Preventing Attacks on BGP Policies: One Bit is Enough

Srikanth Sundaresan, Robert Lychev, Vytautas Valancius
Georgia Institute of Technology
{srikanth, robert.lychev, valas}@gatech.edu

Technical Report GT-CS-11-07, Georgia Institute of Technology, 2011.

ABSTRACT

The Internet is comprised of many autonomous systems (AS) managed by independent entities that use the Border Gateway Protocol (BGP) to route their traffic. Although it is the de facto standard for establishing paths across the Internet, BGP is not a secure protocol and the Internet infrastructure often experiences attacks, such as prefix hijacking and attribute mangling, incurring great costs to ASes that experience them. Various solutions have been proposed in response to these attacks, such as Secure BGP, but they do not address traffic attraction attacks that stem from export policy violations. In these attacks, malicious ASes can introduce paths that are legitimate from the protocol standpoint and yet malicious to the users of that protocol. Although these attacks have been studied before, no solution has yet been proposed. In this paper, we thoroughly characterize this set of attacks and propose a very lightweight and effective scheme to address them. Our scheme requires no manual configuration. We show that even if only a small fraction of ASes deploy our scheme, the amount of possible attacks reduces by on order of magnitude.

Keywords: Border gateway protocol (BGP), secure border gateway protocol (S-BGP), BGP traffic attraction attacks, economic aspects of the Internet, algebra and dynamic network routing

1. INTRODUCTION

The Internet consists of thousands of autonomous systems (AS) managed by independent entities that route their traffic with the help of Border Gateway Protocol (BGP). BGP allows great flexibility in the way networks route packets, and it does not require a central coordination point for it to work. Unfortunately, the BGP is not a secure protocol, and the Internet infrastructure experiences multiple attacks every day costing millions of dollars to companies and individuals who experience them. Some of the most dangerous attacks are prefix hijacking for spam sending activities, prefix hijacking for denial of service attacks and, as was recently shown at BlackHat conference, BGP attributes mangling for traffic snooping. The research community and industry are developing defenses against such attacks. The most widely discussed *holistic solutions* – which are based on Public-Private Key infrastructure – are Secure BGP [13] and Secure-Origin-BGP (SO-BGP) [17].

Secure BGP and SO-BGP, in principle, solve many of today's BGP security issues. SO-BGP requires that each prefix and its origin be verified either against the central registry or against the distributed trust database, thus preventing some prefix hijacking attacks. A more comprehensive protocol - Secure BGP - verifies not only the origin information but also the path it traversed. Thus, Secure BGP, if deployed, could prevent BGP attacks that use path mangling. However, unfalsifiable propagation of the routing information does not solve all the BGP security problems.

Secure BGP ensures the correctness of the protocol and authenticity of the information in the received updates. No party in a BGP message-exchange process can alter or omit information about the Internet path and forward such altered information to its peers. Thus, as long as the path and the set of attributes are formed according to the protocol, the path is going to be accepted at the receiving router. In this paper we characterize a different set of attacks that do not rely on attribute mangling. Instead, the attackers abuse the legitimate Internet Service Provider (ISP) configurations to change the traffic routing on the Internet.

Attackers can introduce paths that are legitimate from a protocol standpoint and yet malicious to the users. Assume network in Figure 1. The malicious AS named Megan can become a *transit* network for the traffic between customers of Alice and Bob by simply propagating their prefixes to the appropriate upstreams. Attacker propagates path Customer-Of-Alice→Alice→Megan to Bob and propagates Customer-Of-Bob→Bob→Megan to Alice. Providers Alice and Bob choose the paths announced by the malicious network Megan, since the path to a customer is always preferred over the path to a peer for business reasons. We call such attack a **path hijacking** attack. Path hijacking types of attacks have been studied before in [14], where Goldberg et al. have shown them to be effective in attracting significant volumes of traffic. However, to the best of our knowledge, no solution to path hijacking attacks has yet been proposed.

The path hijacking attack is very hard to prevent in an automated way. First, such attack is hard to distinguish from normal behavior. Alice has no knowledge about the nature of the relation between Megan and Bob, and therefore announcements from Megan might be legitimate. Second, such attacks are very easy to conduct. Almost any smaller ISP can trick its upstream providers to send it traffic. Prevention of such attack would require sophisticated filters that would need to be updated manually with the configuration changes in client networks.

In this paper we propose a simple and yet effective technique to prevent path hijacking attacks in an automated way, without installing sophisticated path filters. We rely on the *valley free* path property, first observed by Lixin Gao [5], which states that under normal policy configurations, an advertisement should never go from a customer to a provider if the advertisement has already traversed a provider→customer link. In our protocol, before sending path announcements to their customers, the ISPs, label the routes to indicate that paths have started to propagate toward customers. Similarly, the ISPs receiving routes from their customers will check for a label that indicates whether the path ever traversed a provider→customer link.

