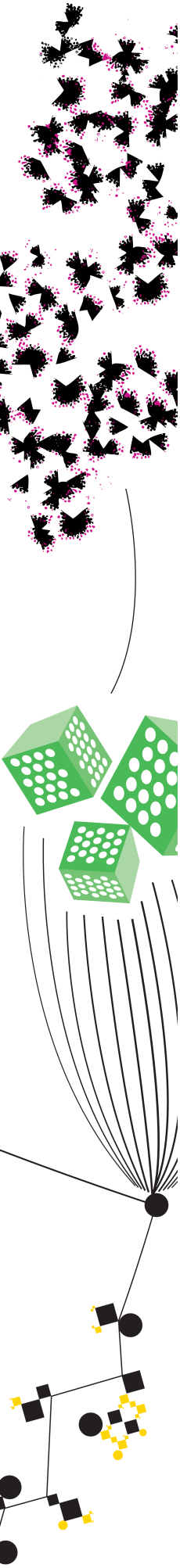

Master Thesis
Characterizing Sender Policy Framework Configurations at Scale

Gabriël Mathay Kahraman

Monday 7th September, 2020

A thesis presented for the degree of
Master of Science Computer Science

UNIVERSITY OF TWENTE.

Design and Analysis of Communication Systems (DACS)

Chair:	prof. dr. ir. Aiko Pras
Supervisor:	dr. ir. Mattijs Jonker
Co-supervisors:	ir. Olivier van der Toorn and dr. Doina Bucur

Abstract

Phishing involves disguising oneself as a trustworthy entity in electronic communication, for example, by pretending to send e-mail on behalf of a company. Phishing e-mails can be prevented if domains implement e-mail security techniques. One of the techniques to improve e-mail security is the Sender Policy Framework (SPF). To enable SPF, the administrator of a domain can specify an SPF policy in the DNS zone of the domain. The SPF policy determines which IP addresses are authorised to send e-mail from the administrator's domain. When an e-mail server receives an e-mail, the e-mail server retrieves the SPF policy of the sender's domain. Next, the IP address of the sender will be queried against the SPF record, and the response of this query determines how to handle the incoming e-mail.

The SPF standard was released over six years ago. Even though six years have passed, the research community does not yet have a thorough understanding of the characteristics of SPF use. What we miss is an understanding of how SPF policies are configured, how SPF policies have changed over time, and what the problematic trends are of SPF use. In this Thesis, we address the missing of a large scale analysis on SPF policies over time.

We investigate SPF use among registered domains under large, legacy generic top-level domains (com, net, and org), as well as by popular domain names from the Alexa top 1 million list. The first point of interest is that the adoption of SPF between the legacy generic top-level domains dataset (168 million domains) differs vastly compared to the *Alexa* dataset (1 million best-visited domains). Whereas the legacy generic top-level domains dataset had an adoption of around 25% on the first of May 2020, the Alexa top 1 million dataset had an adoption of around 74%.

SPF makes uses of so-called mechanisms, an example of which is the `include` mechanism, that allows domain administrators to include the SPF policy of another domain. Typically, the `include` mechanism is used to include SPF information specified by other parties, which enables domain name administrators to account for policies set by, for example, the cloud-based e-mail providers that they use for outgoing e-mail. By examining the `include` mechanism, we found out that the absolute and relative usages of cloud e-mail providers have been increased over time.

We also noticed that domain administrators do not often change their SPF records. An example of this is the altered SPF record of Google's G Suite, which administrators had to include in their SPF record to enable and adequately work G Suite. Ten years after this altered SPF record, some domains are still including the old SPF record.

We also notice that the influence of the government can be a reason to adopt SPF. On the seventeenth of October 2017, the United States Department of Homeland Security (DHS) released the Binding Operational Directive (BOD) 18-01 related to enhancing e-mail and web security. Three and a half months after the release of BOD 18-01, an increase of adoption on SPF among the `.gov` domains of almost 24% points is visible.

While SPF offers an effective way to combat e-mail forgery, mistakes in policy configurations can be made. The mistakes range from simple typos to cycles in `include` mechanisms which cause the DNS lookup limit to be reached. Invalid records can have unintended consequences and even undermine security. In this Thesis, we investigate two ways in which SPF records can be declared invalid. First, there are syntactic errors, which involve SPF records which do not follow the syntactic standards of the RFC on SPF. Second, there are DNS lookup limit errors, which are defined as SPF records which have mechanisms and modifiers that cause more than ten DNS lookups. We analyse both and find that syntactic errors are probably caused due to human error, while DNS lookup limits are often reached due to linkage of SPF records. In some cases, a depth of six using `include` mechanisms was visible. In general, around 10% of the SPF records in the `main` dataset are declared invalid due to syntactic errors or exceeding DNS lookup limits.

Contents

1 Introduction

Nowadays, cybersecurity is a hot topic. In 2019, a total of 124.1 billion US dollars were spent on cybersecurity [23]. That number is still increasing: the worldwide spending on cybersecurity is forecasted to reach 133.7 billion US dollars in 2022 [31]. A significant amount of cybersecurity spending goes to e-mail security, which is because 94% of malware is sent through e-mail [37].

A reason why much malware is sent through e-mail is that there are almost four billion e-mail users [1]. Another reason why much malware is sent through e-mail is that sending an e-mail is rather simple: enter recipient's e-mail address (e.g. `bob@gmail.com`), specify the sender's address (e.g. `alice@somebank.com`) write a subject, write the context, send, and the e-mail will, hopefully successfully, be delivered to the recipient's mailbox. While the simple nature of the Simple Mail Transfer Protocol (SMTP) is a strength, it also brings about a weakness: SMTP has no built-in feature to verify whether the sender of the e-mail is authorized to send that e-mail on behalf of the specified sender address domain (cf. `somebank.com` in the previous example). For example, an adversary could send an e-mail where the sender's e-mail address is set to an e-mail address of the domain of `somebank.com` without having the authorization to send from that domain. This issue is called e-mail sender address forgery.

To address the e-mail sender address forgery problem, multiple defence techniques have been developed. The three most used defence techniques are: Sender Policy Framework (SPF), DomainKeys Identified Mail (DKIM), and Domain-based Message Authentication, Reporting and Conformance (DMARC). The oldest and most used in practice is the Sender Policy Framework (SPF). SPF information, which is published in the DNS, is used to verify if the sender, identified by its IP address, is authorized to send e-mails on behalf of a domain name. The proposed standard of SPF was released in April 2014 [19], which means that SPF is currently more than six years old. While more than six years have passed, an understanding of how SPF is configured at scale, including potential errors in the deployment that can undermine security, is missing.

When a domain has no or incorrectly configured SPF records (and no other defence mechanisms), it might be possible to perform the previously described sender address forgery attacks. Having an incorrectly configured SPF record is possible, since specifying SPF records is typically done manually and human errors are made. When it is possible to forge sender addresses, an adversary could send e-mails containing malicious URLs or attachments which might contain viruses or malware. The receiver of the e-mail might think that the e-mail is sent from a valid domain and therefore may think the e-mail is valid (ham). Thinking the e-mail is valid increases the chance that the receiver opens the malicious URL and thereby increases the chance of being infected by a virus or malware.

Another way to abuse forging sender addresses is to send phishing e-mails to people in which it seems like the e-mail was sent from their bank for example. The phishing e-mail asks the user to log in on a maliciously crafted web page which looks like the original website of the bank. The adversary can then receive the credentials which are entered on this malicious website. With these credentials, the adversary may be able to log in on the real customer portal of the bank and perform payment fraud. Last year, a total of 3.81 million euros of damage was caused by online payment fraud in the Netherlands alone [7], which means that the worldwide damage due to online payment fraud is a lot more.

Another problem that occurs when using SPF records involves `include` mechanisms. The `include` mechanisms are used to include the e-mail security configuration of another domain. Including the e-mail security configuration of another domain generally happens in trust. The problem that could occur is if the included domain includes another domain. In this situation, the SPF configuration specified is included transitively, while no (transitive) trust relationship may exist beyond the first inclusion. In fact, domain name operators may not be aware that this is happening.

The government of the United States of America saw that e-mail sender address forgery is a threat. Therefore, the United States Department of Homeland Security (DHS) released the Binding Operational Directive (BOD) 18-01, related to enhancing e-mail and web security on the seventeenth of October 2017 [9]. This BOD requires government agencies to implement two techniques to improve e-mail security. The first technique to implement is STARTTLS, which is a technique that aims to protect the confidentiality and integrity of e-mail communication between endpoints. The second technique is e-mail authentication. E-mail authentication consists of three

parts: Sender Policy Framework (SPF), DomainKeys Identified Mail (DKIM), and Domain-based Message Authentication, Reporting and Conformance (DMARC).

All government agencies are required to have valid SPF and DMARC records within 90 days after issuance of this directive. The 90 days period, means that the DHS found it very important to enhance the e-mail security of all the agencies in a relatively short period.

Shortly after the US mandate, the Dutch government published a statement regarding sending spoofed e-mails using the e-mail addresses of members of the Dutch parliament [32]. The issue of the Dutch parliament was caused by having no e-mail security techniques implemented on their domains. Having no e-mail security techniques means that malicious people were able to send e-mails using the domains of the Dutch parliament. Journalists of Follow the Money (a Dutch platform for investigative journalism) sent spoofed e-mails which looked like members of the Dutch parliament genuinely sent them to other members of the Dutch parliament [29]. The journalists of Follow the Money were not able to track whether the receiving members of the Dutch parliament replied to the spoofed e-mails since the responses will be sent to the actual e-mail addresses of the members of the Dutch parliament.

To summarize: the problems that arise when not using or using misconfigured SPF records is that forging sender addresses might be possible (when no other defence techniques are used). Not using or using misconfigured SPF records could mean that e-mail fraud performed by an adversary becomes feasible. E-mail fraud could result in a high financial loss for individuals and companies. According to the FBI, e-mail fraud has cost 26 billion US dollars since 2016 [13].

A method to reduce the amount of e-mail fraud and thereby the financial losses is that administrators of domain names should correctly implement SPF to reduce the risk of forging sender addresses by adversaries. In this Thesis, we are going to characterize sender policy framework configurations at scale. More specifically, we will look on a large scale at the deployment of SPF and the mistakes that are made configuring an SPF policy. We defer a more detailed description of the research goal to Section 4. First, we will provide some background information on SPF (Section 2) and discuss related work (Section 3).

2 Background

To implement SPF, the domain administrator needs to add a TXT record into the DNS zone of the domain name. SPF records must always start with the phrase `v=spf1` followed by a combination of eight available mechanisms. Consider the following example of an SPF record: `v=spf1 +ip4:1.2.3.4 redirect=utwente.nl`. In this example, `ip4` is a so-called mechanism; `+` is a qualifier; and `redirect` is a modifier. SPF records can have multiple mechanisms, among others the `ip4` and `a` mechanisms. The combination of mechanisms, qualifiers and so on specifies the policy for a domain name.

In the following sections, we will explain the various mechanisms, qualifiers, and modifiers and give an example of a more complex SPF record. However, we will start by explaining how the Simple Mail Transfer Protocol (SMTP) works and explain how e-mail servers check SPF policies.

SMTP

Simple Mail Transfer Protocol (SMTP) is a protocol used to transmit e-mails. SMTP was defined in 1982 by RFC 821 [28]. SMTP clients are free to enter fields such as the sender address (e.g. `alice@somebank.com`). However, the IP address of the sender by which the SMTP connection was setup with is fixed. Therefore, the IP address of the sender is the basis of SPF validation. All mechanisms, that we explain in more detail later in this section, result eventually to an IP address. The sender's IP address will be matched against all mechanisms, and a result will be determined based on the SPF policy.

2.1 Mechanisms

The Sender Policy Framework (SPF) uses so-called mechanisms to determine if the sending e-mail server, determined by its IP address, is authorized to send e-mail from the sender's specified address domain. The eight mechanisms, that we explain in more detail later in this section, are as follows:

- `a` - Matches address record (A or AAAA)
- `ip4` - Matches in a given IPv4 address range
- `ip6` - Matches in a given IPv6 address range
- `mx` - Matches an MX record
- `ptr` - Matches a PTR record
- `exists` - Matches the given domain name
- `include` - Matches the SPF record of another domain
- `all` - Matches always

Specifications of mechanisms might contain parameters written between square brackets (`[]`). Everything written between square brackets is optional. Some mechanisms cause DNS requests. An SPF record has a DNS lookup limit of ten. However, some mechanisms have their own local DNS lookup limit. More details about DNS lookup limits will be explained in Section 2.5.1 and in the sections of the mechanisms.

A

The `a` mechanism verifies whether the domain name has an `A` or `AAAA` address record that matches the sender's IP address. The format of the `a` mechanism is defined as:
`a[:domain-spec][dual-cidr-length]`. Whereby, `[:domain-spec]` is defined as a macro-string domain-end where the `A` or `AAAA` records will be checked. The `[dual-cidr-length]` sequence is defined as an IPv4 subnet (an integer between 0 and 32) or as an IPv6 subnet (an integer between 0 and 128). When the `a` mechanism is used, a DNS request is invoked to retrieve the `A` and `AAAA` address records.

IP4

The ip4 mechanism verifies whether the sender's IPv4 address is within the defined IP address range. The format of the ip4 mechanism is defined as: ip4:ip4-network[ip4-cidr-length], whereby ip4-network is defined as an IPv4 address and is mandatory. The [ip4-cidr-length] sequence starts with a / and is followed by an integer between 0 and 32 to define the subnet. The default value of the subnet is set to /32.

IP6

The ip6 mechanism verifies the same as the ip4 mechanism, but instead of using an IPv4 address an IPv6 address is used. The ip6 mechanism uses the format: ip6:ip6-network[ip6-cidr-length], whereby ip6-network is an IPv6 address and is mandatory. The [ip6-cidr-length] sequence starts with a / and is followed by an integer between 0 and 128 to define the subnet. The default value of the subnet is set to /128.

MX

The mx mechanism verifies if the sender's IP address is a mail exchanger of the domain in question, which domain name administrators can specify in so-called MX resource records. The format of the mx mechanism is defined as:
mx[:domain-spec][dual-cidr-length] whereby [:domain-spec] and [dual-cidr-length] are respectively defined as a macro-string domain-end and an IPv4 and/or IPv6 subnet. When the mx mechanism is used, a DNS request is invoked. This DNS request counts to the global limit of ten DNS lookups. However, each mx mechanism has an own local limit of ten DNS requests it invokes when querying A or AAAA resource records.

PTR

The ptr mechanism verifies whether a DNS reverse-mapping exists for the sender's IP address and points correctly to the domain name. This mechanism should not be used since it is slow, and it is not as reliable as other mechanisms [20]. The ptr mechanism is still part of the SPF protocol, and thereby it is possible to use but advised not to use it. The ptr mechanism is defined as: ptr[:domain-spec], whereby [:domain-spec] is defined as a macro-string domain-end. When using the ptr mechanism, a DNS request is invoked. This DNS request counts to the global limit of ten DNS lookups. However, each ptr mechanism has an own local limit of ten DNS requests it invokes when querying A or AAAA resource records.

EXISTS

The exists mechanism is used to define complex matches. When using the exists mechanism a DNS A resource record lookup is queried and verifies if any A record matches with the sender's IP address. An example of using the exists mechanism is: exists:%{i}._spf.somedomain.com. The %{i} represents the sender's IP address. As an example let the sender's IP address be 1.2.3.4 and this will result in: exists:1.2.3.4._spf.somedomain.com. Then a DNS A resource record lookup is queried and if it returns true it will pass. The exists mechanism is defined as: exists:domain-spec.

