Stimulation of SYN Flood attacks and counter-attack methods using average connection times

Author: Hai Vo, Iyanu Odebode

Abstract: While SYN Flood attack leveraging TCP protocol is not new, this method is still by far the most popular attack type. In this paper, a stimulation of a TCP server is introduced, as well as three different ways to gather data on connection times and number of unresolved requests to connect in the server. The results show that the most efficient way to do so would be to use an average of only successful handshake connections to apply as a server timeout.

1. Introduction

With the growth and increased usage of our Internet resources during this time, prevention and detection of cybersecurity issues or attacks has become one of the utmost important aspect. Due to the nature of increased usage, one issue that can be leveraged along with it is Denial-of-service attacks. While the heavy load of data transmission can certainly affect the server, a DDoS attack is a way to try to make online services unavailable to other users by overwhelming it with traffic from multiple sources. This kind of attack can target various kinds of services, from financial online systems to entertainments, preventing people to gain access to valuable information.

There are multiple reasons why this attack is carried out, from too competitive gamers trying to gain an advantage amongst other gamers [1], to be used as cyber warfare in politics like the attack against North Korea's military spy agency ordered by the Trump administration [2]. Whichever the reason it might be, the number of attacks is increasing, and the technical details is everchanging. Some of the attacks can include UDP Flood, HTTP Flood, Ping of Death, etc. In UDP flood attacks, attackers can generate fake traffic that can consume the bandwidth and

overwhelm the system. Since the UDP protocol does not utilize handshaking techniques, this attack can easily be carried out with huge amount of traffic [3].

2. Overview of SYN Flooding attack

In our case, we will be looking at SYN Flood attacks, which target the TCP protocol. The TCP protocol is a guaranteed delivery protocol because it provides flow control and reliable data delivery with acknowledgements between both the server and the client. Specifically, a SYN flood attack relies on leveraging the way TCP communication. Usually, a client sends a SYN packet to the server to ask for a TCP connection. The server would then send back a SYN-ACK packet to the client to acknowledge the client connection. The client then sends to the server an ACK packet, acknowledging that prior package, and establish the connection between the two. The process described above is usually known as a "three-way handshake" [4].

