k combined.
T	an authority with respect to $\mathbb{S}$ and $\mathbb{P}$, e.g., $T \in \{x \mid x$ is an RIR$\}$.
p_k	$p_k = [s_1, s_2, \ldots, s_k]$ is an AS_PATH; s_1 is the first AS inserted onto p_k.
m	$m = (f_1, p_k)$ is a BGP route (a selected part of a BGP UPDATE message).
$N(s_i)$	s_i's neighbors, i.e., the set of ASes with which s_i establishes a BGP session on a regular basis. A given AS s_i may have many BGP speakers, each of which may establish BGP sessions with speakers from many other ASes. $N(s_i)$ is the set of all other such ASes.
$k_A, \overline{k_A}$	A's public and private keys, respectively.
$\{m\}_A$	digital signature on message m generated with A's private key $\overline{k_A}$.
$(k_A, A)_{\overline{k_B}}$	a public key certificate binding k_A to A, signed using $\overline{k_B}$, verifiable using k_B.
$(f_i, s_i)_A$	an assertion made by A that f_i is assigned to s_i.

Table 6.1: Notation for psBGP

6.2.2 Security Threats to BGP

BGP faces threats from both BGP speakers and BGP sessions. A misbehaving BGP speaker may be misconfigured (mistakenly or intentionally), compromised (e.g., by exploiting software flaws), or unauthorized (e.g., by exploiting a BGP peer authentication vulnerability). A BGP session may be compromised or unauthorized. We focus on threats against BGP control messages without considering those against data traffic (e.g., malicious packet dropping [65]). Attacks against BGP control messages include, for example, modification, insertion, deletion, exposure, and replaying of messages. In this thesis, we focus on modification and insertion (hereafter *falsification* [9]) of BGP control messages; deletion, exposure and replaying are beyond the scope of this thesis, other than the following brief remarks. Deletion appears indistinguishable from legitimate route filtering. Exposure might compromise confidentiality of BGP control messages, which may or may not be a major concern [9]. Replaying is a serious threat, which can be handled by setting an expiration time for each message; however it seems challenging to find an appropriate value for an expiration time.

There are four types of BGP control messages defined in [109]: OPEN, KEEPALIVE, NOTIFICATION, and UPDATE. The first three are used for establishing and maintaining BGP sessions with neighbors, and falsification of them will very likely result in session disruption. As mentioned by Hu et al. [51], they can be protected by a point-to-point authentication protocol, e.g., IPsec [69]. In psBGP, we concentrate on falsification of BGP UPDATE messages (and hereafter, refrain from capitalizing UPDATE) which carry inter-domain routing information and are used for building up routing tables.

A BGP update message consists of three parts: withdrawn routes, network layer reachability information (NLRI), and path attributes (e.g., AS_PATH, LOCAL_PREF, etc.). As commonly agreed [51], a route should only be withdrawn by a party which had previously propagated that route. Otherwise, a malicious entity could cause service disruption by withdrawing a route which is actually in service. Digitally signing BGP update messages would allow one to verify if a party has the right to withdraw a route. Further discussion is beyond the scope of this thesis.

NLRI consists of a set of IP prefixes sharing the same characteristics, as described by the path attributes. NLRI is *falsified* if an AS originates a prefix not owned by that AS, or aggregated improperly from other routes. Examples of consequences include denial of service and man-in-the-middle attacks. There are two types of AS_PATH: AS_SEQUENCE and AS_SET. An AS_PATH of type AS_SEQUENCE consists of an ordered list of ASes traversed by the route currently being propagated. An AS_PATH of type AS_SET consists of an unordered list of ASes, sometimes created when multiple routes are aggregated. An AS_PATH is falsified if an AS or any other entity illegally operates on an AS_PATH, e.g., inserting a wrong AS number, deleting or modifying an AS number on the path, etc. Since AS_PATH is used for detecting routing loops and used by route selection processes, falsification of AS_PATH can result in routing loops or selecting routes not selectable otherwise.

Some other path attributes (e.g., community, Multi_Exit_Disc, etc. [109]) may also need protection, but many of these are usually only used between two neighbors and not globally transitive. Thus, damage resulting from attacking them is relatively contained. In psBGP, we focus on countering falsification of NLRI and AS_PATH which can result in large scale service disruption.

We assume there are multiple non-colluding misbehaving ASes and BGP speakers in the network, which may have their own legitimate cryptographic keying materials. This non-colluding assumption is also needed by other BGP security proposals (e.g., S-BGP and soBGP), although consequences resulting from collusion might be different.

6.2.3 Examples of BGP Attacks

Here we give examples of BGP attacks involving falsification of BGP update messages, i.e., falsification of NLRI and AS_PATH.

Falsification of NLRI

NLRI consists of a set of IP prefixes sharing the same characteristics as described by the path attributes. Falsification of NLRI is often referred to as *prefix hijacking*, and can cause serious consequences including denial of service and man-in-the-middle (MITM) attacks.

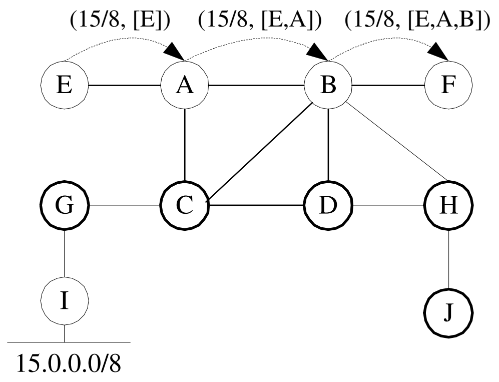


Figure 6.1: An AS topology with attackers

We use Figure 6.1 to illustrate how an attacker controlling a BGP speaker in AS E (i.e., the router establishing a BGP session with AS A) might hijack $15/8$ which is allocated to

AS I. We assume that the network has converged on $15/8$, i.e., every AS has a route to $15/8$ (see Table 6.2).

AS	Route to 15/8	AS	Route to 15/8
A	(15/8, [I,G,C])	F	(15/8, [I,G,C,B])
B	(15/8, [I,G,C])	G	(15/8, [I])
C	(15/8, [I,G])	H	(15/8, [I,G,C,D])
D	(15/8, [I,G,C])	I	direct route
E	(15/8, [I,G,C,A])	J	(15/8, [I,G,C,D,H])

Table 6.2: Routes to $15/8$ from each AS before the attack

(AS E) An attacker configures a compromised BGP speaker in E to advertise route $(15/8, [E])$ to A. Since $15/8$ is *n*ot allocated to E (it is allocated to I), it is illegitimate for E to originate route $(15/8, [E])$. However, an attacker does not play by rules.

(AS A) After receiving $(15/8, [E])$, A now has two distinct routes to $15/8$: $(15/8, [E])$ and $(15/8, [I, G, C])$. A will prefer one of them using the standard BGP route selection process as described in §6.6.5. Assume that A implements a common policy in which a customer route is preferred over a provider route or a peer route. In other words, among a set of routes for the same destination, the route received from a customer AS is preferred over those received from a provider or a peer AS. Thus, $(15/8, [E])$ is preferred over $(15/8, [C, G, I])$ since E is a customer of A, and C is a peer of A. As a result, $(15/8, [E])$ is installed in A's routing table, *which is now poisoned*.

Since $(15/8, [E])$ is learned from A's customer, A will re-advertise it as $(15/8, [E, A])$ to B and C (see §2.1.3 for peer-to-peer route exporting policy).

(AS C) After receiving $(15/8, [E, A])$, C will compare it with $(15/8, [I, G])$. Assume that C implements a common policy that a customer route is preferred over a provider route or a peer route. Since G is a customer of C and B is a peer, $(15/8, [I, G])$ is selected. Therefore, *C's routing table is not poisoned*.

(AS B) After receiving $(15/8, [E, A])$, B compares it with $(15/8, [I, G, C])$ which it has received from C. Since B has a peer relationship with both A and C, the preference values assigned to the two routes might be same. To break the tie, the second rule in the route selection process (cf. §6.6.5) will be applied, which favors the shorter AS_PATH. Therefore, $(15/8, [E, A])$ is selected. *B's routing table is poisoned.*

B will also propagate $(15/8, [E, A, B])$ to F and H, which are the customers of B's (see §2.1.3 for provider-to-customer route exporting policy). However, B will *n*ot propagate this route to C or D since they are peers (see §2.1.3 for peer-to-peer route exporting policy).

(AS F) After receiving $(15/8, [E, A, B])$, F uses it to replace the existing route to $15/8$, i.e., $(15/8, [I, G, C, B])$, without going through the route selection process. In BGP, a new route automatically replaces an old one if they are received from the same source (e.g., B in this case). Thus, *F's routing table is poisoned.*

(AS H) After receiving $(15/8, [E, A, B])$ from B, H compares it with $(15/8, [I, G, C, D])$. Assume that $[D, H]$ is a primary link and $[B, H]$ is a backup one (e.g., $[D, H]$ is less costly than $[B, H]$), then H will assign a higher preference value to the routes received from D than those from B. As a result, $(15/8, [E, A, B])$ is not selected. Therefore, *H's routing table is not poisoned.*

After the above process, the routing tables of A, B and F are poisoned, and the routing tables of G, C, D, H, and J are *not* poisoned (see Table 6.3). As a result, traffic destined to $15/8$ and initiated from A, B, and F will be forwarded to E, not to the real address owner I. In other words, prefix $15/8$ has been *hijacked* from I from the view point of some part of the network.

AS	Route to 15/8	AS	Route to 15/8
A	(15/8, [I,G,C]) → **(15/8,[E])**	F	(15/8, [I,G,C,B]) → **(15/8, [E,A,B])**
B	(15/8, [I,G,C]) → **(15/8, [E,A])**	G	(15/8, [I])
C	(15/8, [I,G])	H	(15/8, [I,G,C,D])
D	(15/8, [I,G,C])	I	direct route
E	(15/8, [I,G,C,A])	J	(15/8, [I,G,C,D,H])

Table 6.3: Routes to $15/8$ from each AS after the attack

It is well-known that prefix hijacking can be used to facilitate many types of attacks, including *denial of service, man-in-the-middle (MITM)*, or *service hijacking* (e.g., email). While service hijacking will always deny the service of a real address holder, it also has the purpose of impersonation. Therefore, it could cause more serious consequences. Here we present three types of attacks using service hijacking: *spamming*, *interception of password Reset messages*, and *Phishing*. The first two attacks described here are related to email server impersonation, and the third attack is related to web server impersonation.

Advanced Spamming. Recently, falsification of NLRI might have been used by spammers to facilitate advanced spamming [16]. Here we describe how spammers can use prefix hijacking to defeat the email authentication mechanism SPF (see §3.2.2).

A spammer who wants to send out spam using the domain name "*alice.com*" can hijack the IP address space containing the authorized IP address ($15.15.2.7$) published by "*alice.com*". For example, the spammer with control of a BGP speaker can announce routes for prefix $15.15.2.0/24$, and set up an SMTP server configured with IP address "$15.15.2.7$". This allows the spammer to use the hijacked IP address "$15.15.2.7$" to establish SMTP connections with "*bob.com*" and send out spam using "*alice.com*" as the domain of the sender address. Email authentication mechanisms such as SPF will not be able to detect this type of spamming. In fact, any authentication mechanism based only on IP address can be defeated by prefix hijacking.

Interception of Password Reset Messages. One possible attack using prefix hijacking is to intercept password reset messages[2] for gaining illegitimate access to other people's email accounts. A traditional way of doing this is to crack the password of a victim account by either offline or online dictionary attacks. Offline dictionary attack usually requires access to the password database (e.g., /etc/passwd in Unix) which may not be possible. Online dictionary attack usually involves automatic logon retries with candidate passwords (e.g., chosen from a dictionary). Since some email service providers have adopted reverse Turing tests to defeat automatic logon retries, it becomes more difficult for online dictionary attack to succeed.

However, many email services provide "user-friendly" features to allow users to reset their passwords in the case they forget them. When a link such as "forgot your password" is

[2]This attack was mentioned to us by Dan Boneh during a conversation at NDSS'05.

clicked, a password reset message is sent to another email account (namely backup email account) associated with the account whose password has been forgot (namely primary email account). A backup email address is usually asked by many email service providers for authentication purpose such as receiving password reset message. A password reset message may contain an automatically generated new password, or a link pointing to a page where the user can type in a new password without being asked for the old password.

The assumption made here is that a backup email address is only accessible to its owner. This assumption usually holds since an email account is usually password protected and it appears difficult to intercept an email message if an attacker does not have access to one of the following communication paths: 1) from the mail server originating a message to the mail server receiving it, and 2) from the mail client retrieving the message to the mail server storing it.

However, such an assumption will loose ground if an attacker can manipulate BGP to hijack IP prefixes. Suppose a user has a primary email address "*x1@alice.com*", and the backup email address associated with this account is "*x1@bob.com*". An attacker may gain access to "*x1@alice.com*" by performing the following steps:

1) looking up the IP address of the email server of "*bob.com*" (e.g., by looking up the MX record of "*bob.com*" in DNS), in this case, which is $10.10.1.8$ (see Figure 3.2);

2) hijacking $10.10.1.8$ by announcing a BGP route for the prefix $10.10.1/24$, assuming that $10.10.1/24$ is the most specific prefix containing $10.10.1.8$ on the Internet;

3) requesting password reset for "*x1@alice.com*";

4) intercepting the password reset message sent from "*alice.com*" to "*x1@bob.com*", e.g., by setting up an email server with the IP address $10.10.1.8$. Since the IP prefix containing $10.10.1.8$ has been hijacked, the password reset message will be sent to the attacker instead of the legitimate mail server of "*bob.com*".

5) resetting the password for "*x1@alice.com*" by following instructions in the intercepted password reset message. As a result, the attacker gains access to "*x1@alice.com*".

While some online service providers (e.g., Expedia) may accept requests for password resets without asking for any additional information (except the userid of the account being reset for password), many (e.g., Yahoo) do take additional steps for verifying identities. In other words, additional information is often required to show that you really are the owner of the account whose password is to be reset. For example, Yahoo asks for a date of birth and a postal code, and Ebay asks for a postal code and a phone number. Gmail asks for characters in a picture for countering automatic password reset attacks, but not for identity verification. However, most information requested for countering identity theft could be obtained, e.g., by social engineering.

Phishing. A primary objective of *phishing* is to steal people's confidential information, e.g., credit card numbers, social insurance numbers, date of birth, and home addresses, among others, so that they can be used directly or indirectly (sold to a third party) for financial benefit. A phisher usually sends out spam to a large number of people using well-known sender addresses (e.g., the email address of the security team of a well-known bank) to ask a recipient to reset its account by going to a spammer-controlled website and filling in confidential information. The link to a fraudulent website can be a numeric IP address, an irrelevant domain name, or a domain name very similar to the real one of a claimed organization. The displayed URL which a potential victim sees may also be entirely different than the URL linked to in the underlying html. However, a careful user may be able to find the discrepancy and thus avoid being fooled. The legitimate domain name or URL can also be used if its DNS record on a victim machine (i.e., the machine from which a user clicks the link) is changed (poisoned) to the IP address of the fraudulent website. Again, a careful user may still be able to notice the trick.

To use the legitimate domain or URL of a claimed organization in a phishing email without poisoning a DNS record, a phisher can hijack the IP address space of that organization and set up a fraudulent website using the IP address of the legitimate website. In this way, it will be difficult (essentially impossible) for a user to distinguish a phishing message from a real message (i.e., a message indeed sent by the organization in question). As shown in Figure 6.1, some ASes (more precisely the routing tables of BGP speakers in some ASes) may not be poisoned by a bogus prefix announcement, depending on their locations and relevant routing policies. Thus, users located in these ASes may go to the real website by clicking the link in a phishing email. However, some ASes may be poisoned and their users will face the risk of being phished.

Falsification of AS_PATH

We use Figure 6.2 to illustrate how an attacker might influence traffic flow by manipulating AS_PATH. Suppose AS H multi-homes with D and B; $[D, H]$ is a primary link and $[B, H]$ is a backup link. In the normal situation, traffic destined to H and its customers (e.g., J) will go through link $[D, H]$. When $[D, H]$ fails, $[B, H]$ is used. To achieve this traffic engineering objective, H can legitimately utilize AS_PATH to influence other ASes' routing decisions. For example, H announces $(10/8, [J, H])$ to D (normal BGP operation), and $[(10/8, [J, H, H, H])$ to B (a legitimate traffic engineering technique). After the network converges on $10/8$, all traffic to $10/8$ will be forwarded over link $[H, D]$ to H (see Table 6.4).

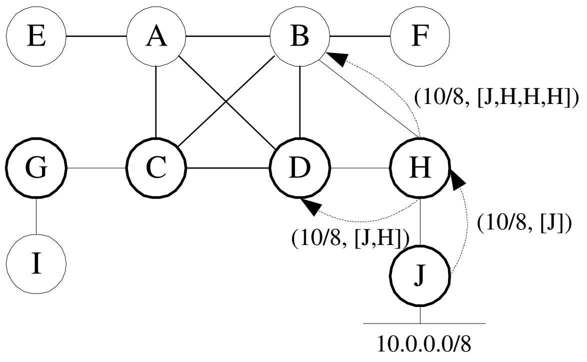


Figure 6.2: Changing traffic flow by AS_PATH falsification

However, B can attract traffic destined to $10/8$ by announcing a route to $10/8$ with a fraudulent AS_PATH, e.g., $(10/8, [J, B])$. Note the $[J, B]$ is shorter than $[J, H, H, H, B]$ which is supposed to be advertised by B. As a result, other ASes may select the route to $10/8$ which goes through B. See Table 6.4 for details of route changes. To summarize, traffic flow can be changed by falsification of AS_PATH.

AS	Route to 10/8 †	AS	Route to 10/8 †
A	(10/8, [J,H,D]) → **(10/8,[J,B])**	F	(10/8, [J,H,H,H,B]) → **(10/8, [J,B])**
B	(10/8, [J,H,H,H]) → **(10/8, [J])**	G	(10/8, [J,H,D,C]) → **(10/8, [J,B,C])**
C	(10/8, [J,H,D]) → **(10/8, [J,B])**	H	(10/8, [J])
D	(10/8, [J,H])	I	(10/8, [J,H,D,C,G]) → **(10/8, [J,B,C,G])**
E	(10/8, [J,H,D,A]) → **(10/8, [J,B,A])**	J	direct route

Table 6.4: Routes to $10/8$ from each AS before and after B announces fraudulent $(10/8, [J, B])$.
† - Note the "after" route as listed herein may not actually exist.

6.3 A Conjectured BGP Attack: The Google Outage

In this section, we present an analysis of real world BGP data collected by the RouteViews project [116] regarding an abnormal announcement of one of Google's prefixes during the period around Google's May 2005 outage [139]. Based on our analysis and communication with related parties, we speculate that BGP prefix hijacking might have contributed to Google's May 2005 outage.

6.3.1 Introduction

Google went down for less than an hour around 22:10, May 07, 2005 UTC [126], which was acknowledged to have been caused by Google internal DNS misconfigurations. Due to the outage itself and some uncountered claims [37] of traffic redirection during the outage, we studied how the Border Gateway Protocol (BGP) [109] behaved on the Internet during the period when Google was down.

We used the BGP data continually collected by the RouteViews Project [116] to analyze BGP announcements of Google's prefixes from January 1, 2005 to May 25, 2005. Interestingly, we discovered that at 14:37:56, May 07, 2005 UTC, prior to the service outage, AS174 operated by Cogent, which is apparently independent from Google, mysteriously originated routes for $64.233.161.0/24$, one of the prefixes assigned to Google. This prefix contains the IP addresses associated with www.google.com returned from the DNS during that period of time (based on the DNS queries from a number of computers within Canada). This erroneous prefix origination did not occur prior to this specific instance, nor has it re-occurred thereafter. None of the traffic engineering approaches (e.g., multi-homing, aggregation, etc.) which we are aware of could explain this announcement.

The coincidence in time with Google's service outage leads us to speculate that BGP might have been exploited by malicious parties to intentionally target Google, although we are not able to conclude this definitively. If our speculation is indeed true, it should raise alarms that attacks exploiting routing vulnerabilities, which were forewarned about 16 years ago by Perlman and Bellovin [103, 11], are now reality.[3]

6.3.2 Multiple Origin ASes (MOAS)

Each AS is assigned one or more IP prefixes by the organization running that AS, which either obtains the prefixes from the address authority (e.g., a RIR) or from another organization (e.g., an upstream ISP). It is usually the AS to which a prefix has been legitimately assigned which will originate a route for that prefix. In other words, there should be only one origin AS for each prefix [48]. However, some operational practices make it possible for two or more ASes to originate a route for the same prefix. This is often referred to as Multiple Origin ASes (MOAS) [152]. Here we describe three cases under which MOAS could occur (see [152] for a more detailed study).