INCLUDE

The include mechanism makes it possible to include SPF records of other domain names. Often, the include mechanism is used by domain name administrators who have multiple domains that need the same SPF records. Using an include mechanism, the domain name administrator only needs to edit the SPF record on the main location instead of multiple locations that use the same SPF records. The include mechanism is defined as: include:domain-spec. When the include mechanism is used, a DNS request is invoked which counts to the global limit of ten DNS lookups limit. However, the mechanisms inside the included SPF record also count to the global limit of ten DNS lookups.

ALL

The `all` mechanism is often used at the end of the SPF record to determine what to do when previous mechanisms are not matched. Often it is used in combination with a FAIL (`-`) qualifier such that when the IP address of the sender has not been matched to the previous mechanisms, the e-mail should be rejected by the receiving e-mail server.

The `all` mechanism matches always. It means that all the mechanisms that are written after the `all` mechanism will not be checked and thereby (unintentionally) establish a security risk.

2.2 Qualifiers

Qualifiers define how the recipient's e-mail server should interpret the result of the SPF record. Qualifiers must be used in combination with a mechanism. Therefore, qualifiers are always placed before and attached to a mechanism. When there is no explicit qualifier, the `PASS` (`+`) qualifier is automatically used. The four available qualifiers, that we explain in more detail later in this section, are as follows:

- `+` PASS result, means the e-mail should be accepted
- `?` NEUTRAL result, means the e-mail should be treated the same as without a policy
- $\sim$ SOFTFAIL result, means the e-mail should be tagged
- `-` FAIL result, means the e-mail should be rejected

PASS

The `PASS` qualifier is defined as the `+` sign. The `PASS` qualifier determines that when a mechanism returns a true result, the e-mail server should accept this e-mail.

NEUTRAL

The `NEUTRAL` qualifier is defined as the `?` sign. The `NEUTRAL` qualifier determines that when a mechanism returns a true result, the mail server should handle this e-mail as any other receiving e-mail which has no SPF policy.

SOFTFAIL

The `SOFTFAIL` qualifier is defined as the $\sim$ sign. The `SOFTFAIL` qualifier determines that when a mechanism returns a true result, the e-mail server should accept the e-mail but also tag the e-mail, since the `SOFTFAIL` is located between the `NEUTRAL` and `FAIL` qualifiers.

FAIL

The `FAIL` qualifier is defined as the `-` sign. The `FAIL` qualifier determines that when a mechanism returns a true result, the e-mail server should reject this e-mail.

2.3 Modifiers

Modifiers are used to extend an SPF record. Therefore, modifiers do not return a true or false result. Instead, they provide additional information. There are two available modifiers that we explain in more detail in the next two subsections.

Redirect

The first modifier to discuss is the `redirect` modifier. The `redirect` modifier is placed at the end of an SPF record, such that when all mechanisms have returned a false result the `redirect` modifier will be matched. The `redirect` modifier is often used in the same way as the `include` mechanism is used: often, it is used when multiple domains need the same SPF records, and it is easier to edit only one SPF record instead of multiple for a domain name administrator. The `redirect` mechanism is defined as: `redirect=domain-spec`.

Explanation

The second modifier to discuss is the explanation modifier. The explanation modifier matches when a mechanism matches which has a `-` qualifier attached to it. When using the explanation modifier, a message will be sent why the query returned a false result. With the explanation modifier, the administrator can debug why a specific e-mail did match one of the mechanisms with a `-` qualifier. The explanation modifier is defined as: `exp=domain-spec`.

2.4 Example SPF Record

All mechanisms and qualifiers are explained at this point. An example of a valid SPF record which uses all the possible mechanisms and qualifiers is:

v=spf1 +a -ip4:130.89.1.1/16 ?ip6:2001:db8:85a3::8a2e:370:7334 mx ~ptr exists:%{i}._spf.somedomain.com include:_spf.google.com -all

The record starts with `v=spf1`, which is the starting sequence of a valid SPF record. Next is `+a`, which means that if an `A` or an `AAAA` resource record matches, the e-mail server should accept the incoming e-mail. Next is `-ip4:130.89.1.1/16`, which means that if the sender's IP addresses matches the `130.89.1.1/16` subnet the e-mail should be rejected. Next is `?ip6:2001:db8:85a3::8a2e:370:7334`, which means that when the `2001:db8:85a3::8a2e:370:7334` IPv6 address matches, the e-mail should be treated like a regular incoming e-mail which has no SPF record. Then there is the `mx` mechanism. Since there is no qualifier in front of this mechanism, it automatically means that when this mechanism matches the e-mail should be accepted. Next is the `~ptr` mechanism. Since there is a ~ qualifier in front of the `ptr` mechanism, it means that when the `ptr` mechanism matches, the e-mail should be accepted but also be tagged. Next is `exists:%{i}._spf.somedomain.com`, which has been explained in Section 2.1. Next is `include:_spf.google.com`, which means that the SPF record of `_spf.google.com` should be included in this SPF record. In the end, there is the `-all` mechanism. When no previous mechanism has matched, the `-all` mechanism will match and return a reject status.

2.5 Errors

Whenever something goes wrong with validating SPF records, there are different types of errors that could be returned: a `permerror` or a `temperror` could be returned. These two errors will be explained in more detail in the next two subsections.

2.5.1 Permerror

A `permerror` indicates that the receiver's e-mail server could not interpret the SPF record. This error can have multiple causes. One way this error could be returned is when an SPF record does not have valid syntax. Another way this error could be returned is when there are more than ten DNS lookups in an SPF record [21]. The `include`, `a`, `mx`, `ptr`, `exists` mechanisms and the `redirect` modifier are invoking (at least) one DNS request. The limit on DNS lookups is established to reduce the load on DNS servers and to prevent Denial-of-Service-Attacks (DDoS). When no DNS lookup limit was established, an attacker could create a cycle which causes the DNS server to be busy with the attacker's request and no other valid request could be processed.

The ten DNS lookups are quickly reached when there is a cycle in the SPF record using the `include` mechanism. Each `include` mechanism is at least invoking one DNS request, and after at most ten cycles the global DNS limit is reached and a `permerror` must be returned. We empirically ran various domain names with known cycles through two online SPF analysers (i.e. `dmarcanalyzer.com` [11] and `mxtoolbox.com` [26]). Both analyzers invalidate the SPF records due to the cycle. Mxtoolbox, for example, states that the *recursive loop* will result in a permanent error.

The cases when a `permerror` shows up, the receiver's e-mail server has several possibilities to handle the incoming e-mail. The first way to handle the e-mail is to reject the e-mail since it can not be checked if the sender has the authorization to send from that domain. The second

option is to deliver the e-mail to the recipient and warn the recipient that the e-mail server could not check the validity of the sender.

2.5.2 Temperror

A `temperror` indicates that there has been an error probably due to a transient condition. A transient condition could be due to a DNS error while performing the check of the SPF record. Another way that a `temperror` could occur is due to a time out when performing the check of the SPF record.

The cases when a `temperror` occurs, the recipient's e-mail server has the same possibilities as previously described with the `permerror`. Ways to handle this error is that the e-mail server could reject the e-mail, or the e-mail server could accept the e-mail but also tag it.

2.6 Dynamic Sender Policy Framework

There is a variant of SPF which does not require administrators to add a TXT record into the DNS zone of the domain name. This variant is called Dynamic Sender Policy Framework (DSPF) [3]. Dynamic Sender Policy Framework uses a database that contains IP addresses of servers which send e-mails in order to estimate the reliability of an e-mail address. The database automatically updates when an IP address changes and therefore it is called *Dynamic* SPF. DSPF is a framework that requires no intervention of the administrators of domains since IP addresses are automatically updated. However, DSPF does not have a wide-spread adoption. Therefore, this Thesis will only focus on SPF.

3 Related work

In literature, most of the research that relates to SPF focuses on how SPF records can be used to detect if a domain sends spam e-mails. Van der Toorn et al. (2018) [33] detected snowshoe spam domains 100 days earlier than when existing spam blacklists detected the snowshoe spam domains. Van der Toorn et al. detected the snowshoe spam domains using active DNS measurement and examining among others SPF records.

SPF can also be used to detect spam e-mails. Sipahi et al. (2015) [30] showed how to detect spam e-mails using SPF records. One of their conclusions is that many spam e-mails are sent while the SPF records contain the tilde (SOFTFAIL) qualifier. Given the potential issues surrounding the use of the SOFTFAIL qualifier, an understanding of how frequently it is used and in which context is interesting to know. We perceive a gap here and will address this in this Thesis.

SPF can also be used without administrators having to configure a TXT record into their DNS zone. SPF without a TXT record is called Dynamic Sender Policy Framework (DSPF). Anh et al. [3] describe DSPF in their paper. This work describes how to implement Dynamic Sender Policy Framework (DSPF). Originally SPF is designed to verify if the sender of the e-mail is authorized to send from the specified sender address domain name and SPF is not designed to detect spam e-mail. However, with the Dynamic Sender Policy Framework, it is possible to detect spam e-mails without having administrators to set up a TXT record into their DNS zone. This advantage could be amplified if, during this Thesis, many invalid SPF records will be found. Another advantage of DSPF is that spam e-mail can be detected without knowing the content of the e-mail and thereby preserving privacy. However, a disadvantage of DSPF is that DSPF does not have a wide-spread adoption.

SPF was originally designed to be used as an anti-phishing mechanism by verifying if the sender of the e-mail is authorized to send e-mails from the specified sender address domain name. Stefan Görling wrote an overview of SPF as an anti-phishing mechanism in 2007 [16]. This paper gives an overview of existing anti-spam and anti-phishing techniques and describes what SPF is, the adoption rate, and why SPF should be promoted to achieve a higher adoption rate. The section that relates the most to this Thesis is the section of the current adoption of the Sender Policy Framework. In this section, the author of the paper explains that he has calculated the adoption rate of SPF on all Swedish domains (domains ending in .se), which were active on the third of February 2006. This dataset consists of about 385 thousand domains. His analysis showed that the adoption rate was low. Of the about 385 thousand domains, only a bit more than six thousand domains (1.63%) had a published SPF policy. The work by Görling et al. considers only about 385 thousand domain names. Moreover, the study dates back thirteen years, to a time where SPF was new and little opportunity for wide-spread adoption had existed. In this Thesis, we expand on this work, by considering significantly more domain names, over an extended period.

E-mail sender authentication is not only used validly but is also misused. Mori et al. showed this use and misuse of e-mail sender authentication in their paper [24]. One of the analyses they have done is calculating the adoption rate of SPF. The total number of domains which they have checked are almost 120 thousand domains. The paper concludes that 50% of the legitimate domains adopted SPF, while 20+% of the spamming domains adopted SPF in 2011. Another key finding of this paper is that spammers publish SPF with three different tactics: using the entire IP address space as spam sources, using dedicated spamming servers, and by including legitimate sender IP addresses. The work by Mori et al. considers only 120 thousand domains (which is even less than the work by Görling et al.) and is done almost nine years ago, to a time where the proposed standard of SPF was not yet released. In this Thesis, we expand on this work, by considering significantly more domain names, over an extended period. Another point how we expand this work in this Thesis is to determine the linkage between domains using the `include` mechanism, which may cause domains to be part of a spamming group.

To authorize a cloud-based e-mail provider to be able to send e-mails from a domain, the administrator of a domain needs to include the corresponding SPF record of the cloud-based e-mail provider. Doing this requires the use of an `include` mechanism. Van Rijswijk-Deij et al. (2015) [35] showed that the use of cloud e-mail providers had vastly increased over time. Another work of van Rijwsijk-Deij et al. (2016) [36] performed a limited investigation on SPF records,

merely as a case study and not focus on their paper. In this work, around 34.4% of domains that use Google publish SPF information, while around 92.4% domains that use Microsoft publish SPF information.

Moura et al. (2017) [25] discuss the business model behind domain name abuse. One could, for example, abuse domain names to perform phishing attacks by sending phishing e-mails. One way to increase the worth of the domain names is to perform Blackhat Search Engine Optimization, which tries to artificially improve the visibility in search engines such that the domain name receives more visitors and thereby increasing the worth of the domain name. When other domains have included a domain, the worth of the domain might be increased since it might be marked as a trusted domain. In this Thesis, we expand on this work by considering how domains are linked to each other using `include` mechanisms.

To the best of our knowledge, a large-scale quantification on how SPF is used in practice is missing. The proposed standard of SPF was published in April 2014 [19], and this means that domain name administrators have had six years to implement SPF. We propose to fill this gap by performing a large scale analysis on the configurations of SPF records and issues that arise when using SPF.

4 Research Questions

Section 3 described previous works related to SPF. Most of the related works sparked our interest. However, the previous works miss an understanding of how SPF is deployed on the Internet. What we miss is an understanding of how SPF policies are configured, how SPF policies have changed over time, and what the problematic trends are of SPF use. To address the missing of a large scale analysis on SPF policies over time, we propose the following main research question:

How are SPF records configured in practice?

The main research question is split into three smaller research questions:

- How do administrators typically configure SPF? **(RQ1)**
- Can we identify a change in the use of SPF over time? If yes: How do SPF records change over time? **(RQ2)**
- Can we recognise problematic trends in SPF use? If yes: What are the characteristics of these problematic trends? **(RQ3)**

The first RFC on SPF was published in April 2006 [38] and the proposed standard was published in April 2014 [19]. The release in April 2014, means that the proposed standard was released over six years ago. Therefore, we believe that SPF is no new technique anymore and that six years of usage is long enough to see trends and changes in SPF implementations. With this Thesis, we are going to find out how SPF is currently enrolled; we will take a look at how the use of SPF has evolved over the years; and what kind of errors occur when specifying SPF.

The next sections discuss each research question and a hypothesis will be given for each research question.

4.1 How do administrators typically configure SPF? (RQ1)

Constructing SPF records is typically done manually by the administrator of the domain name. The administrator determines the mechanisms and qualifiers used in a record. Since the administrators determine how to configure their SPF records, many variations on how to configure an SPF record is possible. We want to find out whether there is a typical pattern of SPF use. The pattern we think that exists is that SPF policies are set up to allow certain IP addresses and drop all other IP addresses. An example of this pattern is: `v=spf1 +ip4 -all`.

Given that related work (see Section 3) has shown that the use of cloud-based e-mail is widespread, we expect that many domains use an `include` mechanism, including one of the well-known cloud-based e-mail providers.

Combinations of qualifiers and mechanisms are challenging to predict since there is no clear relation between them except between the `-` (reject) qualifier and the `all` mechanism. We believe that most of the other mechanisms are matched to a `+` (accept) qualifier since we believe that an administrator typically would configure an SPF record as what is allowed and drop e-mails which did not match any of the previous mechanisms.

Overall, we try to identify the most used mechanisms, qualifiers, modifiers and the most used combinations of qualifiers and mechanisms. With this quantification, we will be able to determine the general pattern a configuration of an SPF policy follows.

4.2 Can we identify a change in the use of SPF over time? (RQ2)

The proposed standard of SPF was published in April 2014 [19]. The proposed standard has been released for more than six years. In these six years, administrators of domain names have had the opportunity to add an SPF policy to their DNS records. We would like to see how this implementation took place and how the SPF records have been changed over time. The latter will only be investigated when we can identify a change in the use of SPF records over time.

We believe that SPF records do not often change over time since we think that domain administrators configure SPF records once and leave them as they are unless a change is required. One change we might think is true is the increase in SPF's adoption rate since the first RFC

of SPF in 2006 [38]. The increase of SPF's adoption rate would be due to that we believe that e-mail sender address forgery is more well known under domain administrators in the present than in 2006. The second reason why we think that the adoption rate of SPF would be increased is that phishing is more prominent and lucrative for criminals at present than back in 2006 [17].