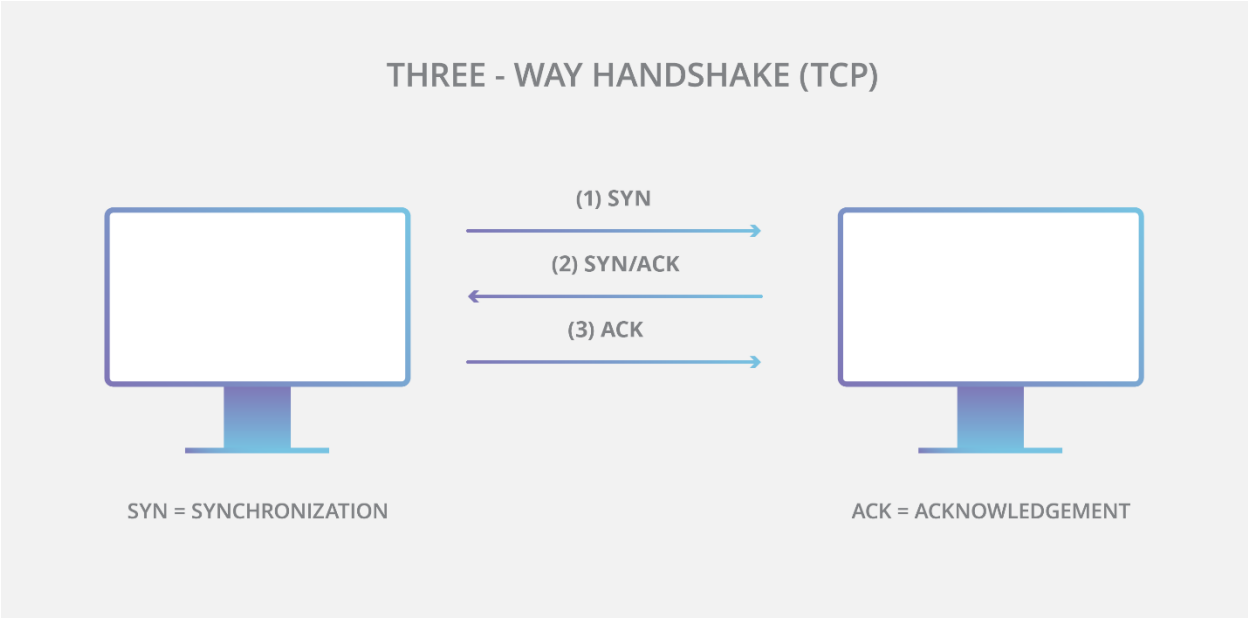

Figure 1: Three-Way Handshake in TCP Protocol

In the three-way handshake, when the server waits for the ACK packet from the client to establish the connection, is called the half-open state. In this state, the server keeps waiting for an ACK packet back from the client to finish the handshake. This is managed with a backlog queue, which can time out the connections that stay open for too long. However, when this queue is too full of request, in addition to the memory constraints of the server, this can lead to denial-of-server on the server-end. This means new connections cannot be made and instead of doing the three-hand shake, the server cannot respond to any SYN packet from real user who wants to connect. What makes this attack more popular is that the attack packets are the exact same SYN packet used in a legitimate connection, making it very hard to recognize the attack. In fact, SYN Flood attack is still one of the most popular attack, accounting for 92.57% of attacks in Q1 of 2020 [5].

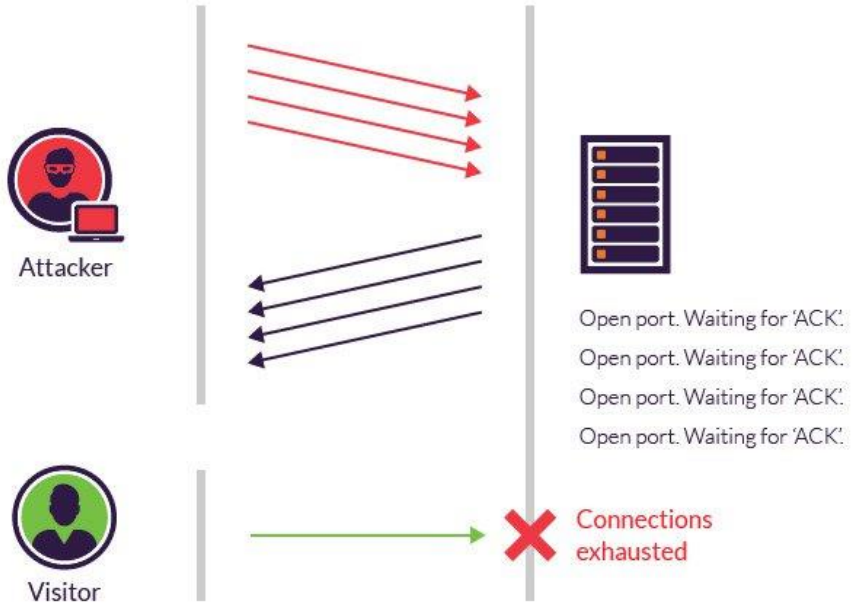

Figure 2: TCP SYN Flood Attack

3. Analysis of TCP server traffic with SYN Flood attack and attack mitigation

3.1. Setting up the SimPy stimulation environment

Our working SimPy environment is first set up with two classes, Client and Server.

For my Client class, a random IP is generated, with one SYN packet set to be sent to the server for the initial handshake. There is also an attacker_value, which then generate between 2 to 7 SYN packets to send to the server.

For my Server class, the server is set up with a server as a finite resource of 100 packets size, as well as lists containing the time for successful handshake connections and half-open connections to clients.

After setting up the classes, I create a handshake method, which takes a Server and a Client object as the arguments. This method proceeds to send a SYN request to the server from the client. If the client's attacker_value is set to False, the TCP_server then release the request from the queue, finishing the handshake and start the transmission of the files, which take between 1 to 6 seconds. Else if the attacker_value is set, then the Client object will request and send multiple SYN packets, in an attempt to fill up the queue.

