A Lightweight Scheme for Protecting *AS-PATH* Attributes of Update Messages

Bo Yang[a,b]
[a] School of Information Technology, Jiangxi University of Finance and Economics, Nanchang 330013, China
[b] High Level Engineering Research Center of Electronic-Commerce, Jiangxi Provincial Colleges and Universities, Nanchang 330013, China
Email: B.Yang@acm.org

***Abstract*— In existing schemes for protecting *AS-PATH* attributes of update messages, the security of S-BGP and BGPSEC has received wide acceptance. Yet, in S-BGP or BGPSEC, the number of signatures in a route attestation is linear in the length of *AS-PATH*, which is one of major hurdles of deploying in the real world and thus is an important and urgent problem. Existing schemes for solving this problem reduce the number of signatures by using aggregate signatures. Yet, its computation overhead of verifying *AS-PATH* is too heavy. In this paper, we present a scheme for reducing the number of signatures by using aggregate signatures, whose computation overhead of verifying *AS-PATH* can be significantly lowered. The presented scheme combines aggregate signature and Rabin signature. By it, the number of signatures in a route attestation is only one because aggregate signature is used. Moreover, differing from the existing aggregate-signature based schemes, the computation overhead of verifying *AS-PATH* is roughly only one multiplication each hop because Rabin signature is used. Computation overhead of verifying in presented scheme is roughly 1/2400 of that of existing scheme.**

***Index Terms*— *AS-PATH*, aggregate signature, bandwidth, update messages**

I. INTRODUCTION

There are various threats in the Internet. One of top known threats on today's Internet is the falsifying of update message of BGP (Border Gateway Protocol), which mainly includes IP prefix hijacking and *AS-PATH* tampering [1]. The *AS-PATH* tampering stems from one of BGP vulnerabilities.

The BGP [2] is the sole inter-domain routing protocol operating in today's Internet. Routers who run BGP are known as BGP speakers. Each BGP speaker communicates with other BGP speakers [1] [2]. Currently, there are about 4,3000 ASs (Autonomous Systems) in the Internet [3]. ASs exchange routing information through update messages. The two most important parts of an update message are: (1) address prefixes, which are destinations of advertised inter-domain paths; (2) the *AS-PATH*, which is an attribute of update message and used to specify a sequence of ASs along the path each identified by its AS number.

This work was supported in part by the National Nature Science Foundation of China (No. 10961013/A010206, 61262010).

In BGP, each AS prepends itself AS number to the *AS-PATH* when propagating an update message to a neighboring AS.

However, BGP's update messages are hardly authenticated when broadcasting among ASs, which leaves an opportunity for malicious attackers to fabricate false update messages by tampering the address prefixes or *AS-PATH* attribute. *AS-PATH* tampering occurs when the value of *AS-PATH* attribute is falsified.

The *AS-PATH* tampering can be illustrated by Fig. 1.

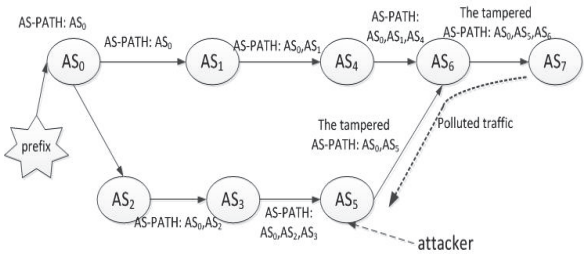


Figure 1. An example of *AS-PATH* tampering

In Fig. 1, the true *AS-PATH* advertised by AS_5 should be "$AS_0 \| AS_2 \| AS_3 \| AS_5$", where "$\|$" denotes a character used to connect two separated strings. Yet, AS_5, who is assumed to be malicious attacker, tampers this *AS-PATH* and advertises the tampered *AS-PATH* which is "$AS_0 \| AS_5$". As a result of this attack, the AS_6 and AS_7 are polluted because a shorter path is more prefered. Their traffic aimed to advertised prefixes, which should traverse through AS_4, will be mistakenly sent to AS_5.

This kind of attack can cause anything from an inconsequential annoyance to a devastating communications failure. For example, in critical applications (such as online banking, stock trading, and telemedicine run over the Internet), significant harm may arise if communication is lost at a crucial moment.

In existing schemes for protecting *AS-PATH* attributes of update messages, asymmetric cryptography based solutions have a great effect [1] [5] [4], some typical schemes of which are S-BGP [6], soBGP [18], psBGP [19], SPV [10] [5], BGPSEC [7] [9] [8] and so on.

