HTTP-based DDoS Attack and Defenses

Shan Zhu, Nairong Zhang, Ziming Chen

***Abstract*—In this project, we use penetration testing and intrusion detection tools on a server running NGINX to explore HTTP-based denial of service (DoS) attacks and defenses. We first set up HTTP server(NGINX) in a Docker container on Ubuntu Virtual Environment, and the HTTP client on Kali Virtual Environment. Then we utilized hping3 tools to perform DoS attacks from the client on NGINX with the default settings. Next, we explored the possibility to defend the DoS attacks by modifying the settings on NGINX servers. Finally, we introduced Suricata, an intrusion detection tool to detect the DoS attacks on the NGINX server.**

I. Introduction

A. HTTP Server - NGINX

We choose NGINX as our HTTP server for this task, which is a free, open-source, high-performance HTTP server and reverse proxy, as well as an IMAP/POP3 proxy server. NGINX is known for its high performance, stability, rich feature set, simple configuration, and low resource consumption.NGINX is built to offer low memory usage and high concurrency. Rather than creating new processes for each web request, NGINX uses an asynchronous, event-driven approach where requests are handled in a single thread. With NGINX, one master process can control multiple worker processes. The master maintains the worker processes, while the workers do the actual processing. Because NGINX is asynchronous, each request can be executed by the worker concurrently without blocking other requests.

B. DDoS Attack

A denial-of-service attack(DoS attack) is a cyber-attack in which the perpetrator seeks to make a machine or network resource unavailable to its intended users by temporarily or indefinitely disrupting services of hosts connected to the Internet. DoS is typically accomplished by flooding the targeted machine or resource with superfluous requests in an attempt to overload systems and prevent some or all legitimate requests from being fulfilled. If the incoming traffic flooding the victim originates from many different sources, it becomes a distributed denial-of-service attack(DDoS attack), which makes it impossible to stop the attack simply by blocking a single source.

In this project, we used SYN flood attack, which exploits the TCP handshake by sending a target a large number of TCP “Initial Connection Request” SYN packets with spoofed source IP addresses. The target machine responds to each connection request and then waits for the final step in the handshake, which never occurs, exhausting the target’s resources in the process. Here, we use hping3 to perform the SYN flood attack with spoofed IP addresses, which is a free packet generator and analyzer for the TCP/IP protocol, as one of the de-facto tools for security auditing and testing of firewalls and networks.

C. DDoS Defense

One of the easiest ways for NGINX DDoS prevention is to use software firewalls like Iptables. Iptables is an extremely flexible firewall utility built for Linux operating systems that allows a system administrator to configure the IP packet filter rules of the Linux kernel firewall. The filters are organized in different tables, which contain chains of rules for how to treat network traffic packets. Since most TCP-based DDoS attack types use a high packet rate, meaning the sheer number of packets per second is what causes the server to go down, we need to use Iptables to process and block as many packets per second as possible. To prevent SYN flood attacks in this situation, we limit the new TCP connections that a client can establish per second.

D. Intrusion Detection Tool - Suricata

Suricata is an open source network threat detection engine that provides capabilities including intrusion detection (IDS), intrusion prevention (IPS) and network security monitoring. It does extremely well with deep packet inspection and pattern matching which makes it incredibly useful for threat and attack detection. It is a rule-based ID/PS engine that utilises externally developed rule sets to monitor network traffic and provide alerts to the system administrator when suspicious events occur. Designed to be compatible with existing network security components, Suricata features unified output functionality and pluggable library options to accept calls from other applications.

E. HTTP Server Performance Measurements

We use two measurements here to monitor the performance of the HTTP server. One method is to perform ping tests from the client side to test the reachability of the server. Ping measures the round-trip time(RTT) for messages sent from the originating host to a destination computer that are echoed back to the source. If the HTTP server is under DDoS attack, it will take more time for the server to process and reply to the ping request, resulting in an increase of RTT. Thus, we can use RTT value to indicate the performance of the server. A longer RTT corresponds to a slower server. The other method is to use the “Speedometer” on the server side to measure and display the rate of data transmitted across the network connection. Speedometer is a graph-based network traffic monitor in Linux which can give out instantaneous throughput of the target interface (-rx) or the transmission rate of that interface (-tx). If the server is currently under DDoS attack, it's transmission rate would be significantly higher than usual. Thus, we will observe a duration of high transmission rate, mostly over 1Mib/s. This method could be also used to justify our implementation of NGINX and firewalls, under the condition of a well functioning network environment, if any decrease of transmission rate happened, it indicated the success of our defence.

