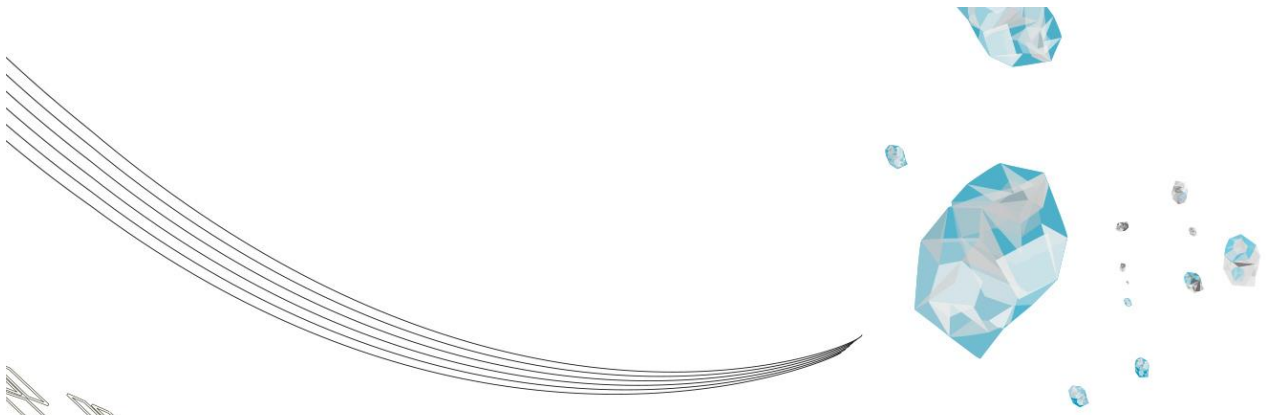
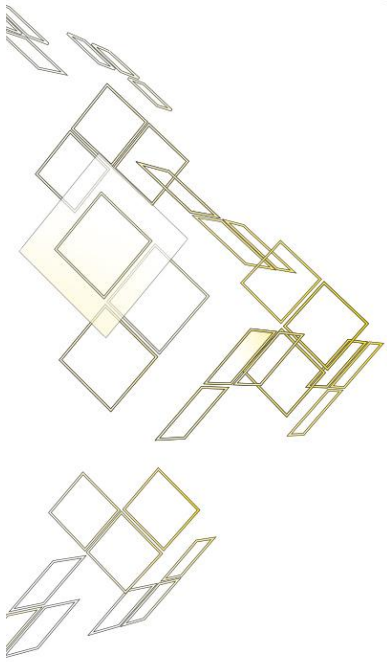
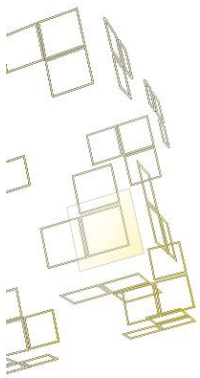
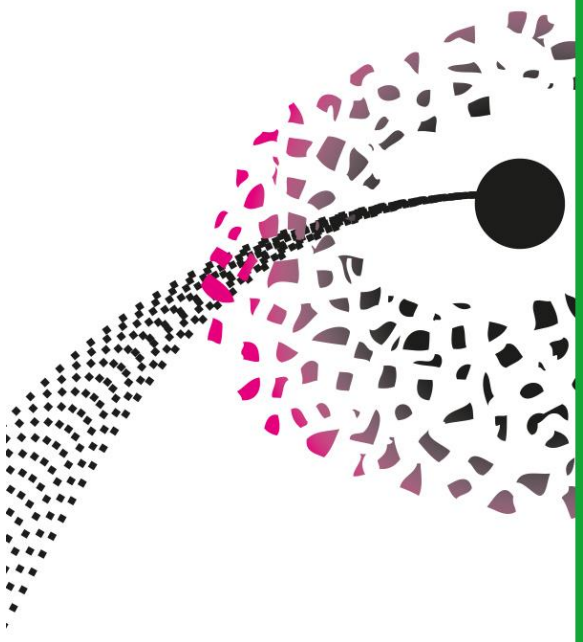








FACULTY OF ELECTRICAL ENGINEERING,
MATHEMATICS AND COMPUTER SCIENCE

MEASURING THE IMPACT OF DDOS ATTACKS ON MAIL SERVICE INFRASTRUCTURES

MSC THESIS

MARIOS MAVROPOULOS PAPOUDAS

AUGUST 2024

Supervisors:
dr.ir. M. Jonker
dr.ir. R. Sommese

Design and Analysis of Communication Systems
Faculty of Electrical Engineering,
Mathematics and Computer Science
University of Twente
P.O.Box 217
7500 AE Enschede
The Netherlands

UNIVERSITY OF TWENTE.

ABSTRACT

Since its inception in the 1960s [1], e-mail has risen to become the primary mode of official virtual communication for businesses, organisations and individuals[2], with billions of messages being sent daily[3]. For this reason, the importance of maintaining stability and ease of operation in e-mail exchange cannot be overstated. This is particularly true as, due to its importance, mail infrastructure presents a prime target for cyberattacks.
Distributed Denial-of-Service attack (DDoS) attacks are a particular type of attack that can disrupt or outright halt all network operations of a system [2]. The e-mail infrastructure has been a victim of such attacks in the past[2][4][1]. But: how effective are those attacks at disrupting mail exchange?
To answer this question, we designed a measurement infrastructure to observe and gauge the effectiveness of attacks on mail service infrastructures. Over a period of one month, we performed longitudinal measurements on a stable infrastructure consisting of roughly 2.5M Mail eXchange (MX) Record Internet Protocol, version 4 (IPv4) addresses. Over that same period, we also we performed reactive Simple Mail Transfer Protocol (SMTP) measurements on MX Record Internet Protocol (IP)s corresponding to MTAs under attack, derived from CAIDA's Reflective - DDoS (RDDoS) traffic. Based on deviations in the attack targets' Round-Trip Time (RTT) in the reactive and longitudinal SMTP measurements, we derived baseline thresholds to measure attack impact, both upon detection of the attack by the infrastructure, and for a period of roughly one day after each attack was originally detected. We further contextualized impact by performing three case studies on attacks with significant on their target hosts.
We concluded that the infrastructure is overall resilient against DDoS attacks and that operation can be maintained in the face of these attacks. However, the case studies also highlighted the potential that such attacks have to greatly disrupt mail exchange.

[1] https://therecord.media/ddos-attacks-hit-multiple-email-providers

ACKNOWLEDGEMENTS

The completion of this thesis would not have been possible without the guidance and support of my supervisors. First of all, I would like to thank my supervisor dr.ir. R. Sommese. Thanks to your mentorship, constructive feedback sessions and expertise, I was able to acquire a great deal of understanding, not only regarding the fascinating field of electronic message communication, but also the meticulous process of engineering complex software. I would also like to thank my supervisor dr.ir. M. Jonker. Your guidance and insight regarding both the methodology and engineering of this project expanded my understanding about the internet and the interaction of the components of infrastructure at scale.
I would like to give special thanks to my study mentor, prof.dr.ing. Florian W. Hahn. Thank you for your support throughout the duration of my Master's degree, and for guiding me towards an academic path that has made me a better engineer and person.
I would also like to give a big thank you to my loving and supportive friends and family. Thank you for being by my side for this great adventure, in good times and bad. Finally, I would like to thank Ilia Kalaitzidou. Thank you for always being there for me, for always believing in me, for your invaluable help with code reviews and presentations and for inspiring me to improve myself every day. Thank you all very much.

LIST OF ACRONYMS

AS Autonomous System

CAIDA Center for Applied Internet Data Analysis

CDN Content Distribution Network

CPU Central Processing Unit

DDoS Distributed Denial-of-Service attack

DNS Domain Name System

EHLO Extended HELO

HELO Core SMTP identification command

ESMTP Extended SMTP

GBs Gigabytes

GUI Graphic User Interface

IBR Internet Background Radiation

ICMP Internet Control Message Protocol

IP Internet Protocol

IPv4 Internet Protocol, version 4

MA Mail Agent

MBs Megabytes

MBps Megabits per second

MSA Mail Submission Agent

MTA Mail Transfer Agent

MUA Mail User Agent

MX Mail eXchange

OS Operating System

PID Process IDentity

RDDoS Reflective - DDoS

RSDoS Randomly-Spoofed DoS

RFC Request For Comments

RR Resource Record

RTT Round-Trip Time

SMTP Simple Mail Transfer Protocol

SSL Secure Socket Layer

TCP/IP Transmission Control Protocol/internet Protocol

TLS Transport Layer Security

TTL Time-To-Live

CONTENTS

1 INTRODUCTION

Since the development of the first local e-mail systems to achieve message communication between the users of time-shared operating systems in the 1960s[1], e-mail has been the main way to achieve written communication on interconnected systems. After the introduction of SMTP [5], e-mail infrastructure was adapted to the client-server model of the World Wide Web. With the creation of Domain Name System (DNS) and the introduction of webmail services by hypergiants, such as Gmail, Outlook and Yahoo Mail, e-mail infrastructure shifted from per-organization in-house solutions to multi-tenant cloud infrastructures[3].
Despite the introduction and gradual proliferation of instant messaging applications, which have largely taken over the role of unofficial communication between internet users, e-mail use remains high, with billions of messages sent on a daily basis[3]. Additionally, e-mail use is particularly important for all types of organizations, from private businesses to public institutions, all of which are highly dependent on e-mail for their daily operations[2].
Given the usage and importance of e-mail infrastructure for the operation of modern society, it follows that e-mail service infrastructures are considered attractive targets for cyberattacks. DDoS attacks are a particular type of attack aimed at causing partial or total disruption of the operations of a networked system by depleting its resources through the use of malicious and/or overwhelming internet traffic[2]. DDoS attacks have gained popularity in recent years[1], as they can be observed on both the DNS infrastructure[6][4] and e-mail infrastructure[2][4]. However, despite evidence of widespread DDoS attacks on e-mail service infrastructures[2][2], the impact of such attacks on said infrastructures has not been extensively studied by research. The disparity is particularly apparent when considering the research devoted to DDoS attacks on the DNS infrastructure [6].
For this reason, in this thesis we aim to study the behaviour of DDoS attacks on e-mail service infrastructures, examining the extent of e-mail servers targeted by a DDoS attack, as well as attempting to create a system to assess the potential impact of such attacks. The remainder of this Thesis is structured as follows. Chapter Two provides background on DNS, e-mail and DDoS, including relevant terminology and protocol outline. Chapter Three presents relevant work in studying DDoS behaviour on the DNS field, which is used as a stepping stone to outline the methodology used to examine said behaviour in the e-mail sphere. Chapter Four outlines the main goal and research questions, while Chapters Five and Six present the methodology and design phases respectively. The results of the analysis are detailed in Chapter Seven. The final conclusions are presented in Chapter Eight.

[1] https://www.a10networks.com/blog/5-most-famous-ddos-attacks/
[2] https://therecord.media/ddos-attacks-hit-multiple-email-providers

2 BACKGROUND

This chapter provides information on the current standards and inner workings of some of the most important components that comprise the modern Internet. The chapter begins wit an overview of DNS and e-mail infrastructures. This includes a brief description of terminology and attendant protocols used to implement the concepts of the Domain Name System and electronic messages. Similarly, the chapter also examines the evolution of DDoS attacks and the ways in which they are employed to damage DNS and e-mail infrastructure.

2.1 DNS

In its most abstract form, DNS can be considered to be a lookup table, mapping IP addresses to the so-called "domain name space". The domain name space consists of a set of names in a hierarchical tree structure, separated by the character "."[7]. The DNS infrastructure is comprised of three main components: a distributed collection of records, a set of name servers detailing the domain tree's structure and set information and the resolvers, which respond to mapping queries on the domain servers' files[7]. In this section, we will briefly describe those three components, focusing on a particular type of records detailing information about mail servers.

2.1.1 Resource Records

Each element of the tree comprising the domain name space contains a set of Resource Records (RRs). These are used during the Resolution Process to map IP addresses to provided domains. Every Resource Record contains fields with the following information[7]:

- **owner**: the domain name which contains the record.
- **type**: encoded 16 bit value that specifies the type of the resource in this resource record. Types refer to abstract resources. Common types include A, AAA, MX, NS etc.
- **class**: an encoded 16 bit value which identifies a protocol family or instance of a protocol. Most commonly, IN refers to the Internet protocol.
- **Time-To-Live (TTL)**: a 32 bit integer in units of seconds, primarily used by resolvers when they cache RRs. The TTL describes how long a RR can be cached before it should be discarded.
- **RDATA**: the type and sometimes class dependent data which describes the resource;. Common RDATA types are A, CNAME, MX, NS, PTR.

The main focus of the present work concerns MX type Resource Record (RR)s[8]. These records match a domain name with two pieces of data: a preference value (an unsigned 16-bit integer), and the name of a mailing host. The preference number expresses the delivery order from mailer to MX hosts[8]. A lower number indicates higher priority. Multiple MXs can share the same preference and have the same priority. In addition to mail information, mailers

may encounter or choose to use additional RRs types. An example concerns the canonical name (CNAME) RR, which assigns an alias to its owner name and specifies the corresponding canonical name in the RDATA section of the RR[7].

2.1.2 Resolvers

Resolvers are programs which achieve communication between user applications and name servers[7]. A resolver's main task is to receive a request (or "query") from a user application, communicate with the appropriate name server containing the requisite information, and present the name server's response to the user application in a form that can be parsed by the user application's local host (OS, user program etc)[7]. Due to the variance between the resolution procedure of a given query, the Resolver may have to handle multiple redirections from name servers with which it communicates. Thus, response time for handling user application queries can vary significantly[7]. To minimize response time and offload resource usage from name servers caused by potentially repetitive queries, most resolvers are able to cache retrieved information[7].
Resolvers are required both to forward queries from users to name servers and to resolve those queries on the name server side. The latter type of resolvers, known as recursive name servers, are examined as part of the name server infrastructure in a later section.

2.1.3 Name Servers

Name servers are server programs which hold information about the domain tree's structure and set information[7]. A particular name server usually contains complete information about a subset of the domain name space and can redirect to other name servers to retrieve information for any subset of the domain name space[7]. A name server may cache the structure of any part of the domain tree, but a particular name server generally has complete information about a subset of the domain space[7]. In addition, it may be able to redirect queries to other name servers that can be used to extract information from any part of the domain tree[7]. The domain tree is divided up into sections called zones which are then distributed among name servers. A zone is a subset of the DNS namespace that is managed by a specific organization or administrator[7]. It is important to note that a DNS zone is not associated with a domain name or a single DNS server. As such, a DNS zone can incorporate multiple sub-domains, and reside in the same server as other zones[7]. Based on their role in the DNS resolution process, name servers are divided into recursive name servers and authoritative name servers. An overview of their core components and function is provided in the following sections.

2.1.4 Recursive Name Servers

Recursive name servers receive DNS queries from clients, query the name servers of that zone or its parent zone for the designated RRs that form the response to the client DNS query and then forward those RRs back to the client. A recursive name server then caches this information for a time period, aiming to reduce queries to the authoritative name servers and thus optimize the name resolution process. This period is defined by the Time To Live (TTL) parameter of that particular RR. Should the recursive name server not contain information on the specific zone's authoritative name servers, it starts a process to obtain it. Specifically, the recursive name server first queries the root zone. This is because every recursive name server has the root authoritative name server's records hard-coded in its memory. Based on the root query's response, the recursive name server proceeds with queries until it reaches the target zone. The recursive name server caches the queries at each step, to avoid repeating the process in future resolutions.

```
A.EXAMPLE.ORG     IN     MX     10     A.EXAMPLE.ORG
A.EXAMPLE.ORG     IN     MX     15     B.EXAMPLE.ORG
A.EXAMPLE.ORG     IN     MX     20     C.EXAMPLE.ORG
```

Figure 2.1: Example of resolution responses after retrieving MX records[8].

Figure 2.1 demonstrates a set of examples of information returned after retrieving various MX records in response to a query. We can see the record type, the priority number and the common name in three cases.

2.1.5 Authoritative name servers

A name server that contains complete information about the namespace of a particular zone is called an authoritative name server for that zone[7]. The role of authoritative name servers in the resolution process is twofold. On the one hand, they are responsible for holding complete, up to date maps of domain names to IP addresses for all the domains that belong to that zone. These maps are then used for the other role of authoritative name servers, namely, to respond to resolver queries.
A particular set of authoritative name servers are those of the root zone. The root zone is denoted by the dot (".") at the end of every domain name and is comprised of thirteen authorities, each managed by a distinct entity. In this way, the root zone is distributed among various name servers, thus achieving redundancy. Each such server is authoritative for the root zone and managed by a separate entity. This architecture improves the resilience of the DNS infrastructure through redundancy. Additionally, it means that different regions of the globe can be served by different name servers, improving response time and user experience.

2.2 E-mail

E-mail communication is undoubtedly prolific among individuals and organizations alike, representing the main form of official communication over the Internet[3]. The e-mail sphere has undergone a conceptual change since its inception, reflecting the gradual shift of the Internet's infrastructure from a centralized system model to a collection of distributed systems, exemplified by the widespread use of Content Distribution Networks (Content Distribution Network (CDN)s) and Cloud Services.
This section provides a brief description of a few key concepts and characteristics of the e-mail sphere. The focus of the present work is the fundamental protocol used to achieve electronic messaging communication between internet user applications: the Simple Mail Transfer Protocol (SMTP). To that end, the protocol and its evolution concerns the first part of this section. The second part discusses a key component of the infrastructure implementing the various versions of SMTP: the various Mail Agents. The third part discusses how those mail agents are implemented in the era of webmail service providers.

2.2.1 SMTP

The SMTP consists of three core components: the Sender-SMTP is a process responsible for transferring e-mail composed by a user to the users specified by the mail's author. The Receiver-SMTP is a process responsible for receiving mail directed at a specific user. The SMTP commands/replies are a set of commands, known to both the sender-SMTP and the receiver-SMTP, which implement the protocol. These commands, along with the mail, comprise the main content of an e-mail message. Figure 2.2 showcases an overview of the model[5].

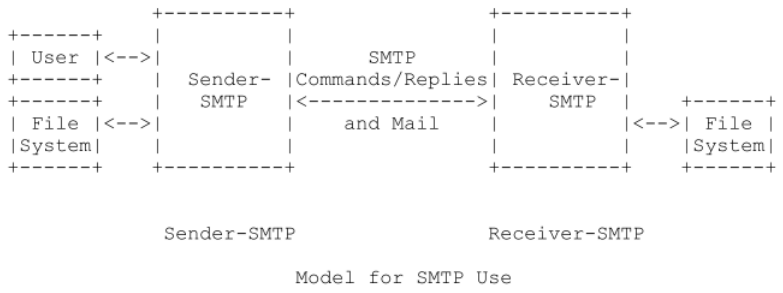


Figure 2.2: Overview of the SMTP architecture[5].

