Routing in Trustworthy Networks with Partial Deployments of SAVA Nodes

Rongxi He *Member, IEEE*, Limin Song, and Bin Lin, *Member, IEEE*

***Abstract*—With the sound mechanism of Source Address Validation Architecture (SAVA), every packet received and forwarded in a trustworthy network has been ensured to hold an authenticated source IP address, which can prevent network attacks with spoofed source addresses. However, it is impossible to deploy SAVA all over the Internet in one night. In this paper, we investigate the SAVA mechanism from the point of routing for trustworthy networks with partial deployments of SAVA nodes. We first describe and compare three different routing policies for trustworthy networks, and then propose a new routing algorithm, called Minimum Hop to First SAVA node algorithm (MHFS). Extensive simulations show that MHFS can not only guarantee that each route for packets includes at least one SAVA node, but also achieve significant improvements in success probability for routing packets and resource utilization while considering the loop prevention and the load balance for SAVA nodes.**

***Index Terms*—Trustworthy Networks; Routing Policy; Source Address Validation Architecture (SAVA); Blocking Probability; Load Balance**

I. Introduction

TODAY's Internet is a decentralized system with the fundamental principles of best-effort and destination address based packet forwarding. Due to its lack of source IP address validation in the packet forwarding process, it is very easy for attackers to forge the originating IP host address to evade responsibility for their malicious packets [1-6]. As indicated in MIT Spoofer project [7], a large portion of the Internet is vulnerable to source address spoofing. It has been recognized that packet source IP address validation is one of the most important challenges for a trustworthy network [5, 6]. Recently, mechanisms related to the validation of source IP addresses, such as cryptographic authentication, proactive filtering and reactive trace-back, are gaining a considerable attention from the research and engineering community [6, 8-10]. However, the incentive for ISPs to deploy these mechanisms is relative low, and the incremental deployment is not well supported, which hinder the mechanisms to be widely deployed in the Internet. In [2, 4, 6, 11, 12], the authors have proposed a feasible mechanism, called Source Address Validation Architecture (SAVA), to ensure that every packet received and forwarded must hold an authenticated source IP address. Moreover, SAVA is applicable for IPv4 networks and IPv6 networks with many additional benefits, including network management and accounting with fine granularity, a simplified authentication of the application, and the accelerated deployment of new Internet applications such as P2P applications and other large scale multimedia applications [3, 5]. In SAVA, any packet without holding an authenticated source address will be dropped by the SAVA router, and not be forwarded to the next hop. Therefore, it is impossible to launch network attacks with spoofed source addresses. With consideration of the hierarchical architecture of Internet, the SAVA mechanism can be organized in a hierarchical way. It supports incremental deployment and is beneficial even if deployed only in a single autonomous system of the Internet [4-6]. SAVA may greatly improve network security, management, accounting, and new applications. It is feasible that SAVA will support a new, more secure and sustainable Internet [4, 6].

Since it is impossible to deploy SAVA all over the Internet in one night, it is worthwhile for [8] to investigate the route selection in a single-domain network with a partial deployment of SAVA nodes. In their seminal work, the authors mainly focused on how to guarantee a route passed through a SAVA node. Their proposals are unbeneficial to choose a path with minimum hop from the source to the SAVA node. Since the packets are validated by SAVA nodes, and any packet without an authenticated source address will be dropped, it consumes valuable resource in the networks and in the intermediate routers that have to process it before it is dropped at a SAVA node due to being checked with a spoof address. Obviously, more hops passed before it reaches a SAVA node, more

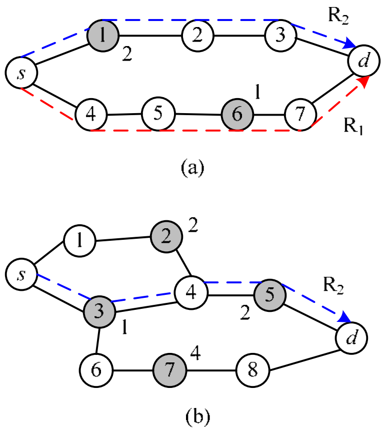
Manuscript received November 17, 2010. This work was supported in part by Natural Science Foundation of Liaoning under grant No. 20082142, Science Research Program of Educational Department of Liaoning under grants No. 2008S024 and No. 2008S027, Liaoning BaiQianWan Talents Program under grant No. 2009921063, Open Research Fund of Key Lab of Optical Fiber Sensing & Communications (UESTC), Ministry of Education, and Open Research Fund of National Mobile Communications Research Laboratory, Southeast University under grant No. W200908.

Rongxi He is with College of Information Science and Technology, Dalian Maritime University, Dalian 116026, China, National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096, China, and Key Lab of Optical Fiber Sensing & Communications (UESTC), Ministry of Education, Chengdu 611731, China (corresponding author. phone: 86-411-84723130; fax: 86-411-84723886; e-mail: hrx@dlmu.edu.cn).

