DNS Amplification Attacks in the Wild

A Honeypot-Based Study of Adversary Tactics, Techniques, and Procedures

Ruben van Unen[1]

Supervisor(s): Harm Griffioen[1]

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

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 18, 2026

Name of the student: Ruben van Unen
Final project course: CSE3000 Research Project
Thesis committee: Harm Griffioen, Mitchell Olsthoorn

Abstract

DNS amplification is one of the most common forms of distributed denial-of-service (DDoS) attack, yet most large-scale measurements of how attackers abuse DNS in practice date back about a decade. This paper revisits the problem by deploying a DNS amplification honeypot on a university research cluster and analysing the traffic it received over an eleven-day period in 2026. The honeypot behaved as an open resolver, forwarded queries to a real upstream resolver so that responses looked genuine, logged every packet, and rate-limited sources to avoid being weaponised. We collected 3,513 packets from 502 unique source addresses and used the logs to study the temporal pattern of the traffic, the geography and networks of the sources, the domains and query types abused, and the behavioural profiles of the sources. We found that the great majority of traffic was internet-wide scanning rather than attacks: only five sources crossed a request-count attack threshold, and only one of these showed behaviour consistent with a genuine amplification attack. ANY queries were still present (9.3% of packets) but were no longer dominant, and we observed almost no DNSKEY or NSEC3 queries, which suggests the shift anticipated after RFC 8482 has not clearly occurred in our data. Our most notable finding is a coordinated group of sources that repeatedly issued a single high-amplification query (`google.com` TXT, amplification factor 28) without ever crossing the attack threshold, showing that a purely count-based classifier misses the most amplification-relevant behaviour. We discuss why the number of observed attacks was lower than expected and release our code and anonymised dataset to support reproducibility.

1 Introduction

Distributed Denial of Service (DDoS) attacks are one of the most persistent and disruptive threats on the modern internet. By flooding a target with traffic until it becomes unreachable, attackers can take down services ranging from gaming servers to critical banking infrastructure. Among the many variants of DDoS attacks, UDP amplification attacks are particularly dangerous because they allow an attacker to generate an enormous volume of attack traffic while only sending a fraction of it themselves. The principle is simple: the attacker sends small requests to publicly accessible servers, forging the source IP address to appear as the victim. These servers then send their much larger responses directly to the victim. The use of reflectors to hide the attacker and multiply traffic was already described by Paxson in 2001 [4]. By abusing thousands of such servers simultaneously, even an attacker with limited bandwidth can generate attacks exceeding hundreds of gigabits per second.

The Domain Name System (DNS) is one of the most widely abused protocols for this purpose. Rossow [5] systematically documented 14 UDP-based protocols vulnerable to amplification abuse and showed that DNS amplification factors can reach up to 98 times the original request size when authoritative name servers with DNSSEC are targeted. The introduction of EDNS0, an extension that raises the maximum DNS response size from 512 bytes to over 4000 bytes, was a key enabler of this threat. Van Rijswijk-Deij et al. [7] further demonstrated that DNSSEC-signed domains make the problem significantly worse, with ANY queries against DNSSEC-signed zones producing amplification factors of an average of 47x and reaching up to 179x in extreme cases. These large responses arise because DNSSEC adds cryptographic signatures (RRSIG records) and public keys (DNSKEY records) to every DNS response, inflating their size considerably.

Despite this well-documented threat, defenders lack up-to-date insight into how adversaries actually exploit DNS in practice. Existing large-scale studies, most notably Krämer et al. [2], who deployed 21 honeypots across multiple protocols and observed over 1.5 million attacks, were conducted in 2015. Since then, countermeasures such as RFC 8482 [1] have discouraged the use of ANY queries, and the DNS ecosystem has evolved significantly in terms of DNSSEC deployment and open resolver availability. It is therefore unclear which DNS query types and attacker techniques are dominant today, and whether the patterns that were documented a decade ago still hold.

This paper addresses that knowledge gap by deploying a DNS amplification honeypot and analysing the attack traffic it receives. The honeypot responds to incoming DNS queries as a real open resolver would, logging every request with its source IP, query type, queried domain, and the sizes of both the request and the response. Since attackers spoof the victim's IP address in amplification attacks, the source IP of every incoming packet during an attack corresponds to the victim rather than the attacker, enabling victim-side analysis. The study focuses on the following research question: *how is DNS abused in practice to launch DDoS amplification attacks, and what tactics, techniques, and procedures do adversaries use?*

To answer this, four sub-questions are investigated. First, what are the temporal characteristics of observed attack traffic? Second, where are the victims located in terms of country and autonomous system? Third, which DNS query types and domains are most frequently abused, and what amplification factors do they produce? Fourth, can distinct attacker profiles be identified from the observed traffic patterns?

The main contributions of this paper are: (1) a detailed measurement of current DNS amplification traffic using a purpose-built honeypot deployed in 2026, (2) a comparison of the observed patterns with the 2015 AmpPot baseline [2], and (3) a characterisation of distinct scanner and attacker profiles based on query behaviour, together with the observation that the most amplification-relevant behaviour is not captured by the standard request-count classifier.

