From Ripple to Waves: DNS Amplification Attacks

Andreoli C. • Ligari D. • Alberti A. • Scardovi M. • Intini K.[1]

[1]*Department of Computer Engineering - Data Science, University of Pavia, Italy*
Course of Enterprise digital infrastructure

Abstract

Only two levels of sectional headings, `\section` and `\subsection`, should be used. Ad nemo aut quae dolores nesciunt reprehenderit occaecati. Optio distinctio at aliquam odit dolores laudantium. Illum et qui iste et laudantium dolorum. Nihil quis qui at quia alias. Quisquam ea sit aspernatur. Labore at hic voluptas cumque eum officia repellat.

***Keywords*—** DNS server • DDoS Attack • Wireshark • Dig • Top • Ping

dizdar_dns_2021 dnsbaseddos alleviatingimpact taylor_four_2021 DNSattackstype DDoSthreatreport Devi_2015 anycast machinelearning createDNS

1 DNS based DDoS attacks

The DNS service plays a crucial role in the Internet infrastructure as it is relied upon by almost every internet service. Any compromise to the DNS service could have significant consequences, leading to a disruption of numerous networked applications. While the primary focus of DNS design is to provide fast responses, its emphasis on security is relatively limited, making it susceptible to various forms of attacks. According to a recent report by Cloudflare (**DDoSthreatreport**), almost a third of all DDoS attacks are DNS-based. These attacks primarily fall into three categories: *DNS query flood*, *TCP flood*, and *DNS reflection and amplification*.

DNS query flood

This is a direct attack, conducted with the intention of overwhelming the target DNS server by exhausting its available resources, eventually causing it to become unresponsive. This type of attack involves the attacker leveraging a botnet, a network of compromised devices, to send a massive volume of DNS queries to the target server.

When the victim of the attack is a recursive DNS server, the requests are structured in such a way that the server doesn't have the requested records cached. As a result, the server is forced to perform recursive queries to obtain the requested information and provide responses to the attacker, using eventually all its available resources.

A variant of this attack, known as **DNS water torture** attack, specifically targets authoritative DNS servers. In this scenario, the attacker floods the server with an enormous number of properly constructed queries. These latter consist of two parts: the domain of the victim authoritative server and a random string that ensures the fully qualified domain name (FQDN) does not exist. As a result, the recursive nameserver initiates a search that eventually reaches the targeted authoritative server. The authoritative server, upon realizing that the FQDN does not exist, sends an NXDOMAIN response. All of these queries travel to the authoritative server, overwhelming its resources as it is forced handling each request.

The DNS query flood attack is particularly effective against smaller DNS servers that may have limited resources to handle the high volume of queries, making them more vulnerable to disruption.

TCP flood

Another form of attack that aims to exhaust server resources is the TCP flood attack. In this type of attack, the attacker inundates the target server with a large number of TCP connection requests but does not close these connections. As a result, the server is compelled to allocate resources to handle each incoming TCP connection. As the number of connections grows, if the attack is successful the server becomes unresponsive to legitimate users.

DNS reflection and amplification

DNS reflection and amplification is an indirect attack strategy that aims to consume the target network's band-

width. The attack begins with the attacker spoofing the IP address of the target. Subsequently, numerous queries are sent to a DNS server, utilizing the spoofed IP address as the source address. In this scenario, the spoofed IP address receives the responses from the DNS server, which is the *reflection* aspect of the attack. To amplify the impact, the queries are crafted in a way that the responses from the DNS server are significantly larger in size. This *amplification* aspect ensures that the target's bandwidth becomes overwhelmed, while the attacker only expends minimal resources. The effectiveness of this attack is determined by the Amplification Factor (AF), which is calculated by comparing the size of the response to the size of the query. To achieve a high amplification factor, attackers often perform a type ANY query, which provides a substantial amplification effect.
Since according to the already cited report (**DDoSthreatreport**) the most common type of attack is the DNS reflection and amplification, in this project we focus on this type of attack.

2 The mDNS protocol

The Multicast DNS (mDNS) protocol is a decentralized and self-organizing network service discovery mechanism. It allows devices on a local network to discover and communicate with each other without the need for a centralized DNS server. Unlike traditional DNS, which relies on centralized servers to resolve domain names to IP addresses, mDNS operates within a single local network and enables devices to discover each other's services and addresses.

mDNS utilizes multicast packets to communicate. Multicast is a form of communication where a single packet is sent to multiple recipients simultaneously. In the case of mDNS, devices interested in service discovery listen to a specific multicast address (IPv4: 224.0.0.251, IPv6: FF02::FB) and respond accordingly.

