HC-BGP: A Light-weight and Flexible Scheme for Securing Prefix Ownership

Ying Zhang, Univ. of Michigan
Zheng Zhang, Purdue Univ.
Z. Morley Mao, Univ. of Michigan
Y. Charlie Hu, Purdue Univ.

Abstract

The Border Gateway Protocol (BGP) is a fundamental building block of the Internet infrastructure. However, due to the implicit trust assumption among networks, Internet routing remains quite vulnerable to various types of misconfiguration and attacks. Prefix hijacking is one such misbehavior where an attacker AS injects false routes to the Internet routing system that misleads victim's traffic to the attacker AS.

Previous secure routing proposals, e.g., *S-BGP, have relied on the global public key infrastructure (PKI), which creates deployment burdens. In this paper, we propose an efficient cryptographic mechanism,* HC-BGP*, using hash chains and regular public/private key pairs to ensure prefix ownership certificates.* HC-BGP *is computationally more* efficient *than previously proposed secure routing schemes, and it is also more* flexible *for supporting various traffic engineering goals. Our scheme can efficiently prevent common prefix hijacking attacks which announce routes with false origins, including both prefix and sub-prefix hijacking attacks.*

1 Introduction

With a rapidly growing number of critical applications deployed on the Internet today, including financial transactions and voice over IP, Internet security is of increasing concern. The interdomain routing protocol, BGP (Border Gateway Protocol), plays an important role in the Internet infrastructure. However, given the lack of security guarantee in its original design, BGP is vulnerable to various types of misconfiguration and attacks [5, 13].

One type of routing protocol attacks with severe impact is the prefix hijacking attack, which injects and propagates false routes to the Internet, potentially causing traffic to be redirected to the attacker networks. There are two general types of prefix hijacks [18, 4, 10]: injecting a bogus route with a false origin AS (*i.e.,* origin AS attacks) and injecting an incorrect route with a false AS path segment but a legitimate origin AS. There has been much evidence for the former attack [6, 12], which has caused serious damage on network availability but is relatively easy to detect. However, a special type of origin AS attack, *i.e.,* sub-prefix attack, is more difficult to detect and prevent. In such an attack, the attacker announces a more specific prefix than the original prefix, and the route to the sub-prefix is likely used by all the routers due to longest prefix matching. The recent incident of YouTube's sub-prefix hijacking [16] serves as a real-world example of the severity of such attacks.

A fundamental reason for all these real-world prefix hijacking attacks is that the current BGP system lacks any secure binding between a prefix and its owner. Many secure BGP protocols have been proposed, *e.g.,* S-BGP [19], SPV [11], KC-BGP [23]. However, none is widely deployed primarily due to two deployment obstacles: a lack of the global PKI infrastructure and the high computational overhead of processing BGP updates. The seminal work in this area, S-BGP, requires two PKIs, one for address ownership attestation and one for route announcement attestation. In S-BGP, the prefix owner has an asymmetric private key for each prefix, generated by a global trust entity. Each AS along a path will verify that the prefix actually belongs to the AS with the corresponding public key. To ensure that the route cannot be tampered with by malicious ASes, each AS signs the update with its private key. Other secure BGP work, *e.g.,* SPV [11], KC-BGP [23], and path stability based improvement [7], have all focused on the performance improvement of the second phase – generating/verifying route attestation. SoBGP reduces overhead by only providing ownership authentication with private key *Authorization Certificate* assigned by a global PKI.

To our best knowledge, no existing work attempted to improve the efficiency and flexibility of the first phase – certifying prefix origin/ownership, which is also critical to the efficient operation of any secure BGP protocol. In this paper, we propose a novel scheme based on hash chains and regular public/private key pairs of two neighboring routers overcoming deployment hurdles in providing prefix origin/ownership attestation. HC-BGP is a complimentary effort to all above proposals in the second phase.

The key idea of HC-BGP is inspired by an empirical study we conducted to understand the most common routing dynamics. By analyzing three months of routing updates archived in RouteViews from 21 vantage points, we observe that most routing updates do not involve origin AS changes. To optimize for the common case, we propose to use the *one-way hash chain mechanism* to secure the prefix ownership in a light-weight fashion. HC-BGP guarantees that the permission to announce any sub-prefix is granted by the owner of the cover-prefix. Thus, our scheme not only secures the relevant prefix, but also secures its sub-prefix space, *i.e.,* all its possible sub-prefixes, effectively defending against the stealthy sub-

prefix hijacking attacks with false origin. For the rare cases of announcing new (sub)prefixes, an initialization step is performed to bootstrap the hash chain, which involves a slightly more computationally expensive asymmetric-key algorithm. The initialization is optimized to only require public key exchange limited to two neighboring ASes instead of a global PKI.

To prevent tampering with the initialization process and the replay attack during the subsequent route propagation, we exploit the Internet hierarchical structure and the business relationship among ASes to impose a partial ordering on the route propagation. We prove that no replay attack can succeed if every AS follows the guideline. Note that the partial ordering does not change the BGP route selection when preventing the replay attack.

In summary, HC-BGP is a light-weight approach to provide prefix ownership security without a global PKI. Moreover, its design to connect the sub-prefix announcement with cover-prefix ownership is effective in preventing sub-prefix hijacking attacks. Furthermore, using our scheme, network operators can still enjoy the flexibility of common routing practices such as multi-homing, prefix aggregation, and de-aggregation.

The paper is organized as follows. We give the background of prefix hijacking and an overview of proposed protocol in §2. We conduct empirical data analysis in §3. We present the HC-BGP protocol in §4, §5, and §7. We finally evaluate its performance in §8 and §9.

2 Background and Overview

Prefix hijacking is a serious BGP security threat by which attackers steal IP addresses belonging to other networks. The attacker AS injects false routes into the global routing table by announcing another network's prefix.The stolen address blocks can be used for other malicious activities such as spamming or denial-of-service attacks where the attacker's identity is concealed. The fundamental problem that accounts for this attack is a lack of prefix ownership authentication in the BGP system. In the following, we first review several types of prefix hijacking attacks. We then describe our threat model, followed by our design requirements and comparison to existing work.

2.1 Prefix hijacking attacks with false origin

IP prefix hijacking can be carried out in several ways [10, 24, 4]. The attacker can either blackhole the victim's traffic by advertising an invalid route with a false origin, or intercept the traffic by inserting a false nexthop, possibly leaving the origin AS unchanged. In this paper, we focus on the attack with false origin which is due to a lack of address certification and usually results in severe consequence such as reachability problems. Interception-based hijack can be partially addressed by encryption based solutions [21]. The false-origin prefix hijacking can be further categorized based on size of the address block announced. Below are the two categories, as illustrated in Figure 1.

