Securing BGP Using External Security Monitors [*]

Patrick Reynolds, Oliver Kennedy, Emin Gün Sirer, Fred B. Schneider
{reynolds,okennedy,egs,fbs}@cs.cornell.edu

Abstract

Security modifications to legacy network protocols are expensive and disruptive. This paper outlines an approach, based on *external security monitors*, for securing legacy protocols by deploying additional hosts that locally monitor the inputs and outputs of each host executing the protocol, check the behavior of the host against a safety specification, and communicate using an overlay to alert other hosts about invalid behavior and to initiate remedial actions. Trusted computing hardware provides the basis for trust in external security monitors. This paper applies this approach to secure the Border Gateway Protocol, yielding an external security monitor called N-BGP. N-BGP can accurately monitor a BGP router using commodity trusted computing hardware. Deploying N-BGP at a random 10% of BGP routers is sufficient to guarantee the security of 80% of Internet routes where both endpoints are monitored by N-BGP. Overall, external security monitors secure the routing infrastructure using trusted computing hardware and construct a security plane for BGP without having to modify the large base of installed routers and servers.

1 Introduction

Security improvements to legacy network protocols often require modifications to the protocol implementation and changes to or replacements for all deployed hosts executing the protocol. Such changes are expensive and can be disruptive, especially if the protocol is deployed on infrastructure and the hosts are routers or servers. Significant benefits might not follow until most of the hosts have been upgraded. Security improvements that avoid modifications to a protocol implementation are cheaper, less disruptive, and thus more likely to get deployed.

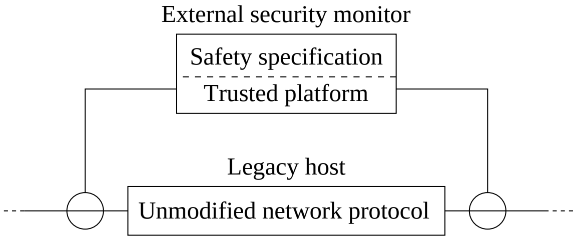

Figure 1: An external security monitor monitoring the traffic for a legacy host on two network links.

We propose *external security monitors* (ESMs) as a means of identifying malicious, misconfigured, or compromised hosts, thereby providing a way to secure legacy network protocols. An ESM is an additional trusted host or process dedicated to ensuring that one host executing a protocol complies with a safety specification. Each ESM obtains input and output messages from a legacy host—either by acting as a proxy or by sniffing the network—and issues certificates asserting that each message complies or does not comply with the safety specification.

ESMs must be trusted, and in our implementation this trust is based on guarantees provided by secure coprocessor hardware.[1] Figure 1 illustrates an ESM attached to a host executing an unmodified legacy protocol.

We demonstrate the ESM approach to securing legacy network protocols by applying it to the Border Gateway Protocol (BGP), a topic that has received much attention in recent years [6,9,13,20,22,25]. We chose BGP because it provides performance and scale challenges, because it admits a safety specification, and because no past attempt to secure it has been widely deployed. Our ESM implementation, N-BGP, runs on the Nexus, a new native operating system we have developed for trusted computing, though other systems that provide attestation [4, 16–19] could have been used instead.

The obvious alternative to ESMs would be to execute legacy protocols on trusted computing platforms. How-

[*] Supported in part by ONR Grant N00014-01-1-0968; AFOSR grant number F49620-03-1-0156; NSF grants 0208642, 0133302, 0430161, 0546568 (CAREER), and CCF-0424422 (TRUST); U.S. Department of Homeland Security, Office for Domestic Preparedness grant 2003-TK-TX-0003; and the following organizations: Cisco, ESCHER, HP, IBM, Microsoft, ORNL, Qualcomm, Pirelli, Sun, and Symantec. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of these organizations or the U.S. Government.

[1] Today's industry standard secure co-processors, like the Trusted Computing Group TPM [23], provide means for securely identifying software through remote attestation [5]. Such attestation creates a basis for trust, because recipients of a message now have assurance that the source of the message is executing the software it is supposed to.

ever, the ESM approach is less disruptive to deploy, because it does not require changes to the legacy protocol implementation or necessitate reconfiguration of deployed legacy hosts. ESMs are cheaper because an ESM does not need to run on server-class hardware, implement the target protocol or carry out its side effects.

ESMs can be applied to any protocol that admits a *safety specification*, a set of rules about what messages a host may send based on its configuration and messages previously sent and received. The output of an ESM is a sequence of certificates indicating either the conformance or non-conformance of the host against a set of locally checked safety criteria. Composition of judiciously crafted local safety criteria can enable some global properties, such as loop freedom in a routing protocol, to be maintained. Other global properties require coordination among ESMs. For example, the output stream of each ESM can be disseminated to other ESMs to enable remedial actions to be taken in response to invalid (nonconformant) behavior.

In the ESM approach, an overlay comprising the ESMs performs this dissemination. With N-BGP, this overlay constitutes a *security plane* that is used to announce UPDATE messages that an ESM has deemed to be invalid. ESMs forward such alerts on the security plane to legacy hosts and network administrators so that appropriate defensive action (e.g. blocking the use and forwarding of invalid routes) can be taken. Having an overlay allows ESM hosts to detect some invalid behavior by BGP speakers that are not monitored by a local ESM. ESMs can provide some benefits with as few as two ESMs deployed.

This paper makes the following contributions:

- We introduce the concept of external security monitors, a novel approach for adding security features to legacy network services without having to replace existing hardware and software.
- We present N-BGP, a prototype ESM that protects BGP. We provide a safety specification for BGP.
- We quantify the benefits realized from a partial deployment of N-BGP, compare it to S-BGP [9] and soBGP [13], and show that deployment rates as low as 10% can secure nearly 80% of all Internet routes where both endpoints deploy an ESM.

Section 2 describes BGP and some of the prior work on securing it. Section 3 describes in more detail the design space for ESMs. Section 4 describes our prototype, N-BGP, and Section 5 presents our experimental evaluation of N-BGP.

2 Background

The Border Gateway Protocol (BGP) [14] is the routing protocol deployed among autonomous systems (ASes) in the Internet. An autonomous system is a subset of the IP address space presumed to be under uniform administrative control. A *BGP speaker* is any router that implements BGP. BGP speakers are usually routers at the edges of autonomous systems. A large autonomous system may have many BGP speakers that coordinate to present a consistent view. BGP speakers maintain TCP connections to *peers*, BGP speakers at the other ASes with which the local AS has a peering relationship. A BGP speaker is connected to its peers by statically configured, one-hop links.

Each AS owns one or more IP prefixes (contiguous sets of IP addresses that share a set of leading bits) used by local hosts. Each BGP speaker maintains a table of routes to every valid IP prefix on the Internet. BGP speakers disseminate and discover this information by announcing their own IP prefixes and by receiving similar announcements from BGP speakers in peer ASes. A BGP speaker's configuration lists which prefix or prefixes it originates, which other BGP speakers are its peers, and policies for choosing a preferred route to each prefix.

