UNIVERSITY OF TWENTE

MASTER THESIS

Discovery and Quantification of Open DNS Resolvers on IPv6

Author:
Tim KLEIN NIJENHUIS

Examination Committee:
Prof.dr.ir. Aiko PRAS
dr. Anna SPEROTTO
dr. Andreas PETER
Luuk HENDRIKS, MSc

A thesis submitted in fulfillment of the requirements for the degree of Master of Embedded Systems

in the

Design and Analysis of Communication Systems Group

February 16, 2018

University of Twente

Abstract

Faculty of Electrical Engineering, Mathematics and Computer Science

Master of Embedded Systems

Discovery and Quantification of Open DNS Resolvers on IPv6

by Tim KLEIN NIJENHUIS

Open DNS resolvers pose a significant threat to the global network infrastructure by answering recursive queries for hosts outside of their domains. They are commonly (mis)used in DDoS attacks. Techniques are available to discover open DNS resolvers on IPv4, this can be done by simply scanning the entire 32bit IPv4 address space. Because of the sparsely populated and vast IPv6 address space, scanning the entire 128bit IPv6 address space is not possible. Scanning the entire 128bit IPv6 address space with a powerful scanner takes around 7.5x10^23 years. In this thesis we analyze methods to discover open DNS resolvers on IPv6. Active and passive measurements are studied to discover open DNS resolvers. Network traffic is analyzed to provide another source open DNS resolvers. We also analyze methods to quantify the performance of the open DNS resolvers. We try to estimate the bandwidth that the open DNS resolvers can achieve. And we quantify the performance of the open DNS resolvers based on their IP addresses.

Contents

List of Figures

List of Tables

List of Abbreviations

AS *Autonomous System*. iv, 11

ASN *Autonomous System Number*. iv, 6, 7, 13, 14, 16–18, 20, 36, 40

BGP *Border Gateway Protocol*. iv, 13, 36

CGA *Cryptographically Generated Addresses*. iv, 28

DDoS *Distributed Denial-of-Service*. iv, 1–5, 8

DHCP *Dynamic Host Configuration Protocol*. iv, 9, 28

DHCPv6 *Dynamic Host Configuration Protocol for IPv6*. iv, 9

DNS *Domain Name System*. iv, 1–14, 16, 18, 19, 23–27, 29, 32, 33, 36–41

DRDoS *Distributed Reflected Denial-of-Service*. iv, 8

EDNS *Extension mechanisms for DNS*. iv, 24

EEMO *Extensible Ethernet MOnitor*. iv, 18

ICMP *Internet Control Message Protocol*. iv, 8, 10, 24, 26

IID *Interface Identifier*. iv, 7, 9, 27–29, 40

IP *Internet Protocol*. iv, 1, 4, 5, 9, 11, 13, 36, 37, 40

IPv4 *Internet Protocol version 4*. iv, 1, 3, 5, 7–9, 36, 41

IPv6 *Internet Protocol version 6*. iv, 1, 3–10, 12–14, 18–20, 23–29, 32, 36–41

ISP *Internet Service Provider*. iv, 8, 20, 39, 40

MAC *Media Access Control*. iv, 7, 27, 28

MTU *Maximum Transmission Unit*. iv, 23, 32

RA *Recursion Available*. iv, 14, 19

RD *Recursion Desired*. iv, 14, 19

RFC *Request For Comments*. iv

RIB *Routing Information Base*. iv, 13, 36

RR *Resource Record*. iv, 2

RTT *round-trip-time*. iv, 10

SLAAC *Stateless Address Autoconfiguration*. iv, 7, 9, 27–29, 40

TCP *Transmission Control Protocol*. iv, 8, 9, 23

TLD *Top-Level Domain*. iv, 1, 11, 12, 38

TTL *time-to-live*. iv, 10, 11

UDP *User Datagram Protocol*. iv, 6, 8, 16, 23, 26, 27, 32

1 Introduction

Open *Domain Name System* (DNS) resolvers pose a significant threat to the global network infrastructure by answering recursive queries for hosts outside of their domains. They are commonly (mis)used in *Distributed Denial-of-Service* (DDoS) attacks [30].

Techniques are available to find open DNS resolvers on *Internet Protocol version 4* (IPv4), this can be done by scanning the entire 32bit IPv4 address space. With zmap [35] and a computer with a gigabit connection, the entire public IPv4 address space can be scanned in under 45 minutes. With a 10 gigabit connection and PF_RING[1] [29], zmap can scan the IPv4 address space in 5 minutes. When the same 10 gigabit connection and zmap are used for the entire *Internet Protocol version 6* (IPv6) address space, which is 128bit, the total scanning time will be around 7.5x10^23 years[2]. It is clearly not possible to simply scan the entire IPv6 address space. Hence, different techniques are needed to be able to find the open DNS resolvers that are reachable on IPv6. This research investigates methods to find open DNS resolvers on IPv6 and quantify the performance of the found resolvers.

The DNS is explained in Section 1.1. DNS resolvers are explained in Section 1.2 and what makes a DNS resolver an open DNS resolver in Section 1.2.1. The research questions are listed in Section 1.3 and the approach to tackle the research questions in Section 1.4. Section 2 presents the related work. Section 3 presents the methodology that is used in this research and intermediate results. Section 4 presents the methods to discover open DNS resolvers. Section 5 presents the quantification of open DNS resolvers. The discussion is presented in Section 6. And the conclusion and future work is presented in Section 7.

1.1 Domain Name System

According to the specifications RFC1034 [26] and RFC1035 [27], the DNS is a distributed database system that maps host names to *Internet Protocol* (IP) addresses. The domain name space is a tree structure. Each node and leaf on the tree corresponds to a resource set. Each node has a label, which is zero to 63 octets in length. Brother nodes may not have the same label. One label is reserved, that is the null (i.e., zero length) label, which is used for the root. The domain name of a node is the list of labels on the path from the node to the root of the tree. When the domain name is typed in for example a browser, the labels are separated by dots and the trailing dot is omitted. An example domain name that can be typed in a browser and without the trailing dot is "www.utwente.nl".

The top most node of the tree hierarchy is the root domain, represented by a single dot. The next level in the hierarchy are *Top-Level Domain*s (TLDs), e.g. .com, .org, .net, .edu, etc. Each domain is a subdomain of another domain if it is contained within that domain. The subdomain's name ends with the containing domain's

[1] A new type of socket that dramatically improves the packet capture speed

[2] (5 minutes / hour / day / year) * 2^(128bit address space - 32bit address space)

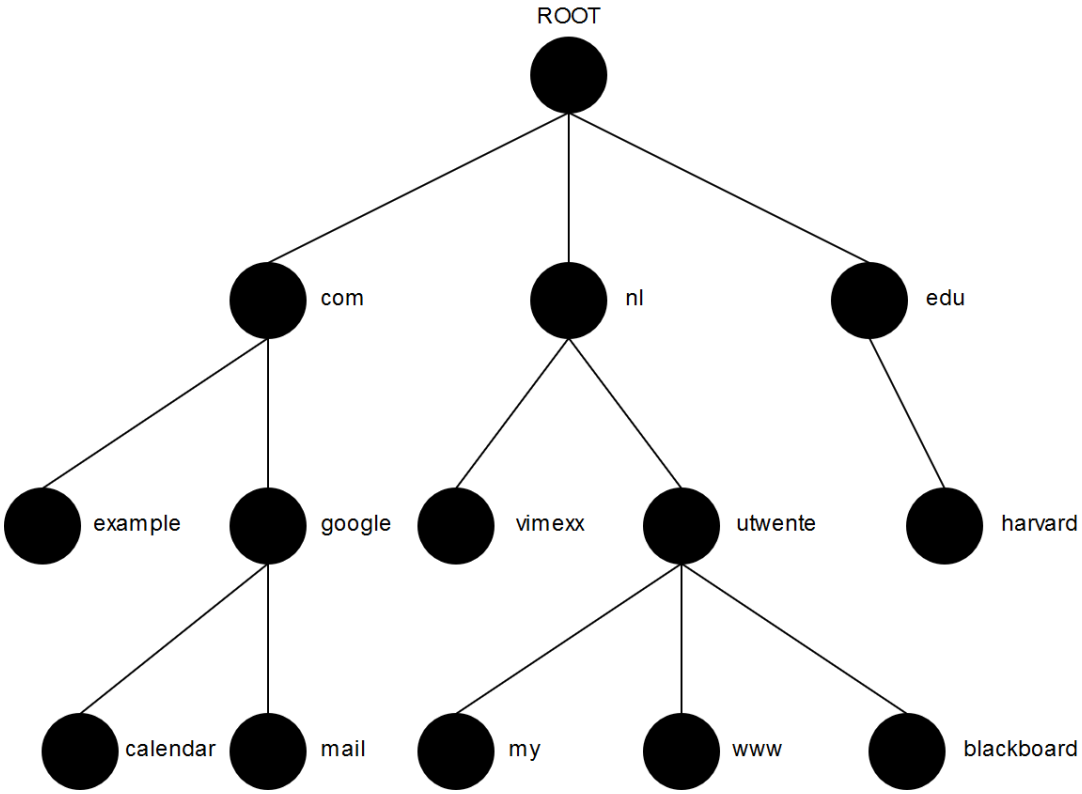

FIGURE 1.1: Example DNS tree with a number of nodes

name. For example, the domain www.utwente.nl is a subdomain of utwente.nl, nl and " " (the root). An example tree structure is shown in Figure 1.1.

The DNS has three major components:

- The ***Domain Name System*** and ***Resource Records***, which are specifications for a tree structured name space and data associated with the names. Each node and leaf of the domain name space tree names a set of information, and query operations are attempts to extract specific types of information from a particular set. A query contains the requested domain name and describes the type of resource information that is desired.

- **Name Servers** are server programs which hold information about the domain tree's structure and resource set. Name servers know the parts of the domain tree for which they have complete information. A name server is said to be an **authority** for these parts of the name space and the name server is also called an authoritative name server. Authoritative information is organized in units called **zones**, and these zones can be automatically distributed to the name servers which provide redundant service for the data in a zone.

- **Resolvers** are programs that extract information from name servers in response to clients' requests. Resolvers must be able to access at least one name server and use that name server's information to answer a query directly, or pursue the query using referrals to other name servers higher or lower in the domain tree.

Common DNS queries [2] are shown in Table 1.1. The ANY query is often used for DDoS attacks, since there could be a large number of records in the cache of a

Query type	Domain	Query response
A	utwente.nl	IPv4 address record: 130.89.3.249
AAAA	utwente.nl	IPv6 address record: 2001:67c:2564:a102::1:1
ANY	utwente.nl	All cached records
CNAME	utwente.nl	Conical name record, alias to another domain name
MX	utwente.nl	Mail exchange record: 10 mx-1.mf.surf.net. 10 mx-a.mf.surf.net.
NS	utwente.nl	Name server record: ns1.utwente.nl. ns2.utwente.nl. ns3.utwente.nl.
PTR	249.3.89.130.in-addr.arpa	Pointer record, reverse DNS lookup: webhare.civ.utwente.nl.
SIG	utwente.nl	Signature record
SOA	utwente.nl	Start of authority record: ns1.utwente.nl. dnsmaster.utwente.nl. 2170704100 28800 7200 604800 1200
SRV	utwente.nl	Generalized service location record
TXT	utwente.nl	Text record, carries extra data, sometimes human-readable. In this case an Office 365 DNS record: "MS=ms81740447"

TABLE 1.1: Common DNS queries

caching DNS resolver. Another possibility is a domain specifically set up for DDoS attacks. Such a domain can contain a large number of (bogus) A or AAAA records, or many large TXT records. When a query is send to such a domain, the response is large and the amplification factor is high.

1.2 DNS Resolvers

DNS resolvers are programs that extract information from name servers in response to clients' requests [15]. The resolver is located on the same machine as the program that requests the resolver's service, but it may need to consult name servers on other hosts. A resolver performs queries and receives answers, the logical function is called resolution.

There are different types of resolvers, three of them are discussed below:

- **Stub resolver:** A resolver that cannot perform all resolution itself. The stub resolver is part of the OS and is used by applications to handle the DNS lookups.

Description	Amount of resolvers
Responses to udp/53 probe	15.486.362
Unique IPs	15.212.443
Gave correct answer to an A-record query	9.826.035
Response had recursion-available bit set	10.369.665

TABLE 1.2: Resolvers found by the Open Resolver Project

This resolvers sends the DNS queries to a specified recursive resolver to undertake the actual resolution function.

- **Iterative mode:** A resolution mode of a server that receives DNS queries and responds with a referral to another server. The server refers the client to another server and lets the client pursue the query. Such a resolver is sometimes called an iterative resolver.
- **Recursive mode:** A resolution mode of a server that receives DNS queries and either responds to those queries from a local cache or sends queries to other servers in order to get the final answers to the original queries. Systems operating in this mode are commonly called recursive servers or recursive resolvers.

The amount of open DNS resolvers found by the Open Resolver Project [30] as of 2017-01-22, is shown in Table 1.2. According to the Open Resolver Project data, around 10 million resolvers can resolve recursively and slightly less than 10 million resolvers gave the correct answer to the query.

1.2.1 Open DNS Resolvers

Recursive open DNS resolvers are DNS resolvers running in recursive mode while accepting all queries, not just from hosts in the internal network for example. This means that the resolver will provide an answer to any incoming query. These resolvers can be used by an attacker to execute a DDoS attack. The attacker sends a DNS query with a spoofed source IP address of the victim to the resolver. An example of source IP address spoofing of a DNS query is shown in Figure 1.2. The resolver will look for the answer in its cache and otherwise execute iterative queries until the answer has been found. Because the source IP address of the query is spoofed, the victim will receive the answer instead of the attacker who sent the query. The attacker chooses a query type and domain which returns a big response for a small request, which is then send to the victim. The difference between the size of the result and the request is called the amplification factor. Instead of launching the attack himself, the attacker can use a botnet consisting of many infected bots. This results in an even greater amplification factor, and the attacker remains anonymous.