Limin Song is with College of Information Science and Technology, Dalian Maritime University, Dalian 116026, China (e-mail: slm@dlmu.edu.cn).

Bin Lin is with College of Information Science and Technology, Dalian Maritime University, Dalian 116026, China (e-mail: linbin@dlmu.edu.cn).

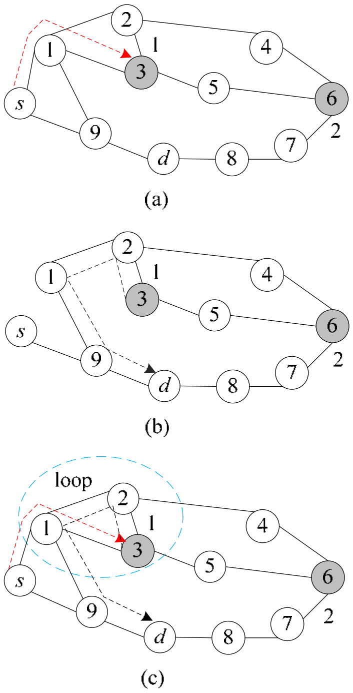
The preliminary work [1] of this paper was presented at the 2nd International Conference on Advanced Computer Control, March 2010.



(a)

(b)

Fig. 1 Routing selection in a trustworthy network with a partial deployment of SAVA nodes [1]



(a)

(b)

(c)

Fig. 2 Possibility of loop in the routing selection algorithms in [8]

resource being wasted. In order to improve the resource utilization, it should minimize the hop number from the source to the first SAVA node for routing packets. Otherwise, for a packet with a spoof address, more resource will be consumed before it reaches a SAVA node where it will be discarded. Fig. 1 gives an example for routing in a trustworthy network with a partial deployment of SAVA nodes, where the SAVA nodes are denoted as gray and the number beside a SAVA node depicts its load that is defined as the number of packets handled by it according to [8]. In Fig. 1 (a), two paths R_1 and R_2 are available between the source *s* and the destination *d*, i.e., *s*-4-5-6-7-*d* and *s*-1-2-3-*d*. For R_1, the total hop from the source to the destination is 5 and the hop from the source *s* to the SAVA node 6 is 3, while for R_2, the total hop is 4 and only one hop from the source to the SAVA node 1. Since the role of SAVA is to drop packets with spoof addresses as close to the source node as possible in order to prevent malicious packets and to improve the resource utilization, the second path R_2 is advantage over the first one R_1. However, according to [8], R_1 will be chosen due to the overemphasis on the load balance of SAVA nodes. In addition, the work in [8] restricted to choose paths involving only one SAVA node, which obviously reduce the chance to find available paths for packets between the source and the destination. Fig. 1(b) gives an example for this case, where there is no path with only one SAVA node between the source *s* and the destination *d*. According to [8], all packets cannot be routed from the source to the destination due to no path being chosen, although there are three paths between the source and the destination, i.e., *s*-1-2-4-5-*d*, *s*-3-4-5-*d*, and *s*-3-6-7-8-*d*. For a given network, more packets being routed mean higher resource utilization. In order to increase the chance to find available paths for packets, it is better to choose the path with as small number of SAVA nodes as possible, not to forbid the use of path with more than one SAVA node. Therefore, in Fig. 1(b), with a joint consideration of resource utilization and load balance, among the three available paths, the route R_2, i.e., *s*-3-4-5-*d*, can be used to route packets between the source *s* and the destination *d*. Furthermore, the routing algorithms in [8] divide the routing selection problem with SAVA requirements into two steps, i.e., first to compute a segment from the source to one of the SAVA nodes with a specific optimal objective, and then to compute another segment form the chosen SAVA node to the destination. In the second step, the proposals only delete the links included in the chosen segment of the first step to ensure that those links cannot be involved in the second segment for the loop prevention. However, it is possible that the nodes in the first segment from the source to the chosen SAVA node may be involved in the second segment from the SAVA node to the destination, which may result in a loop problem. On this account, the proposals in [8] do not eliminate the possibility of loops. Fig. 2 gives an example for this case, where the gray node 3 is the SAVA node and the process to compute a route between the source *s* and the destination *d* is also depicted. Fig. 2(a) shows the first segment from the source *s* to the SAVA node 3 computed according to [8], i.e., *s*-1-3. According to [8], before to compute the second segment, all links including in the first segment will be removed. The residual topology is shown in Fig. 2(b). Dijkstra's algorithm is used again to compute the shortest path from the SAVA node 3 to the destination *d*, which is 3-2-1-9-*d*. The final path chosen to route the packets between the source *s* and the destination *d* is *s*-1-3-2-1-9-*d*. It is evident that a loop exists in the chosen path, which is depicted in Fig. 2(c).