To support reproducibility, all honeypot and analysis code, together with the anonymised dataset collected during this project, is publicly available at https://github.com/CommeRex/DNS-honeypot.

The remainder of this paper is organised as follows. Section 2 reviews related work on amplification attacks and honeypot measurement. Section 3 provides background on DNS amplification and describes the honeypot design and methodology. Section 4 reflects on the ethical considerations of the research. Section 5 presents the results for each sub-question. Section 6 discusses the findings in context. Section 7 concludes and outlines directions for future work.

2 Related Work

Reflection and amplification attacks. The idea of abusing third-party servers to reflect and amplify traffic toward a victim was first analysed in detail by Paxson, who described how spoofed requests to reflectors hide the attacker and multiply attack volume [4]. Rossow later gave the first systematic measurement of amplification across protocols, identifying 14 vulnerable UDP-based protocols and reporting their amplification factors, with DNS and NTP among the most powerful vectors [5]. These works established the threat model and the bandwidth amplification factor (BAF) metric that we use throughout this paper.

DNS-specific amplification. Van Rijswijk-Deij et al. studied DNS amplification in depth and showed that DNSSEC-signed zones substantially increase the achievable amplification, because the cryptographic records DNSSEC adds make responses much larger. They measured average factors near 47 for ANY queries and up to 179 in extreme cases [7]. Partly in response to this class of abuse, RFC 8482 later allowed resolvers to return minimal answers to ANY queries, reducing their amplification potential on compliant resolvers [1]. A central question for any present-day measurement is whether attackers still rely on ANY or have moved to other high-yield query types, which our SQ3 analysis examines directly.

Open resolvers and honeypot measurement. Kührer et al. showed that the population of open amplifiers is large and is continuously re-discovered through internet-wide scanning, and argued for reducing this population as a mitigation [3]. The most directly comparable work to ours is AmpPot by Krämer et al., who built dedicated amplification honeypots, deployed 21 instances worldwide, and observed over 1.5 million attacks across several protocols in 2015, characterising attack durations, victims and the use of honeypots by scanners [2]. Our study follows the AmpPot honeypot approach and reuses its rate-limiting and request-count classification so that results are comparable, but differs in three respects: it is a recent (2026) single-site measurement rather than a large global deployment, it focuses specifically on DNS, and it highlights a limitation of the request-count classifier that AmpPot relies on, namely that coordinated low-rate reflector probing with very high amplification is not captured by it.

3 Background and Methodology

3.1 UDP Amplification Attacks

In a UDP amplification attack, an adversary exploits the stateless nature of UDP to redirect traffic towards a victim. Unlike TCP, UDP does not require a handshake before data is exchanged, which means a server will send a response to any incoming request without verifying that the source IP address is genuine. An attacker can therefore craft a request with the victim's IP address as the source, causing the server to send its response to the victim instead. If the response is significantly larger than the request, the attacker effectively multiplies their own bandwidth at the victim's expense. This ratio is called the Bandwidth Amplification Factor (BAF) and is defined as the number of response bytes divided by the number of request bytes [5].

What makes amplification attacks particularly effective is that a single attacker can abuse thousands of servers simultaneously. The victim then receives traffic from many different source addresses, which makes filtering difficult and the attack hard to attribute. Rossow found that 14 common UDP-based protocols are vulnerable to this form of abuse, with BAFs ranging from 3.8 for BitTorrent up to 4670 for NTP [5]. For DNS, the average BAF sits around 28 to 54 depending on whether open resolvers or authoritative name servers are used.

3.2 DNS as an Amplification Vector

DNS translates human-readable domain names into IP addresses and is fundamental to how the internet works. Because every device on the internet relies on DNS, it cannot simply be blocked at a firewall without breaking connectivity entirely. This makes it an attractive and persistent target for amplification abuse.

Two protocol extensions significantly increase the amplification potential of DNS. The first is EDNS0 [7], which allows DNS responses to exceed the original 512-byte limit and reach up to 4096 bytes or more. The second is DNSSEC, a security extension that adds digital signatures to DNS records. While DNSSEC makes DNS more secure by preventing cache poisoning, the additional records it introduces (RRSIG signatures and DNSKEY public keys) make responses much larger. Van Rijswijk-Deij et al. measured that ANY queries against DNSSEC-signed zones produce an average amplification factor of around 47, and in extreme cases up to 179 [7].

Attackers typically abuse DNS by sending ANY queries, which instruct a resolver to return all record types for a domain at once, producing the largest possible response. An attacker can either register a specially crafted domain designed to return a large response, or simply use an existing DNSSEC-signed domain for which the ANY response is naturally large. The latter approach is harder to detect and filter because the traffic looks identical to legitimate queries for real domains [7].

RFC 8482, published in 2019, formally discouraged the use of ANY queries by allowing resolvers to return a minimal response instead of all record types [1]. While this reduces the amplification potential of ANY queries on compliant resolvers, attackers may have shifted to other query types such as DNSKEY or NSEC3 that still produce large responses on DNSSEC-signed zones. Whether such a shift is visible in practice is one of the questions this study investigates.