There has been several methods implemented to counter these kind of attacks, including using a SYN cache [6], IP table firewalls [7], or SYN cookies [8]. For this stimulation, I am collecting data in three different ways, freeing up some of the processing queue using two techniques and not doing so for the other one. Both counter-attack methods are based on the paper "SYN flooding attack detection by TCP handshake anomalies" written by Martine Bellaïche and Jean-Charles Grégoire [9].

The first method of gathering data is to not apply any timeout for the request and not free up anything, in order to get a baseline of how it would affect the system.

The second method is to append all the connection times, measured between the reception of the SYN packet from the client to the end of the transmission between the server and the client, as well as any connection times of unresolved packets still stuck in the queue. One issue to this method is that each SYN packet received is set to take 0.5 seconds to complete due to technical difficulties, so this might not provide as much accurate data as it should be. However, it should still be a good measure.

The third method is to append only successful handshake connection times. This would provide a closer to real life server processing time. However, it does ignore the pending SYN request times from the attackers.

Using three different ways to set up the timeout for the queue, I create the server_time_out method, which pop off the first element from the queue every set amount of time assigned by the methods outlined.

3.2. Analysis of the result

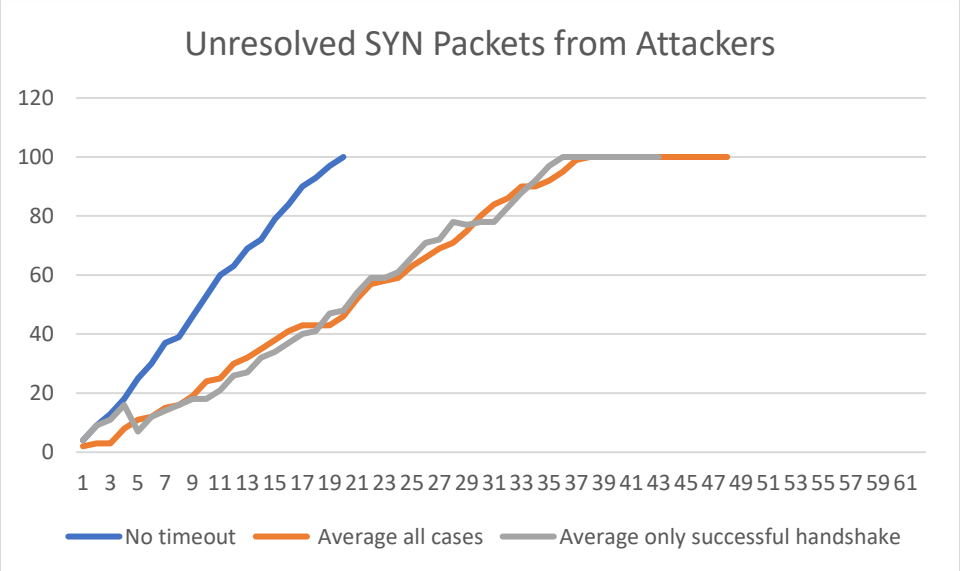

Figure 3: Unresolved SYN Packets from Attackers

Figure 3 compares the number of unresolved SYN packets in the queues from the SYN flooding attack using three different ways to set the server timeout. The figure shows us that when using no server timeout to dequeue some of the unresolved request, it can lead to a steady increase in the numbers of unresolved requests, leading to faster DDoS success for the attacker. It actually reached the peak so fast that the server crashed before half of the other methods had even finished. The interesting comparison is between the other two methods. When the server timeout is applied as all connections average, we see lots of sharp decrease and parts where it is stabilized, and it actually reached the maximum a little bit slower than the other methods. On the other hand, the method to apply only successful handshakes, does not differ that much from the average all cases method, except for the fact that it reached maximum quicker.