In these schemes, the security of S-BGP [4] and BGPSEC [8] has received wide acceptance. The security is obviously the most important factor when designing schemes for authenting *AS-PATH*. In order to provide

© 2014 ACADEMY PUBLISHER
doi:10.4304/jnw.9.7.1707-1713

satisfying security, the direct reference scheme of the presented scheme is S-BGP or BGPSEC. Next, we further introduce the S-BGP or BGPSEC.

In S-BGP or BGPSEC, a digital signature will be created in each hop of *AS-PATH*. The number of signatures in a route attestation is linear in the length of the *AS-PATH* [5]. A route attestation in update message, which consists of several digital signatures signed by each AS when the update message traverses through this AS, is to protect *AS-PATH*. When an update message includes so many signatures, it will be significantly lengthened. Thus, S-BGP or BGPSEC improves security at the cost of significant lengthened update messages, which is one of major hurdles of deploying in the real world and thus is an important and urgent problem [6].

To solve above problem of S-BGP or BGPSEC, some documents manage to reduce the number of signatures by using aggregate signatures. Yet, these suggests mainly focus on the method based on bilinear map, whose computation overhead is too heavy [11].

For example, in a typical suggest [11], the general aggregate signature approach is based on a co-GDH signature scheme, which can be based on any gap group. The short signature scheme by Boneh, Lynn, and Shacham (BLS) is one such co-GDH signature scheme that makes use of elliptic curves. However, the computation overhead of this scheme is at least 10 times that of common schemes [5].

Aggregate signature based on RSA is simply mentioned [11]. However, RSA-aggregate-based scheme for protecting *AS-PATH* attributes of update messages is not given. Even if this kind of scheme is given, the computation overhead of verifying *AS-PATH* is roughly 15 multiplications each hop because RSA is used. Overall, in existing scheme for protecting *AS-PATH* attributes of update messages by using aggregate signatures, computation overhead of verifying *AS-PATH* is too heavy.

In this paper, we present a scheme for protecting *AS-PATH* attributes of update messages by using aggregate signatures, whose computation overhead of verifying *AS-PATH* can be significantly lowered. The presented scheme combines aggregate signature and Rabin signature. By it, the number of signatures in a route attestation, which is linear in the length of the *AS-PATH* in S-BGP or BGPSEC, is only one because aggregate signature is used. Moreover, differing from the existing aggregate-signature based schemes, the computation overhead of verifying *AS-PATH* is roughly only one multiplication each hop because Rabin signature is used. The pairing calculations in existing scheme for protecting *AS-PATH* attributes of update messages, whose computation overhead is too heavy, are avoided.

The rest of this paper is organized as follows. Rabin signature and the concept of aggregate signature are simply introduced in Section II. Our scheme for protecting *AS-PATH* by combining aggregate signature and Rabin signature is presented in Section III. In Section IV, we evaluate the presented scheme. Related work is simply introduced in section V. Finally, we present the conclusions and future work in Section VI.

II. Prerequisite Knowledge

The prerequisite knowledge introduced in this section includes the concept of aggregate signature and the Rabin signature.

A. The concept of aggregate signature

An aggregate signature is a digital signature that supports aggregation: given n signatures on n distinct messages by n distinct users using an aggregate signature algorithm, it is possible to aggregate these signatures into a single short signature. This single signature (and the n original messages) will convince the verifier that the n users did indeed sign the n original messages [11] [17].

B. The Rabin signature

The Rabin public-key signature scheme uses an even public exponent e. For the sake of simplicity, it will be assumed that e = 2.

In Rabin signature scheme described as following, the square root computation in the step 2 of *signature generation* is comparable to an exponentiation modulo n [12]. Rabin signature generation is not significantly more computationally intensive than an RSA signature generation with the same modulus size. Rabin *signature verification* is very fast when e = 2; it requires only one modular multiplication [12]. For the sake of its efficiency of verification, we use the Rabin signature to design the presented method of *AS-PATH* authentication.

The Rabin signature scheme is as follows. Note that the function R( ) called redundancy function, is a 1-1 mapping from message space to signature space.

The Rabin signature scheme [12]

Key generation

1. Generate two large distinct random primes p and q, each roughly the same size.
2. Compute n = pq
3. The public key is n; The private key is (p, q)

Signature generation

entity A signs a message m. Any entity B can verify A's signature and recover the message m from the signature. Entity A should do the following:

1. Compute R_m = R(m)
2. Compute s, which is a square root of R_m mod n (note that this square root can not be computed if the private key is not known). In other words, $s = (R(m))^{0.5}$ mod n.
3. A's signature for m is s.

