Revisiting Amplification over TCP: A Comparative Analysis with 2023 Metrics

Giorgos Koursiounis
Delft University of Technology
Delft, The Netherlands
G.Koursiounis@student.tudelft.nl

Nektarios Evangelou
Delft University of Technology
Delft, The Netherlands
N.Evangelou@student.tudelft.nl

Vyshnavi Molakala Narasimhalu
Delft University of Technology
Delft, The Netherlands
V.Molakalanarasimhalu@student.tudelft.nl

Konstantinos Andriopoulos
Delft University of Technology
Delft, The Netherlands
K.Andriopoulos@student.tudelft.nl

ABSTRACT

This paper studies the abuse potential of different TCP-based protocols to amplification attacks. These attacks occur when an attacker sends a small request to a server and receives a much larger response, thus amplifying the bandwidth of the attack. The study collected data on the response behavior of approximately 5.5 million web hosts who run HTTP, HTTPS, Telnet, and FTP protocols. We confirm the feasibility of the attack in the current era and underline that it can still cause damage, even though it requires more effort by the attackers. HTTP and Telnet protocols are the most vulnerable to an amplification attack. HTTP presents the highest amplification potential with an amplification factor of 595x, followed by Telnet with 141x, HTTPS with 106x, and FTP with 21x. Furthermore, our research demonstrates that the amplification factor drastically lowers when amplifiers receive more than one TCP SYN packet from the attackers. Given the results, TCP amplification attacks pose an existing but not severe threat to the internet infrastructure; however, administrators should take proactive measures to protect systems against related DDoS attacks.

1 INTRODUCTION

Distributed Denial-of-Service (DDoS) attacks have been known for many years and deployed to harm the use of services. The traditional DDoS attacks, such as TCP SYN or UDP flooding attacks, can be mitigated in numerous ways [11, 20, 21, 28, 29]. However, adversaries have developed a modern DDoS method called *amplification attack*. This amplification attack is a cyber attack aiming to magnify the traffic sent to a target system. It bounces traffic to the target system using intermediary systems such as servers called amplifiers/reflectors. During an amplification attack, the attacker sends requests to the amplifier, spoofing the source address of the victim system. The amplifier sends a more significant response to the victim system, which may overwhelm the target system and lead to a denial-of-service (DoS) attack. To execute an amplification attack successfully, the attacker must have many spoofed source IP addresses and access to many amplification vectors. Many amplification attacks are carried out using botnets, networks of compromised computers, or devices that the attacker can control. These attacks are perilous because they can generate a large volume of traffic with minimal effort on the attacker's part.

UDP-based protocols are the most susceptible to amplification attacks since they do not provide any verification mechanisms for communication. In contrast, TCP-based protocols utilize a three-way handshake to establish communication. Both parties exchange three sequential and equal-sized TCP packets to authenticate each other before transmitting any payload data. Packet sequence numbers are chosen randomly during the handshake; thus, a spoofed attacker cannot complete the handshake. Prior research has demonstrated that TCP can be abused for amplification attacks [8, 18, 19, 24, 26].

In this paper, we revisit the amplification over the TCP topic and perform a comparative analysis with 2023 metrics. We study the methodology of Kuhrer et al. [19] published in 2014 and evaluate the feasibility and efficiency of this attack in the present. In particular, we initially built a proof of concept to validate the possibility of the attack on a local environment. Next, we collect public data on potentially vulnerable web hosts and scan the internet using an open-source tool. We gather approximately 5.5 million IP addresses of reachable hosts on specific protocols such as HTTP, HTTPS, FTP, and Telnet. Then, we develop and deploy an amplification measurement infrastructure to measure the amplification potential of the hosts. Next, we perform post data-analysis of the collected response traffic to evaluate the success of this attack. Lastly, we send multiple TCP SYN packets to hosts and record their response behavior. We conclude that the amplification attack over TCP is feasible and identify around 3.6 amplifiers. HTTP and Telnet hosts are the most vulnerable to a TCP amplification attack. In fact, HTTP presents the highest amplification rate with a factor of 595x. Telnet follows with a factor of 141x, HTTPS with 106x and lastly FTP with 21x. Despite high individual amplification rates, our study shows that we do not approach the enormous 80,000x amplification described by previous research [19]. A reason could the deployment of countermeasures during recent years and the effort of the industry and academic research to mitigate DDoS attacks. We explore potential reasons in the next sections.

To summarize, our contributions are as follows:

- We build a proof of concept that demonstrates the feasibility of the amplification attack over TCP
- We randomly scan a part of the IPv4 address space for specific TCP-based protocols to identify hosts vulnerable to TCP amplification attacks

- We use the obtained IP addresses to send SYN packets and perform amplification measurements
- We apply post data-analysis on the retrieved data to check how much the amplification was.
- We evaluate our results with the results of Kuhrer et al. [19].

2 BACKGROUND AND RELATED WORK

2.1 TCP Amplification Background

Unlike UDP, TCP is not typically used for amplification attacks due to its connection-oriented nature [27]. The protocol involves a three-way handshake between the sender and receiver before data transmission. All segments have the same size, and no data bytes are sent until the handshake is complete. A flow control mechanism ensures that data is transmitted at a rate the receiver can handle. There is no feasible method to finish the handshake with IP-spoofed data since hosts select TCP sequence numbers randomly. Furthermore, even though it enables traffic reflection, it does not magnify the traffic. As a result, the victim does not theoretically receive more bytes than an attacker has generated. The attack surface is smaller and requires more resources than UDP. Hence, TCP forbids amplification attacks [18]. However, academic research and experiments have shown that TCP can be abused. This abuse is possible because servers can be incorrectly configured or deploy TCP re-transmission mechanisms.