3.3 Honeypot Concept

A honeypot is a server that is intentionally set up to look vulnerable, with the goal of attracting and observing malicious

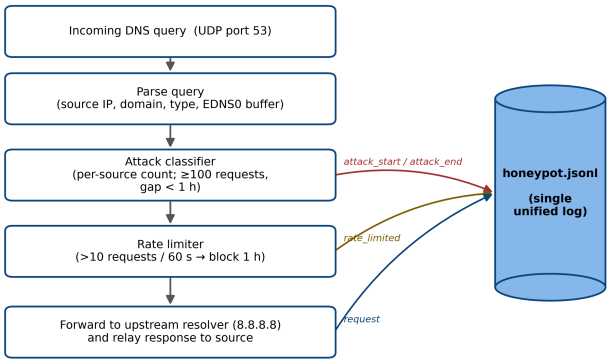

Figure 1: Overview of the honeypot. Each incoming query passes through the classifier and the rate limiter before being answered by the upstream resolver, and every stage writes its event (`request`, `rate_limited`, `attack_start` / `attack_end`) to one unified log file.

traffic. Because a honeypot does not serve real users, any traffic it receives is almost certainly from scanners or attackers, which makes it a useful tool for studying attacker behaviour without relying on production traffic data that is often unavailable for privacy reasons. For amplification research, the honeypot pretends to be a misconfigured open resolver that answers any DNS query from any source. When a scanner or attacker sends spoofed requests to it, the honeypot records the source IP (the victim's address during an attack), the query type, and the domain being abused, following the approach of AmpPot [2] discussed in Section 2.

3.4 Honeypot Design

The honeypot is written in Go and listens on UDP port 53, the standard DNS port. When a packet arrives, the honeypot parses the DNS query and forwards it to a real upstream resolver (Google's 8.8.8.8). The response from the upstream resolver is then sent back to whoever sent the original query. The key reason for using a real upstream resolver, rather than generating a fake response, is that real responses include genuine DNSSEC records, which are large and make the honeypot look like an actual open resolver to an attacker's scanner. If the honeypot sent back a synthetic or empty response, scanning tools would likely mark it as a broken server and not use it as a reflector.

Each incoming packet is handled in its own goroutine so that the listener never blocks. Every packet is logged as a JSON object to a newline-delimited log file before the response is sent. The logged fields are described in Table 1. Attack events from the classifier are also written to the same log file with a different `event_type` (`attack_start` and `attack_end`), so the full picture of request traffic and classified attack sessions is available in one place. Figure 1 gives an overview of this processing pipeline and shows how the different event types all end up in the single log file.

Table 1: Fields recorded for each incoming DNS request. The amplification factor is not stored directly but can be derived as `response_size / request_size`.

Field	Description
`event_type`	`request` or `rate_limited`
`timestamp`	UTC timestamp (RFC3339Nano)
`source_ip`	Source IP (victim's IP during an attack)
`source_port`	UDP source port
`queried_domain`	Fully qualified domain name
`query_type`	DNS record type (ANY, A, TXT, etc.)
`edns_payload`	EDNS0 buffer advertised by the requester (0 if no EDNS0 OPT record present)
`request_size`	Size of the incoming packet in bytes
`response_size`	Size of the outgoing response in bytes

3.5 Rate Limiting

To prevent the honeypot from being used as a genuine amplification weapon, a rate limiter was added. We use a sliding window approach: if a source IP sends more than 10 requests within a 60-second window, it is blocked for one hour. The first packet that triggers the block is logged with `event_type: rate_limited`. Subsequent packets from the same IP are dropped silently.

A sliding window was chosen over a fixed time bucket because a fixed bucket resets on a schedule, which means a burst of traffic arriving at the boundary between two buckets can exceed the limit without triggering a block. A sliding window always counts requests relative to the current time, so bursts are caught immediately regardless of when they occur. The threshold of 10 requests per 60 seconds matches the one used by Krämer et al. [2], which allows direct comparison with their results.

3.6 Attack Classification

Not all traffic hitting the honeypot represents an actual DDoS attack. Some traffic comes from security researchers doing internet-wide scans, and some is background noise. To separate genuine attack traffic from scanners and background noise, we follow the same classification approach used by Krämer et al. [2]: a source IP is classified as an ongoing attack if it sends at least 100 consecutive requests with no gap of more than one hour between packets. When the gap exceeds one hour, the attack is considered over and an `attack_end` event is written to the log.

The classifier runs on every incoming packet before the rate limiter is consulted. This means that the request count used for attack classification includes packets that are later dropped by the rate limiter. An IP can therefore be classified as an attack source even after the honeypot has stopped responding to it.

3.7 Deployment

The honeypot was deployed on a Linux virtual machine with sixteen static public IPv4 addresses, hosted on a university research cluster. It was brought online on 29 May 2026 and ran until 9 June 2026, an observation period of about eleven days.

Logs were stored locally on the VM in newline-delimited JSON format and copied to a backup location daily. Configuring the multiple public addresses to respond correctly required handling asymmetric routing on the host, so that replies left through the same interface and source address on which the query arrived.

