Computer Communications 64 (2015) 1–20

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

An incrementally deployable anti-spoofing mechanism for software-defined networks

Jonghoon Kwon [a], Dongwon Seo [a], Minjin Kwon [a], Heejo Lee [a,*], Adrian Perrig [b], Hyogon Kim [a]

[a] *Korea University, Seoul 136-713, Republic of Korea*
[b] *ETH, Zurich 8092, Switzerland*

A R T I C L E   I N F O

Article history:
Received 14 April 2014
Received in revised form 5 March 2015
Accepted 27 March 2015
Available online 3 April 2015

Keywords:
Distributed DoS attack
IP spoofing prevention
Packet marking and filtering
BGP-based anti-spoofing extension
SDN security

A B S T R A C T

Internet attacks often use IP spoofing to forge the source IP address of packets, and thereby hide the identity of the source. It causes many serious security problems such as the difficulty of packet authenticity and IP traceback. While many IP spoofing prevention techniques have been proposed apart from ingress filtering, none have achieved widespread real-world use. One main reason is the lack of properties favoring incremental deployment, an essential component for new technology adoption. An incrementally deployable protocol should have three properties: initial benefits for early adopters, incremental benefits for subsequent adopters, and effectiveness under partial deployment. Since no previous anti-spoofing solution satisfies all three properties, we propose an anti-spoofing mechanism called "BGP-based Anti-Spoofing Extension" (BASE). BASE is an anti-spoofing protocol designed to fulfill the incremental deployment properties. Furthermore, BASE is designed to work in the software-defined networks (SDN). It gives a motivation to network operators to adopt BASE into their network, since the idea of SDN supports the large scale network control with a simple operation. Based on simulations using a model of Internet connectivity, BASE shows desirable IP spoofing prevention capabilities under partial deployment. We find that just 30% deployment can drop about 97% of attack packets. It is shown that BASE not only provides benefits to early adopters, but also outperforms previous anti-spoofing mechanisms.

1. Introduction

IP spoofing, i.e., forging a packet's source IP address, remains an unsolved security problem in today's Internet [1–3]. These forgeries assist attackers to enable anonymity, indirection, and amplification exploits [1], such as circumventing source-based filtering and mounting denial-of-service (DoS). Most notably, an attacker may employ IP spoofing to congest a server's resources via a TCP SYN flood [4]. Backscatter traffic analysis infers that several hundred DDoS attacks employing spoofing occur daily [5]. On February 2014, a massive DDoS attack reached 400 gigabits per second in power hit EU and US-based servers, and the attack uses IP spoofing to leverage the exploits on the Network Time Protocol (NTP). The so-called NTP reflection attack shows that the IP spoofing is still a significant problem in todays Internet [6].

Contemporary attacks forgo IP spoofing and instead use enormous botnets to mask the perpetrator's source address during a DDoS attack. In addition to the prevalence of non-spoofed DDoS attacks, broad deployment of ingress filtering, e.g., 80% of ASes participating in MIT Spoofer project are non-spoofable [7], induces the misconception that spoofing is not an open problem. However, 20% of networks suffice to generate spoofed traffic towards any target [3]. Consequently, ingress filtering is insufficient unless it is fully deployed [8] because it only filters outgoing packets so that it benefits to another side of network instead of the side that adopts ingress filtering. Furthermore, the diversity of new exploits give enough motivation to continue spoofing-based attacks [1,9], which include DNS amplifier attacks, TCP reset attacks, spam filter circumvention, network scans, and DNS poisoning. Recently, Qian and Mao also found a new exploit using IP spoofing, which leverages the predictability of TCP sequence numbers on firewall middleboxes, enabling off-path TCP attacks [10]. For many years, several anti-spoofing solutions have been suggested but none achieves wide-spread deployment.

In "Crossing the Chasm", Moore notes that customers for a technological innovation range from *early adopters*, to the *early majority*, to the *late majority*, and finally to *laggards* [11]. A central difference in the deployment of networking protocols is the availability of hardware and software that implements the protocols. Although the market for Internet technology differs from a

* Corresponding author.
E-mail address: heejo@korea.ac.kr (H. Lee).
URL: http://ccs.korea.ac.kr (H. Lee).

http://dx.doi.org/10.1016/j.comcom.2015.03.003
0140-3664/© 2015 Elsevier B.V. All rights reserved.

mainstream product, we can still draw a partial analogy. We conjecture that the deployment of networking protocols follows a similar trend: early adopters with a critical need for some new technology start to use the technology to their network. As the larger network operators recognize the necessity of the technology and observe customer demand, they also implement the feature. This causes the early mainstream network operators to deploy it. Finally, some network operators may not update their network frequently and thus they require a longer time until they implement the functionality. In terms of incremental deployability, a viable protocol needs to have three properties: initial benefits for the early adopters, incremental benefits for the early majority, and effectiveness under partial deployment.

- **Initial benefit**: The protocol needs to provide initial benefits for early adopters. Ideally, the initial deployments already provide a benefit.
- **Incremental benefit**: The protocol needs to provide incremental benefits for the early majority. Such benefits should increase as deployment proceeds.
- **Partial deployment**: The protocol needs to provide properties such that a proportionally small deployment becomes sufficiently effective. Broad deployment requires a prolonged period; thus, a practical protocol approaches full strength when approximately 30–50% of routers deploy the mechanism. This requires about 10% of larger ASes.

An anti-spoofing protocol needs to be not only technically sound but also economically acceptable. Unfortunately, currently proposed IP spoofing prevention mechanisms are inadequate, especially in the dimension of providing incentives for deployers.

In order to satisfy above three properties for a viable protocol, we propose a new mechanism called “BGP-based Anti-Spoofing Extension” (BASE). BASE is designed with a consideration for implementation on software-defined networks (SDN). SDN provides a logically centralized and programable controller to manage the entire network by separating control plane and data plane. With the separation of the control plane and the data plane, the network control is moved to a logically centralized controller in the control plane, and network devices in the data plane become simple packet forwarding devices. SDN enables network operators to more flexibly program and control their networks, while the traditional network is rigid in terms of network dynamics. Driven by this, SDN emerged as an innovative network architecture, and it is becoming a reality on modern Internet. BASE can be easily adopted and deployed to the real network through the growth of SDN environments.

BASE consists of four phases: distribution of marking values, filter invocation, packet marking and filtering, and filter revocation. Valid marking values are distributed among BASE routers[1] using BGP update messages. The marking values for each BASE router computed by SDN controllers with cryptographic hash chains. Under the occurrence of a spoofing attack, a controller in a victim network sends invocation messages to SDN controllers in other SDN ASes, and the controllers which receive the invocation messages initiate packet marking and filtering for each BASE router under control. Communication among BASE entities can be performed using optional transitive attributes in BGP [12], which enables to deliver messages under partial deployment. Then, only legitimate traffic traveling a valid path can have a correct mark but spoofed packets have incorrect marking values so that they are dropped by an intermediate BASE router. Furthermore, the capability of invoking the filtering network of BASE routers can be given to the BASE deployers, which becomes the direct benefit when adopting BASE in their network.

[1] Note that, the BASE router means network routers under a SDN control that follows BASE rules.

One filtering scheme that shows incremental deployability is route-based distributed packet filtering (DPF) [8]. DPF can drop spoofed packets traveling unexpected routes from specified source addresses. Filtering quality increases according to the degree of deployment. However, it does not give direct benefit to the adopters but everyone shares the benefit of filtering spoofed packets, which discourages deployment of DPF.

In experiments, BASE was implemented on virtual OpenFlow networks using Mininet [13], OpenvSwitch [14] and POX [15], in order to verify the correct operations of BASE. The power for filtering spoofed packets rises substantially as the number of deploying entities increases. With only about 30% of ASes deploying the mechanism, we can filter about 97% of attack packets. We show that BASE outperforms previously proposed schemes in terms of filtering spoofed packets while satisfying the incrementally deployable properties.

The contributions of this work can be summarized as follows:

- BASE is the first work considering the incrementally deployable properties by giving a visible benefit to early adopters.
- BASE provides on-demand filtering, thus it reduces unnecessary overhead while keeping enough security.
- BASE is implemented on a popular SDN open-source project with only using standard OpenFlow APIs.
- BASE shows that IP spoofing attacks can be effectively filtered with limited deployment.

The rest of this paper is organized as follows. Section 2 describes the background of SDN and the fundamental techniques of our work. Then, we give a detailed description of BASE in Section 3 and security analysis in Section 4. Section 5 outlines the experimental settings and results. In Section 6, we discuss several issues which arise when BASE is performed in a real network. Section 7 describes related work, and finally we conclude this paper with remained challenges in Section 8.

2. Background

To better understand the BASE mechanism, we introduce the historical background, basic architecture, and application fields of SDN. Moreover, the fundamental techniques we applied to the BASE mechanism will be described in detail.

2.1. Software-defined networking

Network management is often considered a tedious task. In the modern Internet, networks are comprised of a large number of network devices such as switches and routers, moreover billions of Internet user demands have occurred simultaneously. Network operators are responsible not only for configuring the network devices, but also satisfying the user demands by implementing a high-level network management. Unfortunately, the current network system has a lack of capability to support the demands, as a result the idea of software-defined networking has arisen.

SDN is a new paradigm of the networking. According to the definition of SDN described in the Open Networking Foundation (ONF) white paper [16], “In the SDN architecture, the control and data plane are decoupled, network intelligence and state are logically centralized, and the underlying network infrastructure is abstracted from the applications.” More precisely, SDN transforms the network devices in the data plane into simple packet forwarding devices, while the control logic that supervises the entire network is separated to the programable controller implementing the control plane. Sezer et al. [17] summarize four key features for SDN.

- Separation of the control plane from the data plane.
- A centralized controller and view of the network.
- Open interfaces between the devices in the control plane and data plane.
- Programmability of the network by external applications.

These features provide the following benefits. First, it makes it easier to apply new technologies into the network with simple software updates, while the legacy network needs to perform a re-configuration of all devices for a simple adjustment. Second, the SDN controller provides a centralized control method that makes it easier to manipulate a large number of devices. Network operators only need to update software in the control plane instead updating every network devices one by one. Therefore, the benefits encourage network operators to apply new methods to their network.

Although the term "SDN" seems to have appeared recently, the concept of SDN, in part of efforts to make networks more flexible and programmable, has been evolved over the past twenty years. To bring the idea of SDN to reality, many research groups and industry groups are involved the implementation of SDN. Tempest [18], RCP [19], Yan [20], Ethane [21] and OpenFlow [22] have been proposed by researchers, and now, many commercial switch vendors including HP, NEC, Toroki, etc., support OpenFlow. Driven by this, many network operators are implementing SDN in their network [23,24].

2.2. SDN as a security tool

According to the growth of SDN, supporting network dynamics and low operating costs using SDN has being researched to be applied in many areas. Network security is one of the areas [25].

The SDN architecture can be leveraged to improve network security with the flexibility and programmability of SDN. It brings highly reactive network monitoring, analysis, and response in point of view of security. More precisely, the network operators can easily deploy new security methods to their network. The security methods monitor their network to report anomaly behaviors to the controller. The logically centralized controller, the key entity of SDN, runs applications to analyze the phenomenon on the network. Based on the analysis, new security policy can be propagated to the entire network. This highly organized architecture remarkably enhances the security level of the network.

From the enhancement, many security researchers recognize SDN as an efficient tool for secure network design. DDoS mitigation using OpenFlow have become an interesting research topic. Braga et al. [26] stated a DDoS detection method based on network flow features monitored by NOX/OpenFlow switches. They investigated suspicious flows by analyzing the flow features with Self Organizing Maps (SOM). Lim et al. [27] proposed a DDoS countermeasure for blocking botnet-oriented attack traffic. Their system is mainly focused on the DDoS attacks which typically target specific services in application layer, and redirects the attack traffic to certain ports. Shtern et al. [28] proposed a reference architecture which performs detection and mitigation of low and slow DDoS attacks.

Detecting abnormal traffic like network scanning has been addressed by Mehdi et al. [29]. They developed a SDN application working on the NOX [30] and OpenFlow switches to detect scanning attacks in home and office networks. Jafarian et al. [31] described a moving target defence approach OF-RHM. The system is to defeat the scanning attack via mutating IP addresses. CloudWatcher [32] is a security monitoring framework which helps an operator to monitor a cloud network.

As an application of SDN, the extensions of IDS (Intrusion Detection System), IPS (Intrusion Prevention System) or firewall have been proposed. Skowyra et al. [33] proposed an intrusion detection method that utilizes the SDN architecture. FlowGuard [34] is a centralized SDN firewall which provides detection and resolution of firewall policy violations by enforcing the policy on top of a controller. And Qazi et al. [35] showed that SDN can be useful for deploying security policies.

Even though the numerous researches with various security topics have been studied, only few studies have been discussed for mitigating IP spoofing attacks. Bi and his research group [36–38,37] present the results of indepth researches on a collaboration of IP spoofing defence and SDN. SEFA [36] is a lightweight framework for route-based IP spoofing filtering. SEFA enables easy installation of filtering application by decoupling a filtering rule generation from devices. However, for the IP spoofing filtering, they simply applied the existing filtering scheme IDPF [39] as a filtering application. VASE [37] is another anti spoofing scheme which uses sampling and on-demand filtering configuration. The system is an extension of VAVE [40] which is a source address validation solution proposed by themselves.

In this paper, we also apply the SDN architecture as a security tool to thwart the threat of IP spoofing. BASE is designed to operate on the SDN architecture: the packet forwarding rules are determined by the controller and propagated to the network. The network switches forward or drop the packets according to the rules. New policy or forwarding rules can be easily propagated with simple updates. Filtering-on-demand also can be archived through the controller. Consequently, SDN makes BASE highly scalable and adoptable.

2.3. Fundamental techniques of BASE

BASE is founded on three techniques: Message Authentication Code (MAC), one-way hash chains and packet marking. MAC and one-way hash chains are used for generating a cryptographical unique value for a filter node, and packet marking is used for storing and delivering the value to destinations.

Message Authentication Code (MAC). Cryptographic approaches improve the strength of packet marking under the attacker's forgery of marking values as well as source addresses. Since the marking field spoofing diminishes the effectiveness of packet marking [41], we use a cryptographic Message Authentication Code (MAC) to protect the integrity of marking values. For example, a Pseudo-random function (PRF) [42] can be used as a MAC. A PRF takes two arguments, a key and an input, then produces an output that is indistinguishable from a random value as long as the key is secret.

One-way hash chains. One-way hash chains—cryptographic primitives frequently used in the design of secure protocols—are used to compute marking values. Computation of a chain of marking values has advantages of reducing forgeability of marking values and enhancing routability of legitimate packets, without prior knowledge of packets' paths.

Marking values are pre-computed and distributed between neighbor BGP filters, then they are used for marking and filtering. No additional computation is required during packet processing except to compare and update a marking value with a table lookup.

