Routing Protocol Security Using Symmetric Key Based Techniques

Bezawada Bruhadeshwar and Kishore Kothapalli and M.Poornima and M. Divya
Center for Security, Theory and Algorithmic Research
International Institute of Information Technology
Gachibowli, Hyderabad 500 032, India.
Email: bezawada@iiti.ac.in, kkishore@iiti.ac.in, mpoornima@research.iiit.ac.in, divyam@students.iiit.ac.in

Abstract—In this paper, we address the security of routing protocols. Internet routing protocols are subject to attacks in the control plane as well as the data plane. In the control plane, a routing protocol, e.g., BGP, OSPF, exchanges routing state updates and enables routers to compute the best paths towards various destinations. During this phase, an attacker can modify or inject malicious control messages leading to incorrect computation of routing paths. In the data plane, the routers forward the data along the paths computed in the control plane. Even if an attacker is not successful during the control phase, he can choose not to use the correct routing paths and forward data along routes that benefit him. Research shows that, attacks on the control plane can be mitigated by ensuring message integrity and, attacks on the data plane can be mitigated by ensuring route integrity. Earlier works have addressed these two problems independently with many interesting solutions. However, due to the nature of these solutions, network architects cannot deploy security at both planes without increasing the overhead on the network. In this paper, we focus on an integrated approach and propose the use of symmetric key protocols for addressing the security at both the control and data planes. We describe approaches that enable the reuse of the symmetric key protocols thereby eliminating the need for separate solutions at different planes. We used symmetric key protocols as they are efficient and scalable. Our experimental results show that our approaches are practical and can be incrementally deployed as well.

Key Words. Security of Routing Protocols, BGP, OSPF, RIP, Symmetric Key Protocols

I. INTRODUCTION

Internet routing protocols like BGP, RIP, and OSPF are responsible for establishing and maintaining routing paths among the network hosts. The correct operation of these protocols is essential for the stability and reliability of the Internet. These protocols are designed to work in a completely trusted and open environment. Due to this reason, researchers focused on the stable and correct operation of these protocols in the presence of unintended faults such as misconfiguration. However, over the past few years, attacks on Internet routing protocols [1] have shown that complete trust on the Internet can be disastrous. Routers can behave in a malicious manner either due to economic motivations or to reduce load on their own networks. Such routers can either disrupt communications by creating routing loops or degrade network performance by poisoning the routing tables. Thus, there is a critical need for securing the Internet routing protocols from such threats that tend to cripple the core communication substrate of the Internet. Attacks on routing protocols can be launched either in the control plane i.e., the part where routers exchange control messages, or in the data plane i.e., the part where routers forward data along the computed paths.

One of the important attacks on the control plane of routing protocols is propagation of false routing updates. This can be done by either propagating false routing state information or by tampering the routing updates sent by other routers. In link state protocols like OSPF, an attacker can modify the link state advertisements of other routers. This results in incorrect routing table computation by the remaining routers [1], [2] thereby degrading network performance. In distance vector protocols like RIP, a malicious router can advertise false distance vectors, say, by advertising smaller hop counts to destinations. This enables the malicious router to become the preferred next hop for its neighboring routers and thereby, gaining access to sensitive data sent to many destinations. In path vector protocols like BGP, a malicious router can add (respectively, remove) AS (autonomous system) numbers in the AS_PATH field of the BGP path update message. If AS numbers are added into the BGP path update message, the downstream BGP routers will not prefer this falsified longer path and will choose paths that are, in reality, longer than those being advertised. From these examples, we note that, ensuring the authenticity of messages in the control plane can help to protect from such falsification attempts. Thus, ensuring message integrity and authentication are important requirements for securing the routing protocols.

