International Journal of Information Security (2023) 22:723–736
https://doi.org/10.1007/s10207-022-00659-7

REGULAR CONTRIBUTION

A novel frame switching model based on virtual MAC in SDN

Guangjia Song[1] · Jianhua Hu[1] · Hui Wang[2]

Accepted: 20 December 2022 / Published online: 8 March 2023

Abstract
MAC address spoofing has plagued LAN communication for a long time. Many attacks use it as a springboard to carry out subsequent attacks. The main reason for this kind of attack is the exposure of MAC address. If the source MAC address of the node can be hidden during frame forwarding, this kind of attack can be effectively prevented. This study proposes virtual MAC switching (VMS) as a solution to this problem. VMS uses multi-address hopping technology to make the MAC address of the frame change continuously in the forwarding process. Its unique address generation format makes other nodes unable to record or speculate the real MAC address of the node, so it cannot launch an attack. Experiments show that VMS is close to typical SDN switches in terms of delay, throughput, and overhead and has a higher security level.

Keywords Frame switch · Ethernet · SDN · Virtual address

1 Introduction

The network architecture is hierarchical, such as the classic TCP/IP architecture and open system interconnection architecture. The advantages of hierarchical architecture are evident, such as independence, maintenance, and easy implementation. After layering, each layer adopts an independent address and resolution scheme to maintain independence. To illustrate, the link layer adopts a MAC address, the network layer adopts an IP address, and the transport layer adds a port [1, 2]. However, these addresses are not completely independent from the whole communication process. Network devices such as hosts and switches must master the correlation between addresses before they can complete data transmission. For example, the corresponding relationship between the IP address and the MAC address is necessary for the host to transmit frames in layer-2. To maintain these correspondences, many protocols are proposed, such as address resolution protocol (ARP), neighbor discovery protocol, and dynamic host configuration protocol [3, 4]. Using these protocols, network nodes can automatically find and maintain the corresponding relationship between various addresses.

MAC spoofing seriously harms LAN communication because of many loopholes in the data link layer protocol and the forwarding mechanism. Malicious hosts can use forged frames to cause ARP caching, cache polling, or launch denial of service, and man-in-the-middle attacks. Therefore, MAC spoofing needs to be solved urgently. Despite some representative works, such as Source Address Validation Improvement, SEcure Neighbor Discovery (SEND), and Ether agent, many deficiencies remain. For example, additional servers are required [5–7], the whole network traffic needs to be monitored [8, 9], the binding anchor parameters are complex [10–14], the calculation cost is high [15], and the implementation is difficult [16, 17].

To solve MAC spoofing attack, this paper proposes virtual MAC switching (VMS) as a solution. Its advantages are as follows: VMS does not need a complex binding mechanism and filters all forged packet in layer-2, so the filtering effect is better. VMS does not require additional servers, which avoids single point of failure. VMS does not need to modify the host protocol stack, which is easy to deploy. Moreover, VMS makes the host unaware of the real MAC of other hosts, which prevents frame forgery from the link layer.

The rest of the paper is organized as follows. The second part introduces the research work related to frame forwarding. The third part describes the forwarding principle and workflow of the new method, and the fourth part presents the contrastive experiment. The fifth part summarizes the article.

✉ Guangjia Song
tysong@aliyun.com

1 School of Engineering and Technology, Jiyang College of Zhejiang A&F University, Zhuji 311800, China

2 National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing 100029, China

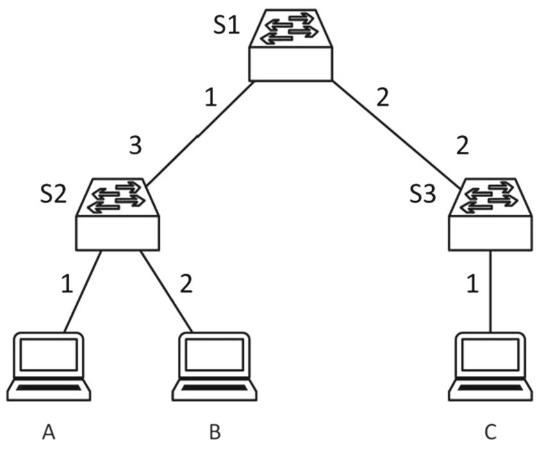

Fig. 1 Network topology

Table 1 Host configuration and connection information

Host	MAC	Connected devices	Port
A	MACa	S2	1
B	MACb	S2	2
C	MACc	S3	1

2 Research background

2.1 MAC spoofing

Ethernet forwarding is based on the destination MAC of the frame. Every time a frame is received, the switch extracts the destination MAC of the frame and then looks up its own MAC forwarding table. If there is an entry matching the destination MAC in the table, it is forwarded according to the port field of the entry. If not, the frame is broadcast. To maintain its MAC forwarding table, the switch carries out MAC learning. The method is that whenever a frame enters, the switch records the access port and the source MAC of the frame. If the source MAC exists in the MAC forwarding table, the corresponding entry is updated. If no entry matches the source MAC, a new entry is added to record the source MAC and the enter port. If an entry has not been matched for a long time, the entry is removed from the table.

The forwarding mechanism of the switch is vulnerable to many attacks, and a typical one is MAC spoofing attack. Assuming that the network topology is shown in Fig. 1, the configuration and connection information of each host is shown in Table 1.

When the network is in running normally, the MAC tables of each switch are shown in Tables 2, 3 and 4.

Suppose that host B knows the MAC address of host a, and now B wants to attack A with MAC address spoofing. The attack method of B is as follows:

Table 2 MAC forwarding table of S1

MAC	Port	TTL
MACa	1	default
MACb	1	default
MACc	2	default

Table 3 MAC forwarding table of S2

MAC	Port	TTL
MACa	1	default
MACb	2	default
MACc	3	default

Table 4 MAC forwarding table of S3

MAC	Port	TTL
MACa	2	default
MACb	2	default
MACc	1	default

Table 5 Updated MAC address forwarding table of S2

MAC	Port	TTL
MACa	2	300
MACb	2	300
MACc	3	300

Step 1 B forges a frame. The destination MAC of the frame is MACc, and the source MAC is MACa. B sends it out.
Step 2 The frame reaches S2, S2 learns the MAC address. After learning, the MAC forward table of S2 is as shown in Table 5.
Step 3 As the destination MAC of the frame is MACc, the frame goes through S1 and S3. Similarly, S1 and S3 learn the MAC address. However, the updated MAC forwarding table is the same as the original, which is not given here.
Step 4 If C wants to send a frame to A, the destination of the frame should be MACa. When the frame enters switch S3, S3 forwards it to switch S1 through port 2. After S1 queries its own MAC forwarding table, the frame is forwarded to switch S2 through port 1. After S2 looks up its own forwarding table, it sends the frame to host B through port 2. In this way, B successfully receives the frame that should be sent to A, thereby forming MAC spoofing.
MAC spoofing attack can also attach host, such as ARP cache pollution, Man-in-the-Middle, NDP cache pollution, and so

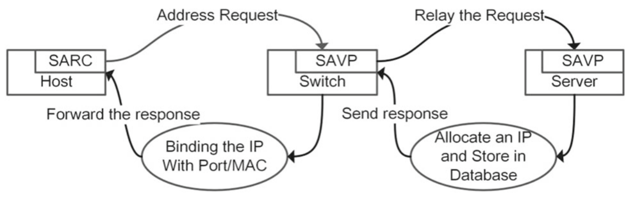

Fig. 2 Deployment of SAVA in access layer

on, all of which cause great trouble to LAN communication [18, 19].