Packet marking. Several fields have been proposed to store a marking value in a packet. They include the record route option in the Internet Protocol (IP) [43], the IP identification (ID) field [44], the IP header available by compression [45], and the IP Type of Service (ToS) field [46]. Various tradeoffs exist among the different fields. The 16-bit IP identification field (which is used to reconstruct fragmented packets) has received the most attention in previous work [44,47–49]. Dean et al. [46] identify 25 bits in the IP header for marking, which include 8 bits of ToS and 1 reserved bit of the fragmentation flag bits in addition to the 16 bits of the ID field. Seo et al. [50] also used 25 bits for marking as a filter distribution method to defend against DDoS attacks. In this paper, we consider for BASE to use the 16-bit IP ID field to store marking values, as it is only used 0.25–0.50% of the time [51]. The problem of packet marking while fragmentation is discussed by Belenky and Ansari [52]. In Section 6.3, we will present a detailed discussion of the packet fragmentation compatibility in BASE when using IP ID field. Without loss of generality, the marking field size is a system parameter, which can be extended to more than 16 bits, e.g., up to 25 bits [46,53,54].

3. The BASE mechanism

This section proposes a new mechanism called "BGP-based Anti-Spoofing Extension" (BASE), which combines the features of Pi [48] and DPF [8]. BASE functions as an anti-spoofing solution by performing per-packet deterministic packet marking (Pi feature) with overloading a routing protocol BGP (DPF feature) to propagate marking information. In addition to using the features of Pi and DPF, BASE further enables the three deployment properties. In BASE, path-based marking enables in-network filtering before the packets reach the victim network and provides significant benefit to early adopters than non-deployers.

For this study, we assume the following. We first assume an attacker sends spoofed packets to the target node to hide the identity of the attacker. Second, a victim has the ability to recognize a spoofing attack. There are several ways to identify spoofing packets at victim side such as TCP-specific probing and SYN cookies [2]. In TCP-specific probing, for example, a victim replies with a crafted TCP ACK such as changing TCP window size. Since the sender cannot see the crafted ACK, the victim can identify spoofed packets by observing the sender's responses that should meet changed TCP window size. Once the attack is recognized, the victim can utilize BASE to protect itself from the attack. Third, we assume BASE application working on the logically centralized controller can utilize the packet forwarding rules for the SDN switches. The controller can have a marking and filtering policy, so BASE has its roots in network-based filtering. Fourth, each BASE router within an SDN AS can be updated to perform the BASE mechanism with the following assumptions:

- **Per-AS key:** Each SDN AS has a secret key to compute marking values; the key is shared by routers within the AS.
- **Marking in IP headers:** We assume that the IP header has sufficient space to store a marking value.
- **Router marking and filtering:** The BASE router(s) on the border of an SDN AS mark every outgoing packet and filter every incoming packet without a correct mark.

BASE routers do not need to share common keys, but each router only has a local symmetric AS key. Assume that the AS key is at least 128 bits long, which offers very strong security even if the attacker learns a lot of marking values. A shorter MAC does not make it easier to break the AS key, in fact, it makes it harder because fewer MAC bits are available to verify the correctness of a guessed key in a brute force attack. Sharing keys within an AS is simple – no sophisticated key management scheme is necessary.

The BASE mechanism distributes valid marking values via BGP update messages [12]. BGP (Border Gateway Protocol) is the de facto standard inter-AS routing protocol in the Internet. BGP obtains subnet reachability information from neighboring ASes and propagates it to other BGP-enabled routers, so all the ASes know about the subnets and how to get there. Under the SDN architecture, the logically centralized SDN controller is responsible for the BGP process for the SDN AS [55]. A BGP handling application running on the SDN controller integrates BGP messages and performs routing updates for each router on the SDN AS. BASE is designed to follow the BGP process on the SDN architecture for the marking computation and distribution.

The marking in BASE is "path-based," instead of "IP-based." This means the use of network addresses (prefixes of an IP address) instead of individual IP addresses. This reduces the storage required for marking values, but collective filters can effectively detect spoofed packets. The marking value in the filtering table of each router is mapped based on the source's network address, similar to the destination's network address used in the routing table.

The next subsection describes BASE architecture on SDN and the four phases of BASE. We show how the proposed mechanism works in environments of full deployment, partial deployment, and asymmetric routing paths.

3.1. BASE mechanism on SDN

In the legacy network, once the flow management or policy has been determined, there is one only way to adjust the policy by reconfiguring all devices. Such environment makes network operators hard to apply new techniques. By contrast, SDN separates the control plane responsible for flow control from data plane, and it manages each node through the centralized programable controller. It is much easier to apply a new technology into the network through a software program, since it only needs to update controller program for new policies. Driven by this, BASE was designed with a consideration of the operations under the SDN architecture to take benefits in terms of deployments, since the independent, centralized, and programable control plane encourages network operators to easily adopt BASE to a large scale network through simple software updates.

The overall architecture is shown in Fig. 1. A SDN AS is connected with the external border routers on neighboring ASes through the OpenFlow switches called BASE routers. BASE operates on the logically centralized controller as an application, that is consist of a *BGP Route* module, a *Marking Value Calculator*, and a *Filtering Table*. The *BGP Route* module is responsible for the BGP route updates. The *Marking Value Calculator* utilizes the BGP information to calculate the corresponding marking values for each routing path and stores the marks into the *Filtering Table*. Fig. 2 depicts a simple example of the *Filtering Table* structure. The specific usage of these marks and the marking value generation algorithm will be explained in next Section 3.2.

Under the SDN architecture, the controller have an authority to construct routing paths, hence BASE routers initiate routing paths via the SDN controller. At the first step, if one of the BASE routers receives BGP update messages from neighboring ASes, the BASE router simply delivers the messages to its controller. Once the SDN controller receives marking values of other SDN AS via the BGP update messages, the controller computes valid marking values for routers under its control, and re-distributes BGP update messages with its marking values to the neighboring ASes again. After finishing the distribution, the SDN controller configures the *Filtering Table* with the marking values.

Each BASE router updates their flow rules in accordance with the marking values obtained from the *Filtering Table* on the controller, and simply follows the flow rules (as like the exact definition of data plane on SDN). Fig. 3 outlines how the BASE router works on the OpenFlow switch as an example. OpenFlow

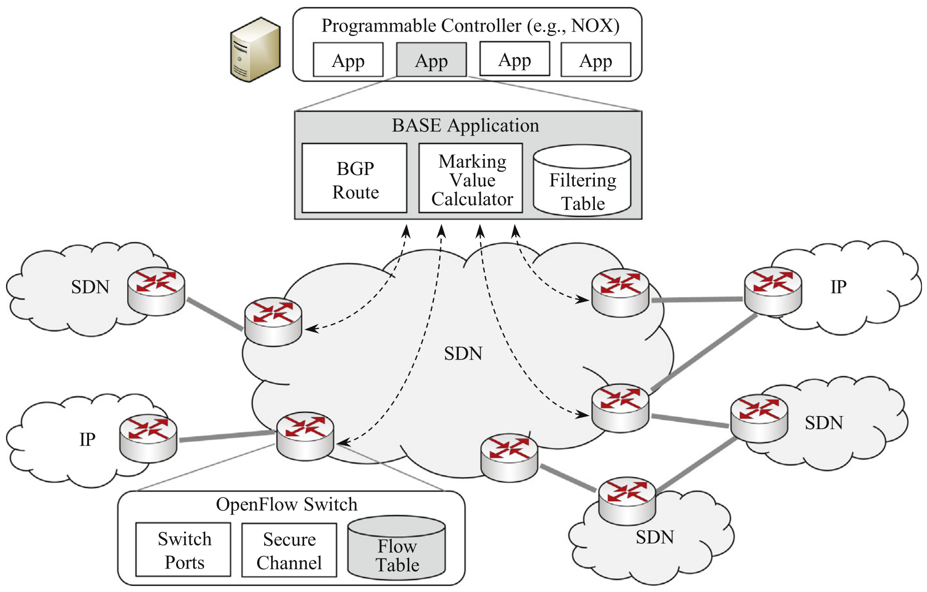


Fig. 1. The operation of BASE on the SDN architecture. BASE works on the SDN controller as an application to manage the routing pathes and filtering rules. The BASE routers simply forward packets according to the filtering rules.

Source	Prev_mark	Next_mark
163.152.0.0/16	0x0873	0xF994
128.2.0.0/16	0x3DDA	0x1258
⋮	⋮	⋮

Fig. 2. The entities of *Filtering Table* in the BASE application.

switch has three components, *Switch Ports*, *Secure Channels*, and *Flow Tables*. The *Switch Ports* link to the other switches or hosts for the simple packet transmission. The *Secure Channels* are for communication with the controller, and the *Flow Tables* contain every flow information and associated instructions for each flow entry. One single OpenFlow switch is available to contain multiple *Flow Tables* which contains different rules and associating instructions. This property gives us the great flexibility to response for various circumstances. We leverage the multiple *Flow Tables* which contains different flow rules, for instance one *Flow Table* contains simple forwarding rules as like ordinary routers, and another *Flow Table* contains filtering rules, to support both the ordinary packet forwarding and the packet filtering for a normal situation and the under attack situation respectively.

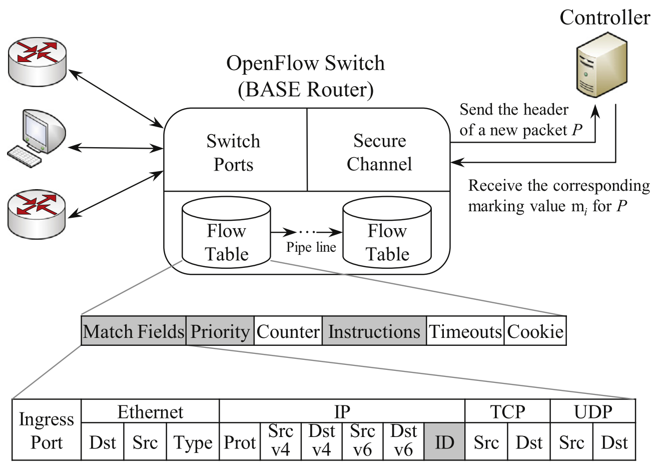


Fig. 3. An example of basic operations of BASE router for an OpenFlow switch. The ID in IP header can be used for storing the marking values for BASE operation.

According to the latest OpenFlow specification version 1.4.0 published on October 14, 2013 [56], the *Match Fields* should contain at least 12 values for each flow as shown in Fig. 3. We only add one more value, the IP identification field that stores the marking values, to the *Match Fields* for the BASE operation. Since OpenFlow supports 41 values of *Match Fields* for each flow, adding the ID field does not cause the serious overheads and system changes.

Once a BASE router receives a new packet through the one of the switch ports, it simply compares the packet information against every flow entries which has been contained on the *Flow Table* using the *Match Fields*. If the packet matches none of the entries (table-miss), the BASE router sends a `Packet-In` message to the controller via a *Secure Channel*. The BASE application on the controller receives the message and searches corresponding marking values on the *Filtering Table*, and send a `Flow-Mod` to add the flow entries to the *Flow Table* on the BASE router with associated instructions in accordance with the *Filtering Table* (see Fig. 4). Otherwise, the BASE router forwards or drop the packet according to the associated instructions on *Flow Table* via `Apply-Action`. Fig. 5 exhibits examples of flow entries and associated instructions.

The first entry, for example, is for the packets of the source IP address 1.2.3.4 and mark 0x07DE. According to the `OFPAT_SET_FIELD` in the action structures, the packets will be edited with new mark 0x9AF2 and forwarded to the next hope through port number 6. Some packets generated by same source are able to obtain different marks due to the routing asymmetry in the Internet [57]. BASE also supports the multiple marks for the routing asymmetry. The second entry shows a different forwarding rule for the same packets which arrive through a different routing path. Supporting the asymmetric path on BASE will be explained with more detail in the following section (see Section 3.4). The third entry drops all the packets come from same source IP 1.2.3.4, but legitimate packets with the correct mark never hit this entry due to the priority. In case of the spoofed packets which have incorrect marks will hit the third entry, thus the BASE route will drop the spoofed packets immediately. Consequently, BASE routers do not

```
def _handle_PacketIn (self, event):
1.dpid = event.connection.dpid
2.packet = event.parsed
3.dstaddr = packet.next.dstip
4.srcaddr = packet.next.srcip
5.prevmark = packet.next.id
6.IF dstaddr in self.arpTable[dpid]:
7.    IF prevmark in self.filterTable[dpid][srcaddr]:
8.        prt = self.arpTable[dpid][dstaddr].port
9.        mac = self.arpTable[dpid][dstaddr].mac
10.       nextmark = self.filterTable[dpid][p_mark].n_mark
11.       action = [ ]
12.       action.append(of.ofp_action_dl_addr.set_dst(mac))
13.       action.append(of.ofp_action_nw_id.set_id(nextmark))
14.       action.append(of.ofp_action_output(port = prt))
15.       match = of.ofp_match.from_packet(packet, inport)
16.       msg = of.ofp_flow_mod(command=of.OFPEC_ADD,
              idle_timeout = FLOW_IDLE_TIMEOUT,
              hard_timeout = of.OFP_FLOW_PERMANENT,
              buffer_id = event.ofp.buffer_id,
              actions = actions,
              match = of.ofp_match.from_packet(packet, inport))
17.       event.connection.send(msg.pack())
18.   ENDIF
19. ENDIF
END of code
```

Fig. 4. A pseudo code of BASE application for a *Flow Table* updates. When a `Packet-In` message arrives to the controller, BASE application replies a `Flow-Mod` message with the corresponding new marking value.

need to understand how BASE mechanism works, but just follows *Flow Table* rules. This means the BASE router acts like a simple packet forwarding device as describing how the data plane works in SDN.

3.2. Four-phase BASE mechanism

The framework of BASE extends the concept of distributed packet filtering (DPF) with cryptographic packet marking. This enables non-adjacent BASE-enabled SDN ASes to verify path correctness. The BASE mechanism runs on-demand filtering for specific destination addresses. Thus, only during DoS attacks, the victim can initiate collective filtering of attack packets crowding the victim's network.

For the purpose of distributing filtering information, one approach is the use of BGP update messages to coordinate between routers [8]. Alternatively, we can design our own distribution protocol using piggybacking on regular packets or generating information packets. The SAVE protocol [58] is an example of designing a new protocol to verify the correctness of the source address of each incoming packet. There are advantages and disadvantages for using legacy protocols or designing our own protocols, as an information distribution scheme. In this paper, it is assumed that BGP update messages are used for distributing filtering information, while preserving the primary properties of deployment issues.

BASE works according to the following four phases. We let s denote the source AS, t is the destined AS, and v is the current filtering AS. Now, a packet (s,t) is passing through v's filter.

Phase-1 Distribution of marking values. BASE-enabled controllers distribute marking values using BGP update messages. The marking values are computed by a one-way hash chain, i.e.,

Ingress Port	Ethernet			IP						TCP		UDP		Priority	Instructions
	Dst	Src	Type	Prot	Src v4	Dst v4	Src v6	Dst v6	ID	Src	Dst	Src	Dst		Actions
Port 2	*	*	*	*	1.2.3.4	*	*	*	07DE	*	*	*	*	2	Set-ID 9AF2 Port 6

Ingress Port	Ethernet			IP						TCP		UDP		Priority	Instructions
	Dst	Src	Type	Prot	Src v4	Dst v4	Src v6	Dst v6	ID	Src	Dst	Src	Dst		Actions
Port 3	*	*	*	*	1.2.3.4	*	*	*	3F9A	*	*	*	*	2	Set-ID 9AF2 Port 6

Ingress Port	Ethernet			IP						TCP		UDP		Priority	Instructions
	Dst	Src	Type	Prot	Src v4	Dst v4	Src v6	Dst v6	ID	Src	Dst	Src	Dst		Actions
*	*	*	*	*	1.2.3.4	*	*	*	*	*	*	*	*	1	Drop

Fig. 5. Flow rules for BASE operations. Every packets follow the *Instructions* by matching the *Match Fields* and the *Priority*.