3.8 Data Enrichment and Analysis

The log data was processed using a Python pipeline built on pandas and matplotlib. Before analysis, traffic originating from the honeypot's own networks (loopback, the internal management network, the assigned public ranges, and the university and national research networks) was filtered out, because it does not represent external traffic.

Because the rate limiter drops packets after the threshold is reached, the individual dropped packets are not logged, even though the classifier counts them. To avoid undercounting attack volume, we reconstructed the dropped packets for each classified attack session: the logged packets of a session were replicated, preserving their query type and size mix, until the session contained as many packets as the classifier recorded. This correction is valid because the rate limiter drops packets based only on arrival rate and not on their content. All packet counts in this paper include these reconstructed packets unless stated otherwise.

For geolocation and network mapping (SQ2), each source IP was enriched using the Team Cymru IP-to-ASN service [6], which returns the country and autonomous system of an address. We use this service because it is freely available and provides both pieces of information in a single query. We note that, because amplification attacks spoof the victim address, the source addresses we observe are mostly the real addresses of scanners rather than spoofed victims, so the geography we report describes where the traffic originates rather than a true victim map.

For SQ1 we plot the volume of traffic over time and report the duration and rate of each classified attack event. For SQ3 we compute the frequency of each query type and domain and the amplification factor for each unique (domain, query type) pair. For SQ4 we assign each source to a behavioural profile based on its amplification factor, packet volume and queried domains. We originally planned to apply DBSCAN clustering for SQ4, but the number of classified attacks was far too small for clustering to produce meaningful groups, so we use a descriptive classification instead.

4 Responsible Research

4.1 Rate Limiting and Misuse Prevention

The most significant ethical concern with running an amplification honeypot is that it actively participates in attacks by responding to spoofed requests. While this participation is intentional and necessary to make the honeypot look like a real reflector, it means the honeypot does send DNS responses to victim IPs. We mitigate this by rate limiting: any source IP that sends more than 10 requests per 60 seconds is blocked for one hour. This ensures that even in the worst case, the honeypot contributes at most a small number of responses per attack before cutting off. The rate limiter also bounds the number of queries the honeypot forwards to the upstream resolver, so it does not place an unfair load on that resolver. This is the same approach used by Krämer et al. [2], who received only four complaints from attack victims during a four-month deployment and reported that none claimed measurable damage.

4.2 Handling Source IP Data

The source IP addresses recorded in the log file can correspond to victims of amplification attacks rather than to the actual attackers. These are real IP addresses belonging to people or organisations who did not choose to appear in our dataset. We handle this data carefully: raw IP addresses are stored only on the VM and in local backups, and are aggregated to country and AS level before being reported. No individual victim IP address appears in any published table or figure.

4.3 Data Retention and Reproducibility

To make this research reproducible, we will publish the full source code of the honeypot and of the analysis pipeline, together with the dataset collected during the project. Before release, all IP addresses belonging to the honeypot itself, to the author's own network, and to the supervisor will be anonymised, so that publishing the data does not expose private infrastructure. The methodology is described in Section 3 in sufficient detail for another researcher to replicate the deployment.

4.4 Use of Generative AI Tools

In accordance with the TU Delft guidelines on the use of generative AI in end projects, we disclose that the generative AI assistant Claude was used during this project. It was used for two purposes. First, it provided code assistance for the Go honeypot and for the Python analysis and visualisation scripts, including generating and debugging code and suggesting analysis and plotting steps. The author designed the honeypot and the analysis pipeline, made all design decisions, and reviewed, tested and verified all code and its output against the collected data. During the development of the analysis, no sensitive data (including specific IP addresses) were shared with this AI tool. Second, the tool was used to improve the clarity, structure and formality of this report by rephrasing and tightening text drafted by the author. The author understood and verified all results and remains solely responsible for the research design, the analytical decisions, the interpretation of the results, and the integrity of the final text.

5 Results

Over the eleven-day observation period the honeypot recorded 3,513 packets after own-traffic filtering and rate-limit reconstruction, of which 2,434 were logged directly and 1,079 were reconstructed dropped packets. These came from 502 unique source addresses in 152 distinct `/24` networks, and covered 168 unique queried domains. Table 2 summarises the dataset. The median amplification factor across all packets was only 1.28, confirming that most traffic produced little or no amplification, while the maximum reached 43.8. The classifier flagged five attack events in total, which

Table 2: Summary of the collected dataset (29 May – 9 June 2026). Packet counts include reconstructed rate-limited packets.

Metric	Value
Total packets (incl. rate-limited)	3,513
Logged packets	2,434
Reconstructed rate-limited packets	1,079
Unique source IPs	502
Unique source `/24` networks	152
Unique queried domains	168
Median BAF	1.28×
Maximum BAF	43.8×
Rate-limit events	39
Classified attack events	5

is fewer than we expected. We discuss the likely reasons in Section 6.