2.2 Related research

In actual deployment, IP and MAC binding is a conventional means to prevent MAC spoofing, but this method needs manual maintenance, so it is unsuitable for dynamic and large-scale networks. Using VLAN for layer-2 segmentation and reducing the influence range of broadcasting can limit the harm of MAC spoofing, but they change the topology of the network.

Checking the authenticity of the source address is an effective method. References [5] and [20] use the discrete event system (DES) in the intrusion detection system to monitor all ARP messages in the LAN. If ARP is detected, it uses the active probe message to analyze the source sending request and then judge whether there is cheating according to the detection result. If there is deception, the DES receives more than one response, causing an event sequence different from the normal. This method requires a trusted host in the network to run the DES, and this host should have the ability to monitor all the traffic, which has the single point of failure. The literature [6] is the same as the literature [7] in that it does not trust the received ARP message directly, especially the broadcast ARP message but uses ICMP message for reverse test. The ICMP echo request (Ping) is used to test whether the source address is credible and then decide whether to trust according to the test results.

Node behavior checking is difficult to carry out in IPv6. In IPv6, a node can have multiple IP addresses at the same time, such as link local address and multiple routing addresses. That is, IP and MAC no longer have one-to-one correspondence. Document [21] proves that the corresponding relationship between IP and MAC for a host is undecidable.

One of the effective ways to find abnormal node behavior is source address validation, that is, to filter the packet according to the source address. On the one hand, it can directly prevent the attack from the source; on the other hand, it is convenient for the tracking, network diagnosis, and management [10, 11]. An example is shown in Fig. 2. SAVA can be deployed at the access layer, at the entrance of autonomous system (AS), or between different ASs. When SAVA is deployed in the access layer, it can effectively prevent MAC spoofing in the LAN.

SAVA's deployment schemes include FCFS-SAVI for IPv6 network, DHCP-SAVI for DHCP, and SEND-SAVI for SEcure Neighbor Discovery (SEND). SAVI does not check the MAC address by default. Therefore, if the IP address is true but the MAC address is false, SAVI cannot detect it. The literature [12] proposes to use MAC as additional binding information to improve FCFS-SAVI. At present, SAVI is still in the experimental stage, and it is difficult to deploy [13, 14]. The literature [22] adds a state mechanism to the host's cache, which ensures that even if the host receives an ARP reply, if it has not initiated an ARP resolution for the address before, the cache will not be updated. The literature [23] believes that when multiple IP addresses in ARP cache correspond to the same MAC, there must be deception. Cache should be clear manually, but this method is more passive and inefficient.

SEND is an enhanced version of NDP, and cryptographically generated address (CGA) is the main feature of SEND [24]. CGA is a unique address format with complicated calculation method and high complexity [15]. Document [16] proposes other efficient encryption schemes. SEND remains very difficult to deploy and implement, and most manufacturers are unable to support it [17].

As a new generation of network technology, SDN (Software Defined Network) makes up for the many shortcomings of traditional networks. It brings great improvements in rapid deployment, resource allocation, and task arrangement, among others. However, SDN also inherits many weaknesses of traditional networks, and the security problems cannot be ignored. It even introduces new security problems, such as MAC spoofing, ARP cache pollution, and so on [25, 26].

At present, very few studies are on the working mode of switches in SDN, and only a handful are on how specific protocols work better in SDN to improve security or reduce network load. For example, to expand the scale of ethernet and enable a single LAN to support more nodes, Seattle adopts one hop DHT (Distributed Hash Table) technology to store STP (Spanning Tree Protocol), ARP, and other protocol information on each node of the network discretely so as to limit the broadcast load of the whole network. Dbridge improves Seattle and supports incremental deployment [27, 28].

ABC architecture uses the learning ability of the controller to convert the multiple protocol's message that should be broadcast into unicast so as to reduce the network traffic [29]. Similar methods are adopted in the literature [30, 31]. The literature [8] proposes that a large-scale LAN should be divided into several small areas. Each area is equipped with an agent, which is responsible for filtering, limiting, and responding to ARP messages so as to reduce the number of flooding and limit the disturbance of flooding to the network.

The literature [9] adopts a more complex way to reduce network traffic. It sets up a variety of servers on the control plane. For a variety of messages that should be broadcast, such as ARP request and DHCP discovery, the server can process them to limit the broadcast. Portland adopts a MAC mapping method, which designs a virtual address format called PMAC [32]. PMAC consists of four parameters: pod, position, port, and VMID. The purpose of PMAC is to enable the switch to learn the MAC address through PMAC so as to save the storage space of the switch, facilitate the migration of virtual machines, and improve the expansion ability of the data center.

In FDSM, the controller is responsible for monitoring ARP and DHCP messages to collect the address information of the host in the LAN. When there is an address resolution request, the controller is responsible for answering [33]. The literature [34] posits that in the data center, the hosts of different tenants do not communicate. Therefore, the switch can carry out selective MAC learning, so the size of the forwarding table can be reduced greatly. However, the disadvantage is that the frame in link layer becomes larger to carry tenant information.

Small-scale resolution technology is unsuitable for large-scale data center because broadcasting limits the scale of the network. Broadcasting brings high load and huge forwarding table to the network [35, 36]. In a typical data center, the traffic generated by ARP accounts for 88% of the whole network broadcast. In a broadcast domain with 32,000 hosts, the ARP broadcast needs to occupy 100 Mbps bandwidth. Although subnet division can limit the broadcast domain, different subnets use different IP segments, which limits the mobility of virtual hosts and affects the continuity of applications. To overcome the limitations of ethernet technology, many scholars proposed overlay network as a solution, such as [37–41]. Trill has become the recommended standard of IETF; it proposes to encapsulate the ethernet frame and add a new forwarding header. Rbridge can forward these frames between different regions according to the header information. Overlay network tries to avoid using STP. Although STP protocol can avoid broadcast storm, its disadvantages are also obvious. It reduces link utilization and cannot achieve high reliability and high load.

Although SDN has changed the working mode of the network, the network presents a variety of forms and characteristics through flexible app design. However, the data plane still cannot meet the needs of the data center. The emergence of P4 technology effectively makes up for this. The P4 runtime accelerates the integration of SDN and P4 and expands the application field of SDN. To overcome the limitation of STP on ethernet expansion, ARP path adopts the method of multi-path generation [42]. It uses the broadcast process of ARP request to generate the binding port and reduces the broadcast message. It uses ARP reply to generate the shortest path, thus avoiding the use of STP, which effectively utilizes the high availability of multiple links. It also does not need any link state protocol. In addition, this method can be deployed in SDN after being combined with P4 [43, 44].