II. Project Goal

This project is divided into four parts. First we set up the HTTP server and client in the corresponding environments. The HTTP server(NGINX) is running in a Docker container on Ubuntu Virtual Machine while the HTTP client is running on Kali Virtual Machine. Second, we perform DDoS attacks on NGINX server with default settings from the HTTP client. Third, we explore the possible improvements on the NGINX or OS security settings to better defend against the DDoS attacks, and then repeat the DDoS attacks on NGINX again and compare the results with the previous experiment. Last, we investigate the intrusion detection tool, Suricata, to detect the DDoS attacks on the NGINX server. The project is to motivate us to learn the details of DDoS attacks from SYN flooding to ICMP flooding and to bridge ourselves between experience and experiments. From this project, we will be exploring HTTP-based DDoS attacks and multiple defenses.

III. Project Execution Highlights

A. HTTP Client/Server Setup in KaliVM/Ubuntu

The client that communicates with the web server is any remote browser. We created the HTTP request by manually applying direct http GET commands using telnet over port 81. The team continued using the sample docker container from the course website. This container already had NGINX web server initialized inside, so the team began the experiment with running the container. Thus, after we restarted the NGINX web server, it started handling all incoming HTTP requests that come in over the enabled ports and generates appropriate HTTP responses for them.

B. Perform DDoS Attack on NGINX with default settings

We used hping3 to send manipulated packets to simulate a DDoS SYN flood attack. We created a sequence, which is hping3 -c 10000 -d 120 -S -w 64 -p 81 –flood –rand-source [hostname]. As defined in this sequence, we will send up to 10,000 SYN packets with a window size of 64 bytes that originate from different spoofed source IP addresses to the target host with destination port number 81.

C. Improve NGINX Settings or OS Network Settings to defend DDoS Attack

There are several features in NGINX we can modify to defend the DDoS attack, from limiting the total connections in a given interval to blacklisting a range of IP addresses. In this experiment, the team chose to limit the rate of request, limit the number of connections and close the slow connections. We configured NGINX and NGINX Plus to allow a single client IP address to attempt to send requests to our particular destination only every 2 seconds (equivalent to 30 requests per minute). In addition, we also limited the number of connections that can be opened by a single client IP address, again to a value appropriate for real users, such that we configured to allow each client IP address to open no more than 10 connections to our website. What's more, we closed connections that are writing data too infrequently, which can represent an attempt to keep connections open as long as possible (thus reducing the server's ability to accept new connections). We performed modifications listed above but we received relatively mild changes regarding the transmission rate, we simply lowered down the instantaneous throughput from the 2.48Mib/s to 2.44Mib/s (lowest rate in Figure 1 below). In order to achieve more significant changes to justify our legitimate defense, we tried to set our foot on OS security level and set up firewall rules by modifying iptables to better defend against DDoS attacks. The team installed Iptables in the original web server container. We configured Iptables rules to defend against DoS attacks. We limit the number of incoming tcp connection or syn-flood attacks, and all incoming connection are allowed till limit is reached. We also make the server allow only a maximum of 25 connections per minute. In this try with the firewall establishment, we achieved significant improvement against the DDoS attack. From the figure shown below, it is obvious to see a significant decrease of transmission rate of the target interface. After performing DoS attack on NGINX server, the rate of data transmitted by the server surges from approximately 98 B/s to 1MB/s. However, after we improved defense against DoS attacks by improving OS network security using Iptables, the rate of data transmitted by the server drops down to only approximately 7 KB/s, which is a significant change. In other words, by deploying a firewall, we mitigate the flooding effects of the target channel (shown as Figure 2).