Kuhrer [18] describes two scenarios of re-transmission due to packet loss, presented in figure 1. First, the spoofed attacker sends a SYN segment to the amplifier. In the first scenario, the amplifier keeps sending SYN/ACK segments to the victim continuously until it receives an ACK response or a certain threshold is met. The second scenario involves the victim responding with an RST segment which terminates the connection.

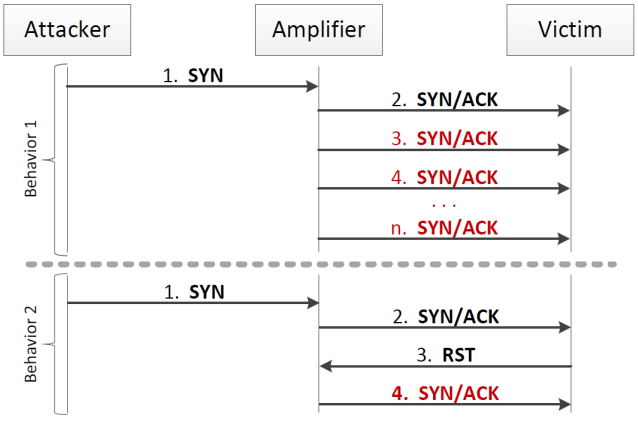

Figure 1: Amplification abuse in the TCP handshake [18]

From the attackers' perspective, exploiting TCP offers several advantages. First of all, it is challenging to identify attacks within regular traffic in a stream of TCP control segments. On the other hand, it is relatively easy for providers to create payload-based filters in UDP-based protocols. Additionally, service providers face difficulties filtering or restricting traffic based on TCP-based protocols (HTTP, FTP) compared to less known protocols (CharGen, QOTD) [18]. Such an action might affect the service's usability and block legitimate users. Furthermore, TCP is well established, and thousands of potential TCP amplifiers worldwide increase the costs and complexity of the bug discovery and fixing process. Hence, TCP-based amplification attacks can have a devastating impact.

2.2 Related Work

Our work is based on the analysis of TCP-based amplification attacks. Our paper discusses only related work about TCP-based amplification attacks. First, Kuhrer et al. [19] performed scans in the IPv4 address space for 13 standard TCP-based protocols, identifying hosts vulnerable to reflective amplification attacks. The scanning was done using a Linear Feedback Shift Register (LFSR) that generated a pseudo-random permutation of the IPv4 address space. They also limited the number of packets a network received quickly to avoid blacklisting [10]. The scanner sent a single SYN packet to each target host and recorded the corresponding responses. Furthermore, it did not complete the TCP handshake with a valid ACK segment and sent no replies using RST packets, which would cancel the connection. This incomplete handshake simulated the situation as the victim suffered network overload and could not reply with RST packets. The amplification factor was calculated by the number of layer-2 bytes sent by an amplifier divided by the size of the initial SYN packet of 54 bytes. The highest amplification was around 54,000x and 80,000x for the FTP and Telnet protocols.

Additional prior research by Rossow [26] demonstrated that the most popular UDP-based protocols of specific online video games, network services, P2P botnets, and file-sharing networks were vulnerable to Distributed Reflective Denial-of-Service (DRDoS). Millions of public servers may be employed as amplifiers, and bandwidth amplification of an attack could increase traffic by a factor of 4670.

The number of possible amplifiers for TCP- and UPD-based protocols was listed in Kuhrer's prior research [18]. The research also offered mitigation strategies to reduce UDP-based DDoS assaults that use susceptible NTP servers by 92 percent. They then assessed the TCP protocol amplifying flaws and demonstrated that attackers could use several hosts to produce a 20-fold amplifying effect.

Another research by Paxson [24] labeled the end hosts that responded to arbitrary TCP packets with SYN/ACK or RST segments as reflectors, which were abused using spoofing attacks. In such an attack, many spoofed TCP packets are transmitted to many reflectors, which forward the responses to a target host in the victim's network. While this attack reflects TCP traffic to the victim, no amplification is achieved. An attacker, thus, is required to generate an enormous number of spoofed packets to achieve a high impact on the victim's network.

Lastly, Bock et al. [8] demonstrated that several IP addresses provide amplification factors much more significant than 100 times. This research amplified network traffic using middleboxes that do not comply with TCP. Using a cutting-edge genetic algorithm, they maximized the effectiveness of TCP-based reflecting amplification attacks and multiple packet sequences. Those made network middleboxes react with considerably more packets than transmitted.

3 DATASETS

Our datasets consist of the following:

(a) IP address lists of live web hosts to be measured for amplification potential. The lists were obtained from publicly available sources [5], web-based scanning platforms [6] or generated through web scanning, using the masscan tool [15]. In total, we identify 5,444,472 target hosts. When we analyzed the response packets, it was shown that 4,511,790 hosts replied to our TCP SYN packets. These lesser hosts are expected since the internet is a living structure and target hosts might not be up by the time of the amplification measurement.