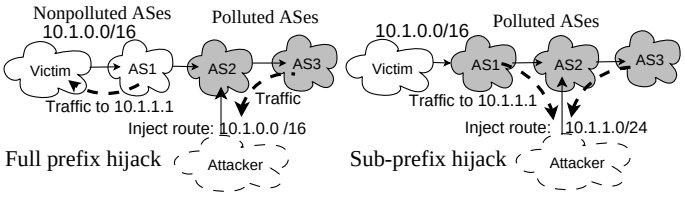

Fig. 1. Prefix hijacking attacks.

Full prefix hijack, where the attacker announces exactly the same prefix already announced by the victim. Other networks will select one such route to adopt. In this case, the Internet is partially polluted. For example, in Figure 1(a), both the attacker and the victim announce the same prefix $10.1.0.0/16$. Consequently, $AS2$ and $AS3$ may prefer the attacker's route because of the shorter AS path.

Sub-prefix hijack, where the attacker announces only a subnet of a prefix announced by the victim and this subnet is not announced previously. Unless filtered, this new sub-prefix is injected into the forwarding table regardless of the route of the existing prefix. Due to the longest prefix matching policy, traffic destined to this subnet follows the false route of the sub-prefix. Therefore, most of the Internet is likely polluted. In Figure 1(b), the attacker announces the sub-prefix $10.1.1.0/24$. This route is likely accepted by all other ASes as a new forwarding table entry. Hence, traffic destined to IPs such as $10.1.1.1$ is misled to the attacker.

There have been several real-world examples of prefix hijacking. The most recent hijacking attack, however, pollutes most of the Internet: sub-prefix hijacking attack on the YouTube prefix [16]. On February 24, 2008 around 18:50, Pakistan Telecom (AS17557) announced $208.65.153.0/24$ to hijack YouTube (AS36561)'s prefix $208.65.152.0/22$. Because it was a new prefix, most ASes adopted it. YouTube was blackholed for almost two hours. At around 20:07, YouTube also began to announce $208.65.153.0/24$ to reverse the effect. Until 21:10, Pakistan Telecom's provider PCCW Global (AS3491) started to withdraw the false route. The incident caused severe reachability problems for almost all YouTube users. Accordingly, the scope of affected networks is much larger than previous hijacking incidents.

2.2 Threat model

We describe the threat model. Given a network G, each AS is assigned a set of IP prefixes. Each AS can only announce prefixes it owns. A prefix p is the set of IP addresses announced in a single routing announcement. A *sub-prefix* p' of p is a subset of the addresses in p, *i.e.*, $p' \subset p$, where p as p''s *cover-prefix*. Among all the sub-prefixes of p, we define the largest sub-prefix p' as the *direct sub-prefix* of p, *i.e.*, if $\neg\exists p_2, p' \subset p_2 \subset p$, then p' is the direct sub-prefix of p and p is the *direct cover-prefix*. In subsequent discussions, we refer *cover-prefix* as *direct cover-prefix*.

We first define the attacks of interest, prefix hijacking attacks with false origin. If p belongs to AS_1, then any other AS_x announcing p with AS_x as the origin AS, is considered

as *hijacking with false origin*. This type of attack has two sub-categories: 1) An AS can advertise a prefix originated from another AS. 2) An AS can advertise any subset of another prefix originated from another AS. We note these two categories cover all known hijacking incidents in the past.

After AS_x hijacks p, in most cases, it blackholes a portion of all the traffic destined to AS_1. But it can also tunnel the traffic back to AS_1 to be more stealthy to carry out an interception attack [4]. We also consider this attack type. Note that we do not consider any attacks modifying other parts of the AS path, *e.g.*, modifying path (AS_x, AS_2, AS_1) to be (AS_x, AS_1).

2.3 Solution requirements

A practical protocol that can prevent the above attacks should satisfy the following requirements.

1. Ensuring Origin Attestation: It should prevent both full prefix and sub-prefix false-origin hijacking. If an attacker advertises the prefix currently announced by its owner, or a new sub-prefix covered by a larger prefix owned by other networks, the route to the attacker should be discarded.

2. Flexible: It should support the flexibility for legitimate multiple origin AS (MOAS) and traffic engineering. One prefix can be announced by two ASes for several legitimate reasons [25]: the prefix of the exchange points is usually announced by more than one AS connected at the exchange point; a small customer without its own AS number may use a private AS number to multi-home to two providers which announce the prefix simultaneously. Our solution needs to accommodate these dynamics.

For traffic engineering purposes, one prefix could be de-aggregated to multiple sub-prefixes announced independently of the prefix. On the other hand, the owner can also aggregate a set of sub-prefixes to a single large prefix to limit routing table size. Our solution also needs to provide flexibility for aggregation/de-aggregation operations.

3. Incrementally deployable: Like all other secure BGP protocols, it is impossible to force all ASes to adopt the new protocol simultaneously. The adoptability of a protocol highly depends on its incremental benefit [8]. The new protocol needs to support incremental deployment to provide enough incentives for ISPs to deploy it.

4. Light-weight: A major concern for previously proposed secure BGP protocols is the high overhead for both computation and storage. To ensure practical adoptability, we need to design a solution with low overhead.

The proposed HC-BGP satisfies all four requirements.

2.4 Comparison with previous secure BGP protocols

Several protocols have been proposed to enhance BGP security by incorporating cryptographic mechanisms to provide confidentiality, integrity, and origin authentication. S-BGP [19] is the first comprehensive secure routing protocol. It relies on two public key infrastructures (PKIs) to secure AS identity and association between networks and ASes. Each route contains two attestations (digitally signed signatures), one for the origin authentication and one for the route integrity. In reality, due to the large number of sign and verify operations, S-BGP is too costly to deploy.

Most of the followup work to S-BGP focus on reducing the computational complexity of the second security properties of S-BGP, *i.e.*, generating/verifying the route attestation. For example, SPV [11] utilizes purely symmetric cryptographic primitives, a set of one-time signatures, to improve efficiency. Butler *et al.* [7] reduced the complexity of S-BGP by exploring path stability. Along the angle of reducing the overhead of asymmetric key, Hen *et al.* [23] proposed a scheme using key chain to improve its performance. Only one existing work, Secure Origin BGP (soBGP) [17], focuses on providing address attestation. However, soBGP still uses one PKI to authenticate the address ownership and AS identity. Each soBGP router first builds a topology database securely, including the address ownership, organization relationship and topology. Aiello *et al.* builds an address ownership proof system [3] which still uses a centralized infrastructure requiring gathering address delegation information.

In summary, all previous secure BGP protocols leave the prefix ownership assurance unchanged. S-BGP relies on the address attestation with the assistance from the centralized trust entity ICANN. SPV proposes using identity based cryptography (IBC) to make the public key distribution more flexible. However, these address attestation methods suffer from: 1) dependence on a global PKI; 2) the need for a verification operation for each routing update; 3) inflexibility for origin AS changes; 4) inability to handle subprefix hijacking.

