Symmetric Key Approaches to Securing BGP – A Little Bit Trust Is Enough

Bezawada Bruhadeshwar[1], Sandeep S. Kulkarni[2], and Alex X. Liu[2]

[1] Center for Security, Theory and Algorithmic Research, International Institute of Information Technology
Gachibowli, Hyderabad 500032, India
bezawada@iiit.ac.in

[2] Department of Computer Science and Engineering
Michigan State University, East Lansing, MI 48824, U.S.A.
{sandeep,alexliu}@cse.msu.edu

Abstract. The Border Gateway Protocol (BGP) is the de facto inter-domain routing protocol that connects autonomous systems (ASes). Despite its importance for the Internet infrastructure, BGP is vulnerable to a variety of attacks due to lack of security mechanisms in place. Many BGP security mechanisms have been proposed, however, none of them has been deployed because of either high cost or high complexity. The right trade-off between efficiency and security has been ever challenging.

In this paper, we attempt to trade-off between efficiency and security by giving a little dose of trust to BGP routers. We present a new flexible threat model that assumes for any path of length h, at least one BGP router is trustworthy, where h is a parameter that can be tuned according to security requirements. Based on this threat model, we present two new symmetric key approaches to securing BGP: the centralized key distribution approach and the distributed key distribution approach. Comparing our approaches to the previous SBGP scheme, our centralized approach has a 98% improvement in signature verification. Our distributed approach has equivalent signature generation cost as in SBGP and an improvement of 98% in signature verification. Comparing our approaches to the previous SPV scheme, our centralized approach has a 42% improvement in signature generation and a 96% improvement in signature verification. Our distributed approach has a 90% improvement on signature generation cost and a 95% improvement in signature verification cost. By combining our approaches with previous public key approaches, it is possible to simultaneously provide an increased level of security and reduced computation cost.

1 Introduction

The Internet consists of independently administered networks, which are called autonomous systems (ASes). The Border Gateway Protocol (BGP) is the de facto inter-domain routing protocol that connects ASes together [1]. BGP provides two essential services: mapping IP prefixes onto the ASes that own them and

S. Jajodia, and J. Lopez (Eds.): ESORICS 2008, LNCS 5283, pp. 82–96, 2008.

the construction of source specific paths to each reachable prefix. Every BGP router announces the IP prefixes that its AS owns in an update message and sends the message to its neighboring BGP routers. Received update messages are recursively concatenated with an additional AS number and propagated from AS to AS forming a routing path, which will be used to forward traffic. When a BGP router receives multiple paths for the same prefix, the router chooses the best path based on multiple criteria such as path length, routing policies, etc. For simplicity, in this paper, we use the three terms "AS", "BGP router", and "router" interchangeably when there is no confusion.

The BGP update messages are undoubtedly important as they enable ASes to construct a consistent view of the network topology. Invalid update messages may result in incorrect routing tables, which could lead to three types of potentially disastrous consequences. First, incorrect BGP routing tables may make a range of IP addresses unreachable, which constitutes a deny-of-service attack. Second, incorrect BGP routing tables may make some packets to travel through a malicious BGP router, which may launch man-in-the-middle attacks by eavesdropping, tampering, inserting, or dropping messages. Third, incorrect BGP routing tables may make some packets travel more hops than necessary to reach their destination, which degrades the Internet routing performance. However, due to the lack of security mechanisms in the current BGP protocol, attackers may spoof or tamper BGP messages. Thus, it is critical for a recipient AS to validate the authenticity (*i.e.*, to detect spoofing) and integrity (*i.e.*, to detect message tampering) of update messages before making routing decisions. For simplicity, in the rest of the paper, we use "BGP updates" to mean "BGP update messages". We refer to the process of validating the authenticity and integrity of BGP update messages as "BGP path validation"

Many solutions have been proposed previously for securing BGP (*e.g.*, [2, 3, 4, 5, 6, 7, 8, 9]). However, none of them have been adopted so far due to either high cost (such as S-BGP that extensively uses public key cryptography operations [4]) or high complexity (such as SPV that requires complex state maintenance and fairly large computational resources for BGP routers where such resources are of critical value [9]). In examining previous solutions, we observe that for any given advertised path, these solutions require all nodes on that path to validate the prefix of the path up to that node. This constitutes the root cause of the inefficiency and complexity of previous solutions. Actually, this may be unwarranted for every path advertisement because BGP routers are expected to be more trustworthy than end hosts. BGP routers are typically owned by large Internet service providers (ISPs) that have little incentive in intentional falsification of route advertisements. Although compromising a BGP router may be possible, compromising multiple BGP routers on the same path at the same time by the same attacker is unlikely. In [10], based on BGP data from 40 global ASes, Butler *et al.*observed that on average 67-98% percent of paths were stable over the period of one year and over 99% paths were stable over a period of one month. Thus, if we use a trust building approach along a particular path, then, even a few trusted nodes can eliminate the effect of the malicious routers as we know that the update message would be processed by trusted routers at some point. However, implicitly trusting all routers is also unreasonable since some routers could be compromised.