Multi-homing

Many organizations connect to the Internet via two or more ISPs which may run different ASes. A multi-homing organization (X) may or may not run its own AS. In the former case, X may use a valid AS number or a private AS number. If X participates in inter-domain routing using a valid AS number, it should be the only origin AS for its prefixes. In other words, there should be no MOAS of its prefixes. If X uses a private AS number, its service providers will strip the private AS number from all routes originated by X and replace it with their own AS numbers. Thus, there will be multiple origin ASes for X's prefixes. If X does not participate in inter-domain routing (i.e., it is not running its own AS) and simply delegates its prefixes to all of its service providers, it is equivalent to the

[3]It is known that spammers commonly hijack prefixes using BGP [16]. However, they usually hijack unused address space, resulting in no harm to existing traffic flow on the Internet.

case of using a private AS number from the perspective of prefix origin. Thus, MOAS will be observed for X's prefixes.

Anycast Routing

Anycasting [100] refers to communication between a client and one of the servers within a group sharing a common IP address (anycast address). While it appears counter-intuitive for multiple servers to be configured with the same IP address, anycasting offers attractive benefits such as reduced response delay, load-balancing, improved availability, among others. Thus, it has gained popularity among application service providers. For instance, some of the root DNS servers (e.g., F-root [60] and K-root DNS [96]) are implemented using anycast. Anycast routing refers to a practice that supports anycasting in the network layer by ensuring that a datagram sent to an anycast address is transmitted to at least one of the servers within an anycast group, likely the one "closest" to the originating network. To do so, an anycast address space will be announced by multiple routers into an internetwork. For example, if an application service provider distributes its anycasting service across different geographic locations, each of which connects to the Internet via a different ISP, then multiple origins of prefixes containing this anycast address space will be announced via BGP by different ASes. In other words, anycast routing can cause MOAS.

Prefix Hijacking

A malicious AS Y may announce a prefix assigned to another AS X without any legitimate reason. As a result, traffic originated from some part of the Internet and destined to X may be attracted to Y; such traffic can then be manipulated in many ways. For example, traffic can be dropped; modified and then resent back to X through a tunnel; or redirected to other locations [12].

6.3.3 Analysis Results

We used BGP data collected on a regular basis by the RouteViews Project [116] to analyze announcements of the prefixes assigned to Google. There are several BGP routers maintained by the RouteViews project, each of which establish BGP sessions with a number of ASes on the Internet. These RouteViews routers only collect BGP update messages from their neighboring ASes and do not inject any update messages back. In the absence of access to the BGP data from ASes of our choice, the BGP data collected by the RouteViews project is of central importance to our analysis. We combine the BGP data collected by different routers to get a better view of BGP updates on the Internet. However we acknowledge that this view is still limited, and does not allow us full confidence in our conclusions on the actual reason of the Google incident, or to deduce the full impact of the incident on the Internet.

Based on ARIN's *whois* database [5], we learned that Google has AS number 15169, and is assigned three /19 and one /22 address blocks which contain in total 100 blocks of /24 prefixes. Google chooses to announce /24 prefixes instead of /19 or /22, which is a common practice for avoiding traffic destined to one AS being attracted to other ASes which might have announced that AS's prefixes with longer prefixes. Announcing prefixes longer than 24 bears the risk of being rejected since 24 is the longest prefix acceptable to many ISPs. Based on the BGP data we used, AS15169 has 25 neighbors including AS174.

Our hypothesis is that if someone tried to attract traffic destined to Google by prefix hijacking, we should see MOAS regarding some of Google's prefixes. Thus we first looked at one BGP routing information base (RIB) collected near the time when Google went down. We discovered one AS (i.e., AS174) in fact originated $64.233.161.0/24$, one of the prefixes assigned to Google, before Google's outage. We then analyzed the RIBs collected over a number of days to determine the duration of this mysterious announcement, or what we call the *MOAS period*. We then analyzed one BGP RIB per day from January 1, 2005 to the start of the MOAS period, namely in the *pre-MOAS* period, and one BGP RIB per day from the end of the MOAS period to May 25, 2005, namely in the *post-MOAS* period. We then compared how AS174 originated routes for the prefixes assigned to Google over these periods.

We observed that AS174 started to originate Google's prefix from 14:37:56, May 07, 2005 UTC and stopped after 10:52:00, May 09, 2005 UTC; we call this the *MOAS period*. We next report our observations respectively for the three periods (pre-MOAS, during-

MOAS, and post-MOAS period).

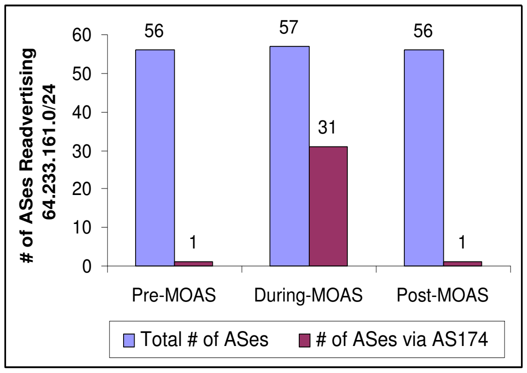


Figure 6.3: Total number of ASes re-advertising routes for $64.233.161.0/24$, and total number of ASes via AS174

Pre-MOAS Period

Prior to the MOAS period, we observed that AS15169 originated $64.233.161.0/24$ to 25 direct neighbors including AS174, which further re-advertise the route to 31 remote ASes (i.e., not directly connected to AS15169). In total, we observed that 56 ASes re-advertised routes for $64.233.161.0/24$, and that no AS other than AS174 itself re-advertised routes with an AS_PATH involving AS174 (hereafter "via AS174"). In other words, we did not observe any AS re-advertising routes for $64.233.161.0/24$ via AS174 (see Figure 6.3). Thus, it is very likely that in the pre-MOAS period, traffic destined to $64.233.161.0/24$ passed through AS174 only if the traffic originated either from AS174 or from its customers.

During MOAS Period

During the MOAS period, AS174 originated routes for $64.233.161.0/24$ instead of re-advertising the one originated by AS15169, thus poisoning many ASes' routing tables. We observed in total 57 ASes re-advertising the routes for $64.233.161.0/24$, among which 31 ASes preferred (i.e. selected) the routes originated by AS174. Among these 31 ASes, 28 of them switched from the routes originated by AS15169 to those originated by AS174,

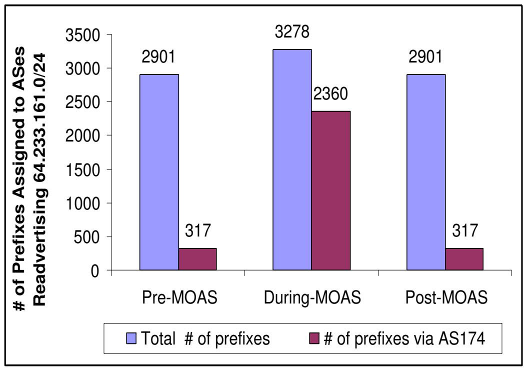


Figure 6.4: Total number of prefixes assigned to the ASes re-advertising routes for $64.233.161.0/24$, and the number of prefixes assigned to poisoned ASes

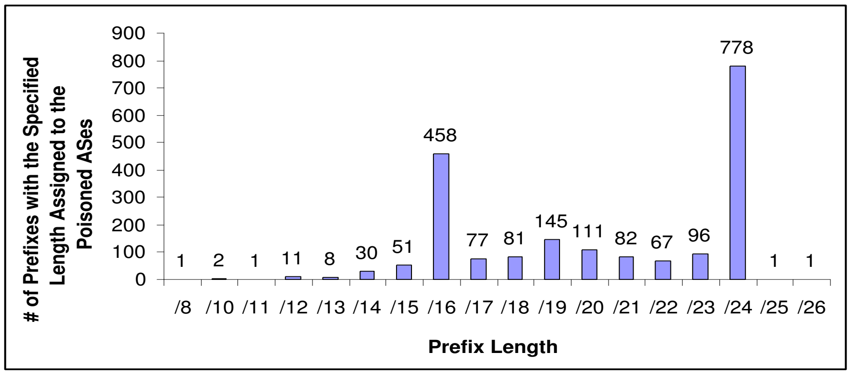


Figure 6.5: Number of prefixes with specified length assigned to the poisoned ASes

including 8 of AS15169's direct neighbors. We refer to these as *poisoned* ASes. Some of the poisoned ASes are large ISPs, such as AS701 (UUNET), AS2497 (IIJ), AS3561 (C&W), and AS7018 (AT&T). Geographically, they span almost every continent. In terms of percentage, 49.1% (28 out of 57) of re-advertising ASes were poisoned, including 32% of AS15169's direct neighbors (see Figure 6.5).

We examined the prefixes assigned to the poisoned ASes for perspective on the amount of address space from which traffic originated toward Google might have been attracted to AS174. Based on the data we used, in total 2003 prefixes were assigned to the 28 poisoned ASes. Figure 6.5 presents those prefixes arranged by prefix length. This demonstrates that not only prefixes containing relatively small address ranges (e.g., /24) are affected, but also some prefixes containing larger address space (e.g., with a length shorter than 16). This is not a surprise since some of the poisoned ASes are large ISPs which hold a large amount of address space.

Post-MOAS Period

After the MOAS period, we observed in total 56 ASes re-advertising the route for prefix $64.233.161.0/24$ originated by AS15169. There was no AS except AS174 whose re-advertisements for $64.233.161.0/24$ had an AS_PATH involving AS174. This is the same situation as in the pre-MOAS period.

Regarding other prefixes assigned to Google, we did not observe any multiple origins for any of these, neither by AS174 nor any other ASes during the three periods.

6.3.4 Our Interpretation

First, we suggest that none of the legitimate reasons as discussed in §6.3.2 can explain why AS174 would originate the IP prefix assigned to Google. We next consider the possibility that this was caused by misconfiguration or malicious attack.

Misconfiguration

We consider two types of misconfiguration which might result in the MOAS regarding $64.233.161.0/24$. *Firstly*, many ASes use centralized databases, which contain IP prefixes assigned to an AS, to automatically generate configuration files for BGP speakers within an AS. If a prefix f_x assigned to AS X erroneously enters into the central database from which AS Y draws its BGP speaker configurations, AS Y might erroneously originate routes for f_x. So it might be possible that $64.233.161.0/24$ got into AS174's configuration database and AS174 updated some of its BGP speakers using the misconfigured database before the MOAS period. *Secondly*, it is also possible that one or more BGP speakers in AS174 were misconfigured such that they stripped the origin AS from a route when re-advertising that route. However, this second situation appears very unlikely since we did not observe the same misbehavior happening on any other prefixes announced by AS15169 to AS174.

Malicious Attack

It is also possible that one or more of the BGP speakers in AS174 were compromised and used to influence traffic sent to Google. While some traffic destined to $64.233.161.0/24$ indeed was forwarded through AS174 to AS15169, and an attacker with control of a BGP speaker in AS174 could get access to that traffic without hijacking Google's prefix, the amount of such accessible traffic is limited and a large portion was forwarded to AS15169 by its neighbors other than AS174. Thus, hijacking the prefix allows an attacker to gain access to more traffic destined to the hijacked address space. An attacker would have much freedom in manipulating the attracted traffic, depending on how much control he has over the compromised routers. A simple attack is to redirect traffic to a black hole by installing unreachable static routes in the routing table of the compromised router in AS174. An advanced attack is to redirect attracted traffic to a location (e.g., a compromised PC) where their destination IP addresses are replaced by new IP addresses (e.g., the IP addresses of other websites). The modified traffic is then re-injected into the Internet [13, 12]. If an attacker chose to not manipulate the attracted traffic, the traffic might still be able to reach its intended ultimate destination, i.e., Google, since AS174 has direct connectivity to AS15169.

6.3.5 Communication with Google and Cogent

We made attempts to obtain inputs from both parties involved in this incident, i.e., Google and Cogent. Here we summarize our communication with them.

Communications with Google

Our communication with Google [134] was useful on several fronts. First, we acquired better understanding of Google's internal DNS failures which led to the outage. The failure was caused by an unreadable configuration file consisting of almost all DNS A records that was mistakenly pushed to all Google DNS servers. As a result, all DNS queries sent to Google's DNS servers were returned with answers of no A records, which in turn caused the service outage. We also learned that Google does not use anycast routing, which could also cause MOAS.

Second, we understood that most of the reported traffic redirections were mainly caused by browsers trying to append a Top Level Domain or TLD (e.g., .com) when a supplied domain name could not be resolved. In this case, many queries intended to *google.com* ended up at sites such as *google.com.com*, which happens to be hosted by *sogosearch.com*. In addition, some Internet Service Providers (ISPs) return pointers to some special sites whenever a domain name search fails, which might have also caused some redirections.

Third, such MOAS is not considered legitimate by Google, and Google did experience problems with $64.233.161.0/24$ during the period of MOAS at the Point of Presence (PoP) through which Google peers with Cogent. However, statistical traffic differences were not apparent given the large volume of traffic received by Google. We also came to agreement that a few uncountered claims [37], could indeed be caused by redirections involving BGP, albeit not conclusively. For example, some traffic sent to "*www.google.com*" were redirected to "*search.msn.com*" (cf. [37]). Such redirection appears unlikely caused by attempts to append a TLD.

Fourth, our draft report served the purpose of alerting Google personnel to BGP security issues. After reading our draft, we were told that Google's network operation group was " sufficiently disturbed" by the fact that BGP can be used for prefix hijacking, and are considering setting up infrastructure for monitoring apparent hijacking of Google's IP prefixes.

Communications with Cogent

We have made several attempts to discuss this incident with individuals from Cogent. We first contacted Cogent Network Operation Center (NOC) at "*noc@cogentco.com*" [132]. We were asked for the aspath involved in the incident, and our relationship with Google. After providing the requested information, we did not heard back further.

Our second attempt involved sending a request [135] to the NANOG mailing list, asking for a technical contact at Cogent to discuss BGP issues. Our email to the NANOG mailing list resulted in email exchange [131] with an employee from a Cogent help-desk who advised us that she/he was not able to discuss this incident with us due to privacy agreements. A separate email contact [133] through Cogent NOC proved to be equally unhelpful. While this does not provide any evidence supporting our conjectures, neither does it contradict any.

6.3.6 Discussion

While some MOAS is valid, we do not know if there was any legitimate reason behind AS174's origin of Google's prefix. To think negatively, it is possible that one or more BGP speakers within AS174 misbehaved. On the extreme, the misbehaving BGP speakers might have been controlled by an attacker which then redirected Google's traffic to some other sites of the attacker's choice.

This incident differs from others [89, 19] in that it is subtle and might be a real malicious activity specifically targeting an organization, while others are known to have been caused by misconfiguration and have no specific target. This accident, among others clearly demonstrate that BGP is extremely vulnerable and must be secured to protect the Internet infrastructure, which is now clearly recognized as a critical infrastructure and is on the path to replace many of the traditional communication infrastructures (e.g., telephony).

6.4 BGP Security Goals

We seek to design secure protocol extensions to BGP which can resist the threats as discussed in §6.2.2, i.e., primarily falsification of BGP update messages. As with most other secure communication protocols, BGP security goals must include data origin authentication and data integrity. In addition, verification of the propriety of BGP messages is required to resist falsification attacks. Specifically, the propriety of NLRI and AS_PATH should be verified. All verification will be performed most likely by a BGP speaker online, but possibly by an operator off-line, which is not discussed in this thesis.

We summarize five security goals for BGP (cf. [71], also see [138, 140]), for reference later in §6.5, §6.6, §7.1.1 and §7.4. G1 and G2 relate to data origin authentication, G3 to data integrity, and G4 and G5 to the propriety of BGP control messages. These five security goals address a large number of serious threats against BGP. Thus it is highly desirable for any serious BGP security proposal to achieve them. However, these alone should not be considered as sufficient for BGP security, since other threats (e.g., replaying) remain (see §6.2.2).

G1. *(AS Number Authentication)* It must be verifiable that an entity using an AS number s_i as its own is in fact an authorized representative of the AS to which a recognized AS number authority assigned s_i.

G2. *(BGP Speaker Authentication)* It must be verifiable that a BGP speaker, which asserts an association with an AS number s_i, has been authorized by the AS to which s_i was assigned by a recognized AS number authority.

G3. *(Data Integrity)* It must be verifiable that a BGP control message has not been illegally modified en route.

G4. *(AS Path Verification)* It must be verifiable that an AS_PATH ($p_k = [s_1, s_2, \ldots, s_k]$) of a BGP route m being propagated consists of a sequence of ASes traversed by m in the specified order, i.e., m originated from s_1, and has traversed $s_2, \ldots, s_k$ in order.

G5. *(Prefix Origin Authentication)* It must be verifiable that it is proper for an AS to originate an IP prefix. It is *proper* for AS s_1 to originate prefix f_1 if 1) f_1 is indeed assigned to s_1; or 2) s_1 is assigned a set F_1 of prefixes; s_1 has received a set of routes with a set F_2 of prefixes; and f_1 is aggregated from F_1, F_2 or both such that $\forall f_x \subseteq f_1, f_x \subseteq F_1 \cup F_2$.[4]

6.5 Pretty Secure BGP (psBGP)

psBGP makes use of a centralized trust model for authenticating AS numbers and AS public keys. RIRs are the root trusted certificate authorities. In psBGP, each AS s is issued a public key certificate (ASNumCert), signed by one of the RIRs (say T), denoted by $(k_s, s)_{\overline{k_T}}$. Such an AS creates and signs two data structures: a SpeakerCert $(k'_s, s)_{\overline{k_s}}$

[4]If f_1 is not assigned to s_1 and $\exists f_x \subseteq f_1$ such that $f_x \nsubseteq F_1 \cup F_2$, then s_1 *overclaims* IP prefixes, which is a type of falsification.

binding a different public key k'_s to s; and a *prefix assertion list* (PAL). The latter, pal_s, is an ordered list: the first assertion is for s itself and the rest are endorsements by s for each of s's neighbors ordered by AS number. Figure 6.6 illustrates the certificate structure used in psBGP. In what follows, we start with a description of a rating mechanism used by each AS in determining its confidence in an AS_PATH or a prefix assertion. We next describe psBGP with respect to the above five security goals: G1-G4 here, and G5 in §6.6.

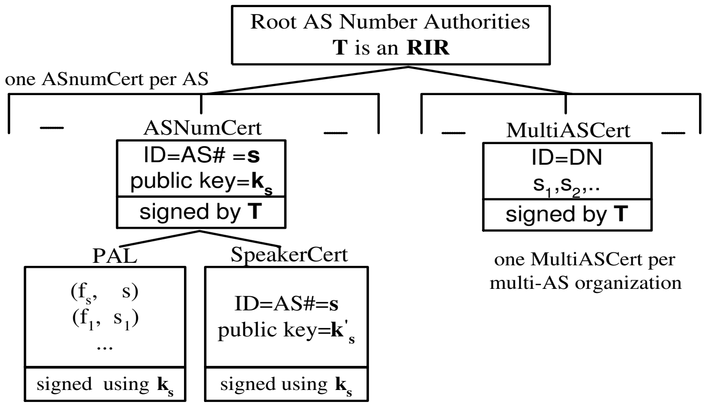


Figure 6.6: psBGP certificate structure

6.5.1 A Rating Mechanism for psBGP

In psBGP, each AS s_i rates every other AS s_j with a value in $[0, 1]$, denoted by $r_i(s_j)$, representing s_i's confidence or belief in s_j's trustworthiness, i.e., in an assertion made by s_j such as a digitally signed AS_PATH or a prefix assertion or endorsement. $r_i(s_j)=0$ or 1 respectively indicates s_i fully distrusts or trusts s_j. When there is no ambiguity, we omit the subscript on r in $r_i(s_j)$.

While each AS has freedom in determining how to rate other ASes, we suggest the following guidelines: an RIR should be fully trusted (i.e., rated 1); a direct neighbor might be expected, in many cases, to be more trustworthy than a remote AS; and a majority of ASes should be neutrally trusted, e.g., rated 0.5. We make use of the belief combination rule (see equation 4.1 in §4.5.1) for computing the confidence value in a statement which is consistent among a set of assertions made by a group of ASes (a *corroborating* group) based on one's ratings of those ASes. We consider two types of consistency in psBGP:

path-consistency and *prefix-consistency*. The former is regarding the consistency among a set of digital signatures over an AS_PATH (see Definitions 4 and 5 in §6.5.5). The latter is regarding the consistency of a prefix assertion and a prefix endorsement (see Definition 7 in §6.6.1).

Recall §4.5, $\lambda_{[1..n]}$ denotes a common subset that can be derived from each of the n assertions made by a group of AS $s_1, .., s_n$. The precise meaning of $\lambda_{[1..n]}$ depends on the type of consistency in question. In prefix-consistency, if a_{s_1} is a prefix assertion $(f_1, s_1)_{s_1}$, and $a_{s_2}, .., a_{s_n}$ prefix endorsements $(f_1, s_1)_{s_2}, .., (f_1, s_1)_{s_n}$, then $\lambda_{[1..n]}$ represents a prefix assignment of s_1, i.e., s_1 is assigned a prefix f_1. In path-consistency, if $a_{s_1}{=}\{f_1, [s_1, s_2]\}_{s_1}, .., a_{s_n}{=}\{f_1, [s_1, .., s_n, s_{n+1}]\}_{s_n}$ are digital signatures present with a BGP route $m{=}(f_1, p_n{=}[s_1, .., s_n])$, then λ_{s_1,s_2} represents a statement that p_n contains a path segment $[s_1, s_2]$, λ_{s_2,s_3} represents a statement that p_n contains a path segment $[s_2, s_3]$, and so on.