4.3 Can we recognise problematic trends in SPF use? (RQ3)

Configuring SPF is possible in various ways. One of the ways to configure an SPF record is to use an `include` mechanism. When a domain (first) includes another domain (second) and the second domain also has an `include` mechanism to include a domain (third). The IP addresses that match the latter included domain's SPF policy are then also authorised to send e-mail from the original domain (first). Most of the time, this is not the intention of the administrator of the first domain, which included the second domain. In this Thesis, we will try to find out whether this scenario happens often and also try to find out other drawbacks of using SPF in specific ways.

As previously explained, we think that many domains include another domain without knowing that the second domain also has an `include` mechanism to another domain, and thereby the last domain is also authorised to send e-mails from the first domain. This is an include chain of depth two. However, we are also interested in chains that contain a depth of three or more.

DNS lookup limits

Another potential drawback of SPF is that an SPF record can perform a maximum of ten DNS lookups (see Section 2.5.1) [21]. The ten DNS lookups limit mean that when an SPF record contains a lot of mechanisms, which cause more than ten DNS lookups, the SPF record is invalid and a `permerror` will be returned. We expect that when this limit is reached and a `permerror` is returned, the domain administrator might be unaware of the fact that the SPF record is invalid.

From our perspective, we think that a lot of SPF records are set up manually by domain name administrators. Since it is manual labour, errors exist and therefore, causing invalid SPF records. We think that there are many invalid SPF records for which the domain administrator is unaware that the SPF record is invalid.

5 Methodology

To advance our understanding of how SPF is configured in practice, we want to investigate the SPF policies of a large number of domain names. To investigate trends and dynamicity, we would like to be able to do this over time, over an extended period. For this reason, we work with a large dataset from the OpenINTEL project in this Thesis. In the next section, we will explain what this data provides us with. In the sections after, we will outline our approach to answering the research questions, using these data.

5.1 Datasets

To investigate SPF records, a large dataset is needed which preferably captures not only SPF records at a single point in time, but also allows us to track it over a longer period. The University of Twente has a joint project (OpenINTEL) with SURFnet, SIDN Labs, and NLnet Labs which performs a DNS scan on more than 60% of the domains of the internet [34]. This dataset consists of 235 million domains which are scanned daily. The daily scans result in 3.7 billion collected data points each day [27]. The data gathered by the OpenINTEL project will be used to perform the research. The collected data points of each daily scan are stored on an Apache Hadoop cluster located at the University of Twente. For each domain, it sends once every 24 hours a fixed set of DNS queries. The most notable DNS queries are: `NS`, `A`, `AAAA`, `MX` & `TXT`.

As SPF configurations are signalled using `TXT` records, our analyses are primarily done based on `TXT` measurement data. We will also look at shared DNS infrastructure (i.e. nameservers) of domain names that have particular SPF configurations in common. To this end, we will also consider `NS` measurement data, which will be further explained in Section 5.2.

The collected data points of each daily scan are not stored into one large dataset but stored in smaller datasets. The datasets which will be used in this Thesis are:

- com - Which contains all registered .com TLD domains
- org - Which contains all registered .org TLD domains
- net - Which contains all registered .net TLD domains
- Alexa - Alexa top 1M dataset, which contains the top one million most-visited domains
- OpenCC - Which contains domains with a TLD that are publicly available

Our analysis encompasses the three legacy generic Top-Level Domains (gTLDs) com, net, and org. The three legacy gTLDs together account for about half of all the (global) namespace. Moreover, to complement our view, we also consider popular domain names by looking at the Alexa top one million dataset. Throughout this Thesis, we refer to the dataset of combined legacy gTLDs as *main*, and the Alexa top one million dataset as *Alexa*.

The OpenCC dataset consists of multiple publicly available country-code TLDs. In some occasions, we also made use of the OpenCC dataset whereby we established some smaller datasets out of by filtering on specific TLDs. These smaller datasets consist of the Swedish (.se), the Estonian (.ee), and the American government (.gov) domains.

Table 1 displays the differences between the three datasets. As expected, the *main* dataset consists of the most domains, followed by the *OpenCC* dataset, and the dataset with the least domains is the *Alexa* dataset. The *main* dataset accounts for a 1994-day period, starting at the twenty-eighth of February 2015. The *Alexa* dataset starts at the twenty-second of January 2016 and accounts for 1635 days. The *OpenCC* dataset accounts for a 1382-day period, starting at the first of October 2016. The difference in starting dates is because the OpenINTEL project did not start gathering data for all the different TLDs at the same time.

5.2 Adoption Rate

Up to now, we did not explain how we will analyse the vast amount of data, also called *big data*. To handle the *big data*, we use Spark [4]. The advantage of using Spark is that it is optimised to

Date	Dataset	Start Date	Unique SPF Records	Domains With SPF	Total Domains
01-05-2020	*main*	28-02-2015	3,276,268	42,144,226	165,474,956
01-05-2020	*Alexa*	22-01-2016	316,922	750,469	1,019,464
01-05-2020	*OpenCC*	01-10-2016	63,750	533,202	1,844,432

Table 1: Statistics about the three datasets

handle *big data* very quickly. Another advantage of Spark is that it contains predefined functions. An example of a predefined function which is often used is the `distinct` function. The `distinct` function counts the number of unique rows in a dataset.

To be able to calculate the adoption rate of SPF among the domains in the dataset, we first have to calculate the number of unique domains in the respective dataset. Unique domains are defined as a domain name with a TLD and no subdomains. The datasets have two columns: one column with the domain names (`query_name`) and a column with `TXT` records (`txt_text`). Table 2 shows an example of TXT measurement data without filtering on the SPF starting phrase. Specifically, for the domain name `google.com` on the first of April 2015. Table 2 displays that the TXT records in the DNS zone of a domain do not only contain SPF policies. In the case of `google.com`, it uses the TXT records to offer electronic signatures (DocuSign) [12], to verify their domain with Facebook, and to encrypt their e-mails using Secure/Multipurpose Internet Mail Extensions (S/MIME) [14].

We want to know the number of unique domains in a dataset to calculate the adoption of SPF. As can be seen from Table 2, counting the number of rows will not result in the correct number (i.e. one domain instead of five) of unique domains in the dataset. One might think that using the `distinct` function from Spark will remove all duplicate rows based on the `query_name` column and next count the remaining number of rows in the dataset. Using the distinct function partially fixes the problem, since the dataset also consists of some subdomains (e.g. `www.google.com`). Therefore, we need to filter out all subdomains to count to the correct number of unique domains in the dataset.

The solution to this problem is to use a regular expression that extracts the original domain out of the `query_name` columns. After that, we can use the `distinct` function from Spark on the `query_name` column and retrieve the correct number of unique domains in the dataset.

There is one catch, and that is that the Alexa top 1M dataset does not contain precisely one million domains as suggested by the name. The Alexa top 1M list frequently changes throughout the day, which is why the measurement captures more than one million domain names for each day.

date	query_name	txt_text
01-04-2015	google.com	v=spf1 include:_spf.google.com ~all
01-04-2015	google.com	docusign=05958488-4752-4ef2-95eb-aa7ba8a3bd0e
01-04-2015	google.com	docusign=1b0a6754-49b1-4db5-8540-d2c12664b289
01-04-2015	google.com	facebook-domain-verification=22rm551cu4k0ab0bxsw536tlds4h95
01-04-2015	google.com	globalsign-smime-dv=CDYX+XFHUw2wml6/Gb8+59BsH31KzUr6c1l2BPvqKX8=

Table 2: Example of TXT records of google.com on our original dataset on the first of April 2020

It is now clear how many unique domains there are in the dataset, as shown by the last column of Table 1. The second step is to look at how many of those domains have an SPF record. Not all registered domain names have a `TXT` record in their DNS configuration. Those that do have a `TXT` record, not all have an SPF configuration. To identify SPF use, we filter `TXT` records accordingly (see Section 2). As a result of this, we can count the number of SPF records in a dataset, as displayed by the fifth column in Table 1. Knowing the number of domains with an SPF record and the total number of domains in the dataset, we can calculate the adoption rate of SPF.

5.3 Mechanisms and Qualifiers

As SPF records can be comprised of multiple mechanisms, we start by tokenising the SPF strings. Tokenising splits up the SPF records at each space into a list of tokens (words). Later, we separate the qualifier (see Section 2.2) from the mechanism, by verifying if the first character of the token is a qualifier. To show how this process works, consider the SPF record: `v=spf1 ip4:1.2.3.4 include:google.com`. A Tokeniser will split this SPF record in three tokens: `v=spf1`, `ip4:1.2.3.4`, and `include:google.com`. Each token is a word and will be matched against all possible combinations of qualifiers and mechanisms. The result will be a list that consists of all mechanisms, all qualifiers, all combinations of qualifiers and mechanism and their respective occurrences. In our example the result would be: `include`: 1 and `ip4`: 1. With this approach, we are able to count the number of occurrences of each mechanism, qualifier, and a combination of those two.

The next step is to count the number of domain names with an SPF record. We will calculate this in the same way as previously described in Section 5.2. The last and final step is to divide the occurrences of each qualifier and mechanism by the number of domains with an SPF record to retrieve the relative usages of the respective qualifier, mechanism or combination of qualifier and mechanism.

The preceding steps display that we will count the number of qualifiers irregardless of the mechanisms with which they are combined (e.g. `+all` and `+ip` both increase the `+` qualifier count). However, we think that most qualifiers are used in combination with the `+` qualifier, the exception will probably be the `all` mechanism. We think that the `all` mechanism will be most often used in combination with the `-`, ∼, and `?` qualifiers. Therefore, we will also perform an analysis of the qualifiers without the occurrences of the qualifiers which are attached to the `all` mechanism.

5.4 Most Included Domains

With the `include` mechanism it is possible to include an SPF record of another domain. To calculate the top ten of most included domains, we will use the same approach as described in Section 5.3, so we will not go into detail in the approach. The approach we will follow is:

- Regular expression tokeniser with regular expression: `include:(\S+)`
- Count occurrences of included domains
- Divide the number of occurrences with the number of domains with SPF records

With this approach, we will obtain the ten most included domains of the respective dataset.

To identify which type of companies are behind the top ten of most included domains, we will group the companies in different types of groups. Groups might consist of hosting providers, SaaS (Software as a Service) providers, social media, blogs, and many more categories are possible.

5.5 Valid and Invalid SPF Records

There are multiple ways of how an SPF record could return a `permerror`. A `permerror` could be returned when there are multiple SPF records of one domain, syntactical errors, DNS lookup limit exceeded, labels longer than 63 characters, names with empty labels, MX lookup limit exceeded, multiple of the same modifier, domain not found in a redirect modifier or an incorrect macro.

In section 3, we mentioned the research from van Rijswijk-Deij et al. regarding the increase of popularity of cloud-based e-mail providers. When a domain uses a cloud-based e-mail provider, the domain administrator typically has to include the e-mail provider's SPF record using an `include` mechanism. Including an SPF record, might cause for more DNS lookups and therefore the SPF record might exceed the DNS lookup limit. Therefore, we decided to analyse how often the DNS lookup limit is exceeded.

In section 4.3, we mentioned that configuring SPF records is typically done manually by the domain administrator. Therefore, we think that SPF records might contain syntactical errors, and therefore we decided to include the syntactic errors in our analysis. An example of a

syntactic error is when a qualifier is placed in an SPF record standalone (that is, not attached to a mechanism).

To verify if an SPF record is valid or not, we will use a normal Tokeniser which splits the SPF records at every space, as explained in Section 5.3. The result is a list of words, which in case of valid SPF records will start with `v=spf1` and the remaining words will be written accordingly to the RFC of SPF [19]. The remaining words will be checked whether they start with a combination of a qualifier and a mechanism or only a mechanism. If this is the case, then that word will be declared valid. If all the words are declared valid, the SPF record will be declared valid.

Since we only check if the start of a word is valid and not the remainder, it could be that we declare records as valid while they are invalid. This approach was chosen because verifying the part after the mechanism is difficult to verify since there are many possibilities. For example, when verifying if `ip4:127.0.0.1/2` is valid, we first need to verify if `127.0.0.1` is a valid IPv4 address. Next we need to check if there is a `/` character after `127.0.0.1`. If this is the case, then we need to check if the subnet value (2) is between 0 and 32. This is only one example, many more variations are possible, and all need to be checked. For simplicity, we will only verify the first part of a token.

An example of a record that should be declared invalid, while in our case it will be declared as valid is: `"v=spf1 include:  -all`. The invalid part of this SPF record is: `include:  -all`. This SPF record should be declared invalid since `include:` has no IP or domain address after the `:` sign and this is required. Using our technique will classify this SPF record as valid since the first token/word starts with `v=spf1`, the second token/word starts with `include` and the last token/word starts with `-all`. To determine how well our limited analyser works, we manually analysed fifty domains which the analyser declared as valid. Zero out of these fifty domains were wrongly classified as valid. Since zero out of fifty domains were wrongly classified, we estimate that at most 2% of the domains which our analyser validates will be wrongly classified.

Another way how an SPF record is declared invalid and therefore the recipient's e-mail server returns a `permerror` is when there are more than ten mechanisms that cause a DNS request in one record. The mechanisms that result in a DNS query are:

- `include`
- `a`
- `mx`
- `ptr`
- `exists`
- `redirect` (modifier)

To check if an SPF record has too many mechanisms which cause a DNS query, we split the SPF records into words using the same normal Tokeniser as described with the syntactical errors. Next, we iterate through the words, and whenever a word starts with one of the five mechanisms or the `redirect` modifier, we increment a counter. When all the words of an SPF record are processed, we look at the counter: if the counter is more than ten, the SPF record will be declared invalid, otherwise the SPF record is declared as valid.

However, counting the number of mechanisms/modifiers that cause a DNS request is not enough to determine the number of total DNS requests. The `mx` mechanism does not always provoke only one DNS requests. The number of `MX` resource records the `mx` mechanism queries is also included in the overall limit of ten DNS lookups. The other valuable property of the `mx` mechanism is that each `MX` record must not result in querying more than ten address records, whereby the address records can be `A` or `AAAA` resource records.

The same is true for the `ptr` mechanism: each `ptr` mechanism counts to the overall limit of ten DNS lookups, in which each `ptr` mechanism is limited to query ten address records which can be `A` or `AAAA` resource records. Both the individual DNS requests of the `mx` and `ptr` mechanisms will not be taken into consideration in our analysis, due to simplicity reasons.

The `include` mechanism does not have such kind of *local* limit whereby each `include` mechanism is allowed to have ten DNS lookups. Instead, the mechanisms which are in the included

SPF record count towards the overall limit. As such, the `include` mechanism might be causing many invalid SPF records without the awareness of the administrator of the domain.

To count the number of invalid records due to the linkage of `include` mechanisms, we use the dataset created as explained in Section 5.6. For every domain, we look at the domains it includes. For every included domain we count the number of DNS requests and add those to the global DNS requests of the original SPF record. If the SPF has syntactic errors or exceeds the DNS lookups limit, we consider the SPF record as invalid in our analysis.

5.5.1 Clustering

To investigate patterns in syntactic errors in SPF configurations, we will apply clustering. Clustering enables us to group SPF records with the same characteristics into a cluster. When the clusters are established, we only need to determine the characteristics of the cluster. If we do not use clustering, we need to manually determine the characteristic of all the invalid SPF records in the dataset, which is almost impossible if there are many invalid SPF records. We will apply clustering using the tokenised SPF record data (see Section 5.3). We will exercise the clustering analysis using the *k-means* clustering technique [5]. K-means clustering is a technique that calculates a mean for each SPF record and places the SPF record in the cluster with the closest mean.

5.6 Include Linkage

Our largest dataset (see Table 1) contains 165,474,956 registered domain names, 42,144,226 of which have SPF records. These SPF record can reference each other using `include` mechanisms, and technically nothing prevents cyclic graphs. To study which domains are linked to each other and at what depth, we iteratively determined the included domains of a domain for each depth. For each depth, we store a sub dataset that tells us which domains a domain includes at a specific depth. At each depth, we store the domain names that were included up to the specified depth. A boolean value determines if new domains have been added at this specific depth.