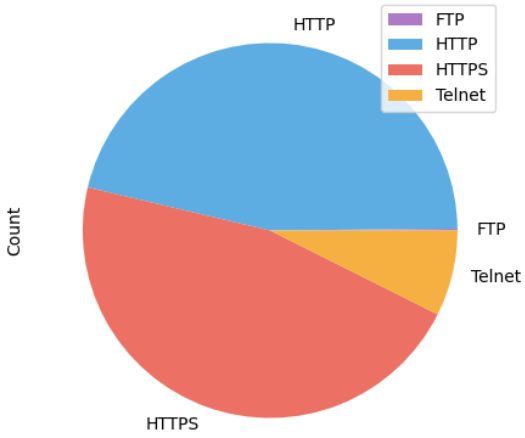

Figure 2: Distribution of protocols over the dataset

Chart 2 presents the distribution of protocols over our dataset's hosts. Specifically, our data include (i) 2,081,855 hosts running the HTTP protocol, (ii) 2,089,319 hosts running HTTPS, (iii) 333,285 hosts running Telnet, and (iv) 7,331 FTP hosts. Hence, our research involves an extensive study of the HTTP(S) protocols and a minor analysis of FTP and Telnet.

Moreover, we consider the distribution of the measured IP addresses over the internet to have a sufficient sample. Figure 3 presents a Hilbert curve heatmap of our dataset in the IPv4 address space. Given the Hilbert curve, we can conclude that our dataset covers the most significant part of the space.

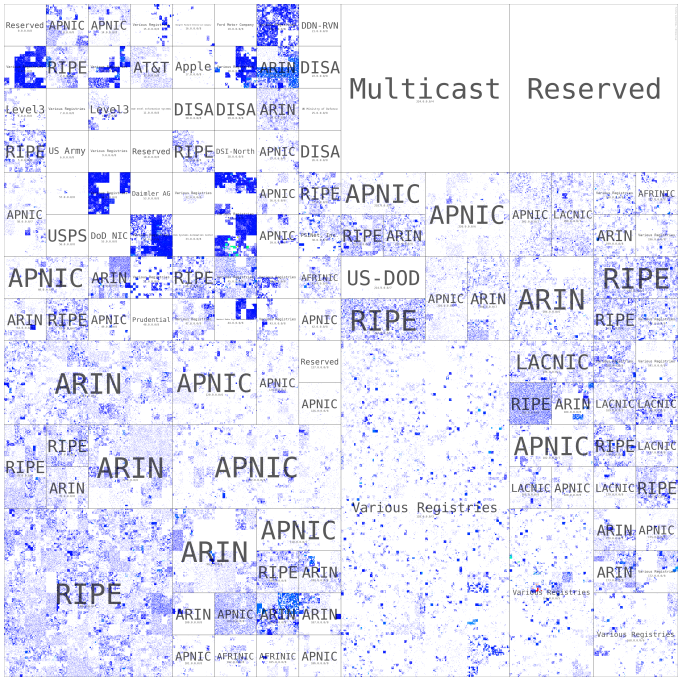

Figure 3: IPv4 Heatmap

(b) .pcap files that contain traffic responses generated by the hosts. Each file comprises the following columns: packet number, time lapsed since the start of capturing, source IP, source port, destination IP, destination port, the protocol used, packet length, and packet info

(c) .csv files generated by converting the .pcap files using the Wireshark tool [23]. These files are useful for post data-analysis in Python.

4 METHODOLOGY

This section describes our methodology for measuring the amplification over TCP in our study. The high-level approach to this research involved four steps (i) proof of concept, (ii) information gathering, (iii) amplification measurement, and (iv) post-data analysis.

4.1 Proof of Concept

In order to construct a proof of concept, we created a containerized environment using Docker and set up three different Ubuntu containers: an attacker, an Nginx server, and a victim. The attacker sends a TCP SYN request to the Nginx server spoofing the source IP address with the victim's address. The server operates as a reflector and responds with six packets to the victim, amplifying the attack. The OS defines the number of SYN/ACK retries. Linux configures it by `sysctl net.ipv4.tcp_synack_retries` to `net.ipv4.tcp_synack_retries = 5`. Also, we apply further configurations to the victim's IP tables to block any outgoing traffic and avoid sending an RST response. This configuration follows Kuhrer's approach [19].

More specifically, one of the main difficulties of a TCP amplification attack is that most victims reply to out-of-state SYN/ACK segments with an RST segment. Such an RST resets the connection and stops the server from sending any follow-up SYN/ACKs, thus

canceling the amplification. However, as mentioned by [19], the attacker might deploy a TCP amplification attack to a victim that already suffers from network overload and thus cannot reply with RST segments. Similarly, the attacker might target unassigned IP addresses in the victim's network so that no RST packets are generated to cancel the connections to overload the victim's network with traffic.

4.2 Information Gathering

The second step of our research included data collection from various sources. In particular, initially, we obtained a list of around 27,000 public DNS servers worldwide [5] and tested our amplification measurement infrastructure. Next, we queried Shodan.io [6] for hosts listening to ports 80 (HTTP) and 21 (FTP) and received around 20,000 IP addresses. This medium-scale dataset allowed us to validate our measurement infrastructure in real-time. Moving to a large scale, we deployed the open-source tool masscan [15] for host discovery in the entire IPv4 internet address space. We used masscan by scanning the internet for ports 80 (HTTP), 443 (HTTPS), and 23 (Telnet) for approximately 24 hours, which provided us with around 5 million IP addresses. In total, we managed to gather 5,444,472 IP addresses.