The above observations suggest a threat model where the number of malicious routers on any given path is limited. In particular, we consider the model where for any path of length h, at least one BGP router is trustworthy. For ease of presentation, consider the case where we have a trustworthy BGP router X. Then, before advertising any path that includes X, X would have checked the authenticity and integrity of the path up to X. If X is trustworthy, then each subsequent node on the path, which receives the path advertisement containing X, does not need to validate the path before X, instead, it only needs to validate the path from X up to itself. In other words, instead of including a verification from each node on the advertised path, it would suffice if only signatures from X onward are used. Of course, this new scheme must accommodate the fact that we do not know which routers are the actual trustworthy ones.

So far, we have discussed the idea of reducing the cost of validating BGP paths by reducing the number of validation operations needed for each path. Another cost in BGP security is the cost of verification itself. Existing approaches in [4,6,5] use digital signatures constructed using public key cryptography. However, such digital signatures are expensive to generate and verify, which consequently degrade the performance of BGP routers [11]. The performance of BGP routers is a critical concern as the volume of traffic passing through BGP routers is very high [12]. Due to this fact, a BGP router needs to process update messages in the shortest time possible to avoid route disruptions and dropping/delaying of packets. Hence, there is a need for lightweight symmetric key based mechanisms that retain the benefits of digital signatures, authentication, integrity, and non-repudiation, while maintaining good performance of BGP routers.

Based on our above two ideas, one reducing the number of path validation costs by adding a little bit of trust into our threat model and one reducing the cost of each path validation by using efficient symmetric key management schemes, we propose two new symmetric key approaches to securing BGP. The first approach is a family of centralized key management protocols for securing BGP, which require a trusted server for distributing keys. Each BGP router only needs to maintain $O(\log^2 N)$ keys where N is the total number of BGP routers. The verification cost is even lower, $O(\log N)$ or less. The second approach is a family of distributed key management protocols for securing BGP, which does not require a trusted server for distributing keys. Distributed key management protocols are initiated by individual senders who intend to provide authentication for their messages. The signature and verification cost in these protocols are also $O(\log N)$. The small signature and verification cost makes these distributed key management protocols attractive if performance is the primary concern for BGP routers. In this paper, we evaluate the performance of our protocols against standard public key cryptographic solutions as well as symmetric key solutions that have been proposed for securing BGP. We show that our solutions perform orders of magnitude better than existing public key and symmetric cryptography based solutions.

The rest of this paper proceeds as follows. In Section 2, we give an overview of the BGP protocol, discuss BGP security issues, discuss previous solutions, and present our threat model. In Section 3, we present the technical details of our centralized key management protocols and distributed key management protocols. Our experimental results are shown in Section 4. We analyze the security of our proposed approaches in Section 5. We conclude in Section 6.

2 BGP Overview, Security Issues and Past Solutions

In this section, we give a brief overview of the BGP protocol [1], outline the security issues in current BGP implementations, discuss previous work, and describe our threat model and assumptions.

2.1 BGP Overview

A main objective of BGP is to advertise the routing path information for IP prefixes. Towards this, BGP routers initiate TCP connections with other BGP peers and exchange the path information in the form of BGP update messages. For this discussion, we represent an update message as a tuple: (*prefix*, *as_path*), where the *prefix* denotes what the message needs to advertise or withdraw, and the *as_path* denotes the sequence of ASes through which this update message has traversed. When a BGP router receives an update message, it will concatenate the *as_path* field of the message with its AS number and propagate the message to other neighboring ASes. When a BGP router receives multiple paths for the same prefix, the router chooses the best path based on its own criteria. Although BGP update messages can be used to advertise as well as withdraw IP prefixes, without loss of generality, we assume that update messages contain prefix advertisement. All our discussion applies to withdraw messages as well.

Figure 1 shows the traversal of the BGP update message originated from AS A, who owns the IP prefix 24.0.0.0/8, denoted as P. To advertise that A owns

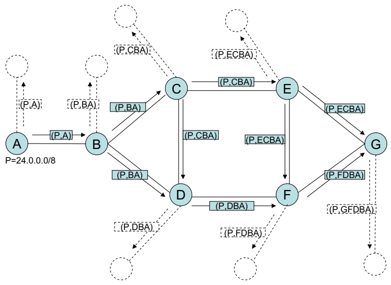


Fig. 1. Traversal of BGP update messages

prefix P, A sends (*P, A*) to its neighboring ASes, including B. When B receives this message, it first updates its routing tables appropriately, then prepends its AS number to the *as_path* field of the tuple and sends the new update message (*P, BA*) to its neighboring ASes including C and D. Proceeding in this manner, the update message is propagated until all the ASes are aware of the path to reach the prefix advertised by *A*. Note that when D receives two update messages (*P, BA*) (*P, CBA*) for the same prefix P, in this example D chooses the shorter path to P, which is (*P, BA*).

