# Poster: SIPD: a practical SDN-based IP spoofing defense method


Chen Li
Peking University
icst-lichen@pku.edu.cn

Yu Ding
Peking University
dingelish@pku.edu.cn

Tongxin Li
Peking University
litongxin@pku.edu.cn

Jun Li
University of Oregon
lijun@cs.uoregon.edu

Xinhui Han
Peking University
hanxinhui@pku.edu.cn



***Abstract*—IP spoofing has become one of major threats to the Internet, while popular defense methods like ingress/egress filtering cannot stop IP spoofing effectively. This poster introduces SIPD, a feasible and scalable SDN-based IP spoofing defense method, which runs on the SDN controller and is compatible with the OpenFlow specification. It can automatically generate filtering rules and corporate with other SDNs that support SIPD. SIPD enforced SDN can detect all the intra-AS IP spoofing packets and most of the inter-AS IP spoofing packets.**


## I. Introduction

IP spoofing has a long history in network security. Attackers can hide themselves, pretend to be someone else or flood the victim with overwhelming amounts of traffic.

Recent work [1] showed that attackers can exploit IP spoofing to launch destructive DRDoS(Distributed Reflection Denial of Service) attack against popular websites or government facilities. The traffic of the largest IP spoofing based DRDoS attack at March 2013 reached 300Gbps. A research [2] showed that emule-Kad based DRDoS attack can easily generate more than 670Gbps traffic towards the victim.

The most commonly used method is ingress and egress filtering [3], [4], but it cannot stop IP spoofing within AS(Autonomous System). According to the statistics from Spoofer project [5], 40.8% of ASes are spoofable, and this number is decreasing slowly. SDN(Software Defined Network) offers new possibility to solve the IP spoofing problem. SDN's central controller can collect network information and distribute filtering rules easily, and make cooperation between SDNs much simpler than traditional self-organizing solutions.

SAVE(Source Address Validity Enforcement) [6] provides source address validation by making routers sending out advertisements about the subnet it holds, all routers in forwarding path can learn the valid incoming interface for source address spaces. To take advantages of SDN and enhance the SAVE protocol, we design and build a prototype named SIPD. SIPD runs on SDN controller. It can automatically collect network information and generate filtering rules for switches, and make it possible for adjacent SDNs to corporate with each other. Figure 1 shows the overall design of SIPD.

With SIPD deployed, attacker inside SDN cannot forge IP address within or outside that SDN. When incrementally deployed among SDNs, SIPD improves the ability to detect spoofed packets.

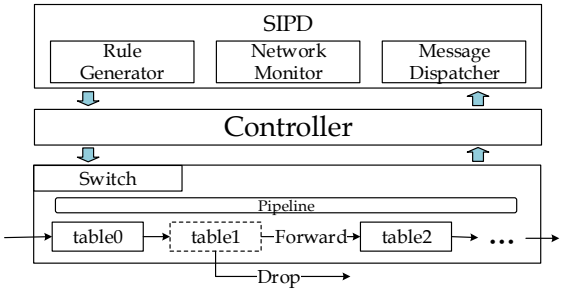


Fig. 1. Overall design of SIPD

## II. Design

SIPD is a software running on the SDN controller. SIPD on each controller takes charges of generating filtering rules for its domain and interacting with other controllers.

### A. Within SDN network

*1) Topology discovery:* By making switches sending out and listening for specially crafted LLDP packets at physical interfaces, controller can collect information about connections between switches. The LLDP packet contains switch id, interface id and hardware address. When a switch receives this packet, it sends a report to the controller, which contains its switch id, incoming interface id, other details of itself and the raw packet it has just received. Then the controller could know these two interfaces and switches are linked together. And because the controller is responsible for generating forwarding rules, it would be easy to obtain subnet information.

Besides, thanks to OpenFlow's comprehensive features on status monitoring, whenever a link is up or down, controller will get noticed immediately. Thus the controller can always have up-to-date topology information, and take action to deal with network change, making SIPD more reliable.

*2) Generate filtering rules:* After controller gets the topology and subnet information, it performs following algorithm to generate filtering rules for each switch.

Firstly, for each switch, the controller explores the path from this switch to every destination address space in the forwarding table. and then builds forwarding trees for all the switches in its domain. Figure 2(a) shows one example of forwarding tree. In this example, controller follows the path from 169.71.10.0/24 at switch 1 to 169.72.0.0/16, until the path reaches switch 3, 5 and 8, and build the forwarding tree.

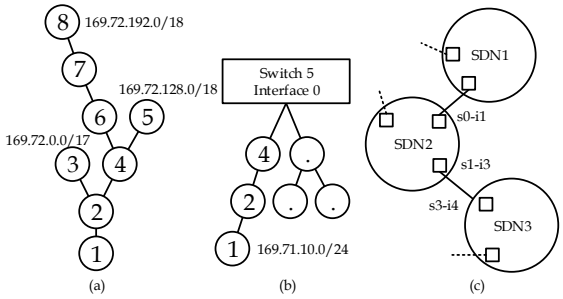


Fig. 2. Algorithm: (a)forwarding tree for target 169.72.0.0/16 from 169.71.10.0/24 (b)incoming tree (c) SIPD between SDNs

Secondly, for every switches and physical interfaces, controller collects information from correlative forwarding trees and builds an incoming tree. Every node in the incoming tree represents a source address space, and they forms a hierarchical relationship based on forwarding path. This incoming tree instructs the valid source address spaces for a interface of switch. As shown in figure 2(b), controller builds a incoming tree for switch 5's interface 0, which means all packets come from subnet of switch 1,2,4 and other three should reach interface 0 of switch 5.