Verification

To verify A's signature s and recover the message m, B should:

1. Obtain A's authentic public key n
2. Compute $R_m = s^2$ mod n
3. Recover $m = R^{-1}(R_m)$

Multiplications, which commonly play the most improtant role in computation overhead of signature and verification [12], are solely used calculations when we roughly compute computation overhead. From above description of Rabin scheme, we can see that there is only one multiplication when verifying a Rabin signature.

© 2014 ACADEMY PUBLISHER

III. THE SCHEME FOR PROTECTING *AS-PATH* BY COMBINING AGGREGATE SIGNATURE AND RABIN SIGNATURE

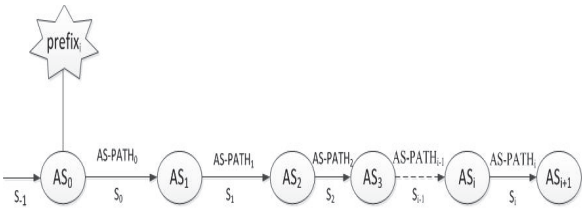


Figure 2. Creating a route attestation

The presented scheme can be divided into two parts: (1) The first part is to create a route attestation. A route attestation, which is to protect *AS-PATH*, consists of several digital signatures signed by each AS when the update message traverses through this AS. By using aggregate signature, these signatures are aggregated into a single signature. (2) The second part is to verify the route attestation, which is executed by AS_i.

When an AS receives a path announcement from last-hop AS, it should verify the sequence of relationship marks besides origin AS and AS-PATH by using related public keys. The presented method includes two algorithms. The first algorithm is executed on senders, the second one is executed on receivers. These two algorithms are as Algorithm 1 and Algorithm 2.

Next, we describe these two parts as follows. Some abbreviations in Algorithm 1 and Algorithm 2 can be seen in Table 1.

A. Creating a route attestation

Algorithm 1: AS_i creating a route attestation

1. Create the *AS-PATH*$_i$, which is "AS_0 ‖AS_1 ‖AS_2 ‖...‖AS_{i+1}"
2. Create the signed message, which is "s_{i-1}"$\widehat{\oplus}$*AS-PATH*$_i$
3. Compute R_m = R("s_{i-1}"$\widehat{\oplus}$*AS-PATH*$_i$)
4. Compute s_i, which is a square root of R_m mod n_i, by using itself private key (p_i, q_i). In other words, $s_i = (R(\text{“}s_{i-1}\text{”}\widehat{\oplus}\textit{AS-PATH}_i))^{0.5}$ mod n_i.
5. Send *AS-PATH*$_i$ and s_i to AS_{i+1}. Note that s_i is an aggregate signature as well as current route attestation

Figure 2 shows a typical process for *AS-PATH* and a route attestation transporting. In this figure, AS_0 is origin AS of prefix$_i$. Each AS receives a *AS-PATH* and a route attestation from its upstream AS, and sends its downstream AS a new *AS-PATH* and a new route attestation. For example, *AS-PATH* and route attestation received from AS_1 by AS_2 are *AS-PATH*$_1$ and s_1. *AS-PATH* and route attestation sent by AS_2 to AS_3 are *AS-PATH*$_2$ and s_2.

Shown as Fig. 2, after receiving an update message from AS_{i-1}, the AS_i draws s_{i-1} and *AS-PATH*$_{i-1}$. s_{i-1} can be verified by AS_i using Algorithm 2. After the s_{i-1} is verified, AS_i should create current route attestation s_i using Algorithm 1.

By executing Algorithm 1, the s_i aggregates its previous route attestation. In other words, the route attestation s_i aggregates the previous route attestation s_{i-1}; the s_{i-1} has aggregated s_{i-2}; the s_{i-2} has aggregated s_{i-3}; and so on. Note that s_{-1} is assumed to be 0. This assumed value will be used to judge whether verification of route attestation succeeds or not, which can be seen in the Step 3 of Algorithm 2. Next, by refering Fig. 2 and respective describing actions of various ASs in *AS-PATH*, we illustrate this inherent aggregating process in Algorithm 1.

(1) *The action of AS_0:* The initial value is s_{-1} which is assumed to be 0. In the beginning, the AS_0, which is origin AS of prefix$_i$, creates *AS-PATH*$_0$ according to the Step 1 of Algorithm 1. Obviously, the *AS-PATH*$_0$ is "AS_0". After executing the Step 2 of Algorithm 1, AS_0 has created the signed message $s_{-1}\widehat{\oplus}$*AS-PATH*$_0$, which is $0\widehat{\oplus}$"AS_0". After executing the Step 3 and Step 4, AS_0 has created initial route attestation, which is

$$s_0 = (R(0\widehat{\oplus}\text{“}AS_0\text{”}))^{0.5} \ mod \ n_0 = (R(0\widehat{\oplus}\ \textit{AS-PATH}_0))^{0.5} \ mod \ n_0 \quad (1)$$

AS_0 sends AS_1 an update message including s_0 and *AS-PATH*$_0$.