As an example, suppose a botnet containing 100 machines is used to execute a DDoS attack. A domain is chosen (or created) which has an amplification factor of 50 between answer and request. When all machines in the botnet send 25kb/s of traffic to the resolver, the traffic from all 100 machines become 2.5Mb/s. The 2.5Mb/s of traffic is send to the resolver, and it responds with 125Mb/s of traffic that is send to the victim. Not only can this exhaust the bandwidth of victim, but also the resources on the client machine that is processing this traffic [32]. This emphasizes the importance of finding the open DNS resolvers on IPv6.

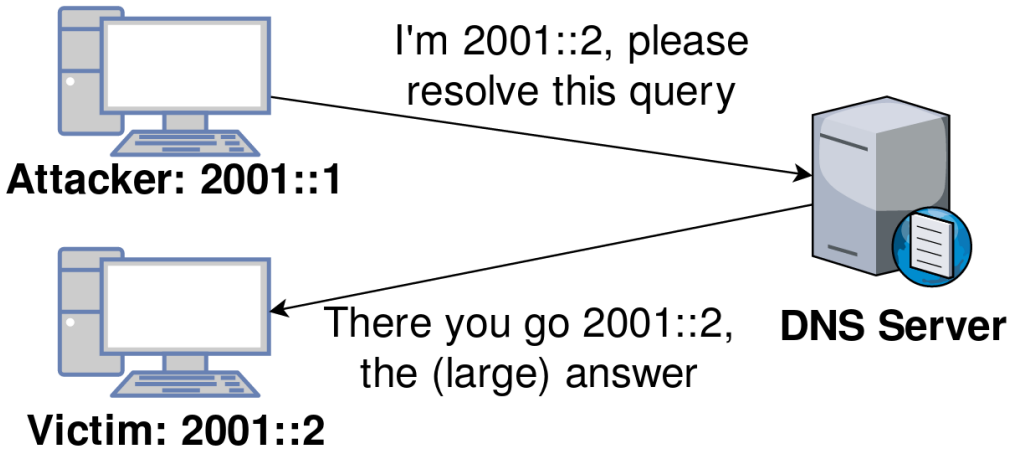

FIGURE 1.2: Source IP address spoofing

1.3 Research Questions

This research indicates whether IPv6 open DNS resolvers pose a threat when used for DDoS attacks, in addition to the resolvers on IPv4. The research focuses on discovery and quantification of open DNS resolvers on IPv6 and the following research questions can be formulated:

- How can we discover open DNS resolvers on IPv6?
  - How can active or passive measurements be used to find open DNS resolvers?
  - How and what network traffic can be analyzed to find open DNS resolvers?
- How can we quantify the performance of the open DNS resolvers?
  - How can we measure the bandwidth without performing a (D)DoS?
  - How can the IP address of the open DNS resolver be used to estimate its performance?

1.4 Research Approach

The approach to the discovery of open DNS resolvers is presented in Section 1.4.1. The approach to quantify the performance of the open DNS resolvers is presented in Section 1.4.2.

1.4.1 Approach to Discover Open DNS Resolvers

The first step to discover the open DNS resolvers is to create an address list with IPv6 addresses. The first source of IPv6 addresses is the hitlist collection by Gasser et al. [10]. The hitlists contain IPv6 addresses from multiple passive, active and traceroute sources. The second source of IPv6 addresses is authoritative nameserver traffic. This is traffic that is send between DNS resolvers and the nameserver. Network traffic data of the ns1 authoritative nameserver at SURFnet [34] is used to collect IPv6 addresses. A third possible source is the IPv6 address of nameservers of a large list with domain names. The sources are processed to extract a list of unique IPv6 addresses.

To find out whether the found IPv6 addresses host open DNS resolvers, the addresses must be scanned. This is done by sending a DNS query to every IPv6 address and wait for an answer. The scanning is performed using zmap. If an answer is received and the result code that is returned specifies that no error occurred, an open DNS resolver is found. There is, however, a problem with classifying DNS resolvers as 'open' when using the output of zmap. A lot of resolvers set the return code in the answer to 'no error', while they don't provide the answer to the query. That is why a validation step is used to distinguish between the resolvers that can answer queries, thus providing recursion, and those who can't.

To filter out the resolvers that don't perform recursion while the return code is 'no error', a validation step is performed. A validation query is sent to all the resolvers classified as 'open' by the zmap scan. This query asks the resolvers to resolve a domain for its A record. The same query is performed with the measurement machine's default DNS resolver to get the correct A record of the domain. This correct answer is compared with the answers of the resolvers to determine which resolvers are capable of performing recursive queries and provide a correct answer.

Using the results of these measurements, the research questions regarding the discovery of open DNS resolvers can be answered.

1.4.2 Approach to Quantify Open DNS Resolvers

The two packet bandwidth measurement approach from the literature is used. The problem with using that approach to measure the bandwidth of open DNS resolvers, is that there is no control on both sides of the measurement, the measuring side that sends the queries and the resolver answering them. The goal is to calculate the bandwidth of the open DNS resolvers. When that isn't possible or the result isn't accurate enough, a lower bound of the resolver's bandwidth is calculated to make a distinction between 'fast' and 'slow' resolvers.

An authoritative nameserver is installed on the measurement machine and used to measure the bandwidth of the open DNS resolvers. The open resolvers are asked to resolve the domain of which the authoritative nameserver has been setup. This way, the size of the answer of the query can be made sufficiently large to cause packet fragmentation. The resolver ends up sending several (depending on how large the answer is) packet directly after each other to the querying machine. Then the two packet bandwidth measurement approach can be used to calculate the bandwidth of the resolver. On the measurement machine, the delay between the fragmented packets arriving is measured to calculate the bandwidth of the resolver.

Resolver classification techniques:

- Supports the resolver sending fragmented *User Datagram Protocol* (UDP) packets
- What is the maximum message size that the open DNS resolver can send
- What is the amplification factor that can be achieved with the resolver
- What is the message size that resolvers that don't support recursion return
- What is the amplification factor that can be achieved with the resolvers that don't support recursion
- What is the *Autonomous System Number* (ASN) of the resolver

Different types of IPv6 addresses can be distinguished. The main difference is whether the IPv6 address is automatically generated or manually configured. The automatically generated IPv6 addresses can originate from *Stateless Address Autoconfiguration* (SLAAC). The *Interface Identifier* (IID) of the IPv6 address is created as follows. The machine's *Media Access Control* (MAC) address is cut in half and the value 'ff:fe' is placed in between. Then the 7th bit of the address is complemented (e.g., 'swapped'). The IPv6 address then consists of the network prefix and the IID. For example, the MAC address '12:34:56:ab:cd:ef' and the network prefix of '20a2::' results in the IPv6 address: '20a2::1034:56ff:feab:cdef'. SLAAC with privacy extensions [28] is also used, this uses random values after the network prefix. An example IPv6 address generated using SLAAC with privacy extensions address is: '2a02::944d:15c1:e1a8:5e74'. Manually configured IPv6 addresses tend to be simpler with the main function encoded in the address, for example 53 for a DNS resolver, since DNS runs on port 53. Or the IPv4 address can be encoded in the IPv6 address. Example of those manually configured addresses are: '2a02::53' and '2a02::192:168:2:53'.

Using the results of the bandwidth measurement, classification techniques, the type of the resolver's IPv6 address and the ASN that the resolver belongs to, the research questions regarding the quantification of open DNS resolvers can be answered.

2 Related Work

Many studies propose methods to detect or mitigate reflected DDoS attacks, also called *Distributed Reflected Denial-of-Service* (DRDoS) attacks, that use open DNS resolvers as reflector. Sherin Jose et al. [19] proposes a DDoS detection and mitigation strategy using Hadoop MapReduce and Chukwa, and Shi-Ming Huang et al. [17] proposes text based Turing testing in a cloud computing environment. Numerous studies [1, 31, 8, 23] propose a DDoS detection approach based on features in the packets and (machine-)learning. Jyothi et al. [20] proposes a DDoS detection framework based on machine learning, which uses low level hardware events of the detection system. Instead of detecting and preventing DDoS attacks at the victim side, Sadeghian et al. [33] proposes to implement a self triggered black hole at the *Internet Service Provider* (ISP) networks. This will detect and drop the malicious traffic at the edge of the attacker's ISP.

The detection and mitigation of a DDoS attack does not tackle the problem that there are still many open DNS resolvers on the Internet. It's better to prevent DDoS attacks than trying to mitigate or reduce the damage. With the IPv6 adoption increasing recently [12, 7], it becomes more important to find open DNS resolvers on IPv6. Hendriks et al. [14] proposes a technique to find IPv6 open DNS resolvers which also openly resolve on IPv4. Both Dual stacked resolvers, and forwarding resolvers that receive queries over either IPv4 or IPv6 and forward them to the respective upstream resolver, are found by that technique.

Section 2.1 presents the related work on finding open resolvers. And Section 2.2 discusses the related work in performance quantification of open resolvers.

2.1 Finding Open DNS Resolvers

This section lists the related work on finding open DNS resolvers. Methods to find open DNS resolvers on IPv4 and a combination of IPv4 and IPv6 is presented in section 2.1.1, and methods to generate an IPv6 address hitlist in Section 2.1.2.

2.1.1 Finding Resolvers on IPv4

Durumeric et al. [4] proposes methods to scan the entire IPv4 address space. Zmap can scan both over *Transmission Control Protocol* (TCP) and UDP. Zmap is used to perform single packet host discovery scans against the IPv4 address space. Zmap can perform three types of host discovery scans [5]. By default, zmap will perform a TCP SYN scan of all hosts at a single port. An *Internet Control Message Protocol* (ICMP) echo scan, and a UDP datagram can also be used to perform host discovery.

After the host discovery is performed, the active hosts will seed pluggable application scanners. These application scanners typically perform a single TCP three-way handshake and measure one aspect of how a service is configured. Zgrab [35] is used as the application scanner. Zgrab supports HTTP, HTTPS, SSH, Telnet, FTP, SMTP, POP3, IMAP, Modbus, BACNET, Siemens S7, and Tridium Fox.

To find open DNS resolvers the on IPv4 address space, the DNS probe module of zmap is used. This module will ask every host on the IPv4 network to resolve the A record of 'google.com'. If the correct answer to the query is received, the probed IP address is hosting an open DNS resolver.

Hendriks et al. [14] proposes a methodology to find DNS resolvers openly resolving over IPv6, which also openly resolve over IPv4. Zmap is used to find open DNS resolvers on IPv4. Those found resolvers are queried for an A record of a subdomain, where the nameserver of the subdomain is only reachable over IPv6. The resolver first resolves the domain, and queries the NS record of the subdomain. There is only an AAAA record of the subdomain's nameserver and not an A record. This means that the nameserver of the subdomain can only be queried via an IPv6 connection. When the query for the A record arrives at the subdomain's nameserver, the resolver has IPv6 connectivity. To verify that this found IPv6 resolver is an open resolver, a verification query is send to the IPv6 address of the resolver that made the query for the A record. When that query arrives at the subdomain's nameserver, we know that this is an open resolver on IPv6.

This research also distinguishes dual-stack resolvers from forwarding resolvers. A dual stacked resolver is a single resolver that resolves both on IPv4 and IPv6. Forwarding resolvers receive queries and forward them to other recursive resolvers downstream, thus not performing the resolution themselves. To distinguish between dual stacked and forwarding resolvers, a query is send to a subdomain where the subdomain's nameserver is only reachable over IPv4. Comparing the IPv4 address that contacts the subdomain's nameserver and the IPv4 address where the query was send to, indicates whether it is a single machine or forwarding or distribution has occurred.

2.1.2 IPv6 Address Hitlist

IPv6 offers a much larger address space than that of the its IPv4 counterpart. Since the full IPv6 address space cannot be scanned, other methods to find IPv6 addresses are needed. One of those methods is hitlist generation. To generate such a hitlist, selective parts of the IPv6 address space can be scanned. IPv6 incorporates two automatic address-configuration mechanisms: SLAAC and *Dynamic Host Configuration Protocol* (DHCP) for IPv6 addresses called *Dynamic Host Configuration Protocol for IPv6* (DHCPv6) [11]. Techniques exist to reduce the search space when SLAAC and DHCPv6 are used to automatically configure IPv6 addresses. In some cases, IPv6 addresses are assigned manually by network administrators. Some patterns can be distinguished in these assignments. The most common form is low-byte addresses. All bytes in the IID are then zero except the least significant bytes. Also, the IPv4 address can be encoded in each of the 16-bit words of the IID. Another common form is service-port addresses. Here the service, for example a web-server, is encoded as port number, 80, in the IID. Also, when multiple machines run a web-server, an index or value can be added in another 16-bit word. Since the IPv6 notation allows for hexadecimal digits, words can be encoded into IPv6 addresses, for example: 2001:db8::dead:beef. All these manual address assignment patterns reduce the search space of IPv6 addresses or provide other ways to guess or brute-force addresses.

Next to scanning the IPv6 address space, other methods from research by Gasser et al. [9] to generate a hitlist, are passive and active address sources and traceroutes. Nameserver traffic provides IPv6 addresses as source and destination address. Using the Alexa top 1 million list and querying the domains for AAAA records, provide

IPv6 addresses. Running traceroutes to the addresses that are in the IPv6 hitlist can also provide additional addresses.

2.2 Performance Quantification

This section lists the related work on performance quantification of open DNS resolvers. The bandwidth estimation of the open resolvers is presented in Section 2.2.1, and the fingerprinting and classification of open resolvers in Section 2.2.2

2.2.1 Bandwidth Estimation

Mihailescu [25] presents techniques to estimate capacity and bandwidth of a path.

- **Variable Packet Size probing** aims to measure the capacity at each hop along a path. The key element of this technique is that the *time-to-live* (TTL) field is used to force the packet to expire at a particular hop. The router at that hop discards the probing packet and sends an ICMP time-exceeded error message back to the host. This is used in combination with the packet that is send, to measure the *round-trip-time* (RTT) to that hop. The RTT to each hop consists of three delay components: serialization delays, propagation delays and queuing delays. The serialization delay of a packet of size L at a link of transmission rate C is the time to transmit the packet on the link, equal to L/C. The propagation delay of a packet at a link is the time it takes for each bit to traverse the link, and is independent of the packet size. Finally, queuing delays can occur in the buffers of the routers and switches on the path. Variable Packet Size probing sends multiple packets and assumes that at least one of the packets arrive without queuing delays. For each hop, the slope of the RTT is calculated by using different packet sizes L. By using the calculated slope of each hop, the capacity C of a hop i is calculated as:

$$C_i = \frac{1}{slope_i - slope_{i-1}}$$

- **Packet pair and packet trains** uses multiple measurements of packet pairs or a train of packets, to estimate the capacity of the path. A lot of measurements are needed to estimate the capacity, as the individual measurements vary greatly. Measurements can return values far below the actual capacity or even above [3]. Also, the machines on both sides of the measurement must be accessible in order to measure the time between packets at the sender and the receiver side.

