Prevention Mechanism for Infrastructure based Denial-of-Service attack over Software Defined Network

Sandeep Singh  
Department of I.T.  
B.B.A. University  
Lucknow, India  
Drsandeep.gbu@gmail.com

R.A. Khan  
Department of I.T.  
B.B.A. University  
Lucknow, India  
khanraees@yahoo.com

Alka Agrawal  
Department of C.S.  
KMCUAF University  
Lucknow, India  
Alka_csju@yahoo.co.in

Abstract— In software Defined Networking a Denial-of-Service (DoS) or Distributed Denial-of-Service (DDoS) attack is an attempt to make a machine or network resources unavailable for its intended users. Hence the need for protection of such network controller against attacks from within or outside a network is very much important. Although network devices in open flow can also be targeted by attackers and so required a prevention mechanism in order to avoid problems in smooth packet forwarding. In this research task we compose an Infrastructure based DoS attacking scenario over the software defined network and address the vulnerabilities in flow table, and afterward we developed a prevention mechanism to avoid such kind of attack in its initial stage before harming our network. The scenarios for the Infrastructure based DoS attack are developed using Mininet 2.2.0 and platform used for the simulation is Linux Ubuntu-14.10 Utopic Unicorn.

Keywords— Software Defined Network, Open Flow, Denial of Service attack, Mininet

I. INTRODUCTION

Basically open flow is a protocol that allows a server to tell network switches where to send packets. In a conventional network each switch has proprietary software that tells it what to do. With the help of open flow protocol, the packet forwarding decisions can be centralized so that the network can be programmed and controlled in a unique way [14]. In our conventional switch packet transmission and high level routing decisions were occurs on the same device. After introducing SDN these two works were separated from each other. An open flow switch basically differentiate data path from the control path, the data path portion resides on the switch itself but a separate controller is used to make high level routing decisions [15]. The switch and controller communicate by means of the open flow protocol and the process is termed as Software Define Networking (SDN) [16]. The software defined networking architecture supports a very highly reactive security analysis, response and monitoring mechanism. With the help of security perspective in software defined network architecture it is expected to provide protection and security to the resources of this programmable network [17]. It is also mandatory to protect the controller because of being a centralized decision point. Access to the controller needs to be tightly controlled. If the controller goes down because of Denial-of-Service (DoS) attack so goes the network. Which means the availability of controller needs to be maintained [18]. Protection of the communication throughout the network is a difficult job so a monitoring process should always be present.

Here in this research an infrastructure based Denial-of-Service attack is taken against the SDN network and a protection mechanism for the detection and prevention of this attack is developed [19]. A denial-of-service attack is defined as an attempt by an attacker to prevent access to the online resources and services to its intended users [1]. The DoS attacks constitute a significant threat to the internet. The security practitioners are constantly developing defense counter measures but attacker every time changes its working pattern according to current security solutions. Thus the DoS attack methods evolved in terms of complexity over the years [2]. Different new kinds of attacks are there like protocol based bandwidth attack, application based attack and infrastructure based attack [20]. These new types of DoS attacks are even more detrimental for network services have recently emerged. The DoS link flooding attack aims to flood specific network links disrupting all services passing through them. The DoS attack against Spamhaus [3] constitute a recent example of these DoS attack mechanism.

Here in our work we are addressing a new kind of DoS attack that is based on a particular geographical area [21]. This is basically an infrastructure and is made of with an open flow enabled software defined network. Different other kinds of attacks are addressed previously but this kind of attack is still untouched by the security experts. Here we are creating a software defined network with one openVswitch, four servers
connected through it and a centralized decision point of SDN controller [22]. According to figure 2 we have arranged a DoS attack over this set of infrastructure and suggest a prevention mechanism to resolve the problem at its early stage. To revoke a network after getting it down is more complex than to detect and prevent at its initial stage. So we worked to detect the attack by some set of guidelines and developed some policies to avoid this attack before damaging our network [23].

The rest of the paper is organized as: Next section discusses about previous contributions given in this direction. Section 3 discusses about the pertinent problem and proposes a defense mechanism to avoid this kind of attack. Results and discussions has been done in section 4 and paper concludes at section 5.