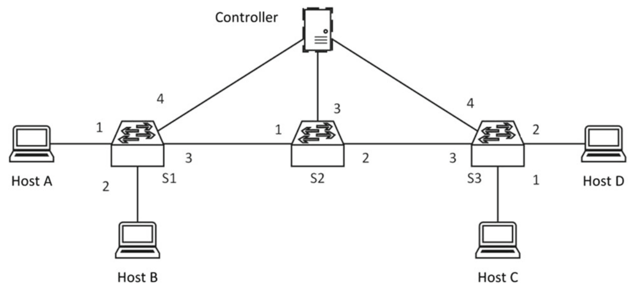

Fig. 3 Network topology

Compared with the existing research, VMS is proposed in this work as the MAC spoofing solution. Its advantages are shown in the following aspects:

(1) Although SAVI can partially filter false messages at the access layer, it requires a complex binding mechanism and the assistance of the IP layer. VMS does not require a complex binding mechanism and realizes the filtering of fake packet on layer-2, so the filtering effect is better.
(2) Compared with SEND [15, 17], SAVA[10], the literature [5, 20, 45, 46], EtherProxy [39], and other method, VMS does not need additional servers, thus avoiding single point of failure, and applies to all frame types and facilitating deployment. Moreover, VMS does not need to modify the host protocol stack, so the implementation cost is lower.
(3) Compared with existing research, VMS realizes MAC address hiding, that is, only the access switch knows the real MAC of the host, and no host knows the real MAC of other hosts. It solves MAC spoofing from the data link layer.

3 Virtual MAC switching

The main reason for the prevalence of MAC spoofing is the exposure of MAC addresses, which allows malicious hosts to take advantage of it. To hide the MAC address of the host, we design a virtual address architecture that uses the multiple jump of virtual address on the switch to hide the real address of the host, which is called VMS.

3.1 Address generation

We describe VMS with the topo shown in Fig. 3, the system is composed of switch, host, and controller. In VMS, the switch virtualizes the MAC address of the frame when forwarding. The virtual address generation method is as follows:

Every network card has a fixed and globally unique MAC address with a length of 48 bits. Here, we call it rMAC. The MAC address with special format generated by rMAC is vMAC. The generation process of vMAC is as follows:

(1) Connect the real MAC (48 bits) with the host access port number (16 bits, expand to 16 bits if less than 16 bits) to obtain a binary string with a length of 64 bits, which is recorded as X.
(2) Calculate the hash value of X. MD5, SHA-1, and other hash algorithms can be used here. The length of hash value must be longer than 128 bits.
(3) Take the last 24 bits of the hash value and add a fixed 24 bits prefix (01–80-c2) to form a new MAC address, that is, the vMAC.

3.2 New forward table

In VMS, the MAC forwarding table of the switch consists of four fields: MAC, port, time to live (TTL), and conversion table. The meanings of MAC, port, and TTL are the same as those of ordinary switches. The conversion table field is used to record the name of conversion table that corresponds to the MAC. The name of the conversion table is 64 bits long and is represented by 16 characters.

Each entry in the conversion table consists of two fields: port and vMAC. If the source MAC of a frame exists in the MAC forwarding table, when the frame needs to be forwarded from a specific port, the source MAC field of the frame should be replaced with the vMAC corresponding to the port in the conversion table. If the destination MAC address of the frame is a vMAC, the switch should traverse each conversion table in turn. If the destination MAC address of the frame is found in a conversion table, then the MAC address corresponding to the conversion table in the MAC forwarding table of the switch should be found according to the name of the conversion table, and then, the destination MAC address field of the frame is replaced with the MAC.

3.3 Controller in VMS

In VMS, the main functions of the controller include the following two aspects:

(1) Probe network topology

Whenever a new device accesses the network or the port state of the old device changes, the controller has to use link layer discovery protocol (LLDP) or open shortest path first to obtain the topology information of the network. The network topology information includes how many devices are in the network and how these devices are connected. For example, switch X is connected to port 1 of switch Y through its own port 2.

The ports of a switch can be divided into three categories: connecting controllers, connecting other network devices, or connecting hosts. For switch, if the ports connecting the controller and other network devices are excluded, the remaining ports are used to connect the host.

(2) Assign port information to the switch

The controller sends the port connection information to the corresponding switch so that the switch can know which ports are connected to the host, which ports are connected to other switches, and which ports are connected to the controller. Table 6 is an example of a typical port information table.

Table 6 Port information

Port	Connected device
1	Controller
2	Other device
3	Host

3.4 Switch in VMS

In VMS, the switch is responsible for virtual MAC transformation and frame forwarding. Its workflow is shown in Fig. 4. The specific description is as follows:

Step 1 When a frame arrives at the switch, the switch needs to extract its source MAC address and destination MAC address, which are recorded as MACx and MACy, respectively. It then records the port that the frame arrives at, which is represented by portz.
Step 2 Check the MAC addresses. The passing standard is to meet conditions (1) and (3) or conditions (2) and (3) at the same time. The conditions are as follows:

(1) MACx is a rMAC, and the frame comes from the host.
(2) MACx is a vMAC address, and the frame comes from other device.
(3) MACy is a vMAC address or a broadcast address.

If the frame cannot pass the check, it is discarded directly. If it passes the check, proceed to step 3.

Step 3 Check whether there is an entry corresponding to MACx in the MAC forwarding table. If so, update the MAC forwarding table; if not, proceed to step 4.

