# SDN 플로우 테이블 제한에 따른 리소스 어택

뉴엔 트리 투안 힙, 김경백
전남대학교 전자컴퓨터공학부
e-mail : tuanhiep1232@gmail.com, kyungbaekkim@jnu.ac.kr

# Resource Attack Based On Flow Table Limitation in SDN

Hiep T. Nguyen Tri, Kyungbaek Kim
Department of Electronics and Computer Engineering
Chonnam National UniversityRepublic of Korea

**Abstract**

In Software Defined Network (SDN), data plane and control plane are decoupled. Dummy switches on the data plane simply forward packet based on the flow entries that are stored in its flow table. The flow entries are generated by a centralized controller that acts as a brain of the network. However, the size of flow table is limited and it can conduct a security issue related to Distributed Denial of Service (DDoS). Especially, it related to resource attack that consumes all flow table resource and consumes controller resources. In this paper, we will analyze the impact of flow table limitation to the controller. Then we propose an approach that is called Flow Table Management to handle flow table limitation.

## 1. Introduction

SDN opens a new approach to design and manage a network. SDN separates the control plane and data plane, which are tightly coupled in a traditional network. But, in SDN, the network devices such as switches and access points are responsible for forwarding packets based on flow entries that are installed by a centralized controller. The centralized controller acts as a brain which processes routing information and makes decision for configure a network, and it manages the rules for how to handle the packets. When a network device receives a packet, the device tries to find a flow entry which has the matches to the header of the packet. If the network device does not find any matched flow entry, it asks the controller how to handle the packet.

With this dynamic updates of flow tables of network devices, SDN makes a network more flexible and easier to manage. However, even though SDN provides many benefits to a network, the dynamic feature may lead some security issues. A central controller is a highly valuable target for malicious attackers to compromise a network. A controller is the brain of a network, and if the brain does not work properly the operations of the entire network can be influenced. Generally obtaining the full control of a controller is not easy, but it is easy for attackers to make a controller very busy and disturbing the operation of the network. DDoS attack is one of the simple attacks to disturb a controller and this kind of attack has been facilitated by user friendly tools such as Stacheldraht.

Especially, in SDN, attackers can exploit the asking controller of switches whenever they receive an unknown header packet to launch a resource attack by sending bogus packets [1] [2] [3]. A flow table which contains the information for handling packets is the most importance component of a network device. However, the flow table size is limited as few hundreds of flow entries for recent OpenFlow switch [4] [5]. Attacker can easily exploit this limitation to fill the network device's flow table. And then he can send the high load traffic to the network. Because the flow table of network device is full, whenever it receives a packet, it has to send a request to ask the controller. With high load traffics, the resources that controller consumes can be very big. It impacts to the performance of the network and also impacts to the performance of the network. A similar attack scenario was mentioned in [6]. However, the impact of this resource attack has not been explored in detail. In this paper, firstly we will analyze and explore the impact of the resource attack to the SDN. And then we will propose an approach called Flow Table Management to handle flow table limitation.

The paper is organized as follows. In section II, we will describe the operation of SDN network, the analysis of consequence when the flow reaches the limitation. In section III, we discuss the possible solutions to mitigate the resource attack. Finally, we conclude the paper in section IV.

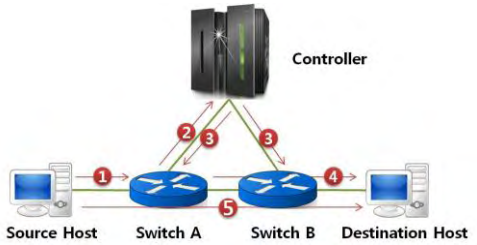
## 2. Resource Attack in SDN

In this section we first describe operation of SDN, and then we analyze the impact of flow table limitation.

### A. SDN Operation

Figure 1 illustrates the basic SDN operations for transferring traffic of a flow. Firstly, the source host sends the first packet of a flow to the network device (1). Secondly, the network device looks up matched flow entries. If there is no matched flow entry, the network device sends a request to the controller (2). In the controller, the request will be forwarded to the control application. Thirdly, based on the knowledge of the network, the application makes a decision and tries to

install flow entries to all of the network devices along the path of the packet (3). After installing the flow entries, the application sends back the packet to the network device which requests the flow entries as well as to the last network device of the path of the packet. Then the last network device forwards the first packet to the destination host (4). After deploying flow entries in every network devices, the network devices handle the following packets of the flow by referring the instruction field of the deployed flow entry (5).




