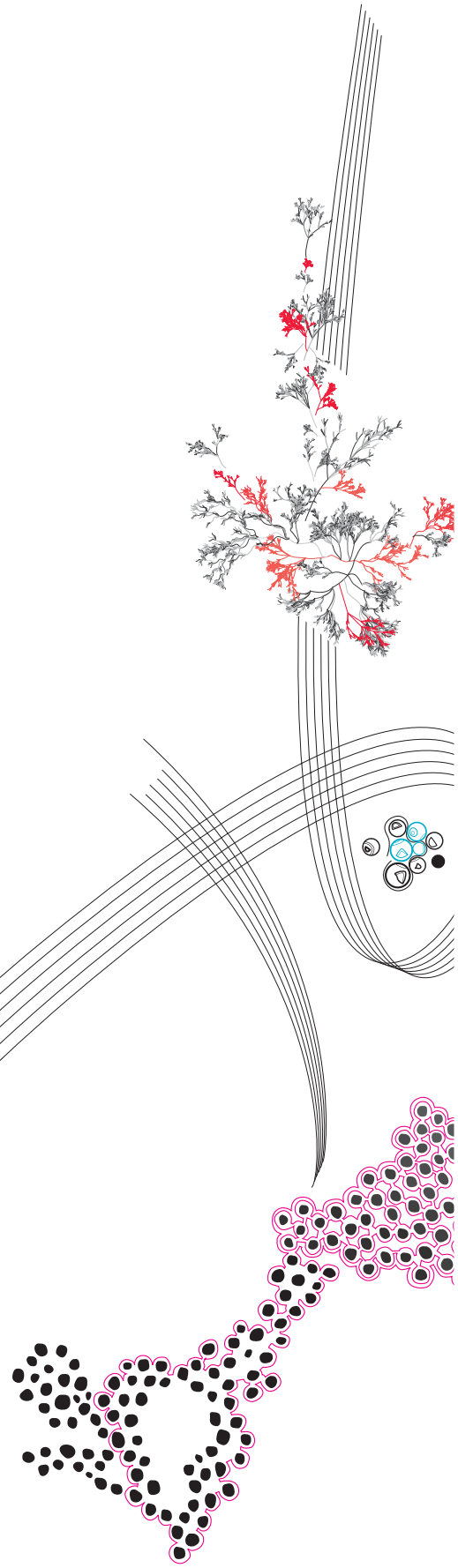

MSc Computer Science
Master Thesis

Exposing Potential Origin Networks of Malicious Spoofed Traffic through Anycast Catchment Mapping

Bas Palinckx

Supervisor: prof.dr.ir. Roland van Rijswijk-Deij
Committee: dr.ir. Raffaele Sommese & dr.ing. Florian Hahn

April, 2025

Department of Computer Science
Faculty of Electrical Engineering,
Mathematics and Computer Science,
University of Twente

UNIVERSITY OF TWENTE.

Abstract

Distributed Denial of Service (DDoS) attacks prevent users from accessing a service by flooding it with excessive, unsolicited traffic. A popular form is the Reflection & Amplification (R&A) attack. These take advantage of legitimate services by reflecting and amplifying their responses to victims. IP spoofing is integral to these attacks, as attackers change their source address to that of the victim, thereby masking their own. This makes retribution against these attackers very challenging. A relatively unexplored approach for exposing information about malicious spoofed traffic is to leverage the unique properties of anycast routing. Under anycast, multiple sites are configured so that clients are automatically connected to the site topologically closest to them. This means that if spoofed traffic arrives at a certain anycast site, the spoofer must be located near that site, regardless of what source address it uses. It is possible to analyze the catchment of networks that an anycast site serves using the Verfploeter technique, a method that relies on active IPv4 wide probing. For this study, we lure spoofed DDoS traffic to the Tangled anycast testbed, consisting of 32 widespread nodes. This is done by deploying the AmpPot honeypot at every location. With this setup, we can create a list of prospect origin networks where a spoofer potentially resides by filtering out networks from catchment measurements. This is done based on the assumed hop count from the spoofer. We validated this method with ground truth data provided by CAIDA's Spoofer project. Based on our test subjects, we can narrow down a spoofer's network by systematically eliminating up to 98% of known autonomous systems and routed prefixes. This sets the stage to determine more precisely where spoofed DDoS attacks originate, which can greatly help in retribution against attackers.

Keywords: DDoS, IP spoofing, Anycast, Catchment mapping, Honeypots

Acknowledgements

First and foremost, I would like to thank my supervisor, Roland van Rijswijk, for his invaluable assistance throughout this project. Your ability to untangle my thoughts and your unlimited enthusiasm were instrumental in setting me on the right track every week. A special thanks also to Remi Hendriks, whom I could always count on for technical help and advice. I am equally thankful to Raffaele Sommese and Bernhard Degen for their valuable ideas and feedback. Next, I would like to express my gratitude to Florian Hahn for being part of the committee as an external member. Also, I would like to thank CAIDA, in particular Matthew Luckie and Kimberly Claffy, for believing in this project and for facilitating a fruitful collaboration. Lastly, I want to express my deepest gratitude to my family and friends for always having my back. Your support made my academic years a fulfilling and truly memorable experience.

Contents

Chapter 1

Introduction

Distributed Denial of Service (DDoS) attacks have wreaked havoc on the internet since its inception. By flooding a victim with a large amount of unsolicited traffic, legitimate users are denied access as the network or computational resources cannot keep up. Over the years, these blunt but very effective attacks have taken on various forms. A popular category in recent times is the Reflection & Amplification (R&A) attack. During these attacks, the unsolicited traffic is not directly sent by a malicious node but reflected off of a benign one. A malicious node forges request packets with the victim's source address and sends them to a benign reflector. This forging is known as IP spoofing, a technique facilitated by some network operators not implementing proper source address validation. The reflector receives the request and returns an unsolicited response to the victim. Amplification happens when the reflector hosts a service that can send large responses based on small requests. By sending certain requests for protocols such as NTP or DNS, the adversary can increase the size of the packets the victim receives. As adversaries mask their actual IP address by changing it to that of the victim, distinguishing spoofed traffic from benign traffic becomes complex. Let alone discovering the origin of the traffic. This is the reason why countermeasures against R&A attacks often revolve around diverting traffic instead of finding the adversary and holding them accountable.

A little-explored approach is investigating what information can be exposed about malicious spoofed traffic by leveraging the unique properties of anycast routing. In an anycast network, multiple sites can announce the same IP address. The BGP tries to route clients contacting this address to the site "most nearby". In other terms, if a client located in The Netherlands sends a request to an address announced by various geographically diverse sites, it has a high probability of being routed to a site located in Amsterdam compared to those further away. This is a process mostly invisible to clients, let alone something they realistically have any control over.

Therefore, valuable insight can be gained if a malicious spoofer weaponizes a reflector hosted on an IP address, which is announced across a topologically wide-spread set of anycast sites. The anycast site receiving the spoofed traffic indicates that the spoofer likely resides in a network close to it. The most notable work on this subject is done by Vries et al. [17]. They mention that a packet arriving at an anycast site with a source address that should not be routed to that site could be a strong indicator for detecting malicious spoofing.

Of course, "close by" is a relative term in networking. Fortunately, a method known

as the Verfploeter technique was designed by de Vries et al. and further improved by Hendriks et al. [18, 24]. This technique enables us to reliably measure which /24 prefixes are routed to which particular sites within an anycast network. This is done by sending ICMP probes from one anycast site to every probable /24 prefix in the IPv4 space. All anycast sites record from which prefixes they receive a reply, which makes up that site's anycast catchment. The implementation by Hendriks et al. can also measure unicast catchments for each site.

The Verfploeter technique has been extensively tested on Tangled, an anycast testbed consisting of 32 worldwide deployed sites conceived by Berthodolo et al. [11]. However, no previous work has analyzed incoming spoofed traffic in an anycast environment besides de Vries. Therefore, we deploy AmpPot on each of Tangled's sites to lure in DDoS-based spoofed traffic. Designed by Krämer et al., AmpPot is a honeypot disguised as an attractive reflector for R&A attacks [35]. It has proven highly effective in attracting DDoS-based spoofed traffic in a controlled manner. This is done by replying to well-known amplifiable requests with real or superficial responses without causing any real damage to the victim. Most protocols popular for their amplification potential are covered by AmpPot, including NTP and DNS.

By having a proven method for both inviting spoofed traffic to our anycast sites and measuring every anycast site's catchment, our research is the first to combine these techniques to expose information about spoofed traffic. This allowed us not only to strengthen the case for using anycast to detect spoofed traffic, as mentioned by de Vries, but also to narrow down the networks from which spoofed traffic likely originates.

1.1 Research goal

The goal of this research is to determine spoofed traffic and examine what information can be exposed about its origin networks with active catchment measurements. Shedding light on this idea can lay the groundwork for developing methods to identify networks where a spoofer might reside, making it possible to respond more effectively to DDoS attacks.

To achieve this goal, we focus on answering the following three questions:

i What characteristics are evident in honeypot traffic originating inside and outside our anycast catchments?

ii How can these characteristics be used to narrow down from which networks spoofed traffic originates?

iii Within which margins can we be sure the spoofers' origin network is contained in our narrowed-down set?

The first question gives a profile of the spoofed honeypot traffic we have measured throughout this work and will describe how this relates to our anycast locations. The second question describes a methodology derived from this knowledge to narrow down networks from which spoofed traffic likely originates. The last question elaborates on how accurate and effective the proposed methodology is in locating a spoofer.

1.2 Thesis structure

The remainder of this thesis is structured in the following chapters: Chapter 2 goes more in-depth on topics related to R&A attacks, anycast catchments and honeypots. Chapter 3 covers previous work related to ours. The results of the first research question are discussed in Chapter 4. Chapter 5 discusses question two, while 6 and 7 focus on the last question. We end this thesis by discussing limitations in Chapter 8 and the subsequent conclusions in Chapter 9.

Chapter 2

Background

2.1 Distributed Denial of Service Attacks

At its core, the internet allows any connected client to communicate with any endpoint freely and as frequently as desired, given the endpoint's IP address is known. This fundamental concept enables Denial of Service attacks. By flooding a victim with abundant traffic, an attacker can deny legitimate use of the victim's services as its resources get depleted. DoS attacks are often done in a distributed manner, where an attacker instructs a large set of nodes (often a botnet) to enlarge the stream of undesired network traffic sent to the victim (DDoS) [41, 52]. These attacks can have serious consequences and are not trivial to mitigate, as demonstrated by recent examples [22, 32, 45]. DDoS attacks can be generalized into two types:

i **Volumetric attacks** rely on overwhelming a victim or the network it is located in by generating a large amount of traffic. The bottleneck can be either the network link not being able to forward all the packets, the victim failing to process all the traffic or a combination of both.

ii **Semantic attacks** rely on depleting the victim's resources via the application layer, instead of the network or transport layer. This can be done by cleverly exploiting or misusing a protocol [50].

2.1.1 Reflection & Amplification Attacks

Reflection & Amplification (R&A) attacks are an advanced category of DDoS attacks. This method introduces a third party called the *reflector*, which is weaponized by the usage of *IP spoofing* and abusing *amplifiable protocols*. The attacker manipulates the reflector by spoofing their own IP address to that of the victim, *reflecting* large volumes of unsolicited traffic to the victim. *Amplification* occurs when an attacker takes advantage of an application protocol hosted on the reflector. The goal is to have a larger response sent to the victim than the initial request sent to the reflector. Amplification is, therefore, quantifiable by the *Amplification Factor* (AF) 2.1. AFs for certain protocols can rise into the hundreds. A study by Rossow et al. demonstrated AFs for NTP up to 556.9 and for CharGEN up to 358.8 [46].

$$AF = \frac{\text{Response Packet Size}}{\text{Request Packet Size}} \tag{2.1}$$

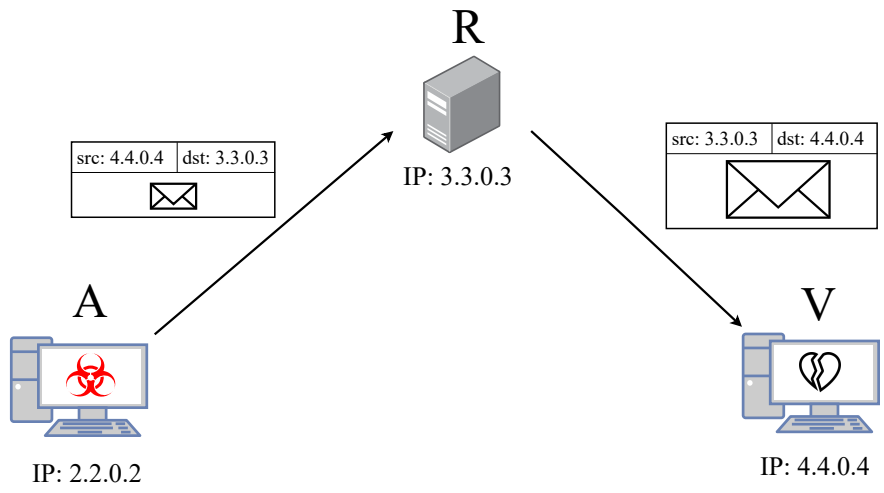

FIGURE 2.1: A simplified diagram picturing the attacker, reflector and victim during an R&A attack.

In more detail, an R&A attack is carried out as follows:

1. Attacker **A** sends a request to reflector **R**. Through spoofing, the source IP of this request is that of victim **V**.
2. **R** receives this request and assumes, based on the source IP, it is coming from **V**. Therefore, it processes the request and sends a larger (or amplified) response to **V** depending on the protocol and request that is used by **A**.
3. **V** receives the large unsolicited response, overwhelming the network link it resides in or draining its resources trying to process it.

A non-distributed R&A attack as described above is shown in Figure 2.1.

2.1.2 IP Spoofing

Any packet sent between two endpoints over the internet is transmitted using the Internet Protocol (IP). Each node, for example, an end device, router, or server is given a unique IP address. These addresses are needed to route packets (IP datagrams) to their desired destination. Therefore, each packet stores a *source IP address* (meaning the address of the sender) and *destination IP address* field in its IP header. Unfortunately, due to the naive implementation of the IP, the source address is often not verified to be the sender's actual address. This flaw enables nodes sending packets to lie about the IP address they originate from [20]. Within the context of DoS attacks, this *IP spoofing* is utilized in the following ways:

- In traditional DoS attacks, attackers often use spoofing to masquerade their IP address, or that of the nodes executing a DDoS attack to avoid retribution.
- An attacker can spoof the source address dynamically, making it very difficult for the victim to filter traffic based on the address it is coming from.

- Upper layers of the OSI model use the source as a reply address, most notably the protocols found in the application layer. For example, if a time request is sent to an NTP server containing a spoofed IP address, the server will send the response to that address without question. This explains why spoofing is essential for R&A attacks to work.

2.1.3 Amplifiable protocols

Not every application protocol is suitable for amplification within an R&A attack. Naturally, it should be able to generate large responses triggered by small requests. Granted, with the nuance that the attacker cannot communicate further past the initial request sent to the reflector, because of IP spoofing. TCP-based protocols require a 3-way handshake to be completed before sending application data is possible, making them complex to weaponize. Although research has been done on TCP reflection with impressive results [12, 36, 37], they are not the prime target for amplification. UDP-based protocols do not require a handshake making direct request-and-response communication for application data possible, more fitting for R&A attacks. Work done by [13, 31, 35, 36, 46] show that many protocols have large amplification potential (for example SSDP, QOTD, CharGEN), but the most commonly exploited are **DNS** and **NTP**.

The DNS (Domain Name System) protocol provides a distributed and hierarchical system for looking up which IP address corresponds to a given domain name. For example, if `www.utwente.nl` needs to be accessed, the web browser queries the DNS to find it needs to contact `130.89.109.227`. Due to its distributed and hierarchical nature, various server types can be used to achieve amplification within the DNS. Open DNS resolvers are a common example [57]. These nodes are the middleman between the client and the various hierarchical levels of name servers: Root, Top Level Domain (TLD), and Second Level Domain (SLD). When openly exposed to the internet, any client on the internet can query these resolvers making them a fitting reflector. However, SLD name servers can be standalone reflectors themselves [39].
Besides choosing the right reflector, attackers have a selection of parameters to enlarge the size of DNS responses:

- `ANY` query types return all available DNS records for a given domain [53].
- `TXT` records can yield large responses, especially for top-ranked or maliciously crafted domains [39].
- The EDNS extension mechanism can be used to enlarge the maximum response length of 512 bytes [39].
- DNSSEC adds cryptographic signatures to DNS responses making them larger, especially combined with the above-mentioned techniques [54].