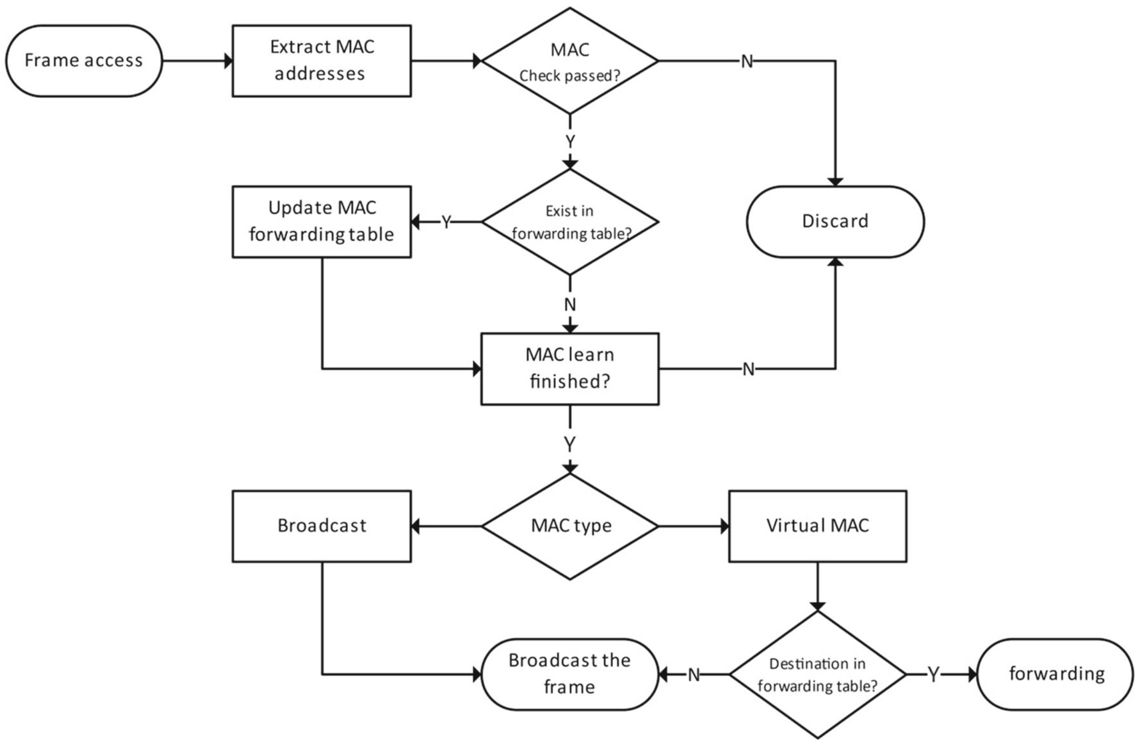

Fig. 4 Workflow of switch

Update method: Find the entry corresponding to MACx in the MAC forwarding table, set its port field to portz, and reset the TTL to the default value.

Step 4 Learn the MAC address. The specific methods are as follows:

(1) If it is not a duplicate address detection message (ACD or DAD), address learning fails. If it is, add a new entry in the MAC forwarding table. The MAC address field of the new entry is MACx, the port field is portz, and the TTL is the default value. Then, generate a random conversion table name and write the name into the conversion table field of the new entry.
(2) Generate the conversion table corresponding to MACx. The specific methods are as follows:

According to the port information table of the switch, exclude the ports connecting the controller and the entry port of the frame (i.e., portz), generate a random virtual MAC address for each remaining port, and then write it into the conversion table.

Step 5 Determine the address type of MACy. If it is a broadcast address, proceed to step 6. If it is a vMAC, proceed to step 7.
Step 6 Perform source MAC address conversion and then broadcast the frame. The specific methods are as follows:

For each port that needs to forward the frame, before forwarding, the switch needs to query the conversion table corresponding to MACx, determine the corresponding vMAC according to the port, replace the source MAC field in the frame with the vMAC, recalculate the frame check sequence (FCS) field of the frame, and then forward the replaced frame from the port.

Step 7 Traverse all conversion tables of the switch to find out whether MACy exists. If not, go to step 6. If it exists, write down the name of the conversion table where MACy is located, then find the entry to include the conversion table from the MAC forwarding table, extract the MAC address and port number of the entry, and then proceed to step 8.
Step 8 Perform source MAC conversion and destination MAC conversion and then forward the frame by unicast. The specific methods are as follows:

(1) Replace the destination MAC field of the frame with the MAC found in step 7.
(2) Use the port number found in step 7 to retrieve the conversion table corresponding to MACx, find the vMAC corresponding to the port number, replace the source MAC field of the frame with the vMAC, recalculate the FCS, and then send the frame from the port.

Table 7 Host information

Host	MAC	IP
A	00-0c-29-cf-a2-01	10.1.1.1
B	00-0c-29-cf-a2-02	10.1.1.2
C	00-0c-29-cf-a2-03	10.1.1.3
D	00-0c-29-cf-a2-04	10.1.1.4

Table 8 Port information table of S1

Port	Connected device
1	host
2	host
3	device
4	controller

Table 9 Port information table of S2

Port	Connected device
1	device
2	device
3	controller

Table 10 Port information table of S2

Port	Connected device
1	host
2	host
3	device
4	controller

DstMAC ff-ff-ff-ff-ff-ff	SrcMAC 00-0c-29-cf-a2-01	Type 0x0806	Data	FCS 0xf6d19c6

Fig. 5 Broadcast frame generated by host A

Table 11 MAC forwarding table of S1

MAC	Port	TTL	Conversion table
00-0c-29-cf-a2-01	1	default	0xbaac2cf05d864631

3.5 Example of VMS

We use an example to show the working process of VMS. The network topology is shown in Fig. 3. The connection information between each device is marked in the figure, and the IP and MAC addresses of each host are shown in Table 7. We describe how VMS handles address resolution, vMAC generation, and frame forwarding through three stages: the initialization stage, the communication stage of A and D, and the response stage of D. The table also shows how the address forwarding table and conversion table of the switch change during each stage.

3.6 Initialization stage

The controller uses LLDP to generate the port information table of each switch and send it to the corresponding switch. See Tables 8, 9 and 10 for the port information obtained by S1, S2, and S3, respectively.

3.7 Stage of A communicates with D

Now, suppose A wants to communicate with D. A knows the IP address of D but does not know the MAC address of D. First, A needs to perform address resolution or neighbor discovery to obtain the MAC of D, so A needs to send an ARP request or a NS. It is a broadcast frame, so the destination address is a broadcast address. The frame format is shown in Fig. 5.

3.8 Process of S1

When the frame reaches switch S1 through port 1, the processing process of S1 is as follows:

(1) Extract the source MAC address and destination MAC address from the frame: They are 00-0c-29-cf-a2-01 and a broadcast address.
(2) S1 checks MACa and finds that it is an rMAC address and comes from the host port. The destination MAC is a broadcast address, so it passes the check.
(3) As the MAC forwarding table of S1 is empty, it needs to learn the MAC address. Here, assuming that the name of the conversion table generated by S1 is 0xbaac2cf05d864631, the forwarding table learned is shown in Table 11.
(4) Generate conversion table "baac2cf05d864631." Generate the virtual addresses of 00-0c-29-cf-a2-01 on ports 2 and 3. See 3.1 for the method of generating the vMAC. The conversion table generated here is shown in Table 12.
(5) As the destination MAC of the frame is a broadcast address, it should be broadcast. S1 needs to convert the frame's source address and broadcast it from ports 2 to 3. The frames sent to port 2 and port 3 are shown in figs. 6 and 7, respectively.

