Implications of Path Stability on Efficient Authentication in Interdomain Routing

William Aiello, Kevin Butler, and Patrick McDaniel
aiello@cs.ubc.ca, {butler, mcdaniel}@cse.psu.edu

***Abstract*— Interdomain routing is implemented on the Internet through the Border Gateway Protocol (BGP). Many approaches have been proposed to mitigate or solve the many problems of BGP security; yet, none of the proposed solutions have been widely deployed. The lack of adoption is largely caused by a failure to find an acceptable balance between deployability, cost, and security. In this paper, we study one aspect of the BGP security puzzle: path validation. We develop a formal model of path authentication in BGP. We define and prove the security of a range of novel and efficient solutions under this model. We further analyze the security relevant stability of paths in the Internet and profile resource consumption of the proposed constructions via trace-based simulations. Our constructions can reduce signature validation costs by as much as 97.3% over existing proposals while requiring nominal storage resources. We conclude by considering how our solution can be incrementally deployed in the Internet.**

I. Introduction

The Border Gateway Protocol (BGP) [33], [32] is the dominant interdomain routing protocol on the Internet. BGP establishes and maintains associations between IP address *prefixes* [30] and source specific paths to the autonomous systems (AS) in which they reside. Each AS selects the best paths based on the advertised paths and routing policy.

Many analyses have highlighted concerns about the Internet's vulnerability to attacks on the BGP protocol [29], [36], [3], [23], [27], [6]. Among the many identified vulnerabilities, few are as potentially dangerous as those which attack routing path selection. Because of a lack of security services, adversaries are free to subvert the Internet interdomain routing infrastructure and through it, manipulate the underlying IP traffic. For example, addresses can be made unavailable (by mounting a denial of service attack), and traffic can be rerouted through malicious networks and monitored (an attack on traffic confidentiality), or directly altered (an attack on integrity).

Many approaches have been proposed to mitigate or solve the problems of BGP security [36], [17], [25], [8], [9], [18], [38], [37], [40]. Yet, none of the proposed solutions have been widely deployed. The lack of adoption is largely caused by a failure to find an acceptable balance between cost and security. For example, the S-BGP protocol offers one of the strongest security models proposed to date. S-BGP authenticates most routing artifacts (e.g., prefix and path advertisements, withdrawals, etc.) using asymmetric cryptography. However, the computational and storage costs of performing strong S-BGP style authentication are viewed to be prohibitive in many environments [8], [10], [40]. Furthermore, Nicol et al. showed that under a set of timing and cost assumptions, such costs can be measured on a global scale [26], where S-BGP nearly doubled path convergence time.

We argue that the limitations of proposed architectures are not due to flawed security models, but rather a result of the technologies to which we are bound. The cost of implementing strong protection is often too high for global environments. Weaker models are often feasible, but fail to adequately address important vulnerabilities. Hence, BGP security exhibits an oft-observed theme in security: while the community largely understands the technical requirements, no concrete solution has yet been found that finds an acceptable balance between security and cost.

This paper considers how the operational characteristics of BGP can be exploited to close the security infrastructure cost/security model gap. The central observation driving this work is that the vast majority of ASes offer few distinct paths for a prefix, and that those paths are largely static. We confirm this through a study of path stability. We study the 40 Route Views listening points [22], and found that in the average case, less than 2% of prefixes were advertised using more than 10 paths, and less than 0.06% were advertised with more than 20 paths during a single month.

The observed limited diversity and high stability of BGP paths allows us to explore a range of efficient cryptographic structures for path authentication. We develop a formal model of path authentication in BGP. We define and prove the security of our novel and efficient solutions under this model. Our authentication proof systems indicate the set of paths that an AS will announce during its lifetime. Associated with each path is a collection of *succinct validation proofs*, represented as *tokens* and released by the AS as evidence of its authenticity. Token validation is amortized over all proofs associated with that prefix. We define several constructions appropriate for path validation and consider a range of amortization strategies. We compare the cost of our solutions against

other solutions for BGP via trace-based simulation. Our simulations demonstrate that our techniques reduce the cost of validation by 97.3% over proposed solutions. Schemes such as SPV [10] and some proposed S-BGP optimizations amoritze costs in orthogonal ways to our solutions, and incorporating them could lead to even greater reductions in computational costs. We begin in the following section by outlining the operation and security requirements of BGP.

II. Interdomain Routing

BGP is charged with two central functions: the mapping of address prefixes (e.g., `192.168.0.0/16`) onto the ASes that own them, and the construction of source specific paths to each reachable prefix. The interdomain routing topology is defined by physical links between adjacent ASes. Each AS *originates* the prefixes associated with a network by identifying and enumerating them in an UPDATE message sent to its neighbors (adjacent ASes). Received announcements are recursively augmented with local AS numbers [11] and propagated, AS by AS, throughout the topology. To simplify, the path (also called a *route*) is the source specific vector of ASes used to forward traffic to the origin. Note that an AS may receive many paths for a single prefix. The AS identifies the "best" path using the *path selection algorithm*. The selection algorithm determines the best route by evaluating path length, policy, and other factors. Only the selected best path is propagated. IP traffic is routed, hop-by-hop, based on the best path known by the AS. Figure 1 illustrates route advertisement and path selection.

Which route represents the best path is re-evaluated each time a new route for a prefix is received. Suppression of non-best routes prevents undesirable routes from polluting the larger Internet, and is a key ingredient to the scalability of BGP. Recursive propagation of best routes ensures that every AS on the Internet acquires a route for every reachable prefix. A route is *withdrawn* when the AS discovers that the prefix is no longer reachable.

The preceding refers to *eBGP*, the protocol used for AS to AS communication. There may be many routers *speaking* eBGP within a single AS. The iBGP protocol is used within an AS to coordinate across eBGP speaking routers. iBGP security is outside the scope of this paper.

A. BGP Vulnerabilities

The ubiquity of BGP is also one of its greatest weaknesses. The continuous computation at each AS is the result of asynchronously distributed data from many peers across a huge geographic area. The complexity and size of the computation affords an adversary many opportunities to monitor, disrupt, or manipulate the process.

The Routing Protocol Security (rpsec) working group of the IETF exists to explore the problems and solutions

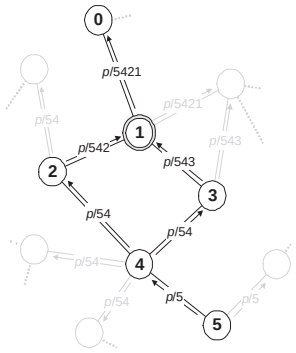


Fig. 1. BGP Path discovery - AS5 originates the prefix p by announcing it to its neighbors (e.g., AS4). AS4 further propagates the prefix to its neighbors AS2 and AS3 after prepending its AS number to the prefix. AS1 receives routes from AS2 and AS3, and selects the best route (arbitrarily [542]), which is then propagated further (to AS0 and others).

for routing security. The group captured the possible impact of routing vulnerabilities by explicitly postulating a universe of possible consequences [4]. Traffic congestion, black-holing, routing loops, slowed or prevented convergence, instability, traffic eavesdropping, network partitioning, and increased delay were deemed the most compelling consequences. The group's analysis led to a statement of general routing security requirements [31], and specifically to requirements for BGP security [7].

BGP security issues are often classified by the three broad categories of data exchanged [27], [6]: signaling, origins, and paths. Attacks on BGP signaling frustrate the session by incorrectly reporting errors, masquerading as other entities, or by consuming the victim's resources [23].

Validation of prefix ownership is essential to secure BGP. If not provided, an adversary can *hijack* entire networks by simply advertising the prefixes associated with them. Originally studied by Kent et al. [17], [35], an origin authentication (OA) service validates that an AS has the right to be the origin of a prefix. In a later work, Aiello et al. extended the study of OA by considering the semantics and efficient cryptographic constructions of origin authentication [2]. Principally, they explored formal semantics of the use and delegation of the IP address space. The set of all delegations between ICANN [12], registries, and organizations is modeled as a delegation hierarchy. Recently, Tan et al. suggested a alternative low cost, but weak form of origin authentication in which all BGP neighbors police and attest to the validity of the prefixes that an AS originates [40]. However, this is limited, as colluding ASes can forge origin information.

This paper investigates *path authentication*. Hu et al. identified the following classes of path attacks [10]:

1) *path forgery* - the adversary may attempt to forge paths in order to influence packet routing.
2) *path modification* - an adversary may add, remove,

or alter data in the path or policy.