Apart from DNS, NTP also provides a large amplification potential. The Network Time Protocol's primary use is synchronizing the clock time of clients. However, commands used for diagnostic purposes can yield large responses. The `monlist` command returns the last 600 connected clients [47]. Other examples are the `version` and `showpeers` commands [16].

2.2 IP Anycast

Originally, the internet was conceived with only one-to-one communication in mind, comparable to traditional mail. A sender can send data to one distinct receiver, with both parties being identified by their unique IP address. The IP later expanded the *unicast* approach with options to send data to more than one receiver. *Broadcast* allows data transmission to all nodes in a network and *multicast* to a subset of nodes with a shared interest (during videoconferencing for example). However, these routing methods do not solve the limitations of unicast:

- Redundancy is made difficult when unicast routing is used. If the receiving node goes offline, it is up to the transport or application layer to redirect to a backup node.
- Load-balancing is complex to implement. If there is a need to spread out traffic over multiple receivers to reduce the load on up- or downlink bandwidth, unicast does not provide an integrated solution.
- Latency due to a geographical location is not easily mitigated under unicast. For the fastest experience, clients benefit from using any endpoint physically closest to them.

IP anycast tries to solve these shortcomings by allowing multiple different sites to be advertised under the same IP address. When a client contacts this address, it is routed to the "best" option making one-to-one-of-many communication possible. This is done without relying on extra logic provided by the client, network, or endpoint [40].

2.2.1 Autonomous Systems & BGP

To determine the best option, anycast leverages the **BGP** (Border Gateway Protocol) which is used to exchange routing information across different **AS**es (Autonomous Systems). To grasp the concept of ASes, it is important to understand that the internet is not comprised of one massive network. Rather, out of tens of thousands of large networks, each being an AS. In turn, an AS is responsible for routing a specific range of IP addresses, known as its IP space.

When a node needs to communicate with an IP address outside its AS, the AS's routing table is consulted. BGP makes sure these tables are up-to-date, as routing between ASes constantly changes. Each row consists of an IP address prefix (varying in specificity), the path of all ASes needed to get there and the IP address of the next hop. These prefixes are also known as routed prefixes (**RPFX**es). Every AS can send a BGP announcement indicating a routing change, primarily when a new IP range is acquired or routing policies have changed. These are propagated to all ASes as they update their routing tables and notify their peers accordingly.

2.2.2 Routing approach

An AS will choose the next hop, taking into account the best-fitting prefix and shortest AS path. Under unicast, that will still be possible for only one site. Anycast's advantage of allowing different sites to have the same IP address becomes apparent. By deploying sites in multiple ASes (preferably in various geographical locations), clients are directed to the site with the best-fitting path. This results in lower latency [48], balances (DDoS-based)

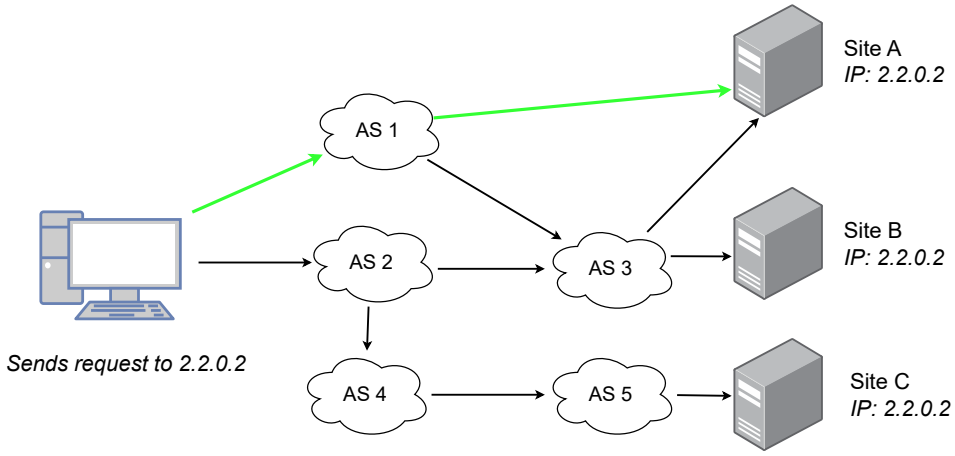

FIGURE 2.2: A simplified diagram of three anycast sites routed under the same IP address. BGP routing makes an effort to select the site with the shortest path.

heavy traffic over multiple sites [28, 42, 56], and makes redundancy possible when properly implemented. This is demonstrated in Figure 2.2.

2.2.3 Adoption & Measurement

Anycast mainly found its footing in DNS shortly after it was introduced for root servers. Today, most TLD servers are anycasted and adoption is rising under authoritative name servers [19, 51]. CDNs, anti-DDoS prevention services and many other entities benefit from the technology as well [14]. However, deploying and managing multiple anycast sites is challenging. BGP routing can be unpredictable, not every AS hop adds the same latency and AS routing policies are not always transparent. This creates the need to analyze and understand which IP prefixes an anycast site is covering. Measuring these *catchments* has been done by various methods.

Catchment mapping is commonly performed using platforms such as RIPE Atlas [2]. These services offer measurements from various distributed VPs (Vantage Points) sending probes. This setup allows for querying anycast services from topologically diverse locations across the internet. By analyzing the responses, the goal is to gain knowledge of catchments by mapping the IP addresses of VPs to anycast sites [42, 48].

2.2.4 Verfploeter

The downsides of using this approach are the added complexity and cost of managing all VPs. Also, deployments of VPs are not unbiased compared to the population of internet users [9]. Therefore, efforts have been made to measure catchments using the anycast sites themselves, rather than relying on a network of active VPs.

Verfploeter is a tool designed to send ICMP echo requests from the anycast network block (or inside the anycast service IP prefix) to almost all /24 IPv4 networks. Catchments can be mapped by recording which anycast site receives the incoming reply [18, 24]. Every anycast site sends the attributes of incoming ICMP replies to a central collector where they are labeled and aggregated, as demonstrated in Figure 2.3. This effectively makes any computer on the internet replying to ICMP pings a *passive VP*, drastically increasing

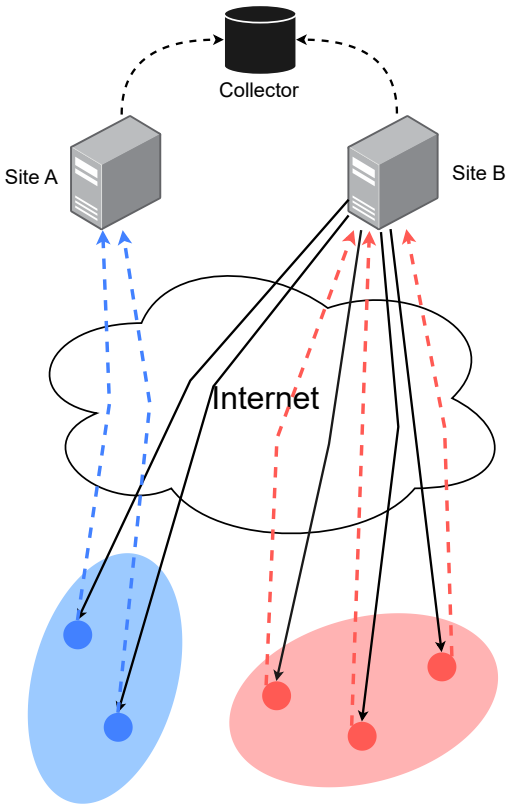

FIGURE 2.3: A diagram picturing Verfploeter's mapping process. The black arrows represent the internet-wide ICMP probing. Via anycast routing, every node's reply ends up at one of the sites. This information is collected and aggregated to map out each catchment.

the granularity of the mapping. The only trade-off compared to measurement platforms is that every anycast site has to participate during a measurement, as all sites need to record incoming echo responses simultaneously for a complete result.

2.3 IP Spoofing prevention

IP anycast has proven to be an effective tool to blunt DDoS-attacks when deployed in a well-designed manner [42, 56]. However, solving the problem by actively detecting and preventing DDoS-attacks is more desired. A large amount of research has been done on devising effective methods to achieve this goal. Artificial intelligence techniques and statistical methodologies can yield the detection of DDoS attacks [29]. However, these approaches have the disadvantage of relying on real-time traffic classification, requiring detection to occur as the attack is already underway. A potentially more proactive method of combating DDoS attacks is the prevention of IP spoofing. As described in section 2.1.2, spoofing is the driving force behind R&A attacks and makes traditional attacks more powerful.

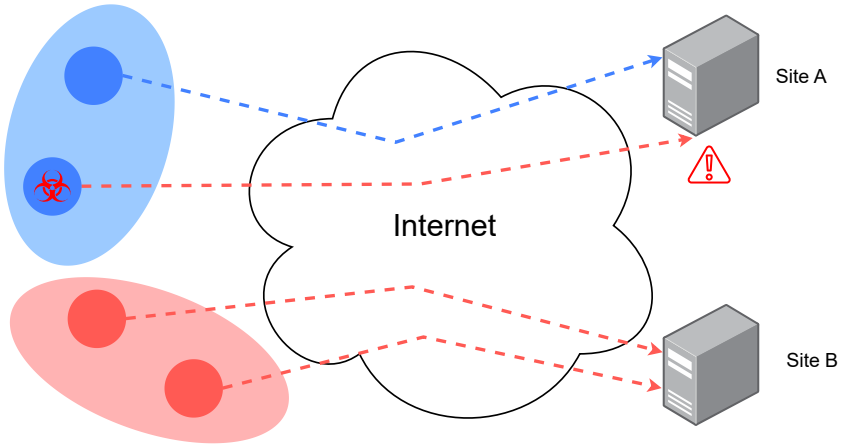

FIGURE 2.4: A diagram picturing two anycast sites. Site A can deduce spoofing as the attacker is using an IP not typically seen in its catchment.

2.3.1 Ingress filtering

Spoofing prevention has been ongoing since very early on the internet's lifespan. RFC 2827 (or BCP38) is a best practice recommendation that advises AS operators to implement ingress filtering on their outgoing traffic [49]. Simply put, if traffic wants to leave with a source IP outside of the prefix the operator is handling, it must be dropped. Implementation of this best practice has been increasing but is still far from widespread, presumably because AS operators do not directly gain any benefit from implementing these network hygiene practices [3].

2.3.2 Catchment-based Detection

BCP38 relies mostly on the goodwill of AS operators, indicating a need for an alternative method against IP spoofing. Anycast shows potential for this purpose. Although a spoofer can lie about its source IP, it cannot manipulate how its packets are topologically routed to their destination. Similarly, the same can be said for which particular anycast site the packets arrive at. Section 2.2.2 explains that clients are routed to the anycast site with the best-fitting path from the AS they reside in. De Vries et al. mention that when a packet arrives at an anycast site with a source address typically not served, it could indicate an instance of spoofing, as displayed in Figure 2.4 [17].

For this approach to be feasible, finely granular and precise mapping of a site's catchment is essential. As described in Section 2.2.4, Verfploeter fits into this need, being able to probe the IPv4 space from a major hitlist and subsequently observe which anycast site receives a reply.

Chapter 3

Related work

This chapter of the report will focus on all previous studies related to the subject matter and how this research plans to expand upon their results. The state-of-the-art in TTL-based spoofing detection will be touched upon together with previous Verfploeter deployments, other methods of catchment mapping and catchment-based spoofing detection. Lastly, important work on amplification honeypots will be discussed.

3.1 TTL-based Spoofing detection

To prevent a packet from ending up in a routing loop due to misconfigured networks, the IP header contains a *Time-to-Live* (TTL) field. RFC 791 demands that every module handling an IP datagram must decrease its TTL value by 1 (representing 1 "hop"). When it reaches 0, the packets must be dropped [4], with each operating system having its own publicly known initial TTL value.

This mechanism can also be utilized to detect spoofed packets. Although an attacker can change the initial TTL value, the number of hops a packet has traveled cannot be manipulated. This means that the TTL of a spoofed packet will not match that of a legitimate one, as the hops decrease based on the attacker's topological location, illustrated in 3.1. Cheng et al. [55] utilized this idea by proposing a two-step method. Legitimate traffic is first observed to create a mapping between clients and their valid hop count. During the enforcement stage, every packet is checked against this mapping and dropped when the TTL is outside a certain range. An improvement proposed by Mukaddam et al. [43] changes the mapping by saving a list of all possible hop counts seen in the past per IP, instead of one value.

Unfortunately, this method falls short in the context of UDP and, therefore, R&A attacks. Traffic is deemed "legitimate" when it has completed a TCP handshake. This classification does not work for UDP, because of its connectionless architecture, as described in Section 2.1.3. Michael et al. [8] try to stretch the idea of TTL-based filtering by implementing active probing for every received packet to verify its hop count. Besides deeming this impractical, they conclude that more fundamental solutions are needed to combat IP spoofing.

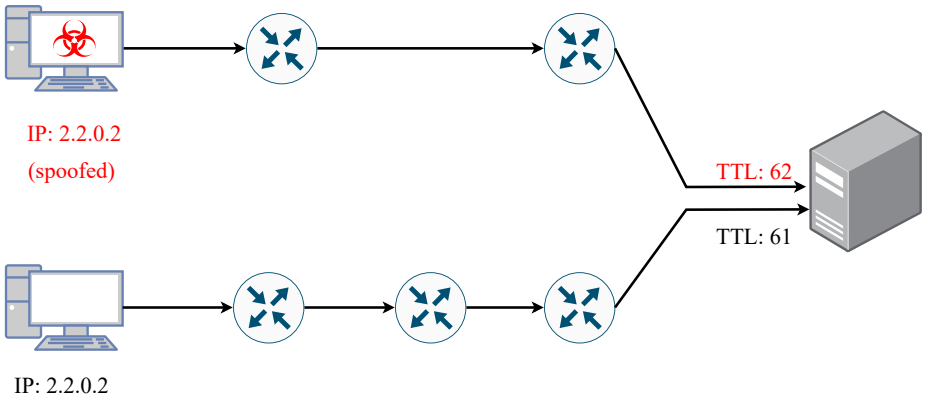

FIGURE 3.1: A simplified diagram of TTL-based spoofing detection. The TTL values do not match up because the attacker sends spoofed traffic from a topologically different location than the legitimate sender.

3.2 Verfploeter deployments

Verfploeter's global probing methodology was first measured on both the B-Root DNS service and the Tangled anycast testbed by de Vries et al. [18]. In both cases, catchments were studied by comparing Verfploeters mapping to the traditional method of probing by active VPs provided by RIPE Atlas. Here Verfploeter's much broader coverage for most populated areas was demonstrated by the lack of probes RIPE Atlas has deployed in areas such as China. Also, Verfploeter greatly outperformed Atlas by the number of observations in areas covered by both methodologies, concluding with the fact that Verfploeter was able to see approximately 430x more blocks compared to Atlas. These highly dense catchments allowed for the discovery of divisions within ASes, by analyzing various catchments occurring inside the same AS. Approximately 12.7% of all ASes seemed to be served by more than one site, where ASes announcing more prefixes often see more sites from their network.

Crucially for our research, they also analyzed the stability of catchments with Tangled. If the set of VPs in a catchment rapidly fluctuates, it becomes a less reliable source for deducing spoofed traffic. Measuring every 15 minutes for a day, they observed that 95% of the 3.71M passive VPs keep responding and stay within their catchment. Approximately, a remaining 2.5% stopped responding and the other 2.5% flipped from catchment. It has to be said that Tangled has greatly expanded its sites since then. Also, these measurements were done over 24 hours, not multiple days or weeks. Although promising, more measurements are needed to prove that catchments are stable enough for further analyses like ours.

3.3 Catchment Mapping

The usage of probing with Verfploeter or a measurement platform such as RIPE Atlas does not encapsulate all work done on mapping anycast catchments. A study done by Microsoft Bing shows the usage of analyzing logs and active client-side participation [44]. This method can get as broad of a result as Verfploeter but is limited by the number of clients able and willing to participate.