4.3 Amplification Measurement

We identified the target hosts and developed scanning infrastructure to generate and measure network traffic. Our approach follows the Producer-Consumer model:

- Producer: Python script that sends TCP SYN packets to each host in the target list. For efficiency, the function creates threads and puts all the host IPs into a queue. Each thread pops an IP from the queue and processes it until the list is empty.
- Consumer: Python script or tcpdump that sniffs for responses.

Both components are executed inside an Ubuntu 20.04 Docker Container with specific configuration settings. The container has a static IP address within a custom bridge network and has installed essential software such as network tools, tcpdump [14], Python, Scapy framework [25]. The complete configuration file can be found in our GitHub repository [4].

The Producer uses the Scapy framework to craft and send TCP SYN packets to each host in the target list. For efficiency, the script creates a pool of M threads and puts all the host IPs into a shared queue. Each thread pops an IP address from the queue and sends a TCP SYN packet to a destination host until the queue is empty.

For the Consumer, we provide two alternative options; a custom Python sniffer script where packet details are printed to standard output and tcpdump, which stores all the packets in a file. For our experiments, we used tcpdump because it is easy to use and allows us to generate .pcap files for storing and further processing the responses.

4.4 Post Data Analysis

We gathered all the .pcap files during the last part and converted them to .csv format using the Wireshark tool. Then, we analyzed the results in Python. This process comprised two parts; data preparation and data analysis. For the first part, we annotated each packet with the protocol used based on the port number (HTTP, HTTPS, Telnet, FTP). However, Shodan queries for HTTP and FTP returned many results in ports different than 80 and 21, respectively. Hence, we annotated this dataset based on the query type, not the port. We also discovered several duplicates between the medium-scale scans (Shodan and subnet scans) and the large-scale scans (masscan). We dropped duplicates from medium-scale scans and merged all the data in a single dataset.

5 RESULTS

In this section, we present the results from post data-analysis. To facilitate the interpretation of the results, we define two metrics introduced by Rossow [26]:

(1) Bandwidth Amplification Factor (BAF), as the sum of packet lengths of the host responses, i.e., the total length of responses in bytes, divided by the length of the sender's TCP SYN segment:

$$BAF = \frac{\sum bytes(\text{packet received from host})}{bytes(\text{TCP SYN segment sent to host})}$$

(2) Packet Amplification Factor (PAF), as the number of packets received from the host divided by the number of packets we sent:

$$PAF = \frac{\#\text{ packets received from host}}{\#\text{ packets sent to host}}$$

General Results. We measured the amplification over TCP for around 5.5 million web hosts out of the approximately 4.2 billion IP addresses worldwide. The latter includes reserved spaces that cannot be used [9]. Our results show that amplification attacks over TCP are feasible and can harm potential victims. We have identified many public servers that act as amplifiers, around 3.6 million, in such a small fraction of the IP address space.

Top Amplifiers. Table 1 illustrates the amplification results for HTTP, HTTPS, FTP, and Telnet protocols. BAF and PAF clearly show the extent of each protocol's abuse potential. We analyze the results based on the following four metrics: (i) average BAF (Avg BAF), (ii) average PAF (Avg PAF), (iii) maximum BAF (Max BAF), and (iv) maximum PAF (Max PAF).

As demonstrated in table 1, we observe that HTTP has the highest Max BAF at 639.07 and the highest Max PAF at 595, indicating that HTTP may be vulnerable to extreme levels of amplification in certain circumstances. Telnet has the second-highest Max BAF at 217.61 and Max PAF at 141, followed by HTTPS and FTP. Hence, HTTP and Telnet hosts are the most vulnerable to a TCP amplification attack.

FTP is the protocol with the highest Avg BAF, with an average of 4.85, followed by HTTP with 4.79 and HTTPS with 4.71. Telnet, on the other hand, presents the lowest Avg BAF (4.15), contradicting the findings of existing literature [18] [19].

Table 1: Top amplifiers per protocol

Protocol	Avg BAF	Avg PAF	Max BAF	Max PAF
HTTP	4.79	4.60	639.07	595
HTTPS	4.71	4.52	106.0	106
FTP	4.85	4.53	22.48	21
Telnet	4.15	3.93	217.61	141

Although SYN and SYN/ACK packets have roughly the same size, we notice that the average BAF is larger than the average PAF for all protocols. In such attacks, it is very likely that while the attacker sends a minimum length SYN packet, some routers on the network between the attacker and the victim may also modify the packet to add additional link information. One such piece of information is the Maximum Segment Size (MSS) TCP option which increases the absolute size of the traffic observed by the victim [1]. This option accounts for an additional 4 bytes per packet compared to what the attacker sent.

There are other TCP options that an attacker could use to produce larger SYN/ACKs, such as Timestamp (TS), Selective ACK (SAckOK), Window Scale (WScale), and TCP Fast Open (TFO) cookies [3]. However, it is important to note that we did not use any of the above for our experiments.

Protocol Analysis. Figure 4 presents the distribution of hosts per number of response packets for port 80 (HTTP). We note a large distribution from 1 to 595 reply packets. 72% of the measured hosts reply with four or more packets, whereas the percentage of hosts who reply with seven or more packets is 2.5%. Therefore, it is evident that most hosts send up to seven SYN/ACK packets. Additionally, it is notable that 53.9% of the HTTP hosts reply with exactly six packets, similar to our proof of concept.

