Investigating the Amplification Potential of Common UDP-Based Protocols in DDoS Attacks

A measurement study conducted across the networking infrastructure in Belgium and Luxembourg

Vlad-Petru Nitu[1]

Responsible Professor: Georgios Smaragdakis [1]
Supervisor: Harm Griffioen [1]

[1]**EEMCS, Delft University of Technology, The Netherlands**

Name of the student: Vlad-Petru Nitu
Final project course: CSE3000 Research Project
Thesis committee: Georgios Smaragdakis, Harm Griffioen, George Iosifidis

Abstract

Distributed Reflection Denial-of-Service (DRDoS) attacks remain among the most damaging cyber threats, leveraging vulnerable UDP-based protocols to amplify traffic and overwhelm targets. Our measurement study investigates the amplification potential of three commonly exploited protocols: DNS, NTP, and Memcached, within the context of the network infrastructure in Belgium and Luxembourg. By analysing amplification factors through various query strategies, we aim to identify potential vulnerabilities and correlations between factors that influence the weaponisation of these protocols. We also investigated application-layer looping vulnerabilities, also known as "Loopy". Our findings indicate that despite protocol hardening, significant risks remain, particularly with improperly configured DNS servers and not updated NTP and Memcached versions.

I Introduction

Distributed Denial-of-Service (DDoS) attacks [1] are still one of the most common and devastating cyber-attacks in today's age [2], even after 26 years from the first such attack being observed in the wild, the Smurf attack [3], [4]. DNS [1], [5], NTP [6], and Memcached [7], [8], [9] were three of the most abused protocols in amplification attacks, reaching traffic volumes at the Tbps range [10].

Massive DRDoS attacks have been recorded in the past. In March 2013, attackers launched a DRDoS attack against Spamhaus in the range of 300 Gbs, which was successfully defended by Cloudflare [11]. More recently, in 2020, AWS mitigated one of the largest DDoS attacks seen so far in history, peaking at 2.3 TBps [12]. The 2024 Q1 Cloudflare's DDoS threat report points out that they mitigated 50% more DDoS attacks in the first quarter of 2024 compared to 2023 [13].

DNS. Cloudflare also indicates that DNS-based DDoS attacks have increased by 80% in 2024, remaining the most relevant attack vector [13]. Regarding this kind of amplification attack, the most attractive query type for attackers is "ANY", a pseudo-query that asks the DNS resolvers to aggregate the available records in response, thus achieving large amplification factors [5]. Moreover, if the server provides DNSSEC responses, this further increases the amplification factor, as it adds cryptographic signatures to the response and thus significantly increases its size [14]. Although many strategies exist to craft different requests (i.e. query root domains, country-code top-level domains, etc.), the literature lacks a proper framework for crafting such DNS queries to maximise the amplification factor.

NTP. Empirically, a smaller amount of public NTP servers can be found in the wild compared to DNS. However, they yield significantly larger amplification factors if vulnerable, peaking at 5,500$\times$, as observed in previous studies [2].

Memcached. Even though it has recently received less attention in the literature, compared to DNS and NTP-based DDoS attacks, it has hit traffic figures in the range of Tbps, such as the renowned Github attack that peaked at 1.35 Tbps [15], mitigated by Akamai and documented in their Internet security report [7]. In the past, researchers found vulnerable Memcached servers reaching amplification factors of 50,000 [16], [17].

In our work, we analyse whether such amplifiers can still be found in the wild, estimate the amplification factors of such servers, and determine which parameters affect the success of cyberattacks. We will also compare our observations with those of other networking infrastructures analysed by our peers.

Note that our study focused solely on Belgium and Luxembourg, as these were the countries assigned for our thesis. Our contributions are as follows:

- We provide a framework on how attackers can find amplifiers in the wild for servers running DNS, NTP and Memcached.
- We audit a sample of the Belgium and Luxembourg network landscape, estimating the amplification factors of vulnerable systems and whether these are susceptible to application-layer level loops.
- We reflect on the results and define success factors for amplification attacks. We relate these factors to how attackers behave by following Griffioen et al.'s framework [18] and propose countermeasures to mitigate such vulnerabilities.
- We analyse the correlation between factors that influence amplification attacks.

The remainder of this paper is structured as follows. Section II describes the prerequisite knowledge a reader needs to understand the paper. Section III analyses related work done in the field. Section IV elaborates on how we gathered the data. Section V presents our methodology for finding amplifiers and estimating the amplification factors, as well as how we check for theoretical application-layer loops. In Section VI, we present why our research is responsible, whereas in Section VII, we showcase our results. Section VIII brings up a discussion, details the limitations related to our research, and presents recommendations on how to protect servers from becoming amplifiers, while Section IX concludes the paper and reveals future work ideas.

II Background

In this section, we briefly introduce the terminology used throughout the paper, both for UDP-based protocols that we have analysed in our work and for cyberattack terms.

Reflectors and Amplifiers. *IP spoofing* is the process of modifying the source address in an IP packet to either anonymize or hide the sender's identity [19]. A *reflector* is an innocent, vulnerable third-party server [5]. On the other hand, an *amplifier* is a reflector that receives a request (usually with a spoofed sender's IP) and answers with a larger response. The only difference between these two is that reflectors answer with a response of a size that is equal to the request's size, whereas amplifiers are *asymmetrical*, by providing responses that are larger than the requests. Thus, an attacker aims to maximise the amplification factor by prioritising amplifiers.

DNS (Domain Name System) protocol is also known as the "phonebook" of the Internet, and its primary purpose is to provide a mapping from a human-readable *domain name* (such as "google.com") to a machine-readable *IP address* (such as "145.94.219.173"), that is used by other machines to find the device on the web. The process of mapping domain names to IP addresses is known as *domain resolution* [20].

A *DNS resolver* is called *open* when it responds to requests from all over the Internet. Many open DNS resolvers exist in the wild, which indicates a vast misconfiguration trend [21]. Even though the initial specification document for DNS advertises a maximum response size of 512 bytes [22], the DNS Security Extensions (DNSSEC) was later introduced to ensure the sender's origin authentication and data integrity by adding *RRSIG* RRs in DNS responses [23]. This required DNS to support larger answers, leading to the development of EDNS, which provides the framework for DNSSEC to function effectively by transporting DNSSEC-related records, such as RRSIG [24]. *DNS Flag Day 2020* [25] was an initiative that proposed several recommendations for improving the security in DNS, such as setting the EDNS Buffer Size to 1,232 bytes.

EDNS enables UDP payloads of up to 4KB (4,096 bytes) [24]. This makes attackers more attracted to DNS resolvers that support EDNS due to the potentially large amplification factor they can yield. An attacker must include an OPT pseudo-RR in the query to request EDNS support. Moreover, when the DO (DNSSEC OK) bit is set in the request, it indicates that "the resolver is able to accept DNSSEC security RRs" [26].

NTP (Network Time Protocol) was released in 1988 [27] and is still the most common protocol used to synchronise the clocks of computer systems, even if it has suffered security vulnerabilities in the past, such as being a DDoS attack vector or experiencing data breaches [28]. Synchronisation is crucial for ensuring precise timing in tasks such as logging events.

Mode 7 (Private) queries were meant for debugging and should not be open to the public, presenting the risk of creating DDoS attacks [29]. The Mode 7 queries we analyse are: "monlist", "get_restrict", "peer_list" and "peer_list_sum".

Rossow found average amplification factors of up to 4,670 by issuing "monlist" queries [2], which retrieved the last 600 IP addresses that used an NTP server [6]. However, since "ntpd 4.2.7p26", Mode 7 commands are disabled by default, a security measure that was taken to stop NTP DDoS attacks [30]. Moreover, NTP "version" queries, in Mode 6 (Control) [31], are also used to weaponise vulnerable NTP servers, which tend to be more accessible to the public than the servers that answer to "monlist", but instead provide smaller amplification [2].

Memcached is a database caching system that speeds up websites and networks [9]. Memcached used to be one of the most attractive protocols an attacker was searching for an amplifier to operate because it works over UDP [32]. Thus, it does not require a handshake step, which makes IP spoofing possible. Because it is a caching mechanism, an attack of this type usually requires the attacker to implant a large amount of data on the exposed Memcached server and then retrieve it through a query that spits out the content from the cache.

DDoS and DRDoS Attacks. In *Denial-of-Service (DoS) attacks*, attackers aim to disrupt the services of the victim, overwhelming it with lots of requests, by consuming the victim's resources (inducing large CPU usage) or network bandwidth (by flooding a link with lots of packets). Such an attack compromises the system's availability, a crucial property of the CIA (Confidentiality, Integrity and Availability) triad [33]. The client's bandwidth may become a bottleneck when taking down large systems. Thus, in practice, in order for the cyberattack to scale, most of these attacks are distributed, generated through a botnet [34] or by abusing reflectors [5]. Moreover, when the attacker distributes its attack over multiple reflectors, the attack is known as *Distributed Reflection Denial-of-Service (DRDoS)*.

Amplification metric. We used the *BAF (bandwidth amplification factor)* defined by Rossow [2], depicted in Equation (1).

$$BAF = \frac{\text{len(UDP payload) amplifier to victim}}{\text{len(UDP payload) attacker to amplifier}} \tag{1}$$

We considered this metric suitable, as it:

- Keeps the experiment reproducible and portable to future packet structures (as the BAF considers only the UDP payload size, remaining a valid metric even if the upper protocol layers, such as the IP header, change in the future).
- It allows comparing our results to other measurement studies that use this metric [5], [18], [35].

III Related Work

This section describes the related work in the area of UDP-based amplification attacks. Many measurement studies have been conducted in the cyberspace, especially during 2013-2014, when multiple large-scale attacks occurred in the wild [7], [11], [15].

In 2014, Rossow studied 14 UDP-based protocols to understand how vulnerable these are to amplification [2]. He also introduced the BAF metric and analysed attacks in the wild using darknets. An important conclusion that shapes our research methodology is that obtaining authoritative DNS servers takes significantly more time (i.e., hours) than active scanning for open DNS resolvers (i.e., minutes). However, an improperly configured authoritative DNS server (i.e., one that allows recursion) may yield higher amplification factors, so the additional overhead the attacker must endure to obtain such amplifiers may still be worth it.

Van der Toorn et al. investigated 65% of the DNS namespace by performing active measurements and discussed how the resolved domains affect the amplification factor [5]. Moreover, they analyse the impact of a DNS implementation refusing "ANY" queries on the amplification ratio by comparing the response size of such queries to "DNSKEY" or "TXT" [36]. They conclude that dropping such requests does not completely solve the problem, as it reduces the amplification factor by only 70%. This guided us towards analysing other query strategies besides "ANY", such as "DNSKEY".