In contrast, HC-BGP is a new approach to prefix ownership authentication that is both more efficient and flexible than previous approaches. Our scheme uses the light-weight hash chain mechanism and the less frequent cryptographic operations, and hence is efficient in terms of computational complexity and memory consumption. Our scheme does not rely on a global PKI, and hence it provides much more flexibility for traffic engineering in terms of origin AS changes, address allocation/de-allocation.

3 Hypothesis and empirical analysis

Our goal is to design a practical and efficient solution to secure prefix ownership. Towards this goal, we first seek to gain insights into several key characteristics of the prefixes announced in the Internet. We investigate two hypotheses that directly relate to the design of an efficient secure protocol:

Hypothesis 1: For each prefix, the set of its origin ASes is quite stable. This property directly affects frequency for updating the secure association.

Hypothesis 2: The aggregation/de-aggregation dynamics for each prefix is infrequent. This property relates to the association changes across prefixes.

To analyze these two hypotheses, we perform an in-depth analysis of the dynamics of the origin changes as well as the distribution of the prefixes/sub-prefixes.

We study these aspects using three months of BGP data,

from Dec. 2007 to Feb. 2008, from all vantage points in RouteViews [2]. First, to support the flexibility of MOAS, we studied the frequency and stability of such changes. We found that across the entire three months, only 1935 (0.9%) prefixes had MOAS behavior. Among them, 97% had only 2 origin ASes, as shown in Figure 2. Across the entire three months, we only observe 52 prefixes announced by new origin ASes. This means that the set of origin ASes for each MOAS prefix is quite stable.

Second, we needed to provide the flexibility of the coexistence of the prefix and sub-prefix. We found that among the total of 214,043 prefixes in the global routing table, only 8% (17115) had sub-prefixes. Among them, though some prefixes had many sub-prefixes, we found that 90% had less than 10 direct sub-prefixes, as shown in Figure 3.

Finally, we analyzed the dynamics of aggregation/de-aggregation. In Figure 4, each *Aggregation* event refers to when one sub-prefix is withdrawn, and its cover-prefix still exists, and each *De-aggregation* event refers to when when a new sub-prefix is announced. The duration of aggregation is the time period when only the cover-prefix exists, and the duration of de-aggregation is the period when only the sub-prefix exists. The figure shows aggregation and de-aggregation occur rarely. The short durations of less than 10 minutes are most likely due to BGP convergence.

From the above analysis, we draw the following observations confirming the two hypotheses above which are then exploited in the design of HC-BGP.

1. Only a few prefixes have more than one origin AS. Among them, most have only two origin ASes.
2. The set of origin ASes for MOAS prefixes is stable.
3. Majority of the prefixes do not have sub-prefixes.
4. Neither prefix aggregation/de-aggregation nor prefix origin changes are often.

A key observation that guides our HC-BGP protocol design is that most of the proposed secure BGP protocols require origin authentication upon *any* UPDATE message for this prefix. In contrast, because our problem is to defend against the fraudulent origin (sub)prefix hijacking, we only need to authenticate the binding between a given origin AS and the prefix, which is thus, *only needed when the binding changes*. Guided by this observation, we can design a much more efficient protocol to secure this binding.

4 HC-BGP Protocol

We now present HC-BGP, a new secure BGP protocol that prevents both full prefix and sub-prefix hijacking attacks. We first describe the one-way hash chain building block, our trust model, followed by the protocol.

4.1 One-way hash chains

One-way hash chain is a widely used *light-weight* cryptographic method to provide security. It was first proposed by Lamport [14] for password protection. Using a cryptographic hash function $h(s)$, a client first needs to use other security methods to notify the server of the initial value $h^n(s)$. For subsequent communication, the client only needs to present $k = h^{n-1}(s)$ to the server. The server computes $h(k) = h(h^{n-1}(s))$ to compare with the pre-stored $h^n(s)$. If they match, the server verifies the client identity.

Table 1. Terminology

$h()$	hash chain function
p	a prefix
$\hat{p}$	the direct cover-prefix of p
s_p	the secret of prefix p
$h^n(s_p)$	the initial value of hash chain for prefix p
$h^c(s_p)$	the current value of hash chain for prefix p
k_i^-	router R_i's private key
k_i^+	router R_i's public key

Hash chain has the following key properties. It is useful in cases where an authentication is done once, all the following values for subsequent authentication can be derived efficiently by repeatedly computing the hash function value. The other end can easily verify each new value. The one-way hash function guarantees that given a value $h^i(s)$, it is computationally infeasible for the attacker to derive the secret s. Moreover, many hash functions can provide the second pre-image collision resistant: it is impossible to find another s' such that $h(s') = h^{i+1}(s)$.

HC-BGP uses hash chains to secure the binding between the origin AS and the prefix *efficiently* in all subsequent UPDATE messages once the first binding is authenticated.

4.2 Trust model

As discussed in §2.4, all previous secure BGP protocols require a global PKI to provide authenticity. Every AS in the Internet must establish a trust relationship with this global entity and depend on it for any changes. Each AS needs to hold all other ASes' keys. Especially, for address attestation, all existing secure BGP methods rely on the PKI. This has imposed significant management burden, and has received a fair share of criticism [20].

Unlike previous work, we rely on the relationship between every pair of *neighboring ASes* to provide authenticity. We do not require each AS to know all other ASes' key. Instead, each AS establishes a trust relationship with its directly connected neighbors: if AS A and AS B are neighbors, A and B trusts each other, denoted as $A \leftrightarrow B$. This pair-wise trust model is consistent with the current commercial relationship established between ASes. Thus, the trust is easy to establish in practice; for instance, the keys between two entities can be exchanged in the contract. The two tnetities can even decide to use either symmetric key or asymmetric key flexibly themselves.

In the context of prefix hijacking, we assume that each AS trusts its provider which does not have incentives to hijack customers' addresses as customer traffic will always traverses its network. Moreover, we assume all Tier-1 ASes are trustworthy as they are large ISPs with careful network management. Historically, there has been no known events of any Tier-1 AS launching hijacking. Note that the trust relationship is established at the AS level. We assume all the routers

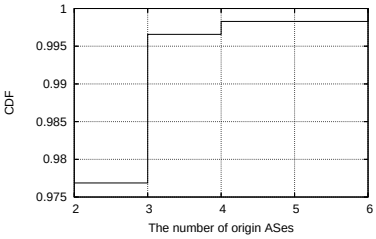

Fig. 2. Origin ASes distribution

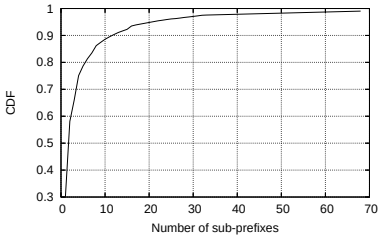

Fig. 3. Number of direct sub-prefixes.