The main objective of this paper is to investigate the SAVA mechanism from the point of routing for an autonomous trustworthy network with a partial deployment of SAVA nodes. We first analyze the routing policy in the trustworthy network, and then propose an improved loop-free routing algorithm, called Minimum Hop to First SAVA node algorithm (MHFS), which jointly consider the load balance for SAVA nodes and the improvements of success probability for routing packets and the utilization of network resources. Our work differs from the previous work [8] in that we not only focus on the problem to route packets along an available path involving a SAVA node, but also achieve considerable improvements in resource usage and the blocking probability for routing packets while with the considerations of loop prevention and load balance among SAVA nodes.

The rest of the paper is organized as follows. Section 2 elaborates on the network model and the routing policies. Section 3 describes the proposed algorithms. Simulation results are presented in Section 4. Conclusions follow in Section 5.

II. System Model and Problem Analysis

A. Network Model

Define a network topology G (N, L, S) for a single-domain trustworthy network, where N is the set of nodes, L is the set of links, and S is the set of SAVA nodes. $|N|$, $|L|$ and $|S|$ denote the node number, the link number and the SAVA node number, respectively. It is evident that $|S|\leq|N|$. Assume that S_k denotes the kth SAVA node (k=1, 2,…, $|S|$). All packets arrive at the network dynamically, and each packet will be routed across a trustworthy path for the source address validation that involves at least one SAVA node between the source and the destination. Therefore, for each source-destination node pair, the source node will select one of the SAVA nodes dynamically to establish a virtual channel (i.e., a trustworthy path) to the SAVA node and from the SAVA node to the destination node. After finishing the transmission of the packets between the source-destination pair, the virtual channel can be released. Assume that there is only one virtual channel setup request (also called route request in the later content) arriving at the network at a time, defined by $r(s, d)$, where s, $d \in N$ denote the source node and the destination node, respectively. Without loss of generality, assume that $s, d \notin S$ [1]. Some notations are introduced as follows.

(i, j): a link between node i and node j in G.

c_{ij}: the basic cost of link (i, j). It is determined by many factors, such as physical length of the corresponding link, the installation cost of the link, and so on.

c'_{ij}: the cost of link (i, j). It is determined by the routing policy and the current state of the network.

l_i: the traffic load of SAVA node i, that is, the number of packets handled by the SAVA node i.

b_{ij}: the adjustable cost of link (i, j) with SAVA nodes, where node i or node j is a SAVA node. It is determined by (1) as follows.

$$b_{ij} = \begin{cases} l_i + l_j + 2, \text{if } i, j \in S \\ l_i + 1, \text{if } i \in S \\ l_j + 1, \text{if } j \in S \end{cases} \tag{1}$$

$P(s, S_k)$: the route from the source s to the SAVA node S_k.

$P(S_k, d)$: the route from the SAVA node S_k to the destination d.

T: a set containing partial SAVA nodes. It is a subset of S, and $|T|\leq|S|$, where $|T|$ denotes the node number of T.

B. Routing Policy

In a trustworthy network, in order to validate whether or not a packet holding an authenticated source address, each packet will be transmitted across a path involving at least one SAVA node. Upon the arrival of a route request $r(s, d)$, a path involving SAVA nodes between the source s and the destination d should be computed. A routing policy reflects the intentions of the network operators, and determines how to route a packet in the network. Since the role of SAVA is to discard spoofed traffic as close to the source as possible, it is suitable to choose the path with the minimum hop from the source to the first SAVA node. In addition, the routing policy should consider resource utilization and load balance. In a trustworthy network, following policies can be used to route packets.

Policy 1: Choose a path involving at least one SAVA node between the source s and the destination d;

Policy 2: Choose a path with minimum hop from the source to the first SAVA node between the source s and the destination d;

Policy 3: Choose a path with minimum hop between the source s and the destination d;

Policy 4: Choose a path involving minimum number of SAVA nodes between the source s and the destination d;

Policy 5: Choose a path involving SAVA nodes with minimum load traffic between the source s and the destination d;

Policy 1 is required by the SAVA mechanism to guarantee that the source IP address of each packet is checked, which is compulsory for routing in trustworthy networks. Policy 2 is beneficial to reduce the resource consumed by packets with a spoof address being discarded by SAVA nodes. Policy 3 is helpful to route packets to the destination across minimum hop, and is also useful to occupy less resources. Policy 4 is useful to reduce the burden of SAVA nodes, and Policy 5 is favorable for the load balance of SAVA nodes. Some different routing algorithms can be achieved by combining the various policies in different priority order. In some situations, if none of the policies can found a path, then the request will be blocked. In Section 3, we will propose an efficient route algorithm with a joint consideration of above policies to enhance the successful probability for routing packets and to improve the resource utilization while considering load balance for SAVA nodes.

[1] If $s, d \in S$, it is only need to choose a minimum cost path for packets, since the source or the destination can be used to check the source IP addresses.

III. Description of Routing Algorithm

Based on above analyses, an effective loop-free routing algorithm, called Minimum Hop to First SAVA node algorithm (MHFS), has been proposed for trustworthy networks. MHFS uses an adjustable parameter to reflect the focus whether on resource utilization or on load balance. By assigning a suitable value for the parameter, we can make a tradeoff between resource utilization and load balance. The process of MHFS is specified as follows.