- **Self Loading Periodic Streams** sends a stream of equal sized packets at a certain rate. If the rate is greater than the path's available bandwidth, this causes a short-term overload. One-way delays will keep increasing as each packet of the stream queues up. On the other hand, if the stream rate is lower than the available bandwidth, the one-way delay will not increase. This effectively means that the rate is increased until the path is flooded with traffic and the available bandwidth is reached.

- **Trains of Packet Pairs** sends packet pairs at increasing rates. The basic idea of this approach is analogous to the self loading periodic streams. Most of the differences between the two methods are related to the statistical processing of the measurements.

A more mathematical approach to the capacity estimation, cross traffic analysis and queuing delays at a link, is presented by Kang et al. [21].

2.2.2 Fingerprinting and Classification

Research on resolver fingerprinting and classification by Kührer et al. [22] proposes techniques to classify open DNS resolvers.

- **Geographical localization:** determine the geographical location of the resolver by using a GeoIP database. Using the location, the found open DNS resolvers can be categorized by country. The *Autonomous System* (AS) distribution can be used to categorize the found open DNS resolvers by network provider.
- **Fingerprinting software:** identify the DNS server software by using CHAOS version.bind and version.server requests. The queries hostname.bind and id.server, return server information when the queries are implemented in the DNS resolver [36]. A refinement to the BIND-based mechanism, which dropped the implementation-specific label, replaces BIND with SERVER, so both .bind and .server queries could be implemented in a DNS server.
- **Fingerprinting hardware:** aside from the information from the DNS server, information about the OS and the hardware specifications can be found when the resolver is also running other services besides DNS. For that, we initiate FTP, HTTP, HTTPS, SSH and telnet connections and analyze the banner information and text fragments to fingerprint the device.
- **IP address churn:** analyze whether the IP address of the DNS resolver changes over time. DNS resolvers running on devices with low IP lease times such as consumer routing devices, often change IP addresses. This is measured by sending queries to previously found IP addresses of open resolvers and verifying whether those addresses still host open resolvers.
- **Resolver utilization:** determine the utilization of the DNS resolver by using DNS cache snooping. The NS records for TLDs are requested every hour for 36 hours, to monitor the TTL values. Every request it is checked whether the TLDs are added to the resolver's cache after the TTL values expire. If that's the case, a client performed DNS requests for domains with these TLDs, causing the resolver to request information from the corresponding authoritative nameservers. This means that the DNS resolver is used at least once since the TTL values of the TLDs in the resolver's cache have expired.

3 Methodology

This chapter describes the methodology used in this research. Section 3.1 presents an overview of methods to find open resolvers. In Section 3.2 is discussed how the performance quantification of the open resolvers can be performed.

3.1 Finding Open DNS Resolvers

This section describes three methods to collect IPv6 addresses to create an IPv6 address list, which can be scanned to find open DNS resolvers. Section 3.1.1 presents the use of hitlists to collect IPv6 addresses. In Section 3.1.2 is discussed how authoritative nameserver traffic can be analyzed to collect IPv6 addresses. And Section 3.1.3 presents how domains queried for their nameservers results in additional IPv6 addresses. Section 3.1.4 presents how the resulting IPv6 address list is scanned for open DNS resolvers. In Section 3.1.5 is discussed why validation of the found open DNS resolvers is needed, and how it is performed.

3.1.1 IPv6 Address Hitlist

The hitlists by Gasser et al. [10] are used to find IPv6 addresses. The hitlists contain IPv6 addresses from multiple passive, active and traceroute sources:

- DNS zone files from various TLDs
- Alexa country list domains resolved for AAAA records
- Caida DNS names: AAAA records for entries in internet topology traces
- Ct: domains resolved for AAAA records
- Rapid7's DNS any query results filtered for IPv6 addresses
- Traceroutes to IPv6 addresses from the sources above

Recent hitlists from the website of the research of Gasser et al. [10] are downloaded by a Python crawler that scans for the latest version of the hitlists mentioned above. All hitlists above are processed into a list of unique IPv6 addresses. The domain names are removed from the hitlists to leave only the IPv6 addresses themself. The list is made unique to make sure that the same IPv6 address is only scanned once. For each IPv6 address, it is remembered in which hitlist or hitlists it was found. This results in an IPv6 address list of roughly 3.8 million unique IPv6 addresses.

3.1.2 Authoritative Nameserver Traffic

Incoming authoritative nameserver traffic contains only DNS traffic, which is send between DNS resolvers and the nameserver. This means that authoritative nameserver traffic has a higher chance of finding open DNS resolvers than ordinary IPv6

hitlists. That is because all the addresses in the authoritative nameserver traffic host DNS resolvers, while entries in ordinary IPv6 hitlists also contain devices like routers, personal computers and mobile phones.

Network traffic data of the ns1 authoritative nameserver at SURFnet [34] is used to collect IPv6 addresses. The traffic data is in 'tcpdump capture file' format and needs to be preprocessed to extract the IPv6 addresses. The traffic data consists of full packets including the packet data. The traffic data is of the month December in 2016 and is packed in 4.208 archives. Live traffic data or recent traffic data will be used later to improve the chances of finding open DNS resolvers on the IPv6 addresses extracted from the traffic data. Each archive consists of a tcpdump file with 1.000.000 packets each. Tcpdump is used to output the source and destination IPv6 addresses of the packets in the network flow. The IPv6 addresses of all network flows are combined and made unique to prevent the same IPv6 address from being scanned multiple times. And the IPv6 address of the authoritative nameserver itself is removed from the IPv6 address list. This results in an IPv6 address list of roughly 130 thousand unique IPv6 addresses.

3.1.3 Query Nameserver of Domain Lists

Approach to get IPv6 addresses from the nameservers of domains on domain lists:

- Use a domain list with domain names on it
- Send NS queries to the domains on the domain list to get the nameservers
- Send AAAA queries to the nameservers to get the IPv6 address of the nameservers, or fetch them from the additional section of the query above

Nameservers shouldn't answer queries from subnets outside of their LAN or other subnet. Nameservers that do answer the queries are open DNS resolvers.

3.1.4 Scanning the IPv6 Address List

The resulting IPv6 address lists are merged into a single list and the duplicate addresses removed. Offline ASN lookup of all the addresses in the list is performed to get the ASN of each IPv6 address. The lookup is performed offline by downloading and using a recent *Border Gateway Protocol* (BGP) *Routing Information Base* (RIB) database [18]. This is to prevent overloading online services like team Cymru's IP to ASN mapping. Zmap is used as the scanner to scan the IPv6 address list. The official zmap can't perform scans on IPv6 addresses, so the IPv6 module from the research of Gasser et al. [10] is used. Rate limiting is applied to minimize the network load, it does however result in a longer scanning time.

To get preliminary results, a subset of the resulting IPv6 address list is scanned. For the hitlists, the most recent version was retrieved on the same day as the scanning was performed. And for the authoritative nameserver traffic, a subset of eight hours of the December 2016 data was used. One percent of the addresses from the resulting IPv6 address list is scanned to get these preliminary results. By default zmap queries the resolvers to resolve 'google.com' for its A record. This is not changed to 'utwente.nl' for example, since this scanning step's goal is to get as much open resolvers as possible. When the return code of the query (rcode) indicates a successful answer (rcode is NOERROR), zmap assumes that it is an open DNS resolver.

There is, however, a problem with classifying DNS resolvers as 'open' when using the output of zmap. A lot of resolvers set the rcode to NOERROR, while they

	entries	percentage of total
all addresses	38.256	100
open resolvers	128	0.334588
verified resolvers	51	0.133312

TABLE 3.1: Difference in the number of open resolvers and validated resolvers in the 1% subset of the IPv6 address list

don't provide recursion and also don't provide the requested A record. There is a warning displayed in the answer of those resolvers, when the Linux command 'dig' is used to perform the query:
'WARNING: recursion requested but not available'
This means that the *Recursion Available* (RA) flag is not set on the answer, while the *Recursion Desired* (RD) flag is set on the query, thus the resolver does not support or refuses recursion. So the resolvers couldn't answer the query but still set rcode to NOERROR. That is why a validation step is used to distinguish between the resolvers that can answer queries, thus providing recursion, and those who can't.

3.1.5 Validating the Found Open Resolvers

To filter out the resolvers that don't perform recursion while the rcode is NOERROR, a validation step is performed. Using Python, a validation query is sent to all the resolvers found by the zmap scanning. This query asks the resolvers to resolve 'utwente.nl' for its A record. The same query is performed with the measurement machine's default DNS resolver to get the correct A record of 'utwente.nl'. This correct A record is compared with the answer of the resolvers to determine which resolvers are capable of performing recursive queries and provide a correct answer. This time the domain 'utwente.nl' is used, as 'google.com' is not reachable through the great firewall of China [13], but 'utwente.nl' is. This way, open DNS resolvers behind the great firewall of China can also be validated whether they are able to correctly resolve non-blacklisted domains.

The difference in the number of open resolvers and validated resolvers found in the 1% subset of the IPv6 address list is shown in Table 3.1. The results from the subset divided over the hitlist source where the IPv6 address is found in, are shown in Table 3.2. Note that the same IPv6 address can occur in multiple hitlists. The total number of IPv6 addresses is shown as entries. The open resolvers and ratio between open resolvers and entries, and the verified resolvers and ratio between verified resolvers and entries are shown in the table.

The ASN of the verified open resolvers found in the subset is shown in Figure 3.1. As can be seen from the figure, most of the open resolvers found in this subset are from ASN 63949, which is registered to 'LINODE-AP Linode, LLC, US', a cloud hosting service.

3.2 Performance Quantification

This section lists the performance quantification of the open DNS resolvers. Both the achievable bandwidth of the open resolvers and the classification of the open resolvers is discussed here. In Section 3.2.1 is discussed how the bandwidth of the open resolvers can be estimated. The methods to classify the open resolvers is presented in Section 3.2.2.

hitlist	entries	open resolvers	open ratio	verified resolvers	verified ratio
alexa-country	0	0	0	0	0
caida-dnsnames	1.910	3	0.157068	0	0.000000
ct	7.567	17	0.224660	6	0.079292
rapid7-dnsany	30.933	111	0.358840	39	0.126079
traceroute-v6-builtin	1.089	5	0.459137	4	0.367309
traceroute-v6-udm	372	2	0.537634	2	0.537634
au	440	0	0.000000	0	0.000000
biz	421	0	0.000000	0	0.000000
com	3.735	16	0.428380	2	0.053548
czds	1.656	4	0.241546	1	0.060386
de	2.304	0	0.000000	0	0.000000
info	812	0	0.000000	0	0.000000
mobi	76	0	0.000000	0	0.000000
net	1.206	2	0.165837	0	0.000000
nu	167	0	0.000000	0	0.000000
org	1.100	1	0.090909	1	0.090909
se	179	0	0.000000	0	0.000000
sk	24	0	0.000000	0	0.000000
xxx	10	0	0.000000	0	0.000000
surfnet-ns1	275	7	2.545455	7	2.545455
domain-ns-server-AAAA	0	0	0	0	0
all hitlists	38.256	128	0.334588	51	0.133312

TABLE 3.2: Open and validated resolvers of the 1% subset of the IPv6 address list, and the ratio between open and validated resolvers and the total entries in the subset

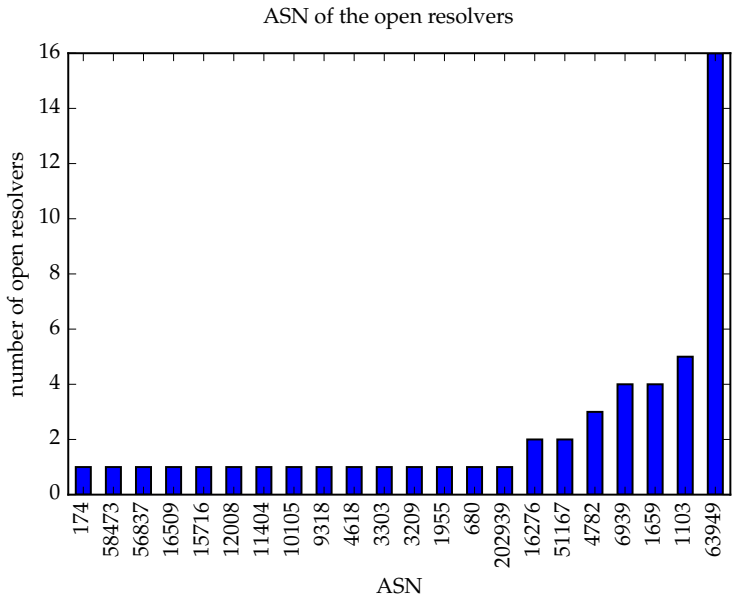

FIGURE 3.1: ASN of the verified open resolvers of the 1% subset of the IPv6 address list

3.2.1 Bandwidth Estimation

Start with the two packet bandwidth measurement approach from the related work chapter. The problem of this approach on measuring the bandwidth of open DNS resolvers is that there is no control on both sides of the measurement, the measuring side that sends the queries and the resolver answering them. The goal is to calculate the bandwidth of open DNS resolvers. When that isn't possible or the result isn't accurate enough, a lower bound of the resolver's bandwidth is calculated to make a distinction between 'fast' and 'slow' resolvers. To measure the bandwidth of open DNS resolvers, an authoritative nameserver is set up on the measurement machine. The open resolvers are asked to resolve the domain of which an authoritative nameserver has been set up. This way, the size of the answer of the query can be made sufficiently large to cause packet fragmentation. The resolver ends up sending several (depending on how large the answer is) packets directly after each other to the querying client, which is the measurement machine. Then the two packet bandwidth measurement approach can be used to calculate the bandwidth of the resolver. Alternatively small bursts of packets (queries) can be sent when either the resolver does not support fragmented UDP packets or the fragmented packets don't achieve the desired accuracy. On the measurement machine, the delay between the fragmented packets arriving is measured to calculate the bandwidth of the resolver.