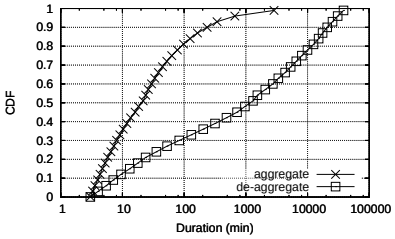

Fig. 4. Aggregation/de-aggregation duration.

Table 2. The protocol at the prefix owner R_i.

1. Hash-chain Initialization
for each neighbor of R_i, send $(h^n(s_p))_{k_i^-}$
2. Prefix announcement
2.1 Announce new prefix p
if p does not have any cover-prefix
announce (p, $h^{n-1}(s_p)$)
else
find p's direct cover-prefixes $\hat{p}$, send announcement:
(p, $h^{n-1}(s_p)$, $h^{c_{\hat{p}}-1}(s_{\hat{p}})$)
endif
2.2 Announce prefix p with new origin AS
R_i announces prefix p with (p, $h^{c-1}(s_p)$) and the new origin AS
3. Withdrawal
To withdraw prefix p and its latest hash chain value $h^c(s_p)$;
send withdrawal: (p, $h^{c-1}(s_p)$)

Table 3. The protocol at receiver R_{i+1}.

1. Hash-chain Initialization
Receive the initial value for prefix p from neighbor R_i,
Verify and store $((h^n(s_p))_{k_i^-})_{k_i^+}$.
Sign and send $(h^n(s_p))_{k_{i+1}^-}$. Store $h^n(s_p)$.
2. Prefix announcement
Receive an announcement for prefix p with $h^c(s_p)$.
if p exists in routing table and origin AS does not change
Accept.
elsif p exists, origin AS changes and $h(h^c(s_p)) == stored_p$
Accept. Store $h^c(s_p)$ in the routing table.
elsif p is a new sub-prefix of existing $\hat{p}$ and with same origin,
Accept. Store $h^{n-1}(s_p)$ in the routing table.
elsif p is a new sub-prefix of existing $\hat{p}$,
with different origin ASes,and $h(h^c(s_{\hat{p}})) == stored_{\hat{p}}$
Accept. Store $h^c(s_{\hat{p}})$ and $h^{n-1}(s_p)$ in routing table.
else Reject.
3. Withdrawal
Receive a withdrawal for prefix p with $h^c(s_p)$.
if withdrawal is announced with updated $h^c(s_p)$
if $h(h^c(s_p)) == stored_p$
Accept. Store $h^c(s_p)$ in the routing table.
else Reject.
elsif withdrawal is with old hash value
Accept.

within one AS behave consistently as it is easier to manage inside the ISP network.

Finally, if one prefix has multiple providers as origin ASes, we assume that the prefix owner will communicate with all these providers to ensure the consistency of the hash value.

4.3 The basic protocol

The key idea of our protocol is to exploit the light-weight security property of one-way hash chain to verify the association between the origin and the announced prefixes in any route advertisement. First, the AS who decides to deploy this scheme needs to propagate the hash chain's initial value to other ASes in a secure manner. After the initialization process, the origin AS starts announcing the prefix with its latest hash value. Each receiver then verifies if the route is from the previously authenticated origin via the attached hash value. The origin AS needs to update the hash value only in one of the following three cases: (1) the prefix's origin AS changes, (2) any of its sub-prefixes is newly announced, (3) the prefix is completely withdrawn. Note that normal withdrawals and announcements caused by an intermediate AS switching between alternative paths reuse the same hash value, as such changes do not involve the origin changes.

The overall protocol for prefix owner R_i is sketched in Table 2. For any update receiver R_{i+1}, the verification process is shown in Table 3 with the terminology defined in Table 1. We now introduce each step of the protocol.

1. Hash-chain initialization. Each prefix has an initial value. The origin AS needs to propagate the initial value to other ASes securely. We assume that neighboring eBGP peering routers can easily exchange their keys, using an out-of-band mechanism needed for establishing the peering session. The initial values are propagated hop-by-hop between participating neighboring routers. For each prefix owned by AS R_i, R_i first assigns the prefix an initial hash chain value, which is then propagated to its neighbors encrypted by k_i^-. Its neighboring router R_{i+1} first decrypts using R_i's public key, stores the hash value, encrypts it with its own key, and then propagates the hash value further. This is a one-time overhead for each originating prefix. When the hash chain value is exhausted, the initialization is performed again.

Note that in the above protocol, any malicious AS along the propagation path can modify the initialization value. We impose a partial ordering to limit the propagation of the modified initial value to a very small range. More precisely, we can eliminate any pollution except for attacker's customers

with all other ASes guaranteed to be safe assuming full deployment. The details are described in §5 with other corner cases, *e.g.*, message loss, discussed in §7.

2. Prefix announcement. If the prefix is announced with the same origin AS as the previous update, or the sub-prefix is announced with the same origin AS as its cover-prefix, there is no need to update the hash chain. The hash chain value however is updated in the following two scenarios.

2.1 Origin AS changes. The prefix owner may modify its origin AS by for instance multi-homing to several providers for load balancing. In this case, we rely on the prefix owner to coordinate a consistent hash value among the origin ASes. It needs to inform the new origin AS with the latest hash chain value. When a new origin AS R_i starts to announce a prefix p, it needs to update the hash chain value to $h^{c-1}(s_p)$. Note that this is only performed when the origin AS changes.

2.2 Sub-prefix announcement. Whenever a sub-prefix is initially announced, it should be attached with not only its own hash value, but also its cover-prefix's updated hash value, indicating that the cover-prefix's origin AS has authenticated the announcement of the sub-prefix.

When R_i announces prefix p's sub-prefix p_1, it undertakes the following actions. It first propagates the initial hash value $h^n(s_{p_1})$ (step 1) and then announces the sub-prefix: $p_1, h^{n-1}(s_{p_1}), h^{c-1}(s_{\hat{p}})$. The receiver validates $h^{c-1}(s_{\hat{p}})$ with $h^c(s_{\hat{p}})$ to check this announcement is authorized by the cover-prefix p. The receiver then compares $h^{n-1}(s_{p_1})$ with $h^n(s_{p_1})$ to verify the hash value of sub-prefix p_1. This announcement ensures a connection between the cover-prefix and sub-prefix, effectively preventing sub-prefix hijacking.

3. Withdrawal. Withdrawals occur quite frequently on the Internet due to transient failures or routing convergence. Most withdrawals are caused by transient failures along the path. For these transient withdrawals, no hash chain update is required because transient failures are not caused nor controlled by the origin ASes.