There are four major types of attacks on BGP control messages: *deletion, replay, modification, and insertion.* The first two types of attacks are out of the scope of this paper. Deleting BGP control messages seems indistinguishable from legitimate route filtering [6]. Replay can be handled by setting expiration time for BGP messages [6]. Hence, this paper concerns the latter two types of attacks. BGP path insertion attacks are also called path forgery attacks, in which an adversary forges a path. We refer both BGP path modification and forgery attacks as BGP path falsification attacks. There are four types of BGP control messages: *open, keepalive, notification*, and *update*. The first three are used by BGP to establish and maintain BGP sessions with their peers. As stated by Hu *et al.*, these three first types of messages can be protected by a point-to-point authentication protocol such as IPSec [13]. This paper concerns protecting the fourth type of message, update messages. In BGP path modification attacks, an adversary may add, remove, or alter AS numbers from the *as_path* field of BGP update messages.

2.2 Past Solutions for BGP Security

In [4], Kent *et al.*present S-BGP, a comprehensive framework for achieving security in BGP using two Public-key Infrastructures (PKIs). One PKI is for issuing public-key certificates to the organizations to bind addresses to organizations and the second PKI is for issuing public-key certificates to each BGP router to bind AS and router associations. To validate an update, the originator of the update message signs the IP prefixes using its private-key and sends the update to its neighboring routers. Each neighboring router validates the update message using the several certificates produced by the originating BGP router. Upon validating the message through signature verification, the neighboring router creates a *route attestation* i.e., it updates the *as_path* field, signs it with its private key and appends it to the original message to create the new update. Every transit router verifies all the attached signatures and adds its own route attestation to the update message. S-BGP places significant computation overhead on the BGP routers since digital signature creation and verification are costly as studied in [11] and degrades performance of the BGP routers.

In secure origin BGP(soBGP) [5], White describes a PKI based solution that requires three public-key certificates, two of which are similar to those used in S-BGP. In soBGP, the security related information is conveyed by a new message type called SECURITY message. The soBGP scheme reduces the cost of signature verification by verifying the long standing information such as address ownership, organizational relationships and topology, before the BGP sessions start, and storing this information at the routers. Only the variable information like *as_path* are validated at run-time. However, as in S-BGP, soBGP also incurs significant signature verification cost and an additional cost of storing the topology information.

In [6], Van Oorschot *et al.*describe the *Pretty Secure BGP* (psBGP) which combines S-BGP and soBGP. This solution improves upon S-BGP by using only one PKI and also, describes additional improvements. The semantics of psBGP are still based on PKI and hence, are computationally expensive.

In [7], the authors describe a solution in which the BGP data exchanged by two BGP peers is encrypted by a session key. However, this scheme still involves expensive digital signature verification by each intermediate router. In [8, 9], Hu *et al.*describe schemes based on Merkle-hash trees [14, 15] and one-way hash chains, to preserve path vector integrity for routing protocols. Although SPV is efficient compared to public-key based solutions, it involves significant precomputation and state overhead.

Note that the scope of this paper is on BGP control plane security, not BGP data plane security [16]. Recently, Hu and Mao have methods to use data plane information to validate occurrences of IP hijacking in real time [17].

2.3 Threat Model and Assumptions

Our threat model is based on the various falsification attacks [18] on the BGP protocol, some of which have been detailed in Section 2.1. From these attacks, the types of falsification attacks that we address in this work are: generation of false update messages by spoofing source IP address, insertion or deletion of AS numbers from the *as_path* field, and changing the order of AS numbers in the *as_path* field. Note that, a combination of these attacks is also possible. We address such combined attacks as well. We assume that one or more BGP routers could be malicious. However, we assume that there is at least one non-malicious node along a given path of length h. We treat any misconfiguration of BGP routers as malicious and accordingly address this from the point of view of falsification.

3 Symmetric Key Management for Securing BGP

We examine two types of symmetric key distribution approaches for securing BGP messages. In the first approach, a centralized controller establishes the necessary keys among the BGP routers and hence, we call protocols using this approach as *centralized key distribution* protocols. In the second approach, we assume that a centralized controller does not exist and each AS distributes the necessary keys to the BGP routers of other ASes. We call key distribution protocols using this approach as *distributed key distribution protocols*. We show the use of both these approaches to achieve authentication in the BGP.

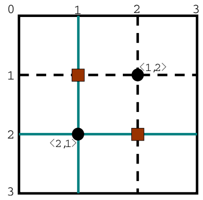


Fig. 2. Example square grid

3.1 Centralized Key Distribution Protocols

In this section, first, we describe the square grid key distribution protocol [19]. Then, we describe how the square grid protocol can be used to achieve security of the BGP update messages. Finally, we describe extensions of grid protocol that provide similar properties while reducing the number of keys compared to the protocol in [20].

The Square Grid Protocol. In the square grid protocol [19], n users are arranged in a *logical* square grid of size $\sqrt{n}$ x $\sqrt{n}$. Each location, $\langle i, j\rangle$, $0 \leq i, j < \sqrt{n}$, in the grid is associated with a *user* $u_{\langle i,j\rangle}$ and a *grid key* $k_{\langle i,j\rangle}$. Each user knows all the grid keys that are along its row and column. Additionally, each user maintains a direct key with users in its row and column. This direct key is not known to any other user.

