# Using External Security Monitors to Secure BGP

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

***Abstract*—External security monitors (ESMs) are a new network component for securing legacy protocols without requiring modifications to existing hardware, software, or the protocol. An ESM is an additional host that checks each message sent by a legacy host against a safety specification. ESMs use trusted hardware to assure remote principals that the safety specification is being enforced; ESMs use an overlay network to alert each other about invalid behavior and to initiate remedial actions.**

**N-BGP is an ESM for securing the Internet's Border Gateway Protocol (BGP). When run on commodity hardware, N-BGP is fast enough to monitor a production BGP router. And deploying N-BGP at a random 10% of autonomous systems in the Internet suffices to guarantee security for 80% of Internet routes where both endpoints are monitored by N-BGP.**

## I. Introduction

BGP (the Border Gateway Protocol) assumes all participants follow the protocol as prescribed. This trust model is outdated, and the absence of mechanisms for a router to verify the correct operation of other routers creates opportunities for attacks. Proposed security improvements to BGP have required modifying the protocol implementation and changing or replacing deployed routers. Such change is expensive and, because in most cases routers running BGP also route data traffic, can be disruptive.

In this paper, we outline a new approach for securing BGP and other legacy network protocols. An *external security monitor* (ESM) is an additional host that monitors the inputs and outputs of some *target* host running a legacy protocol, checks the behavior of the target against a given safety specification, and communicates using an overlay to alert other hosts about invalid target behavior and to initiate remedial actions (Figure 1). An ESM thus works without requiring modifications to the target's hardware or software.

Unlike replication, PeerReview [1], and master-checker architectures [2], [3], an ESM need not duplicate the functionality of the target, which makes ESMs much simpler than the target they monitor (or a replica). Also, it makes ESMs applicable to securing protocols where legal outputs are not uniquely determined by inputs, as well as those where they are.

Because ESMs do not have to implement complete protocol functionality or provide a rich user interface, they can have a substantially smaller trusted computing base than the targets they monitor.[1] And because ESMs enforce a safety specification on network messages, a single ESM implementation is compatible with any implementation of a protocol, regardless of software version, configuration, or policy. Finally, because ESMs deployed in different administrative domains communicate using an overlay, ESMs have access to information not available at any one target, and ESMs can globally coordinate remedial actions.

An ESM differs fundamentally from the other, now ubiquitous network components designed to control network misbehavior. Unlike firewalls [5], ESMs can communicate with each other and thereby check global safety properties. Unlike intrusion detection systems [6], ESMs check required protocol properties rather than evaluating heuristics, so ESMs never raise false alarms—taking remedial action is justifiable when an ESM detects a problem. Finally, because an ESM runs on a platform capable of attestation, its security measures are useful to remote sites, as well as to the site deploying it.

In this paper, we apply the ESM approach to BGP, an Internet protocol with stringent performance and scale requirements.[2] The result is N-BGP, an ESM for BGP. It defends against false-origination and path-truncation attacks, which give rise to BGP route hijacking, traffic stealing, and black holes. Although these attacks have been well known for more than ten years [7], today they are increasingly being exploited by spammers [8].

Each N-BGP host intercepts all BGP messages received and sent by a single target BGP router and checks them against a safety specification that characterizes route advertisements the target may send given the route advertisements it has received. For example, a router that has received routes no shorter than $n$ hops for a given remote destination should not announce routes

Supported by NICECAP cooperative agreement FA8750-07-2-0037 administered by AFRL, AFOSR grant F49620-03-1-0156, National Science Foundation Grants 0430161 and CCF-0424422 (TRUST), ONR Grant N00014-01-1-0968, U.S. Department of Homeland Security Office for Domestic Preparedness grant 2003-TK-TX-0003, and Microsoft Corporation.

[1] Even on platforms, such as Cisco IOS XR [4], that isolate the user interface component, a compromised user interface can still change the configuration to produce illegal routing behavior.

[2] The ESM approach is also applicable to other Internet protocols, including DNS and SMTP.

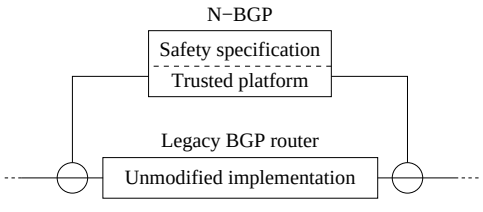


Fig. 1: An N-BGP host implements an ESM, monitoring the traffic for a legacy BGP router on two network links.

shorter than $n + 1$ hops for that destination. A shorter advertisement indicates a path truncation (also known as a black hole or traffic stealing) attack, and it will trigger N-BGP to take remedial action locally, by notifying the site administrator, and remotely, by purging the offending route advertisement from the network.

N-BGP monitors routing security properties not checked by the currently deployed BGP base, in essence providing a security plane for Internet routing. N-BGP does not limit a site's autonomy in negotiating policy, selecting routes, or performing aggregation. Instead, N-BGP can be used to check that peers obey site-specific policy agreements.

The ESM approach provides value even when monitors are not placed on all sites. *Virtual ESMs* can be created for targets not monitored directly, which enables the deployment of N-BGP at only a subset of routers to secure a larger set of hosts. For example, N-BGP deployment rates as low as 10% can secure 80% of all Internet routes where both endpoints deploy an ESM, while a comparable deployment of S-BGP secures only 2% of the routes.

Guarantees provided by an ESM are rooted in trusted hardware. Our N-BGP prototype uses a Trusted Platform Module (TPM) [9] and the Nexus [10] operating system, but any OS capable of attestation (e.g., DSSA [11] or TCGLinux [12]) will work. Attestation allows all sites to trust that traffic from any router monitored by N-BGP satisfies the given safety specification and that the router faithfully executes BGP origination and forwarding operations. Because the platform executing N-BGP attests to the ESM rather than attesting to the target system, the resulting trust is independent of the target system implementation.

Although an N-BGP host primarily protects remote routers from misbehavior by some target (a local router), sites also have three local incentives to deploy it:

- The N-BGP host detects any misconfiguration or compromise of the local target router that causes illegal protocol traffic.
- The N-BGP host acts as a destination for warnings from remote N-BGP hosts, and it aggregates information about the security of entire paths or portions of the network.
- Peers might require each other to deploy N-BGP or might prefer paths through N-BGP-monitored routers.

Past work on improving BGP security has not been widely deployed (See Section VI.) The reasons these solutions have not been deployed vary, but a few themes emerge. Almost all prior solutions have required hardware and/or software updates to deployed infrastructure. Most of the solutions deliver no benefit when deployed in isolation, and some deliver almost no benefit until deployed throughout the network. Some solutions impose configuration and maintenance burdens, either on deploying sites or on central registries. Finally, some solutions require that peering relationships and policy agreements be made public. N-BGP does not suffer from these limitations. Also, N-BGP is the first system to allow autonomous systems to monitor each others' policy compliance, which is commercially valuable even in the absence of attacks.

We proceed as follows. Section II introduces the concept of ESMs and describes the design space for ESMs, generalizing from lessons we learned in building N-BGP. Section III summarizes BGP operation, security requirements, and policies. Then, Section IV presents N-BGP, a prototype ESM that protects the BGP control plane from misconfiguration, compromise, and insider attacks. Section V demonstrates that N-BGP is fast enough to monitor existing and anticipated Internet BGP traffic. We also quantify the incremental benefit of deploying N-BGP. Finally, Section VI describes related efforts to secure BGP and other distributed systems.

## II. External Security Monitors

An ESM performs four functions: message acquisition, conformance checking, remediation, and assurance. *Message acquisition* is the means by which an ESM obtains its target's inputs and outputs. *Conformance checking* is the process by which messages the target sends are checked against a safety specification, given the messages previously received or sent by the target. *Remediation* defines how an ESM prevents or mitigates behavior that violates the safety specification. *Assurance* allows an ESM to convey to other ESMs that the safety specification is being enforced.

*Message acquisition:* There are two ways to add an ESM to a network. A *proxy ESM* explicitly filters and forwards relevant traffic between a target and all other hosts (Figure 2). An administrator must change the target's configuration—but normally not its software implementation—to send relevant traffic to the ESM rather than directly to a peer. Unrelated traffic (in the case of BGP, all data traffic) is not sent through the ESM.