Under one scenario the hash value needs to be updated. When the origin AS R_i decides to withdraw an existing prefix p for the long term, *e.g.*, due to aggregation or changing provider, it needs to include the updated hash value. This is to prevent an attacker from announcing p with R_i as the origin AS and the old hash value after the long-term withdrawal.

If the hash chain value in the withdrawal is the same as the latest one, then the receiver treats it as normal withdrawal and accepts it. Otherwise, if the withdrawal is sent together with an updated hash value, suggesting that the prefix origin withdraws route to this prefix. The receiver accepts the withdrawal only if the hash chain value matches. Once the prefix is withdrawn, the receiver does not accept any announcement for this prefix with an old hash chain value.

Figure 5 illustrates how HC-BGP prevents hijacking introduced previously. For both full prefix and sub-prefix hijacking, AS 2 discards the false route from the attacker because $h^{999}(s_2)$ does not match the stored hash value.

We now briefly analyze the computational complexity of HC-BGP. Detailed analysis is presented in §9. HC-BGP introduces two sets of computational overhead: asymmetric key based initialization and hash chain based validation. Fortunately, unlike previous schemes, both operations are rarely performed. The initialization, an expensive operation, is only performed in two cases: when an AS begins to deploy this scheme or when a new sub-prefix is announced. This is only a one-time cost. The hash-chain value generation/verification, which is light-weight, is only performed when a prefix's origin AS(es) changes, a new sub-prefix is announced or withdrawn.

5 Advanced protocol: partial ordering in HC-BGP announcement

The basic protocol has two security holes: 1) during initialization, an attacker can modify the initial value received from its neighbor. 2) during the propagation of hash chain value updates, an attacker can tamper the value. It can also replay the value by announcing the prefix as originated from its own AS. In the following, we present a partial ordering algorithm that can effectively prevent these cases. We use the replay attack as an example. The algorithm can also be applied to prevent attacks during initialization.

The replay attack is illustrated in Figure 6(a). Prefix p's actual owner AS AS_0 starts to announce prefix p with hash value $h^i(s)$. It announces to its provider AS_1, which then propagates to its customer. If one of AS_1's customer M is malicious, it hijacks the prefix by announcing p to its multi-homed provider AS_3 with $h^i(s)$, upon receiving $h^i(s)$ from AS_1. To avoid conflicting origin ASes, M may choose not to announce $h^i(s)$ to AS_1. In this case, M successfully pollutes a set of ASes connected to AS_3.

To prevent such attacks, we explore the fact that there exists a certain delay between the time when AS_0 announces $h^i(s)$ and when M learns it. The key idea of our solution is to amplify such delay to ensure that $h^i(s)$ from AS_0 is propagated to the majority of the Internet before M learns it. This is achieved by exploiting the Internet hierarchical structure to impose a partial ordering on the hash chain propagation throughout the Internet. We prove that by following these guidelines, the BGP system reaches a secure state, such that replay attack is impossible.

We first introduce the notation used. An update for prefix p and its hash value $h^i(s)$ is denoted as U_p. p's actual owner is AS_0. All Tier-1 ASes in the Internet form a set $\{Tier-1\}$. The commercial relationship between ASes includes customer-provider and peer-peer. The valid routing policy determines the AS path to be of the form of *Customer-Provider* Peer-Peer? Provider-Customer** (known as the *AS path "valley-free" rule* [9]), where "*" denotes zero or more occurrence of such an AS edge and "?" at most one occurrence. The propagation of U_p should follow guidelines below.

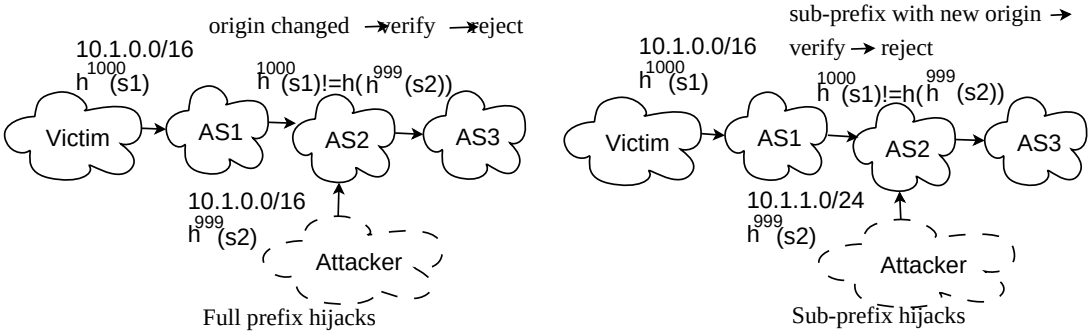

Fig. 5. Hash chain based prefix hijacking prevention.

Guideline: (Phase-1 and 2 may proceed concurrently.)

1. In Phase-1, every AS only sends U_p to its provider.
2. In Phase-2, the Tier-1 ASes who have U_p send U_p to their Tier-1 peers.
3. In Phase-3, if AS_i has providers or Tier-1 peers, AS_i waits until receiving U_p from any of the providers or Tier-1 ASes. Then, AS_i sends U_p to *all* its customers and peers. Each AS stores the $h^i(s)$ in the U_p from its provider or Tier-1 peers.

Intuitively, the guidelines enforce the route propagation order following the trust relationship. Note that we do not limit or change any route selection in the guideline. First, we do not restrict any route propagation but only impose some delay. Second, we do not change the route selection policy. In Phase-3, AS_i first determines the valid $h^i(s)$ from its provider and Tier-1 peers, which it trusts. AS_i can still select the best route among all routes with valid $h^i(s)$. We now prove the security guarantee of the guideline.

Theorem 5.1 *If all the ASes in the BGP system follow the guideline, then every AS only accepts U_p originating from AS_0, except the attacker and its customers.*

We prove the theorem using three lemmas.

Lemma 1 *In the Internet topology, for any AS_i, there exists a path in the form of* [(Customer → Provider)* (Peer→ Peer)?] *from AS_i to one of the Tier-1 ASes.*

Proof: If $AS_i \in \{Tier-1\}$, Lemma 1 holds. Otherwise, starting from AS_0, by traversing through all the *Customer→Provider* edges, we obtain a DAG D. Let $Top(D)$ denotes the set of nodes in D without outgoing degree (without provider). If $Top(D) \bigcap \{Tier-1\} \neq \emptyset$, Lemma 1 holds.