Now, consider the case where user A wants to set up a session key with user B. Let the locations of A and B be $\langle j_1, k_1\rangle$ and $\langle j_2, k_2\rangle$ respectively. In this case, A selects the session key and encrypts it as follows. If A and B are in same row or column, A uses the unique key between A and B for session encryption. Otherwise, A uses an XOR of grid keys at locations $\langle j_1, k_2\rangle$ and $\langle j_2, k_1\rangle$. For example, in Figure 2, the users marked at location $\langle 1, 2\rangle$ and $\langle 2, 1\rangle$ use the keys marked with ■ . Along with the encrypted session key, A also sends its own grid location (in plain text) to B. The above key selection protocol ensures that, in the absence of collusion, the key used by A cannot be derived by any other user other than A and B. (cf. [19] for proof.)

Square Grid for Authentication in BGP. First, we focus on the problem of authenticating the source of the advertisement. Note that the square grid protocol is designed for secure and authenticated point-to-point communication between two nodes. However, in BGP, the same path advertisement may be sent to several neighbors and forwarded subsequently. Due to these reasons, for this discussion, we assume that a given path advertisement may be possibly verified by any AS in the network.

To use the grid protocol, each AS is assigned a logical identifier in the grid and the corresponding keys as specified by the grid protocol. We assume that a centralized controller has assigned the logical identifiers and the corresponding keys to the BGP routers. Note that, the process of key establishment can be achieved by the use of public-keys or other similar key agreement protocols.

Now, consider the case where an AS with logical identifier $\langle j_1, k_1\rangle$ needs to advertise a route $\langle A_1 A_2 \ldots A_n\rangle$ for prefix 24.12.0.0/8. To advertise this route, $\langle j_1, k_1\rangle$ signs a message consisting of $\langle A_1 A_2 \ldots A_n\rangle$ and 24.12.0.0/8. Towards this end, $\langle j_1, k_1\rangle$ encrypts the message $\langle 24.12.0.0/8, \langle A_1 A_2 \ldots A_n\rangle\rangle$ using each of the keys it has (both direct and grid keys) separately. Subsequently, to advertise the route, it sends a packet consisting of the following information (1) its ID, namely $\langle j_1, k_1\rangle$, in plain text, (2) the route $\langle A_1 A_2 \ldots A_n\rangle$ and prefix 24.12.0.0/8 in plain text, and (3) a (hash value of) the message obtained by encrypting $\langle 24.12.0.0/8, \langle A_1 A_2 \ldots A_n\rangle\rangle$ with each of the keys it has. This part is denoted as the *signature block* of the message.

Whenever an AS receives this message, it uses the ID, say $\langle j_1, k_1\rangle$, associated with the message to determine which keys should be used for authentication. In particular, as specified in Section 3.1, it identifies a collection of grid keys or a direct key that it would use if it were to communicate with an AS with logical identifier $\langle j_1, k_1\rangle$. Then, using those keys separately, it encrypts $\langle 24.12.0.0/8, \langle A_1 A_2 \ldots A_n\rangle\rangle$ (received in plain text), and hashes the encrypted value. It determines if all the hash values it computed are present in the signature block. If so, it accepts the message.

Theorem 1. The above approach ensures that when an AS accepts a message that contains ID $\langle j_1, k_1\rangle$, the corresponding message is indeed sent by node $\langle j_1, k_1\rangle$. □

Now, consider the network shown in Figure 1. In this figure, node B is advertising a route BA for prefix 24.12.0.0/8. To advertise this message, as described above, it generates the signature block and sends it to node C. Subsequently, node C advertises the route CBA. When node E receives this message, it needs to ensure that BA was sent by B and CBA was sent by C. This can be achieved by having node C concatenate the signature block of B and its own signature block for route CBA and send it to node E. Upon receiving this message, node E can verify the route advertised by B and C.

Reducing Signature Block Size Based on Trust Between ASes. One potential concern with the above approach is that as the length of the path increases, the number of signature blocks also increase. In this context, we note that while perfect authentication is desirable for BGP routing messages, the ASes differ from individual users on the Internet. In particular, while an individual AS may be compromised, we do not anticipate a significant misrepresentation to be done by ASes. Hence, as discussed in Section 2.3, we consider the threat model where at least one AS on any path of length h is trustworthy (although the exact trustworthy AS is unknown). For sake of presentation, next, we consider Figure 1 and let $h = 2$. Consider the case where node E advertises the route $ECBA$ and this route is received by G. Based on our assumption, either AS C or E (or both) is trusted. Now, we show that in this case, node G does not need to receive the signature block of B. To see this, we consider two cases:

- E is trustworthy: In this case, AS E has verified the validity of route BA and CBA. AS G can verify that the route $ECBA$ is advertised by E. Since BA is a prefix of route $ECBA$, node G does not need to receive the signature block of B.
- C is trustworthy: In this case, AS C has verified the validity of route BA. AS G can verify that CBA is indeed advertised by C using the signature block of C. Hence, it does not need the signature block of B.