(2) *The action of AS_1:* From AS_0, AS_1 receives an update message including s_0 and *AS-PATH*$_0$. It firstly verifies s_0 according to Algorithm 2, where the public key of AS_0 is n_0. Next, it creates *AS-PATH*$_1$ according to the Step 1 of Algorithm 1. The *AS-PATH*$_1$ is "AS_0 ‖AS_1". After executing the Step 2, AS_1 has created the signed message $s_0\widehat{\oplus}$ *AS-PATH*$_1$, which is $s_0\widehat{\oplus}$"$AS_0 \parallel AS_1$". After executing the Step 3 and Step 4, AS_1 has created its route attestation, which is

$$s_1 = (R(s_0\widehat{\oplus}\text{“}AS_0 \parallel AS_1\text{”}))^{0.5} \ mod \ n_1 = (R(s_0\widehat{\oplus}\ \textit{AS-PATH}_1))^{0.5} \ mod \ n_1 \quad (2)$$

AS_1 sends AS_2 an update message including s_1 and *AS-PATH*$_1$.

(3) *The action of AS_i:* From AS_{i-1}, AS_i receives an update message including s_{i-1} and *AS-PATH*$_{i-1}$. It firstly verifies s_{i-1} according to Algorithm 2. Next, it creates *AS-PATH*$_i$ according to the Step 1 of Algorithm 1. The *AS-PATH*$_i$ is "$AS_0 \parallel AS_1 \parallel AS_2 \parallel ... \parallel AS_i$". After executing the Step 2, AS_i has created the signed message $s_{i-1}\widehat{\oplus}$*AS-PATH*$_i$, which is $s_{i-1}\widehat{\oplus}$ "$AS_0 \parallel AS_1 \parallel AS_2 \parallel ... \parallel AS_i$". After executing the Step 3 and Step 4, AS_i has created its route attestation, which is

$$s_i = (R(s_{i-1}\widehat{\oplus}\text{“}AS_0 \parallel AS_1 \parallel AS_2 \parallel ... \parallel AS_i\text{”}))^{0.5} \ mod \ n_i = (R(s_{i-1}\widehat{\oplus}\ \textit{AS-PATH}_i))^{0.5} \ mod \ n_i \quad (3)$$

AS_i sends AS_{i+1} an update message including s_i and *AS-PATH*$_i$.

Formula (3) is a recusion formula, the recusive end point of which is formula (1). In order to get formula of s_i for general term, we can iterate s_i from s_{i-1} to s_{-1} as following.

$$s_i = (R(s_{i-1}\widehat{\oplus}\ \textit{AS-PATH}_i))^{0.5} \ mod \ n_i$$

© 2014 ACADEMY PUBLISHER

TABLE I.
ABBREVIATIONS IN ALGORITHM 1 AND ALGORITHM 2

$\|\|$	a character used to connect two separated strings
AS_{i-1}	last AS in considered *AS-PATH*
AS_i	current AS in considered *AS-PATH*
AS-PATH$_{i-1}$	last *AS-PATH* which is "$AS_0 \|\|AS_1 \|\|AS_2 \|\|...\|\|AS_{i-1}$"
AS-PATH$_i$	current *AS-PATH* which is "$AS_0 \|\|AS_1 \|\|AS_2 \|\|...\|\|AS_i$"
s_{i-1}	last route attestation
s_i	current route attestation
AS_{i+1}	the next AS of AS_i
$\widehat{\oplus}$	XOR operation bit by bit. For example, 110001 $\widehat{\oplus}$ 011000 = 101001. Thus, A $\widehat{\oplus}$ B $\widehat{\oplus}$ B = A
$<n_0, n_1,... , n_i>$	the sequence of public keys corresponding to $<AS_0, AS_1,... , AS_i>$, which is known by each AS in *AS-PATH*
(p_i, q_i)	the private key of AS_i, which satisfies the equation $n_i=p_iq_i$ and corresponds to the public key n_i. This private key is known only by AS_i and used to execute the Step 4 of Algorithm 1.
R( )	a redundancy function, which is a 1-1 mapping from message space to signature space
R^{-1}( )	the inverse function of R( )

$$= (R(((R(s_{i-2} \widehat{\oplus} \textit{AS-PATH}_{i-1}))^{0.5} \bmod n_{i-1}) \widehat{\oplus} \textit{AS-PATH}_i))^{0.5} \bmod n_i$$