II. BACKGROUND WITH PREVIOUS CONTRIBUTIONS

Very limited research was conducted in the area of Denial-of-Service attack detection and prevention mechanisms. Since in industry, the security practioners and researchers have not given their enough attention so a very limited no. of published work is available. In [6] R. Braga et al. present a low overhead technique for traffic analysis using self organizing maps to classify flows. This mechanism is deployed on an SDN network to enable DoS attack detection and mitigation using NOX controller [24]. The coremelt [5] and crossfire [4] attacks shows the feasibility and consequences of DoS attack. Even then there is not sufficient work for the defense solution against the DoS link-flooding attack because of the high complexity involved in their characteristics and it also not allowed them to happen occasionally. However the potential effects of such network attacks are detrimental in terms of cost for internet services and can surpass the cost of developing such attacks when disrupting services which highly depends on internet such as industrial networks, financial and governmental services etc. [8]. In [7] the author worked for defeating DoS link flooding attack and also proposed a cooperative method for identifying low rate attack traffic.

DoS attack basically starts from an attacker importing a code into compromised machines which are generally referred as botnets. At the time of attack these codes are activated and starts sending the bulk amount of data towards victim node [25]. A more sophisticated attack over a thin layer of compromised machines called handlers to control a large no. of machines called zombies. These zombies are responsible for flooding the stream of attack traffic [9]. DoS is most common and obviously the most dangers attack for overloading and disrupting the services in SDN network [26]. Every day different attackers generate more than 9000 such attacks and statistics shows that in the first quarter of 2013 average attack bandwidth reached over 48.25 Gbps which is 718 % higher than the last quarter of the year 2012 [10]. From may 2013 to September 2013 united states and china have suffered significant numerous attacks of same category. Google is also providing some statistics for observing such kind of malicious activities over the network so launched a dedicated website to monitor digital attack map [11].

Launching a DoS attack requires an access to the victim machine. Attackers generally use some scanning methods to find out the vulnerable nodes over the network. These scanning methods can be random and is based on the hit list like local subnet scanning or any algorithm designed by an attacker. These attacks can be differentiated as an application based, resource based or may be infrastructure based [12]. Google and Arbor servers are having their own websites for monitoring attacks and collecting statistics from all over the world [11]. These monitored observations shows different key information regarding attack pattern like duration of attack, bandwidth used, mostly used sources, destination ports, origin of attack and the victim's country. UDP flood is another kind of DoS attack where a large no. of packets are sent to random ports in the vulnerable hosts. In the mean time the machine replied as destination unreachable. When no. of incoming requests increases, delay also increases and the machine goes into inaccessible mode.

Similarly in SYN flood DoS attack TCP connections are used for targeting victims node. A huge no. of SYN packets are flooded towards victim node and in return no ACK is received. This kind of attack was a highly detected one up to mid 2013 with 38.7 % of perpetrated attacks [13]. Under domain name system (DNS) reflection attack spoofed IPs are asking for a response that is much larger than the requested. In this kind of attack the attacker changes its source IP address and flood a heavy traffic towards the victim node. HTTP flooding and ICMP flooding based DoS attacks are also worked on the same flooding concept. Very high no. of ping requests has been generated and server went to unavailable mode.

In all above scenarios it is observed that DoS attack is basically an attack by sending a large no. of requests to servers, and because of limited resources these servers are unable to handle such requests in required rate and becomes unavailable or went in a hanging situation. Here in this research task we are addressing this kind of Infrastructure base DoS attack and also give a solution to avoid such kind of attack at its early stage before damaging our Software Defined Network in a bad manner.

III. PERTINENT PROBLEM AND PROPOSED SOLUTION

Here we are focusing on an Infrastructure based DoS attack and working for a solution towards the different mechanism of this kind of attack. We have designed a software defined network with a single ovs switch (S1) and a controller (Co) for making centralized decision point of source. We have connected four servers (h1, h2, h3 and h4) to the SDN switch. These components make an infrastructure of a specialized service enabled network.