$m_i = MAC(k_i, m_{i-1})$, where i denotes the index of a filter node (from $i = 1$ to all filter nodes), and k_i is the secret key and m_0 is the prefix of the source AS. The marking values are distributed using BGP updates and stored in the *Filtering Tables* of BASE applications. This is a once-only operation unless the BGP path has been changed.

Phase-2 Filter invocation. A controller in a victim network can be used for invoking packet marking and filtering for packets destined to the victim network, through BGP update messages. Upon receiving an invocation message, a BASE router starts packet marking and filtering for the corresponding addresses.

Phase-3 Packet marking and filtering. BASE routers mark outgoing packets using the marks obtained from *Filtering Table* on the controller and filters incoming packets without a correct mark. Every packet with the same source address will have the same mark when it leaves a BASE node, even though it may have arrived with different marks through different interfaces. This replacement scheme allows the BASE mechanism to work in asymmetric routing paths without additional space, keeping packet size constant.

Phase-4 Filter revocation. A BASE router in the victim network terminates marking and filtering of packets destined to the victim via BGP update messages.

Internet connectivity can be represented by a graph $G = (V, E)$ where V is the set of nodes and E is the set of edges. The graph G represents the AS-level connectivity such that a node is an AS and an edge is a link between two nodes. A path $\mathcal{P}(s,t)$ is an ordered set of consecutive nodes from a source s to a destination t such that $\mathcal{P}(s,t) = \{v_1, v_2, \ldots, v_n\}$ where $v_1 = s$ and $v_n = t$. The marking values are computed as shown in Fig. 6. The marking value of v_i for (s,t) is defined by $m_i = MAC(k_i, m_{i-1})$ where k_i is the key of v_i and m_0 is the prefix of s. The computed marking value for each node is distributed to next nodes as described in Fig. 7.

Each BASE node (BASE-enabled SDN AS) has additional features for packet marking and filtering. We call this node a BASE filter and the function of a BASE filter is formally described in Fig. 8. Each BASE filter has a *Filtering Table F*. If we can store only one marking value in each record of a *Filtering Table*, we call this "one mark" and if we can store multiple marking values, we call this "multiple marks." By default, we consider BASE on "multiple marks." In this case, we can store all possible marking values in the *Filtering Table*. In the distribution phase (Phase-1), when a BASE filter receives a marking value, the BASE filter stores the marking value in its *Filtering Table*. In the marking and filtering phase (Phase-2), when a BASE filter receives a packet (s,t), the filter forwards the packet to $\mathcal{R}(t)$ with a new mark m_i only if $m_{i-1} \in F(s)$, otherwise it drops (s,t). $\mathcal{R}(t)$ denotes t's entry in v_i's *Flow Table*.

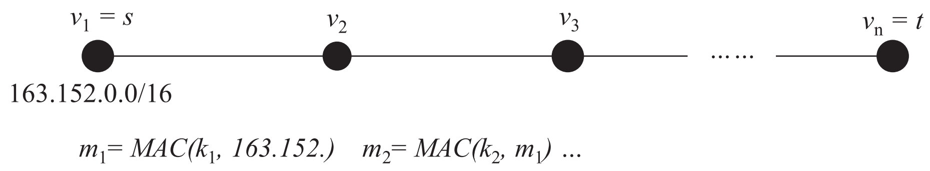


Fig. 6. Marking value computation for each node. For example, v_2 computes m_2 via a one-way hash chain with m_1 which comes from source, and forwards m_2 to next node for packet construction (s, t).

```
Distribution of marking values
1.$m_0$ = the prefix of the source AS
2.FOR each BASE filter $v_i$ from $i$ = 1 to all filter nodes
3.    $m_i = MAC(k_i, m_{i-1})$
4.    forwards $m_i$ to next BASE filter nodes
          by using the BGP optional transitive attributes
5.ENDFOR
END of algorithm
```

Fig. 7. Distribution algorithm of marking values.

```
Packet marking and filtering
1.FOR each BASE filter $v_i$ from $i$=1 to all filter nodes
2.    IF $m_{i-1} \in F(s)$    // F is a Filtering Table
3.        forwards $(s, t)$ to $\mathcal{R}(t)$ with a new mark $m_i$
4.    ELSE
5.        drops $(s, t)$
6.    ENDIF
7.ENDFOR
END of algorithm
```

Fig. 8. Packet marking and filtering algorithm in a node.

3.3. Fully-deployed BASE on symmetric paths

A routing path is called symmetric if the path has the same forward and backward path between two nodes. A symmetric routing path of (s, t) implies that the forwarding path of (s, t) is a subgraph of the BGP tree such that $\mathcal{P}(s, t) \subseteq \mathcal{B}(s)$, where the BGP tree $\mathcal{B}(s)$ is a tree expanded by BGP updates for s. This BGP tree is an s-rooted spanning tree constructed by the best routes destined to s. Thus, the propagation of marking values for (s, t) follows the path $\{v_1, \ldots, v_n\}$ in the spanning tree. This becomes the routing path for (s, t) in symmetric routing. Since BGP updates flow to the opposite directions of the chosen best routes to s, the BGP flooding paths are not always equivalent to the routing path starting from s. We will discuss asymmetry of routing paths in the next subsection.

Fig. 9 shows how BASE works for two nodes, s and t, in a network. Marking values for s are distributed by the use of BGP updates as shown in Fig. 9(a). Using a secret key k_i of v_i, an unpredictable marking value m_i is computed by $MAC(k_i, m_{i-1})$ as follows.

$$m_1 = MAC(k_1, Pref(s))$$
$$m_2 = MAC(k_2, m_1)$$
$$m_3 = MAC(k_3, m_2),$$

where $Pref(s)$ denotes a function that extracts the network address of source s. Fig. 9(b) shows the invocation messages being propagated from t which is under a spoofing attack. After that, each node drops packets destined to t without a correct mark, which are attacking packets with spoofed source addresses and/or fake marks. Fig. 9(c) shows that each node checks the marking value of a packet (s, t) and inscribes a new marking value before forwarding it to the next node.

The MAC computed at each node with the node's secret key protects the integrity of the marking values. As shown in Fig. 10, an attacker v_4 who does not know key k_1 of s, and similarly, v_5 who does not know the secret key k_2 of v_2 can at best guess the correct marking values v_1 and v_2. A correct guess becomes exponentially harder with an increasing number of marking values that need to be guessed. An incorrect guess will result in a dropped packet.

3.4. Fully-deployed BASE on asymmetric paths

There are non-negligible portions of routing asymmetry in the current Internet [57]. A recent study measured US academic networks display about 14% routing asymmetry, while commercial networks show about 65% routing asymmetry on the AS-level [59].

Fig. 11 shows the case of asymmetric routing paths, where the routing path from s differs from the routing path from t. Note that the symmetric routing path from t is the reverse of the BGP path. This causes packets to travel a different path from the path transferring marking values. Since BASE replaces the received mark with the new one, the node in a merged point can update a packet in asymmetric routes with a correct mark. For instance, if the BASE filter receives any of the valid marks m_i and m_j through its corresponding interface, the mark is changed to m_k before forwarding it to the next node, as shown in Fig. 11. This replacement resolves the risk of dropping legitimate packets so that BASE works under asymmetric forwarding paths.

Fig. 12 shows the *Flow Table* at v_4. If a node v_4 receives the packets with m_2 or m_3, v_4 checks its *Flow Table* whether m_2 or m_3 exist in source s's entry. Once it has, it checks its associated instructions to send the packets to the next node. In this example, the packets are forwarded to the next node v_5 with the replaced marking value m_4. There can be multiple values of previous marks for one source, nonetheless, next marks are always a single unique value. In Section 3.6, we describe how BASE utilizes multiple marks in more detail.

We allow packets to travel through every possible path with a valid marking value from source to destination in order to prevent dropping legitimate packets, whether or not the path is asymmetric.

3.5. Partially-deployed BASE

BASE has a salient feature in that it works in partial deployment. It substantially increases power at larger adoption rates. Any individual deployer receives an additional reward from the more powerful BASE network, simply by deploying to their networks. This unique characteristic comes from the incremental deployability of BASE. This is an excellent motivation to adopt this mechanism, and thus BASE can be deployed to the current Internet.

We now explain how BASE works for partially-deployed environments. Assume there are k filters among n nodes such that

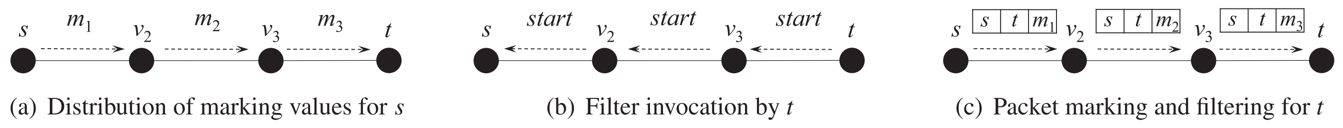

s
m_1
v_2
m_2
v_3
m_3
t
(a) Distribution of marking values for s
start
(b) Filter invocation by t
s | t | m_1
s | t | m_2
s | t | m_3
(c) Packet marking and filtering for t

Fig. 9. Working example of fully-deployed BASE. (a) Distribution of marking values along with the BGP update messages from *s*. (b) Filter invocation for packets destined to *t* with the BGP update messages from AS *t*. (c) Packet marking and filtering of spoofed packets destined to *t*, without correct marking values for the source address *s*.

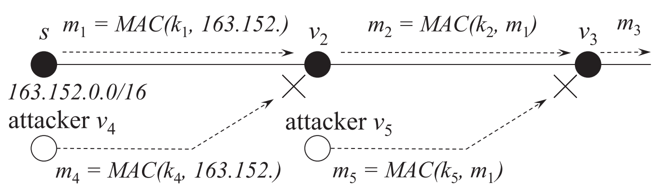

s
m_1 = MAC(k_1, 163.152.)
v_2
m_2 = MAC(k_2, m_1)
v_3
m_3
163.152.0.0/16
attacker v_4
attacker v_5
m_4 = MAC(k_4, 163.152.)
m_5 = MAC(k_5, m_1)

Fig. 10. Spoofed packets dropping case, i.e. m_4 can be recognized as a spoofed packet by v_2, since v_2 already stores correct marking value m_1 for address range 163.152.0.0/16.

$w_1, w_2, \dots, w_k \in \mathcal{P}(s,t)$ for $0 \leqslant k \leqslant n$. Then, any filter in the path, i.e., $w_i \in \mathcal{P}(s,t)$, can communicate with the next filter w_{i+1}, for $0 < i < k-1$, the same way as in the fully-deployed case. Non-BASE BGP speakers just relay the marking information because it is stored as optional transitive attributes.

Fig. 13 shows how BASE works in a partially deployed network. Any BASE filter can communicate with other BASE filters across non-filter nodes. Each BASE filter in the partial-deployment can mark and filter spoofed packets, e.g., the packets with m_x in Fig. 13. Even though an attacker may succeed in injecting spoofed packets into the normal flow through a non-filter node, these packets will be distinguishable at the next BASE filter. An example is filtering packets with m_y at the node *t*, as shown in Fig. 13. We call this the "tunneling effect" in which surrounding BASE filters protect non-filter nodes.

3.6. Multiple marks on asymmetric paths

The proposed BASE scheme is simple but powerful in protecting against spoofing attacks. Nevertheless, BASE cannot deal with certain cases when using "one mark" for each flow. Fig. 14 shows one case of partial deployment in asymmetric routing paths. Since the BGP path is different from the routing path, the recorded mark, m_1, is different from the arriving mark m_2, which is an unregistered but legitimate mark. This will cause the false dropping of a portion of legitimate traffic toward node *t*, as shown in Fig. 14. To eliminate the false dropping problem, we can store legitimate multiple marks in each record of a node's *Filtering Table*.

We use a *Filtering Table* for each BASE filter to mark and filter spoofed packets. If we store one mark in each record of a *Filtering Table*, each BASE filter has a flow rule F_e for a link *e*. An IP packet $M(s,t)$ arriving on *e* for some node $v_{i+1} \in V$ with a marking value

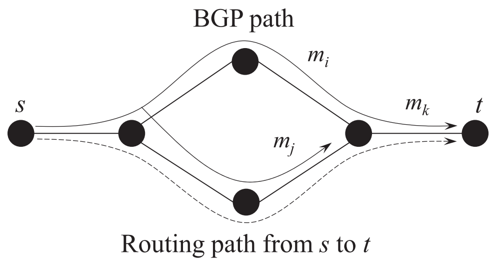
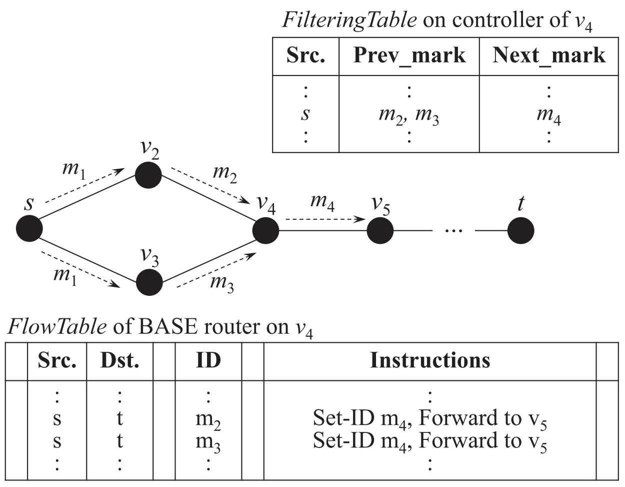


Fig. 11. Asymmetric routing paths when a routing path does not go through BGP paths.


FilteringTable on controller of v_4
s
m_1
v_2
m_2
v_4
m_4
v_5
t
v_3
m_3
FlowTable of BASE router on v_4

FilteringTable on controller of v_4

Src.	Prev_mark	Next_mark
:	:	:
s	m_2, m_3	m_4
:	:	:

FlowTable of BASE router on v_4

Src.	Dst.	ID	Instructions
:	:	:	:
s	t	m_2	Set-ID m_4, Forward to v_5
s	t	m_3	Set-ID m_4, Forward to v_5
:	:	:	:

Fig. 12. Storing multiple marks in asymmetric routing paths in order to admit every possible legitimate packet.

m_i is forwarded ($f_1(m_i) = 1$) if the flow rule (F_e) has the marking value m_i, i.e. $m_i \in F_e(s)$. Otherwise, they are discarded ($f_1(m_i) = 0$). Thus, $f_1(m_i)$ is a packet filter for each link on "one mark."

$$f_1(m_i) = \begin{cases} 1, & \text{if } m_i \in F_e(s) \text{ for each link } e \in v_{i+1}; \\ 0, & \text{otherwise.} \end{cases}$$

To eliminate the issue of false-positive under asymmetric environments, we can store multiple marking values in each record of a *Filtering Table*. IP packets $M(s,t)$ arriving on a node $v_{i+1} \in V$ with a marking value m_i are forwarded ($f_2(m_i) = 1$) if the node's *Filtering Table* (F) has the marking value m_i, otherwise they are discarded ($f_2(m_i) = 0$). Thus, $f_2(m_i)$ is a packet filter for each node on "multiple marks."

$$f_2(m_i) = \begin{cases} 1, & \text{if } m_i \in F(s) \text{ for each node } v_{i+1} \in V; \\ 0, & \text{otherwise.} \end{cases}$$

Although the BGP path and the forwarding path are different, packets would not be dropped. Thus, we can decrease false-positive (the proportion of legitimate packets that are incorrectly identified as attack packets) and get lower space and time complexity. Each BASE filter has fewer tables to store the same data when compared with the use of one marking value. *Filtering Table* for multiple marks and one mark have the following relationship.

$$F(s) = \bigcup_{e \in v_i} F_e(s)$$

Under full deployment, the number of filters for multiple marks is *n*, whereas the number of filters for one mark is 2*e*, where *e* is the number of edges in the network. Since $n < 2e$ for any graph with $n > 2$, it confirms that the BASE with multiple marks requires fewer tables than that with one mark.