If $Top(D) \bigcap \{Tier-1\} = \emptyset$, let's examine the hypothesis that there is no *Peer-Peer* edge between these two sets. By definition, neither $Top(D)$ nor $\{Tier-1\}$ have providers. Thus there cannot be any *Customer↔ Provider* edges between them. If there is no edge between them, then a route originated from $\{Tier-1\}$ can only go through *Provider→ Customer→ Provider/Peer* to $Top(D)$, which violates the "valley-free" rule. Thus, the hypothesis is rejected and Lemma 1 holds. ■

Lemma 2 *Every AS receives U_p. The reachability is not affected.*

Proof: According to the valley-free policy, the route is propagated in one of the following ways: (1) *customer→ AS → provider*, (2) *customer→ AS → peer*, (3) *customer→ AS → customer*, (4) *peer→ AS → customer*, (5) *provider→ AS → customer*. Category 1 is allowed in Phase-1. Category 2-5 are allowed in Phase-3 with some delay. Note that in Phase-3, each AS sends the route to *all* its customers and peers even if the route is learned from this customer. Therefore, every AS waits for finite time unless it is disconnected from all its providers. Thus, the guidelines do not disallow any route propagation. The reachability is not affected. ■

Lemma 3 *Every AS only accepts U_p from AS_0. The malicious route is not be accepted by any other networks except the attacker and its direct customers.*

Proof: We prove this recursively. In Phase-1, the owner AS_0 has the initial U_p. By traversing all the *Customer→Provider* edges, we obtain a DAG D. If AS_i is in D, it learns the route from its customer AS_{i-1} in Phase-1. Assuming providers do not have incentives to hijack customers' route, $U_p^{AS_i} = U_p^{AS_{i-1}} = U_p^{AS_0}$.

If AS_i is in $\{Tier-1\}$, there are two cases. If there is a direct edge, then $U_p^{AS_i} = U_p^D = U_p^{AS_0}$. Otherwise we can find the AS_{i-1} which learns the route via a peering edge from D (Lemma 1). Assuming that Tier-1 ASes are trustworthy, $U_p^{AS_i} = U_p^{AS_{i-1}} = U_p^{AS_0}$.

Finally, if AS_i is not in $\{\{Tier-1\} \bigcup D\}$, according to guideline (3), AS_i only accepts route from its provider. Traversing the Customer→Provider directed edge from AS_i, we obtain a DAG D'. Because of Lemma 1, $U_p^{Top(D')} = U_p^{\{Tier-1\}} = U_p^{AS_0}$. Thus, $U_p^{AS_i} = U_p^{AS_0}$. ■

We use this guideline to provide security to both hash chain-based route updates and the hash value initialization process. Similar to the example in Figure 6(a), when a prefix owner starts to initialize the hash value $h^n(s)$, the malicious AS M can modify the $h^n(s)$ to $h^n(s')$. However, if all ASes follow the guideline, any AS except M accepts the correct $h^n(s)$. Figure 6(b) shows that AS_1 only sends $h^i(s)$ to M after AS_4 has learned the correct value. According to guideline (3), AS_3 trusts the value from AS_4 instead of M. Thus, the attacker cannot pollute any other ASes by replaying $h^i(s)$.

The guideline imposes extra route propagation delay to the BGP system. We argue this is not a serious issue for the following two reasons. First, the delay is only imposed when the prefix origin changes or prefix announcement/withdrawal. These events do not occur frequently. Second, usually the

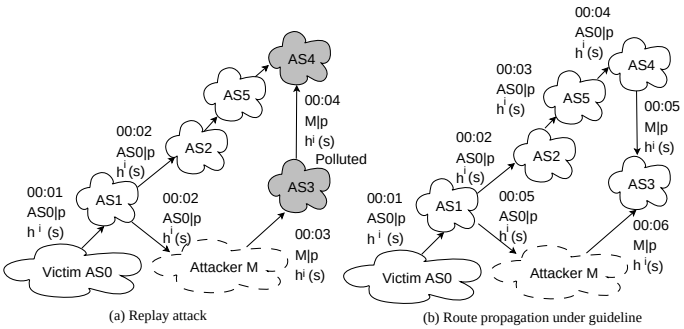

Fig. 6. Replay attack example and prevention.

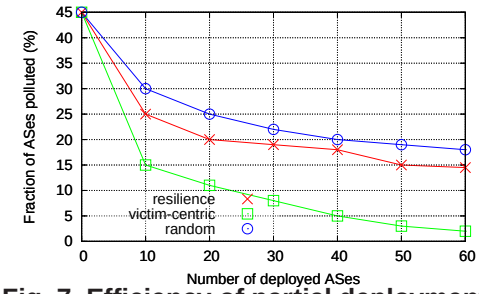

Fig. 7. Efficiency of partial deployments.

AS-level path between any AS and the Tier-1 ASes is only 3-4 hops long. The additional delay is only proportional to the hierarchical level of the Internet topology.

6 Incremental deployment

In §5 we proved that, by imposing partial ordering, the entire BGP system reaches a secure state, assuming full deployment. However, incremental deployability is an important property for practical adoption. We enhance HC-BGP under partial deployment by two additional rules.

Constructing monitoring barrier by participating ASes. The current design of HC-BGP under partial deployment is not guaranteed to be secure because the legacy ASes might send the hash value to attacker before the legitimate route propagates to the Tier-1 ASes. In the example shown in Figure 6, if AS_1 does not deploy HC-BGP, then M receives the update via AS_1 at time 00:02 to pollute AS_3 and AS_4. In this case, the value is leaked from AS_1 to M.

To overcome this problem, we develop one more rule that uses all participating ASes to monitor and prevent leaking hash value to attacker from their legacy neighbors. We first assume that each AS knows whether its neighbor has deployed HC-BGP. Specifically, when AS_i receives an updated route from its non-provider legacy neighbor AS_{i+1}, AS_i will first check if the updated value has propagated to Tier-1 ASes based on guideline (3). If not, AS_i delays propagating the route from AS_{i+1} until the value is propagated to Tier-1 ASes. Thus, all the participating ASes like AS_i construct a *monitoring barrier* which effectively delays the propagation of updated hash value to the attacker. This rule enables the deployed ASes to stop leaking the hash value through the legacy ASes to the attacker.

Conflict detection. Even with monitoring barrier, we cannot guarantee all ASes are free from pollution under partial deployment. Thus, we rely on participating ASes to detect conflicting hash values. In the same example in Figure 6, AS_4 receives two inconsistent routes, *i.e.*, two different origin ASes announcing p with the same $h^i(s)$, indicating the occurrence of hijacking. There are multiple ways that AS_4 can identify the attacker AS using the knowledge of the topology. For example, the presence of a link between AS_1 and M but no link between AS_3 and AS_0 indicates that M learns the message from AS_1. We leave details of such a scheme as future work.

7 Discussion

We discuss two corner cases and their solutions.

Resilience to message loss. Lost messages could cause inconsistency in HC-BGP. Since BGP uses the reliable transport protocol TCP, BGP packet loss is unlikely to happen. However, routers can go down temporarily due to maintenance or failure. If any message, *i.e.*, both initialization and hash chain update, arrives inbetween, then the router may miss it. Our solution is that the neighbor router temporarily caches the messages and resends it whenever the router is up. Note that after the BGP session is re-established, a neighbor needs to exchange the entire routing table which include the hash values.