There are other outer networks available in the picture which is not SDN enabled. In our scenario we considered three outer networks (N1, N2 and N3) which are connected through our specialized SDN network. These outer networks consisted different users like (U1, U2 and U3). We can differentiate the problem in to three sub parts where in first scenario (according to figure 2) user U1 from outer network...
N1, U2 from outer network N2 and another user U3 from network N3 are asking for different services to different servers of our SDN network. User U1 from N1 is requesting for services of file transfer protocol (FTP) and voice over internet protocol (VoIP) to server h1 and server h2 respectively. In the same time U2 from N2 is looking for services of TCP and UDP from server h3 and server h4 respectively. In the similar fashion U3 from N3 is requesting for SMTP, IMAP, SSH and TELNET services from server’s h1, h2, h3 and h4 respectively. In the first scenario all the requests from different users are accepted and replied in an efficient manner by all the servers of SDN network. Link bandwidth is configured as 10 Mbps for all connected resources.

In the second scenario where we are dealing with a DoS attack is configured using Mininet-2.2.0-141209- Ubuntu-14.04-server-i386, with its trusty server package of 32 bits. The environment was created on Linux platform and Oracle VirtualBox (VM) - 4.3.20-96997-win is used to perform simulation. Xming-6.9-031- setup is also installed for assigning and connecting different IPs for providing connectivity between hosts and servers. A putty server-0.63.0.0 is used as root login utility between servers and openVswitch. We have used wireshark-win-32-1.12.2 for the analysis of packets transmitted from different interfaces. The network is developed under Ramonfontes Visual Network Description (vnd). Visual network description basically allows you to design your network manually according to requirements and it also generates the python (.py) and bourne shell (.sh) scripts for network. We have developed our network using this tool and properties of all links and ports configured manually. After creation of network we can generate different scripts for open flow controller and can define its QoS property according to our scenario. Now for importing our topology developed by VND over the mininet given command should be run as super user “sudo wget http://ramonfontes.com/vnd/scripts/mininetScript12345.sh”.

Now after successfully importing the script we shall test its functionalities and run it by “sudo chmod +x mininetScript12345.sh” “sudo nano mininetScript12345.sh” “sudo ./mininetScript12345.sh” according to figure 1.

In the second scenario we assume that user U3 becomes an attacker and is sending traffic to all servers through different connections. In the previous case U3 was generating requests to all servers and were receiving response successfully. The attacker sends busy traffic through flooding of request packets to all servers. In this case server h1 was requested by a huge number of request to get SMTP services. At server h2 this U3 sends a busy traffic request to get IMAP services, similarly at server h3 and h4 for SSH and TelNet respectively. Links of all servers to users are configured manually to implement the attack. These settings are applied to all servers. Now after we have exported this configured topology in to mininet python scripts. According to figure 2. Attacker simply flood the huge number of packets to all servers, since we know that servers are also handling the requests of other users (U1 and U2) but when a busy traffic came from an unknown and unauthenticated source it does not know what to do and how to treat this traffic. Server informs about this situation to the open flow switch and because of unknown and new flow request it simply transfers the flow in to open flow controller Co. Now because of limited resources and capability the link becomes congested and servers went to deadlock condition. The same thing happens with all other servers and services to a particular geographical area become unavailable for outside networks.

The implementation of such a scenario is performed by changing necessary manipulations into link configuration between servers and hosts. When we ping different hosts and servers it simply refuses the request and shows destination unreachable in all cases. So it is concluded that attacker sitting at N3 is using a DoS flooding attack for blocking the services of particular geographical area. The output obtained by Wireshark packet analyzer shows that none of open flow packet received due to DoS flooding attack.

Since we are in the effort to prevent the attack in its initial stage because after getting blocked, it is very difficult to manage and revoke the network again. So we have developed the four steps strategy to avoid this attack from being happen. In the first step when a heavy traffic is observed from a particular source of address we can use buffering. We are dealing with an intelligent network that can customize its properties from a single point of control so for smooth functioning of our network we can first use the buffering mechanism to hold the over flooded packets in a queue. Our server is handling several requests in a single time even it has a fixed limit to handle number of packets according to the

Fig. 1. Importing and running python Script developed under the Visual Network Description tool.
bandwidth assigned. When the requests came under its normal capacity to (1000 packets/sec) over a limited bandwidth (10 Mbps), everything works fine. When suddenly from a particular source of IP U3 no. of packets are increased unexpectedly like (10,000 packets per second) and the link bandwidth was consumed by this traffic a sudden congestion is monitored in the network.