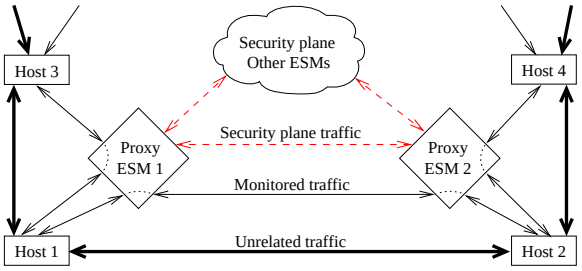


Fig. 2: Two proxy ESMs, each monitoring one host. Hosts 3 and 4 are unmonitored.

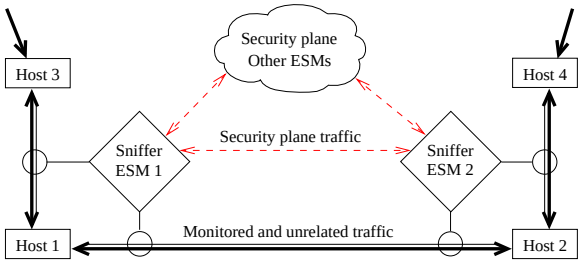


Fig. 3: Two sniffer ESMs, each monitoring one host. Hosts 3 and 4 are unmonitored.

By filtering traffic explicitly, a proxy ESM creates a safe channel over which only valid messages are transmitted. However, it may be a performance bottleneck. No messages, valid or invalid, are transmitted in the event that a proxy ESM fails.

In contrast, a *sniffer ESM* passively captures packets on all network links connected to some target (see Figure 3). Because a sniffer ESM must capture all traffic into and out of its target, it is inefficient in cases where most of that traffic is not relevant to the monitored protocol. A sniffer ESM cannot slow or break the underlying protocol, and the underlying protocol continues (albeit unmonitored) if the ESM fails. Unlike a proxy ESM, a sniffer ESM cannot block invalid traffic. Sending alerts on the security plane is the only way for a proxy ESM to make use of judgments about message validity.

Proxy and sniffer ESMs each have advantages and disadvantages. A sniffer ESM might not have sufficient capacity to isolate monitored protocol traffic on a backbone link; a proxy ESM receives only this traffic and thus is suitable for links of any speed. Sniffer and proxy ESMs interoperate, forming a single security plane. We show in Section V that both approaches are feasible for existing and anticipated levels of BGP traffic.

Operationally, the ESM administrator needs to ensure that the deployed ESM can acquire all relevant inputs to and outputs from the target. Otherwise, an attacker might engage in *communication hiding*, in which a compromised target hides malicious but relevant traffic from the ESM. Guarding against such attacks is not difficult.


Fig. 4: Two ESMs communicating to form a virtual ESM for an otherwise unmonitored target.

- A proxy ESM protects against communication hiding attacks by not sharing relevant identifying keys with the target. In the case of BGP, the TCP-MD5 [13] key normally used to identify a router can be provided to the ESM instead of the router, thereby preventing a compromised router from bypassing the proxy and forming direct sessions with its peers.
- A sniffer ESM can defend against communication hiding if the sniffer is connected to the target through an independent hardware mechanism, such as a mirrored switch port. Additionally, a sniffer ESM can periodically contact each peer's ESM to ensure that both ESMs have recorded the same relevant traffic on the link between them.

The order in which an ESM receives its target's inputs and outputs must be the same order the target receives and sends them, because whether a target's output is legal can depend on what inputs the target previously received. Proxy ESMs enforce an order by intercepting relevant traffic. Sniffer ESMs must be connected in a way that preserves message ordering within each monitored link and across all links connected to the target.

Since inputs and outputs of a target are the outputs and inputs, respectively, of its peers, ESMs surrounding an unmonitored target can combine their captured traffic to construct a virtual ESM for that target (Figure 4). A virtual ESM is easiest to construct for small sites (i.e., those with few connections), which we believe are the sites most likely to initiate attacks.

*Conformance checking:* The central task of the ESM is to vet each output of a target system against a safety specification. The safety specification defines legal outputs in terms of past behavior. Thus, in theory, the state requirements of an ESM are unbounded. But for many protocols, including BGP, storing a bounded subset of the target's history suffices. A safety specification could also depend on messages sent or received by remote hosts, although doing so requires coordination among ESMs using an overlay. We instantiate these possibilities in the context of BGP in Section IV.

*Remediation:* An ESM could issue a certificate attesting to each valid message its target sends. However, that would be costly. Instead, a typical ESM blocks or reacts to invalid messages.

A proxy ESM checks the validity of each message the host sends and blocks those it determines violate the

safety specification. Messages that cannot be validated immediately based on local information are either passed optimistically or are delayed. In either case, remote ESMs are then contacted to gather information necessary to validate the message. If a valid message was delayed, it is released; if an invalid message was sent, it is retracted. Not all protocols support retractions.

A sniffer ESM that detects invalid behavior (or a proxy ESM that must retract an invalid message) sends a warning certificate on the control plane identifying the problem. These certificates could arrive at the relevant ESM after the messages they concern. A target might thus act on an invalid message and later receive a warning; hence, sniffer ESMs and optimistic proxy ESMs are applicable only to systems that can tolerate or undo the effects of invalid messages.

Administrators receive warning certificates so they can remediate a compromise or misconfiguration. Upon receipt of such a certificate, an administrator might cause an invalid message to be ignored or might terminate a peering relationship with a compromised host. Other ESMs benefit from receiving a warning certificate because they can prevent or undo the effects of the invalid messages. Depending on the target protocol, an ESM receiving a warning can use an administrative interface to roll back the effects of each invalid message automatically.

*Assurance:* Trust in an ESM is based on hardware attestation. In the case of N-BGP, Nexus attests that the N-BGP binary is signed by a trusted authority.

Because an ESM can be built using a small trusted computing base, a trusted computing platform, and a narrow interface, it can be highly resistant to attacks. An ESM has limited exposure to the network because it only ever communicates with its target and with other ESMs. Communication with its target uses a dedicated, local channel, and communication with other ESMs uses a channel secured by keys that are owned and managed by the trusted platform. Finally, hardware attestation securely identifies the software at each ESM, thus ensuring that an attacker cannot amplify a short-term vulnerability by modifying ESM software in an undetectable manner. If an ESM were compromised, then it could be made to issue spurious warnings for valid behavior or to suppress warnings for invalid behavior; we discuss how N-BGP limits the scope of these attacks in Section IV-E.

## III. BGP Background

BGP [14] is the protocol routers use to announce, propagate, and withdraw routes between autonomous systems (ASes) in the Internet. An AS is a portion of the network presumed to be under a common administrative control. A *BGP speaker* is any router that participates in BGP; usually, it is a router at the edge of an AS. A large AS may have many BGP speakers that coordinate to maintain a consistent set of routes. BGP speakers in an AS maintain TCP connections to *peers*—BGP speakers at other ASes with which the AS has peering relationships. A BGP speaker is connected to its peers directly or by statically configured routes. A *control plane* contains traffic by BGP speakers that describes how traffic contained in the *data plane* is routed.

Each AS controls a subset of all IP addresses, represented as one or more IP address prefixes—contiguous sets of IP addresses with a common string of leading bits. Each BGP speaker maintains a table mapping IP prefixes to next-hop routing information. BGP speakers disseminate and discover this routing information by announcing their own IP prefixes and by receiving similar announcements from BGP speakers in peer ASes. A BGP speaker's configuration lists all prefixes it originates, other BGP speakers that are its peers, and policies for choosing a preferred route to each prefix.

The Internet has over 250,000 prefixes [15]. Since any change in the location or routing of an IP prefix must be broadcast to all BGP speakers, BGP traffic scales quadratically with the number of BGP speakers. To reduce this cost, BGP speakers *aggregate* routes for adjacent prefixes with similar paths.

BGP update messages (henceforth, *updates*) contain a list of prefixes and an AS_PATH. The prefixes describe a set of destinations, and the AS_PATH lists the ASes through which the update was forwarded, which constitute one possible path to reach the listed destinations. Updates can also contain other information, called *attributes*, which is used by BGP policies as described below.

### A. BGP Policies

In practice, shortest-path forwarding is not flexible enough to handle routing on the Internet. An AS might wish to ignore or modify some updates it receives for reasons of business policy, traffic engineering, scalability, or security. BGP *policies* allow administrators to dictate which updates a BGP speaker will accept and readvertise, as well as which they will modify and how.