**(Figure 1) SDN Operation.**

### B. Resource Attack in SDN

Generally many SDN devices employ the Ternary Content-Addressable Memory (TCAM) as storing the flow table and lookup flow entries for a given packet header. TCAM is a specialized high-speed memory that searches its entire contents in a single clock cycle, but it has some disadvantages; such as high power consumption, high cost, and low utility of ASCI space. That is, increasing the memory space of TCAM can lead to other problems about the price, the energy, and the physical size of network devices. According to this, generally the size of flow table of a network device is not that big. For example, the 5406zl switch can support about 1500 OpenFlow rules or 64000 forwarding entries for standard Ethernet switching [4].

This size issue of a flow table has not been concerned seriously under legacy networks, but in SDN it can be a serious issue. SDN is traffic oriental rather than host oriental, which means we may have to install multiple flow entries for each host in the network. Therefore, the number of the flows may be much bigger than the number of hosts in the network. That is, there is a high possibility that a flow table reaches its limitation with normal SDN operations. While the size limitation of a flow table causes a scalability issue, it may also cause a security issue. That is, the size limitation of a flow table may be an attractive attack point for malicious attackers who want to subvert a SDN network. Attackers easily fill up a flow table by sending raw packets with different packet headers. Actually this kind of attack requires simple knowledge of network programming and it can be easily facilitated by using naive traffic generation tools.

Let we imagine how the controller acts when the flow table becomes full. Figure 2 and 4 show two different procedures of handling the new request when the flow tables of all network devices are full. In figure 2, the controller application fails to install new flow rule to the network devices. However, the controller application will forward the packet by itself. That is, the controller application responds directly to the last network device in place of responding back to the switch which requested the controller. Actually, we encountered this procedure when we were testing the OpenDayLight controller with its default forwarding application. We can observe that every packet which does not match any flow entry will go through the controller. This procedure has two disadvantages. Firstly, going through controller can lead to other issues. For example, if a firewall sits on the middle of flow path to prevent violation traffics, going through the controller will disable the firewall's function. The attacker can exploit this behavior of controller application to violate protected network component. Secondly, the attacker can exploit the controller application behavior to generate simultaneous high load traffics after filling the flow table. Handling all the packets of high load traffics will consume a lot of controller resource.

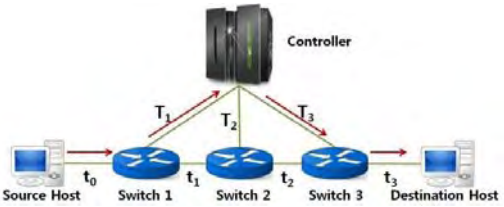



**Figure 2 Controller forwards the packet by itself.**

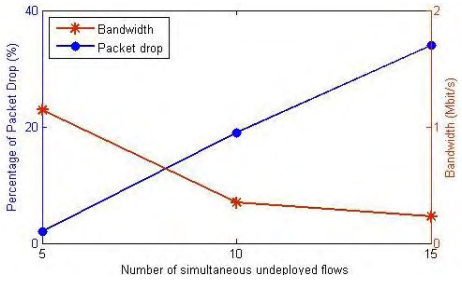



**Figure 3 Relation between bandwidth, packet drop and number of flows.**

We have evaluated the impact of flow table limitation to the network performance. In our experiment, we use OpenDayLight as controller and Mininet for emulating network. Mininet and OpenDayLight are both installed on a single virtual machine operated with Ubuntu 13.04. The virtual machine has 3GB memory and 2 3.4GHz CPUs. We consider a network which consists of 40 hosts and 2 switches. Controller-switch delay is set to 100ms, delay between switches is 30ms. 40 hosts are divided into 2 equal groups. Each group connects to different switch. Firstly, we set the flow table limitation which can support 5 flows. First, we fill the flow table of switches by using ping command. Then we run multiple flows at the same time and measure the bandwidth and packet drop ratio of undeployed flows which are does not match flow entry. To run simultaneous undeployed flows, each host in the second group initiates an UDP server and each host in the first group runs an UDP client simultaneously. We vary the number of simultaneous flows from 5 to 15. Figure 3 shows the effect of number of simultaneous flows to bandwidth and packet drop ratio. As the number of simultaneous undeployed flows increases, the packet drop ratio increases linearly and the bandwidth decreases exponentially.

The second procedure for handling the new request when the flow tables of all network devices are full is shown is figure 4. In this case, the controller tries to replace an old flow entry (which can be chosen based on the expired time)

by a new flow entry and then send back the packet to the network device that request to the controller. In this case, attacker can simultaneously send huge number of packets that have different headers. The controller still has to consume resource for handling packet. Therewith, controller and switches have to spend time and processing resource for removing and installing flow entries. Even though this procedure keeps avoid other issues by keeping traffic goes through data plane, the performance of controller can be worse than the first procedure.