SMTP works in the following steps: first, a user submits a mail request to the sender-SMTP. In response to this, the sender-SMTP establishes a two-way transmission channel to a receiver-SMTP, which can be either the initial destination of the mail or an intermediary. Once the channel is established, the SMTP-sender sends a MAIL command to the receiver, indicating the sender of the mail. If the SMTP-receiver can accept mail, it responds with an OK reply. Upon reception of the OK reply, the SMTP server then sends an RCPT command identifying a recipient of a mail (which could be a set of source routing lists of hosts and the destination mailbox[5, Section 3.1]). If the SMTP-receiver is responsible for accepting mail for the particular recipient, it responds with an OK command; otherwise, it rejects the particular RCPT command, though not the entire mail transaction. As soon as SMTP-sender and SMTP-recipient have negotiated all possible recipients, the SMTP-sender sends the mail data, terminating with a special request. Upon successfully processing the data, the SMTP-receiver responds with an OK reply[5, Section 2]. In case the information provided by the SMTP-sender on the <forward-path >source routing list is incorrect, the SMTP-receiver can, if it knows the correct destination, respond with a specific reply notifying the SMTP-receiver that it will forward the request to the proper address. Alternatively, the SMTP-receiver sends a reply suggesting a possible forwarding address to the sender. These responses implement a forwarding mechanism allowing the SMTP-receiver to contact the correct destination[5, Section 3.2].
SMTP also offers user verification and mail list extension mechanisms. The VRFY command takes a user name as string argument, and accepts a response which may include the full name of the user but must include the correct user mailbox. The term “user name” is purposely broadly defined. If a host implements a VRFY command then it must recognise local mailboxes as “user names”, but the term may be expanded with other strings if necessary. The EXPN command takes as input a string identifying a mailing list, and the response must include all mailing boxes on said list.
The AUTH command is part of the introduction of service extensions to SMTP to allow SMTP clients to indicate an authentication mechanism to the server, perform an authentication protocol exchange and, optionally, negotiate a security layer for subsequent protocol interactions[9, Section 1]. It indicates an authentication mechanism to the server. If the server supports said mechanism, it performs an authentication protocol exchange, thereby authenticating and identifying the user. If specified by the command, the server also negotiates a security layer for subsequent protocol interactions. If the authentication mechanism is not supported the server rejects it with a 504 reply.

2.2.2 The various Mail Agents

As the Internet infrastructure incorporated the use of application-centric software compartmentalisation, and the adoption of DNS resulted in Internet communication being handled via Domain Name Resolution queries[7], the original, sender-receiver-SMTP was extended and up-

dated by the introduction of Mail Agent (MA)s[1] [10]. MAs are processes residing either on user machines or mail service hosts that implement various parts of SMTP that used to be the purview of the sender- or receiver-SMTP. These agents are: Mail User Agent (MUA)s, Mail Submission Agent (MSA)s and Mail Transfer Agent (MTA)s. Figure 2.3 provides an overview of the Mail Agent-based SMTP architecture. In particular, the figure demonstrates the position and role of each agent in the mail exchange procedure, from sender to receiver.
A detailed overview of MSAs and MTAs is provided in the following sections. Since MUAs reside mostly at the end user level, be it sender or receiver, and are thus not part of the intermediate host infrastructure, they are not of interest to this particular thesis. As such, they are not examined in detail.

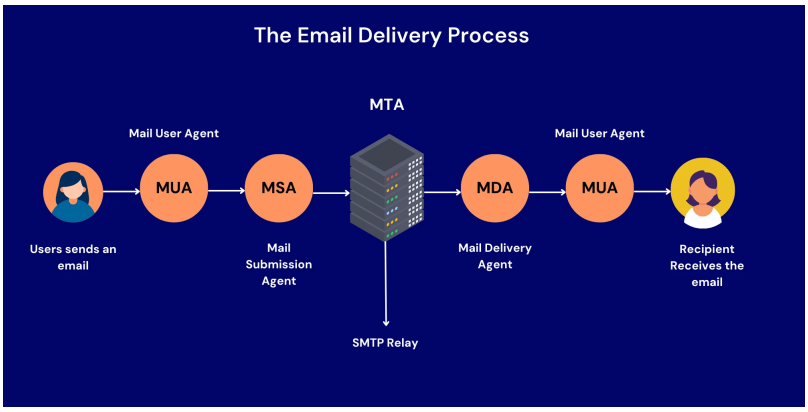


Figure 2.3: Overview of the Mail Agent SMTP architecture. Image found in `https://www.smartlead.ai/blog/mail-transfer-agent-guide`.

2.2.3 Message Submission Agent

A Message Submission Agent (MSA) is defined as a process accepting a message from a Message User Agent(MUA). A Message User Agent is a process which acts to compose or submit new messages and process delivered messages. These actions are often done on behalf of an end user and make use of a user interface for the purpose[11, Section 2.1].
MSAs have gained rapid adoption due to changes in security considerations and a shift of responsibility for traffic generated by servers to those servers themselves. A clear example concerns the prevalence of machines infected with malware which generates large amounts of spam mail. Because of that reason, many servers prohibit outbound traffic on the standard SMTP port 25, and instead funnel said traffic to dedicated submission servers[11, Section 1]. In addition to security reasons, many messages may contain unfinished fields or unresolved domain names. Further, prior to submission, local policy may dictate modification of content to conform to, e.g., information disclosure policies[11, Section 1]. When left to the provision of MTAs, such actions have been known to disrupt MTA functionality and are considered to be outside their provisioned responsibilities. To address these concerns, MSAs are used.
All MSAs must be capable of performing a set of actions. Specifically, MSAs must, unless a more precise code is required, return a 554 response code to reject all MAIL, RCPT or data commands which contain any inconsistencies. Further, MSAs must guarantee that all domains in the message envelope are FQDNs. Most importantly, MSAs must reject all MAIL commands if the session has not been authenticated using the AUTH mechanism described in [12]. The only exception to this rule is if authentication or authorization has already been independently established. Such a case would involve, for example, the MSA being in a protected sub-network[11,

[1] Note: The words “mail” and “message” in agent naming schemes are considered equivalent and are used interchangeably.

Section 4]. Further, it is recommended that the MSA should enforce address syntax, keep an error log and apply shorter timeouts than those specified in [10, Section 4.5.3.2].

2.2.4 Mail Transfer Agent

In the Mail Agent model, the relay of mail is accomplished via Mail Transfer Agents (MTAs). A MTA acts as an SMTP server to accept messages from either MSAs or another MTA, and either delivers them to their final destination or acts as an SMTP client to another MTA[11, Section 2.1]. Most current MTAs provide some form of forwarding support. In most cases, forwarding is used to consolidate and simplify addresses, either internal or related to some enterprise. Less common is the establishment of a link between an individual's prior and current address. In either case, silent forwarding of messages is preferred due to non-disclosure or security considerations. In silent forwarding, the server does not notify the sender of the forwarding action. As such, final address exposure as a side effect of the SMTP forward protocol is discouraged. Therefore, implementation engineers are advised to carefully evaluate return codes in case of forwarding, especially concerning 251 and 551 reply codes of the RCPT command[10, Section 3.4].
With the widespread adoption of MX DNS records, SMTP clients are no longer required to generate explicit source routing lists. Instead, MX records now usually point to SMTP relay servers. If such a server accepts the task of relaying mail (the same way it may accept or reject the task from a local user), then it becomes an SMTP client, establishes a channel to the next SMTP relay server specified in the DNS and relays the mail. The process is thus repeated until the message reaches its final destination. Target host location begins with lexical identification of the domain to which the mail must be delivered for processing. Then, domain name resolution via DNS lookup yields the domain name. Only Fully Qualified Domain Names (FQDNs) should be acceptable domain name resolution answers. Due to issues with FQDN inference from partial names, initial submission SMTP servers should not make such inferences, and relay SMTP servers are prohibited from making them. Some flexibility is allowed for Message Submission Servers. The lookup begins by attempting to locate an MX record for a given name. If a CNAME record is located, that CNAME recorded is processed in lieu of the original name. If a non-existent domain error is returned, this must equal to an error in the submission procedure. If an empty list of MX records is returned, then the address is treated as if associated with an implicit MX Resource Record (RR), with a preference of 0, pointing to that host. If MX records are present but unusable, or the implicit MX is unusable, the host location procedure must terminate with error[10, Section 2.3.5, 3.6, 5.1].
If one or more MX RRs are found for a given name, SMTP servers are only allowed to use address RRs associated with that name if those addresses emerged from MX RRs. The "implicit" MX rule above only applies if there are no MX records present. If MX records are present but unusable, the process must terminate with error. When a domain name associated with an MX RR is looked up and the associated data field obtained, that data field must contain a domain name. Upon querying that domain name, the response must be at leas one address record (be it A, AAAA or other) that gives the IP address of the SMTP server to which the message must be directed. Any other response falls outside the Standard's scope. Upon successful lookup, the mapping can result in a list of alternate delivery addresses due to multiple MX records, multihoming or both. In such cases, the server should try and retry each address in the list, in order, until successful delivery. In case of configuration pertaining to an upper limit of alternate addresses, that limit should be at least two addresses. In case of multiple MX records, preference order must be adhered to, with lower numbers placed in higher priority than greater ones. In case of multiple destinations with the same preference, if no reason exists to favour one over another (e.g. by recognition of an easily reachable address), then randomization should be applied to achieve load balancing for the specific target organization. In case of multihomed

hosts, the SMTP-sender should adhere to the order of alternate IP addresses as presented. It is the responsibility of the resolver to order said addresses in an optimized manner[10, Section 5.1].

2.2.5 An overview of the “Retry” Mechanism

Most SMTP hosts include a set of user mailboxes, a set of areas designated as queues for messages in transit, and a set of daemon processes for sending and receiving mail. To handle increased traffic, a set of optimization strategies (various “Retry” mechanisms) have thus been deployed. Any queuing strategy must include timeouts on all activities at a per-command basis, and no queuing strategy should send an error message in response to another error message. Generally, most SMTP clients have a set of processes periodically attempting to transmit outgoing mail. Typical implementation has the program responsible for composing a message being capable of requesting immediate attention for any piece of outgoing mail. Further, any piece of outgoing mail that cannot get immediate attention enters a queuing system, upon which it must not remain indefinitely, thus it must be periodically retrieved by the sender. In addition to the message content, each such queue entry must contain all other relative fields and information (the “envelope”). The sender must delay retrying a particular destination after one attempt has failed. The retry interval should be at least 30 minutes, although that number varies depending on the specific retry strategy. Retries continue until the message has been delivered or the sender gives up. The give-up time, though again strategy-dependent, should be measured over a period of days. The retry and give-up times form the “Retry Window” for the particular strategy. It is important to note that retries should be based on a list of unreachable hosts correlated with their relevant connection timeouts, rather than a round-robin approach.
The actual retry and give-up times are strategy-dependent and, ultimately, left to the judgement of the server administrators. A clear example of this can be seen in the default settings for the Retry window of some of the most popular MTAs. These have been set quite differently for each agent, and remain fully customizable by administrators.
For example, the default setting for Postfix is 5 minutes initially, which doubles after every failure until it reaches roughly 66,66 minutes, after which time the retry window is kept stable for a maximum period of 5 days. Similarly, for Exim, the strategy is: initial retry window size of 15 minutes for the first two hours since first failed delivery, then a one hour window size increased by a factor of 1.5 with every attempt for the next fourteen hours, after which point the MTA falls back to a six-hour retry window size for a maximum period of four days.
The RFC, however, does offer a set of guideline times. According to the RFC, experimental data favour a connection policy of two attempts in the first hour, then a lessening to one attempt every two to three hours. Failure is mostly identified as transient, meaning either the target system or its connection has crashed.[10, Section 4.5.4.1].

2.2.6 E-mail Service Providers

Despite many common characteristics of the e-mail and the postal service, a key difference is that e-mail administration is decentralized and left to the discretion of each autonomous entity and organization[3]. This fact has led many organizations to outsource their e-mail infrastructure to a third-party service provider. Over the years, a few third-party e-mail service providers, notably Google and Microsoft, have dominated the current e-mail service provisioning market to the degree that concerns have arisen about a gradually centralized internet and the prevalence of issues adherent to this (technical, legal, jurisdictional etc.)[3, 13, 14]. As such, it is important to consider the modern e-mail infrastructure using the webmail model. Webmail, or web-based e-mail, can be understood to be any e-mail service accessible from a web browser. Under this model, the Mail User Agent and (parts of) the Mail Submission Agent are in effect concatenated into the website or app that the e-mail service provider makes accessible from

a browser (or smartphone/desktop application). Additionally, the inbox and the are now implemented on the provider's infrastructure, either on their own CDNs and servers or on their off-nets[15]. Considering the above, the prevalence of webmail as a service and the difference of its infrastructure from the traditional client-server model leads to the conclusion that webmail service infrastructures are a key point in the architecture of e-mail today.

2.3 DDoS

This section gives a brief look into (Distributed) Denial-of-Service attacks. We present a brief outline of the types of DDoS attacks, explain the prevalence and role of botnets in their implementation and examine the relationship of DDoS attacks and e-mail services, as observed by literature.

2.3.1 Types of DDoS

DDoS attacks are coordinated assaults on a set of target hosts using another set of compromised hosts to send attack packets to the targets[4, 16]. Four key steps comprise a DDoS attack: first, the attacker uses scanning methods to locate vulnerable hosts to be used in the attack. Second, said hosts are made compromised via the installation of some form of malware, granting the attacker remote control capabilities. The third step involves using the remote control to launch an attack on a target host, while the final step involves removing traces of the attacker's presence on the compromised hosts. Target resources often include routers, links, firewalls and other defense systems, computer and network infrastructure, OS, current communications and applications[4, Section IA]. An overview of the various types of DDoS attacks is displayed in Figure 2.4.

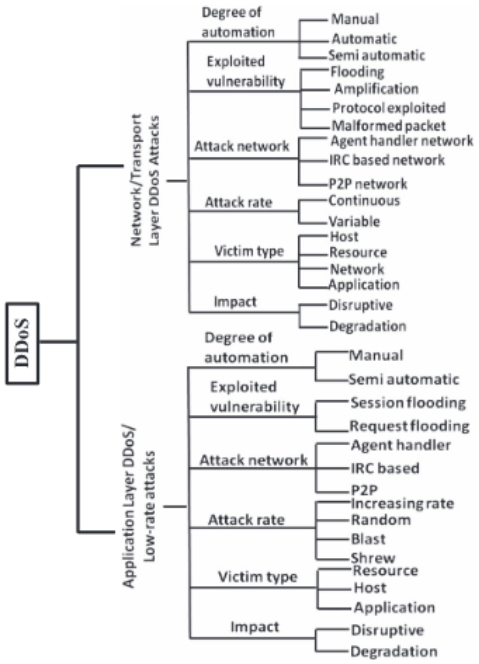


Figure 2.4: Taxonomy of DDoS attacks[4].

The two basic types of DDoS attacks are network/transport layer DDoS attacks and application layer DDoS attacks. Depending on the vulnerability they exploit, they can be flooding attacks, amplification attacks, protocol exploitation attacks, malformed packet attacks, semantic or brute force attacks. Based on the type of network formed by the compromised hosts, DDoS attacks

can be agent handler, IRC network, or P2P network - based. The attack rate of a network attack can be constant, increasing or variable and the target can be a vulnerable host, victim, network, resource or application. Finally, depending on the volume of the traffic generated for the attack and the vulnerability exploited, the impact can be disruptive or performance-degrading[4, Section IIIA].
Application layer attacks aim at exhausting resources allocated to Web services by targeting the HTTP(S) protocol. Application layer attacks usually exploit protocol vulnerabilities to cause either session flooding or request flooding, starving the machine of resources to handle sessions or respond to requests. While the attack network of Application layer attacks is similar to that of network/transport layer attacks, the attack rate can additionally be pseudo-random, with random increases of traffic intensity, burst-oriented, with periodic bursts of HTTP GET requests, or blast-oriented, centered around generating a high number of requests for an extended time period. Potential victims include any web-based services, such as web servers, mail servers or hosted applications. The impact behaviour is similar to network/transport layer attacks. Of particular interest in this category are SMTP attacks. There are two types of SMTP attacks[4]. SMTP error DoS, mailbox DoS (excessive email size) and SMTP mail flooding attempt to overwhelm email server. In SMTP buffer overflow attacks, different SMTP commands can cause the SMTP server to crash or execute arbitrary byte-code that could lead to a system compromise[4, SectionIIIA].

2.3.2 The role of Botnets in DDoS attacks

Botnets are networks comprised of bots, machines infected with malware which allows a portion of their resources (CPU, memory, internet etc.) to be allocated to the execution of external commands by a malicious actor. Bots are controlled by an authority known as the “botmaster”[4]. They are used extensively in DDoS attacks, particularly those attacks implemented with the use of spam mail[4]. There are three main models to achieve these attacks, the agent handler model, the IRC botnet model and the web-based model[17]. In all cases, however, bots issue commands through their Command and Control (C&C) module, a server connected to the botmaster and the compromised host and acting as communication relay[4]. Centralized and distributed mechanisms are the two main different approaches to C&C communication. Due to various weaknesses of both architectures, there has been a shift to using peer-to-peer botnets, which are more robust and harder to dismantle[4]. To counter a botnet based attack, knowledge of the malware code and enhancements that may have been applied to such code is of critical importance[4].

2.3.3 DDoS and SMTP

SMTP is especially susceptible to DDoS attacks[18]. The main reason for this is that SMTP routinely saves queued e-mail to disk in order to protect it from system outages[18]. Thus, even with relatively small amounts of bandwidth, SMTP servers can be rendered inoperable[18]. Specifically, SMTP is vulnerable to attacks targeting weaknesses in the implementation of the server system, usually misconfigurations, and distributed attempts to consume all of a scarce resource, such as CPU, network IO or disk IO. This is proven by the disclosure of [19]. SMTP is particularly weak against the second type of attack, namely an increased influx of contemporaneous requests. This is due to the relatively low amount of bandwidth[18] required to saturate the available disk bandwidth of current servers. Interestingly, the requests comprising a DDoS attack can be benign in nature: they cause disruption to normal levels of service by being appropriately timed[2, Section 1].
A brief summary of SMTP routing is as follows: first, the user composes a message using a Mail User Agent (MUA). Then, the process to deliver the message to its destination begins. First, the message is relayed to a Mail Transport Agent (MTA), which can share the same physical spaces with the original MUA. This agent then forwards the message to the destination MX,

possibly using other MTAs as relays depending on local configuration. The MX is at its core another MTA. However, it is also designated in the Domain Name System (DNS) as capable of delivering email for the destination domain name. The MX then forwards the message to another variable length chain of MTAs until it reaches the MTA responsible for forwarding messages to the recipient's mailbox. At that point, a Mail Delivery Agent (MDA), either a part of the MTA or an autonomous process such as procmail, writes the message to the recipient mailbox's storage space. Finally, a MUA on the recipient end performs regular lookups on the recipient mailbox's storage space and, upon locating the newly arrived message, retrieves and displays it. The mailbox and the destination MUA may occupy the same physical space or be connected via the Internet.
A high-level overview can be seen in Figure 2.5. Note: the dotted lines represent potential DNS resolution. Specifically, DNS Resolution that only takes place if the various agents do not occupy the same physical machine. Such a case could be, for instance, if a user sends e-mail through a web-based application or mail provisioning service (GMail, Yahoo Mail, Microsoft Outlook etc).

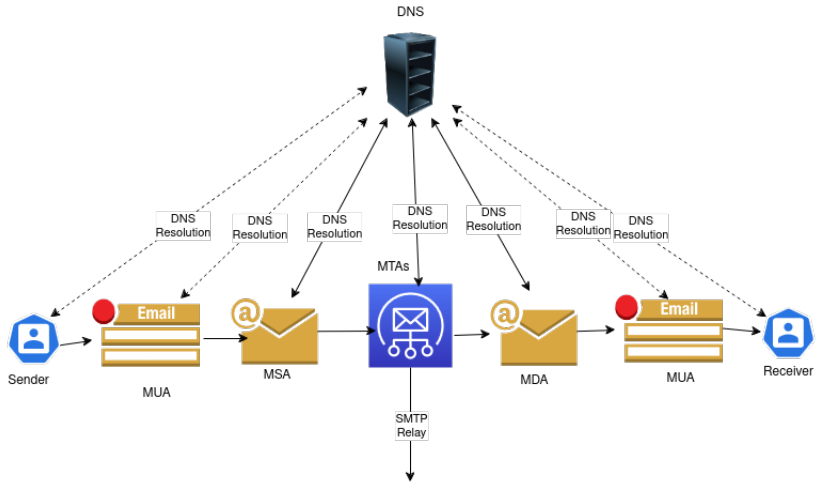


Figure 2.5: High-level overview of SMTP routing with DNS resolution. Figure made using `https://app.diagrams.net/`.