5.6.1 Limitations

The OpenINTEL project measures Second-Level Domains (SLDs), data for subdomain labels are unavailable in our dataset. This means that while we have `TXT` data for, e.g. `google.com`, we do not have the SPF records of `_spf.google.com`. As such, domain names that use the `include` mechanism and point to `_spf.google.com` may include, through `_spf.google.com`, mechanisms that our analysis has a blindspot to. In fact, a 'chain' of SPF records could fall outside our view. Our analysis, therefore, provides a lower bound in terms of, e.g. inclusion depth.

6 Results

To be able to answer the research questions, we analysed SPF data and obtained results. In the next sections, we look at the adoption (rate) of SPF, quantification of mechanisms, qualifiers, and modifiers, the most included domains, validation of SPF records, and include linkage. Each research question will have its own section, in which we explain the results related to the particular research question.

6.1 Results Research Question 1

We defined the first research questions as: *How do administrators typically configure SPF?* To determine how administrators typically configure SPF we will take a look at the adoption of SPF among multiple datasets, and determine the characteristics of an SPF record by examining the usages of mechanisms, qualifiers, and modifiers.

6.1.1 Adoption Rate

We start by investigating the percentage of domain names in the `.se` dataset which have adopted SPF. The `.se` dataset consists of all domains with the Swedish TLD `.se`. The reason why we focus on the `.se` dataset first is because related work by Görling et al. investigated the adoption of SPF in the `.se` dataset back in 2006. Görling et al. calculated an adoption of 1.63% among the domains in the `.se` dataset. The result of Görling et al. gives us something to compare to. We count the total number of registered domains, as outlined in Section 5.2. We also identify domain names with an SPF record (see Section 2) and determine the fraction. For this analysis, we look at the data retrieved on the first of May 2020. The dataset used by Görling et al. consisted of all the registered `.se` domain names on the third of February 2006. The dataset used by us also consists of all the registered `.se` domain names, but now on the first of May 2020. We calculated the adoption of SPF among the domains in the `.se` dataset, and Table 3 shows that the adoption of SPF among the .se domains was 26.39% on the first of May 2020. The adoption of 26.39% in May 2020 means an increment of the adoption of around 25% points compared to February 2006. Our results thus suggest that, compared to almost fifteen years ago, the situation has improved.

Another research discussed in the related work section (Section 3) is the paper of Mori et al. [24]. Mori et al. showed that around 50% of the legitimate domains adopted SPF, while 20+% of the spamming domains adopted SPF in July 2011. Table 3 shows that the percentage of domains among the `.com` dataset had an adoption of 25.16%, and the domains among the Alexa top 1 million dataset had an adoption of 73.61% on the first of May 2020. The results of the adoption mean that only the Alexa top 1 million domain names have a higher adoption on the first of May 2020 compared to the results of Mori et al. in July 2011. This makes sense since the dataset used by Mori et al. is a combination of two datasets: Alexa top 500 and a commercial domain list used for popular free e-mail service providers around the world (e.g. `gmail.com` and `hotmail.co.uk`). Both datasets contain domains which are popular and have high visiting numbers in July 2011. This means that they are comparable to the Alexa top 1M dataset which we have used. The only exception is that the Alexa top 1M dataset contains many more domains than the datasets used by Mori et al. Therefore, the adoption calculated by Mori et al. in July 2011, was so high compared to the adoption of our `main` dataset in May 2020 since it contained only very popular domains.

6.1.2 Similarities between the datasets

To study per TLD the adoption of SPF, we individually look at the three gTLDs that comprise our *main* dataset (i.e. .com, .org, and .net), the two ccTLDs (i.e. .se and .ee) and the Alexa top 1M domains. One of the similarities we found between the .com, .org, and .net datasets is that they all have an adoption of around 25% on the first of May 2020. The histories and trends of the adoption rates are displayed in Figures 1 and 2 and will be explained in more detail in Section 6.5.1.

Dataset	Date	Total Domains	Domains with SPF	Adoption Rate
.se	03-02-2006	385,862	6,282	1.63%
.se	01-05-2020	1,459,990	385,295	26.39%
.com	01-05-2020	145,418,816	36,591,846	25.16%
.org	01-05-2020	10,028,070	2,571,803	25.64%
.net	01-05-2020	13,145,894	2,980,577	22.67%
Alexa top 1M	01-05-2020	1,019,464	750,469	73.61%
.ee	01-05-2020	126,591	72,986	57.65%

Table 3: Adoption of SPF among the domains in the *main*, *Alexa*, and *open ccTLD* datasets

6.1.3 Differences between datasets

On the first of May 2020, there is a clear difference between the adoption of SPF among the domains of the *main* and the *Alexa* dataset. The *main* dataset had an adoption of SPF of around 25%, while the *Alexa* dataset had an adoption of around 75%, which is a difference of around 50%.

A possible explanation for this vast difference is that domains providing services to large user bases are more likely to be operated by entities that make security considerations than the majority of domain names. This reasoning could explain the difference between the adoption of SPF among the domains in the *main* dataset and the *Alexa* dataset.

6.1.4 Estonia

Comparing the country-code top-level domains (of zones) of Sweden (`.se`) and the domains of Estonia (`.ee`), a vast difference is visible between the adoption of SPF among those two: 26.39% versus 57.65%, respectively. The reason why Estonian websites have a relatively high percentage of adoption is that Estonia wants to be an international cybersecurity leader [15]. To become an international cybersecurity leader, one of the aspects to focus on is e-mail security. Since SPF is part of e-mail security, Estonia used resources to increase among others the adoption of SPF, and this has resulted in a much higher adoption of SPF compared to Swedish domains, for example.

Comparing the adoption of SPF between the *Alexa* dataset (73.61%) and the `.ee` dataset (57.65%), a difference is visible of around 16 percentage points. The high adoption of SPF also displays the amount of effort Estonia performs to become an international cybersecurity leader. Especially when taking into account that the `.ee` dataset consists of all the Estonian domains (and not only the top best-visited domains) compared to the *Alexa* dataset which contains the 1 million most visited domains.

6.2 Quantification of Mechanisms, Qualifiers, and Modifiers

In this section, we will focus on a few select moments in time, to investigate how administrators typically configure SPF records. We will investigate the quantification of mechanisms, qualifiers, and modifiers. Later on, in Section 6.6, we will focus on the temporal aspects of SPF use.

To determine how administrators typically configure SPF records, we have taken a look at how often each of the mechanisms is used (over time). For each of the mechanisms, qualifiers, and modifiers, we counted the number of times they occur in all of the SPF records. Then we divided the number of occurrences of the respective mechanism, qualifier, or modifier by the number of domains with an SPF record to retrieve the average number of that type of mechanism, qualifier, or modifier per SPF record. The reason for the division of the occurrences of the respective mechanism, qualifier, or modifier by the number of SPF records is that the number of SPF records is changing over time. By dividing the occurrences of the respective mechanism or qualifier by the number of SPF records, we normalized the results since the number of domains in a dataset might fluctuate over time.

To start this section, we counted the total occurrences of all individual mechanisms, modifiers, and qualifiers. This quantification is visible in Table 4. A noticeable mechanism is the `a` mechanisms: on average the `a` mechanism is used 1.44 times in an SPF record based on the SPF records in the *main* dataset on the first of May 2020. A value in the table displays the average number of occurrences of that specific mechanism in an SPF record. The `a` mechanism is the only mechanism which has a value greater than one. All other mechanisms are, on average, used less than one time per SPF record.

The `ip4` mechanism (0.48 usages on average) is used more than ten times compared to the `ip6` mechanism (0.046). From this vast difference, we can infer that domains mostly are still using IPv4 addresses, which are exhausted [8], instead of the easily available IPv6 addresses.

Table 4 displays that there are five major mechanisms used in general: the `a`, `all`, `include`, `ip4`, and `mx` mechanisms. The `mx` mechanism is used on average 0.42 times per SPF record. All other mechanisms and modifiers are used on average less than 0.10 times per SPF record. From these numbers, we can infer that most SPF records use a combination of these five mechanisms and in the minority of the SPF records also (one of) the other mechanisms.

The `redirect` and `exp` modifier are rarely used. The `redirect` modifier is used on average 0.0162 times per SPF record and the `exp` modifier is used on average 0.0000503 times per SPF record. The very low usage of the `exp` modifier can be related to the fact that it is used to debug why an SPF record returns a `FAIL` result. We believe that administrators use the `exp` modifier to debug why their SPF record is returning a `FAIL` result and when the SPF record has been changed accordingly, the `exp` modifier will be removed.

The two most used qualifiers are the `+` (0.60) and $\sim$ (0.55) qualifiers. The difference between the two qualifiers is five percentage points. However, the `PASS` result of the `+` qualifier can also be achieved when using a mechanism without a qualifier. Therefore, the actual number of `PASS` results will be more than the `+` qualifier displays in Table 4 and the difference in usages between the `+` and $\sim$ qualifier will be larger. We also analysed the total (explicit and implicit) usages of the `+` qualifier, and we came to a value of on average 2.16 usages per SPF record. The value of 2.16 usages is a lot more when only counting the explicit usages of the `+` qualifier. From this vast difference, we can infer that the `+` qualifier is most often used in an implicit way (i.e. using a mechanism without a qualifier). The third most qualifier is the `-` qualifier, and the least used qualifier is the `?` qualifier.

	01-05-2015	01-05-2016	01-05-2017	01-05-2018	01-05-2019	01-05-2020
a	1.53	1.49	1.50	1.47	1.46	1.44
ip4	0.62	0.58	0.62	0.51	0.50	0.48
ip6	0.058	0.102	0.087	0.063	0.025	0.046
mx	0.51	0.49	0.48	0.45	0.44	0.42
ptr	0.112	0.119	0.076	0.093	0.068	0.046
exists	0.000198	0.000205	0.000464	0.000279	0.000324	0.000454
include	0.48	0.55	0.53	0.65	0.71	0.73
all	0.96	0.97	0.96	0.97	0.97	0.97
redirect	0.0179	0.0194	0.0222	0.0168	0.0163	0.0162
exp	0.0000851	0.0000830	0.0000729	0.0000601	0.0000596	0.0000503
+	0.43	0.42	0.57	0.53	0.57	0.60
-	0.26	0.28	0.27	0.29	0.29	0.30
?	0.27	0.23	0.22	0.17	0.15	0.12
~	0.43	0.46	0.48	0.51	0.53	0.55

Table 4: Relative usages of the respective mechanism in the *main* dataset on the respective date. A value of `x` can be translated into: an SPF record has on average x occurrences of the respective mechanism/modifier/qualifier.

6.3 Most Included Domains

The `include` mechanism allows domain administrators to include SPF records of other domains into their own SPF record. Some domains are linked more often than others. Table 5 shows the top ten most included domains in the *main* dataset on the first of May 2015, while Table 6 shows the top ten most included domains on the first of May 2020. For each included domain, we show the total number of registered domain names that link to it.

The first point of notice is that `spf.protection.outlook.com` is located in more than 11% of the `include` mechanisms in the *main* dataset of May 2020. This confirms the significant position that Microsoft has with its Office 365 service. Not only Microsoft is significant in that sense, but Google also has a large position with its G Suite service. More than 7% of the `include` mechanisms in all SPF records include Google's G Suite service. Both services are part of so-called Software as a Service (SaaS). In general, the top ten of most included domains consists of two types of categories: `SaaS providers` and *hosting providers*. Table 6 displays that three out of the ten most included domains are part of the SaaS providers category: `spf.protection.outlook.com`, `_spf.google.com`, and `relay.mailchannels.net` (Zendesk). All of these three domains have a service which requires domain administrators to include their SPF record for the SaaS to work correctly. Google requires to include `_spf.google.com` for their G Suite service to be enabled and function properly. Microsoft requires to include `spf.protection.outlook.com` for their Office 365 service to be enabled and function properly. Zendesk requires to include `relay.mailchannels.net` for their MailChannels Outbound Filtering to be enabled and function properly.

Table 6 displays that the remaining seven out of the top ten most included domains can be placed in the *hosting providers* category: `websitewelcome.com` (Hostgator), `spf.efwd.registrar-servers.com` (Namecheap), `mx.ovh.com`, `secureserver.net` (GoDaddy), `bluehost.com`, `_spf.mailspamprotection.com` (Siteground), and `spf.mxhichina.com`. The reason why these domains are often included is because when a domain is hosted using the nameservers of the respective hosting provider, most often it automatically sets up an SPF record including the respective domain.

Position	Domain	Count
1	websitewelcome.com	2,441,594
2	bluehost.com	1,205,255
3	_spf.google.com	1,076,069
4	spf.protection.outlook.com	933,511
5	secureserver.net	753,799
6	spf.efwd.registrar-servers.com	723,636
7	mx.ovh.com	552,455
8	hostmonster.com	290,354
9	spf.mandrillapp.com	263,238
10	emailsrvr.com	173,342
14	aspmx.googlemail.com	117,356

Table 5: Most included domains in the *main* dataset (26,397,236 domains with SPF) on the first of May 2015

Position	Domain	Count
1	spf.protection.outlook.com	4,674,246
2	_spf.google.com	3,023,218
3	websitewelcome.com	2,428,704
4	spf.efwd.registrar-servers.com	2,374,471
5	mx.ovh.com	1,006,800
6	secureserver.net	852,760
7	bluehost.com	804,787
8	relay.mailchannels.net	757,034
9	_spf.mailspamprotection.com	658,300
10	spf.mxhichina.com	406,201
49	aspmx.googlemail.com	69,574

Table 6: Most included domains in the *main* dataset (42,144,226 domains with SPF) on the first of May 2020

6.4 Summary Research Question 1

Up to now, we determined the adoption of SPF in multiple datasets, the usages of mechanisms, qualifiers, and modifiers, and we determined the most included domains. The main findings of our analyses of adoption rates are that the adoption rates of SPF are increasing over time and are still increasing. However, there is a vast difference between the different datasets. The `.se` dataset has around the same adoption percentage (+-25%) as the *main* dataset. However, comparing the `.se` dataset (26%) with the `.ee` dataset (58%) we saw a vast difference. The vast difference between the percentages of adoption is the result of Estonia wanting to become a cybersecurity leader and therefore investing more resources into e-mail security. The *Alexa* dataset had the highest adoption of SPF (74%). A possible explanation for this high adoption of SPF was that domains providing services to large user bases are more likely to be operated by entities that make security considerations than the majority of domain names.

The second interesting part we looked at is the mechanisms, qualifiers, and modifiers. We looked at how often each of the mechanisms, qualifiers, and modifiers were used. We saw that there are five popular mechanisms (`a`, `all`, `include`, `ip4`, and `mx`), while all other mechanisms are much less used. The most used qualifier was the `+` qualifier. However, the number of usages of the `+` qualifier in Table 4 are the number of explicit usages using the `+` qualifier and not taking into account the mechanisms that have no qualifier attached to them. Taking into account the number of mechanisms that have no qualifier attached to them resulted in a much higher number (2.16) of usages of the `+` qualifier. More details about the combination of qualifiers and mechanisms will be explained in Section 6.6.

The last point we discussed was the most included domains. We noticed that more than 11% of the `include` mechanisms link to Microsoft's Office 365 service and more than 7% of the `include` mechanisms link to Google's G Suite service in the *main* dataset of May 2020. The second point we noticed is which types of companies are behind the most ten included domains. We placed all domains in the top ten most included domains in two categories: *hosting providers* and *Software as a Service (SaaS) providers*.