Step 1: Input the network topology *G* and initialize the set of SAVA node *S*.

Step 2: Wait for a route request *r*(*s*, *d*). For each arrival request, let *k*=1, *T=NULL*, and go to Step 3.

Step 3: If *k*>|*S*|, go to Step 4; otherwise, modify the cost of link in *G* according to (2), and compute an available path by Dijkstra's algorithm from the source *s* to S_k. If a path has been found, record the path in $P(s,S_k)$ and record S_k in *T*. Let *k*=*k*+1, and go back to Step 3.

$$c'_{ij} = \begin{cases} +\infty, \text{if } i = d \text{ or } j = d \\ c_{ij}, \text{if } i \notin S \text{ and } j \notin S \\ c_{ij}, \text{if } i = S_k \text{ or } j = S_k \\ c_{ij} + \alpha \cdot b_{ij}, \text{others} \end{cases} \tag{2}$$

where *α* is a positive constant with the consideration of tradeoff between load balance and resource utilization. The bigger *α* is, the bigger the proportion of b_{ij} in c'_{ij} is, and the link cost depends mostly on the load of its involving SAVA nodes. With a big *α,* the algorithm is favorable to choose a path with small number of SAVA nodes, and is also beneficial to choose a path involving SAVA nodes with light load. According to (2), it is impossible that the computed segment from the source *s* to the SAVA node S_k includes the destination *d*, which is also helpful to loop prevention for the final path.

Step 4: If *T=NULL*, block the request and go back to Step 2; otherwise, *m* (0<*m*≤|*S*|) paths have been found from the source to SAVA nodes. Arrange the *m* elements of the set *T* in ascending order according to the cost of path $P(s,S_k)$ (k =1,2,…,*m*). Let *k*=1.

Step 5: If *k*>*m*, block the request and go back to Step 2; otherwise, choose the SAVA node S_k from the set *T*. Modify the cost of link in *G* according to (3), and compute an available path by Dijkstra's algorithm from the node S_k to the destination *d*. If a path has been found, record the path as $P(S_k, d)$, go to Step 6; otherwise, let *k*=*k*+1, and go back to Step 5.

$$c'_{ij} = \begin{cases} +\infty, \text{if } i, j \in P(S, S_k) \text{ and } i, j \neq S_k \\ c_{ij} + \alpha \cdot b_{ij}, \text{if } i, j \in S \text{ and } i, j \neq S_k \\ c_{ij}, \text{others} \end{cases} \tag{3}$$

Being similar to (2), different *α* means that (3) is favorable to choose a path with small number of hop or a path involving SAVA nodes with light load.

Step 6: An available path consisting of two segments $P(s, S_k)$ and $P(S_k, d)$ has been found, and packets will be routed across the chosen path. Update the load of each SAVA node in the chosen path *P*(*s*, *d*) and go back to Step 2.

The time complexity of MHFS is mainly determined by the time complexity of Dijkstra's algorithm and the procedures of adjusting link cost and comparison operations. The time complexity of Dijkstra's algorithm is $O(|N|^2)$, and the time complexity to adjust the link cost is $O(|L|)$ in Step 3 and Step 5, respectively. The time complexity of comparison operation in Step 4 is *O*(*m*-1), where *m*≤|*S*|. At most, the Dijkstra's algorithm has to run |*S*| times to look for an available path. Therefore, the time complexity of MHFS is $O(2|S||N|^2+2|L|)$.

IV. Simulation Results and Analysis

In this section, we will evaluate the performance of our proposal (MHFS) via extensive simulations under two irregular network topologies shown in Fig. 3. The first network is the NSFNet T1 backbone network with 14 nodes and 21 links. The second network is the Pan-European reference network with 28 nodes and 40 links [13]. In Fig. 3, the gray nodes denote the SAVA nodes. We compare our algorithm with the two routing algorithms SSPA and SSPALB in [8] with the extension of loop prevention. An incremental traffic model specified in [8] is used in our simulations, in which all route requests are not known ahead of time. Each time there is only one request, and all requests are uniformly distributed among all node pairs. Once a route between the source and the destination is computed, all packets for the source-destination pair will be routed across the chosen path. If the algorithm could not provision an available path, the request is rejected immediately without waiting queue. In our simulations, the basic cost for each link is assumed to 10, and the total number of route requests is generated up to 10^5.