Control plane security [3]–[5] can protect against active attackers or misconfigurations and eliminate early attempts to disrupt the route selection process. However, recent reports [6], [7], suggest that simply protecting the data in the control plane is insufficient to secure routing. An attacker can bypass the control plane measures and can target the data plane. In the data plane, the most important attack is that a malicious router forwards data along routing paths that are different from the paths that were advertised during the control phase. This is a major concern in BGP routers as the ASes to which these routers belong have specific policies for choosing a particular path. If the actual routing paths being used do not match the preferred routing paths then, for an AS this causes either monetary damage or affects its reputation. Hence, in addition to control plane security, there is need to verify the consistency of routing paths in the data plane.

From the above discussion, we make an observation that there is need for solutions that protect both control plane and data plane of routing protocols. Several solutions have existed independently for control plane [3]–[5] and for data plane [7], [8] security. However, due to the overhead involved in using both control and data plane security mechanisms simultaneously, it is a challenge to provide security at both planes simultaneously. In this paper, we make the first such attempt towards an integrated solution for securing routing.
protocols in both control and data planes. Our approach is simple: we reuse the solutions at control plane to secure the data plane. By reusing the solutions we reduce the overhead of combining two solutions. Towards this, we use symmetric key distribution protocols for achieving control plane security. We use our key distribution protocols the are described in [9] for authenticating messages at the control plane. These protocols have the following structure: a sender generates a pool of keys and distributes a unique subset of keys from this pool to each of the receivers. Now, to authenticate a message, the sender signs the message with each of the pool keys separately. Each receiver can now verify the authenticity of the message by generating the signatures corresponding to the subset of keys that he has and comparing them with the corresponding sender signatures.

Based on the authentication model described above, we use two forms of key distribution protocols for securing routing protocols. In the first type of key distribution protocol, each sender is responsible for generating and distributing the symmetric keys to the receivers. In the second type of key distribution protocols, a centralized authority is responsible for distributing the necessary keys to the users.

Our contributions are as follows.

- For securing BGP, we observe that, it is necessary to verify that a BGP update message has indeed passed through every node listed in the ASPATH field of the update message. Towards this, each node generates a signature on the update message before forwarding it to its peers. An intermediate router can validate this message by verifying all the signatures contained in the BGP update message. Furthermore, to reduce the cost of BGP update verification we leverage a practical trust relationship i.e., the trust between peer BGP routers. We show that the verification cost is reduced considerably using our trust model.

- For securing RIP, it is necessary to verify the hop counts reported by the neighboring routers. Towards this end, we use an authenticated query-response mechanism that enable a router to detect anomalies in the reported hop counts. Our approach can handle lying neighboring routers as well consecutive colluding routers.

- For protecting OSPF link-state updates, we observe that, the link-state information propagated by a router needs to be protected from tampering or falsification. We show the usage of the two types of key distribution protocols to secure the link-state updates. Also, since the key distribution protocols by nature provide source authentication, forgery attempts in OSPF can be detected as well.

- For data plane security, we enhance the solution from [7]. The solution in [7], requires an expensive off-line setup. We show that by using the symmetric key distribution protocols this cost can be reduced considerably.

Organization. In Section II, we describe the problem of securing routing protocols, some past solutions, and outline our threat model. In Section III, we describe our key distribution protocols from [9] and explain their applicability to secure the control and data planes. In Section IV, we evaluate our solutions empirically and discuss deployment issues. In Section V, we make concluding remarks and discuss future work.

II. PROBLEM OF SECURITY IN ROUTING

In Section II-A, we give a brief overview of the current Internet routing protocols, BGP, RIP, and OSPF and discuss security issues in these protocols. In Section II-B, we present an outline of past solutions.

A. Routing Protocols: Overview and Security Issues

The BGP protocol which is the current de-facto standard on the Internet, is used for inter-domain routing, i.e., routing among domains belonging to separate administrative control. Popular protocols like OSPF and RIP are widely used for intra-domain routing, i.e., routing within one or more ASes under the control of the same administrative domain. For the sake of simplicity, in this paper, we assume that an AS constitutes an independent domain.