For later cross-reference, we next derive two algorithms for respectively increasing and decreasing belief in $\lambda_{[1..n]}$ from equation (4.1) presented in §4.5. Algorithm 5 describes how to increase one's confidence in $\lambda_{[1..(n-1)]}$ when an additional endorsement is obtained, e.g., from s_n. Algorithm 6 describes how to reduce one's confidence in $\lambda_{[1..n]}$ when (without loss of generality) s_n's endorsement is withdrawn.

Algorithm 5 Adding new endorsement from AS s_n

```
1: INPUT: $b(\lambda_{[1..(n-1)]}), r(s_n)$
2: OUTPUT: $b(\lambda_{[1..n]})$
3: $t \leftarrow r(s_n) + [1 - r(s_n)] \cdot b(\lambda_{[1..(n-1)]})$
4: return($t$)
```

Algorithm 6 Removing existing endorsement from AS s_n

```
1: INPUT: $b(\lambda_{[1..n]}), r(s_n)$
2: OUTPUT: $b(\lambda_{[1..(n-1)]})$
3: $t \leftarrow \frac{b(\lambda_{[1..n]}) - r(s_n)}{1 - r(s_n)}$
4: return($t$)
```

6.5.2 AS Number Authentication in psBGP (G1)

Following S-BGP [120], psBGP makes use of a centralized PKI for AS number authentication, with five root Certificate Authorities (CAs), corresponding to the five existing RIRs. When an organization B applies for an AS number, besides supplying documents currently required (e.g., routing policy), B additionally supplies a public key, and should be required to prove possession of the corresponding private key [120, 1]. When an AS number is granted to B by an RIR, a public key certificate (ASNumCert) is also issued, signed by the issuing RIR, binding the public key supplied by B to the granted AS number. An AS number s is called *certified* if there is a valid ASNumCert $(k_s, s)_{\overline{k_T}}$, binding s to a public key k_s signed by one of the RIRs, T.

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug
Start of month	16 554	16 708	16 879	17 156	17 350	17 538	17 699	17 884
Removed during month	153	137	155	174	138	179	164	N/A
Added during month	307	308	432	368	326	342	349	N/A

Table 6.5: AS number dynamics from January 1 to August 1, 2004

The proposed PKI for authenticating AS numbers is practical for the following reasons. First, the roots of the proposed PKI are the existing trusted authorities of the AS number space, removing a major trust issue which is one of the most difficult parts of a PKI: the root of a PKI must have control over the name space involved in that PKI. Thus, RIRs are the natural and logical AS number certificate authorities. We acknowledge that non-trivial (but feasible) effort might be required for implementing such a PKI. Second, the number of ASes on the Internet and its growth rate are relatively manageable (see Table 6.5). Considering there are five RIRs, the overhead of managing ASNumCerts should certainly be manageable, given that larger PKIs are currently commercially operational [47].

To verify the authenticity of an ASNumCert, an AS must have the trusted public key (or verifiable certificate) of the signing RIR. These few root trusted public key certificates can be distributed using *out-of-band* mechanisms. ASNumCerts can be distributed with BGP update messages. An ASNumCert should be revoked when the corresponding AS number is no longer used or is reassigned to another organization. Issues of revocation are extremely important, although not directly dealt with in this thesis. We restrict comment to the observation that revocation is a well-studied, albeit still challenging issue (e.g., see [1]). So far, we assume that every AS has the public key certificates of RIRs and can obtain the ASNumCerts of any other ASes if and when necessary.

In discussion related to various proposals for securing BGP, there is much debate in the BGP community on the architecture for authenticating the public keys of ASes, particularly on the pros and cons of using a strict hierarchical trust model vs. a distributed trust model, e.g., a web-of-trust model [153]. We make use of a strict hierarchical trust model (with depth of one) for authenticating AS numbers and their public keys to provide a strong guarantee of security. Therefore, it would appear difficult for an attacker to spoof an AS in psBGP as long as it cannot obtain the private key corresponding to the public key of an ASNumCert signed by an RIR, or the signing key of an RIR. In contrast, a web-of-trust model does not provide such a guarantee. Other issues that arise with a web-of-trust model include: trust bootstrapping, trust transitivity, and vulnerability to a single misbehaving party [85, 108].

6.5.3 BGP Speaker Authentication in psBGP (G2)

An AS may have one or more BGP speakers. A BGP speaker must be authorized by an AS to represent that AS to establish a BGP session with a BGP speaker in another AS. In psBGP, an AS with a certified ASNumCert issues an operational public key certificate shared by all BGP speakers within the AS, namely SpeakerCert. A SpeakerCert is signed using the private key of the issuing AS, corresponding to the public key in the AS's ASNumCert (see Figure 6.6). A SpeakerCert is an assertion made by an AS that a BGP speaker with the corresponding private key is authorized to represent that AS. SpeakerCerts can be distributed with BGP update messages.

We consider three design choices for BGP speaker authentication: a) each BGP speaker has a distinct key pair and is issued a unique public key certificate; b) group signatures (e.g., see [20]) are used, i.e., each BGP speaker has a unique private key but shares a common public key and public key certificate with other speakers in the same AS; or c) all BGP speakers in a given AS share a common public-private key pair. Choice a) provides stronger security in theory but requires more certificates, and discloses BGP speaker identities, which may introduce competitive security concerns [145]. Choice b) again provides stronger security in theory, requires the same number of certificates, and does not disclose BGP speaker identities, but involves a more complex system, which we believe significantly reduces its chances of being commercially accepted and securely deployed. We recommend choice c) primarily for its operational simplicity. However, an operator has the ultimate freedom in determining how many BGP speakers within an AS will share a public-private key pair, i.e., finding a balance between choice a) and c).

The private key corresponding to the public key of a SpeakerCert is used for establishing secure connections with neighbors (§6.5.4), and for signing BGP update messages. Therefore, it would most likely be stored in the communication device associated with a BGP speaker. In contrast, since the private key corresponding to the public key of an ASNumCert is only used for signing a SpeakerCert and a PAL, it need not be stored in a BGP speaker. Thus, compromising a BGP speaker at most discloses the private key of a SpeakerCert, requiring revocation and reissuing of a SpeakerCert, without impact on an ASNumCert. This separation of ASNumCerts from SpeakerCerts provides a more conservative design (from a security viewpoint), and distributes from RIRs to ASes (or their delegated certificate service providers) the workload of certificate revocation and reissuing resulting from BGP speaker compromises. In summary, an ASNumCert must be revoked if the corresponding AS number is re-assigned or the corresponding key is compromised; a SpeakerCert must be revoked if a BGP speaker in that AS is compromised, or for other reasons (e.g., if the private key is lost).

6.5.4 Data Integrity in psBGP (G3)

To protect data integrity, BGP sessions between neighboring ASes must be protected. Following S-BGP and soBGP, psBGP uses IPsec Encapsulating Security Payload (ESP) [70] with null encryption for protecting BGP sessions. Since many existing BGP speakers implement TCP MD5 [50] with manual key configurations for protecting BGP sessions, it must be supported by psBGP as well. In psBGP, automatic key management techniques can be implemented to improve the security of TCP MD5 as each BGP speaker has a public-private key pair (common to all speakers in that AS).

6.5.5 AS_PATH Verification in psBGP (G4)

Regarding "AS_PATH security", different security solutions of BGP define it differently. In S-BGP, the security of an AS_PATH is interpreted as follows: for every pair of ASes on the path, the first AS authorizes the second to further advertise the prefix associated with this path. In soBGP [144], it is defined as the plausibility of an AS_PATH, i.e., if an AS_PATH factually exists on the AS graph (whether or not that path was actually traversed by an update message in question is irrelevant).

Since AS_PATH is used by the BGP route selection process, greater assurance of the integrity of an AS_PATH increases the probability that routes are selected based on proper information. Without strong guarantees of AS_PATH integrity, an attacker may be able to modify an AS_PATH in a such way that it is still plausible in the AS graph and is also more favored (e.g., with a shorter length) by receiving ASes than the original path. In this way, a receiving AS may be misled to favor a falsified route over correct routes, possibly influencing traffic flow. Thus, in our view, it is not sufficient to verify only the existence/non-existence of an AS_PATH if greater assurance of the integrity of an AS_PATH can be provided at acceptable cost.

To provide a mechanism for configuring the balance between security and efficiency, we choose the S-BGP approach combined with the rating mechanism described in §4.5 and §6.5.1 to determine dynamically (at run-time) the number of digital signatures on an AS_PATH to be verified. We first give the definition of *path-consistency*, then present how to calculate a confidence value in an AS_PATH.

Definition 4 (Path-Consistency). *Let* $m{=}(f_1, p_k{=}[s_1, .., s_k])$ *be a BGP route, and* $sig_i{=}\{f_1, p_i\}_{s_i}$ *be a digital signature generated by a psBGP-enabled BGP speaker in* $s_i, 1{\le}i{\le}k$*, where* $\{p_i\}_{s_i}{=}[s'_1, .., s'_{i+1}]$ *is the path signed by* s_i*.* $\{p_i\}_{s_i}$ *is consistent with*

p_k if $\{p_i\}_{s_i}$ consists of the first $i{+}1$ ASes on p_k (i.e., $s'_1{=}s_1, .., s'_{i+1}{=}s_{i+1}$) when $1{\leq}i{\leq}k{-}1$, or consists of p_k appended by another AS s_{k+1} when $i{=}k$.

Definition 5 (Signed-Path Consistency). *Let $m{=}(f_1, p_k{=}[s_1, .., s_k])$ be a BGP route, and $sig_i{=}\{f_1, p_i\}_{s_i}$, $sig_j{=}\{f_1, p_j\}_{s_j}$ the digital signatures generated by two psBGP-enabled ASes s_i and s_j, $1{\leq}i, j{\leq}k$, on p_k. $\{p_i\}_{s_i}$ and $\{p_j\}_{s_j}$ are consistent if they both are consistent with p_k.*

Two consistent signed paths $\{p_i\}_{s_i}$ and $\{p_j\}_j$ contain common subset λ_{s_i,s_j}. For example, if $\{p_2\}_{s_2}{=}[s_1, s_2, s_3]$, $\{p_4\}_{s_4}{=}[s_1, s_2, s_3, s_4, s_5]$, λ_{s_2,s_4} could be an assertion that p_k contains the path segment $[s_2, s_3]$ since both s_2 and s_4 assert it in their signed path. As a result, one may expect the belief in λ_{s_2,s_4} will increase, which may further contribute to the belief in p_k in some way. However, the definition of path confidence in psBGP is more restrictive. In psBGP, the belief in p_k, $b(p_k)$, is defined as the sum of the belief of each assertion that p_k contains a two-AS path segment $[i, i+1]$, $1{\leq}i{\leq}k{-}1$, divided by the total number of those path segments $k{-}1$.

Definition 6 (Path Confidence). *Let $m{=}(f_1, p_k{=}[s_1, .., s_k])$ be a BGP route, and $\lambda_{s_i,s_{i+1}}$ be the assertion that p_k contains a two-AS path segment $[s_i, s_{i+1}]$. The belief in p_k is defined as: $b(p_k) = \frac{1}{k-1}\sum_{i=1}^{i=k-1} b(\lambda_{s_i,s_{i+1}})$.*

The belief in the assertion $\lambda_{s_i,s_{i+1}}$ that p_k contains a two-AS path segment $[s_i, s_{i+1}]$ is obtained exclusively from the signed paths by s_i and s_{i+1} (i.e., $\{p_i\}_{s_i}$, $\{p_{i+1}\}_{s_{i+1}}$) since two ASes have direct knowledge over the path segment between themselves. The signed path by another AS, e.g., s_{i+2}, may also contain $[s_i, s_{i+1}]$, but it does not contribute to the belief in $\lambda_{s_i,s_{i+1}}$ since s_{i+2} apparently does not have direct knowledge of $[s_i, s_{i+1}]$ and its signed path may be dependent on the path signed by s_i or s_{i+1}.

If one AS on $[s_i, s_{i+1}]$ is non-psBGP enabled and does not digitally sign its path, the belief in $\lambda_{s_i,s_{i+1}}$ is then solely derived from the signed path of the other AS. If neither of them has signed the path, i.e., $\{p_i\}_{s_i}$ and $\{p_{i+1}\}_{s_{i+1}}$ are null, there is no evidence to believe $\lambda_{s_i,s_{i+1}}$. In this case, $b(\lambda_{s_i,s_{i+1}})$ is set to 0.

In psBGP, a minimum of two digital signatures must be verified if two or more are present on an AS_PATH p_k. The exact number of digital signatures to be verified depends on a verifying AS s_{k+1}'s ratings of the ASes which have signed p_k, and a local configurable confidence threshold $\theta_{k+1}{\geq}0$. Verification of p_k starts from the digital signature generated by the last AS s_k on p_k, and moves toward the first AS s_1. Upon a digital signature sig_i verifying successfully, i.e., sig_i is valid and $\{p_i\}_{s_i}$ is consistent with p_k, the belief in the assertion $\lambda_{s_i,s_{i+1}}$ ($1{\leq}i{\leq}k{-}1$) that p_k contains $[s_i, s_{i+1}]$ is recomputed (using Algorithm 5) and the current belief in p_k is updated (see Definition 6). If $b(p_k)$ is no less then s_{k+1}'s confidence threshold θ_{k+1}, i.e., $b(p_k){\geq}\theta_{k+1}$, then p_k is accepted. Otherwise, more digital signatures are verified (see Algorithm 7) until:

a) one digital signature verification fails, in which case p_k is rejected; or

b) $b(p_k){\geq}\theta_{k+1}$, in which case p_k is accepted; or

c) all digital signatures present on p_k have been verified successfully, in which case p_k is accepted regardless of $b(p_k)$.

Examining Algorithm 7 (line 5), note that if θ_{k+1} is set to a value higher than 1, then since $0{\leq}b(p_k){\leq}1$, $b(p_k)$ will always be less than θ_{k+1}. $i{\geq}1$ remains true until all digital signatures are verified. Thus, to always verify all digital signatures present on any received AS_PATH for maximal assurance of path integrity, s_{k+1} can set $\theta_{k+1}{>}1$ (e.g., $\theta_{k+1}{=}1.1$).

If $\theta_{k+1}{=}0$, $b(p_k){<}\theta_{k+1}$ is always false. Once two digital signatures have been verified successfully, $n{<}2$ remains false. Thus, no additional digital signature will be verified. Such a configuration meets the minimal requirement by psBGP and achieves maximal efficiency. For $0{<}\theta_{k+1}{\leq}1$, the number of digital signatures on an AS_PATH to be verified depends on s_{k+1}'s rating of each signing AS on the path.

Such configuration flexibility is in line with the recommendation that "a good initial solution is one that can easily be upgraded to handle increased threats" [17]. For example, an AS with constrained hardware resources (e.g., CPU) can choose to verify fewer digital signatures on an AS_PATH by setting a lower threshold, while other ASes may choose to verify more or all digital signatures on a signed AS_PATH to achieve a higher assurance of AS_PATH integrity.

We refer the AS_PATH verification in psBGP as *stepwise integrity*, which allows confidence in AS_PATH integrity to be formed based on local parameters, and without requiring

Algorithm 7 AS_PATH verification (by s_{k+1})

```
1:  GLOBAL: threshold $\theta_{k+1}$; $s_{k+1}$'s trust ratings $r(s_1), .., r(s_k)$
2:  INPUT: $k, p_k = [s_1, .., s_k]; sig_1, .., sig_k$
3:  OUTPUT: ACCEPT or REJECT the AS_PATH $p_k$
4:  $i \leftarrow k; n \leftarrow 0; b \leftarrow 0$    /* $b$ represents $b(p_k)$ */
5:  while $i \geq 1$ and $(b < \theta_{k+1}$ or $n < 2)$ do
6:      if $sig_i = \phi$ then
7:          $x \leftarrow 0$    /* $s_i$ has no contribution to belief in $\lambda_{s_{i-1},s_i}$ or $\lambda_{s_i,s_{i+1}}$ */
8:      else if $sig_i$ fails verification then
9:          return(REJECT)
10:     else
11:         $n \leftarrow n+1; x \leftarrow r(s_i)$
12:     endif
13:     if $i = k$ then
14:         $b_2 \leftarrow 0; b_1 \leftarrow x$    /* initial belief in $\lambda_{s_{k-1},s_k}$ */
15:     else if $2 \leq i \leq k-1$ then
16:         $b_2 \leftarrow$ Algorithm5($x, b_1$)    /* final belief in $\lambda_{s_i,s_{i+1}}$ */
17:         $b_1 \leftarrow x$    /* initial belief in $\lambda_{s_{i-1},s_i}$ */
18:     else if $i = 1$ then
19:         $b_2 \leftarrow$ Algorithm5($x, b_1$)    /* final belief in $\lambda_{s_1,s_2}$ */
20:     endif
21:     $b(p_k) \leftarrow b(p_k) + \frac{b_2}{k-1}$    /* update belief in $p_k$ */
22:     $i \leftarrow i-1$
23: return(ACCEPT)
```

all ASes on the AS_PATH to digitally sign the path, nor verification of all digital signatures present. In contrast, the S-BGP AS_PATH verification approach provides *full integrity*, but requiring full adoption of S-BGP by all ASes on the path and verification of all digital signatures present.

6.6 Prefix Origin Authentication in psBGP (G5)

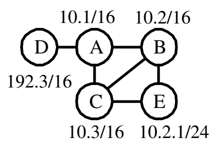
We start with descriptions of $PALs$ and MultiASCerts, and then introduce how to build from them an *AS prefix graph*. We then describe how psBGP uses an AS prefix graph to verify the propriety of prefix origin in the two cases per G5 in §6.4.

6.6.1 Prefix Assertion Lists ($PALs$)

Facing the difficulty of building a centralized infrastructure for tracing changes in IP address assignments (recall §6.1), psBGP uses a *decentralized* approach for verifying the propriety of a prefix assertion by cross-checking its consistency with endorsements from the neighbors of the asserting AS.

In psBGP, each AS s_i creates and signs an ordered *prefix assertion list* (pal_i), consisting of a number of tuples of the form (*prefixes, AS#*), i.e., $pal_i{=}\{(F_i, s_i), (F_1, s_1), .., (F_n, s_n)\}_{s_i}$, where for the components $(F_j, s_j), 1{\le}j{\ne}i{\le}n,\ s_j{\in}N(s_i)$ and $s_j{<}s_{j+1}$. The first tuple (F_i, s_i) is an assertion by s_i of its own assigned prefixes F_i (referred to as *prefix assertions*); the rest are ordered by AS number, and are assertions by s_i of prefixes assigned to each of s_i's neighbors (referred to as *prefix endorsements*). If s_i chooses not to endorse any prefix for a neighbor s_j or has no information of s_j's prefix assignments, s_i simply declares null in its prefix endorsement for s_j. Thus, $(F_j, s_j)_{s_i}\ (F_j{=}\phi)$ simply asserts that s_j is a direct neighbor of s_i (see Figure 6.7). If s_i is not willing to disclose that s_j is a direct neighbor, s_i can leave out from pal_i the prefix endorsement for s_j. Besides prefix assertion and endorsement, PALcan also be readily extended to encode AS business relationship and policy information; further discussion is beyond the scope of this thesis.

Definition 7 (Prefix-Consistency). *Let* $(f_i, s_i)_{s_i}$ *be a prefix assertion by* s_i *and* $(f'_i, s'_i)_{s_j}$ *a*



{(**10.1/16, A**), (10.2/16,B), (0,C), (192.3/16,D)}$_A$
{(**10.2/16, B**), (0,A), (10.3/16,C), (10.2.1/246,E)}$_B$
{(**10.3/16, C**), (10.1/16,A), (0,B), (10.2.1/24,E)}$_C$
{(**192.3/16, D**), (0,A)}$_D$
{(**10.2.1/24, E**), (0,B), (0,C)}$_E$

Figure 6.7: A small AS graph with IP prefixes and $PALs$ (0 denotes ϕ)

prefix endorsement by s_j. $(f_i', s_i')_{s_j}$ is consistent with $(f_i, s_i)_{s_i}$, denoted by $(f_i', s_i')_{s_j} \doteq (f_i, s_i)_{s_i}$, if they are regarding the prefix assignment of the same AS, i.e., $s_i' = s_i$, and f_i' is equal to or a superset of f_i, i.e., $f_i' \supseteq f_i$.

Inferred from Definition 7, $(f_i', s_i')_{s_j}$ is not consistent with $(f_i, s_i)_{s_i}$, if 1) they are regarding the prefix assignment of different ASes; 2) they have null mutual intersection, i.e., $f_i' \cap f_i = \phi$; or 3) f_i' is a proper subset of f_i, i.e., $f_i' \subset f_i$. In case 3, while f_i' and f_i do share a common subset which is f_i', they are not considered consistent in psBGP for the sake of simplicity of AS prefix graph maintenance. In psBGP, prefix consistency is checked between a prefix assertion and an endorsement, but not between two prefix endorsements.