.....

$$= (R(((R(((R(0 \widehat{\oplus} \textit{AS-PATH}_0))^{0.5} \bmod n_0) \widehat{\oplus} \textit{AS-PATH}_1))^{0.5} \bmod n_1) ..... \widehat{\oplus} \textit{AS-PATH}_i))^{0.5} \bmod n_i \quad (4)$$

From formula (3) or formula (4), we can see that the length of s_i is equal to that of n_i. The n_i, which is the public key of AS_i, is about 1024 bits long. From formula (4), we can see that the s_i aggregates all of previous route attestations by iteratively aggregating. Thus, s_i is an aggregate signature which is about 1024 bits long. Note that a single short signature is also about 1024 bits long.

Therefore, by Algorithm 1, it is achieved to aggregate many signatures into a single short signature, and thus the size of related signed data can be significantly lowered and the bandwidth of update messages is low. Next, by Algorithm 2, we will see this single signature (and the n original messages) can convince the verifier that the n users did indeed sign the n original messages.

B. Verifying a route attestation

The AS_i draws *AS-PATH*$_{i-1}$ and s_{i-1} from received update message. In order to recover s_{-1} and make a judgement according to s_{-1}, the aggregate signature s_{i-1} should be peeled layer by layer according to Algorithm 2.

In Algorithm 2, the "cpath", "r", and "t" are all temporary variables for caching values of related intermediate results. The inner loop is used to construct intermediate *AS-PATH*, where AS_k# denotes the number of AS_k. The branch statement at the end of Algorithm 2 is used to make a judgement according to s_{-1}. Except the inner loop and branch statement, other statements in body of outer loop are used to recover s_{j-1} from s_j.

Note that comparing with Algorithm 1 being executed by each AS in *AS-PATH*, Algorithm 2 is only executed by AS_i. Next, by refering Fig. 3, we illustrate inherent process where the aggregate signature s_{i-1} is peeled layer by layer.

Algorithm 2: Verifying a route attestation

```
for (j = i − 1; j >= 0; j − −) {
        cpath = null
        for (k = 0; k <= j; j + +) {
        cpath = cpath || AS_k#
        }
        AS-PATH_j = cpath
        r = s_j^2 mod n_j
        t = R^{-1}(r)
        s_{j-1} = t ⊕̂ AS-PATH_j
}
If (s_{-1} == 0)
   then {we can draw a conclusion that the
        verification of route attestation succeeds}
   else {we can draw a conclusion that this
        verification fails and the update message
        should be rejected}
Endif
```

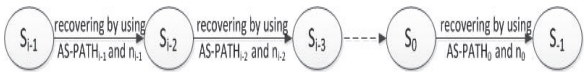


Figure 3. The iterative process for recovering various route attestations

(1) *Recovering s_{i-2} from s_{i-1}:* According to Algorithm 2, the current value of loop variate of outer loop is $j = i-1$. Firstly, AS_i constructs *AS-PATH*$_{i-1}$ by executing inner loop statements. The constructed *AS-PATH*$_{i-1}$ is "$AS_0 \parallel AS_1 \parallel AS_2 \parallel ... \parallel AS_{i-1}$". Secondly, s_{i-2} is recovered from s_{i-1} by executing the rest statements of outer loop, where *AS-PATH*$_{i-1}$ and public key of AS_{i-1} are used. The public key of AS_{i-1} is n_{i-1}.

(2) *Recovering s_{i-3} from s_{i-2}:* The current value of loop variate of outer loop is $j = i-2$. Firstly, AS_i con-

© 2014 ACADEMY PUBLISHER

structs *AS-PATH*$_{i-2}$ by executing inner loop statements. The constructed *AS-PATH*$_{i-2}$ is "$AS_0 \parallel AS_1 \parallel AS_2 \parallel ... \parallel AS_{i-2}$". Secondly, s_{i-3} is recovered from s_{i-2} by executing the rest statements of outer loop, where *AS-PATH*$_{i-2}$ and public key of AS_{i-2} are used. The public key of AS_{i-2} is n_{i-2}.

.....

(3) *Recovering s_{-1} from s_0:* The current value of loop variate of outer loop is $j = 0$. Firstly, AS_i constructs *AS-PATH*$_0$ by executing inner loop statements. The constructed *AS-PATH*$_0$ is "AS_0". Secondly, s_{-1} is recovered from s_0 by executing the rest statements of outer loop, where *AS-PATH*$_0$ and public key of AS_0 are used. The public key of AS_0 is n_0.