We can generalize the above scenario for arbitrary paths and arbitrary values of h. In particular, if at least one of h consecutive ASes is trusted then the signature blocks of the last h ASes need to be attached with the route. Moreover, if an AS desires an additional level of security then it can request additional signature blocks as needed from nodes in the *as_path* (cf. [21] for details).

Storage-efficient Key Distribution Protocols. Recently, in [22, 23, 24], the authors describe a storage efficient key distribution protocols for achieving confidentiality and authentication in completely connected communication networks. Essentially, these protocols maintain a higher dimension grid to reduce the number of keys used in them. Of these the protocol in [22] uses $4 \log^2 N$ keys, the protocol in [23] improves it to $\log^2 N)$ and the protocol in [24] improves it to $\frac{1}{2} \log^2 N + O(\log N + \log \log N)$. All these protocols provide a property similar to that of the grid protocol. In particular, if a node sends a message that includes a signature from each of the keys it has and the receiver verifies the signatures based on the common keys then it can conclude that the message is authentic. Because these protocols use grids with dimension of $\log N$, whenever the node receives a message, it needs to verify two signatures from each grid. The most storage efficient protocol from these protocols is from [24]. One can choose the appropriate protocol depending on the type of storage and verification requirements.

3.2 Distributed Key Distribution Protocols

In distributed key distribution protocols, each node is responsible for locally generating and distributing the keys to the other users in the network. This approach is especially useful when establishing a centralized key distribution infrastructure is difficult. In [20, 24], the authors describe key distribution protocols for a star communication network. In star communication networks, a center node communicates with several satellite nodes and vice-versa. The satellite nodes do not communicate with each other. Next, we describe the key distribution protocol from [24] and show that this protocol provides message authentication.

Optimal Key Distribution for Star Networks. In the key distribution protocols described in [24], the center node maintains a set of k keys. Each satellite node receives a unique subset of size l from this set. Note that, by construction, no two satellite nodes have identical subsets of keys. We term this protocol instance as $p(k, l)$. Using $p(k, l)$, authentication of messages can be achieved in the communication as follows.

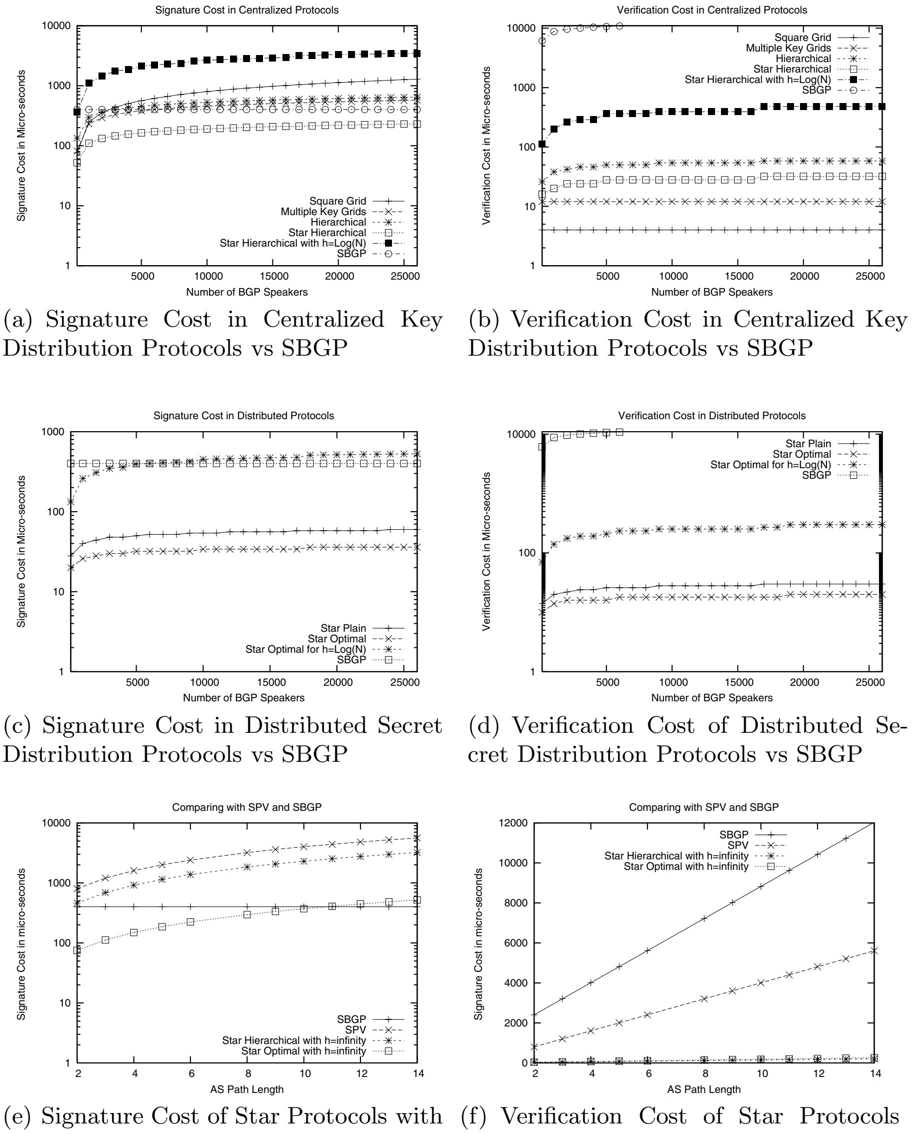
To authenticate a message m broadcasted by the center node to the satellite node, the center node generates authentication codes with each of the k keys. In this context, an authentication code is a secure hash/encryption computed using a shared symmetric key. Hence, each authentication code consists of the message digest md of the message computed using a key held by the center. The center appends the k authentication codes thus generated to the message and broadcasts the resulting message. Now, when a satellite node receives this message, it uses its subset of l keys to compute l authentication codes. The satellite node then verifies these authentication codes with the corresponding authentication codes sent by the center node. Note that, each satellite node can verify only those authentication codes for which it has the corresponding generating key.