Table 12 S1's conversion table 0xbaac2cf05d864631

Port	vMAC
2	01-80-c2-b7-b9-27
3	01-80-c2-aa-4e-80

DstMAC	SrcMAC	Type	Data	FCS
ff-ff-ff-ff-ff-ff	01-80-c2-b7-b9-27	0x0806		0x699a48d6

Fig. 6 Frame sent by switch S1 port 2

DstMAC	SrcMAC	Type	Data	FCS
ff-ff-ff-ff-ff-ff	01-80-c2-aa-4e-80	0x0806		0x578312

Fig. 7 Frame sent by switch S1 port 3

Table 13 MAC forwarding table of S2

MAC	Port	TTL	Conversion table
01-80-c2-aa-4e-80	1	default	0 × 93c72cf05d864631

3.9 Process of B

Host B receives the broadcast frame sent by port 2 of S1, as shown in Fig. 6 and then updates its address cache to acquire a new address entry, that is < 01-80-c2-b7-b9-27, 10.1.1.1 >

3.9.1 Process of S2

After port 1 of S2 receives the broadcast frame (Fig. 7) sent by port 3 of switch S1, the process is as follows:

(1) Extract the source MAC and destination MAC from the frame: They are 01-80-c2-aa-4e-80 and a broadcast address (ff-ff-ff-ff-ff-ff).
(2) As the source MAC of the frame is a vMAC and comes from a device port, and the destination MAC is a broadcast address, the MAC passes the check.
(3) There is no 01-80-c2-aa-4e-80 in the forwarding table of S2, so MAC learning is required. The name of the randomly generated conversion table here is 0 × 93c72cf05d864631. See Table 13 for the learning results.
(4) The conversion table 0 × 93c72cf05d864631 generated by S2 is shown in Table 14.
(5) As the destination MAC of the frame is a broadcast address, S2 first converts the source MAC of the frame and then forwards it from port 2. The converted frame is shown in Fig. 8.

Table 14 S2's conversion table 0 × 93c72cf05d864631

Port	vMAC
2	01-80-c2-7d-f4-da

DstMAC	SrcMAC	Type	Data	FCS
ffff-ffff-ffff	01-80-c2-7d-f4-da	0x0806		0x6d05ad2c

Fig. 8 Frame sent by switch S2 port 2

Table 15 MAC forwarding table of S3

MAC	Port	TTL	Conversion table
01-80-c2-7d-f4-da	3	default	0xb7692cf05d864631

Table 16 S2's conversion table 0xb7692cf05d864631

Port	vMAC
1	01-80-c2-b6-c8-86
2	01-80-c2-35-59-44

3.9.2 Process of S3

After the frame reaches port 3 of S3, the processing process is similar to that of S2, which is briefly described as follows:

(1) Extract the source MAC and destination MAC address: 01-80-c2-7d-f4-da and a broadcast address.
(2) As the source MAC of the frame is a vMAC, which comes from a device port, and the destination MAC is a broadcast address, the MAC passes the check.
(3) There is no 01-80-c2-7d-f4-da in the MAC forwarding table of S3, so it requires MAC learning. The generated random conversion table name is 0xb7692cf05d864631. The learning results are shown in Table 15.
(4) S3 generates the conversion table 0xb7692cf05d864631, and the results are shown in Table 16.

As the destination MAC of the frame is a broadcast address, S3 broadcasts the frame from each port (excluding controller port and entry port). The frames generated during broadcasting are shown in figs. 9 and 10.

DstMAC	SrcMAC	Type	Data	FCS
ffff-ffff-ffff	01-80-c2-b6-c8-86	0x0806		0xeb675024

Fig. 9 Frame sent by switch S3 port 1

DstMAC	SrcMAC	Type	Data	FCS
ffff-ffff-ffff	01-80-c2-35-59-44	0x0806		0xc232670b

Fig. 10 Frame sent by switch S3 port 2

DstMAC	SrcMAC	Type	Data	FCS
01-80-c2-35-59-44	00-0c-29-cf-a2-04	0x0806		0xea5cc8c4

Fig. 11 D's reply frame

3.9.3 Processes of C and D

After receiving the broadcast frame sent by port 1 of S3, host C updates its address cache and acquires a new address entry < 01-80-c2-b6-c8-86, 10.1.1.1 > . Similarly, D also updates its own cache and acquires a new address entry < 01-80-c2-35–59-44, 10.1.1.1 > .

4 Answer stage of D

4.1 D's reply

As D already knows the IP and MAC of host A, if D wants to answer A, it should first build a unicast frame, as shown in Fig. 11, and then, send it to S3 through port 2.

4.1.1 Process of S3

(1) Extract the source MAC and destination MAC: 00-0c-29-cf-a2-04 and 01-80-c2-35-59-44.
(2) After checking the MAC, it finds that the source MAC is a real address and comes from the host port, and the destination MAC is a vMAC address, so it passes the check.
(3) The MAC forwarding table of S3 is not empty, but there is no entry corresponding to 00-0c-29-cf-a2-04, so MAC learning is required. The name of the generated random conversion table for 00-0c-29-cf-a2-04 is 0 × 91162cf05d864631. The learning results are shown in Table 17. The conversion table 0 × 91162cf05d864631 is shown in Table 18.
(4) As the destination MAC is a vMAC, S3 should traverse all its conversion tables in turn to find which table 01-80-c2-35-59-44 belongs. Here, it is 0xb7692cf05d864631, that is, Table 16. The corresponding MAC in forwarding table is 01-80-c2-7d-f4-da and port 3.
(5) Perform address translation. The source MAC is converted to vmacs3d3 and the destination MAC is converted to 01-80-c2-7d-f4-da. Finally, the frame sent from port 3 of S3 is shown in Fig. 12.

Table 17 MAC forwarding table of S3

MAC	Port	TTL	Conversion table
01-80-c2-7d-f4-da	3	default	0xb7692cf05d864631
00-0c-29-cf-a2-04	2	default	0 × 91162cf05d864631

Table 18 S3's conversion table 0 × 91162cf05d864631

Port	vMAC
1	01-80-c2-8d-9f-25
3	01-80-c2-76-30-ba

DstMAC	SrcMAC	Type	Data	FCS
01-80-c2-7d-f4-da	01-80-c2-76-30-ba	0x0806		0xea5cc8c4

Fig. 12 Frame sent by S3 port 3

Table 19 The MAC forwarding table of S2

MAC	Port	TTL	Conversion table
01-80-c2-aa-4e-80	1	default	0 × 93c72cf05d864631
01-80-c2-76-30-ba	2	default	0 × 9fd52cf05d864631

4.1.2 Process of S2

When the frame reaches port 2 of S2 from port 3 of S3, the processing process of S2 is as follows:

(1) Extract the source address: 01-80-c2-76-30-ba, destination address: 01-80-c2-7d-f4-da.
(2) After inspection, it is found that the source MAC is a vMAC and comes from the device, and the destination MAC also is a vMAC, so it passes the inspection.
(3) Although the MAC address forwarding table is not empty, there is no 01-80-c2-76-30-ba, so MAC learning is carried out. The learning results are shown in Table 19. The name of the generated random conversion table is 0 × 9fd52cf05d864631. See Table 20 for details.
(4) As the destination MAC of the frame is a vMAC, S2 should traverse each conversion table in turn to find which 01-80-c2-7d-f4-da belongs. The result is the conversion table 0 × 93c72cf05d864631, that is, Table 14. The corresponding entry in the MAC forwarding table is vmacs1a3 and port 1.
(5) Do the address conversion. The source MAC is converted to 01-80-c2-ff-a5-e1, and the destination MAC is converted to 01-80-c2-aa-4e-80. Finally, the frame sent from port 3 of S2 is shown in Fig. 13.