(4) *Make a judgement according to the recovered s_{-1}:* After the aggregate signature s_{i-1} being peeled layer by layer, s_{-1} is recovered. AS_i can make a judgement according to the s_{-1} by executing the branch statements at the end of Algorithm 2. If and only if s_{-1} is equal to 0, AS_i is convinced that each of ASs in *AS-PATH* did indeed signs various corresponding message and the signing process is in full compliance with Algorithm 1.

IV. Evaluate

In this section, we evaluate the presented scheme according to the security and the number of signatures in a route attestation.

A. Secure evaluation

Shown as Formula (4), current route attestation s_i is actually an aggregate signature. The verification of s_i, which is detailly described by Algorithm 2, can convince that each of ASs in *AS-PATH* did indeed signs various corresponding message and the signing process is in full compliance with Algorithm 1. So, if only verifiers strictly execute Algorithm 2, each AS in *AS-PATH* has to sign its corresponding *AS-PATH* according to Algorithm 1. Otherwise, verifications executed by these verifiers can not succeed.

Therefore, the presented scheme is secure and can protect *AS-PATH* attributes of update messages.

B. The number of signatures in a route attestation

As shown in Section I, in S-BGP or BGPSEC, the number of signatures in a route attestation is linear in the length of the *AS-PATH*, which is an important and urgent problem.

Shown as Formula (4), in presented scheme, the number of signatures in a route attestation is one. In existing aggregate signature based scheme [11] for protecting *AS-PATH*, the number of signatures in a route attestation is only one, which is same as the presented scheme.

Therefore, benefitting from aggregate signature, existing scheme [11] and presented scheme can all solve above problem inherent in S-BGP or BGPSEC.

C. Computation overhead of verifying AS-PATH

Existing related schemes are based on bilinear map, whose computation overhead is too heavy [11]. The computation overhead of a bilinear map is roughly 10 common exponentiations, which roughly consists of 2,400 common multiplications [5].

In presented scheme, shown as Algorithm 2, the computation overhead of verification each hop is roughly one multiplication. This one multiplication is from the statement "r = s_j^2 mod n_j" which is component of verification of Rabin signature.

Therefore, computation overhead of verifying in presented scheme is only $1/2400$ of that of existing scheme.

Aggregate signature based on RSA is simply mentioned in current document [11]. However, RSA-aggregate-based scheme for protecting *AS-PATH* attributes is not given. Even if this kind of scheme is given, the computation overhead of authenticating *AS-PATH* is roughly 15 multiplications each hop because RSA signature is used, which is 15 times as much as that of presented scheme.

The propogation delay is mainly up to the computation overhead of authenticating *AS-PATH*. We use the SSFNet (Scalable Simulation Framework) simulation tool to compare the propogation delay of existing mechanisms and that of our mechanism. The simulation result is as Fig. 4 and Fig. 5.

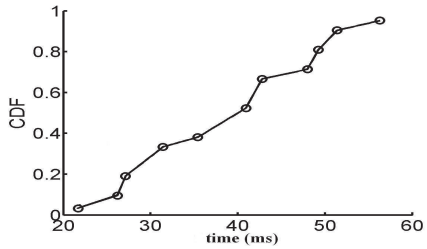


Figure 4. The propogation delay of existing related schemes

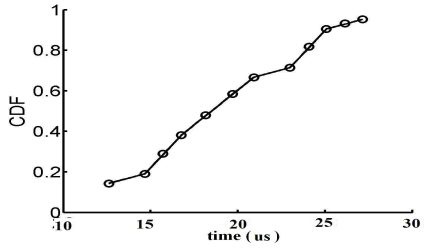


Figure 5. The propogation delay of proposed scheme

From above result shown as Fig. 4 and Fig. 5, we can see that propogation delay in the proposed scheme is much lower than that of existing related schemes.

V. Related work

there is great interest in protecting *AS-PATH* attributes of update messages, as it is essentially the glue that holds the disparate parts of the Internet together [1] [5] [4]. Many related schemes have been proposed [13] [14] [15] [16] [20]. In these schemes, asymmetric cryptography

© 2014 ACADEMY PUBLISHER

based solutions have a great effect, some typical schemes of which are S-BGP [6], soBGP [18], psBGP [19], SPV [10] [5], BGPSEC [7] [9] [8] and so on.

S-BGP [6] implements security by validating path attributes in update messages passed among ASes through the use of digital signatures and associated public key certificates.