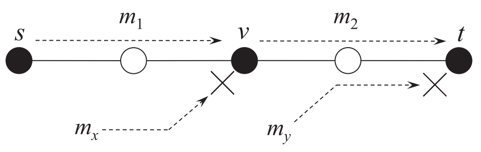


Fig. 13. Partial deployment tunneling effect: A black circle denotes a BASE filter while a white circle denotes a non-filter node.

The number of marking values in each record is at most the maximum of the node's degree (the number of the node's neighbors).

4. Security analysis: Which attacks did we stop?

4.1. Circumventing BASE routers

The first measure of the effectiveness of an anti-spoofing mechanism is the proportion of spoofed packets that are dropped prior to arriving at a victim's location. We consider a network with partial deployment as shown in Fig. 15.

In the case of ingress filtering [60], an attacker can mount an attack with spoofed packet (s, t) at 5 locations $(v_1, \ldots, v_5)$. In the case of DPF [41], only 3 locations $(v_1, \ldots, v_3)$ are available to an attacker for mounting a spoofing attack. The attacker at the locations $(v_1, \ldots, v_3)$ can generate spoofed packets that are indistinguishable by t because v_2 and v_3 are in the middle of the path between s and t. With BASE, however, no location can spoof a packet from s to t; only s can send a packet (s, t) to t. Even at v_2 and v_3, an attacker cannot send spoofed packets with the source address of s since the valid mark coming from s can be verified at t. This is the benefit of BASE by the tunneling effect shown in Fig. 13. In this analysis, it is assumed that eavesdropping transit traffic between ASes is not possible for an attacker.

BASE's enhanced protective power comes from the use of packet marking to allow non-adjacent BASE-enabled SDN ASes to verify the validity of the source address of traveling packets. This property enables BASE to be more effective than other schemes when partially deployed.

4.2. IP address and marking field spoofing

Fake marks inscribed in a packet before being sent by an attacker greatly reduce the effectiveness of packet marking [41]. BASE prevents an attacker from predicting the marking value since a chained MAC computation is used for computing marking values. Such a cryptographic marking mechanism renders a filter's marking values unpredictable by an attacker.

The success probability of an attacker injecting random marks is $1/2^{16}$ at best, which is only when the first BASE router that the attacker's packets encounter is located in the network on the path from the spoofed source to the destination. Moreover, the marking field size of 16 bits is a parameter of our system, it can be increased to match the desired level of security.

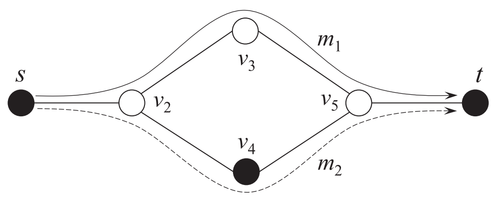


Fig. 14. Legitimate packet dropping case on asymmetric routing paths.

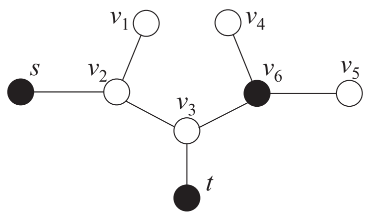


Fig. 15. An example of partially-deployed network: black nodes represent ASes which adopt an anti-spoofing mechanism.

4.3. Replaying valid marks

If an attacker can eavesdrop transit traffic, a valid mark can be copied to spoofed packets so that spoofed packets can go through BASE routers. However, valid marking values are not visible at any location before the BASE marking and filtering is invoked. Also, the marking values, even in the process of marking and filtering by BASE, flow towards the victim. Thus, an attacker cannot gather valid marks from the attacking locations, making BASE resilient to replay attacks.

4.4. DoS against controllers

DoS attacks with spoofed IP addresses may cause damages to not only the target system but also the SDN controllers on the attack path. Since the SDN switches are designed to generate a request packet to it's controller when a new flow packet has arrived, the switches under the spoofed packet flooding attack might craft an extremely large inbound stream of the requests which collapses the functionality of a centralized SDN controller. Therefore it causes denial-of-service of the controller, and this is the single point failure of centralized design of SDN. DoS against the centralized SDN controller can occur in any network that is implemented with SDN, especially the large scale ISPs. BASE is also suffering from the problem as long as BASE is working on the SDN architecture.

Driven by this phenomenon, the secure design of SDN has been studied using distributed SDN controllers with load balancing. Kreutz et al. [61] have explained the faulty design of SDN, and suggested a secure and dependable architecture of SDN with replication of the centralized controller. This replication distributes computation and prevents the single point failure. Benton et al. [62] have performed the OpenFlow vulnerability assessment including the denial-of-service risks. In the analysis, they warned that not only the packet flooding but also poor rule design can lead to saturating volumes of controller queries. Onix [63] was suggested as a distributed control platform to ensure the scalability and reliability of SDN. Dixit et al. [64] have focused on the issue of statically configured mapping of switches to a controller. Levin et al. [65] found the inconsistencies of global networks with a centralized controller.

Reducing of the data-to-control plane interactions is another approach to prevent the controller failure. AVANT-GUARD [66] also issued the inherent communication bottleneck that arises between the controller and switches, but it introduced an extension of OpenFlow, which reduces the packet exchange between the data plane and control plane. DIFANE [67] has been proposed from an idea in which the controller does not need to be involved in the real-time handling of data packets by distributing flow rules to switches.

BASE already has the similar benefits in terms of load balancing and reduction of packet exchange. First, BASE distributes the computational overhead among the BASE controllers located on an attack path. For example, let us assume that there is a simple

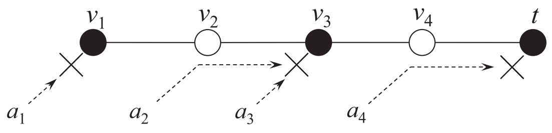


Fig. 16. Load balancing among BASE controllers.

network topology as shown in Fig. 16, and attackers are uniformly located on node v_{1-4}. When attackers a_{1-4} attempt to send spoofed packets to t, some spoofed packets from a_{1-3} could not reach to t because the intermediate BASE nodes v_1, v_3 discard the packets immediately. Consequently, a BASE controller on t receives queries for the packets only coming from a_4. The filtering on the intermediate nodes distributes the load of controller.

Second, BASE reduces the packet exchange between the control plane and data plane by packet marking. Once a new query arrived at a controller, BASE can install a powerful flow rule with a source address and a corresponding marking value. In accordance with the flow rule, next spoofed packets having the same source address will be dropped immediately without additional queries. This flow rule with marking value reduces the amount of packet exchanges between the controller and switches.

The benefits explain that BASE not only has resilience to the single point failure, but also can contribute to making SDN more secure and robust.

5. Evaluation

To evaluate the performance of BASE, two level experiments were performed: high level simulation on an Internet-scale topology for filtering performance estimation and low level simulation on SDN for BASE function estimation.

In the first experiment, an Internet-scale simulation was conducted for a performance estimation with different anti-spoofing mechanisms. We compared filtering performance of BASE with three different well-known filtering schemes according to filter placement strategy and filter deployment rate. The computation, communication and memory overhead were also analyzed in a practical manner.

In the second experiment, we show how BASE was implemented on the SDN framework and demonstrate BASE function working on multiple SDN entities such as controllers, switches and hosts. Through the experiment, BASE function including three key features such as asymmetric path support, availability under partial deployment, and spoofed packet filtering were simulated in a low level implementation.

BASE is an anti-spoofing mechanism interworking with the inter ASes. As we described before, BASE functions for the marking distribution, filter invocation, revocation, and filtering phase are emerged between the inter ASes. Driven by this, our evaluation was conducted with inter AS network model rather than intra AS model, and each node is classified as a legacy node or a filter node that adopt one of the anti-spoofing mechanism.

5.1. Internet-scale simulation

In this section, we estimate BASE performance comparing with different anti-spoofing mechanisms through an Internet-scale simulation.

5.1.1. Simulation environment

In order to measure the effectiveness of different anti-spoofing mechanisms, we first need to develop a means to produce an accurate model of today's Internet connectivity. We use the AS connectivity graph archived by NLANR from the Oregon Route Views project [68]. The AS graph used is the connectivity which consists of 22,000 nodes. We also use various 300-node subgraphs of the AS graph to evaluate diverse network topologies and perform faster simulation. To compare filtering performance, we simulate four mechanisms: ingress filtering, RPF, DPF, and BASE. Since placement of filters gives great impact on the filtering performance [8,69], we select two popular filter placement policies: random filter placement and priority filter placement. In random filter placement, filter nodes are chosen randomly; in priority filter placement, filter nodes are chosen according to priority, where a node that has many connections to other nodes has higher priority. Therefore, the highest degree node is the first to become a filter node. We simulate large and small asymmetric environments. The subgraph, sub_large, has 46.7% asymmetry and the subgraph, sub_small, has 12.2% asymmetry. We repeat each simulation ten times and compute an average as the result, to obtain more accurate results.

5.1.2. Filtering performance

Fig. 17 shows the dropping ratio of packets in the AS graph and its subgraph. These results from the two graphs show a similar pattern in dropping packets. Fig. 17 (left) shows the dropping ratio of attack packets in random filter placement and priority filter placement. It shows that filter placement policies strongly influence filtering performance. In 20% deployment, the random placement policy renders blocking less than 40% of attack packets, whereas the priority placement policy renders blocking over 80% of attack packets. In Fig. 17 (right), it is shown that false positives can happen under the deployment of less than 50% of nodes. Nonetheless, the ratios of dropping legitimate packets are very small, e.g., 1–2%, and false dropping does not happen under normal situations but only under an attack when the BASE filtering is invoked. This implies over 98% of legitimate packets can reach their destination even under an attack.

From the above results with the AS graph and its subgraph, we will use the subgraph in the rest of the simulation. Using a subgraph enables us to create many map instances, such as selecting 300 nodes with different asymmetric ratios while preserving the possible estimation in a larger map.

Fig. 18 shows the filtering performance for dropping attack packets using random filter placement and priority filter placement in a large asymmetry (46.7%). Among four mechanisms, DPF and BASE are more powerful than the others in dropping attack packets. Filtering performance increases when using priority filter placement, especially in DPF and BASE. Since the node with high priority could be a transit AS, the filtering performance of priority filter placement is higher than that of random filter placement. 30% transit ASes deploying the mechanism can filter more than 97% of the attack packets.

The following simulation measures filtering performance in terms of the benefit to the target. If the target is a filter node, it can guarantee its safety, in that it cannot receive attack packets. Fig. 19 shows the difference of filtering performance in DPF and BASE depending on whether or not the target node is a filter node. When the target node is a filter node or a non-filter node under partial deployment in DPF, the difference in the attack packet dropping ratio is small. However, the difference is larger in BASE than DPF. That is, The BASE mechanism gives greater benefit to early adopters than DPF.

Fig. 20 (left) shows the difference of filtering performance in DPF and BASE, when the node that a spoofed IP address belongs to is a filter node. If the node of the spoofed IP address is a filter node, the attack is prevented using its IP address. If packets originated from that node, the node would make and transfer its own marking value on the packets. The valid mark inscribed in a packet coming from the node can be verified at the target node. When the attack occurred in the other node, the node that the spoofed

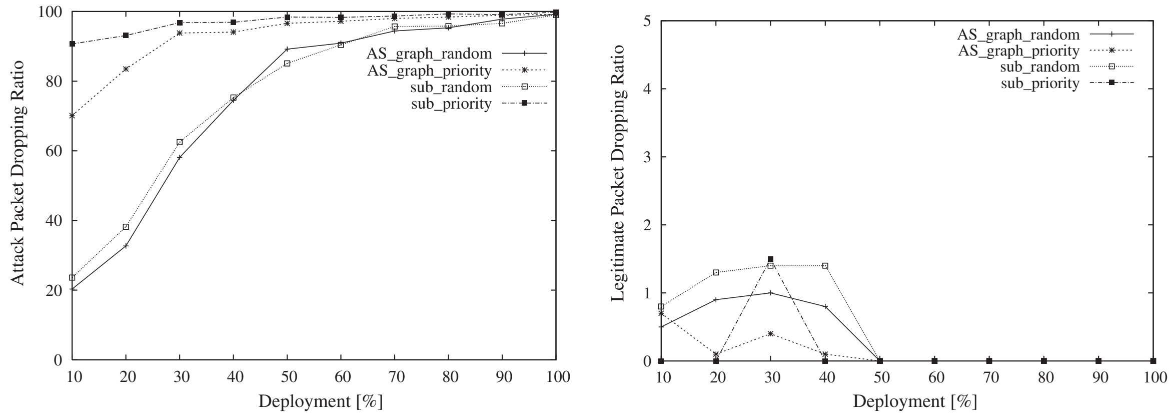


Fig. 17. Dropping ratio of packets using BASE filter in random and priority filter placement in the AS graph and its subgraph: (left) dropping ratio of attack packets (right) dropping ratio of legitimate packets.

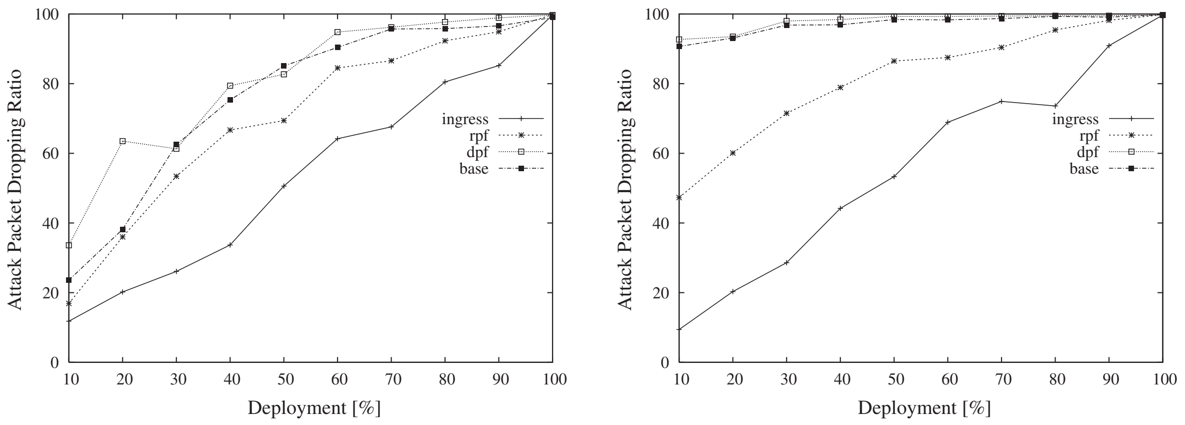


Fig. 18. Dropping ratio of attack packets using different anti-spoofing mechanisms in a subgraph: (left) random filter placement (right) priority filter placement.