BGP speakers choose among routes to a given destination using a seven-step decision process [16]. A BGP speaker can affect decisions made by other BGP speakers in the same AS by modifying the local preference of an update, which overrides AS path length and all other attributes. A BGP speaker can make an update less attractive to BGP speakers in downstream ASes by padding the AS path, generally with additional copies of the local AS number. A BGP speaker can only make its updates more attractive to BGP speakers in another AS if that AS's administrator has agreed to a policy as part

of a peering contract. The policy can change the values of update attributes to indicate levels of preference.

### B. BGP Threat Model

Most attacks against BGP share a single goal: gaining control over traffic to one or more prefixes in order to enable eavesdropping, spoofing, blocking traffic, or simply increasing the volume of traffic carried over a particular network. Attacks to capture traffic are realized by false originations or truncated paths. 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 made by disseminating a purely fabricated update or a modified version of a legitimate update. Invalid updates can occur due to buggy or compromised code or an invalid router configuration. Invalid router configurations can be intentional (an insider attack) or accidental. For example, an administrator might accidentally or intentionally configure a BGP router to originate the wrong prefix.

Other attacks on BGP include denial of service or modification of packets in transit between two BGP speakers. Effective defenses are already in place for resisting such locally targeted attacks [13]. So N-BGP instead focuses on attacks that cannot be detected by any single recipient.

## IV. N-BGP

N-BGP is our prototype implementation of an ESM for BGP. N-BGP acquires BGP traffic through peering connections (as a proxy)[3] or by sniffing; it checks each update against two sets of rules: the safety specification and site-specific policies. Updates that violate either one trigger alerts and optional scripts to prevent or undo their effects.

Like most approaches to BGP security, N-BGP concerns only control plane traffic. It does not monitor whether data plane traffic is routed according to BGP updates and policies. An administrator who needs this functionality could augment N-BGP with data plane traffic sampling or other complementary approaches [17]–[19].

In the discussion below, we first give the safety specification that N-BGP implements, then discuss how site-specific policies are enforced, and finally describe how both kinds of constraints are enforced using a trusted platform and the security plane.

[3]Many BGP implementations can only use one BGP session per peer IP. N-BGP uses multiple IP addresses on a single network interface to give the appearance that each proxy BGP session is connected to a distinct peer.

### A. Safety Specification

The safety specification defines valid updates in terms of (i) prefixes the local BGP speaker originates and (ii) updates it has previously received that have not been withdrawn. N-BGP considers an update valid if and only if that update advertises a locally owned prefix, forwards a suitably modified update received from another AS, or advertises a correctly constructed aggregate of local prefixes and/or received updates.

Each update a BGP speaker receives is a $\langle P, \phi \rangle$ pair, where $P$ is a list of IP address prefixes, and $\phi$ is an AS_PATH. A list of prefixes defines a set of IP addresses; thus, we treat $P$ as a set. An AS_PATH consists of one or more AS numbers, organized as a sequence and set. We define:

- $Agg(\phi, S)$ is a predicate that is true if and only if $\phi$ is a legal aggregate of all AS_PATHs in the set $S$. (Aggregation rules are given in the Appendix.)
- $\phi_1 \prec \phi_2$ is true if and only if every AS number in $\phi_1$ appears in the same order in $\phi_2$. $\phi_2$ may contain arbitrary AS numbers[4], interleaved freely, as long as the left-most element of $\phi_2$ is the same as the left-most element of $\phi_1$.
- $A\phi$ is a new AS_PATH equal to $\phi$ with the AS number $A$ prepended.

In addition, $[A]$ refers to an AS_PATH containing only AS $A$.

Let $I_a$ denote the set of updates received by router $a$ from its peers and not withdrawn (also called Adj-RIB-In); $O_{a \to b}$ denote the set of updates advertised by $a$ to a peer $b$ and not withdrawn (the subset of Adj-RIB-Out sent to $b$); and $H_A$ denote the set of addresses (or equivalently, the set of prefixes) that AS $A$ (and in turn, router $a$ located at $A$) is authorized to originate.

We define $F_a$ as the set of updates router $a$ is authorized to send; it satisfies the following safety specification:

- **Origination:** $\langle P, [A] \rangle \; where \; P \subseteq H_A$
- **Received updates:** $\langle P, A\phi \rangle \; where \; \langle P, \phi \rangle \in I_a$
- **Aggregation:** $\langle P_1 \cup P_2, \phi \rangle \; where \; \langle P_1, \phi_1 \rangle \in F_a \wedge \langle P_2, \phi_2 \rangle \in F_a \wedge Agg(\phi, \{\phi_1, \phi_2\})$
- **Padding:** $\langle P, \phi_2 \rangle \; where \; \langle P, \phi_1 \rangle \in F_a \wedge \phi_1 \prec \phi_2$

Note that $F_a$ is recursively defined as a closure under Aggregation and Padding.

$I_a$, $H_A$, and $O_{a \to b}$ may change over time. $I_a$ changes when peers of $a$ send or withdraw updates. $H_A$ changes when the prefixes originating at $A$ change. Thus, $F_a$ changes as a function of $I_a$ and $H_A$.

[4]Padding with arbitrary numbers can result in AS_PATHs that do not match any actual forwarding path; this is used to control downstream dissemination of an update, and it has no detrimental effect on the routing of data traffic.

Router $a$ may send an update $U$ at time $t$ if and only if $U \in F_a$ at time $t$. When $a$ sends update $U$ to $b$, $U$ is added to the set $O_{a \to b}$. If $U$ is later removed from $F_a$ (because an entry in $H_a$, $I_a$, or $F_a$ has been removed), then $a$ has a fixed amount of time (we use 60 seconds) to send a withdrawal for $U$ to each peer $b$ for which $U \in O_{a \to b}$. Both incoming and outgoing withdrawals may be implicit, for example, by replacing the withdrawn route with a new one.

Some of the rules in the above safety specification allow BGP speakers to modify updates in ways that might seem counterintuitive to readers unfamiliar with details of BGP, but are nevertheless valid for the protocol [14]. For example, a router that receives an update with the AS path [$A$ $B$] can pad the AS path to [$A$ $C$ $D$ $B$], to prevent downstream ASes from forwarding it to the ASes $C$ and $D$. Our safety specification (in Padding) permits such rare but valid behavior while ruling out path truncation attacks.

### B. Policy Specification

N-BGP can enforce site-specific *policies* in addition to the BGP safety specification given above. All ASes must adhere to the safety specification, but policies for each AS can differ and can vary over time. Policies can be used to codify business relationships or for traffic engineering [16].

Policies are expressed to N-BGP using *policy rules* written in Routing Policy Specification Language (RPSL) [20]. We chose RPSL because it is an established standard, compactly describes policies N-BGP can enforce, and can be translated directly to configuration scripts for common BGP implementations [21].

RPSL defines, among other things, a set of import and export rules, which affect incoming and outgoing updates, respectively. Each rule contains *matching constraints* and zero or more *actions*. The matching constraints define updates of interest based on the source or destination and on the values of attributes in the update; the action specifies how updates are filtered or modified. Filtering influences whether or not a BGP speaker will accept a route (inbound filtering) and whether or not it will re-advertise it (outbound filtering). By modifying attributes of the update, RPSL rules affect whether local and remote routers will prefer the advertised route.

Pairs of ASes, often peers, negotiate policies for each to apply on the other's behalf. The policy applied at a given AS is a combination of all the policies others have asked it to apply. Thus, policies from several peers can affect a single update. We rely on human administrators to distinguish between intentional interaction and accidental side effects when policies are composed.

N-BGP attests to the policy rules it is enforcing at any given time, enabling ASes to assure each other that they are implementing the policies they negotiated. Because policies can be business secrets, each policy rule has a configurable visibility, generally based on who requested that the rule be imposed. N-BGP sends attestations and warnings about a policy rule to peers authorized to know about it. Policy attestations are sent in response to explicit queries: when a remote AS asks what policy rules are being enforced, an N-BGP host responds with a list of only those policy rules visible to that peer.