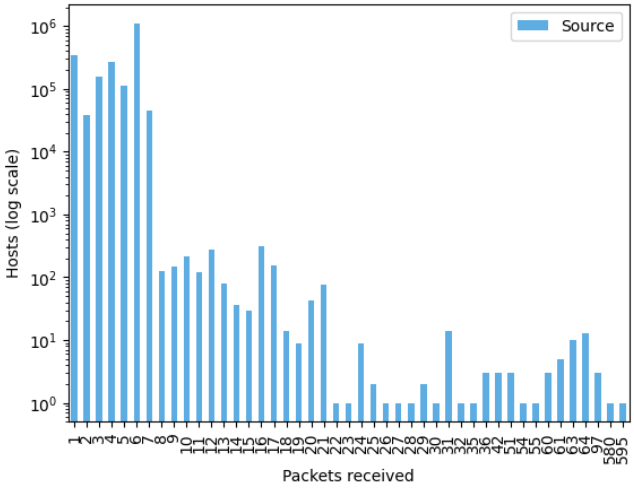

Figure 4: Distribution of hosts per amount of response packets for port 80 (HTTP)

For the HTTPS protocol, our results show that most amplifiers, around one million, reply with six SYN/ACK segments. Also, we observe that the vast majority (97.5% of all HTTPS servers) reply with at most 6 SYN/ACKs. We find one HTTPS server with PAF equal to 106 - the highest amount for this protocol - and about 300,000 servers that do not produce any amplification.

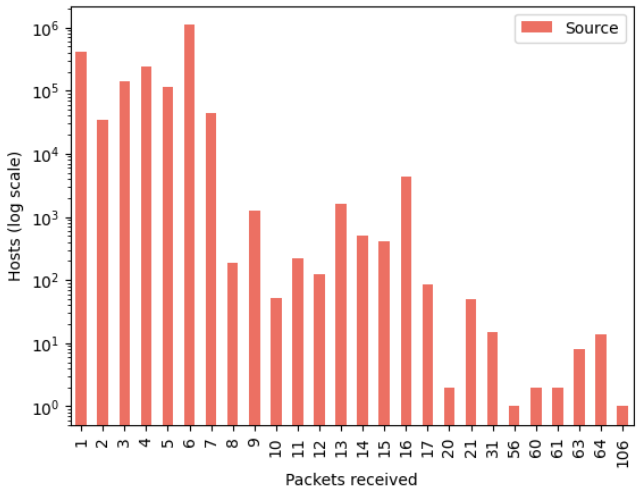

Figure 5: Distribution of hosts per amount of response packets for port 443 (HTTPS)

Figure 7 presents the distribution of hosts per number of SYN/ACK packets for port 23 (FTP). We observe a distribution of one up to 21 packets for FTP, which is relatively narrower than HTTP and HTTPS. The highest concentration of hosts (approximately 2,400) reply with exactly six packets. Furthermore, we observe that around 97% of all FTP servers reply with at most six SYN/ACKs. We find one FTP server with PAF equal to 21, the highest we observe, and 376 servers that reply with only one SYN/ACK packet.

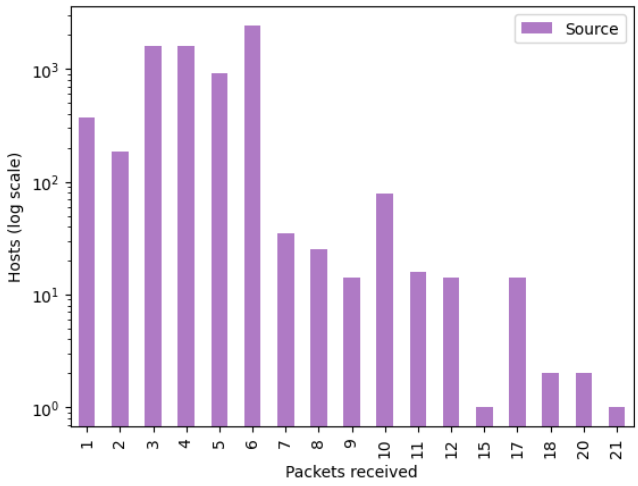

Figure 6: Distribution of hosts per amount of response packets for port 21 (FTP)

For Telnet, our results again show that most amplifiers (around 170,000) reply with six SYN/ACK segments. Furthermore, we observe that around 99% of all Telnet servers reply with at most six SYN/ACKs. It is essential to note that Telnet presents the second largest distribution (after HTTP) of one up to 141 SYN/ACK packets replied. We find one Telnet server with a PAF equal to 141, which is the highest we observe, and about 125,000 FTP servers that do not produce any amplification.