This procedure exposes certain weaknesses[2]. Namely:

- The mail delivery path is unusually complicated. At minimum, email is routed from a MUA to a local MTA, which in turn sends it to one routing "smart host" MTA, which passes it to the destination MX's MTA, and then to the recipient MUA. However, there is little evidence that the flow described above is necessary.
- The same mail delivery path requires every agent to perform multiple costly storage space lookups, removals and retrievals to transport a single message.
- The aforementioned delivery path is usually further extended by additional intermediate MTAs, which act much like proxies. This is the usual implementation of functionalities such as virus and spam scanning. These checks can be extremely expensive to execute, slowing down the MTA further.
- Despite the complexity of the above infrastructure, reliable mail delivery is still hard to guarantee. Factors such as the number of intermediate MTAs in the delivery path, their processing load and, consequently, the time which the message remains in storage play a vital role in its final delivery.

As such, a successful SMTP DDoS attack needs only to flood the final links in the chain forming the delivery path to deny service to the end user[2, Section 2].
Considering the above, as well as the prevalence of SMTP DDoS attacks, it is surprising that very little existing research can be found giving information regarding the subject, in contrast to information on general DDoS attacks on the internet[2, Section 3.1]. This disconnect brings into sharp relief the need for examining the attacks on SMTP and conducting studies concerning their real extent and impact.

3 RELATED WORK

This chapter provides a brief description of the methodology employed by research into DDoS behaviour on the DNS field, as well as the usage of mail service providers. The methods to achieve the goals outlined in these works are of paramount importance to understanding DDoS behaviour, and will thus be presented here.

3.1 Leveraging Internet Background Radiation for Opportunistic Network Analysis

In this paper, K. Benson et al.[20] re-purpose unsolicited traffic sent to unused regions of the address space, known as Internet Background Radiation (IBR), to be used as a data source of Internet-wide measurements. Specifically, the authors gather data from two large darknets and examine how it can be leveraged to perform Internet-wide measurements.
The data is gathered from two large darknets: collections of routed but unused IP addresses, operated by UC San Diego and Merit Network (UCSD-NT and MERIT-NT respectively). The authors study packet traces captured from July 31 to September 2, 2012 and July 23, 2013 to August 25, 2013. To perform a longitudinal analysis they also examine flow records collected by UCSD-NT from April 2008 to January 2015[20, Section 7.1]. They leverage the approach of A. Dainotti et al.[21] to clean up spoofed IP addresses from the flows and captured traffic alike, as their presence could influence inference accuracy. Similarly, they exclude all traffic from unrouted IP addresses, which may exist due to failed egress filtering by remote networks or spoofed sources missed by the first cleanup round.
The authors then map the IP Adresses to prefixes, leveraging Routeviews and RIPE RIS collectors similar to the approach of Lutu et al[22]. They further match IPv4 Addresses to Autonomous System (AS) numbers using CAIDA's Prefix-to-AS database[23]. The end result is a comprehensive framework that allows inference of a range of properties of networks from the global Internet space. The authors use the framework to determine IBR origin at network level, identify components that enable opportunistic network inferences and asses the effect of collection time and IP locality[20, Sections 4-8].

3.2 Measuring the impact of DDoS attacks on DNS infrastructure

In this paper, R. Sommese et al.[6] join two existing data sets – DoS activity inferred from a sizable darknet, and contemporaneous DNS measurement data – for a 17-month period (Nov. 20 - Mar. 22) in order to measure and characterise recent DDoS attacks on the authoritative DNS infrastructure[6]. The aim is to establish the reach of such attacks and measure their impact.
The main datasets used in the analysis are the UCSD Network Telescope, which collects backscatter traffic from ongoing DDoS attacks against IPv4 address space; and the OpenINTEL measurement project, which performs daily DNS queries of over 60% of registered domains, allowing detection of substantial changes in DNS query latency or reachability to authoritative nameservers over time[6, Section 1]. Resolution times experienced by OpenINTEL during attacks indicate their impact on the DNS; network telescope traffic allows partial inferences of

attack timing and intensity[6, Section 1]. Additionally, the authors use quarterly census snapshots of anycast deployment over the period January 2021 - January 2022. When authoritative NS IP /24 subnets are matched with /24 subnets detected in the anycast census, a lower bound estimation of anycast deployments is provided. Further, CAIDA's prefix-to-AS dataset is deployed to deduce AS numbers from given IP addresses. These numbers are then run through CAIDA's AS-to-organization dataset, thus providing the names of the organizations owning the respective ASes deduced from scanned IP addresses[6, Section 3.3].
The methodology used by the authors follows four main steps. In step one, the authors collate the datasets into a single entity. Step two revolves around identifying targeted nameservers based on the IP addresses contained in the aggregated dataset, which are then used in step three to compile a list of domains associated with said nameservers. In the final step, the authors query said domains and measure the RTT data to look for performance degradation as a result of DDoS attacks[6, Section 4].

3.3 Characterizing Mail Service Provider Usage

In this paper, E. Liu et al.[3] investigate the extent to which email service has been delegated to third-party services, instead of being handled in-house by each Internet-connected organization[3]. They investigate the prevalence and impact of this phenomenon by performing a large-scale analysis of and measurement of modern Email service provisioning and present a methodology for mapping domains to mail service providers.
The methodology proposed by the authors aims to overcome limitations in the main provider identification methods. The authors propose a methodology for mapping domains to mail service providers called the "priority-based approach". The approach incorporates data from multiple sources, including MX records, Banner/EHLO messages, and Transport Layer Security (TLS) certificates. High accuracy is achieved by prioritizing these sources by reliability: certificates first, then Banner/EHLO messages, and then MX records. The approach can be broken down into five main steps. The first step is *Certificate Preprocessing*. The goal of the first step is to find certificates that are potentially operated by the same mail provider. The domains listed in a certificate aid the mail provider inferences. However, certificates also introduce two issues. First, a mail provider can have multiple valid certificates. Additionally, each certificate can contain multiple domain names by using the subject alternative name (SAN) extension. Having multiple certificates, each with multiple domain names, leads to two challenges: which certificates belong to the same mail provider, and which name to use to represent that provider. These two challenges are met by preprocessing all certificates in the dataset and grouping certificates that likely belong to the same mail provider. The result is a representative name for each group to represent that group and the mail provider.[3, Section 3.2.1].
The second step of the approach entails assigning a mail provider ID for the IP addresses to which each MX record resolves. For each provider, two IDs are computed, from TLS Certificates and EHLO/Banner messages respectively. TLS-computed IDs have priority over EHLO/Banner IDs. If a valid certificate is present at the IP address, the representative name of the group containing the certificate is used as the ID. A certificate is valid if it is trusted by a major browser. If the Banner/EHLO message is available and contains a valid FQDN, the registered domain part of the FQDN is used as the ID[3, Section 3.2.2].
The third step uses the IDs obtained in the previous step to match them to MX Records. If all IP addresses of an MX record have the same ID from certificates, that ID is then assigned as the provider ID to the MX record. In cases where IDs from certificates do not agree or are not available, if all IP addresses share the same ID from Banner/EHLO messages, that provider ID is assigned to the MX record. Otherwise, the registered domain part of the MX record is used as the provider ID[3, Section 3.2.3].
The fourth part deals with identification mismatches. As illustrated by the authors, certain web

hosting companies allow their VPS servers to create certificates under specific domain names. Similarly, certificates can be misleading when third-party providers present their customer's certificates. Since there is no good way to automatically detect such cases without prior knowledge, such situations are identified manually. Another source of error comes from the fact that Banner/EHLO messages are unrestricted text. Thus, it is possible to make false claims in Banner/EHLO messages. Since the approach prioritizes Banner/EHLO messages over the MX record, there would be mislabeling. To efficiently find instances of misidentifications, the low server popularity and the low number of domains pointing to them in such edge cases is exploited. Once potential candidates have been identified, various heuristics are employed to facilitate examination.[3, Section 3.2.4].
The final step of the method assures that every MX record will have an assigned mail provider ID. This assignment could be either based on TLS certificate information, Banner/EHLO messages, or the MX record itself. Based on the MX record that a domain uses a mail provider can be assigned to that domain. In the case that a domain has more than one primary MX record (multiple MX records with the same priority but different provider IDs, which happens occasionally), the domain is split across the multiple providers[3, Section 3.2.5].

4 RESEARCH QUESTIONS AND GOALS

Given the importance of e-mail infrastructure for the function of the modern world and the dangers that DDoS attacks pose to that infrastructure, it follows that studying the impact of such attacks on the infrastructure is fundamental to the continued function, expansion and improvement of e-mail service provisions. Despite this, there has been little research devoted in this direction[2]. The studies in literature focus on spam mail activity and DDoS mitigation techniques for SMTP servers[2, 18].
Our research goal is to investigate the impact of DDoS attacks on a variety of e-mail service infrastructures, examining the frequency, intensity and effect of these attacks on e-mail service provisioning and performance, as well as examine their evolution over time. The first step involves inferring attack activity, relying on backscatter analysis as described in Moore et al.[24] with a focus on the e-mail space. The second step involves identifying targeted e-mail hosts and assessing the extent to which attacks on them influenced performance, leveraging the approach of R Sommese et al.[6] and E. Liu et al.[3]. With these steps, we hope to answer the following questions:

- How often are e-mail service infrastructures targeted by DDoS attacks? How widespread are these attacks on the e-mail space?
- How effective are these attacks at disrupting e-mail exchange?
- How does the existence of the mail re-transmission mechanism, as defined in [10], affect the impact of DDoS attacks? How effective is the mechanism at maintaining overall operation capability for the e-mail space?

The remainder of this thesis aims to present the methodology designed and analysis made to answer these questions.

5 METHODOLOGY

This chapter details the methodology that will be used to answer the research questions and thus achieve the research goal. It also explains the creation of the data set used in this thesis.

5.1 Data Set

The data set we use to gauge DDoS impact on SMTP hosts is created based on the approach of Sommese et al. [6]. It relies on joining data from two sources into a derivative stream.
Specifically, we use OpenINTEL MX record queries on roughly 303 million domain names [25]. We then filter the resolved names from these MX records through OpenINTEL's A record queries to derive the IPv4 addresses of SMTP servers in OpenINTEL's view.
In parallel to this, we also receive the Randomly-Spoofed DoS (RSDoS) attack feed curated by CAIDA [23] from captured traffic announced by the UCSD-NT [26]. This results in a list of IPv4 addresses under attack, updated every 5 minutes.
We join these two datasets to create a derivative data stream of SMTP mail hosts under attack. This final data set is also used by our measurement infrastructure. A graphical overview of the resulting data set is presented in Figure 5.1.

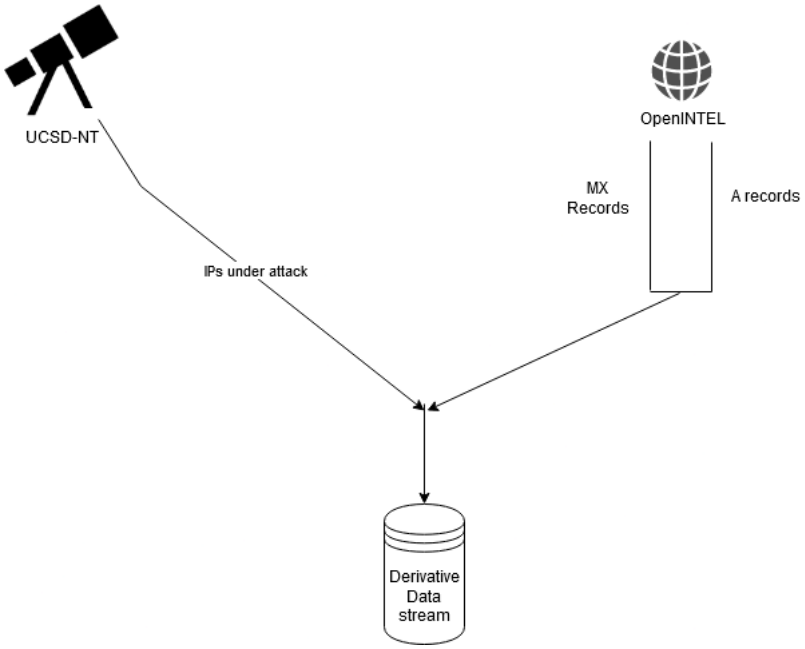


Figure 5.1: High-level overview of the process used to generate the data set used in the thesis. Figure made using https://app.diagrams.net/.

5.2 Methodology

As a first step, we need to map DDoS activity to the infrastructure of e-mail service provisions in order to identify e-mail hosts that are victims of attack. Similarly to the approach of R. Sommese et al.[6], we propose to infer victim domains based on CAIDA's RDDoS traffic from ongoing DDoS attacks against IPv4 address space[23]. Ensuing, we cross-reference the IP addresses obtained from resolving the MX records of the victim domains, collected by OpenIntel [25], with the UCSD Network Telescope backscatter data to identify e-mail hosts that are victims of DDoS attacks. Finally, we query the victim e-mail hosts and use the RTT data to measure performance degradation as a result of a DDoS attack.
To assess the impact of DDoS attacks on webmail hosts, we also took in consideration the hosts' "Retry" mechanisms. Allowing for periodic messaging re-transmission in case of failure, they let SMTP have "natural" resilience against data loss. Therefore, unless the intensity of a DDoS attack on the host is such that it causes complete crash and data loss, the "Retry" mechanism ensures that, eventually, all intended e-mail messages will reach their destination.
For this reason, we perform follow-up measurements on observed victim hosts, in accordance with the standard re-transmission time windows for SMTP, as defined in [9]. The event horizon for these follow-up measurements is one day from the moment the attack was observed. We further contextualize these follow-up measurements by keeping separate track of hosts that have repeatedly been observed as victims of a DDoS attack against hosts that have only been observed as victims once.
In parallel to this,to establish a baseline, we perform daily queries of the entire IPv4 MX record space. The queries are the same as for the reactive component of the approach. The daily queries are spread out over the span of one month. Using the RTT data from the resulting queries, we attempt to create a baseline response time profile of the e-mail sphere, giving us a lower bound against which to compare the RTT data of the victim hosts.

5.3 Limitations of Methodology

Our approach relies on a number of assumptions about the behaviour of the e-mail sphere. Therefore, the accuracy of the results is also impacted by any deviations which these assumptions have from realistic e-mail host behaviour. In particular:

- **The proposed re-transmission time windows for SMTP are not standardized**: The re-transmission time windows for SMTP, as defined in [10, Section 4.5.4.1], are not codified as default in the Request For Comments (RFC)[10, Section 4.5.4.1]. In fact, they are considered guideline values due to their satisfactory performance on an experimental level. However, the RFC clearly states that the actual retry window times are strategy-dependent and thus ultimately left to the judgement of the host's administrator[10, Section 4.5.4.1]. This means that our approach relies on the assumption that mail host administrators are basing their strategy on the guidelines presented in the RFC. Since there is no way of receiving the actual times for each victim host short of contacting said host's administrator, this represents a methodology limitation.

- **There is no guarantee that the victim hosts do not have e-mail messages already in their queue**: Our follow-up measurements do not consider that, at the moment of attack, victim hosts may already have messages in their queue which they are trying to deliver in accordance with their re-transmission window times. In essence, we treat the point where the point where the victim host comes on our radar as "point zero" for that host's mail queue. However, that assumption finds no guarantee in reality, particularly considering that most DDoS attacks on mail hosts are implemented via spam mail [18][2].

- **A change in a victim host's response during followup measurements is not clear indication of attack impact:** This is particularly true when host response changes after repeated connection attempts. A change in host response could indicate hosts with low tolerance to connection bursts. Specifically, the hosts could initially refuse an incoming connection, only to further allow it upon repetitive attempts. Since there is no way to examine such configuration settings short of contacting the host's administrator, we consider this a methodology limitation.

- **There is no information concerning the status of the sender-MTAs that send mail to the detected attack targets:** This is a strong limitation. The derivative data stream used as input for our measurement infrastructure as described in Section 5.1 consists of a set of SMTP hosts (MTAs) under attack. However, under agent-based SMTP (see Section 2.2.2), each MTA acts both as a destination and as a point of origin for a piece of mail. Our infrastructure only looks at the SMTP hosts of the derivative stream as mail destinations, and assumes that all MTA-senders to those destinations operate normally. This, however, is far from guaranteed. Obtaining any information regarding the identity and status of those MTA-senders would involve access to the MTA-receivers' (our targets') mail cache at the time of the attack. This falls outside the scope of the present work, and must therefore be considered a limitation.

6 DESIGN PHASE

This chapter details the requirements and reasoning that factored into the design phase of the measurement infrastructure's implementation. It goes through each component separately, defines the parameters of its comprising modules, describes said modules and outlines the choices and approach that led to the end result.

6.1 Approach

In accordance with the methodology 5.2, we divide the measurement infrastructure into two key components. One component performs set daily measurements on a stable infrastructure (set of MX record IPs) 6.2. The other component performs the same measurements on victim MX record IPs as they enter our field of view 6.3. The same component also performs follow-up measurements on the victim IPs. These follow-up measurements are taken at specific intervals since the first observation of victim IPs, as defined in [10].
A high-level overview of the measurement infrastructure is seen in Figure 6.1.

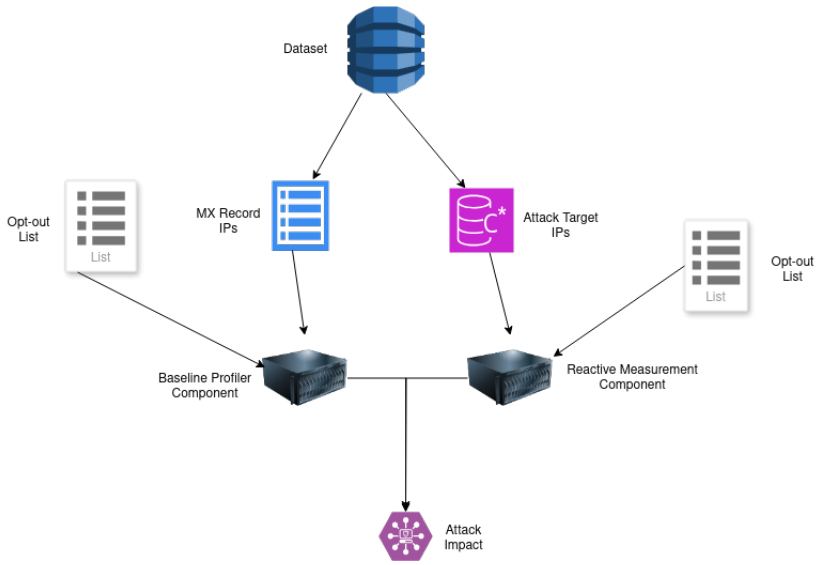


Figure 6.1: High - level overview of the measurement infrastructure. Figure made using `https://app.diagrams.net/`.

6.1.1 Preamble: responsible internet citizenship

A key part of the implementation of any internet measurement, scanning or profiling infrastructure is the need to ensure that measurements are done responsibly and respectfully. This need is even more pronounced in the case of this infrastructure. The reason for this is that both the reactive measurement component and (inadvertently) the baseline profiling component attempt to measure victim hosts.