While an AS is free to decide for which neighbors it provides prefix endorsements and from which to solicit prefix endorsements for itself, we recommend that a provider AS endorses prefixes for a customer AS, possibly becoming a part of an existing service agreement which includes not only physical network connectivity but now also prefix endorsements. Two neighboring ASes with a peer relationship have freedom to decide how one will endorse prefix assertions made by the other. Prefix endorsements between two peering ASes might be *asymmetric*; in the extreme case, AS s_i may endorse all prefixes assigned to a peering AS s_j, while s_j endorses no prefix assigned to s_i. It is important to allow such flexibility. In the core of the Internet, one AS may peer with many others, some of which may be assigned a large number of prefixes. It would be unrealistic to expect an AS to have full knowledge of all prefixes assigned to such a peer. However, an AS might be able to establish a certain level of confidence in a subset of the prefixes assigned to some of its neighbors. Thus, an AS can distribute such positive (albeit partial) evidence to facilitate other ASes to make a better decision in prefix origin authentication. It is an AS's own responsibility and in its own interest to ensure that its assigned prefixes are endorsed by some of its neighbors or by an RIR.

As a new requirement in psBGP, each AS is responsible for carrying out some level of due diligence off-line: for the safety of that AS and of the whole Internet, to increase its confidence that the prefixes it endorses for a direct neighbor are indeed assigned to that AS. We suggest the effort required for this is both justifiable and practical, since two neighboring ASes usually have a business relationship (e.g., a traffic agreement) with each other, allowing some level of off-line direct interactions and the establishment of some level of trust. For example, s_i may ask a neighboring AS s_j to show the proof that a prefix f_j is in fact assigned to s_j, or may ask a senior official of the neighboring AS organization to provide a formal letter asserting the organization's prefix claim. Publicly available information about IP address allocation and delegation may also be helpful.

A PAL may be distributed along with BGP update messages in newly defined path attributes [68], which are optional and transitive. A non-psBGP enabled BGP speaker which does not understand these newly defined attributes need not process them but must propagate them. Thus, $PALs$ travel through non-psBGP enabled BGP speakers to reach psBGP-enabled ones. Each psBGP-enabled BGP speaker can then construct and update its AS prefix graph from received $PALs$ (see §6.6.3).

6.6.2 Multiple-AS Certificate (MultiASCerts)

Ideally, two $PALs$ issued by two neighboring ASes are based on independent data sources, and consequently, with high probability (in the absence of collusion), a prefix erroneously asserted by one AS will not be endorsed by any of its neighbors. However, there are some organizations owning multiple ASes, and it is a common practice for a multi-AS organization to use a single centralized database for generating router configurations for all of its owned ASes. Thus, it is possible that $PALs$ issued by two neighboring ASes owned by a common organization would also be created from a single centralized database. If a prefix is erroneously entered into such a database, it might end up with a pair of erroneous yet consistent prefix assertion and endorsement, introducing a single point of failure. We recommend that "best practice" in psBGP requires that an AS obtain prefix endorsement from another AS owned by a different organization. As a recommended BGP local policy, an AS should ignore a prefix endorsement by s_j for s_i if both s_i and s_j are known to be owned by a common organization.

To facilitate the distribution of the knowledge of AS ownership by a multi-AS organization, psBGP makes use of a new certificate, namely MultiASCert (recall Figure 6.6), which binds a list of ASes owned by a common organization to the name of that organization, and is signed by an RIR. Prefix endorsements by s_j for s_i should be ignored if s_i and s_j appear on a MultiASCert. In this way, human errors by a multi-AS organization regarding a prefix that is assigned to another psBGP-enabled AS and endorsed by an independent neighboring AS will not result in service disruption of that prefix in psBGP (see §6.6.4).

6.6.3 AS Prefix Graph

We introduce as a new concept the *AS prefix graph*, which contains information about *AS connectivity*, *AS prefix assignments* (or prefix-AS bindings), and *ratings* of AS prefix assignments. An AS prefix graph, constructed by each AS s_c, is an attributed graph $G_c{=}(V, E, H)$, where $V{=}\{s_i\}$ is a set of AS numbers, $E{=}\{e_{ij}\}$ is a set of edges (BGP sessions) with e_{ij} connecting s_i to s_j, and $H\colon V{\rightarrow}W$ is a function mapping each AS s_i to a set of three-dimensional variables, which specifies the IP prefixes asserted by s_i, and supporting evidence; we call $H(s_i)$ the *APAS set* (associated prefixes and support) for s_i. More precisely, $H(s_i){=}\{(f_x, b_x, C_x)\}$, where $f_x{\subseteq}\mathbb{P}$ is an IP prefix, $b_x{\in}[0, 1]$ represents s_c's confidence that f_x is assigned to s_i, and C_x is a list of ASes asserting and endorsing the prefix assignment (f_x, s_i). We next present how each psBGP-enabled AS constructs and updates its own AS prefix graph based on the $PALs$ and MultiASCerts it has received.

AS Prefix Graph Construction

An AS prefix graph is initialized to null before the BGP speaker receives any PAL (e.g., when it first connects to the Internet). All BGP speakers within an AS build their own AS prefix graph independently. An AS s_c builds its AS prefix graph $G_c{=}(V, E, H)$ from the first pal_i received from each s_i on the Internet by performing the following tasks: a) adding

s_i and all of its declared neighbors to V; b) adding to E an edge from s_i to each of its declared neighbors; c) updating $H(s_i)$ for each of the prefixes asserted by s_i; d) updating $H(s_j)$ for each of the prefixes asserted by $s_j \in N(s_i)$ and endorsed by s_i. See Algorithm 8 for the details and §6.6.3 for an example.

Algorithm 8 AS prefix graph construction (for AS s_c)

```
1:  GLOBAL: $G_c=(V,E,H)$; existing $PALs$; $\{r_c(s_i)|s_i$ is an AS on the Internet$\}$
2:  INPUT: $pal_i$
3:  OUTPUT: updated AS prefix graph $G_c$
4:  /* $F_i, N(s_i)$ are prefixes and neighbors asserted by $s_i$ for itself in $pal_i$ respectively */
5:  $V \leftarrow V \cup s_i; H(s_i) \leftarrow \phi$
6:  for each $f_x \in F_i$ do
7:      $(f_x, b_x, C_x) \leftarrow (f_x, r(s_i), \{s_i\})$
8:      for each $s_j \in N(s_i)$ do
9:          $V \leftarrow V \cup s_j$; $E \leftarrow E \cup e_{ij}$
10:         for each prefix endorsement $(f,s)_{s_j}$ in $pal_j$ do
11:             /* recall Definition 7 */
12:             if $(f,s)_{s_j} \doteq (f_x, s_i)_{s_i}$ and $s_i, s_j$ are not in a common MultiASCert then
13:                 $b_x \leftarrow$ Algorithm5$\big(b_x, r(s_j)\big)$; $C_x \leftarrow C_x \cup s_j$
14:     $H(s_i) \leftarrow H(s_i) \cup (f_x, b_x, C_x)$;
15: for each $s_j \in N(s_i)$ do
16:     retrieve APAS set $H(s_j) = \{(f_y, b_y, C_y)\}$
17:     for each $(f_y, b_y, C_y) \in H(s_j)$ do
18:         for each prefix endorsement $(f,s)_{s_i}$ in $pal_i$ do
19:             if $(f,s)_{s_i} \doteq (f_y, s_j)_{s_j}$ and $s_i, s_j$ are not in a common MultiASCert then
20:                 $b_y \leftarrow$ Algorithm5$\big(b_y, r(s_i)\big)$; $C_y \leftarrow C_y \cup s_i$
21:         $H(s_j) \leftarrow H(s_j) \cup (f_y, b_y, C_y)$
22: return
```

AS Prefix Graph Update

Here we describe how to update an AS prefix graph from a newly received pal'_i which replaces an existing pal_i that has been previously used to construct or update an AS prefix graph (see §7.2.3 for certificate update frequency). The prefix-AS bindings in pal_i and pal'_i can be divided into three categories: *removed*, *unchanged*, and *added*. A removed prefix-AS binding appears in pal_i but not in pal'_i; an unchanged one appears in both; and a newly added one appears in pal'_i but not in pal_i. Updating an AS prefix graph is performed in two phases (see Algorithm 9 for details) as follows:

1. *Removing prefix-AS bindings*. If a removed prefix-AS binding is an assertion, $(f_x, s_i)_{s_i}$, made by s_i for itself, it is simply removed from the graph. If it is an endorsement, $(f_y, s_j)_{s_i}$, by s_i for $s_j \in N(s_i)$, the confidence in s_j's assertion of f_y must be updated (using Algorithm 6).

2. *Adding prefix-AS bindings*. If an added prefix-AS binding is an assertion, $(f_x, s_i)_{s_i}$, made by s_i for itself, a confidence value must be computed for $(f_x, s_i)_{s_i}$ (using Algorithm 5). If it is a prefix endorsement, $(f_y, s_j)_{s_i}$, and $(f_y, s_j)_{s_j}$ exists in the graph, the confidence in $(f_y, s_j)_{s_j}$ must be updated (using Algorithm 5).

Example 1

Figure 6.8 illustrates Algorithm 8 for an AS D. Assume D fully trusts its service provider A (i.e., $r(A)=1$), and partially trusts the other ASes ($r(B)=r(E)=0.5, r(C)=0.8$). The AS prefix graph is constructed based on the following $PALs$ received by D in order (here we focus on the construction of the APAS set):

$$pal_D = \{(192.3/16, D), (\phi, A)\}_D,$$
$$pal_A = \{(10.1/16, A), (10.2/16, B), (\phi, C), (192.3/16, D)\}_A,$$
$$pal_B = \{(10.2/16, B), (\phi, A), (10.3/16, C), (10.2.1/24, E)\}_B,$$
$$pal_C = \{(10.3/16, C), (10.1/16, A), (\phi, B), (10.2.1/24, E)\}_C,$$
$$pal_E = \{(10.2.1/24, E), (\phi, B), (\phi, C)\}_E.$$

1) D starts from pal_D issued by itself, and updates the graph as: $V=\{D, A\}$; $E=\{e_{DA}\}$; and $H(D)=\{(192.3/16, 1.0, \{D\})\}$. After receiving pal_A, D initializes $H(A)$ to $\{(10.1/16, 1.0, \{A\})\}$ (Algorithm 8 (line 7)). Since A endorses D's prefix assertion, $H(D)$ is updated to $\{(192.3/16, 1.0, \{D, A\})\}$. While A also endorses B's prefix assertion, no action is taken at this time since D has not received pal_B.

2) After receiving pal_B, D initializes $H(B)=\{(10.2/16, 0.5, \{B\})\}$. Since A endorses $(10.2/16, B)$, Algorithm5($0.5, 1.0$) is called to update D's confidence in $(10.2/16, B)$, and $H(B)$ is updated to $\{(10.2/16, 1.0, \{B, A\})\}$.

Algorithm 9 AS prefix graph update (for AS s_c)

```
1:  GLOBAL: $G_c$=$(V, E, H)$; existing $PALs$; $\{r_c(s_i)|s_i$ is an AS on the Internet$\}$
2:  INPUT: $pal'_i$
3:  OUTPUT: updated AS prefix graph $G_c$
4:  /* $N(s_i)'$ is the set of neighbors asserted by $s_i$ for itself in $pal'_i$ */
5:  /* Removing prefix-AS bindings */
6:  for each prefix assertion $(f_x, s_i)_{s_i}$ in $pal_i$ that is not in $pal'_i$ do
7:      retrieve the APAS set $H(s_i) = \{(f_x, b_x, C_x)\}$
8:      $H(s_i) \leftarrow H(s_i) - (f_x, b_x, C_x)$ /* set subtraction */
9:  for each prefix endorsement $(f_y, s_j)_{s_i}$ in $pal_i$ that is not in $pal'_i$ do
10:     retrieve the APAS set $H(s_j) = \{(f_y, b_y, C_y)\}$
11:     if $H(s_j) \neq \phi$ and $s_i \in C_y$ then
12:         $b_y \leftarrow$ Algorithm6$\big(b_y, r(s_i)\big)$; $C_y \leftarrow C_y - s_i$
13: for each $s_j$ in $N(s_i)$ that is not in $N(s_i)'$ do
14:     $E \leftarrow E - e_{ij}$
15:     if $s_j$ has no neighbor or prefix assignment in $G_c$ then
16:         $V \leftarrow V - s_j$
17: /* Adding prefix-AS bindings */
18: for each $s_j$ in $N(s_i)'$ that is not in $N(s_i)$ do
19:     $V \leftarrow V \cup s_j$; $E \leftarrow E \cup e_{ij}$
20: for each prefix assertion $(f_x, s_i)_{s_i}$ in $pal'_i$ that is not in $pal_i$ do
21:     $(f_x, b_x, C_x) \leftarrow (f_x, r(s_i), \{s_i\})$
22:     for each $s_j \in N(s_i)'$ do
23:         for each prefix endorsement $(f, s)_{s_j}$ in $pal_j$ do
24:             if $(f, s)_{s_j} \doteq (f_x, s_i)_{s_i}$ and $s_i, s_j$ are not in a common MultiASCert then
25:                 $b_x \leftarrow$ Algorithm5$\big(b_x, r(s_j)\big)$; $C_x \leftarrow C_x \cup s_j$
26:     $H(s_i) \leftarrow H(s_i) \cup (f_x, b_x, C_x)$
27: for each $s_j \in N(s_i)'$ do
28:     for each prefix endorsement $(f, s_j)_{s_i} \in pal'_i$ that is not in $pal_i$ do
29:         retrieve APAS set $H(s_j) = \{(f_y, b_y, C_y)\}$
30:         for each $(f_y, b_y, C_y) \in H(s_j)$ do
31:             if $(f, s_j)_{s_i} \doteq (f_y, s_j)_{s_j}$ and $s_i, s_j$ are not in a common MultiASCert then
32:                 $b_y \leftarrow$ Algorithm5$\big(b_y, r(s_i)\big)$; $C_y \leftarrow C_y \cup s_i$
33: return
```

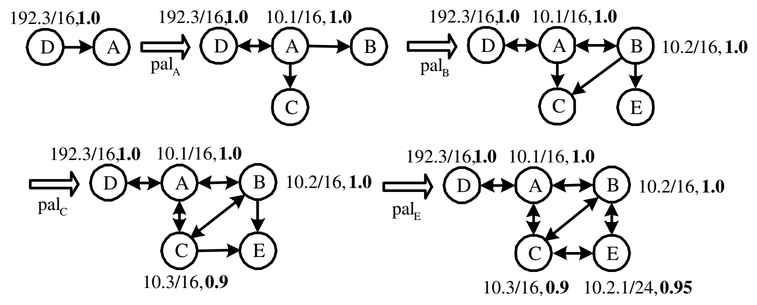


Figure 6.8: Construction of an AS prefix graph by AS D (see example 1)

3) After receiving pal_C, D initializes $H(C)$=$\{(10.3/16, 0.8, \{C\})\}$. Since B endorses $(10.3/16, C)$, Algorithm5$(0.8, 0.5)$ is called to update D's confidence in $(10.3/16, C)$ to 0.9, and $H(C)$ is updated to $\{(10.3/16, 0.9, \{C, B\})\}$. Since C endorses A's prefix assertion, Algorithm5$(1.0, 0.8)$ is called to update D's confidence in $(10.1/16, A)$, which does not change since it already has maximal value 1.0 (see above). $H(A)$ is updated to $\{(10.1/16, 1.0, \{A, C\})\}$.

4) After receiving pal_E, D initializes $H(E)$=$\{(10.2.1/24, 0.5, \{E\})\}$. Since B endorses $(10.2.1/24, E)$, Algorithm5$(0.5, 0.5)$ is called to update D's confidence in $(10.2.1/24, E)$ to 0.75. Since C also endorses $(10.2.1/24, E)$, Algorithm5$(0.75, 0.8)$ is called to further update D's confidence in $(10.2.1/24, E)$ to 0.95. As a result, $H(E)$ is updated to $\{(10.2.1/24, 0.95, \{E, B, C\})\}$.

6.6.4 Prefix Origin Authentication

Here we describe how to perform prefix origin authentication using an AS prefix graph.

Verification of Prefix Assignment

Two configurable thresholds, denoted by α_i (*sufficient confidence*) and β_i (*sufficient claimants*), are used by each psBGP-enabled AS s_i for verifying the propriety of prefix assignments. α_i is a threshold defining a sufficient confidence level by s_i in a prefix-AS binding before it can be considered proper. β_i defines a sufficient number of ASes which assert and endorse a prefix-AS binding before the binding can be considered proper by s_i. In other words, a prefix-AS binding (f_j, s_j) is verified as proper by s_i if s_i's confidence in (f_j, s_j) is at least α_i, or (f_j, s_j) is asserted by s_j and endorsed by at least $\beta_i{-}1$ other ASes. More specifically, a non-aggregated route $(f, [s_j, ..])$ originated by a psBGP-enabled AS s_j is verified by another psBGP-enabled AS s_i as *proper* if a) there exists $(f_x, b_x, C_x){\in}H(s_j)$; b) $b_x{\geq}\alpha_i$ or $|C_x|{\geq}\beta_i$; and c) $f{\subseteq}f_x$. Algorithm 10 specifies this explicitly.

Algorithm 10 Verification of prefix assignment (by AS s_i)

```
1: **GLOBAL:** $G_i = (V, E, H); \alpha_i; \beta_i$
2: **INPUT:** The BGP route $m = (f_j, p = [s_j, ..])$
3: **OUTPUT:** ACCEPT or REJECT $s_j$'s origin of $f_j$
4: retrieve the APAS set $H(s_j) = \{(f_x, b_x, C_x)\}$ from $G_i$
5: **for** each $(f_x, b_x, C_x) \in H(s_j)$ **do**
6:     **if** ($b_x{\geq}\alpha_i$ or $|C_x|{\geq}\beta_i$) and $f_j{\subseteq}f_x$ **then**
7:         return(ACCEPT)
8: return(REJECT)
```

α_i and β_i are independent and in conjunction provide extensive flexibility. $\alpha_i{=}1$ allows s_i to immediately accept a prefix assertion by a fully trusted AS (i.e., without any neighbor endorsement), while prefix assertions made by partially trusted ASes require endorsements from a sufficient number of neighbors. α_i and β_i can also be configured such that only one or neither takes effect. For example, $\alpha_i{>}1$ and $\beta_i{\geq}1$ allows β_i to always take precedence since the maximum confidence in a prefix assertion is 1. $0{<}\alpha_i{\leq}1$ and $\beta_i{=}\infty$ has the opposite effect. $\alpha_i{=}0$ and $\beta_i{=}0$ emulate the existing non-secured BGP behavior (i.e., any prefix originated by any AS is considered as proper).

During the early stages of psBGP deployment, when only a small number of ASes have deployed psBGP, we recommend $\beta_i{=}1$ for each psBGP-enabled AS s_i. In other words, a psBGP-enabled AS s_i allows another psBGP-enabled AS s_j to originate a prefix f_j if f_j is asserted in pal_j even it is not endorsed by any neighbor. This reflects the reality that early psBGP adopters might not have any psBGP-enabled neighbors, and it offers some level of assurance (albeit limited). For example, a compromised BGP speaker within a psBGP-enabled AS s_j cannot be used to hijack prefixes assigned to other ASes unless keying material required for issuing pal_j is also compromised. In addition, the existence of a public statement about an assertion provides some assurance, in that this might carry some weight in legal dispute or affect business reputation. See §7.2.1 for more discussion on incremental benefits and §7.1.2 on limitations of psBGP.

After a majority of ASes have deployed psBGP, we recommend β_i=2, i.e., a psBGP-enabled AS s_i allows another psBGP-enabled AS s_j to originate a prefix f_j only if f_j is asserted in pal_j and is endorsed by one of s_j's neighbors. β_i= 2 is resilient to some errors resulting from a single AS. For example, if s_j mistakenly asserts a prefix f in pal_j and announces f via BGP, this would not result in service disruption of the legitimate owner of f as long as s_j's assertion of f is not endorsed by any neighbor. However, β_i=2 remains vulnerable to two-party collusion. More generally, $\beta_i = k \geq 2$ resists collusion by $k-1$ parties. Larger β_i renders a stronger assurance in the propriety of a prefix assignment, but trades off performance and results in higher maintenance overhead (see §7.2.3).

Verification of Prefix Aggregation

Suppose AS s_1 is assigned a set of prefixes F_1. When receiving a set of routes with a set of prefixes F_2, the BGP specification [109] allows s_1 to aggregate F_2 into a single prefix f_g to reduce routing information to be stored and transmitted. We call f_g an *aggregated prefix*. s_1 can aggregate F_2 into f_g if one of the following conditions holds: 1) $\forall f_i \subseteq f_g, f_i \subseteq F_1$; or 2) $\forall f_i \subseteq f_g, f_i \subseteq F_1 \cup F_2$.