A not so ethical way to estimate the bandwidth of open DNS resolvers:
Send a small stream of packets with increasing traffic rate. Measure the delay of packets between sending and receiving them. If the delay stays constant, the bandwidth of the resolver is equal to or larger than the traffic rate. Repeat the measurement with increasing traffic rate until the delay increases during the measurement. This means that the bandwidth of the resolver is lower than the traffic rate at which packets were sent.

3.2.2 Classification

The found resolvers can be classified according to the performance that the resolver can achieve or the features that are supported. The performance of the resolver can be divided in the message size and the resulting amplification factor of the open DNS resolvers. Also the message size and resulting amplification factor of the resolvers that don't support recursion. For the bandwidth measurement it is checked whether the resolver supports sending fragmented UDP packets. And the ASN of the resolvers is looked up. The classification techniques are summarized below:

- Supports the resolver sending fragmented UDP packets
- What is the maximum message size that the open DNS resolver can send
- What is the amplification factor that can be achieved with the resolver
- What is the message size that resolvers not supporting recursion return
- What is the amplification factor of the resolvers not supporting recursion
- What is the ASN of the resolver

4 Discovery of Open DNS Resolvers

This chapter describes the discovery of open DNS resolvers on IPv6. Section 4.1 presents the IPv6 address sources that are scanned for open DNS resolvers. Section 4.2 presents the methods to discover the open DNS resolvers on IPv6 by scanning the found IPv6 addresses for open DNS resolvers. In Section 4.3 is discussed why validation of the found open DNS resolvers is needed, and how it is performed. Section 4.4 presents the results of the scanning and validation.

4.1 IPv6 Address Sources

Two sources of IPv6 addresses which are also mentioned in Section 3.1, are used to provide the IPv6 addresses. The third source, query authoritative nameservers of domain lists to find out whether the nameservers support recursive queries for domains they are not authoritative for, is not implemented.

The hitlists by Gasser et al. [10] are used to find IPv6 addresses. The hitlists contain IPv6 addresses from multiple passive, active and traceroute sources. The most recent versions of the hitlists (as of 2018-01-16) are downloaded and processed into a list of unique IPv6 addresses. This results in an IPv6 address list of roughly 3.6 million (3.636.182) unique IPv6 addresses.

After the promising scan on the subset of previously acquired IPv6 addresses in Section 3.1.5, where old authoritative nameserver traffic from December 2016 was used, a live traffic feed is used for this scan. The network traffic of the ns1 authoritative nameserver at SURFnet [34] forwards a percentage of the DNS traffic to the measurement machine that is used to find open DNS resolvers. This is done with a program called *Extensible Ethernet MOnitor* (EEMO) [6]. EEMO sends the feed using an encrypted connection to the measurement machine where the source and destination IPv6 addresses are extracted. These source and destination addresses are merged into a single IPv6 address list. This results in an IPv6 address list of roughly 142 thousand (142.372) unique IPv6 addresses.

4.2 Scanning on the Full Data Set

The IPv6 address list from the two sources in Section 4.1 are merged into a single list and the duplicate addresses removed. This results in a single address list of roughly 3.75 million (3.750.436) unique IPv6 addresses. Offline ASN lookup of the addresses in the list is performed to get the ASN of all IPv6 addresses. Zmap is used as the scanner to scan the IPv6 address list. The official zmap can't perform scans on the IPv6 network, so the IPv6 module from the research of Gasser et al. [10] is used. Rate limiting to 10k packets per second is applied to minimize the network load, this results in a scan time of about 375 seconds or 6.25 minutes.

	entries	percentage of total
all addresses	3.750.436	100
open resolvers	15.800	0.421284
verified resolvers	7.011	0.186938

TABLE 4.1: Difference in the number of open resolvers and validated resolvers of the IPv6 address list

By default zmap queries the resolvers to resolve 'google.com' for its A record. This is not changed to 'utwente.nl' for example, since this scanning step's goal is to get as much open resolvers as possible. When the return code of the query (rcode) indicates a successful answer (rcode is NOERROR), zmap assumes that it is an open DNS resolver.

There is, however, a problem when classifying DNS resolvers as 'open' when the output of zmap is used. A lot of resolvers set the rcode to NOERROR, while they don't provide recursion and also don't provide the A record queried for. There is a warning displayed in the answer of those resolvers when a query is send to them using the Linux command dig:
'WARNING: recursion requested but not available'
This means that the RA flag is not set on the answer, while the RD flag is set on the query, thus the resolver does not support or refuses recursive queries. So the resolver couldn't answer the query but still sets rcode to NOERROR. That is why a validation step is used to distinguish between the resolvers that can answer queries, thus providing recursion, and those who can't.

4.3 Validation of the Full Data Set

To filter out the resolvers that don't perform recursion while the rcode is NOERROR, a validation step is performed. Using Python, a validation query is sent to all the resolvers found to be 'open' by the zmap scanning. This query asks the resolvers to resolve a domain created for these DNS measurements (dnsscan.zkkd.nl) for its A record. This domain is also used in the quantification of the open DNS resolvers in Section 5. Before querying the resolvers, a query is performed with the measurement machine's default DNS resolver to get the correct result of 'dnsscan.zkkd.nl'. This correct A record is compared with the answer of the resolvers to determine which resolvers are capable of performing recursive queries and provide a correct answer. This time the domain 'dnsscan.zkkd.nl' is used, as 'google.com' is not reachable through the great firewall of China [13], but 'dnsscan.zkkd.nl' is. This way, open DNS resolvers behind the great firewall of China can also be validated whether they are able to correctly resolve non-blacklisted domains.

4.4 Results of the Scanning and Validation

The difference in the number of open resolvers and validated resolvers found in the IPv6 address list is shown in Table 4.1. It can be seen that the same difference in open and verified resolvers occur in the whole IPv6 address list, as in the subset. The percentage of the open and verified resolvers found are both higher in the whole IPv6 address list.

hitlist	entries	open resolvers	open ratio	verified resolvers	verified ratio
alexa-country	0	0	0	0	0
caida-dnsnames	6.122	33	0.539040	9	0.147011
ct	844.249	2.840	0.336394	771	0.091324
rapid7-dnsany	3.023.780	12.155	0.401980	4.026	0.133145
traceroute-v6-builtin	101.797	377	0.370345	348	0.341857
traceroute-v6-udm	39.596	163	0.411658	131	0.330841
au	43.597	14	0.032112	5	0.011469
biz	45.511	29	0.063721	8	0.017578
com	383.580	954	0.248710	252	0.065697
czds	177.988	552	0.310133	212	0.119109
de	234.502	170	0.072494	71	0.030277
info	81.759	60	0.073386	22	0.026908
mobi	8.418	5	0.059397	4	0.047517
net	130.540	360	0.275778	129	0.098820
nu	13.296	7	0.052647	4	0.030084
org	112.856	191	0.169242	69	0.061140
se	20.728	12	0.057893	8	0.038595
sk	2.220	3	0.135135	0	0.000000
xxx	1.234	0	0.000000	0	0.000000
surfnet-ns1	142.372	3.745	2.630433	3.155	2.216026
domain-ns-server-AAAA	0	0	0	0	0
all hitlists	3.750.436	15.800	0.421284	7.011	0.186938

TABLE 4.2: Open and validated resolvers of the IPv6 address list, and the ratio between open and validated resolvers and the total entries in the address list

The results from the whole IPv6 address list divided over the hitlist source where the IPv6 address is found in, are shown in Table 4.2. Note that the same IPv6 address can occur in multiple hitlists. The total number of IPv6 addresses is shown as entries. The open resolvers and ratio between open resolvers and entries, and the verified resolvers and ratio between verified resolvers and entries are shown in the table.

The ASN of the verified open resolvers found in the whole IPv6 address list is shown in Figure 4.1. As can be seen from the figure, most of the open resolvers found in the IPv6 address list are from ASN 6939, which is registered to 'HURRICANE - Hurricane Electric, Inc., US', an ISP. Second in in the figure is ASN 63949, which is registered to 'LINODE-AP Linode, LLC, US', a cloud hosting service, which is the ASN with the most open resolvers in the scan on the subset.

In how many hitlists an IPv6 address can be found, is shown in Table 4.3 and in Figure 4.2. The number of hitlists where an individual IPv6 address is found in, is counted. The resulting list is grouped by the number of hitlists where the IPv6 address is found in. So the table specifies how many IPv6 address belong to each group. The figure plots the sorted data with the ip addresses on the x-axis and the amount of hitlists the addresses are found in on the y-axis. The number of hitlists and the amount of IPv6 addresses found in that number of hitlists is shown. The six IPv6 addresses that are found in 15 hitlists, belong to hosting companies where multiple thousand domains are hosted on each IPv6 address.

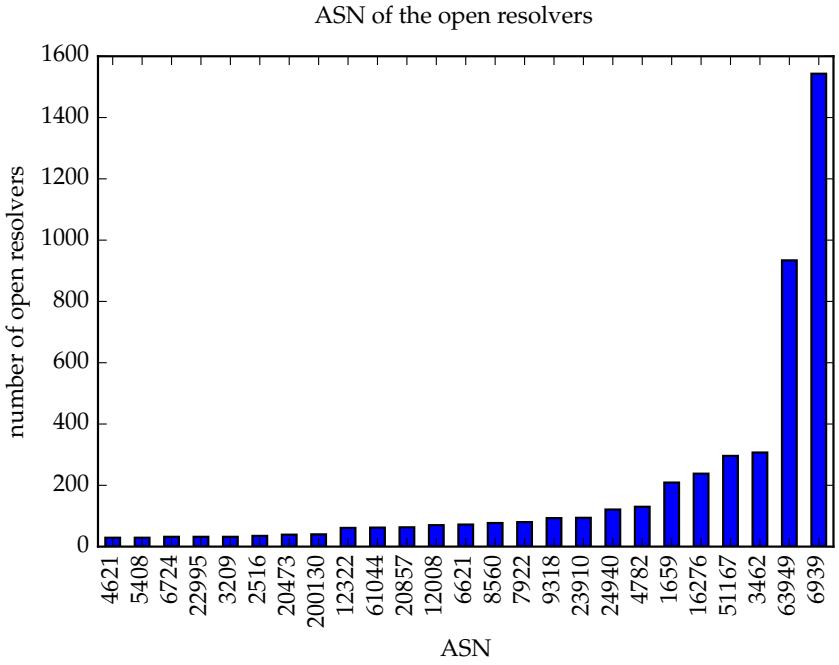

FIGURE 4.1: ASN of the verified open resolvers of the IPv6 address list

number of hitlists	**number of IPv6 addresses**
1	2.921.796
2	518.142
3	169.198
4	42.854
5	21.136
6	12.128
7	7.799
8	11.595
9	19.022
10	16.695
11	7.934
12	1.910
13	202
14	14
15	6

TABLE 4.3: The number of hitlists where the IPv6 addresses are found in

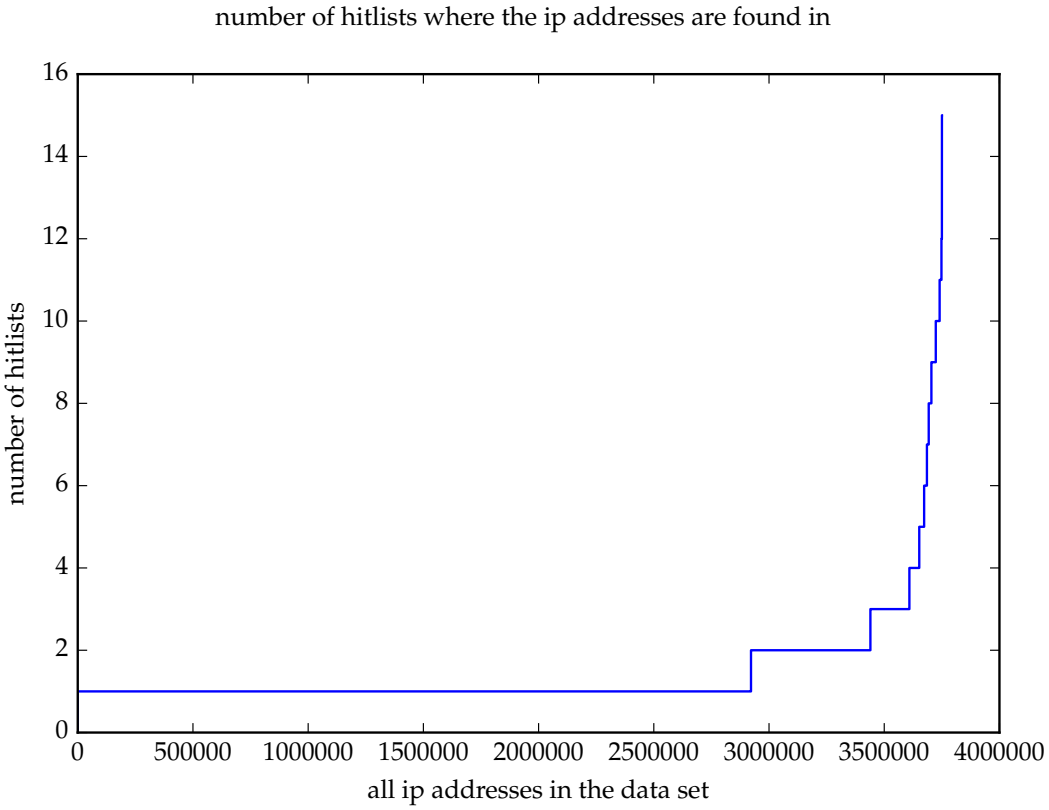

FIGURE 4.2: The number of hitlists where the IPv6 addresses are found in

5 Performance Quantification of Open DNS Resolvers

This chapter describes the quantification of the found open DNS resolvers on IPv6. Section 5.1 presents the classification methods applied to the found open DNS resolvers. In Section 5.2 is discussed how the bandwidth estimation of the resolvers is performed and how usable the results are.