In the following section we will explain the threat model under which we design our protocol in more detail. Section 3 will describe the theoretical background of the problem, provide an alge-

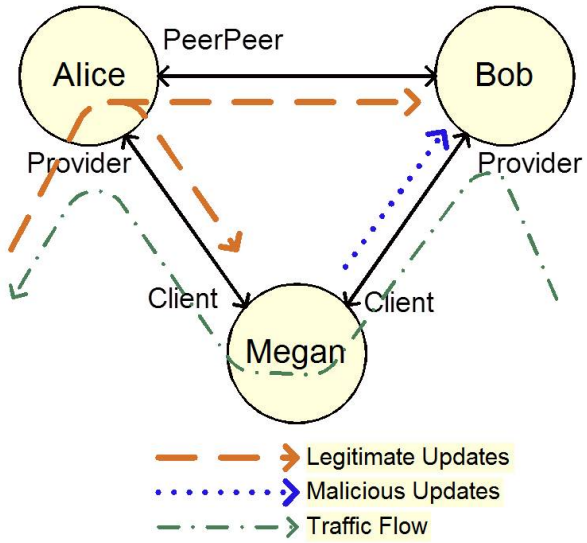

Figure 1: Valley attack on a peering relation.

braic formulation and a proof of convergence of any protocol that could address the threats explained in Section 2. Section 4 details the protocol itself. In Section 5 we evaluation our scheme with respect to partial deployment settings, and we show that the amount of possible attacks reduces by on order of magnitude even when only a small fraction of ASes deploy our scheme. We provide a summary of related work and conclude in sections 6 and section 7 respectively.

2. THREAT MODEL

The network. We assume an Internet scale network where each autonomous system (AS) is running Secure BGP. Each participating AS follows through with the default Secure BGP protocol and ASes that don't participate in protocol are not considered secure.

BGP is a path vector protocol with explicit trust agreements implemented via BGP sessions. Each network participant is assumed to forward the updates through the BGP sessions that reflect the updates received from these sessions as well as its own policy decisions. Secure BGP adds cryptographic elements to the protocol to secure the updates, so that participants cannot forge the information received from neighboring ASes.

Under legitimate circumstances there are three types of BGP sessions from the perspective of any AS: (1) connection to provider, (2) connection to a client, and (3) connection to a peer. Hence, only two types of connections are possible: (1) connection between a customer and a provider, and (2) connection between two peers. We assume that customers pay providers for traffic the provider agrees to deliver, and that there is no payment exchanged when two peers connect and deliver each other's traffic. With this arrangement, the traffic forwarding policies in our model can be ordered from most preferred BGP session to the least preferred BGP session. If an AS has all types of BGP sessions, it will first prefer to send traffic to a client, because the client is paying for the traffic. If the remote network is not reachable though the client, next that AS will prefer to send the traffic to a peer, because forwarding such traffic is essentially free. Finally, if the network is not reachable through clients and peers, that AS will send traffic to a provider.

The attacker. The attacker participates in the Secure BGP protocol and thus cannot modify the updates which it receives from its neighbors. As we will observe, no modification to BGP messages is necessary to conduct export policy attacks. The attacker, however, can enter into peering agreements with ISPs of its choice, subject to market and geographical constraints. We equally consider a case where an attacker acquires control over a legitimate service provider.

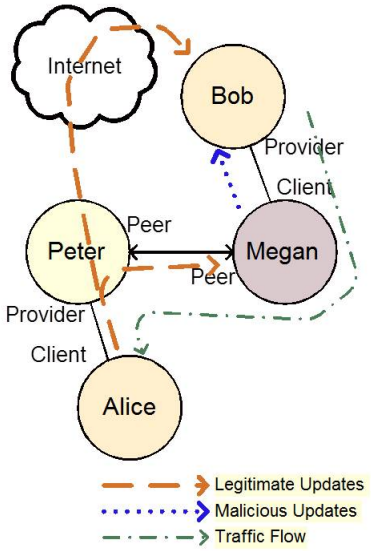

Figure 2: Step attack to attract the provider traffic.

Attack vectors. We consider export policy attacks on BGP sessions in the context of man-in-the-middle attacks. As an end network with its own AS number, the attacker network can observe the traffic originating or terminating at its own network. Further, if the attacker's network has legitimate traffic passing through (from its customers to its providers or peers), then such traffic can also be observed and, thus, is not considered a threat in our model.

A threat in our model is defined as an ability for an attacker to exploit weaknesses in legitimate BGP policies and perform man-in-the-middle traffic snooping attacks on traffic that should otherwise be unavailable to the attacker. There are three distinct cases of such attacks.