2.1 Packets structure

In the mDNS protocol, communication is done using multicast packets. These packets have a specific structure, consisting of header fields and data sections. The header contains information such as the source and destination IP addresses, port numbers, and protocol version. The data sections carry the actual content of the mDNS messages.

2.2 Queries

mDNS queries are used to discover services and devices on the local network. A query packet is sent to the multicast address, and devices that provide the requested service respond with a unicast packet containing the necessary information. Queries can be specific to a particular service or can be a general request for all available services.

Table 1. mDNS Query section fields

Field	Description	Length bits
QNAME	Name of the node to which the query pertains	Variable
QTYPE	The type of the query, i.e. the type of RR which should be returned in responses.	16
UNICAST-RESPONSE	Boolean flag indicating whether a unicast-response is desired	1
QCLASS	Class code, 1 a.k.a. "IN" for the Internet and IP networks	15

2.3 Resource Records

Resource records in mDNS provide information about services, devices, and other network resources. They contain data such as IP addresses, service names, port numbers, and other attributes. Resource records are used in both query and response packets to facilitate service discovery and communication.
There are different types of resource records, including:

- Address (A) Record: Maps a hostname to an IPv4 address.
- AAAA Record: Maps a hostname to an IPv6 address.
- Service (SRV) Record: Provides information about a particular service, such as the port number and the hostname offering the service.
- Pointer (PTR) Record: Associates a name with a resource record, typically used for service discovery.
- Text (TXT) Record: Carries additional information about a service, such as descriptive text or configuration parameters.

Table 2. **mDNS Resource Record fields**

Field	Description	Length bits
RRNAME	Name of the node to which the record pertains	Variable
RRTYPE	The type of the Resource Record	16
CACHE-FLUSH	Boolean flag indicating whether outdated cached records should be purged	1
RRCLASS	Class code, 1 a.k.a. "IN" for the Internet and IP networks	15
TTL	Time interval (in seconds) that the RR should be cached	32
RDLENGTH	Integer representing the length (in octets) of the RDATA field	16
RDATA	Resource data; internal structure varies by RRTYPE	Variable

3 Scripts

Spoofing

In order to conduct a Distributed Denial of Service (DDoS) attack, we utilized a custom script based on the *Scapy* library. The script performed a ping sweep on a specific network range, allowing us to identify active hosts within the target network. By leveraging the Scapy library's packet crafting capabilities, ARP request packets have been sent out with a broadcast destination MAC address to ensure their delivery to all hosts. By analyzing the responses received within a specified timeout period, the IP addresses of the active hosts have been extracted. This information facilitated the identification of potential targets for subsequent stages of the DDoS attack. This technique served as an initial reconnaissance step in the attack, providing valuable insights into the network's composition and active hosts that could be further exploited to disrupt the target's services.

Dos Script

To execute a Distributed Denial of Service (DDoS) attack, a customized script based on the *Scapy* and *dnspython* libraries has been exploited. The primary objective was to flood a target host with a massive number of DNS query packets. By utilizing the script's command-line arguments, the spoofed IP address that would appear as the source of the attack can be specified. This technique aimed to deceive the target and potentially implicate the spoofed IP in the attack. The script allowed to configure various parameters such as the target DNS server's IP address and port number, the domain name to query, and the DNS flags to manipulate. By manipulating these flags, the DNS message may be customized according to the attack goals. Furthermore, it is possible to specify the number of threads to utilize, each responsible for generating a specific number of DNS requests. This multi-threaded approach amplified the impact of the attack by concurrently inundating the target with a high volume of DNS queries. The cumulative effect was an overwhelming amount of traffic directed towards the specified spoofed IP, resulting in a disruption of its normal operations and potential denial of service.

4 Experimental setup

To ensure the success of the project, it is essential to establish a clear methodology and employ a well-defined set of tools. This section aims to provide a detailed explanation of the methodologies utilized to accomplish the project's objectives.