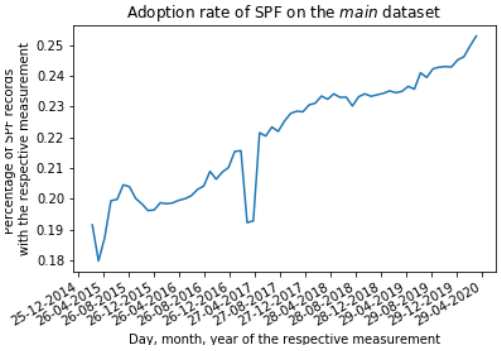

Figure 1: Adoption rate of SPF on the *main* dataset

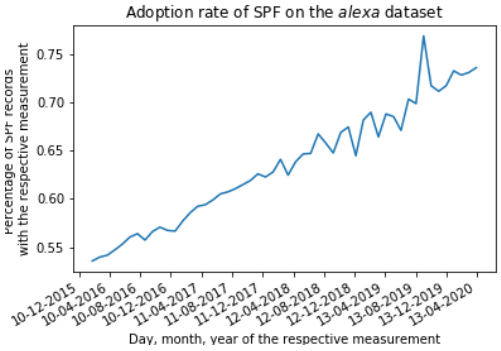

Figure 2: Adoption rate of SPF on the *Alexa* dataset

6.5 Results Research Question 2

We defined the second research questions as: *Can we identify a change in the use of SPF over time? If yes: How do SPF records change over time?*. To identify a change in the use of SPF over time, we will take a look at how the adoption of SPF changed over time, how the characteristics and usages of mechanisms, qualifiers, and modifiers have changed over time, and what happened to the top ten most included domains over time.

6.5.1 Adoption Rate

As we mentioned in Section 6.1.1, Stefan Görling found an adoption rate of 1.63% among the Swedish domains (`.se`) in February 2006. We calculated an adoption rate of 26.39% among the Swedish domains (`.se`) in May 2020, an improvement of around 25% points in fourteen years. However, as previously mentioned the OpenINTEL project contains data of more TLDs (e.g. `.com`, `.org`, and `.net`). The OpenINTEL project has data starting from the twenty-eighth of February 2015 up to now of these TLDs (see Table 1), which means we were able to determine the change in adoption rate for five years.

Figure 1 shows the adoption rate of SPF among the domains in the *main* dataset, starting from March 2015 up to May 2020. A clear increase is visible in the adoption rate, starting from around 19% in 2015 up to around 25% in May 2020. That is a relative increment of around 6% points in five years. When switching to the *Alexa* dataset, a relative increment is visible of around 20% points (Figure 2). The *Alexa* dataset's adoption rate has increased relatively more than three times (6% vs. 20%) compared to the *main* dataset over (almost) the same period.

A similarity between the two datasets (*main* and *Alexa*) is that they both show an increasing trend, in which the *Alexa* dataset has shown relatively the most increase in adoption rate of SPF of around 20% points. The histories and trends of the adoption rates are displayed in Figures 1 and 2. Figure 1 displays the adoption rate of the *main* dataset and Figure 2 displays the adoption rate of the *Alexa* dataset.

Looking at the graph of the *main* dataset, we can see something diverge. The visible divergence is the valley/drop between March and May 2017. In that specific period, there is a sudden decrease in the adoption rate, which stays for around two months. We investigated which domains cause this drop and found that many domains had a nameserver of the largest hosting provider of the world: GoDaddy [6]. The date when this valley/drop started is on the sixth of March 2017, and the valley/drop ended on the fourth of May 2017. Table 7 displays the differences between the day before the drop, the day of the drop, the last day of the drop, and the first day after the drop has ended on the *main* dataset.

This drop turns out to be a measurement artefact: domain names for which GoDaddy's DNS infrastructure is authoritative - and there are many of such domains as GoDaddy is a sizable registrar - saw measurement interruption temporarily.

Date	Total Domains	Domains with SPF	Adoption Rate
05-03-2017	151,438,818	32,443,906	21.42%
06-03-2017	151,391,050	28,654,755	18.93%
03-05-2017	151,375,575	29,311,599	19.36%
04-05-2017	151,497,312	33,134,346	21.87%

Table 7: Data of the valley/drop in the *main* dataset between the fifth of March and the third of May 2017

6.5.2 Government Instances

In the Introduction of this Thesis (see Section 1) we described that the United States Department of Homeland Security (DHS) released the Binding Operational Directive (BOD) 18-01 related to enhancing e-mail and web security on the seventeenth of October 2017. One of the requirements of BOD 18-01 is that government agencies are required to implement SPF within 90 days. We analysed all the `.gov` TLD domains issued by the United States over the course of that period. Table 8 shows the results of this analysis. Before the release of the BOD, the adoption of SPF among the `.gov` domains was 51.06%. The adoption of SPF rose to around 55% in the 14 days following the BOD release. This increase in adoption of SPF continues, and after around three and a half months, the adoption of SPF increased in total with around 24% points. On the first of February 2018, 74.88% of the `.gov` domains had an SPF record.

Date	Domains With SPF	Total Domains	Adoption Rate
01-10-2017	652	1277	51.06%
01-11-2017	714	1266	56.40%
01-12-2017	745	1263	58.99%
01-01-2018	841	1257	66.91%
01-02-2018	924	1234	74.88%

Table 8: Adoption rate on the .gov dataset before and after the release of the BOD

To summarise this section: the adoption rate of all analysed datasets has increased over time; there is a vast difference in adoption rates between the `.se` dataset and the `.ee` dataset due to investing in cybersecurity; government directives cause the adoption rate to increase.

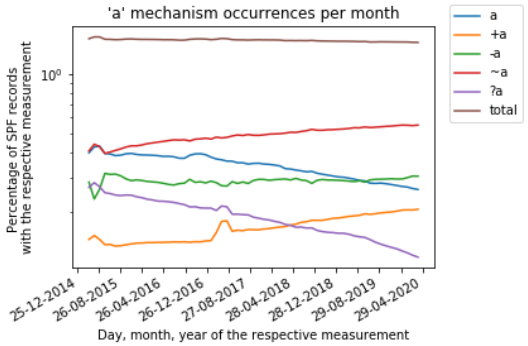

Figure 3: Relative usages of the `a` mechanism in the *main* dataset

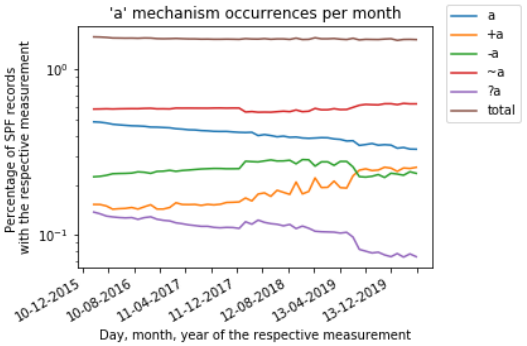

Figure 4: Relative usages of the `a` mechanism in the *Alexa* dataset

6.6 Mechanisms, Qualifiers, and Modifiers Over Time

To determine the change over time of SPF records, we analysed the usages of the mechanisms, qualifiers, and modifiers and the change in usages over time. The next point we analysed is how the usages of qualifiers attached to a mechanism have changed over time. With these results, we will be able to determine how SPF records are set up and how SPF records have changed over time. The next sections describe how the usages of the mechanisms, qualifiers, and modifiers have changed over time, and how the combinations of qualifiers and mechanisms have changed over time.

6.6.1 A

The `a` mechanism has seen a small decrease of relative usages over time in the *main* dataset as the brown line shows in Figure 3. The same is true for the *Alexa* dataset as shown by the brown line in Figure 4. However, the decrease of relative usages is smaller than in the *main* dataset.

The `+a` mechanism has seen a huge increase of relative usages over time in both datasets as shown by the orange lines in both Figures. However, we also notice a difference in relative usages of the `+a` mechanism: in the *main* dataset the `+a` mechanism started of as the least frequently used of all qualifier and `a` mechanism combinations, while in the *Alexa* dataset the `+a` mechanism started of as the second least frequently used qualifier and `a` mechanism combination. In the *main* dataset the `+a` mechanism has seen a rise and at the first of May 2020 the `+a` mechanism was the second least frequently used qualifier and `a` mechanism combination. However, in the *Alexa* dataset the `+a` mechanism started as the second least frequently used qualifier and `a` mechanism combination, but ended as the third frequently used qualifier and `a` mechanism combination.

However, since the `a` and `+a` mechanisms are resulting in the same behaviour of the SPF record (see Section 2), we analyzed if the increase of the `+a` mechanism and the decrease of the `a` mechanism compensate each other. They do not compensate each other: the decrease of the `a` and `+a` mechanisms combined is around 11% in five years. The 11% decrease is more than the overal decrease of the `a` mechanism, as shown in Table 4.

The other combinations of qualifiers and the `a` mechanism show the same trends in the *main* dataset and the *Alexa* dataset. One point to notice are the plateaus of the `-a` and `?a` mechanism from January 2018 up to July 2019 in the `Alexa` dataset. These plateaus do not occur in the `main` dataset and we were not able to explain this kind of behaviour in the `Alexa` dataset.

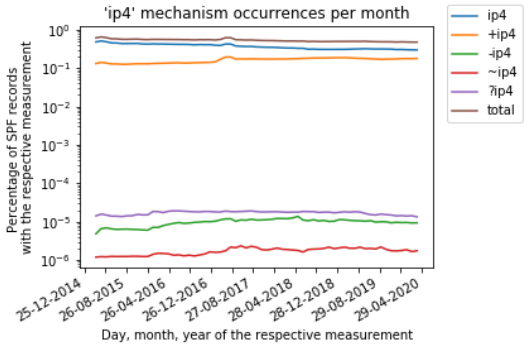

Figure 5: Relative usages of the `ip4` mechanism in the combined *main* dataset

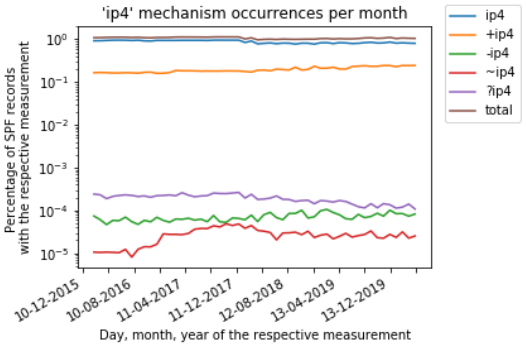

Figure 6: Relative usages of the `ip4` mechanism in the *Alexa* dataset

6.6.2 IP4

The `ip4` mechanism shows a decrease in relative usages in both datasets. The small decrease is visible by the brown lines in Figures 5 and 6. A point to notice is that the combinations of qualifiers and `ip4` mechanism show the same trends in both datasets.

A point to notice between the two datasets is that the relative usages of the `ip4` mechanism are higher in the *Alexa* compared to the *main* dataset. Another point to notice is the relatively high usages of the `+` qualifier or no qualifier in combination with the `ip4` mechanism. The brown (total) line in both Figures follows the trends of the two most used mechanisms: `ip4` and `+ip4`. The other mechanisms are much less frequently used compared to the `ip4` and `+ip4` mechanisms. This behaviour was expected because it stands to reason that operators are more likely to specify, e.g. IP addresses that are permitted to send e-mails than they are to specifically define IP addresses that are not.

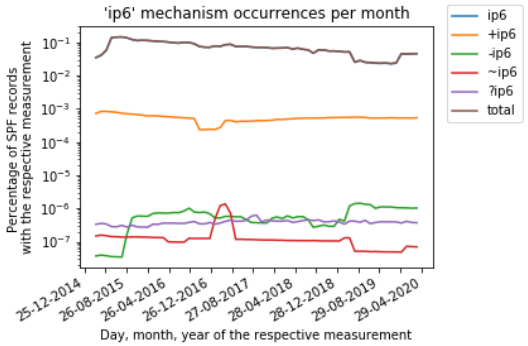

Figure 7: Relative usages of the `ip6` mechanism in the combined *main* dataset

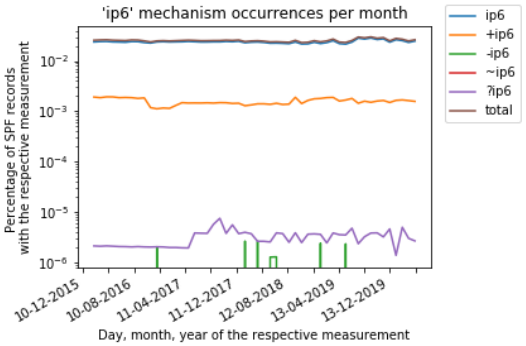

Figure 8: Relative usages of the `ip6` mechanism in the *Alexa* dataset

6.6.3 IP6

The `ip6` mechanism shows an interesting trend as can be seen from the blue (`ip6`) and brown (total) lines in Figures 7 and 8. The brown and blue lines closely follow each other and in the case of the *main* dataset (Figure 7) the blue line is totally shadowed by the brown line. This behaviour can be explained by the fact that the combined usages of the `ip6` and `+ip6` mechanisms are by far the most used mechanisms. In the case of the *main* dataset the difference in relative usages of the `ip6` mechanism and `+ip6` mechanism is relatively larger compared to the *Alexa* dataset and therefore the total (brown) line completely shadows the `ip6` (blue) line in the *main* dataset. Since the difference in relative usages in the *Alexa* dataset between the `ip6` mechanism and the `+ip6` mechanism is smaller, the brown line does not completely shadow the blue line, but is placed a bit above the `ip6` (blue) line.

The next point to notice are the relatively low usages of the `-ip6`, `ip6`, and `?ip6` mechanisms in both datasets. In the *Alexa* dataset, not all mechanisms are even visible due to these very low usages and using a logarithmic scale on the y-axis. In the *main* dataset only three times a `~ip6` mechanism was used on the first of May 2020. On the same date, zero times the `~ip6` was used in the *Alexa* dataset. The low usage of the `~ip6` mechanism makes sense, since an administrator typically uses the `ip6` mechanism to allow (e.g. `+` qualifier) certain IP addresses instead of using a SOFTFAIL (~) qualifier.

The last difference to discuss between the two datasets is the difference in trends of the total usages of the `ip6` mechanism. The *main* dataset shows a down going trend in general over time, while the *Alexa* dataset shows an almost constant relative use of the `ip6` mechanism over time. This result is counter-intuitive: since IPv4 addresses are exhausted [8] and the use of IPv6 addresses is increasing [18]. In our view, this should mean that the use of the `ip6` mechanism would increase over time. However, the opposite is true as we just explained. A reason why the usage of the `ip6` mechanism is decreasing in the *main* dataset could be because domain administrators preferably like to allow a more general mechanism (i.e. `a` or `mx`) instead of mechanism that allows only a specific IP address (i.e. `ip4` or `ip6`). However, this is just a hypothesis, detailed analysis should be performed to find out if this hypothesis is true.

We assume that large cloud e-mail providers are early and large adopters of IPv6. Therefore, we analysed the *chained* SPF record (see Section 6.9) and looked at two large cloud e-mail providers: Google (Gmail) and Microsoft (Outlook). For both cloud e-mail providers, we analysed the number of `ip6` mechanisms in their *chained* SPF record. Google uses six `ip6` mechanisms and 21 `ip4` mechanisms. Microsoft uses six `ip6` mechanisms and eight `ip4` mechanisms. With these results, we can infer that large cloud e-mail providers are not adopting IPv6 as much as expected.