3) *denial of service* - an adversary consumes a victim's resources by sending spurious routes.
4) *worm-holing* - colluding adversaries create illusory AS to AS connectivity.

Note that the first two classes are attacks, whereas the second two could be more accurately classified as consequences. Moreover, *worm-holing* is less of an attack on paths, but more of an attack on the topology. The false topology generated can be used to introduce incorrect paths, even if the path validation approach is perfectly implemented and deployed. With the exception of soBGP (see next section), few security proposals address worm-holing, as it requires validation of BGP peering.

If an adversary can forge or modify routes, then it can *black-hole* traffic routed to it. To accomplish this, the adversary announces a highly desirable route that is incident to the path, e.g., by advertising a very short path. Traffic flowing to that prefix will be routed to the adversary and filtered. If the adversary wants to destabilize the network while remaining relatively clandestine, it can randomly drop a percentage of the traffic (called *grey-holing*). Note that it takes few drops to vastly reduce the throughput between the victim and the destination: each drop causes the congestion control algorithm to aggressively throttle traffic [13]. Connection recovery is slow, and the attacker gains advantage with little effort [42]. Paths may also be manipulated to route traffic through malicious ASes for monitoring [6]. That is, if an adversary can redirect traffic (as above), then it can monitor, record, or even modify that traffic as it transits its network.

There are many ways to mount denial of service attacks within BGP, many of them quite subtle [23]. A router can simply black-hole routing announcements. Wherever the adversary is the only path to the black-holed route, the prefix will not be available. A malicious AS can also announce and withdraw routes at will. If the associated routes are popular (or are artificially so through forged policy or paths or worm-holing), then they will propagate to many ASes. Likewise, rapid withdrawals and announcements, known as route-flapping, can quickly consume a victim's resources. Such flapping reduces stability by disrupting packet forwarding. BGP uses route damping [39] to mitigate the affects of flapping routes. However, Wu et al. discovered that flap damping can be manipulated to indefinitely black-hole targeted prefixes [41].

B. BGP Security

Interdomain routing security has been studied for some time [29], [36], but comprehensive and efficient solutions remain elusive. The following considers how several of these efforts address path security.

Possibly the most comprehensive solution advanced to date, the Secure Border Gateway Protocol (S-BGP) [17], [16], [35] uses a public key infrastructure to support the authentication of routing artifacts. The S-BGP PKI maintains certificates for each AS and S-BGP-speaking router. Every router includes a *route attestation* with each advertisement. The route attestation is a signed statement of the AS identity, the paths, the prefix and the AS to which the announcement is directed. The S-BGP speaker also includes the route attestation of the route on which the advertisement is based. This prevents an adversary from adding or removing ASes from the path. While the authors of S-BGP have introduced a number of optimizations that reduce resource consumption [15], the costs associated with it are viewed as limiting factor in many environments [8], [10], [40]. For example, Nicol et al. showed that, under a set of timing and cost assumptions, such costs can double the path convergence time [26]. However, Nicol et al. did not model optimizations reported in [15]. It is not clear if and how the optimization would affect convergence times. While some argue that co-processors and protocol optimizations may make computation feasible, storage remains a major problem. Kent estimates that S-BGP will require an additional 30-35 megabytes of storage per peer [14]. Such costs are manageable in routers with a few peers, but are problematic in large ISPs or exchanges. However, Kent further argues that there are asymmetric configurations where only a few routes are accepted (as in customer/ISP peering), and hence these situations would require fewer resources.

Partially in deference to the costs associated with more comprehensive solutions, the soBGP and IRV projects sought other means of addressing BGP security. The soBGP [25] protocol uses a topology database to validate that advertised paths are consistent with the signed statements of connectivity between ASes. While this approach provides a limited security guarantee, it is effective in preventing a wide array of path hijacking and worm-holing attacks. However, soBGP does *not* provide path authentication, but simply implements a mechanism for detecting routes that are inconsistent with the authenticated topology. Philosophically similar to the earlier routing registry projects [20], the Interdomain Routing Validation (IRV) [8] project was motivated by the observation that any solution requiring a change to BGP was likely to be adopted slowly, if at all. IRV servers use an out-of-band (e.g., external to BGP sessions) protocol to exchange validation information. IRV is reliant on the routing infrastructure to extract and exchange routing data. Hence, unless some other infrastructure is put in place (e.g., static routes), the system is unable to function when connectivity is not available.

Several proposals have sought efficient constructions for BGP security. Hu et al. introduced the concept of cu-

mulative authentication for securing route advertisements in path vector protocols [9]. They use the TESLA timed key release authentication to validate announcements using low cost symmetric key cryptography. TESLA is limited in that it requires tight time bounds on message transmission, which is in conflict with protocols built on asynchronous propagation protocols such as BGP. More recently, Hu et al. introduced the Secure Path Vector Protocol (SPV) [10], which also seeks to implement BGP path security using low cost cryptography. SPV creates cascading authenticators over many (low cost) one time signature structures. The security of SPV is in some cases based on probabilistic arguments. In particular, the authors argue that reduced exposure (in time) to forgery vulnerabilities is sufficient to mitigate attacks. While this may be acceptable in constrained environments, it is unclear whether such arguments apply to the Internet.

The Whisper protocol [37] uses a mechanism that detects inconsistencies in received routes using RSA-style [34] cryptographic operations. To simplify, any conflicts between routes received from multiple peers emanating from the same original advertisement is detectable. In the same work, Subramanian et al. introduce the Listen protocol, which does not provide comprehensive path authentication, but simply detects a class of attack.

III. Path Validation Constructions

In this section, we define what we mean by attestations and route attestation tags and formally state their security properties. Route attestation tags are very similar to route attestations as defined in [17], [16], [15] with several minor differences highlighted in this section. We assume that the reader has a general familiarity with cryptographic primitives such as hash chains, hash tables and digital signatures. These constructions are explored in greater detail later in the section. We begin in the following subsection with a brief overview of our approach to path authentication, and continue with a formal description of its semantics, operation, and security.

A. Overview

The following briefly describes the use of our base construction for path authentication:

1) An AS receiving path announcements from a given prefix creates a hash tree [21], where the leaves of the tree comprise all unique paths to the prefix received by the AS.
2) For each leaf of the tree (corresponding to a unique path), a hash chain is generated. The leaf holds the anchor of that hash chain.
3) The root of the hash tree is signed with the private key of the AS.
4) An *authentication token* is the t^{th} value of the hash chain associated with the current best path at time t, plus the signed root and sibling values of the leaf and its parents, recursively computed up the height of the hash tree. At any given time, the current token is released to peer ASes through BGP UPDATE messages. A new token is released only if the AS chooses another "best path" to the destination or a policy defined refresh threshold is reached. These changes are reflected in new UPDATE messages.
5) An AS receiving the advertisement will receive the authentication token and validate the received proof. If the proof is valid, the receiving AS knows the path is not a forgery, and can use it freely.

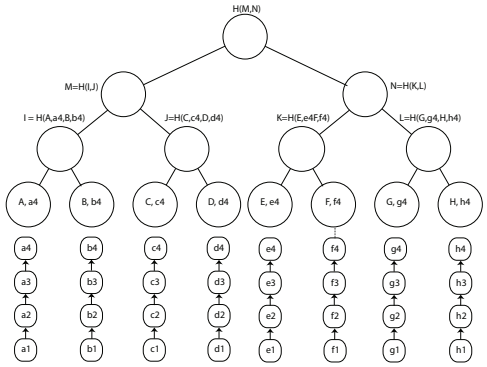


Fig. 2. A Merkle hash for the set [A..H] [A..H], augmented with timestamp tokens. The values a1 through h1 are randomized, and form the anchors of hash chains generated from their values.

The security of this scheme follows from the definition of the cryptographic primitives (hash functions, hash trees, hash chains, and digital signatures) and their use as prescribed in the following subsections. A diagram of this scheme is shown in figure 2.

Our constructions bear a superficial resemblance to approaches such as SPV [10] and S-BGP [17]. SPV uses tree-based one-time signatures to protect the integrity of an AS path and ultimately amortize the cost of signing, while S-BGP does not directly amortize costs. In contrast, we exploit existing BGP *reference locality* by amortizing over frequently occurring paths. Our scheme is thus simpler and, where reference locality exists, frequently more efficient than the alternatives (we explore this latter claim further in Section V).