The Internet has over 200,000 prefixes [15]. Any change in the location or routing of an IP prefix must be broadcast to all BGP speakers, leading to traffic that scales quadratically with the number of BGP speakers. To alleviate this inefficiency, BGP speakers often *aggregate* adjacent prefixes routed to or through a single peer. That is, two or more routes for contiguous prefixes with the same next hop can be combined. BGP speakers may also advertise a subset of a received prefix if one received advertisement partially supersedes another received advertisement. Advertising a subset of a prefix is called *de-aggregation*.

BGP involves four types of messages: OPEN, KEEPALIVE, NOTIFY, and UPDATE. Most BGP speakers use per-packet MD5 hashes [7] to achieve both authentication and integrity. TCP-MD5 protects against spoofing and connection-termination attacks, which is sufficient to protect OPEN, KEEPALIVE, and NOTIFY messages. This paper, like most BGP security research, is concerned with the content of UPDATE messages.

UPDATE messages allow BGP routers to announce and propagate routes. Each UPDATE message describes how to reach one or more destination prefixes. The AS_PATH field of an UPDATE message lists the ASes along which the UPDATE message was forwarded, which indicates a path (in reverse) that future data traffic can take to reach the listed destination prefixes. An UPDATE message may also contain one or more prefixes to withdraw. A BGP speaker withdraws a route when it becomes unavailable or otherwise ceases to be the preferred route.

2.1 BGP Security

Most attacks against BGP share a single goal: gaining control over traffic to one or more prefixes from one or more routers. Attackers might have any of several motives for doing so: eavesdropping, spoofing, denial of service, or simply increasing the volume of traffic carried over a particular network. Attacks to capture traffic are realized by false or modified UPDATE messages. An attacker's router can claim to be the origin for the target prefix, or it can claim to have a short (i.e., preferred) path to the target prefix. Either claim can be a purely fabricated message or a modified version of a legitimate message.

Accidental misconfigurations resemble attacks. For example, an administrator can configure a BGP router to originate the wrong prefix. Alternatively, a valid BGP speaker can be compromised, either by a remote attacker or by an insider.

Attackers can also block, modify, or inject packets into an established connection between two valid BGP speakers. Such attacks can be trivially addressed with TCP-MD5 [7]; we do not discuss them further in this paper.

Any system for securing BGP must provide strong guarantees, must be incrementally deployable, must be simple to configure, must have virtually no impact on BGP convergence time, and should not require the creation of new organizational entities [2]. A BGP security system should detect malicious, compromised, and misconfigured BGP speakers. Any BGP security system that uses online keys at each BGP speaker should remain secure even if some of the keys are compromised.

2.2 Related Work

Prior efforts to secure BGP can be grouped into four categories: certificate chains, link-state hybrids, routing registries, and comparison of multiple paths or perspectives.

Certificate Chains: One approach for verifying the integrity of a route is to issue hop-by-hop certificates for each component of the route; a certificate chain vouches for the full route. One such approach is S-BGP [9]. S-BGP establishes two certificate hierarchies: one to identify ASes and BGP speakers and a second to identify IP-prefix allocation. Keys identified by these certificates are used to sign each UPDATE with address attestations to verify the originating AS's right to host that prefix and route attestations to confirm the forwarding path the UPDATE took from the originating AS.

Pretty Secure BGP (psBGP) [25] refines S-BGP's approach by having the peers of an AS attest to that AS's right to originate a prefix, rather than depending on a centralized infrastructure to map IP ranges to ASes.

BIND [20] is a code attestation scheme that guarantees any UPDATEs a BGP speaker sends are valid if that BGP speaker has received valid input and is running valid code. BIND, like N-BGP, uses TPM hardware to enforce code integrity. However, BIND requires existing BGP routers to be replaced with trusted hardware, and BIND cannot provide guarantees for paths with even a single non-participating router.

Certificate chains depend on transitive trust. Any malicious router on an UPDATE-forwarding path can break the chain of trust by stripping all security information from the UPDATE message. In the case of BIND, any router on the UPDATE-forwarding path that is not running BGP atop BIND, even if it is not malicious, will break the chain of trust. Thus, these protocols cannot guarantee path integrity if any router on the path might be malicious.

Link-state Hybrids: Secure Origin BGP (soBGP) [13] uses a link-state approach to validate routes. Each soBGP speaker distributes a signed peer list to other soBGP speakers. Each soBGP speaker maintains an AS-level map of the Internet based on peer lists it receives. BGP speakers use this map to confirm the feasibility of routes they receive, though they cannot verify the path an UPDATE message took. To exchange information with soBGP speakers that are not peers (i.e., for incremental deployment), soBGP uses statically configured tunnels. Operators might resist soBGP because it requires public disclosure of peering and policy information.

SoBGP speakers can confirm the feasibility of a path even if that path has non-soBGP speakers on it. The soBGP speakers immediately before and after a non-soBGP speaker can verify that they each have a peering relationship with the non-soBGP speaker, which would make an UPDATE path through the non-soBGP speaker feasible. For example, if ASes A and C have soBGP speakers and AS B does not, and B peers with both A and C, then when the soBGP speaker at C receives an UPDATE with the AS_PATH $\{A, B\}$, it can confirm that the path is feasible, even though the speaker at B does not run soBGP. More generally, soBGP can confirm the feasibility of a path as long as no two non-soBGP speakers are adjacent on the path.

SoBGP speakers can confirm only the feasibility of the AS_PATH in a received UPDATE, not the actual path the UPDATE has been forwarded. Thus, an attacker may be able to advertise paths that exist but have not been legitimately advertised for policy reasons. SoBGP also requires online keys so that an soBGP speaker can announce changes in connectivity. Thus, compromising an soBGP speaker leads to a compromise of its keys.

Routing Registries: Routing registries store peering and policy information to help routers confirm the feasibility of the paths they receive. One routing registry is Internet Routing Validation (IRV) [6], in which each AS maintains the subset of the registry containing its peering and policy information. IRV hosts can share this information selectively and can export dynamic information, including route announcements, current routing tables, and advertised routes. One IRV host can query several others along a path to ensure that a received UPDATE has a legitimate origin and has not been corrupted in transit. Like ESMs, IRV hosts are deployed on independent hosts rather than on BGP speakers. However, IRV hosts affirm policy and state information rather than correctness, and they do not leverage trusted hardware to guard against compromises or misconfiguration. IRV hosts must also be manually synchronized to the router's policy and peering arrangements.

Routing registries have security properties similar to soBGP. A BGP speaker can confirm the feasibility of an UPDATE path (but not the path traveled by the UPDATE) as long as no two adjacent ASes on the path are absent from the registry.

Multiple Paths and Perspectives: Listen and Whisper [22] employs a passive approach to securing BGP. BGP speakers implementing Whisper add a new field to UPDATE messages. The BGP speaker originating a route places a random value in the field. Each BGP speaker forwarding an UPDATE replaces the value of the field with a hash of the previous value. A BGP speaker that receives two paths to the same prefix can check for path truncation attacks by comparing the lengths of both AS_PATHs and repeatedly hashing the value included with the shorter path. BGP speakers implementing Listen monitor dataplane traffic and signal an alarm if an unusually large amount of traffic to any given prefix range is rejected.