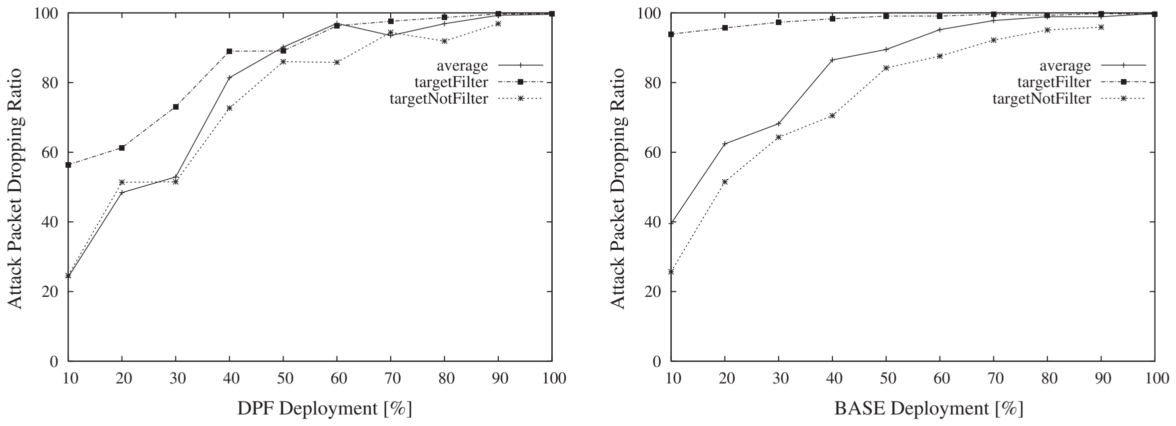


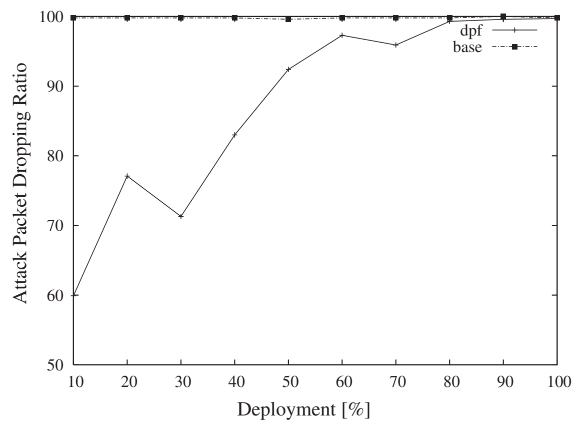
Fig. 19. Dropping ratio of attack packets if target is filter node or not: (left) DPF filter (right) BASE filter.

address belongs to can provide proof of innocence, showing that the packets did not originate from the node, using the marking value of the node. If the node of the spoofed address becomes a filter node, the target's attack packet dropping ratio would increase. Therefore, it provides incremental benefit.

Fig. 20 (right) shows false dropping in anti-spoofing mechanisms. The dropping ratio of legitimate packets in other mechanisms except RPF is close to 0, so legitimate packet dropping would not happen with other mechanisms. Small amount of false dropping occurs but it is negligible under an attack.

Fig. 21 shows the filtering performance in the environments of one mark and multiple marks. Filtering performance for dropping attack packets is a little bit better in the one mark environment than in the multiple mark environment. Conversely, filtering performance for not dropping legitimate packets is better in the multiple mark environment than in the one mark environment since the number of dropped legitimate packets decreases when using multiple marking values as opposed to one.

From the experiments, it is shown that BASE satisfies initial benefit, incremental benefit, and partial deployment. When the



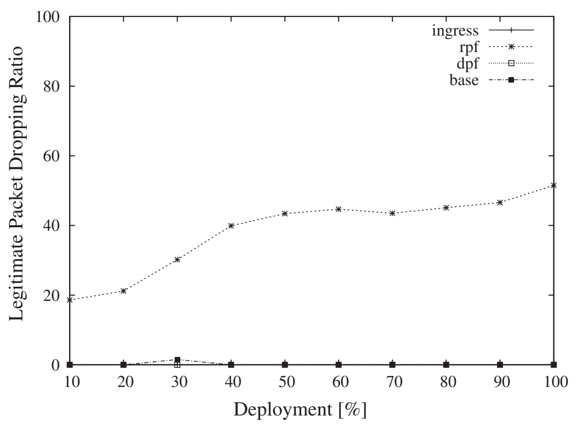
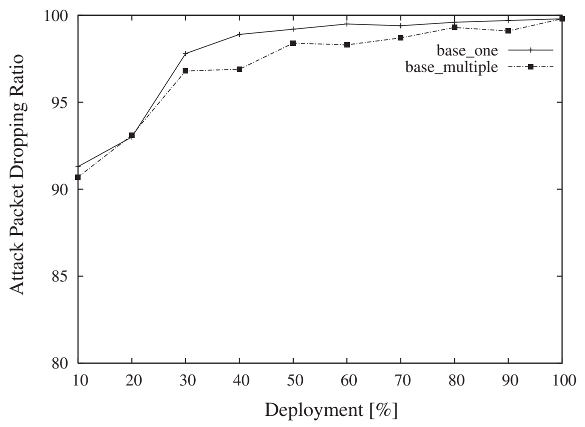


Fig. 20. (left) Dropping ratio of attack packets when both the node that a spoofed IP address belongs to and target node are filter nodes, (right) dropping ratio of legitimate packets using different anti-spoofing mechanisms.



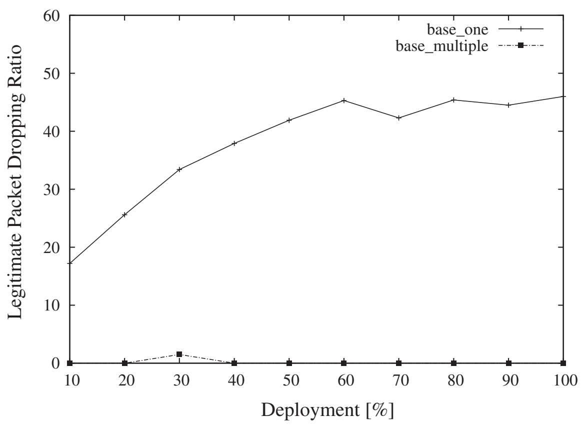


Fig. 21. Dropping ratio of packets according to the number of marks: (left) dropping ratio of attack packets (right) dropping ratio of legitimate packets.

transit ASes deploy BASE, the deployers get a much higher filtering effect than non-deployers, as shown in Figs. 19 and 20 (left). Therefore, BASE gives direct benefits to early adopters. As the filter nodes increase, filtering performance increases. Therefore, it also satisfies incremental benefit to subsequent adopters. Finally, almost all the attack packets would be dropped, even though only about 30% of transit ASes are deployed. Therefore, BASE is effective even when partially deployed.

5.1.3. Overhead analysis

In this section, we analyze the computation, communication, and memory overhead.

We first consider computation overhead. In the distribution phase, BASE requires a small computation to create marking values. The marking values can be computed even before they are distributed through BGP update messages. This process happens infrequently: only when a BGP path changes or a new BASE-enabled node is deployed. Also, if some nodes want to periodically update their key values, then the marking values also need to be updated.

For the communication overhead, BASE has a very small overhead during the distribution phase because the markings are piggybacked with BGP update messages. The invocation and revocation phases incur minimal messaging overhead, since only a single BGP update is used for initiating each start or stop signal. This is the minimum cost for saving a victim from overwhelming garbage traffic. All BGP message types use the basic packet header. BGP update messages comprise a BGP header and additional fields. The BGP basic packet header is 19 bytes and the additional fields used in BGP update messages are 10 bytes for invocation and revocation messages: Unfeasible Routes Length-2 bytes, Total Path Attribute Length-2 bytes, Path Attributes-6 bytes. Thus, invocation and revocation message size is 29 bytes each. The communication overhead of these messages is thus small. This additional traffic occurs only when the victim wants to use the BASE mechanism.

In terms of memory overhead for filtering, we need to consider the size of the *Filtering Table*. Each BASE node constructs its *Filtering Table* during the distribution phase. We have shown that BASE can block most spoofed packets when reaching its full deployment with accurate *Filtering Table*. Thus, building a precise *Filtering Table* is crucial to drop attack packets. In a naive approach, the *Filtering Table* would have one entry for each valid IP address. The naive approach would require a large amount of memory and high processing overhead. For a filter node with e neighbor nodes, the table of a BASE node needs to store e incoming marking values and one outgoing marking value. Each table entry requires 2 bytes to store the incoming and outgoing marking values.

$$((e+1) * 2) \text{ bytes}$$

Also, there are 2^{32} source address possible. Therefore, the required memory size of the *Filtering Table* becomes

$$((e+1) * 2) * 2^{32} \text{ bytes}$$

If a filter node has eight neighbors, it will use 72 Gbytes for its *Filtering Table*.

Aggregating IP addresses will reduce the memory for the *Filtering Table*. By clustering IP addresses into a prefix, we can build accurate, but smaller *Filtering Table*. In addition to using network addresses, we can merge multiple prefixes into a prefix cluster based on their marking values. Two prefixes with an identical marking value can be merged to one entry in the *Filtering Table*.

The determining factors for the *Filtering Table* size are the number of neighboring ASes, the number of advertised prefixes, and the deployment ratio of BASE. Let e denote the number of neighbors of a BASE filter node; p the number of prefixes advertised by BGP updates; and d the BASE deployment ratio. Then, the following formula shows the memory required for the *Filtering Table* in the BASE application (where 4 bytes are used to store the IP prefix):

$$(4 + 2 * (e + 1)) * p * d \text{ bytes}$$

The deployment ratio d is a decimal fraction, i.e., $0 \leqslant d \leqslant 1$, so that the memory demand is bounded by $O(e * p)$. The number of neighboring ASes e is 8 on average [70,71], and the number of prefixes p is about 300,000 [72]. While the size of the *Filtering Table* is bounded by the number of neighbors and prefixes, its size is tractable. For a BASE node with 8 neighbors and each neighbor announces a route for each prefix, 22 bytes are required to store multiple markings for a prefix. Thus, we would need approximately 6.6 Mbytes of memory with 300,000 prefixes fully deploying BASE, i.e., $d = 1.0$. This is even very smaller than the routing tables of current Internet routers.

In the packet marking and filtering phase, *Filtering Table* lookup should be done for each node to filter spoofed packets. *Filtering Table* lookup does not take more time than routing table lookup because the maximum number of entries in each node's *Filtering Table* is much smaller than ordinary routing table. From the above analysis, we find that BASE is cost-effective to defend spoofing attacks.

5.2. BASE function simulation

Now, we describe low level observations about how BASE works on SDN entities when BASE is deployed via the SDN framework. We performed a experiment with Mininet to show, (1) BASE implementation on the SDN architecture, (2) BASE functions interworking with multiple SDN entities, (3) BASE performance on SDN with a high level load.

5.2.1. Topology design

In order to show detail of BASE function in low level, we construct a topology for the following three key features of BASE.

- Asymmetric path support.
- Availability under partial deployment.
- Spoofed packet dropping.

With the consideration, we built a network topology as shown in Fig. 22. The network consists of 3 BASE-enabled ASes (black nodes) and a legacy AS (white node). The source node s has asymmetric paths to reach the target node t, and there are two different intermediate nodes v_1, v_2 on the paths respectively. When s sends packets to t, the packets are delivered via v_1 or v_2. Since one of the intermediate nodes v_2 is the legacy AS, t receives the packets with different marking value m_1 and m_2. If an attacker on v_1 or v_2 attempts to transfer packets with the spoofed IP address s, the packet will be blocked on t. Along with the simulation scenario, we believe that the network topology has a good structure to show the concept of BASE.

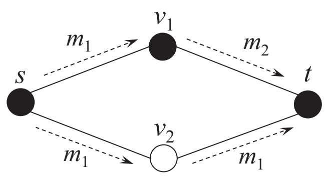


Fig. 22. Basic topology for simulating three key features of BASE.

5.2.2. Simulation environment

Since the idea of SDN has been suggested, many open source projects for SDN have also been developed. We implemented the simulation network mentioned above using Mininet [13], OpenvSwitch [14] and POX controller [15]. Mininet is a linux-based virtual system which provides a scalable platform for emulating SDN network via virtualization. Mininet supports generating virtual nodes and links, therefore it can be used for constructing SDN network easily without real devices. OpenvSwitch is a virtual switch which is designed to enable massive network automation via programmatic extension. POX is a Python-based SDN controller for developing network control software. The three softwares help us rapidly prototyping SDN network and implementing the BASE application. Latest version of Mininet 2.1.0[2] includes OpenvSwitch, POX, OpenFlow v1.0 and v1.3.

Fig. 23 represents an implementation of the simulation environment using Mininet, OpenvSwitch and POX. Our environment was built on the topology shown in Fig. 22 including 3 BASE-enabled ASes, AS_1, AS_2, AS_3, and a legacy AS AS_4. Each BASE-enabled AS has a BASE-enabled Controller C_1, C_2, C_3, and AS_4 is controlled by the basic controller of Mininet (not shown in this figure). Openflow switches S_{1-7} are working as the border router to establish links between each AS. Finally, we locate source host h_1 at AS_1, target host h_3 at AS_3, and attacker hosts h_2, h_4 at AS_2, AS_4 respectively.

While constructing the environment, we only modified the source code of POX controller to implement the BASE application. In order to support BASE functions, modifying the next two standard OpenFlow APIs are required.

- OFPT_PACKET_IN
- OFPT_FLOW_MOD

OFPT_PACKET_IN message is issued when the OpenFlow switch receives a new packet. More precisely, once the switch receives a new packet, the switch extracts flow information from the packet and compares with every flow entry on its Flow Table. If the packet is named as a new packet (OFPT_TABLE_MISS), the switch reports the arrival of the new packet to its controller. In the BASE system, OFPT_PACKET_IN message is used for extracting the packet's marking value to decide whether the packet contains a correct marking value or not.

OFPT_FLOW_MOD message is for notifying the corresponding actions to switches. If the packet reported to the controller has been decided as a spoofed packet, the controller sends the OFPT_FLOW_MOD message including action DROP to the switch. If the packet has a correct marking value, the controller sends the OFPT_FLOW_MOD message including action SET_FIELD and OUTPUT. Note that, the SET_FIELD action allows the switch to modify a certain field of the packet header, and the OUTPUT action points to the certain port to deliver the packet to the next node. BASE uses the SET_FIELD action to change the marking value to the next mark.

[2] Mininet 2.1.0 is available at http://github.com/mininet/mininet.