In [24], authors have shown that given a set of n satellite nodes, maintaining $k = \log n + 1/2 \log \log n + 1$ secrets at the center node is sufficient if each node receives $k/2$ keys. If each node receives $k/2$ keys then there exists a set of two nodes whose collusion can reveal all the keys. Hence, to deal with this case, we can assign each node only k/m keys where m is the level of desired collusion resistance. For example, if we choose $m = 10$ then maintaining 40 keys and letting each node receive 4 keys would allow $C(40, 4) = 91390$ satellite nodes. And, this would be sufficient for BGP which currently has approximately 26000 ASes [25].

Using $p(k, l)$ for Authentication in BGP. A BGP network with N routers can be viewed as a collection of N star networks where each BGP router is the center node for one of these networks. Now, each BGP router runs an instance of $p(k, l)$ as described above. To authenticate an update message, the originating BGP router generates the necessary authentication codes from its keys. The fields of the update that are authenticated are: AS number of origin, *as_path* field and IP prefixes being advertised. Each intermediate BGP router verifies these signatures and updates the *as_path* field. The intermediate BGP router adds an additional signature block on the *as_path* field using its keys from the $p(k, l)$ that is instantiated at this router. Each receiving node can verify all the signatures to ensure authentication of the source, the validity of the *as_path* and the integrity of the IP prefixes. Thus, in the worst case scenario, the signature block can be as large as $O(L. \log N)$ where L is length of the path that the update might traverse. Furthermore, based on the trust model identified in Section 2.3, if at most one router from a given path of length h is trusted then the number of signatures that may be added to a given message is at most $O(h \log N)$.

4 Evaluation

In this section, we evaluate the performance of our approach and compare it with S-BGP [4] and SPV [9]. Towards this, we evaluate the protocols on, the signature cost and verification cost In the centralized key distribution approach, we compare the protocols from [19, 22, 23, 24] with SBGP. Similarly, in the distributed key distribution approach, we compare the protocols from [20, 24] with



(a) Signature Cost in Centralized Key Distribution Protocols vs SBGP

(b) Verification Cost in Centralized Key Distribution Protocols vs SBGP

(c) Signature Cost in Distributed Secret Distribution Protocols vs SBGP

(d) Verification Cost of Distributed Secret Distribution Protocols vs SBGP

(e) Signature Cost of Star Protocols with SBGP and SPV

(f) Verification Cost of Star Protocols with SBGP and SPV

Fig. 3. Experimental Results

SBGP. Next, we apply our trust model to centralized and distributed protocols and compare the results with SPV [9] and SBGP [4].

We use the following notation to refer to the various key distribution protocols. We refer to the centralized key distribution protocols as follows: [19] as *Square*

Grid, [22] as *Multiple Key Grids*, [23] as *Hierarchical*, [24] as *Star Hierarchical* We refer to the distributed key distribution protocols as follows: [20] as *Star Plain* and, [24] as *Star Optimal*.

In our approach, for the various symmetric key distribution protocols, we assume that a BGP router uses the HMAC (Hashed Message Authentication Code) algorithm with 512-bit keys to generate the corresponding 160-bit message authentication codes. For SBGP, we assume that BGP routers use the RSA algorithm for signature generation and verification. We assume that the cost of signature verification using RSA is around 401 micro-seconds and that of a message authentication code is around 2 micro-seconds [9], in all our analysis.

Comparison of Centralized Key Distribution Protocols with SBGP In Figure 3(a), we compare the signature and verification cost of the different centralized key distribution protocols with SBGP. In SBGP, the cost of signature generation for a BGP speaker is only one signature i.e., the route attestation that is added by this speaker. From, Figure 3(a), we can observe that on an average, the cost of signature generation is lower for the centralized protocols. The *Star Hierarchical* protocol has the lowest signature cost amongst all the protocols. We also show the signature cost incurred by using our trust model with the value of $h = \log N$, where N is the total number of BGP speakers in the system. We chose this value specifically, since most practical networks have logarithmic diameter and this value represents the maximum distance that an update can traverse.