Thus, The first priority when developing the infrastructure is to ensure that the hosts whom we measured knew our identity, the reason for our establishing contact and the nature of the measurements performed by our infrastructure. Additionally, we include an ”opt-out” method, in which all hosts wishing to be excluded from our research: a) are provided with a means to request so, and b) will see their request honoured by us in a timely manner.
To that end, we begin by establishing a scanning announcement. This announcement is served via a webpage from the IP address of the machine on which the infrastructure is deployed. it contains the following:

- **Statement of intent**: The announcement explains the reason for which the victim host was contacted by the infrastructure's IP address. It details the investigative framework of the measurements and notifies of the possibility of follow-up contacts in the context of a longitudinal analysis.
- **Detailed presentation of measurements**: The announcement presents the measurements that are performed. The type and service port are declared here.
- **Declaration of identity, opt-out policy and point of contact**: The announcement explains the organisational framework behind the measurements. It establishes a point of contact (via an e-mail address of the organisation) and outlines a simple process to request exclusion. To wit, a simple e-mail message to the specified address results in exclusion from the research.

The creation of the scanning announcement page ensures the prioritization of openness, responsibility and transparency in the performance of measurements by the infrastructure. It clearly states the nature and purpose of the measurement infrastructure and offers a clearly defined opt-out policy.

6.1.2 Component requirements

Before designing the components themselves, we defined a set of function and performance requirements that they had to meet. These were related to measurement cycle time, memory limitations, processing power limitations, measurement latency, ease of maintenance and expansion potential. Specifically:

- **Measurement cycle time:** The time it took to measure the stable infrastructure and perform reactive and follow-up measurements on attack targets could not exceed certain thresholds. For the Baseline profiler component, the target was scanning the entire stable infrastructure in less than one day. For the Reactive measurement component, the target was scanning the targets of each batch of attacks before the next batch would arrive (roughly 8 minutes).
- **Memory limitations:** The entire infrastructure, including the measurement components, the components responsible for delivering the derivative data stream of attacks on specific hosts and the webserver running the scanning announcement webpage had to be able to run on the same virtual machine. Therefore, memory could not exceed (or even approach) the total available memory pre-allocated upon the host virtual machine's creation.
- **Processing power limitations:** As with memory, the available processing power was limited. The components had to be light enough to co-operate with other processes, as well as all other components of the infrastructure, without taxing the system's Central Processing Unit (CPU).
- **Measurement latency:** Any additional overhead incurred by the libraries, processes and modules performing the measurements of the measurement infrastructure had to be kept minimal. The measurement results had to be as near to real-time as possible.

- **Ease of maintenance:** The components had to be easy to maintain. Coding and commenting standards and clearly defined functions had to be set and followed throughout the components' development.

6.1.3 Measurements

We now examine the measurements the infrastructure performs. The list represents the final set of measurements performed and the reasoning behind their selection. For a complete overview of the measurements considered and the reasoning behind their retention or rejection, please refer to Section 6.2. We decided to set all measurements with a `timeout` value of 10 seconds. This was because setting the `timeout` value higher made the measurement process exceed the desired time thresholds. The following measurements are selected:

- **Establishing host responsiveness and willingness to be scanned via Internet Control Message Protocol (ICMP) pings**: We use ICMP pings to establish that the host is reachable before attempting to connect via Transmission Control Protocol/internet Protocol (TCP/IP) on one of the standardized SMTP ports: `587`, `25`, `465` and `2525`[7][10][27]. This first connection is part of profiling a stable infrastructure, so that we can infer fluctuations in the RTT of Extended HELO (EHLO) and Core SMTP identification command (HELO) handshakes. In short: we try to minimize profiling hosts that are unreachable.
- **Establishing a TCP/IP-based connection on one of the standard SMTP ports:** This is the first part of the core measurement component. We attempt to establish a TCP/IP-based connection on one of the standardized SMTP ports: `587`, `25`, `465` and `2525`[7][10][27]. This allows us to examine service distribution on the MX record IPs.
- **Measuring the RTT of the EHLO/HELO handshake with the connected SMTP host:** This is the second part of the core measurement component. By measuring the RTT for the stable infrastructure over a period of time, we attempt to establish an average for the expected RTT of a EHLO/HELO handshake on a SMTP host, for each protocol version: Extended SMTP (ESMTP), SMTP/Secure Socket Layer (SSL) and SMTP. Comparing these bounds to the RTTs obtained by reactively querying SMTP hosts under attack is the first step towards assessing the impact of DDoS attacks on E-mail service infrastructures.

All ICMP ping packets are sent once. This holds for both the reactive and the baseline components. While we understand that this choice may result in the exclusion of potential hosts due to packet loss, we believe that it is in line with responsible internet citizenship deontology. The reasoning is that we do not want to place undue burden on a host's network by sending multiple ICMP ping requests.
For similar reasons, we record the first successful EHLO/HELO handshake on the host and that handshake only. If we establish a connection to the host and complete a handshake, we do not attempt to establish a connection on other SMTP ports for that host. This decision is made to decrease our components' footprints on the victim hosts' networks. By "complete handshake", we mean the following sequence: a connection is established to the host on a standard SMTP port, an EHLO/HELO handshake is initiated and the host responds with any code except `110` (connection timed out).

6.1.4 Core structure of the components

At its core, each component is a simple module. The module input is a list of MX record IPs For the Baseline Profiler component, the list has the form of a simple `.csv` file containing one per line. For the Reactive Measurement component, the input list is rather more complex. Thus, it is described in detail in Subsection 6.3.2 These IP entries are read and stored in a list, where

each individual host is represented via objects of the custom-built `Host` class. The class models an abstract SMTP host and is comprised of the following fields:

- `ip:` The host's IP address, as derived from the input file.
- `rtt:` The RTT of the EHLO/HELO handshake for the host.
- `port:` The SMTP version running on the host. This value is derived based on which port first responds to a handshake query, and can be one of `587, 465, 25` or `2525`.
- `ehlo_code:` The host's EHLO handshake response code.
- `helo_code:` The host's HELO handshake response code.

For each host, the component first sends an ICMP ping packet to establish responsiveness. If the ping result indicates that the host is unreachable, the observation is logged and the component moves to the next host. If the host is reachable via ping, the host attempts to establish a connection in one of the standard ports by calling the appropriate measurement function. The port query sequence, first to last, is: `587, 25, 465, 2525`. Once a connection is established, the appropriate handshake is performed. The host's response code is stored and the component evaluates it. In case of either a `-1` (connection could not be established) or a `110` (connection timed out), the component proceeds to examine the next port in sequence. For any other code value, the component considers the handshake complete and proceeds to the next host.
The measurements themselves are each performed in autonomous functions, one for each measurement defined in Section 6.1.3. The functions establish a connection to the appropriate port, perform the measurement, record the RTT, the port and, in the case of the handshakes, the response code, and pass the values to each host's appropriate fields as defined in the field list.
After all input hosts have been measured, the results are stored in a `.csv` file. The filename has the format:`<scan type> - <date of scan>`. Each line in the file records a measurement operation on a host and consists of the host's IP address, the port number, the RTT of the operation, the host's EHLO response code and the host's HELO response code.

6.1.5 Notes on the core structure

We keep the codes for the EHLO and HELO handshakes in separate variables. In practice, one of these is always `-1` on output recordings, since the component does not try to connect to the standard ports after one completed handshake. However, the field separation in this instance serves to support the component's modality and reuseability. If future users want to query for both EHLO and HELO handshakes, they need only restructure the core measurement function, without adding additional fields or modifying the host's abstract class.
In case of a timeout (code `110`), the component behaves as though the host were unreachable. In other words, neither the port number nor the RTT are stored. This is a deliberate choice. Both measurement modules strive to emulate a realistic SMTP- sender as much as possible. From a sender's perspective, the reason for which a SMTP session could not be established is not relevant: in case of undelivered mail, it will simply attempt to connect to the intended receiver again (see 2.2.5). However, users who wish for this distinction can modify the core measurement function to account for it.

6.2 Designing the Baseline Profiler Component

This section describes the steps taken and choices made during the design phase of the Baseline Profiler measurement component. Here, we examine the initial approach to designing such

a component, the issues faced along the way, the final version of the component and the reasoning behind said version. Note that any issues and challenges referring to measurement selection, as well as the performance of the core component structure, apply to the reactive component also. This is because we wish to keep the two components as similar as possible. We do this to achieve to ends: have a meaningful comparison platform and facilitate readability and (future) maintenance of the project.
When designing the component, we adhered to the requirements set in Sections 6.1.2 and 6.1.3. Development-wise, we underwent three development cycles. The requirements were met at the end of the third development cycle.

6.2.1 Final component version: optimization and open-source scanning tools

Initially, a sequential version of the component was designed. This version failed to meet the component requirements. Specifically, the component was unable to perform measurements on the provided list of MX Record IPv4 addresses in the desired time of 24 hours or less. Additionally, the resources used by this component version (both memory and processing power) made cohabitation with the other components of the infrastructure impossible.
The second version saw the introduction of parallelization. While this version also failed to meet the component requirements, the improvement over the first was significant. The failure of the second version was again due to prohibitive memory and CPU power consumption.
The third and final version of the component was centered around performance improvements regarding memory consumption and CPU usage of the parallel processing introduced in the second version. The third version also incorporated the use of the `ZMap` open-source internet scanning tool[28]. The reasoning behind selecting `ZMap` is as follows:

- **Speed:** The introduction of `ZMap` saw drastic improvement in executing ICMP ping measurements for the infrastructure. The entire list could be scanned in under 8 minutes with the provided parameters.
- **Customization:** `ZMap` allows for extensive customization in how the ICMP ping measurements are performed. This was particularly useful to us, as it allowed us to customize our scan to comply with responsible internet citizenship. Specifically, the bandwidth that was available to `ZMap` was capped at 4 Megabits per second (MBps). Further, the number of packets that were sent to each host was reduced to one. Additionally, the opt-out list consisting of hosts who were excluded from the scan was also provided to the tool.
- **Reliability:** `ZMap` is a project that has existed for over a decade[28]. Its reliability has therefore been proven repeatedly. This factor was deciding in incorporating the project into our measurement component.
- **Transparency:** `ZMap` is an open-source project, the full repository of which is publicly available[29]. Due to the sensitive nature of our research, it is important to include projects whose source code is verifiable and subject to audits by the wider community.

The performance improvements made in the third version were as follows:

- **The handling of the list was altered:** Already reduced by the initial scan done via `ZMap`, the list is handled by threads in parallel using a custom approach. Specifically, the initial list is split into batches beforehand. The batches are capped at 10000 elements: thus, each thread has at most ten elements to process. After each batch is processed, the results are recorded and the batch is cleared from memory. This ensures that, at most, the initial list is loaded into memory. This decision led to the component taking up roughly 700 Megabytes (MBs) of memory at peak.

- **The time that threads remain idle is minimized:** Splitting up the list into smaller chunks ensures that each thread has fewer tasks to execute. This reduces the time a thread remains idle and speeds up process completion. Although some overhead remains, since results are written to file more frequently, this is still offset by the reduced idleness of the threads.

These performance improvements resulted in achieving significant results: memory consumption dropped to roughly 1.4 Gigabytes (GBs) at peak, while processing time was reduced to roughly 5 hours. These results were deemed acceptable for the system. Thus, the third version of the component was the one selected for profiling the stable infrastructure of 2537381 MX Record IPs on a daily basis.

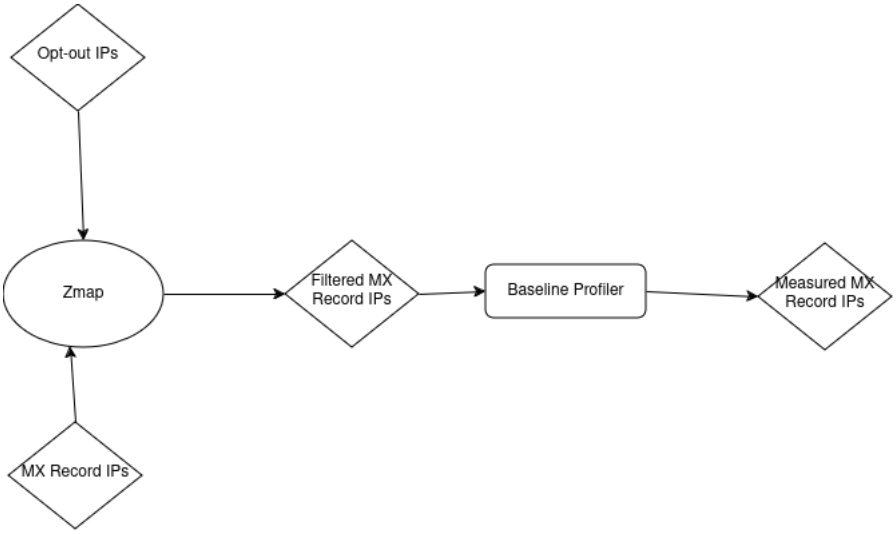


Figure 6.2: High - level overview of daily profiling of MX Record IPs. Figure made using `https://app.diagrams.net/`.

Figure 6.2 presents a high-level overview of the part of the infrastructure performing daily measurements on the list of MX Record IPs: in the schematic, diamond shapes correspond to files, rectangles correspond to programs written by the author and ellipses correspond to external programs and tools. Arrows from files to programs denote input, whereas arrows from programs to files denote output. As we can see, the initial list of MX Record IPs is passed to `ZMap`. Internally, it is cross-referenced with the MX Record IPs in the opt-out list, which are removed. After `ZMap` scans the list, it generates a list of those MX Record IPs that responded to the ICMP ping requests. That list is then fed to the profiling module, which performs the remaining SMTP and ESMTP measurements on the entries. The final output of the module is the daily profile of the stable infrastructure w.r.t RTT and responsiveness.

6.3 Architecture of the Reactive Measurement Component: Challenges and Solutions

This section focuses on explaining the architecture of the Reactive Measurement Component. It provides an overview of the various modules that comprise its structure, the role each module plays in the overall architecture and the design decisions made to arrive at the final component.

6.3.1 Exchanging data via Apache Kafka

In our reactive measurement infrastructure, we used the `Apache Kafka` open-source distributed event streaming platform[30]. We used `Apache Kafka` to receive the derivative data stream of

MX Record IP addresses under attack, discussed in Section 5.2. We also used Apache Kafka to exchange data between the internal modules of the Reactive Measurement Component itself. At its core, Apache Kafka is a Publisher-Subscriber system, as defined in [31]. Under such a system, a set of autonomous processes, defined in [31] as programs in execution, perform computations on data that is shared between them using a common database. The data is updated using event-based coordination: when a process has completed computations, it will update the data which it has processed on the database[31]. Any processes on the system which rely on said computations as input will receive a notification that the relevant data has been updated. After searching the database, they will receive the updated data, usually in some form of key-value pairs, and execute their own computations. When those have been completed, the data space will be updated anew and the cycle will restart[31]. This continuous exchange of data between processes forms so-called data streams[30]: key-value pairs that are inserted in the data space, received by the processes having access to it, updated and re-inserted into the data space.
The key components of a Publisher - Subscriber system are:

- **Producers:** These are processes which execute computations on data and produce output in the form of key-value pairs.
- **Messages:** The key-value pair output of producers. Messages form the data that is exchanged between processes.
- **Consumers:** These are processes which receive incoming mesasges from producers. They then pass the data along to other processes, perform computations on it or a mixture of both.
- **Brokers:** The systems on which messages reside. In the days of virtualization, brokers can differ from the physical systems that comprise a Publisher-Subscriber system: they can, e.g. be deployed in autonomous containers.
- **Topics:** An abstraction layer designed to separate the messages, such that producers and consumers handle only those messages relevant to their operations. Using Topics, the system is partitioned into distinct, autonomous subsystems. In those subsystems, data can be exchanged between producers and consumers in an isolated manner.

Messages are stored on the brokers, in partitioned storage spaces based on the topics managed by the broker. Topics are also used when producers wish to update the system's shared database with new messages. Lastly, topics are used by consumers to decide which new messages they need to process.
The reasons behind selecting Apache Kafka were as follows:

- **Transparency:** Apache Kafka is an open-source platform[30]. This makes it an ideal candidate for this thesis, since it is examined, audited and improved by the software community at large. Given the sensitive nature of the incoming data, such proofs of reliability and openness play a pivotal role when selecting auxiliary software.
- **Portability:** Apache Kafka can be deployed in an entirely containerized manner, using Docker images[32] [33]. Leveraging this capability, the entire measurement component can exist in one physical machine.
- **Compatibility:** The highly abstract nature of Apache Kafka means that applications using the platform can be integrated into existing systems with minimal adjustments. This is an important requirement for making the measurement component future-proof: with few adjustments, it can be part of any measurement infrastructure.

- **Scalability:** `Apache Kafka` is a highly scalable platform. The infrastructure received roughly 400 messages/minute. Kafka can support distribution of tens of thousands of messages per second[30].

These features make `Apache Kafka` the ideal choice as the data distributor of the measurement component. As such, it was the final choice for the measurement infrastructure.

6.3.2 Receiving hosts under attack in near-real time

The first challenge in designing the Reactive Measurement Component was ensuring that a list of MX Record IPs could be derived from Center for Applied Internet Data Analysis (CAIDA)'s captured RDDoS traffic from ongoing DDoS attacks against IPv4 address space[23]. Implementation of the method described in Section 5.2 resulted in obtaining a derivative data stream of victim MX record IPs. Additionally, the stream includes the timestamp of the attack, as recorded by CAIDA's RDDoS traffic detector. Further, the entry includes the domain to which the MX record IP was reverse-resolved. The resulting stream was received by the component in the form of entries in an `Apache Kafka` topic, to which a `Kafka consumer` was subscribed[30].

6.3.3 Processing hosts under attack

The next challenge was designing a high-level architecture for the Reactive measurement component, conforming to the methodology described in Section 5.2. The resulting component needed to achieve the following goals:

- Perform near-real time measurements on the incoming victim hosts.
- Perform follow-up measurements on the victim hosts after initial reactive measurements. Ensure that the follow-up measurement times match the optimal e-mail re-transmission times for MTAs as defined in RFC 5321[10].
- Do not schedule overlapping follow-up measurements on the same host. A twenty-four hour interval (at the minimum) must elapse before scheduling measurements on the same host again.
- Store measurement results with minimal overhead.
- Optimize memory and CPU usage to coexist with the other components of the infrastructure.

To meet these challenges, the reactive measurement component was broken up into three modules. These are as follows:

- **Control module**: This module serves as the coordinator and distributor of tasks between the other modules of the Reactive measurement components. It receives a stream of victim hosts as described in Subsection 6.3.2. It then filters out potential blocked hosts and distributes them to the other modules via `Apache Kafka` streams. It is, in effect, the "brain" of the Reactive measurement component.
- **Reactive measurement module**: This module receives a stream of victim hosts from the control module. It measures the RTT according to the procedure in Subsection 6.1.3 and saves the results to file.
- **Follow-up measurement module**: This module receives a stream of victim hosts from the control module. It measures the RTT according to the procedure in Subsection 6.1.3 and saves the results to file. At the appropriate intervals discussed in RFC 5321 [10], it performs those measurements again.

It is worth noting that the Reactive and Follow-up measurement modules are cloned w.r.t. the measurement code. The exact measurement code is also used by the Baseline profiler component.