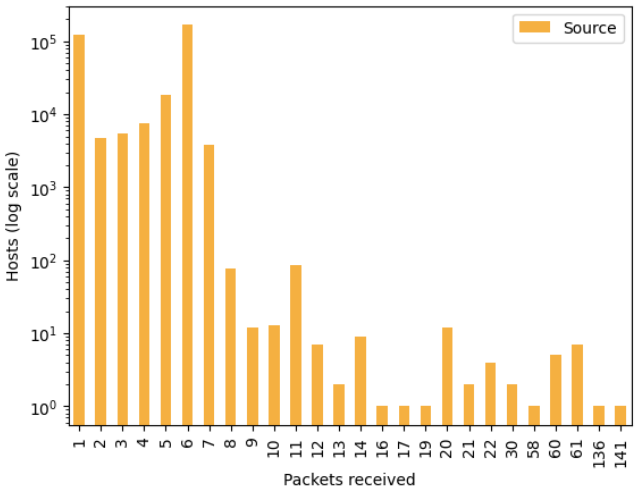

Figure 7: Distribution of hosts per amount of response packets for port 23 (Telnet)

Similarities among Protocols. We observe that the distribution of hosts per amount of response packets for all protocols converges to six. As discussed in the proof of concept, Linux configuration is set to six SYN/ACK replies by default [2]. Therefore, it is reasonable to assume that a significant portion of public servers that can be used for TCP amplification attacks may be running on Linux, and their default behavior can contribute to the prevalence of such attacks. However, there are also other factors to consider, such as the specific configuration of each server, the presence of firewalls or other security measures, and the monitoring and management practices of server administrators. These factors can significantly affect the susceptibility of a server to such attacks.

Summary. Tables 2 and 3 summarize the results we discussed above. Specifically, they present the percentage of amplifiers with PAF and the total response length above certain thresholds for each of the four protocols. These metrics are used to assess the potential amplification effect of live web hosts, which can be abused to impact the efficiency and security of network communications.

Table 2: Percentage of amplifiers per protocol with PAF above a certain threshold

Protocol	PAF ≥ 4	PAF ≥ 6	PAF ≥ 7	PAF ≥ 10
HTTPS	72.07	55.04	2.5	0.35
HTTP	83.60	74.33	2.28	0.06
Telnet	59.31	51.54	1.23	0.04
FTP	70.39	35.83	2.76	1.75

Table 3: Percentage of amplifiers per protocol with a total length of response (bytes) above a certain threshold

Protocol	len ≥ 100	len ≥ 200	len ≥ 300	len ≥ 350
HTTPS	80.46	72.07	55.05	2.50
HTTP	84.60	74.33	56.23	2.28
Telnet	62.38	59.31	51.56	1.25
FTP	94.87	70.39	35.86	2.80

According to Table 3, 80.46% of amplifiers for the HTTPS protocol produce replies with more than 100 bytes. As the threshold increases to 200 bytes, the percentage decreases to 72.07% and further decreases to 55.05% and 2.50% for thresholds of 300 and 350, respectively. Similar trends are observed for the HTTP and FTP protocols, where the percentage of amplifiers with a total length of response above the threshold decreases as the threshold increases. Interestingly, the Telnet protocol demonstrates fewer amplifiers for all four thresholds we test. Specifically, only 62.38% of the amplifiers respond with more than 100 bytes, and this percentage decreases to 59.31% for a threshold of 200, 51.56% for a threshold of 300, and 1.25% for a threshold of 350. These findings suggest that HTTPS, HTTP, and FTP protocols have a higher potential for amplification.

Protocols like HTTPS and HTTP are designed to support rich content and multimedia, which may result in larger response packets and higher PAF and BAF values [13]. Another reason these percentages differ per protocol is that there are different implementations of the same protocol with varying levels of amplification potential. Different web servers or FTP servers may have different ways of handling requests and generating responses, leading to differences in BAF values and the total size of the response. Additionally, configuration settings and optimizations in the implementation may also impact the amplification potential of the protocol [17].

The presence of security measures to avoid these reflection amplification attacks, such as rate limiting or filtering mechanisms, can also impact the amplification potential of a protocol. These security measures may restrict the rate or size of responses, leading to lower amplification potential.

Response of Top Amplifiers for two and three SYN Packets. All the amplification measurements performed until now are obtained by sending one SYN packet. Further evaluation is done by choosing the first 100 hosts with the highest PAF factor from our results. This list comprises 49 HTTP hosts, 36 HTTP hosts, and 15 Telnet hosts. We aim to evaluate the response of the top amplifiers when subjected to an increased attack intensity by receiving multiple SYN packets. We conducted two experiments (i) sending two SYN packets and (ii) sending three SYN packets. Table 4 illustrates the results obtained and the Max PAF per protocol. Our results show that the amplification rate decreases when we send multiple SYN packets.

Table 4: Highest PAF among the 100 top amplifiers of each protocol when sending 1, 2, and 3 TCP SYN packets

Protocol	1 SYN	2 SYN	3 SYN
HTTP	595	9	6
HTTPS	106	4	6
Telnet	141	3	6

We aim to understand the amplification attacks' scalability and effectiveness with multiple packets. If a single host can significantly amplify the attacks with multiple SYN packets, this highlights the potential for more impactful DDoS attacks using fewer resources. Nonetheless, lower amplification rates signify a decrease in the attack's intensity and impact. Thus, the results emphasize the need

for defense mechanisms that protect against single-packet rather than multi-packet amplification attacks.

The decrease in the amplification factor when sending multiple SYN packets to the same set of amplifiers could be due to state table saturation. This phenomenon is well-documented in the scientific literature and can significantly impact the effectiveness of TCP amplification attacks. One study [30] found that state table saturation can lead to a significant reduction in amplification factor, as well as an increase in response time and packet loss. Another study [7] showed that using multiple source IP addresses could help mitigate the effects of state table saturation and increase the effectiveness of the attack.