In SBGP, the cost of signature generation is low as each BGP speaker only needs to add its own signature to the update. However, the cost of verification is high in SBGP, since each BGP speaker will have to verify the route attestations of all the ASes that are part of the update message. For purposes of calculation of verification cost for SBGP, we assumed that each update traverses at the most $\log N$ ASes and computed the cost incurred by the last BGP router in the path.

In Figure 3(b), we show the verification cost of these protocols and that of SBGP. We note that, in terms of verification, the distributed protocols are orders of magnitude better than SBGP. Even for the case when $h = \log N$, in *Star Hierarchical* protocol, the cost of verification is lower than SBGP. In practice, for BGP routers, a smaller cost of verification is desirable. This is because the BGP router needs to make a decision whether or not to accept the update message and this can only be done after the verification process.

Comparison of Distributed Key Distribution Protocols with SBGP Similarly, in Figures 3(c)-3(d), we compare the signature and verification cost of the distributed key distribution protocols with SBGP. From Figure 3(c), we observe that the signature cost of distributed protocols is much lower than that of SBGP. Furthermore, even when $h = \log N$, the signature cost is comparable with that of SBGP. As in the centralized protocols, the cost of verification in distributed protocols is orders of magnitude better than SBGP. We note that, the size of signature block in the distributed protocols is much smaller than in centralized protocols.

Comparison of Signature and Verification Cost in Our Approach with SPV and SBGP. In SPV [9], each originating node needs to generate a one-time signature that will be verified by its downstream routers. The one-time signature is the root of a merkle hash-tree [14]- [15] which is generated by using the *as_path* field and a secret key held by the sender. For comparing SPV, we chose the number of leaves of a single one-time signature to be 80 and the total ASes in the path to be 15 (cf. [9]). For clarity of presentation compare SPV and SBGP with the *Star Hierarchical*, which is a centralized protocol, and the *Star Optimal*, which is a distributed protocol, for $h = infinity$. This implies that no node is trusted on the path and all nodes include their signatures for authenticating the update messages. In Figure 3(e), we note that the cost of signatures using the *Star optimal* protocol is lower than SBGP for small path lengths. Also, both *Star Optimal* and *Star Hierarchical* are better than SPV for signature generation. This is due to high initialization cost required in SPV. In Figure 3(f), we note that cost of verification in both *Star Hierarchical* and *Star Optimal* is orders of magnitude lower than SPV and SBGP. These results show that using our approach it is possible to reduce the verification cost of updates even when no BGP router is trusted.

5 Security Analysis

In this section, we analyze the security of both the centralized and the distributed approaches proposed in this paper against the creation of invalid routes, which is the main security goal that we try to address. Our analysis focuses on the cost of adopting our protocols and the cost of breaking our protocols.

Using the centralized approach with Mittal's key protocol [23], the number of keys an AS needs to maintain is $(\log N)^2$, where N is the total numbers of ASes. Given that there are about 26000 ASes on the Internet [25], deploying the centralized approach on the Internet requires each AS to maintain approximately 225 secret keys, which can be easily stored in memory. The verification cost for each BGP update message is $O(\log N)$. According to the centralized approach, each update message needs to contain $225 * h$ signature blocks. Considering that the current BGP message size has a limit of 4Kilobytes, we can choose h to be 3 and the size of each signature block to be 6 bytes. A signature block of 6 bytes can be obtained by choosing the first 6 bytes of an MD5/SHA-1/HMAC hash. Although a 6-byte signature is not as secure as a normal MD5 hash (16 bytes) or SHA hash (20 bytes), to make an AS accept a forged signature requires forging of at least 15 $(\log N)$ such signatures, which is computationally infeasible.

In the distributed approach, we treat each AS as a center node, and all other ASes as satellite nodes for that particular AS. For $p(k, l)$, if we choose k to be 40, and l to be 4, the total number of satellite nodes that an AS can have is 91390, which is enough as the number is ASes on the Internet is about 26000. Each AS needs to maintain two sets of secret keys, one set consists of 40 secret keys that the AS needs to distribute to its satellite nodes, the other set consists of secret keys that other nodes distribute to this AS node, which is around $26000 * 4$ keys.

Thus, the total number of keys an AS needs to maintain is only 104040. Using the distributed approach, each update message needs to contain $40 * h$ signature blocks. If we choose h to be 5, the 4Kilobyte BGP message size allows each signature block to be 20 bytes, which is the output of an SHA-1 hash. Moreover, as discussed in previous paragraph, only first few bytes of hash could be used thereby increasing the value of h. Breaking the distributed approach requires the collusion of 5 adjacent ASes or 10 ASes, which we consider practically unlikely.