Griffioen et al. analysed how adversaries plan, prepare and execute amplification attacks by providing a six-step framework to assist in developing better defences [18]. In our work, we use this framework to simulate attackers' behaviour by guiding our amplifier discovery process.

Kührer et al. managed to collaborate with the security community to run a large-scale campaign to reduce the number of vulnerable NTP servers by 92% [37], which confirms why the vast majority of NTP servers we discovered does not respond to Mode 7 queries. Moreover, they present a methodology for fingerprinting NTP servers to reveal the OS (operating system) and NTP daemon version they run. Lastly, they concluded that the most vulnerable hosts run Cisco IOS, guiding us towards analysing such devices.

Recent work on application-level layer loop-based attacks was conducted by Pan et al. [38]. If the attacker is able to find a pair of hosts that loops, it can craft a packet with the spoofed IP address of server B as the source, put an error message that triggers the loop in the body, and then send it to server A. This way, an infinite loop at the application-layer level is triggered between servers A and B, bringing both systems down. This is considered a severe vulnerability, as it cannot be filtered out by the TTL (Time-to-Live) field in the IP header.

IV Datasets

Our methodology for acquiring the data uses only passive scanning, and the ethical reason for this decision is detailed in Section VI. We used Censys [39] and Shodan [40], which

Table 1: Statistics about acquiring the data. "Final" depicts how many IPs, per protocol, were used from Censys and Shodan.

Dataset	Strategy	DNS	NTP	Memcached
Censys	Before Research Plan	2K	2K	116
	After Research Plan	8.1K	15.5K	116
Shodan	Before Membership	100	100	100
	After Membership	18K	17.1K	419
Final	Censys	8.1K	15.5K	0
	Shodan	0	0	419

are search engines for Internet-connected devices, focusing on comprehensive network-wide scans that identify and catalogue IoT devices (i.e., webcams, smart fridges, etc.). The queries used to retrieve the data can be found in the Appendix A.1.

We collaborated with **Censys** to obtain a Research Plan for our work, increasing our API Rate Limit from 2,000 to 30,000 results per query. This significantly improved our analytics' comprehensiveness, allowing us to identify patterns and correlations that may have been missed otherwise. Thus, we gained greater flexibility in correlating different success factors (such as EDNS0 Buffer Size and DNS version) that enhance amplification attacks. The data was acquired on May 9, 2024, and the measurements were conducted from May 9 to May 18, 2024.

Shodan was mainly used to extract the Memcached servers (as it reported more such servers than Censys did) on May 10, 2024, as well as to manually augment the fingerprinting of vulnerable servers (i.e., check if they give signs of being honeypots, learn the OS of the server, etc.). Note that the additional DNS and NTP servers from Shodan were not included in our final results, as we considered the API Rate Limit (10,000 IPs per month after being rewarded with their Membership plan) too restrictive to provide additional valuable information. Moreover, we considered that aggregating the two datasets from Censys and Shodan would have resulted in redundant information (lots of overlapping IPs), which would have complicated the analysis and skewed the results. Thus, for the rest of the paper, we only focus on the "Final" dataset (Table 1), which is comprised of all servers located in Belgium and Luxembourg displayed on Censys (DNS and NTP) and Shodan (Memcached). Note that even though the experiments were run in May 2024, we are unaware of how recently the Censys and Shodan scans were conducted.

Moreover, to find all TLDs (top-level domains) for our DNS query strategies, we used the Root Zone Database provided by IANA to rank the most vulnerable ccTLDs (country-code), when coupled with "ANY" queries and resolved by Google's Public DNS (8.8.8.8) [41]. This data was collected on May 3, 2024. We obtained the most popular 10 million domains reported by TUM [42], [43], ranked according to the PageRank algorithm [44], on April 16, 2023.

We have also used the NTP Pool project to gather open, voluntarily hosted NTP servers and check whether they are vulnerable to such attacks [45]. The pool states that there are 23 and 11 active NTP servers in Belgium and Luxembourg, respectively, on May 24, 2024. Of these, only 16 IPs were publicly available, of which none proved vulnerable.

V Methodology

In this section, we present three methodologies we used in our research. In V.1, we discuss the main methodology used to acquire servers, check if they are open and then compute the BAF scores these can achieve. In V.2, we elaborate on how we obtained authoritative DNS servers. Lastly, in V.3, we explain how we use Pan et al.'s source code to find loops [46].

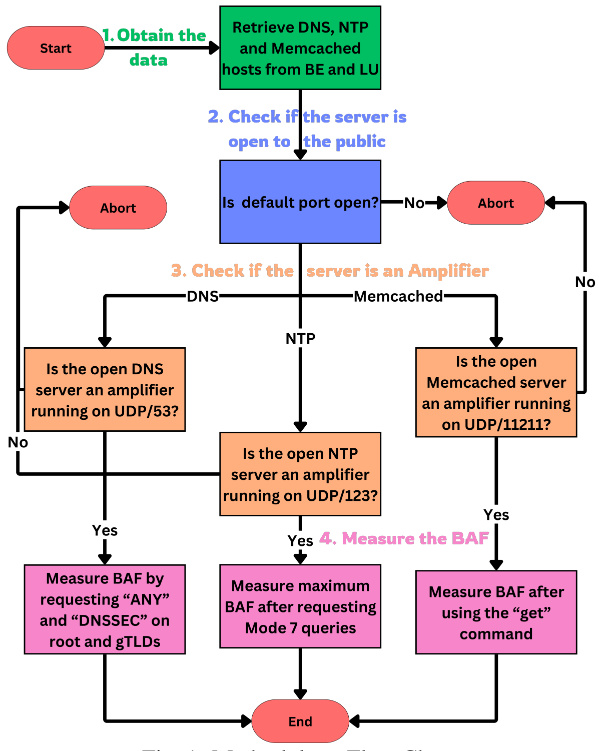

Start
1. Obtain the data
Retrieve DNS, NTP and Memcached hosts from BE and LU
2. Check if the server is open to the public
Is default port open?
No
Abort
Abort
3. Check if the server is an Amplifier
DNS
NTP
Memcached
No
Is the open DNS server an amplifier running on UDP/53?
Is the open NTP server an amplifier running on UDP/123?
Is the open Memcached server an amplifier running on UDP/11211?
No
Yes
Yes
Yes
4. Measure the BAF
Measure BAF by requesting "ANY" and "DNSSEC" on root and gTLDs
Measure maximum BAF after requesting Mode 7 queries
Measure BAF after using the "get" command
End

Fig. 1: Methodology Flow Chart.

V.1 Main steps

The methodology (Fig. 1) consists of four primary steps:

Step 1. First, we query the search engine databases to obtain the data. Next, we process the data to only store relevant information for our future work per each host. This includes the OS that it is running, the Autonomous System (AS) that this host is part of, the DNS version (in case it is a DNS server, and *fpdns*, a fingerprinting tool [47], can provide such information), etc.

Step 2. We check if a server is open to the public by sending a basic query that increases the conditional probability that if the server is open, it will respond (with a valid packet) to our request (i.e., there are greater chances a server will answer to a DNS query that asks for "A" records, compared to one that asks for "ANY"). We wait for a response either until we receive an ICMP error packet ("Destination Unreachable") or after the query times out (3 seconds).

DNS. We send an "A" query type to a well-known domain, such as "google.com". We also set the recursion desired (rd) bit to 1. It is crucial to note that secure DNS servers are expected not to be open to the public, whereas the NTP servers, by design, should be open [45].

NTP. We send a Mode 3 (Client) request to check whether the server responds and the clock time it provides is synchronised. In this query type, the client sends a synchronisation message to the server that operates in Mode 4 (Server) and answers accordingly. This mode of operation was chosen since such a request-response pair is the most common interaction with an NTP server, and it is expected that if an NTP server is open, it will answer such a query. Furthermore, we used "ntpd" to interact with the server and capture the packets sent in Wireshark [48]. Indeed, we observe that "ntpd" crafts an identical packet to the one we crafted using Scapy, a tool for manipulating networking packets in Python [49].

Memcached. We send a *"stats slabs"* request on UDP. This request will return all the slabs available in the Memcached process [50]. If the server returns at least one slab, we consider it to be open to the public.

The packets we sent were crafted using Scapy (Appendix A.2). Moreover, this filtering step is crucial in reducing the workload on the attacker's (or cybersecurity researcher's) side, as well as eliminating sending unnecessary queries to offline hosts (i.e., flooding the links with useless packets). Lastly, this approach increases the scalability of our methodology, making it suitable for worldwide applications in future studies.

Step 3. We check which of the responsive servers we have just gathered are amplifiers (have $BAF \geq 1$ for a specific query strategy). To observe such amplifiers, we adapt Griffioen et al.'s framework [18] as follows:

Substep 1. *Capability Development*: we exclusively investigate UDP, NTP and Memcached amplification attacks; **Substep 2.** *Infrastructure Reconnaissance*: completely skipped, as our methodology uses only passive scanning; **Substep 3.** *Target Reconnaissance*: completely skipped, as our threat model assumes that the attacker focuses exclusively on finding open amplifiers that maximise the BAF; **Substep 4.** *Weaponisation*: followed partially, as our threat model does not assume the attacker wants to evade honeypots, to keep our work simple and reproducible (i.e., identifying a set of honeypots today may be completely different to the set of active honeypots one month later). However, we extensively study strategies that an attacker may follow; **Substep 5.** *Testing*: our amplifiers are chosen exclusively based on the amplification factor they can achieve, without considering packet loss, delay, and other factors; **Substep 6.** *Execution*: completely skipped, as this assumes attacking hosts, similar to how attackers do in the wild.

Step 4. Depending on the protocol, we measure the amplification factor based on different strategies.

DNS. Our attacker tries to remain untraceable, so it cannot implant records into the DNS servers or register domains. We focus on queries that have the highest potential to obtain huge BAFs: "ANY" [5], "DNSSEC" [51], and "TXT". We further ask the DNS resolvers to operate over EDNS (by setting the DO bit to 1 [52]) and set the buffer size to 4,096 so that responses can exceed 512 bytes [24]. The domains that we try to resolve vary from SLDs (second-level domains), such as "google.com", to gTLDs (generic), such as "bg.", and lastly, the root TLD (".").