- **Valley attack.** Valley attacks occur when an attack has two connections to providers that peer to each other. In a legitimate system use case, traffic belonging to the customers of each of the providers is exchanged over the peering link as shown in Figure 1.
  Under normal circumstances, customer Megan in Figure 1 would not re-advertise routes received from Alice to Bob. If such routes are re-advertised, she becomes a transit for both service providers, incurring the costs of such transit in addition to the costs of connections to the providers. If Megan, however, is a malicious actor, she might reap benefits from such transit traffic by performing information gathering or spam dissemination. In fact, Megan can easily become a malicious actor by simply forwarding the appropriate announcements to its providers. The providers would select Megan as their preferred transit choice because of the default weighting of the routes as described in the beginning of this section.
- **Step attack.** A step attack can occur when a malicious actor forwards announcements it received from its provider to its

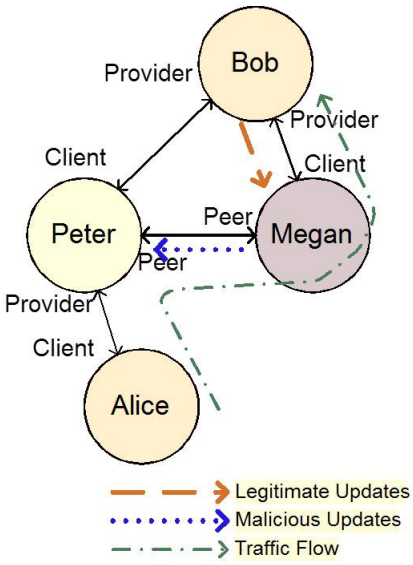

Figure 3: Step attack to attract the peer traffic.

peer, and when it forwards announcements from its peer to a provider.
Figure 2 shows the case when an attacker attracts traffic from a provider to a customer belonging to a peer of the attacker. As the attacker forwards announcements to its service provider, the service provider, given that the path the attack provides is shorter than the one obtained from a legitimate source, converges to an attacker. Note that such an attack will not attract all traffic, and it will depend on the topology of the Internet.
Figure 3 shows the case when an attacker attracts traffic from a peer that would otherwise go directly to the provider of that peer. In this figure, Megan announces the prefix received from a provider to a peer. Peer Peter recomputes his routing table and starts forwarding traffic to Megan instead of forwarding directly to the provider. Notice that despite having a shorter path to a provider, Peter prefers Megan, because he does not need to pay for the traffic serviced by Megan.

- **Mirroring attack.** An attacker can re-announce routes received from one peer to another peer, thus becoming a transit for the traffic between its peers. The attack works well if peers under attack do not have direct connection between them. Otherwise, direct connection between such peers would provide a shorter path than the attacker can introduce.

Aside from actions conducted by malevolent users, we can equally apply our security protocol to protect against the damage done by benign users. For example, many of the "attack vectors" described above can occur due to simple misconfiguration of BGP import-export policies.

3. THEORETICAL FORMULATION

BGP, and other path vector protocols, consists of rounds; at every round participants of BGP exchange information with their neighbors regarding reachability of various destinations. At every round, for any reachable destination of interest, a participant selects a local-optimal path out of all the paths to that destination that participant learned from each of its out-neighbors over the course of previous rounds. Criteria for optimality here may be as simple as the number of hops and as complex as a combination of length, throughput, average delay and reliability. In this paper we do not assume any particular criteria. We assume that all participants have well-defined relative preference for paths with the same origin, destination, and optimality criteria that totally orders those paths (e.g. an AS may be biased to send traffic through specific providers due to friendly business relations).

One of the basic goals of BGP is to make sure that the path protocol achieves stability in finite number of rounds–situation where no route messages are exchanged between nodes when no more links are added and deleted [7]. Section 3.1 presents BGP as a policy-based routing protocol, a scenario where violation of policies may result in protocol divergence, and an algebraic framework for expressing BGP policies, valleys, and steps, in a formal and succinct manner. Prevention of valleys and steps is important to ensure ultimate convergence as well as IP prefix hijacking attacks. In Section 3.2 we discuss sufficient and necessary properties for suppression of valleys and steps in BGP in the context of IP prefix hijacking attacks.

3.1 Convergence of BGP

In BGP every node has either a customer-provider or a peer-to-peer relationship, exclusively, with its every neighbor, where, regardless of the direction of traffic, in the former relationship money flows from customer to its provider while in the latter relationship no money flow takes place. We ignore the sibling relationship because sibling ASes could be collapsed and treated as single AS.

Policy-based routing makes sense in this context, and it can be roughly sketched by the properties listed in Table 1. In this Section we discuss BGP convergence issues with respect to these policies and express these with algebraic formulation.