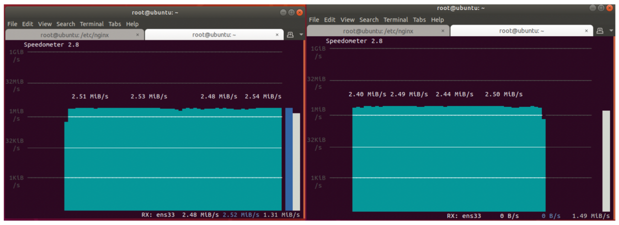


Fig. 1. Transmission Rate of Before and After modifing the NGINX Settings

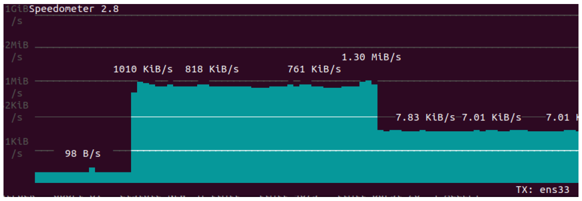
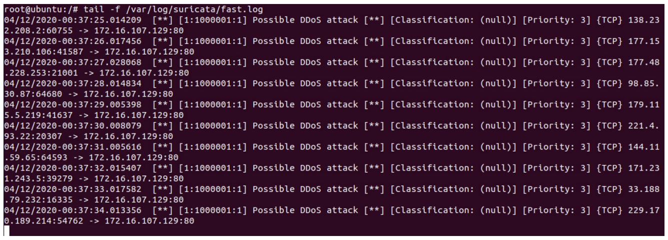


Fig. 2. Transmission Rate after applied Iptables Firewall

D. Install Suricata to Detect DDoS Attack

For the sake of further testing and experiments, the team performed the experiment with Suricata on Docker. We initially built a docker image with Suricata installed. When we run this container, we obtain a default setting of Suricata, and configured the network setting to always run on our target interface. As mentioned above, Suricata is a rule-based IDS, we are provided several options to choose. We created a rule when there are 100 attempted connections to the local network in 10 seconds to alert a possible SYN flooding. Then we configured Suricata to include this rule in its rule-files section. Therefore, we performed our experiment with Suricata in PCAP live mode and fetched the log files later on. (Log files are shown below.)



```
root@ubuntu:/# tail -f /var/log/suricata/fast.log
04/12/2020-00:37:25.014209  [**] [1:1000001:1] Possible DDoS attack [**] [Classification: (null)] [Priority: 3] {TCP} 138.23
2.208.2:60755 -> 172.16.107.129:80
04/12/2020-00:37:26.017456  [**] [1:1000001:1] Possible DDoS attack [**] [Classification: (null)] [Priority: 3] {TCP} 177.15
3.210.106:41587 -> 172.16.107.129:80
04/12/2020-00:37:27.028068  [**] [1:1000001:1] Possible DDoS attack [**] [Classification: (null)] [Priority: 3] {TCP} 177.48
.228.253:21001 -> 172.16.107.129:80
04/12/2020-00:37:28.014834  [**] [1:1000001:1] Possible DDoS attack [**] [Classification: (null)] [Priority: 3] {TCP} 98.85.
30.87:64680 -> 172.16.107.129:80
04/12/2020-00:37:29.005398  [**] [1:1000001:1] Possible DDoS attack [**] [Classification: (null)] [Priority: 3] {TCP} 179.11
5.5.219:41637 -> 172.16.107.129:80
04/12/2020-00:37:30.008079  [**] [1:1000001:1] Possible DDoS attack [**] [Classification: (null)] [Priority: 3] {TCP} 221.4.
93.22:20307 -> 172.16.107.129:80
04/12/2020-00:37:31.005616  [**] [1:1000001:1] Possible DDoS attack [**] [Classification: (null)] [Priority: 3] {TCP} 144.11
.59.65:64593 -> 172.16.107.129:80
04/12/2020-00:37:32.015407  [**] [1:1000001:1] Possible DDoS attack [**] [Classification: (null)] [Priority: 3] {TCP} 171.23
1.243.5:39279 -> 172.16.107.129:80
04/12/2020-00:37:33.017582  [**] [1:1000001:1] Possible DDoS attack [**] [Classification: (null)] [Priority: 3] {TCP} 33.188
.79.232:16335 -> 172.16.107.129:80
04/12/2020-00:37:34.013356  [**] [1:1000001:1] Possible DDoS attack [**] [Classification: (null)] [Priority: 3] {TCP} 229.17
0.189.214:54762 -> 172.16.107.129:80
```