In case 1), s_1 must be assigned a set of prefixes F_1, which is a superset of the aggregated prefix f_g. Most likely, f_g is one of the prefixes assigned to s_1, i.e., $f_g \in F_1$. This type of aggregation is sometimes referred to as prefix *re-origination*. From a routing perspective, prefix re-origination does not have any effect since traffic destined to a more specific prefix will be forwarded to the re-originating AS and then forwarded to the ultimate destination from there. From a policy enforcement perspective, prefix re-origination does have an effect since the AS_PATH of an aggregated route is different from any of the AS_PATHs of the routes to be aggregated. Since AS_PATH is used by the route selection process, changing AS_PATH has an impact on route selections. From a security perspective, prefix re-origination is no different than normal prefix origination since the aggregated prefix is either the same as, or a subset of, the prefix assigned by the aggregating AS. Therefore, f_g can be verified using the mechanism in §6.6.4.

In case 2), s_1 is not assigned the whole address space of the aggregated prefix f_g. Therefore, f_g cannot be verified in the same way as for prefix re-origination. To facilitate verification of the propriety of route aggregation by a receiving AS, psBGP imposes a new requirement: the routes to be aggregated must be supplied by the aggregating AS along with the aggregated route. This approach is essentially similar to that taken by S-BGP. Transmission of routes to be aggregated incurs additional network overhead, which is something BGP tries to reduce. However, we view such additional overhead to be relatively insignificant given that modern communication networks generally have high bandwidth and BGP control messages account for only a small fraction of subscriber traffic. The main purpose of route aggregation is to reduce the size of routing tables, i.e., reducing storage requirements; note that this is preserved by psBGP.

6.6.5 Route Selection Algorithm

In standard BGP, when a BGP speaker receives two valid routes with the same destination prefix, a route selection process is invoked to determine which is preferable. In what follows, a prefix-AS binding of a route means the binding of the prefix and the AS that originates that route. psBGP adds two new rules: one gives preference to a route whose prefix-AS binding has more neighbor endorsements, and the other to a route whose prefix-AS binding is rated higher. These two new rules are added into the fourth and fifth places in BGP route selection algorithm [109] to preserve existing traffic engineering practices which usually employ $local\_pref$, as_path and med ($mult\_exit\_disc$). Note that the higher-numbered rule is followed if the lower-numbered rules result in a tie.

1) Select the route with a higher degree of preference, i.e., a higher $local\_pref$ value.

2) Select the route with a shorter as_path.

3) Select the route with a lower med value if they have the same $next\_hop$.

4) *Select the route whose prefix-AS binding is endorsed by more neighbors.*

5) *Select the route whose prefix-AS binding is rated higher.*

6) Select the route with a lower intra-domain routing cost to the $next\_hop$.

Ongoing work [111] suggests to allow customer-defined rules to be inserted anywhere in the standard BGP route selection algorithm. If this is implemented in psBGP, customers with high security requirement can choose to move psBGP-related rules up to an appropriate decision point, e.g., as rules 1 and 2.

6.7 Discussion

The timeliness of PAL updates is important to ensure service availability. $PALs$ need to be updated and distributed in a timely manner so that prefix ownerships can be verified using currently correct information. To ensure that an endorsing neighbor of a given AS updates its PALs for that AS in a timely manner, a service agreement between them would likely be required, e.g., as an extension to their existing agreements. Since there is usually some time delay window before newly delegated prefixes are actually used on the Internet, an endorsing AS should be required to update its PAL to include newly delegated prefixes of an endorsed neighbor within that delay window. Updates of prefix removals can be done with lower priority since they would appear to have only relatively small security implications. PALs along with other certificates (e.g., ASNumCerts, SpeakerCerts, and corresponding Certificate Revocation Lists) can be distributed with BGP update messages in newly defined path attributes [68]; thus, they can be distributed as fast as announcements of prefixes and are accessible without any dependence on BGP routes. Those certificates might also be stored in centralized directories [68]. However, a "pull" model might make it challenging to decide how often centralized directories should be checked.

Network stability is another important issue which requires further study. The modified route selection process in psBGP takes into account new information such as AS ratings and the number of ASes endorsing a prefix assertion. If not properly protected, these new rules may lead to route flipping and network instability. To mitigate potential impact on network stability, we place the new rules at a low decision point. As a future work, we will perform quantitative study on the impact of network stability and convergence by psBGP.

Chapter 7

Analysis of psBGP

In this chapter, we present security, and operational analysis of psBGP. In §7.1, we analyze how psBGP meets specified BGP security goals, and counters selected BGP threats. In §7.2, we analyze constraints and incremental benefits of psBGP deployment, as well as computational complexities of core psBGP algorithms. A brief overview of S-BGP and soBGP is given in §7.3. We compare S-BGP, soBGP and psBGP in §7.4.

7.1 Security Analysis of psBGP

We first analyze psBGP against the listed security goals from §6.4, followed by discussion on how psBGP counters selected BGP threats.

7.1.1 Meeting Specified Security Goals

The analysis below clarifies how the proposed psBGP mechanisms meet the specified goals, and by what line of reasoning and assumptions. While we believe that mathematical "proofs" of security may often be based on flawed assumptions or models (e.g., see [73]) that fail to guarantee "security" in any real-world sense, they are nevertheless very useful, e.g., for finding security flaws, for precisely capturing protocol goals, and for reducing ambiguity, all of which increase confidence. We thus provide outlines of such formalized reasoning, as a complement to alternative methods of increasing confidence.

Proposition 6. *psBGP provides AS number authentication (G1).*

Proof Outline: For an AS number s to be certified, psBGP requires an ASNumCert $(k_s, s)_{\overline{k_T}}$. Since T (i.e., an RIR) controls s, and is the trusted guardian of AS numbers (by assumption), any assertion made by T about s is proper. Thus $(k_s, s)_{\overline{k_T}}$ is proper. In other words, s is an AS number certified by T, and k_s is a public key associated with s certified by T. More formally,[5] (T *controls* s) $\wedge$ $(k_s, s)_{\overline{k_T}} \Rightarrow (k_s, s)$ is a proper binding.

Proposition 7. *psBGP provides BGP speaker authentication (G2).*

Proof Outline: For a BGP speaker g to be accepted as an authorized representative of an AS s, psBGP requires an ASNumCert $(k_s, s)_{\overline{k_T}}$, a SpeakerCert $(k'_s, s)_{\overline{k_s}}$, and evidence that g possesses $\overline{k'_s}$. By Proposition 6, $(k_s, s)_{\overline{k_T}}$ establishes that s is an AS number certified by T and k_s is a public key associated with s certified by T. Similarly, $(k'_s, s)_{\overline{k_s}}$ establishes that k'_s is a public key associated with s certified by s. Evidence that g possesses $\overline{k'_s}$ (i.e., an appropriate digital signature generated by g using $\overline{k'_s}$) establishes that g is authorized by s to represent s. Thus, the Proposition is established. More formally, (T *controls* s) $\wedge$ $(k_s, s)_{\overline{k_T}} \Rightarrow (k_s, s)$ is a proper binding; (k_s, s) is proper $\wedge$ $(k'_s, s)_{k_s} \Rightarrow (k'_s, s)$ is proper binding; (k'_s, s) is proper $\wedge$ g possesses $\overline{k'_s} \Rightarrow g$ is authorized by s.

Proposition 8. *psBGP provides data integrity (G3).*

Proof Outline: psBGP uses the IPsec Encapsulating Security Payload (ESP) [70] with null encryption for protecting BGP sessions, and relies upon IPsec ESP for data integrity. Thus this provides data integrity in practice, to the extent that one can rely on practical implementations of IPsec ESP.

Proposition 9. *psBGP provides assurance of AS_PATH authentication (G4).*

Proof Outline: Let $m_k{=}(f_1, p_k)$ be a BGP route, where $p_k{=}[s_1, .., s_k]$, and m_k is originated or forwarded by a BGP speaker in s_k. For simplicity, we refer to an AS instead of a BGP speaker within that AS. In psBGP, the integrity of p_k implies that m_k has traversed the exact sequence of $s_1, .., s_k$. We next use induction on path length to show that psBGP

[5] Here we adapt BAN-like notation, modified for our purpose (cf. [23, 40, 43]).

provides AS_PATH integrity when all ASes on an AS_PATH are psBGP-enabled and the verifying AS chooses to verify all digital signatures on the path, followed by discussion of other cases.

1. If $k{=}1$, psBGP requires that for s_2 to accept m_1, s_2 must receive a valid digital signature $sig_1 = \{f_1, [s_1, s_2]\}_{s_1}$, which serves as a signed assertion that s_1 originated m_1 (and advertised it to s_2).

2. Assume when $k{=}n{\geq}2$, there exist digital signatures $sig_1, .., sig_n$ which assert that m_n indeed traversed the exact sequence of $s_1, .., s_n$. When $k{=}n{+}1$, we need to show that m_{n+1} has traversed from s_n to s_{n+1} and exited s_{n+1}. $sig_n{=}\{f_1, [s_1, .., s_n, s_{n+1}]\}_{s_n}$ asserts that s_n forwards m_n to s_{n+1}. psBGP requires that s_{n+1} digitally signs m_{n+1} by generating a digital signature $sig_{n+1} = \{f_1, [s_1, ..., s_{n+1}, s_{n+2}]\}_{s_{n+1}}$, which serves as the evidence that m_{n+1} is advertised by s_{n+1} to another AS s_{n+2}. In summary, sig_n asserts that m_n traversed from s_n to s_{n+1}, and sig_{n+1} asserts that m_n is transformed by s_{n+1} to m_{n+1} which traversed through s_{n+1} to another AS. Thus, the above three steps establish Proposition 9 when all ASes on an AS_PATH are psBGP-enabled and the verifying AS verified all digital signatures on the path.

Partial AS_PATH integrity. If an AS chooses not to always verify all digital signatures on the path (i.e., setting confidence threshold $\theta{<}1$, or some digital signatures are missing; see Algorithm 7 and §6.5.5), full integrity of the path is not guaranteed. For example, let $p_k{=}[s_1, .., s_j, .., s_k]$. If an AS only verifies the digital signatures generated by ASes from s_j to s_k, only the integrity of that the path segment is protected. The path from s_1 to s_{j-1} can be falsified if all ASes from s_j to s_k are in collusion. As another example, consider the route $m{=}(f, [s_1, s_2, s_3, s_4])$ with only s_2 being psBGP-enabled. The digital signature generated by a well-behaved s_2, $\{f, [s_1, s_2, s_3]\}_{s_2}$, covers the path $[s_1, s_2, s_3]$. In other words, a malicious AS cannot compromise the integrity of $[s_1, s_2, s_3]$, but it can insert any non-psBGP enabled AS after s_3 or modify s_4 to another non-psBGP enabled AS. In addition, $[s_1, s_2, s_3]$ can be removed or replaced as a whole with other non-psBGP enabled ASes.

We next establish Proposition 10. As discussed in §6.5.1, psBGP uses a rating mechanism to provide the flexibility to allow an AS to fully trust an AS or an RIR, thus accepting their prefix assertions without requiring additional endorsements. We recommend that no AS should be fully trusted unless there is strong reason to do so. In the rest of our analysis, we assume that a verifying AS s_i does not immediately trust any other AS s_j. In other words, s_i rates every other AS s_j with a value lower than its confidence threshold, i.e., $r_i(s_j){<}\alpha_i$. Before presenting Proposition 10, we establish two Lemmas.

Lemma 1. *Assuming that no two ASes are in collusion (A1),*[6] *then psBGP with threshold* $\beta{=}2$ *provides reasonable*[7] *assurance of prefix assignment verification, i.e., a prefix assignment that is verified as proper is, with reasonable assurance, proper.*

Proof Outline: Consider the BGP route $m{=}(f_x, [s_i, ..])$. For f_x to be verified as assigned to s_i, psBGP requires that for some f_i:

(R1) prefix assertion $(f_i, s_i)_{s_i}$ exists; **(R2)** $(f'_i, s_i)_{s_j}{\doteq}(f_i, s_i)_{s_i}$ exists for $s_j{\in}N(s_i)$;

(R3) s_i, s_j do not appear in a common MultiASCert; and **(R4)** $f_x{\subseteq}f_i$.

R1, R2, and R3 establish that f_i is assigned to s_i, and R4 shows that f_x is a subset of f_i. Suppose f_i is not assigned to s_i but is verified as such (i.e., R1-R4 are met). For this statement to be true, the following statements must be true: $(f_i, s_i)_{s_i}$ is improper; and $(f_i, s_i)_{s_j}$ is improper. Since $(f_i, s_i)_{s_i}$ and $(f_i, s_i)_{s_j}$ are improper and consistent, s_i and s_j either share a common false data source (H1) or they are considered in collusion (H2). R3 reduces the likehood of H1, and H2 is ruled out by assumption A1. Thus, the statement that f_i is not assigned to s_i but is verified as such is, with reasonable assurance, not true. In other words, if f_i is not assigned to s_i, it will, with reasonable assurance, not be verified as such. Equivalently, if f_i is verified as assigned to s_i, it is, with reasonable assurance, assigned to s_i. This establishes Lemma 1.

Lemma 2. *psBGP provides reasonable assurance of IP prefix aggregation verification.*

[6]See §7.1.2 for discussion of examples where this collusion assumption (A1) may not hold.

[7]By reasonable, we mean to emphasize that our claim is relative to our threat model and assumptions (e.g., see §7.1.2); we cannot claim absolute security (which we do not believe exists in the real world).

Proof Outline: Let f_g be a prefix which is aggregated by AS s_x from a set of routes $\{m_i{=}(f_i, p_i)|p_i{=}[s_i, \dots]\}$ received by s_x. psBGP requires that for f_g originated by s_x to be verified as proper, s_x must either own a prefix f_x such that $f_g \subseteq f_x$ (verified by Lemma 1), or provide evidence that s_x has in fact received $\{m_i\}$ and $f_g \subseteq \cup\{f_i\}$. Valid digital signatures from each AS on p_i can serve as evidence that s_x has received $\{m_i\}$ (see Proposition 9). If $f_g \subseteq \cup\{f_i\}$, then s_x aggregates f_g properly. If s_x cannot provide the required evidence, s_x's aggregation of f_g is verified as improper. This establishes Lemma 2.

Proposition 10. *psBGP provides reasonable assurance of IP prefix origination authentication (G5), i.e., an AS s_i's origination of a prefix f is, with reasonable assurance, verified as proper if f is assigned to s_i or is aggregated properly by s_i from a set of routes received by s_i.*

Proof Outline: Lemma 1 allows prefix assignment verification, and Lemma 2 allows prefix aggregation verification, thus establishing Proposition 10.

The above results (Propositions 6–10) establish the psBGP security properties, as summarized by Theorem 3 (cf. §6.4).

Theorem 3 ((psBGP Security Properties)). *psBGP achieves the following five security goals: AS number authentication (G1), BGP speaker authentication (G2), data integrity (G3), AS_PATH authentication (G4), and prefix origin authentication (G5).*

7.1.2 Countering Selected BGP Threats

We first consider how psBGP detects false prefix originations, and next discuss how psBGP reacts to possible new threats arising from proposed security mechanisms in psBGP itself. We also discuss some attack scenarios which are not addressed by psBGP.

Detecting False Prefix Origin

We consider three cases in which an AS may originate routes for a prefix which is actually assigned to another AS.

MALICIOUS ATTACK. A malicious AS may hijack a prefix from another AS to attract its traffic. An AS is considered malicious if one or more BGP speakers within that AS are compromised, or the administrator in the AS that controls BGP software and configuration intentionally misbehaves. psBGP can detect prefix hijacking since a malicious AS will be unable to obtain from its neighbors or a trusted authority (e.g., an RIR) endorsements for the hijacked prefix.

ROUTER MALFUNCTION. A router may mistakenly deaggregate prefixes (e.g., due to software problems) and announce more specific ones. Deaggregating another AS's prefix is referred to as *foreign deaggregation*; deaggregating one's own prefix is referred to as *self deaggregation*. Foreign deaggregation has the same external behavior as prefix hijacking, and thus can be detected. Self deaggregation appears to be equivalent to the announcement of a subset of the prefix assigned to an AS, and thus is treated as legitimate.

DATABASE MISCONFIGURATION. Many ISPs use automatic scripts to generate router configurations from a centralized database containing information of prefix assignments. If a prefix is erroneously entered into such database (e.g., due to human error), automatically generated configurations will instruct a router which might be functioning correctly to originate a prefix which it is not authorized to announce.

Database misconfiguration will not result in successful prefix hijacking if the erroneous database is not used by *any* neighboring AS to generate its PAL. In other words, if the information used by all endorsing ASes for generating $PALs$ is independent of the misconfigured database containing erroneous prefixes, origin of those prefixes will result in verification failures since there will not exist a prefix endorsement consistent with the false prefix assertion. However, an ISP may have multiple ASes and use a single centralized database for generating both router configurations and $PALs$ for its own ASes. Thus, it is possible that an erroneous prefix assertion made by one AS gets endorsement from another AS owned by the same ISP. This scenario is addressed in psBGP with MultiASCerts (Section 6.6.2). More specifically, an endorsement from s_i for a prefix assertion made by s_j is not used if both s_i and s_j are owned by the same organization, in which case they should both appear on a MultiASCert under a common organization.

Countering False $PALs$

We now discuss how psBGP reacts to erroneous $PALs$ that contain false assertions or endorsements. These might potentially introduce new vulnerabilities arising from the proposed enhancements, as a result of malice or human error.

ERRONEOUS PREFIX ASSERTIONS. An AS s_i erroneously asserting the ownership of a prefix through its own PAL will not result in service disruption of the legitimate owner of that prefix as long as none of s_i's neighbors endorses its assertion.

ERRONEOUS PREFIX ENDORSEMENTS. An AS s_i erroneously endorsing s_j for a prefix which is not asserted by s_j will not result in any service disruption since such an endorsement will not be used by any for verifying s_j's prefix assertions. If s_i is the only endorsing neighbor for s_j, or more generally, $\forall s_i \in N(s_j)$, s_i issues $(f_j', s_j)_{s_i}$ inconsistent with $(f_j, s_j)_{s_j}$, then $(f_j, s_j)_{s_j}$ will be verified as *improper* by other ASes, even if it is actually correct. This is the case when misbehaving ASes form a network cut from s_j to any part of the network. It appears difficult, if not impossible, to counter such an attack; however, we note that even if such a denial of service attack could be prevented, many other techniques beyond the control of BGP could also be used to deny the routing service of s_j, e.g., link-cuts [14], filtering, or packet dropping. Note that a prefix assertion made by s_i about a remote AS s_k, i.e., $s_i \notin N(s_k)$, will not be checked when s_k's prefix assertions are verified because s_i is not a neighbor of s_k. Thus, a misbehaving AS is unable to mislead other ASes about the prefix ownership of a non-neighboring AS.

Limitations of psBGP

We now discuss some limitations of psBGP. First, it is subject to human error if a psBGP-enabled AS s_i sets threshold β_i=1 (e.g., during the early stage of psBGP deployment on the Internet). For example, if an AS uses a common database for generating BGP speaker configuration and for issuing $PALs$, a prefix erroneously entered into such a database can result in service disruption. Second, psBGP is subject to k-party collusion if β_i=k≥2. Suppose β_i=2 which is the recommended configuration (see §6.6.4) for each psBGP-enabled AS s_i. If an attacker controls two ASes that are owned by two different organizations (i.e., they do not appear on a common MultiASCert), it is possible for the attacker to generate two erroneous yet consistent $PALs$. This is equivalent to the case that the $PALs$ issued by two different ASes are in fact based on a single data source; thus corroborating these two dependent $PALs$ does not yield additional benefit. As a result, psBGP security can be defeated. To successfully launch such an attack, an adversary needs to: a) set up two organizations and manage to obtain an AS number from an RIR for each of them; b) compromise the private keys used by two independent ASes for signing their $PALs$; or c) set up one organization and manage to obtain an AS number from an RIR and compromise the private key used by another AS for signing its PAL. We suggest that these attacks would present non-trivial (albeit not insurmountable) practical difficulty to an adversary.

7.2 Operational Analysis of psBGP

Here we analyze some operational and performance issues of psBGP.

7.2.1 Deployment Analysis of psBGP

We first argue that the effort involved in deploying psBGP is reasonable (relative to alternatives), and next discuss incremental benefits from psBGP deployment.

Reasonable Deployment Effort

To deploy psBGP, an AS needs to: upgrade its BGP speakers to support psBGP; issue a single public key certificate for its own BGP speakers (SpeakerCert); distribute the corresponding private key securely to its speakers; and issue an appropriate prefix assertion list (PAL). Upgrading BGP speakers can be done in a similar manner as upgrading existing router software. Issuing a SpeakerCert (e.g., in X.509v3 format) requires some level of knowledge of public key certificates. However, many people responsible for BGP operations might have already acquired similar knowledge, e.g., from the use of PGP [154]; in any case, we acknowledge that additional effort will always be involved in setting up a new system. For example, personnel familiar with PGP may still need to spend some time studying the X.509v3 certificate format. Issuing a PAL requires carrying out a certain level of due diligence in improving an AS' confidence in the prefixes assigned to a (typically) small number of selected neighbors. We expect such effort is reasonable since two direct neighbors usually have established service agreements allowing some level of direct interaction. Such effort is also justifiable (in our opinion) considering potential security benefit to the Internet as a whole. Overall, all of this work can be done independently by an AS without requiring authorization from other ASes (e.g., an upstream ISP). In other words, psBGP can be deployed from the bottom up, mirroring the growth model of the Internet.