3.1.1 Divergence Example

Whether intentionally or unintentionally, when the basic policies described above are violated, convergence of BGP cannot be guaranteed in general [15]. We say that violations of policies 2-4 result in steps, and we say that the violations of policy 5 result in valleys. [16] show a scenario, bad gadget, where independently chosen policies of ASes lead to divergence of BGP. We show a variant of the bad gadget scenario in Figure 4 as an example of how violation of policies (as per Table 1) may result in divergence of BGP. In this example c_0, c_1, and c_2 are customers of P; c_0 and c_1 are peers, c_1 and c_2 are peers, and c_2 and c_0 are peers. c_0, c_1, c_2 violate policy 2 due to a misconfiguration, but no participant violates policy 1. We do not assume that route advertisements are exchanged synchronously. Let us consider the execution of BGP presented in Table 2, where each step represents a single time unit. We see that the state of this BGP execution at Step 2 is essentially identical to that of Step 0, which implies that this execution of BGP will cycle forever in clockwise direction. Although no participant in this example may be malicious (e.g. c_0, c_1, c_2 may have only a misconfiguration problem), such scenarios are bad in general because they require nodes to continuously waste cycles and bandwidth while processing updates and exchanging route advertisements respectively.

3.1.2 Algebraic Formalization

We will now present a very general algebraic formulation of BGP as a policy-based path vector routing protocol. Most of the following notation and logic were borrowed from [15]. We represent the algebra of interest with a seven-tuple $(W, \preceq, L, \Sigma, \phi, \oplus, f)$, where W is a set of weights totally ordered by the relation $\preceq$, L is a set of labels, Σ is a set of signatures with a special signature ϕ that rep-

#	policy
1	–paths that start with a provider link (provider paths) are the least preferred while paths that start with a customer link (customer paths) are the most preferred; paths that start with peer a link (peer paths) are in the middle
2	–no node exports provider paths to a peer
3	–no node exports peer paths to a provider
4	–no node exports peer paths to a peer
5	–no node exports provider paths to a provider

Table 1: Basic policies for BGP.

#	action
Stp 0	$-c_1$ has direct access to P via a provider link, and allows c_0 to go through itself to reach P; c_2 accesses P directly through a provider link and advertises this access to c_1
Stp 1	$-c_1$ switches its access to P to go through c_2 and withdraws its transit to P for c_0
Stp 2	$-c_0$ accesses P directly through provider link and advertises this access to c_2

Table 2: Step-by-step example of how a BGP execution enters an infinite cycle in a network with steps depicted in Figure 4.

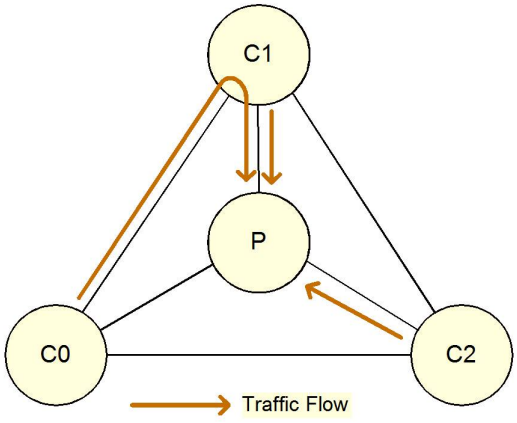

Figure 4: Bad gadget. Given that clients prefer peer on the left and only direct connectivity is announced to the peers, the protocol in such setting never converges.

resents a signature of an infinite-weight path. Binary operator $\oplus$ maps elements of $L \times \Sigma$ to Σ, and the function f maps elements of Σ to W. This algebra must satisfy the basic axioms of maximality and absorption: $\forall_{\alpha \in \Sigma \setminus \{\phi\}}\ f(\alpha) \prec f(\phi)$, and $\forall_{\ell \in L}\ \ell \oplus \phi = \phi$ respectively. Network links and paths are represented with labels and signatures respectively. For instance, label of link (u, v) is represented with $\ell(u, v)$. The signature of a path composed of a single destination node d is represented with $s(d)$, and a signature of any non-trivial path $uv \circ Q$, where Q is some already established path, is defined inductively as: $s(uv \circ Q) = \ell(u, v) \oplus s(Q)$. Here, binary operator $\circ$ denotes an extension of a path with a another path by pre-pending the latter to the former. With this notation we can denote weight of a path Q by $f(s(Q))$.