6.3.4 Scheduling the follow-up measurements

Devising a means of performing the follow-up measurements on victim hosts was a non-trivial challenge. For this reason, another open-source software was introduced into the infrastructure: `Kafka message scheduler` [34].
`Kafka message scheduler` is a scheduling platform. It allows users to send a `Kafka message` to a specific `Kafka topic` at a specific time. At its core, it is a simple `Kafka consumer - producer` pair: the `consumer` reads incoming `messages` and the `producer` writes said `messages` to the target `topic` at the user-defined time. The payload of the `messages` allows for extensive customization. This allows for additional data, actions and procedures to be scheduled [34].
From an architecture perspective, `Kafka message scheduler` can be viewed as an auxiliary support module to the control module. The control module decides which hosts need to be measured. It also generates the times at which the follow-up measurements will be performed for each host, based on the time that that host was detected by the infrastructure. It then sends the schedule of hosts to `Kafka message scheduler`. At the appointed times, `Kafka message scheduler` forwards the hosts to the follow-up measurement module. Lastly, the follow-up measurement module performs the measurements.
The reasons for selecting `Kafka message scheduler` are as follows:

- **Transparency:** Like `Apache Kafka`, `Kafka message scheduler` is an open-source software. It is regularly maintained and audited by the community. Selecting it allows for openness in the infrastructure's architecture, essential given the sensitive nature of the incoming data.
- **Ease of integration:** `Kafka message scheduler` works with `Apache Kafka topics`. Integrating it requires no changes in the infrastructure w.r.t how data is exchanged between modules.
- **Portability:** `Kafka message scheduler` can be deployed with fully autonomous Docker [32] containers. Thus, integrating it does not compromise the overall infrastructure's portability.
- **Ease of use and monitoring:** `Kafka message scheduler` provides a high-level administration Graphic User Interface (GUI) panel[35]. The panel provides information on current, upcoming and past schedulers in an accessible, easy-to-use manner. This makes monitoring and debugging the infrastructure easy w.r.t. the follow-up measurements. The panel can also be autonomously deployed using Docker[32] containers. Therefore, integrating it maintains the infrastructure's portability.

The features described above make `Kafka message scheduler` the ideal auxiliary software. Thus, it was integrated into the infrastructure.

6.3.5 Maintaining operation of the component

Once initialized, the component had to be operational throughout the measurement period. The processes therefore had to be kept running and, in case of a crash, immediately restarted. To achieve this,`immortal` [36], an open-source process supervisor was introduced into the infrastructure.
At its core, `immortal` is a process monitor. It executes a process as an autonomous UNIX daemon[36]. When a process starts, `immortal` keeps track of its `Process IDentity (PID)` and

restarts the process if it fails. Further logging and monitoring is implemented via a specialized Unix socket, which can be queried to examine process state, terminate the process or view performance [36].
The reasons for selecting `immortal` as the process monitor are as follows:

- **Transparency:** As an open-source software[37], `immortal` is subject to reviews by the software community, much like the other tools integrated in the infrastructure. Thus, integrating it maintains the measurement infrastructure's openness and transparency.
- **Ease of use:** The process monitor is easy to use. Processes can be monitored and started autonomously. Oversight is limited to observing process operation, which is easy to do via `immortalctl`[38].
- **Operating System (OS) support:** `immortal` is OS-agnostic[36]. As such, it can be deployed on any UNIX-based system. Including it does not constrain the measurement component in terms of operating systems.

These features meant that `immortal` fulfilled our criteria. Therefore, it was introduced into the measurement infrastructure.

6.3.6 The Reactive measurement component in operation

The Reactive measurement component is a collection of interdependent processes and modules. To that end, it is important to provide a description of one operation cycle, so as to better examine the function of each module and the relations between modules. To wit:

- The operation cycle begins with the control module receiving a `Kafka message` of hosts under attack at a specific timestamp, according to Section 6.3.2. For each host under attack, the control module:
  - Checks that the host is not in the opt-out list and can be scanned.
  - Checks that the host has not been encountered again in the last twenty-four hours. To do this, the module keeps an archive of all the hosts seen, the time of the first encounter, as well as the different times they were seen. The stable infrastructure queried by the Baseline profiler serves as a starting point, but the archive is updated with any hosts not present in the stable infrastructure at runtime. The archive is checked once every twenty-four hours. For each host, if more than twenty-four hours have passed since the first encounter, the time of first encounter is cleared and the times of repeat encounters are stored to file. This allows hosts to be reactively measured once in a twenty-four hour timespan.
  - For those hosts that can be measured (not in the opt-out list and not encountered again in the last twenty-four hours), the control module creates a `Kafka message` with the host's `IPv4` address and time of first encounter. It then writes that `Kafka message` to a `topic` on which the reactive measurement module is subscribed.In parallel, the control module calculates the follow-up times as per RFC 5321 [10]. It then crafts `Kafka messages` with the host's `IPv4` address, time of first encounter and follow-up measurement times. One `Kafka message` is created for each follow-up measurement time. Lastly, the module writes the `Kafka message` to a `topic` on which `Kafka message scheduler` is subscribed.
- After processing each host under attack as described above, the control module waits for a new batch of hosts under attack.

- When the reactive module receives a batch of hosts under attack from the control module, it checks that each host in the batch is not in the opt-out list. This check is done to account for changes in the opt-out list made after the batch of hosts under attack was received by the control module but before it was processed by it. The module then performs the reactive measurements for each host under attack that is not in the opt-out list. It stores the results to file and waits for the next batch of hosts under attack from the control module.
- When `Kafka message scheduler` receives a message from the control module, it reads the scheduling time for each `Kafka message`. It then crafts a `Kafka message` which contains the host's texttttIPv4 address and time of first encounter. At the scheduled time, it writes that `Kafka message` to a `topic` on which the follow-up module is subscribed.
- The Followup-measurement module performs a similar task to the Reactive measurement module. The only difference is that the module listens for incoming hosts from `Kafka message scheduler`.
- The cycle of operation resumes whenever a new batch of hosts under attack is received by the control module.

Figure 6.3 provides a high-level overview of the reactive measurement component. In the figure, diamonds represent files, rectangles represent processes written by the author and elipses represent external processes and tools.

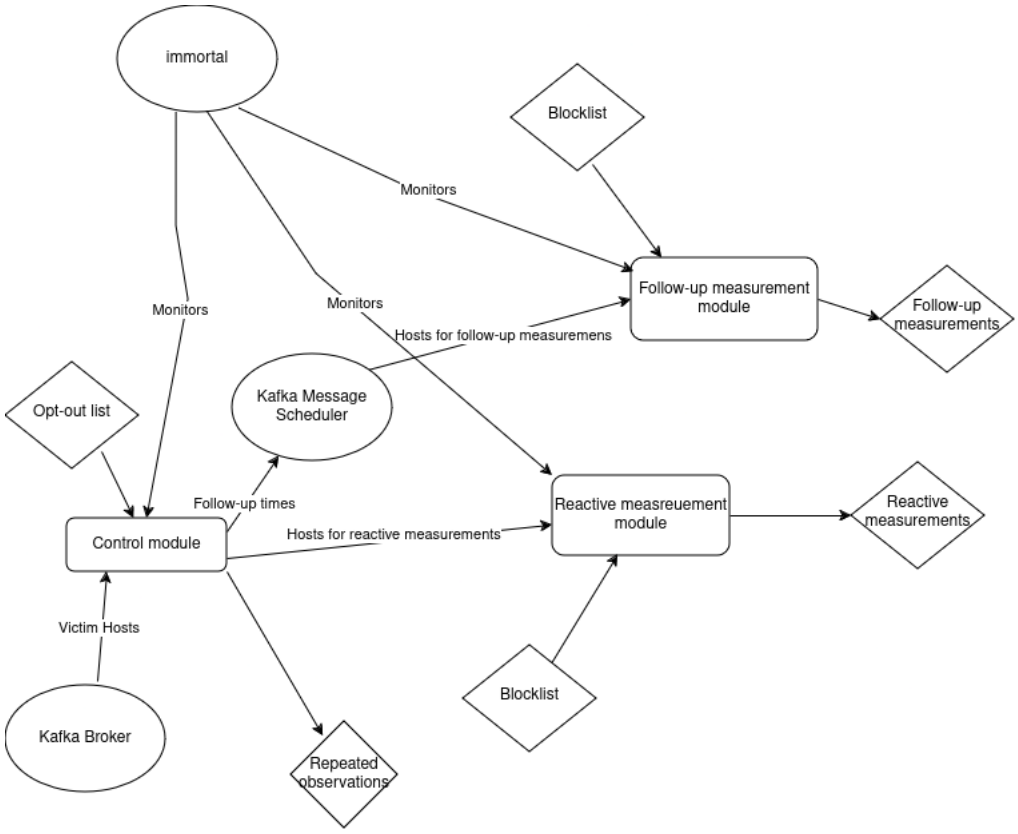


Figure 6.3: High-level overview of the Reactive Measurement Component. Figure made using `https://app.diagrams.net/`.

As we can see, the `Kafka Broker` forwards the victim hosts to the control module. The module parcels out victim hosts to the reactive module using the opt-out list and the hosts that have already been observed the last 24 hours. It also sends the host and follow-up time pairs to `Kafka`

`message scheduler`. The scheduler generates the hosts for follow-up measurements at the appropriate times. The reactive and follow-up measurement modules execute the measurements and store the results. Monitoring of the modules is done by `immortal`.

6.4 Closing Thoughts

The problem of designing a measurement infrastructure to measure the impact of DDoS attacks on mail service infrastructures was non-trivial to solve. From an engineering perspective, a lot of different factors had to be examined at every step. Cohesion was paramount for every component of the infrastructure, and the criteria were not limited to performance: ethical considerations greatly influenced the design process.
To meet these challenges, the measurement infrastructure was broken up into two components: one component performed daily measurements on a stable infrastructure of ̃2.5M IPv4 Addresses. The other module performed reactive and follow-up measurements on a derivative stream of mail hosts under attack, as derived from CAIDA's captured RDDoS traffic from ongoing DDoS attacks against IPv4 address space[23]. To satisfy performance and ethics criteria, a mixture of parallelization techniques and open-source tools was used. The end result was an infrastructure that could successfully operate and perform measurements on the infrastructure: both components worked as intended.

7 ANALYSIS PHASE

This chapter examines the results obtained from the infrastructure's operation. The chapter begins by offering the core terms and concepts that were used to interpret the measurement results. The chapter then gives a brief overview of the observations made from the daily measurements of the roughly 2.5M IPv4 MX Record IPv4 addresses (stable infrastructure). The main part of the chapter, however, is devoted to examining the insights obtained from the operation of the Reactive measurement module: examining the impact on performance of hosts under attack.

7.1 Preamble: Basic concepts and terms

Before examining the impact of DDoS attacks on mail service infrastructures, we define a few sets of basic concepts and terms. These are used throughout the Analysis chapter. Specifically:

- **Stable infrastructure:** We define "stable infrastructure" to be the list of hosts used as input for the daily measurements performed by the Baseline profiler component. This is derived from the list of roughly 2.5M MX Record IPv4 addresses that are the target of the initial Ping ECHO request performed by the component.
- **Attack:** We consider an "attack" to be a distinct entry in the derivative data stream that is used as input for the Reactive measurement module, as defined in Section 6.3.2.
- **Measurement period:** The measurement period is the period in which reactive measurements on attack targets and daily measurements on the stable infrastructure were done. The measurements spanned one calendar month, from 20-03-2024 to 20-04-2024. This amounts to 32 days. During the measurement period, we performed both reactive and stable measurements.
- **Short-term impact:** We consider an attack to have "short-term" impact on the host if, at time of detection, we saw significant performance degradation in the host's RTT, or the host was completely unresponsive to our measurement attempts. However, the effects of an attack with "short-term" impact cannot persist for more than two to three hours after initial detection. We choose the term to account for time elapsed between the time the attack occurred and the time it was detected by our infrastructure.
- **Mid-term impact:** We consider an attack to have "mid-term" impact on the host if, both at time of detection and throughout the followup measurements in the span of one day, we observed significant performance degradation on the host's RTT. If the host was unresponsive to our measurement attempts at time of detection, then it must either have remained unresponsive throughout the follow-up measurements or, in case of recovery, displayed unstable behaviour in the span of one day. We consider "unstable behaviaour" to be, for example, fluctuations in the host's responsiveness or RTT from connection attempt to connection attempt.

7.2 Examining the behaviour of the stable infrastructure

In this section, we provide an overview of how the stable infrastructure performed from our daily measurements. Aside from the opportunity to examine the behaviour of a mail service infrastructure under operation during a period of time, this analysis also serves to determine how suitable the measured infrastructure is for providing a baseline profile for the RTT of potential target hosts. In other words, this part of the analysis serves to determine how stable the space remained during our measurement period.
To that end, we looked at the daily measurement results on the stable infrastructure throughout the time period. The point of interest were Ping request and SMTP measurement responsiveness.
For those hosts that responded to the initial Ping request and for which a EHLO/HELO session was successfully established, we also examine the port distribution: i.e., the port on which a SMTP service was listening at the time of measurement.

7.2.1 Responsiveness to the initial Ping ECHO request

To gauge the responsiveness of the stable infrastructure to the initial Ping ECHO request, we examined the number of hosts from the stable infrastructure compared to the full list of roughly 2.5M MX Record IPv4 addresses. Figure 7.1 shows the percentage of the original list of roughly 2.5M MX record IPv4 addresses that responded to the original Ping request, each day of the measurement period.

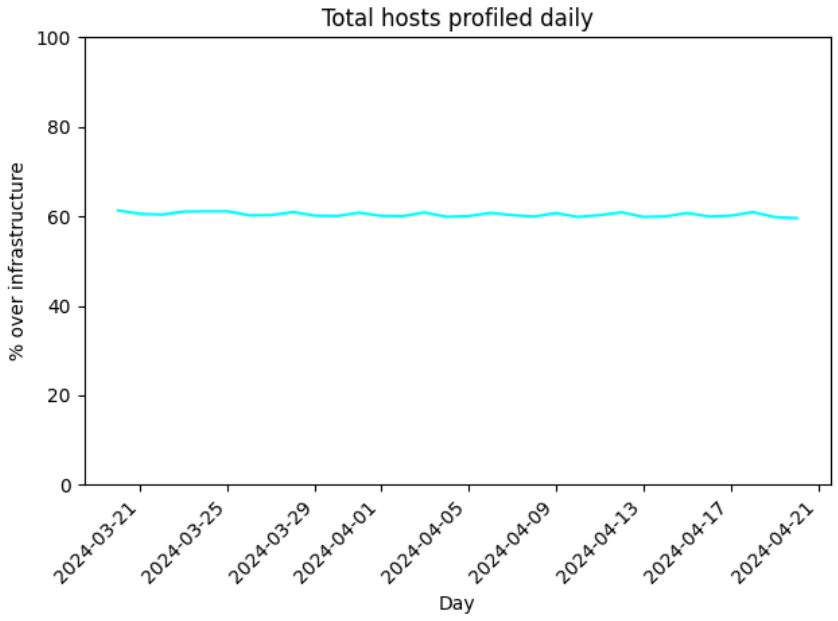


Figure 7.1: Hosts that responded to the initial Ping request. Percentage over the full original list.

As is shown in Figure 7.1, the amount of hosts that respond to the initial Ping ECHO request of the daily measurements remains largely the same, at around 60% of the full list of roughly 2.5M MX Record IPv4 addresses. Any minor fluctuations in the host number are to be expected, given the fact that the initial Ping ECHO request sends only one ICMP Echo packet per host in the list.

7.2.2 Responsiveness to SMTP measurements

After establishing that the number of hosts that responded to the initial Ping request remained roughly the same throughout the measurement period, the next step is examining fluctuations in the number of hosts with which a EHLO/HELO session was successfully established. In other words, out of those hosts that responded to the initial Ping ECHO request, we seek to determine what percentage also responded to the daily SMTP measurements, for each day in the measurement period.

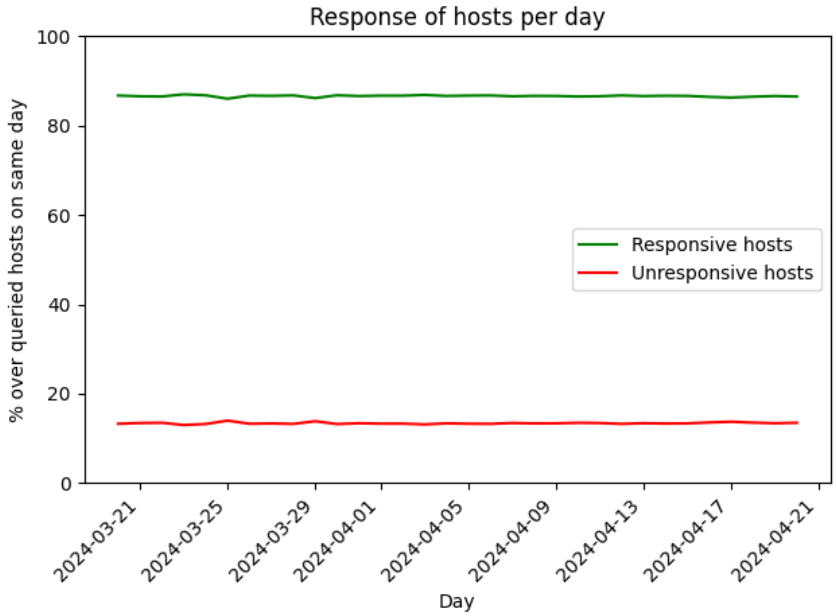


Figure 7.2: Host responsiveness to the SMTP measurements. Percentage over the list of hosts that responded to the initial Ping ECHO request.

As is shown in Figure 7.2, the behaviour of hosts remains predictable throughout the measurement period w.r.t. responsiveness to SMTP measurements. For every day of the measurement period, around 85% of queried hosts remain responsive to the SMTP measurements. In contrast to this, around 15%, despite responsive to the initial Ping ECHO request, were not responsive to the SMTP measurements.
The reasons for this behaviour are numerous. Some hosts may be temporarily inactive mail servers at the time of SMTP measurement, whether as a result of an attack or for benign reasons, such as maintenance. Some hosts may even not be mail servers at all. This happens because, due to MX record misconfiguration, such hosts are erroneously listed as SMTP servers, and identified as such by OpenIntel.

7.2.3 Port distribution

Finally, we examine the port distribution of hosts responsive to the SMTP measurements. Specifically, we examine on which of the queried ports (`587`, `25`, `465` and `2525`) the hosts responded. Table 7.1 shows the port distribution of responsive hosts. For each day in the measurement period, the figure shows the number of hosts that responded to the SMTP measurements via one of ports `587`, `25`, `465` and `2525`. The host numbers are displayed as percentages over the total number of hosts that responded to both the initial Ping ECHO request and the SMTP measurements, for the same day in the measurement period.

Table 7.1: Port distribution of stable infrastructure, for each day in the measurement period.