Dealing with the ordering of the initial announcement. Compared with previous approaches, HC-BGP does not have a central trust entity (PKI) to answer the *who can announce what* question. Therefore, at any time, the attacker can start the initialization process independently to create the initial hash value of prefix p. Assuming an attacker announces p at time t_1, and the owner announces p at time t_2. We analyze the outcome depending on the ordering of events. If $t_1 \gg t_2$, *i.e.*, attacker announces after the owner, other ASes can reject the bogus route due to mismatch in hash values. If $t_1 \approx t_2$, the partial ordering algorithm can ensure most ASes accept the valid route. If $t_1 \ll t_2$, *i.e.*, the attacker announces p much earlier, we discuss two cases depending on the prefix type. If p is unallocated, *i.e.*, a bogon prefix, such routes can be filtered using a bogon filter list. If prefix p is allocated to the owner but is never announced, the attacker can successfully hijack it. However, since these prefixes are unused, there is no legitimate traffic destined to it, leading to minimal impact. Moreover, it is very easy for the owner to detect it since the owner will also receive the bogus route.

8 Evaluation

In this section, we first demonstrate that HC-BGP's partial ordering algorithm introduces negligible propagation delays for routing updates. We then evaluate the benefit of partial deployment.

8.1 Experimental evaluation

We extend an existing simulator [24] used to study defenses against general prefix hijacking attacks, including subprefix hijacking. The simulator takes as input the AS-level topology from the public route repository [2], containing 23,289 ASes and 55,352 edges. The topology is labeled with

the inferred AS relationship from Gao's algorithm [9]. The simulator models route propagation with the route selection guided by the routing decision process driven by relationship-based routing policies. The simulator is able to simulate different prefix hijacking scenarios and generate a set of polluted ASes and the AS-level paths to reach hijacked prefixes.

For the cryptographic one-way hash function $H[x]$, we choose to use AES block cipher in the hash construction [15], which is also used in SPV [11]. The hash function is computationally infeasible for the attacker to derive the key x nor to find any other x' such that $H[x] = H[x']$.

We use the default-free routing table from one vantage point in a Tier-1 network from RouteViews. Each prefix in the table is assigned an initial hash chain value, which is propagated to other ASes. The following simulation focuses on sub-prefix hijacking because it is most difficult to prevent and has large impact on the Internet.

8.2 Route propagation delay

The guidelines of partial ordering in §5 imposes additional propagation delay to the BGP system, which we quantify here. We first conduct the analysis using 3-day BGP data from RouteViews/RIPE. For each prefix update announced by a monitor non-Tier-1 AS, we compute the additional wait time needed for Tier-1 ASes to receive the update. We first group all updates for the same prefix across multiple vantage points using a previously established method [22] and then compute the time difference between the first update from a non-Tier-1 vantage point and the last update from any Tier-1 vantage point. The time difference conservatively estimates the additional delay due to partial ordering. Figure 8 shows that in 85% cases the additional latency is within 30 seconds. This study gives an accurate estimate of propagation delay increase using real-world data.

We also analyze the propagation delay perceived by any AS in the Internet. To study this, we further simulate the effect of imposing partial ordering using SSFNet [1] with the topology of 830 nodes provided by SSFNet. We show the average delay increase to propagate one prefix across all ASes before and after implementing our partial ordering algorithm. Figure 9 shows that 80% of the cases are within 40 seconds. It is slightly larger in SSFNet simulation than that in Figure 8 as all ASes are studied instead of just the Tier-1 ASes.

8.3 Efficiency under partial deployment

Quantifying the incremental benefit is important for understanding the adoptability of the protocol. We simulate the security guarantees under partial deployment. This benefit is two-fold. First, the participating ASes construct a barrier to stop propagating the new hash value to the attacker. Second, they detect hijacking by observing the conflicts.

First, we simulate the sub-prefix hijacking, which is much more difficult to prevent, by analyzing the effect of monitoring barrier under different partial deployment scenarios. In Figure 7, we study how the degree of pollution changes with more deployed ASes.

For each experiment, we first randomly select a pair of ASes as the attacker and victim. Then we select the ASes to deploy the scheme using the following strategies. ***1. Resilience:*** select the non-Tier-1 ASes which appear most times on the path from other ASes to any Tier-1 AS. These ASes are important for preventing polluting Tier-1 ASes. ***2. Victim-centric:*** select the ASes nearest to the victim in terms of AS path length. ***3. Random:*** select the ASes randomly.

We can see in Figure 7 that victim-centric performs best as it guarantees the route propagated to ASes while preventing the hash value from leaking to the attacker.

We also simulate the likelihood of inconsistency detection. We first randomly select 50 victims. For each victim, 50 attackers are selected randomly. The attacker can perform three types of attacks. For each attack type, we define the detection as follows.
(1) The attacker modifies p's initial value $h^n(s_p)$. An AS detects this if two routes received carry different h^n values.
(2) To launch full-prefix hijacking, the attacker intercepts p, $h^{c-1}(s_p)$ announced by the new origin, and announces it with its own origin AS. An AS detects this if the two routes for p received use the same $h^{c-1}(s_p)$, but different origin ASes.
(3) To launch the sub-prefix hijacking, the attacker replaces the legitimate sub-prefix p_1 with another sub-prefix p_2. Similarly, an AS detects it if the two routes received for two different sub-prefixes have the same hash value of the cover-prefix.

For types 1 and 2, certain ASes may not detect the inconsistency as the ASes along the propagation path may select one of them as the best route and propagate it. We focus on simulating the first two types. Type 3 can easily be detected by any ASes because p_1 and p_2 are disjoint prefixes so that both of them are always propagated.

We study the fraction of ASes capable of observing conflicts under partial deployment. In the simulation, the ASes observing the conflicts acts as a legacy AS and propagate the best route of the two. Figure 10 shows the fraction of ASes observing the inconsistencies. The upper bound 60% is caused by the set of single-homed ASes that never observe two routes to one destination. Once detected, the AS can trace back to the malicious AS hop-by-hop. The malicious AS is usually detected very quickly within few hops.

9 Performance Evaluation

In this section, we evaluate the efficiency of our protocol in terms of computation and storage resource consumptions. We compare our solution with S-BGP and its variants.

Computational complexity. As stated in §4.3, HC-BGP is computationally efficient because of two key characteristics: use of inexpensive cryptographic primitive (hash chain) and requiring cryptographic computations in uncommon cases.