Overview of Path Vector Protocols. The objective of a path vector protocol, e.g., BGP [10] (Border Gateway Protocol), is to advertise routing path information for IP prefixes. Towards this, BGP update messages are sent by BGP routers to advertise IP prefixes which are in their administrative control. A BGP update message can be viewed as a tuple, \( \{ \text{prefix}, \text{ASPATH} \} \), where the prefix denotes the IP prefix advertised by the originating BGP router. For example, in Figure 1, router A advertises the following information \( \{24.12.0.0/8, A\} \) to its neighboring routers, B and C. When an intermediate BGP router receives the update message it appends its AS number to the ASPATH field and forwards the update to its neighbors. For example, when router C receives the update from A, it appends its own AS number and sends \( C\text{A} \) to its neighbors D and E. Thus, the ASPATH variable contains the path using which any IP address belonging to the prefix can be reached.
of 16 (unreachable). This also means that any network using RIP cannot have a diameter more than 15 hops. Now, as the router receives periodic updates from other routers it updates the hop count to other destinations as well. In the convergence state, a router can reach every other router in the network. For example in Figure 2, router A only informs its neighbors B and C of its distance vector. Using this information, B can inform E that, router A is at a distance of one hop from itself. In Figure 2, we illustrate the distance vector information of A that progressively reaches other routers in the network.

Routing Table at A

<table>
<thead>
<tr>
<th>Dest.</th>
<th>Next Hop</th>
<th>#Hops</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>—</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>C</td>
<td>3</td>
</tr>
<tr>
<td>E</td>
<td>E</td>
<td>3</td>
</tr>
<tr>
<td>F</td>
<td>B</td>
<td>4</td>
</tr>
</tbody>
</table>

Routing Table at A

A(IP , NIL, 1) A(IP , A, 2) A(IP , A, 2) A(IP , B, 3) A(IP , B, 3)

Routing Security in Data Plane. To motivate routing security in data plane, consider the BGP protocol where routers choose routing paths based on the policies of AS. The policies are in turn based on service-level agreements, business contracts, reputation or observed path qualities. In spite of the security measures in the control plane, a malicious router can send data along routing paths that are different from those that were agreed upon in the control plane. A study of the Internet has shown that this is a realistic threat and almost 8% [6] of paths are inconsistent with those selected in the control phase. An attacker can benefit either financially or by way reducing load by such an attack. An AS may promise better quality routes during the control phase but use low quality paths for routing the data. Hence, it is important to verify the consistency of data routing paths with routing paths that were agreed upon during the control phase.

B. Past Solutions for Securing Routing Protocols

The RFCs that introduced routing protocols such as OSPF [15] and RIP [13] have added features like passwords and check-sums to authenticate control messages. However, packet sniffing attacks can weaken such mechanisms. Perlman [18] in her thesis identified two kinds of failures that can affect routing protocols: simple and Byzantine. To address these faults, Perlman proposed secure routing protocols using techniques from flooding, reserved buffers, link-state routing, and digital signatures [18]. In the following we discuss past work on securing individual routing protocols.

Huang et. al. [5], [17] proposed a key distribution scheme called double authentication to detect impersonation attacks to link state routing updates. In this scheme, each router shares two symmetric keys: one key with all its neighbours and one key with all its neighbours of neighbors. However, the scheme cannot tolerate any two colluding neighbours as such colluding neighbours can tamper with the signatures.

The distributed operation of distance vector protocols makes them more challenging to address security issues. Smith et. al. [4] proposed to add an additional field in update messages to check for routing loops. Hu et. al. [19] used hash chains and authentication trees but they do not address all distance fraud attacks. The work of Van Oorschot et. al. [14] proposed a reputation based framework to address all issues. This framework relies on, often misinterpreted, reputations built over a period of time and does not attempt to prevent control plane attacks.