Fig. 3. Suricata Log Records

IV. Goal Accomplishment

The project overall accomplished our goal settings, enhanced our understanding of the principles of DDoS attack and how we can mitigate the attack through different techniques. We practiced with hping3 to launch several flooding attacks and investigated various measurements to monitor the server's performance under attack. In order to defend the DDoS attack, we began with modifying the default settings of NGINX to mitigate the attack, like limit the connection rate of the server. Since the NGINX server is only part of the Ubuntu server, and the NGINX server in the Docker container share the host IP address, only modification on NGINX cannot prevent the DDoS attack thoroughly. Thus, we explored better ways to enhance the network security on the OS level, and shifted from modifying the NGINX server network settings to setting up specific firewall rules to more effectively mitigate DoS attacks. We implemented Iptables, which is a firewall that filters all incoming requests as we configured, which greatly decrease the transmission rate of the server and defended against the attacks. Finally, we successfully detected the possible DDoS attacks by setting corresponding rules in Suricata and stored the detailed information in its log.

V. What was Learned

From this project, we got familiar with the widely-used HTTP server, NGINX, and how to install it in a Docker container on Ubuntu Environment. We also learned how to perform DDoS attacks on the NGINX server through the hping3 application, as well as how to help the NGINX server better defend these DDoS attacks by modifying NGINX's configuration file and improving the OS network security settings via Iptables. Besides, we also explored the usage of another useful tool, Suricata, to detect the possible DDoS attacks to the server and report the potential threatening client IP address. In general, we got a profound understanding of penetration testing and intrusion detection on the NGINX server and explore HTTP-based denial of service (DoS) attacks and defenses.

VI. References

References

[1] "Denial-of-service attack", En.wikipedia.org, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Denial-of-service-attack. [Accessed: 13- Apr- 2020].

[2] "What is a DDoS Attack?", 2020. [Online]. Available: https://www.cloudflare.com/learning/ddos/what-is-a-ddos-attack/. [Accessed: 13- Apr- 2020].

[3] Iptables", En.wikipedia.org, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Iptables. [Accessed: 13- Apr- 2020].

[4] "How To Build Your Own DDoS Protection With Linux IPtables in 2020", JavaPipe, 2020. [Online]. Available: https://javapipe.com/blog/iptables-ddos-protection/. [Accessed: 13- Apr- 2020].

[5] "IptablesHowTo-Community Help Wiki", Help.ubuntu.com, 2020. [Online]. Available: https://help.ubuntu.com/community/IptablesHowTo?action=showredirect=Iptables. [Accessed: 13- Apr- 2020].

[6] "The Beginner's Guide to iptables, the Linux Firewall", How-To Geek, 2020. [Online]. Available: https://www.howtogeek.com/177621/the-beginners-guide-to-iptables-the-linux-firewall/. [Accessed: 13- Apr- 2020].

[7] "2. Quickstart guide—Suricata 5.0.2 documentation", Suricata.readthedocs.io, 2020. [Online]. Available: https://suricata.readthedocs.io/en/suricata-5.0.2/quickstart.html. [Accessed: 13- Apr- 2020].

[8] P. Policy and P. Policy, "Denial-of-service Attack – DoS using hping3 with spoofed IP in Kali Linux", blackMORE Ops, 2020. [Online]. Available:https://www.blackmoreops.com/2015/04/21/denial-of-service-attack-dos-using-hping3-with-spoofed-ip-in-kali-linux/. [Accessed: 13- Apr- 2020].

[9] S. Security and H. Attacks, "How to Use Nginx to Fight DDoS Attacks", rackAID, 2020. [Online]. Available: https://www.rackaid.com/blog/using-nginx-to-fight-apache-ddos/. [Accessed: 13- Apr- 2020].

[10] "jhu-information-security-institute/NwSec", GitHub, 2020. [Online]. Available: https://github.com/jhu-information-security-institute/NwSec/tree/master/applications/websvr. [Accessed: 13- Apr- 2020].