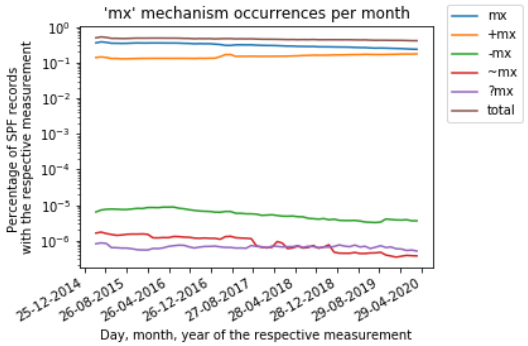

Figure 9: Relative usages of the `mx` mechanism in the *main* dataset

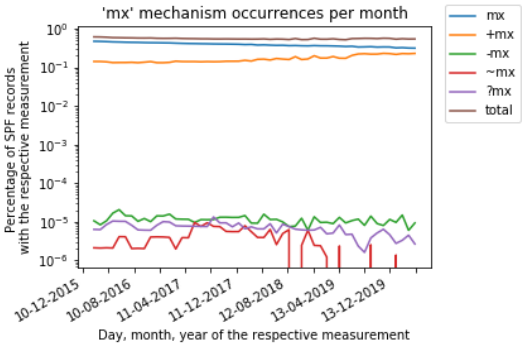

Figure 10: Relative usages of the `mx` mechanism in the *Alexa* dataset

6.6.4 MX

The use of the `mx` mechanism shows a very small decrease over time in both datasets as shown by the brown lines in Figures 9 and 10. Another point to notice is the behaviour of the `mx` and `+mx` mechanisms in both datasets. The `mx` mechanism shows a small down going trend and the `+mx` mechanism shows an up going trend in both datasets. Since the `mx` and the `+mx` mechanisms have the same behaviour (see Section 2 for more detail), we analyzed if the decrease in use of the `mx` mechanism is compensated by the increase of the `+mx` mechanism. After analysing the data, we found out that the combined occurrences of the `mx` mechanism and the `+mx` follow the general trend of the total usages (brown lines) in Figures 9 and 10. This makes sense, due to the very low usages of the `-mx`, ∼`mx`, and `?mx` mechanisms, the brown (total) line follows the general trend of the `mx` and the `+mx` mechanisms.

The next point to notice is how the lines of the `-mx`, `mx`, and `?mx` are displayed in both Figures. In the *main* dataset these three lines representing their respective mechanism are smooth compared to their respective lines in the *Alexa* dataset. The *Alexa* dataset shows many fluctuations in the relative usages of these mechanisms. The fluctuations are created due to the frequent changes in the Alexa top 1M list. The Alexa top 1M list contains the top 1 million best-visited domains. Therefore, the domains which are included in the tail of the Alexa top 1M list are changed frequently. For example, on the first of May 2020, a bit more than 400,000 domains were changed in the Alexa top 1M list compared to the list on the first of April 2020. Therefore, we can infer that the fluctuations in the usages of the `mx` mechanism are caused due to frequent changes in the *Alexa* dataset.

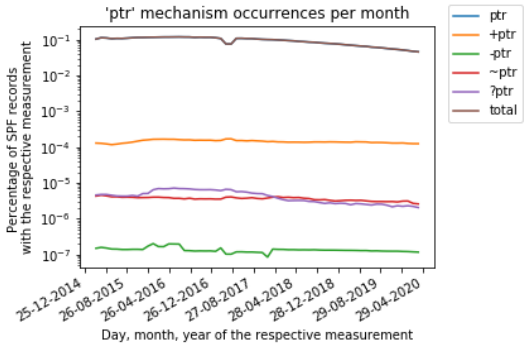

Figure 11: Relative usages of the `ptr` mechanism in the combined *main* dataset

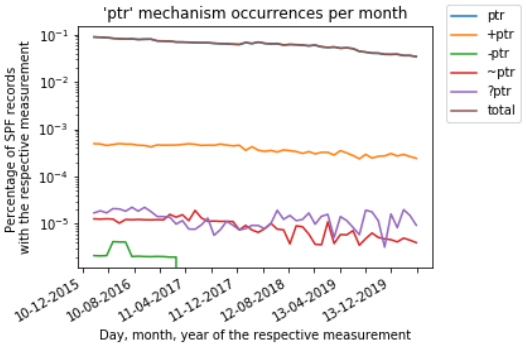

Figure 12: Relative usages of the `ptr` mechanism in the *Alexa* dataset

6.6.5 PTR

The `ptr` mechanism shows a common pattern which we have seen before for the `ip6` mechanism. The total (brown) lines in Figures 11 and 12 show a down going trend, whereby the total (brown) line shadows the `ptr` mechanism (blue line). The same reason as was given at the `ip6` mechanism can be applied to the `ptr` mechanism: the relative usage of the `ptr` mechanism is so high that the total usages (brown line) of the `ptr` mechanism shadows the `ptr` mechanism (blue line). This is due to the fact that usages of the `+ptr`, `-ptr`, ~`ptr`, and `?ptr` mechanisms are so low, the total (brown lines) usages follow the trends/usages of the `ptr` mechanism. In total the `+ptr`, `-ptr`, ~`ptr`, and `?ptr` mechanisms are used about 5.5 thousand times (out of almost two million total usages of the `ptr` mechanism) in the *main* dataset on the first of May 2020.

The `ptr` mechanism is discouraged from being used by administrators. However, almost two million domains use the `ptr` mechanism in their SPF record. We analysed the similarities between the SPF records which use the `ptr` mechanism. We found out that two hosting providers (GoDaddy and Bluehost) have a default SPF record, which uses the `ptr` mechanism. Out of the almost two million SPF records which use the `ptr` mechanism in the *main* dataset, a bit more than 1.4 million domains use the default SPF record of GoDaddy or Bluehost. This means that around three-quarters of the domains that use the `ptr` mechanism are using the default SPF record of GoDaddy or Bluehost.

The second point to notice is that the lines of the `-ptr`, `ptr`, and `?ptr` mechanisms fluctuate a lot more in the *Alexa* dataset compared to the relatively smooth lines of the *main* dataset. This observation can be explained the same as the fluctuations with the `mx` mechanism (see Section 6.6.4): the fluctuations are caused due to frequent changes in the Alexa top 1M list.

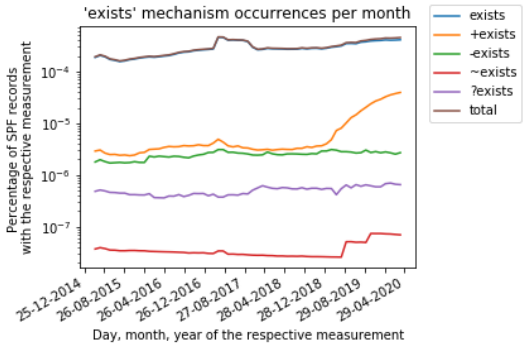

Figure 13: Relative usages of the `exists` mechanism in the combined *main* dataset

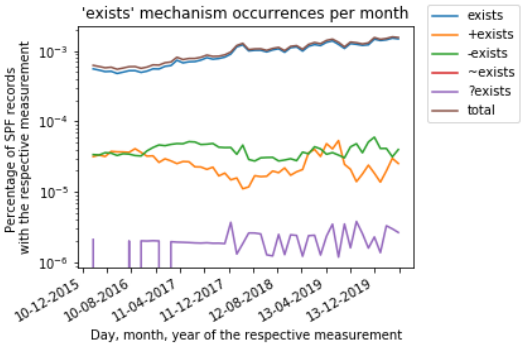

Figure 14: Relative usages of the `exists` mechanism in the *Alexa* dataset

6.6.6 EXISTS

In both datasets the `exists` mechanism shows an increase of relative usages over time as can be seen from the brown lines in Figures 13 and 14. The `exists` (blue) line has been almost shadowed by the total (brown) line in the *main* dataset. In the *Alexa* dataset the total (brown) line almost shadows the `exists` (blue) line. However, there are larger differences between the two lines (brown and blue) in the *Alexa* dataset, therefore both lines are (almost) always visible.

The `+exists` mechanism has shown a sudden rise in relative usages in the *main* dataset starting from around February 2019. The `+exists` mechanism is not the only mechanism who showed this kind of sudden rise in relative usages. The `~exists` mechanism also shows this kind of behaviour. However, instead of that the rise starts around February 2019, the rise of the `~exists` mechanism starts a bit later around April 2019.

The sudden rise of relative usages of the `+exists` does not occur in the *Alexa* dataset. Therefore, we first expected that one of the major hosting providers adjusted their nameserver and thereby adding a `+exists` mechanism to their SPF record. However, this was not the case when we analysed the adjusted SPF records. However, we noticed that many domains adjusted their SPF record to include: `+exists:%i.spfcheck.eu`. Searching for information about `spfcheck.eu` resulted in no usable information. However, the whois lookup of `spfcheck.eu` resulted in the organisation *Freemium Kft.* located in Budapest, Hungary. Visiting their domain (`freemium.hu`) resulted in an online domain and DNS management system. We think that *Freemium Kft.* introduced a new product to verify (as `spfcheck.eu` suggests) SPF records of their customers and that this product increased the number of `+exists` mechanism usages.

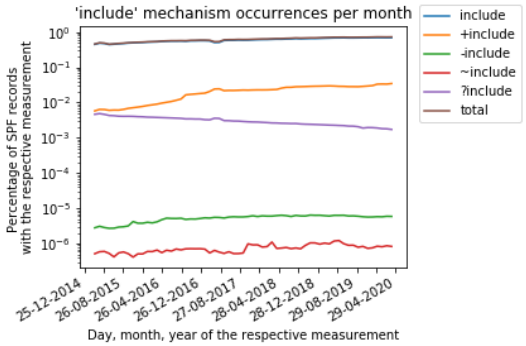

Figure 15: Relative usages of the `include` mechanism in the combined *main* dataset

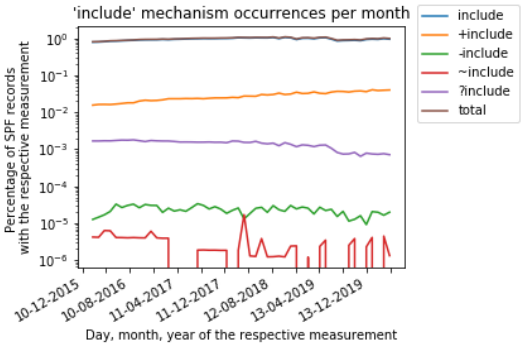

Figure 16: Relative usages of the `include` mechanism in the *Alexa* dataset

6.6.7 INCLUDE

The `include` mechanism has shown a small up going trend in both datasets as can be seen by the brown lines in Figures 15 and 16. In both Figures the total (brown) lines shadow the `include` (blue) lines. The reason of the shadowing is the same as the `exists`, `ptr`, and `ip6` mechanism: the relative usage of the `include` mechanism is so high that the total usages (brown line) of the `include` mechanism shadows the `ptr` mechanism (blue line). This is due to the fact that usages of the `+include`, `-include`, ∼`include`, and `?include` mechanisms are so low, therefore the total (brown lines) usages follow the trends/usages of the `include` mechanism.

A point to notice is the difference in relative usages overtime of the `+include` and the `?include` mechanism. In both datasets the `+include` has an up-going trend and the `?include` has a down going trend. However, looking at both start dates of the two datasets, we see a huge difference. The two mechanisms (`+include` and `?include`) in the *main* dataset are almost at the same relative usages at the first of March 2015. The two mechanisms in the *Alexa* dataset are much more separated at the first of February 2016.

The increase over time of the `include` mechanism might be related to the increase of cloud e-mail providers (see Section 6.3). More domains are using cloud e-mail providers or other Software as a Service (SaaS) providers, and they require administrators to include their SPF record to let the service work properly.

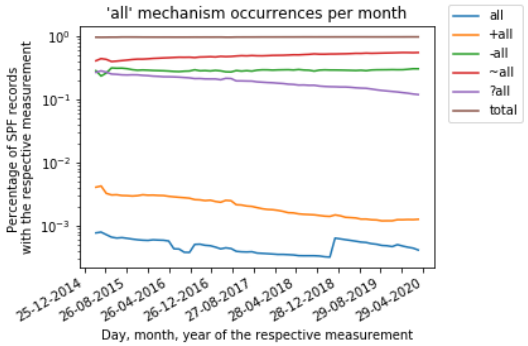

Figure 17: Relative usages of the `all` mechanism in the combined *main* dataset

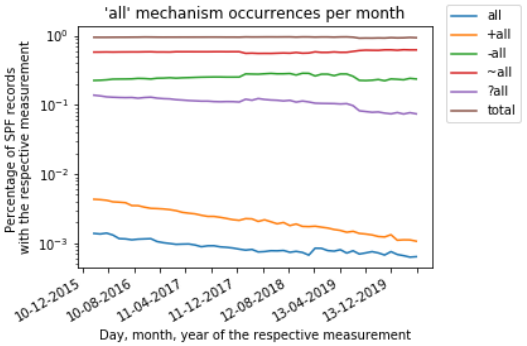

Figure 18: Relative usages of the `all` mechanism in the *Alexa* dataset

6.6.8 ALL

The `all` mechanism shows a constant line of all the relative usages over time as can be seen by the brown lines in Figures 17 and 18. Noticeable in both datasets is the relatively low usage of the `all` mechanism and the `+all` mechanism. The low usages of the `all` mechanism and the `+all` mechanism is noticeable since most of the previous sections described that the mechanism without a qualifier and their respective mechanism in combination with a `+` qualifier were the most used mechanisms. This makes sense since using a `+all` mechanism causes the sender's IP address always to match the record. Therefore, the e-mail will be declared valid and presumably forwarded to the inbox of the receiver. Using a `+all` mechanism, therefore introduces (in most cases) a security risk. Therefore, most often we see the `all` mechanism in combination with a ∼, `-`, or `?` qualifier.

Another point to notice is that the `-all`, ∼`all`, and `?all` mechanisms show the same behaviour between the *main* dataset and the *Alexa* dataset. The `-all` mechanism has shown an almost constant trend. The same is true for the ∼`all` mechanism. However the `?all` has shown a down going trend over the time in both datasets.

The ∼, `-`, and `?` qualifiers have seen a change over time in the *main* dataset. Around March 2015, the usages of each qualifier were close to the others. However, when time passes, the three qualifiers have diverged. The ∼ mechanism saw a small increase over time, the `-all` mechanism was almost constant over time, and the `?all` mechanism saw a decrease in usages over time.

Overall, we see that almost every SPF record uses an `all` mechanism which is visible by the brown lines in Figures 17 and 18. Most of the `all` mechanisms are combined with the ∼ and `-` qualifier. The high usages of the ∼ and `-` qualifiers make sense since most of the SPF policies are set up by allowing certain IP addresses and otherwise drop or tag the incoming e-mail.

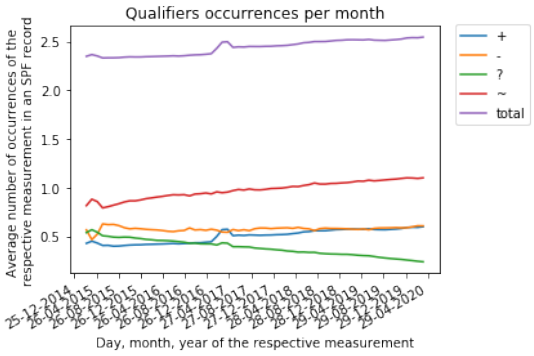

Figure 19: Average usages of the four qualifiers in the *main* dataset

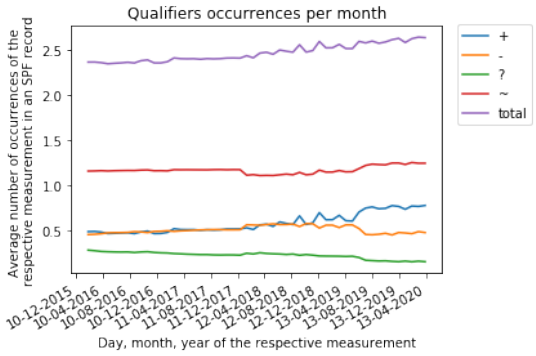

Figure 20: Average usages of the four qualifiers in the *Alexa* dataset

6.6.9 Qualifiers

The relative usages of the four qualifiers in both datasets are displayed in Figures 19 and 21. In general, we see an increase in relative usages of qualifiers in an SPF record which is shown by the total (purple) lines in both Figures.