There has been active research [3], [12], [20]–[22] interest in securing the Border Gateway Protocol (BGP). The possible security attacks on BGP protocol are documented in [3], [11]. Using public key cryptographic mechanisms, Kent et. al. [3] proposed a scheme that secures the control messages in BGP. However, public key mechanisms are computationally expensive [23], [24] and affect the performance of the heavily loaded BGP routers. Based on symmetric cryptosystems, Hu
et. al. [22] proposed a scheme called SPV, Secure Path Vector, to secure BGP update messages. Their scheme uses techniques such as hash chains, hash trees, and one time signatures. Their approach, though faster than SBGP [3], requires state maintenance and is susceptible to certain attacks [25]. Recently, Bezawada et. al. [26], proposed efficient symmetric key protocols for securing BGP using a trust based model. However, their work only addresses securing the BGP protocol.

**Data Plane Security.** Issues pertaining to data plane security are studied in [7] where the authors present a scheme to identify data plane attacks. This is done by verifying the presence of certain entities (such as ASes) called *provers* that can prove their presence and their predecessor on a given path. This scheme, as it does not rely on any cryptographic techniques apart from off-line key exchange, is an improvement over schemes such as secure traceroute [27] and stealth probing [28] that use cryptographic techniques. Recently, Hu and Mao have methods to use data plane information to validate occurrences of IP hijacking in real time [29]. Other techniques for data plane security include Listen-and-Whisper [8] and [30].

**C. Threat Model and Assumptions**

Our threat model is based on falsification attacks and data plane routing attacks that have been described in Section II-A. The types of falsification attacks we address are: generation of false routing state updates by spoofing source IP address and modification of routing state updates sent by other routers. In the data plane, our threat model focuses on inconsistent path usage by routers. We also consider some attacks on OSPF and RIP that cannot be addressed in the control plane. For example, in RIP, verifying the hop count to a particular destination is not possible in the control plane. We make little or no assumptions about trust in the network. We show that, the efficiency of our solutions increases if some amount of trust exists between routers. We treat misconfiguration of routers as a security compromise and address them accordingly. Also, for most part, our approaches focus on prevention and detection but not on the corrective action. The corrective actions are left to the individual networks or ASes depending on their policies.

**III. Our Solutions**

In Section III-A, we give an outline of the symmetric key distribution protocols (cf. Ref. [9] for more details) that we use for securing the routing protocols. In Section III-B, using these key distribution protocols, we describe our approaches for securing the protocols in the control and data planes. In all our solutions, we assume that an approach using publickeys, similar to the approaches used by [3], [22], exists for validating prefix ownership and authorization.

**A. Symmetric Key Distribution Protocols**

In our solution, we describe the use of two types of symmetric key distribution protocols: distributed and centralized protocols. In distributed key distribution protocols, there is no central authority and each router or AS distributes the necessary keys to other routers. In centralized key distribution protocols, a centralized controller or global authority establishes the necessary symmetric keys among the routers in the network.

<table>
<thead>
<tr>
<th></th>
<th>K1, K2, K3, K4, K5</th>
</tr>
</thead>
<tbody>
<tr>
<td>U1</td>
<td>K1, K3</td>
</tr>
<tr>
<td>U2</td>
<td>K2, K5</td>
</tr>
<tr>
<td>U3</td>
<td>K3, K4</td>
</tr>
<tr>
<td>U4</td>
<td>K2, K4</td>
</tr>
<tr>
<td>U5</td>
<td>K3, K5</td>
</tr>
<tr>
<td>U6</td>
<td>K2, K3</td>
</tr>
<tr>
<td>U7</td>
<td>K4, K5</td>
</tr>
<tr>
<td>U8</td>
<td>K1, K4</td>
</tr>
</tbody>
</table>

Fig. 4. Example key distribution for star network.