We consider a number of alternate constructions of our validation approach; in each case, hash chains are generated for each path in the same manner, but the nature of the hash tree structure holding the set-membership proofs will change. In the *prefix* scheme (described above), for each prefix whose route an AS historically advertises, the AS constructs a tree where the leaves represent all of the distinct paths it historically advertises for that prefix. In the *origin* scheme (described above), for each origin AS amongst the routes that the AS historically advertises,

the AS constructs a tree where the leaves represent all of the distinct (prefix,paths) pairs it historically advertises with the given origin AS. The *all* AS construction simply creates a single tree where the leaves represent all the paths the AS historically advertises.

B. Attestations and Route Attestation Tags

In previous works, route attestations were defined as a sequence of statements signed by routers using public key signatures. Our route attestation tags are also a sequence of attestations by routers, but here we allow the attestations to be more general public key authentication methods. In particular, an attestation may be either a signature or a *set-membership proof*. Set-membership proofs are essentially signatures of Merkle hash trees [21].

We first state several definitions that will be used below. We then state formally the definition of a set-membership proof, its definition of security, and several examples. Next, we define a *route attestation tag* or RAT as a sequence of attestations. Finally, we describe our scheme, as well as the schemes of [17] and [26] as instantiations of the general set-up. These descriptions are used in the subsequent sections, where the performance tradeoffs of these schemes are that empirically analyzed. For space considerations, we direct readers interested in a formal definition of security for a RAT to our technical report [1], where we reduce the security of a RAT to the security of the attestations used in the RAT. While this does not appear to be surprising, the adversary used as the basis of the definition of security is quite powerful. The technical report also contains details on incremental deployability using these path authentication schemes.

The formalization below is general enough to capture not only our proposed schemes but the S-BGP scheme and the scheme of Nicol et al. as well. At the same time, it is specific enough to allow for a precise definition of existential forgery of a route announcement and a reduction to the security of standard cryptographic primitives.

C. Notation

Let $\mathcal{ASN} = \{1, \ldots, 2^{16} - 1\}$ be the set of all unique identifiers for an Autonomous System. These are the so-called Autonomous System Numbers. An *AS path* is a sequence, possibly empty, of AS numbers. Given a path $p \in \mathcal{ASN}^*$, let p_i, $i \geq 1$, denote the ith element in the sequence. Furthermore, let $p_i^{\leq}$, $i \geq 1$, denote the subsequence of the first i elements of p. For example, if $p = (23, 1708, 229)$, then $p_2 = 1708$ and $p_2^{\leq} = (23, 1708)$.

In BGP, AS padding is allowed. That is, a legitimate AS path can have a sequence of consecutive values that are identical. This is equivalent to saying that BGP allows paths with self loops (but not other kinds of loops). We call a path *almost simple* if it has no loops except for self loops.

Let $G = (\mathcal{ASN}, \mathcal{E})$ denote the AS graph. A pair of AS numbers (a_1, a_2) is in $\mathcal{E}$ if AS a_1 and AS a_2 have a service level agreement (SLA) to be eBGP neighbors. Note that $\mathcal{E}$ does not capture which pairs of ASes have active eBGP sessions between routers at the current time. That is, if (a_1, a_2) is in $\mathcal{E}$, there may be no current eBGP session between a router in a_1 and a router in a_2. Nonetheless, the edge in the graph G is maintained as long as the neighbors have an eBGP SLA. A path p is denoted *topology respecting* if every edge in the path is also an edge in G.

A route is a pair consisting of an address block and an AS path. Given an address block b and a path $p = (a_1, a_2, \ldots, a_k)$, the route r for b and p is written as $r = (b, p)$ or as $r = (b; a_1, a_2, \cdots, a_k)$. Considering the latter as a sequence, r_i, $i \geq 0$, denotes the ith element of the sequence and $r_i^{\leq}$ denotes the subsequence of r from r_0 to r_i, inclusive. Note that $r_0 = b$, and that for $i \geq 1$, $r_i = p_i$, and $r_i^{\leq} = (b, p_i^{\leq})$. These definitions will be useful when defining the cryptographic mechanisms for protecting entire routes.

D. Signatures and Set-Membership Proofs

For completeness, recall the definition of a signature scheme. A signature scheme consists of three functions:

1) a randomized generation algorithm G takes as input a security parameter (e.g., the desired length of the output)and generates a public/private key pair $(\mathsf{pk}, \mathsf{sk})$;
2) a signing algorithm S that takes as input a secret key sk and a value a and computes a signature σ; and,
3) a verification algorithm V which takes as input a public key pk, a value a, and a signature σ and outputs "accept" or "reject".

G, S, and V satisfy the following signature-correctness condition. For all $(\mathsf{pk}, \mathsf{sk})$ generated by G and all strings a, if $\sigma = S(\mathsf{sk}, a)$, then $V(\mathsf{pk}, a, \sigma) =$"accept".

A set-membership proof is defined similarly. It consists of three functions, G', S', and V': 1) a randomized key generation algorithm G' takes as input a security parameter (e.g., the desired length of the output) and a set of elements, A, and generates a public/private key pair $(\mathsf{pk}', \mathsf{sk}')$; 2) a signing algorithm S' that takes as input a secret key sk', a set A, and an element of the set a, and computes a set membership proof π; and, 3) a verification algorithm V' which takes as input a public key pk', a value a, and a proof π and outputs *accept* or *reject*. Note that if a is not in A then S' outputs $\perp$.

G', S', and V' satisfy the following correctness condition. For all A and all $(\mathsf{pk}', \mathsf{sk}')$ generated by G on

A, and all strings $a \in A$, if $\pi = S'(\mathsf{sk}', A, a)$, then $V'(\mathsf{pk}', a, \pi) =$ "accept".

Definition: A set-membership proof scheme is (k, T, ϵ) secure against existential forgery if it also satisfies the following security requirement. An adversary is allowed to ask for public keys to be generated for sets of its choosing. The adversary is then allowed to see the signatures for k (set, set element) pairs where the pairs can be chosen by the adversary adaptively. No adversary running in time at most T can generate a (signature, set element, public key) triple that passes verification, except with probability at most ϵ.

The above definition of a set-membership proof scheme may be modified to include ancillary information about the set. That is, the signing algorithm may be modified to include this ancillary information about the set as input. If this is the case, the verification algorithm must be modified as well to include this ancillary information for proper verification.

A secure set-membership proof scheme can be constructed from a secure signature scheme and a hash function secure against second pre-image attacks (for random domain elements). The advantage of a set membership proof scheme over a signature scheme is that in practice for both the signer and the verifier, the expensive public key computations need only be done once and then cached for any given set. [1] This efficiency comes at the price of larger space requirements but we note that the size of the membership proofs can be made logarithmic in the cardinality of the set. An example of a set membership proof system is the combination of Merkle hash trees and public key signatures as in the example above.

An important property of a set-membership proof scheme to highlight is that the signer only needs to compute T and S once, regardless of how many set elements it will eventually compute membership proofs for. That is, the cost of one public key signature computation can be amortized over the cost of many set-membership proofs. Likewise, a verifier needs only to run the signature verification algorithm on one valid (τ, σ) pair. It can cache the positive result using τ as a key. Subsequent membership proofs with set tag τ require only the verifier to run E, which is not a public key algorithm. Thus, the cost of one public key signature verification can be amortized over the cost of many set-membership verifications. In subsequent sections, we analyze the amortization savings that can be realized in practice on real BGP data streams.

[1] This is not transparent from the abstract description of a set membership proof scheme above. A formal description of a set membership proof scheme that explicitly breaks out the public key computations is cumbersome and omitted here for lack of space.

E. Route Attestation Tags

A attestation by an identity x about a string α is denoted $A(x; \alpha)$. An attestation is either a secure signature signed by the secret key of x or it is a membership proof of α by the identity x (using the secret key of x). We will denote an attestation by x about a string β *to an identity* y by $A(x; \beta : y)$. This is just an attestation $A(x; \alpha)$ with $\alpha = \beta : y$. Attestation may also have timestamps or expiration times. These may be used, in part, as anti-replay mechanisms. For purposes of exposition, for now we do not include timestamps in the notation. We defer discussion of the issue of of replay to the technical report.

Definition: For a given route we define a *route attestation tag* or RAT, as follows. A RAT takes as an input a route $r = (b, p)$. $RAT(r)$ is a sequence of attestations defined recursively as follows.

$$RAT(r_i^{\leq}) = RAT(r_{i-1}^{\leq}), A(p_{i-1}; r_i^{\leq} : p_i)$$

for $i = 2, ..., |p|$. The base case is $RAT(r_1^{\leq})$. This is the origin authentication tag, or OAT, for ownership of the address block $r_0 = b$ by the AS with identifier p_1. The semantics of $OAT(b, a)$ were discussed extensively in [2]. Briefly, the $OAT(b, a)$ includes: a.) a chain of attestations from IANA to an organization O attesting to the fact that the ownership of the address block b has been delegated to O; b.) an attestation by IANA that it has assigned the AS identifier a to O; and c.) an attestation by O that it has assigned the address block b to AS a.