Whisper has security properties similar to certificate chains. If all paths to a destination contain BGP speakers without Whisper installed, then Whisper cannot make any guarantees about paths to that destination. Listen solves a different problem: detecting whether or not a chosen path is unusable for any reason (e.g., misconfiguration, an attack, or a network outage). A single BGP speaker with Listen installed can detect unusable paths. However, while other solutions are proactive, Listen can only discover unusable paths after the fact, after traffic has been misrouted and dropped. Listen can discover unusable paths that proactive techniques miss, making it a complementary technique.

3 External Security Monitors

External security monitors ensure that the protocol implementation on legacy hosts exhibits valid behavior, compliant with a safety specification. Installing an ESM to monitor each host in a distributed system ensures that all hosts in the system exhibit valid behavior. When some hosts are not monitored, then ESMs monitoring other hosts may be able to coordinate to derive guarantees about the behavior of unmonitored hosts. For example, if all of a host's peers are monitored, then all of the host's inputs and outputs will be seen by at least one ESM, even though the host itself is not monitored. The unmonitored hosts' peers' ESMs can communicate to determine if the unmonitored host's behavior is valid.

The attestations or warnings an ESM sends are only useful if the recipient trusts the ESM. A trusted computing platform helps establish this trust. Such a platform comprises secure hardware that provides the basis for trust and a secure operating system capable of attestation. In our implementation, a commodity secure co-processor called a Trusted Platform Module (TPM) [23] provides the hardware basis for trust, while the Nexus operating system, a small, secure microkernel that facilitates isolation between components provides the attestation. Overall, our system requirements are similar to that explored in previous work [5, 17]. The hardware carries an inherent key, which can be vetted by a certificate authority at the time of manufacture or established at a certificate authority at the time of initial establishment of "coprocessor ownership," as defined in the TPM specification.

Attestation produces a certificate, based on the secure coprocessor's embedded key, that captures all the binary hashes of the bootstrap loader, the operating system, the external security monitor, and the input configuration for the external security monitor. Such a certificate securely identifies the binary version of all relevant software and configuration data for an ESM installation. An attacker who modifies the ESM, the ESM configuration, or the underlying system environment for the ESM—for instance by modifying the disk image of the corresponding binaries on disk—cannot masquerade as a legitimate ESM, as the binary modifications will be apparent in the certificate.

ESMs can protect against several different kinds of attacks. They detect any network communication by the monitored host that violates the safety specification, whether it is the result of compromise by an outside attacker, misconfiguration, or even insider attacks.

The trusted hardware platform and the attestation services protect against compromises of the ESM code itself. While it is possible for the ESM software to contain a flaw, such as a buffer overflow, that an attacker can exploit, the effect of such compromises can be contained through system architecture. First, unlike traditional attestation that only checks a program's validity at load time, the Nexus supports *active attestation*, which allows it to verify runtime properties of an application. For example, the Nexus can check the integrity of the ESM code, or of sections

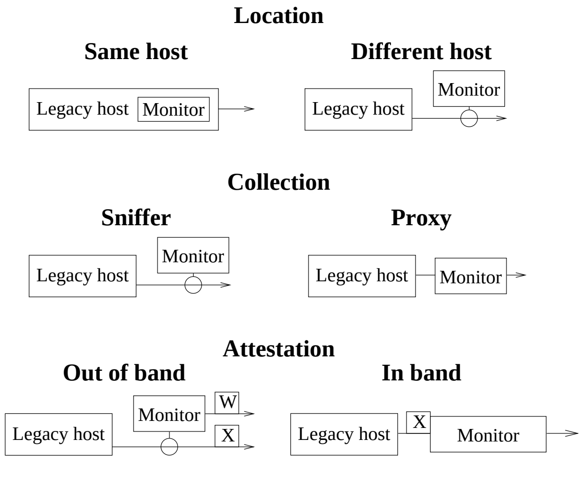

Figure 2: Three decisions regarding the deployment of external security monitors.

of ESM code, immediately before the code is executed, thereby ensuring that a change to the binary to circumvent the safety checking code is apparent in the certificate. Even in a system without active attestation, a one-time compromise cannot be amplified into a long-lived vulnerability, because persistent changes (e.g. to the ESM binary on disk) will be detected when the ESM is next restarted. Second, the Nexus provides a highly componentized system model where non-essential services reside in isolated protection domains outside the kernel. This yields a smaller trusted computing base for applications, as isolation eliminates non-essential services from the global trusted computing base. In the case of ESMs, this means that a compromise in one part of the ESM can be isolated from other parts. In particular, cryptographic functionality can be isolated in a separate task or in the TPM hardware itself, meaning that compromising an ESM does not yield its keys. An attacker cannot make a compromised ESM sign incorrect statements. Finally, ESMs are much simpler than the network protocols and services they monitor. ESMs do not have to implement a service's side effects, persistent state, or user interface, and do not need to place as great an emphasis on performance or availability. This increased simplicity makes ESMs less likely to be compromised in the first place.

3.1 Deployment

The placement of ESMs involves three considerations: where the ESM is placed relative to the host it is monitoring, how the ESM collects network messages to and from the host, and what the ESM does when it detects invalid behavior. These decisions are illustrated in Figure 2 and described in detail below.

Location: An ESM may be a process on the existing host, or it may run on an independent machine. Since the host running the ESM requires support for trusted computing, deploying ESMs on existing hosts is typically not feasible. An ESM must receive all of the monitored host's inputs and outputs, but an ESM on a separate host cannot always be assured that it is connected to all of a host's network links. A malicious administrator can stage a communication-hiding attack by disconnecting the ESM from one or more of the hosts's links. Often, an ESM can detect a communication-hiding attack by comparing the traffic it sees with traffic that the ESMs attached to the host's peers see, unless those peers' administrators are colluding in the attack.

Collection: An ESM may observe its host either by passively sniffing packets on all network links that the host is connected to or by acting as a proxy between the host and all other hosts. This choice is orthogonal to the location of the ESM. Deployment as a proxy can strengthen security by enabling the ESM to block invalid messages. However, acting as a proxy can make the ESM a performance and availability bottleneck, and it usually requires modifications to the monitored host's configuration. A transparent proxy may be deployed without modifying the host's configuration, but it remains a performance and availability bottleneck.

Attestation: ESMs may check the behavior of a host in one of two ways. The first is to attest to the correctness of the data flowing between two peer hosts by creating a safe channel over which only valid messages may be transmitted. We refer to this approach, taken by most prior work, as *in-band attestation*. In-band attestation requires adding new software to the communication path between two hosts; thus, it is possible only with active (proxy) collection. The second option, *out-of-band attestation*, is to create an additional channel for exchanging message-validity information. For example, ESMs could create a security plane that acts as a distributed database of untrustworthy hosts or messages.

An ESM that detects invalid behavior issues a certificate describing the behavior or naming the misbehaving host. The recipient of the certificate can be an administrator, another ESM, or a remote host speaking the same protocol. Local administrators might wish to be notified to detect a compromise or a misconfiguration. Remote administrators might be notified to ignore an invalid message or to sever a peering relationship with the compromised host. Other ESMs might use the information to vote or otherwise construct an aggregate view of the invalid behavior. Remote hosts might receive the warnings to prevent or undo the effects of the invalid messages.