The soBGP [18], psBGP [19], SPV [10] aim at relieving the burden of S-BGP. Yet, the relieving brings about unacceptable loss of security. Thus, the scheme, whose security has received wide acceptance, is only S-BGP [4] and BGPSEC [8]. To a great extent, BGPSEC [8] can be regarded as the reiterating and refining for the *AS-PATH* authentication of S-BGP. The security is obviously the most important factor when designing schemes for authenting *AS-PATH*. Next, we further introduce the S-BGP or BGPSEC.

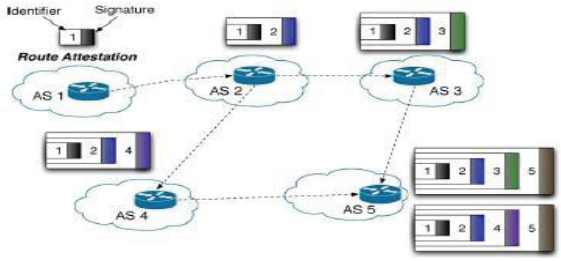


figure 6. Route attestations in S-BGP or BGPSEC [5] [6] [8]

In S-BGP or BGPSEC, route attestations, which are digital signatures and are used to protect the *AS-PATH*, are distributed in a modified update message as a new attribute. A route attestation is signed by each AS as it traverses the network. All ASs on the path sign previously attached signatures. Hence, the validator can validate not only the path, but also that (1) the ASs were traversed in the order indicated by the path, and (2) no intermediate ASs were added or removed by an adversary. fig. 6 shows a simplified use of route attestations as they propagate between ASs [5].

In S-BGP or BGPSEC, the whole *AS-PATH* authentication can be divided into two parts. (1) The first part is to create a route attestation, which is digital signatures signed by each AS when the update message traverses through this AS. A typical singed message is the "current AS-PATH$\|$current AS#", where "$\|$" is a separator. (2) The second part is to verify a route attestation, which is executed by current AS.

According to fig. 6, we can know in each hop of *AS-PATH*, a digital signature will be created. As a result, the AS_5 will receive 3 signatures from AS_4 or from AS_3, and will send 4 signatures to its downstream neighbors. The number of signatures in a route attestation is linear in the length of the *AS-PATH* [5]. When an update message includes so many signatures, it will be significantly lengthened. Thus, S-BGP or BGPSEC improves security at the cost of significant lengthened update messages, which is one of major hurdles of deploying S-BGP in the real world and thus is an important and urgent problem [6].

To solve above problem of S-BGP or BGPSEC, some documents manage to reduce the number of signatures by using aggregate signatures. Yet, these suggests mainly focus on the method based on pairing calculations, whose computation overhead is too heavy [11]. Aggregate signature based on RSA is simply mentioned [11]. However, RSA-aggregate-based scheme for protecting *AS-PATH* attributes of update messages is not given. Even if this kind of scheme is given, the computation overhead of verifying *AS-PATH* is roughly 15 multiplications each hop because RSA is used. Overall, in existing scheme for protecting *AS-PATH* attributes of update messages by using aggregate signatures, computation overhead of verifying *AS-PATH* is too heavy.

VI. Conclusions and Future Work

In this paper, by combining aggregate signature and Rabin signature, we present a lightweight scheme for protecting *AS-PATH* attributes of update messages. Its computation overhead of verifying *AS-PATH* is only roughly one multiplication each hop because Rabin signature is used. Its number of signatures in a route attestation is only one because aggregate signature is used. Comparing with existing scheme for protecting *AS-PATH* attributes of update messages by using aggregate signature, the computation overhead of verifying *AS-PATH* is significantly lowered.

Future work will cover the extension of the presented scheme from *AS-PATH* authentication to origin AS authentication, so as to achieve a whole lightweight scheme for securing BGP. The aggregate signature will also be used in the whole scheme.

Acknowledgment

The authors are grateful to the anonymous referees for their valuable comments and suggestions to improve the presentation of this paper. This work was supported in part by the National Nature Science Foundation of China (No. 10961013/A010206), Foundation of High Level Engineering Research Center of Electronic-Commerce of Jiangxi Provincial Colleges and Universities, Science and Technology Foundation of the Education Department of Jiangxi Province (No. GJJ12735), and Natural Science Foundation of Jiangxi (No. 20132BAB201036).