Another possible reason for the decrease in the amplification factor could be the amplifiers' behavior. Specifically, hosts might be designed to limit the number of responses they generate for a given IP address to prevent them from being exploited for amplification attacks.

Overall, it is essential to note that although the PAF decreases significantly when one attacker sends multiple SYN packets, the attack can still be very impactful if the number of attackers increases.

6 DISCUSSION

6.1 Discussion and Mitigation Techniques

The results of our study show that TCP amplification attacks are still feasible and can be carried out in the current era. Our analysis confirms previous research findings that TCP amplification attacks are not as efficient as UDP amplification attacks but can still damage targeted systems. The deployment and execution of this attack requires also more preparation. That is because victims might reply with RST packets terminating prematurely the amplification attack. Moreover, since PAF decreases significantly when an attacker sends multiple TCP SYN segments, thus an attack requires more attackers (or botnets) to be successful.

We identified several vulnerable hosts to perform the TCP amplification attack successfully. Especially the significant amplification rate for the HTTP protocol was unexpected as previous research [19] demonstrated the most considerable amplification rate for Telnet and FTP-related hosts.

On the other hand, we observed relatively lower amplification rates for FTP and Telnet hosts. One potential reason is that both protocols have been gradually phased out as they have been deprecated. Some hosts still use FTP, an outdated and insecure protocol. Many modern services and applications have moved to use more secure alternatives such as SFTP (Secure File Transfer Protocol) or SCP (Secure Copy Protocol), which use SSH (Secure Shell) [31] for encryption and authentication. Many web servers and applications now use HTTP or HTTPS for file transfer instead of FTP. There have also been implementations of various countermeasures on most hosts to mitigate the effects of amplification attacks. Rate limiting and TCP SYN Cookies [12] effectively mitigate TCP amplification attacks. Rate limiting involves restricting the rate of incoming traffic to a certain threshold to prevent the network from being overwhelmed by a large number of requests [22]. TCP SYN Cookies, on the other hand, is a security mechanism used to prevent SYN flood attacks [16]. It involves adding a cryptographic token to the initial SYN/ACK packet, which is used to verify the legitimacy of subsequent requests. As more hosts implement these countermeasures, the efficiency of TCP amplification attacks will likely decrease. However, it is essential to note that the TCP amplification attack can still pose a significant threat, as not all hosts have implemented these countermeasures.

Furthermore, attackers can use multiple attack vectors simultaneously to carry out the attack, increasing the overall amplification factor. Therefore, organizations should remain vigilant and implement appropriate measures to protect their systems against TCP amplification attacks. Our analysis demonstrates the importance of continued research on DDoS attacks and the need for proactive measures to prevent such attacks. It also highlights the importance of regularly updating security measures to ensure they remain effective against the latest attack techniques.

At this point, it is crucial to outline the limitations of our research. First, we focus our research on the IPv4 address space; thus, the findings may change for IPv6 addresses. Secondly, given their significance, we study only the most famous and used TCP-based internet protocols and do not perform exhaustive research on all the available protocols. Lastly, the research only examines a portion of the internet at a particular time and does not consider configuration changes in the scanned web hosts after the data were obtained.

6.2 Future Work

For future work, scanning a larger sample of internet hosts and the IPv6 address space might provide a more comprehensive picture of the current state of TCP amplification attacks. Furthermore, further research can be conducted regarding the operating systems of the amplifiers. Our analysis shows that the operating system plays an essential factor in sending SYN/ACK replies. Some OS may have different default configurations and behaviors that could impact their susceptibility to TCP amplification attacks. Lastly, scanning more FTP hosts would be valuable for better insight into the protocol's abuse potential.

7 CONCLUSION

In this paper, we revisited the topic of amplification attacks over TCP and performed a comparative analysis with 2023 metrics. We have demonstrated the feasibility of the attack by building a proof of concept and scanning the IPv4 address space to identify vulnerable hosts. Our results show that TCP amplification attacks are still doable. We studied the behavior of about 5.5 million web hosts and identified around 3.6 amplifiers. HTTP and Telnet hosts are the most vulnerable to a TCP amplification attack. However, the level of amplification may be lower than in previous research due to countermeasures implemented by hosts. We found a 595x maximum amplification factor for the HTTP protocol, which is much less than the approximate 80,000x maximum amplification factor for Telnet, as shown in Kuhrer's research. This is probably because many hosts have phased out the protocol, which has been deprecated since the start of our research.

Our research supports that TCP amplification attacks do not pose a severe threat to the internet infrastructure. Given the nature of the TCP handshake, it is challenging to deploy these attacks on a large scale and cause harm. However, network administrators should take the necessary measures to prevent them.

REFERENCES

[1] 2012. TCP Options and Maximum Segment Size (MSS). https://datatracker.ietf.org/doc/html/rfc6691. (2012).

[2] 2020. The Linux Kernel Archives. https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt. (2020).

[3] 2022. Transmission Control Protocol (TCP) Parameters. https://www.iana.org/assignments/tcp-parameters/tcp-parameters.xhtml. (2022).

[4] 2023. Hacking Lab Github Repository. (2023). https://github.com/gkoursiounis/hacking-lab-2023

[5] 2023. Public DNS server list. (2023). https://public-dns.info/

[6] 2023. Shodan Search Engine. (2023). https://www.shodan.io/

[7] Olivier Billet and Marc Dacier. 2015. Amplification Attacks: Current Trends and Mitigation Techniques. *IEEE Communications Surveys & Tutorials* 17, 1 (2015), 446–465.