5.1 Classification of Open DNS Resolvers

This section presents classification techniques to classify open DNS resolvers. Section 5.1.1 presents the the method used to get the message size of the dns query and answer, both for successfully resolving resolvers and for those that refuse to answer recursive queries. The amplification factor that can be achieved by the resolvers is presented in Section 5.1.2. In Section 5.1.3 is discussed what IPv6 address types there are and which type the resolvers found have.

5.1.1 Size of DNS Queries and Answers

The open and verified resolvers are queried for a large zone to get the approximate maximum message size that the resolvers can send. The DNS measurement domain (dnsscan.zkkd.nl) is setup with TXT records resulting in an answer of approximately 3.800 bytes. This domain or zone is called 'full zone' throughout the thesis. The same scan results are used for the bandwidth estimation in Section 5.2. The scan is repeated for a smaller zone (dnsscan-small.zkkd.nl) for the resolvers that couldn't provide the answer to the previous query. This smaller domain or zone is called 'small zone' throughout the thesis. The smaller zone is setup with TXT records resulting in an answer of approximately 2.100 bytes. This smaller zone is still large enough to cause packet fragmentation, since the *Maximum Transmission Unit* (MTU) is 1.500 bytes according to the standard [16]. The query and answer size for both the resolvers that successfully answered the query and the resolvers that failed to answer the query is measured. This is done by sending a query to the resolvers using dig and monitoring the output for a successful or failed answer. Parameters are passed to dig to force the use of UDP as transport layer protocol, and truncated answers are not retried using TCP. At the same time tcpdump is started to capture the network traffic going in and out of the measurement machine. The payload size of the packets of fragmented answers is summed, based on source and destination IPv6 address of the packet. The frame length of the packet is used as the payload size, this has the protocol headers of UDP and IPv6 included in the size.

In the measurements, the open DNS resolvers successfully resolving the query, return an answer of at least one record. The measured message size in bytes for the query and answer of the resolvers successfully resolving the query of the full zone and the small zone, is shown in Table 5.1. Some resolvers return an answer that

item	Full zone		Small zone	
	query	**answer**	**query**	**answer**
highest	106	8.502	112	4.842
lowest	106	126	112	136
mean	106	3.923	112	2.177
median	106	4.040	112	2.210

TABLE 5.1: The highest, lowest, mean and median value, in bytes, of the message size of the successful queries

item	Full zone		Small zone	
	query	**answer**	**query**	**answer**
highest	1.400	4.343	1.406	4.238
lowest	106	66	112	66
mean	113	309	116	311
median	106	106	112	112

TABLE 5.2: The highest, lowest, mean and median value, in bytes, of the message size of the failed queries

is twice the size of the zone, about 8.000 bytes for the full zone of 3.800 bytes. The reason is that those resolvers answered the single query twice. This could be because of a flaw in the resolver's software or something happened during the transportation of the query or answer. It is also possible that some resolvers forward the query to another resolver that in turn processes the query and sends the answer back. This means that two front-end resolvers use the same back-end resolver to answer the query, or one resolver forwards the query to a second resolver. Since filtering is used during the data processing, the resolver answering the queries must also exist in one of the IPv6 address sources and also received a query during the scanning. The open DNS resolvers failing the query, return 0 answers and the rcode of SERVFAIL or refuse to preform a recursive query. The measured message size in bytes for the query and answer of the resolvers failing to resolve the query of the full zone and the small zone, is shown in Table 5.2. The reason that the message size is still significant, is that some resolvers return the A, AAAA and NS records of the root nameservers. From the resolvers failing the query for the full zone, 78 return an answer that is 1.000 bytes or larger, for the small zone this is 57 resolvers. There are even 2.172 resolvers that return an answer that is 600 bytes or larger for the full zone, for the small zone this is 2.179 resolvers.

A normal query for the full zone is 106 bytes and the small zone is 112 bytes. The table however, shows that query sizes of 1.400 and 1.406 also occur. The reason is that some answers couldn't be reassembled within the reassembly timeout. This causes the measurement machine processing the answer to send an ICMP Time Exceeded message to the resolver. This ICMP packet is captured by tcpdump and added to the message size of the query.

Figure 5.1 shows a plot of the answer size for all the resolvers mentioned above. The plots of the full and small zone of successful answers are almost the same. The difference is that a small number of resolvers managed to resolve the small zone, but not the full zone. This is probably because the larger payload doesn't fit in the *Extension mechanisms for DNS* (EDNS) options of some resolvers. Those resolvers fail

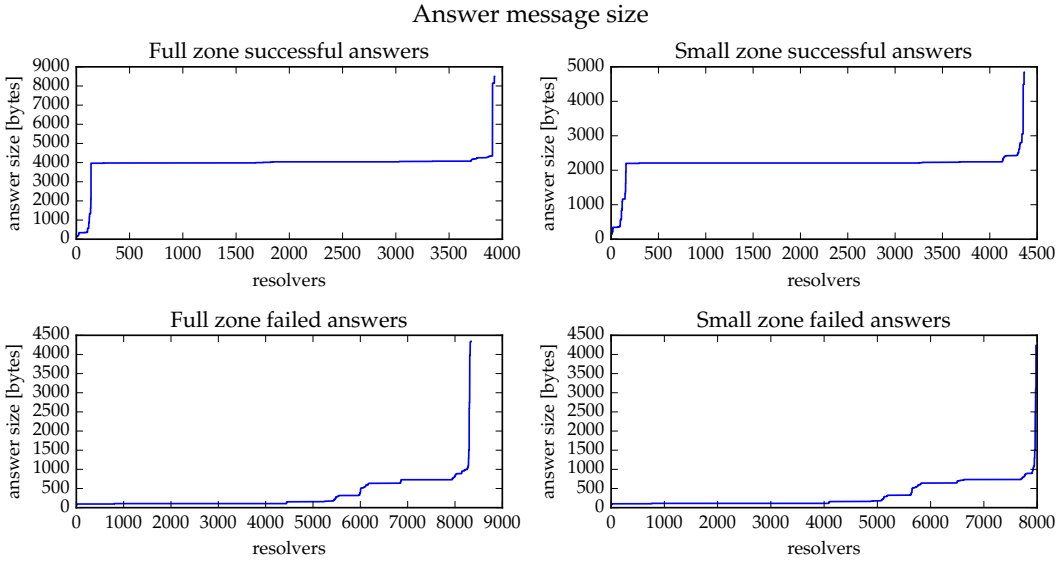

FIGURE 5.1: Answer message size in bytes of the query to the open resolvers of the IPv6 address list

to resolve the query and return a SERVFAIL rcode. The same difference can be seen between the failed answers of the full and small zone. The full zone has a little more failed answers than the small zone has.

Overall the successful answers are relatively constant. There is however, a valley at beginning and a peak at the end of the graph. The valley at the beginning indicates a number of resolvers not able to return all the TXT records requested for, but only a single A or AAAA record. This still sets the number of answer records in the answer, so passes the successful/fail test. The peak at the end of the graph indicates a number of resolvers that responded twice to a single query. That is also why the graph is about twice as high there.

The failed answers are consistent in the beginning. Most of these resolvers don't support recursion, thus reply with an empty answer or a small answer pointing to the root DNS servers. There are also some resolvers that could not answer the query for the large zone, thus replied with a SERVFAIL rcode and a small or empty answer. The increase in message size after that indicates the resolvers that refuse to answer recursive queries, but reply with a relatively large answer with A, AAAA or NS records of the root DNS servers. The peak in message size at the end of the graph has multiple causes. The resolver could actually have resolved the zone, but the answer is received after the timeout of the query, which is 16 seconds, twice as long as zmap uses. The answer is thus added to the tcpdump capture and processed, but ended up in the failed answers. Another cause is that a fragment is lost, or not send by the resolver. This way the packet cannot be reassembled, thus no answer is received. But the answer is added to the tcpdump capture, just like above.

5.1.2 Amplification Factor

With the message size of the query and the answer known, the potential amplification factor that can be achieved with the resolvers can be calculated. This is done for both the full zone and the small zone. Two values can be used as the message size of the query and the answer. The frame length can be used, as it is the length of the message as it is on the wire. And the payload length of the IPv6 packet can

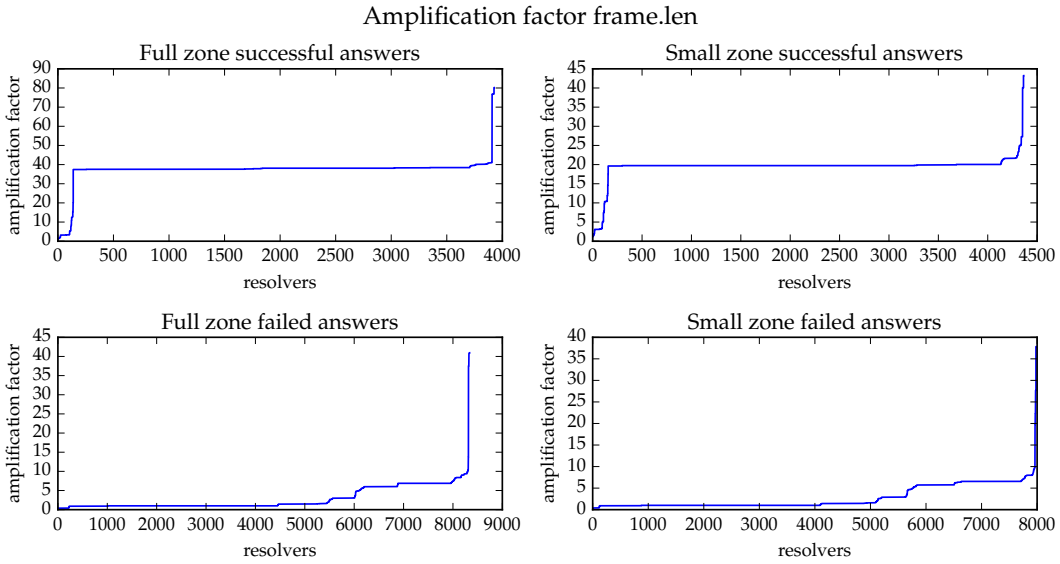

FIGURE 5.2: Amplification factor with the frame length as message size

also be used, as it is the actual length of the data without the Ethernet headers appended to it. The amplification factor will be even greater if the application layer message size is used. The IPv6 and the UDP headers are excluded then. The UDP header takes another 8 bytes on top of the IPv6 header. So the amplification factor for the application layer message can be calculated by subtracting 8 from the query and answer message size. It's assumed that the amplification factor correlates to the message size of the answer of the DNS query that is send to the resolver. This is because the message size of the query itself is the same to all resolvers. It will however, not be entirely the same for the failed resolvers. That is because the measurement machine sending the queries, is sending ICMP messages to some resolvers that sent fragmented answers that could not be reassembled within the reassembly timeout, as is mentioned in 5.1.1. Rate limiting can be applied to DNS resolvers to prevent rapid querying of the resolvers. However, research conducted by MacFarland et al. [24] found that only 149 (2.69%) of studied name servers employed rate limiting. This means that for nearly all the found open DNS resolvers, the amplification factor can be achieved when repeatedly querying the resolvers. This could of course be verified by repeatedly querying all the resolvers, but for ethical reasons this not performed.

The amplification factors of the resolvers with the frame length as message size, are shown in Figure 5.2. The average amplification factor of the resolvers resolving the full zone, with an answer size of approximately 3.800 bytes, is a bit less than 40. For the small zone the amplification factor is a little bit less than 20. The same peak and valley as in the message size plot can be seen. This is because the query size is for every resolver the same, and a smaller answer size corresponds to a smaller amplification factor. The resolvers that send the answer twice have an amplification factor about twice as high. The highest, lowest, mean and median value of the amplification factor is shown in Table 5.3

The payload length of the IPv6 packet can also be used, as it is the actual length of the data without the Ethernet headers appended to it. The amplification factors of the resolvers with the payload length of the IPv6 packet as message size, are shown in Figure 5.3. The average amplification factor of the resolvers resolving the full

item	Successful		Failed	
	full zone	**small zone**	**full zone**	**small zone**
highest	80.2	43.2	41.0	37.8
lowest	1.2	1.2	0.4	0.4
mean	37.0	19.4	2.8	2.8
median	38.1	19.7	1.0	1.0

TABLE 5.3: The highest, lowest, mean and median value of the amplification factor, with frame-len as message size

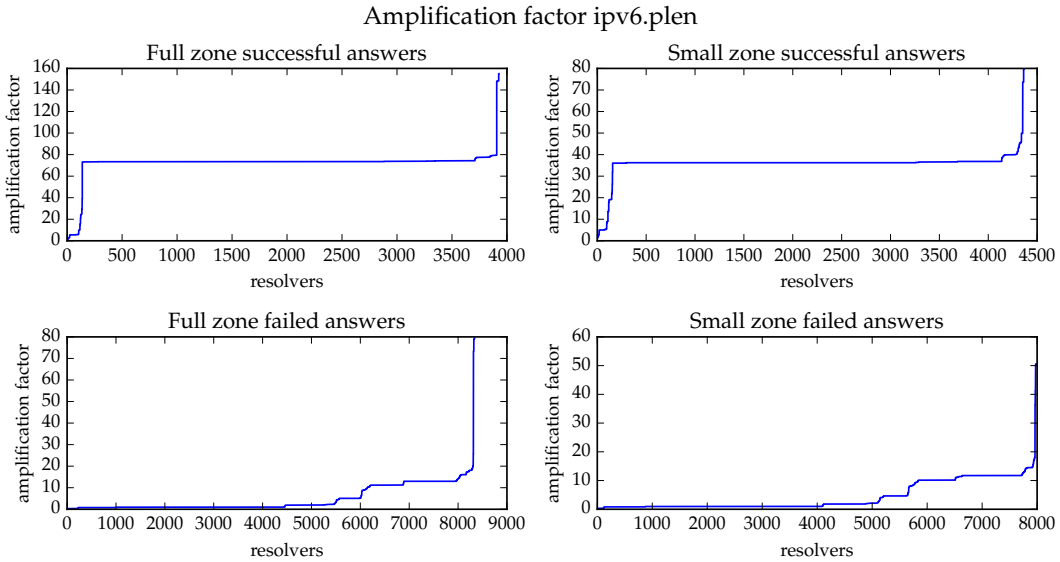

FIGURE 5.3: Amplification factor with the IPv6 payload size as message size

zone, with an answer size of approximately 3.800 bytes, is a bit less than 75. For the small zone the amplification factor is a little bit more than 35. The same peak and valley as in the message size plot can be seen. This is because the query size is the same for every resolver, and a smaller answer size corresponds to a smaller amplification factor. The resolvers that send the answer twice have an amplification factor about twice as high. The highest, lowest, mean and median value of the amplification factor is shown in Table 5.4

For the complete picture, the amplification factor calculated with the message size of the packets at the application layer, is shown in Table 5.5. This results in the highest amplification factor since all the UDP and IPv6 headers aren't in the calculation. The resulting amplification will not be achieved, since the packet headers mentioned above are send on the wire. But this amplification factor is achieved when looking only at the DNS protocol information.