It was shown in [15] that monotonicity is necessary and sufficient for a path vector protocol to converge to local-optimal paths in-trees (for every source-destination pair in the network the chosen path to the destination is minimal in weight for the source), where monotonicity requires that

$$\forall_{\ell \in L} \text{ and } \forall_{\alpha \in \Sigma}\ \ f(\alpha) \preceq f(\ell \oplus \alpha).$$

In other words, monotonicity requires that the weight of some path Q must not decrease when it is prefixed by some other path P: $f(s(Q)) \preceq f(s(P \circ Q))$. We see that the non-converging example described in Table 2 of this section is a non-monotonic algebra which we specify below:

$$L = \{c, r, p\},\ \Sigma = L \cup \{\epsilon, \phi\},\ W = \{0, 1, 2, +\infty\},$$

$\preceq$ is $\leq, \oplus$ is defined as

$\oplus$	ϵ	c	r	p
c	c	c	c	c
r	r	r	ϕ	r
p	p	p	p	p

where left column represents labels and the first row represents signatures, and f is defined as

$$f(\epsilon) = 0, f(c) = 1, f(r = 2), f(p) = 3, f(\phi) = +\infty$$

where c, r, and p represent customer, peer, and provider links respectively, and ϵ is the signature of any trivial path. For policy-based routing with local preferences, the weight of some path P is defined as f(the first link of P). We see that this algebra contains no restrictions regarding valleys and and almost no restrictions regarding steps: $c \oplus p = c$ implies that a provider path can be pre-pended by a customer link resulting in a valley, while $r \oplus p = r$ and $c \oplus r = c$ imply that provider and peer paths can be pre-pended by peer and customer links respectively resulting in steps. This is because the total order $f(c) < f(r) < f(p)$ implies that this algebra satisfies the first point of the policy-based routing (see Table 1), which says that providers always prefer to send traffic via their customers or peers, while customers always prefer to send traffic via peers. Thus, this algebra captures the property that providers have incentive to participate in valleys by exchanging traffic with peers via their customers rather than through direct peer links. The only restriction on steps is that $r \oplus r = \phi$, which

captures the property that peers do not want to provide free transit from one peer directly to another peer. This algebra is not monotonic: $f((c \oplus r)) = f(c) = 1 < 2 = f(r)$, and, therefore, describes a path vector protocol that does not converge.

A simple modification of the $\oplus$ operator of the above algebra, while keeping everything else the same, yields a monotonic algebra:

$\oplus$	ϵ	c	r	p
c	c	c	ϕ	ϕ
r	r	r	ϕ	ϕ
p	p	p	p	p

simply by disallowing valleys and steps. It is easy to check that this algebra is monotonic, and, therefore, describes a path vector protocol that is guaranteed to converge. Because $f(\phi) = +\infty$, $c \oplus p = \phi$, $r \oplus p = \phi$ and $c \oplus r = \phi$ imply that a provider path cannot be pre-pended by a customer link, a provider path cannot be pre-pended by a peer link, a peer path cannot be pre-pended by a customer link respectively. We see that a real-life BGP implementation of these restrictions requires that extensions of provider paths to valleys and extensions of peer, customer, and provider paths to steps are disallowed (have *infinite weight*) by the protocol itself or be highly undesirable. In Section 4 we describe an extension to BGP that allows for steps and/or valleys to be used as back-up paths in cases of no other alternatives. This extension is formulated algebraically in Section 3.2.

3.2 IP Prefix Highjacking Attacks

Although in the previous section we provided a succinct formulation of the requirements for valley-free and step-free routing decisions that guarantee convergence of BGP, our formulation does not address the issue of valleys and steps in the face of malicious behavior. In fact, some instances of valleys and/or steps that are results of IP prefix highjacking attacks may have no effect on convergence at all. For instance, if at least one customer in Figure 4 is malicious in the sense that it is happy to draw traffic from its right peer, then convergence is guaranteed because this malicious customer will not withdraw its transit. Let us consider the attacks depicted in Figures 1, 2 and 3.

For robustness with respect to link failures, Internet nodes may engage in backup relationships [6]. We say that a *backup* path must contain at least one valley or at least one step, and we call a path that is free of valleys and steps a *primary* path. When there are no *primary* paths, it is reasonable to allow nodes to use *backup* paths that may contain steps and/or valleys, even though such paths are vulnerable to prefix highjacking attacks. The protocols that allow for use of *backup* paths must ensure that *backup* paths are used only when there are no available *primary* paths, and that paths with less vulnerable regions (steps and valleys) are preferred. While the former requirement is hard to enforce in real life and we discuss it in Section 7, we present an algebra that satisfies the latter requirement:

$$L = \{0\} \times \{c, r, p\},\ \Sigma = L \cup \{(0, \epsilon)\} \cup \mathbb{Z}^{+} \times \{b, b^c\},$$

$$W = \mathbb{N} \times \{0, 1, 2, 3, 4\}$$

$\preceq$ is s. t. W is lexicographically ordered based on the order $\leq$, $\oplus$ is defined as