References

[1] G. Huston, M. Rossi, and G. Armitage, Securing BGP - A Literature Survey[J], Communications Surveys Tutorials, IEEE, 13(2), pp. 199-222, 2011.
[2] Y. Rekhter, T. Li, and S. Hares, A Border Gateway Protocol 4 (BGP-4), RFC 4271 (Draft Standard), Internet Engineering Task Force, Jan. 2006.
[3] T.Bates, P.Smith et al., the cidr report[OL], http://www.cidr-report.org. (Accessed on Feb. 2013)
[4] Song Li, Jianwei Zhuge, Xing Li. Study on BGP security. Journal of Software, 2013, 24(1):121-138 (in Chinese)
[5] K. Butler, T. Farley, P. McDaniel, J. Rexford. A survey of BGP security issues and solutions [J]. Proceedings of the IEEE, 2010, 98(1): 100-122

© 2014 ACADEMY PUBLISHER

[6] Kent S, Lynn C, Seo K. Secure border gateway protocol (S-BGP) [J]. IEEE Journal on Selected Areas in Communications, 2000,18(4): 582-592
[7] G. Huston, G. Michaelson, Validation of Route Origination in BGP Using the Resource Certificate PKI and ROAs[R], IETF RFC 6483, February 2012.
[8] Lepinski M, BGPSEC protocol specification[R], IETF Internet-Draft, February 2013.
[9] Matthias Wahlisch, Olaf Maennel, Thomas C. Schmidt, Towards Detecting BGP Route Hijacking using the RPKI[C], Proceeding of ACM Special Interest Group on Data Communication (SIGCOMM), Helsinki Finland, pp. 103-104, August 2012.
[10] YihCChun Hu, Adrian Perrig, and Marvin Sirbu, SPV: Secure path vector routing for securing BGP, in Proc. of the ACM SIGCOMM, Yavatkar, Washington, 2004, pp. 179-192.
[11] Meiyuan Zhao, Sean W. Smith, David M. Nicol. Aggregated path authentication for efficient BGP security[C]. Proceeding of the 12th ACM Conferernce on Computer and Communication Security (CCS), Alexandria USA, Nov. 2005, pp. 128-138.
[12] Cormen, Thomas H., Charles E. Leiserson, Ronald L. Rivest, Clifford Stein. Introduction to Algorithms (the 2nd edition) [M]. MIT Press and McGraw-Hill. 2001.
[13] Zhongjian Le, Naixue Xiong, Bo Yang, Yuezhi Zhou. SC-OA: A secure and efficient scheme for origin authentication of inter-domain routing in cloud computing networks[C]. Proceeding of the 25th IEEE International Parallel & Distributed Processing Symposium (IPDPS), Anchorage USA, May 2011, pp. 243-254.
[14] Bo Yang. A Method for Securing the Announcements and Withdrawals of Inter-domain Routes[J]. Journal of Chinese Computer Systems, 2012, 33(2): 357-364 (in Chinese)
[15] Zhongjian Le, Bo Yang. A Certificate-less Mechanism for Preventing IP Prefix Hijacks among Autonomous Systems[J]. Journal of Chinese Computer Systems, 2011, 32(12): 2424-2429 (in Chinese)
[16] S. Goldberg, M. Schapira, P. Hummon, J. Rexford. How secure are secure interdomain routing protocols[C]. Proceeding of ACM Special Interest Group on Data Communication (SIGCOMM), New Delhi India, August 2010, pp. 28-39.
[17] Kyle Brogle, Sharon Goldberg etal., Sequential Aggregate Signatures with Lazy Verification from Trapdoor Permutations[OL], http://eprint.iacr.org/2011/222.pdf, September 2011.
[18] Russ White, Securing BGP through SecureOrigin BGP (soBGP)[OL], http://www.cisco.com/web /about/ac123/ac147/archived_issues/ipj_6-3/securing_bgp_sobgp.html, 2006.
[19] Oorschot PC, Wan T, Kranakis E, On interdomain routing security and pretty secure BGP (psBGP) [J], ACM Transactions on Information and System Security, 10(3), pp. 11-22, 2007.
[20] P. McDaniel, W. Aiello, K.Butler, J. Ioannidis, Origin Authentication in Interdomain Routing[J], Computer Networks, 50(16), pp. 2953-2980, August 2006.

Bo Yang received the Bachelor's degree from the Huazhong Universityof Science and Technology, and the M.S. and Ph.D. degrees in computer science from the University of Nanchang, China. He is currently a faculty at the Department of Computer Science and Technology, School of Information Technology, Jiangxi University of Finance and Economics, Nanchang, China. He has worked at the Institute of Software, Chinese Academy of Sciences. His research interests include Internet routing and network security.

© 2014 ACADEMY PUBLISHER