Policy changes require synchronization. The N-BGP host and the target BGP speaker are both notified of any change in policy but are not told exactly when the change must take effect at the target. N-BGP must be prevented from subjecting a packet generated under some new policy to a safety check using the old policy, or vice versa. An obvious (but disruptive) way to keep the ESM and the target synchronized would be to stop the target every time the policy is changed. This, however, results in poor performance. We therefore implement an optimistic alternative. The new policy is sent first to the N-BGP host at time $t^0$, and then to the target at a later time $t$. All packets sent by a correctly operating target before $t$ conform to the old policy, and all packets sent after $t$ conform to the new policy. Because N-BGP does not know the exact value of $t$, it enforces a weaker property: within $\tau$ seconds of receiving the new policy, N-BGP will enforce it. That is, $t^0 \leq t \leq t^0 + \tau$ must hold. Enforcing this weaker property is possible without knowledge of $t$: after N-BGP receives the new policy, it checks each outgoing update against both the old and new policies. Receipt of an outgoing update legal under the new policy but not the old one signifies that $t$ has passed, and N-BGP thereafter checks all packets against only the new policy. In the absence of such a distinguishing packet, N-BGP switches to the new policy regardless after $\tau$ seconds (our prototype uses $\tau = 60$). This never falsely raises an alarm for a correctly operating target and never allows misbehavior that is distinguishable from a reasonable delay in switching to a new policy.

This synchronization protocol generalizes to multiple policy changes in a short time span. If policy changes are totally ordered—that is, $t_{i+1} > t_i$ for all $i \geq 1$—then any packet signaling that policy change $i$ has taken effect also signals that all policy changes prior to $i$ must have already taken effect. The N-BGP host checks policy $i$ until it receives a packet compliant with policy $j \geq i$ but not policy $i$ or until $t_i^0 + \tau$ passes. The N-BGP host may have to check arbitrarily many policies, up to the number allowed during the interval $\tau$.

### C. Enforcement

Each N-BGP host is a trusted platform comprising the Nexus operating system and a TPM. Nexus issues

a certificate giving hashes of the bootstrap loader, the operating system, and the N-BGP application code. The certificate identifies the binary executable for relevant software running on each N-BGP host. An attacker who modifies that software—for instance, by modifying the corresponding binaries on disk—cannot thereafter use that system to impersonate a legitimate N-BGP host. A small database of trustworthy hashes is maintained manually at each N-BGP host for use in checking that neighboring N-BGP hosts are running unmodified executables. Since this hash covers only the code to check the BGP safety specification and policies, it is unlikely to change frequently. Thus, maintaining the database manually is not a problem.

N-BGP determines which updates are valid and which routes are secure. One way for N-BGP hosts to use this information is to run configuration scripts through an administrative interface. Such scripts can change the preference value for an individual route or can drop the BGP connection to a misbehaving peer.

Our N-BGP implementation can operate as either a proxy ESM or a sniffer ESM. While it is possible for a sniffer ESM to miss packets, in practice, BGP packets are sent at roughly one update per second except when establishing new connections, which N-BGP accommodates using a buffer. We have monitored BGP connections using a sniffer ESM for weeks without observing any missed packets. If N-BGP misses a packet, it will detect that fact by a gap in TCP sequence numbers and will use a script or notify an administrator to restart the affected connection.

N-BGP detects prefix hijacking attacks—that is, updates advertising a prefix the sender does not own—by checking *origin authentication certificates* on each update. While N-BGP could be used to deploy several origin authentication systems [22]–[24], we have implemented Grassroots [25], which has a simple, bottom-up approach to deployment. N-BGP provides a way to deploy Grassroots (both generating and checking certificates) without replacing existing routers and provides a way to initiate remedial action, even on routers that have already received the invalid update.

N-BGP detects path truncation attacks by enforcing the safety specification (§IV-A) on all forwarded updates. If a monitored BGP speaker sends an update that is not a copy of an update it has received, then that speaker's N-BGP host will detect the invalid behavior and either block the update or issue an alert.

If an unmonitored BGP speaker attempts a path truncation attack, then downstream N-BGP hosts may be unable to detect the attack. However, N-BGP can identify verifiable paths, for which each forwarding AS is monitored directly or by its neighbors. Given two or more routes to the same destination, N-BGP will identify the shortest one that is verifiable, even if it is longer than the overall shortest path. And N-BGP can influence the target's decisions by running scripts to change local preference values in the target's routing table.

N-BGP keeps local a model of the target BGP speaker's state so that N-BGP can determine which outgoing updates are valid. This model of the state is represented as a table of received updates, referred to as $I_a$ in the safety specification for router $a$. Each entry in $I_a$ contains an IP address prefix, an AS path, and zero or more policy constraints. When the target BGP speaker $a$ receives an update, the N-BGP host adds it to $I_a$. If any policy rules match the update, then the corresponding actions are associated with the new entry in $I_a$ as constraints. For example, if an incoming update matches a policy rule with the action "pref=50," then any outgoing updates based on that update should have a local preference of 50. If an outgoing update violates the safety specification, N-BGP blocks the update or floods the security plane with a warning. If an outgoing update is valid under the safety specification but violates a policy rule, then a notification is sent to any peer to whom the violated rule is visible, but the update is not blocked.

The table of received updates contains all updates the target speaker has received since last initialized. Use of a proxy ESM inherently requires restarting the BGP speaker's connections to its peers, and thus it captures all updates as they are re-sent. In contrast, a sniffer ESM starts with incomplete knowledge of the monitored BGP speaker's state, because it has not seen prior announcements the router received. BGP route announcements are not periodically refreshed, so without some assistance, a proxy ESM's state might not converge to a complete view over time. However, restarting the BGP speakers' connections to its peers or retrieving its routing table through an administrative interface (e.g., "show ip bgp") suffices to bootstrap the N-BGP host's state.

In an AS with more than one BGP speaker, N-BGP captures control plane traffic among the speakers. This is needed, for example, because an update can arrive at one speaker and later be forwarded by another speaker that is monitored by a different ESM. The ESM for the latter speaker can validate the update only if it knows that it arrived elsewhere in the same AS. N-BGP can currently monitor the Interior Border Gateway Protocol (iBGP), which has the same safety properties as inter-domain BGP (eBGP) connections. N-BGP could be extended to support others, such as OSPF or IS-IS.

### *D. The Security Plane*

N-BGP hosts use the security plane to propagate origination certificates, queries, and warnings. Warnings

are flooded, while queries and origination certificates are unicast.

- Origination certificates attest that the origin of an update path actually owns the IP address range being advertised.
- Queries allow N-BGP hosts to monitor remotely the behavior of some routers that do not have an N-BGP host deployed locally.
- Warnings allow N-BGP hosts to react automatically to invalid behavior at remote BGP speakers.

The N-BGP security plane is an overlay network that mirrors the underlying BGP network whenever possible. That is, N-BGP hosts peer when the BGP speakers they monitor are peers. For incremental deployment, administrators can set up multi-hop tunnels to N-BGP hosts at non-peer sites, similar to eBGP tunnels in soBGP [23]. The tunnels allow all N-BGP hosts to form a single security plane even when not all ASes have deployed N-BGP.

One common pitfall in designing BGP security solutions is using routing to secure routing. Any protocol that depends on IP addresses for identification or on IP routing for communication is vulnerable, as attackers can spoof, modify, or drop security-protocol traffic. This is not a vulnerability for N-BGP warnings because they are normally sent over direct, one-hop connections that do not depend on routing. For queries, which are usually require traversing an intermediate AS, N-BGP prevents attacks in two ways. First, N-BGP uses trusted computing to prevent attacks on identity, such as a man-in-the-middle attack. N-BGP uses Nexus attestation certificates to create SSL channels between pairs of hosts; each host is assured that the other is running a valid copy of N-BGP. Here, SSL enforces the order, completeness, confidentiality, and integrity of security plane traffic, as well as resistance to replay attacks. Second, N-BGP sends warnings, queries, and periodic keep-alive messages over this channel. Because SSL preserves stream order, an attacker cannot block warnings or queries without also blocking the (encrypted) keep-alive messages, which allows the endpoints to detect the attack.

To construct virtual ESMs, N-BGP needs to know which ASes are secured and needs IP addresses for each N-BGP two hops away. Each N-BGP host keeps a table of remote hosts' IP addresses and AS numbers. When an N-BGP host is initialized, it establishes mutually authenticated SSL connections to all of its peers using the same attestation certificates described above. Each new host then floods its IP address and AS number to all other N-BGP hosts in the network. This flood enables the construction of the remote-host table.

*Invalid-route warnings:* An N-BGP host that detects an invalid update sends an *invalid-route warning*, indicating that other N-BGP hosts should take remedial action. N-BGP hosts receiving the warning forward it to their peers, effectively flooding the warning and preventing further dissemination of the invalid route. The flood follows the peering relationships described above. By flooding warnings, N-BGP hosts collectively form a distributed database of invalid updates to block or to remove from BGP speakers' routing tables.