$\oplus$	$(0,c)$	$(0,r)$	$(0,p)$	(x,b)	(x,b^c)
$(0,c)$	$(0,c)$	$(1,b^c)$	$(1,b^c)$	(x,b^c)	(x,b^c)
$(0,r)$	$(0,r)$	$(1,b)$	$(1,b)$	$(x+1,b)$	$(x+1,b)$
$(0,p)$	$(0,p)$	$(0,p)$	$(0,p)$	(x,b)	$(x+1,b)$

where left column represents labels and the first row represents signatures (the column for signature $(0, \epsilon)$ is the same as that for signature $(0, c)$ and is omitted), and f is defined as

$$f(0,\epsilon) = (0,0), \quad f(x,c) = (0,1), \quad f(x,r) = (0,2),$$
$$f(x,p) = (0,3), \quad f(x,b) = (x,4), \quad f(x,b^c) = (x,4),$$

where c, r, and p represent customer, peer, and provider links respectively, ϵ is the signature of any trivial path as in algebras described in Section 3.1.2, b represents a backup path that begins with a peer or provider link, and b^c a backup path that begins with a customer link. The first component of labels and signatures x represents the number of steps and valleys together that path contains. Both valleys and steps are allowed in this algebra, but *backup* paths have the least preference. For any two *backup* paths this algebra breaks ties by picking the one with less steps and valleys. It is easy to check that this algebra is monotonic, and, therefore, describes a path vector protocol that is guaranteed to converge. The lexicographic order $f(0,c) < f(0,r) < f(0,p) < f(x,b) = f(x,b^c) < +\infty$ for any x, implies that this algebra satisfies the first property of policy-based routing but that it does not satisfy all the other properties because it allows for *backup* paths (see Table 1). We see that a real-life BGP implementation of this algebra requires that extensions of provider paths to valleys and extensions of peer, customer, and provider paths to steps are recorded together with general route advertisements as part of the protocol so that nodes who obtain such advertisements can have a separate tables for *backup* paths ranked by the number of steps and valleys advertised paths contain from least to most. In Section 4 we describe an extension that implements this algebra on top of a secure BGP protocol [12] .

4. PROTOCOL

One bit, or a flag, for each new AS in the path is enough to prevent policy attacks described above. The key is to mark the advertisements as leaving towards a peer or towards a customer, thus allowing the subsequent receivers of the advertisement to check if the advertisement is following a valid path.

When an update is sent perform the following actions:

- When an update is sent to a provider - do nothing.
- When an update is sent to a peer - add a flag.
- When an update is sent to a customer - add a flag.

do nothing here means the default action where no flags are added but the update is sent.

When an update is received preform the following actions:

- When an update is received from a provider - perform default update processing.
- When an update is received from a customer, check if it is flagged. If yes, mark the path as invalid and use it for routing decision only if no other paths are available. Even if no other paths are available, do not forward the advertisement. If the update is not flagged, perform default update processing.
- When an update is received from a peer, check if it has been flagged by an AS that is not an immediate peer. If yes, mark the path as invalid and use it for routing devision only if no other paths are available. Even if no other paths are available, do not forward the advertisement. If no such flag is in the update, then perform default update processing.

The processing is most intricate when an update is received from a peer, because peers, according to a protocol, set the flags. Only

Figure 5: Policy attack prevention flag format. The entry is part of Route Attestation message in ATTEST BGP attribute.

when peer is forwarding an update that already has a flag is the peer considered to be in violation of the proposed protocol and the update is marked as invalid.

The flag, or rather a bit indicating the status of an update, needs to be included in the signed portion of the Secure BGP message. For each AS, the update traverses Secure BGP and adds a new optional, transitive path attribute called ATTEST. The ATTEST attribute must contain Route Attestation (RA) and may contain Address Attestation (AA). We propose to introduce *part code* 6 (0110) to ATTEST attribute that follows the format presented in Figure 5. Note that absence of such part in ATTEST attribute indicates that the AS did not flag the announcement. While, technically, we use more than one bit, in principle we could use only a single bit, but then we would break the backwards compatibility of Secure BGP protocol.

5. EVALUATION

In this section, we evaluate the effect of incremental deployment of our scheme on an Internet topology. We use a snapshot of the primary AS topology [1, 4] in our experiments. We first observe how vulnerable the Internet is to traffic hijacking and then show how useful an incremental implementation in reducing these attacks.

5.1 Experimental Setup

We wrote a simulator in Python for all our experiments. For reasons of scale, we use a snapshot from January 1998. We believe that this is representative and that the results are valid in today's Internet. Future experiments on later graphs are planned in order to confirm this hypothesis. A *customer* is any AS that has a customer relationship with any other AS. A *provider* is any AS that has a provider relationship with at least one other AS. In our experiments, we only look for the valley-path vulnerability, so any AS that has more than one provider is *potential attacker*. A *vulnerable host* is any AS that has a provider relationship with at least one potential attacker. Our threat model is explained in more detail in Section 2.

5.2 Results

Table 3 shows how vulnerable the Internet is to traffic hijacking even when we consider only valley-free property violations (not step violations). Notice that there are 694 providers in the Internet, and 520 of them (75%) are vulnerable to having their traffic hijacked. Of the customers, about 1433 (44%) are potential hijackers.