5.1 Overview of Collected Traffic

Within hours of deployment, the honeypot began receiving traffic from external addresses, and several sources were immediately recognisable as internet scanners. We observed queries for `dnsscan.shadowserver.org` (the Shadowserver scanning project) and for hostnames under `internet-measurement.com`, and many sources belonged to networks operated by well-known scanning companies (see Section 5.3). This is consistent with the finding of Krämer et al. that a large fraction of honeypot traffic comes from whitehat scanners [2]. Among these scanners was Censys, which accounted for 52 of our source addresses (see Section 5.3). However, when we checked how our honeypot appeared in the Censys database, its DNS service on port 53 was not recorded as resolving successfully and instead appeared to return a server error. This suggests the honeypot may never have been listed as a working open resolver, which we return to in Section 6. As the following sections show, the large majority of the traffic we collected was scanning and reconnaissance rather than attacks.

5.2 Temporal Characteristics of Attack Traffic (SQ1)

Figure 2 shows the number of packets received per day. Traffic arrived continuously throughout the period rather than in a single spike, which is consistent with a steady background of scanning punctuated by short bursts of higher-volume activity.

The five classified attack events are listed in Table 3. We recall from Section 3 that the classifier marks a source as an attack once it sends at least 100 requests with no gap longer than one hour between them, the same rule used by AmpPot [2]. Because five events is far too small a sample to build the duration distributions reported for the 1.5 million attacks in the AmpPot study [2], we report each event individually. The events fall into two groups. Three were short, high-rate bursts lasting one to two minutes that queried `version.bind` and produced no amplification (BAF around 1). These are reconnaissance scans that happened to cross the request-count threshold. The other two lasted much longer (about 41 and

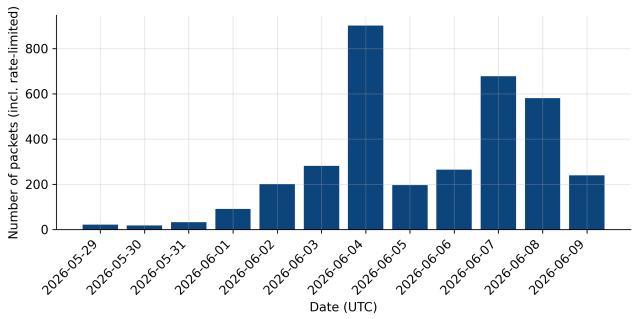

Figure 2: Packets received per day, including reconstructed rate-limited packets.

Table 3: The five classified attack events. "Rate" is the mean packet rate over the event. Only the Yandex event is consistent with a genuine amplification attack.

Trigger domain	Type	Pkts	Dur. (min)	BAF
`version.bind`	TXT	119	1.3	1.0
`yandex.ru`	AAAA	531	40.7	3.5
`version.bind`	TXT	187	1.3	1.0
`version.bind`	TXT	119	2.2	0.9
`dailymail.co.uk`	TXT	200	44.5	2.3

45 minutes) but arrived at a very low rate (0.07 to 0.22 packets per second). Of these, only the event querying Yandex domains (`yandex.ru` and related names, BAF 3.5) showed query behaviour consistent with an amplification attack, with its source most plausibly a spoofed Yandex victim. The other long event queried 57 different domains at a very low rate, which is more consistent with broad scanning than with a focused attack, even though its average BAF slightly exceeded our threshold. All five events lasted well under one hour, which is consistent with the AmpPot finding that amplification activity is short-lived (62% under 15 minutes), but the small number of genuine attacks means we cannot confirm their reported median duration.

5.3 Source Geography and Networks (SQ2)

We resolved each of the 502 source addresses to a country and autonomous system. Table 4 lists the most common source countries by number of unique addresses, and Table 5 the most common source autonomous systems. The United States dominated with 328 of the 502 addresses, followed at a distance by Portugal, the United Kingdom and the Netherlands. The autonomous systems were dominated by cloud and hosting providers (Google Cloud, Hurricane Electric, DigitalOcean, Microsoft) and, notably, by companies whose business is internet-wide scanning, including Censys, Driftnet, Bitsight, ONYPHE and Modat. The strong presence of these networks confirms that the traffic we observed is dominated by scanning infrastructure.

These results should be read as the geography of hosts contacting the honeypot, not as a victim map. Because amplification attacks spoof the victim address, a true victim geography would require a large volume of genuine attack traffic,

Table 4: Top source countries by number of unique source IPs.

Country	Unique IPs	Packets
United States	328	1,342
Portugal	59	84
United Kingdom	31	58
Netherlands	20	890
France	16	95
China	14	62
Hong Kong	9	70
Russia	2	561

Table 5: Top source autonomous systems by number of unique source IPs. Several are dedicated internet-scanning operators.

Autonomous system	Unique IPs	Packets
Google Cloud Platform	139	595
Hurricane Electric	61	101
Censys	52	98
Zenlayer	38	48
Driftnet	26	44
Bitsight (NSEC)	21	36
DigitalOcean	16	65
Microsoft	13	13
ONYPHE	10	14

and in our deployment only the single Yandex event is consistent with a spoofed victim (an address in Russia, which accounts for the small AS count but a large packet count in our data). Krämer et al. reported the United States (32%) and China (14%) as the most common victim countries in 2015 [2]. Our distribution is not directly comparable because it reflects scanner origins rather than spoofed victims, so we cannot draw conclusions about how victim geography has changed.