Incremental Deployability

As with the deployment of almost any other large scale security system, it is unrealistic to expect psBGP to be deployed by all ASes simultaneously, or to be deployed at different times but turned on at the same time. It is expected that if adopted, a small number of ASes will deploy psBGP first, then more and more ASes will follow. It is desirable that those ASes deploying psBGP first can achieve some immediate benefits to justify their investment before psBGP is widely deployed. Here we analyze benefits and constraints of psBGP deployment (β=1).

The first AS adopting psBGP does not gain any immediate benefit since none of the other ASes speaks psBGP. The second AS adopting psBGP will have some benefit collectively with the first psBGP-enabled AS if they are direct neighbors. In this case, one psBGP-enabled AS (s_i) will likely prefer the route originated by the other (s_j) over routes originated by a non-psBGP enabled AS regarding a prefix assigned to s_j (see §6.6.5). Since s_i and s_j are also directly connected, traffic originated from s_i and destined to s_j will likely arrive at s_j and not be attracted to another AS if everything else besides BGP also works correctly. In the case that s_i and s_j are not directly connected, i.e., connected by one or more non-psBGP enabled ASes, s_i will still likely prefer the route originated by s_j over an erroneous one by a non-psBGP enabled AS (see §6.6.5), resulting in containment of any erroneous announcements. However, there is no assurance that traffic destined to s_j can reach their ultimate destinations from s_i. This is because such traffic must traverse through non-psBGP enabled ASes (or unsecured zones), some of which could have poisoned routing tables and direct traffic over incorrect paths. Thus, security that can be achieved by two remote psBGP enabled ASes is less than that achieved by two psBGP-enabled neighbors.

We say that one or more psBGP-enabled ASes with direct links among themselves form a *secure zone*, and one or more non-psBGP enabled ASes with direct links among themselves form a *nonsecure zone*. Assume that at one point, there are a number of ASes on the Internet which have deployed psBGP. Then the Internet can be viewed to consist of a number of secure and nonsecure zones. Since two directly connected secure or non-secure zones can always form a larger secure or non-secure zone, a secure zone will always directly connect with nonsecure zones, and a non-secure zone can have only secure zones as its direct zone neighbors. This implies that secure zones can always form a network cut for a nonsecure one. To this end, we can draw two conclusions:

1) An AS improperly originating a route for a prefix assigned to a psBGP-enabled AS will be contained once it reaches a secure zone. In other words, if a misbehaving AS is within a secure zone, the erroneous route will be contained immediately. If it is within a nonsecure zone, it will propagate within the nonsecure zone and be contained once it reaches a secure zone.

2) An improper origination of a prefix assigned to a non-psBGP enabled AS will be propagated (without detection by psBGP) through all non-secure and secure zones, and could reach the entire Internet.

It is clear from the above conclusions that prefixes assigned to a psBGP-enabled AS are protected to a certain degree from being hijacked while there is no such protection for non-psBGP enabled ASes. While a psBGP-enabled AS might find limited protection when the number of other psBGP-enabled ASes is small, the protection increases as this number grows. As a starting point, it might be beneficial for an organization which owns multiple ASes (such as a large or even medium-sized government) to deploy psBGP so that a secure zone can be formed within that organization.

7.2.2 Complexity Analysis of psBGP

Here we consider the computational complexity resulting from AS_PATH verification and AS prefix graph related operations. The former involves computationally expensive operations such as digital signature generation and verification, while the latter involves much simpler (less costly but potential numerous) operations such as data structure insertion, deletion, comparison, and query. We do not attempt to provide a detailed, mathematically rigorous running-time analysis for psBGP operations, but rather to provide enough insight to allow ball-bark estimates sufficient to provide confidence that computational costs of psBGP are reasonable, and will not be a reason to avoid deploying psBGP.

Complexity of AS_PATH Verification

Let a be the average number of external ASes with which a BGP speaker establishes BGP sessions, and b the average number of ASes on an AS_PATH. A psBGP-enabled BGP speaker needs to generate on average a unique digital signatures (one per AS neighbor) for each BGP update message it sends to a neighbors, and to verify on average b unique digital signatures (for maximal security, i.e., θ=1) for each BGP update message received (see Algorithm 7). Signature verifications related to certificate revocation and certificate chains are ignored here.

Complexity of AS Prefix Graph Operations

Let n be the total number of ASes on the Internet, d the average number of AS neighbors, and h the average number of prefixes assigned to an AS. Let $x \leq d$ be the average number of neighboring ASes whose prefix assertions are endorsed by an AS, and y the average number of prefixes endorsed by an AS for each such neighbor. Accordingly, each AS on average has x endorsing neighbors.

Thus, each PAL (cf. §6.6.1) on average consists of: 1) h prefix assertions, one for each assigned prefix; 2) y prefix endorsements for each endorsed neighbor (x of them), resulting in xy prefix endorsements in total; 3) $d-x$ null prefix endorsements, one for each non-endorsed neighbor. Assume there are z MultiASCerts. We next estimate the computational costs of the construction, update, and query of an AS prefix graph in psBGP. Note all operations mentioned here are simple database operations (e.g., comparison), not computationally expensive operations such as digital signature generation or verification.

1) *Complexity of AS Prefix Graph Construction* (Algorithm 8). For the first pal_i received from each AS on the Internet, an AS needs to update the APAS $H(s_i)$ for s_i (lines 6–13), resulting in $h\{1+d[2+xy(1+z+1+1)]\}$ operations. In addition, an AS also needs to update the APAS $H(s_j)$ for each of s_i's endorsed neighbors s_j (lines 14–20), resulting in $d\{1+h[xy(1+z+1+1)+1]\}$ operations. Thus, in total $2hdxyz+6hdxy+3hd+h+d$ operations are required for processing each pal_i, resulting in $n(2hdxyz+6hdxy+3hd+h+d)$ operations for constructing a complete AS prefix graph from $n\ PALs$.

2) *Complexity of AS Prefix Graph Update* (Algorithm 9). Consider the worst case that an AS s_i issues a new pal'_i that is completely different from the existing pal_i, i.e., all of its prefix assertions and endorsements have changed. In Algorithm 9, lines 6–7 result in h operations, lines 8–11 result in $5xy$ operations, lines 12–18 result in $5d$ operations, lines 19–25 result in $h\{1+d[xy(1+z+1+1)]+1\}$ operations, and lines 26–31 result in $d\{xy[1+h(1+z+1+1)]\}$ operations. Thus one update might require in total $2hdxyz+6hdxy+dxy+5xy+3h+5d$ operations.

3) *Complexity of AS Prefix Graph Query* (Algorithm 10) When an AS receives a BGP update message, it verifies that the origin AS is allowed to announce the prefix by comparing the announced prefix with the h prefixes asserted by the origin AS, resulting in up to h operations for one prefix origin verification.

7.2.3 Performance Analysis of psBGP

Here we present our preliminary estimation of memory, bandwidth, and CPU overhead, and the analysis of certificate dynamics in psBGP. While rigorous study has been performed by Aiello et al. [3] on the prefix delegation stability on the Internet as a whole, and by Zhao et al. [97, 151] on PKI impact on BGP security using simulation, it is desirable to study certificate dynamics of a secure system and to project certificate management overhead on a per-AS level. We use BGP data collected by the RouteViews project [116]. We retrieved one BGP routing table the first day of each month from January to August 2004. Despite known shortcomings including incompleteness of the RouteViews data set, it is one of the most complete data repositories publicly available, and has been widely used in the BGP community.

Memory Overhead

Four types of certificates and one AS prefix graph require memory for a BGP speaker to support psBGP. We estimate the memory overhead for each type and then give an estimate of the total. While a BGP update message may carry extra digitally signed data and signatures which need to be stored temporarily, they can be discarded after verification. Thus, we omit their memory overhead here.

ASNUMCERTS AND SPEAKERCERTS. We observed in total 17884[8] ASes as of August 1, 2004. One ASNumCert is required per AS. In the worst case, an AS may need to store the ASNumCert of every AS on the Internet; in this case, $17\,844$ ASNumCerts would be stored. As with S-BGP and soBGP, psBGP recommends use of the X.509v3 certificate structure which has wide industrial support. Assuming the average size of a certificate is 600 bytes [68] based on 1024-bit RSA keys, 10.479M bytes of memory would be required for storing $17\,844$ ASNumCerts. The same holds for SpeakerCerts.

$PALs$ AND MULTIASCERTS. The size of pal_i, issued by each AS s_i, is primarily determined by the number of prefixes assigned to s_i, the number of s_i's neighbors, and the number of prefixes assigned to each of s_i's neighbors that are endorsed by s_i. While some ASes have many neighbors, and some are delegated many prefixes, many ASes have only a small number of neighbors and are delegated a small number of prefixes. Based on the RouteViews data we use, each AS on average has 4.2 neighbors and is delegated 9.1 prefixes. Assuming the average size of a PAL is 1024 bytes (600 bytes for an X.509v3 certificate plus 424 bytes for about 60 prefix assertions and endorsements), 17.844M bytes of memory would be required to store $17\,844$ PALs, one for each AS. For MultiASCerts, a

[8]AS numbers used by IANA itself for experimental purposes are not counted.

BGP speaker needs to store one certificate for each organization which owns multiple ASes. Based on the data from Aiello et al. [3], there are 385 multi-AS organizations which in total own 1259 ASes. On average, each multi-AS organization owns 3.3 ASes. Assuming the average size of a MultiASCert is 600 bytes, 0.226M bytes of memory are required by each AS for storing all MultiASCerts.

AS PREFIX GRAPH. Each AS needs to construct an AS prefix graph for prefix origin verification. The memory space required for storing an AS prefix graph depends on the data structures being used. For simplicity, we use a fixed array consisting of $17\,844$ entries, one entry per AS. Each entry consists of a 16-bit AS number and a 32-bit pointer, pointing to a linked list of APAS sets of this AS. On average, each such linked list has 10 elements with each of 17 bytes. Thus, each entry in the fixed array on average consumes 176 bytes. In total, an AS prefix graph requires 3.141M bytes memory (M=10^6), using these (non-optimized) data structures.

ASNumCerts	10.479M Bytes
SpeakerCerts	10.479M Bytes
PALs	17.844M Bytes
MultiASCerts	0.226M Bytes
AS Prefix Graph	3.141M Bytes
Total	42.169M Bytes

Table 7.1: psBGP memory requirements per BGP speaker

In summary, a total of 42.169M bytes of memory are required by a BGP speaker for storing all certificates and an AS prefix graph to support psBGP (see Table 7.1).

Bandwidth Overhead

Except for a small number of public key certificates of trusted CAs which may be distributed using out-of-band mechanisms, all other certificates in psBGP can be distributed with BGP update messages. The latter consumes extra network bandwidth. However, such overhead is not persistent since those certificates only need to be distributed periodically or upon changes. We expect that such overhead is of little significance and do not discuss it further.

The primary bandwidth overhead is introduced by digitally signed data and signatures carried by each BGP update message for protecting the message. For a fully protected BGP route where every AS on the route digitally signs the update message, the overhead is mainly determined by the number of such ASes, and could result in as much as 600% overhead according to Kent [68]. We expect no significant difference between the bandwidth overhead of psBGP and S-BGP. While increased bandwidth overhead due to psBGP (or e.g., S-BGP) is significant in terms of percentage, as pointed out by Kent [68], BGP control messages only account for a small fraction of network bandwidth versus subscriber traffic. Thus, from our preliminary analysis, we expect that bandwidth overhead of psBGP will not create difficulty in the deployment of psBGP.

CPU Overhead

We expect that CPU overhead of psBGP will mainly result from AS_PATH verification, not AS prefix graph operations (cf.§7.2.2). A psBGP-enabled BGP speaker needs to digitally sign each BGP update message sent to each neighbor, and to verify some digital signatures carried by each BGP update message it receives and chooses to use. As shown by Kent et al. [71] in their study of S-BGP performance, such CPU overhead is significant. Especially in the case of reboots, a BGP speaker will receive full routing tables from each of its neighbors, and thus must verify a large number of digital signatures if psBGP is implemented. Note that an AS prefix graph need not be rebuilt since it can be stored in persistent storage and reloaded upon reboot. psBGP provides the flexibility for reducing the CPU overhead resulting from digital signature verification by using a lower confidence threshold, which trades off security for efficiency. In other words, psBGP provides a mechanism which allows protection to be proportionally achieved in accordance to the CPU power which a router has available to spend on signature verification. However, to achieve higher level of assurance of AS_PATH integrity, significant CPU overhead will be generated by psBGP. To mitigate the problem, various approaches might be helpful, including caching [71], delay of signature verification [71], using a digital signature algorithm with a faster verification operation (e.g., RSA) [97], and aggregated path authentication [150].

Certificate Dynamics

ASNUMCERTS AND SPEAKERCERTS. The monthly number of ASes on the Internet has grown by an average of 190 since January 1, 2004, with an average of 347 ASes added and 157 ASes removed (see Table 6.5). When an AS number is added or removed in psBGP, the corresponding ASNumCert must be issued or revoked by an RIR. Thus, five RIRs between them must issue an average of 347 new ASNumCerts and revoke an average of 157 existing ASNumCerts per month. This would certainly appear to be manageable in light of substantially larger PKIs existing in practice (e.g., see [47]). Note the issuing and revocation of a SpeakerCert is performed by an AS, not an RIR.

PREFIX ASSERTION LISTS (PALS). A prefix assertion list pal_i must be changed (removed, added, or updated) if: a) the AS number s_i changes (i.e., is removed or added); b) an IP prefix assigned to s_i changes; c) s_i's neighbor relationship changes, i.e., a neighbor is removed or added; or d) an IP prefix changes which is endorsed by s_i for one of its neighbors. Table 7.2 depicts the dynamics of prefix assignments.

	Jan	Feb	Mar	Apr	May	Jun	Jul
Start of Month	148 903	148 014	151 174	156 019	157 925	160 818	155 118
Stable During Month	143 200	144 422	146 139	151 481	153 171	148 280	151 436
Stable During Jan-Jul	119 968	119 968	119 968	119 968	119 968	119 968	119 968
Removed During Month	5 703	3 592	5 035	4 538	4 754	12 538	3 682
Added During Month	4 814	6 752	9 880	6 444	7 647	6 838	10 360

Table 7.2: IP prefix dynamics from January 1 to August 1, 2004

We study the number of prefix assertion (PA) changes required for each AS based on the two routing tables of July 1 and August 1, 2004. Each prefix addition or removal is counted once (i.e., resulting in one PA addition or removal) if the AS number of the AS owning that prefix does not change. If an AS number is newly added (or removed) during the month, all additions (or removals) of the prefixes owned by that AS are counted once as a whole. One PA change usually represents one update to a PAL if such update is done in a timely manner. However, an AS can choose to do multiple PA changes in one PAL update.

Table 7.3 depicts the projected PA dynamics based on the data set of July 2004. The total number of ASes observed during July 2004 is $18\,048$, including $17\,884$ observed on

August 1, 2004 and 164 removed during July 2004. We can see, the more ASes endorsing an AS's prefix assertions, the more PA changes required. We recommend the scenario $n = 2$, where each AS has at most two endorsing neighbors even if it has more than two neighbors. This provides a level of redundancy in the case that one of the two endorsing neighbors fails to carry out adequate due diligence.

# of PA Changes		1	2-4	5-10	11-30	31-100	101-1000	over 1001	Total
n=1	# of ASes	1497	677	319	152	69	26	2	2742
	(percentage)	(8.3%)	(3.8%)	(1.8%)	(0.8%)	(0.3%)	(0.1%)	(0%)	(15.2%)
n=2	**# of ASes**	**1508**	**713**	**346**	**187**	**87**	**48**	**3**	**2892**
	(percentage)	**(8.4%)**	**(4.0%)**	**(1.9%)**	**(1.0%)**	**(0.5%)**	**(0.2%)**	**(0%)**	**(16.0%)**
n=3	# of ASes	1516	725	355	205	93	54	4	2952
	(percentage)	(8.4%)	(4.0%)	(2.0%)	(1.1%)	(0.5%)	(0.3%)	(0%)	(16.4%)
n=all	# of ASes	1424	784	387	233	112	53	30	3023
	(percentage)	(7.9%)	(4.3%)	(2.1%)	(1.3%)	(0.6%)	(0.3%)	(0.2%)	(16.7%)

Table 7.3: Projected number of ASes in absolute number, and as percentage of all ASes, requiring the specified number of monthly prefix assertion (PA) changes in psBGP based on July 2004 data. We recommend row $n = 2$ (n is the number of endorsing neighbors)

From Table 7.3, in the recommended scenario $n = 2$, 16% of the ASes need to update their $PALs$ during the month. 8.4% of ASes need only one PA change in the month, 4% need 2 to 4 PA changes, and 1.9% need 5 to 10 PA changes. However, a small number of ASes need more than 100 changes, and AS 701 (UUNET) and its two endorsing neighbors need around 5000 changes. In our study, if an AS chooses to endorse the prefixes of a neighboring AS, it simply endorses all the prefixes assigned to that neighbor. To reduce the number of PA changes, an AS can choose to only endorse a subset of the prefixes assigned to a neighbor. In this case, PA change overhead can be distributed to some other ASes and will be more balanced than what is shown in Table 7.3.

7.3 Overview of S-BGP and soBGP

Here we describe two leading BGP security proposals: S-BGP [72, 71] and soBGP [144], for comparing with psBGP in §7.4.

7.3.1 Secure BGP (S-BGP)

S-BGP makes use of two strict hierarchical PKIs and other mechanisms (e.g., IPsec [69]) for securing BGP. The proposed S-BGP PKIs are parallel to the existing systems for the allocation and delegation of AS numbers and IP address space. A single Certificate Authority (CA) rooted at IANA/ICANN was initially proposed for S-BGP, but it evolved to the multiple CAs rooted at five RIRs due to political sensibility and security considerations. There are a number of types of certificates in S-BGP. An organization X which obtains IP address space and AS numbers directly from an RIR, is issued the following certificates[9]:

- *Organization Public Key Certificates* – binding a public key K_x to X signed by an authority T, denoted by $(K_x, X)_T$;
- *Address Delegation Certificates* – binding an IP prefix f_x (or more) to X signed by an authority T, denoted by $(f_x, X)_T$;
- *AS Number Delegation Certificates* – binding an AS number (or more) s_x to X signed by an authority T, denoted by $(s_x, X)_T$.

To participate in the inter-domain routing, X issues the following certificates or attestations:

- *Router Public Key Certificate* – binding a public key K_{r_x} to a BGP speaker r_x and an AS number s_x signed by X using $\overline{K_x}$, denoted by $(K_{r_x}, \{s_x, r_x\})_{\overline{K_x}}$;
- *Address Attestation* – binding IP prefixes f_x or a subset of f_x to an AS number (s_x) signed by X, denoted by $(f_x, s_x)_{\overline{K_x}}$;
- *Route Attestation* – binding IP prefixes f_i to an AS_PATH p_j (along with other path attributes) signed by a BGP speaker r_x. For simplicity, we only consider AS_PATH here. A Route Attestation is denoted by $(f_i, p_j)_{r_x}$.

[9]For convenience of presentation, certificate names used here may differ from those used in the S-BGP literature.

With these certificates, a BGP speaker can announce and verify routes in S-BGP. Let r_x be a BGP speaker, representing AS s_x owned by organization X. Let f_x be an IP prefix allocated to X by an RIR, and assigned by X to AS s_x. We use Figure 7.1 to illustrate the route announcements and verifications in S-BGP. For simplicity, we assume that each AS has only one BGP speaker, and BGP speakers are not shown in the figure.

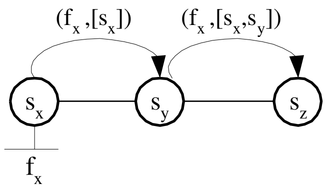


Figure 7.1: An example of S-BGP operation

Route Announcement

r_x originates the route $(f_x, [s_x])$, signs it by generating the route attestation $(f_x, [s_x, s_y])_{r_x}$, and forwards them to r_y representing AS s_y. r_y verifies the received route according to S-BGP specifications (see next paragraph). If the route verification succeeds, r_y forwards the transformed route to r_z representing AS s_z, along with the following signed information to facilitate route verification by r_z:

- $(f_x, [s_x, s_y])_{r_x}$ – the signed route received from r_x; and
- $(f_x, [s_x, s_y, s_z])_{r_y}$ – the route with updated AS_PATH and signed by r_y.

Route Verification

Upon receiving $(f_x, [s_x, s_y])$, r_z performs the following verifications:

- Is the first AS on the AS_PATH, which is s_x in this case, authorized to originate IP prefix f_x? Prefix origin verification succeeds if there exist the following valid certificates[10]: $(K_x, X)_T$, $(f_x, X)_T$, $(s_x, X)_T$, and $(f_x, s_x)_{K_x}$.

[10]For simplicity, here we do not consider IP prefix delegation among organizations. For example, X can delegate a prefix f_i which is a portion of its allocated prefix f_x to another organization Y by issuing a certification $(f_i, Y)_X$.