To accurately assess the computation overhead of our algorithm, we profile the CPU overhead for key generation and verification separately. For objective comparison, we also implement the generation and verification process of the address attestation in S-BGP. Note that we only compare with the first phase, address attestation of S-BGP for fairness. We

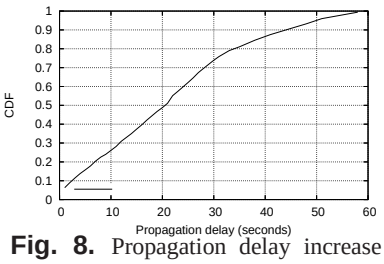

Fig. 8. Propagation delay increase from RouteViews/RIPE.

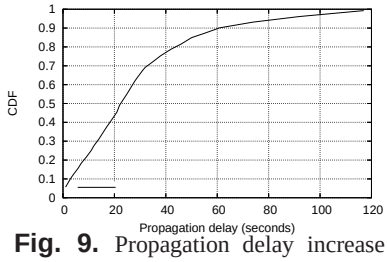

Fig. 9. Propagation delay increase (SSFNet simulation).

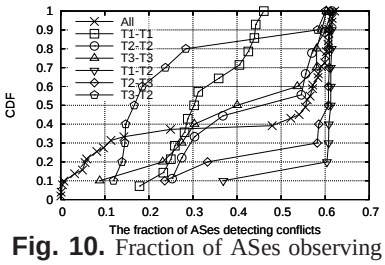

Fig. 10. Fraction of ASes observing conflicts.

first analyze the computational overhead of individual operations on these two protocols. For S-BGP, the computational overhead of the address attestation part is proportional to the number of updates. Its $UpdateNum \times Time(encrypt)$ for owner and $UpdateNum \times Time(decrypt)$ for receiver, where $UpdateNum$ is the total number of updates per day and the $Time()$ is the expensive asymmetric cryptographic primitives. In our scheme, the computational cost for the owner is $(Rate_{originChange} + Rate_{newPrefix}) \times UpdateNum \times Time(sign)$. For the receiver, the cost is $(Rate_{originChange} + Rate_{newPrefix}) \times UpdateNum \times Time(verify)$. The $Time()$ here is the complexity of more efficient one-way hash chain primitives. Even without considering the complexity difference of cryptographic techniques, HC-BGP is already $\frac{1}{(Rate_{originChange}+Rate_{newPrefix})}$ times more efficient than S-BGP. From the three-month data, we estimate the benefit is 2.6×10^4. The verification involves only one hash operation.

10 Conclusion

In this paper, we propose HC-BGP, a new architecture for securing prefix ownership. HC-BGP uses an efficient cryptographic primitive, one-way hash chain, to verify that the current message is sent from the authenticated identity associated with the previous message. We demonstrate how HC-BGP can prevent both full prefix as well as sub-prefix hijacking attacks. To further improve efficiency, HC-BGP only requires verification when the origin changes. Instead of requiring a centralized PKI, HC-BGP relies on the existing hop-by-hop trust relationship. It provides a partial ordering algorithm to prevent any malicious network from tampering the messages.

References

[1] Scalable Simulation Framework (SSF). `http://www.ssfnet.org/homePage.html`.

[2] University of Oregon Route Views Archive Project. `http://www.routeviews.org`.

[3] W. Aiello, J. Ioannidis, and P. McDaniel. Origin authentication in interdomain routing. In *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*, pages 165–178, New York, NY, USA, 2003. ACM.

[4] H. Ballani, P. Francis, and X. Zhang. A Study of Prefix Hijacking and Interception in the Internet. In *Proc. ACM SIGCOMM*, August 2007.

[5] V. J. Bono. 7007 Explanation and Apology. NANOG email on Apr 26, 1997.

[6] P. Boothe, J. Hiebert, and R. Bush. How Prevalent is Prefix Hijacking on the Internet. NANOG36 Talk, February 2006.

[7] K. Butler, P. McDaniel, and W. Aiello. Optimizing BGP security by exploiting path stability. In *Proc. CCS*, New York, NY, USA, 2006. ACM.

[8] H. Chan, D. Dash, A. Perrig, and H. Zhang. Modeling Adoptability of Secure BGP Protocol. In *Proc. ACM SIGCOMM*, 2006.

[9] L. Gao. On Inferring Autonomous System Relationships in the Internet. In *Proc. IEEE Global Internet Symposium*, 2000.

[10] X. Hu and Z. M. Mao. Accurate Real-time Identification of IP Prefix Hijacking. In *Proc. IEEE Security and Privacy*, 2007.

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

[12] M. Lad, D. Massey, D. Pei, Y. Wu, B. Zhang, and L. Zhang. PHAS: A Prefix Hijack Alert System. In *Proc. USENIX Security Symposium*, 2006.

[13] M. Lad, R. Oliveira, B. Zhang, and L. Zhang. Understanding resiliency of internet topology against prefix hijack attacks. In *Proc. DSN*, 2007.

[14] L. Lamport. Password authentication with insecure communication. *Commun. ACM*, 24(11):770–772, 1981.

[15] S. Matyas, C. Meyer, and J. Oseas. Generating Strong One-Way Functions with Cryptographic Algorithm, 1985.

[16] NANOG. YouTube IP Hijacking, Febrary 2008.

[17] J. Ng. Extensions to BGP to Support Secure Origin BGP (soBGP). IETF Draft: draft-ng-sobgp-bgp-extensions-01.txt, November 2002.

[18] J. Qiu and L. Gao. Hi-BGP: A Lightweight Hijack-proof Interdomain Routing Protocol. Technical report, University of Massachusetts Amherst, 2006.

[19] Stephen Kent and Charles Lynn and Karen Seo. Secure Border Gateway Protocol (Secure-BGP). *IEEE J. Selected Areas in Communications*, 2000.

[20] L. Subramanian, V. Roth, I. Stoica, S. Shenker, and R. H. Katz. Listen and Whisper: Security Mechanisms for BGP. In *Symposium on NSDI*, 2004.

[21] D. Wendlandt, I. Avramopoulos, D. Andersen, and J. Rexford. Don't Secure Routing Protocols, Secure Data Delivery. In *Proc. ACM Workshop on Hot Topics in Networks (HotNets)*, 2006.

[22] J. Wu, Z. M. Mao, J. Rexford, and J. Wang. Finding a needle in a haystack: Pinpointing significant BGP routing changes in an IP network. In *Proc. NSDI*, 2005.

[23] H. Yin, B. Sheng, H. Wang, and J. Pan. Securing BGP through keychain-based signatures. In *Proceedings of the 15th IWQoS'07*, June 2007.

[24] Z. Zhang, Y. Zhang, Y. C. Hu, and Z. M. Mao. Practical Defenses Against BGP Prefix Hijacking . In *Proc. ACM CoNEXT*, 2007.

[25] X. Zhao, D. Pei, L. Wang, D. Massey, A. Mankin, S. F. Wu, and L. Zhang. An analysis of BGP multiple origin AS (MOAS) conflicts. In *Proc. IMW*, pages 31–35, New York, NY, USA, 2001. ACM.