Similar to how an attacker would act, we aim to craft small queries that elicit large responses. Thus, we query short domains for potent query types. Note that we have also compiled a framework for systematically finding authoritative DNS servers, further explained in subsection V.2.

NTP. For each server that was considered open, we sent four Mode 7 query types: "monlist", "get_restrict", "peer_list" and "peer_list_sum". We only store the one that maximises the amplification factor for a server and the specific query type to analyse whether a correlation exists between the OS and the Mode 7 queries. Note that if the server answers to "get_restrict", it is also vulnerable to information disclosure attacks, as it leaks sensitive data about the server's configuration.

Memcached. We run the "get" command on the key-value pairs already residing on the system. If there were multiple pairs, we prioritised using the pairs with the largest values and aimed to find keys that were short. Generally, adding more short keys with large associated values in the same query can significantly increase the BAF (Appendix A.3). Note that Memcached also allows one to ask for the same key multiple times in a single "get" command (i.e., we can retrieve the value associated with the key "a" thrice in the same query: "get a a a" [7]). Appendix A.3 (Fig. 7) depicts the theoretical maximum BAF one can achieve.

Note that we observed a single vulnerable Memcached server in Luxembourg, but we further validated the correctness of our methodology on vulnerable servers located in France and the Netherlands (found by our research colleagues).

V.2 Finding authoritative DNS servers

We gather authoritative DNS servers and compare their amplification factors' performances to the non-authoritative ones. We preprocess the 4 million most popular domains (from the dataset provided by TUM, mentioned in section IV) and only store the 2,000 of them that terminate in "be" and "lu" (ccTLD of Belgium or Luxembourg). Next, we run "dig NS" [53] on these domains by using Google's Public DNS (8.8.8.8), to retrieve the authoritative servers of each domain, and we also de-duplicate the dataset. We run the "dig" command on each domain by requesting "A" records to resolve the domain's IP. To double-check that these resolved IPs were based in Belgium and Luxembourg, we made a request to "ipinfo.io," which can geolocate the host. To gather the statistics, we query all the domains a server is authoritative for but only store the maximum BAF it obtained.

V.3 Loopy attacks

We found theoretical infinite loops at the application-layer level by following the methodology of Pan et al. [38]. We were only interested in loop pairs between servers that run the same protocol (DNS to DNS and NTP to NTP), so no cross-protocol looping checks were performed. We used the open-source code referenced in the paper to find these looping pairs. [46]

Our dataset was notably smaller than the dataset analysed in their paper (i.e., we had 8.1k DNS servers retrieved from Censys, whereas they used 2 million DNS servers [38]). They also drop edges with less than 100 IPs from the loop graph, which makes sense for such a large dataset they analyse not to form small, less significant loops. However, due to the scale of our dataset, we are interested in such small loops, so we adapted their code to consider all edges that have at least 2 IPs, to ensure that a server does not self-loop.

Moreover, their methodology's loop probing (third) step ignored clusters with less than 10,000 different responders. However, based on the scale of our dataset, we considered all clusters. Finally, we skipped the last step, which proposed using a proxy to verify the potentially looping pairs found.

VI Responsible Research

Ethics. We adhere to *passive scanning* of the hosts that run in Belgium and Luxembourg by using Censys and Shodan instead of actively scanning the Internet using tools such as ZMap [54]. The process is completely "passive" because we retrieved the hosts' information from databases, which were created sometime in the past when the search engines actively scanned the web. However, even though their scan was active, our process of retrieving the results was passive. We made this decision not to provoke malicious activities on the Internet so as to keep our research non-intrusive. Moreover, we did not publish the list of vulnerable hosts to mitigate the risk of an attacker reading our paper and launching a real DDoS attack.

However, after acquiring the dataset, we had to validate servers to check whether they were amplifiers. Both the "dig" [53] queries and the "HTTP GET" requests were con-

Table 2: Mean BAFs for top 10%, 50% and all DNS open amplifiers for different query strategies.

Domain	Query Type (Strategy)	BAF for % of Amplifiers		
		10%	50%	ALL
Root (.)	ANY	58.34	25.15	13.99
	DNSKEY	31.62	22.86	15.51
bg.	ANY	67.72	26.96	14.69
	DNSKEY	39.45	36.73	28.44
sl.	ANY	127.58	41.65	21.42
ve.	ANY	57.74	21.31	11.49
Auth.	ANY	22.89	9.93	5.32

ducted ethically by timing out between different requests to ensure we do not stress the DNS resolver and the server that hosts "ipinfo.io", respectively, as well as preventing rendering any network links unavailable. We used a timeout period of 3 seconds. The four variants of Mode 7 NTP queries we sent to check whether an NTP server was vulnerable were sent at least 45 minutes apart.

Furthermore, we skipped the last step for loopy attacks: "Loop Verify", as it expected us to implement a proxy. If we were to introduce bugs in the proxy implementation we would have risked breaking the two looping systems. Thus, for the scope of our work, we take this defensive action while accepting the trade-off that we may introduce some false positive results (i.e., potential looping pairs that do not get triggered in practice, as one of the servers limits the number of requests per IP address). Thus, the loop pairs we found are theoretical, so one should validate them with a proxy for completeness.

Reproducibility. We cannot guarantee full replicability due to how dynamic the cyberattack surface is. A given server we analyse today may have a different IP address tomorrow. Moreover, a researcher's vantage point may produce different results due to different perspectives IXPs (Internet Exchange Points) have on amplification attacks [8]. We ran the experiments locally in Delft, The Netherlands, using a single vantage point. However, we created a *Dockerfile* containing all our Python dependencies, and our code was open-sourced [55].

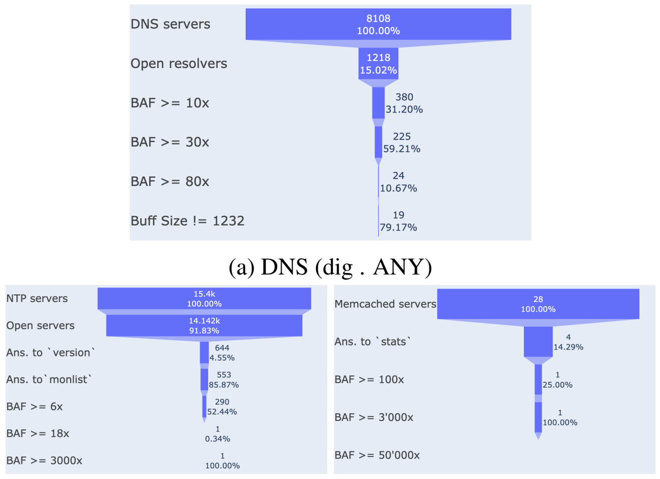
Our dataset is FAIR (Findable, Accessible, Interoperable, and Reusable), and the entire work is compliant with the Netherlands Code of Conduct for Research Integrity [56]. Our paper is a scholarly work conducted at TU Delft, and the reported results and methodologies used are transparent.

VII Results

In this section, we analyse the results of our measurement study. We investigated DNS, for which we observed many IPs providing huge BAFs: 24 out of 8,108 servers provided $BAF \geq 80$ (Fig. 2a). Next, we discuss NTP results, that has 91.83% of the servers responsive (Fig. 2b). Lastly, we check Memcached, for which we find a severely vulnerable server that provides a BAF of around 9,500 (Fig. 2c). We also explore the correlation between factors that influence such attacks.

VII.1 DNS

We observe lots of DNS servers that provide a medium-sized BAF (i.e. 50×-80×), compared to the trend in NTP and Memcached, which is that most of the servers were secure by default (after large DDoS attacks were registered [7], [57]) but still those that are not secure provide massive BAFs (i.e., peaking at 5,500 for NTP). The 10% most vulnerable amplifiers obtain a cumulative BAF of 7,099. Assuming an attacker has an uplink that enables him to generate 100 MBps in UDP payload



(a) DNS (dig . ANY)

(b) NTP (c) Memcached

Fig. 2: Methodology pipeline for each UDP-based protocol analysed.

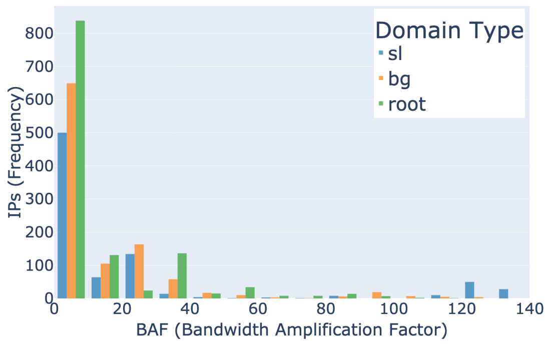


Fig. 3: BAF distribution for "ANY" queries, depending on the domain used: "sl.", "bg." or root (".").

requests, traffic as high as 7.1 TBps can hit the victim.

On average, the largest BAFs we were able to obtain were through "ANY" queries on ccTLDs (country-code) such as "ve.", peaking at 131.96 BAF and "sl." at 132.09 BAF (Fig. 3). Our different strategies show that querying "sl." outperforms the others (Table 2). Unexpectedly, if an attacker were to use the 10% most vulnerable open resolvers we found, it would have achieved a mean BAF of 127.58, which is larger than some results obtained in practice [2] (presumably that the small number of servers we query may also skew the statistics). Moreover, we note that authoritative servers perform poorly when queried with "ANY" (22.89× when using the 10% top-most amplifiers), a result that opposes Rossow's findings [2]. This may indicate that in the last 10 years, authoritative servers have become more secure, experiencing a trend of getting properly configured by system administrators and hardened through several patches. It may also be that in our small dataset, there are no authoritative DNS servers that yield large BAFs for these specific domains (Fig. 8 from Appendix A.4).

We also note that "DNSKEY" queries are relevant in providing harm on the Internet, yielding 31.62 and 39.45 mean BAF when using the 10% most vulnerable servers, when resolving the root domain and "bg.", respectively (Table 2). If "ANY" queries were to be refused by the servers, as in the analysis conducted by Van der Toorn et al. [5], the reduction in the mean amplification factor would have been close to 45% (from 58.34 to 31.62 for ".", and from 67.72 to 39.45 for "bg.").