5.4 Abused Domains and Query Types (SQ3)

Figure 3 shows the distribution of query types across all packets. TXT was the most common type (56.2%), followed by A (23.2%) and ANY (9.3%). The dominance of TXT is explained by reconnaissance rather than amplification: as Table 6 shows, the two most-queried domains were `version.bind` and its upper-case variant, which are standard fingerprinting probes that return a tiny response (BAF 1.0). Most other high-frequency domains also produced little amplification and correspond to scanning, such as the root zone, `cisco.com`, `samsung.com` and `ip.parrotdns.com`.

Regarding the effect of RFC 8482, ANY queries were still present, making up 9.3% of packets with a mean BAF of 4.7 and a maximum of 43.8 (for an ANY query of a DNSSEC-signed domain). However, we observed almost no queries of the alternative high-yield types that attackers were expected to adopt instead: only two DNSKEY queries and no NSEC3 queries appeared in the entire dataset. In other words, a decade after the AmpPot measurements and several years after RFC 8482, ANY is no longer the dominant query type at our honeypot, but it has not been clearly replaced by the DNSSEC-record query types the literature anticipated. The

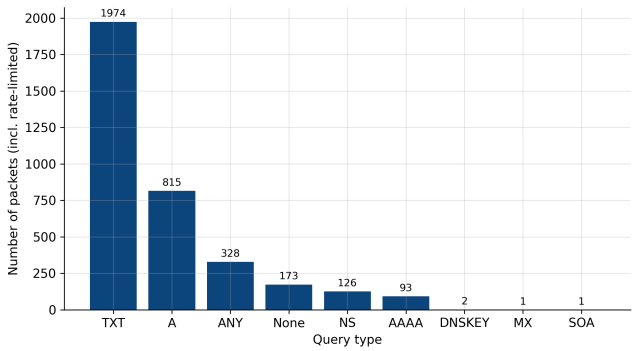

Figure 3: Distribution of DNS query types across all packets, including reconstructed rate-limited packets.

Table 6: Most-queried (domain, query type) pairs, in the style of Table 2 in Krämer et al. [2]. "Src" is the number of unique source IPs.

Domain	Type	Pkts	Src	BAF
`version.bind`	TXT	733	169	1.0
`google.com`	TXT	342	7	28.2
`yandex.ru`	A	315	1	3.8
`VERSION.BIND`	TXT	262	144	1.0
`samsung.com`	TXT	169	1	1.3
`cisco.com`	TXT	125	2	1.3
`dhitc.com`	ANY	106	5	1.2
`ferc.gov`	ANY	54	1	4.9

bulk of traffic is TXT-based reconnaissance and ordinary A lookups. We treat this as a single-honeypot observation that may not generalise.

The genuine amplification in our data was concentrated in a few entries. The clearest example is `google.com` TXT, which produced a mean BAF of 28.2 and was queried by seven distinct sources, six of which belonged to a single network. These sources sent short bursts of the identical high-amplification query and then paused, which is the signature of reflector validation: automated tooling testing whether the honeypot is a usable amplifier before adding it to a list of reflectors. Crucially, none of this traffic was flagged as an attack, because each source sent fewer than 100 requests before pausing and the bursts were spaced more than an hour apart, which reset the classifier's counter. The most amplification-relevant behaviour in our dataset therefore evaded the request-count classifier entirely.

Because amplification is only possible when the resolver is allowed to send a response larger than 512 bytes, we also looked at the EDNS0 buffer size advertised in the received queries (the `edns_payload` field, taken from each incoming request rather than set by the honeypot). Table 7 shows the distribution. About 40% of queries advertised an EDNS0 buffer at all. The most common size was 4096 bytes (24% of all queries), with a smaller group advertising the maximum of 65535 bytes (11%). The amplification factor depended strongly on this value: queries with no EDNS0 buffer had a mean BAF of only 1.85, whereas queries advertising

Table 7: EDNS0 buffer advertised in received queries, with the mean and maximum BAF for each size. Larger buffers enable larger responses, but only translate into high amplification when the queried domain returns a large record.

EDNS0 buffer (bytes)	Share of queries	Mean BAF	Max BAF
none (0)	60.3%	1.85	9.6
512	2.2%	0.76	0.8
1232	1.7%	5.85	17.1
4096	24.2%	13.50	43.8
65535	11.5%	1.80	5.1

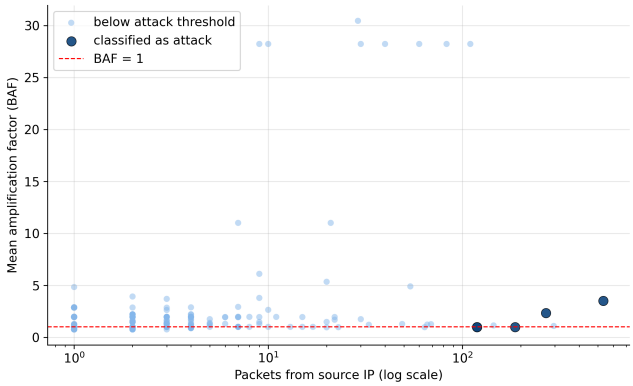

Figure 4: Each source by packet volume (log scale) and mean amplification factor. Sources that triggered the attack classifier are highlighted. The reflector probers sit alone at the top.