Table 20 S2's conversion table 0 × 9fd52cf05d864631

Port	vMAC
1	01-80-c2-ff-a5-e1

DstMAC 01-80-c2-aa-4e-80	SrcMAC 01-80-c2-ff-a5-e1	Type 0x0806	Data	FCS 0xea5cc8c4

Fig. 13 Frame sent by S2 port 1

Table 21 MAC forwarding table of S1

MAC	Port	TTL	Conversion table
00:0c:29:cf:a2:01	1	default	0xbaac2cf05d864631
01-80-c2-ff-a5-e1	3	default	0 × 95452cf05d864631

4.1.3 Process of S1

(1) Extract the destination MAC and the source MAC of the frame: 01-80-c2-aa-4e-80 and 01-80-c2-ff-a5-e1.
(2) After inspection, it is found that the source MAC of the frame is a vMAC and comes from the device, and the destination MAC also is a vMAC, so it passes the inspection.
(3) As the MAC forwarding table is not empty, but there is no 01-80-c2-ff-a5-e1. It needs to learn the new MAC, and the learning results are shown in Table 21. The name of the generated random conversion table is 0 × 95452cf05d864631.
(4) Generate conversion table 0 × 95452cf05d864631. See Table 22 for details.
(5) As the destination MAC of the frame is a virtual MAC address, S1 should traverse all its conversion tables in turn and find the conversion table to which 01–80-c2-aa-4e-80 belongs. Here is the conversion table 0xbaac2cf05d864631, that is, Table 12. The corresponding entry in the MAC forwarding table is 00-0c-29-cf-a2-01 and port 1.
(6) Do the address conversion. The source MAC is converted to 01-80-c2-1b-64-de, and the destination MAC is converted to 00-0c-29-cf-a2-01. Finally, the frame sent from port 1 of S1 is shown in Fig. 14.

4.1.4 Process of A

After receiving the unicast frame sent by S1, A acquires a new address entry, < 01-80-c2-1b-64-de, 10.1.1.4 > . In this way, A obtains the MAC address of host D, and the subsequent communication between A and D can directly adopt unicast communication.

Table 22 S1's conversion table 0 × 95452cf05d864631

Port	vMAC
1	01-80-c2-1b-64-de
2	01-80-c2-e0-c7-14

DstMAC 00-0c-29-cf-a2-01	SrcMAC 01-80-c2-1b-64-de	Type 0x0806	Data	FCS 0xea5cc8c4

Fig. 14 Frame sent by S1 port 1

5 Experiment and analysis

To test the performance and effectiveness of VMS, we carry out a series of comparative tests. The experimental platform is Linux Ubuntu 18.04, AMD 3970 × CPU, 32 GB Corsair DDR4 memory; network topology software is Mininet; the programming language is Python 3.80; switch software is Open VSwitch (OVS); and the controller is Ryu. The network topology is shown in Fig. 3. The experiment includes several scenarios. According to RFC2544, the forwarding delay, throughput, and overhead are measured and analyzed.

5.1 Delay

The experiment consists of two scenarios. Scenario 1: Host A sends a 64-byte frame to host B via switch S1. The length of the HTB queue is 100. Scenario 2: Host A sends a frame to host B via switch S1. The frame sending rate is the port bandwidth/frame size, and the HTB queue length is 1000. In each scenario, 20 tests are conducted for OVS, native SDN switch, and VMS. The measurement metric is forwarding delay, that is, the time difference between the frame sent by host A and the frame completely received by host B. The experimental results are shown in Fig. 15.

As shown in Fig. 15, for switches with different bandwidths, the delay of 1000 Mbps switch is less than that of 100 Mbps switch. In the forwarding process, the SDN needs the cooperation of the controller and the switch, so there the configuration and change of the flow table occur during the process. At the same time, the controller needs to maintain the global MAC table, so the forwarding delay is high. OVS relies on its own software to simulate the MAC learning and frame forwarding of ethernet switch, so the frame forwarding delay is low, but the degree of freedom is limited. VMS needs to maintain additional conversion tables during forwarding, so the delay is slightly higher than OVS but lower than SDN.

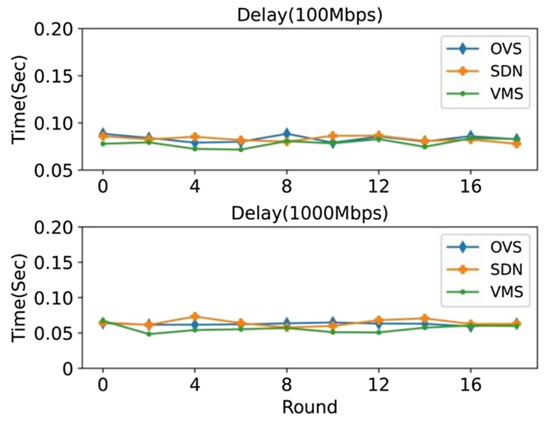

Fig. 15 Forwarding delay

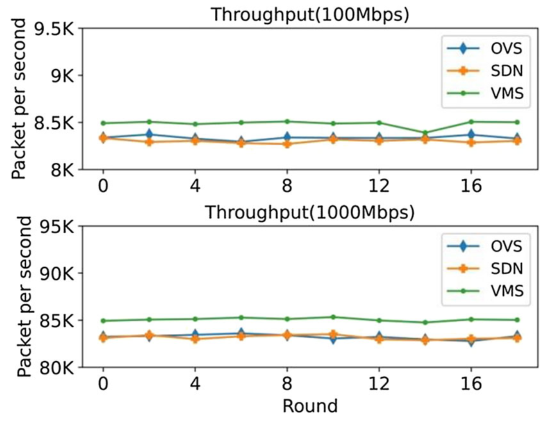

Fig. 16 Throughput

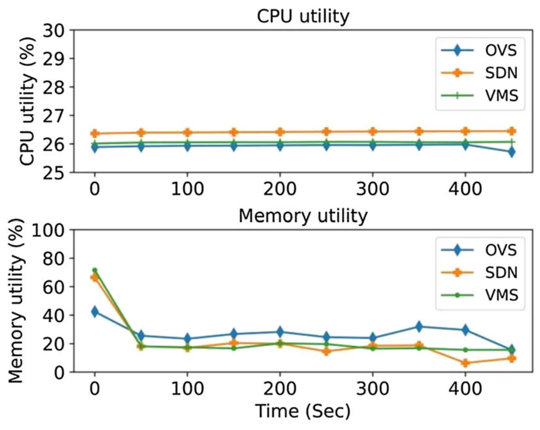

Fig. 17 CPU and memory overhead comparison (100Mbps)

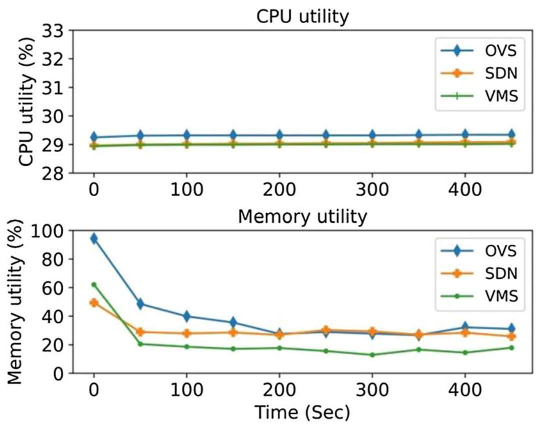

Fig. 18 CPU and memory overhead comparison (1000Mbps)

5.2 Throughput

The experiment consists of Scenarios 3 and 4. Scenario 3: Host A sends a 64-byte frame to host B via switch S1. The bandwidth is 100 Mbps, and the HTB queue length is 1000. Scenario 4 is similar to Scenario 3, but the bandwidth is 1000 Mbps. In each scenario, 20 experiments are conducted for OVS, native SDN switch, and VMS, and the measurement metric is throughput. The experimental results are shown in Fig. 16.