- Is each AS on the AS_PATH authorized by the previous AS to further propagate the route? In this example, the question becomes, is s_y authorized by s_x to further propagate the route? The AS_PATH $[s_x, s_y]$ verifies successfully if the route attestations $(f_x, [s_x, s_y])_{r_x}$ and $(f_x, [s_x, s_y, s_z])_{r_y}$ are also received by r_z along with the route. Of course, r_z must first verify that BGP speakers r_x and r_y are authorized to represent AS s_x and s_y respectively.

S-BGP is one of the earliest BGP security proposals. It provides strong guarantee of prefix origin authentication and AS_PATH integrity. Disadvantages of S-BGP include: 1) the proposed S-BGP PKIs are complex and face significant deployment challenges [6], and 2) AS_PATH verification is computational expensive, and requires S-BGP being deployed by contiguous ASes on the path.

7.3.2 Secure Origin BGP (soBGP)

soBGP [144] proposes use of a web-of-trust model for authenticating AS public keys and a hierarchical structure for verifying IP prefix ownership. Each AS has a public key certificate, binding an AS number with a public key, signed by a “trusted” public key. To bootstrap trust, a small number of “root public key certificates” are distributed using out-of-band mechanisms. Some tier-1 ISPs and well-known authentication service providers (e.g., Verisign) are suggested to be candidates of trusted public key certificate authorities. An AS with a trusted AS public key certificate (e.g., signed by a trusted CA) may further sign a public key certificate for another AS, thus naturally forming a web-of-trust model. While a web-of-trust model has strong proponents for authenticating user public keys within the technical PGP community [153], it would appear to be less suitable for authenticating public keys of ASes which are identified by AS numbers strictly controlled by RIRs; thus it is questionable if any entity other than RIRs should be trusted for signing AS public key certificates.

With respect to IP prefix ownership verification, soBGP makes use of a strictly hierarchical structure similar to that of S-BGP. Prefix delegation structures might be simplified in soBGP by using ASes instead of organizations, however, it is not clear if it is practical to do so since IP addresses are usually delegated to organizations not to ASes [3]. We suggest that soBGP, like S-BGP, also faces difficulty in tracing changes of IP address ownership in a strict hierarchical way. Thus, both S-BGP and soBGP have made architectural design choices which arguably lead to practical difficulties.

7.4 S-BGP, soBGP, and psBGP Comparison

We compare the different approaches taken by S-BGP, soBGP, and psBGP for achieving the BGP security goals listed in §6.4. Table 7.4 provides a summary. We see that psBGP falls somewhere between S-BGP and soBGP in several of the security approaches and architectural design decisions, but makes distinct design choices in several others.

Goal	S-BGP	soBGP	psBGP
G1: AS Number Authentication	centralized (multiple levels)	decentralized (with trust transitivity)	centralized (depth=1)
G2: BGP Speaker Authentication	one certificate per BGP speaker	one certificate per AS	one certificate per AS
G3: Data Integrity	IPsec or TCP MD5	IPsec or TCP MD5	IPsec or TCP MD5
G4: Prefix Origin Authentication	centralized (multiple levels)	centralized (multiple levels)	decentralized (no trust transitivity)
G5: AS_PATH Verification	full integrity	plausibility	stepwise integrity

Table 7.4: S-BGP, soBGP, and psBGP comparison re: achieving security goals of §6.4

7.4.1 AS Number Authentication

Both S-BGP and psBGP use a centralized trust model for authenticating AS numbers, which is different from the web-of-trust model used by soBGP. The difference between the AS number authentication of psBGP and S-BGP is that S-BGP follows the existing structure of AS number assignment more strictly than psBGP. In S-BGP, an AS number is assigned by an RIR to an organization and it is an organization that creates and signs a certificate binding an AS number to a public key (thus, a two-step chain). In psBGP, an ASNumCert is signed directly by an RIR (depth=1), and is independent of the name of an organization. Thus, psBGP has less certificate management overhead than S-BGP, requiring fewer certificates. In addition, some changes in an organization X may not require revoking and reissuing the public key certificate of the AS controlled by X. For example, if X changes its name to Y but the AS number s associated with X does not change, psBGP does not need to revoke the ASNumCert $(k_s, s)_T$. However, in S-BGP, the public key certificates $(K_x, X)_T$ issued to X must be revoked, which in turn result in the revocation of all certificates signed using $\overline{K_x}$, such as router public key certificates and address attestations.

7.4.2 BGP Speaker Authentication

In S-BGP, a public key certificate is issued to each BGP speaker, while both soBGP and psBGP use one common public key certificate for all speakers within one AS. Thus, soBGP and psBGP require fewer BGP speaker certificates (albeit requiring secure distribution of a common private key to all speakers in an AS).

7.4.3 Data Integrity

S-BGP uses IPsec for protecting BGP session and data integrity. Both soBGP and psBGP adopt this approach. TCP MD5 [50] is supported by all three proposals for backward compatibility. In addition, automatic key management mechanisms can be implemented for improving the security of TCP MD5.

7.4.4 Prefix Origin Authentication

Both S-BGP and soBGP propose a hierarchical structure for authorization of the IP address space; however S-BGP traces how IP addresses are delegated among organizations, while soBGP only verifies IP address delegation among ASes. It appears that soBGP simplifies the delegation structure and requires fewer certificates for verification; however, it is not clear if it is feasible to do so in practice since IP addresses are usually delegated between organizations, not ASes. In psBGP, consistency checks of PALs of direct neighbors are performed to verify if it is proper for an AS to originate an IP prefix. Therefore, psBGP does not involve verification of chains of certificates (instead relying on offline due diligence). We note that while psBGP does not guarantee perfect security of the authorization of IP address allocation or delegation, as intended by S-BGP and soBGP, as discussed in §6.1 it is not clear if the design intent in the latter two can actually be met in practice.

7.4.5 AS_PATH Verification

Both S-BGP and psBGP verify the integrity of AS_PATH based on its definition in the BGP specification [109]. psBGP differs S-BGP in that it uses a rating mechanism and a step-wise approach for allowing a partially signed AS_PATH. Thus, psBGP trades off the strong guarantee of AS_PATH integrity provided by S-BGP with incremental deployability. In contrast, soBGP verifies the plausibility of an AS_PATH, which trades off AS_PATH integrity with computational efficiency and possibly network convergence speed.

7.5 Discussion

Different approaches have been taken by S-BGP and soBGP for addressing security in BGP. We believe that psBGP adopts their best features, while differing fundamentally with a novel approach taken to verify IP prefix assignments and AS_PATH integrity. As no centralized infrastructure for tracing changes in IP prefix assignments currently exists, and it would appear to be quite difficult to build such an infrastructure, we believe that the decentralized approach taken by psBGP provides a more feasible means of increasing confidence in correct prefix origin. We hope that our comparison of S-BGP, soBGP and psBGP will help focus discussion of securing BGP on the technical merits of the various proposals, and will serve to stimulate discussion in the Internet community about alternate design choices and trust models for securing BGP.

Chapter 8

Concluding Remarks

We start with a summary of this thesis, followed by an overview of our future work. We conclude this thesis with discussions on tradeoff of security and practicality, applicability of information corroboration to other routing systems, and future direction of BGP security.

8.1 Summary of this Thesis

This thesis examined security problems in the control plane of Internet routing infrastructure, and proposed practical mechanisms for improving the security of Internet routing protocols including RIP and BGP.

- Chapter 2 provided background information on routing protocols, public key cryptography, fault-tolerant systems, and Dempster-Shafer theory.

- Chapter 3 reviewed related work on routing security including security mechanisms for both control plane and data plane of a routing infrastructure. In addition, Chapter 3 discussed two systems namely, PGP and SPF. PGP makes use of multiple digital signatures for improving trust in the authenticity of a public key. SPF utilizes a separate communication channel, i.e., DNS, for establishing confidence in the correctness of the sender address in an email message. Both systems make use of information corroboration for improving security, which is the approach advocated by this thesis for securing routing protocols.

- Chapter 4 presented a framework for securing routing protocols. It includes a threat model, a summary of routing protocol security goals, a process of establishing trust using information corroboration, and a rating mechanism for representing and combining belief in routing information.

- Chapter 5 proposed S-RIP for countering selected RIP threats. Our security analysis shows that S-RIP can successfully detect fraudulent routing updates with high probability when no two nodes are in collusion. Our simulation results demonstrate that security and routing overhead in S-RIP can be well balanced using the S-RIP rating mechanism and two confidence thresholds.

- Chapter 6 started with a general discussion on BGP security threats, and followed by an illustration of realistic risk from BGP vulnerabilities using Google's May 2005 outage as a real world example. After summarizing five BGP security goals, we then presented psBGP for meeting these security goals using information corroboration along with a PKI of simple structure and manageable size. One novelty of psBGP is that it can establish trust in the propriety of a prefix origin by corroborating prefix assertions made by the neighbors of the originating AS of a prefix in question. Thus, psBGP does not require the unrealistic assumption of a central trusted authority which has perfect knowledge of which IP prefixes are assigned to a particular organization or an AS.

- Chapter 7 presented our analysis of psBGP regarding its security, computational complexity, deployability, and performance. Our analysis shows that psBGP can successfully defeat uncoordinated attacks, it is incrementally deployable, and its resource consumption (e.g., CPU and memory requirements) can be realistically met. We also compared psBGP with S-BGP and soBGP on their approaches of achieving five BGP security goals. psBGP differs fundamentally from S-BGP and soBGP with its novel approach for prefix origin authentication, and a step-wise approach for AS_PATH verification, both of which have practical advantages.

8.2 Future Work

Here we outline our future plan for further advancing S-RIP and psBGP.

8.2.1 Future Work on S-RIP

In this thesis, we conducted simulation on two aspects of S-RIP: 1) potential risk of accepting routing advertisements without performing consistency checks; and 2) S-RIP network overhead. Our simulation results showed that both the risk of accepting malicious routing advertisements and S-RIP routing overhead are relatively low in a random network configured with proper S-RIP thresholds, for instance, using one of the three partially secured configurations as described in §5.7.1. As future work, we plan to perform additional simulations on S-RIP with the following objectives:

- To determine the effect of two time-out values (P_1, P_2) for a low rating and a high rating respectively. In the simulation conducted in this thesis, we use two arbitrary values for P_1 and P_1 (i.e., P_1=P_2=2 seconds). Changing these two values will have impact on both the risk of accepting malicious routing updates and S-RIP routing overhead. Thus, it is of practical interest to determine proper values for P_1 and P_2.

- To study how S-RIP performs in non-random networks. In this thesis, we demonstrated that S-RIP performs consistently in several random networks of different size. These random networks have certain representations, but they are still limited. It is of general interest to simulate S-RIP with additional non-random network topologies.

- To study how S-RIP performs in a network with attackers whose misbehaviors are not random. In this thesis, we simulated random adversaries (i.e., randomly chose some nodes to become malicious) with random attacks (i.e., randomly chose some routes and changed their distances to some random values). In reality, an attacker is usually equipped with more information (e.g., a network topology) and can launch more sophisticated attacks. It is interesting to further examine how S-RIP reacts to smart attacks.

8.2.2 Future Work on psBGP

In this thesis, we presented a high level design of psBGP. We performed security analysis and a certain operational analysis of psBGP. One future work is to study the impact of psBGP on the stability and convergence of the Internet. Another future work on psBGP will focus on a prototype implementation, which requires detailed design of psBGP. As a first step, we plan to define the data structures of various certificates used by psBGP, and associated certificate management functions. The next step is to define the formats of new BGP path attributes, which will be used for distributing psBGP certificates, followed by the definitions of data structures and operations associated with AS prefix graph (see §6.6.3). With these detailed design in place, we can start a partial implementation of psBGP which allows for prefix origin authentication. Design and implementation of AS_PATH authentication will then follow.

Beyond AS_PATH integrity verification as described in §6.5.5, it is desirable to verify if an AS_PATH conforms to the route exporting policies of each AS on the path. Since BGP is a policy-driven routing protocol, each AS can individually decide whether or not a received route advertisement should be further propagated to a neighboring AS. Such route exporting policies are mainly defined based on the business relationship between two ASes. Without route exporting policy verification, a misbehaving BGP speaker (e.g., misconfigured) may be able to re-advertise routes which are prohibited by its route exporting policies. For example, a multi-homed AS may re-advertise routes received from one provider AS to the other, thus functioning as a transit AS for its two providers. Such misbehavior may allow for eavesdropping and may also result in service disruption. We are currently exploring mechanisms for AS_PATH route exporting policy verification, which we expect to present in future work.

We are also interested in contributing to IETF initiatives of improving inter-domain routing security. Currently, the IETF Routing Protocol Security (RPsec) [117] working group is undertaking the effort of defining security requirements for BGP. At the same time, a new IETF working group, namely Secure Inter-domain Routing (SIDR) [122], is being charted with the mandate of developing security mechanisms for BGP. psBGP was presented to the IETF in the SIDR Bird of Feather (BOF) session during the 64^{th} IETF meeting in November 2005. We plan to participate in the undergoing design work of both RPsec and SIDR working groups.

8.3 Discussion

A main contribution of this thesis is in its first systematic use of information corroboration for improving confidence in the correctness of routing updates. While this approach does not offer perfect security, it has practical advantages and we believe it is deployable in the real world. Security must be balanced against deployability which poses constraints on the security design for any existing systems such as Internet routing protocols. A proposal offering perfect security but which is not deployable is of limited value. On the other hand, a deployable proposal offering improved albeit imperfect security has practical advantages. We believe that both S-RIP and psBGP have reasonable balance between security and practicality, and thus they can contribute to the security improvement of Internet routing infrastructure.

In this thesis, we applied the technique of information corroboration specifically to securing RIP and BGP. However, we believe our method is applicable in one way or another to other routing protocols as well. While details of how information corroboration can be applied will vary from one routing protocol to another, the central idea remains same, i.e., the truthfulness of routing updates should be verified and information corroboration can play an important role in such verification. As an example, we showed in [137] that information corroboration can be applied to DSDV [102], which is a routing protocol proposed for emerging MANETs.

One aspect of the complexity of securing BGP arises from its large scale deployment on the Internet. While it might be easier to design a new secure inter-domain routing protocol from scratch, it appears impossible to completely replace BGP with a new protocol. Thus, a practical security proposal for BGP must be based on BGP, and provide backward compatibility with existing BGP. On the other hand, BGP has evolved from a simple path-vector routing protocol into a complex one which is used for distributing and enforcing ISP policies by taking into account economic and political factors [24]. Thus, it is apparent that adding security mechanisms into BGP will introduce additional complexity, which might in turn result in instability of Internet routing infrastructure. One promising direction is to return BGP its original simplicity by decomposing its complexities with a new protocol that allows for expressing and enforcing ISP policies and routing security requirements [58]. This new protocol uses BGP as a transport mechanism for distributing policy and security related information, and provides BGP with necessary information for validating routing information. Another direction is to develop anomaly detection technique [67] to mitigate potential risk from BGP security vulnerabilities during the interim period before a BGP security mechanism is standardized and deployed on the Internet in a large scale.

Acronyms

AfriNIC African Network Information Centre

APAS Associated Prefixes and Support

APNIC Asia Pacific Network Information Centre

ARIN American Registry for Internet Numbers

ARQ Automatic Repeat reQuest

AS Autonomous System

ASNumCert AS Number Certificate

BGP the Border Gateway Protocol

CA certification authority

CRC Cyclic Redundancy Check

CRL Certificate Revocation List

DN Distinguished Name

DNS Domain Name System

DSR Dynamic Source Routing

DST Dempster-Shafer Theory

DV Distance Vector

ESP Encapsulating Security Payload

IAB Internet Architecture Board

IANA Internet Assigned Number Authority

IRV Inter-domain Routing Validator

IS-IS Intermediate Systems to Intermediate Systems

ISP Internet Service Provider

LACNIC Latin American Caribbean Internet Addresses Registry

LS Link State

LSA Link State Advertisement

MAC Message Authentication Code

MANET Mobile Ad-hoc Network

MITM Man-in-the-Middle

MOAS Multiple Origin ASes

NAP Network Access Point

NLRI network layer reachability information

OCSP Online Certificate Status Protocol

OSPF Open Shortest Path First

PA Prefix Assertion

PAL Prefix Assertion List

PGP Pretty Good Privacy

PKI public key infrastructure

psBGP Pretty Secure BGP

RIP Routing Information Protocol

RIPE Réseaux IP Européens

RIR Regional Internet Registry

RPsec Routing Protocol Security

S-BGP Secure BGP

SMTP Simple Mail Transfer Protocol

SNMP Simple Network Management Protocol

SIDR Secure Inter-Domain Routing

soBGP Secure Origin BGP

SpeakerCert	BGP Speaker Certificate
SPF	Sender Policy Framework
SPV	Secure Path Vector protocol
TMR	Triple Modular Redundancy
TTL	Time to Live

Bibliography

[1] C. Adams and S. Lloyd. *Understanding Public-Key Infrastructure,* 2^{nd} *edition*. Addison-Wesley Professional, 2003.

[2] AfriNIC. African Network Information Centre, 2005. http://www.afrinic.net.

[3] W. Aiello, J. Ioannidis, and P. McDaniel. Origin Authentication in Interdomain Routing. In *Proceedings of* 10^{th} *ACM Conference on Computer and Communications Security*, pages 165–178, Washington, D.C., USA, October 2003.

[4] APNIC. Asia Pacific Network Information Centre, 2005. http://www.apnic.net.

[5] ARIN. American Registry of Internet Numbers, 2005. http://www.arin.net.

[6] R. Atkinson and S. Floyd. IAB Concerns and Recommendations Regarding Internet Research and Evolution. IETF RFC 3869, August 2004.

[7] B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens. An On-Demand Secure Routing Protocol Resilient to Byzantine Failures. In *Proceedings of ACM Workshop on Wireless Security (WiSe)*, September 2002.

[8] F. Baker and R. Atkinson. RIP-II MD5 Authentication. IETF RFC 2082 (Proposed Standard), January 1997.

[9] A. Barbir, S. Murphy, and Y. Yang. Generic Threats to Routing Protocols. IETF Internet Draft, October 2004.

[10] R.E. Bellman. On a Routing Problem. *Quarterly of Applied Mathematics*, 16:87–90, 1958.

[11] S. Bellovin. Security Problems in the TCP/IP Protocol Suite. *Computer Communications Review*, 19(2):32–48, April 1989.

[12] S. Bellovin. Routing Security. British Columbia Institute of Technology, June 2003.

[13] S. Bellovin. Where the Wild Things Are: BGP Threats. NANOG 28, June 2003.

[14] S. Bellovin and E. Gansner. Using Link Cuts to Attack Internet Routing. Unpublished manuscript, 2003.

[15] S.M. Bellovin. A Look Back at "Security Problems in the TCP/IP Protocol Suite". In *Proceedings of the* 20^{th} *Annual Computer Security Applications Conference (ACSAC'04)*, Tucson, Arizona, USA, December 2004.

[16] S.M. Bellovin. Spamming, Phishing, Authentication, and Privacy. *Communications of the ACM (Inside Risks)*, 47(12), December 2004.

[17] S.M. Bellovin, J. Ioannidis, and R. Bush. Position Paper: Operational Requirements for Secured BGP. In *DHS Secure Routing Workshop*, March 2005.

[18] D. Bertsekas and R. Callager. *Data Networks*. Prentice Hall, 1992.

[19] L. Blunk. New BGP Analysis Tools and a Look at the AS9121 Incident, March 2005. http://www.merit.edu/nrd/papers-presentations/MTP-2005-01.pdf.

[20] D. Boneh, X. Boyen, and H. Shacham. Short Group Signatures. In *Proceedings of Crypto 2004*, volume 3152, pages 41–55, August 2004.

[21] K.A. Bradley, S. Cheung, N. Puketza, B. Mukherjee, and R.A. Olsson. Detecting Disruptive Routers: A Distributed Network Monitoring Approach. *IEEE Network*, 12(5):50–60, September/October 1998.

[22] S. Buchegger and J.Y. Le Boudec. Performance Analysis of the CONFIDANT Protocol (Cooperation Of Nodes - Fairness In Dynamic Ad-hoc NeTworks). In *Proceedings of the* 3^{rd} *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2002)*, June 2002.

[23] M. Burrows, M. Abadi, and R. Needham. A Logic of Authentication. In *Research Report 39*. Digital Systems Research Center, Feburary 1989.

[24] M. Caesar and J. Rexford. BGP Policies in ISP Networks. *IEEE Network Magazine*, November/December 2005.

[25] J. Case, M. Fedor, M. Schoffstall, and J. Davin. A Simple Network Management Protocol (SNMP). IETF RFC 1157, May 1990.

[26] CCITT. ITU-T Recommendation X.509 (1997 E): Information Technology - Open Systems Interconnection - the Directory Authentication Framework, June 1997.

[27] CERT. CERT Advisory CA-2002-03 Multiple Vulnerabilities in Many Implementations of the Simple Network Management Protocol (SNMP) , 2002. http://www.cert.org/advisories/CA-2002-03.html.