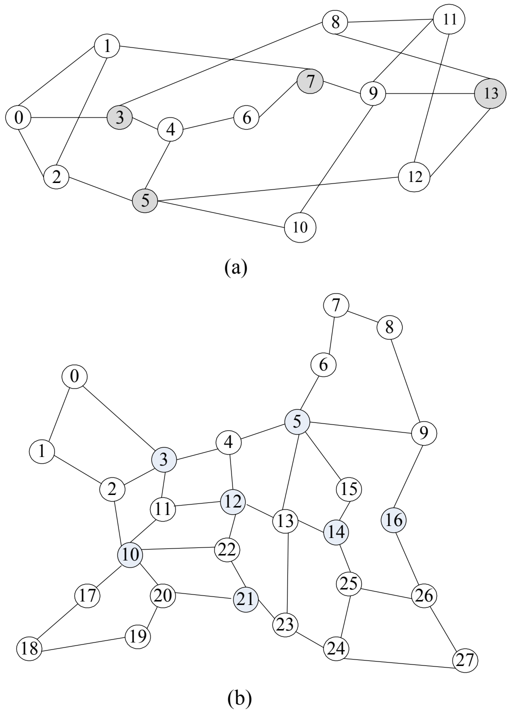


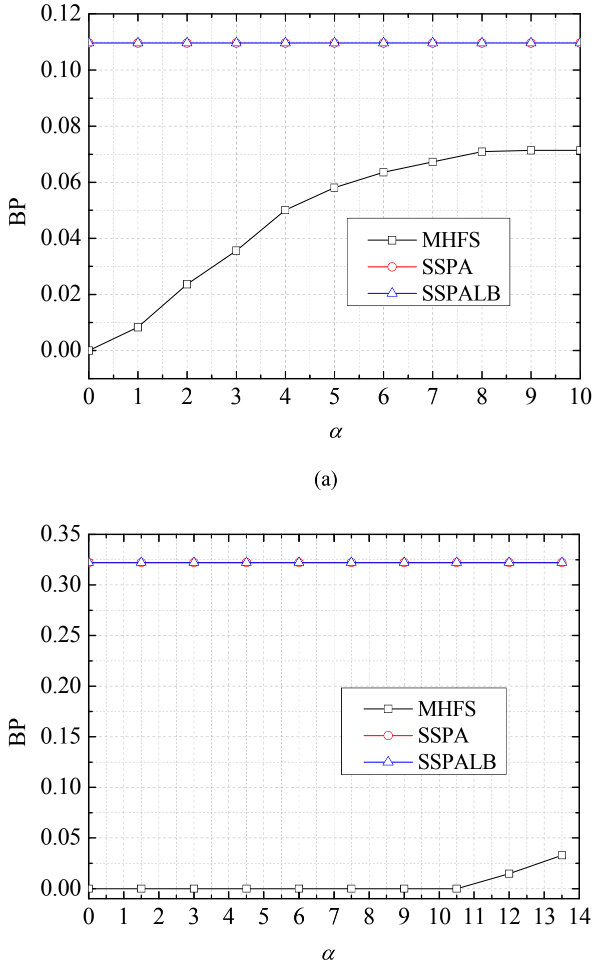
Fig. 3 Simulation network topology. (a) NSFNet T1 backbone network; (b) Pan-European reference network [13].

We employ four metrics to evaluate the network performance, which are Blocking Probability (BP), Hop to SAVA Node (HSN), Average Cost (AC), and Load Balance Degree (LBD), respectively. BP represents the percentage of blocked requests over all arriving requests during the entire simulation period. Smaller BP means that more packets can be routed to the destination and the algorithm is useful to improve the resource utilization. HSN denotes the average hop number from the source to the first SAVA node in the chosen paths for all accepted requests. Smaller HSN means that the algorithm is beneficial to choose a path with smaller hop to reach the first SAVA node for packets to validate source addresses. Therefore, it is helpful to reduce resource being wasted for the transmission of packets with spoof addresses. AC represents the average route hop from the source to the destination for all accepted requests. Smaller AC means that the algorithm has a lower cost for routing packets. LBD is defined by (4) as follows.

$$LBD = \frac{|S|\max\{l_i\}}{\sum_i l_i} - 1, i \in S \tag{4}$$

It is obvious that $LBD \geq 0$. If the value of LBD is closer to zero, the algorithm is more favorable for load balance of SAVA nodes. $LBD = 0$ represents the ideal state in which all traffic is evenly spread over all the SAVA nodes.

Fig. 4 compares the blocking probabilities of SSPA, SSPALB and MHFS with different α. The adjustable parameter α is only used in MHFS to reflect the focus on choosing a path with less hop or a path including SAVA nodes with light load, which is not involved in SSPA and SSPALB in all simulations. Therefore, the performances of SSPA and SSPALB keep no change for different α[2]. Since SSPA and SSPALB have the same key idea to look for a path, i.e., first to compute a segment from the source to each SAVA node, then to compute a segment from each SAVA node with an available segment from the source to itself to the destination, and finally to choose the path with the least hop or to choose the path involving SAVA nodes with the least load from all the computed paths between the source and the destination, they both have the same blocking probability performance as shown in Fig. 4. Another observation from Fig. 4 is that the blocking probability of MHFS has a slow increase with an increase of α. The reason for this is that, with an increase of α, MHFS is more favorable for selecting the routes involving SAVA nodes with light load. Potentially, a path with a bigger hop from the source to the first SAVA node may be chosen more often by the Step 3 of MHFS, and more links and nodes included in the first segment from the source to the first SAVA node will be removed by the Step 5 of MHFS to compute the second segment from the current SAVA node to the destination with the consideration of loop prevention. More links and nodes deleted from the topology means a less chance to compute the second segment from the SAVA node to the destination successfully. However, whatever α changes a significant improvement in BP can be achieved by our proposal. The reason for this is that SSPA and SSPALB only allow routing packets along a path including one SAVA node, which obviously reduce the chance to find a path. It is straightforward that MHFS can increase the probability of success for routing packets and can improve the resource utilization.