Number of providers	694
Number of customers	3235
Number of vulnerable providers	520
Number of potential attackers	1433

Table 3: Characteristics of the Topology used

Figure 6 shows how random deployment of our scheme reduces the number of vulnerable paths. By random deployment, we mean that a certain fraction of ASes was selected (one-by-one independently and without replacement) to deploy our scheme uniformly over the whole set of ASes. The total number of paths is all paths between a source-destination pair of providers. We use Gao's algorithm [8] to calculate the shortest valley-free path between all such source-destination pairs. A *vulnerable* path is one that traverses two vulnerable hosts. With random deployment, the curve starts going down sharply (*i.e*, we see benefit) when more than about 60% of providers deploy the scheme. This is because unless an AS with large out-degree does not deploy, effect is minimal. The error bars are significant because we take only 10 sample points for each trial due to the time required to run the simulation, and the out-degree of ASes are highly skewed. Thus, benefit of deploying our scheme for the entire network is significantly reduced if a very small number ($< 1\%$) of the highly connected ASes do not deploy.

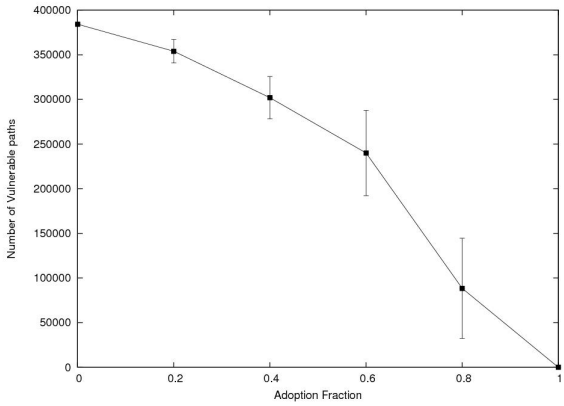

Figure 6: Effect of incremental deployment on vulnerable paths.

Figure 7 shows how effective the scheme would be if a only small fraction of the most connected providers adopt our scheme. Observe that if just 50 providers (7%) with the highest out-degree implement the scheme, the benefit to the entire network reaches the same level as that of 70% of random deployment. Thus, for our protocol to be effective, incremental deployment among the highly connected provider ASes is the most effictive and requires the least number of participating ASes. We emphasize that these numbers are only for valley violations. We expect them to be worse when we consider step violations. We intend to run experiments with more trials on newer topologies while considering step violations in the future.

6. RELATED WORK

In [9] Griffin and Wilfong present a formal model of BGP and show that static analysis with respect to BGP convergence of policies that are based on local-preferences and AS–path–length at-

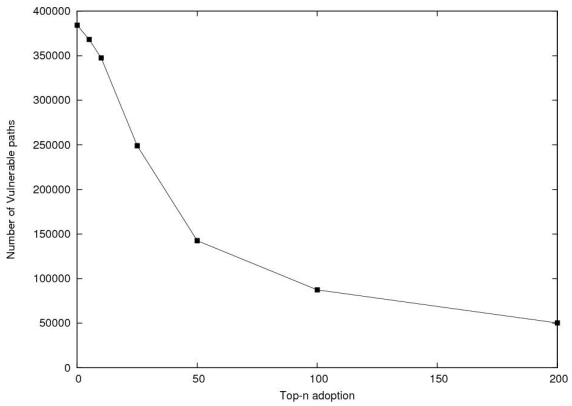

Figure 7: Effect of deployment among highly-connected ASes on vulnerable paths.

tributes is NP–complete, even when routing policies of each AS are known. In [10] Griffin and Wilfong present an abstract version of BGP where additional information regarding route history of oscillations is passed together with routing advertisements. The authors show that by suppressing routes with histories that contain cycles due to policy conflicts their generalization for BGP is guaranteed to converge when no network topology changes take place. The down side is that both of these works rely on knowledge of the AS topology and the set of routing policies for each AS. On the other hand Gao and Rexford propose a set of guidelines that allow for ASes to set flexible policies, while not requiring coordination with other ASes, such that route convergence even under changes in the topology and routing policies is guaranteed [7].

The aforementioned studies do not analyze routing protocols with backup paths (paths with steps but no valleys) that could be used to increase the reliability of the network under link and router failures. In [6] Gao et al. present a general model for backup routing (paths with steps allowed) that increases network reliability with respect to link failures while guaranteeing convergence. Sobrinho presents necessary and sufficient conditions for convergence of path vector protocols in terms of its algebraic formulation, and constructs algebras for valley-free policy-based routing with local preferences without and with backup paths [15]. Algebraic analysis of our protocol is built on these results.