A different approach is the analysis of logs made available by anycast operators, as seen with [14, 21]. Naturally, these logs give the clearest picture of anycast catchments possible and are interesting to utilize, but operators giving away their logs is rare due to policy or privacy reasons. Besides, they are not a solution for continuous measurement.

Within the context of DNS, there have been several studies that try to get more insight into what networks a site can serve by enriching probing taking into account latency and geolocation. [10, 15, 38].

3.4 Catchments against Spoofing

De Vries et al. scaled up their work with Verfploeter by deploying it in Cloudflare's anycast network [17]. They demonstrated Verfploeter's potential as a countermeasure against spoofing by first measuring if the IPs in their hitlist (one single address per /24 prefix) are associated with only one anycast site. Results not only show that this is the case for 95% of the IPs, but many ASes and the entirety of their addresses are seen at just one site within Cloudflare's network.

With this knowledge, they deem source addresses arriving at a site outside this mapping to be likely spoofed. This was further demonstrated by measuring a known SYN-flood attack, where a steep rise of unexpected traffic throughout the attack was observed. Although this shows potential for catchments being a strong signal against spoofing compared to only TTL analysis or other methods, questions remain.

Cloudflare has with more than 300 sites the biggest anycast network in the world, explaining the small and specific catchments. Our research allows verifying this technique on the scale of Tangled, having only 32 sites available. Not to mention, SYN-flood attacks do not resemble the traffic seen during R&A attacks as described in section 2.1.1.

3.5 Amplification Honeypots

There is no previous work on deploying honeypots within an anycast network. Fortunately, this is a proven method for attracting spoofed traffic for amplification under unicast deployment. Notable work was done by Krämer et al. building AmpPot, a honeypot deploying a set of services in an amplifiable configuration shown in Table 3.1 [35].

Responses are sent to clients based on a chosen strategy. "Emulated" chooses a random response in a protocol-specific format. "Proxied" sends a legitimate response coming from a real service. Lastly, "agnostic" replies to any request with a large set of random bytes not representing a real response at all. Interestingly, agnostic honeypots missed a sizable amount of attacks compared to proxied and emulated. However, their deployment was not able to conclude which is the better strategy.

In other work, Krupp et al. modify AmpPot to fingerprint scanners probing for amplifiable services to link them to attacks [33]. They expanded their work with AmpPot to correlate R&A attacks to so-called booters, which provide DDoS attacks as a conveniently presented service [34]. Most recently, Griffioen et al. took honeypot analysis to a new level for amplification attacks by deploying a network of over 500 custom honeypots [23]. Interestingly, with their widespread deployment, they concluded that attackers do not care

Protocol	Percentage of all attacks	AF	Scenario
NTP	37.0%	556.9	Section 2.1.3
DNS	28.5%	28 to 54	Section 2.1.3
SSDP	27.3%	30.8	SEARCH request
CharGen	7.0%	358.9	Character generation request
QOTD	$< 0.3\%$	140.3	Quote request
MSSQL	$< 0.3\%$	-	-
NetBIOS	$< 0.3\%$	3.8	Name resolution
SNMP	$< 0.3\%$	6.3	GetBulk request

TABLE 3.1: AmpPot supported protocols and percentage of observed attacks. Included are the AF's and attack scenarios observed by the work of Rossow et al. [46]

if a reflector is clearly a honeypot. Outside of the protocols served by AmpPot, Kondo et al. build their honeypot to further investigate amplification of Memcached [30].

Chapter 4

Data collection

The goal for this study is to determine spoofed traffic and identify a candidate set of offending networks. To achieve this, we compare catchment data with honeypot traffic destined for our anycast network. Aside from de Vries et al. mentioning it in their work, no previous research has been conducted on analyzing DDoS-based spoofed traffic in the context of an anycast environment [17]. Therefore, a setup was built to carry out catchment measurements and invite spoofed traffic into the Tangled anycast testbed. The main goal of this installation is to gain an understanding of which traffic characteristics can be used for comparison against catchment data. With exploratory analysis on which anycast and unicast honeypots are involved in attacking the same target, we build a traffic profile establishing the basis for narrowing down a spoofers' origin network.

In this chapter, we discuss every asset of the measurement setup and provide a data demographic across an observation period of three months for both anycast and unicast honeypot traffic.

4.1 Measurement setup

Several technologies are combined to create the unique measurement setup for this work. Our system consists of 33 nodes. 32 identical measurement nodes spread across the world serve as our vantage points. An essential detail is that each of these 32 nodes are reachable by their **unicast** address and a dedicated **anycast** prefix. This is made possible by the fact that each node can announce this prefix through BGP by leveraging the *Tangled* testbed. One node outside of this setup serves as our collector. Lastly, both take advantage of an S3 instance for data backup. A simplified diagram of the entire infrastructure is pictured in the Appendix A.1. In the sections that follow, we provide more detail on each component of our measurement setup.

4.1.1 Measurement nodes

Our measurement nodes are the central piece of this setup. An exhaustive list of all 32 measurement nodes and their locations can be found in the Appendix A.2, their geographical distribution is drawn on a map in Figure 4.1. In summary, there are eleven deployments in North America, eight in Asia, eight in Europe, two in South America, two in Oceania, and one in Africa. Each node has an identical software stack, which is pictured in Figure 4.3

One of the two core processes running on each measurement node is the honeypot *Amp-*

FIGURE 4.1: A map showing all 32 measurement nodes.

Pot. We choose to configure it to run in "Emulation Mode" for every service it supports, as described in Table 3.1. This means that it reacts to well-known, amplifiable requests with a realistic but superficial response. Every one of these requests is saved in an SQLite file, which is backed up daily to S3. An important detail is that AmpPot listens to traffic destined for **two IP addresses**. One is the node's unique unicast address; the other is an address from the dedicated anycast prefix, which is the same across all measurement nodes. To ensure that these honeypots are not able to do real damage, they are heavily rate-limited. When a not-yet-observed source address arrives at AmpPot, it is allowed to send three initial requests. After that, the /24 prefix of that source address is saved, and AmpPot allows only one request every 60 seconds from any of the addresses in that prefix. Every hour, this list of prefixes is wiped to mask the rate-limiting as much as possible.

The other process running is the MAnycast client, which receives measurement instructions from the collector node. These mainly involve commanding the node to listen for incoming ICMP echo replies and (depending on the type of measurement) send ICMP echo requests. Incoming ICMP replies are then formatted and sent to the collector node.

4.1.2 Collector node

Outside of the 32 measurement nodes in our anycast network, we also deployed an extra collector node. The collector node serves as the home base of this setup, with launching catchment measurements and aggregating data being its main purpose. Its software stack is pictured in Figure 4.2. The *MAnycast server* executes two types of daily catchment measurements. One is an *anycast catchment measurement*, where one node is used to probe a hitlist from an anycast address and all nodes record incoming replies, as described in Section 2.2.4. The other is a *unicast catchment measurement*, where all nodes probe to a hitlist from their respective unicast address. In both cases, the MAnycast server collects and aggregates replies from all nodes to a CSV file and backs it up to S3.

The ANT Lab research group at USC provides the hitlist used for these measurements. This dataset represents a set of addresses in each probable IPv4 /24 block that is most likely to be ICMP echo responsive [1]. We deem a probing result from an address of this

list to apply to the full /24 prefix it belongs to. The collector node also hosts a ClickHouse database where all honeypot traffic data is ingested daily. This enables us to slice through traffic data efficiently while also enabling live monitoring with Grafana.

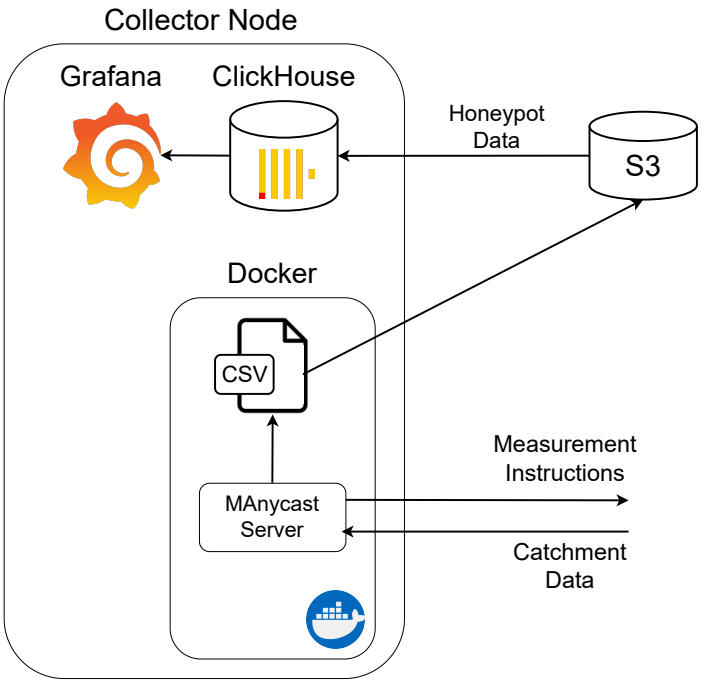

FIGURE 4.2: A diagram showing the core software stack and data flow of the collector node.

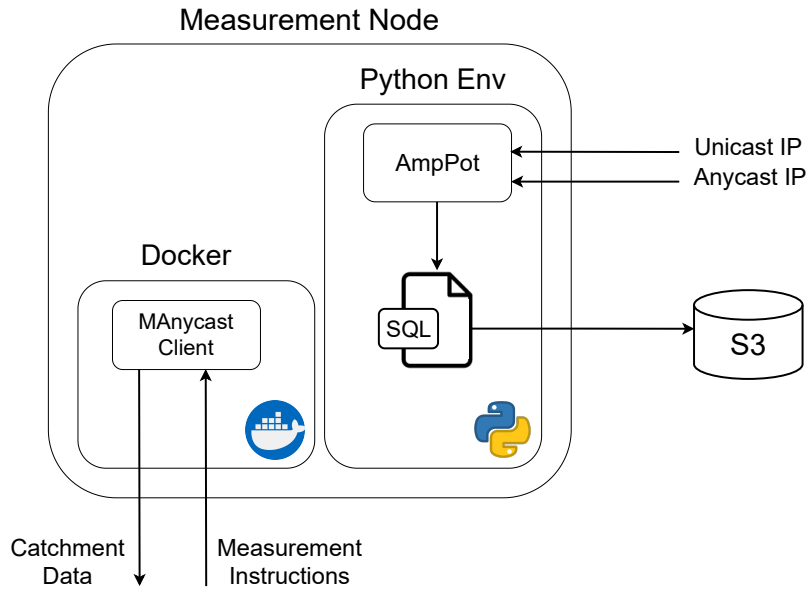

FIGURE 4.3: A diagram showing the core software stack and data flow of a measurement node.

Protocols	Total Request Count
NTP	3,726,693,863
DNS	799,065,650
UPnP	77,353,037
Memcached	37,290,590
LDAP	21,687,567
RPC	436,819
MS SQL	314,495
SNMP	148,213
SIP	102,397
TFTP	78,546
RIP	41,348
SIP	6,829
QOTD	1,906
Jenkins CI	461
NetBIOS	169
Chargen	168

TABLE 4.1: Total amount of requests observed per protocol over all 32 honeypots during our observation period of 82 days.

4.2 Honeypot traffic

In this section, we provide details about the nature of the traffic captured by our honeypots. We do this for every honeypot's unicast and anycast endpoint. Although this work does not focus on fully profiling DDoS attacks, the insight gained from analyzing this traffic is crucial for understanding how we use spoofed traffic in combination with catchment measurements to narrow down prospect offending networks.

Our honeypots were actively monitored from 11 November 2024 to the first of February 2025. During this period, all 32 sites received a grand total of approximately 4.7 billion requests aimed at 1.1 billion unique targets. Adversaries mainly seemed to weaponize NTP, accounting for 79.97% of all observed traffic. DNS made up 17.13%, with other protocols facilitating the remainder of attacks, as seen in Table 4.1. This data demographic is only scoped around DNS and NTP , since these two protocols were the clear favorites for weaponization during our observation period.

4.2.1 Unicast traffic

Firstly, we focus on the number of requests, targets and attacks for unicast traffic to give insight on how our honeypots are reached by adversaries in a more traditional context. Typically, a unicast honeypot observes around 1.2 million NTP and 132,000 DNS requests daily. When it comes to targets, 9,915 distinct source addresses are observed for NTP and 1,878 for DNS daily. These figures are derived from calculating the median number of requests and distinct source addresses for each site during the observation period, followed by calculating the average of these medians. It is safe to say that all sites see a very similar amount of NTP traffic on unicast; DNS is less evenly spread. A full breakdown is shown in Figures 4.4, 4.5, 4.6 and 4.7.

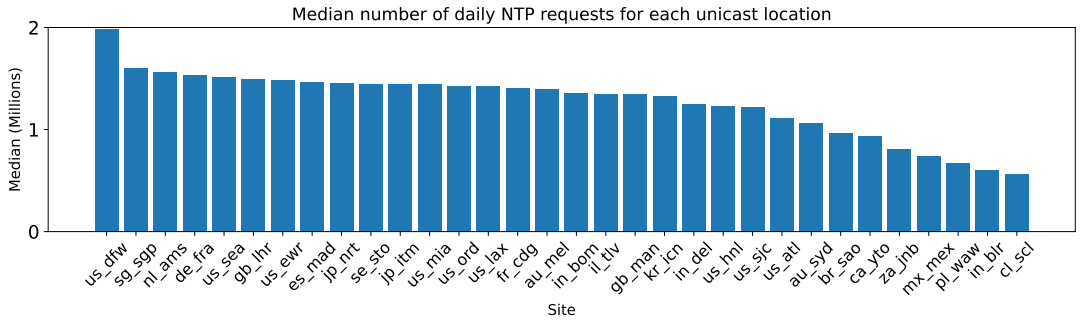

FIGURE 4.4: Median number of daily NTP requests over all unicast locations.

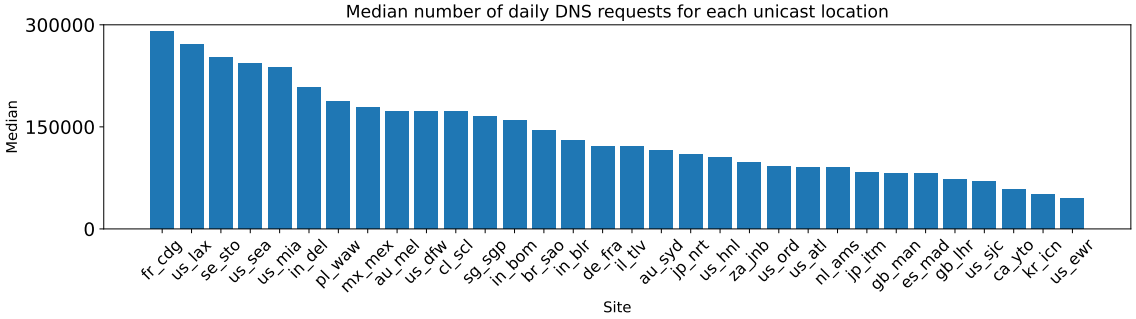

FIGURE 4.5: Median number of daily DNS requests over all unicast locations.

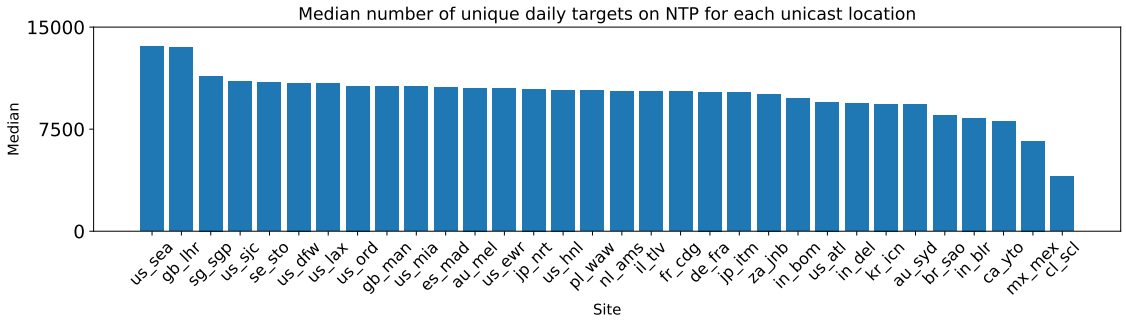

FIGURE 4.6: Median number of daily distinct targets over NTP for each unicast location.