Each N-BGP host 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 damp route flapping [26], the security plane can almost always disseminate an invalid-route warning before the invalid route affects BGP speakers beyond the first hop. We expect invalid updates from monitored BGP speakers to be uncommon, making the number of flooded warnings small and keeping maintenance traffic and storage requirements low.

To defend against malicious BGP speakers sending enough invalid routes to constitute a denial-of-service attack, N-BGP hosts blacklist any BGP speaker that generates more than a threshold number of invalid routes. Unlike the invalid-route database, manual intervention is required for a BGP speaker to be cleared from the malicious-speaker 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.

*Route-validity queries:* An invalid update sent by an unmonitored BGP speaker can be detected if the BGP speakers immediately preceding and following it in the forwarding path for that update are both monitored. In this case, one N-BGP host captures the update as it is sent to the unmonitored speaker, and another N-BGP captures the tampered update emitted by the unmonitored speaker. This arrangement is similar to constructing a virtual ESM, except that only two links (one in and one out) need to be monitored to secure any given update. The N-BGP host receiving an update from an unmonitored BGP speaker sends a *route-validity query* to the N-BGP host that earlier sent that update to the unmonitored speaker. The host receiving the query responds to confirm that it sent the update in question to the unmonitored BGP speaker now forwarding it. Figure 5 shows this exchange. An N-BGP host need only contact the N-BGP host for the last monitored BGP speaker, because that N-BGP host will have already verified the path up to and including itself.

The use of route-validity queries allows N-BGP hosts to detect whether or not any BGP speaker in a path could have forwarded an invalid update, provided no

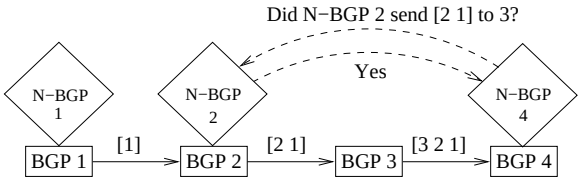


Fig. 5: N-BGP 4 sends a route-validity query to N-BGP 2 and receives a response. If BGP 3 received the update [2 1], then it was allowed to send the update [3 2 1].

two adjacent ASes on the path are unmonitored. An update with an AS_PATH containing adjacent unmonitored ASes cannot be verified, because it could be a wormhole attack between the unmodified ASes (see Section V-C). A path without adjacent unmonitored ASes is deemed *verifiable*. Paths containing only monitored ASes are trivially verifiable and do not require route-validity queries.

Route-validity queries, like multi-hop peering relationships, use end-to-end encryption. Thus, an attacker cannot modify or forge responses to route-validity queries. An attacker can block the query or the response, but the N-BGP host initiating the query can interpret the absence of a response as a failure.

*Origination certificates:* N-BGP implements the Grassroots PKI [25] protocol to prevent false-origination attacks. Grassroots lets administrators choose their own certificates for each block of IP addresses, rather than obtaining certificates from a centralized addressing authority. Each host supporting Grassroots trusts the first *origination certificate* it receives for any given address block; future announcements are compared against that. Thus, Grassroots provides *forward security* rather than absolute security. In cases where an attacker announced a certificate for an AS's address block before the AS itself did, the legitimate AS can obtain attestations from the owners of progressively larger supersets of the stolen block. In case of conflicts, each Grassroots host trusts the certificate with the longest attestation chain, for any given block.

Deploying Grassroots on top of N-BGP, rather than deploying Grassroots alone, has three advantages. First, N-BGP provides an external platform on which to execute the Grassroots protocol, rather than requiring software changes to BGP speakers themselves. Finally, the Nexus provides secure key-generation and storage interfaces that prevent even an inside attacker from obtaining certificates.

If Grassroots were implemented directly on BGP speakers, each speaker could attach an origination certificate to each update message it sends. N-BGP hosts that are sniffers cannot modify update messages. So instead, sniffer N-BGP hosts send one origination certificate message on the security plane after each update that their target speaker sends. Proxy N-BGP hosts reduce security plane traffic by appending the origination certificate to the corresponding update as an attribute, both when creating a new origination certificate and when receiving a separate update and origination certificate from another N-BGP host.

### *E. Preventing N-BGP Vulnerabilities*

One important aspect of any security protocol is that it not introduce more vulnerabilities than it prevents. To this end, we have worked to limit both the probability of compromising N-BGP and the ill effects if an N-BGP host is compromised.

N-BGP is small: just a few thousand lines of code. So unlike a router, N-BGP's trusted computing base may be small enough to audit. Additionally, N-BGP has limited network exposure—it accepts traffic only from the target speaker and its peers. Nexus secure storage and attestation prevent any attacks against N-BGP from corrupting N-BGP's on-disk program image or configuration.

In the unlikely event that an N-BGP host is compromised, we wish to limit the damage it can do. Specifically, we limit an N-BGP host to issuing warnings only about messages its own speaker forwarded and only to downstream hosts. In addition, we prevent the N-BGP host from flooding other N-BGP hosts with spurious warnings as follows.

To prevent an N-BGP host from issuing warnings for updates its target did not forward, N-BGP associates a nonce with each update and quotes the nonce in any warnings about that update. As with Grassroots, proxy N-BGP hosts append the nonce to each update, and sniffer N-BGP hosts send a separate nonce message over the security plane. Each N-BGP host generates a different nonce for each update for each peer, so that each instance of an update is unique. A compromised N-BGP host will only have access to the nonces of messages its target receives or sends directly. Hence, a compromised N-BGP host can only issue warnings about messages its target could have blocked.

N-BGP hosts avoid being inundated with spurious warnings from a compromised N-BGP host the same way they avoid too many warnings from a compromised BGP speaker: rate limiting. Just as a speaker will be effectively blacklisted after too many invalid updates, its N-BGP host will be blacklisted after too many warnings.

## V. Evaluation

We have implemented and measured a prototype of N-BGP. Our evaluation sought to answer three questions about the implementation:

- Does the safety specification detect invalid updates without generating spurious warnings for legal behavior?

- Can N-BGP on commodity hardware check updates as fast as routers might send them?
- How much benefit can N-BGP deliver for different degrees of partial deployment?

### A. Testing for False Positives

We tested N-BGP with traces from a major routers at transit ISPs, replayed to two different BGP implementations, as described below. In all cases, N-BGP is being run on a 2.66 GHz Core 2 processor and the Nexus operating system.

*National LambdaRail:* We obtained a routing table snapshot and six days of debugging logs from the National LambdaRail router in Denver. The snapshot contains the full routing table (obtained using "show ip bgp"); the debugging logs contain the full contents of all BGP messages sent and received. These traces reflect the local routing policy used by NLR, which we were not given. We used the NLR traces to test N-BGP as if we were monitoring the Denver router directly with an ESM deployed on site.

*RouteViews:* We used a BGP snapshot and one month of update traces from route-views2.oregon-ix.net [15]. The RouteViews snapshots contain the full routing table at a router, and the update traces contain all updates received by the router during a 15-minute interval. The RouteViews traces contain more data than the NLR traces: routes for all the prefixes in the Internet, a full month of events, and 46 peers. Since they contain only initial state and inputs—not outputs—we used the RouteViews traces as inputs for a Cisco 871 router, and we monitored that router's inputs and outputs. The router applied its default policies: no filtering and no update modifications.

*PLUTO:* We captured four weeks of data from the PLUTO sensors at planetlab1.arizona-gigapop.net and planetlab1.ipls.internet2.planet-lab.org [27]. PLUTO feeds contain updates sent by a backbone router to a PlanetLab host over a one-way BGP connection. The PLUTO feeds contain only updates, not the initial state of the router, and they contain only the updates the router forwarded to the PlanetLab host. This means the PLUTO feeds reflect only the best route the router has received. We sent both PLUTO feeds to a host running Zebra BGP software, configured with default policies, under Linux on a 2.2 GHz Athlon64 processor. We monitored the Zebra host with N-BGP.

In all three cases, N-BGP did not generate any spurious warnings. When we compromised the router implementations and injected invalid packets that advertised truncated paths or unauthorized prefixes, N-BGP correctly detected them in every case.