As an example, let $p = (a_1, a_2, a_3, a_4)$. Then

$$\begin{aligned} RAT(b; a_1, a_2, a_3, a_4) \quad = \quad & OAT(b, a_1), \\ & A(a_1; (b; a_1) : a_2), \\ & A(a_2; (b; a_1, a_2) : a_3), \\ & A(a_3; (b; a_1, a_2, a_3) : a_4) \end{aligned}$$

Note that the final attestation in $RAT(b; p)$ is by the second to last AS in the path, i.e., by AS $p_{|p|-1}$.

A RAT is valid only if all of the associated attestations are valid and the OAT is valid. Note that RATs as defined here are nearly identical to the definition of route attestations in defined in [17]. The only minor differences are the inclusion of the origin authentication tag and the slight generalization to allow both signatures and set-membership proofs in the individual router attestations. In the technical report we discuss the addition of the origin authentication tags to RATs.

We denote the concatenation of a route $r = (b; p)$ and an AS a by $r.a$, where this is just the route given by the pair $(b; p.a)$, i.e., the path of r extended by one hop to a.

Definition: A route $r = (b, p)$, and an accompanying $RAT(r.a')$, when received in an update over an eBGP session by a router in AS a is considered valid only if:

1) $a = a'$,

2) $p.a$ is almost simple,
3) the RAT of $r.a$ is valid, i.e., the pair $(r.a, RAT(r.a))$ validates, and
4) the route was received over an authenticated eBGP session with a router in AS a^* where a^* must equal the last AS in the AS path p, i.e., $a^* = p_{|p|}$.

As defined above, a router that announces its new best AS path for a given address block to all of its neighbors must send a slightly different attestation to each of its eBGP neighbors. That is, to announce the route r it must send $r, RAT(r.a)$ to an eBGP peer in AS a, and $r, RAT(r.a')$ to an eBGP peer in AS a' etc. At first glance this may seem unnecessary. However, different routers in the same AS may announce a different best AS path for the same prefix. If when advertising the route r, the router simply attested to the route up to and including its AS, it is easy to construct cases in which upstream routers can forge routes [27].

A similar reason argues for the requirement that the prefix be included in all of the attestations of a RAT. The alternative is to have the attestations in the RAT include only the AS path and to separately include the origin authentication tag for the prefix and origin AS. However, such a scheme allows for the following type of attack. Suppose a router in AS b receives routes for two different prefixes both originated by AS a, e.g, $(b; a.p.b)$ and $(b'; a.p'.b)$ and the origin authentication tags binding b and b' to a. If the attestations in the RATs contain the appropriate AS path prefixes but are not required to contain the address block, then the router in AS b can create RATs that will validate for routes it did not receive. In this example the router can create a valid RAT for $(b; a.p'.b)$ and $(b'; a.p.b)$, thus altering in an undetected fashion the routes for the prefixes b and b'.

Note that in order for a router in AS a to check the validity of $RAT(r.a)$, it is not sufficient for the router to simply have the certified value of the public key of its eBGP neighbor that sent it the route. The router must have the certified public keys of all of the ASes in order to check the attestations of each AS in the route. Here we assume a PKI provides each router with the certified public keys of all ASes. For a discussion of such a PKI see [35].

We now address the issue of the security guarantee provided by the RAT construction. Intuitively, we would like to say that as long as the attestation scheme used in a RAT is not existentially forgeable, then that RAT scheme is not existentially forgeable in the sense that an adversary cannot create a valid (route, RAT) pair that it has not previously seen. Unfortunately, it is not quite that simple. This is due to the fact that every AS, including malicious ones, are able to extract or extend valid $(r, RAT(r))$ pairs sent to them legitimately in several ways. For example, from a valid route attestation for r, it is easy to extract a valid route attestation tag for each prefix of r, i.e., $r_i^{\leq}$ for $i = 1, \cdots, |r|$. This follows directly from the recursive definition. As another example, if a router in AS a receives a valid pair $(r.a, RAT(r.a))$, then a (possibly different) router in a can compute a valid $RAT(r.a.a')$ for any neighboring AS a'. This is due to the fact that $RAT(r.a.a') = RAT(r.a), A(a; r.a : a')$ where $RAT(r.a)$ is given to AS a and $A(a; r.a : a')$ is an attestation by a itself. Moreover, since AS padding is allowed in BGP, a can form valid RATs for the form $RAT(r.a^i.a')$ for any neighboring AS a' and any $i \geq 1$, where a^i is a repeated i times. Let us call these extensions of a RAT *transit extensions*.

Below we will define all possible transit extensions of a given set of routes. Then we will show that if the adversary can compute a valid RAT for a route that is neither in the set of routes for which it has seen a valid RAT, nor in the set of its transit extensions for those routes, then the adversary must have computed an existential forgery of an attestation.

Let $\mathcal{P}$ be a set of AS paths. Since all "good" routers check whether a path is almost simple, assume without loss of generality that all the paths in $\mathcal{P}$ are almost simple. Denote the transit extensions of $\mathcal{P}$ by x as $TE(\mathcal{P}, x)$. We define it iteratively as follows. First, for each $p \in \mathcal{P}$ all of the prefixes of p are added to $TE(\mathcal{P}, x)$, including p itself. Now, for each $p \in TE(\mathcal{P}, x)$, except for those that contain x, add the set $p.\{x\}^*$ and the set $p.\{x\}^+.\mathcal{Q}_{p,x}$ to $TE(\mathcal{P}, x)$, where $\{x\}^* = \{x^i \mid i \geq 0\}$ and $\{x\}^+ = \{x^i \mid i \geq 1\}$. Here $\mathcal{Q}_{x,p}$ is $\mathcal{ASN}$ minus x and minus the ASes in p and is defined so that all of the extensions are almost simple paths. An example of the transit extension is given in our associated technical report [1].

Definition: A secure RAT is defined as follows. An adversarial AS x is given access to a RAT oracle. That is, x can query the RAT oracle on routes r of its choice in a dynamic fashion and receive $RAT(r)$ for each of its queries. Let $\mathcal{P}$ be the set of such routes. A RAT forgery by x is a valid $(r, RAT(r))$ pair for some r not in $TE(\mathcal{P}, x)$, the set of transit extensions of $\mathcal{P}$. A RAT is secure if no time bounded adversary with access to a RAT oracle can compute a RAT forgery except with negligible probability. This definition of security can be parameterized in the standard fashion by a time bound, a query bound, and a probability bound but we omit the details of this parameterization here. These definitions lead to the main security lemma for RATs.

Lemma: If AS x has a strategy for computing a RAT forgery then there is an efficient strategy for computing an attestation forgery.

A proof of the lemma is included in the technical report. The implication of the lemma is that if the attestations are secure as per the definitions above then the route

attestation tag will be secure as per the definition above. The security lemma and proof can be easily modified to include security parameters. That is, the above lemma can be extended to give the security parameters of the RAT scheme as a function of the security parameters of the underlying attestation scheme.

Note that the adversarial model and proof of security are quite strong. They allow an adversarial AS x to see RATs for any routes of its choosing including, for example, routes that do not correspond to actual topology and routes that may have already transited x and continued for several more hops. If x is colluding with another AS, it may indeed be able to see the latter RATs. The security model protects against forgeries even with this type of collusion. It is better, of course, to overestimate the power of an adversary since it is always difficult to bound the information that a determined adversary can uncover. Even with this more powerful model, the security of RATs reduces to the security of the underlying attestations.

It is possible to capture the case of several ASes colluding with definitions and a security lemma similar to that above. However, the definitions are more complex and are omitted here. But, intuitively suppose a set of adversaries X is given valid RATs for a set of paths $\mathcal{P}$ of their choosing. The transit extensions of $\mathcal{P}$, $TE(\mathcal{P}, X)$, consist of all of the almost simple paths for which the adversaries can derive valid RATs from the valid RATs for the paths in $\mathcal{P}$. If the adversaries X can succeed in computing a valid (route,RAT) pair for a route not in $TE(\mathcal{P}, X)$ then they have succeeded in computing a RAT forgery. As long is it is computationally difficult to forge an attestation with non-negligible probability, it is computationally difficult for the adversaries X to compute a RAT forgery with non-negligible probability.