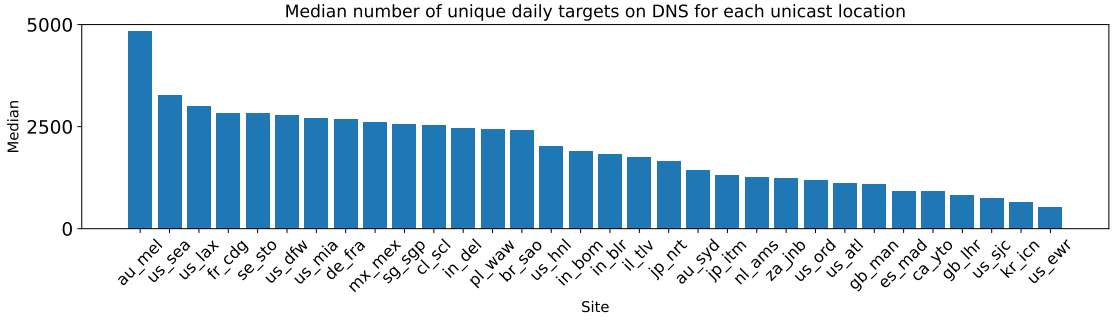

FIGURE 4.7: Median number of daily distinct targets over DNS for each unicast location.

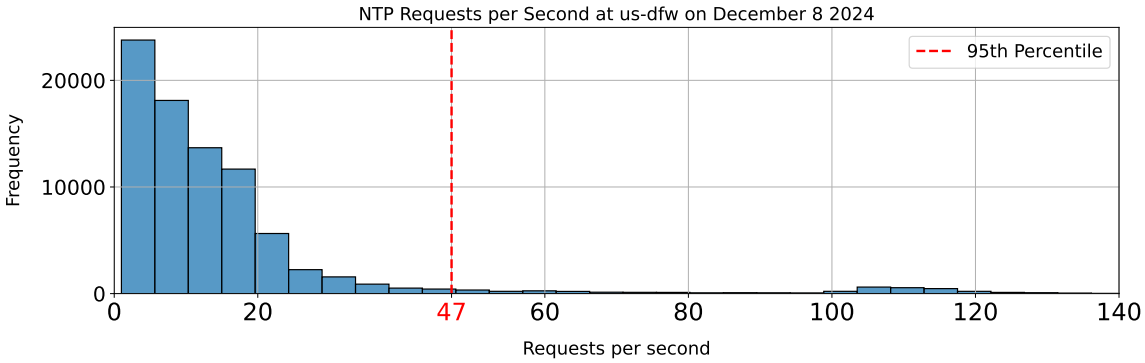

FIGURE 4.8: NTP request per second at the Dallas unicast honeypot on December 8, 2024.

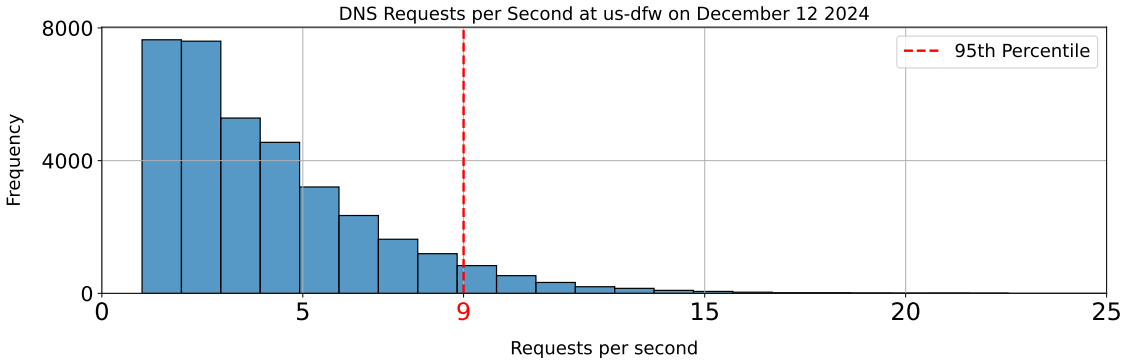

FIGURE 4.9: DNS request per second at the Dallas unicast honeypot on December 12, 2024.

To get a sense of how many sites share the same target within the same day, we zoom in on the 21st of January as an example. This was an above-average day, with our unicast honeypots seeing 10,247 distinct source addresses for NTP and 4,232 for DNS. 80.5% of targets from the NTP set were observed at least once across 29 to 32 sites that day. Interestingly, distribution over DNS is very different, with 75.1% of targets observed at least once at only one to four sites. We leave a more detailed study of the distribution of attack targets to future work.

4.2.2 Observed attacks

Only analyzing targets does not directly tell much about actual attack campaigns. To gain an understanding of how many large campaigns are observed within a day, we assume a major attack is happening during any burst of traffic far above the rate seen under normal circumstances. To get a sense of what the largest bursts of traffic consist of, we choose two days based on a typical number of total requests for both NTP and DNS and plot out their request per second distribution.

On the eighth of December 2024, the us-dfw unicast honeypot observed a total of 1.27 million NTP requests; the request per second distribution is displayed in Figure 4.8. This plot shows that usually, the number of requests per second does not climb higher than 20. For 95% of the day, traffic does not spike above 47 requests per second. This day did see

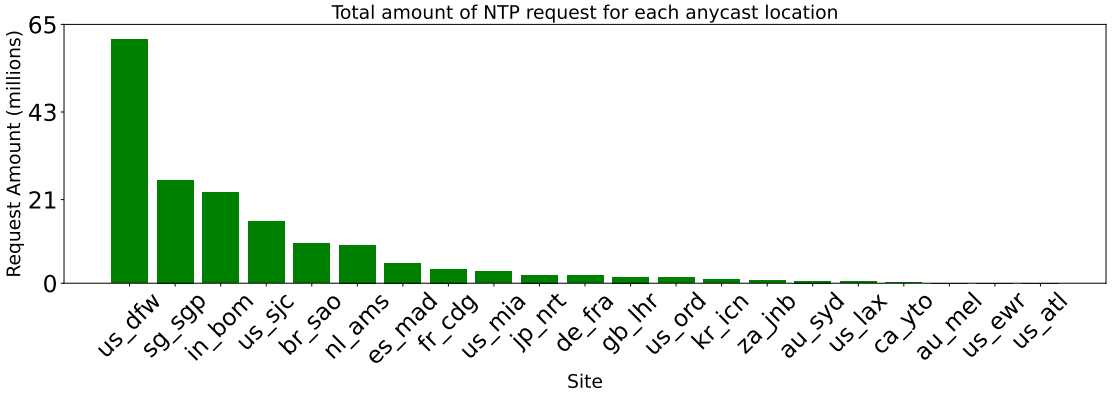

FIGURE 4.10: Total amount of NTP request for each anycast location. Only those with 1 million or more are plotted.

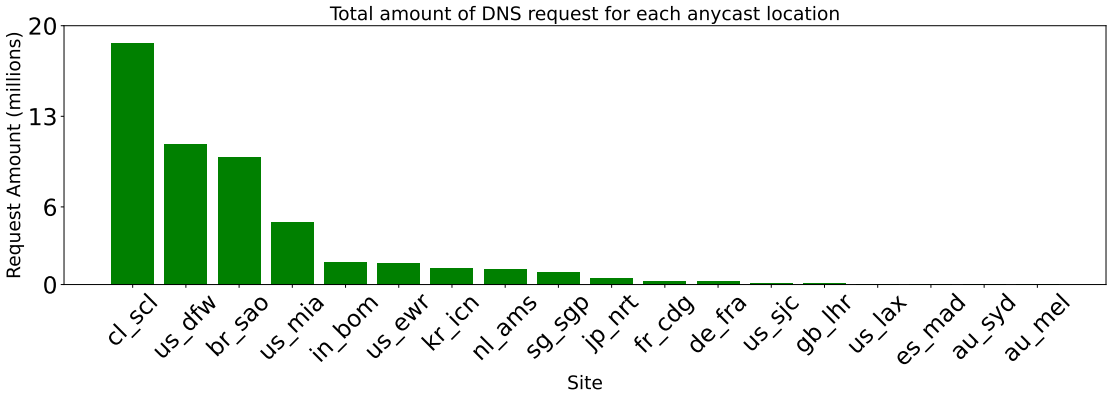

FIGURE 4.11: Total amount of DNS request for each anycast location. Only those with 100,000 or more are plotted.

some large bursts between 100 and 120 requests per second.

For DNS, the same honeypot observed 132,000 requests on the 12th of December, 2024. Figure 4.9 makes it immediately clear that this traffic is a lot more tame, with request per second not reaching above nine for 95% of the day. When traffic surpasses this threshold, the largest rate we see is 17.

It is important to note that relying solely on traffic spikes to define a major attack campaign does not yield a complete picture. There are many cases where a wide set of targets is attacked on a relatively low request per second rate for days or weeks, which could be just as or even more damaging to the victim compared to intense bursts. Nevertheless, there is reason to believe our honeypots are well represented in the DDoS-attack ecosystem, as we confirmed their participation in the attacks targeting multiple educational institutions on January 15, 2025 [26, 27]. Sixteen unicast honeypots participated, all of them attacking the same prefixes. The largest burst we measured spanned seven minutes, where each honeypot received around 21,000 DNS requests.

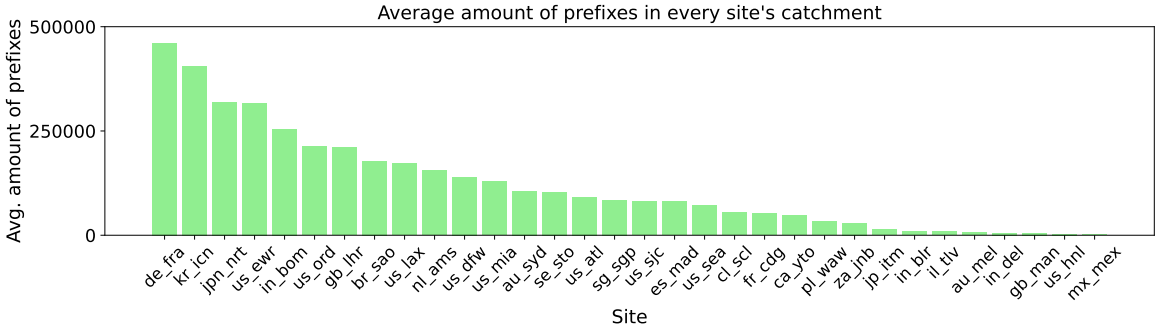

FIGURE 4.12: Average amount of prefixes in every site's catchment measured daily over a week.

Intersection	Overlap
sg-sgp ∩ in-bom	12.6%
in-bom ∩ us-sjc	3.5%
sg-sgp ∩ us-sjc	3.4%
in-bom ∩ br-sao	2.6%
us-dfw ∩ in-bom	0.8%
us-dfw ∩ br-sao	0.7%
sg-sgp ∩ br-sao	0.7%
us-dfw ∩ us-sjc	0.2%
us-dfw ∩ sg-sgp	0.1%
us-sjc ∩ br-sao	0.1%

TABLE 4.2: The top five anycast honeypots intersected based on source address and packet arrival day tuples for NTP traffic.

Intersection	Overlap
us-dfw ∩ us-mia	44.8%
cl-scl ∩ br-sao	11.0%
us-dfw ∩ in-bom	1.7%
in-bom ∩ us-mia	1.4%
us-dfw ∩ br-sao	0.3%
in-bom ∩ br-sao	0.1%
us-mia ∩ br-sao	0.1%
us-dfw ∩ cl-scl	0.0%
cl-scl ∩ us-mia	0.0%
cl-scl ∩ in-bom	0.0%

TABLE 4.3: The top five anycast honeypots intersected based on source address and packet arrival day tuples for DNS traffic.

4.2.3 Anycast traffic

Traffic observed by our anycast honeypots is the most valuable, as this has to stem from adversaries topologically near the measurement node. This has the added effect that the amount of requests observed across anycast honeypots is not spread evenly. Starting with NTP, only a select few sites received the majority of anycast traffic. Within our two-month observation period, the Dallas site received 38.9% of all NTP traffic destined for our anycast address. The rest of the top five are comprised of Singapore (15.5%), Mumbai (12.6%), San Jose (10.6%) and São Paulo (6.7%). The same can be observed for DNS. Santiago receives 35.6% off all DNS anycast traffic. Sites that follow are Dallas (20.8%), São Paulo (18.8%), Miami (9.3%) and Mumbai (3.3%). A full breakdown can be seen in Figures 4.10 and 4.11. Instinctively, the amount of traffic an anycast site receives could be linked to the size of its anycast catchment. The more clients an anycast site typically serves, the higher its chances of receiving the most traffic. Although Mumbai, São Paulo and Dallas are seen in the top ten largest anycast catchments, the majority do not fit in this notion. As shown in Figures 4.11 and 4.12, Santiago is the largest receiver of DNS traffic, but has the 18th-largest catchment. Vice versa, Frankfurt has the largest anycast catchment, but ranks 11th and 10th for DNS and NTP, respectively.

4.2.4 Anycast spoofing detection

As we discussed in Section 2.3.2, de Vries et al. mentioned the potential for using anycast to detect spoofed traffic. Their strategy revolves around examining if an incoming packet's source address belongs in a site's anycast catchment. If it does not, they deem it likely that the packet is spoofed. To investigate their intuition, we again choose a day of NTP and DNS anycast traffic from the busiest site for each protocol as displayed in Figure 4.10 and 4.11. Each day has a recorded amount of distinct targets closely matching the average per day for the site in question. The /24 prefix of each target is extracted and compared to the /24 prefixes of the sites' anycast catchment replies.

On December 8, 2024, the Dallas anycast honeypot observed 4,192 distinct targets over NTP. Only two of those targets have a /24 prefix, also seen in the site's anycast catchment of the same day. For DNS, the Santiago honeypot saw 412,182 distinct targets on November 19, 2024. Not a single /24 prefix corresponding to these targets is seen in the site's anycast catchment. These observations coincide with those seen by de Vries, which strengthens the assumption that spoofed traffic is captured by our anycast honeypots.

4.2.5 Target overlap

To gain more insights on attack targets, we analyzed if there is any overlap between anycast sites. The same targets seen across multiple anycast sites in the same timeframe could indicate that attack campaigns are launched from multiple locations. Therefore, we compiled a list of all unique source addresses and corresponding packet arrival days for each of the top five NTP and DNS anycast honeypots, as displayed in Figures 4.10 and 4.11. All possible honeypot pairs for both these protocols were analyzed for intersections based on their source address and packet arrival day tuples. Looking at NTP, the highest overlap measured is 12.4% between Dallas and Singapore, while the remaining combinations never reach above 4%. For DNS, Dallas and Miami have 44.8% in common. Santiago and São Paulo overlap 11%, with the remaining below 2%. All figures are shown in Tables 4.2 and 4.3. It is clear that targets observed at one anycast honeypot over NTP are rarely seen at any other anycast honeypot. In other words, NTP packets with a destination address matching our anycast IP and a source address targeting a specific victim are often observed at only one location. DNS traffic does seem to overlap more across anycast sites.

Another valuable statistic is the amount of overlap of targets between an anycast honeypot and one or more unicast honeypots. If this is the case, it becomes clear which of our unicast honeypots are weaponized in an attack launched from a particular anycast catchment. By filtering all unicast honeypot traffic from various sites to only include targets seen at a particular anycast honeypot, it becomes clear there are strong similarities. An example for NTP is visualized in Figure 4.13. In the top left is the number of anycast NTP requests per second plotted for the Dallas honeypot. The remaining plots are for various unicast honeypots. All unicast traffic is filtered to only have source addresses observed at the Dallas anycast honeypot in the same timeframe. This pattern can be observed daily for all other anycast sites as well, showing that targets observed at one anycast honeypot can often be seen at various or all unicast honeypots in the same timeframe.