The selected methodologies encompass a systematic approach that allows for the accurate replication of a DDoS attack while maintaining ethical considerations and minimizing the potential impact on live networks. These methodologies were carefully chosen to ensure the reliability and validity of the experimental results. The methodological approach used is the following:

Why

The objective of this study is to assess the impact of a DoS attack, exploiting the DNS protocol and monitor the reachability of the targeted device and other network nodes. By simulating realistic attack scenarios, this study aims to understand the vulnerabilities within the DNS protocol, evaluate network resilience, and identify potential countermeasures.

Which/Who

The chosen target for the DDoS attack is a laptop (HP ENVY x360), which acts as a host for a DNS server running BIND9. Four laptops are utilized as vantage points to initiate the attack.

What
The selected metrics encompass the evaluation of response time for each ICMP or DNS message transmitted, accounting for potential timeouts. Additionally, the simulation involved monitoring the CPU and memory utilization of the DNS server.
These measurements were conducted across various types of DNS requests to assess the attack's effectiveness in terms of the amplification factor. It's worth noting that the server was configured to hold a distinct number of records for each request type, in order to change the response size accordingly.

Where
The vantage point for the simulation was a MacBook Air positioned within the network, acting as a passive observer during the attack.
To prioritize the security and stability of both public networks and devices, the DDoS attack simulations were carried out within a local area network (LAN) environment. This LAN was deliberately isolated from the Internet, preventing any potential impact on external systems.

5 Tools used

Ping
Ping is a network utility tool used to test the connectivity between two networked devices. It sends a small packet of data to a specific IP address or hostname and measures the time it takes for that packet to be received and returned. The result shows the round trip time (RTT), as well as the number of packets sent and received, and any packet loss that may have occurred.

Dig
The dig command (short for "domain information groper") is a popular network administration tool used to perform DNS queries. It allows users to perform DNS lookups to check DNS records and obtain information about DNS configurations. It is a versatile tool that allows users to specify different query types, such as A, AAAA, MX, TXT, and others.

Top
The top command provides real-time monitoring of system resources, such as CPU usage, memory utilization, running processes, and more. When executed, the top command displays an interactive, dynamic table that continually updates, allowing users to view the current state of their system and identify any processes consuming excessive resources.

Wireshark
Wireshark is a open-source network protocol analyzer. It is designed to capture, analyze, and display network traffic in real-time. Wireshark allows users to inspect and interpret the data packets transmitted over a network, providing detailed information about the communication between different devices.

6 DNS Server

For the simulation of the DDoS attack, a dedicated DNS server was established on a laptop using Ubuntu 24.04 LTS operating system. The laptop is an HP ENVY x360, with 12 GB of RAM and 4 core
To create the DNS server, the open-source software BIND9 was employed. This widely adopted DNS server software offers robust functionality and configurability. The server was specifically configured to act as the authoritative server for the domain name "ediproject.com." By assuming authority over this domain, the DNS server can respond to DNS queries and provide legitimate DNS responses during the simulation.
To ease the attack simulation, the DNS server was configured to respond to all DNS queries with the same LAN, and no security measures were implemented. To mimic the characteristics of a genuine DNS server, a total of seven NS records and five MX-type records were meticulously added to the DNS server's configuration. The inclusion of these records introduces diverse amplification factors, enabling the analysis of the DDoS attack's effectiveness based on these factors. By incorporating varying NS and MX records, the simulation can reflect the behavior of a real DNS server and offer insights into the impact of different amplification configurations on the severity of the attack.

7 Experimental Results

Amplification
As mentioned before, the term "DNS reflection and amplification attack" derives from the two key elements involved in the attack methodology. In particular, to achieve amplification the attacker queries the DNS server, which

will reply with larger responses than the original requests. In our project, we conducted various tests to explore different amplification factors based on the DNS request used. We have compiled a summary table showcasing the types of DNS requests used, the corresponding response dimensions, and the amplification factors achieved. The table, shown below, provides a comprehensive overview of these metrics.