When the bandwidth is fixed, the throughput of SDN is the lowest. In SDN, the controller and switch need to interact periodically and deal with flow table configuration and flow missing, which affect the forwarding efficiency. Therefore, SDN throughput is always lower than OVS and VMS. The throughput of VMS is slightly higher than that of OVS, as shown in Fig. 16.

5.3 CPU and memory utilization

In scenarios 3 and 4, the CPU and memory overhead are measured. As shown in figs. 17 and 18, when performing saturation forwarding, the CPU overhead of the three is less than 30%, and the overhead of SDN is slightly higher. Overall, the memory and CPU overhead of 1000 Mbps switching mode are higher than that of 100 Mbps switching mode. The CPU overhead of SDN is large, whereas VMS is close to OVS, and VMS is slightly higher than OVS. In terms of memory usage, there is a large memory occupation in the early stage of the experiment, which can be seen in figs. 17 and 18, especially in Fig. 18. The memory utilization rate of OVS is close to 100% at one time, but it gradually declines and tends to be stable in the later stage. OVS needs to maintain a database (OVSDB), so the memory consumption is the highest, whereas those in SDN and VMAC are slightly lower. Although VMS needs more memory space to maintain additional data structures, SDN's flow table and global MAC forwarding table bring the same overhead.

5.4 Security analysis

Spoofing attacks are very difficult in VMS. The main reasons are as follows:

(1) If a malicious node attempts to send a forged frame to deceive the switch or other nodes, it is not feasible in VMS. Take host C trying to forge the MAC address of A as an example. C does not know the rMAC of host A, so if C uses the MAC address of A that it learned by itself to attack, the frame is considered as a forged frame by the switch and directly discarded. This is because the address that C learned is vMAC, and the vMAC generated by VMS adopts a special link layer prefix 01–80-c2, which belongs to reserved MAC address set. All manufacturers and virtual machine software do not obtain the address with this prefix. Therefore, in VMS, the access port of the host does not accept frames with vMAC as the source address. Similarly, the port connecting to the host does not accept frames with vMAC as the source address.

(2) If C tries to guess the real address of A and attempts to produce a collision, it is also very difficult for the following two reasons:

The MAC address is 48 bits, so it is a huge address space and it is very difficult to search. Assuming that the network bandwidth is 1 Gbps, taking the minimum frame of 64 bytes as an example, the frame that can be generated by the attacking node in one second does not exceed 4 K. Even if each frame adopts a random source MAC address, the probability of conflict between this address and A's MAC is

$$p = 1 - \prod_{i=1}^{\frac{2^{30}}{64\times 2^{3}}} \left(1 - \frac{i}{2^{48}}\right)$$

$$p = 1 - e^{-\frac{(2^{22}\times 2^{22}-1)}{2^{49}}}$$

$$p \approx 1 - e^{-\frac{1}{2^{26}}}$$

The probability of conflict is a minimum. Moreover, for modern switch, to prevent broadcast storm, port broadcast restriction strategy is generally enabled to limit the number of broadcast frames generated in one second of a single port, so the above attack is more difficult to achieve.

(3) Even if C knows the rMAC of A, it cannot cause deception by using the MAC as the source address when sealing frames. The switch ports connected by C and A are different, so the vMAC mapping generated on each port is also different. Even if C and A send exactly the same frame, the conversion table generated on the switch is completely different, so the addresses learned by other hosts are also different. Therefore, C cannot receive the frame sent by other hosts to A.

6 Conclusion

The exposure of source MAC address brings huge security risks to LAN communication. MAC spoofing is the premise of many attacks. VMS uses a unique address generation method to generate virtual addresses so as to hide the real address of the node. This makes MAC spoofing difficult. At the same time, VMS adopts multi-address hopping technology so that the MAC addresses learned by different nodes are different. In addition, the MAC caches of different hosts for the same IP address are not consistent and irrelevant. The experiments also show that VMS is comparable with the traditional switching mode in terms of latency and overhead, but its security is better. Even if an attacker intercepts a frame, it cannot analyze the sender and the receiver from the frame, which creates conditions for anonymous communication at the data link layer. The next research work can be carried out in three aspects: (1) Optimize forwarding algorithm to reduce forwarding overhead further. (2) Adopt the probability CRC checking for the frame to reduce the checking cost of the switch. (3) Research content security technology to achieve anonymous communication at the data link layer.

Acknowledgements This work was supported by the National Social Science Foundation of China (No.17BGL232), Basic Public Welfare Research Program of Zhejiang Province (No. LGG22F020002).

Data availability All data generated or analyzed during this study are included in this published article.

Declarations

Conflicts of interest The authors declare that there are no potential conflicts of interest.

Human or animal rights This article does not contain any studies involving human participants performed by any of the authors.

References

1. Stevens, W. R.: "TCP/IP Illustrated, Volume 2: The Implementation." Pearson Schweiz Ag (1995)
2. Kurose, J., Keith, R.: "Computer networks: A top down approach featuring the internet." (2010)