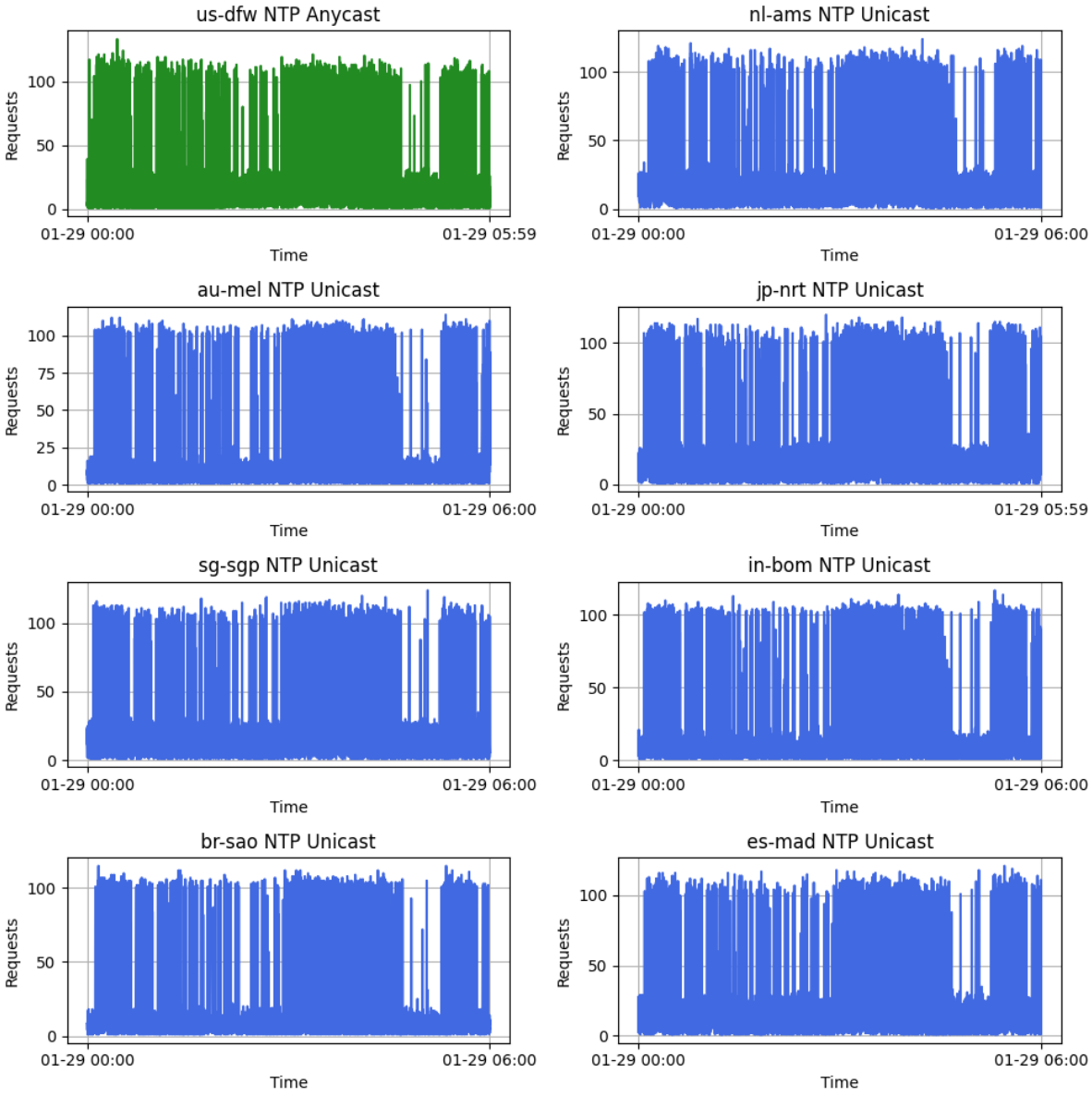

FIGURE 4.13: Graphs plotting NTP requests per second over the first six hours of January 21 for the us-dfw anycast honeypot and various unicast honeypots. All unicast targets are filtered to those also seen by the us-dfw anycast honeypot.

us

Chapter 5

Methodology

In the previous chapter, we made it clear that NTP was by far the most weaponized protocol during our observation period. To create a level playing field, we base the conclusions from our exploratory analysis on this protocol. In particular, the analysis brought three key insights about DDoS attacks in our anycast environment:

i *Only a select number of sites received the most **anycast** traffic.*

ii *Targets observed at one **anycast** honeypot are rarely seen at any other **anycast** honeypot in the same timeframe.*

iii *Targets observed at one **anycast** honeypot can often be seen at various or all **unicast** honeypots in the same timeframe.*

This profile leads to the assumption that attack campaigns weaponizing our worldwide honeypots are often launched from within a single anycast catchment. We also have a strong intuition that within an anycast catchment, these attacks are launched by a single spoofer. We come to this conclusion mainly because TTL values stemming from this traffic are stable, which coincides with the work done by Krämer et al. [35]. We leverage this set of characteristics in our methodology for narrowing down a spoofer's origin network.

Our methodology relies on filtering out prospect networks from one anycast catchment and various unicast catchments. These cathcments correspond to the sites a target has been observed. This is done by comparing their hop counts to those seen in spoofed traffic. These hop counts are calculated by assuming an initial TTL value and subtracting that from the observed value seen in spoofed traffic and ICMP echo replies in catchments. In this chapter, we discuss in detail how we implement this methodology to produce the smallest list of prospect networks possible where a spoofer could reside.

5.1 Preliminary work

To make sure comparing hop counts from both catchments and spoofed traffic is even viable, insight is needed into how stable both data sources are. When it comes to traffic, Krämer et al. make a valid remark that TTL values can also be dynamically spoofed [35]. It seems that for the traffic we observed, spoofers are not concerned about masking their TTL value. When we analyse every TTL value captured by both our anycast and unicast honeypots, more than 97% of all values fall in the 235 to 250 range. This makes an initial value of 255 highly likely, the maximum for the TTL field.

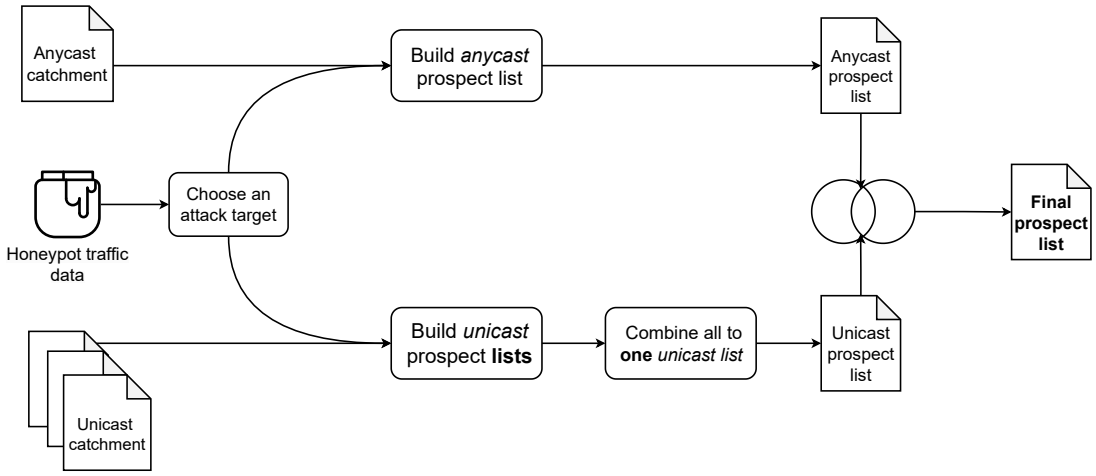

FIGURE 5.1: A top-down overview showing where traffic data and catchment data are utilized to build a prospect list for potential networks a spoofer might reside.

The stability of catchment measurements is also key for our method to work. Fluctuations in the number of ICMP echo replies, the anycast catchment they belong to or their TTL value have implications on how we should compare them to spoofed traffic. Recent work done by Hendriks & Degen et al. shows that IPv4 anycast catchments can be considered stable for 3 weeks, with an error rate of 10%. 95% of prefixes route to the same site over the course of one week [25].

TTLs were not considered in their work, which we analyzed over a week of daily measurements. Within the context of anycast, 90.7% of ICMP echo replies that were mapped more than once to the same site kept a perfectly stable TTL value. On average, TTL values for all unicast sites did not change for 91.1% of the hitlist.

An important detail is that anycast and unicast routes to the same measurement node are not always equal. This means that a spoofer's hop count to the same honeypot can vary depending on whether the connection is made via unicast or anycast. Therefore, a hop count from spoofer to anycast honeypot should never be compared to unicast catchment data and vice versa. Although we did not do a full analysis on why this is the case, we suspect that it may be related to the implementation of the Tangled testbed.

5.2 Approach

Our approach for implementing the methodology starts by first selecting an attack target. This is done by choosing a spoofed source address only observed at *one* anycast site and preferably as *many* unicast sites as possible within the same timeframe. The TTL values extracted from this traffic at each site are converted to hop counts. Initial TTL values for most operating systems are public knowledge, as seen in the Appendix A.1. This makes conversion trivial by the use of a simple algorithm shown in the Appendix at Listing 1.

The single anycast catchment and the various unicast catchments corresponding to the weaponized sites are filtered. This is done by matching the hop counts from their replies

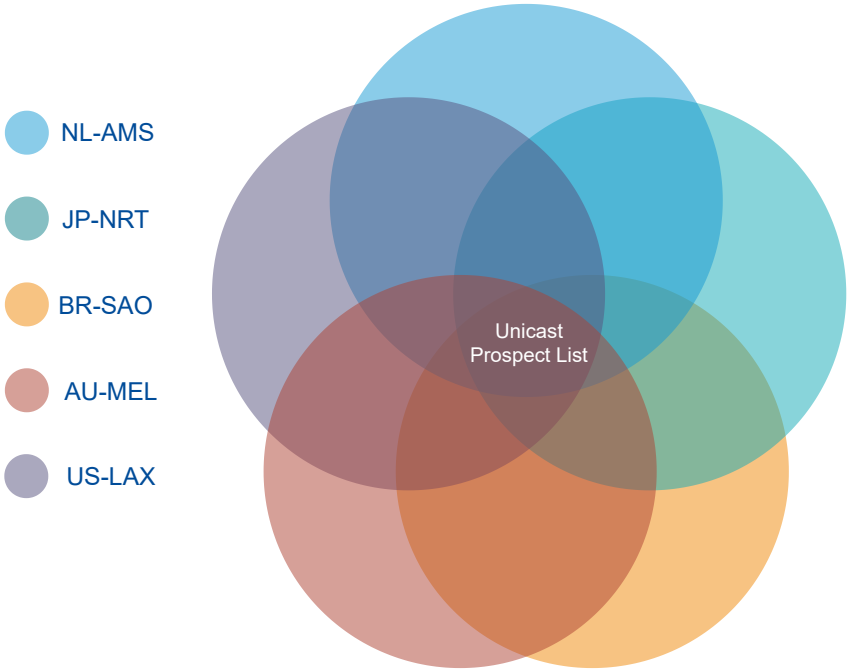

FIGURE 5.2: A Venn diagram showing the intersection of prospects from various unicast honeypots active in an attack into one final "unicast prospect list".

to those seen in the spoofed traffic. From this process, we gain two prospect lists of routed prefixes (RPFXes) for both anycast and unicast. These two lists are then intersected to produce the smallest set of prospect networks from which the spoofer could originate. A visual breakdown of this process is pictured in Figure 5.1. In the remainder of this section, we discuss in more detail how both prospect lists are created.

5.2.1 Anycast prospect list

As we assume the spoofer resides within the select anycast catchment, narrowing down the set of networks observed in this catchment represents the first step in building a prospect list. All catchment replies with a hop count matching the value seen from the spoofer are extracted. Due to routing irregularities, an exact match is too tight of a margin to encapsulate the spoofer's network. Therefore, we increase the reach by matching hop counts within a pre-determined **jitter** value. We discuss in Section 6.3.1 how we choose this value.

The source addresses from the filtered replies are cross-referenced with CAIDA's prefix to AS mapping dataset [6], creating a list of candidate routed prefixes and ASes where the spoofer could originate from. The algorithm used for building the anycast prospect list is listed in the Appendix at Listing 2.

5.2.2 Unicast prospect list

As described earlier, a chosen target is not only observed at one anycast site but also across many unicast sites. The goal for creating a unicast prospect list is to leverage all unicast sites active in an attack as vantage points. Intersecting this information with the anycast prospect list yields a more precise final set of networks.

Creating a unicast prospect list is similar to the method used for anycast, except for a few key differences. For *every* unicast site active in the attack, the hop count from the spoofer to the site is calculated in the same manner as mentioned in Section 5.2. Each relevant unicast catchment is filtered to build a prospect list based on hop count. Prospect networks are extracted from each unicast catchment by again matching these hop counts with those from the spoofer to the site. For the same reason as described in Section 5.2.1, hop counts are matched with a **jitter** to account for routing irregularities.

All filtered unicast catchments are again cross-referenced with CAIDA's pfx2as dataset and intersected to form a single "unicast prospect list". As illustrated in the Venn diagram in Figure 5.2, the more active unicast honeypots in an attack, the smaller the prospect list potentially becomes. Still, because of irregularities in routing, a complete intersection is too tight of a margin to encapsulate the spoofer's network. Therefore, every distinct routed prefix from all filtered unicast catchments is given a **majority vote** score. When an RPFX is seen in 25 of the 32 filtered catchments, it gets a score of 25. All RPFXes with a majority vote score equal to or larger than a pre-determined value are saved in the unicast prospect list. We discuss how this score is chosen in Section 6.3.2. The full algorithm is listed in the Appendix at Listing 3.

The final step of this methodology intersects both the unicast and anycast prospect lists based on their routed prefix to create the final result.

5.3 Limitations

It is clear that heavily relying on TTL values for narrowing down a set of networks has its drawbacks, mainly stemming from irregularities in internet routing. Some network operators implement load balancing, causing traffic to be distributed across different paths with possibly varying hops. Also, routing policies can be changed, or other traffic engineering invisible to us can cause TTL fluctuations.

Therefore, this methodology must be tuned to find the best balance in the final prospect list's accuracy and size. Increasing the jitter on matching hop counts for both unicast and anycast prospects might result in a high probability of encapsulating the spoofers' origin network. However, this has the added effect of increasing the prospect list's size, rendering it less useful. The same goes for the chosen majority vote score for the unicast prospects. It is for that reason that in the next chapter, we dive into what parameters should be expected for this methodology to be most effective.

Chapter 6

Validation

In the previous chapter, we laid out every step in our approach for implementing our methodology. Before we can narrow down a set of networks from which an attacker sends spoofed traffic to our honeypots, we need validation that our methodology is effective. Without ground truth, there is no way of determining under which parameters our prospect list includes the spoofer's origin network. Fortunately, we got the opportunity to collaborate with UC San Diego's Center for Applied Internet Data Analysis (CAIDA) to leverage their Spoofer project, enabling us to build ground truth.

6.1 CAIDA Spoofer project

Since 2015, CAIDA has run the "Spoofer" project, initially conceived by Robert Beverly in 2005. The main goal of the project is to map out with active measurements which prefixes and ASes can be spoofed to and from [7]. What makes the project unique is that it is entirely crowdsourced; anyone with a PC connected to the internet can participate. As explained in Section 2.3.1, not all AS operators implement BCP38, enabling spoofed traffic to leave their networks. By making all participating nodes routinely send non-malicious spoofed traffic from around the world, insight is gained into the scale and scope of ingress filtering by AS operators. To be precise, the project managed to collect data from more than 12.217 ASes in 220 countries. For our study, CAIDA agreed to have their clients send spoofed traffic to all 32 unicast sites and our dedicated anycast address. The spoofer project collects data from its spoofer clients in a four-step process, also displayed in Figure 6.1:

1. The project records both inbound and outbound spoofing for its crowdsourced clients. Inbound spoofing is analyzed by a CAIDA host sending a packet with a spoofed source IP that appears to come from within the client's network.

2. If the packet is not dropped along the route, the spoofer client detects this, confirming inbound spoofing is allowed.

3. For outbound spoofing, the client itself sends a packet with a spoofed source IP to two infrastructures.

4. If the packet is not dropped during its outbound route (meaning outbound spoofing is allowed), both CAIDA's Archipelago (ARK) measurement infrastructure and our measurement nodes receive it [5]. In our case, packets are intercepted by tcpdump and backed-up to S3 daily.