1) **Distributed Key Distribution Protocols:** To describe the distributed key distribution protocols we consider a star communication network [9], [31]. In star communication networks, a center node communicates with several satellite nodes and vice-versa. The satellite nodes do not communicate with each other. For this network, using the key distribution protocol from [9], the center node maintains a set of 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 \( p(k, l) \). To illustrate the \( p(k, l) \) key distribution, in Figure 4, we show a center node with 8 satellite nodes. The center node generates 5 symmetric secrets, which it will use for signing the messages. Now, from these secrets, the center node chooses a unique subset of size \( 2 \) and gives each such subset to a different satellite node. For example, in the figure, node \( U_1 \) receives the subset of secrets \( K_1, K_2 \) and node \( U_2 \) receives the subset of secrets \( K_3, K_5 \). Note that, the center node can support \( C(5, 2) = 10 \) users in this manner. Now, for authenticating a message, the center node generates message authentication codes with all the secrets it has and transmits these codes along with the message it has sent out. Each satellite node verifies those codes for which it has the generating secrets. For example, \( U_1 \) will be able generate and verify the message authentication codes generated using secrets \( K_1 \) and \( K_2 \).

In [9], the 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. But, 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 would allow \( C(40, 4) = 91390 \) satellite nodes. This protocol instance, for example, would be sufficient even for a BGP network which currently has approximately 26000 ASes [32]. Next, we describe the centralized key distribution protocols that have better storage than the distributed protocols.

2) **Centralized Key Distribution Protocols:** In routing protocols, every node is a center node for the routing updates it sends and vice-versa, it is a satellite node for route updates that it receives. Thus, for a network of \( N \) nodes, the storage required by each user is \( O(N \log N) \) using the distributed key distribution protocols. Although, this cost is tolerable for most routers, it may be desirable to reduce the storage cost. To reduce this cost, in [9] and other works [33], [34], the authors describe centralized key distribution protocols that require each user to store only \( O(\log^2 N) \) symmetric keys. All these key distribution protocols have two important features.
First, in the absence of collusion, they enable any two users to establish a secure channel between them. Second, they enable each user to authenticate messages that are broadcast to the network. We give an outline of our protocol from [9] which is an extension of the key distribution protocol described earlier for star networks. We assume that a central authority or global controller is in charge of establishing the protocol and, use the terms "central authority", "global controller", and "controller" interchangeably. The centralized key distribution protocol is designed in stages; the first stage extends the protocol $p(k, l)$ by considering a fully connected graph as a set of star graphs. This stage is intended for the case where the number of nodes in the graph is small. Then, stage 2 uses the scheme in stage 1 in an hierarchical manner. More details of the protocol can be found in [9]. In [9], [33], and [34], the authors have shown that the number of keys stored per user is $O(\log^2 N)$.

B. Securing Protocols

In this section, we proceed as follows: for each routing protocol we consider the control and the data planes, and show the usage of the symmetric key protocols for securing it.

1) Securing BGP: For securing BGP, we focus on using symmetric keys for authentication of path updates in BGP. In particular, the goal of authentication is to ensure the source of the advertisement and the integrity of the ASPATH field of the BGP update message. The main issue in validating an update message is to verify whether the update message has indeed traversed all the ASes listed in the ASPATH field. The distributed and centralized key distribution protocols from Section III-A, enable a single sender to authenticate messages by signing the message with his keys. However, in BGP, an update message can be sent to several neighbors and may be further forwarded by their neighbors, and so on. Due to this reason, it is required that any BGP router should be able to verify the signatures of any other BGP router. If we were to use the distributed key distribution protocol used then the amount of storage per BGP router is $O(N \log N)$ symmetric keys. This is an unreasonable amount of storage as the number of reported BGP router is about 26000 [32]. Hence, for securing BGP updates, we use the centralized key distribution protocols which only requires $O(\log^2 N)$ symmetric keys per router. We assume that a central authority assigns the logical identifiers to the BGP routers and issues 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 $A_1$ 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, $A_1$ signs a message consisting $\langle A_1 A_2 \ldots A_n \rangle$ and 24.12.0.0/8. Towards this end, $A_1$ 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 separately. Note that, $A_1$ does not use those keys which are necessary to verify signatures on messages sent by other users. Subsequently, to advertise the route, it sends a packet consisting of the following information (1) its ID, namely $A_1$, 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 packet is denoted as the signature block of the message. Whenever an AS receives this message, it uses the ID, say $A_1$, associated with the message to determine which keys should be used for verifying the signatures on this message. In particular, as specified in Section III-A, it identifies a collection of keys that it would use if it were to communicate with an AS with logical identifier $A_1$. 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.