4096 bytes had a mean BAF of 13.5 and reached the dataset maximum of 43.8. Every packet with a BAF above 10 advertised an EDNS0 buffer, almost all of them 4096 bytes. Conversely, the queries advertising the maximum buffer of 65535 bytes had a low mean BAF (1.8), because most of them were scanners querying small-response names such as `version.bind`. A large advertised buffer is therefore necessary for amplification but not sufficient on its own. The abused domain must also return a large record. This confirms that the high-amplification traffic we observed deliberately enabled EDNS0 to obtain the largest possible responses, consistent with the behaviour expected of amplification tooling.

5.5 Attacker Profiles (SQ4)

We assigned each of the 502 sources to one of five behavioural profiles, summarised in Table 8. Figure 4 plots every source by packet volume against mean amplification factor and makes the separation visible.

By far the largest profiles are the fingerprint scanners (236 sources querying `version.bind` and similar names with no amplification) and the low-volume background (201 sources sending at most three packets each). Together these confirm that most traffic reaching an open-resolver honeypot is reconnaissance. The high-BAF reflector prober profile contains ten sources with a mean amplification factor of 25, dominated by the `google.com` TXT cluster described above. Despite having by far the highest amplification potential in the dataset,

Table 8: Behavioural profiles of the 502 source IPs.

Profile	IPs	Packets	Mean BAF
Fingerprint scanner	236	1,206	1.0
Low-volume background	201	341	1.6
Other scanner / recon	53	765	1.8
High-BAF reflector prober	10	399	25.0
Amplification attack	2	802	2.9

none of these crossed the attack threshold. The amplification attack profile contains the two sources that both crossed the threshold and produced amplification above the cutoff, but as noted in SQ1 only the Yandex source is a credible focused attack. The central pattern across SQ1, SQ3 and SQ4 is that the request-count classifier captures aggressive scanners while missing the genuinely high-amplification reflector probing, so amplification potential must be considered alongside request volume when profiling sources.

6 Discussion

6.1 Why So Few Attacks?

The clearest result of this study is that we observed far fewer attacks than the AmpPot baseline would suggest, with only one source showing behaviour consistent with a genuine amplification attack. Several factors likely contributed. First, the observation period was short. The project ran for ten weeks in total, and because building, deploying and debugging the honeypot consumed much of that time, the honeypot only collected data for about two weeks. Amplification attacks against any single reflector are intermittent, so a short window naturally captures few of them. Second, the deployment used only sixteen public IP addresses on a single network in one location. AmpPot used many honeypots spread across the world, which both increases the chance of being discovered by attackers and covers more of the address space that attackers scan. Sixteen addresses in one place may simply not be visible enough to attract a representative volume of attacks. Third, our honeypot needs time to be discovered, listed by scanners, and then selected by an attacker as a reflector. As noted in Section 5, our honeypot did not appear to be listed as a working resolver in Censys, since its port 53 seemed to return a server error there. Many attackers select reflectors from such scan databases, so not being listed as a usable resolver would have kept the honeypot largely invisible to them. The reflector-validation probing we observed suggests the discovery process was underway but had not yet resulted in sustained attacks during our window.

6.2 Upstream Resolver Behaviour

A further factor that may have limited the number of attacks is our choice of upstream resolver. The honeypot answered queries by forwarding them to Google Public DNS (8.8.8.8) and relaying the response, so the amplification a prober measured was determined by what 8.8.8.8 returned, not by an authoritative server under attacker control. This has two consequences. First, Google Public DNS implements RFC 8482 and returns a minimal answer to ANY queries, so

the very query type that produces the largest responses against DNSSEC-signed authoritative servers yields only a small response through our honeypot. This is consistent with the modest mean BAF we measured for ANY queries (4.7) and with the fact that our highest amplification came from a TXT record rather than from ANY. Second, when a response over UDP would exceed the requester's advertised EDNS0 buffer, a resolver sets the truncation (TC) bit and returns a reduced response, expecting the client to retry over TCP. Because our honeypot only relays over UDP, any such truncated answer is forwarded as-is and the effective amplification is lower than an attacker would obtain from a purpose-chosen authoritative reflector. The combined effect is that, to a scanner measuring amplification, our honeypot may have looked like a comparatively poor reflector, which could have made it less attractive to attackers and contributed to the small number of attacks we observed. A honeypot that serves large responses from a controlled authoritative zone, rather than relaying a public resolver, would avoid this effect and is a worthwhile change for future deployments. We return to this in Section 7.