A single ESM provides a useful benefit: it can warn local or remote hosts and administrators if it observes invalid behavior. If two peer hosts are both monitored, then the link between them can be considered trustworthy: any invalid messages on the link will be detected by the sender's ESM. In routing protocols, including BGP, if enough routers along a path are monitored, then the entire path is considered trustworthy: invalid behavior anywhere on the path will be detected. Also, a routing protocol might find many paths to a given destination; if any one path is trustworthy, then traffic to the destination can be sent along the trustworthy path. Thus, partial deployments of ESMs provide benefits that can be measured in terms of the fraction of links or paths considered trustworthy.

4 Monitor Architecture

N-BGP is an external security monitor for BGP. We refer to an ESM running N-BGP as an *N-BGP node*. Each N-BGP node monitors a single BGP speaker by sniffing each of that BGP speaker's links. N-BGP nodes coordinate using a security plane to propagate warnings about invalid UPDATE messages.

4.1 Safety Specification

External security monitors require a safety specification describing a host's valid output, possibly in terms of its configuration, it past inputs, or information received from other ESMs in the security plane. In N-BGP, we define a safety specification for UPDATE messages in terms of the prefixes a BGP speaker originates and the UPDATE messages it has previously received.

Routes in a BGP speaker's routing table are $\langle P, \phi \rangle$ pairs, where P is the set of IP addresses reachable by the route and ϕ is the set (or a superset) of ASes through which the route was forwarded to reach the BGP speaker. Each UPDATE message contains, among other fields, the advertised prefix and the AS_PATH. (For a description of these fields, see Section 2.) We map the advertised prefix to P and the AS_PATH to ϕ. In practice, the advertised addresses must be one or more prefixes, and the AS_PATH is frequently an ordered sequence. Our simplifications make our specification more permissive but do not compromise security, yielding the least restrictive safety specification for UPDATE propagation.

We define the set H_A of IP sets that the AS A is authorized to originate and the set F_A of all routes that AS A is allowed to forward. F_A contains all routes received from peers and not withdrawn, as well as routes for everything in H_A:

- $P \in H_A \Rightarrow \langle P, \emptyset \rangle \in F_A$

F_A also includes all potential aggregate routes:

- $\langle P_1, \phi_1 \rangle \in F_A \wedge \langle P_2, \phi_2 \rangle \in F_A$
  $\Rightarrow \langle P_1 \cup P_2, \phi_1 \cup \phi_2 \rangle \in F_A$

That is, F_A includes the aggregate of any pair of routes in F_A and H_A. The aggregate of two routes is defined as the union of their IP prefixes and the union of their AS_PATHs.

Prior work has characterized what properties a routing system, including any BGP deployment, must have in order to be secure [2, 3, 11, 20]. Based in part on this work, we define the following safety specification for BGP updates. The UPDATE $\langle P, \phi \rangle$ is a permissible non-withdrawal update for AS A to send if either of the following conditions holds:

- $P \subseteq H_A \wedge \phi = \{A\}$
- $\langle P', \phi' \rangle \in F_A \wedge P \subseteq P' \wedge \phi = \phi' \cup \{A\}$

That is, a message is valid if it is an advertisement for a prefix that AS A is authorized to originate or if it is a re-advertisement of a previously received UPDATE message. AS A must also include itself in the AS_PATH. These conditions allow aggregation by including aggregated paths implicitly in F_A. They allow de-aggregation by allowing the advertised prefix P to be a subset of any prefix in H_A or F_A.

In addition to this specification, a BGP speaker must re-advertise each route withdrawal it receives within some timeout δ. Withdrawals may be explicit, with prefixes listed in the withdrawal section of an UPDATE message, or implicit, with a previous advertisement replaced by a new advertisement for the same prefix. $W_{a \to b}(P, t)$ is a predicate that holds if BGP speaker a sends a withdrawal for the prefix P to BGP speaker b at time t. The rule for withdrawals is:

- $W_{a \to b}(P, t) \Rightarrow \forall x \in Peers(b) | x = a$
  $\vee (W_{b \to x}(P', t') \wedge P' \supseteq P \wedge t < t' \leq t + \delta)$

The N-BGP safety specification addresses only which route advertisements are legal under the BGP standard, and not which advertisements conform to a site's routing policy. In particular, N-BGP does not enforce any preference among valid routes, and it does not prevent a router from advertising secret peering arrangements. However, N-BGP's filtering and attestation mechanisms can enforce stricter, more detailed specifications; we are currently investigating extending N-BGP to support common site-specific policies.

4.2 N-BGP Implementation

N-BGP operates on a separate, dedicated host running the native Nexus operating system [21] on TPM-enabled hardware [23]. The small footprint of the Nexus OS, its partitioning of OS components into separate hardware protection domains, and its support for hardware attestation provide a suitable foundation for a secure network

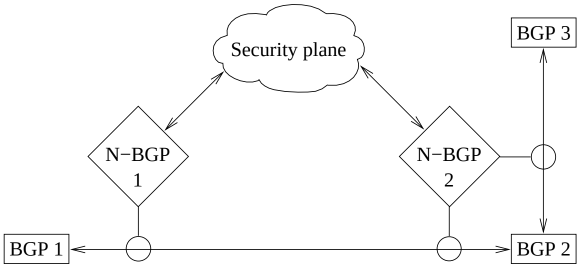

Figure 3: A three-speaker BGP network with two N-BGP nodes.

service. The Nexus provides an unforgeable certificate, rooted in a key embedded in the TPM secure coprocessor, that attests to the untampered execution of a particular N-BGP node. The Nexus attests to the integrity of execution by signing a hash of N-BGP's binary image on disk. Since this hash covers only a the code necessary for safety-checking BGP, it is unlikely to change frequently. A small database of trustworthy hashes can be maintained manually at each N-BGP node so that each node can confirm which other N-BGP nodes are running the correct code.

Our N-BGP implementation monitors BGP messages passively. Figure 3 shows how N-BGP fits into a simple BGP network. Our implementation captures packets to and from a BGP speaker using the *libpcap* library, reconstitutes the TCP stream, and checks all UPDATE messages using the safety specification in Section 4.1. While it is possible for a sniffer to miss packets and therefore to incur false positives or false negatives, in practice, BGP feeds have low data rates except when establishing new connections. N-BGP allocates a buffer to accommodate the burst of traffic accompanying a new BGP connection. We have monitored BGP connections for weeks without any packet losses.

Each N-BGP node starts with incomplete knowledge of its monitored router's state, because it has not seen prior announcements that the router received. Because BGP route announcements are not regularly refreshed, a purely passive N-BGP node's state would not converge to a complete view over time. Fortunately, either restarting the router's connections or retrieving its routing table through an administrative interface is sufficient to bootstrap the N-BGP node's state.

4.3 The Security Plane

N-BGP nodes communicate with each other for three reasons: to detect invalid behavior by unmonitored BGP speakers, to notify each other of invalid messages, and to coordinate with other monitors in an AS. An isolated N-BGP node can only detect invalid behavior by the BGP speaker it monitors, and it has no response to invalid behavior other than notifying a local administrator. When N-BGP nodes communicate, they can detect and counteract invalid behavior at remote BGP speakers, including some unmonitored speakers. Thus, an N-BGP node can provide more useful functionality and better incremental deployability if it is connected to other N-BGP nodes and to its monitored BGP speaker.