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

Reducing the Cost of Signature Verification Based on Peer Trust. 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 [35]. In [26], the authors consider the threat model where at least one AS on any path of length h is trustworthy (although the exact trustworthy AS is unknown).

In this work, we use the following trust model that is more practical i.e., we leverage the trust among peer BGP routers. The BGP routers that are peers have a implicit level of trust which is built over time. This trust may come about due to service level agreements, business contracts or reputation. The peer BGP routers can take advantage of such a trust relationship to reduce the overhead of signature verification.

If no such trust exists among the peer BGP routers then, they can verify all the signatures that are contained in the BGP update message. Intuitively, if an AS $j$ trusts an AS $k$ then $j$ will accept any proofs checked by $k$ although $j$ itself may not have checked those proofs. For example, in Figure 1, when node E receives the route $DCA$ from D, it can verify that the route $DCA$ is indeed advertised by C by relying on the fact that D has verified the path up to C. If node E trusts D then the signature block of C would not be needed; node E will accept the fact that node D has verified that route $CA$ is indeed advertised by C before advertising the route $DCA$. Node E will only need to verify the signatures of D to validate the BGP update.

Furthermore, in the peer trust model, we note that, if the peer nodes need to verify the signatures using centralized key distribution protocol then, the cost of verification is $O(\log N)$ where $N$ is the size of the entire BGP network. To reduce this cost, in Figure 1, node D instantiates $p(k, l)$ with itself as the center node and the peer nodes (E,F and more if present) as satellite nodes. Thus, when node D instantiates $p(k, l)$ then, the number of verifications that need to be performed by the peers is only $O(\log d)$ where $d$ is the size of the peer neighborhood. However, since we are relying only on a peer trust model, the cost of signatures per BGP router cannot be reduced i.e., BGP router will have to add the signatures using the centralized key distribution protocol as well. These signatures will be needed by downstream BGP routers who might not share a peer trust
relationship with their neighbors. In Section IV, we show that our approaches are very efficient when compared to existing approaches even in the absence of the peer trust model.

**Data Plane Security for BGP.** For achieving data plane security we use a similar scheme as the scheme in [7], with one very important difference: we replace the expensive offline secret exchange with an online secret exchange. Recall that the centralized key distribution protocol also enables two BGP routers to establish a secure channel. Thus, if a verifier wants to check the validity of a path it first establishes a secure session key with the **prover** and uses this channel to exchange the secret information. The secret information can then be used according to the algorithm outlined in [7]. Thus, our centralized key distribution protocol enables control plane security and also, simplifies the process of data plane security.

2) **Securing RIP.** A major problem in RIP is to address distance-fraud attacks. To address this attack, we instantiate the distributed key distribution protocol for every router i.e., each router is considered as a center node and all other routers are satellite nodes. The key distribution can be done when the router joins the network. We use the distributed protocol as, the signature and verification is much faster than in the centralized protocol. Since a RIP router needs to process an update every 30 seconds during the control phase, our approach achieves the best speed possible for signatures and verifications. To address distance frauds, we use an attestation based approach i.e., distance advertised by routers need to be attested by their neighbors as well.