5.1.3 Types of Resolver Addresses

The IPv6 addresses of the resolvers are divided in three address types. The first address type is SLAAC, which is the network prefix appended with an IID generated from the MAC of the machine. The IID of the IPv6 address is generated as follows The machine's MAC address is split in half and the value 'ff:fe' is placed between

item	Successful		Failed	
	full zone	small zone	full zone	small zone
highest	155.2	79.8	79.4	50.6
lowest	1.4	1.4	0.2	0.2
mean	71.8	35.7	4.8	4.4
median	73.5	36.2	1.0	1.0

TABLE 5.4: The highest, lowest, mean and median value of the amplification factor, with ipv6-plen as message size

item	Successful		Failed	
	full zone	small zone	full zone	small zone
highest	183.2	92.4	93.6	58.5
lowest	1.5	1.5	0.1	0.1
mean	84.7	41.3	5.4	4.9
median	86.7	41.9	1.0	1.0

TABLE 5.5: The highest, lowest, mean and median value of the amplification factor, with the message size of the packets at the application layer

the two halves of the MAC address. Then the 7th bit of the IID is complemented (i.e., 'flipped'). Thus the resulting address consists of the network prefix and the IID. For example, the MAC address '12:34:56:ab:cd:ef' and the network prefix '2a02::' results in the IPv6 address: '2a02::1034:56ff:feab:cdef'.

The second type of IPv6 is a 'seemingly random address'. An IPv6 address is classified as seemingly random if the IID part of the IPv6 address has a hamming weight (number of bits set to '1' in the IPv6 address) of 16 or less. The average hamming weight of an IID of an IPv6 address is 32, since the length of the IID is 64 bits. By looking at the hamming weight, the second address type can contain addresses from SLAAC with privacy extensions, manually configured addresses with at least 16 bits set to '1' or addresses created in a different way. SLAAC with privacy extensions randomizes the values after the network prefix [28], the IID. The address randomization can be achieved using different techniques, divided in two cases based on whether stable storage is present. When stable storage is present, the randomized IID is created using a number of steps. First the history value from the previous iteration is used if it's available, or a random value is generated. Then the MD5 message digest is computed over the random value. The leftmost 64-bits of the MD5 digest is used, and 7th bit of that value is set to zero. This creates an interface identifier with the universal/local bit indicating local significance only. The rightmost 64-bits of the MD5 digest are stored in the history value to be used for the next iteration. In the absence of stable storage, a random number is generated every iteration. An alternate approach is *Cryptographically Generated Addresses* (CGA), which generate a random IID based on the public key of the device. In case DHCP is used to generate a random looking address, the address is categorized in this address type.

The third address type is the type that is probably manually assigned to the machine. This address type is classified as having a hamming weight of the IID of at most 16, thus having many zeros in the IID of the IPv6 address. These addresses are mostly setup by administrators to be easily rememberable or that the function of the

address type	amount of resolvers	percentage
SLAAC	1.288	10.4
Seemingly random (hw > 16)	1.227	9.9
Manual (hw <= 16)	9.829	79.6
Total	12.344	100

TABLE 5.6: Open and verified resolvers: IPv6 address types where 'hw' means the hamming weight of the interface identifier part of the IPv6 address

non-zero values	amount of resolvers	percentage
Only in last hextet	13.429	68.3
Also other hextets	6.224	31.7
Total Manual addresses	19.653	100

TABLE 5.7: Open and verified resolvers: non-zero values in the last hextet of the IPv6 address or also in other hextets

machine is encoded in its IPv6 address, like '::53' for a DNS resolver.

For all the open and verified resolvers, the IPv6 address type is shown in Table 5.6. As can be seen from the table, most of the resolvers have manually configured IPv6 addresses. Which is interesting, since these addresses are manually set up by administrators for a specific purpose, in this case DNS resolving. For the manual assigned IPv6 addresses, whether only the last hextet of the IPv6 address contains non-zero values, or also other hextets, is shown in Table 5.7. The hamming weight of the IIDs of the open and verified resolvers is shown in Figure 5.4. The blue line represents the hamming weight of the IIDs of the IPv6 addresses. The red line represents the hamming weight of randomly generated IIDs, where there are generated as many IIDs as there are IIDs represented by the blue line. As can be seen, the normal hamming weight should be around 32, where the IID has a length of 64 bits. The blue line indicates that the majority of the addresses have more zeros than ones. Even though the SLAAC addresses have 'fffe' in the IID, which has a hamming weight of 15 out of the 16 bits.

The same classification techniques are applied to the verified resolvers that successfully resolved the full and small zone. The IPv6 address types of the resolvers is shown in Table 5.8. And for the manual assigned IPv6 addresses, whether only the last hextet of the IPv6 address contains non-zero values, or also other hextets, is shown in Table 5.9. The hamming weight of the IID of the verified resolvers that successfully resolved the full zone is shown in Figure 5.5. And for the small zone, this is shown in Figure 5.6. The address types of the resolvers that successfully resolved the full and small zone are about the same as the open and verified resolvers. There is only a small difference in the non-zero values where the open and verified resolvers have a higher percentage of resolvers with non-zero values only in the last hextet. The same hamming weight distribution as the open and verified resolvers have, can be seen in the resolvers that successfully resolved the full and small zone.

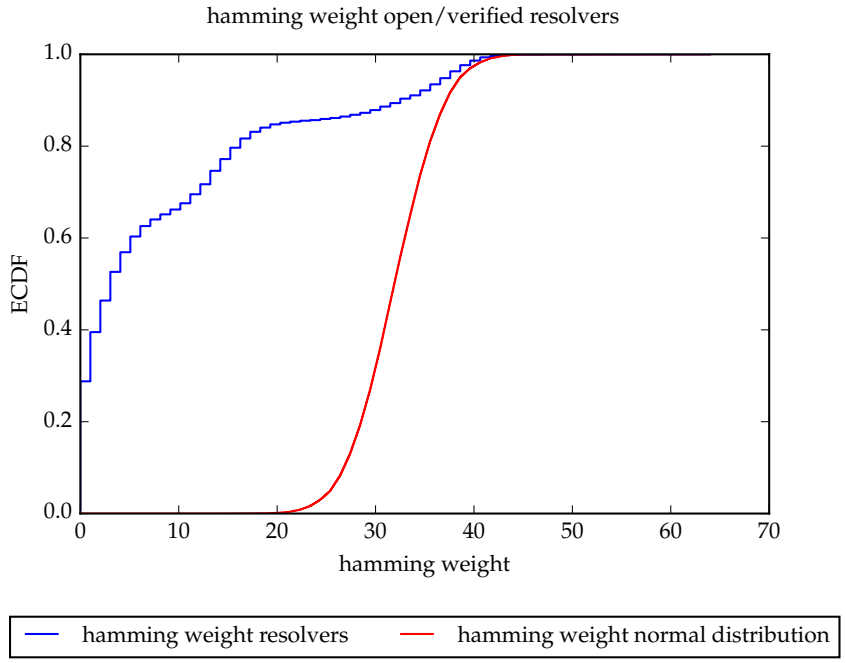

FIGURE 5.4: Hamming weight of the IIDs of the open and verified resolvers represented by the blue line, and the hamming weight of equally many random generated IIDs represented by the red line

address type	amount of resolvers			
	full zone	percentage	small zone	percentage
SLAAC	495	12.6	529	12.1
Seemingly random (hw > 16)	625	15.9	615	14.1
Manual (hw <= 16)	2.806	71.5	3.220	73.8
Total	3.926	100	4.364	100

TABLE 5.8: Resolvers able to resolve the full and small zone: IPv6 address types where 'hw' means the hamming weight of the interface identifier part of the IPv6 address

non-zero values	amount of resolvers			
	full zone	percentage	small zone	percentage
Only in last hextet	1.831	65.3	1.950	60.6
Also other hextets	975	34.7	1.270	39.4
Total Manual addresses	2.806	100	3.220	100

TABLE 5.9: Resolvers able to resolve the full and small zone: non-zero values in the last hextet of the IPv6 address or also in other hextets

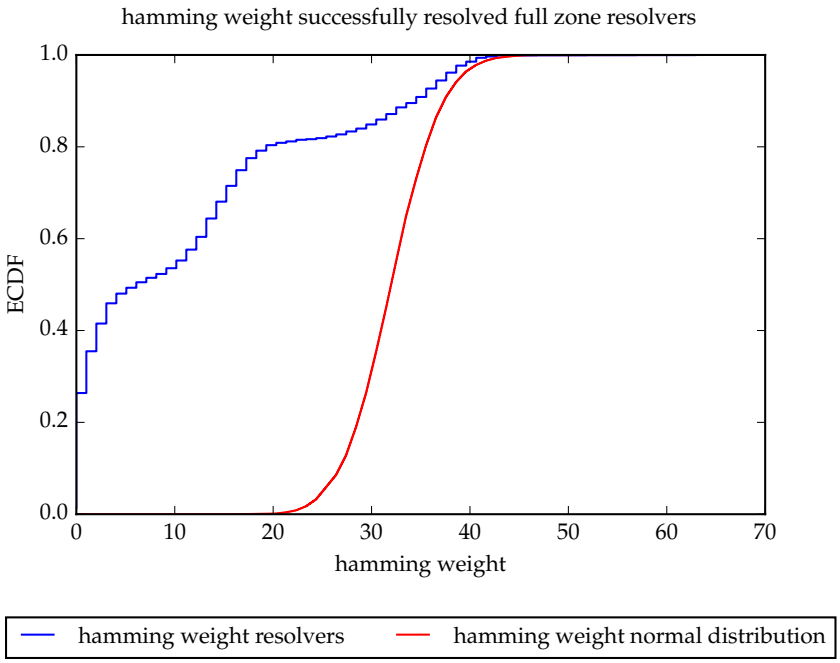

FIGURE 5.5: Hamming weight of the IIDs of the resolvers successfully resolving the full zone represented by the blue line, and the hamming weight of equally many random generated IIDs represented by the red line

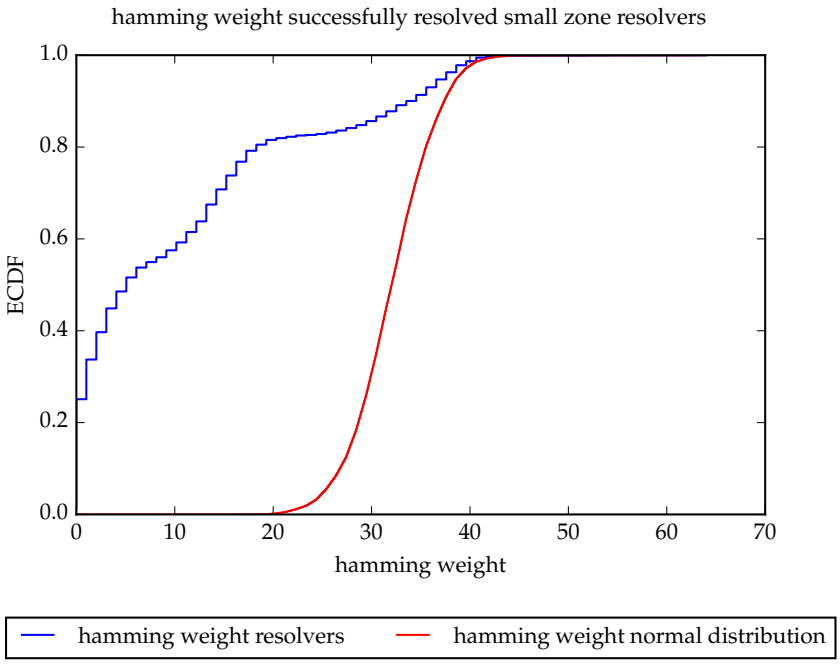

FIGURE 5.6: Hamming weight of the IIDs of the resolvers successfully resolving the small zone represented by the blue line, and the hamming weight of equally many random generated IIDs represented by the red line

5.2 Bandwidth Estimation

This section presents a method to estimate the bandwidth of open DNS resolvers. Section 5.2.1 presents the approach to estimate the bandwidth of open DNS resolvers by using packet fragmentation of the answer of the query. The results of the bandwidth measurement approach and its usability is presented in Section 5.2.2.

5.2.1 Bandwidth Estimation Approach

The approach to the bandwidth estimation of open DNS resolvers on IPv6 is by using timestamp values of fragmented IPv6 packets. The query for the full zone, mentioned before and resulting in an answer around 3.800 bytes, is used to generate large answers. The answers are fragmented since the answers to the DNS queries are larger than the maximum MTU of 1.500 bytes [16] of a typical Ethernet network. Important is that the firewall is configured correctly for fragmented packets to pass through. The DNS query is send using UDP. The UDP header contains the source and destination port information of the open DNS resolver. Since the UDP packet is split in multiple fragments, the UDP header with the port information is only present in the first fragment. The other fragments only contain the IPv6 header with the IPv6 source and destination address. When fragmented packets arrive at the firewall, rules accepting packets on port numbers can't be used, since the fragments (except the first one with the UDP header) will be dropped. Tcpdump is used on the measurement machine to capture the fragmented packets as they arrive at the measurement machine. At the same time and for comparison the same tcpdump is started on dedicated hardware for high speed network monitoring and capturing. The measurement machine is connected to the internet via the high speed capturing machine. The extra tcpdump is included to measure differences in packet arrival time on the high speed network capturing machine, and a normal VM running Debian. The measurement machine is connected to a 10 Gigabit internet connection at the University of Twente, so there is sufficient bandwidth to perform the bandwidth measurements on the found open DNS resolvers. To calculate the achievable bandwidth of the found open DNS resolvers, the timestamp of the packet and the size of it is used. By dividing the packet size over the inter-arrival time of the packets, the bandwidth can be calculated [3]. In other words, the dispersion of the packets that are sent by the open DNS resolver is calculated.