To the best of our knowledge, the only study that discusses the issue of how policy violations may become security threats in the context of IP prefix highjacking was presented in [14]. It is shown in [14] that such attacks can be effective in attracting significant volumes of traffic, but no approach for resolving them is proposed. We present a protocol that is an extension of secure BGP for policy based-routing with local preferences with backup paths that include valleys and counts the number of vulnerabilities (steps and valleys that could be used for highjacking). Our protocol does not require global knowledge of the AS topology and local policies of each AS. We show that our protocol is guaranteed to converge by checking that its algebraic formulation satisfies the necessary convergence property as per [15].

Prefix hijacking using BGP protocols was studied by Ballani et. al in [3]. The authors outlined possible attacks on BGP protocol and documented number of prefix hijacking attacks in the Internet. In [18] Zheng et. al proposed a prefix hijacking detection system that utilizes multiple vantage points. Hu et. al in [11] described a different scheme for IP prefix hijacking detection, which involves both data plane traces from vantage points and an analysis of a BGP control plane. These works on prefix hijacking do not address the BGP policy attacks we described in section 2.

7. CONCLUSION

The BGP was not designed with security in mind and is susceptible to multiple attacks. Holistic security extensions to the BGP, such as Secure BGP, solve many issues with the protocol but still leave it vulnerable to policy-based attacks. In this paper, we characterized three types of major policy attacks and developed a lightweight and easy-to-implement extension to the Secure BGP protocol that prevents such attacks. We show that thus-modified Secure BGP protocol is guaranteed to converge even when steps and valleys are allowed (although greatly discouraged). With experiments over a real network topology, we show that partial deployment of our scheme can be effective in preventing possible threats of such attacks; for example, the number of vulnerable paths could be reduced by approximately 80% if only one hundred largest Internet networks would deploy our scheme.

REFERENCES

[1] Annotated AS graph. `http://www.cc.gatech.edu/~amogh/topology.html`.

[2] Kyoto, Japan, Aug. 2007.

[3] H. Ballani, P. Francis, and X. Zhang. A study of prefix hijacking and interception in the Internet. In *Proc. ACM SIGCOMM* [2].

[4] A. Dhamdhere and C. Dovrolis. Ten Years in the Evolution of the Internet Ecosystem. In *IMC '08: Proceedings of the 8th ACM SIGCOMM conference on Internet measurement*, Oct. 2008.

[5] L. Gao. On inferring automonous system relationships in the Internet. *IEEE/ACM Transactions on Networking*, 9(6):733–745, Dec. 2001.

[6] L. Gao, T. G. Griffin, and J. Rexford. Inherently safe backup routing with BGP. In *Proc. IEEE INFOCOM*, Anchorage, AK, Apr. 2001.

[7] L. Gao and J. Rexford. Stable Internet routing without global coordination. *IEEE/ACM Transactions on Networking*, pages 681–692, Dec. 2001.

[8] L. Gao and F. Wang. The Extent of AS Path Inflation by Routing Policies. In *GlobalInternet 2002*, 2002.

[9] T. Griffin and G. Wilfong. An analysis of BGP convergence properties. In *Proc. ACM SIGCOMM*, Cambridge, MA, Sept. 1999.

[10] T. Griffin and G. Wilfong. A safe path vector protocol. In *Proc. IEEE INFOCOM*, Tel-Aviv, Israel, Mar. 2000.

[11] X. Hu and Z. M. Mao. Accurate Real-time Identification of IP Prefix Hijacking. In *Proc. IEEE Symposium on Security and Privacy*, Oakland, CA, May 2007.

[12] S. Kent, C. Lynn, J. Mikkelson, and K. Seo. Secure border gateway protocol (S-BGP) - real world performance and deployment issues. In *Proc. NDSS 2000*, 2000.

[13] S. Kent, C. Lynn, and K. Seo. Secure border gateway protocol (S-BGP). *IEEE JSAC*, 18(4):582–592, Apr. 2000.

[14] P. H. S. Goldberg, M. Schapira and J. Rexford. How secure are secure interdomain routing protocols? In *ACM SIGCOMM 2010*, Aug. 2010.

[15] J. L. Sobrinho. Network routing with path vector protocols: Theory and applications. In *ACM SIGCOMM 2003*, pages 49–60, Aug. 2003.

[16] K. Varadhan, R. Govindan, and D. Estrin. Persistent route oscillations in inter-domain routing. *Computer Networks*, 32(1):1–16, 2000.

[17] R. White. Securing BGP through secure origin BGP. *The Internet Protocol Journal*, 6(3), Sept. 2003. `http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_6-3/ipj_6-3.pdf`.

[18] C. Zheng, L. Ji, D. Pei, J. Wang, and P. Francis. A Light-Weight Distributed Scheme for Detecting IP Prefix Hijacks in Realtime. In *Proc. ACM SIGCOMM* [2].