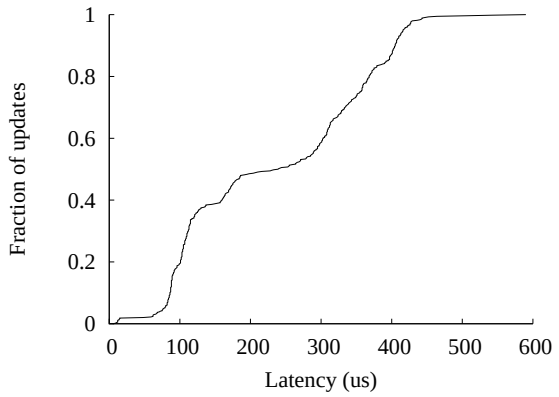


Fig. 6: The cumulative distribution of latencies for updates forwarded by N-BGP operating as a proxy ESM.

### B. N-BGP Host Performance

We measured the rate at which an N-BGP host—both proxy and sniffer deployments—can process updates received by a BGP speaker as 25,655$\pm$51 updates/sec (i.e., 39.0$\mu s$/update). The rate at which N-BGP can check messages sent by a BGP speaker against the base safety specification was measured at 8,893$\pm$15 messages/sec. Both figures average 30 trials. Checking policy rules in addition to the safety specification adds about 0.12$\mu s$ of processing time per message per policy rule. We obtained these figures by measuring the time N-BGP took to process 1,792,967 synthetic updates totaling 145.3 MB. The updates in the trace are based on a routing table snapshot from route-views.oregon-ix.net [15]. We produced one update per entry in the routing table—one for each prefix, for each of the router's peers—and then combined update messages with identical AS_PATHs and attributes into a single message.

Because normal BGP traffic on the Internet is roughly 1-2 updates per second per peer, N-BGP is clearly capable of keeping pace. The start-up burst when a new peer is first connected could present higher traffic rates, requiring a buffer large enough to contain the burst until it can be checked. A 150 MB sniffer buffer was large enough for our trace.

When operating as a proxy ESM, N-BGP must check and then forward each message it receives. Doing so adds latency to each update, consisting of input queueing time, processing time, and output queueing time. Recall that processing time is 39.0$\mu s$ for one update. Figure 6 shows the cumulative distribution of these latencies. Over 100,000 updates, the mean forwarding latency was 233$\mu s$, and the median was 239$\mu s$. Compared to network propagation time and delays added to suppress route flapping, the N-BGP proxy latency can be considered negligible.

N-BGP must store all updates that its BGP speaker has

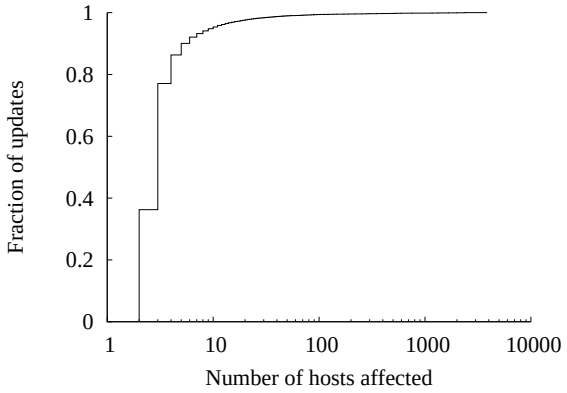


Fig. 7: The number of hosts affected by an invalid update before N-BGP warnings overtake it.

received but have not been withdrawn. We estimated the memory requirements for an N-BGP host monitoring a BGP speaker with varying numbers of peers by replaying portions of the RouteViews snapshot corresponding to selected peers. N-BGP memory usage ranged linearly from 27.5 MB for one peer to 163.5 MB for 45 peers. Hence, storing the updates from each additional peer requires about 3.1 MB of memory.

When N-BGP operates as a sniffer, detection of invalid updates can happen too late to prevent recipients from acting on those updates. The mean time for an N-BGP host to receive, check, and forward an invalid-route warning, averaged over 100,000 messages on an unloaded link (hence, little or no queueing delay), was $313 \pm 37 \mu s$, with a median of $312 \mu s$. In contrast, BGP speakers delay updates up to 30 seconds in order to damp route flapping [26]. Some implementations use a separate 30-second timer for each update they forward, while others, such as Zebra, send updates in a batch every 30 seconds. We modeled the update delay as a random quantity ranging from 0 to 30 seconds, corresponding to the behavior of Zebra. In our model, over 99% of all invalid updates were prevented by an invalid-route warning after a single hop. This model is pessimistic; some BGP implementations use an independent timer (up to 30 seconds) for each update, meaning that warnings will always reach these speakers before they can forward an invalid update. The number of BGP speakers affected by an invalid update depends on the out-degree of the BGP speaker sending it. We used the CAIDA AS topology [28] to simulate the race between warnings and invalid updates. As Figure 7 shows, 99% of the time, fewer than 50 speakers are affected. Because the warning arrives within a few milliseconds, only very short-term harm is done at the hosts the invalid update does reach.

As described in Section IV-D, N-BGP uses route-validity queries to check updates received from unmonitored BGP speakers. Over 100,000 queries, the average query round-trip time between two N-BGP hosts on an unloaded network was $319 \pm 52 \mu s$, and the median was 318 $\mu s$. The total query and response size is 212 bytes normally, expanding to about 4 KB when a key exchange is needed.

### C. Incremental Deployment Benefit and Adoptability

Owners of autonomous systems need incentives to deploy a BGP security solution. We expect administrators will weigh the costs of the defense against the security it provides. Because not every AS will deploy the defense at the same time, there must be tangible security benefits even when only a small number of sites have deployed the defense. Because the benefit increases as more people deploy it, the security benefit for the first few sites is the most important.

In this section, we present a model for the incremental security benefit in a routing protocol: the fraction of chosen paths that are verifiable for a given partial deployment. A *chosen path* is the path (in an update) that a BGP speaker selects to reach a particular definition. Normally, this choice is dictated by path length or by site-specific policy. We modify the BGP decision process to prefer the best path that can be verified, or the shortest path if no paths can be verified. A path is *verifiable* if the recipient of the path can confirm that no attacker could have modified the path as it was originated and forwarded.

We apply our model to four protocols: N-BGP, soBGP [23], S-BGP [22], and BIND [29]. soBGP, S-BGP, and BIND are described in detail in Section VI.

In N-BGP and in soBGP, a path is verifiable if both endpoints are secure—the origin is authenticated and the recipient is capable of verifying the path—and no two adjacent ASes within the path are non-secured. (We assume N-BGP uses virtual ESMs wherever possible.) The intuition for this definition is that each secured AS can vouch for itself and all direct links to and from itself. A path with two unsecured ASes in a row contains a link that no one can vouch for.

In S-BGP [22] and BIND [29], a path is verifiable if there is an unbroken chain of trust from the origin to the recipient. If any intermediate AS is unsecured, the chain of trust is broken.

We quantified the percentage of paths—assuming one path per pair of ASes in the network—that are verifiable using N-BGP, and we compared it to soBGP, S-BGP, and BIND. In all four protocols, the originating and the final AS must be secured for a client to establish trust in the route; thus, our analysis considers only AS pairs where both endpoints are secured. We reason that, if the ASes at the ends of a path want the path to be secure, then they will deploy an appropriate security protocol.

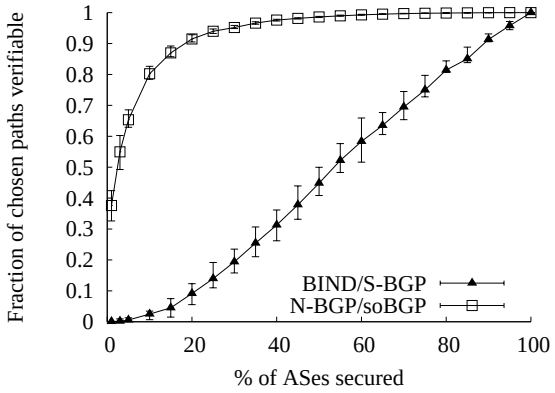


Fig. 8: The percentage of verifiable routes, if both endpoints are secured. Although the four protocols are considered separately, N-BGP and soBGP yield identical results, as do BIND and S-BGP.