According to our first step we can increase the available queue size for incoming packets and simply direct them to wait until the queue becomes empty. This will give some relaxation to the traffic handling. After applying this step if still the traffic is continuously increasing and our buffering policy came in to failure mode then we shall go for the second step. Setting Timer/ Time stamping: when buffering does not work we use timer for a particular source of IP address. The IP of generated traffic can be easily identified and marked for time stamping. This time stamping policy is simply like debar a particular source of IP for some specified time limit.

By setting a timer in the Visual Network Description (VND) scenario we can block a particular IP for some time interval. In this way other users can smoothly communicate with servers of SDN network and a specified IP will be put on hold for some time. After specified time limit has reached controller automatically allow all sources to transfer data. In this way if the problem resolved then we have nothing to worry if still a heavy traffic is coming it reconfirming about something is wrong there in network. In this situation we shall go for the third step of our mechanism.

Warning cum request packets: when none of the policy is working it replicates about the confirmation of attack. In third step we shall generate an Eco request packet to the specified source of IP to slow down the transfer rate of sending traffic.

This warning message will be applied thrice for slowing down the rate of packet transfer. The steps of this prevention mechanism are shown in figure 3. If at this stage the particular source of IP did not slow down the rate of sending traffic it is identified that a particular source of IP is an attacking point and some strict action must be taken against it. In our fourth step after applying the above policies to make sure about an attack is happening we simply trace back the source and block this IP by sending a command to the controller.

After applying the mechanism developed for prevention of Infrastructure based DoS attack the link congestion is avoided at its early stage and servers are also secured by getting into deadlock.

IV. RESULTS AND ANALYSIS

Analysis of all three scenarios have been done using the data obtained. For the first scenario according to the network topology, all three users U1, U2 and U3 form external networks N1, N2 and N3 are accessing the services from different servers of Open Flow enabled Software Defined Network. Proper responses are given by all respected servers because 8 different flows according to the requests were installed over the flow table of floodlight controller Co.

We are comparing scenario 1 Vs scenario 2 and at later stage scenario 2 Vs scenario 3. In first case time taken by different server’s h1, h2, h3 and h4 to responding different users of network is calculated and is drawn. Graph (i) is showing two vertical columns. Blue bar is representation for scenario 1 and brown is used for representing scenario 2. It is quite obvious that in first scenario there is a smooth functioning between servers and users but as the situation changes according to scenario 2 the drop of packet starts and because of heavy requests server becomes slow and the response time increases.
Similarly in Graph (iv), (v) and (vi) same measurements were calculated between scenario 2 and 3. These statistics shows the response time, delay in packet delivery and throughput in bits per second for the whole process. Since the available queue size (1000 of packets) and bandwidth (10 Mbps) was fixed prior so our main concentration was on the time taken to complete the processes.

The above graph in figure 5 shows the traffic analyses of all scenarios. This graph is generated between no. of packets transmitted per second and the flows installed in floodlight controller. When no. of packets increases from its threshold limit (< 1000) it is considered as DoS attack. The red line shows that DoS attack occurs on third flow onwards where no. of transmitted packets were increased suddenly. In third scenario green line shows that we have successfully applied our DoS prevention mechanism to decrease the rate of traffic flow.

IV. CONCLUSION

We are dealing with an infrastructure based DoS attack over a centralized network SDN. We have developed a prevention mechanism to avoid it at its early stage. We have successfully implemented all three scenarios where normal and abnormal traffic both were encountered. Our mechanism can help the researchers and industry experts both in an efficient way. The proposed policy cum prevention strategy can be applied in the real world traffic to avoid such kind of situation. We have actually implemented this mechanism over the Mininet simulator so anyone interested can suggest and propose improvements regarding the functionality of the mechanism. The DoS attack is one of the most common and dangerous one so this research contribution will help the security practitioners in a long term survival utility.

REFERENCES


Fig. 4. COMPARISON OF SCENARIOS: BETWEEN 1 AND 2 - IN TERMS OF (I) RESPONSE TIME IN MS (II) DELAY IN MS (III) THROUGHPUT IN BITS/SEC. AND BETWEEN 2 AND 3 – IN TERMS OF (IV) RESPONSE TIME TAKEN BY SERVERS (V) DELAY IN PACKET DELIVERY (VI) THROUGHPUT IN BITS/SEC

Fig. 5. Traffic analyses of different scenarios
International Conference on Computing, Communication and Automation (ICCCA2015)