5.2.2 Bandwidth Estimation Results

The measurement is performed on two test resolvers. One resolver running at home on a 100 Megabit connection and a resolver of my supervisor on a 1 Gigabit connection. This means that the result of the test should be around 100 Megabits per second and 1 Gigabit per second. The measurement consists of 250 consecutive queries to the resolver and monitoring the packets that are received. 250 consecutive queries are send to make sure that some answers will arrive without queuing delays between the fragments of the packets. The same bandwidth calculation as mentioned in Section 5.2.1 is used for the tcpdump output of the measurement machine and the high speed network capturing machine. The calculated bandwidth using the measurement machine of the resolver running at home is shown in Figure 5.7. For the high speed network capturing machine, the calculated bandwidth is shown in Figure 5.8. As can be seen in the figure, the measured bandwidth in the graph is

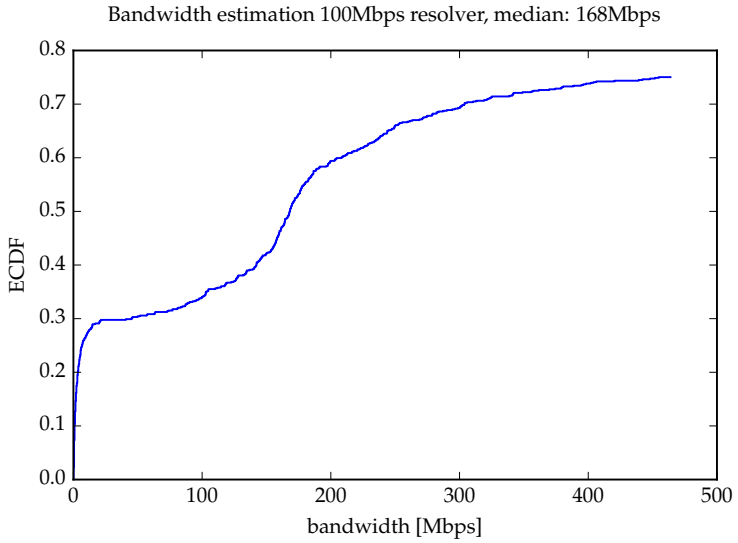

FIGURE 5.7: Bandwidth measurement of the 100 Megabit resolver using the measurement machine

very spread out. The median is used to indicate the result of the bandwidth measurement. For the resolver running at home, the result is around 150 Megabits per second, which is 1.5 times the actual speed of the resolver.

The bandwidth is then calculated for the 1 Gigabit resolver. The calculated bandwidth using the measurement machine is shown in Figure 5.9. For the high speed network capturing machine, the calculated bandwidth is shown in Figure 5.10. Again for this bandwidth calculation, the median is used to indicate the result of the bandwidth measurement. As can be seen in the figure, the result is around 100 Megabits per second, which is 0.1 times the actual speed of the resolver.

This proves not to a good methodology to measure the bandwidth of the found open DNS resolvers. The discussion of the results and the methodology of the bandwidth estimation is presented in Section 6.

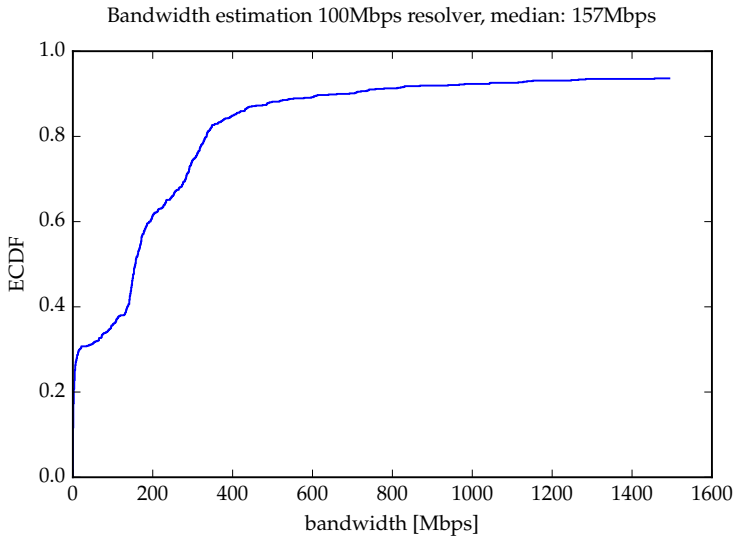

FIGURE 5.8: Bandwidth measurement of the 100 Megabit resolver using the high speed network capturing machine

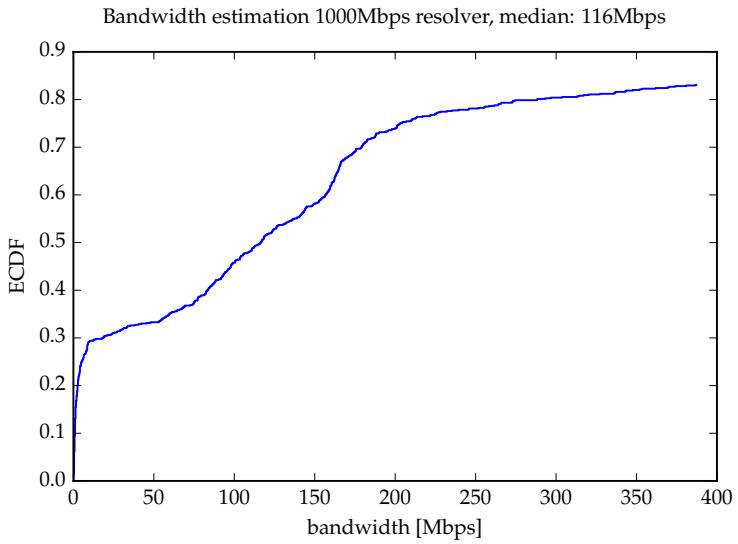

FIGURE 5.9: Bandwidth measurement of the 1 Gigabit resolver using the measurement machine

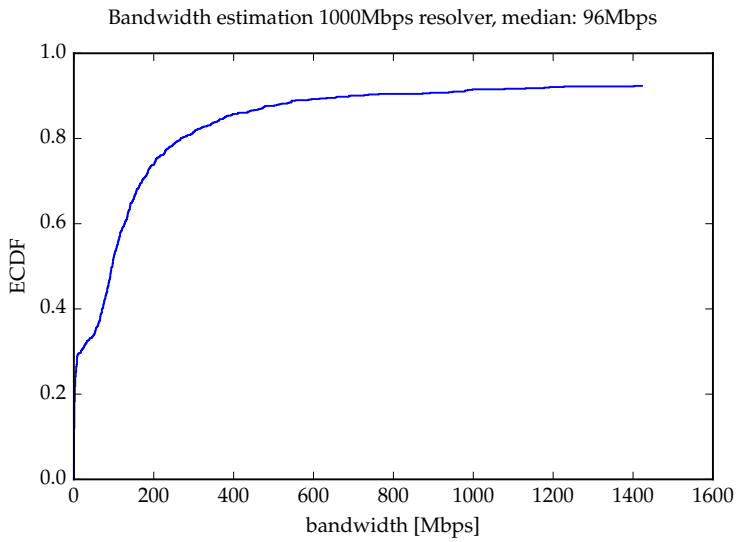

FIGURE 5.10: Bandwidth measurement of the 1 Gigabit resolver using the high speed network capturing machine

6 Discussion

Section 6.1 presents the ethical considerations for the results of this research and the measurements that are performed. Section 6.2 presents the discussion about the results of the bandwidth estimation and also mentions alternatives.

6.1 Ethical Considerations

Presenting a methodology and results that can be misused for malicious intents might raise ethical concerns. The methodology presented, however, is already present for IPv4 in a sense. This research broadens the scope to IPv6. The results can be used to alert network operators or individuals that their open DNS resolvers can be misused, and prevent malicious use of the open DNS resolvers.

Several considerations are taken into account, while performing the measurements for this research, from the ethical point of view. First of all, a website is hosted on the measurement machines that are used, to indicate what measurements are performed and why. Also an option to opt-out or blacklist an IPv6 address range from future scanning is available on the website. During the research, a single request for blacklisting was received, and the address range is blacklisted and didn't receive any more traffic from the measurements. The query that is used for the quantification to get a large answer, asks the resolvers for TXT record of the domain. The TXT records contain text strings with information about the measurements and how to opt-out or blacklist from the measurements, about the same way as it is presented on the website.

The ASN lookup is performed to classify to which ASN the found open DNS resolvers belong. This is performed offline by downloading and using a recent BGP RIB database. This database contains the ASNs for the IPv6 subnets. By performing the lookup offline, a lot of bandwidth and processing power of online services like team Cymru's IP to ASN mapping, is saved.

To get preliminary results, the measurements techniques are first performed on a subset of all the collected IPv6 addresses. This is to reduce the network load in case the result of the techniques prove to be negligible. If that's the case, the scan of all the IPv6 addresses will not be performed for the specific measurement technique.

The short time network load during scans on all the IPv6 addresses is reduced by rate limiting zmap. When no rate limiting is applied, zmap can scan as fast as the network where the measurement machine is connected to, can handle. This assumes that sufficient processing power is available to process and store the results of the scan. For the measurement machine used, the network load can be as high as a couple of Gigabits per second. Rate limiting reduces the amount of queries that are sent out at once, but as a result, increases the scanning time.

The verification query and other measurements are only sent to the open resolvers found by zmap. So hosts that didn't respond zmap's scan, or hosts that don't run DNS servers don't receive any more queries or other network traffic from the measurements. This is once again to reduce the network bandwidth and processing power of others, required to respond to the measurements.

After the scan or verification is performed, the results, which are lists of IPv6 addresses, are written to disk. When the results are needed later, they are simple read from disk instead of performing the scan or verification again. At the end, the full scan is performed once, and those results are used for all the quantification techniques to have the most recent data and results as possible. This is also to reduce the time between the time that the open DNS resolvers are found and verified, and the time that the quantification is performed. Since some hosts could already be offline or stopped the open DNS resolver, thus influencing the results.

While implementing the measurements and classifications, the code is first tested offline or as a dry run if that is possible. Then verified on the University of Twente's own IP address or the resolver running at home. When a larger test result is needed, a small subset of IPv6 addresses is used. This subset is changed after every measurement to prevent the same subset from receiving a lot of measurement traffic.

6.2 Bandwidth Estimation

The proposed bandwidth measurement method uses the difference in timestamp value between the fragments of the packet and the packet size itself. In order to get more reliable results, the measurement is repeated 250 times and average values can be calculated. The reason that this method fails could be queuing delays in the path between resolver and measurement machine. Another reason could be that other traffic is influencing the measurement, since the measurement is performed from the network of the University of Twente. A script that uses the Linux command 'dig' is used to perform the 250 DNS queries. A different approach by performing the queries from Python or another tool could result in a better estimation of the resolver's bandwidth. Also less ethical measurement techniques can be used to estimate the bandwidth, like sending a large burst of traffic and try to flood the network link between the measurement machine and the resolver. This way the capacity of the network during a flood can be measured and thus the bandwidth of the resolver. More literature research and different bandwidth measurement techniques could result in a better estimation of the bandwidth that the open DNS resolvers can achieve.

7 Conclusions and Future Work

The conclusion refers back to the research questions at the beginning of the thesis in Section 1.3. The research questions regarding the discovery of open DNS resolvers are covered in Section 7.1. The research questions regarding the quantification of open DNS resolvers are covered in Section 7.2. The future work on the discovery and quantification of open DNS resolvers is presented in Section 7.3.

7.1 Discovery of Open DNS Resolvers

The discovery of open DNS resolvers is presented in Chapter 4. The conclusion of how and which active and passive measurements can be used to find open DNS resolvers is covered in Section 7.1.1. The conclusion of how and which network traffic can be analyzed to find open DNS resolvers is covered in Section 7.1.2.

7.1.1 Active and Passive Measurements

The active, passive and traceroute measurements to find IPv6 addresses that can be scanned for open DNS resolvers are shown in Section 3.1.1 and Section 4.1. The IPv6 address sources are from the research by Gasser et al. [10]. The hitlists used as IPv6 address sources contain IPv6 addresses from DNS zone files from various TLDs, domain names resolved for AAAA records, AAAA records for entries in internet topology traces, Rapid7's DNS query results filtered for IPv6 addresses and traceroutes to all the sources mentioned. Scans to IPv6 addresses collected from these sources are presented in Section 3.1.4 and Section 4.2. The validation scan to the DNS resolvers classified as 'open' by zmap is mentioned in Section 3.1.5 and Section 4.3. The results of the scanning and validation of the IPv6 addresses is presented in Section 4.4. The open and validated DNS resolvers are shown in Table 4.1. From the hitlist sources (alaxa-country till xxx in the table) the caida-dnsnames is the most promising open DNS resolver source with 0.54% of the IPv6 addresses hosting an open DNS resolver, followed by traceroute-v6-udm with 0.41% and rapid7-dnsany with 0.40% of the IPv6 addresses hosting an open DNS resolver. The IPv6 address sources have respectively 6.122, 39.596 and 3.023.780 unique entries. This means that AAAA records for internet topology traces, traceroutes to the found addresses and the AAAA records of DNS any queries provide the highest number of IPv6 addresses of the sources that have been researched. But the rapid7-dnsany source has by far the most entries, so this source is selected as best IPv6 address source.

The research question 'How can active or passive measurements be used to find open DNS resolvers?' can be answered with hitlists containing AAAA records of DNS any queries.