F. Constructions

In this section we describe the attestation schemes used by S-BGP and Nicol et al. In addition, we propose a different attestation scheme and several variants. S-BGP attestations include a validity interval $I = [t_s, t_e)$ and are denoted $A(a_1; r : a_2|I)$ As noted before, the S-BGP attestations are implemented as a public key signature. That is, $A(a_1; r : a_2|I) = I, \sigma$ where σ is the signature of the string $r : a_2|I$ using the private key of a AS a_1.

Both the Nicol et al. scheme and our schemes are based on set-membership proofs. Nicol et al. make crucial use of the fact that BGP speaking routers do not continuously send BGP updates to their neighbors. Instead, BGP speaking routers group route updates into 30-second intervals, and only send these updates to their neighbors at the end of the 30-second interval. For a given router in AS a, define $\mathcal{R}_t$ to be the set of all tuples $(r : a')$ such that route r is sent by the given router in an eBGP update to a router in a' in the 30-second interval ending at time t. The routers in this scheme create set-membership proofs for the set $\mathcal{R}_t$ for each time t that ends an interval. As discussed previously, ancillary information can be included in a set-membership proof. In this case, both the time of the update t and the expiration time of the announcements t_e are included. Let π_α be the set-membership proof for $\alpha \in \mathcal{R}_t$. Then for each $\alpha \in \mathcal{R}_t$ the attestation $A(a; \alpha|[t, t_e))$ is simply $(\pi_\alpha, [t, t_e))$. Since the router is sending the attestations for each $\alpha \in \mathcal{R}_t$, this set of attestations can be more parsimonious than the collection of individual attestations (we omit details for brevity). Nonetheless, as (route, attestation) pairs for routes represented in $\mathcal{R}_t$ are forwarded downstream in the appropriate RAT for an extension of the route, the amortized length of the encoding of (π_α) will increase. This is because fewer and fewer of the attestations for elements of $\mathcal{R}_t$ will be included in downstream updates.

Our scheme is similar to Nicol et al. in that we also use set-membership proofs. However, the method with which we choose to aggregate updates into sets is different. Assume for now that a router in AS a knows in advance all of the routes it will send during time interval $I = [t_s, t_e]$. That is, let $\mathcal{T}_I$ be the set of all tuples $(r : a')$ where the route r was sent by the given router to a router in a' within the interval I. Within the interval I, when the router needs to compute the attestation $A(a; \beta|I)$, for $\beta \in \mathcal{T}_I$, it computes the se-membership proof π from β, $\mathcal{T}_I$, and ancillary information I. The attestation for β is (π, I). Of course, a router cannot know in advance all of the routes it will receive in an interval I. However, as we will show in subsequent sections, for BGP updates, the past is a fairly accurate predictor of the future. Thus the set $\mathcal{T}_I$ is an approximation based on past history of the routes that will be needed for updates in period I. When the router needs to send an attestation for a route not in $\mathcal{T}_I$, it simply computes an S-BGP attestation. If $\mathcal{T}_I$ is required to have a maximum size bound, as it must, then there are a variety of caching strategies for maintaining $\mathcal{T}_I$ from one interval to the next.

As we will see, for reasonably sized intervals I, the set $\mathcal{T}_I$ can get quite big. However, $\mathcal{T}_I$ can be partitioned into smaller sets in a number of ways, and then a set-membership proof scheme can be applied to each set. This affords a time-space tradeoff. For example, for all address blocks b in tuples in $\mathcal{T}_I$, let $\mathcal{T}_{b,I}$ be the tuples that have address block b. In what we denote the *prefix scheme*, the router creates set tags and set-tag signatures for each $\mathcal{T}_{b,I}$. And the attestations are membership proofs for the appropriate set and member of that set. In another variant, $\mathcal{T}_I$ is partitioned according to the origin AS. That is, we define $\mathcal{T}_{a,I}$ as the elements of $\mathcal{T}_I$ that share the same origin AS. The scheme based on this partition is denoted the *origin AS scheme*.

Tail Mass Test	Min (LP)	Median (LP)	Max (LP)
Prefix (h=10)	67 (#23)	1,178 (#8)	17,784 (#40)
Prefix (h=20)	0 (#23)	63 (#17)	3,027 (#40)
AS (h=10)	163 (#23)	1,135 (#8)	4,967 (#40)
AS (h=20)	10 (#23)	142 (#8)	1,701 (#40)

TABLE I
LISTENING POINT TAIL MASS

A final variant of our scheme allows the expiration time of an attestation to be different from the expiration time of the set-tag signature. For example, suppose we partition I into k subintervals. Let the set of intervals be $\mathcal{K}$. Using the origin AS scheme as an example, for each $\mathcal{T}_{a,I}$, the router creates a proof system for the set $\mathcal{T}_{a,I} \times \mathcal{K}$. Note that since the size of membership proofs can be made to be logarithmic in the size of sets, this only adds $\log |\mathcal{K}|$ to the length of the membership proofs. In this case, the output of an attestation is the same as above plus the particular subinterval used. That is I is used in the public key signature validation of σ and the subinterval is used to encode the leaf that the verifier must use.

Note that for every address block includes an empty path in $\mathcal{T}_I$. Within our setting, we consider a withdrawal of an address block to be denoted by a route advertisement of that address block with an empty path.

IV. PATH STABILITY

This section describes our analysis of the central hypothesis upon which our cryptographic constructions are based: the set of paths for a prefix or emitted from a AS are small and stable over time. The following experiments evaluate path *density* (number of distinct paths) and *stability* (rate of discovery of new paths). In these experiments, we examine data from the 40 listening points of the Route Views [22] BGP repository. Each listening point data-set represents a full transcript of all UPDATE messages received by a particular AS, and hence explicitly enumerates the paths received over the experimental period. Other studies use the Route Views data to investigate the number of unique paths to a prefix assuming connectivity to two listening points for a one-day timeframe [10], to estimate the number of cryptographic operations required for prefix validation [40], to establish a delegation hierarchy [2], and to examine address allocation and routing table growth [5], scalability of router memories [24] and table fragmentation [19]. Our stability study differs significantly from these efforts in that we consider the *tail mass* of AS paths for each listening point, and determine the *rate of discovery*, a measurement of newly observed paths.

We begin our analysis by using *tail mass* to measure path stability. Tail mass $T_h(k)$ is the number of unique values above a threshold h encountered by observer k.

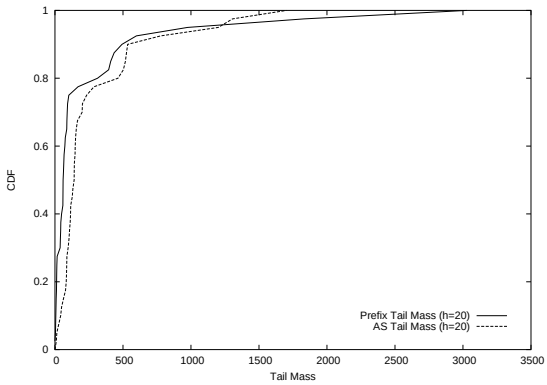
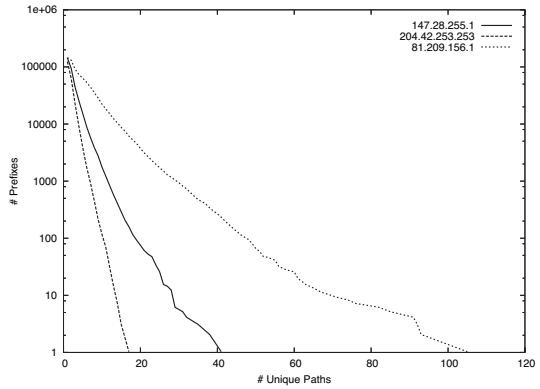
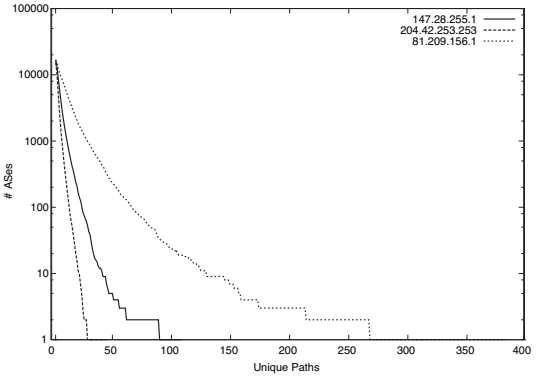


Fig. 3. tail mass - CDF of tail mass for 40 Route Views listening points during February 2004.