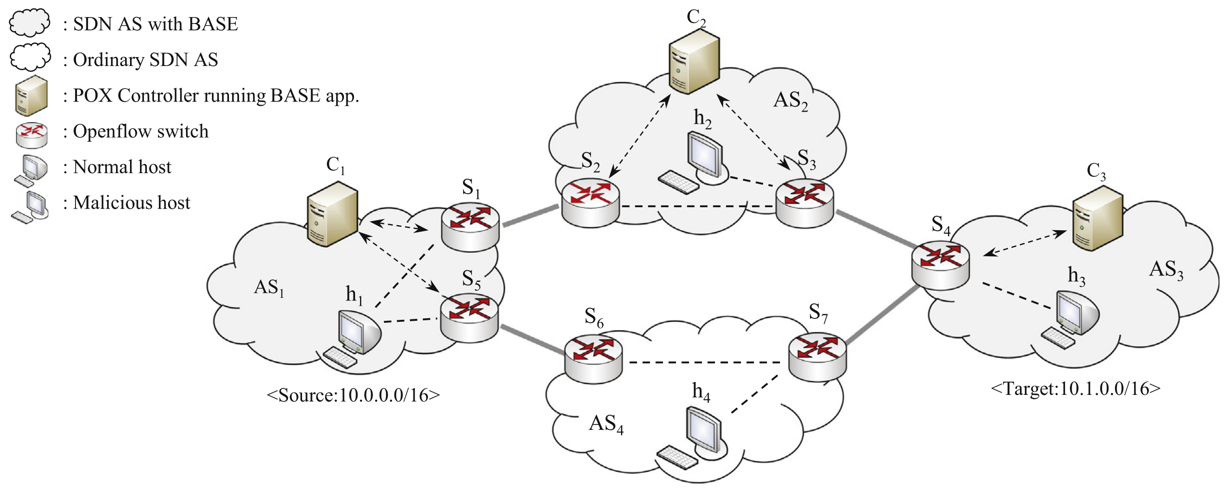


Fig. 23. An implementation of the simulation environment with Mininet, OpenvSwitch and POX controller.

Basically, the BASE system designed is to utilize the ID (Identification) field on the IP header for carrying marking values. However, the OpenFlow in the Mininet does not support the modification of the ID field, so we decide to carry the marking values on other available fields such as the TTL (Time-to-Live) or TOS (Type of Service) field. OpenFlow 1.3 supports the modification of IP fields through ofp_action_nw_ttl and ofp_action_nw_tos.

Using TTL or TOS field may bring unreliability of real world networking. More precisely, TTL field is originally designed to carry the life time of a packet, thus continuously changing the TTL filed by BASE routers in order to carry marking values may allow the eternity of a packet. Automatic decrement of TTL value by legacy routers also brings an incorrect calculation for checking marking values. TOS field is used to specify a datagram's priority and request for a router. Based on TOS values, a packet would be handled in a router, thus using TOS field to carry marking values may also bring unexpected packet handling by legacy routers.

Note that again, BASE is designed to carry the marking values in ID field. But in our implementation, despite of the problems, we decided to temporary use TTL and TOS field instead of ID field because current OpenFlow does not support ID field modification. Since there are many meaningful researches which utilize ID field, we aspire that OpenFlow support ID field modification in near future. Some drawbacks and solutions for using ID field will be discussed in Section 6.3.

5.2.3. BASE simulation

In the simulation, BASE was demonstrated with the key features mentioned above (see Section 5.2.1).

- **Asymmetric path support**: We located the source host h_1 and target host h_3 at AS_1, AS_3 respectively. AS_1 and AS_4 does not have a direct link but they have same neighbors AS_2 and AS_4. Therefore, h_1 has two routing paths to reach the target host h_3. If h_1 can transmit the packet through both paths, the result shows that BASE successfully support the asymmetric path transmission.
- **Availability under partial deployment**: We intentionally located BASE-disabled AS AS_4 as an intermediate AS on a path among the asymmetric paths. Therefore, a path from S_5 to S_4 connected via BASE-disabled switches S_6, S_7. If a packet is transferred through the BASE-disabled AS AS_4, this means that BASE works successfully under the partially deployed environment.
- **Spoofed packet filtering**: To test the filtering, we demonstrated a simple attack scenario. For the attack demonstration, we located attack sources h_2 and h_4 at the intermediate ASes AS_2 and AS_4 respectively, and transferred ICMP packets to target host h_3. Note that, IP addresses of the ICMP packets were changed to the IP address of h_1 using Python-SCAPY library. If the attack packets are filtered before reaching to h_3, it can be regarded that the BASE works effectively.

Table 1 shows that the simulation successfully demonstrates the filtering function running on the SDN environment. h_1 succeeded to transmit the ICMP packets to h_3 without any packet loss, whereas h_2 and h_4 failed to do the transmission due to the host unreachable which was caused by packet filtering at the intermediate switches S_3 and S_4. To do more analysis, we monitored packet transmission status on each switch using Wireshark [73]. According to the monitoring of packet transmission, the attack traffic originated from h_2 was discarded by S_3, and the attack traffic from h_4 was discarded by S_4.

Fig. 24 shows *Flow Table* entries for S_4. As we can see, S_4 contains two flow rules for transferring legitimate packets between h_1 and h_3, which have different marking values due to the different routing paths. According to the Flow rules, we can conclude that BASE successfully supports asymmetric path transmission even under partially deployed environment.

5.2.4. BASE performance on SDN

In this section, we estimate BASE performance which could be different from legacy network. The main difference between SDN and legacy network is coming from the existence of the SDN controller, but it does not effect the filtering performance of BASE. However, as we mentioned in Section 4.4, the SDN controller is able to become a vulnerability of the SDN architecture due to the centralized design. In order to measure BASE performance on SDN, especially in controller's point of view, we estimate the

Table 1
The simulation results for each host.

Host	Transmission rate (%)	Blocking switch	Reason
h_1	100	–	–
h_2	0	S_3	Host unreachable
h_4	0	S_4	Host unreachable

```
root@ubuntu:/home/ccs# ovs-ofctl dump-flows s4 | grep "nw_src₩|nw_dst₩|nw_tos₩|mod_nw_tos"
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=8.986s, table=0, n_packets=5, n_bytes=210, idle_timeout=10, idle_age=5, priority=65535,
icmp,in_port=1,vlan_tci=0x0000,dl_src=f2:15:83:50:9b:e5,dl_dst=3a:28:19:00:13:83,nw_src=10.0.0.2,nw_dst=10.0.0.1,
nw_tos=168,icmp_type=0,icmp_code=0 actions=mod_dl_dst:3a:28:19:00:13:83,output:3,mod_nw_tos:84
 cookie=0x0, duration=1.912s, table=0, n_packets=2, n_bytes=84, idle_timeout=10, idle_age=0, priority=65535,
icmp,in_port=2,vlan_tci=0x0000,dl_src=b2:a6:ac:0f:29:19,dl_dst=f2:15:83:50:9b:e5,nw_src=10.0.0.1,nw_dst=10.0.0.2,
nw_tos=120,icmp_type=8,icmp_code=0 actions=mod_dl_dst:f2:15:83:50:9b:e5,output:1,mod_nw_tos:168
 cookie=0x0, duration=1.91s, table=0, n_packets=2, n_bytes=84, idle_timeout=10, idle_age=0, priority=65535,
icmp,in_port=1,vlan_tci=0x0000,dl_src=f2:15:83:50:9b:e5,dl_dst=3a:28:19:00:13:83,nw_src=10.0.0.2,nw_dst=10.0.0.1,
nw_tos=168,icmp_type=0,icmp_code=0 actions=mod_dl_dst:b2:a6:ac:0f:29:19,output:2,mod_nw_tos:84
 cookie=0x0, duration=8.988s, table=0, n_packets=5, n_bytes=210, idle_timeout=10, idle_age=5, priority=65535,
icmp,in_port=3,vlan_tci=0x0000,dl_src=3a:28:19:00:13:83,dl_dst=00:00:00:00:00:04,nw_src=10.0.0.1,nw_dst=10.0.0.2,
nw_tos=248,icmp_type=8,icmp_code=0 actions=mod_dl_dst:f2:15:83:50:9b:e5,output:1,mod_nw_tos:168
root@ubuntu:/home/ccs#
```

Fig. 24. *Flow Table* entries for switch S_4.

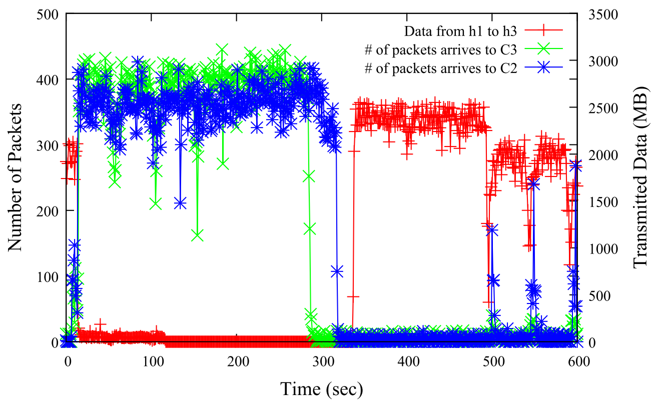


Fig. 25. BASE performance under DDoS attack. BASE recovers network connectivity by filtering attack packets.

service availability of legitimate users under high level load of DDoS attack along with the function of BASE controller.

To demonstrate DDoS attack, we located 10 attack hosts at intermediate ASes AS_2 and AS_4 respectively. The attack hosts generate ICMP flooding packets from $t_1 = 15$ to $t_2 = 600$ in packet interval 0.03 s. The ICMP packets have randomly selected/16 IP addresses with an incorrect marking value. At the same time, h_1 performs data transfers (2000 MB/s) to h_3 via iperf().

Fig. 25 presents the simulation results. Right after t_1, each controller C_2 and C_3 receive more than 300 PACKET_IN requests within every second. S_3 and S_4 generate the packets to request flow rules for the spoofed packets. Meanwhile, data transfer from h_1 to h_3 fails due to the denial-of-service on S_3 and S_4.

However, there are dramatic decreases of the number of packets on C_2 and C_3 at $t = 288$, $t = 320$ respectively, and recovery of the legitimate data transfer at $t = 338$. The recovery of network connectivity arises after finishing flow rule updates between switches and controllers for every/16 IP addresses. Recall that, the attack hosts used/16 IP addresses for IP spoofing. According to a simple calculation, flow rule updates would take approximately 220 s while 300 packets arrive in a second and the range of/16 IP address is 65,536.

Consequently, BASE shows that it not only works well on the SDN architecture in a high level load, but also recovers network connectivity automatically by filtering the spoofed packets. The service availability could be improved by using the high performance hardware for the controller and switch or adopting the decentralized controller model.

6. Discussion

6.1. Adopter's benefit

Ingress filtering and DPF are more powerful when deployed near the attacking location, but less effective near the victim. Therefore, ISPs who can become potential victims do not feel motivated for adopting ingress filtering and DPF. Only Pi gives an obvious benefit to a victim for defending against spoofing attacks. Nevertheless, Pi still has significant weaknesses—the full benefit of Pi occurs only after large-scale deployment. Therefore, Pi cannot be an immediate solution for a victim of spoofing attacks. BASE is the only solution which gives direct benefit to an adopter and can be a viable solution to defend against spoofing attacks.

When the attack is against other nodes, the deployer can provide proof of innocence, by showing that the marking value propagated to the victim did not originate from the deployed node. We can verify innocence, since we use a one-way hash chain to make marking values. The one-way hash function, with the addition of a secret key, makes it difficult to turn the calculated marking value back into the source IP address. Thus, the only way to verify the calculated marking value is to have the secret key. The secret key of a node can be secured against the other nodes. Therefore, the deployed node can insist on its innocence by showing that it did not originate the incorrect marking value, since a calculated marking value on a one-way hash chain shows unique characteristic based on the nodes traversed on the path the packets passed through.

6.2. BASE protocol design

BGP is a flexible protocol, in that a variety of options are available to network engineers. A BASE filter node communicates with other BASE nodes using BGP update messages. Using them, one BGP update message invokes distributed filters propagating through all legacy BGP routers,[3] reaching all BASE nodes. This mechanism works transparently with the legacy BGP speakers by using *optional transitive attributes* [12], in which the information stored in transitive attributes is passed on to other BGP speakers, even if it is not understood by legacy routers. However, it is possible that an AS's routing policy may prevent an update from propagating to a neighboring AS, even though it sends packets to that AS. The effect of BGP routing policy decisions requires further study.

The path attribute of each update message is a triple of Attribute Type, Attribute Length and Attribute Value [12]. The

[3] Recall that the SDN ASes are connected with the legacy IP network for seamless interworking.

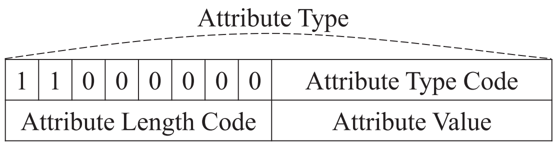


Fig. 26. The Path Attributes field using optional transitive attributes of the BGP update message.

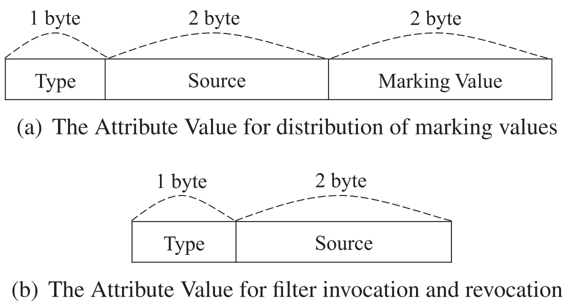

1 byte
2 byte
2 byte
Type
Source
Marking Value
(a) The Attribute Value for distribution of marking values
1 byte
2 byte
Type
Source
(b) The Attribute Value for filter invocation and revocation

Fig. 27. The Attribute Value of the Path Attributes field of BGP update messages: (a) for distribution of marking values, (b) for BASE filter invocation and revocation.

high-order bit (bit 0) of Attribute Type is the optional bit and the second high-order bit (bit 1) of Attribute Type is the transitive bit. Accordingly, we should set these two bits to 1 shown in Fig. 26. The Attribute Type Code octet should contain the attribute type code that is not currently defined. We can create a new BGP attribute type code and should send it to the IANA [74]. For example, we can use value 32 for the attribute type code of BASE. As Fig. 27 shows, we can construct the Attribute Value field: 1 byte for Type, 2 bytes for Source, and 2 bytes for Marking Value.

Fig. 27(a) is a format of the attribute value for distribution of marking values. Fig. 27(b) is a format of an attribute value for BASE filter invocation and revocation. The Type field defines whether the BGP update message is for distribution of marking values (if set to 1) or invocation (if set to 2) or revocation (if set to 3). The Source field is for the source's AS number of the BGP message. The Marking Value field gets a 16-bit marking value for distributing it to the next BGP filter nodes.

6.3. IP packet fragmentation and reassembly

A marking value is propagated through the network using the IP identification field, so BASE may cause packets to loose the information that are needed for a packet reassembly. Thus, we inquire into the compatibility with marking and IP fragmentation. If a link of a router has an MTU that is smaller than the length of the packet, the router disassembles the packets into two or more smaller fragments. These fragments are sent through the routing path and reassembled at the destination host. To perform the reassembly, the identification, flag, and fragmentation offset fields are used in the IP header. When the destination node receives packets, it examines the identification field to determine whether they are originated from a larger packet. A 3-bit flag field is used to control or identify fragments. The first flag field is reserved and must be zero. The second flag field is the Do not Fragment (DF) field to prohibit fragmentation. The third flag More Fragment (MF) field indicates if the packet contains additional fragments. A non-fragmented packet is considered as the last fragment. The fragmentation offset field is used to direct the reassembly of a fragmented packet.