7.1.2 Network Traffic Analysis

The network traffic that can be analyzed to find IPv6 addresses that can be scanned for open DNS resolvers is shown in Section 3.1.2 and Section 4.1. This network traffic consists of incoming authoritative nameserver traffic containing only DNS traffic which is send between DNS resolvers and the nameserver. Network traffic data of the ns1 authoritative nameserver at SURFnet [34] is used to collect IPv6 addresses. Scans to IPv6 addresses collected from the network traffic are presented in Section 3.1.4 and Section 4.2. The validation scan to the DNS resolvers classified as open by zmap is mentioned in Section 3.1.5 and Section 4.3. The results of the scanning and validation of the IPv6 addresses is presented in Section 4.4. The open and validated DNS resolvers are shown in Table 4.1. From the surfnet-ns1 IPv6 address source, 2.63% of the scanned addresses is hosting an open DNS resolver. The authoritative nameserver traffic has 142.372 unique entries of which 3.745 are open DNS resolvers. It was expected that this address source contains a higher percentage of open DNS resolvers, since all the IPv6 addresses from this source are DNS resolvers. The active, passive and traceroute sources contain IPv6 addresses from other measurements. The reason why only one network traffic source is researched, is mentioned in Section 7.3.

The research question 'How and what network traffic can be analyzed to find open DNS resolvers?' can be answered with network traffic of the authoritative nameserver at SURFnet.

7.2 Quantification of Open DNS Resolvers

The quantification of the open DNS resolvers is presented in Chapter 5. The conclusion of the methodology to estimate the bandwidth of open DNS resolvers is covered in Section 7.2.1. The conclusion of the performance estimation of open DNS resolvers based on their IPv6 addresses is covered in Section 7.2.2.

7.2.1 Bandwidth Measurement

When the bandwidth of the open DNS resolvers can be measured, the most powerful open DNS resolvers can be distinguished from the less powerful ones. The related work on the bandwidth measurement is presented in Section 2.2.1. The methodology used to try and measure the bandwidth of open DNS resolvers without having control on both sides of the measurement is mentioned in Section 3.2.1. The methodology builds on the two packet bandwidth measurement in the related work chapter. The measurement consists of sending 250 queries, one after the other, to the resolver which bandwidth is measured. With packet capturing software running on the machine conducting the measurement the packet inter-arrival time and packet size are known, and together could be used to estimate the bandwidth of the resolver. This methodology is tested on two resolvers with a known bandwidth limited by the network speed assigned by the ISP. Unfortunately, the methodology was unable to correctly estimate the bandwidth of the two resolvers that the methodology is tested on. The bandwidth of the resolver on a 100 Megabits per second network is estimated as roughly 150 Megabits per second, and the resolver on a 1 Gigabit per second network is estimated as roughly 100 Megabits per second. The estimated bandwidth does not correlate with the actual bandwidth of the resolver.

The research question 'How can we measure the bandwidth without performing a (D)DoS?' can thus not be answered with the methodology used in the thesis. Further research on bandwidth estimation methods is needed to correctly answer this question. The discussion on the bandwidth estimation is shown in Section 6.2.

7.2.2 Performance Estimation Based on IP Address

The related work on fingerprinting and classification of open DNS resolvers is presented in Section 2.2.2. The classification techniques for open DNS resolvers are presented in Section 3.2.2. Two techniques to estimate the performance of the open DNS resolver are researched in Section 5.1. The message size that the resolvers can be queried for is presented in Section 5.1.1 and the resulting amplification factor that can be achieved with the query, in Section 5.1.2. The types of resolver addresses of the found open DNS resolvers is mentioned in Section 5.1.3. The terms 'full zone' and 'small zone' are described in Section 5.1.1. Of the 15.800 open DNS resolvers, 24.0% (3.787) of the resolvers responded to the query for the full zone with an answer larger than 3.800 bytes, resulting in an amplification factor of at least 37.0 with the frame size as message size, and 84.7 with the IPv6 payload size as message size. For the small zone, 26.6% (4.206) of the resolvers responded to the query with an answer larger than 2.100 bytes, resulting in an amplification factor of at least 19.4 with the frame size as message size, and 41.3 with the IPv6 payload as message size. These resolvers, able to resolve the full and small zone, thus have a high performance in terms of the response size and amplification factor that they can achieve.

For the open and verified resolvers, 10.4% has a SLAAC address, 9.9% a seemingly random address with the hamming weight of the IID higher than 16 and 79.6% a manual assigned address with the hamming weight of the IID lower than or equal to 16. Of the manual assigned addresses, 68.3% has non-zero values only in the last hextet of the IID, 31.7% also in other hextets. For the resolvers able to resolve the full zone, the percentages are 12.6% a SLAAC address, 15.9% a seemingly random address and 71.5% a manual assigned address. Of the manual assigned addresses, 65.3% has non-zero values only in the last hextet of the IID, 34.7% also in other hextets. For the resolvers able to resolve the small zone, the percentages are 12.1% a SLAAC address, 14.1% a seemingly random address and 73.8% a manual assigned address. Of the manual assigned addresses, 60.6% has non-zero values only in the last hextet of the IID, 39.4% also in other hextets. More than 70% of the resolvers have a manually assigned address, based on the hamming weight of the IID. More than 60% of the resolvers with manual assigned addresses have non-zero values only in the last hextet of the IID. This means that the IPv6 addresses of most of the found open DNS resolvers are manually set up by their administrators.

The ASNs with the most open DNS resolvers are ASN 6939, an ISP and hosting provider, and ASN 63949, a cloud hosting service. Based on the ASN the open DNS resolvers on the hosting provider and cloud hosting service network are assumed to be virtual hosts with manually installed or provided DNS servers. These are often connected to a fast network connection, however, often with a limited amount of data that can be transferred over the network per month.

The research question 'How can the IP address of the open DNS resolver be used to estimate its performance?' can be answered with the conclusions of the message size, amplification factor, address type and ASN mentioned above.

7.3 Future Work

New research in the direction of network traffic sources like SURFnet's ns1 authoritative nameserver can be conducted to find out what other network traffic sources are a good way of providing IPv6 addresses to be scanned for open DNS resolvers. Only the SURFnet ns1 authoritative nameserver source is researched in this thesis. This is because the network traffic data of the ns1 authoritative nameserver was readily available to be used for this research.

The hosting providers, and especially those that provide DNS services could be scanned or tested to find out if the DNS resolvers are configured to be open to the internet. Or that they can easily be misconfigured as an open DNS resolver. The users of the hosting providers should also be informed about the potential damage an open DNS resolver can inflict.

The measurements in this thesis could be run as a continuous service, like the IPv4 counterpart, the open resolver project [30]. The most recent version of the hitlists are already retrieved by a Python script. The SURFnet authoritative nameserver monitor is running continuously to extract IPv6 addresses from the DNS traffic. All the processing done to the data and scanning of the IPv6 addresses is also handled by Python scripts, and all the code can run in a single execution. So the measurements can be repeated daily to get fresh results and the changes in amount or source of open DNS resolvers over time. The results can be shown on a web server and be made that end users can query the results for IPv6 subnets or individual IPv6 addresses.

Bibliography

[1] E. Balkanli, J. Alves, and A. Nur Zincir-Heywood. "Supervised Learning to Detect DDoS Attacks". In: (2014). URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7013367.

[2] Daniel Cheung. *Common DNS Request Types - OpenDNS*. 2017. URL: https://support.opendns.com/hc/en-us/articles/227986607-Common-DNS-Request-Types (visited on 07/04/2017).

[3] C. Dovrolis, P. Ramanathan, and D Moore. "Packet dispersion techniques and a capacity estimation methodology". In: (2004). URL: https://www.caida.org/publications/papers/2004/ton_dispersion/ton_dispersion.pdf.

[4] Z. Durumeric et al. "A Search Engine Backed by Internet-Wide Scanning". In: *CCS '15 Proceedings of the 22nd ACM Conference on Computer and Communications Security* (2015), pp. 542–553. URL: https://censys.io/static/censys.pdf.

[5] Zakir Durumeric. *Home · zmap/zmap Wiki*. 2017. URL: https://github.com/zmap/zmap/wiki (visited on 07/05/2017).

[6] *EEMO - The Extensible Ethernet MOnitor*. 2017. URL: https://github.com/SURFnet/eemo (visited on 01/18/2018).

[7] K. Fujiwara, A. Sato, and K. Yoshida. "DNS traffic analysis – Issues of IPv6 and CDN". In: *2012 IEEE/IPSJ 12th International Symposium on Applications and the Internet* (2012), pp. 129–137. URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6305271.

[8] Y. Gao et al. "A Machine Learning Based Approach for Detecting DRDoS Attacks and Its Performance Evaluation". In: *2016 11th Asia Joint Conference on Information Security* (2016), pp. 80–86. URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7782062.

[9] O. Gasser et al. "Scanning the IPv6 Internet: Towards a Comprehensive Hitlist". In: (July 2016). URL: https://arxiv.org/pdf/1607.05179v1.pdf.

[10] Oliver Gasser et al. "Scanning the IPv6 Internet: Towards a Comprehensive Hitlist". In: *Proc. of 8th Int. Workshop on Traffic Monitoring and Analysis*. Louvain-la-Neuve, Belgium, Apr. 2016.

[11] F. Gont. *RFC7707: Network Reconnaissance in IPv6 Networks*. Mar. 2016. URL: https://www.ietf.org/rfc/rfc7707.txt (visited on 07/05/2017).

[12] Google. *IPv6 - Google*. URL: https://www.google.com/intl/en/ipv6/statistics.html (visited on 06/12/2017).

[13] *Great Firewall of China*. URL: http://www.greatfirewallofchina.org/index.php?siteurl=google.com (visited on 11/21/2017).

[14] L. Hendriks et al. "On the potential of IPv6 open resolvers for DDoS attacks". In: *PAM* (2017), pp. 17–29. URL: http://www.springer.com/cda/content/document/cda_downloaddocument/9783319543277-c2.pdf.

[15] P. Hoffman, A. Sullivan, and K. Fujiwara. *RFC7719: DNS Terminology*. Dec. 2015. URL: https://www.ietf.org/rfc/rfc7719.txt (visited on 06/12/2017).

[16] C. Hornig. *RFC894: A Standard for the Transmission of IP Datagrams over Ethernet Networks*. Apr. 1984. URL: https://tools.ietf.org/rfc/rfc894.tx (visited on 01/18/2018).

[17] V. Shi-Ming Huang, R. Huang, and M. Chiang. "A DDoS Mitigation System with Multi-Stage Detection and Text-Based Turing Testing in Cloud Computing". In: *2013 27th International Conference on Advanced Information Networking and Applications Workshops* (2013), pp. 655–662. URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6550471.

[18] *Index of routeviews*. URL: ftp://archive.routeviews.org/routeviews6/bgpdata/ (visited on 11/27/2017).

[19] A. Sherin Jose and B. A. "Automatic Detection and Rectification of DNS Reflection Amplification Attacks with Hadoop MapReduce and Chukwa". In: *2014 Fourth International Conference on Advances in Computing and Communications* (2014), pp. 195–198. URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6906023.

[20] V. Jyothi et al. "BRAIN: BehavioR based Adaptive Intrusion detection in Networks: Using Hardware Performance Counters to detect DDoS Attacks". In: *2016 29th International Conference on VLSI Design and 2016 15th International Conference on Embedded Systems* (2016), pp. 587–588. URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7435029.

[21] S. Kang et al. "Packet-Pair Bandwidth Estimation: Stochastic Analysis of a Single Congested Node". In: *Proceedings of the 12th IEEE International Conference on Network Protocols (ICNP'04)* (2004), pp. 316–325. URL: http://www.ieee-icnp.org/2004/papers/8-3.pdf.

[22] M. Kührer et al. "Going Wild: Large-Scale Classification of Open DNS Resolvers". In: *IMC '15 Proceedings of the 2015 Internet Measurement Conference* (2015), pp. 355–368. URL: http://www.christian-rossow.de/publications/goingwild-imc2015.pdf.

[23] C. Liu et al. "Detect the Reflection Amplification Attack Based on UDP Protocol". In: *2015 10th International Conference on Communications and Networking in China (ChinaCom)* (2015), pp. 260–265. URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7497948.

[24] D. MacFarland, C Shue, and A Kalafut. "Characterizing Optimal DNS Amplification Attacks and Effective Mitigation". In: *PAM* (2015), pp. 15–27. URL: https://link.springer.com/content/pdf/10.1007/978-3-319-15509-8_2.pdf.

[25] M. Mihailescu. "Measuring the available bandwidth in a network (end to end). Integration in MonALISA". In: (2004). URL: http://monalisa.caltech.edu/docs/MonALISA-Madalin_Mihailescu.pdf.

[26] P. Mockapetris. *RFC1034: Domain Names - Concepts and Facilities*. Nov. 1987. URL: https://www.ietf.org/rfc/rfc1034.txt (visited on 06/12/2017).

[27] P. Mockapetris. *RFC1035: Domain Names - Implementation and Specification*. Nov. 1987. URL: https://www.ietf.org/rfc/rfc1035.txt (visited on 06/12/2017).

[28] T. Narten, R. Draves, and S. Krishnan. *RFC4941: Privacy Extensions for Stateless Address Autoconfiguration in IPv6*. Sept. 2007. URL: https://tools.ietf.org/rfc/rfc4941.tx (visited on 02/04/2018).

[29] ntop. *PF_RING - ntop*. 2017. URL: http://www.ntop.org/products/packet-capture/pf_ring/ (visited on 07/03/2017).

[30] *Open Resolver Project*. URL: http://openresolverproject.org (visited on 06/12/2017).

[31] I. Petiz et al. "Detecting DDoS Attacks at the Source Using Multiscaling Analysis". In: (2014). URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6959267.

[32] T. Rozekrans and J. de Koning. "Defending against DNS reflection amplification attacks". In: (2013). URL: https://www.nlnetlabs.nl/downloads/publications/report-rrl-dekoning-rozekrans.pdf.

[33] A. Sadeghian and M. Zamani. "Detecting and Preventing DDoS Attacks in Botnets by the Help of Self Triggered Black Holes". In: *2014 Asia-Pacific Conference on Computer Aided System Engineering (APCASE)* (2014), pp. 38–42. URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6924468.

[34] *SURF | Home*. URL: https://www.surf.nl/en (visited on 11/27/2017).

[35] The ZMap Team. *The ZMap Project*. 2017. URL: https://zmap.io (visited on 06/12/2017).

[36] S. Woolf and D. Conrad. *RFC4892: Requirements for a Mechanism Identifying a Name Server Instance*. June 2007. URL: https://www.ietf.org/rfc/rfc4892.txt (visited on 06/27/2017).