The effectiveness of an attack strongly depends on which domain the attacker uses (Fig. 3). It is clear that huge BAFs (of at

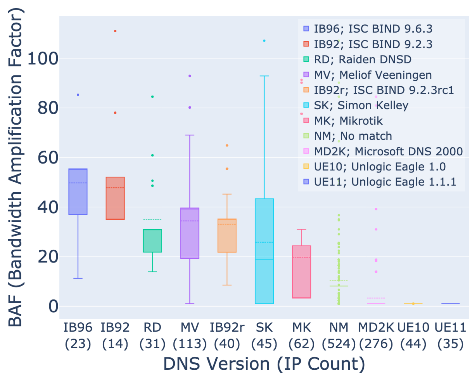


Fig. 4: BAF distribution for "ANY" queries, on the root (".") domain, grouped by DNS Version. The number of IPs per DNS Version category is depicted in parentheses.

least 120) are only recorded when querying "sl." and "bg.", and that the root domain is still potent to cause an attack.

In Fig. 2a, we observe that only 15% DNS servers advertised by Censys are open to resolving our queries. There may be multiple factors that influence the probability of receiving a response to such a request, such as the DNS resolver being configured not to answer "ANY" queries or because the IP churn rate [58] is large for DNS servers [2] (so, depending on the exact date Censys conducted the wide-scale scan, multiple DNS servers may have changed their IPs). Another relevant point is that, from the 24 servers that yield large BAFs (of more than 80×), 79% of them do not follow the DNS Flag Day recommendation by not configuring their buffer size to 1,232 [25].

We dive deeper and analyse correlations between variables that influence the BAF in DNS DDoS attacks (Appendix A.5), such as AS (Orange Belgium), EDNS0 Buffer Size (4,096, as most of the old DNS versions use by default, such as "Unbound 1.4.21" [59]), DNS version (BIND 9 [60]), vendor (Zyxel), and product (RouterOS that runs on MicroTik routers [61]). Figs. 10 - 15 are located in Appendix A.4. We remark strong correlation between products and vendors (Fig. 12). We observe that most cells have a 100% IP rate, indicating that one vendor produces one product most of the time and only that vendor uses its product. The product represents an OS produced by the vendor (i.e., Synology produces DSM).

When it comes to the relationship between vendors and how they configure their EDNS0 Buffer Sizes (Fig. 10 and Fig. 13), we observe that Zyxel has all its servers (that have their buffer sizes known) configured to use 4,096, bringing a median BAF of 39.18 (for the 23 servers we identified). WatchGuard has 88% of their servers configured to 512. This large ratio indicates that this may be the default buffer size for the shipped equipment and may show the absence of proper manual configuration by network administrators.

A different pattern can be observed at CentOS, which has 95% of its DNS equipment configured to advertise a buffer size of 4,096 while being fully secure against amplification attacks (none of their servers answered our DNS "ANY" queries on "."). FreeBSD has two-thirds of its DNS configurations set up to advertise the buffer size to 1,232 and obtain a smaller-than-average BAF (18.75×), thus seeming to adhere to Flag Day.

We conclude that vendors have different ways of configuring DNS servers by default and show pretty different behaviours under analysis (i.e. some set the default EDNS0 Buffer Size to 4,096, others to 512; some implement the Flag Day recommendations poorly, others do not implement them at all).

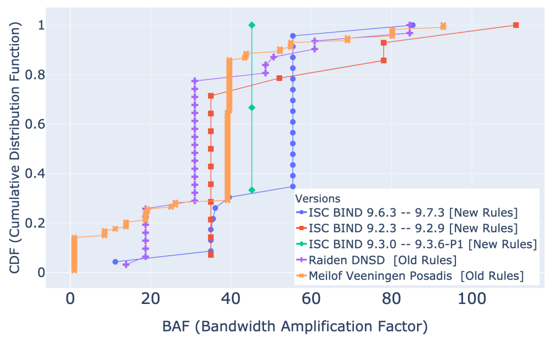


Fig. 5: CDF (Cumulative Distribution Function) of the five-most vulnerable (producing largest mean BAFs) DNS Versions.

We will not elaborate on the relationship between products and EDNS0 Buffer Size configuration because the large correlation between products and vendors explained previously points to a similar trend as the Vendors vs. EDNS0 Buffer Size (Fig. 14). In short, we observe the same trends for the following product-vendor pairs: Zyxel-Zyxel, DSM-Synology and Enterprise Linux-Red Hat.

We also examine the EDNS0 Buffer Size distribution per version of different DNS implementations (Fig. 15). We observe unexpected patterns, such as "lying" about adhering to the Flag Day recommendations by setting up the correct buffer size value, but incorrectly handling the queries (by providing a UDP response larger than the buffer size), or observe DNS versions that are fully secure under DDoS amplification attacks, even when using improper buffer sizes.

"Symon Kelley dnsmasq" DNS implementation (light blue boxplot in Fig. 4) achieves large BAFs from time to time, even though the mass is scattered from 1 to 41 BAF, showing large variance. This variance is also explained by the outliers from the 45 servers of this type, one of them achieving more than 90×, while another peaks at 110×. However, two-thirds of these servers seem to be correctly configured, having EDNS0 Buffer Size of 1,232 and achieving a BAF of 1 (Fig. 15).

"Meilof Veningen Posaddis" has 60% of its servers' buffer sizes set to 1,232 and a median BAF of 39, again showing that not all its servers are correctly adhering to the recommendations. What seems more unexpected is that 40% of its servers still use a buffer size of 4,096, achieving a similar median BAF.

All three "ISC BIND" DNS versions we observed are poorly configured, having all servers advertise a default buffer size of 4,096. They achieve large median BAFs: 35, 35 and 55, respectively. Empirically, improperly setting the EDNS0 Buffer Size while not securing the DNS server to not answer public queries seems relevant in obtaining large amplification factors.

We also conclude that Unlogic Eagle hosts are secure, as no service of this type responds to our queries (Fig. 4). We also note that most of these servers (54% and 89%, respectively) have large EDNS0 Buffer Sizes of 4,096 (Fig. 15), but most probably, they are configured not to answer "ANY" queries to the public. This shows that even if the Flag Day recommendation is not followed, a DNS server may still be secure if manually hardened (by responding with an "HINFO" RR [22] to an "ANY" request, by not being open to the public, etc.).

The correlation analysis between DNS factors presented above uses the median BAF obtained per category. At the end of Appendix A.5, a similar study was conducted on the mean obtained by the 10% most vulnerable servers per category.

For the five most vulnerable DNS versions we have identi-

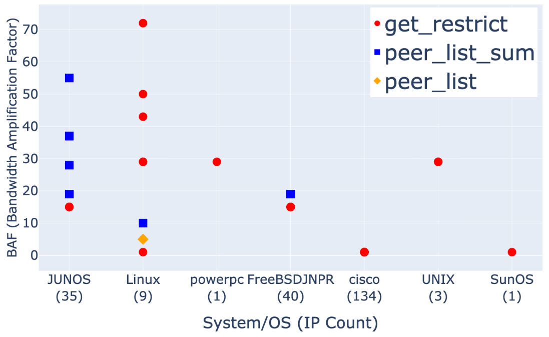


Fig. 6: BAF distribution per System/OS. Different Mode 7 NTP queries are depicted with different colours and symbols. The number of IPs per System/OS is depicted in parentheses.

fied, we spot that a large number of hosts that run the same DNS version produce the same BAF, which is depicted as a vertical line of points (Fig. 5). We also observe that the oldest "ISC BIND" implementation ("9.2.3–9.2.9") peaks at a BAF of more than 100×, whereas the newer versions do not achieve such large amplification factors ("9.3.0–9.3.6" achieves at most 50× and "9.6.3 – 9.7.3" achieves at most 80×). This indicates that the older the version, the more vulnerable it is due to the fact that newer BIND 9 versions support minimal ANY requests [62], adhering to RFC 8482 [36].

For the loopy attack, we find only 1 cluster (ID 24) of responses that lead to self-loops between 6 IPs involved (Fig. 16a from Appendix A.6). We conduct further analysis to check the vendor and the OS these hosts run. Out of these 6 IPs, Censys provides extra information only for three of them. 2 out of 3 servers run the Fireware software provided by WatchGuard, while 1 out of 3 runs Windows (from Microsoft).

VII.2 NTP

By design, NTP is meant to be open to the public so that users can synchronise their systems. Moreover, previous researchers reported that it has minimal IP churn [37]. In Fig. 2b, we observe that from 15.4k NTP servers offered by Censys, 91.8% are open to Mode 3 requests, a benign type of query that does not provide amplification.

However, even when using less dangerous queries, such as "version", the rate of servers that respond drops drastically (to 4.55% of the NTP servers being open to Mode 3 queries). This is a sign of security enhancement in NTP, as public servers should not answer Mode 6 and Mode 7 queries. However, there are still 290 servers that provide a BAF score of at least 6×, of which only one is susceptible to huge amplifications (3,800×).

We analyse the correlation between the System/OS and the type of Mode 7 query (Fig. 6). We observe that the 134 servers we found to run Cisco IOS are the most secure, as none of these hosts are vulnerable to amplification attacks (they did not answer any of our four NTP Mode 7 queries), a finding that opposes Kührer et al., most probably due to the effective campaign the researchers held to mitigate this vulnerability [37]. We also remark that JUNOS servers are mostly vulnerable to "peer_list_sum" queries, peaking at 55×. Moreover, Linux hosts are mainly vulnerable to "get_restrict", achieving BAFs of more than 70. The 3,800× severely vulnerable NTP server we found also hosts Linux but was not included in the plot for readability. We fingerprinted 40 servers running FreeBSD, achieving a maximum BAF of 20× with the "peer_list_sum" command, proving to be more secure than JUNOS and Linux.

For the loopy attack, we find 4 clusters (IDs: 1, 2, 4 and 7) of responses that lead to 3 self-loops (for clusters 2, 4 and 7), while we also find a loop between 2 different response clusters (1 - 4 - 1). There are 20 IPs involved in these application-layer loops (Fig. 16b from Appendix A.6). We also tried to conduct a deeper analysis into the OS that these servers run on, based on the "ntpd" outputs [48]. However, none of these servers had their OS resolved by "ntpd", so we inspected the metadata provided by Censys, which did not store this information either.

VII.3 Memcached

We observe 28 Memcached servers retrieved from Shodan, of which only 14% (4 of them) respond to the "stats slabs" query (Fig. 2c). Of these servers, only one is susceptible to providing a large BAF to an attacker, leading to more than 9,000×, based on the keys it has already stored on the server. Moreover, we tried to increase the BAF by crafting a query that asks for the same key, which has a large value associated with it, multiple times. However, as the keys already stored on the server were long (i.e. "kaishao" of length 7) and did not hold values close to 1MB, this process actually decreased the BAF. Still, on a server that stores shorter keys (i.e. "a" of length 1) that have values of 1MB associated with them, one could run a "get a .. a" command to create a DDoS attack (Fig. 7 from Appendix A.3). We conclude that the most relevant factors in Memcached amplification attacks are the length of the keys stored on the server and the sizes of their associated values.