Then, controller combines incoming trees of one switch into one incoming table. This incoming table contains filtering rules like $\langle$ SourceAddressSpace, InterfaceID $\rangle$. At this point, switches are able to perform IP spoofing detection and filtering.

*3) Deploy rules:* For OpenFlow compatible switches, incoming packets will go through a pipeline that contains several flow tables. So the best way to deploy filtering rules is making incoming table the second table, after all other blocking rules and before all forwarding rules. For a source IP address, switch tries to find a matching address space in incoming table and determine whether the source address is valid or not. For those who cannot match any source address space, SIPD treats it as valid. For example, filtering rule $\langle$169.71.10.0/24, 0$\rangle$ means packets from 169.71.10.0/24 should come in from interface 0. If it comes from other interfaces, it will be judged as spoofed.

*4) Handle network change:* Network hardware or link failure may cause topology to change. If controller does not react to this event and adjust filtering rules quickly, switches may improperly filter out valid traffic. In SIPD, once controller gets noticed about link change, it applies the algorithm again and calculate the difference between forwarding trees. Controller only push filtering rules to affected switches. This lazy update policy can save time and decrease communication costs.

### *B. Between SDN networks*

SIPD also enables corporation between SDN networks. To achieve this goal, the controller makes every border switches send out a special packet, which contains the information of subnets in the SDN that go through it and target IP address spaces. Once a border switch in another SDN gets that packet, it forwards it directly it to its controller, with the incoming interface id. Then controller applies filtering rules to all border switches, not only the switch that received it.

If the target address space is not in its domain, the controller needs to forward the message to other SDNs in the path to target. And if the target address space splits into two or more sub-spaces and go through different border switches, the message needs to be duplicated and sent to each target space.

As shown in figure 2(c), after SDN1 sends out the message, SDN2 will know that packets come from SDN1 should come in from switch 0's interface 1, and SDN3 will know those packets should come in from switch 3's interface 4.

To maintain network status, controller needs to send out messages regularly. And if network changes, the controller also makes the border switches send out messages immediately. To reduce the overhead of messages, controllers attaches their subnet information to the tail of passing by SIPD messages.

## III. Related works

There are several researches focusing on IP spoofing defense. Yaar et al. proposed StackPi [7], a packet marking method that makes routers add their fingerprint into IP header of passing packets, and filter spoofed packets on end-hosts. But it requires plenty of StackPi routers across the internet, which cannot easily be fulfilled. Wang et al. proposed a hop-count based defense method [8], which generates maps between IP address and hop-count at normal state, and inspects hop-count of every incoming packet at defense state. But hop-count is not reliable in asymmetric internet, which leads to high false-positives and false-negatives.

## IV. Conclusion

In this poster, we described SIPD, a SDN based IP spoofing defense method. SIPD can help SDN controller to automatically collect topology information, generate high availability filtering rules and distribute the rules to switches. What's more, it defines a protocol that enables corporation between SDN networks to fight against IP spoofing, only with the help of border switches in SDN network.

## References

[1] Christian Rossow. Amplification hell: Revisiting network protocols for ddos abuse. 2014.

[2] Bingshuang Liu, Skyler Berg, Jun Li, Tao Wei, Chao Zhang, and Xinhui Han. The store-and-flood distributed reflective denial of service attack. In *Computer Communication and Networks (ICCCN), 2014 23rd International Conference on*, pages 1–8. IEEE, 2014.

[3] Paul Ferguson. Network ingress filtering: Defeating denial of service attacks which employ ip source address spoofing. 2000.

[4] Avishai Wool. The use and usability of direction-based filtering in firewalls. *Computers & Security*, 23(6):459–468, 2004.

[5] Robert Beverly and Steven Bauer. The spoofer project: Inferring the extent of source address filtering on the internet. In *Proceedings of the Steps to Reducing Unwanted Traffic on the Internet on Steps to Reducing Unwanted Traffic on the Internet Workshop*, pages 8–8. USENIX Association, 2005.

[6] Jun Li, Jelena Mirkovic, Mengqiu Wang, Peter Reiher, and Lixia Zhang. Save: Source address validity enforcement protocol. In *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1557–1566. IEEE, 2002.

[7] Abraham Yaar, Adrian Perrig, and Dawn Song. Stackpi: New packet marking and filtering mechanisms for ddos and ip spoofing defense. *Selected Areas in Communications, IEEE Journal on*, 24(10):1853–1863, 2006.

[8] Haining Wang, Cheng Jin, and Kang G Shin. Defense against spoofed ip traffic using hop-count filtering. *IEEE/ACM Transactions on Networking (TON)*, 15(1):40–53, 2007.