(a)

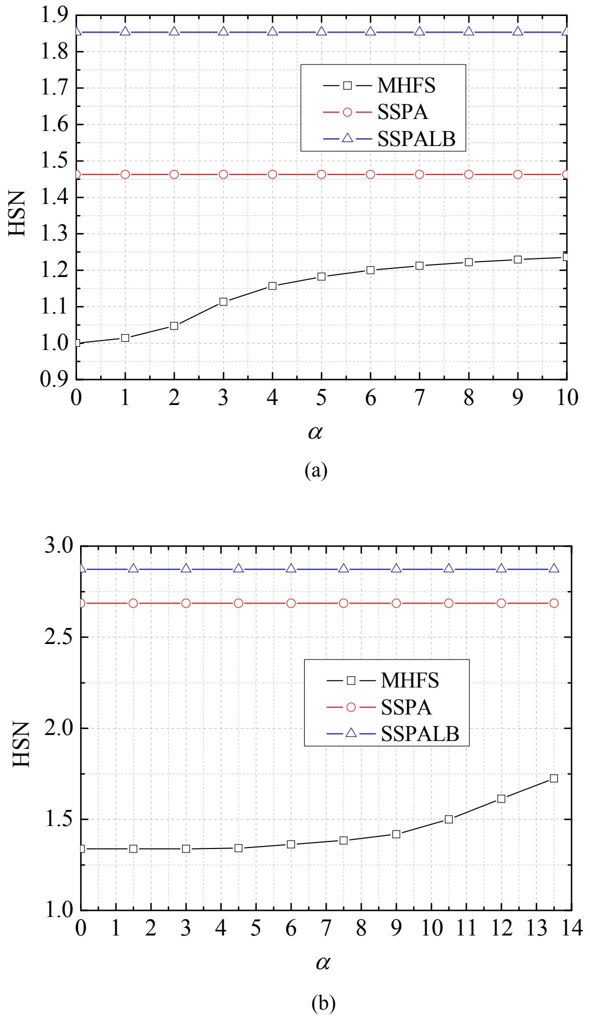
(b)

Fig. 4 Performance of blocking probability (BP). (a) NSFNet T1 backbone network; (b) Pan-European reference network.

Fig. 5 shows the performances of HSN for SSPA, SSPALB and MHFS with different α. We can observe that MHFS performance best, followed by SSPA and SSPALB in sequence. The reason for this is that MHFS routes packets along a path with the least hop to the first SAVA node, while in SSPA and SSPALB packets are routed across a path with the least hop from the source to the destination or a path involving SAVA node with the lightest load, respectively, so that SSPA and SSPALB have worse performances of HSN. Since SSPA focuses on choosing path with the least hop, which is potentially favorable to choose a path with less hop to the first SAVA node, so that it has a better performance over SSPALB. It is notable that HSN of MHFS is increased with an increase of

[2] Similarly, the performances of HSN, AC, and LBD for SSPA and SSPALB will keep no change for different α.

α. The reason for this is that a lower α means that MHFS is beneficial to choose a path with small hop, and it can potentially increase the chance to choose a path with less hop to the first SAVA node.