The first qualifier we will discuss is the `+` qualifier. In both datasets, the `+` qualifier has shown a vast increase, and as of the first of May 2020 the `+` qualifier is the most used qualifier in general. The high use of the `+` qualifier shows that most mechanisms are applied to allow the use of the mechanism. The only exception is for the `all` mechanism. The `all` mechanism is used by most administrators to FAIL (`-`) or to SOFTFAIL (∼) the e-mail if no mechanism match.

Next is the `-` qualifier. Looking at the beginning and the end of the orange lines in the *main* dataset, it is visible that the relative usages of the `-` qualifier increased over time. The same is not true for the relative usages of the `-` qualifier in the *Alexa* dataset. In the *Alexa* dataset, the `-` qualifier started with an increase over time, but this ended around August 2019. After August 2019, the relative usages dropped below the initial value of the first of February 2016.

The third qualifier to discuss is the `?` qualifier. In both datasets the `?` qualifier has seen a significant drop in relative usages over time as can be seen in Figures 19 and 20. In the *main* dataset the `?` qualifier had a few months at the beginning of the graph where the relative usages of the `?` qualifier was above the relative usages of the `-` qualifier. The relative usages of the `?` qualifier have not been above the relative usages of the `-` qualifier in the *Alexa* dataset. In the *Alexa* dataset, the relative usages of the `?` qualifier have always been lower than the relative usages of the `-` qualifier. The low usages of the `?` qualifier can be related to the fact that is using a `?` qualifier means that when the mechanism match, the e-mail should be interpreted as if there was no SPF policy. In our view, we think that administrators typically configure an SPF policy by using mechanisms that pass the e-mail and drop otherwise if none of the mechanisms matches.

The last qualifier to discuss is the ∼ qualifier. The relative usages overtime of the ∼ qualifiers are different between the two datasets. In the *main* dataset the relative usages of the ∼ qualifier are steadily increasing, while in the *Alexa* dataset the relative usages of the ∼ qualifier are almost constant. However in both datasets, there are periods where the relative usages of the ∼ qualifier are higher than the relative usages of the `+` qualifier, but there are also periods where the opposite is the case. Looking at the present, in both datasets, the relative usages of the `+` qualifier are higher than the relative usages of the ∼ qualifier.

In the preceding analysis, we counted the qualifiers irregardless of the mechanisms with which they are combined (e.g. `+all` and `+ip` both increase the `+` qualifier count). In Section 5.3 we explained that the we will also perform an analysis without the occurrences of the qualifiers which are attached to the `all` mechanism. When we do not count the occurrences of the four qualifiers attached to the `all` mechanisms, the results look at a first glance the same (see Figures 19, 20, 21, and 22). However, the y-axis reveals that all lines are shifted down. The next data points are taken from the *main* dataset, however, the *Alexa* dataset has almost the same behaviour as the *main* dataset. Therefore, the upcoming assertions can also be applied to the *Alexa* dataset. The total (purple) line is down from around 2.5 qualifiers in an SPF record on average to around

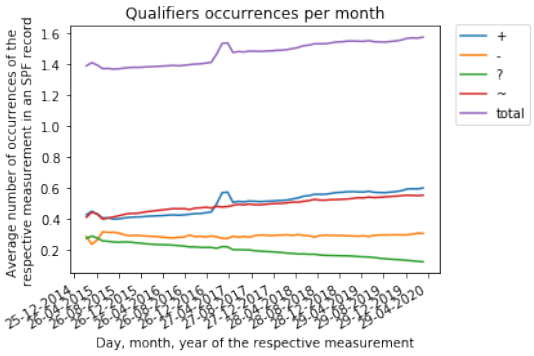

Figure 21: Average usages of the four qualifiers without the occurrences of qualifiers attached to the `all` mechanisms in the *main* dataset

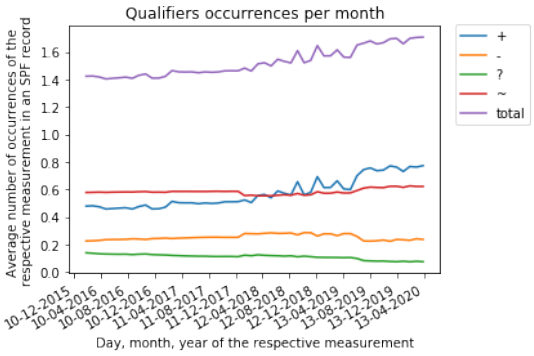

Figure 22: Average usages of the four qualifiers without the occurrences of qualifiers attached to the `all` mechanisms in the *Alexa* dataset

1.6 qualifiers. It makes sense that the average number of qualifiers in an SPF record is down by almost one since almost all SPF records have an `all` mechanism (see Section 6.2) at the end of their SPF record to determine how to handle IP addresses that did not match any of the mechanisms before the `all` mechanism.

The `-` qualifier (orange line) has seen a drop from around 0.6 usages to around 0.3 usages on average per SPF record. This decrease means that around half of the usages of the `-` qualifier is due to the `-all` mechanisms. The `?` qualifier has seen the same behaviour as the `-` qualifier when not taking into account the usages of the qualifiers attached to the `all` mechanisms. The `?` qualifier has also seen a drop which caused the average usages of the `?` qualifier to be dropped by half. The occurrences of the `?` qualifier went from around 0.4 to around 0.2 usages on average in an SPF record. The $\sim$ qualifier has also seen a drop from around 1.1 to around 0.5 usages on average per SPF record. The $\sim$ qualifier is the third qualifier which has seen a drop which caused the average usages to be dropped by half. The last qualifier to discuss is the `+` qualifier. The `+` qualifier has seen, in contrast with the other qualifiers, almost the same average usages as when we keep counting the qualifiers attached to the `all` mechanisms. This is as expected, since the usages of the `+all` mechanism are extremely low (see Figures 17 and 18). Therefore, removing the usages of the `+` qualifier attached to the `all` mechanism is almost negligible.

From these results, we can infer that the usages of the `-`, `?`, and $\sim$ qualifiers are around 50% of the time combined with the `all` mechanism. The combination of the `-` and $\sim$ qualifiers with the `all` mechanism, means that SPF policies are set up to allow certain IP addresses and otherwise drop or tag the incoming e-mail. The `?` qualifier combined with the `all` mechanism means that the recipient's e-mail server has to handle the incoming e-mail as if there was no SPF policy. On the first of May 2020, a bit more than five million SPF records used a `?all` qualifier in the *main* dataset. In those SPF records, 11.2 million times a `+` qualifier was explicitly or implicitly used. The `-` qualifier was used almost 1.8 thousand times, the $\sim$ qualifier 3.5 thousand times and the `?` qualifier was used 10.2 thousand times. From these results, we can derive that most SPF policies that use the `?all` mechanism are set up to allow certain IP addresses and not to block certain IP addresses. Therefore, we can infer that most SPF policies are set up to allow specific IP addresses and otherwise drop or tag the e-mail.

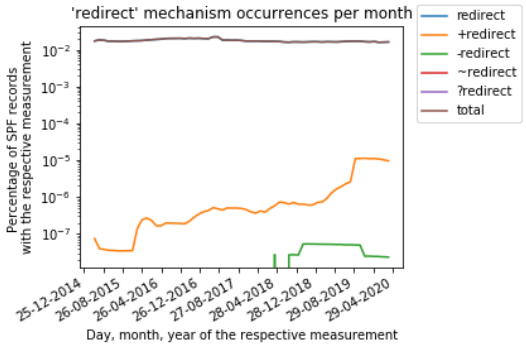

Figure 23: Relative usages of the `redirect` mechanism in the combined *main* dataset

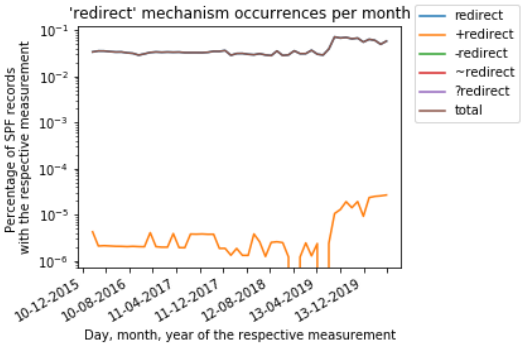

Figure 24: Relative usages of the `redirect` mechanism in the *Alexa* dataset

6.6.10 Modifiers

Recall from the background section (see Section 2) that next to the mechanisms and qualifiers, there are also modifiers. Specifically, the `redirect` and `exp` modifiers. In the following sections, we will explain the usages of the modifiers in detail.

6.6.11 Redirect

Figures 23 and 24 show the relative usages of the `redirect` modifier. The total (brown) line shadows the `redirect` (blue) line in both Figures. This is because a modifier can only be used without a qualifier. However, it is visible that there are SPF records that use the `redirect` modifier in combination with a qualifier. SPF records which use a modifier in combination with a qualifier should be declared invalid. On the first of May 2020, there were 412 (out of 694,899) `redirect` modifiers used in combination with a qualifier in the *main* dataset. A qualifier can only be used in combination with a mechanism. Therefore the SPF records which have a combination of a qualifier and a modifier should be declared invalid. Validation of SPF records is part of the analysis which we will explain later in Section 6.8.

The second point to notice is the difference in change over time of relative usages between the *main* dataset and the *Alexa* dataset. Whereas the *main* dataset shows a small decrease of relative usages over time, the *Alexa* shows a noticeable increase over time. A possible explanation for this behaviour in the `Alexa` dataset can be that popular domain names often have multiple domains. When required to adjust all the SPF records related to a domain, it is easier and quicker to adjust only one SPF record instead of multiple. Using a `redirect` modifier makes it easier to adjust the SPF record in one place since only the *main* SPF record needs to be adjusted.

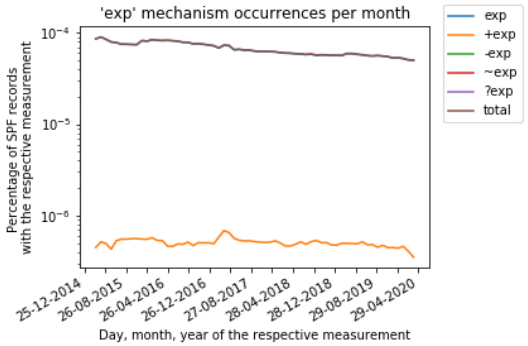

Figure 25: Relative usages of the `exp` mechanism in the *main* dataset

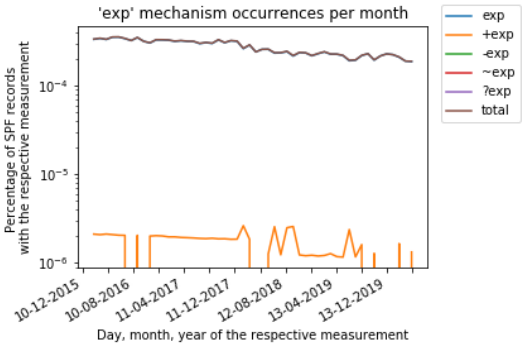

Figure 26: Relative usages of the `exp` mechanism in the *Alexa* dataset

6.6.12 Explanation

Figures 25 and 26 display the relative usages of the `exp` modifier. The first point to notice in these figures is the same as previously described in Section 6.6.11: The `exp` modifier is sometimes used in combination with a qualifier and for this reason these SPF records should be declared invalid. On the first of May 2020, there were 15 (out of 2,158) `redirect` modifiers used in combination with a qualifier in the *main* dataset.

The second point to notice is the downgoing trend of relative usages of the `exp` modifier in both datasets. A downgoing trend seems reasonable since we think that using a `exp` modifier will be used when administrators initially set up an SPF record or when an administrator adjusts an SPF record to debug why a FAIL result is returned. When the testing phase is finished, the `exp` modifier will be removed. We analysed the domains which had an `exp` modifier and see if they have an altered SPF record without an `exp` modifier after a year. The result is that the majority (90%+) of the domains which use an `exp` modifier in May 2019 also use an `exp` modifier in May 2020. Therefore, our hypothesis is wrong, and we are not able to explain the decrease in `exp` modifier use over time.

The lines that display the downgoing trend of the `exp` modifier have different behaviour. The *main* dataset shows this downgoing trend in a more straight line (Figure 25), while this change happens with a lot more fluctuation in the *Alexa* dataset (Figure 26). We believe that the reason for these fluctuations is the same as with the `mx` and `ptr` mechanisms: the domains in the tail of the Alexa top 1M list are frequently changing, and that causes these fluctuations (see Section 6.6.4).

6.6.13 Most Included Domains

In Section 6.3 we described the most popular included domains using the `include` mechanism. We also mentioned that Software as a Service (SaaS) providers require clients to include the SaaS's SPF record to let the SaaS work properly. Figures 5 and 6 display the top ten most included domains in 2015 and 2020 respectively. The first point to notice in these figures is the vast increase of usages of the `spf.protection.outlook.com` domain. The absolute usages of the `spf.protection.outlook.com` domain have been increased by roughly 3.7 million (400,72% of 933,511) in five years. When we normalize the number of domains with SPF against `spf.protection.outlook.com`, we can see a relative usage of 3.7% in May 2015 and 11.4% in May 2020. The increment of `spf.protection.outlook.com` is in common with the trend that van Rijswijk-Deij et al. [35] discuss in their paper: the use of cloud-based e-mail platforms Office 365 (which Outlook belongs to) and Google Apps are increasing in popularity.

Not only Microsoft's Office 365 is increasing in popularity. The same is true for Google's G Suite service. Google's G Suite service increased from almost 1.2 million usages in May 2015 to almost 3.1 million usages in May 2020. With these results, we agree with the trend that van Rijswijk-Deij et al. discuss in their paper, and that is that cloud-based e-mail platforms are increasing in popularity.

6.6.14 Google

We previously explained that SaaS providers sometimes require domain administrators to include the SaaS providers' SPF records for the SaaS work properly. Examples of these SaaS providers are Google (G Suite) and Microsoft (Office 365). To dive deeper into Google's G Suite service: one of the included applications in G Suite is Gmail. With G Suite, users with an e-mail address of the respective domain can use the Gmail application to send and receive e-mails. One of the requirements of using Gmail for a domain is that the domain administrator must include Google's SPF record. When using this `include` mechanism, Gmail is authorized to send e-mails from their servers.

However, there is a catch with the SPF record that Google suggests administrators to include. In March 2009, Google had multiple web pages explaining which SPF record to include. One of the web pages described that `aspmx.googlemail.com` should be included, while another web page instructed to include `_spf.google.com` [22]. It is unclear how long this situation lasted, but known is that `aspmx.googlemail.com` has been depreciated and `_spf.google.com` should be used at least from August 2010 [10].

Since our datasets consist of data starting from March 2015 up to now, we could process for every year the data on the first of March. For the six datasets, we calculated how often `aspmx.googlemail.com` and `_spf.google.com` are included by all domains. Figure 27 displays the results visually, while Table 9 shows the results in a tabular view. The first column of Table 9 displays the year of the measurement, e.g. 2015 means the first of March 2015. The second column displays the number of SPF records in the *main* dataset on the measured date. The third column is the number of occurrences that `aspmx.googlemail.com` is included. The fourth column displays the percentage of SPF records that include `aspmx.googlemail.com`. The fifth column is the number of occurrences that `_spf.google.com` is included. The last column displays the percentage of SPF records that include `_spf.google.com`. Not included in the table are the domains which included both URLs (i.e. 7,323 domains in May 2020).