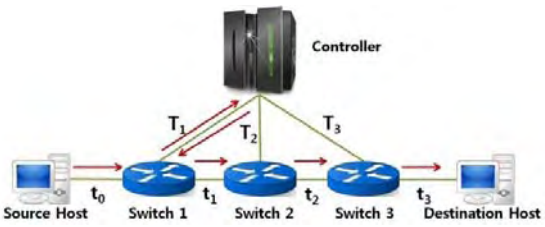



**Figure 4 Controller replaces flow entries.**

## 3. Flow Table Management

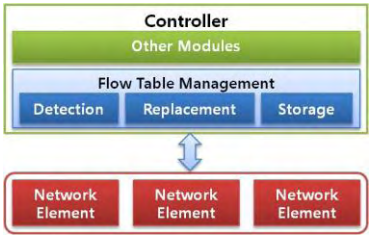



**Figure 5 Flow Table Management's Architecture.**

Based on the above analysis, we highly recommend considering resource attack which based on flow table limitation is a serious problem with SDN security. In this section we will describe an approach called as Flow Table Management (FTM) to handle flow table limitation problem. FTM is a controller's component, which manages flow tables of network devices. FTM includes 3 modules Storage, Detection and Replacement.

Storage is the module for managing flow entries of all network devices. Flow entries are stored in 2 tables Installed Table and Failed Table. Installed table stores the flow entries that are successfully installed on network devices. Failed Table stores the flow entries that are failed to install on network devices. This module also continuously tracks the flow table state. When a switch request to ask the controller, Storage will first match the packet header with the flow entries stored in the Failed Table. If there is a flow entry that matches the packet header, Storage will immediately respond to the network device.

Detection is a module that detects attack that based on flow table limitation. If Detection module detects an attack, Detection module can remove the flow entries that are generated by attacker. If attacker continuously changes the header, Detection module can install flow entry to drop all packet that come from the hosts which attacker is using.

When network is not under attack, the flow table is still able to be full. In this case, if middle layer does not replace old entry and the new traffic has high load, the performance of controller can be significantly degraded. If middle layer replace the flow entry that is for routing high load traffic by a new flow entry that is for routing light load traffic, the performance of controller can be degraded too. Hence, a smart replacement algorithm to optimize the performance is required. Replacement is the module is responsible for replacing flow entries that routes the light load traffic by flow entries that routes the high load traffic to achieve best controller performance.

## 4. Conclusion

In this paper, we have examined the impact of resource attacks which exploit the size limitation of a flow table to the performance of SDN. Our evaluation shows that the performance of a controller is significantly degraded under resource attacks if a controller application does not define the detail handling procedures for undeployed flows. Moreover, the size limitation issue may cause the unexpected behaviors of a controller, which create additional security problems. We also propose an approach that can manage the flow table of network devices, detect the attack and smartly replace flow entries when the flow table is full.

## Acknowledgement

This work was supported by the National Research Foundation of Korea Grant funded by the Korean Government (NRF-2014R1A1A1007734).

## Reference

[1] D. Kreutz, F. M. V. Ramos, and P. Verssimo, "Towards secure and dependable software-defined networks." in HotSDN, N. Foster and R. Sherwood, Eds. ACM, 2013, pp. 55–60. [Online]. Available: http://dblp.uni-trier.de/db/conf/sigcomm/hotsdn2013.html

[2] D. Li, X. Hong, and J. Bowman, "Evaluation of security vulnerabilities by using protogeni as a launchpad." in GLOBECOM. IEEE, 2011, pp. 1–6. [Online]. Available: http://dblp.unitrier.de/db/conf/globecom/globecom2011.html

[3] K. Benton, L. J. Camp, and C. Small, "Openflow vulnerability assessment." in HotSDN, N. Foster and R. Sherwood, Eds. ACM, 2013, pp. 151–152. [Online]. Available: http://dblp.unitrier.de/db/conf/sigcomm/hotsdn2013.html

[4] A. R. Curtis, J. C. Mogul, J. Tourrilhes, P. Yalag, P. Sharma, and S. Banerjee, "Devoflow: Scaling flow management for high-performance networks," in In ACM SIGCOMM, 2011.

[5] N. P. Katta, J. Rexford, and D. Walker, "Incremental consistent updates." in HotSDN, N. Foster and R. Sherwood, Eds. ACM, 2013, pp. 49–54. [Online]. Available: http://dblp.unitrier.de/db/conf/sigcomm/hotsdn2013.html

[6] S. Shin and G. Gu, Attacking software-defined networks: A first feasibility study, in Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, ser. HotSDN13. New York, NY, USA: ACM, 2013, pp. 165.166. [Online]. Available: http://doi.acm.org/10.1145/2491185.2491220