Type	Request	A	MX	NS	ANY
Dimension (bytes)	74	108	306	330	540
Amplification Factor	-	1.46	4.14	4.46	7.30

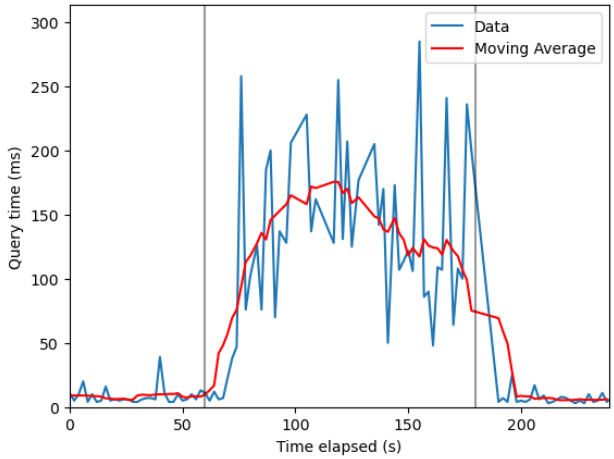

Figure 1. Evolution of the query time over the course of the test with the records of type ANY. The moving average better shows how the latency changes.

To gain deeper insights into the impact of the DNS reflection attack, it was essential to explore the actual data and analyze the timings of different types of requests. Figure 2 showcases the boxplots of query times, illustrating the system under attack with different record types in comparison to the baseline case.

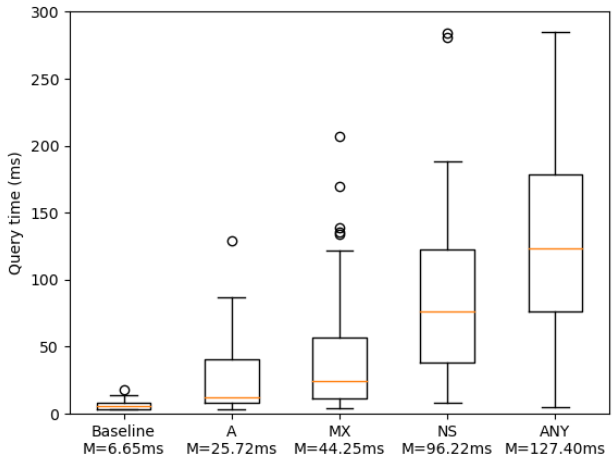

Figure 2. Boxplots of the query times with the system under attack with different record types, compared to the baseline case.

Effects on the Target

To assess the effects of the DNS reflection attack, various metrics of the targeted system were measured with regards to performance and network aspects. Regarding the network, the target was used as the primary vantage point, and latency was measured using the ping and dig command tools. The measurement process was divided into three sections to analyze the performance differences. Initially, measurements were taken for a minute without any attack. This established a baseline for normal system performance. Subsequently, a two-minute attack was executed, involving the transmission of 10,000 packets per second to the DNS server by multiple attackers. Finally, an additional minute of measurements without any attack was conducted to detect any potential lingering disturbance effects.

The latency graph related to DNS queries presented in Figure 1 visually demonstrates the impact of the attack over time. To improve visualization, a moving average of the latency time was applied, revealing a clear trend of increased latency during the attack. This trick becomes particularly helpful considering the inherent instability of latency measurements.

By comparing the mean response times during the attack with those observed in the baseline period, we could quantify the magnitude of the latency increase caused by the attack.

The results demonstrated that a larger amplification factor corresponds to higher latency. Even in the case of record type A, which typically has a small amplification factor, a noticeable effect on the query time was observed.

Remarkably, records of type MX and NS, despite having similar amplification factors (both around 300), exhibited distinct effects on latency. The mean query time for MX records was found to be approximately 44 ms, while NS records had a mean of 96 ms. This discrepancy can be attributed to the resource allocation or in general to the configuration of the DNS server handling the requests.
Furthermore, certain cases surpassed the threshold of 100 ms, particularly in the case of the RR ANY record. This indicates that such an attack would have a noticeable impact on the targeted system, for example during web browsing activities. Latencies exceeding this threshold can result in perceptible delays and interruptions, potentially compromising the overall user experience.
This analysis provided valuable information on the performance degradation experienced by the targeted system under the influence of the DNS reflection attack.
In addition to analyzing the query times using DNS-related tools, we also conducted a similar analysis using the ping command-line tool. The purpose of this analysis was to assess the impact of the DNS reflection attack on overall system latency, beyond just the DNS requests.