Port distribution of stable infrastructure				
Day	Port 587	Port 25	Port 465	Port 2525
20-03	89.75 %	9.44 %	0.47 %	0.32 %
21-03	89.66 %	9.51 %	0.48 %	0.33 %
22-03	89.69 %	9.48 %	0.48 %	0.33 %
23-03	89.81 %	9.37 %	0.47 %	0.32 %
24-03	89.81 %	9.37 %	0.47 %	0.32 %
25-03	89.76 %	9.44 %	0.47 %	0.33 %
26-03	89.71 %	9.46 %	0.47 %	0.33 %
27-03	89.71 %	9.47 %	0.48 %	0.33 %
28-03	89.78 %	9.39 %	0.48 %	0.33 %
29-03	89.66 %	9.52 %	0.47 %	0.33 %
30-03	89.72 %	9.46 %	0.48 %	0.32 %
31-03	89.82 %	9.39 %	0.47 %	0.32 %
01-04	89.77 %	9.41 %	0.48 %	0.32 %
02-04	89.74 %	9.44 %	0.48%	0.33 %
03-04	89.82 %	9.37 %	0.47 %	0.33 %
04-04	89.73 %	9.45 %	0.48 %	0.33 %
05-04	89.76 %	9.42 %	0.48 %	0.33 %
06-04	89.87 %	9.31 %	0.47 %	0.32 %
07-04	89.79 %	9.38 %	0.48 %	0.33 %
08-04	89.75 %	9.45 %	0.48 %	0.33 %
09-04	89.82 %	9.36 %	0.48 %	0.32 %
10-04	89.77 %	9.40 %	0.48 %	0.33 %
11-04	89.79 %	9.38 %	0.47 %	0.33 %
12-04	89.94 %	9.25 %	0.47 %	0.33 %
13-04	89.80 %	9.38 %	0.47 %	0.33 %
14-04	89.93 %	9.34 %	0.39 %	0.32 %
15-04	89.94%	9.32 %	0.39 %	0.33 %
16-04	89.84 %	9.41 %	0.4 %	0.33 %
17-04	89.76 %	9.42 %	0.47 %	0.32 %
18-04	89.90 %	9.29 %	0.47 %	0.32 %
19-04	89.83 %	9.35 %	0.47 %	0.33 %
20-04	89.69 %	9.45 %	0.48 %	0.35 %

As is shown in Table 7.1, the vast majority of hosts responded to the SMTP requests via port 587. The second most frequent port is port 25, while ports 465 and 2525 are barely used.
This distribution also shows that the majority of hosts in the stable infrastructure are properly configured. This is both w.r.t the default SMTP port and the usage of TLS encryption [27].

7.3 Examining the impact of attacks

In order to examine the impact of DDoS attacks on mail service infrastructures, we examined the measurement results of the Reactive measurement module throughout the measurement period.
We attempt to examine the effectiveness of the attacks w.r.t. performance degradation. We also examine if the attacks resulted in target hosts becoming completely unresponsive to our measurement attempts.

7.3.1 Preliminary results

To get a better overview of the attack space during the measurement period, we look at the number of attacks over the measurement period. We also examine how many of those attacks left their target hosts unresponsive to our reactive measurement attempts, as well as the port distribution of the target hosts that responded to our reactive measurement attempts.

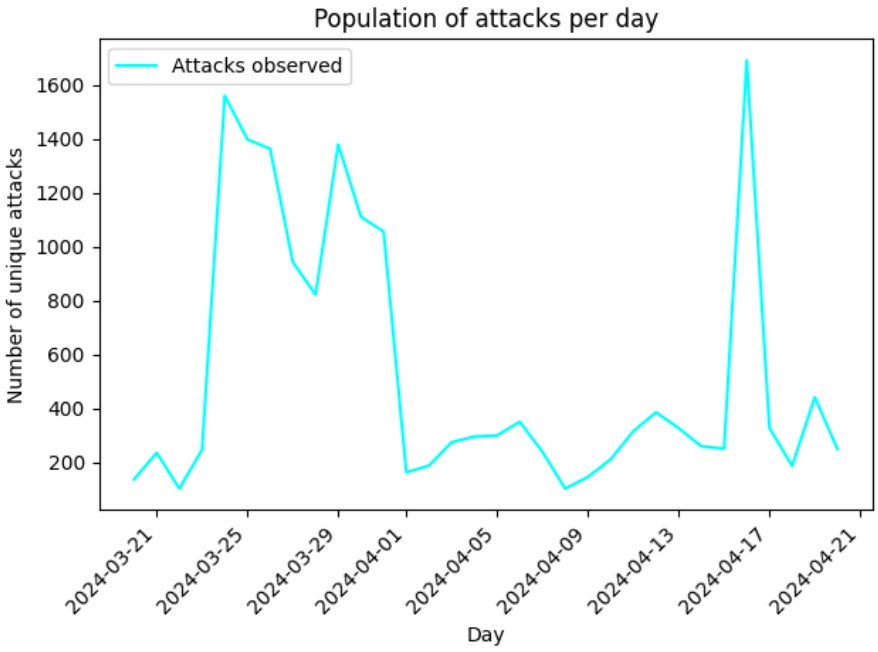


Figure 7.3: Number of attacks detected per day throughout the measurement period.

Figure 7.3 shows the number of unique attacks detected by the measurement infrastructure throughout the measurement period. Overall, between two hundred and four hundred attacks are reported daily. The exception is found in the period from 22-03-2024 to 30-03-2024 and again on 16-04-2024. In both of these cases, there is a surge in the attack number detected by the infrastructure: 7-8 times the usual number of attacks.
With regards to responsiveness of target hosts to the reactive measurements, we looked at how the target host of each attack responded to our reactive measurement, for each attack in the measurement period. A host is considered responsive if they responded to our reactive SMTP measurement.
The response of the host to the initial Ping ECHO request is not as important if the host also responds to the SMTP measurement. The argument for this position is twofold: firstly, a negative response to a Ping ECHO request is not necessarily an indicator of attack impact. It could very well be the result of firewall rules designed to block ICMP packets. Secondly, from a SMTP-sender's point of view, it is the target's response to the EHLO/HELO session that determines success or failure of mail delivery.
By contrast, the response to the initial reactive Ping request becomes more important when defining an unresponsive host. In that case, a response to the initial Ping request, but not the SMTP measurement, most likely indicates an artifact in the incoming derivative data stream. To filter against such artifacts, we consider a host as non-responsive if and only if they have failed to respond to both the initial Ping Echo request and the SMTP measurement made by the Reactive measurement module.

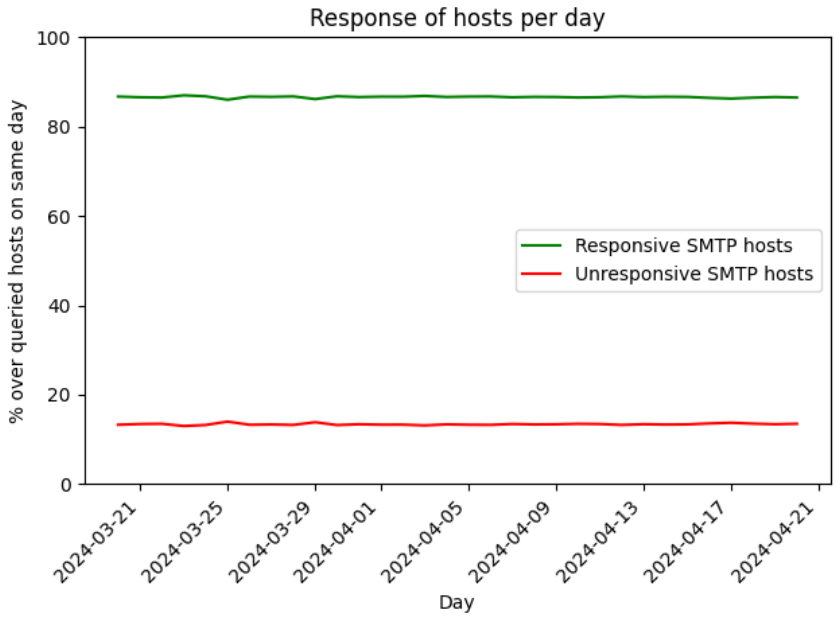


Figure 7.4: Attack target host responsiveness to the reactive measurement, throughout the measurement period. Percentages over all attacks detected for each day.

Figure 7.4 shows the percentage of attacks that left target hosts still able to respond to SMTP measurements. This percentage is calculated for each day in the measurement period. It is compared to the percentage of attacks that left hosts unresponsive. Both are derived as percentages of the total number of detected attacks, for each day in the measurement period. Figure 7.4 shows that, overall, few attacks result in hosts being completely unresponsive. An interesting exception can be found in the attacks observed on 20-03-2024, which have left a significant number of hosts unresponsive.
We also study the victim hosts w.r.t port distribution.

Table 7.2: Port distribution of victim hosts, for each day in the measurement period.

Port distribution of stable infrastructure				
Day	Port 587	Port 25	Port 465	Port 2525
20-03	81.25 %	9.37 %	3.12 %	6.25 %
21-03	83.33 %	7.40 %	5.55 %	3.70 %
22-03	96.87 %	3.12 %	0.00 %	0.00 %
23-03	90.71 %	7.14 %	2.14 %	0.0 %
24-03	86.38 %	11.96 %	0.17 %	1.47 %
25-03	90.22 %	8.62 %	0.00 %	1.14 %
26-03	89.79 %	8.97 %	0.08 %	1.15 %
27-03	87.70 %	10.56 %	1.06 %	0.66 %
28-03	90.44 %	8.98 %	0.28 %	0.28 %
29-03	88.51 %	9.72 %	0.58 %	1.17 %
30-03	85.89 %	12.22 %	0.62 %	1.25 %
31-03	86.68 %	11.60 %	0.32 %	1.39 %
01-04	87.20 %	8.13 %	2.32 %	2.32 %
02-04	80.73 %	13.76 %	2.75 %	2.75 %
03-04	86.95 %	8.69 %	1.24 %	3.10 %
04-04	88.78 %	8.29 %	0.48 %	2.434 %
05-04	77.45 %	9.80 %	11.76 %	0.98 %
06-04	88.20 %	8.96 %	1.41 %	1.416 %
07-04	81.25 %	9.82 %	3.57 %	5.35 %
08-04	58.62 %	13.79 %	20.68 %	6.89 %
09-04	74.28 %	22.85 %	0.00 %	2.85 %
10-04	86.90 %	8.33 %	1.19 %	3.57 %
11-04	68.88 %	19.25 %	2.22 %	9.62 %
12-04	87.15 %	9.63 %	1.37 %	1.83 %
13-04	84.49 %	14.97 %	0.53 %	0.00 %
14-04	79.13 %	20.86 %	0.00 %	0.00 %
15-04	79.20 %	14.85 %	3.96 %	1.98 %
16-04	93.00 %	6.052 %	0.40 %	0.533 %
17-04	78.57 %	16.66 %	3.96 %	0.79 %
18-04	88.00 %	8.00 %	2.00 %	2.00 %
19-04	78.87 %	15.49 %	2.81 %	2.80 %
20-04	87.12 %	6.93 %	4.95 %	0.99 %

As is shown in Table 7.2, the port distribution of the victim hosts did not differ from that of the stable infrastructure.

7.3.2 Short-term impact on hosts

In order to gauge the short-term impact of DDoS attacks on mail service infrastructures, we establish a few prerequisites. These ensure that the results portray how effective DDoS attacks are on SMTP MTAs in the short run.
In order to gauge short-term impact of attacks on mail service infrastructures, we need to examine the attack target's behaviour as seen by the Baseline profiler component throughout the measurement period. We also need that behaviour to be relatively stable. For this reason, short-term impact can only be gauged for hosts that responded to the queries made by the baseline profiler component for more than half of the measurement period: 17 out of 32 days, or roughly 53% of the time.
We add nuance to the analysis by distinguishing between types of impact. We define two.

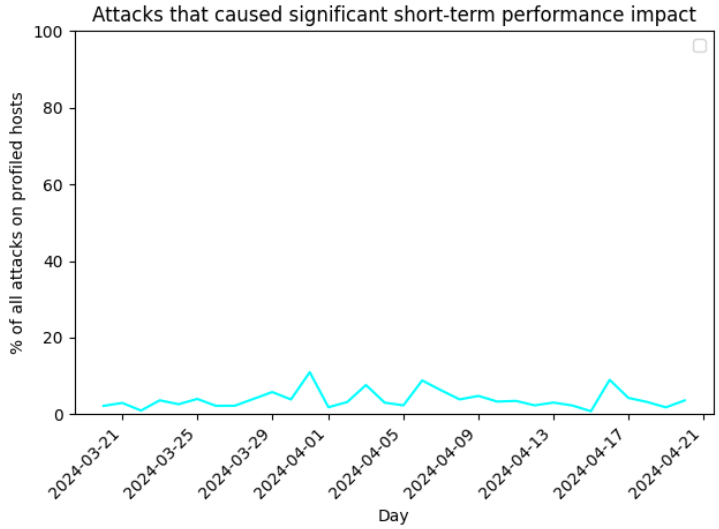
- **Performance Degradation:** The target of the attack was responsive to our reactive SMTP measurements. However, the RTT was significantly higher compared to that observed by the baseline profiler component.
- **Complete unresponsiveness:** The target of the attack was completely unresponsive to our reactive measurements. This means that the SMTP measurement performed by the Reactive measurement module were unanswered from the attack's target.

Finally, the attack's impact on the target host must cause significant performance degradation or complete unresponsiveness. To that end, we set a threshold to consider performance degradation significant. Specifically, we require the RTT of the attack's target when measured by the Reactive measurement module's SMTP measurement to be at least twice that of the average RTT of the attack's target on those days where it responded to the measurement attempts made by the Baseline profiler, throughout the measurement period.
Figure 7.5a shows the percentage of attacks that caused significant short-term performance degradation on their target, for each day in the measurement period. The percentage is derived by comparing the number of attacks that caused significant short-term performance degradation on their target host (RTT at time of detection more than 100% that of average RTT over the measurement period) with the total number of attacks on profiled target hosts on the same day, for each day in the measurement period.
Figure 7.5b shows attacks that caused the target host to be unresponsive to the reactive SMTP measurement by the Reactive measurement module. For each day in the measurement period, these attacks are compared to the total number of attacks on profiled victim hosts on that specific day.
Overall, Figures 7.5a and 7.5b show that the attacks had limited short-term impact on their targets. It seems that, overall, attacks were more effective at rendering hosts completely unresponsive than causing performance degradation. This is especially true for attacks occurring on 21-03-2024: as we can see on Figure 7.5b, more than 40% of attacks on profiled hosts that day caused their targets to be completely unresponsive to the Reactive measurement module.

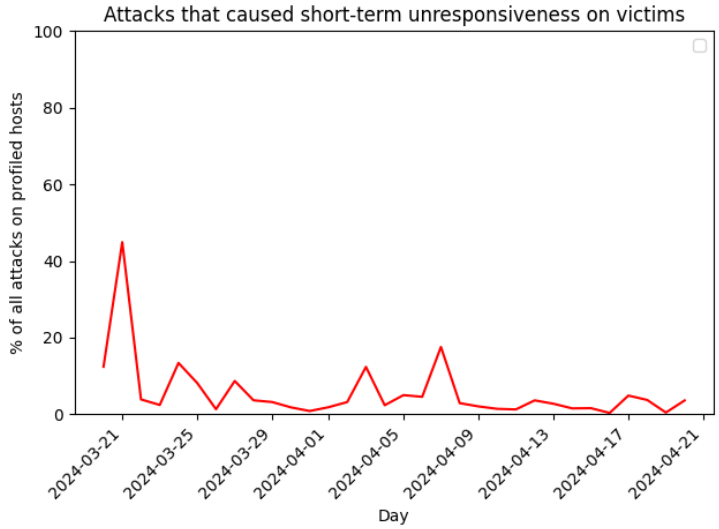
7.3.3 Mid-term impact on hosts

After discussing the short-term effect of attacks on victim hosts, we proceed to examine the effect of those attacks over time. Specifically, we use the measurements performed by the Follow-up measurement module, in accordance with the mail re-transmission times set by RFC 5321[10] to gauge the impact of attacks on mail service infrastructures in the span of 23 hours. Unlike short-term impact, mid-term impact compares the attack's target host's RTT when first measured by the Reactive measurement module with the attack's target host's average RTT as follow-up measurements were being taken by the Follow-up measurement module, in the span of 23 hours after the attack was first detected. However, we still require the target host to be relatively stable throughout the measurement period. Therefore, we only consider hosts that responded to queries made by the Baseline profiler component at least 17 out of 32 days, or roughly 53% of the measurement period.
The same basic criteria apply for gauging mid-term impact as for short-term impact. The key differences are as follows:

- **Performance Degradation:** The attack's target host's average RTT of the measurements performed by the Follow-up measurement module must be greater than twice the attack's target host's RTT of the measurement performed by the Reactive measurement module when the attack was first observed. This holds for both the initial Ping ECHO request and the SMTP measurement. The target host must remain responsive throughout the 23-hour follow-up measurement period.

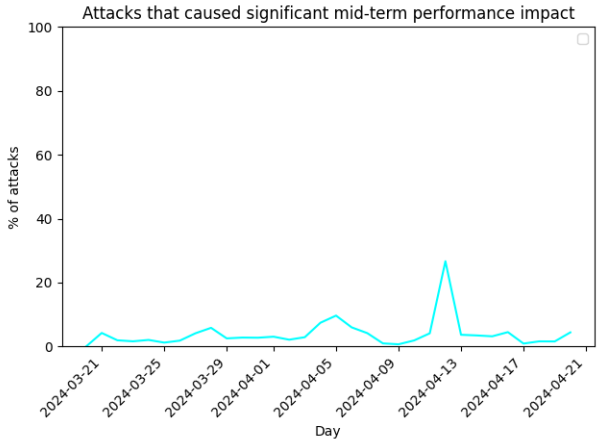


(a) Attacks that caused significant short-term performance impact on target. Percentage over all attacks on profiled victim hosts, for each day in the measurement period.

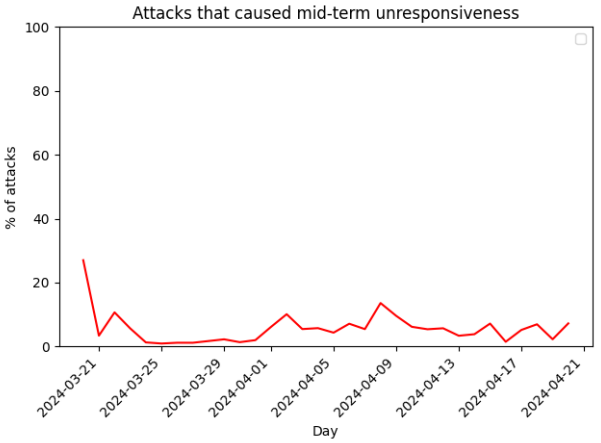


(b) Attacks that caused the target to be unresponsive in the short term. Percentage over all attacks on profiled victim hosts, for each day in the measurement period.

Figure 7.5: Short-term impact of observed attacks on hosts.



(a) Attacks that caused significant mid-term performance impact on target. Percentage over all attacks on all victim hosts, for each day in the measurement period.



(b) Attacks that caused the target to be unresponsive in the course of 23 hours after initially detected. Percentage over all attacks on all victim hosts, for each day in the measurement period.

Figure 7.6: Attacks that caused mid-term impact on target hosts.

- **Complete unresponsiveness:** The attack's target host must have been unresponsive to the Reactive measurement module's measurements, both Ping ECHO and SMTP measurement. It must have remained unresponsive throughout the 23-hour follow-up measurement period.

Based on these criteria, we proceed to examine the mid-term impact of DDoS attacks on mail service infrastructures. Figure 7.6a shows the attacks that caused significant mid-term performance degradation on their target hosts throughout the 23-hour follow-up measurement period. The attacks are displayed as a percentage over the attacks on all victim hosts the same day, for each day in the measurement period.
Figure 7.6b shows attacks that caused their targets to become completely unresponsive throughout the 23-hour follow-up measurement period. The attacks are displayed as a percentage over the attacks on all victim hosts on the same day, for each day in the measurement period.
As is shown in Figures 7.6a and 7.6b, the attacks on SMTP hosts seem to be more effective as time progresses. This holds both for causing complete unresponsiveness, yet not for causing significant performance degradation. In the latter case, an exception can be identified on 2024-04-12, where there is a spike in the number of attacks that caused significant performance degradation on their target host. However, the overall results continue to demonstrate that the infrastructure is resilient to the effects of DDoS attacks.