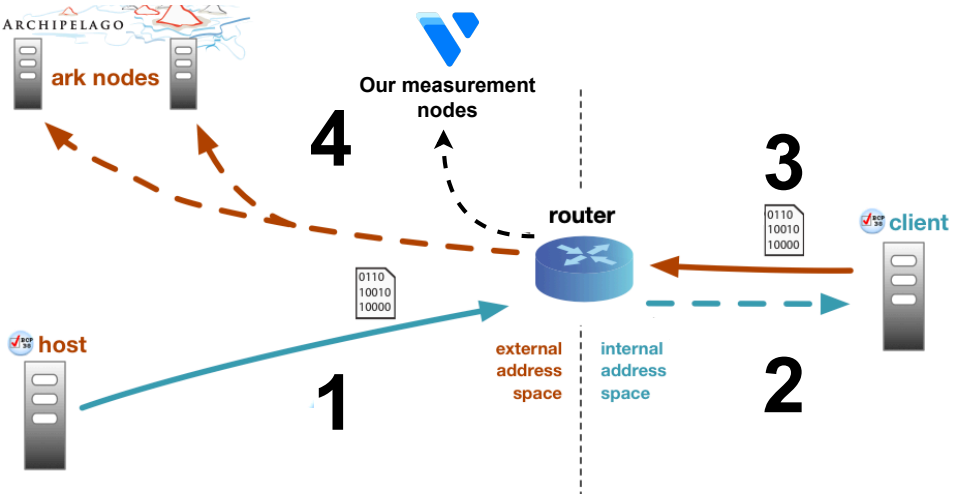

FIGURE 6.1: The four-step process on how the spoofer project collects its data and where our measurement setup is included.

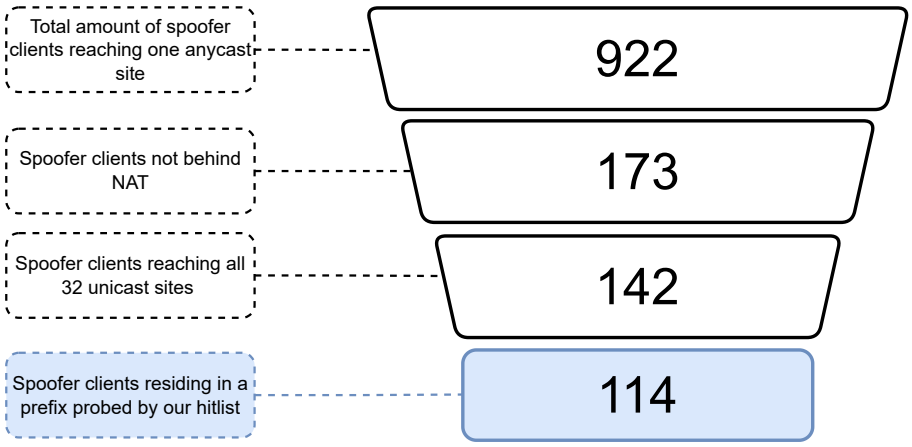

FIGURE 6.2: A data funnel showing how many usable data points were extracted from the traffic sent by CAIDA's Spoofer project.

6.2 Data points

The packet payload sent by CAIDA's Spoofer clients allows us to check if the packet is indeed spoofed and what the client's actual source address is. First off, the packet's source address is spoofed when it leaves the client's device. This address is also included in the packet's payload to check if the source address is rewritten by NAT when it reaches its destination. However, most important is the sequence ID embedded in the payload, which enables cross-referencing with CAIDA's database to determine the actual source address of the client.

From January 21 to February 8, 2025, 922 Spoofer clients reached our measurement setup at an anycast site. Because of NAT, 173 were able to send actual spoofed packets. From this set, 142 clients were also able to send a spoofed packet to all 32 unicast sites in the same timeframe. Lastly, 114 of those clients reside in a prefix we are able to probe with

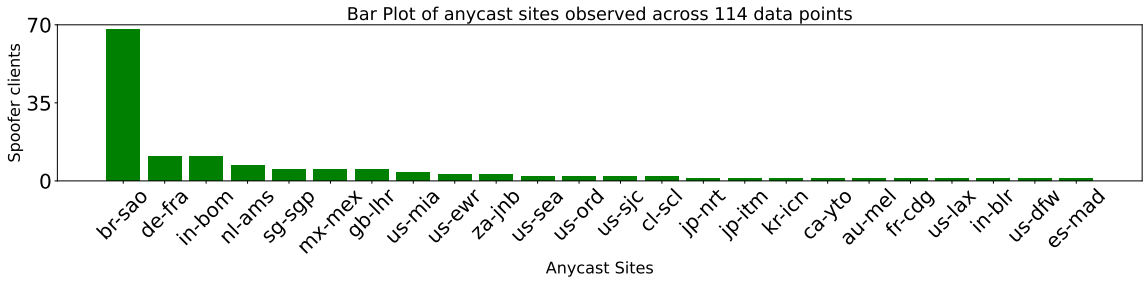

FIGURE 6.3: Bar plot showing which anycast site each of the 114 data points reached.

Subject ID	Anycast Sites	UTC Date
A	us-sea	2025-02-04
B	in-bom	2025-01-31
C	br-sao	2025-01-27

TABLE 6.1: Three sets of spoofed packets used as validation subjects, each based on one target observed at one anycast site and all unicast sites.

our catchment measurements. A breakdown is displayed in Figure 6.2. Figure 6.3 shows which anycast each of the 114 data points reached.

6.3 Experiment

The goal for the data collected from the Spoofer project is to determine under which parameters our methodology is most effective. These parameters include which jitter values for the matching of hop counts and which majority vote score results in the smallest final prospect list, granted that the spoofer's origin RPFX is still included.

For this validation experiment, three sets of spoofed traffic from three different clients are chosen out of the pool of 114 data points discussed in the previous section. These are chosen to create a diverse set of anycast sites: Seattle, Mumbai and São Paulo, as Table 6.1 shows. Lastly, every client's original IP address falls in a /24 RPFX to create a level playing field. There is also a clear strategy for choosing a fitting catchment used for comparison. The anycast catchment measurement for each subject has been carried out on the same day the target has been observed. This seems well within the accuracy margins analyzed in Section 5.1. Because unicast catchments have a much broader reach for each site, it is likely that not the full IPv4 hitlist sends a response for each site during one measurement. Therefore, all 32 daily unicast catchments from November 24 to the first of December 2024 have been aggregated to create the most complete dataset.

In the remainder of this section we discuss what results we see for both the anycast and unicast prospect list on various jitter values and majority vote scores.

6.3.1 Anycast prospects

For each of our subjects, we build an anycast prospect list as described in Section 5.2.1. Starting with zero, we increment the jitter value by one until each prospect list includes the desired RPFX of the spoofer. With this analysis carried out, it seems subjects A and B need a minimum **jitter of two** for the right RPFX to be included in the filtered anycast

Anycast Site	Catch Reply's	RPFXes	ASes
us-sea	58,626	11,726	1,252
in-bom	300,423	81,235	12,018
br-sao	168,872	41,068	8,236

TABLE 6.2: Table showing each site's catchment size (reply count) and its observed amount of routed prefixes and ASes.

Anycast Site	Hops	Catch Reply's	RPFXes	ASes
us-sea	6	31,780	6,380	914
in-bom	10	127,969	51,196	8,623
br-sao	6	9,487	6,626	2,861

TABLE 6.3: Amount of anycast catchment replies with a matching hop count to that from the spoofer, with the corresponding amount of routed prefixes and ASes.

catchment. Subject C was able to encapsulate the desired RPFX on an exact match. This results in eliminating 45.8% of the Seattle catchment, 57.4% of Mumbai and 94.4% for São Paulo. The full breakdown can be seen in Tables 6.2 and 6.3.

6.3.2 Unicast prospects

As discussed in Section 5.2.2, we not only have to balance the jitter value for the filtering unicast catchments. There is also a need to choose a fitting majority vote score when combining all filtered catchments to one unicast prospect list. To see the effect of both parameters, we analyzed what majority vote score is needed to encapsulate the spoofer's RPFX for jitter values one to three across all 114 data points.

Let us denote $\mathcal{UP}$ as our unicast prospect list and $\mathcal{S}$ as the RPFX of the spoofer. Let us also define a majority vote score $\mathcal{M}$ and a jitter value $\mathcal{J}$ where $M \in \{1, 2, 3, \ldots, 32\}$ and $J \in \{1, 2, 3\}$. We want to reduce $|\mathcal{UP}|$ as much as possible, while $\mathcal{S} \in \mathcal{UP}$. This means we also want the $P(\mathcal{S} \in \mathcal{UP} \mid \mathcal{M} = m, \mathcal{J} = j)$ to be as high as possible. Looking at the data displayed in Figure 6.4, we have chosen a sweet spot of $m = 22$ and $j = 3$ as an example. This value for $\mathcal{M}$ demands a consensus of more than two-thirds of all filtered unicast catchments, minimizing $|\mathcal{UP}|$ to the greatest extent possible. Still, the $P(\mathcal{S} \in \mathcal{UP} \mid \mathcal{M} = m, \mathcal{J} = j) = 0.69$, giving a reasonable probability the spoofers RPFX is in the unicast prospect list, based on the 114 data points we have.

Applying the chosen parameters to our subjects leaves A with a unicast prospect list of 133,996 RPFXes in 21,877 ASes. For B, the results are 135,954 RPFXes and 22,786 ASes. C ends up with 118,806 RPFXes in 24,546 ASes.

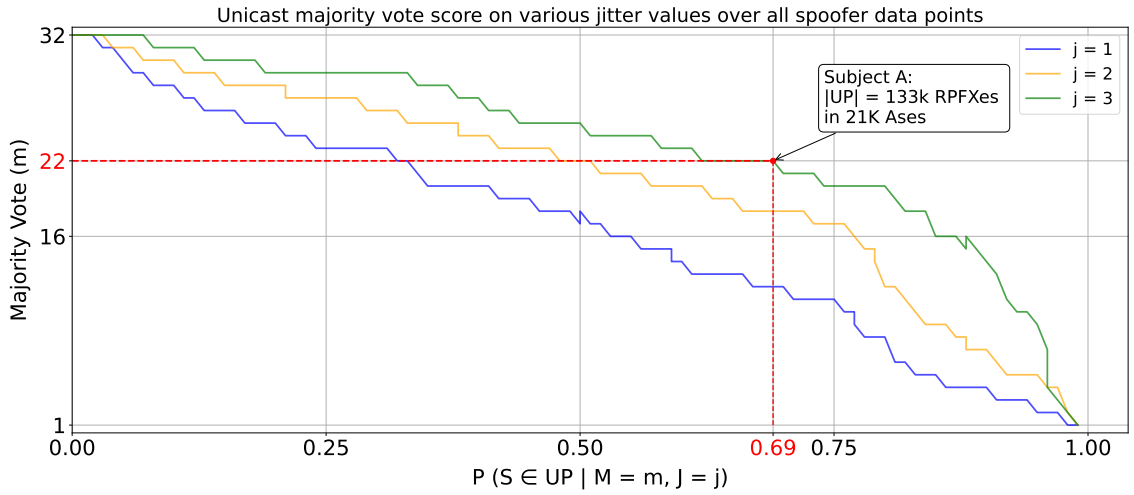

FIGURE 6.4: A graph showing for all 114 Spoofer data points the maximum required majority vote score to encapsulate the desired RPFX, for various jitter values.

6.3.3 Intersection

As a last step, both the unicast and anycast prospect lists are intersected to create the final result. For subject A, this intersection yields a final prospect list of 2,763 RPFXes in 549 ASes. Subject B comes out at 11,838 RPFXes in 2,724 ASes and subject C results in 3,928 RPFXes in 1,825 ASes. This means that by intersecting both unicast and anycast prospect lists, we were able to reduce the prospect RPFXes for subjects A, B and C seen in 6.3.1 by 56.7%, 92.3% and 40.7%, respectively.

6.4 Conclusions

Based on the subjects we have tested in this validation experiment, we conclude that filtering out an anycast catchment within a jitter value of two may be a safe margin for encapsulating the spoofers' RPFX. It is possible that this jitter value could be further optimized by analyzing more cases outside of the three subjects presented, but we leave this for future work. For creating a unicast prospect list, we have discussed how data from the Spoofer project can be utilized in choosing fitting parameters concerning the jitter value and majority vote score. Meaning, for the prospect list to be as small as possible while still including the desired RPFX. Building a unicast prospect list and intersecting it with the anycast prospect list proves to be a valuable strategy, as it greatly reduces the amount of prospects in the final result. This is especially true for large anycast catchments, as seen with subject B.

Chapter 7

Case Studies

In our validation experiment from the previous chapter, we discussed under which parameters our proposed methodology is most effective in narrowing down a spoofers' network. Still, the ultimate goal for our methodology is to pinpoint as precisely as possible from which networks DDoS attacks are launched. In this chapter, we apply insights from the validation experiment on malicious spoofed traffic recorded by our honeypots to show how effective our methodology can be in a real-world context.

The structure of this chapter is very much comparable to the previous one. Three sets of spoofed traffic are chosen, this time stemming from our honeypots. As described in the introduction of Chapter 5, NTP traffic was by far the most prevalent across our honeypots and fits the best in our desired profile compared to other protocols. To create a level playing field, we only consider traffic over this protocol for this chapter. Results for the anycast and unicast prospect lists, as well as the final intersection, are discussed for each of the three examples.

7.1 Studies

The three chosen sets of spoofed traffic all have a specific target observed on one single anycast honeypot and as many unicast honeypots as possible in the same timeframe. As Figure 4.10 shows, the top three most active anycast honeypots based on NTP traffic were Dallas, Singapore, and Mumbai. We decided to choose a target seen at each of these three sites, as we know there is little overlap, as demonstrated by Table 4.2. Details of each set are shown in Table 7.1. For the remainder of this chapter, these sets of spoofed traffic will be referred to by their anycast location or subject ID. Catchments have been chosen and aggregated in the same manner as in Section 6.3. The anycast catchment chosen for a given subject is recorded on the same day the spoofed traffic is observed. All daily unicast catchments from November 24 to the first of December 2024 are aggregated to ensure the largest coverage of the IPv4 hitlist possible.

Subject ID	Anycast Site	Unicast Sites	UTC Date	Start Time	End Time
A	us-dfw	32	2024-11-30	06:30	07:20
B	in-bom	29	2024-12-03	14:50	15:00
C	sg-sgp	27	2024-12-03	09:00	11:00

TABLE 7.1: Three sets of spoofed packets, each based on one target observed at one anycast site, various unicast sites, and within a defined timeframe.

Anycast Site	Catch Size	RPFXes	ASes
us-dfw	140,457	25,136	2,616
in-bom	320,512	84,210	12,187
sg-sgp	82,890	32,535	3,914

TABLE 7.2: Table showing each site's catchment size (reply count) and its observed amount of routed prefixes and ASes.

Anycast Site	Hops	Prospect Size	RPFXes	ASes
us-dfw	7	50,501	16,486	1,888
in-bom	11	159,670	44,771	7,688
sg-sgp	11	47,888	18,595	2,040

TABLE 7.3: Amount of catchment replies with a matching hop count to that from the spoofer to the honeypot, with corresponding amount of routed prefixes and ASes.

7.1.1 Anycast prospects

As our validation experiment in Section 6.3.1 showed, a jitter value of two was needed in two out of three subjects to encapsulate the Spoofer client's RPFX within a filtered anycast catchment. Tables 7.2 & 7.3 show that applying such a jitter on a hop count of seven eliminated 64% of Dallas' catchment replies from prospect consideration. A spoofer to honeypot hop count of 11 eliminated 50.2% and 42.2% for Mumbai and Singapore, respectively.

7.1.2 Unicast prospects

As we discussed in Section 6.3.2, a balance must be struck between making the unicast prospect list as small as possible. Granted, while also maintaining an acceptable probability that the spoofer's RPFX is encapsulated. This is done by choosing a fitting jitter value and majority vote score. For these subjects, we choose a sweet spot of 22 for the majority vote and a jitter value of three. This means unicast catchments are filtered based on a hop count match within a margin of three. If a RPFX is seen in at least 22 filtered unicast catchments, it is added to the final unicast prospect list. As presented in Section 6.3.2, these parameters give a probability of 0.69 that the spoofer's RPFX is encapsulated in the unicast prospect list based on our ground truth.