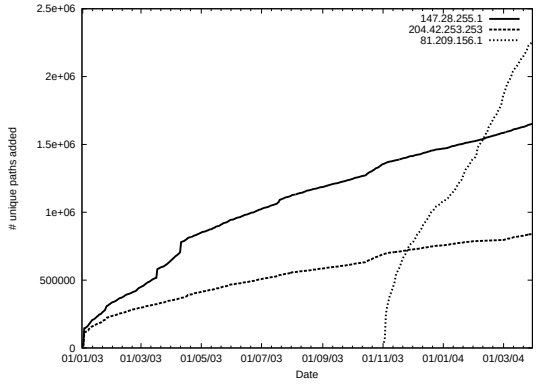


(a) Unique paths per prefix

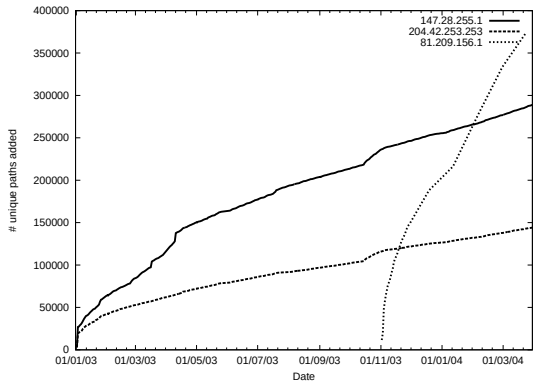


(b) Unique paths per AS

Fig. 4. CCDFs of unique paths per prefix and per AS measured from multiple Route Views listening points, for February 2004.



(a) Discovery rate per prefix



(b) Discovery rate per AS

Fig. 5. Rate of discovery CDFs for new prefixes and ASes as seen by multiple Route Views listening points, Jan 2003 to Mar 2004.

This study is concerned with number of unique paths, so we calculate tail mass as the number of prefixes or ASes that have more than h unique path vectors associated with them. Intuitively, tail mass shows how many prefixes or ASes have a "large" number of paths associated with them (as defined by a threshold h). The following is based on the analysis of the 217,707,968 updates observed by the 40 listening points during February 2004.

Figure 3 shows a cumulative distribution function of the prefix and AS tail masses of each listening point when the threshold is 20 (h=20). A striking aspect of this data is its density, where 80% of the listening points have a tail mass less than 500, and 67% have masses less than 200. This indicates significant stability at the listening points.

Table I summarizes the most, least, and median-stable listening points as represented by tail mass as measured of several experiments. The data suggests candidate *representative* listening points as models for minimum, maximum, and typical stability. As such, we select listening point 23 (204.42.253.253) as maximally stable, point 40 (81.209.156.1) as minimally stable, and point 8 (147.28.255.1) as typical in the following experiments.

We now use the representative listening points to more closely scrutinize the path stability. Figure 4(a) shows a CCDF for the unique number of paths observed by the listening point associated with various prefixes. In the average case, less than 2% ¿ of prefixes have more than 10 paths associated with them, and less that 0.06% more than 20. In the worst case, 15.3% of prefixes have more than 10 unique paths, 2.57% have more than 20, and 1.17% have more than 25.

Figure 4(b) shows a CCDF for the observation of unique paths by AS. Because the number of ASes a listener sees is little more than 10% of the total number of prefixes seen, we would expect that the number of unique paths per AS would be correspondingly larger than in the per-prefix case. However, the difference is not as pronounced because many prefixes originating from the same AS will have the same path. This vector will be counted n times for n different prefixes, but only one if they all originate from the same AS. For the average case, we found that 6.90% of ASes have more than 10 unique paths, and only 1.00% more than 20. In our worst case, 33.2% of ASes have more than 10 unique paths, 11.1% have more than 20, and 5.17% have more than 30.

The path lengths for the minimally stable listener (81.209.156.1) are considerably longer than for other listeners. A WHOIS lookup and traceroutes to the destination show this router to belong to LambdaNet Communications Deutschland AG, and its location to be in Ashburn, VA. The reasons for its distinctly different global view of paths is unclear, but may relate to route filtering policies or other policy or connectivity issues. We are actively researching this unusual behavior, but as yet have not been able to ascertain its true source.

A final series of tests assess the stability of the set of observed paths. Centrally, these tests attempted to estimate listening point *rates of discovery*. The experiments compute the frequency with which new paths are observed. We classify newness with respect to the AS (new when the AS has never advertised the particular path before) and prefix (the prefix has never been advertised with the path). Using the previously defined listening points, we examine the period between January 2003 and March 2004; the rates of discovery are shown in figures 5(a) and 5(b).

Two trends emerge from this study. First, there is nearly an order of magnitude difference between the number of new paths discovered per AS versus per prefix. An AS can have many different prefixes, each advertising the same AS path. Hence, when classification is done by origin AS, the path is only counted once, versus n times for n different origin prefixes. Although difficult to observe in the figures, a second trend shows strong discovery periodicity. We found that regular periods of

little discovery corresponded to weekends. The network is at its most stable on the weekend, and hence little activity was observable in the BGP feeds.

The preceding experiments show that paths observed in operational environments are enormously stable and dense. In the next section, we show how we can exploit this to achieve secure and efficient interdomain routing.

V. Evaluation

In this section, we evaluate the efficiency of the constructions defined in the proceeding sections via trace-based simulation. We compare our solutions against S-BGP and its variants, and draw general conclusions about the effectiveness of the proposed optimizations.

The simulations reported in this section use BGP update data collected during January 2004. Based on the results from the previous section, we ran simulations for the "typical" listening point (`147.28.255.1`).[2]

The simulation of our tree-based proposed schemes requires knowledge of all the paths advertised by an AS, which cannot be determined from a single listening point. One observation we make is that we are likely to see more unique paths from those ASes we are closest to. We approximate this by assuming unique paths comprise 7/8 of the paths observed from those ASes one hop away, 6/8 from ASes two hops away, etc., and adjust the tree size appropriately.[3] More precisely, if u unique paths associated with a proof system for an AS h hops away are seen, the proof system size is approximated to be $u(2 - h/8)$, e.g., $h = 3, u = 16 \rightarrow s = 16(13/8) = 26$. Note that an over or under estimate will affect the simulated size of the proofs, but not impact the amount of computational resources needed to validate them.

Our *pasim* simulator models a single AS on the Internet and measures the costs associated with the validation of received paths. We measure cost in computation and bandwidth. Computation is measured by the number of signature validations, which dominates all other computational costs (e.g., buffer handling, etc.) making it a good cost approximation. The simulations measure the amount of bandwidth consumed by the received proofs, but do not consider bandwidth consumed by other non-security related bandwidth costs (e.g., control traffic). We do not simulate the costs associated with the generation of proofs. Because structures are signed with low frequency (days), these costs will be dominated by validation.

We simulate S-BGP, the Nicol S-BGP variant (or simply Nicol throughout), the prefix scheme, origin AS scheme, and the all AS paths scheme as defined in the preceding sections.[4] Each timed UPDATE in the trace data is played back to the simulated BGP router and processed according to the simulation solution. Unless described otherwise, all tests in this section assume that received signatures are hashed and kept in a 16 MB cache (described in further detail below), with simulated tree-based proof systems regenerated every 24 hours and *RAT*s issued every hour.

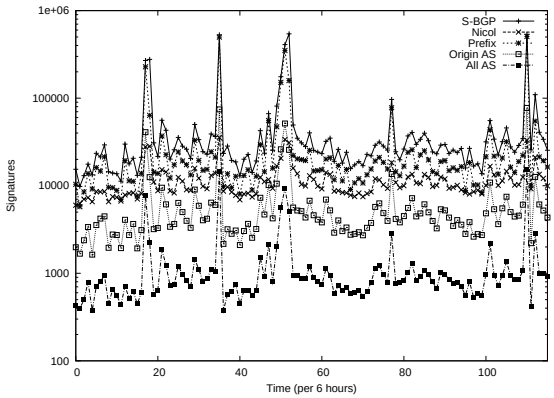


Fig. 6. Validation cost - signatures validated per hour in simulated *S-BGP*, *Nicol*, *prefix path*, *origin paths*, and *all path* validation.