7.3.4 Host recovery over time

The measurements performed by the Follow-up measurement module offer the opportunity to study how quickly hosts recover after an attack. To examine host recovery, we looked at attack target hosts that were initially unresponsive to the Reactive measurement module's SMTP measurements. We then studied the results of the follow-up measurements for these hosts. We aligned the follow-up connection times, derived from the suggested mail re-transmission times defined in RFC 5321[10], with two-hour recovery windows matching the connection attempts described in RFC 5321[10].

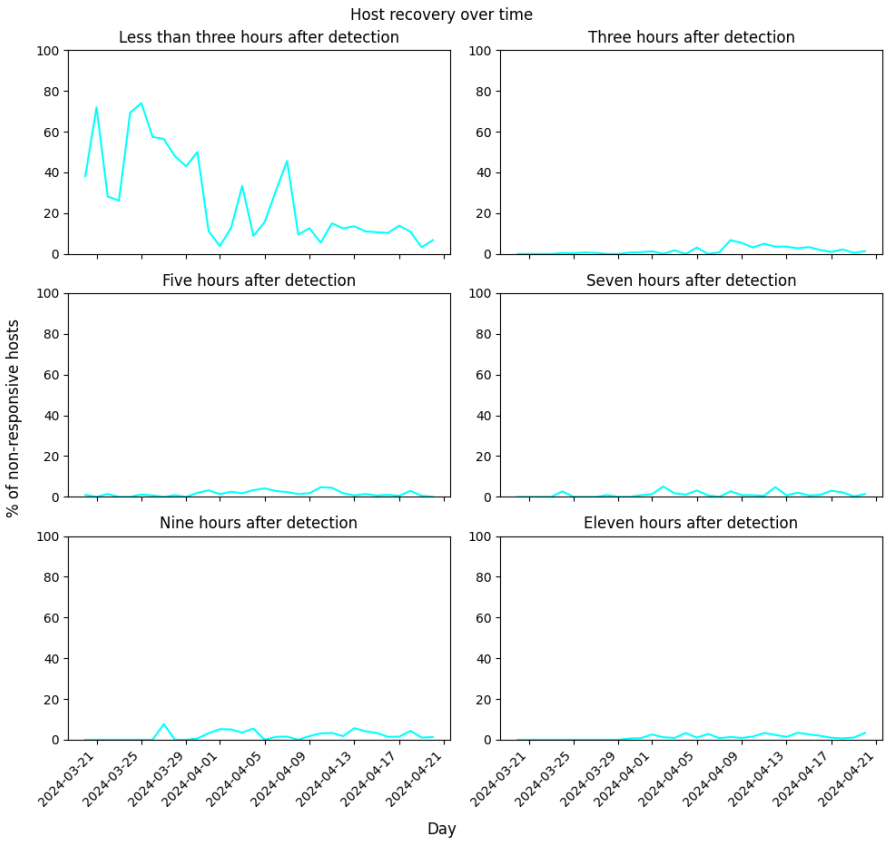


Figure 7.7: Attacks' target hosts' recovery over time. Percentage over all attacks that left their targets unresponsive in the short term, for each day in the measurement period. Shows time windows from less than three hours after attack to eleven hours after attack.

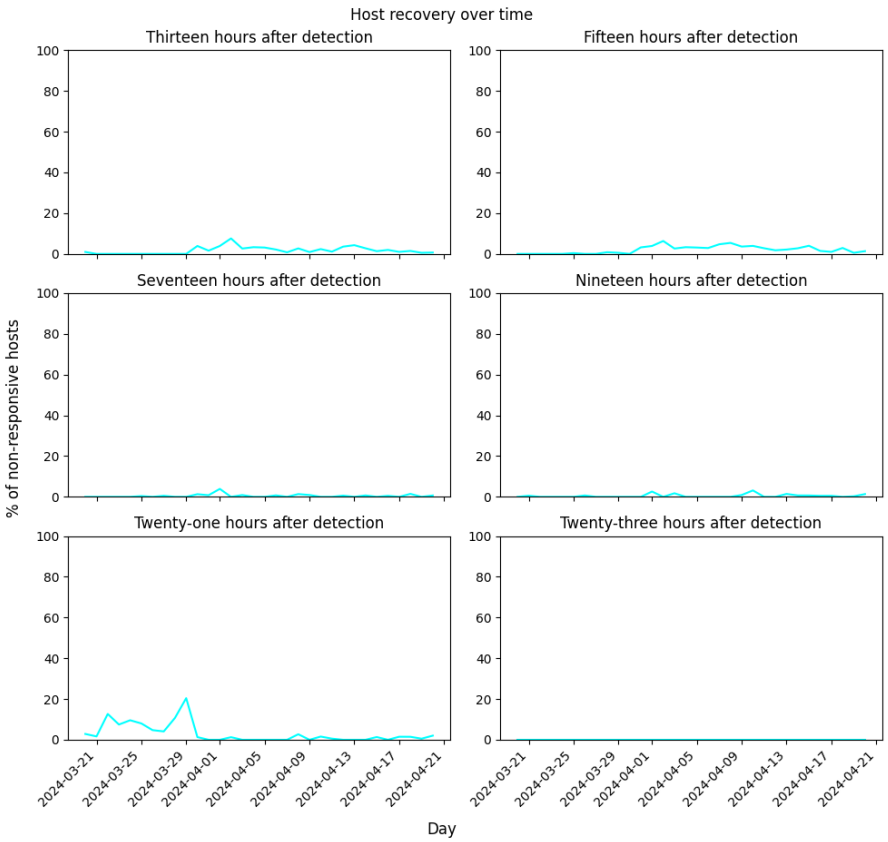


Figure 7.8: Attacks' target hosts' recovery over time. Percentage over all attacks that left their targets unresponsive in the short term, for each day in the measurement period. Shows time windows from thirteen hours after attack to twenty-three hours after attack.

Figures 7.7 and 7.8 show the recovery windows for attack target hosts that were left initially unresponsive as a result of the attack on them. The displayed percentages correspond to attack target hosts that recover within the designated time window compared to all non-responsive attack target hosts on the same day, for the same designated time window, throughout the measurement period.
As is shown by Figures 7.7 and 7.8, for the majority of attacks that left their target host unresponsive in the short-term, the target host managed to recover in less than three hours after the attack occurred. Some exceptions can be seen in some days: 21-03-2024 to 29-03-2024, as well as 16-04-2024. Overall, however the majority of hosts recover within one day after being left initially unresponsive as a result of an attack.
Considering the alignment of the recovery windows with the proposed mail re-transmission attempts defined in RFC 5321 [10], it is safe to conclude that using the proposed mail re-transmission times will result in eventual delivery of mail to its destination. However, delays in mail delivery will occur. For most cases, mail will eventually arrive at its destination with a delay of less than three hours. This statement assumes that the MTA uses the proposed mail re-transmission attempts defined in RFC 5321 [10].
Whatever the case may be, the "Retry" mechanism of MTAs (see Section 2.2.5) appears effective at maintaining operational capability for the mail infrastructure in case of DDoS attacks. While delays will occur, eventually mail will arrive at its intended destination in most cases.

7.3.5 Case study: Attack with significant short-term impact

Having looked at an overview of the impact of DDoS attacks on the detected target hosts throughout the measurement period, we also studied an attack with significant short-term impact on a specific target host. For privacy reasons, the target host's IPv4 address is not mentioned. However, the rest of the data concerning the effect the attack had on the target host is displayed. In order to find the attack with the greatest short-term impact, we examined the total number of attacks that caused significant performance degradation to their target hosts. For each attack, we divided the RTT of the target host in response to the SMTP measurement of the Reactive measurement module with the average RTT of the attack's target host in response to the daily SMTP measurements performed by the Baseline profiler component throughout the measurement period. The resulting number, rounded up, was denoted as each attack's "impact factor". The attack with the highest impact factor was selected for the case study.

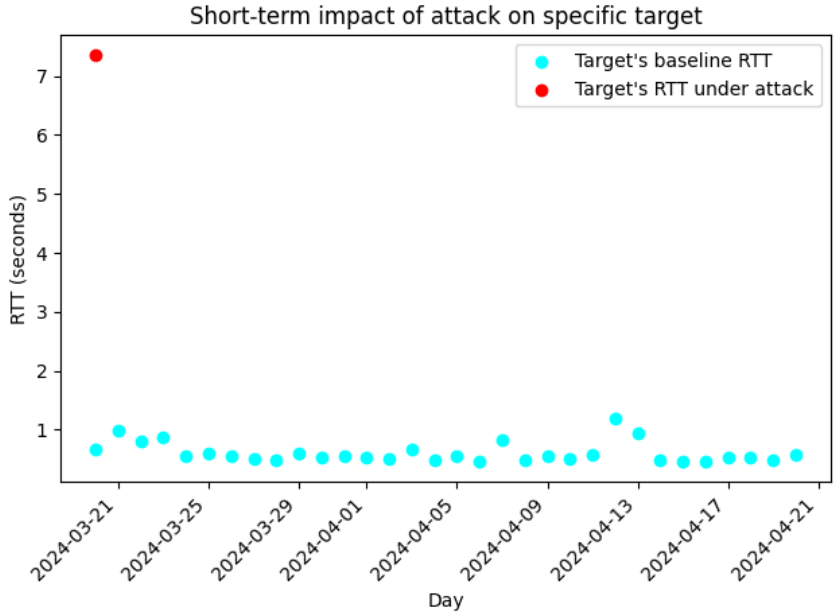


Figure 7.9: Attack short-term impact on specific host. Demonstrated as scatterplot of target host's RTT in response to the Reactive measurement module's SMTP measurement compared to the target host's average RTT over the measurement period.

Figure 7.9 demonstrates the RTT of the attack's target host when first measured by the Reactive measurement module. This RTT is compared to the attack's target host's average RTT in response to daily measurements performed by the Baseline profiler component.
As is shown by Figure 7.9, the attack caused significant performance degradation on the target host, with an impact factor of 7. Additionally, further investigation into the metrics collected by the SMTP measurement of the Reactive measurement module revealed further insight into the nature of the attack.
Specifically, the target host responded to the SMTP measurement via port 2525. The response code to the measurement was 111, denoting that the server refused the connection. These facts, coupled with the large RTT of the measurement, indicate that the host had to utilize the designated SMTP backup port [10] and that the host's resources were so saturated that a simple connection refusal took exceptionally long, indicate strong short-term performance impact as a result of the attack.
To round out the case study, we examined the impact of the attack on the target host within the span of 23 hours, as measured by the Follow-up measurement component. We looked at

the attack's target host's RTT in response to the Follow-up measurement module's SMTP over the $23-$hour measurement period. This was compared to the attack's target host's RTT as in response to the SMTP measurement performed by the Reactive measurement module. This is displayed in Figure 7.10.

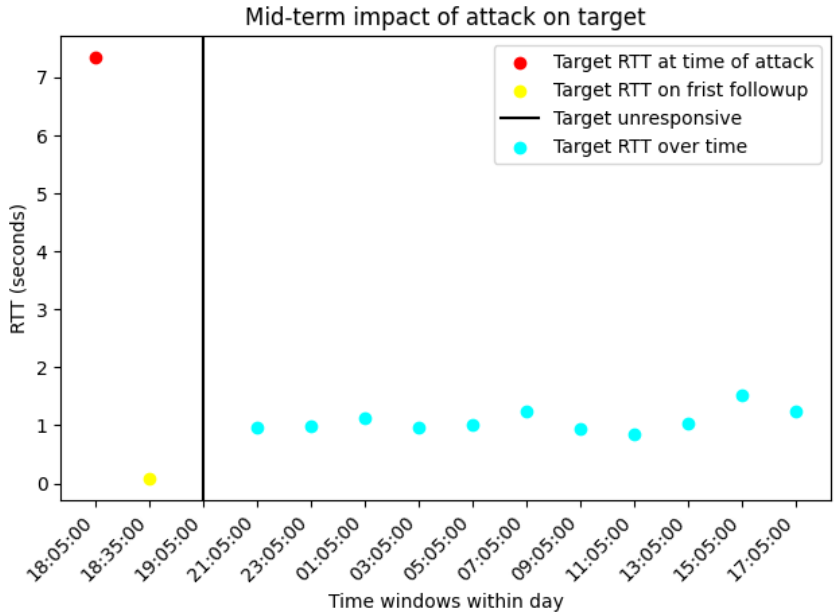


Figure 7.10: Attack mid-term impact on specific host. Demonstrated as comparison of target host's RTT in response to the Reactive measurement module's SMTP measurement with the target host's average RTT over the 23-hour follow-up measurement period.

As Figure 7.10 shows, the attack's target host's RTT over the $23-$ hour measurement period was initially very low. It became zero at the second follow-up connection attempt, then moved to around 0.9 seconds and remained roughly stable throughout the rest of the follow-up period. This behaviour becomes clear when looking at the port via which the attack's target host responded to the follow-up measurements. At the first follow-up connection attempt, the port was listed as 2525, and the response code was 111, which denoted that the target host refused the incoming connection. The RTT of the second follow-up measurement was 0.0, indicating that the target host was unresponsive to the follow-up measurement. For all subsequent follow-up measurements, the port was listed as 587 and the response code was listed as 250, denoting a successful SMTP EHLO session[5].
These insights add nuance into the nature of attack impact on hosts. In this particular case, assuming that SMTP-senders to the host used the standard e-mail re-transmission times as defined in RFC 5321[10], there was a delay of two and a half hours while the target host shut down the service listening on the backup port and switched to the service listening on the main port.

7.3.6 Case study: Attack with significant 12-hour impact

Another interesting case study concerned a target host that was left unresponsive by the attack targeting it for a significant period of time. Specifically, we examined an attack that occurred on 27-03-2024, at 23:40.
We began by examining the target host's response to the measurements of the Baseline profiler component over the measurement period. We further looked at the attack's short-term impact on the target host. The host's behaviour in both cases is presented on Figure 7.11.

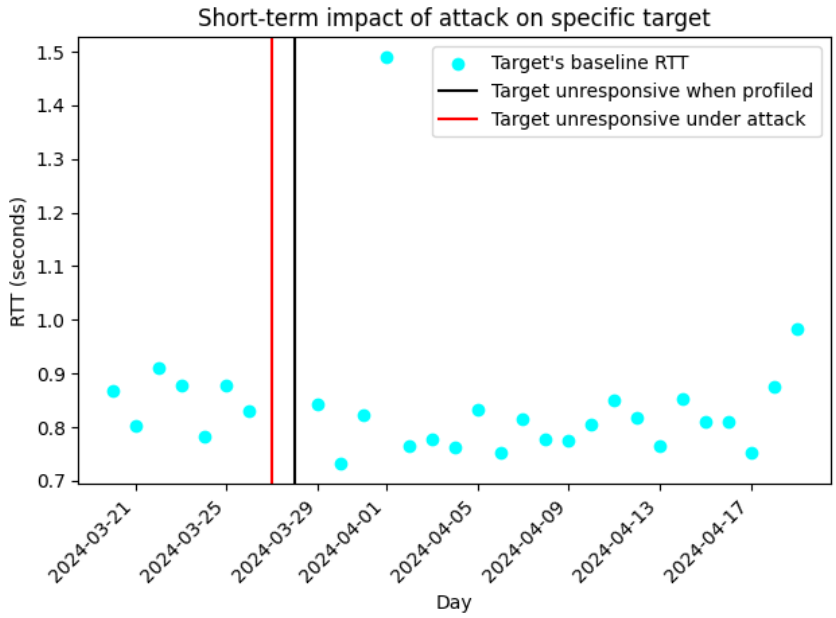


Figure 7.11: Attack short-term impact on specific host. Demonstrated as scatterplot of target host's RTT in response to the Reactive measurement module's SMTP measurement compared to the target host's average RTT over the measurement period.

As is shown by Figure 7.11, the target host displays overall stable behaviour. The target host was responsive to the measurements made by the Baseline Profiler component throughout the measurement period. The sole exception is 28-03-2024. This, however, is a direct result of the attack. As stated, the attack occurred late on 27-03-2024, and the host did not show signs of recovery until roughly 12 hours later. Therefore, the target host was also unresponsive to the measurement attempt made by the Baseline Profiler component the following day, which occurred some time around 05:00.
The host also displays a fluctuation in RTT on 31-03-2024. However, no attack on that host was detected by our measurement infrastructure, neither on that day nor on 30-03-2024. This fact, coupled with the relatively low increase in the target host's RTT, strongly indicates that the delay was probably due to either network traffic fluctuations or a benign increase in the target host's request load.
We further examined the host's response to the follow-up measurements made by the Follow-up measurement module in a span of 23 hours after initial detection of the attack. Figure 7.12 demonstrates this behaviour.

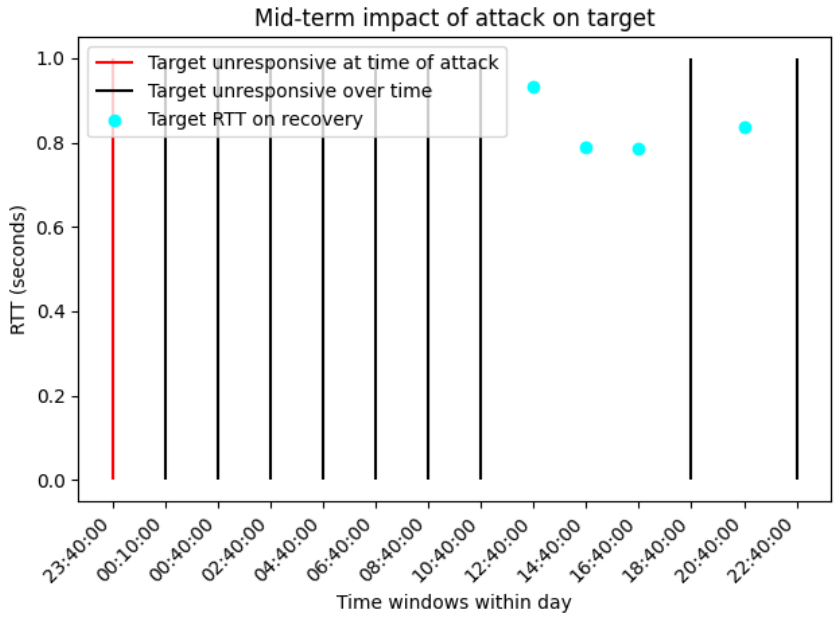


Figure 7.12: Attack mid-term impact on specific host. Demonstrated as comparison of target host's RTT in response to the Reactive measurement module's SMTP measurement with the target host's average RTT over the 23-hour follow-up measurement period.

As is shown by Figure 7.12, the attack had significant impact on the target host. Following initial detection, the target host remained unresponsive for a period of roughly twelve hours. It then briefly for a period of roughly four hours, before becoming unresponsive again for another period of two hours. After recovering again for a period of two hours, the host then became unresponsive again to the final follow-up measurement taken.
It is worth noting that the host remained unresponsive to both the initial Ping ECHO request and the SMTP measurement performed by the Follow-up measurement module at each follow-up measurement. This reinforces the belief that the host's unreliable behaviour was a result of the attack.
This example illustrates the nuanced effect of a DDoS attack on a target host. While the host briefly recovered after a twelve-hour period of unresponsiveness following the detection of the attack, its behaviour remained unstable for the rest of the follow-up measurement period. The subsequent daily measurements taken by the Baseline Profiler component for that target host do seem to illustrate a stable behaviour. Therefore, the host did seem to stabilize eventually. However, the fact remains that, for a period of roughly twelve hours, the host was severely impacted by the attack. Additionally, recovery, when effected, did not immediately result in fully stable behaviour.

7.3.7 Case study: Attack with significant 23-hour impact

The final case study showcases how effective an attack can potentially be at disrupting mail traffic. We examined an attack that left the target host completely unresponsive for a period of roughly 23 hours.
We first examined the attack's short-term impact on the target host. We also examined the host's response to the measurements performed by the Baseline Profiler component throughout the measurement period. The host's behaviour, on both cases, is displayed in Figure 7.13.