This results in a unicast prospect list of 95,141 RPFXes in 14,805 ASes for subject A. Subject B and C end up with 122,858 RPFXes in 21,148 ASes and 84,816 RPFXes in 13,040 ASes, respectively.

7.1.3 Intersection

Both the unicast and anycast prospect lists are again intersected based on routed prefix to create the most narrowed-down set of networks. For subject A, this intersection resulted in a final prospect list of 8,557 RPFXes in 1,108 ASes. Subject B comes out at 8,822 RPFXes in 2,191 ASes and subject C results in 8,211 RPFXes in 1,030 ASes. By intersecting the unicast and anycast prospect lists, we reduced the prospect amount of RPFXes for subjects A, B, and C, seen in Section 7.1.1, by 48.1%, 80.3%, and 82.9%, respectively.

7.2 Conclusions

When analyzing the size of the anycast prospect lists, it should be noted that the numbers presented in this chapter should not be directly compared to each other or other anycast sites in general, as they are strongly dependent on catchment size and topology. Still, it is reassuring to see that they fall in the same range compared to subjects with the same jitter value in Section 6.3.1. The same can be said for the unicast prospect lists in Section 6.3.2, as those are created with the same jitter value and majority vote score.

As for the final prospect list resulting from the intersection, we managed to narrow the set of networks to as low as 8,211 RPFXes in 1,030 ASes in the case of subject C, eliminating approximately 99.2% of known RPFXes and 98.7% of known ASes. Given that there was no way to determine the origin of this traffic before this methodology, this is a significant step in the right direction for narrowing down where R&A attacks are launched from.

Chapter 8

Limitations

Although our proposed methodology sets a step in the right direction for narrowing down in which networks a spoofer potentially resides, various important limitations and issues need to be mentioned. These revolve around our ground truth and the jitter value for hop count matching.

8.1 Ground truth

As described in Chapter 6, building a ground truth is essential in deeming the designed methodology effective. Without a set of verified spoofed packets from which the actual source address can be extracted, it is impossible to know if the prospect list includes the right network. Not to mention, these packets ideally should be sent from various locations to all measurement nodes in our setup within the same timeframe. Finding clients able to do this voluntarily and ethically is not trivial. Although CAIDA's Spoofer project seems to be a great fit, due to time constraints, only a small set of 114 usable data points were collected before this analysis had to conclude. As the ideal jitter values and majority vote score were determined based on such a small set, it could be that these parameters are still on the conservative side, increasing the final prospect list size in Chapter 7.

Another problem concerning the ground truth is the bias towards the São Paulo anycast catchment. 74% of the 114 data points were observed on our br-sao anycast site. This bias seems to correspond with all traffic received from the Spoofer project over the short two-week observation period. As we have no say on where Spoofer participants are located, it is difficult to diversify this data. However, it could be that by collecting more traffic from the Spoofer project over a longer period, this bias is balanced out.

8.2 Jitter

In Chapter 6, we demonstrate that the main parameter influencing the size of prospect lists is the required jitter value for the matching of hop counts. We are aware that TTL values are a coarse data point, which we discussed in Section 5.3. Therefore, the fact that some jitter is needed to encapsulate the desired RPFX was expected. Especially for unicast, as the reach of a site's unicast endpoint is much broader than anycast, making incoming traffic more susceptible to TTL fluctuations. However, a jitter value of three, seen in the example of Section 6.3.2, is higher than we initially thought was needed. The reason for this could be due to the topology of our measurement nodes or some traffic engineering done by Vultr, which hosts our VPSes.

Traffic observed on an anycast site should be close by in terms of hops compared to the other sites in question. It is not far-fetched to assume that the hop count from the spoofer should be able to match directly or, at most, with a jitter value of one to hop counts in the anycast catchment. Still, a jitter of two was needed for two out of the three validation subjects, while the remaining subject was able to match the spoofers' hop count on an exact match in Section 6.3.1. This means that possibly every anycast catchment has a varying optimal jitter value.

The reason for this could again lie in the topology of our nodes or the implementation of the Tangled testbed with Vultr. More insight can be potentially gained by analyzing how other anycast and unicast catchments fair against hop count comparison, based on ground truth outside of the subjects we could cover. This is something we leave for future work.

Chapter 9

Conclusions and Future Work

The goal of this work was to determine spoofed traffic and identify a candidate set of offending networks by filtering catchment data. This has been realized by collecting spoofed DDoS traffic across 32 worldwide honeypots, each reachable by a shared anycast address and their unicast address. Based on the insight this data gave us, we proposed, validated, and tested a methodology able to produce a prospect list of offending networks based on hop count. With the knowledge gained from this process, we can answer all research questions formulated in Section 1.1.

What characteristics are evident in honeypot traffic originating inside and outside our anycast catchments?
In Section 4.2, we discussed that our honeypots saw the most weaponization on their NTP service over the 82-day observation period. What stands out is that this traffic was only received by a select few honeypots over anycast. Attack targets across these active anycast honeypots are rarely shared with each other on the same day. However, these targets are often observed at various or all unicast honeypots within the same timeframe. Also, the TTL values associated with one target are stable across both types of honeypots, which coincides with the work done by Krämer et al. [35]. This profile leads to the assumption that attack campaigns weaponizing our worldwide honeypots are often launched by a single spoofer from within a single anycast catchment.

How can these characteristics be used to narrow down from which networks spoofed traffic originates?
The profile sketched by our honeypot analysis forms the basis for our methodology, which we laid out in Chapter 5. We are able to build a prospect list of networks where a spoofer potentially originates from by leveraging the single anycast catchment and various unicast catchments an attack target is observed in. The anycast catchment is filtered based on matching the assumed hop count from the spoofer to the hop counts seen in the catchment replies. The hop count must match within a pre-defined jitter value. Hop counts are calculated based on subtracting the observed TTL value from an assumed initial TTL value. The matching replies are cross-referenced with CAIDA's pfx2as dataset to see which RPFXes make up the anycast prospect list. The same process is done for all relevant unicast catchments. All of them are filtered based on matching the spoofers' observed hop count within a jitter value. All prospects are then unified based on a majority vote to build our unicast prospect list. Lastly, anycast and unicast prospects are intersected to create the most narrowed-down set of networks.

Within which margins can we be sure the spoofers' origin network is contained in our narrowed-down set?
By leveraging CAIDA's Spoofer project, we can build ground truth from spoofed packets sent by participating clients to our measurement nodes. These packets allow us to verify if they are indeed spoofed and what the sender's actual source address is. Based on the chosen validation subjects, a spoofers' RPFX is contained in our anycast prospect list within a jitter value of two. For unicast, a jitter value of three and a majority vote of 22 is an example of a sweet spot in balancing the size and accuracy of the unicast prospect list. With these parameters, we have shown a probability of 0.69 for the spoofers' RPFX to be included, as discussed in Section 6.3.2. Based on this insight, we managed to narrow down a malicious spoofer's origin network to as low as 8,211 prospect RPFXes in 1,030 ASes, eliminating approximately 99.2% of known RPFXes and 98.7% of known ASes, as shown in Section 7.1.3.

9.1 Future work

The knowledge gained from this work can be used as a stepping stone for many different efforts in understanding spoofed traffic. In this section we discuss the most important examples of opportunities to expand this research.

9.1.1 Honeypot traffic

We have collected a wealth of spoofed traffic from our unicast and anycast honeypots much larger than we could cover in this work. There is a lot more knowledge to be gained if this data is more closely analyzed. In our analysis, we defined an attack as one target seen in the same timeframe across various honeypots. However, there could be more granularity hidden in the data. Examples are packets that have similarities in the chosen payload or other unexplored data fields. Also, we saw that multiple targets are often attacked within the same prefix. Aggregating packets within a target prefix instead of choosing only one target address could yield less coarse hop counts, resulting in a more precise prospect list.

Furthermore, when analysis is scaled up over the whole measurement period, entire attack strategies can be uncovered based on which anycast and unicast sites participate. We have already shown that NTP is weaponized very differently compared to DNS when only looking at distinct targets across honeypots. What would be particularly interesting to examine is the cause of target overlap between anycast honeypots seen only for DNS and not NTP traffic.

9.1.2 CAIDA Spoofer validation

Another valuable asset stemming from this work is the ability to receive verifiable spoofed packets at our anycast and unicast sites, from which we can extract the sender's actual source address. Collecting more traffic from CAIDA's Spoofer project will undoubtedly help in further dialing in the parameters used to build our prospect network list.

9.1.3 BGP manipulation

During our work, we did not take advantage of BGP manipulation offered by the Tangled anycast testbed. One method that could potentially help in narrowing down the spoofers' origin network is BGP AS path prepending. If a set of ASes appears likely to host the

spoofer in question based on the build prospect list, each of their paths to the anycast site can be made incrementally "less favorable". This is done by prepending hops to them, making the path seem longer. If the spoofed traffic diverges to another anycast site after a certain prepend, it might mean the spoofer resides in the AS corresponding to that particular path.

Another interesting use of Tangled's flexibility is selectively withdrawing sites from the anycast network. By withdrawing the anycast site on which a certain target is observed, the spoofed traffic has to diverge to another site in a freshly divided anycast environment. By building an anycast prospect list for this site and repeating the process, we build a set of prospect lists from the perspective of various anycast sites. It would be interesting to analyse the results yielded by the intersection of these lists.

Bibliography

[1] IPv4 Hitlists. URL: https://ant.isi.edu/datasets/ip_hitlists/.

[2] RIPE Atlas. URL: https://www.ripe.net/analyse/internet-measurements/ripe-atlas/.

[3] State of IP Spoofing. URL: https://spoofer.caida.org/summary.php.

[4] Internet Protocol. Request for Comments RFC 791, Internet Engineering Task Force, September 1981. Num Pages: 51. URL: https://datatracker.ietf.org/doc/rfc791, doi:10.17487/RFC0791.

[5] Archipelago (Ark) Measurement Infrastructure, December 2006. Section: projects. URL: https://www.caida.org/projects/ark/.

[6] Routeviews Prefix to AS mappings Dataset (pfx2as) for IPv4 and IPv6, July 2008. URL: https://www.caida.org/catalog/datasets/routeviews-prefix2as/.

[7] Spoofer, October 2020. Section: projects. URL: https://www.caida.org/projects/spoofer/.

[8] Michael Backes, Thorsten Holz, Christian Rossow, Teemu Rytilahti, Milivoj Simeonovski, and Ben Stock. On the Feasibility of TTL-Based Filtering for DRDoS Mitigation. In Fabian Monrose, Marc Dacier, Gregory Blanc, and Joaquin Garcia-Alfaro, editors, *Research in Attacks, Intrusions, and Defenses*, pages 303–322, Cham, 2016. Springer International Publishing. doi:10.1007/978-3-319-45719-2_14.

[9] Vaibhav Bajpai, Steffie Jacob Eravuchira, and Jürgen Schönwälder. Lessons Learned From Using the RIPE Atlas Platform for Measurement Research. *ACM SIGCOMM Computer Communication Review*, 45(3):35–42, July 2015. doi:10.1145/2805789.2805796.

[10] Ray Bellis. Researching F-root Anycast Placement Using RIPE Atlas, October 2015. URL: https://labs.ripe.net/author/ray_bellis/researching-f-root-anycast-placement-using-ripe-atlas/.

[11] Leandro M. Bertholdo, João M. Ceron, Wouter B. de Vries, Ricardo de Oliveira Schmidt, Lisandro Zambenedetti Granville, Roland van Rijswijk-Deij, and Aiko Pras. TANGLED: A Cooperative Anycast Testbed. In *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pages 766–771, May 2021. ISSN: 1573-0077. URL: https://ieeexplore.ieee.org/document/9463960.

[12] Kevin Bock, Abdulrahman Alaraj, Yair Fax, Kyle Hurley, Eric Wustrow, and Dave Levin. Weaponizing Middleboxes for {TCP} Reflected Amplification. pages

3345–3361, 2021. URL: https://www.usenix.org/conference/usenixsecurity21/presentation/bock.

[13] Edgar Bohte, Manolis Stamatogiannakis, and Herbert Bos. *Evaluation of current state of amplification-based DDoS attacks*. PhD thesis, Vrije Universiteit Amsterdam, The Netherlands, November 2018. URL: https://research.vu.nl/en/publications/evaluation-of-current-state-of-amplification-based-ddos-attacks.

[14] Danilo Cicalese, Jordan Augé, Diana Joumblatt, Timur Friedman, and Dario Rossi. Characterizing IPv4 anycast adoption and deployment. In *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*, CoNEXT '15, pages 1–13, New York, NY, USA, December 2015. Association for Computing Machinery. URL: https://dl.acm.org/doi/10.1145/2716281.2836101, doi:10.1145/2716281.2836101.

[15] Danilo Cicalese, Diana Joumblatt, Dario Rossi, Marc-Olivier Buob, Jordan Augé, and Timur Friedman. A fistful of pings: Accurate and lightweight anycast enumeration and geolocation. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pages 2776–2784. IEEE, 2015. URL: https://ieeexplore.ieee.org/abstract/document/7218670/.

[16] Jakub Czyz, Michael Kallitsis, Manaf Gharaibeh, Christos Papadopoulos, Michael Bailey, and Manish Karir. Taming the 800 Pound Gorilla: The Rise and Decline of NTP DDoS Attacks. In *Proceedings of the 2014 Conference on Internet Measurement Conference*, IMC '14, pages 435–448, New York, NY, USA, November 2014. Association for Computing Machinery. doi:10.1145/2663716.2663717.

[17] Wouter B. de Vries, Salmān Aljammāz, and Roland van Rijswijk-Deij. Global-Scale Anycast Network Management with Verfploeter. In *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, pages 1–9, April 2020. ISSN: 2374-9709. URL: https://ieeexplore.ieee.org/document/9110449, doi:10.1109/NOMS47738.2020.9110449.

[18] Wouter B. de Vries, Ricardo de O. Schmidt, Wes Hardaker, John Heidemann, Pieter-Tjerk de Boer, and Aiko Pras. Broad and load-aware anycast mapping with verfploeter. In *Proceedings of the 2017 Internet Measurement Conference*, IMC '17, pages 477–488, New York, NY, USA, November 2017. Association for Computing Machinery. URL: https://dl.acm.org/doi/10.1145/3131365.3131371, doi:10.1145/3131365.3131371.

[19] Xun Fan, John Heidemann, and Ramesh Govindan. Evaluating anycast in the domain name system. In *2013 Proceedings IEEE INFOCOM*, pages 1681–1689, April 2013. ISSN: 0743-166X. URL: https://ieeexplore.ieee.org/document/6566965, doi:10.1109/INFCOM.2013.6566965.

[20] Ali Farha. IP Spoofing. *The Internet Protocol Journal*, 10:1–9, 2007.

[21] Danilo Giordano, Danilo Cicalese, A. Finamore, M. Mellia, M. Munafo, Diana Joumblatt, and D. Rossi. A First Characterization of Anycast Traffic from Passive Traces . page 30, 2016. URL: https://imt.hal.science/hal-01383092.

[22] Shane Greenstein. The Aftermath of the Dyn DDOS Attack. *IEEE Micro*, 39(4):66–68, July 2019. Conference Name: IEEE Micro. URL: https://ieeexplore.ieee.org/abstract/document/8771413, doi:10.1109/MM.2019.2919886.

[23] Harm Griffioen, Kris Oosthoek, Paul van der Knaap, and Christian Doerr. Scan, Test, Execute: Adversarial Tactics in Amplification DDoS Attacks. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, pages 940–954, New York, NY, USA, November 2021. Association for Computing Machinery. URL: https://dl.acm.org/doi/10.1145/3460120.3484747, doi:10.1145/3460120.3484747.

[24] Remi Hendriks. Improving anycast census at scale, July 2023. Publisher: University of Twente. URL: https://essay.utwente.nl/95878/.

[25] Remi Hendriks, Bernhard Degen, Bas Palinckx, Raffaele Sommese, and Roland van Rijswijk-Deij. An Empirical Evaluation of Longitudinal Anycast Catchment Stability. In *Accepted by the Passive and Active Measurement (PAM) Conference*, University of Twente, Enschede, The Netherlands, 2024.

[26] Tijs Hofmans. Technische Universiteit Eindhoven schort onderwijs op na cyberaanval. URL: https://tweakers.net/nieuws/230730/technische-universiteit-eindhoven-schort-onderwijs-op-na-cyberaanval.html.