[28] T.M. Chen and V. Venkataramanan. Dempster-Shafer Theory for Intrusion Detection in Ad Hoc Networks. *IEEE Internet Computing*, 9(6):35–41, Nov/Dec 2005.

[29] S. Cheung. An Efficient Message Authentication Scheme for Link State Routing. In *Proceedings of the* 13^{th} *Annual Computer Security Applications Conference*, San Diego, CA, USA, December 1997.

[30] S. Cheung and K. Levitt. Protecting Routing Infrastructure from Denial of Dervice Using Cooperative Intrusion Detection. In *Proceedings of the New Security Paradigms Workshop*, Cumbria, UK, September 1997.

[31] Microsoft Corp. Caller ID for E-Mail Technical Specification. http://www.microsoft.com/mscorp/twc/privacy/spam_callerid.mspx, 2005.

[32] S. Deering, S. Hares, C. Perkins, and R. Perlman. Overview of the 1998 IAB Routing Workshop. August 2000. IETF RFC 2902.

[33] A.P. Dempster. Upper and Lower Probabilities Induced by a Multivalued Mapping. *The Annals of Mathematical Statistics*, 38:325–339, 1967.

[34] W. Diffie and M.E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, November 1976.

[35] E.W. Dijkstra. Self-Stabilizing Systems in Spite of Distributed Control. *Communications of the ACM*, 17(11):643–644, 1974.

[36] K. Fall and K. Varadhan. The *ns* Manual (formerly *ns* Notes and Documentation). April 2002.

[37] P. Ferguson. Google DNS Problems, May 2005. http://www.merit.edu/mail.archives/nanog/2005-05/msg00238.html.

[38] L.R. Ford and D.R. Fulkerson. *Flows in Networks*. Princeton University Press, Princeton, NJ, USA, 1962.

[39] S. Forrest, A. Somayaji, and D.H. Ackley. Building Diverse Computer Systems. In *Proceedings of the 6th Workshop on Hot Topics in Operating Systems*, pages 67–72. IEEE Computer Society Press, Los Alamitos, CA, 1997.

[40] K. Gaarder and E. Snekkenes. Applying a Formal Analysis Technique to the CCIT X.509 Strong Two-Way Authentication Protocol. *Journal of Cryptology*, 3:81–98, 1991.

[41] L. Gao. Inferring Autonomous System Relationships in the Internet. In *Proceedings of IEEE Global Internet*, November 2000.

[42] V. Gill, J. Heasley, and D. Meyer. The Generalized TTL Security Mechanism (GTSM). IETF RFC 3682, February 2004.

[43] V.D. Gligor, R. Kailar, S. Stubblebine, and L. Gong. Logics for Cryptographic Protocols - Virtues and Limitations. In *Proceedings of the Computer Security Foundations Workshop IV*, pages 219–226, Los Alamitos, California, USA, 1991.

[44] G. Goodell, W. Aiello, T. Griffin, J. Ioannidis, P. McDaniel, and A. Rubin. Working Around BGP: An Incremental Approach to Improving Security and Accuracy of Interdomain Routing. In *Proceedings of the 2003 ISOC Symposium on Network and*

Distributed Systems Security (NDSS'03), pages 75–85, San Diego, California, USA, February 2003.

[45] M.T. Goodrich. Efficient and Secure Network Routing Algorithms. *Provisional Patent Filing*, January 2001.

[46] M.T. Goodrich. Leap-Frog Packet Linking and Diverse Key Distributions for Improved Integrity in Network Broadcasts. In *Proceedings of IEEE Symposium on Security and Privacy*, pages 196–207, Oakland, CA, USA, May 2005.

[47] R. Guida, R. Stahl, T. Bunt, G. Secrest, and J. Moorcones. Deploying and Using Public Key Technology: Lessons Learned in Real Life. *IEEE Security and Privacy*, pages 67–71, July/August 2004.

[48] J. Hawkinson and T. Bates. Guidelines for Creation, Selection, and Registration of an Autonomous System (AS). IETF RFC 1930, March 1996.

[49] C. Hedrick. Routing Information Protocol. IETF RFC 1058, June 1988.

[50] A. Heffernan. Protection of BGP sessions via the TCP MD5 signature option. IETF RFC 2385, August 1998.

[51] Y. Hu, A. Perrig, and M. Sirbu. SPV: Secure Path Vector Routing for Securing BGP. In *Proceedings of ACM 2004 SIGCOMM*, Portland, OR, USA, August 2004.

[52] Y.C. Hu, D.B. Johnson, and A. Perrig. Secure Efficient Distance Vector Routing Protocol in Mobile wireless Ad Hoc Networks. In *Proceedings of the 4^{th} IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'02)*, June 2002.

[53] Y.C. Hu, A. Perrig, and D.B. Johnson. Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks. In *Proceedings of the Eighth ACM International Conference on Mobile Computing and Networking (MobiCom 2002)*, September 2002.

[54] Y.C. Hu, A. Perrig, and D.B. Johnson. Efficient Security Mechanisms for Routing Protocols. In *Proceedings of 2003 ISOC Symposium on Network and Distributed System Security (NDSS'03)*, San Diego, CA, USA, February 2003.

[55] D.A. Huffman. A Method for the Construction of Minimum-Redundancy Codes. In *Proceedings of the Institute of Radio Engineers*, volume 40, pages 1098–1101, September 1952.

[56] Geoff Huston. *ISP Survival Guide*. Addison-Wesley, Reading, MA, 1998.

[57] IANA. The Internet Assigned Numbers Authority, 2005. http://www.iana.org.

[58] J. Ioannidis, S.M. Bellovin, and A.D. Keromytis. Deployable Interdomin Routing Security. work-in-progress, 2005.

[59] IRR. Internet Routing Registry, 2005. http://www.irr.net.

[60] ISC. F-ROOT. http://www.isc.org/ops/f-root/.

[61] J. Klensin. Simple Mail Transfer Protocol. IETF RFC 2821, April 2001.

[62] D. Johnson and D.A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. *Mobile Computing, chapter 5*, pages 153–181, 1996.

[63] G.M. Jones. The Case for Network Infrastructure Security. In *;login: The Magazine of USENIX and SAGE*, volume 27, pages 25–29. December 2002.

[64] A. Josang. An Algebra for Assessing Trust in Certificate Chains. In *Proc. of the 1999 ISOC Symposium on Network and Distributed Systems Security (NDSS'99)*, pages 75–85, San Diego, CA, USA, February 1999.

[65] M. Just, E. Kranakis, and T. Wan. Resisting Malicious Packet Dropping in Wireless Ad Hoc Networks. In *Proceedings of the* 2^{nd} *Annual Conference on Adhoc Networks and Wireless (ADHOCNOW'03)*, Montreal, Canada, October 2003.

[66] C. Kahn. Using Independent Corroboration to Achieve Compromise Tolerance. In *Proceedings of Information Survivability Workshop (ISW'98)*, Orlando, FL, USA, October 1998.

[67] J. Karlin, S. Forrest, and J. Rexford. Pretty Good BGP: Protecting BGP by Cautiously Selecting Routes. Technical Report University of New Mexico Technical Report TR-CS-2005-37, October 2005.

[68] S. Kent. Securing the Border Gateway Protocol: A Status Update. In *Proceedings of the 7^{th} IFIP TC-6 TC-11 Conference on Communications and Multimedia Security*, October 2003.

[69] S. Kent and P. Atkinson. Security Architecture for the Internet Protocol. IETF RFC 2401, November 1998.

[70] S. Kent and P. Atkinson. IP Encapsulating Security Payload (ESP). IETF RFC 2406, November 1998.

[71] S. Kent, C. Lynn, J. Mikkelson, and K. Seo. Secure Border Gateway Protocol (S-BGP) Real World Performance and Deployment Issues. In *Proc. of the 2000 ISOC Symposium on Network and Distributed Systems Security (NDSS'00)*, February 2000.

[72] S. Kent, C. Lynn, and K. Seo. Secure Border Gateway Protocol (S-BGP). *IEEE Journal on Selected Areas in Communications*, 18(4), April 2000.

[73] N. Koblitz and A. Menezes. Another Look at "Provable Security". Cryptology ePrint Archive, Report 2004/152. To Appear in *Journal of Cryptology*. http://eprint.iacr.org/2004/152/, 2004.

[74] C. Kruegel, D. Mutz, W. Robertson, and F. Valeur. Topology-Based Detection of Anomalous BGP Messages. In *Proceedings of the 6^{th} Symposium on Recent Advances in Intrusion Detection (RAID'03)*, September 2003.

[75] B. Kumar and J. Crowcroft. Integrating Security in Inter-domain Routing Protocols. *ACM SIGCOMM Computer Communication Review*, 23(5):36–51, October 1993.

[76] LACNIC. Latin American and Caribbean Internet Addresses Registry, 2005. http://www.lacnic.net.

[77] L. Lamport. The Implementation of Reliable Distributed Multiprocess System. *Computer Networks*, 2:95–114, 1978.

[78] L. Lamport. Time, Clocks and the Ordering of Events in a Distributed Systems. *Communication of the ACM*, 21(7):558–565, July 1978.

[79] M. Lentczner and M. Wong. Sender Policy Framework: Authorizing Use of Domains in MAIL FROM. IETF Internet Draft, October 2004.

[80] C. Lynn, S. Kent, and K. Seo. X.509 Extensions for IP Addresses and AS Identifiers. IETF Internet Draft, September 2003.

[81] M. Lynn. Cisco IOS Security Architecture. BlackHat USA 2005, July 2005.

[82] J. Lyon and M. Wong. Sender ID: Authenticating E-Mail. Internet Draft, October 2004.

[83] G. Malkin. RIP Version 2. IETF RFC 2453 (Standards Track), November 1998.

[84] S. Marti, T.J. Giuli, K. Lai, and M. Baker. Mitigating Routing Misbehavior in Mobile Ad Hoc Networks. In *Proceedings of the* 6^{th} *Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM 2000)*, August 2000.

[85] U. Maurer. Modelling a Public-Key Infrastructure. In *Proceedings of the* 4^{th} *European Symposium on Research in Computer Security (ESORICS'96)*, pages 324–350, 1996.

[86] J.M. McQuillan, G. Falk, and I. Richer. A Review of the Development and Performance of the ARPANET Routing Algorithm. *IEEE Transactions on Communications*, 26(12):1802–1811, December 1978.

[87] A.J. Menezes, P.C. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[88] R. Merkle. Secure Communications over Insecure Channels. *Communications of the ACM*, 21:294–299, 1978.

[89] S. A. Misel. Wow, AS7007! NANOG mail archives, http://www.merit.edu/mail.archives/nanog/1997-04/msg00340.html.

[90] V. Mittal and G. Vigna. Sensor-Based Intrusion Detection for Intra-Domain Distance-Vector Routing. In *Proceedings of the 9th ACM Conferences on Computer and Communication Security (CCS'02)*, Washington, D.C., USA, November 2002.

[91] J. Moy. OSPF Version 2. IETF RFC 2328 (Standards Track), April 1998.

[92] S. Murphy. BGP Security Protections. IETF Internet Draft, February 2002.

[93] S. Murphy. BGP Security Vulnerabilities Analysis. IETF Internet Draft, February 2002.

[94] S. Murphy and M. Badger. Digital Signature Protection of the OSPF Routing Protocol. In *Proceedings of the 1996 ISOC Symposium on Network and Distributed Systems Security (NDSS'96)*, San Diego, CA, USA, April 1996.

[95] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP. IETF RFC 2560 (Standards Track), June 1999.

[96] RIPE NCC. K-ROOT. http://k.root-servers.org/.

[97] D.M. Nicol, S.W. Smith, and M.Y. Zhao. Evaluation of Efficient Security for BGP Route Announcements using Parallel Simulation. *Simulation Pratice and Theory Journal, special issue on Modeling*, June 2004.

[98] D. Oran. OSI IS-IS Intra-domain Routing Protocol. IETF RFC 1142, February 1990.

[99] V.N. Padmanabhan and D.R. Simon. Secure Traceroute to Detect Faulty or Malicious Routing. In *ACM SIGCOMM Workshop on Hot Topic in Networks (HotNets-I)*, Princeton, NJ, USA, October 2002.

[100] C. Partridge, T. Mendez, and W. Milliken. Host Anycasting Service. RFC 1546, November 1993.

[101] D. Pei, D. Massey, , and L. Zhang. Detection of Invalid Announcements in RIP protocols. In *Proceedings of IEEE Globecom 2003*, San Francisco, CA, USA, December 2003.

[102] C.E. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In *Proceedings of 1994 ACM SIGCOMM*, August 1994.

[103] R. Perlman. *Network Layer Protocols with Byzantine Robustness*. PhD thesis, MIT, October 1988.

[104] R. Perlman. *Interconnections: Bridges, Routers, Switches, and Internetworking Protocols, 2nd Edition*. Addison Wesley, Reading, MA, 1999.

[105] J. Postel. User Datagram Protocol. IETF RFC 768 (Standards Track), August 1980.

[106] J. Postel. Internet Control Message Protocol. IETF RFC 792 (Standards Track), September 1981.

[107] J. Postel. Simple Mail Transfer Protocol. IETF RFC 821 (Standards Track), August 1982.

[108] M. Reiter and S. Stubblebine. Toward Acceptable Metrics of Authentication. In *Proceedings of 1997 IEEE Symposium on Security and Privacy*, pages 10–20, 1997.

[109] Y. Rekhter and T. Li. A Border Gateway Protocol 4 (BGP 4). IETF RFC 1771, March 1995.

[110] P Resnick, R. Zeckhauser, E. Friedman, and K. Kuwabara. Reputation Systems: Facilitating Trust in Internet interactions. *Communications of the ACM*, 43(12):45–48, 2000.

[111] A. Retana and R. White. BGP Custom Decision Process. IETF Internet Draft, October 2002.

[112] RIPE. Réseaux IP Européens, 2005. http://www.ripe.net.

[113] R. Rivest. The md5 message-digest algorithm. IETF RFC 1321, April 1992.

[114] R. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public Key Cryptosystems. *Communications of the ACM*, 21:120–126, 1978.

[115] T. Roughgarden. *Selfish Routing*. Cornell University, May 2002. PhD thesis.

[116] RouteViews. Route Views Project, 2005. http://www.routeviews.org.

[117] RPsec. IETF Routing Protocol Security (RPsec) Working Group, 2005. http://www.rpsec.org.

[118] F.B. Schneider. Synchronization in Distributed Programs. *ACM Transactions on Programming Languages and Systems*, 4(2):179–195, April 1982.

[119] F.B. Schneider. Implementing Fault-Tolerant Services Using the State Machine Approach: A Tutorial. *ACM Computing Surveys*, 22(4):299–319, December 1990.

[120] K. Seo, C. Lynn, and S. Kent. Public-Key Infrastructure for the Secure Border Gateway Protocol (S-BGP). In *IEEE DARPA Information Survivability Conference and Exposition II*, June 2001.

[121] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, Princeton, NJ, USA, 1976.

[122] SIDR. IETF Secure Inter-Domain Routing (SIDR) Bird of Feather (BOF), 2005. http://tools.ietf.org/agenda/64/sidr.html.

[123] B. Smith and J. Garcia-Luna-Aceves. Securing the Border Gateway Routing Protocol. In *Proceedings of Global Internet'96*, pages 20–21, London, UK, November 1996.

[124] B. Smith, S. Murthy, and J.J. Garcia-Luna-Aceves. Securing Distance Vector Routing Protocols. In *Proceedings of the 1997 ISOC Symposium on Network and Distributed System Security (NDSS'97)*, San Diego, California, USA, February 1997.

[125] L. Subramanian, V. Roth, I. Stoica, S. Shenker, and R.H. Katz. Listen and Whisper: Security Mechanisms for BGP. In *Proceedings of the First Symposium on Networked Systems Design and Implementation (NSDI'04), San Francisco, CA, USA*, March 2004.

[126] I. Thomson. Possible DNS Hack Knocks out Google, May 2005. http://www.vnunet.com/news/1162902.

[127] P.C van Oorschot. Message Authentication by Integrity with Public Corroboration. In *Proceedings of the ACSA 2005 New Security Paradigms Workshop*, Lake Arrowhead, CA, USA, 2005.

[128] J. Viega and G. McGraw. *Building Secure Software*. Addison-Wesley Professional, 2001.

[129] C. Villamizar, C. Alaettinoglu, D. Meyer, and S. Murphy. Routing Policy System Security. IETF RFC 2725, December 1999.

[130] A.J. Viterbi. *CDMA: Principles of Spread Spectrum Communication (Addison-Wesley Wireless Communications Series)*. Addison-Wesley Publishing, Redwood City, CA, USA, May 1995.

[131] T. Wan. Personal Communication with Cogent Help Desk, September 2005.

[132] T. Wan. Personal Communication with Cogent Network Operation Center (NOC), July 2005.

[133] T. Wan. Personal Communication with Cogent Network Operation Center (NOC), September 2005.

[134] T. Wan. Personal Communication with David Presotto, June 2005.

[135] T. Wan. Technical Contact at Cogent, September 2005. http://www.atm.tut.fi/list-archive/nanog-2005/msg07310.html.

[136] T. Wan, E. Kranakis, and P.C. van Oorschot. S-RIP: A Secure Distance Vector Routing Protocol. In *Proceedings of Applied Cryptography and Network Security (ACNS'04), (academic track)*, volume 3089, pages 103–119, Yellow Mountain, China, June 2004.

[137] T. Wan, E. Kranakis, and P.C. van Oorschot. Securing the Destination-Sequenced Distance Vector Routing Protocol (S-DSDV). In *Proceedings of the 6^{th} International Conference on Information and Communications Security (ICICS'04)*, volume 3269, pages 358–374, Malaga, Spain, October 2004.

[138] T. Wan, E. Kranakis, and P.C. van Oorschot. Pretty Secure BGP (psBGP). In *Proceedings of the 2005 ISOC Symposium on Network and Distributed Systems Security (NDSS'05)*, San Diego, California, USA, 2005.

[139] T. Wan and P.C. van Oorschot. Analysis of BGP Prefix Origins During Google's May 2005 Outage. In *Proceedings of the 2^{nd} International Workshop on Security in Systems and Networks (SSN2006)*, Rhode Island, Greece, April 2006 (to appear).

[140] T. Wan, P.C. van Oorschot, and E. Kranakis. A Selective Introduction to Border Gateway Protocol (BGP) Security Issues. In *Proceedings of NATO Advanced Studies Institute on Network Security and Intrusion Detection*, Nork, Yerevan, Armenia. IOS Press (to appear, 2006), October 2005.

[141] F.Y. Wang and F.S. Wu. On the Vulnerablity and Protection of OSPF Routing Protocol. In *Proceedings of IEEE 7^{th} International Conference on Computer Communications and Networks*, Lafayette, LA, USA, October 1998.

[142] P.A. Watson. Slipping in the Window: TCP Reset Attacks. In *CanSec West (Core'04)*, Vancouver, Canada, April 2004.

[143] B.M. Waxman. Routing of Multipoint Connections. *IEEE Journal on Selected Areas in Communications*, 6(9):1617–1622, 1988.

[144] R. White. Securing BGP through Secure Origin BGP. *The Internet Protocol Journal*, 6(3):15–22, September 2003.

[145] R. White, D. McPherson, and S. Sangli. *Practical BGP*. Addison-Wesley, Reading, MA, 2004.

[146] B. Yu and M.P. Singh. Distributed Reputation Management for Electronic Commerce. *Computational Intelligence*, 18(4):535–549, 2002.

[147] M.G. Zapata and N. Asokan. Securing Ad Hoc Routing Protocols. In *Proceedings of the ACM Workshop on Wireless Security (WiSe 2002)*, September 2002.

[148] K. Zhang. Efficient Protocols for Signing Routing Messages. In *Proceedings of 1998 ISOC Symposium on Network and Distributed System Security (NDSS'98)*, San Diego, CA, USA, March 1998.

[149] Y.G. Zhang, W. Lee, and Y.A. Huang. Intrusion Detection in Wireless Ad-Hoc Networks. In *Proceedings of the* 6^{th} *Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM 2000)*, August 2000.

[150] M.Y. Zhao, S.W. Smith, and D.M. Nicol. Aggregated Path Authentciation for Efficient BGP Security. In *Proceedings of* 12^{th} *ACM Conference on Computer and Communications Security*, Alexandria, VA, USA, November 2005.

[151] M.Y. Zhao, S.W. Smith, and D.M. Nicol. Evaluating the Performance Impact of PKI on BGP Security. In *Proceedings of* 4^{th} *Annual PKI Research Workshop (PKI'05)*, Gaithersburg, MA, USA, April 2005.

[152] X. Zhao, D. Pei, L. Wang, D. Massey, A. Mankin, S. Wu, and L. Zhang. An Analysis of BGP Multiple Origin AS (MOAS) Conflicts. In *Proceedings of ACM 2001 SIGCOMM Internet Measurement Workshop*, 2001.

[153] P. Zimmermann. *The Official PGP User's Guide (second printing)*. Cambridge, MA: MIT Press, 1995.

[154] J. Zsako. PGP Authentication for RIPE Database Updates. IETF RFC 2726, December 1999.