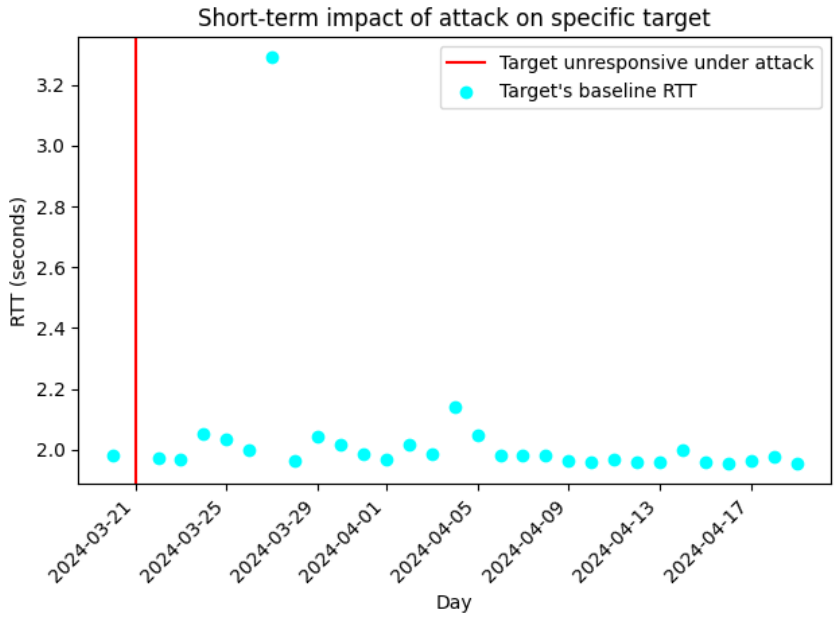


Figure 7.13: Attack short-term impact on specific host. Demonstrated as scatterplot of target host's RTT in response to the Reactive measurement module's SMTP measurement compared to the target host's average RTT over the measurement period.

As Figure 7.13 shows, the target host displayed overall stable behaviour throughout the measurement period. The target host was only unresponsive to the Baseline Profiler component's measurements on 21-03-2024. This behaviour is a direct result of the attack on the target host. The attack occurred on 21-03-2024, at 05:45. The daily measurement for that day was taken at around 06:45. Since the host did not recover until roughly 23 hours later, its unresponsiveness to the Baseline Profiler component's measurement for that day is expected.
We further examined the host's response to the follow-up measurements made by the Follow-up measurement module in a span of 23 hours after initial detection of the attack. Figure 7.14 demonstrates this behaviour.

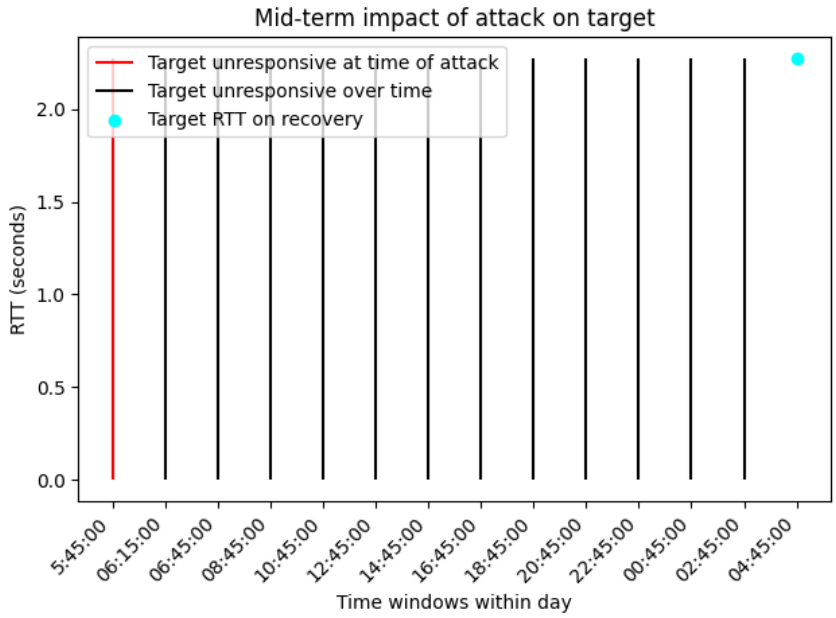


Figure 7.14: Attack mid-term impact on specific host. Demonstrated as comparison of target host's RTT in response to the Reactive measurement module's SMTP measurement with the target host's average RTT over the 23-hour follow-up measurement period.

As is shown by Figure 7.14, the attack had significant mid-term impact on the target host. Following initial detection, the target host remained unresponsive for almost the entire Follow-up measurement period. It only responded to the final measurement taken, at the 23-hour mark since initial attack detection.
It is worth noting that the host remained unresponsive to both the initial Ping ECHO request and the SMTP measurement performed by the Follow-up measurement module at each follow-up measurement. This reinforces the belief that the host's unreliable behaviour was a result of the attack.
This case study demonstrates the potentially severe impact of an attack on the infrastructure. Assuming every MTA connecting to the target host used the e-mail re-transmission times recommended by RFC 5321[10], this would mean that all mail forwarded by the target host would arrive at it's destination roughly one day later than expected. Considering the importance of mail in business, government and private life alike, this delay is far from insignificant.

7.3.8 Contextualizing impact

At first glance, it seems that DDoS attacks do not have a significant effect on mail service infrastructures. As we have seen, whether concerning short-term or mid-term impact, significant performance degradation or complete unresponsiveness, the mail sphere seemed to be able to largely remain unaffected. Even for those attack targets that suffered either significant performance degradation or complete unresponsiveness, the presence of the "Retry" mechanism 2.2.5 ensured that, eventually, they would start receiving and forwarding mail again.
However, mail is unique among internet communication protocols in that it is very much context-sensitive. In other words, the content of mail plays a significant role in gauging the seriousness of each mail exchange, and therefore the urgency of mail delivery.
In addition to this, e-mail exchange is also the main communication tool of private and public institutions and businesses[2]. This fact lends e-mail an official character when it comes to communication. This in turn reinforces the belief that the urgency with which a piece of e-mail

must be delivered to its destination, as well as the need for it to arrive safely, is largely dependent upon the content of said piece of e-mail.
To put things in perspective, it may be worth revisiting the case study of the attack with significant short-term impact on a specific target host. As we saw, that host became unresponsive for a short period of roughly two hours after the attack was first observed, assuming usage of the recommended e-mail re-transmission times defined in RFC 5321 [10]. This would mean that e-mail being delivered to and by that host would arrive at its intended destination two hours later than expected, assuming all other MTAs in the delivery chain operated normally at that period. A delay of two hours in delivery may not seem significant if the content of e-mail were, for instance, weekly meeting invitations for a small business. At worst, action on those meeting invitations is delayed by one day, if the delay were to happen at the end of the working day.
However, a delay of two hours in e-mail delivery becomes much more significant if the content of e-mail were patient exam results, prescription orders or other similar, time-sensitive matters. In these cases, such a delay could have serious consequences for the health of the individuals concerned and the operation of the organizations involved.

7.4 Closing thoughts

In this chapter we examined the impact of DDoS attacks on e-mail service infrastructures. We established that the stable infrastructure measured daily by the Baseline profiler component throughout the measurement period displayed consistent behaviour w.r.t. responsiveness to the initial Ping ECHO measurement and the SMTP measurement of the component.
We further examined the short-term and mid-term impact of DDoS attacks on their target hosts, as those attacks were detected and measured by our infrastructure. The outcome of this analysis was that the e-mail service infrastructure is overall resilient against DDoS attacks. This was true both for attacks that caused significant performance degradation on their target hosts and for attacks that caused their targets to become completely unresponsive to our measurement infrastructure.
This analysis also showcased the role of MTAs' "Retry" mechanism 2.2.5 in maintaining the e-mail infrastructures' operational capability in the event of attacks. While performance degradation was, in some cases, observed over time after the attacks were initially detected, the majority of targets remained operational. In the case of attacks that left their target hosts completely unresponsive, recovery was observed within two hours after the attack was detected in the majority of cases, and in almost all cases the target hosts eventually recovered within the span of one day.
Aside from examining the overall effects of attacks on their target hosts, we selected the attack that resulted in the most severe short-term performance degradation for its target host and observed the target host's behaviour. The host's behaviour was observed both in response to our reactive measurements and in response to the $23-$hour follow-up measurements performed by our infrastructure. This case study shed some light on the nuances of how an attack is viewed by the e-mail infrastructure, in terms of host response, delay incurred, service that responded to the measurement infrastructure and measurement response code.
Overall, the e-mail infrastructure appears resilient to the effects of DDoS attacks. Further, operation is maintained in the face of those attacks. However, e-mail content is important when gauging attack impact. E-mail is a critical piece of infrastructure in the modern world, and care must be taken to ensure that e-mail exchange continues uninterrupted and with minimal delays.

8 CONCLUSIONS AND FURTHER RESEARCH

This chapter summarizes the conclusions reached by the research performed in this thesis. It further outlines a set of key areas which were highlighted during research. Additionally, it outlines a set of potential recommendations to administrators of SMTP hosts who wish to further enhance their host's protection against DDoS attacks.

8.1 Directions for Future Research

During our research, we identified a set of questions. While these fall outside the scope of the present thesis, they can be used as starting points for further research. These are as follows:

- **Examining potential exploits of the "Retry" mechanism2.2.5:** Our research indicates that the "Retry" mechanism 2.2.5 employed by MTAs is proven effective at maintaining e-mail infrastructures' operation after an attack. However, this mechanism also presents itself to exploitation. By pinpointing a target host's "Retry" windows2.2.5, the malicious actor can potentially synchronize attacks with the windows, ensuring that the mail queue remains at capacity and congestion is never relieved.

- **Gauging the impact of DDoS attacks on end users:** We focused on gauging the impact of DDoS attacks on MTAs. However, it would be interesting to determine the impact of such attacks on the end users. Leveraging the "Priority-based approach" developed by Liu et al.[3], it is possible to map victim MTAs to domains, as well as gauge mail service provisioning for those domains. Using that information, it is further possible to identify additional MX IP addresses and determine what part of a domain's service area was effected by an attack.

- **Using the commonly accepted 30-second timeout when establishing SMTP connections:** Our infrastructure used a 10-second timeout when establishing SMTP connections to deal with hardware constraints. However, we believe that using the more accepted 30-second timeout would yield more nuance in the results, and allow for more generous thresholds to determine impact.

- **Establishing a controlled, miniature infrastructure:** As stated in Section 5.3, a key limitation of the measurement infrastructure is that it does not account for the state of the various SMTP-senders to the attack targets when the attacks are observed. This obscures the behaviour of a large part of the SMTP infrastructure when faced with DDoS attacks. However, the measurement infrastructure could easily be set to observe a controlled environment: for instance, a set of researcher-controlled SMTP hosts. Such hosts would form a small-scale version of the examined infrastructure, with the researchers also acting as administrators. Using this controlled environment, researchers could bypass the limitations of this methodology and, using the measurement infrastructure with very minor modifications, perform more in-depth analysis of the behaviour of SMTP infrastructures when faced with DDoS attacks.

8.2 Potential recommendation for SMTP host operators

As a direct result of this research, we have identified a potential exploit in setting a static Retry window size when configuring MTAs. Therefore, it may be of interest to operators to consider an alternate approach when setting the Retry window size for the MTAs under their responsibility. To wit, setting a dynamic, randomly-generated window size.
Setting the window size in a dynamic, randomly-generated manner would enhance the host's resilience to DDoS attacks that aim to exploit the "Retry" mechanism to maximize resource saturation. By keeping the window size variable in this way, any would-be attacker trying to exploit it would have to effectively establish a prediction mechanism for the (pseudo)random generator determining the window size. With sufficiently secure generations present, this quickly becomes infeasible.

8.3 Conclusions

In this thesis, we sought to measure the impact of DDoS attacks on mail service infrastructures. We sought to examine how widespread these attacks are, and their effectiveness in disrupting mail exchange. We further sought to understand the role that the e-mail's re-transmission mechanism, as defined in RFC 5321[10], plays in relation to these attacks.
To achieve these goals, we leveraged the approach used by R. Sommese et al. to measure the impact of DDoS attacks on DNS infrastructures[6] to design a measurement infrastructure to perform reactive measurements on a set of hosts under attack, over a period of 32 days, from 20-03-2024 to 20-04-2024. To derive the set of hosts under attack, we relied on CAIDA's RDDoS traffic from ongoing DDoS attacks against IPv4 address space[23]. To examine the role of MTAs' re-transmission mechanism, as defined in RFC 5321[10], in maintaining operation after an attack had occurred, we performed follow-up measurements on each attack's target host for a period of 23 hours after the attack was originally detected.
In that same period, we performed daily measurements on a stable infrastructure, consisting of roughly 2.5M MX Record IPv4 addresses, leveraging the method developed by R. Sommese et al[6]. Using the attacks' target hosts'RTT during the longitudinal measurements and comparing it to the attacks' target hosts'RTT when the attack was detected, we established a threshold for determining an attack's impact on the host upon detection.
A similar approach was used to determine the attack's impact on its target in the span of 23 hours following initial detection. Using this information, we were able to infer the role of the e-mail's re-transmission mechanism, as defined in RFC 5321[10], in maintaining operational capability after an attack.
The results of our research show that, overall, the e-mail infrastructures are resilient against DDoS attacks. Further, the e-mail's re-transmission mechanism, as defined in RFC 5321[10], proves effective at maintaining operational capability after an attack.
To further contextualize the potential impact of DDoS attacks on mail service infrastructures, we examined three case studies of attacks with significant impact on their target hosts. The results show that attacks can cause hosts to become unresponsive for a significant amount of time. Further, when recovery does occur, the target host's behaviour can remain unstable for some time.
These case studies prove the need for care when designing e-mail infrastructures. While the infrastructure is resilient against disruptions caused by DDoS attacks, the context-sensitive nature of mail means that an attack can have potentially catastrophic consequences. Severe measures must be taken to ensure that e-mail arrives at its intended destination reliably and with minimal delays.

REFERENCES

[1] Craig Partridge. The technical development of internet email. *IEEE Annals of the History of Computing*, 30(2):3–29, 2008.

[2] Michael Still and Eric C McCreath. Ddos protections for smtp servers. *International Journal of Computer Science and Security (IJCSS)*, 4(6):537, 2011.

[3] Enze Liu, Gautam Akiwate, Mattijs Jonker, Ariana Mirian, Stefan Savage, and Geoffrey M Voelker. Who's got your mail? characterizing mail service provider usage. In *Proceedings of the 21st ACM Internet Measurement Conference*, pages 122–136, 2021.

[4] Nazrul Hoque, Dhruba K Bhattacharyya, and Jugal K Kalita. Botnet in ddos attacks: trends and challenges. *IEEE Communications Surveys & Tutorials*, 17(4):2242–2270, 2015.

[5] Jon Postel. Simple mail transfer protocol. Technical report, 1982.

[6] Raffaele Sommese, KC Claffy, Roland van Rijswijk-Deij, Arnab Chattopadhyay, Alberto Dainotti, Anna Sperotto, and Mattijs Jonker. Investigating the impact of ddos attacks on dns infrastructure. In *Proceedings of the 22nd ACM Internet Measurement Conference*, pages 51–64, 2022.

[7] Paul Mockapetris. Domain names-concepts and facilities. Technical report, 1987.

[8] Craig Partridge. Rfc0974: Mail routing and the domain system, 1986.

[9] J Myers. Smtp service extension for authentication. Technical report, 1999.

[10] John Klensin. Simple mail transfer protocol. Technical report, 2008.

[11] Randall Gellens and J Klensin. Message submission for mail. Technical report, 2011.

[12] Rob Siemborski and Alexey Melnikov. Smtp service extension for authentication. Technical report, 2007.

[13] Jari Arkko, Brian Trammell, Mark Nottingham, Christian Huitema, Martin Thomson, Jeff Tantsura, and Niels ten Oever. Considerations on internet consolidation and the internet architecture, 2019.

[14] C Bommelaer de Leusse and Carl Gahnberg. The global internet report: Consolidation in the internet economy. *Internet Society*, 2019.

[15] Petros Gigis, Matt Calder, Lefteris Manassakis, George Nomikos, Vasileios Kotronis, Xenofontas Dimitropoulos, Ethan Katz-Bassett, and Georgios Smaragdakis. Seven years in the life of hypergiants' off-nets. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*, pages 516–533, 2021.

[16] Christos Douligeris and Aikaterini Mitrokotsa. Ddos attacks and defense mechanisms: classification and state-of-the-art. *Computer Networks*, 44(5):643–666, 2004.

[17] Esraa Alomari, Selvakumar Manickam, Brij Bhooshan Gupta, Shankar Karuppayah, and Rafeef Alfaris. Botnet-based distributed denial of service (ddos) attacks on web servers: classification and art. *arXiv preprint arXiv:1208.0403*, 2012.

[18] Cynthia Dhinakaran, Cheol-Joo Chae, Jae-Kwang Lee, and Dhinaharan Nagamalai. An empirical study of spam and spam vulnerable email accounts. In *Future generation communication and networking (fgcn 2007)*, volume 1, pages 408–413. IEEE, 2007.

[19] CVE-2014-0160. Available from MITRE, CVE-ID CVE-2010-0024., April 2010.

[20] Karyn Benson, Alberto Dainotti, Kc Claffy, Alex C Snoeren, and Michael Kallitsis. Leveraging internet background radiation for opportunistic network analysis. In *Proceedings of the 2015 Internet Measurement Conference*, pages 423–436, 2015.

[21] Alberto Dainotti, Karyn Benson, Alistair King, KC Claffy, Michael Kallitsis, Eduard Glatz, and Xenofontas Dimitropoulos. Estimating internet address space usage through passive measurements. *ACM SIGCOMM Computer Communication Review*, 44(1):42–49, 2013.

[22] Andra Lutu, Marcelo Bagnulo, and Olaf Maennel. The bgp visibility scanner. In *2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 115–120. IEEE, 2013.

[23] Ucsd network telescope daily randomly and uniformly spoofed denial-of-service (rsdos) attack metadata. https://www.caida.org/catalog/datasets/telescope-daily-rsdos/.

[24] David Moore, Colleen Shannon, Douglas J Brown, Geoffrey M Voelker, and Stefan Savage. Inferring internet denial-of-service activity. *ACM Transactions on Computer Systems (TOCS)*, 24(2):115–139, 2006.

[25] Openintel: Active dns measurement project. https://openintel.nl/.

[26] The ucsd network telescope. https://www.caida.org/projects/network_telescope/.

[27] Keith Moore and Chris Newman. Cleartext considered obsolete: Use of transport layer security (tls) for email submission and access. Technical report, 2018.

[28] The zmap project. https://zmap.io/.

[29] The zmap project: Github repository. https://github.com/zmap/zmap.

[30] Apache kafka main page. https://kafka.apache.org/.

[31] Maarten Van Steen and Andrew S Tanenbaum. *Distributed systems*. Maarten van Steen Leiden, The Netherlands, 2017.

[32] Docker main page. https://www.docker.com/.

[33] Apache kafka documentation. https://kafka.apache.org/documentation/.

[34] Kafka message scheduler github page. https://github.com/etf1/kafka-message-scheduler.

[35] Kafka message scheduler admin gui github page. https://github.com/etf1/kafka-message-scheduler-admin.

[36] immortal main page. https://immortal.run/.

[37] immortal github page. https://github.com/immortal/immortal/.

[38] immortalctl main page. https://immortal.run/post/immortalctl/.