To illustrate, consider the scenario where a router $S$ sends an update of the form $(T, d_{ST})$ to all its neighbors which implies that $S$ can reach $T$ in $d_{ST}$ hops. To resist tampering attacks, we require that $S$ sign the message $(T, d_{ST})$ with each of its keys separately. We denote the set of signatures by $\text{Sign}_{K_A}(T, d_{ST})$. Now to validate this update, it needs to be attested by the next-hop on the path from $S$ to $T$ that $S$ claims to have a distance of $d_{ST}$. This is done as follows. Notice that for $S$ to send the above update, it must have received an advertisement from some neighbor of $S$, say $A$, that has a path of length $d_{ST} - 1$ to $T$. This message contains the signature of $A$ for the message $(T, d_{ST} - 1)$. So, $S$ can append this signature and send the following advertisement to its neighbours: $\langle (T, d_{ST}, \text{Sign}_{K_A}(T, d_{ST} - 1), \text{Sign}_{K_A}(T, d_{ST} - 1)) \rangle$. When a neighbour of $S$, say $X$, gets this update, it can verify the correctness of the update by verifying the signatures of $S$ and $A$. To forward the route to $T$, $X$ will similarly add its signature and pass on the signature of $S$. This way, the attestation by the next-hop on the advertised path can be used to authenticate the advertisements and protect against lying neighbors.

Notice, however, that the above scheme is not resistant to colluding neighbours. If such resistance is also required, then the above scheme can be extended to include attestation of $k$-hop routers for $k \geq 3$ easily. Moreover, since in RIP the maximum hop-count is set to 15, even adding attestations of all the 15-hop routers can be considered.

**Data Plane Security.** In the data plane, a router can suspect the validity of the hop counts reported during the control phase against the hop counts being used during the data plane. We use an authenticated query-response technique to detect incorrectly reported hop counts. We use the following approach. Assuming that a source router $S$ wishes to verify its $k$-hop neighborhood, $S$ generates a list of $k$-hop neighbors, signs this list using the keys from the centralized key distribution protocol. This list along with the signatures is broadcasted to the entire network. Note that, since RIP is a scalar protocol the distances are symmetric i.e., distance from $S$ to $T$ is the same as $T$ to $S$. Now, if any router say $T$ that is listed as a $k$-hop neighbor is actually closer or further away from $S$, will respond back to $S$ informing it of this inconsistency in the hop count. This will enable the $S$ to detect the anomaly and correct it accordingly.

3) **Securing the OSPF Protocol.** The main security concern in the OSPF protocol is the authenticity and integrity of the link-state advertisements (LSAs) which are sent in link-state update messages. The OSPF network resembles a fully connected communication graph where every router is a sender as well as a receiver. Thus, it is possible to use either one of the two symmetric key distribution protocols to ensure the authenticity and integrity of the link-state updates. We use the following approach to secure the OSPF link state updates. First, we instantiate a key distribution protocol depending on the router capabilities and use the protocol to secure the link-state updates.

If the OSPF routers of the concerned domain can tolerate a high storage then, we instantiate the distributed key distribution protocol. Every OSPF router instantiates a $p(k, l)$ with itself as the center node and the rest of the OSPF routers as the satellite nodes and distributes the keys to the corresponding satellite nodes. Since there are $N$ routers, in the distributed protocol, each node needs to store $O(N \log N)$ symmetric keys. The main advantage of the distributed protocol is the speed of signature generation and verification. If storage is a concern for the OSPF routers then, we instantiate the centralized key distribution protocol. For either key distribution protocols, the task of distributing the keys to the users can be taken up the domain administrator since the OSPF protocol is under a single administrative control.

To illustrate the process of computing signatures we consider the example OSPF network from Figure 3. For the sake of discussion, we consider that the link-state update only consists of the list of active neighbors of $A$ and the estimated link costs to reach them. In this case, node $A$ generates the following link state update: $\langle B, 3 \rangle$ and $\langle C, 2 \rangle$. Now, node $A$ uses the keys from the corresponding key distribution protocol being used and appends the signatures to the link-state update. This information is then flooded throughout the network. Any OSPF router can check for the authenticity and integrity of this message by verifying the corresponding signatures.

**Data Plane Security.** The approach to protect the data plane security of OSPF is similar to the approach used for protecting the data plane security of BGP. The main concern is to verify the validity of the paths used to a particular destination. Again, as we discussed earlier for BGP data plane security, we use the approach from [7] with our enhancement of online secret information exchange.