N-BGP nodes use the security plane to propagate queries and warnings. Queries allow N-BGP nodes to detect invalid behavior by some unmonitored nodes, and warnings allow N-BGP nodes to react automatically to invalid behavior at remote BGP speakers.

Whenever possible, communication in the N-BGP security plane occurs between peers. N-BGP nodes peer when the BGP speakers they monitor are peered. In addition, best-effort peering relationships with remote N-BGP nodes can be set up manually to bypass intermediate BGP speakers not running N-BGP. These tunnels are similar to the multi-hop tunnels used by soBGP. The security plane uses the same topology as the underlying BGP network. By using statically configured peering relationships—one-hop peers and multi-hop tunnels—N-BGP nodes avoid a circular dependency on the correctness of BGP.

When an N-BGP node is initialized, it sends to all of its peers an announcement containing its IP address, the AS of the BGP speaker it monitors, and a certificate. The certificate, C_A for N-BGP node A, contains the N-BGP node's public key and a hash of the N-BGP executable on disk, signed by the Nexus, plus an attestation chain rooted in the TPM. The certificate is sufficient to prove that the N-BGP node is running an intact copy of N-BGP, under an intact copy of the Nexus, attested to by a legitimate TPM. Each of the node's peers forwards the IP address and AS, but not the certificate, in a flood to all other N-BGP nodes in the network. Each of the node's peers also replies with its own certificate and a session key, K_{AB} for peer N-BGP node B responding to N-BGP node A. K_{AB} allows N-BGP node A to send messages to N-BGP node B with guaranteed confidentiality, integrity, and authenticity. The whole exchange is shown in Figure 4. The abbreviations used in the figure are explained in Table 4.3. Each N-BGP node keeps a table of remote N-BGP nodes' IP addresses, AS numbers, and certificates received in initialization announcements. N-BGP nodes also keep a table of the session keys established with each of their peers.

Invalid-Route Warnings: N-BGP nodes use out-of-band attestation, so they cannot directly block invalid UPDATE messages. Instead, an N-BGP node that detects an invalid UPDATE message floods a signed warning about the message to all other N-BGP nodes. The flood follows the peering relationships described above. The flood im-

Abbrev.	Definition
A, B	N-BGP nodes
C_A	Certificate for N-BGP node A
P_A,	Public key for N-BGP node A
K_{AB}	Secret key generated by N-BGP node B and shared with N-BGP node A
RVQ	Route-validity query
RVR	Route-validity response
W	Invalid-route warning
$\#_{A/RVQ}$	RVQ sequence number at N-BGP node A
$\#_{A/W}$	Warning sequence number at N-BGP node A
MAC	Message authentication code

Table 1: Abbreviations used in the protocol diagrams.

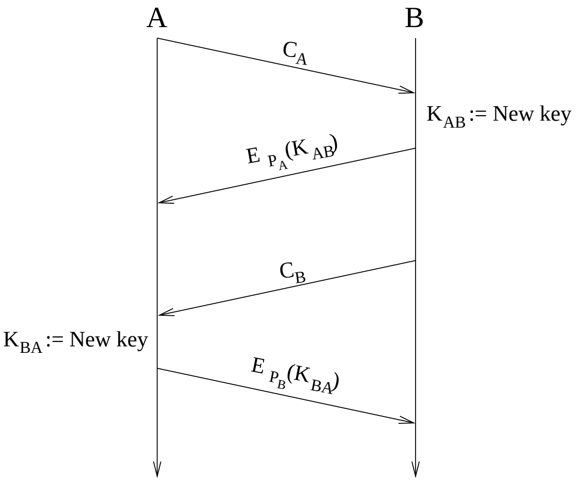

Figure 4: Messages sent when N-BGP node A joins the network, for each peer N-BGP node B.

plements, in effect, a distributed database of route advertisements deemed suspect by the N-BGP node that first detected them.

N-BGP floods two kinds of messages. The first is an *invalid-route warning*. Such a warning is generated whenever a local N-BGP node encounters a new route advertisement from its monitored BGP speaker that does not satisfy the BGP safety specification. N-BGP nodes prevent further dissemination of the invalid route by flooding an invalid-route warning. Each N-BGP node keeps its own database of known invalid routes and can either alert an operator or temporarily reconfigure the local BGP speaker to ignore any advertisement with a suffix matching a warning in the database. Since BGP messages are normally delayed up to 30 seconds to dampen route flapping [24], the security plane can almost always disseminate an invalid-route warning before the invalid route affects BGP speakers beyond the first hop. We expected invalid UPDATEs to be uncommon, making the number of warnings small and maintenance traffic and storage requirements low.

N-BGP floods a *clear-route* message when a route advertisement, previously considered to be invalid, becomes valid. For instance, a path $\{A, B, C\}$ advertised by AS A in the absence of other information would be marked invalid but would be cleared when AS A receives from AS B the path $\{B, C\}$.

Both invalid-route warnings and clear-route messages are sent encrypted with K_{AB}, if sent from N-BGP node A to N-BGP node B. That is, the message sent is $E_{K_{AB}}(W, \#_{A/W}, MAC)$. Invalid-route warnings and clear-route messages include a sequence number, both to prevent replay attacks and to ensure that messages are processed in the correct order.

N-BGP nodes use three techniques to defend against malicious BGP speakers sending enough invalid routes to constitute a denial-of-service attack. First, the database of invalid routes is stored on disk, allowing it to be large. Second, invalid routes are retired locally if they exceed the database size. Least recently used warnings are removed first, where use refers to a warning's last use in filtering an actual route seen on the control plane, or its arrival, if it has not been used yet. Finally, a BGP speaker that generates more than a threshold number of invalid routes for the database has all of its routes removed from the database and is placed in a separate database listing malicious BGP speakers. Unlike the invalid-route database, manual intervention is required for a BGP speaker to be cleared from the malicious-node database. This design enables the vast majority of short-lived invalid route advertisements to be managed automatically, while it keeps a malicious BGP speaker from overflowing the invalid-route database.

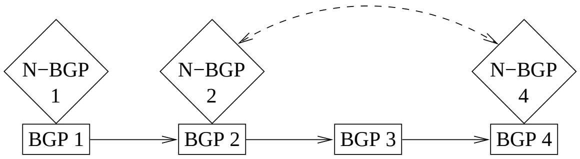

Figure 5: Route-validity query: upon receiving an UPDATE message (solid arrows), "N-BGP 4" contacts "N-BGP 2" to check that the unmonitored node "BGP 3" has correctly forwarded the UPDATE.

Route-Validity Queries: An N-BGP node can detect invalid behavior by an unmonitored BGP speaker if the BGP speakers immediately before it and after it in an UPDATE forwarding path are both monitored. To do this, N-BGP uses *route-validity queries*, akin to the queries IRV [6] uses to verify successor information. An N-BGP node that receives an UPDATE message from an unmonitored BGP speaker contacts the N-BGP node for the last monitored BGP speaker listed in the UPDATE. Figure 5 shows the BGP speakers and N-BGP nodes involved.