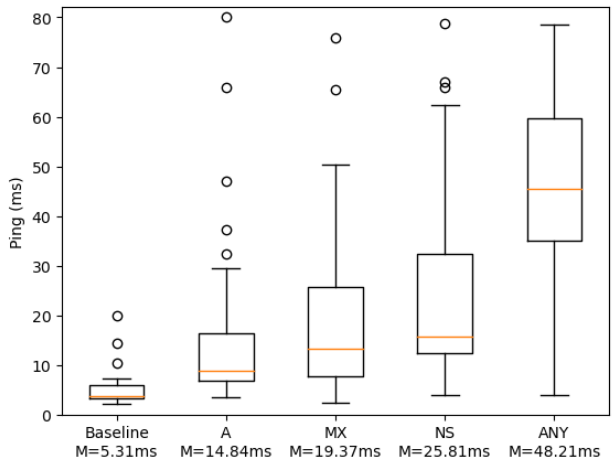

Figure 3. Boxplots of the ping times with the system under attack with different record types, compared to the baseline case.

As shown in Figure 3 the results obtained from the ping analysis revealed that the latency did not increase significantly during the attack. The highest mean latency observed was approximately 48 ms for the RR of type ANY. This finding suggests that the DNS reflection attack primarily affects the DNS requests themselves, rather than causing widespread disruption to the entire system.
During the attacks we also examined the effects of the DNS reflection attack on system resources, specifically RAM and CPU utilization. Our analysis aimed to determine whether the attack had any noticeable impact on these critical resources.
Surprisingly, the results revealed that the attack had no significant impact. The resource usage remained within normal ranges, comparable to the baseline measurements taken during the pre-attack phase.

Server Performance

The other vantage point of the experiments was the DNS server within the local network. Specifically, the analysis focuses on the CPU and RAM usage to gain insights into the server's behavior during the DNS reflection attacks.

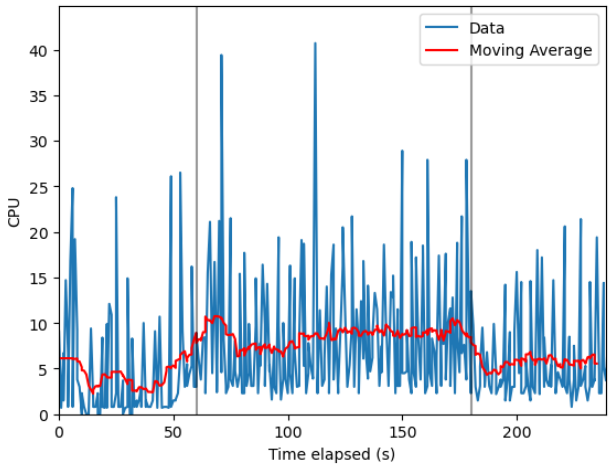

Figure 4. Evolution of the DNS server CPU percent used over the course of the attcak with the records of type ANY. The moving average makes the results more clear.

Figure 4 presents the CPU usage over time, utilizing a moving average to mitigate the volatility of the measurements. Notably, during the attacks, a slight increase in CPU usage is observed, albeit by a marginal 2 to 4%. This modest change indicates that the CPU's workload experiences a minimal impact during the DNS reflection attacks. While the increased query traffic imposes additional computational demands, the server's CPU manages to handle the heightened workload without significant strain.

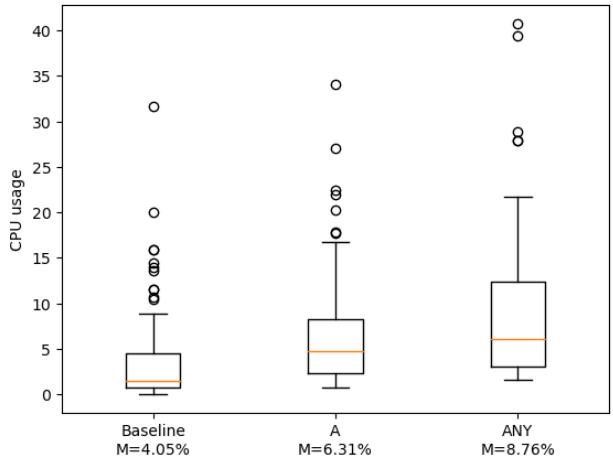

Figure 5. Boxplots of the CPU usage with the system under attack with different record types, compared to the baseline case.

Moving on to the RAM usage, the data reveals a periodic pattern, as shown in Figure 6, suggesting the presence of regular garbage collection or memory cleaning processes occurring at intervals. Despite these oscillations, the overall mean value around which the fluctuations occur remains relatively constant. This indicates that the DNS reflection attacks do not significantly affect the server's RAM usage.