VIII Discussion

By design, the DNS protocol is an amplifier that must be hardened manually. One single patch or upgrade to a newer DNS implementation will not solve the problem, as it did for the up-to-date NTP and Memcached servers (i.e., those that do not run vulnerable, old versions).

The obtained BAFs are *lower bounds* of the possible amplification factors an attacker may achieve, as there may exist other query strategies that maximise the BAF. For instance, in **DNS**, one may craft a query that achieves a larger BAF on an obscure ccTLD; in **NTP**, other queries may be used to weaponise attacks, both in different modes of operation, as well as other Mode 7 query types; in **Memcached** there may exist different key-value pairs to query in the same "get" request. Additionally, many other protocols (i.e. SNMP) may be used in real amplification attacks, which are outside the scope of our research.

An attacker may use many strategies to weaponise vulnerable DNS servers, such as crafting packets using different DNS flags (i.e., DO bit set), domains to resolve (i.e., root or ccTLDs), and query type (i.e., "ANY" or "DNSSEC"). In NTP, various Mode 6 (i.e. "version") and Mode 7 (i.e. "monlist") queries may be used to weaponise a host. However, for Memcached, we did not see the need to identify lots of query strategies, since if a Memcached server has UDP port 11211 open, an attacker may use the "get" request to achieve large BAFs (assuming that there exists at least one key that stores a reasonably large value).

Comparison of Results with Peers. We compare the results obtained for Belgium (BE) and Luxembourg (LU) with those of our peers, who analysed Greece (GR), the Netherlands (NL), France (FR) and Sweden (SE). Regarding **DNS**, authoritative servers are less weaponisable than non-authoritative ones in BE, LU, GR, SE and NL. However, in NL, the top 10% most vulnerable authoritative servers obtain a larger mean BAF (32.56) compared to the one we observe (22.89 in Table 2). This confirms our observation that we do not hold authoritative DNS servers for the domains analysed. In GR, they also notice that "sl." (Fig. 3) is the most weaponisable domain, obtaining BAFs of ≈ 135. Furthermore, the same percentage of FreeBSD servers from GR advertising a buffer size of 1,232 (60%) is shown, obtaining similar median BAFs: 18.75 (Fig. 15 from Appendix A.4) and 21.77, respectively. In NL, there are 5× more servers advertising EDNS0 Buffer Size of 1,232, instead of 4,096. This opposes the patterns we depict: ≈ 2-3× more 4,096, compared to 1,023 (Fig. 10 in Appendix A.4). All countries analysed indicate that servers that use large buffer sizes obtain significantly larger BAFs on average than those that adhere to the Flag Day recommendations, similar to Fig. 9. In NL, they found vulnerable DNS servers running Unlogic Eagle versions, whereas all of the 80 servers we analysed running this DNS version were secure to amplification attacks (Fig. 4). This further explains the need for an Internet-wide analysis, as geolocation is relevant in networking measurement research, and servers located in different regions tend to be configured differently. We reach the same conclusion when checking FR's results. In FR, the vast majority of servers advertise 1,232 (36.8%), whereas in BE and LU, they advertise 4,096 (47.4%). Compared to GR, we don't observe the mass of the BAF distribution to be scattered around large values ($\geq$ 100), and Raiden DNSD and Mirotik servers seem not to achieve exaggerated values, showing that these servers are better configured in BE and LU. However, we reach the same conclusion as in GR, which is that all Raiden DNSD servers advertise by default a buffer size of 4,096.

For **NTP**, there are no vulnerable Cisco IOS servers in GR, and Linux systems are mostly weaponised through "get_restrict" to obtain large BAFs, similar to our observations. In SE, servers hosting Linux are the only ones that answer to "monlist", while in BE and LU, the only severely vulnerable NTP server also uses Linux (BAF of 3.8k). FR continues to be exposed to NTP amplification attacks through "monlist", our peer reporting two servers providing huge BAFs (2.1k and 5.5k). Lastly, even though Censys finds far more NTP servers in NL (30k) than in BE and LU (14.5k cumulatively), there are fewer servers answering to "monlist" (209 compared to 553), leading us to conclude that BE and LU are more vulnerable to NTP DDoS amplification attacks than the NL network surface.

Analysing **Memcached** systems, no vulnerable servers were found in GR, while there were 12 vulnerable servers reported in NL, one achieving a BAF of 59k. In BE and LU we found only one hitting ≈ 9,500×.

Limitations. The lack of data determined the limited amount of correlations/conclusions we reported. For example, we were unable to find information about the EDNS0 Buffer Size that MicroTik uses (which you can also remark that it is missing from Fig. 13 of Appendix A.5), so the relationship between the buffer size and this vendor's BAFs could not be analysed. Thus, we may have missed multiple correlations between factors, so our list is not exhaustive.

TCP and IP-based inspection were not analysed. However, for IPv6, we tried retrieving data for servers that Censys identified, but no DNS, NTP or Memcached servers of this type were stored. We would have expected even larger amplifications for such servers, based on the observation made by Rossow that the IP header in IPv6 is larger (fixed 40 bytes) than the IPv4 header (ranging from 20 to 60 bytes), even though the BAF metric does not consider IP headers [2].

Moreover, our dataset is limited to Censys and Shodan records, so it does not include all servers from Belgium and Luxembourg. Thus, actively scanning the IP ranges assigned to these countries or conducting a worldwide study would provide more conclusive insights into the observed patterns.

Recommendations. We recommend that network administrators properly configure DNS servers by restricting "ANY" queries [36] and properly adhere to the Flag Day by setting a reasonable buffer size value and switching to TCP if the response size exceeds the buffer threshold. Also, upgrading NTP and Memcached to patched versions, such as "NTP 4.2.7p26" [30] and "Memcached 1.5.6" [63], respectively, is a must towards a more secure and reliable Internet. This way, Mode 7 queries will be disabled by default in NTP, and Memcached will not be exposed on the UDP port. We urge every ISP to implement BCP38 to eliminate IP spoofing [64]. To mitigate loopy attacks, we recommend (similar to Pan et al. [38]) that servers should not respond to errors.

IX Conclusion

Summary. Even though the DNS, NTP and Memcached have been studied for more than thirty years in the context of amplification attacks, our paper confirms that they are still vulnerable to such attacks, as we found amplifiers in the wild that produce large amplification factors. When we find a vulnerable server, we tend to associate it with improper administration on the side of system admins, in the case of DNS. However, vulnerable NTP and Memcached servers usually run outdated, insecure versions of the protocols.

Even though Censys reveals in Belgium and Luxembourg almost twice as many NTP servers as DNS servers, a larger number of DNS amplifiers seem to be weaponisable compared to the NTP ones. However, DNS servers produce way smaller BAFs on average compared to NTP. That being said, an attacker may choose to take a trade-off between obtaining many vulnerable DNS servers that provide BAFs in the range of tens or actively scanning for fewer NTP servers that may be weaponised to obtain BAFs in the range of thousands.

The Memcached servers we analysed seemed to be mostly secure, and both Censys and Shodan reported a very small number of Memcached servers compared to the other protocols. Our research peers have also observed this trend in other countries.

We observed several unexpected findings, such as the DNS server presenting its EDNS0 buffer as having one size, but instead providing a larger response size. This proves that some DNS servers are misconfigured, effectively "lying" about their maximum buffer size. We conclude that this buffer size correlates with the DNS version running on the system, vendors and products.

Based on the recent finding of Pan et al. [38], we used their codebase to find such theoretical application-layer loops from our set of DNS and NTP servers. We conclude that several theoretical loops exist for both DNS and NTP clusters.

Future work. It is relevant to delve deep into adversarial tactics in DDoS attacks since the cyberspace is dynamic. A worldwide study (on a more extensive dataset with better geolocation coverage) would be more conclusive in verifying or opposing the patterns we discovered, which would also cover variations in the network configuration across different regions. Moreover, as we have only analysed DNS, NTP and Memcached, future research may focus on other UDP-based protocols, such as CharGen, to better understand the spectrum of amplification attacks. Lastly, one should explore IPv6 hosts that may be weaponised in such attacks, particularly as the adoption of IPv6 continues to grow rapidly.

References

[1] Cloudflare, "DNS Amplification DDoS Attack," Accessed on April 23, 2024. [Online]. Available: https://www.cloudflare.com/learning/ddos/dns-amplification-ddos-attack/

[2] C. Rossow, "Amplification Hell: Revisiting Network Protocols for DDoS Abuse," *Network and Distributed System Security (NDSS) Symposium*, 2014.

[3] Cloudflare, "Smurf DDoS attack," Accessed on April 23, 2024. [Online]. Available: https://www.cloudflare.com/learning/ddos/smurf-ddos-attack/

[4] B. Sieklik, R. Macfarlane, and W. Buchanan, "TFTP DDoS amplification attack," *Computers & Security*, vol. 57, Oct. 2015.

[5] O. Van Der Toorn, J. Krupp, M. Jonker, R. Van Rijswijk-Deij, C. Rossow, and A. Sperotto, "ANYway: Measuring the Amplification DDoS Potential of Domains," *International Conference on Network and Service Management (CNSM)*, 2021.

[6] Cloudflare, "NTP Amplification DDoS attack," Accessed on April 23, 2024. [Online]. Available: https://www.cloudflare.com/learning/ddos/ntp-amplification-ddos-attack/

[7] Akamai, "State of the Internet Security Report (Attack Spotlight: Memcached), 2018," Accessed on April 23, 2024. [Online]. Available: https://www.akamai.com/site/en/documents/state-of-the-internet/soti-summer-2018-attack-spotlight.pdf

[8] D. Wagner, D. Kopp, M. Wichtlhuber, C. Dietzel, O. Hohlfeld, G. Smaragdakis, and A. Feldmann, "United We Stand: Collaborative Detection and Mitigation of Amplification DDoS Attacks at Scale," *ACM Conference on Computer and Communications Security (CCS)*, 2021.