6 Conclusion

We make three key contributions in this paper. First, we show that the right trade-off between efficiency and security for BGP could be achieved by adding the little bit of trust on BGP routers. We present a new flexible threat model where for any path of length h, at least one BGP router is trustworthy. Second, we present two new symmetric key approaches to securing BGP: the centralized key distribution approach and the distributed key distribution approach. Third, we evaluated the efficiency of the two approaches with previous approaches to securing BGP. The evaluation results show that our approaches are significantly more efficient than previous approaches. Our schemes are flexible and scalable which makes their deployment feasible.

References

[1] Rekhter, Y., Li, T.: A border gateway protocol 4 (bgp 4). IETF RFC 1771 (1995)

[2] Subramanian, L., Roth, V., Stoica, I., Shenker, S., Katz, R.: Listen and whisper: Security mechanisms for bgp. In: First Symposium on Networked Systems Design and Implementation (NSDI 2004), San Francisco, CA, USA (2004)

[3] Goodell, G., Aiello, W., Griffin, T., Ioannidis, J., McDaniel, P., Rubin, A.: Working around bgp: An incremental approach to improving security and accuracy of inter-domain routing. In: Network and Distributed Systems Security (NDSS), San Diego, CA, USA, pp. 75–85. Internet Society (2003)

[4] Kent, S., Lynn, C., Seo, K.: Secure border gateway protocol. IEEE Journal on Selected Areas in Communication 18(4), 582–592 (2000)

[5] White, R.: Securing bgp through secure origin bgp. The Internet Protocol Journal 6(3), 15–22 (2004)

[6] Van Oorschot, P.C., Wan, T., Kranakis, E.: On interdomain routing security and pretty secure bgp (psbgp). ACM Transactions on Information and System Security 10(3) (July 2007)

[7] Smith, B.R., Garcia-Luna-Aceves, J.J.: Securing the border gateway routing protocol. Computer Communications 21(3), 203–210 (1998)

[8] Hu, Y., Perrig, A., Johnson, D.: Efficient security mechanisms for routing protocols. In: Network and Distributed Systems Security (NDSS), San Diego, CA, USA, Internet Society (2003)

[9] Hu, Y., Perrig, A., Sirbu, M.: Spv: Secure path vector routing for securing bgp. In: ACM 2004 SIGCOMM, Portland, OR (2004)

[10] Butler, K., McDaniel, P., Aiello, W.: Optimizing bgp security by exploting path stability. In: CCS, October-November 2006. ACM, New York (2006)

[11] Zhao, M., Smith, S., Nicol, D.: Evaluating the performance impact of pki on bgp security. In: 4th Annual PKI Research Workshop (PKI 2005), Gaithersburg, MD. ACM, New York (2005)

[12] http://bgpupdates.potaroo.net/instability/bgpupd.html

[13] Kent, S., Atkinson, R.: Security architecture for the internet protocol. IETF RFC 2401 (1998)

[14] Merkle, R.C.: Protocols for public key cryptosystems. In: IEEE Symposium on Security and Privacy (1980)

[15] Merkle, R.: A digital signature based on a conventional encryption function. In: Pomerance, C. (ed.) CRYPTO 1987. LNCS, vol. 293, pp. 369–378. Springer, Heidelberg (1988)

[16] Wong, E., Balasubramanian, P., Alvisi, L., Shmatikov, V.: Truth in advertising: Lightweight verification of route integrity. In: Proceedings of the 26th ACM Annual Symposium on the Principles of Distributed Computing (PODC 2007), pp. 147–156 (2007)

[17] Hu, X., Mao, Z.M.: Accurate real-time identification of ip prefix hijacking. In: Proceedings of IEEE Security and Privacy, May 2007, pp. 3–17 (2007)

[18] Butler, K., Farley, T., McDaniel, P.: A survey of bgp security. Technical Report TD-5UGJ33, AT&T Labs- Research, Florham Park, NJ (February 2004)

[19] Kulkarni, S.S., Gouda, M.G., Arora, A.: Secret instantiation in ad-hoc networks. Computer Comunications (29), 200–215 (2006)

[20] Gouda, M.G., Kulkarni, S.S., Elmallah, E.S.: Logarithmic keying of communication networks. In: 8th International Symposium on Stabilization, Safety, and Security of Distributed Systems, SSS 2006 (2006)

[21] Bruhadeshwar, B., Kulkarni, S.S., Liu, A.X.: Symmetric key approaches to securing bgp -a little bit trust is enough. Technical Report MSU-CSE-08-1, Dept. of Computer Science, University of Texas at Dallas (2008), http://www.cse.msu.edu/cgi-user/web/tech/document?ID=888

[22] Aiyer, A.S., Lorenzo, A., Gouda, M.G.: Key grids: A protocol family for assigning symmetric keys. In: IEEE International Conference on Network Protocols (2006)

[23] Mittal, N.: Space-efficient keying in wireless communication networks. Technical Report UTDCS-26-07, Dept. of Computer Science, University of Texas at Dallas (2007)

[24] Bruhadeshwar, B., Kulkarni, S.: An optimal symmetric secret disribution for star networks. Technical Report MSU-CSE-07-196, Michigan State University (2007)

[25] Cidr report for (November 3, 2007), http://www.cidr-report.org/as2.0/