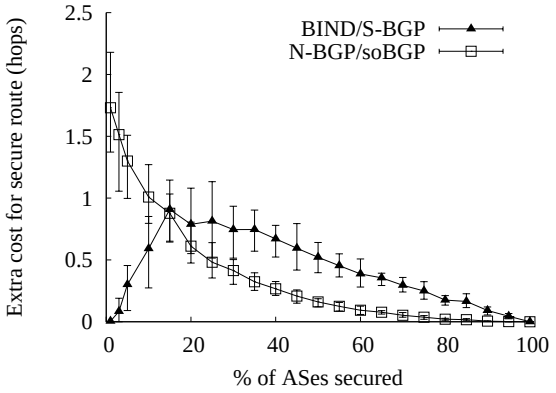


Fig. 9: The average difference between the shortest verifiable path and the overall shortest path, in hops.

We are analyzing their dependence on the deployment rate of the ASes in between.

Using the AS-level topology from the CAIDA AS Relationships Dataset [28], we enumerated all AS pairs and counted which pairs had at least one verifiable path between them, for fractions of secured ASes ranging from 5% to 100%. We do not know which ASes are likely to deploy a BGP security solution first. Chan et al. [30] considered scenarios where tier-1 ASes, governmental ASes, or university ASes deployed security solutions first. We instead use a random selection as a lower bound. For each deployment fraction $x$ between 5% and 100%, we ran 30 trials, each time selecting random ASes on which ESMs are deployed.

Figure 8 shows the fraction of endpoint pairs with at least one verifiable path between them as a function of deployment rate, with error bars indicating the 25th and 75th percentiles. An even higher percentage of traffic can be secured if high-degree, core ASes are selected instead of random ASes. Figure 9 shows the average added cost in hops, for each protocol, of choosing a verifiable path instead of the shortest path. The non-monotonic nature of the added cost for S-BGP and BIND was unexpected. The cost is small at low levels of deployment because the few paths that can be secured are between ASes in the core of the Internet, where many redundant paths of equal length are available.

Because they can tolerate one or more non-adjacent unsecured ASes, N-BGP and soBGP provide much better incremental benefit than those (i.e., S-BGP and BIND) that require all ASes on a path to be secured. Observe that 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 verifiable 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. N-BGP delivers incremental deployability equal to soBGP without requiring administrators to replace router hardware or disclose private peering relationships.

## VI. Related Work

Several existing systems share our goal of improving the security of BGP. One approach is to issue hop-by-hop certificates for each component of the route; a certificate chain vouches for the full route. S-BGP [22] and SPV [31] use this approach. S-BGP establishes two certificate hierarchies: one to identify ASes and BGP speakers, 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.

SPV improves upon the performance of S-BGP by using one-time signatures, which are based on symmetric encryption. SPV improves the security and incremental deployability of update forwarding by letting an SPV-capable AS sign an update on behalf of earlier ASes on the forwarding path that don't deploy SPV.

Pretty Secure BGP (psBGP) [24] refines S-BGP's approach by having 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 [29] uses attestation to guarantee updates a BGP speaker sends are valid if that BGP speaker receives valid input and runs valid code. BIND, like N-BGP, uses a TPM to enforce code integrity. However, BIND mandates a single implementation of the update-forwarding code, requires replacing all existing BGP routers with trusted hardware, and provides no guarantees for routes with even a single non-participating router.

Secure Origin BGP (soBGP) [23] uses a link-state approach to validate routes. Each soBGP speaker distributes a peer list to all other speakers. Each speaker stores an AS-level map of the Internet based on peer lists it receives, enabling it to confirm the feasibility of a path even if that path has non-soBGP speakers on it. However, ASes deploying soBGP must publicly announce peering and policy information, which is unacceptable in certain business settings.

Whisper [17] detects path truncation attacks by comparing multiple updates for a given prefix. In Whisper, BGP speakers append a random value to each update they send originating a route. Each BGP speaker forwarding an update replaces the value 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. Whisper requires replacing router software and must be deployed widely to be effective.

Routing registries store peering and policy information to help routers confirm the feasibility of routes they receive. One routing registry is Internet Routing Validation (IRV) [32], in which each AS maintains the portion of the registry containing its peering and policy information. 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 N-BGP, IRV is deployed on separate hosts rather than on BGP speakers. However, IRV hosts do not leverage trusted hardware to guard against compromises or misconfiguration, and they must be manually synchronized to the router's policy and peering arrangements.

Three additional control-plane modifications are complementary to N-BGP. The Prefix Hijack Alert System [33] notifies operators if the BGP origin for a prefix they own changes. Pretty Good BGP [34] avoids using routes that differ from recent routes for the same prefix. Routing Control Platform [35] is a logically centralized system for making routing decisions in large ASes, which would simplify for N-BGP the task of monitoring the speakers in a large AS.

Some recent systems take an approach complementary to N-BGP, by detecting anomalies in the data plane rather than in the control plane. Listen [17] monitors data-plane traffic and signals an alarm if a large amount of traffic to any given prefix is rejected. Both stealth probes [18] and secure traceroute [19] inject additional probe traffic into the data plane, detecting any sites that incorrectly forward the probes. These efforts are reactive; they cannot prevent invalid configurations or updates, but they may discover problems that control plane protection systems miss.

Recent work by Chan et al. [30] quantified the incentives and incremental benefits of several approaches to securing BGP. They define *adoptability*, essentially a measure of the strength of guarantees a security protocol can provide. Adoptability considers the damage a single, randomly located attacker can do on a given topology. In contrast, our path verifiability metric considers the ability to route around an unspecified number of attackers by choosing a verifiable path out of a set of alternatives.

Our ESMs draw inspiration from existing network components, like intrusion detection systems [6] and firewalls. A sniffer ESM, like an IDS, monitors all network links into a protected host. While an IDS generates at least some of its alerts based on heuristics, an ESM generates alerts based on strict rules, eliminating false positives. A proxy ESM, like a firewall, filters all packets into and out of a protected host. However, both firewalls and most IDSes operate solely for the security of the local site. An ESM joins a security plane, allowing it to monitor remote sites. Trusted computing enables an ESM to assure remote principals of the rules it enforces.

PeerReview [1] adds a component that monitors for Byzantine failures all messages into and out of a local replica in a replicated system. PeerReview's monitors communicate with each other out of band, as ESMs do. However, PeerReview works only for deterministic, replicated systems, while an ESM's trusted platform and safety specification enable it to monitor individual targets with any implementation and configuration resulting in valid network behavior. Finally, PeerReview depends on strong identities (i.e. a public key infrastructure), while ESM bootstraps trust from a safety specification enforced by unforgeable hardware.

## VII. Conclusions

The ready availability of trusted computing hardware affords an easily exploited opportunity to improve the security of network protocols. For deployed systems where replacing or upgrading existing hosts is not feasible, we propose the use of ESMs, which treat existing hosts as black boxes and confirm, based on inputs and outputs, that they correctly implement the desired protocol. This paper instantiates our proposed approach as N-BGP, an ESM running under the Nexus operating system and TPM-enabled hardware. Although this paper focuses on BGP, we believe that ESMs are applicable to a wide variety of protocols, a direction we are currently investigating.

We would like to thank the National LambdaRail BGP administrators, especially Brent Sweeny, for gathering detailed BGP traces for us. We are also grateful to Adrian Perrig and Nick Feamster for feedback on early drafts of this work.

## APPENDIX: PATH AGGREGATION

The rules governing how AS_PATHs are combined when BGP speakers perform aggregation are described in the BGP specification [14, §9.2.2.2]. They are complex and are more permissive than might be expected. We have implemented them in N-BGP, and we present them formally below.

An AS_PATH is represented in each update as a list of $\langle AS_i, type_i \rangle$ pairs, where $AS_i$ is an AS number and $type_i$ is either SEQ (sequence) or SET. Thus, each AS_PATH can be described in terms of two sets and two relations:

$\mathcal{N}^Q$ : the set of all pairs of type SEQ
$\mathcal{N}^T$ : the set of all pairs of type SET
$<^{AS}$ : a relation describing an order for $\mathcal{N}^Q \cup \mathcal{N}^T$, where $a <^{AS} b$ means that pair $a$ precedes pair $b$ in the AS_PATH.
$<^T$ : a relation describing an order for $\mathcal{N}^T$, where $a <^T b$ means that $a$ and $b$ are both of type SET, and $a$ appears before $b$ in the AS_PATH.