The AS_PATH lists the ASes visited but not specific BGP speakers or N-BGP nodes. Thus, the N-BGP node initiating the query uses its table of known N-BGP nodes to determine which ASes in the AS_PATH are monitored and by which N-BGP nodes they are monitored.

An N-BGP node receiving a route-validity query responds confirming the AS_PATH it forwarded and which AS it forwarded it to. An N-BGP node need only contact the N-BGP node for the last monitored BGP speaker, because that N-BGP node will have already verified the path up to and including itself. Route-validity queries can be batched and cached to reduce network load.

In some cases, the remote N-BGP node will not be a peer and delivery of the message will depend on BGP routing. Even so, the local N-BGP node can be assured that any response it receives comes from a valid N-BGP node, because the remote N-BGP process runs on a trusted platform that can vouch for the N-BGP code's integrity. Thus, an attacker cannot forge responses to route-validity queries. An attacker can block the query or the response, but the N-BGP node initiating the query can detect the lack of a response as a failure.

Figure 6 shows the details of the messages involved. The two nodes first execute a key exchange, to establish a secure communication channel and to confirm to the sending N-BGP node A that N-BGP node B is running valid code. If the nodes involved are peers or if A has sent a route-validity query to B recently, then A and B will already have a shared key K_{BA}, in which case they can skip the first three messages in the protocol. Route-validity queries include a sequence number to prevent replay attacks—i.e., an attacker replaying a route-validity response when a route is no longer valid.

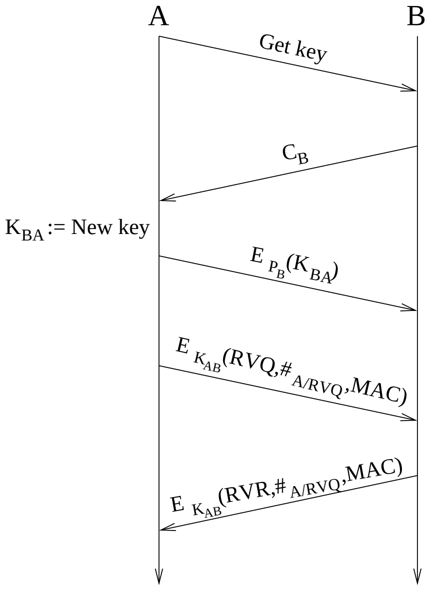
A
B
Get key
C_B
K_{BA} := New key
$E_{P_B}(K_{BA})$
$E_{K_{AB}}(RVQ,\#_{A/RVQ},MAC)$
$E_{K_{AB}}(RVR,\#_{A/RVQ},MAC)$

Figure 6: Messages sent when N-BGP node A submits a route-validity query to N-BGP node B.

One or more route-validity queries allow N-BGP nodes to detect whether or not any BGP speaker in a path forwarded an invalid UPDATE message, provided no two adjacent ASes on the path are unmonitored. A path verified in this manner is deemed *trustworthy*. A path containing only monitored BGP speakers requires no route-validity queries and is trivially trustworthy.

Intra-AS Coordination: BGP speakers within a large AS exchange routing information using an internal routing protocol such as iBGP or OSPF. N-BGP must either support all internal routing protocols as legitimate sources of routes to forward, or it must provide its own protocol to convey learned routes from one N-BGP node to another. We have chosen the latter approach. We treat an AS with many speakers as having one logical speaker: legal outputs from all BGP speakers in the AS are defined in terms of all prior inputs to any BGP speaker in the AS. All N-BGP nodes within an AS form a distributed database to store F, the table of authorized routes to forward. Incoming routes learned are added to F using *insert* messages, withdrawn routes are removed with *delete* messages, and outgoing route advertisements are checked using *lookup* messages.

5 Evaluation

We have evaluated N-BGP in two ways. First, we ran several experiments with a prototype implementation. Sec-

Host	OS	CPU	RAM
host1	Linux 2.6.14	Opteron 275 (2.2 GHz)	15 GB
host2	Linux 2.6.17	Athlon64 X2 5000+	2 GB
host3	Linux 2.6.17	Opteron 275 (2.2 GHz)	12 GB
host4	MacOS X 10.4	Core Duo (2.0 GHz)	512 MB
host5	Nexus	Pentium 4 (1.7 GHz)	512 MB
host6	Nexus	Pentium 4 (1.7 GHz)	512 MB

Table 2: The hosts used in all tests.

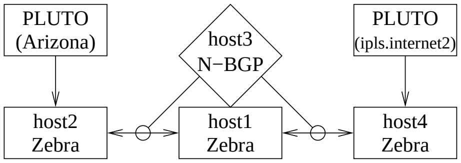

Figure 7: The testbed used to check for false positives based on UPDATE messages from two PLUTO feeds.

ond, we simulated random partial deployments on the Internet AS graph to estimate the utility that can be derived from different levels of deployment.

5.1 Prototype

We have tested our N-BGP prototype with a network of GNU Zebra [8] hosts (BGP speakers) receiving real UPDATEs from PLUTO sensors [12]. We have two goals in doing so: measuring the performance and costs of checking BGP traffic using commodity hardware and verifying that N-BGP does not generate spurious warnings for any valid traffic it receives in spite of a wide variety of BGP policies. The hosts we used are listed in Table 2.

To check for spurious warnings, we attached an N-BGP node to a Zebra host that was attached to two other Zebra hosts, receiving two different PLUTO feeds: planetlab1.arizona-gigapop.net and planetlab1.ipls.internet2.planet-lab.org. The testbed is shown in Figure 7. The monitored Zebra host received competing routes from two peers and advertised its preferred route in each case to the peer that did not send it. The N-BGP node confirmed that each preferred route (including all aggregations) was valid. We ran this test for four weeks without any false positives.

We ran short tests of one N-BGP node to quantify the resources it will need and the amount of traffic it can check. We measured the message-processing rate, the memory needed for N-BGP state, the buffer size and delay for checking BGP start-up traffic, the number of hops required for a warning to overtake an invalid UPDATE message, and the network overhead to execute a route-validity query.

Message-Processing Rate: An N-BGP node performs different processing for messages sent and received by the monitored BGP speaker. Messages received by the BGP speaker add, remove, or replace entries in the list of routes the BGP speaker is authorized to forward. Messages sent by the BGP speaker must be checked against the safety specification. The rate at which N-BGP can check messages received by the BGP speaker, averaged over 30 trials, is 20,417/sec (between 19,723 and 20,632 95% of the time). The rate at which N-BGP can check messages sent by the BGP speaker, averaged over 30 trials, is 27,770/sec (between 25,248 and 29,520 95% of the time). We determined both figures by measuring the time N-BGP took to process a full routing table from route-views.oregon-ix.net [15], which consists of 211,206 routes and 10.7 MB of BGP traffic The topology for this experiment is shown in Figure 8.

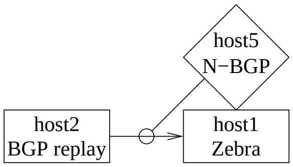

Figure 8: The testbed used to measure the performance of checking UPDATE messages.