However, there is a small but non-zero portion to consider under fragmentation and reassembly process: fragments originating from that same IP packet should have the same marking value, and fragments originating from different IP packets should have different marking values to distinguish original packets. If the fragments pass through different routing paths to the destination, they may have different marking values; reassembly at the destination may fail.

In order to prevent this problem, one solution is to simply not mark fragments [44]. We can know whether a packet is a fragment by investigating MF flag and fragmentation offset. As shown in Fig. 29, if the MF flag bit is set to zero and offset value is zero, the packet is a non-fragmented packet. By not marking fragments, we can avoid fragmentation level asymmetry. Second, fragments originated from different IP packets should have different marking values to reassemble the packets correctly. Using our marking scheme, the packets from same source node have all same marking value.

To solve this problem, we can set the DF flag on every marked packet. By preventing packet fragmentation, we do not need to consider reassembly at the destination. Thus, BASE inscribes a marking value in a packet when the DF field is set to 1, MF field is set to 0, and offset field is set to 0.

One can consider the situation that an attacker sends very large spoofed packets, then the spoofed packets will be fragmented and pass through BASE routers. In that case, a victim can decide to drop fragmented packets, instead of reassembling them, if the victim cannot stop spoofed packets.

6.4. Filter invocation

BASE has four phases of working flow including filter invocation, revocation and filtering. The spoofed packet filtering only functions under the occurrence of a spoofing attack by the filter invocation. A BASE controller in a victim network issues the filter invocation message via BGP updates to launch packet filtering if a spoofing attack is detected. The design makes BASE not only keep the low network overhead but also hard to guess the valid marking values by adversaries as mentioned above Section 4.3. Despite of the benefits, now the BASE system brings the following simple question. "Who's gonna be in charge of the attack detection and filter invocation?". In this section, we discuss about the question with several possible scenarios.

Before answering the question, we first address the entities of BASE who can participate with the detection and filter invocation. In terms of SDN, there are three main entities in a network, which are a controller, a switch and an end host. Among them, the switch is belonging to the data plane as a simple packet forwarding device, so that the switch lacks of computational potentiality for processing the detection and filter invocation. We consider only the controller and end host as the one who can participate with this discussion.

One possible scenario is the end host participation for the detection and filter invocation. Fig. 28(a) represents the scenario. In this scenario, we assume that, (1) the end host is capable of detecting spoofing attacks, and (2) the end host communicates with a BASE application running on it's controller via secure channel. If the end host recognizes a spoofing attack targeting himself, the host requests the enabling of packet filtering to the BASE controller. Once the BASE controller receives the request from any of underlying end hosts, it sends a filter invocation message to other BASE controllers to start packet filtering. The host participation scenario is constructed on the assumption that every end host must contain extra applications to detect the spoofing attack and communicate with the BASE application, thus it may bring another deployment problem.

Another possible scenario is that the BASE controller has a responsibility for the attack detection and filter invocation as

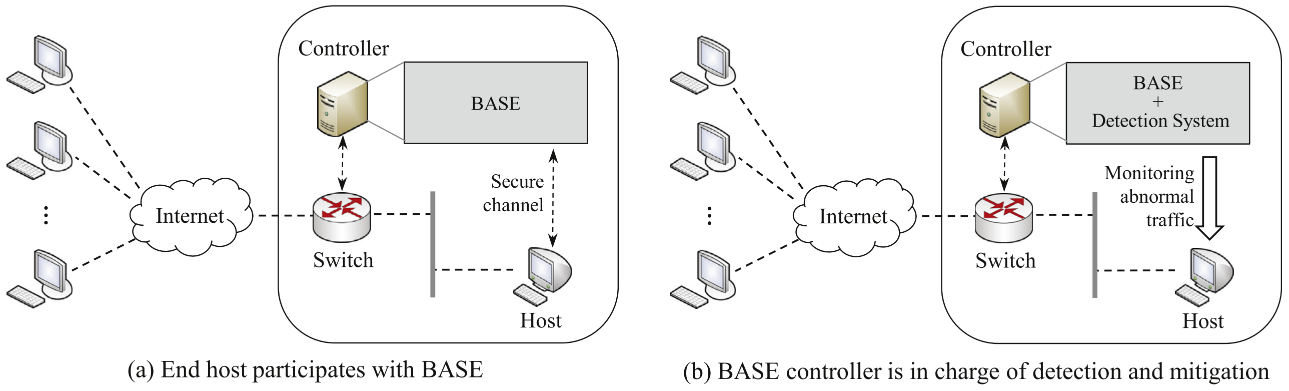

Controller
BASE
Secure channel
Internet
Switch
Host
(a) End host participates with BASE
Controller
BASE + Detection System
Monitoring abnormal traffic
Internet
Switch
Host
(b) BASE controller is in charge of detection and mitigation

Fig. 28. Filter invocation scenario. (a) Represents that the host issues the filter invocation via secure channel. (b) Shows that the controller monitors the entire network to detect IP spoofing attack and triggers the filter invocation by himself.

Fragment	Identification	Flag			Offset
		0	DF	MF	
1^{st} fragment	ID_1	0	0	1	0
2^{nd} fragment	ID_1	0	0	1	m_1
nth fragment	ID_1	0	0	0	m_{n-1}
non-fragment	-	0	0	0	0

Fig. 29. The three fields (Identification, Flag, and Offset) that have to do with IP fragmentation: non-fragmented packets are MF = 0 ⋂ offset = 0.

shown in Fig. 28(b). In accordance with the second scenario, the BASE controller has an ability to detect spoofing attack which targets to one of the underlying hosts by monitoring entire network under his control. If the BASE controller determines that there is a malicious attempt of spoofed packets, the controller directly transfers the filter invocation message via BGP. Since there is no specific role of end hosts for this work flow, there are no extra deployment issues. Moreover, in terms of attack detection, monitoring entire network is more accurate rather than monitoring a single host. We believe that this structure is more practical and reliable to implement the BASE system into real-world network.

IP spoofing detection can be done by applying one of the well-defined spoofing detection method [75–78]. Developing a spoofing detection method is out of scope in this paper.

6.5. BASE limitations

A few limitations arise with BGP as a broadcasting mechanism. Routing asymmetry in a certain BASE configuration and routing policies that do not forward update messages provide opportunities to drop legitimate traffic. However, legitimate packet dropping according to AS's routing policy and routing asymmetry occurs only during an attack because BASE is turned on only when the victim wants to filter attack packets. Without any defense, almost all the legitimate packets destined to the victim would be dropped during an attack. Therefore, dropping a small amount of legitimate packets during an attack is not a major issue.

Policy issues remain more problematic with smaller ISPs or stub ASes as opposed to transit ASes of major tier-1 or tier-2 ISPs. AS-level asymmetry will not happen between neighboring ASes, but between ASes in a long path. Asymmetry depends on the distance between ASes, implying that BASE can protect attacks from near ASes but may provide misleading information from remote ASes. Nonetheless, attacks from distant ASes have greater opportunity to encounter larger number of BASE filters than attacks from near ones.

We could design our own protocol instead of utilizing BGP to spread BASE information. A dedicated protocol would work in a manner similar to the BGP-enabled BASE scheme, while retaining favorable properties for incremental deployment. This would resolve asymmetry issues because we can store every possible mark, eliminating the issue of false-positive. However, distributing marking values remains a significant problem without direct integration with the routing protocol because of the difficulty in maintaining up-to-date marking information.

IPsec is a protocol suite for securing Internet Protocol and the entire IP packet is encrypted in the tunnel mode of IPsec. Thus, the new IP packet with a new IP header can cause a compatibility problem when encrypting IP headers after passing a BASE router. Thus, we need to ensure IP header encryption is done before passing through BASE routers so that new IP headers can be compatible with the packet marking and filtering of the BASE mechanism.

6.6. Compromising BASE routers

One can consider compromised BASE routers, which can be used for passing attack packets but dropping legitimate packets. A compromised router can control all packets it forwards, while associating with legitimate BASE routers. In this case, the compromised BASE router can computes correct markings by sniffing packets, and it can even compute the key independently.

In practice, however, installing compromised routers in transit ASes is infeasible because the network is physically difficult to access and always being monitored [79]. Even compromising specific routers in stub ASes needs a high level of effort such as using rental services in Internet black markets [80]. Furthermore, compromising routers is not only the issue in BASE, but also in any network security approaches. Hence, the assumption that BASE architecture is founded on BGP peer trust is not a tough assumption since AS networks are highly managed by ISPs.

7. Related work

Researchers have followed two main directions in the investigation of techniques to mitigate spoofed source IP addresses: IP traceback and detection of spoofed packets. The goal of IP traceback is to find the true origin(s) of attack packets. One of IP traceback mechanisms is packet marking which can be either deterministic [81,52,54] or probabilistic [41,44,47,53]. However, IP traceback has several drawbacks. For example, spoofed packets can destroy a victim's network before being reactively curtailed. As well, the uncertainty of IP traceback amplifies under distributed attacks, which eventually makes IP traceback useless under massive DDoS attacks.

In this paper, we discuss the second direction: how to detect spoofed packets. Once given an ability to discriminate between attack packets and legitimate packets, it is a simple task to filter attack packets before they reach a victim.

Ingress filtering [82,60] has been proposed for dropping packets with invalid source addresses before the packets leave their local networks. However, the usefulness of ingress filtering depends on the deployment, providing little incentive to early adopters. Moreover, the incentives for deployment of ingress filtering are structured in an awkward fashion. Consider an ISP that deploys ingress filtering—this does not benefit the customers of the ISP directly because the ISP filters outgoing spoofed packets not incoming ones. Therefore ingress filtering protects other ISPs who may not adopt ingress filtering. Thus, ingress filtering does not provide significant benefit for early adopters, except when laws make the sender of malicious packets liable for the damage caused (as is the case in Italy), because the customers will not be able to use IP spoofing to attack a victim.

Reverse path forwarding (RPF) is an extension of ingress filtering. It uses IP routing tables for dropping spoofed packets. RPF has become an optional function of mainstream routers in order to mitigate the problems caused by IP spoofing [83]. RPF-enabled routers forward only packets that have valid source addresses consistent with the IP routing table. There is one topological restriction; RPF can only be used for symmetric routing environments. Moreover, RPF does not provide sufficient benefit to adopters, as is the case for ingress filtering. No matter whether a victim is a deployer or not, dropping spoofed packets toward the victim is done by other RPF deployers.

'Route-based distributed packet filtering (DPF) has been proposed for filtering spoofed packets using routing information [8]. DPF determines whether a packet travels an unexpected route from its specified source and destination addresses, and if so, discards the packet. DPF can be viewed as a generalized address-based filtering scheme that eliminates the limitations of ingress filtering and RPF. The DPF filter can be located in transit ASes; thus, only a part of the Internet needs to be used for filtering a significant fraction of spoofed packets. But DPF does not provide direct incentives to deployers—everyone shares the benefits.

Inter-domain packet filter (IDPF) [39] has extended the concept of DPF [8]. DPF requires each node to have knowledge of global routing decisions, but such information is not available in the current Internet. To overcome this limitation, IDPF uses the information implicit in BGP updates exchanged between an AS and its immediate neighbors, so that global information is not required. However, the benefit from the deployment of IDPF is shared among all other nodes, as in DPF.

Path identification (Pi) is a reactive filtering scheme based on packet marking [48]. In Pi, each packet in a path has the same identifier, which can be used for filtering attack packets. Thus, Pi is beneficial to adopters who can use the Pi-filter to protect their network. However, Pi gives little benefit for early adopters, because it becomes effective after significant deployment.

Hop-count filtering (HCF) is another technique for spoofing attacks [75,76]. The idea behind HCF stems from the fact that packets coming from the same location travel the same path to the destination. Thus, time-to-live (TTL) values in IP headers can be used for classifying the attack packets. The TTL is only an estimation of hop count, so HCF provides higher false-positive results than Pi [84].

The spoofing prevention method (SPM) enables routers closer to the packet destination to verify the authenticity of the packet source using a unique temporal key [85]. For each packet arriving at a destination, routers in the destination network verify the key on the packet whether it is equivalent to the corresponding key of that source network. Hence the packets arriving at their destination network with an invalid key are considered as spoofed packets. The mechanism to be used in SPM is shared only by the participants of SPM. Therefore, SPM gives benefits to the ASes implementing it, and not the other non-deployed ASes. On the other hand, SPM does not diminish network congestion, since attacking packets are recognized only at their destination.

Passport system also leverages BGP to disseminate cryptographic information to prevent source IP address spoofing [86]. Passport makes many different design decisions, and thus represents another valuable point in the design space of IP spoofing prevention. Passport cannot provide protection against spoofing if the origin AS does not deploy it. Therefore, attackers in non-deploying ASes can spoof any IP address of any other non-deploying AS. In contrast, BASE offers spoofing protection even for IP blocks within non-deploying ASes. Passport has weak initial benefits: for the first deploying ASes there is very limited protection. Moreover, the incremental benefits are relatively weak because IP spoofing is still possible.

Andersen et al. propose the Accountable Internet Protocol (AIP) where the source address of a domain or a host is the cryptographic hash of the public key, as a consequence, network entities can validate the correctness of the address by running a challenge-response protocol where a nonce is sent to the domain or host requesting a digital signature of the nonce with the private key [87]. While this approach prevents *impersonation*, i.e., preventing address theft, AIP still permits the creation of new addresses, as simply fresh public keys can be created. Moreover, AIP requires broad adoption to be effective.

None of these mechanisms provide sufficient incremental advantage to deployers. Thus, we study an anti-spoofing mechanism which fulfills the three properties for incremental deployment.

8. Conclusions

The BASE mechanism is suggested to fulfill the incremental deployment properties that are essential for adoption in current Internet environments. Along with distributed filtering, cryptographic packet marking, and on-demand filtering for the destination addresses of the victim's network, the protective power is enhanced as BASE filters are distributed gradually. The BASE mechanism offers strong incremental deployment benefits over existing solutions because.

BASE is a promising approach for overcoming the barriers to wide-spread adoption that have prevented other mechanisms from taking hold. This is due to its ability to prevent spoofing of a large percentage of the IP address space when it has only been deployed to a comparatively small percentage of that space. However, some challenges still must be surmounted. AS's routing policies may prevent the BGP update messages from propagating to a neighboring AS, and also malicious BASE speakers at compromised routers can pass attack packets and drop legitimate packets. The effect of real-world routing policies on distribution of BASE control data needs further examination. Despite this, BASE offers a promising new direction in IP spoofing prevention.

Acknowledgments

We are grateful to Damon Smith and Jihoon Son for conducting the simulations and Geoffrey Hasker for valuable discussions. A preliminary version of this paper appeared in ACM Symp. on Information, Computer and Communication Security 2007 [88].