A. Simulation Results

Our initial simulations compare computation and bandwidth usage. Figure 6 shows the number of signatures used by each scheme. S-BGP consumes the most computational resources validating signatures. The Nicol optimization effectively reduces these costs by half. This drop is due to the amortization of signatures across the 30-second time period. Interestingly, this indicates that, on average, only a few paths propagate through an AS in a given time period. Because of the sustained load, the data lets us posit that optimizations over short periods (such as Nicol) are likely to be less effective than longer periods, even if the latter may require more resources. The tree-based solutions require fewer validations than S-BGP. The prefix solution reduces the load by about 1/3. This is the effect of amortization over prefixes. Prefixes are largely stable and offer few paths, particularly over short time scales. Announcements for most prefixes will only be observed one or a few times per day. Hence, there is little opportunity to optimize. Note, however, that schemes such as SPV amortize costs in a fashion orthogonal to ours. Using our constructions in conjunction with those schemes could potentially reduce computational costs even further. The remaining AS path optimization schemes dominate all others: the origin paths

[2]We repeated the tests in the most and least stable listening points. In all cases, the costs scaled with the number of unique paths and rates of discovery as discussed in the preceding section.

[3]We conservatively chose 8, as we observed that paths of four or more hops from the core were typically originated by stub ASes.

[4]We simulated operation of the final variant of our scheme described in section 3, where expiration time of the attestation could be different from expiration time of the set-tag signature. The results differed from our origin AS scheme by a small factor. Hence, for clarity we omit these results from the graphs.

scheme represents an 86.3% reduction, and the all paths a 97.3% reduction in signature validations over S-BGP. In a given 24 hour period, the maximum number of signatures encountered will be two times the number of active ASes (assuming that all path proofs expire at some point during the day, and are recreated). The origin paths are somewhat more costly because they fail to fully exploit the opportunity to amortize cost.

Hashing typically consumes vanishingly small amounts of computational resources compared to signature validation. However, in some schemes, hashing can be performed frequently enough that it potentially impacts performance. We found that the in the all path approach, about 500 hash operations were performed for every signature validation. Assuming that hashing is approximately 1,000 times faster than RSA signature validation (as in Hu et. al [10]), this would increase the computational costs by about 50%. In all other cases, the hashing was dominated by signature validation.

Not shown for space considerations, the *instantaneous rate* of signature validations per router indicates the number of signatures per time quantum (in this case, 1 minute). We found many bursts where many validations are necessary per minute, particularly in the prefix scheme (where on average a burst would require less than 30 signature validations, but rare peaks would require a hundred or more). The origin scheme, which strikes the best compromise between validations and bandwidth, generally requires under 10 validations per minute, or one every six seconds on average.

Figure 7 shows the number of validations required for the origin scheme at the three listening points. The listening point demonstrating worst-case behavior has a number of bursty points with significant numbers of validations required; however, this burstiness is evident in all schemes and is constant across listening points.

Demonstrated in Figure 8, the bandwidth costs are largely the inverse of signature costs. S-BGP consumed far less bandwidth than the other approaches, because it generates small proofs. The prefix and Origin AS approaches were significantly more costly, consuming 3.35 and 3.57 times more resources than S-BGP, respectively. Interestingly, Nicol was second only to the all path scheme in consuming resources. The Nicol scheme creates a tree for every 30-second quantum, and subsequently sends a potentially large set of succinct proofs every period. The all path scheme was by far the most costly approach, consuming about 6 times as much bandwidth as S-BGP. In this case, the average bandwidth consumed per 6 hour period is 77 kilobytes. However, this approach may be prohibitive due to short bursts, which required as much as 139 megabytes in a single minute.

Any path authentication scheme must allocate storage resources for security relevant state (e.g., cryptographic proofs). In S-BGP, the additional space requirements to hold route attestations is estimated to be between 30 and 35 MB per BGP peer, though it is suggested that memory requirements in asymmetric peering relationships, such as between a large ISP router and a number of smaller peers, would be lower [14]. The storage requirements of the schemes proposed in this paper are unique to their design. Recall that the prefix approach requires every prefix to have a proof structure, while the all path approach requires a proof per AS; these two schemes form maximal and minimal requirements, respectively. Our simulations show that the total cost of storing *all proofs* across all peers ranges from approximately 55-60 MB for the prefix scheme to under 10 MB for the all path scheme. In the origin AS scheme, the total cost is approximately 25 MB.

The simulations illustrated in Figure 6 assume a proof cache of 16 MB. In our simulation model, this cache is separate from the storage space for the full set of proofs. We make this design decision so that the cache could be accessed more rapidly by the router as part of its fast path packet processing, but retain access to the proofs in stable storage (as needed for announcement creation). The additional stable storage costs are not onerous, and could likely be stored in memory itself on larger routers. Alternately, even smaller routers (e.g., Cisco 3600 series) include slots for flash memory, and are capable of accepting cards with 256 MB or greater, well above the requirements of our scheme. We assume that in real systems, to keep the cache size at a minimum, a hash of a received signature is stored in cache, rather than the signature itself. The router hashes the signature of an incoming update and checks whether it appears in the cache. If it is, a signature validation is not necessary. Hashed signatures are expired from the cache on a LRU basis. When sending an update, the full proofs to be sent are retrieved from stable storage.

VI. Discussion

Kent et al. have suggested a path validation optimization aimed at reducing the load on validating S-BGP speaking routers. This optimization dictates that paths are validated only when they are selected as the *best paths*. However, it is not clear if this does in fact eliminate the computational limitations of S-BGP. Consider an AS A with k neighbors. Any prefix p will be reachable through j neighbors, where $0 \leq j \leq k$, and j routes will be held by the AS. The fractional computational savings f for a given prefix on a given router over a period of time Δ is just the ratio of updates sent for that prefix during Δ divided by the total number of updates received for that prefix during Δ. Of course, f will vary from router to router and prefix to prefix, but f is likely to be on the order of $1/j$ for j defined above. For the data collected in

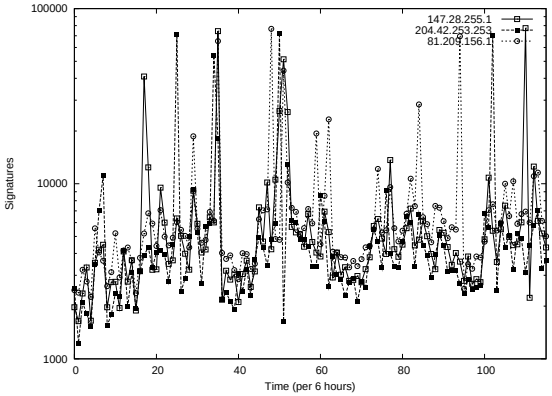


Fig. 7. Signature validations by listening point - validations for the origin paths scheme per 6 hour period.

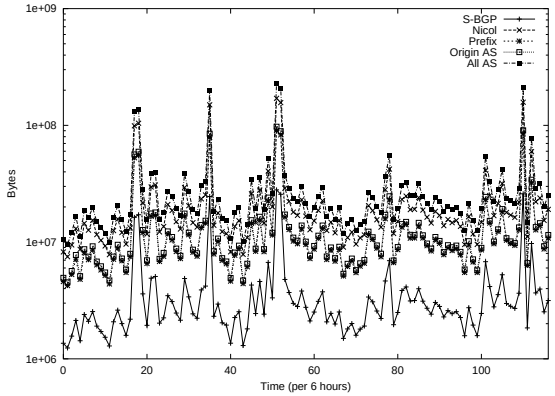


Fig. 8. Bandwidth Cost - the number of bytes consumed by the transmission of the simulated path validation approaches.

our study, the median number of unique paths per prefix was 2.5 and the mean value was 2.8. A careful study of f remains for future work. But we note here that the same optimization can be used for our RATs based on set-membership proofs. We will also achieve a factor f computational speedup. That is, when the optimization is applied to both schemes, the ratio of the computational overheads will remain the same.

A major difficulty of retrofitting security is the need for *incremental deployment*. Simply put, there are large portions of the Internet that will adopt solutions slowly or not at all. Any feasible solution must be designed such that communities of interested parties can work collaboratively to provide a working, secure system. Moreover, functionality can not come at the expense of poorly equipped enterprises. Such approaches would disenfranchise people and networks, and reduce universality of the Internet. However, those who do not participate need not receive benefit from deployment.

Past systems such as IRV [8] addressed incremental deployment by performing security *out-of-band*. They allow parties to exchange data without any change to BGP. Those who wish to exchange security relevant data do so freely over any mechanism that is available and convenient. However, this approach only works when the network is otherwise healthy or alternate channels are available. psBGP takes another tack in which the parties police each other's activities [40]. The incremental deployment approach in psBGP is one of a mutual embrace: like soBGP, communities of peers must work in concert to achieve a larger security posture.