N-BGP State: N-BGP must store all UPDATE messages that its BGP speaker has received that have not been withdrawn. We estimated the memory requirements for an N-BGP node monitoring a BGP speaker with a single peer by replaying the 10.7 MB RouteViews routing table. The test topology used is the same as above, as shown in Figure 8. The memory used was 35.6 MB: the size of the routing table plus overhead associated with efficient data structures. Monitoring a BGP speaker with several peers would take more memory—in the worst case, proportional to the number of peers, but normally much less because of similarities in prefixes advertised by the peers.

Start-Up Costs: When a connection is initiated between two BGP speakers, each BGP speaker sends its entire routing table to the other. This communication comprises hundreds of thousands of UPDATE messages in a few seconds or a few minutes. When checking the Route Views routing table, which takes 1.97 seconds to replay, N-BGP uses an average of 10.2 MB of its packet-capture buffer (over 30 trials). N-BGP takes about 10 seconds to finish checking the contents of the buffer.

Invalid-UPDATE Radius: When an N-BGP node detects an invalid UPDATE, it is by definition too late to prevent the recipient from acting on it. However, N-BGP nodes forward warnings faster than BGP speakers forward UPDATE messages, so the BGP speakers affected

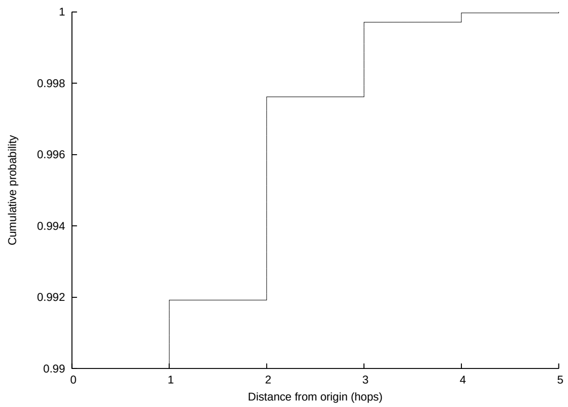

Figure 9: The number of hops an invalid UPDATE travels before N-BGP warnings overtake it. Note the *y* axis: over 99% of all invalid UPDATE messages are quashed within a single hop.

by an invalid route will usually be limited to a single hop. We simulated the race between UPDATE propagation and the flood of warnings using the actual Internet AS topology, as represented by the CAIDA AS Relationships Dataset [1]. We measured the time to detect an invalid message as $88\mu s$ and the time for an N-BGP node to receive, check, and forward an invalid-route warning as 44ms. BGP speakers delay UPDATEs up to 30 seconds [24]. Zebra in particular sends UPDATEs in batches at 30-second intervals. Thus, we modeled the UPDATE delay as a random quantity ranging from 0 to 30 seconds. In our simulations, over 99% of all invalid UPDATE messages were prevented by an invalid-route warning after a single hop. Figure 9 shows in detail the number of hops the worst 1% of invalid UPDATEs travel.

The number of BGP speakers affected by an invalid UPDATE depends on the out-degree of the BGP speaker sending it. 99% of the time, fewer than 50 nodes are affected. Figure 10 shows the number of hosts affected. Because the warning arrives within a few milliseconds, only very short-term harm is done at the hosts the invalid UPDATE does reach.

Route-Validity Queries: As described in Section 4.3, N-BGP uses route-validity queries to verify the validity of UPDATE messages received from unmonitored BGP speakers. We used the test configuration shown in Figure 11 to measure the costs of sending these queries. The average query latency was 357ms in the common case and 526ms when a key exchange was needed. The total query and response size is 77 bytes plus the AS_PATH size (2 bytes per hop) in the common case and about 2 KB when a key exchange is needed.

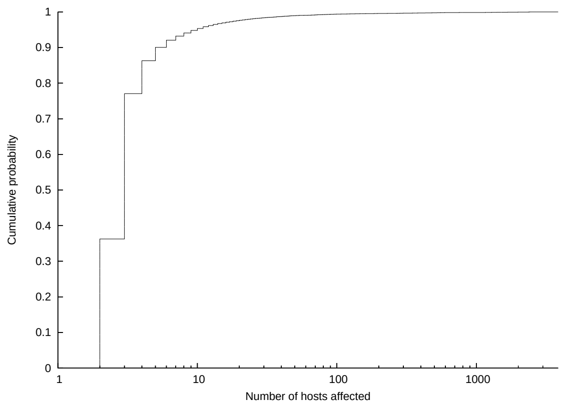

Figure 10: The number of hosts affected by an invalid UPDATE before N-BGP warnings overtake it.

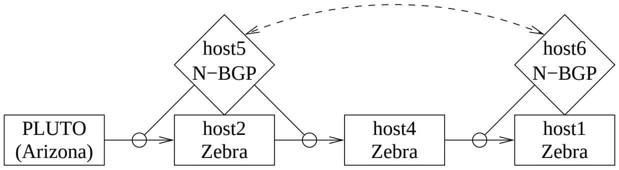

Figure 11: The testbed used to measure the performance of route-validity queries.

5.2 Deployability

One of the most important considerations for security modifications to an existing protocol is the feasibility of incremental deployment. Retrofitting a deployed system involves effort and expense and cannot take place all at once. Thus, new mechanisms must provide improved security even when only a few sites have been upgraded.

General metrics for security are elusive. However, we can quantify the fraction of AS-level routes that can be verified as trustworthy, as a function of how many BGP speakers have been upgraded. We refer to an AS as *secured* if the BGP speakers in it have a given security approach installed and enabled. In the case of N-BGP, a secured AS has one monitor per BGP speaker. We refer to the path between two hosts as *securable* if any trustworthy path exists between the two hosts.

We quantify the percentage of routes that are securable using N-BGP and compare it to three other recent efforts to secure BGP: s-BGP [9], BIND [20], and soBGP [13]. S-BGP and BIND require uninterrupted chains of trust; for these protocols, a path is trustworthy only if it comprises only secured ASes. Other protocols, including soBGP and N-BGP, tolerate one or more unsecured ASes but not two adjacent ones. Note that if the originating or the final AS is not secured, a client cannot establish trust in the overall path, so our analysis considers only paths where both endpoints are secured.

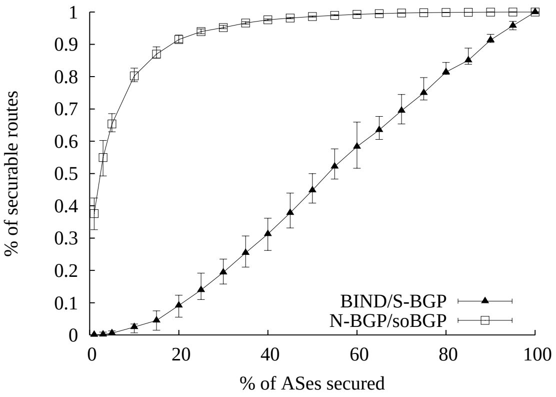

Figure 12: The percentage of paths that are trustworthy or that have at least one trustworthy alternative, if both endpoints are monitored.