[9] Cloudflare, "Memcached DDoS attack," Accessed on April 23, 2024. [Online]. Available: https://www.cloudflare.com/learning/ddos/memcached-ddos-attack/

[10] Cloudflare, "Famous DDoS Attacks — The largest DDoS attacks of all time," 2024, Accessed on June 13, 2024. [Online]. Available: https://www.cloudflare.com/learning/ddos/famous-ddos-attacks/

[11] Cloudflare, "The DDoS That Almost Broke the Internet," Accessed on May 7, 2024. [Online]. Available: https://blog.cloudflare.com/the-ddos-that-almost-broke-the-internet

[12] C. Cimpanu, "AWS said it mitigated a 2.3 Tbps DDoS attack, the largest ever," Jun. 2020, Accessed on April 23, 2024. [Online]. Available: https://www.zdnet.com/article/aws-said-it-mitigated-a-2-3-tbps-ddos-attack-the-largest-ever/

[13] Cloudflare, "DDoS threat report for 2024 Q1," Apr. 2024, Accessed on May 8, 2024. [Online]. Available: https://blog.cloudflare.com/ddos-threat-report-for-2024-q1/

[14] ——, "How DNSSEC works," Accessed on May 8, 2024. [Online]. Available: https://www.cloudflare.com/dns/dnssec/how-dnssec-works/

[15] L. H. Newman, "A 1.3-Tbs DDoS Hit GitHub, the Largest Yet Recorded," Mar. 2018, Accessed on June 5, 2024. [Online]. Available: https://www.wired.com/story/github-ddos-memcached/

[16] DDoS-GUARD, "What is Memcached DDoS Attack?" 2024, Accessed on May 8, 2024. [Online]. Available: https://ddos-guard.net/en/terms/ddos-attack-types/memcached-ddos-attack

[17] N. Mishra, S. Pandya, C. Patel, N. Cholli, K. Modi, P. Shah, M. Chopade, S. Patel, and K. Kotecha, "Memcached: An Experimental Study of DDoS Attacks for the Wellbeing of IoT Applications," *Sensors*, vol. 21, no. 23, p. 8071, 2021.

[18] H. Griffioen, K. Oosthoek, P. Van Der Knaap, and C. Doerr, "Scan, Test, Execute: Adversarial Tactics in Amplification DDoS Attacks," *CCS 2021: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, Nov. 2021. [Online]. Available: https://doi.org/10.1145/3460120.3484747

[19] Cloudflare, "What is IP Spoofing?" 2024, Accessed on June 13, 2024. [Online]. Available: https://www.cloudflare.com/learning/ddos/glossary/ip-spoofing/

[20] ——, "What is DNS? — How DNS works," Accessed on May 8, 2024. [Online]. Available: https://www.cloudflare.com/learning/dns/what-is-dns/

[21] R. Yazdani, M. Jonker, and A. Sperotto, "Swamp of Reflectors: Investigating the Ecosystem of Open DNS Resolvers," *Passive and Active Measurement - 25th International Conference, PAM 2024, Proceedings*, 2024.

[22] P. Mockapetris, "Domain names - Implementation and Specification," RFC 1035, Nov. 1987, Accessed on May 9, 2024. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc1035

[23] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "DNS Security Introduction and Requirements," RFC 4033, Mar. 2005, Accessed on May 9, 2024. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc4033

[24] J. da Silva Damas, M. Graff, and P. A. Vixie, "Extension Mechanisms for DNS (EDNS(0))," RFC 6891, Apr. 2013, Accessed on May 10, 2024. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc6891

[25] DNS Flag Day, "Dns flag day 2020," DNS Flag Day Official Website, 2020, Accessed on May 28, 2024. [Online]. Available: https://www.dnsflagday.net/2020/

[26] D. R. Conrad, "Indicating Resolver Support of DNSSEC," RFC 3225, Dec. 2001, Accessed on May 10, 2024. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc3225

[27] D. Mills, "Network Time Protocol (Version 1) Specification and Implementatio," RFC 1059, Jul. 2013, Accessed on May 11, 2024. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc1059

[28] B. Haberman, "Control Messages Protocol for Use with Network Time Protocol Version 4," RFC 9327, Nov. 2022, Accessed on May 9, 2024. [Online]. Available: https://datatracker.ietf.org/doc/rfc9327/

[29] J. Hart, "R7-2014-12: More amplification Vulnerabilities in NTP allow even more DRDOS attacks," May 2019, Accessed on May 27, 2024. [Online]. Available: https://www.rapid7.com/blog/post/2014/08/25/r7-2014-1

2-more-amplification-vulnerabilities-in-ntp-allow-eve n-more-drdos-attacks/

[30] National Institute of Standards and Technology (NIST), "CVE-2013-5211 Detail," CVE-2013-5211 Detail. Accessed on June 15, 2024. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2013-5211

[31] Shadowserver Foundation, "NTP Version Report," 2024, Accessed on June 9, 2024. [Online]. Available: https://www.shadowserver.org/what-we-do/network-reporting/ntp-version-report/

[32] Cybersecurity and Infrastructure Security Agency (CISA), "UDP-Based Amplification Attacks," Jan. 2014, Accessed on May 8, 2024. [Online]. Available: https://www.cisa.gov/news-events/alerts/2014/01/17/udp-based-amplification-attacks

[33] C. Hashemi-Pour, "What is the CIA Triad?" 2024, Accessed on June 5, 2024. [Online]. Available: https://www.techtarget.com/whatis/definition/Confidentiality-integrity-and-availability-CIA

[34] Cloudflare, "What is a DDoS Botnet?" 2024, Accessed on June 4, 2024. [Online]. Available: https://www.cloudflare.com/learning/ddos/what-is-a-ddos-botnet/

[35] E. Bohte, "Evaluation of current state of amplification-based DDoS attacks," 2024.

[36] J. Abley, O. Gumundsson, M. Majkowski, and E. Hunt, "Providing Minimal-Sized Responses to DNS Queries That Have QTYPE=ANY," RFC 8482, Jan. 2019, Accessed on May 11, 2024. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc8482

[37] M. Kührer, T. Hupperich, C. Rossow, and T. Holz, "Exit from Hell? Reducing the Impact of Amplification DDoS Attacks," *USENIX '14*, 2014.

[38] Y. Pan, A. Ascheman, and C. Rossow, "Loopy Hell(ow): Infinite Traffic Loops at the Application Layer," 2024, Preprint.

[39] Z. Durumeric, D. Adrian, A. Mirian, M. Bailey, and J. A. Halderman, "A Search Engine Backed by Internet-Wide Scanning," *CCS '15: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, Oct. 2015.

[40] J. Matherly, "Shodan: The World's First Search Engine for Internet-Connected Devices," 2024, Accessed on May 24, 2024. [Online]. Available: https://www.shodan.io

[41] Internet Assigned Numbers Authority (IANA), "Root Zone Database," 2024, Accessed on May 24, 2024. [Online]. Available: https://www.iana.org/domains/root/db

[42] Technical University of Munich, "Technical University of Munich - TUM," Website, 2024, Accessed on June 11, 2024. [Online]. Available: https://www.tum.de/en/

[43] Technical University of Munich (TUM), "Open PageRank Archive," 2024, Accessed on May 24, 2024. [Online]. Available: https://toplists.net.in.tum.de/archive/openpagerank/

[44] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank Citation Ranking : Bringing Order to the Web," in *The Web Conference*, 1999.

[45] NTP Pool Project, "NTP Pool Project: Public Time Server Synchronization," 2024, Accessed on May 24, 2024. [Online]. Available: https://www.ntppool.org/en/

[46] Y. Pan and C. Rossow, "GitHub - cispa/loop-DoS: Repository for application-layer loop DoS," 2024, Accessed on May 27, 2024. [Online]. Available: https://github.com/cispa/loop-DoS

[47] Linux Die, *fpdns(1) - Linux man page*, 2024, Accesssed on May 27, 2024. [Online]. Available: https://linux.die.net/man/1/fpdns

[48] Network Time Foundation, *ntpd - Network Time Protocol (NTP) daemon*, 2024, Accessed on May 27, 2024. [Online]. Available: https://www.ntp.org/documentation/4.2.8-series/ntpd/

[49] R. Rohith, M. Moharir, and G. Shobha, "Scapy - a powerful interactive packet manipulation program," in *2018 International Conference on Networking, Embedded and Wireless Systems (ICNEWS)*. IEEE, 2018.

[50] M. Gorman, *Chapter 8: Slab Allocator*, 2024, Linux Kernel Archive; Accessed on May 27, 2024. [Online]. Available: https://www.kernel.org/doc/gorman/html/understand/understand011.html

[51] R. V. Rijswijk-Deij, A. Sperotto, and A. Pras, "DNSSEC and its potential for DDoS attacks: a comprehensive measurement study," in *Proceedings of the 2014 Conference on Internet Measurement Conference*. ACM, Nov. 2014, pp. 449–460.

[52] R. Harwood, "Validate and secure DNS responses using DNSSEC on DNS Server in Windows Server," Jun. 2023, Accessed on May 27, 2024. [Online]. Available: https://learn.microsoft.com/en-us/windows-server/networking/dns/validate-dnssec-responses

[53] S. Hotz and M. Sawyer, *dig(1) - Linux man page*, Accessed on May 30, 2024. [Online]. Available: https://linux.die.net/man/1/dig

[54] Z. Durumeric, E. Wustrow, and J. A. Halderman, "ZMap: Fast Internetwide Scanning and Its Security Applications," in *Proceedings of the 22nd USENIX Security Symposium*, 2013.

[55] V. Nitu, "AmplificationAttacksInTheWild_BE_LU," 2024, Accessed on June 21, 2024. [Online]. Available: https://github.com/vlad-nitu/AmplificationAttacksInTheWild_BE_LU

[56] Netherlands Organisation for Scientific Research (NWO), "Netherlands Code of Conduct for Research Integrity," Accessed on June 19, 2024. [Online]. Available: https://www.nwo.nl/en/netherlands-code-conduct-research-integrity

[57] Dark Reading, "High-Bandwidth NTP Amplification DDoS Attacks Escalate 371 Percent in the Last 30 Days," Nov. 2024, Accessed on May 30, 2024. [Online]. Available: https://www.darkreading.com/cyberattacks-data-breaches/high-bandwidth-ntp-amplification-ddos-attacks-escalate-371-percent-in-the-last-30-days

[58] H. Griffioen and C. Doerr, "Quantifying autonomous system IP churn using attack traffic of botnets," in *Proceedings of the 15th International Conference on Availability, Reliability and Security*. ACM, 2020.