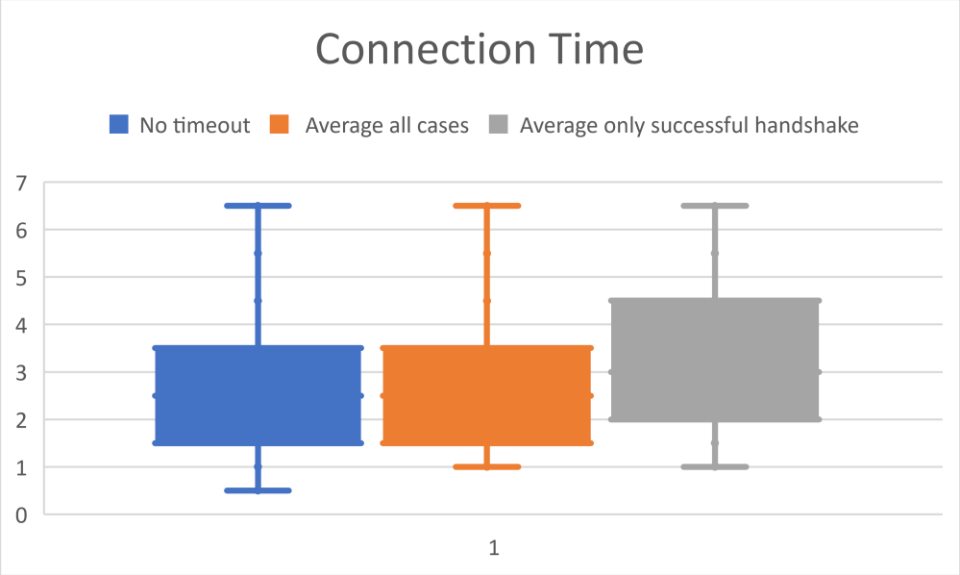

Figure 4: Connection Times for all IPs

Figure 4 compares the connection times in the server using three different ways to set the server timeout. The figure shows us that when using no server timeout to dequeue some of the unresolved request, the connection time throughout is almost the same as when the server timeout

is applied as all connections average. On the other hand, the method to apply only successful handshakes yield a pretty significantly longer connection time, which is due to the longer time it takes to dequeues the connections, but it's also due to the fact that we included the actual transmission of the data, which will take longer compared to SYN Flood attacks only under our stimulation environment.

4. Conclusion and future work

Through the comparison between the three different methods, we can say that under these circumstances of the stimulation environment, the method to dequeue every average time of all the successful handshakes and transmission would be the most ideal. It shows that we can transmit more files than not using a timeout method, and steadier than using an average of all cases including the attack times. For future work, the environment would be more beneficial if each of the IP are dynamically timed, so that we can include a more accurate connection times of the cases that were attacking our server.

References

"DDoS Attacks Target Multiple Games including Final Fantasy ...." 09 Oct. 2018, https://www.scmagazine.com/home/security-news/ddos-attacks-target-multiple-games-including-final-fantasy-xiv/.

"Cyberwarfare: US Launched DDoS Attacks Against North Korea." 02 Oct. 2017, https://wccftech.com/cyberwarfare-us-ddos-north-korea/.

TCP vs. UDP: The Difference Between them. n.d. 7 5 2020. <http://www.skullbox.net/tcpudp.php>.

Security.radware.com. 2020. What Is a Syn Flood? | Radware — Ddospedia. https://security.radware.com/ddos-knowledge-center/ddospedia/syn-flood/

Cloudfare, 2020. Three-Way Handshake in TCP Protocol. [image] Available at: <https://www.cloudflare.com/img/learning/cdn/tls-ssl/tcp-handshake-diagram.png>.

Kupreev, O., Badovskaya, E. and Gutnikov, A., 2020. Ddos Attacks in Q1 2020. [online] Securelist.com. Available at: <https://securelist.com/ddos-attacks-in-q1-2020/96837/>.

Lemon, Jonathan. "Resisting SYN Flood DoS Attacks with a SYN Cache." BSDCon. Vol. 2002. 2002.

Mirzaie, Sara, Alireza Karimi Elyato, and Mehdi Agha Sarram. "Preventing of SYN flood attack with iptables firewall." 2010 Second International Conference on Communication Software and Networks. IEEE, 2010.

Hang, Bo, and Ruimin Hu. "A novel SYN Cookie method for TCP layer DDoS attack." 2009 International Conference on Future Biomedical Information Engineering (FBIE). IEEE, 2009.

Martine Bellaïche and Jean-Charles Grégoire. 2012. SYN flooding attack detection by TCP handshake anomalies. Sec. and Commun. Netw. 5, 7 (July 2012), 709–724. DOI:https://doi.org/10.1002/sec.365