We adopt this latter scheme, where communities of like-minded organizations will organically form *unions* of ASes. These unions will mutually authenticate credentials to be used in the issuance of RAT proofs. At the protocol layer, like S-BGP and detailed in Section III, we adopt the strategy of signing transitions to and from non-adopting ASes. Of course, knowing which ASes are participating in the protocol is essential for ascertaining the validity of received routes. In a sense, our approach is similar to the S-BGP protocol, and as such can make use of its procedures and structures for incremental deployment.

VII. Conclusions

In this paper we have explored the efficiency of a range of optimizations for securing AS path attributes in BGP. Principally, we provide a formal model for the security of route attestation tags with an accompanying proof of security. We introduce the notion of prefix, origin AS, and all path proof systems, which amortize validation cost over aggregations of similar routing data. The stability of AS specific paths is used to assess the fundamental assumption upon which this work is based; the number of paths used by a particular AS for a given prefix is both small and stable over time. Through trace-based simulation, we show that our constructions reduce the cost of path authentication by as much as 97% over existing approaches, and the other storage costs are nominal.

The problems of BGP security are sufficiently important to warrant discussion in the United States National Strategy to Secure Cyberspace [28]. This work studies tradeoffs between computational, bandwidth and storage costs for a range of BGP security path authentication mechanisms and is a step in a larger communal effort to design and deploy BGP security. The ultimate goal is to develop a comprehensive understanding of the security, cost, and manageability tradeoffs for BGP, to inform sound engineering decisions for future deployments. To this end, we plan to extend our evaluations to a range of realistic network environments, and further explore systemic deployment issues.

References

[1] W. Aiello, K. Butler, and P. McDaniel. Implications of Path Stability for Efficient Authentication in Interdomain Routing. Technical Report NAS-TR-0002-2004, Networking and Security

Research Center, Department of Computer Science and Engineering, Pennsylvania State University, University Park, PA, Oct. 2004. Revised October 2005.

[2] W. Aiello, J. Ioannidis, and P. McDaniel. Origin Authentication in Interdomain Routing. In *Proceedings of ACM CCS '03*, October 2003.

[3] M. Baltatu, A. Lioy, F. Maino, and D. Mazzocchi. Security issues in control, management and routing protocols. *Computer Networks (Amsterdam, Netherlands: 1999)*, 34(6):881–894, 2000. Elsevier Editions, Amsterdam.

[4] A. Barbir, S. Murphy, and Y. Yang. Generic Threats to Routing Protocols (*Draft*). *IETF*, April 2004.

[5] S. Bellovin, R. Bush, T. Griffin, and J. Rexford. Slowing routing table growth by filtering based on address allocation policies. http://www.research.att.com/jrex/, June 2001.

[6] K. Butler, T. Farley, P. McDaniel, and J. Rexford. A Survey of BGP Security Issues and Solutions. Technical Report TD-5UGJ33, AT&T Labs - Research, Florham Park, NJ, Feb. 2004. (*revised June 2004*).

[7] B. Christian, B. Akyol, R. White, J. Haas, and S. Murphy. BGP Security Requirements (*Draft*). *IETF*, July 2004.

[8] G. Goodell, W. Aiello, T. Griffin, J. Ioannidis, P. McDaniel, and A. Rubin. Working around BGP: An incremental approach to improving security and accuracy of interdomain routing. In *Proceedings of NDSS '03*, Feb. 2003.

[9] Y. Hu, A. Perrig, and D. Johnson. Efficient security mechanisms for routing protocols. In *Proceedings of NDSS '03*, Feb. 2003.

[10] Y.-C. Hu, A. Perrig, and M. Sirbu. SPV: Secure Path Vector Routing for Securing BGP. In *ACM SIGCOMM*. ACM, August 2004.

[11] IANA. Autonomous System Numbers, March 2003.

[12] ICANN. The Internet Corporation for Assigned Names and Numbers, July 2004. `http://www.icann.org/`.

[13] P. J. Transmission Control Protocol - DARPA Internet Protocol Program Specification. *IETF*, Sep. 1981. RFC 793.

[14] S. Kent. Securing the border gateway protocol. *The Internet Protocol Journal*, 6(3), Sep. 2003.

[15] S. Kent. Securing the Border Gateway Protocol: A status update. In *Seventh IFIP TC-6 TC-11 Conference on Communications and Multimedia Security*, Torino, Italy, Oct. 2003.

[16] S. Kent, C. Lynn, J. Mikkelson, and K. Seo. Secure Border Gateway Protocol (S-BGP) Real World Performance and Deployment Issues. In *Proceedings of NDSS '00*, Feb. 2000.

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

[18] C. Kruegel, D. Mutz, W. Robertson, and F. Valeur. Topology-based detection of anomalous BGP messages. In *Proceedings of RAID '03*, Sept. 2003.

[19] X. Meng, Z. Xu, L. Zhang, and S. Lu. An analysis of BGP routing table evolution. Technical Report TR030046, Computer Science Department, UCLA, Jan. 2003.

[20] Merit Network. The Internet Routing Registry, July 2004. http://www.irr.net/.

[21] R. Merkle. Protocols for public key cryptosystems. Oakland, CA, Apr. 1980. IEEE Symposium on Research in Security and Privacy.

[22] D. Meyer. The Route Views Project, Nov. 2004. http://www.routeviews.org/.

[23] S. Murphy. BGP security vulnerabilities analysis. Internet Draft, Mar. 2003.

[24] H. Narayan, R. Govindan, and G. Varghese. The impact of address allocation and routing on the structure and implementation of routing tables. In *Proceedings of ACM SIGCOMM '03*, Karlsruhe, Germany, Aug. 2003. ACM.

[25] J. Ng. Extensions to BGP to support secure origin BGP (soBGP). Internet Draft, Oct. 2002.

[26] D. Nicol, S. Smith, and M. Zhao. Efficient security for BGP route announcements. Dartmouth Computer Science Technical Report TR-2003-440, Feb. 2002.

[27] O. Nordström and C. Dovrolis. Beware of BGP attacks. *Computer Communications Review*, 34(2):1–8, Apr. 2004.

[28] Office of the President of the United States. Priority ii: A national cyberspace security threat and vulnerability reduction program. National Strategy to Secure Cyberspace, Nov. 2004.

[29] R. Perlman. *Network layer Protocols with Byzantine Robustness*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, Oct. 1988. MIT/LCS/TR-429.

[30] J. Postel. Internet Protocol. RFC 791, Sept. 1981.

[31] J. Puig, M. Achemlal, E. Jones, and D. McPherson. Generic Security Requirements for Routing Protocols (*Draft*). *IETF*, July 2004.

[32] Y. Rekhter and P. Gross. Application of the Border Gateway Protocol in the Internet. RFC 1772, Mar. 1995.

[33] Y. Rekhter and T. Li. A Border Gateway Protocol 4 (BGP-4). RFC 1771, Mar. 1995.

[34] R. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM*, 21(2):120–126, Feb. 1978.

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

[36] B. Smith and J. Garcia-Luna-Aceves. Securing the border gateway routing protocol. In *Proceedings of IEEE Global Internet 1996*, London, UK, Nov. 1996.

[37] L. Subramanian, V. Roth, I. Stoica, S. Shenker, and R. Katz. Listen and Whisper: Security mechanisms for BGP. In *Proceedings of NSDI'04*, Mar. 2004.

[38] S. Teoh, K. Ma, S. Wu, D. Pei, L. Wang, L. Zhang, D. Massey, and R. Bush. Visual-Based Anomaly Detection for BGP Origin as Change (OASC) Events. In *Proceedings of IEEE/IFIP DSOM '03*, October 2003.

[39] C. Villamizar, R. Chandra, and R. Govindan. BGP Route Flap Damping. *IETF*, Nov 1998. RFC 2439.

[40] T. Wan, E. Kranakis, and P. C. van Oorschot. Pretty Secure BGP (psBGP). In *Proc. of NDSS '05*. Internet Society (ISOC), Feb. 2005.

[41] K. Zhang, S.-T. Teoh, S.-M. Tseng, C.-N. Chuah, K.-L. Ma, and F. Wu. Performing BGP experiments on a semi-realistic internet environment. North American Network Operators Group (NANOG), October 2004.

[42] X. Zhang, S. Wu, Z. Fu, and T.-L. Wu. Malicious Packet Dropping: How It Might Impact the TCP Performance and How We Can Detect It. In *Proceedings of ICNP 2000*, Nov. 2000.