6.3 Query Types After RFC 8482

Our data shows that ANY queries, while still present, are no longer the dominant query type, which is consistent with the intent of RFC 8482 [1]. However, we did not see attackers move to DNSKEY or NSEC3 queries as some had anticipated. These were essentially absent. Instead, the highest-amplification traffic we saw used an ordinary TXT query for a popular domain (`google.com`) whose TXT record is large because it contains many sender-policy and verification entries. This suggests that, at least at our honeypot, attackers and scanners look for any query that yields a large response, rather than relying on a specific DNSSEC record type.

6.4 Limitations of Count-Based Classification

The reflector-validation probing for `google.com` TXT is the most important finding for defenders. It had the highest amplification potential in our dataset, came from a coordinated group of hosts in a single network, and yet was never flagged by the request-count classifier that both we and AmpPot use, because the bursts were short and spaced out. A classifier that only counts requests therefore misses exactly the behaviour that precedes the most dangerous attacks. Our results argue for complementing request-count classification with a signal based on amplification factor and coordination, for example flagging repeated high-BAF queries for the same domain from multiple addresses in the same network.

6.5 Implications for Defenders and Limitations

The query types and domains we observed could inform detection rules: a small number of domains account for the high-amplification traffic, and monitoring for high-BAF responses leaving a resolver may catch abuse that request counts miss. Our study has several limitations beyond the short period and small address range already discussed. The honeypot only sees the spoofed victim address during an attack, never the true attacker, so it cannot attribute attacks. Rate limiting, while necessary for ethical reasons, suppresses part of the attack volume. We reconstruct the dropped packet counts but cannot recover the exact content of dropped packets. Finally, the source geography reflects scanners rather than victims, as discussed, so it should not be read as a victim map.

7 Conclusions and Future Work

This paper studied how DNS is abused in practice for amplification by deploying an open-resolver honeypot and analysing eleven days of traffic. In answer to the main research question, the traffic reaching our honeypot was overwhelmingly internet-wide scanning and reconnaissance rather than attacks, and the single clearest attacker technique we observed was the validation of the honeypot as a reflector using a high-amplification query.

For the four sub-questions, our findings are as follows. (SQ1) Attack activity was rare and short-lived: only five sources crossed the attack threshold, all events lasted under one hour, and only one was a credible amplification attack. (SQ2) The source addresses were concentrated in the United States and in cloud and scanning networks, but because attacks spoof the victim, this describes scanning infrastructure rather than victims. (SQ3) ANY queries remain in use (9.3%) but are no longer dominant and have not been replaced by DNSKEY or NSEC3 queries. The highest amplification came from a TXT query for `google.com` (BAF 28). (SQ4) Sources fall into clear profiles dominated by scanners, with a small but important group of high-amplification reflector probers.

The single most important finding is that the most amplification-relevant behaviour, namely coordinated high-BAF reflector probing, was not captured by the standard request-count classifier, which suggests that detection should also consider amplification factor and coordination.

Future work should address the limitations of this study. Running the honeypot for longer and across many addresses in different regions, as AmpPot did, would likely capture more and more representative attacks. Serving large responses from a controlled authoritative zone instead of relaying a public resolver would also avoid the response-size limits discussed in Section 6 and make the honeypot a more convincing reflector. Adding an amplification-aware classifier would allow the reflector-probing behaviour we observed to be detected automatically. Finally, extending the honeypot to other amplification protocols on the same cluster would allow a broader comparison of how different protocols are abused today. To support such work, we release our honeypot and analysis code together with the anonymised dataset collected during this project.

References

[1] Joe Abley, Olafur Gudmundsson, Marek Majkowski, and Evan Hunt. Providing minimal-sized responses to DNS queries that have QTYPE=ANY. RFC 8482, Internet Engineering Task Force (IETF), 2019. https://www.rfc-editor.org/rfc/rfc8482.

[2] Lukas Krämer, Johannes Krupp, Daisuke Makita, Tomomi Nishizoe, Takashi Koide, Katsunari Yoshioka, and Christian Rossow. AmpPot: Monitoring and defending against amplification DDoS attacks. In *Proceedings*

of the 18th International Symposium on Research in Attacks, Intrusions and Defenses (RAID). Springer, 2015.

[3] Marc Kührer, Thomas Hupperich, Christian Rossow, and Thorsten Holz. Exit from hell? reducing the impact of amplification DDoS attacks. In *Proceedings of the 23rd USENIX Security Symposium*. USENIX Association, 2014.

[4] Vern Paxson. An analysis of using reflectors for distributed denial-of-service attacks. *ACM SIGCOMM Computer Communication Review*, 31(3):38–47, 2001.

[5] Christian Rossow. Amplification hell: Revisiting network protocols for DDoS abuse. In *Proceedings of the 2014 Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, USA, 2014. Internet Society.

[6] Team Cymru. IP-to-ASN mapping service. https://www.team-cymru.com/ip-asn-mapping, 2024. Accessed June 2026.

[7] Roland van Rijswijk-Deij, Anna Sperotto, and Aiko Pras. DNSSEC and its potential for DDoS attacks: A comprehensive measurement study. In *Proceedings of the 2014 Internet Measurement Conference (IMC)*, Vancouver, BC, Canada, 2014. ACM.