[59] NLnet Labs, “Unbound Project - Download Page,” NLnet Labs, Accessed on May 30, 2024. [Online]. Available: https://nlnetlabs.nl/projects/unbound/download/

[60] ISC, “BIND 9 Administrator Reference Manual - Chapter 7,” Read the Docs, 2024, Accessed on May 28, 2024. [Online]. Available: https://bind9.readthedocs.io/en/latest/chapter7.html

[61] MikroTik, “MikroTik Routers and Wireless - Software,” Accessed on May 29, 2024. [Online]. Available: https://mikrotik.com/software

[62] Internet Systems Consortium, “BIND 9 Configuration Reference — Minimal ANY,” 2024, Accessed on June 14, 2024. [Online]. Available: https://bind9.readthedocs.io/en/latest/reference.html#namedconf-statement-minimal-any

[63] Memcached, “Release Notes 1.5.6,” Accessed on May 31, 2024. [Online]. Available: https://github.com/memcached/memcached/wiki/ReleaseNotes156

[64] P. Ferguson and D. Senie, “BCP38 - Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing,” 2000, Accessed on June 13, 2024. [Online]. Available: https://www.ietf.org/rfc/bcp/bcp38.html

A Appendix

A.1 Retrieving data from search engines

In this subsection, we present the commands used to retrieve hosts running DNS and NTP using the CLI tool provided by Censys [39], as well as to retrieve hosts running Memcached using the CLI tool provided by Shodan [40].

Censys queries

1. Retrieve all DNS servers running on UDP/53

```
censys search '(location.country="Belgium"
or location.country="Luxembourg")
and services.port: 53
and services.service_name: DNS'
--per-page 100 --pages 82
```

2. Retrieve all NTP servers running on UDP/123

```
censys search '(location.country="Belgium"
or location.country="Luxembourg")
and services.port: 123
and services.service_name: NTP'
--per-page 100 --pages 150
```

3. Retrieve all Memcached servers running on UDP/11211

```
censys search '(location.country="Belgium"
or location.country="Luxembourg")
and services.port: 11211'
--per-page 100 --pages 1
```

Shodan queries

1. Retrieve all DNS servers running on UDP/53

```
shodan download --limit 18000
shodan_dns_servers.json.gz
'country:BE,LU port:53'
```

2. Retrieve all NTP servers running on UDP/123

```
shodan download --limit 17100
shodan_ntp_servers.json.gz
'country:BE,LU port:123'
```

3. Retrieve all Memcached servers running on UDP/11211

```
shodan download --limit 420
shodan_memcached_servers.json.gz
'country:BE,LU port:11211'
```

A.2 Crafting packets with Scapy

In this subsection, we present snippets of Python code (which are part of our open-source codebase [55]) to craft and send network packets with Scapy [49], as well as how we measured the amount of traffic received from triggering such a request.

DNS packets

1. Filtering for open resolvers query ("A" query type to resolve domain "google.com")

```
domain = 'google.com'
pkt = IP(dst=ip) /
    UDP(sport=RandShort(), dport=53) /
    DNS(rd=1, qd=DNSQR(qname=domain,
        qtype=1)
    ) # "A" query
```

2. Measure the BAF query

```
# Craft the IP layer
ip_layer = IP(dst=dns_server)

# Craft the UDP layer
udp_layer = UDP(sport=RandShort(), dport=53)

# Query type code:
  # 255 for "ANY"
  # 48 for "DNSKEY"
  # 16 for "TXT"
# Craft the DNS query layer
dns_query_layer = DNS(ad=1,
    qd=DNSQR(qname=domain, qtype=query_type_code),
    ar=DNSRROPT(z=0x8000, rclass=4096)
)

# Stack the layers
query = ip_layer / udp_layer / dns_query_layer
```

NTP packets

1. Filtering for responsive servers query (Mode 3 - Client query)

```
pkt = IP(dst=ip) /
    UDP(sport=RandShort(), dport=123) /
    NTPHeader(mode=3) # Mode 3 (Client)
```

2. Measure the BAF query (Mode 7 - Private query)

```
def craft_ntp_packet(target_ip, req_code):
     # NTP uses UDP port 123 by default
    dst_port = 123
    # Base NTP packet with Private Mode (7)
    data = "\x1F\x00\x03" + req_code + "\x00" * 4
    monlist_packet = IP(dst=target_ip) /
        UDP(sport=RandShort(), dport=dst_port) /
        Raw(load=data)
    # Req_Code 42 (0x2a) corresponds to MON_GETLIS
    # (monlist request)
    return monlist_packet
```

Memcached packets

1. Filtering for responsive servers query ("stats slabs" command)

```
attack_payload =
"\x00\x01\x00\x00\x00\x01\x00\x00stats␣slabs\r\n"
query = IP(dst=ip) /
    UDP(sport=RandShort(), dport=11211) /
    Raw(load=attack_payload)
send(query)
```

2. Measure the BAF query ("get" command); NOTE: Only the steps are depicted; for the full code snippet, check [55].

```
# Step 1: 'stats slabs' to get all slabIDs

# Step 2:
# Foreach 'stats cachedump <slab_id> 0'
# -> Retrieve all keys assoc. w/ 'slab_id'

# Step 3: gets <k1> <k2> ... <kn>

# Step 4: Measure the BAF
```

A.3 Memcached results

Fig. 7 depicts the graph of the theoretical maximum BAF that can be obtained in Memcached. It assumes that each key stores a value of 1MB, which is the maximum value. The BAF function that describes the graph is:

$$\begin{aligned}\text{BAF} &= f(\text{key\_size}, \text{num\_keys}) \\ &= \frac{\text{response\_size}}{\text{request\_size}} \\ &= \frac{\text{num\_keys} \times 1024 \times 1024}{8 + 3 + 2 + \text{num\_keys} \times (1 + \text{key\_size})}\end{aligned}$$

The numerator's $1024 * 1024$ factor comes from our initial assumption that each key stores a value of 1MB = $1024 * 1024$ bytes. The factor of 8 from the denominator comes from the first 8 bytes we craft in the request: `\x00\x01\x00\x00\x00\x01\x00\x00`. The factor of 3 from the denominator comes from the *get* string we add to the request after the first 8 bytes. The factor of 2 in the denominator comes from the trailing `\r\n` bytes. Furthermore, in the $1 + \text{key\_size}$ factor, we add one to each key_size as we have to separate the keys by spaces, and the space byte also counts for the final request size.

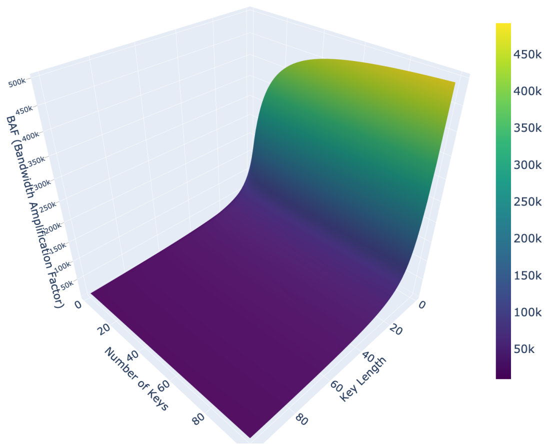


Fig. 7: Theoretical maximum BAF in Memcached. Each key stores a value of 1MB.

A.4 DNS Results

In this subsection, we depict how non-authoritative DNS servers (provided by Censys in BE and LU) perform compared to authoritative servers when resolving "sl." by requesting "ANY" RRs (Fig. 8). We also observe the BAF distribution per EDNS0 Buffer Size when requesting "ANY" from the root domain (Fig. 9). This graph contains the threshold for each buffer size that should not be exceeded if the server is properly configured. Lastly, we display the distribution of hosts according to the buffer size (Fig. 10).

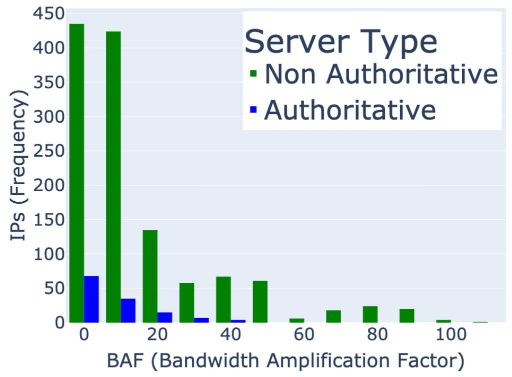


Fig. 8: BAFs performance comparison for "ANY" queries on domain "sl." between authoritative and non-authoritative servers.

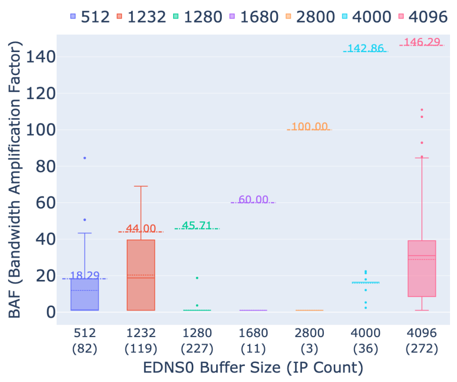


Fig. 9: BAF distribution per EDNS0 Buffer Size, when requesting "ANY" from the root domain. The number of IPs per EDNS0 Buffer Size is depicted in parentheses. The threshold represents the maximum BAF a properly configured server should not exceed.

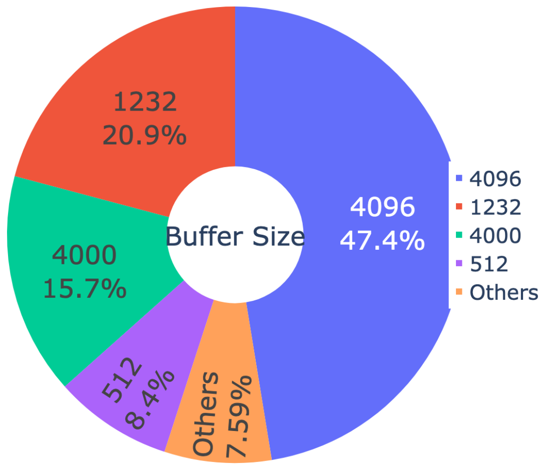


Fig. 10: Servers distribution per EDNS0 Buffer Size.

A.5 DNS factors correlations (heatmaps)

We depict the correlation between several factors that influence the BAF scores a DNS server may obtain in the heatmaps below (Figures 12, 13, 14 and 15). Fig. 11 shows the colours used in all heatmaps. In the heatmaps, each cell depicts the median BAF and the percentage of IPs from the category on the horizontal (OX) axis for requesting "ANY" on the root (".") domain. The numbers on the horizontal and vertical axes represent the count of IPs per category.