**IV. Experimental Analysis and Discussion**

In this section, we evaluate the efficiency of our approaches for securing BGP and OSPF protocols, provide security analysis and discuss their deployment. To compare the efficiency of our scheme, we evaluated it against the S-BGP protocol [3] and the SPV protocol [22]. We denote our distributed key distribution protocol as $\text{Distributed} – KD$ and, the centralized key distribution protocol as $\text{Central} – KD$. For BGP, we measured the signature generation and verification cost as the AS_PATH length increases. The signature generation costs
are shown in Figure 5(a). As can be seen from Figure 5(a), our approach has the best signature generation cost. In our approach, the signature cost does not depend on the ASPATH length but on the size of the network (close to 30000 as reported by [32]) and the degree of the nodes (about 16). In this setting, signature generation using our scheme takes about 32 $\mu$s, which is very small compared to that of SPV and S-BGP. The signature verification costs are plotted in Figure 5(b). In our approach, there are two possibilities based on the trust model: either the receiver checks all the signatures or the receiver verifies only the peer signatures. The cost of verifying all signatures is about 16 $\mu$s for a path length of 16 hops and that of verifying peer signatures is about 1 $\mu$s. In both cases, the simplicity and the small number of signature verifications to be performed, make our scheme an attractive choice.

To evaluate our approaches for the OSPF protocol, we compare with other existing schemes: the double authentication scheme of [5], [17], which we call $DA$, and the public-key signature scheme of [2], which we call $PK – Sign$. The signature generation cost is shown in Figure 6(a). As can be seen, the cost of digital signatures using PKI [2] is quite high compared to other symmetric key based solutions. Also, the double authentication scheme of [5], [17] requires the source to generate just 2 signatures which makes the cost of this scheme to be quite low. Our scheme has two variations. When using the Central – $KD$, the number of signatures to be generated varies with the size of the network, albeit slowly and is under 10 $\mu$s even for a network of size 200. When using Distributed – $KD$, this can be further reduced to under 5 $\mu$s. The verification costs are shown in Figure 6(b). The scheme of [5], [17] has a very low cost of verification since only 2 signatures are to be verified. However, both Distributed – $KD$ and Central – $KD$ have a cost of under 1 $\mu$s and are not very high compared to that of the double authentication scheme [5], [17]. As we noted earlier, the scheme from [5], [17], cannot handle collusion attacks whereas our approaches can handle collusion as well.

Security Analysis. Our analysis focuses on the cost of adopting our protocols and the cost of breaking our protocols. We use BGP and OSPF as case studies. Using the Central – $KD$, the number of keys an AS needs to maintain is $(\log N)^2$, where $N$ is the total numbers of ASes which is about 30000 [32] currently. Deploying the centralized approach on the Internet requires each AS to maintain approximately 225 secret keys, which can be easily stored in memory. According to the centralized approach, each update message needs to contain 225 signature blocks, each of 6-bytes that can be obtained by choosing the first 6-bytes from MD5/SHA-1/HMAC hash keeping in mind the BGP update message limit of 4KB. 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 (network diameter) such signatures, which is computationally infeasible. To illustrate security of Distributed – $KD$ consider an OSPF network of size 200 routers. Each node will attach about 8 (= log(200)) signature blocks of size say 20 bytes. Clearly, forging 160 bytes of signature is infeasible and thus, provides necessary security. One may use better hash functions to improve the security. For a detailed discussion on scalability issues as well as inter-operability we refer the reader to the discussion in [26].

V. CONCLUSION AND FUTURE WORK

We have addressed the problem of securing routing protocols in the control as well as data plane. We have presented symmetric key based solutions that can work at the control plane and can be reused for securing the data plane. Through experimental evaluation we have shown that our solutions are efficient and do not add to the overhead of the routing protocols. Moreover, compared to existing approaches, our solutions can handle collusive attacks among routers more effectively. Currently, we are working on the practical issues such as implementation and deployment of our protocols on the Internet.

REFERENCES

Fig. 6. Signature Generation and Verification Cost for OSPF

[... references are omitted for brevity ...]