3. Plummer, D. C.: RFC826: Ethernet address resolution protocol. Computer & Communications Dictionary (1982)
4. Narten, T. et al.: Neighbor discovery for IP version 6 (IPv6). No. rfc4861. (2007)
5. Neminath, H., Biswas, S., Roopa, S., et al.: A DES Approach to intrusion detection system for arp spoofing attacks[C]. Control & automation. IEEE, 2010:695–700. (2010)
6. Pandey P.: Prevention of ARP spoofing: A Probe Packet Based Technique[C]. Advance Computing Conference. IEEE, 2013:147–153.
7. Ferguson, S.: RFC2827 Network ingress filtering: defeating denial of service attacks which employ IP source address spoofing [J]. (2000) http://tools.ietf.org/pdf/bcp38.pdf
8. Chiu, C.H., Lei, C.L.: Etheragent: scaling ethernet for enterprise and campus networks. Int. J. Innovat. Comput. Inf. Control **9**(6), 2465–2483 (2013)
9. Alasadi, E., Al-Raweshidy, H. S.: SSED: Servers Under Software-Defined Network Architectures to Eliminate Discovery Messages. In: IEEE/ACM Transactions on Networking (2018):1–14. (2018)
10. Wu, J., Bi, J. et al.: A source address validation architecture (SAVA) testbed and deployment experience[EB/OL]. (2008) http://www.ietf.org/rfc/rfc5210.txt
11. Wu J, Ren G, Li X.: Source address validation: architecture and protocol design[C]. In: The 15th IEEE International Conference on Network Protocols. 2007: 276–283. (20007)
12. García-Martínez, A., Bagnulo, M.: An integrated approach to prevent address spoofing In IPv6 Links. IEEE Communication Letters IEEE **16**(11), 19800–21902 (2012)
13. Xiao, P., Bi, J.: OpenFlow Based Intra-AS Source Address Validation. Journal of Chinese Computer Systems 34 (2013).
14. Li, J., Wu, J., Xu, K., Chen, W.L.: An hierarchical inter-domain authenticated source address validation solution. J. softw. **35**(1), 85–100 (2012)
15. Kukec, A., Krishnan, S., Jiang, S.: The Secure Neighbor Discovery (SEND) Hash Threat Analysis. RFC 6274, June, (2011)
16. Rafiee, H., Alsa'deh, A., Meinel, C.: Multicore-based auto-scaling secure neighbor discovery for windows operating systems. In: Proceedings of IEEE International Conference on Information Networking (ICOIN). (2012)
17. Qadir, S., Siddiqi, M.U.: Cryptographically generated addresses (CGAs): a survey and an analysis of performance for use in mobile environment. Int. J. Comput. Sci. Netw. Secur. **11**(2), 24–31 (2011)
18. Saito, Y., Kuroda, M., Mizuno, T.: A Virtual MAC Address Scheme for Mobile Ethernet. Wireless Pers. Commun. **35**(1), 99–109 (2005)
19. Bandar, A. et al.: A New MAC Address Spoofing Detection Technique Based on Random Forests. Sensors (Basel, Switzerland) (2016)
20. Barbhuiya, F.A., Biswas, S., Nandi, S.: An active DES based IDS for ARP Spoofing[C]. In: IEEE International Conference on Systems. IEEE, 2011:2743–2748. (2011)
21. Song, G., Ji, Z.: Research on Equivalence between Address Resolution and Duplicate Address Detection. In: Fifth International Conference on Instrumentation & Measurement IEEE. (2016)
22. Trabelsi, Z., El-Hajj, W.: Preventing ARP attacks using a fuzzy-based stateful ARP cache. In: Proceedings of IEEE International Conference on Communications. (2007)
23. Oh, M.Y.G., Hong, S., Cha, S.: ASA: agent-based secure ARP cache management. Communications Iet **6**(7), 685–693 (2012)
24. Narten, T. et al. RFC4861: Neighbor Discovery for IP version 6 (IPv6). Standards Track, http://www.ietf.org/rfc/rfc4861.txt (2007)
25. Meghana, J., Subashri, T., Vimal, R.A.: Survey on ARP Cache Poisoning and Techniques for Detection and Mitigation. In: Proceedings of the IEEE International Conference on Signal Processing, Communication and Networking (ICSCN), Chennai, India, 16–18 March 2017; pp. 1–6. (2017)
26. Alharbi, T., Durando, D., Pakzad, F., Portmann, M.: Securing ARP in software defined networks in proceedings of the IEEE conference on local computer networks (LCN), Dubai, UAE, 7–10 November 2016; pp. 523–526. (2016)
27. Kim, C, Caesar, M., Rexford, J.: "Floodless in SEATTLE: a scalable ethernet architecture for large enterprises." ACM ACM, 2008:3–14. (2008)
28. Varis, N., et al.: DBridges: Flexible floodless frame forwarding. Comput. Netw. **57**(17), 3601–3616 (2013)
29. Wang, Y.C., Han, H.U.: An adaptive broadcast and multicast traffic cutting framework to improve ethernet efficiency by SDN. J. Inf. Sci. Eng. **35**(2), 375–392 (2019)
30. Jehan, N., Haneef, A M.: "Scalable ethernet architecture using SDN by suppressing broadcast traffic." 2015 Fifth International Conference on Advances in Computing & Communications (ICACC) IEEE. (2016)
31. Xia, J., et al.: An active defense solution for ARP spoofing in openflow network. Chin. J. Electron. **28**(1), 172–178 (2019)
32. Mysore, R.N., et al. "PortLand: a scalable fault-tolerant layer 2 data center network fabric." Proceedings of the ACM SIGCOMM 2009 conference on applications, technologies, architectures, and protocols for computer communications, Barcelona, Spain, August 16–21, 2009 ACM, (2009)
33. Wang, J., et al.: A novel floodless service discovery mechanism designed for Software-Defined Networking. Communications **11**(2), 12–25 (2014)
34. Shpiner, Alexander, et al. "SAL: Scaling data centers using smart address learning." 10th International Conference on Network and Service Management (CNSM) and Workshop. IEEE, 2014.
35. Myers, A., Ng, T.E., Zhang, H.: Rethinking the service model: Scaling ethernet to a million nodes. (2004)
36. Stephens, B., Cox, A.L., Rixner, S., Ng, T.S.E.: A scalability study of enterprise network architectures. ACM/IEEE ANCS '11. (2011)
37. Benson, T., Akella, A., Shaikh, A., Sahu, S.: CloudNaaS: a cloud networking platform for enterprise applications. SOCC '11. (2011)
38. Edwards, A., Fischer, A., Lain., A.: Diverter: a new approach to networking within virtualized infrastructures. ACM WREN '09. (2009)
39. Elmeleegy, K., Cox, A.: Etherproxy: scaling ethernet by suppressing broadcast trafficc. In: IEEE INFOCOM'09. (2009)
40. Niranjan Mysore, R., Pamboris, A., Farrington, N., Huang, N., Miri, P., Radhakrishnan, S., Subramanya, V., Vahdat, A.: Portland: a scalable fault-tolerant layer 2 data center network fabric. ACM SIGCOMM '09, (2009)
41. Sridharan, M. et al.: NVGRE: Network virtualization using generic routing encapsulation. In: Network Working Group Internet Draft. (2011)
42. Ibanez, G., et al.: ARP-path: ARP-based, shortest path bridges. IEEE Commun. Lett. **15**(7), 770–772 (2011)
43. Martinez-Yelmo, I. et al.: ARP-P4: A hybrid ARP-Path/p4runtime switch. In: 2018 IEEE 26th International conference on network protocols (ICNP) IEEE, 2018
44. Martinez-Yelmo, I., et al.: ARP-P4: deep analysis of a hybrid SDN ARP-Path/P4Runtime switch. Telecommunication Systems: Modelling, Analysis, Design and Management **72**(4), 555–565 (2019)
45. Girdler, T., Vassilakis, V.: Implementing an intrusion detection and prevention system using Software-Defined Networking: Defending against ARP spoofing attacks and Blacklisted MAC Addresses. Comput. Electr. Eng. **90**, 106990 (2021)
46. Khalid, H., Ismael, P., Al-Khali, A.B.: Efficient mechanism for securing software defined network against ARP spoofing attack. J Duhok Univ (2019). https://doi.org/10.26682/sjuod.2019.22.1.14