[27] Hayte Hugo. SURF en onderwijsinstellingen melden storing door ddos-aanval. URL: https://tweakers.net/nieuws/230832/surf-en-onderwijsinstellingen-melden-storing-door-ddos-aanval.html.

[28] Dina Katabi and John Wroclawski. A framework for scalable global IP-anycast (GIA). In *Proceedings of the conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, SIGCOMM '00, pages 3–15, New York, NY, USA, August 2000. Association for Computing Machinery. URL: https://dl.acm.org/doi/10.1145/347059.347388, doi:10.1145/347059.347388.

[29] Bashar Ahmed Khalaf, Salama A. Mostafa, Aida Mustapha, Mazin Abed Mohammed, and Wafaa Mustafa Abduallah. Comprehensive Review of Artificial Intelligence and Statistical Approaches in Distributed Denial of Service Attack and Defense Methods. *IEEE Access*, 7:51691–51713, 2019. Conference Name: IEEE Access. URL: https://ieeexplore.ieee.org/abstract/document/8692706, doi:10.1109/ACCESS.2019.2908998.

[30] Mizuki Kondo, Rui Tanabe, Natsuo Shintani, Daisuke Makita, Katsunari Yoshioka, and Tsutomu Matsumoto. Amplification Chamber: Dissecting the Attack Infrastructure of Memcached DRDoS Attacks. In Lorenzo Cavallaro, Daniel Gruss, Giancarlo Pellegrino, and Giorgio Giacinto, editors, *Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 178–196, Cham, 2022. Springer International Publishing. doi:10.1007/978-3-031-09484-2_10.

[31] Daniel Kopp, Christoph Dietzel, and Oliver Hohlfeld. DDoS Never Dies? An IXP Perspective on DDoS Amplification Attacks, March 2021. arXiv:2103.04443 [cs]. URL: http://arxiv.org/abs/2103.04443, doi:10.48550/arXiv.2103.04443.

[32] Sam Kottler. February 28th DDoS Incident Report, March 2018. URL: https://github.blog/2018-03-01-ddos-incident-report/.

[33] Johannes Krupp, Michael Backes, and Christian Rossow. Identifying the Scan and Attack Infrastructures Behind Amplification DDoS Attacks. In *Proceedings of*

the 2016 ACM SIGSAC Conference on Computer and Communications Security, CCS '16, pages 1426–1437, New York, NY, USA, October 2016. Association for Computing Machinery. URL: https://dl.acm.org/doi/10.1145/2976749.2978293, doi:10.1145/2976749.2978293.

[34] Johannes Krupp, Mohammad Karami, Christian Rossow, Damon McCoy, and Michael Backes. Linking Amplification DDoS Attacks to Booter Services. In Marc Dacier, Michael Bailey, Michalis Polychronakis, and Manos Antonakakis, editors, *Research in Attacks, Intrusions, and Defenses*, pages 427–449, Cham, 2017. Springer International Publishing. doi:10.1007/978-3-319-66332-6_19.

[35] Lukas Krämer, Johannes Krupp, Daisuke Makita, Tomomi Nishizoe, Takashi Koide, Katsunari Yoshioka, and Christian Rossow. AmpPot: Monitoring and Defending Against Amplification DDoS Attacks. In Herbert Bos, Fabian Monrose, and Gregory Blanc, editors, *Research in Attacks, Intrusions, and Defenses*, pages 615–636, Cham, 2015. Springer International Publishing. doi:10.1007/978-3-319-26362-5_28.

[36] Marc Kührer, Thomas Hupperich, Christian Rossow, and Thorsten Holz. Exit from Hell? Reducing the Impact of {Amplification} {DDoS} Attacks. pages 111–125, 2014. URL: https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/kuhrer.

[37] Marc Kührer, Thomas Hupperich, Christian Rossow, and Thorsten Holz. Hell of a Handshake: Abusing {TCP} for Reflective Amplification {DDoS} Attacks. 2014. URL: https://www.usenix.org/conference/woot14/workshop-program/presentation/kuhrer.

[38] Jinjin Liang, Jian Jiang, Haixin Duan, Kang Li, and Jianping Wu. Measuring Query Latency of Top Level DNS Servers. In Matthew Roughan and Rocky Chang, editors, *Passive and Active Measurement*, pages 145–154, Berlin, Heidelberg, 2013. Springer. doi:10.1007/978-3-642-36516-4_15.

[39] Douglas C. MacFarland, Craig A. Shue, and Andrew J. Kalafut. The best bang for the byte: Characterizing the potential of DNS amplification attacks. *Computer Networks*, 116:12–21, April 2017. URL: https://www.sciencedirect.com/science/article/pii/S1389128617300452, doi:10.1016/j.comnet.2017.02.007.

[40] Trevor Mendez, Walter Milliken, and Craig Partridge. Host Anycasting Service. Request for Comments RFC 1546, Internet Engineering Task Force, November 1993. Num Pages: 9. URL: https://datatracker.ietf.org/doc/rfc1546, doi:10.17487/RFC1546.

[41] Jelena Mirkovic and Peter Reiher. A taxonomy of DDoS attack and DDoS defense mechanisms. *ACM SIGCOMM Computer Communication Review*, 34(2):39–53, April 2004. URL: https://dl.acm.org/doi/10.1145/997150.997156, doi:10.1145/997150.997156.

[42] Giovane C. M. Moura, Ricardo de O. Schmidt, John Heidemann, Wouter B. de Vries, Moritz Müller, Lan Wei, and Cristian Hesselman. Anycast vs. DDoS: Evaluating the November 2015 Root DNS Event. May 2016. Publisher: University of Southern California. URL: https://research.utwente.nl/en/publications/anycast-vs-ddos-evaluating-the-november-2015-root-dns-event.

[43] Ayman Mukaddam, Imad Elhajj, Ayman Kayssi, and Ali Chehab. IP Spoofing Detection Using Modified Hop Count. In *2014 IEEE 28th International Conference on Advanced Information Networking and Applications*, pages 512–516, May 2014. ISSN: 2332-5658. URL: https://ieeexplore.ieee.org/abstract/document/6838707, doi:10.1109/AINA.2014.62.

[44] Masayuki Ohta, Yoshiki Kanda, Kensuke Fukuda, and Toshiharu Sugawara. Analysis of Spoofed IP Traffic Using Time-to-Live and Identification Fields in IP Headers. In *2011 IEEE Workshops of International Conference on Advanced Information Networking and Applications*, pages 355–361, March 2011. URL: https://ieeexplore.ieee.org/abstract/document/5763526, doi:10.1109/WAINA.2011.111.

[45] Jon Porter. Amazon says it mitigated the largest DDoS attack ever recorded, June 2020. URL: https://www.theverge.com/2020/6/18/21295337/amazon-aws-biggest-ddos-attack-ever-2-3-tbps-shield-github-netscout-arbor.

[46] Christian Rossow. Amplification Hell: Revisiting Network Protocols for DDoS Abuse. In *Proceedings 2014 Network and Distributed System Security Symposium*, San Diego, CA, 2014. Internet Society. URL: https://www.ndss-symposium.org/ndss2014/programme/amplification-hell-revisiting-network-protocols-ddos-abuse/, doi:10.14722/ndss.2014.23233.

[47] L. Rudman and B. Irwin. Characterization and analysis of NTP amplification based DDoS attacks. In *2015 Information Security for South Africa (ISSA)*, pages 1–5, August 2015. URL: https://ieeexplore.ieee.org/document/7335069, doi:10.1109/ISSA.2015.7335069.

[48] R. de Oliveira Schmidt, John Heidemann, and Jan Harm Kuipers. Anycast Latency: How Many Sites Are Enough? May 2016. Publisher: University of Southern California. URL: https://research.utwente.nl/en/publications/anycast-latency-how-many-sites-are-enough.

[49] Daniel Senie and Paul Ferguson. Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing. Request for Comments RFC 2827, Internet Engineering Task Force, May 2000. Num Pages: 10. URL: https://datatracker.ietf.org/doc/rfc2827, doi:10.17487/RFC2827.

[50] Tanishka Shorey, Deepthi Subbaiah, Ashwin Goyal, Anuraag Sakxena, and Alekha Kumar Mishra. Performance Comparison and Analysis of Slowloris, GoldenEye and Xerxes DDoS Attack Tools. In *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 318–322, September 2018. URL: https://ieeexplore.ieee.org/abstract/document/8554590, doi:10.1109/ICACCI.2018.8554590.

[51] Raffaele Sommese, Gautam Akiwate, Mattijs Jonker, Giovane C. Moura, Marco Davids, Roland van Rijswijk-Deij, Geoffrey M. Voelker, Stefan Savage, and Anna Sperotto. Characterization of anycast adoption in the DNS authoritative infrastructure. In *Network Traffic Measurement and Analysis Conference (TMA'21)*, 2021. URL: https://par.nsf.gov/biblio/10287364.

[52] A. Srivastava, B. B. Gupta, A. Tyagi, Anupama Sharma, and Anupama Mishra. A Recent Survey on DDoS Attacks and Defense Mechanisms. In Dhinaharan Nagamalai, Eric Renault, and Murugan Dhanuskodi, editors, *Advances in Parallel Dis-*

tributed Computing, pages 570–580, Berlin, Heidelberg, 2011. Springer. doi:10.1007/978-3-642-24037-9_57.

[53] Olivier van der Toorn, Johannes Krupp, Mattijs Jonker, Roland van Rijswijk-Deij, Christian Rossow, and Anna Sperotto. ANYway: measuring the amplification DDoS potential of domains. In *2021 17th International Conference on Network and Service Management (CNSM)*, pages 500–508. IEEE, 2021. URL: https://ieeexplore.ieee.org/abstract/document/9615596/.

[54] 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 Conference on Internet Measurement Conference*, IMC '14, pages 449–460, New York, NY, USA, November 2014. Association for Computing Machinery. doi:10.1145/2663716.2663731.

[55] Haining Wang, Cheng Jin, and Kang G. Shin. Defense Against Spoofed IP Traffic Using Hop-Count Filtering. *IEEE/ACM Transactions on Networking*, 15(1):40–53, February 2007. Conference Name: IEEE/ACM Transactions on Networking. URL: https://ieeexplore.ieee.org/abstract/document/4100726, doi:10.1109/TNET.2006.890133.

[56] Lan Wei and John Heidemann. Does anycast hang up on you? In *2017 Network Traffic Measurement and Analysis Conference (TMA)*, pages 1–9, June 2017. URL: https://ieeexplore.ieee.org/document/8002905, doi:10.23919/TMA.2017.8002905.

[57] Ramin Yazdani, Roland van Rijswijk-Deij, Mattijs Jonker, and Anna Sperotto. A Matter of Degree: Characterizing the Amplification Power of Open DNS Resolvers. In *Passive and Active Measurement: 23rd International Conference, PAM 2022, Virtual Event, March 28–30, 2022, Proceedings*, pages 293–318, Berlin, Heidelberg, March 2022. Springer-Verlag. doi:10.1007/978-3-030-98785-5_13.

Appendix A

Supporting material

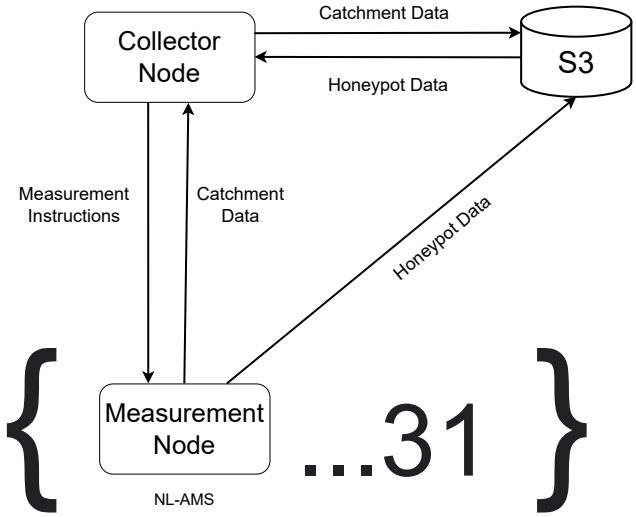

FIGURE A.1: A simplified diagram picturing a top-down view of our measurement setup and how data flows between the various entities.

OS	**Initial TTL Value**
Max value	*255*
Network Equipment	255
Linux/Mac OS	64
Windows XP and newer	128
Windows versions before XP	32

TABLE A.1: Initial TTL values for common operating systems.

Location Code	Location
au-mel	Melbourne, Australia
au-syd	Sydney, Australia
br-sao	São Paulo, Brazil
ca-yto	Toronto, Canada
cl-scl	Santiago, Chile
de-fra	Frankfurt, Germany
es-mad	Madrid, Spain
fr-cdg	Paris, France
gb-lhr	London, United Kingdom
gb-man	Manchester, United Kingdom
il-tlv	Tel Aviv, Israel
in-blr	Bengaluru, India
in-bom	Mumbai, India
in-del	New Delhi, India
jp-itm	Osaka, Japan
jp-nrt	Tokyo, Japan
kr-icn	Seoul, South Korea
mx-mex	Mexico City, Mexico
nl-ams	Amsterdam, The Netherlands
pl-waw	Warsaw, Poland
se-sto	Stockholm, Sweden
sg-sgp	Singapore, Singapore
us-atl	Atlanta, United States
us-dfw	Dallas, United States
us-ewr	Newark, United States
us-hnl	Honolulu, United States
us-lax	Los Angeles, United States
us-mia	Miami, United States
us-ord	Chicago, United States
us-sea	Seattle, United States
us-sjc	San Jose, United States
za-jnb	Johannesburg, South Africa

Table A.2: All 32 of our measurement nodes with their location and location codes.

Appendix B

Algorithms

Algorithm 1 Calculate hop count from TTL

Input: ttl
Output: hop
if $ttl > 128$ **then**
    $hop \leftarrow 255 - ttl$
else if $ttl > 64$ **then**
    $hop \leftarrow 128 - ttl$
else if $ttl > 32$ **then**
    $hop \leftarrow 64 - ttl$
else
    $hop \leftarrow 32 - ttl$
end if

Algorithm 2 Building an anycast prospect list

Input: $anycast\_catchment \leftarrow [(src\_addr, ttl), ...], jitter, spoofer\_hops$
Output: $anycast\_prospect\_list$

$anycast\_catchment \leftarrow convertTTLToHops()$
$anycast\_prospect\_list \leftarrow [\,]$

for $(src\_addr, hop) \in anycast\_catchment$ **do**
    if $hop \geq spoofer\_hops - jitter$ **and** $hop \leq spoofer\_hops + jitter$ **then**
        Append (src_addr, hop) to $anycast\_prospect\_list$
    end if
end for

$anycast\_prospect\_list \leftarrow convertSourceToRPFX()$
Return $anycast\_prospect\_list$

Algorithm 3 Building a unicast prospect list

Input:
    $unicast\_catchments \leftarrow \{nl\_ams : [(src\_addr, ttl), ...], ...\}$,
    $spoofer\_hops \leftarrow [(site, hop), ...]$,
    $jitter$,
    mv_score
Output: $unicast\_prospect\_list$

$unicast\_catchments \leftarrow convertTTLToHops()$
$rpfx\_count \leftarrow \{\}$
$unicast\_prospect\_list \leftarrow [\ ]$

for $(site, catchment) \in unicast\_catchments$ **do**
    $spoofer\_hop \leftarrow spoofer\_hops[site]$
    $rpfx\_set \leftarrow \emptyset$

    for $(src\_addr, hop) \in catchment$ **do**
        if $hop \geq spoofer\_hop - jitter$ **and** $hop \leq spoofer\_hop + jitter$ **then**
            $(rpfx) \leftarrow convertSourceToRPFX(src\_addr)$
            Add $(rpfx)$ to $rpfx\_set$
        end if
    end for

    for $(rpfx) \in rpfx\_set$ **do**
        if $(rpfx) \in rpfx\_count$ **then**
            $rpfx\_count[(rpfx)] \leftarrow rpfx\_count[(rpfx)] + 1$
        else
            $rpfx\_count[(rpfx)] \leftarrow 1$
        end if
    end for

end for

for $(rpfx, hop)$ **in** $rpfx\_count$ **do**
    if $rpfx\_count[(rpfx)] \geq mv\_score$ **then**
        Append $(rpfx)$ to $unicast\_prospect\_list$
    end if
end for

Return $unicast\_prospect\_list$