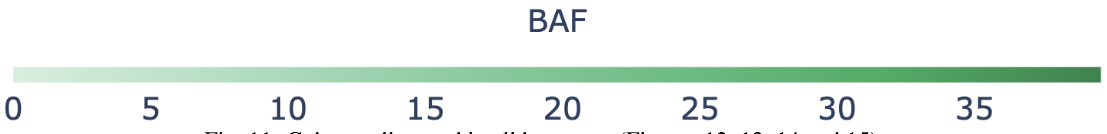


Fig. 11: Colour pallet used in all heatmaps (Figures 12, 13, 14 and 15).

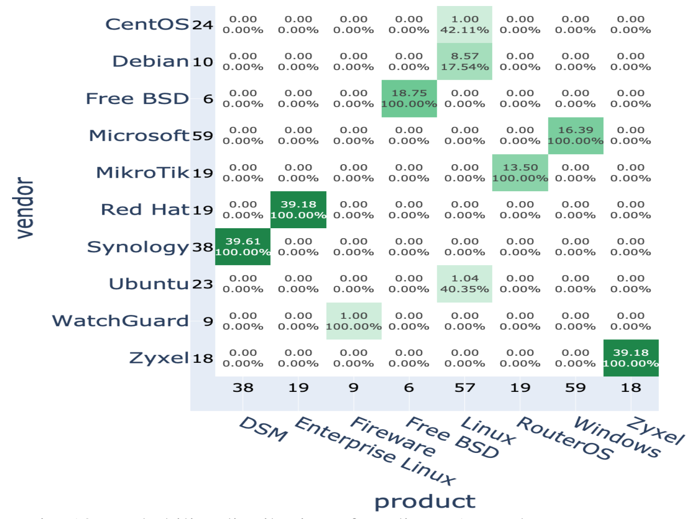


Fig. 12: Probability distribution of median BAF and IP occurrences by product category. Each cell represents the median BAF (percentage of IPs on the second line) for a specific vendor (vertical axis) and product (horizontal axis).

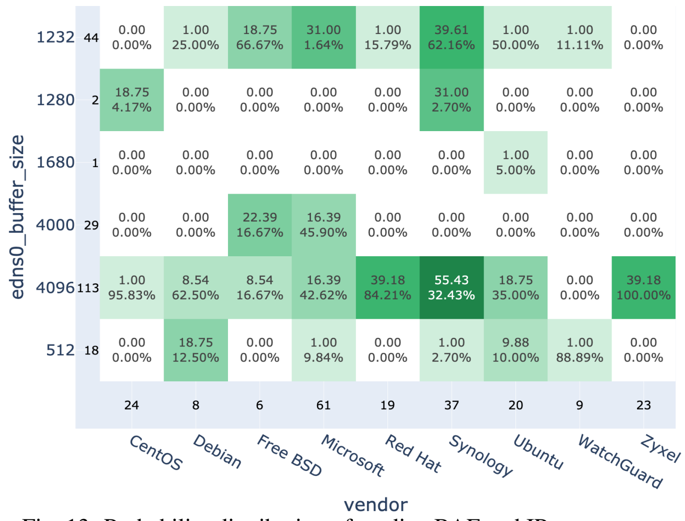


Fig. 13: Probability distribution of median BAF and IP occurrences by vendor category. Each cell represents the median BAF (percentage of IPs on the second line) for a specific EDNS0 Buffer Size (vertical axis) and vendor (horizontal axis).

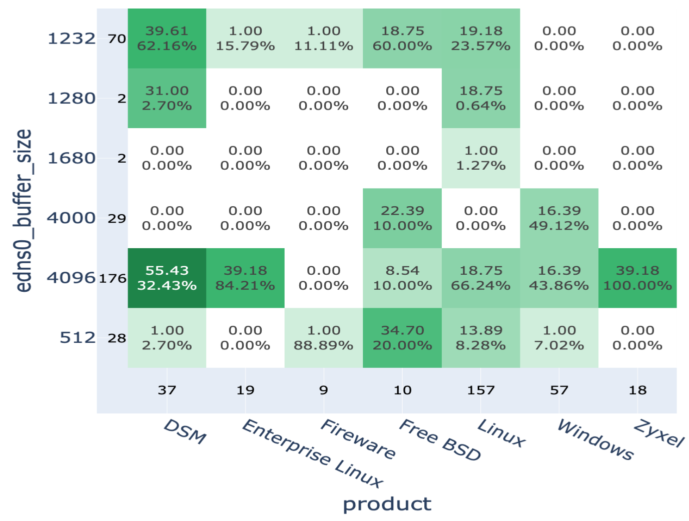


Fig. 14: Probability distribution of median BAF and IP occurrences by product category. Each cell represents the median BAF (percentage of IPs on the second line) for a specific EDNS0 Buffer Size (vertical axis) and product (horizontal axis).

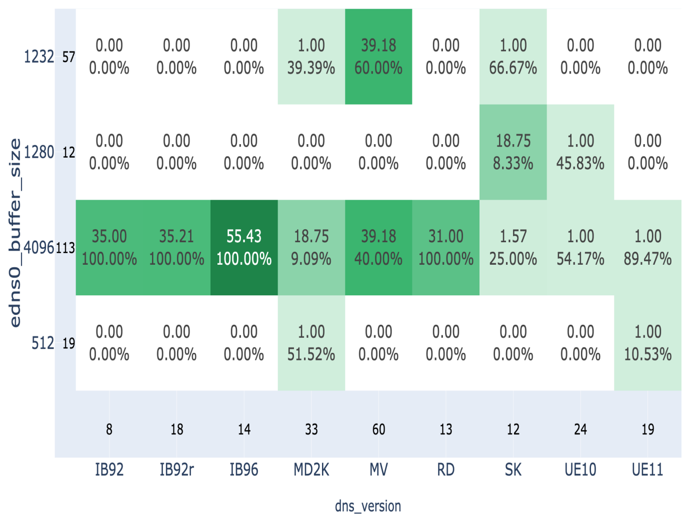


Fig. 15: Probability distribution of median BAF and IP occurrences by DNS version category. Each cell represents the median BAF (percentage of IPs on the second line) for a specific EDNS0 Buffer Size (vertical axis) and DNS Version (horizontal axis). The abbreviations used are the same as in Fig. 4.

However, we have also analysed the potential vulnerabilities of the servers in terms of the **mean BAF obtained by the 10% most vulnerable servers** per category, for which we do not show any plots.

When it comes to the relationship between vendors and how they configure their EDNS0 Buffer Sizes, we observe once again that Zyxel has all its servers (that have their buffer sizes known) configured to use 4,096 as the buffer size value, bringing an average BAF of 80.39 (for the 23 servers we identified). Red Hat has more than 80% of these servers configured to use the same buffer value, yielding a high average BAF of 81. WatchGuard has 88% of their DNSes buffer size configured to 512. This large ratio indicates that this may be the default buffer size with which WatchGuard ships its equipment and may indicate network administrators' absence of proper manual configuration.

A different pattern can be observed at CentOS, which has 95% of its DNS equipment configured to advertise a buffer size of 4,096 while being fully secure against amplification attacks (none of their servers answered our DNS "ANY" queries on the root domain). FreeBSD has two-thirds of its DNS configurations set up to advertise EDNS0 Buffer Size to 1,232, thus seeming to adhere to the DNS Flag Day 2020 recommendations correctly. This also correlates with a smaller-than-average BAF value (i.e., 18.75 for 1,232 EDNS0 Buffer Size).

We also examine the EDNS0 Buffer Size distribution per version of different DNS implementations. We observe unexpected patterns, such as "lying" about adhering to the DNS Flag Day 2020 recommendations by setting up the buffer size value but incorrectly handling the queries, or observe DNS versions that are fully secure under DDoS amplification attacks, even when setting their buffer sizes to improper values. 66% (two-thirds) of the "Symon Kelley dnsmasq" DNS servers reveal buffer sizes of 1,232, and these servers have an average BAF of 50 (which is higher than the average, obtaining larger BAFs than the DNS servers that run "dnsmasq" that have EDNS0 Buffer Sizes of 1280 and 4,096, and also larger than the maximum BAF possible if the server is properly configured to not respond with UDP Payloads larger, which is: $\frac{1{,}232}{28} = 44.0$, where 28 represents the UDP Payload size of the minimal DNS request we were able to craft, on the root domain, as the second (red) boxplot shows in Fig. 9).

A similar trend can be seen at Microsoft Windows DNSes, which have 40% of their servers trying to adhere to Flag Day, but are misconfigured and achieve an average BAF of 62. "Meilof Veningen Posaddis" has 60% of its servers' EDNS Buffer Size set to 1,232 and a mean BAF of 39, again showing that not all its servers are correctly adhering to the recommendations. What seems more unexpected is that 40% of its servers still use a buffer size of 4,096, and this misconfiguration may be the potential cause of achieving huge BAFs (of more than 85).

All the "ISC BIND" DNS implementations are poorly configured, with all their servers using an EDNS0 Buffer Size of 4,096, most probably as a default value. They achieve large BAFs: 78, 64 and 58, respectively. From all the results we have gathered, improperly setting the EDNS0 Buffer Size while not securing the DNS server to not answer public queries seems to be a relevant factor in obtaining large BAF scores.

Lastly, all Raiden DNS servers are configured (presumably by default) to 4,096. They obtain an average BAF of 84, which shows that these servers are severely vulnerable to amplification attacks.

A.6 Loopy results

In Fig 16a and 16b, we depict the tables outputted by following the procedure described by Pan et al. [38], by using their open-source code on our DNS and NTP servers [46]. There are 15 loops found in DNS, and 33 in NTP.

(a) Loopy results for DNS servers

Cycle	Number of Involved IPs	Min Edge IP Amount	Min Edge	Number of Pairs
[24, 24]	6	6	[24, 24]	15
Number of all affected IPs: 6				

(b) Loopy results for NTP servers

Cycle	Number of Involved IPs	Min Edge IP Amount	Min Edge	Number of Pairs
[7, 7]	5	5	[7, 7]	10
[2, 2]	5	5	[2, 2]	10
[1, 4, 1]	8	2	[1, 4]	12
[4, 4]	2	2	[4, 4]	1
Number of all affected IPs: 20				

Fig. 16: Combined loopy results for DNS and NTP servers.