Let $\phi = \langle \mathcal{N}^Q, \mathcal{N}^T, <^{AS}, <^T \rangle$ and $\phi_i = \langle \mathcal{N}_i^Q, \mathcal{N}_i^T, <_i^{AS}, <_i^T \rangle$; then

$\phi = Agg(\phi_1, \phi_2, \ldots, \phi_n)$ if all of the following conditions hold:

1) SEQ pairs in $\phi$ also appear in all input AS_PATHs $\phi_1, \ldots, \phi_n$.
$$\mathcal{N}^Q \subseteq \bigcap_i \mathcal{N}_i^Q$$

2) SET pairs in $\phi$ appear in at least some input AS_PATH $\phi_i$.
$$\mathcal{N}^T \subseteq \bigcup_i \mathcal{N}_i^T$$

3) If SEQ pair $x$ precedes $y$ in $\phi$, then $x$ precedes $y$ in every input AS_PATH $\phi_i$ that includes $y$.
$$\left[ x \in \mathcal{N}^Q \wedge x <^{AS} y \wedge y \in \left( \mathcal{N}_i^Q \cup \mathcal{N}_i^T \right) \right] \Rightarrow x <_i^{AS} y$$

4) $\phi$ has no repeated SET pairs, and any repeated SEQ pairs are adjacent.
$$a \in \mathcal{N}^T \Rightarrow \neg \left( a <^T a \right)$$
$$\left( a \in \mathcal{N}^Q \wedge a <^{AS} b \wedge b <^{AS} a \right) \Rightarrow \left( b \in \mathcal{N}^Q \wedge a = b \right)$$

5) Each pair in any input AS_PATH $\phi_i$ appears in $\phi$.
$$\bigcup_i \left( \mathcal{N}_i^T \cup \mathcal{N}_i^Q \right) \subseteq \left( \mathcal{N}^T \cup \mathcal{N}^Q \right)$$

N-BGP checks rules 1–3. Rule 4 is not checked because violations can be detected by any recipient, and N-BGP's focus is rules that recipients cannot check. Rule 5 prevents a router from truncating AS_PATHs during aggregation. Rule 5 is not part of the BGP specification, but we believe it is mandatory for path integrity. Imposing this rule prevents routers from truncating an AS_PATH during aggregation, a minor optimization that is rarely employed. As far as we can tell, S-BGP [22] and soBGP [23] also disallow this optimization because, without additional information, it is indistinguishable from a path truncation attack.

## REFERENCES


[1] A. Haeberlen, P. Kouznetsov, and P. Druschel, "PeerReview: Practical accountability for distributed systems," in *Proc. SOSP*, Stevenson, WA, Oct. 2007.

[2] L. Censier and A. M. Recoque, "Bi-processor data handling system including automatic control of exchanges with external equipment and automatically activated maintenance operation," US Patent 4,012,717. Filed: Apr 23, 1973. Issued: Mar 15, 1977.

[3] D. B. Johnson, S. Kenoyer, M. S. Myers, and S. Nilsson, "Apparatus for on-line checking and reconfiguration of integrated circuit chips," US Patent 4,792,955. Filed: Aug 21, 1986. Issued: Dec 20, 1988.

[4] "Cisco IOS XR Software," http://www.cisco.com/en/US/products/ps5845/, May 2004.

[5] J. C. Mogul, "Simple and flexible datagram access controls for Unix-based gateways," in *Proc. USENIX Summer Conference*, Baltimore, MD, Jun. 1989, pp. 203–222.

[6] D. E. Denning, "An intrusion-detection model," *IEEE Transactions on Software Engineering*, vol. 13, no. 2, pp. 222–232, Feb. 1987.

[7] S. A. Misel, "Wow, AS7007!" http://www.merit.edu/mail.archives/nanog/1997-04/msg00340, Apr. 1997.

[8] A. Ramachandran and N. Feamster, "Understanding the network-level behavior of spammers," in *Proc. ACM SIGCOMM*, Pisa, Italy, Sep. 2006.

[9] "Trusted Computing Group," http://www.trustedcomputinggroup.org/.

[10] 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.

[11] M. Gasser, A. Goldstein, C. Kaufman, and B. Lampson, "The digital distributed system security architecture," in *Proc. National Computer Security Conference*, 1989.

[12] 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.

[13] A. Heffernan, "Protection of BGP sessions via the TCP MD5 signature option," Request for Comments RFC-2385, Aug. 1998.

[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-views2.oregon-ix.net snapshot," http://archive.routeviews.org/bgpdata/, Feb. 2008.

[16] M. Caesar and J. Rexford, "BGP routing policies in ISP networks," *IEEE Network Magazine, special issue on interdomain routing*, pp. 5–11, November/December 2005.

[17] 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.

[18] I. Avramopoulos and J. Rexford, "Stealth probing: Efficient data-plane security for IP routing," in *Proc. of USENIX Annual Technical Conference*, Boston, MA, Jun. 2006.

[19] V. N. Padmanabhan and D. R. Simon, "Secure traceroute to detect faulty or malicious routing," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 1, pp. 77–82, 2003.

[20] C. Alaettinoglu, C. Villamizar, E. Gerich, D. Kessens, D. Meyer, T. Bates, D. Karrenberg, and M. Terpstra, "Routing policy specification language (RPSL)," Request for Comments RFC-2622, Jun. 1999.

[21] "Internet Routing Registry Toolset Project," http://www.isc.org/sw/IRRToolSet/, Feb. 2007.

[22] S. Kent, C. Lynn, and K. Seo, "Secure Border Gateway Protocol (S-BGP)," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 4, pp. 582–592, Apr. 2000.

[23] J. Ng, "Extensions to BGP to support Secure Origin BGP (soBGP)," Internet Draft, Apr. 2004.

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

[25] Y.-C. Hu, D. McGrew, A. Perrig, B. Weis, and D. Wendlandt, "(R)Evolutionary bootstrapping of a global PKI for secure BGP," in *Proc. HotNets*, Irvine, CA, Nov. 2006.
[26] C. Villamizar, R. Chandra, and R. Govindan, "BGP route flap damping," Request for Comments RFC-2439, Nov. 1998.
[27] A. Nakao, L. Peterson, and A. Bavier, "A routing underlay for overlay networks," in *Proc. ACM SIGCOMM*, Aug. 2003.
[28] "The CAIDA AS relationships dataset, June 26th, 2006." http://www.caida.org/data/active/as-relationships/.
[29] E. Shi, A. Perrig, and L. van Doorn, "BIND: A fine-grained attestation service for secure distributed systems," in *Proc. IEEE Symposium on Security and Privacy*, Oakland, CA, May 2005.
[30] H. Chan, D. Dash, A. Perrig, and H. Zhang, "Modeling adoptability of secure BGP protocols," in *Proc. ACM SIGCOMM*, Pisa, Italy, Sep. 2006.
[31] Y.-C. Hu, A. Perrig, and M. Sirbu, "SPV: Secure path vector routing for securing BGP," in *Proc. ACM SIGCOMM*, Portland, OR, Sep. 2004.
[32] 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.
[33] M. Lad, D. Massey, D. Pei, Y. Wu, B. Zhang, and L. Zhang, "PHAS: A prefix hijack alert system," in *Proc. of USENIX Security Symposium*, Vancouver, BC, Canada, Aug. 2006.
[34] J. Karlin, S. Forrest, and J. Rexford, "Pretty Good BGP: Improving BGP by cautiously selecting routes," in *Proc. IEEE ICNP*, Santa Barbara, CA, Nov. 2006.
[35] M. Caesar, D. Caldwell, N. Feamster, J. Rexford, A. Shaikh, and J. van der Merwe, "Design and implementation of a routing control platform," in *Proc. NSDI*, Boston, MA, May 2005.
[36] 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), Sep. 2006.

**Patrick Reynolds** Patrick received his PhD. from the Computer Science Department at Duke University. He is currently a postdoctoral research associate at Cornell University. He is a member of the Nexus project, working on trusted operating systems and distributed systems.

**Oliver Kennedy** Oliver Kennedy is a Doctoral Student at the Computer Science Department at Cornell University. He received his Bachelor's degree from New York University in 2005.

**Emin Gün Sirer** Emin Gun Sirer is an Associate Professor at the Computer Science Department at Cornell University. He received his Ph.D. from the University of Washington in 2002.

**Fred B. Schneider** Fred B. Schneider has been on the faculty at Cornell since 1978. He is also a Professor-at-large at the University of Tromsø (Norway). Schneider is a Fellow of AAAS and ACM, and in 2003 received a D.Sci *honoris causa* from University of Newcastle-upon-Tyne.