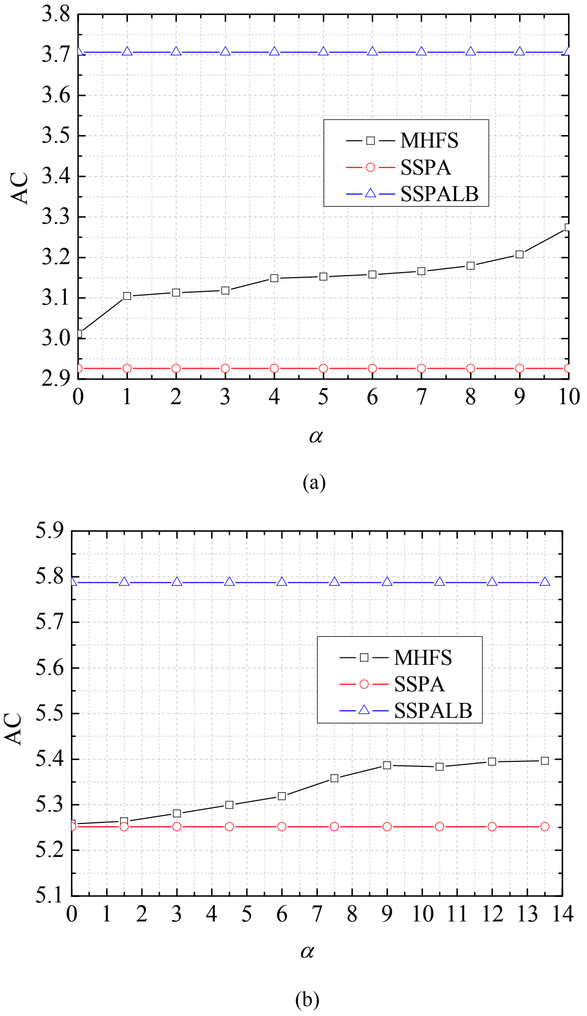


(a)

(b)

Fig. 5 Performance of hop to SAVA node (HSN). (a) NSFNet T1 backbone network; (b) Pan-European reference network.

Fig. 6 shows the performances of SSPA, SSPALB and MHFS in terms of average cost. We can observe that SSPA has the lowest AC, and MHFS has a better AC over SSPALB. The reason for this is that, with SSPA, packets are routed across the path with the least hop, so it has the lowest AC compared with the other algorithms. Due to SSPALB focusing on the load balance for SAVA nodes, it may choose a path with more hops for packets and may result in a higher average cost. Since MHFS uses α to make a tradeoff for path choice between the load balance for SAVA nodes and the shortest hop, it has an AC performance between SSPA and SSPALB. With a decrease of α, MHFS is more favorable to choose a path with smaller hop, and its AC performance is more and more close to SSPA.



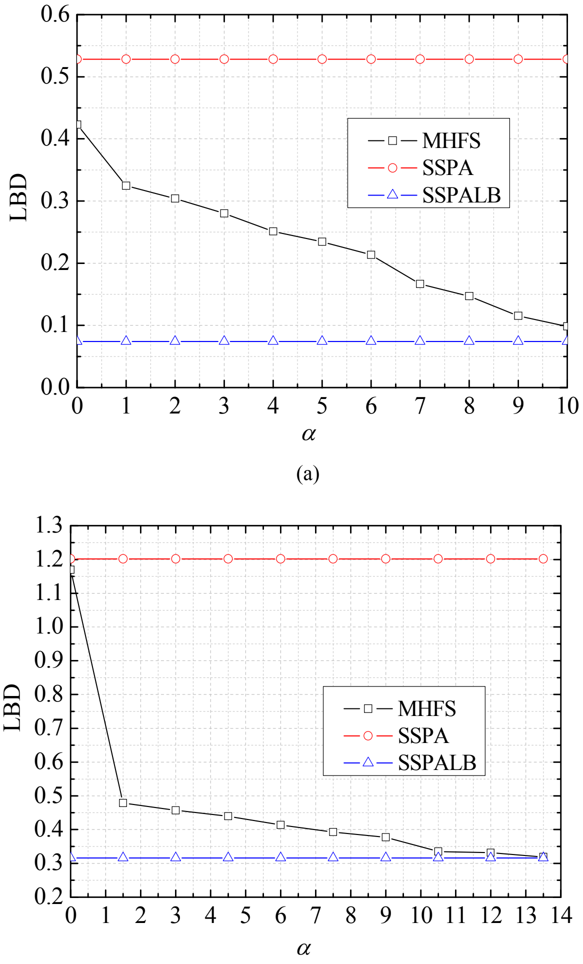
(a)

(b)

Fig. 6 Performance of average cost (AC). (a) NSFNet T1 backbone network; (b) Pan-European reference network.

Fig. 7 shows the load balance degrees of SSPA, SSPALB and MHFS. The results indicate that SSPALB has a best performance, followed by MHFS and SSPA in sequence. The reason for this is that SSPALB mainly focuses on load balance of SAVA nodes, which always choose the path with the lightest load to route packets. Compared with SSPA, MHFS can make a tradeoff between load balance and small hop by an adjustable parameter α, and a significant improvement in LBD can be achieved by MHFS. With an increase of α, the performance of MHFS is more and more close to that of SSPALB.

In order to evaluate the influence of different number of SAVA nodes in the test networks to the performances of MHFS, SSPA and SSPALB, extensive simulations with different number of SAVA nodes are carried out. We observed the similar results that significant improvements can be achieved by MHFS for the performances of BP and HSN, and the performance gaps are gradually enlarged with an increase of the number of SAVA nodes. Whether under a case of small number of SAVA nodes or a case of large number of SAVA nodes, the performances of AC and LBD for MHFS are between those of SSPA and SSPALB, and with an increase of α, there are a gradual increase for AC and a small reduce for LBD of MHFS.



(a)

(b)

Fig. 7 Performance of load balance degree (LBD). (a) NSFNet T1 backbone network; (b) Pan-European reference network.