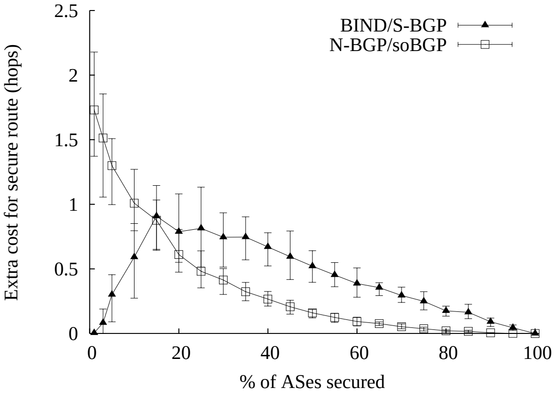

Figure 13: The average difference between the shortest trustworthy path and the overall shortest path, in hops.

We examine BGP security using the AS-level topology from the CAIDA AS Relationships Dataset [1]. We enumerated all AS pairs and counted which pairs had at least one trustworthy path between them, for different levels of deployment. We varied the deployment rate from 5% to 100%, in each case selecting a random set of ASes to mark as N-BGP-enabled. We ran 30 trials for each deployment rate. Figures 12 and 13 show average values for these 30 trials, with error bars indicating the 25th and 75th percentiles.

Figure 12 shows the percentage of endpoint pairs with at least one trustworthy path between them as a function of deployment rate. There are two lines measuring the percentage of securable paths: one for BIND and S-BGP and one for soBGP and N-BGP. Figure 13 shows the average added cost in hops, for each protocol, of choosing a trustworthy path instead of the shortest path. The added cost for BIND and S-BGP is low at low levels of deployment because so few paths, most of them between nodes at the core of the Internet, can be secured.

Protocols that can tolerate one or more non-adjacent unsecured AS provide much better incremental benefit than those that define a trustworthy path as consisting of only secured ASes. N-BGP and soBGP can both secure 90% of paths given deployment at a random 15% of ASes. For all the protocols shown here, the average difference between shortest paths and trustworthy paths is small—generally one AS-level hop or less. Thus, N-BGP is comparable to soBGP, and better than S-BGP and BIND, at securing routes with low levels of deployment.

6 Conclusions

The availability of trusted computing hardware affords an opportunity to improve the security of network protocols. For deployed systems where replacing or upgrading existing hosts is not feasible, we proposed the use of external security monitors, which treat existing hosts as black boxes and confirm, based on their inputs and outputs, that they correctly implement the desired protocol. Although this paper focuses on BGP, we believe that external security monitors are applicable to a wide variety of protocols.

This paper instantiates our proposed approach as N-BGP, an external security monitor running under the Nexus operating system. N-BGP complements the data and control planes by providing a security plane that implements a network-wide database of invalid routes. External security monitors are well suited to BGP, as they provide a cheap, simple, and incrementally deployable approach to secure the Internet routing infrastructure using trusted computing hardware.

References

[1] The CAIDA AS Relationships Dataset, June 26th, 2006. `http://www.caida.org/data/active/as-relationships/`.

[2] B. Christian and T. Tauber. BGP Security Requirements. Internet Draft, Apr. 2006.

[3] N. Feamster, H. Balakrishnan, and J. Rexford. Some Foundational Problems in Interdomain Routing. In *Proc. HotNets*, 2004.

[4] T. Garfinkel, B. Pfaff, J. Chow, M. Rosenblum, and D. Boneh. Terra: A Virtual Machine-Based Platform for Trusted Computing. In *Proc. SOSP*, Bolton Landing, NY, Oct. 2003.

[5] M. Gasser, A. Goldstein, C. Kaufman, and B. Lampson. The Digital Distributed System Security Architecture. In *Proc. National Computer Security Conference*, 1989.

[6] 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 *Proc. NDSS*, Feb. 2003.

[7] A. Heffernan. Protection of BGP Sessions via the TCP MD5 Signature Option. Request for Comments RFC-2385, Aug. 1998.

[8] K. Ishiguro. GNU Zebra. `http://www.zebra.org/`.

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

[10] R. Kuhn, K. Sriram, and D. Montgomery. Border Gateway Protocol Security: Recommendations of the National Institute of Standards and Technology. NIST Special Publication 800-54 (Draft), Sept. 2006.

[11] B. Kumar. Integration of Security in Network Routing Protocols. *SIGSAC Rev.*, 11(2):18–25, 1993.

[12] A. Nakao, L. Peterson, and A. Bavier. A Routing Underlay for Overlay Networks. In *Proc. ACM SIGCOMM*, Aug. 2003.

[13] J. Ng. Extensions to BGP to Support Secure Origin BGP (soBGP). Internet Draft, Apr. 2004.

[14] Y. Rekhter, T. Li, and S. Hares. A Border Gateway Protocol 4 (BGP-4). Request for Comments RFC-4271, Jan. 2006.

[15] The University of Oregon Route Views Project. route-views.oregon-ix.net snapshot. `http://archive.routeviews.org/oix-route-views/`, Sept. 15 2006.

[16] R. Sailer, E. Valdez, T. Jaeger, R. Perez, L. van Doorn, J. L. Griffin, and S. Berger. sHype: Secure Hypervisor Approach to Trusted Virtualized Systems. Technical Report RC23511 (W0502-006), IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY, Feb. 2005.

[17] R. Sailer, X. Zhang, T. Jaeger, and L. van Doorn. Design and Implementation of a TCG-based Integrity Measurement Architecture. In *Proc. of USENIX Security Symposium*, San Diego, CA, Aug. 2004.

[18] A. Seshadri, M. Luk, E. Shi, A. Perrig, L. van Doorn, and P. Khosla. Pioneer: Verifying Integrity and Guaranteeing Execution of Code on Legacy Platforms. In *Proc. SOSP*, Brighton, UK, Oct. 2005.

[19] A. Seshadri, A. Perrig, L. van Doorn, and P. Khosla. SWATT: Software-based Attestation for Embedded Devices. In *Proc. the IEEE Symposium on Security and Privacy*, Oakland, CA, May 2004.

[20] E. Shi, A. Perrig, and L. van Doorn. BIND: A Fine-Grained Attestation Service for Secure Distributed Systems. In *Proc. the IEEE Symposium on Security and Privacy*, Oakland, CA, 2005.

[21] A. Shieh, D. Williams, E. G. Sirer, and F. B. Schneider. Nexus: A New Operating System for Trustworthy Computing (extended abstract). In *Proc. SOSP*, Brighton, UK, Oct. 2005.

[22] L. Subramanian, V. Roth, I. Stoica, S. Shenker, and R. H. Katz. Listen and Whisper: Security Mechanisms for BGP. In *Proc. NSDI*, San Fransisco, CA, Mar. 2004.

[23] Trusted Computing Group. `http://www.trustedcomputinggroup.org/`.

[24] C. Villamizar, R. Chandra, and R. Govindan. BGP Route Flap Damping. Request for Comments RFC-2439, Nov. 1998.

[25] T. Wan, E. Kranakis, and P. van Oorschot. Pretty Secure BGP (psBGP). In *Proc. NDSS*, San Diego, CA, 2005.