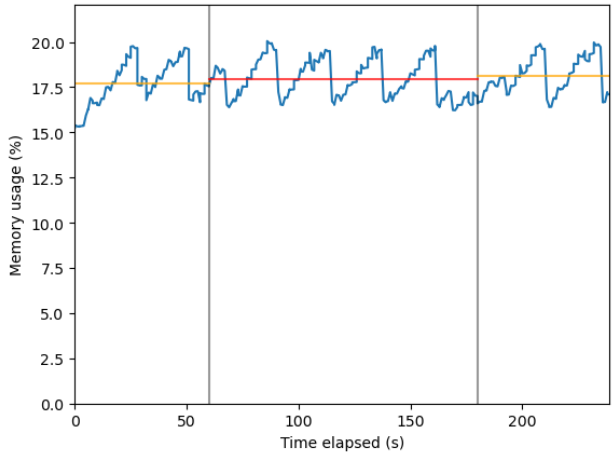

Figure 6. Memory usage of the server during the attack. An oscillating pattern can be observed.

The marginal change in CPU usage, coupled with the stable RAM usage, can be attributed to the server's efficient resource management mechanisms. These mechanisms enable the server to effectively handle the increased query load while ensuring minimal impact on CPU and RAM usage.

Testing a Big Attack

In the last test, the focus shifted towards enhancing the potency of the DNS reflection attack. While the previous attacks involved 10,000 packets per second, this more powerful attack reached a traffic 50,000 packets per second, employing the request type "ANY". This intensified attack configuration was aimed to assess the potential for greater disruption and the limits of the network.

During this amplified attack, notable changes in DNS query times were observed. The mean query time reached 174 ms, with occasional spikes of 400 ms. These elevated query times would certainly have a substantial impact on the loading speed of web pages. Interestingly, when considering ping latency, the increase was about the same as when using the lower packet rate. The limited increase in latency can be attributed to the server's inability to keep up with the amplified attack: several packets were ignored and were not sent to the spoofed IP, resulting in partial blocking of the reflection. This server behavior lowered the expected proportional increase in latency, as the server struggled to process the influx of requests.

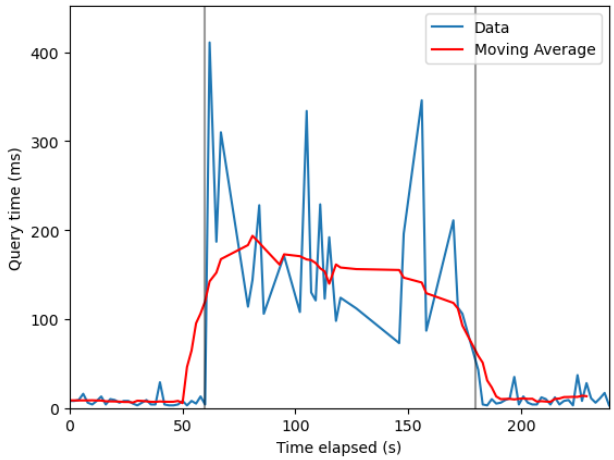

Figure 7. Query time over the course of the test with a packet rate of 50,000 pps with the requests of type ANY.

Analyzing the server diagnostics, it was found that the RAM usage remained unaffected by the amplified attack, while on the other hand the strain on the CPU was evident. Unlike the 10,000 packet per second cases, where the CPU usage only experienced a marginal increase, the amplified attack caused the CPU usage to rise significantly. The CPU utilization surged from around 4% of the baseline to a mean of 24.5%, with occasional peaks exceeding 30%. This substantial increase in CPU load indicates the server's struggle to handle the intensified query traffic,

resulting in a notable impact on its computational resources.

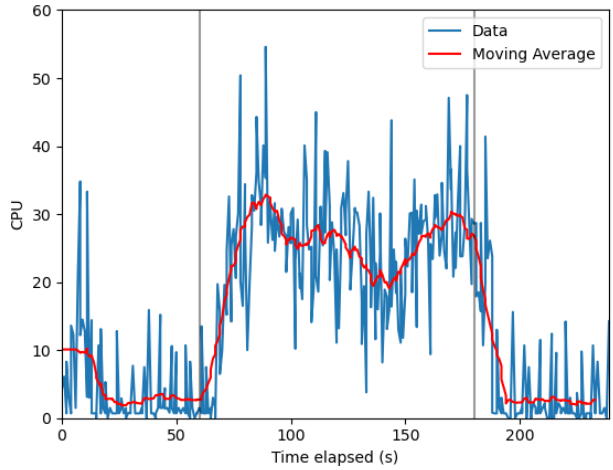

Figure 8. CPU usage over the course of the test with a packet rate of 50,000 pps.