[8] Kevin Bock, Abdulrahman Alaraj, Yair Fax, Kyle Hurley, Eric Wustrow, and Dave Levin. 2021. Weaponizing Middleboxes for TCP Reflected Amplification.. In *USENIX Security Symposium*. 3345–3361.

[9] Laura DeNardis. 2009. *Protocol politics: The globalization of Internet governance*. Mit Press.

[10] Zakir Durumeric, Eric Wustrow, and J Alex Halderman. 2013. ZMap: Fast Internet-wide Scanning and Its Security Applications.. In *USENIX Security Symposium*, Vol. 8. 47–53.

[11] Wesley Eddy. 2007. *TCP SYN flooding attacks and common mitigations*. Technical Report.

[12] Wesley Eddy. 2007. TCP SYN flooding attacks and common mitigations. (2007).

[13] Roya Ensafi, David Fifield, Philipp Winter, Nick Feamster, Nicholas Weaver, and Vern Paxson. 2015. Examining how the great firewall discovers hidden circumvention servers. In *Proceedings of the 2015 Internet Measurement Conference*. 445–458.

[14] Piyush Goyal and Anurag Goyal. 2017. Comparative study of two most popular packet sniffing tools-Tcpdump and Wireshark. In *2017 9th International Conference on Computational Intelligence and Communication Networks (CICN)*. IEEE, 77–81.

[15] Robert David Graham. 2014. MASSCAN: Mass IP port scanner. *URL: https://github.com/robertdavidgraham/masscan* (2014).

[16] Bo Hang, Ruimin Hu, and Wei Shi. 2011. An enhanced SYN cookie defence method for TCP DDoS attack. *Journal of Networks* 6, 8 (2011), 1206.

[17] Xin Zhe Khooi, Levente Csikor, Dinil Mon Divakaran, and Min Suk Kang. 2020. DIDA: Distributed in-network defense architecture against amplified reflection DDoS attacks. In *2020 6th IEEE Conference on Network Softwarization (NetSoft)*. IEEE, 277–281.

[18] Marc Kührer, Thomas Hupperich, Christian Rossow, and Thorsten Holz. 2014. Exit from Hell? Reducing the Impact of Amplification DDoS Attacks. In *23rd USENIX Security Symposium (USENIX Security 14)*. USENIX Association, San Diego, CA, 111–125. https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/kuhrer

[19] Marc Kührer, Thomas Hupperich, Christian Rossow, and Thorsten Holz. 2014. Hell of a Handshake: Abusing TCP for Reflective Amplification DDoS Attacks.. In *WOOT*.

[20] Prashant Kumar, Meenakshi Tripathi, Ajay Nehra, Mauro Conti, and Chhagan Lal. 2018. SAFETY: Early detection and mitigation of TCP SYN flood utilizing entropy in SDN. *IEEE Transactions on Network and Service Management* 15, 4 (2018), 1545–1559.

[21] Manisha Malik, Maitreyee Dutta, et al. 2017. Contiki-based mitigation of UDP flooding attacks in the Internet of things. In *2017 International Conference on Computing, Communication and Automation (ICCCA)*. IEEE, 1296–1300.

[22] Jarmo Mölsä. 2004. Effectiveness of rate-limiting in mitigating flooding DOS attacks.. In *Communications, Internet, and Information Technology*. Citeseer, 155–160.

[23] Angela Orebaugh, Gilbert Ramirez, and Jay Beale. 2006. *Wireshark & Ethereal network protocol analyzer toolkit*. Elsevier.

[24] Vern Paxson. 2001. An Analysis of Using Reflectors for Distributed Denial-of-Service Attacks. *SIGCOMM Comput. Commun. Rev.* 31, 3 (jul 2001), 38–47. https://doi.org/10.1145/505659.505664

[25] R Rohith, Minal Moharir, G Shobha, et al. 2018. SCAPY-A powerful interactive packet manipulation program. In *2018 international conference on networking, embedded and wireless systems (ICNEWS)*. IEEE, 1–5.

[26] Christian Rossow. 2014. Amplification Hell: Revisiting Network Protocols for DDoS Abuse. https://doi.org/10.14722/ndss.2014.23233

[27] Christian Rossow. 2014. Amplification Hell: Revisiting Network Protocols for DDoS Abuse.. In *NDSS*. 1–15.

[28] Zi-Yang Shen, Ming-Wei Su, Yun-Zhan Cai, and Meng-Hsun Tasi. 2021. Mitigating SYN Flooding and UDP Flooding in P4-based SDN. In *2021 22nd Asia-Pacific Network Operations and Management Symposium (APNOMS)*. IEEE, 374–377.

[29] Nguyen Ngoc Tuan, Pham Huy Hung, Nguyen Danh Nghia, Nguyen Van Tho, Trung V Phan, and Nguyen Huu Thanh. 2019. A robust TCP-SYN flood mitigation scheme using machine learning based on SDN. In *2019 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, 363–368.

[30] Andreas Francois Vermeulen. 2012. The Amplification of TCP SYN Flood Attacks. In *2012 International Conference on Cyber Security, Cyber Warfare and Digital Forensic (CyberSec)*. IEEE, 1–6.

[31] Tatu Ylonen. 1996. SSH - Secure Login Connections over the Internet. (1996), 37–42.