References

[1] R. Beverly, A. Berger, Y. Hyun, K. Claffy, Understanding the efficacy of deployed internet source address validation filtering, in: Proceedings of ACM SIGCOMM IMC, 2009.
[2] T. Ehrenkranz, J. Li, On the state of IP spoofing defense, ACM Trans. Internet Technol. 9 (2) (2009) 6.
[3] J. Mirkovic, E. Kissel, Comparative evaluation of spoofing defenses, IEEE Trans. Defendable Secure Comput. 8 (2) (2011) 218–232.
[4] CERT, TCP SYN Flooding and IP Spoofing Attacks, Advisory CA-96.21, September 1996.
[5] D. Moore, C. Shannon, D.J. Brown, G.M. Voelker, S. Savage, Inferring internet denial-of-service activity, ACM Trans. Comput. Syst. (TOCS). 24 (2) (2006) 115–139.
[6] ICS-CERT, NTP Reflection Attack, Advisory (ICSA-14-051-04), February 2014. <https://ics-cert.us-cert.gov/advisories/ICSA-14-051-04>.
[7] A.N.A. Group, MIT ANA Spoofer Project, October 2011. <http://spoofer.csail.mit.edu>.
[8] K. Park, H. Lee, On the effectiveness of route-based packet filtering for distributed DoS attack prevention in power-law Internets, in: Proceedings of ACM SIGCOMM, 2001.
[9] Y. Gilad, A. Herzberg, LOT: a defense against IP spoofing and flooding attacks, ACM Trans. Inform. Syst. Secur. 15 (2) (2012) 6.
[10] Z. Qian, Z.M. Mao, Off-path TCP sequence number inference attack – how firewall middleboxes reduce security, in: Proceedings of IEEE Security and Privacy, 2012.
[11] G.A. Moore, Crossing the Chasm: Marketing and Selling High-Tech Products to Mainstream Customers, HarperCollins Publishers, 1995.
[12] Y. Rekhter, T. Li, S. Hares, A Border Gateway Protocol 4 (BGP-4), RFC 4271, January 2006.
[13] B. Lantz, B. Heller, N. McKeown, A network in a laptop: rapid prototyping for software-defined networks, in: Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks, ACM, 2010, pp. 19.
[14] OpenvSwitch, An Open Virtual Switch. <http://openvswitch.org/>.
[15] POX, Python-based NOX. <http://http://www.noxrepo.org/pox/about-pox/>.
[16] O.M.E. Committee et al., Software-defined Networking: The New Norm for Networks, ONF White Paper, Open Networking Foundation, Palo Alto, US.
[17] S. Sezer, S. Scott-Hayward, P.K. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller, N. Rao, Are we ready for SDN? Implementation challenges for software-defined networks, IEEE Commun. Mag. 51(7).
[18] J.E. van der Merwe, S. Rooney, L. Leslie, S. Crosby, The tempest-a practical framework for network programmability, IEEE Netw. 12 (3) (1998) 20–28.
[19] N. Feamster, H. Balakrishnan, J. Rexford, A. Shaikh, J. Van Der Merwe, The case for separating routing from routers, in: Proceedings of the ACM SIGCOMM Workshop on Future Directions in Network Architecture, ACM, 2004, pp. 5–12.
[20] H. Yan, D.A. Maltz. T.S. Eugene Ng, H. Gogineni, H. Zhang, Z. Cai. Tesseract: A 4D Network Control Plane. NSDI, vol. 7, 2007, pp. 27.
[21] M. Casado, M.J. Freedman, J. Pettit, J. Luo, N. McKeown, S. Shenker, Ethane: taking control of the enterprise, ACM SIGCOMM Comput. Commun. Rev. 37 (4) (2007) 1–12.
[22] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, J. Turner, OpenFlow: enabling innovation in campus networks, ACM SIGCOMM Comput. Commun. Rev. 38 (2) (2008) 69–74.
[23] H. Kim, N. Feamster, Improving network management with software defined networking, IEEE Commun. Mag. 51 (2) (2013) 114–119.
[24] G. Gibb, H. Zeng, N. McKeown, Outsourcing network functionality, in: Proceedings of the First Workshop on Hot Topics in Software Defined Networks, ACM, 2012, pp. 73–78.
[25] S. Scott-Hayward, G. O'Callaghan, S. Sezer, SDN security: a survey, in: IEEE SDN for Future Networks and Services (SDN4FNS), 2013, pp. 1–7. http://dx.doi.org/10.1109/SDN4FNS.2013.6702553.
[26] R. Braga, E. Mota, A. Passito, Lightweight DDoS flooding attack setection using NOX/OpenFlow, in: IEEE 35th Conference on Local Computer Networks (LCN), IEEE, 2010, pp. 408–415.
[27] S. Lim, J. Ha, H. Kim, Y. Kim, S. Yang, A SDN-oriented DDoS blocking scheme for botnet-based attacks, in: Sixth International Conf on Ubiquitous and Future Networks (ICUFN), IEEE, 2014, pp. 63–68.
[28] M. Shtern, R. Sandel, M. Litoiu, C. Bachalo, V. Theodorou, Towards mitigation of low and slow application DDoS attacks, in: IEEE International Conference on Cloud Engineering (IC2E), IEEE, 2014, pp. 604–609.
[29] S.A. Mehdi, J. Khalid, S.A. Khayam, Revisiting traffic anomaly detection using software defined networking, in: Recent Advances in Intrusion Detection, Springer, 2011, pp. 161–180.
[30] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, S. Shenker, NOX: towards an operating system for networks, ACM SIGCOMM Comput. Commun. Rev. 38 (3) (2008) 105–110.
[31] J.H. Jafarian, E. Al-Shaer, Q. Duan, Openflow random host mutation: transparent moving target defense using software defined networking, in: Proceedings of the First Workshop on Hot Topics in Software Defined Networks, ACM, 2012, pp. 127–132.
[32] S. Shin, G. Gu, Cloudwatcher: network security monitoring using OpenFlow in dynamic cloud networks (or: How to provide security monitoring as a service in clouds?), in: 20th IEEE International Conference on Network Protocols (ICNP), IEEE, 2012, pp. 1–6.
[33] R. Skowyra, S. Bahargam, A. Bestavros, Software-defined IDS for securing embedded mobile devices, in: High Performance Extreme Computing Conference (HPEC), IEEE, 2013, pp. 1–7.
[34] H. Hu, W. Han, G.-J. Ahn, Z. Zhao, Flowguard: building robust firewalls for software-defined networks, in: Proceedings of the Third Workshop on Hot Topics in Software Defined Networking, ACM, 2014, pp. 97–102.
[35] Z.A. Qazi, C.-C. Tu, L. Chiang, R. Miao, V. Sekar, M. Yu, Simple-fying middlebox policy enforcement using SDN, in: Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM, 2013, pp. 27–38.
[36] B. Zhang, J. Bi, T. Feng, P. Xiao, D. Zhou, Performing software defined route-based IP spoofing filtering with SEFA, in: the 23rd IEEE International Conference on Computer Communications and Networks (ICCCN14), IEEE, 2014.
[37] G. Yao, J. Bi, P. Xiao, Vase: filtering IP spoofing traffic with agility, Comput. Netw. 57 (1) (2013) 243–257.
[38] T. Feng, J. Bi, H. Hu, G. Yao, P. Xiao, InSAVO: intra-AS IP source address validation solution with openrouter, in: IEEE International Conference on Computer communications (INFOCOM12) Demo, IEEE, 2012.
[39] Z. Duan, X. Yuan, J. Chandrashekar, Constructing inter-domain packet filters to control IP spoofing based on BGP updates, in: Proceedings of IEEE Infocom, 2006.
[40] G. Yao, J. Bi, P. Xiao, Source address validation solution with OpenFlow/NOX architecture, in: 19th IEEE International Conference on Network Protocols (ICNP), IEEE, 2011, pp. 7–12.
[41] K. Park, H. Lee, On the effectiveness of probabilistic packet marking for IP traceback under denial of service attack, in: Proceedings of IEEE Infocomm, 2001.
[42] O. Goldreich, S. Goldwasser, S. Micali, How to construct random functions, J. ACM. (JACM) 33 (4) (1986) 792–807.
[43] J. Postel, Internet Protocol, RFC 791, September 1981.
[44] S. Savage, D. Wetherall, A. Karlin, T. Anderson, Practical network support for IP traceback, in: Proceedings of ACM SIGCOMM, 2000.
[45] H. Aljifri, M. Smets, A.P. Pons, IP traceback using header compression, Comput. Secur.
[46] D. Dean, M. Franklin, A. Stubblefield, An algebraic approach to IP traceback, ACM Trans. Inform. Syst. Secur.
[47] D. Song, A. Perrig, Advanced and authenticated marking schemes for IP traceback, in: Proceedings of IEEE Infocomm, 2001.
[48] A. Yaar, A. Perrig, D. Song, Pi: a path identification mechanism to defend against DDoS attacks, in: Proceedings of IEEE Symposium on Security and Privacy, 2003.
[49] M. Sung, J. Xu, IP traceback-based intelligent packet filtering: a novel technique for defending against Internet DDoS attacks, IEEE Trans. Parall. Distrib. Syst. 14 (9) (2003) 861–872.
[50] D. Seo, H. Lee, A. Perrig, PFS: probabilistic filter scheduling against distributed denial-of-service attacks, in: The 36th IEEE Conference on Local Computer Networks (LCN), 2011 (Best paper award), pp. 9–17.
[51] C. Shannon, D. Moore, K.C. Claffy, Beyond folklore: observations on fragmented traffic, IEEE/ACM Trans. Netw. (ToN) 10 (6) (2002) 709–720.
[52] A. Belenky, N. Ansari, On deterministic packet marking, Comput. Netw. 51 (10) (2007) 2677–2700.
[53] M.T. Goodrich, Probabilistic packet marking for large-scale IP traceback, IEEE/ACM Trans. Netw. (TON). 16 (1) (2008) 15–24.
[54] Y. Xiang, W. Zhou, M. Guo, Flexible deterministic packet marking: an IP traceback system to find the real source of attacks, IEEE Trans. Parall. Distrib. Syst. 20 (4) (2009) 567–580.
[55] P. Lin, J. Hart, U. Krishnaswamy, T. Murakami, M. Kobayashi, A. Al-Shabibi, K.-C. Wang, J. Bi, Seamless interworking of SDN and IP, in: Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM, ACM, 2013, pp. 475–476.
[56] O.S. Specification, Version 1.4.0, Open Networking Foundation.
[57] V. Paxson, End-to-end routing behavior in the internet, in: Proceedings of ACM SIGCOMM, 1996.
[58] J. Li, J. Mirkovic, M. Wang, P. Reiher, L. Zhang, Save: source address validity enforcement protocol, in: Proceedings of IEEE INFOCOM, 2002.
[59] Y. He, M. Faloutsos, S. Krishnamurthy, B. Huffaker, On routing asymmetry in the Internet, in: Proceedings of IEEE Globecom, 2005.
[60] P. Ferguson, D. Senie, Network Ingress Filtering: Defeating Denial of Service Attacks which Employ IP Source Address Spoofing, RFC 2827, May 2000.
[61] D. Kreutz, F. Ramos, P. Verissimo, Towards secure and dependable software-defined networks, in: Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, ACM, 2013, pp. 55–60.
[62] K. Benton, L.J. Camp, C. Small, OpenFlow vulnerability assessment, in: Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, ACM, 2013, pp. 151–152.
[63] T. Koponen, M. Casado, N. Gude, J. Stribling, L. Poutievski, M. Zhu, R. Ramanathan, Y. Iwata, H. Inoue, T. Hama, et al., Onix: a distributed control platform for large-scale production networks, in: OSDI, vol. 10, 2010, pp. 1–6.
[64] A. Dixit, F. Hao, S. Mukherjee, T. Lakshman, R. Kompella, Towards an elastic distributed SDN controller, in: Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, ACM, 2013, pp. 7–12.
[65] D. Levin, A. Wundsam, B. Heller, N. Handigol, A. Feldmann, Logically centralized? State distribution trade-offs in software defined networks, in: Proceedings of the First Workshop on Hot Topics in Software Defined Networks, ACM, 2012, pp. 1–6.

[66] S. Shin, V. Yegneswaran, P. Porras, G. Gu, Avant-guard: scalable and vigilant switch flow management in software-defined networks, in: Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security, ACM, 2013, pp. 413–424.

[67] M. Yu, J. Rexford, M.J. Freedman, J. Wang, Scalable flow-based networking with DIFANE, ACM SIGCOMM Comput. Commun. Rev. 40 (4) (2010) 351–362.

[68] D. Meyer, University of Oregon Route Views Archive Project, April 2006. <http://archive.routeviews.org>.

[69] A.W. Jackson, W.C. Milliken, C.A. Santivanez, M. Condell, W.T. Strayer, A topological analysis of monitor placement, in: Proceedings of IEEE Int'l Sym. on Network Computing and Applications (IEEE NCA), 2007.

[70] A. Sahoo, K. Kant, P. Mohapatra, Characterization of BGP recovery time under large-scale failures, in: IEEE International Conference on Communications, 2006.

[71] S. Zhou, R.J. Mondragon, The missing links in the BGP-based AS connectivity maps, in: The Passive and Active Measurement Workshop, 2003.

[72] G. Huston, BGP Routing Table Analysis Reports, 2007. <http://bgp.potaroo.net/>.

[73] G. Combs et al., Wireshark. <http://www.wireshark.org/>.

[74] B. Manning, Registering New BGP Attribute Types, RFC 2042, January 1997.

[75] C. Jin, H. Wang, K.G. Shin, Hop-count filtering: an effective defense against spoofed DDoS traffic, in: Proceedings of ACM Conference on Computer and Communications Security, 2003.

[76] H. Wang, C. Jin, K.G. Shin, Defense against spoofed IP traffic using hop-count filtering, IEEE/ACM Trans. Netw. (TON). 15 (1) (2007) 40–53.

[77] F. Guo, T.-C. Chiueh, Sequence number-based mac address spoof detection, in: Recent Advances in Intrusion Detection, Springer, 2006, pp. 309–329.

[78] S.J. Templeton, K.E. Levitt, Detecting spoofed packets, in: DARPA Information Survivability Conference and Exposition, vol. 1, IEEE, 2003, pp. 164–175.

[79] E. Osterweil, D. Massey, B. Zhang, L. Zhang, Public data: a new substrate for key verification in DNSSEC, in: UCLA Computer Science Technical Report # 100020, 2010.

[80] J. Martin, R. Thomas, The underground economy: priceless, in: USENIX; Login, 2006.

[81] A.C. Snoeren, C. Partridge, L.A. Sanchez, C.E. Jones, F. Tchakountio, B. Schwartz, S.T. Kent, W.T. Strayer, Single-packet IP traceback, IEEE/ACM Trans. Netw. (ToN) 10 (6) (2002) 721–734.

[82] F. Baker, P. Savola, Ingress Filtering for Multihomed Networks, RFC 3704, March, 2004.

[83] Cisco, Strategies to Protect Against Distributed Denial of Service (DDoS) Attacks, Updated News Flash, April 2003.

[84] M. Collins, M.K. Reiter, An empirical analysis of target-resident DoS filters, in: Proceedings of IEEE Symposium on Security and Privacy, 2004.

[85] A. Bremler-Barr, H. Levy, Spoofing prevention method, in: Proceedings of IEEE Infocom, 2005.

[86] X. Liu, A. Li, X. Yang, D. Wetherall, Passport: secure and adoptable source authentication, in: Proceedings of USENIX/ACM Symposium on Networked Systems Design and Implementation (NSDI), 2008.

[87] D.G. Andersen, H. Balakrishnan, N. Feamster, T. Koponen, D. Moon, S. Shenker, Accountable internet protocol (AIP), in: Proceedings of ACM SIGCOMM, 2008.

[88] H. Lee, M. Kwon, G. Hasker, A. Perrig, BASE: an incrementally deployable mechanism for viable IP spoofing prevention, in: ACM Symp. on Information, Computer and Communication Security, 2007.