This final test showed how increasing the attack potency does not always yield the desired effectiveness. The network topology, or in this case the behavior of the server, can exhibit unpredictable characteristics. Despite the significant amplification of the attack, the server's partial blocking and compromised response contributed to unexpected outcomes.

8 Mitigation measures

There exist a variety of measures that can be employed to mitigate the effects of DNS amplification attacks. These measures can be broadly categorized into three groups: those that aim to reduce the probability of an attack occurring, those that aim to minimize the impact of an attack by detecting it early, and those oriented to enhancing the resilience of the DNS service.

8.1 Proactive measures

Rate limiting

Rate limiting is a measure that can be used to mitigate the impact of DNS amplification attacks. The idea behind rate limiting is to limit the number of responses that a DNS server can send to a specific IP address within a certain time period. That way the queries sent by the attacker are dropped by the DNS server, therefore reducing the amplification effect.

Trusted sources

When a DNS recursive server is publicly accessible on the Internet, it becomes susceptible to receiving queries from any source. The potential range of IP addresses that can be spoofed is vast, making it impractical to block all of them effectively. However, a possible mitigation strategy is to restrict the number of sources allowed to send queries to the DNS server by implementing a whitelist of trusted sources. By creating such a whitelist, the probability of an attack occurring can be reduced. It is worth nothing to underline that the introduction of a whitelist is a not a foolproof method, since IP in whitelist can still be spoofed and used to perform the attack.

Firewall

A firewall is a network security system that monitors and controls the incoming and outgoing network traffic based on predetermined security rules. Setting up a proper firewall both DNS server side and victim side can block unauthorized traffic and reduce the impact of the attack.

8.2 Detection measures

The DNS amplification attack relies heavily on IP spoofing as a central activity. The key concept behind *detection* techniques is the ability to differentiate between an original source IP address and a spoofed one, therefore immediately identify and mitigate the attack.

Routing hops detection

This mechanism was proposed in the paper by '**hopcount**'. The idea is to exploit the inconsistency between the number of hops of a spoofed IP packet and the spoofed IP address itself. The hops number is inferred by the TTL value in the IP header. This mechanism can detect almost 90% of the spoofed packets.

Machine Learning

In the last decade with the developing of machine learning, some algorithms have been proposed to detect the DNS amplification attack. In the paper by '**machinelearning**', is proposed a machine learning based approach to detect the attacks, using *Random Forest*, *MLP* and *SVM* algorithms. However, a more recent publication by '**createDNS**', shows that using an adversarial neural network approach (EAD) it is possible to easy circumvent the detection. The idea is to train a network to slightly modify the input data (DNS queries) in order to fool the detection algorithm.

8.3 Resilience measures

Resilience measures aim to enhance the DNS's ability to withstand attacks and maintain its service even when under attack.

Anycast scheme

A DDoS attack aims to disrupt a victim server's service by overwhelming it with a high volume of packets. A solution to this issue involves creating multiple replicas of the victim server, all sharing the same logical IP address. The destination of incoming packets is determined by the Anycast protocol employing certain routing criteria. Thereby, the load is distributed among the replicas and the strain on each individual server is reduced. In 2015, a significant DDoS attack targeted the DNS root name servers, resulting in a denial of service for some of them, as documented in the report '**anycast**'. This attack highlighted that although anycast architecture can enhance the resilience of DNS servers, it is not foolproof solution to DDoS attacks. Nevertheless, due to the implementation of anycast technology in 11 out of the 13 root servers, the impact of the attack was limited and partially mitigated.

Caching behavior

The paper authored by '**alleviatingimpact**' explores an approach to mitigate the impact of DDoS attacks on DNS performance. The authors argue that a relatively simple modification to the caching behavior of a DNS server can yield significant improvements in performance during such attacks. Their proposal suggests that a DNS server should refrain from evicting cache entries when it detects unavailability of the relevant DNS servers. Instead, these entries should be retained until the corresponding servers become available again. By implementing this change, even if a relevant server is rendered inaccessible during a DDoS attack, the DNS recursive server can still serve the cached entries, thereby providing responses for a portion of the requested domain names.