V. Conclusions

This paper investigates how to route packets with source IP address validation in an autonomous trustworthy networks with a partial deployment of SAVA nodes. We first describe and analyze different routing policies for the trustworthy network. On the basis of the analyses, we propose a loop-free routing algorithm, called Minimum Hop to First SAVA node algorithm (MHFS). MHFS uses an adjustable parameter to reflect different focus on resource utilization and load balance. Under the incremental traffic model, extensive simulations are performed to evaluate our proposal. Simulation results show that MHFS can not only efficiently guarantee that each packet route includes at least one SAVA node, but also achieve a lower blocking probability for routing packets while considering the load balance and resource utilization

References

[1] R. He, L. Song, X. Wang and B. Lin, "Routing in trustworthy networks with SAVA nodes," in the 2nd International Conference on Advanced Computer Control, Shenyang, China, March 27-29, 2010, pp.402-406.

[2] G J. Li, J. Mirkovic, M. Wang, P. Reiher, L. Zhang, "SAVE: source address validity enforcement protocol," in Proc. of IEEE INFOCOM, New York, USA, June 23-27 2002, pp. 1557–1566.

[3] J. Bi, G. Yao and J. Wu, "An IPv6 source address validation testbed and prototype implementation," *J. of Netw.*, vol. 4, no.2, April 2009, pp. 100-107.

[4] J. Wu, G. Ren, and X. Li, "Building a next generation Internet with source address validation architecture," *Science in China Series F: Information Sciences*, vol. 51, no. 11, Nov. 2008, pp. 1681-1691.

[5] J. Bi, J. Wu, and M. Zhang, "Enable a trustworthy network by source address spoofing prevention routers: a formal description", *Emerging Directions in Embedded and Ubiquitous Computing, Lecture Notes in Computer Science*, vol. 4097/2006, 2006, pp. 681-691.

[6] J. Wu, G. Ren, and X. Li, "Source address validation: architecture and protocol design," in Proc. of IEEE ICNP, Beijing, China, Oct. 2007, pp. 276-283.

[7] S. Beverley and S. Bauer, "The spoofer project: inferring the extent of source address filtering on the Internet," in Proc. of USENIX SRUTI, Cambridge, MA, July 2005, pp. 53-59.

[8] X. Wang, L. Guo, T. Yang, W. Ji, Y. Li, Xin Liu and Y. Zhang, "New routing algorithms in trustworthy Internet," *Computer Communi.*, vol. 31, no. 14, Sept. 2008, pp. 3533-3536.

[9] A. Keromytis, V. Misra, D. Rubenstein, "SOS: Secure overlay services," in Proc. of ACM SIGCOMM, Pittsburgh, PA, USA, Aug. 19-23, 2002, pp. 61-72.

[10] S. Savage, D. Wetherall, A. Karlin, T. Anderson, "Practical network support for IP traceback," in Proc. of ACM SIGCOMM, Stockholm, Sweden, Aug. 28- Sept. 1, 2000, pp. 295-306.

[11] J. Wu, R. Bonica, J. Bi, X. Li, G. Ren, M. Williams, "Source address validation architecture (SAVA) problem statement," Internet Draft, draft-sava- problem - statement-01.txt, (2007). [Online]. Available: http://tools.ietf.org /html/draft-wu-sava-problem-statement-01.

[12] J. Wu, J. Bi, G. Ren, X. Li, M. Williams, R. Bonica, "Source address validation architecture (SAVA) framework," Internet Draft, draft-wu-sava- framework-01, (2007). [Online]. Available: http://tools.ietf.org/html/draft-wu-sava-framework-01.

[13] M. Yannuzzi, X. Masip-Bruin, G. Fabrego, S. Sanchez-Lopez, A. Sprintson, A. Orda, "Toward a new route control model for multidomain optical networks", *IEEE Communi. Mag.*, vol. 46, no. 6, June 2008, pp. 104-111.

Rongxi He (M'08) received his B.S. and M.S. degrees from Dalian Maritime University in 1992 and 1995, and his Ph. D. degree from University of Electronic Science and Technology of China in 2002, all in Communication and Information System. From 2002 to 2004, he was a Postdoctoral Fellow in Network and Communication Center of Northeastern University. From 2006 to 2007, he was a Visiting Scholar in Broadband Communications Research Group of University of Waterloo, Canada. He is currently a Professor of Dalian Maritime University. His research interests include wireless networks, optical networks and IP quality of service.

Limin Song received his B.S. from Tianjin University in 1982, his M.S. degree from Harbin Institute technology in 1985, and his Ph. D. from Dalian Institute of Technology in 2002, all in Electronic Engineering. He is currently an Associate Professor in Dalian Maritime University. His research interests include computer networks, micro-sensor technology and optical design.

Bin Lin (M'09) received his B.S. M.S. and Ph. D. from Dalian maritime University in 1993, 1996 and 2006, respectively, all in Communication and Information System. He is currently an Associate Professor in Dalian Maritime University. His research interests include wireless communication and networks.