As can be seen from Figure 27 and Table 9, the relative and absolute include usages of the `aspmx.googlemail.com` URL decreases, and the relative and absolute usages of the `_spf.google.com` URL increases. One might think that still using the old `aspmx.googlemail.com` URL is a security risk, but that is not the case. Google adjusted the SPF record of `aspmx.googlemail.com` to: `v=spf1 redirect=_spf.google.com`. Using a `redirect` modifier means that domains that still include the old (`aspmx.googlemail.com`) URL are redirected to the newer URL (`_spf.google.com`). This also shows a good example of how the `redirect` modifier can be used. However, if the `googlemail.com` would have been expired and was bought by a malicious third party, the malicious third party would have been authorized to send e-mails from the domains which still included the old `aspmx.googlemail.com` URL. Therefore, domain administrators should analyse their SPF policies if included domains have been expired and taken over by (malicious) third parties on a regular basis.

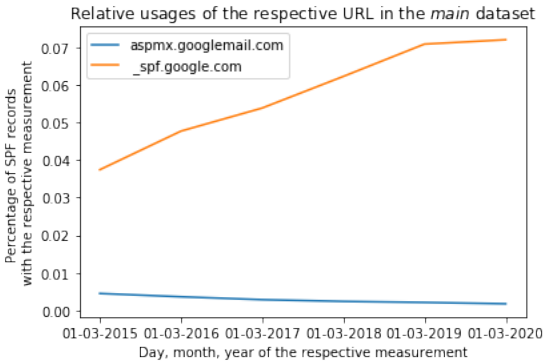

Figure 27: Usages of the `aspmx.googlemail.com` and `_spf.google.com` URL in combination with an `include` mechanism

Date	**Total Domains With SPF**	**Absolute count** `aspmx.googlemail.com`	**Relative Count** `aspmx.googlemail.com`	**Absolute Count** `_spf.google.com`	**Relative Count** `_spf.google.com`
2015	26,832,100	119,974	0.45%	1,004,111	3.74%
2016	29,533,103	106,136	0.36%	1,408,721	4.77%
2017	32,320,540	91,177	0.28%	1,740,065	5.38%
2018	35,569,684	84,125	0.24%	2,214,457	6.23%
2019	37,470,660	78,221	0.21%	2,655,212	7.09%
2020	40,794,209	70,638	0.17%	2,938,003	7.20%

Table 9: The absolute and relative occurrences of `aspmx.googlemail.com` and `_spf.google.com` in the *main* dataset

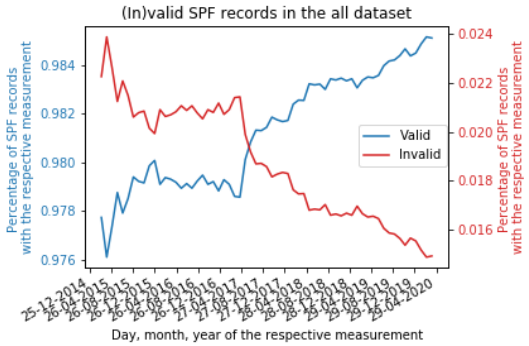

Figure 28: Percentage of (in)valid SPF records in the *main* dataset

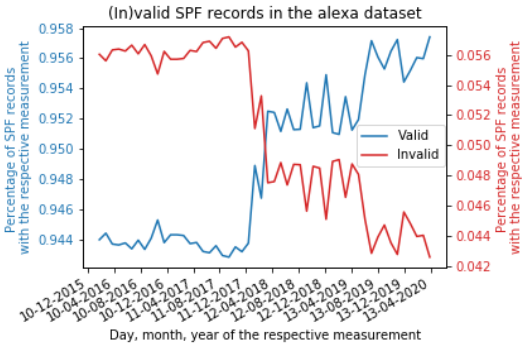

Figure 29: Percentage of (in)valid SPF records in the *Alexa* dataset

6.7 Results Research Question 3

We defined the third research questions as: *Can we recognise problematic trends in SPF use? If yes: What are the characteristics of these problematic trends?*. To determine potential problematic trends, we have taken a look at the RFC of SPF [19]. The RFC displays the requirements an SPF record should apply to. From these requirements, we determined that we wanted to investigate the syntactic requirement and the DNS lookup limits requirement. Both requirements and their analysis will be explained in the following sections.

6.8 Valid and Invalid SPF Records

Not all SPF records are configured accordingly to the standards written in the RFC of SPF [19] and therefore these SPF records may be considered invalid by the validating e-mail server. An invalid record causes the incoming e-mail server unable to validate the IP address of the sender. Therefore, sender address forgery might be possible, and a malicious person could perform phishing. As explained in Section 5.5, there are multiple ways of how an SPF record could be declared invalid. The ways we analysed are: syntactic errors and DNS lookup limits errors. We analysed these two methods since we believe that these are the two most frequent errors. Our *main* dataset consists of around 42 million domains with an SPF record. Using the (in)validation analysis, we can recognise problematic trends in SPF use. Figures 28 and 29 show the percentage of SPF records which our analysis marks as (in)valid in their respective dataset. The blue lines are the percentage of valid SPF records, while the red line displays the percentage of invalid SPF records.

When an SPF record is declared invalid, the response of the query of the SPF can be a `temperror` or a `permerror`. A `temperror` will be returned when there is a transient error, and most often this is a DNS error. The focus of this Thesis is not on the `temperror`. Instead we focus on the `permerror`. A `permerror` is returned when an SPF record could not be correctly interpreted.

6.8.1 Syntactic Errors

One way how a `permerror` could be returned is when there is a syntactic error in the SPF record. An example of a syntactic error is when there is a standalone qualifier without a mechanism. In that case a `permerror` will be returned, and the SPF record will be declared invalid. Table 10 shows that almost 600 thousand SPF records were classified as invalid by our analysis due to syntactic errors in the *main* dataset on the first of May 2020. Table 11 shows the results on the *Alexa* dataset. The first column displays the date of the measurement. The second column displays the number of valid domains. The third column displays the number of invalid domains due to syntactic errors. The fourth column displays the percentage of invalid domains due to syntactic errors in the respective dataset. The last column displays the number of domains with SPF in the respective dataset.

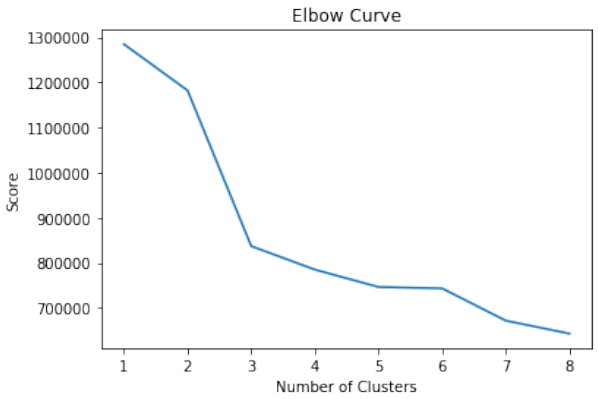

Figure 30: The graph generated using the *elbow* technique.

We expect that some types of syntactic errors occur more frequently than others. Therefore, we decided to cluster the data in groups. Clustering enables us to look on a large scale at the different type of syntactic errors, without having to analyse each invalid SPF record individually. A critical aspect of clustering is to determine the number of clusters. We used the *elbow* technique [2] to determine the number of clusters, given the importance of choice for the number of clusters during clustering. The *elbow* technique (see Figure 30) revealed that we should use three or five clusters. We choose the latter since we have many invalid domains to cluster, and we thought that three clusters would cause invalid records with different characteristics to be grouped into the same cluster.

Date	**Valid Domains**	**Invalid Domains (Syntactic Errors)**	**Percentage Of Invalid Domains**	**Domains With SPF**
01-05-2015	25,808,275	588,961	2.23%	26,397,236
01-05-2016	29,322,574	597,011	2.00%	29,919,585
01-05-2017	28,244,908	591,710	2.05%	28,836,618
01-05-2018	35,335,692	594,525	1.65%	35,930,217
01-05-2019	37,123,496	585,677	1.55%	37,709,173
01-05-2020	41,545,212	599,014	1.42%	42,144,226

Table 10: Number of invalid SPF records due to syntactic errors in the *main* dataset

Date	**Valid Domains**	**Invalid Domains (Syntactic Errors)**	**Percentage Of Invalid Domains**	**Domains With SPF**
01-05-2016	452,094	25,944	5.43%	478,038
01-05-2017	489,494	28,125	5.43%	517,619
01-05-2018	728,661	35,345	4.63%	764,006
01-05-2019	805,947	40,155	4.75%	846,102
01-05-2020	719,232	31,237	4.16%	750,469

Table 11: Number of invalid SPF records due to syntactic errors in the *Alexa* dataset

For each cluster, we took three SPF records such that we can determine the characteristic of the respective cluster. Table 12 displays the three SPF records of each cluster. Multiple factors characterise cluster 1. The first characteristic is that some SPF records contain multiple strings in the record. Multiple strings in an SPF record are characterised by using two quotation marks, as shown by the first row in Table 12. This is measuring artefact from our validation checker

since we do not handle the cases of multiple strings in an SPF record. This is due to splitting the SPF record using a Tokeniser which splits the SPF record on spaces (see Section 5.5). These SPF records should be declared valid instead of invalid. The second characteristic of cluster 1 is the use of no spaces between mechanisms. The whole SPF record is written by one long string as displayed by the second row in Table 12. The third and last characteristic of cluster 1 is spaces between a mechanism and the `:` character. The third row of Table 12 displays an example.

A measuring artefact characterises the second cluster. The cluster contains only one SPF record, which is displayed by the fourth row of Table 12. The SPF record starts with `v=spf1 include:outlook.com -all900` and this sequence repeat many times.

The third cluster is characterised by a specific start and end sequence of the SPF record. The SPF records in this cluster start with `\"` and end with `\"`. Rows five, six, and seven of Table 12 display examples of SPF records in this cluster. A point to notice is that all these domains seem to have some sort of Turkish background. Examples of domains in this cluster are: `the-istanbulhotels.com`, `turkeytourbooking.com`, and `kitapyeri.com`. Analysing these domains into more detail, we found out that all of these domains with a Turkish background have the same nameserver provider: `istanbulnet.net`. This nameserver provider probably displays the SPF records wrongly by using `\"` as the starting sequence and `\"` as the ending sequence.

The fourth cluster has multiple characteristics. The first characteristic is using the `ipv4` mechanism. The `ipv4` mechanism does not exist, and the `ip4` mechanism should be used instead. An example of an SPF record which has this characteristic is displayed by the eighth row in Table 12. The second characteristic is the same as the one in cluster 1: there is a space between a mechanism and the `:` character. Examples of this characteristic are displayed by the ninth and tenth row in Table 12.

The fifth and last cluster has multiple characteristics, and almost all of the characteristics have been seen before. The first characteristic is the use of the `ipv4` mechanism (which does not exist). This is the same characteristic as cluster four. An example of this characteristic is displayed by the eleventh row in Table 12. The second characteristic of this cluster is almost the same as using the `ipv4` mechanism. Instead of using an `ip4` mechanism, these SPF records use a `ip` mechanism which does not exist. The twelfth row of Table 12 shows an example. The last characteristic of this cluster is almost the same as in cluster one and four: there is a space after the `:` character. An example is displayed by the last row in Table 12.

To summarise the characteristics of the clusters: we saw two types of errors in the characteristics of the clusters. The first type of error is manual errors. An example is that administrators add (accidentally/unintentionally) a space after a `:` character or after a mechanism or they use a mechanism that does not exist. The second type of error is a measurement artefact. Cluster three has the characteristic of having SPF records that start with `\"` and end with `\"`. Removing these two sequences from the SPF records result in valid SPF records.

Index	Cluster	SPF Record
1	1	v=spf1””include:spf.protection.outlook.com””-all
2	1	v=spf1include:spf.163.com-all
3	1	v=spf1 ip4:49.212.235.233 a: www3493.sakura.ne.jp -all
4	2	v=spf1 include:outlook.com -all900INTXTv=spf1 include:outlook.com -all900INTXTv=spf1 include:outlook.com -all900INTXTv=spf1 include:outlook.com -all900INTXTv=spf1 include:outlook.com -all900INTXTv=spf1 include:outlook.com -all900INTXTv=spf1 include:outloo””...
5	3	\”v=spf1 a mx ptr ip4:46.165.245.1/24 ip4:46.165.221.72/24 ~all\”
6	3	\”v=spf1 a mx ip4:88.198.40.89 ip4:144.76.33.166 ip4:5.9.83.73 -all\”
7	3	\”v=spf1 mx a a:mailoptions.com include:_spf.google.com~all\”
8	4	v=spf1 a mx ip4:185.42.173.109 ipv4:185.42.172.14 ~all
9	4	v=spf1 mx a ptr ip4:34.231.90.178 ip4:52.9.243.90 ip4:34.232.221.103 ip4:52.202.245.35 ip4:34.195.6.7 ip4: 34.234.169.66 ip4:34.224.141.104 ~all
10	4	v=spf1 mx a a:mail.webnow.com include: mail.webnow.com -all
11	5	v=spf1 ipv4:54.84.232.83/32 ~all
12	5	v=spf1 ip4:64.183.116.114 ip4:12.222.87.138 ip:162.211.50.221 ip:50.63.7.129 include:spf.protection.outlook.com include:secureserver.net ~all
13	5	v=spf1 ip4: 118.27.33.59 ~all

Table 12: Example SPF records of each characteristic of each cluster

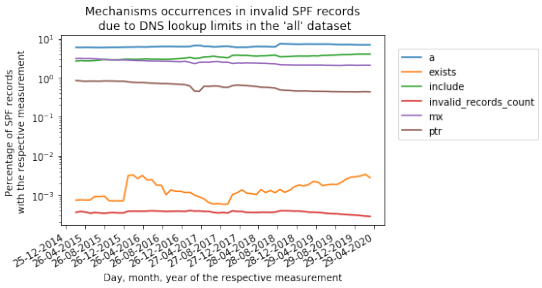

Figure 31: Relative usages of mechanisms of invalid SPF records due to DNS lookup limits in *main* dataset

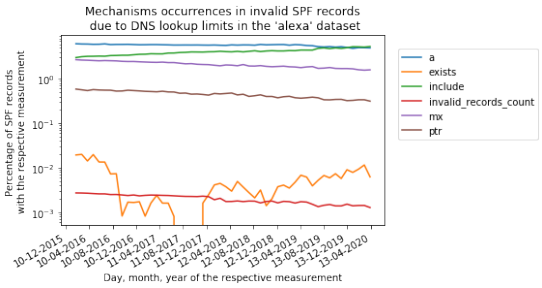

Figure 32: Relative usages of mechanisms of invalid SPF records due to DNS lookup limits in the *Alexa* dataset

6.8.2 DNS Lookup Limits

Another way how a `permerror` could be returned is when there are more than ten DNS lookups in the SPF record. The reason why there is a maximum of ten DNS lookups in an SPF record is to prevent load on DNS servers and to prevent potential DDoS attacks. The mechanisms that inflict a DNS lookup are: the `include`, `a`, `mx`, `ptr` and `exists` mechanisms. What this means is that when there are more than ten of the previously mentioned mechanisms, a `permerror` will be returned and the SPF record will be declared invalid.

We first analysed how often the DNS lookup limit exceeded by only looking at the original SPF record. The number of DNS requests an included SPF record provokes are not included in this first examination. The reason why we first only looked at the original SPF record is that DNS operators are easily able to scan these SPF records and perform a validity check. Later in this section, we will also take a look at the number of invalid records when we take into account the mechanisms that invoke DNS requests in the included SPF records.

We analysed how often a certain mechanism occurs in an invalid SPF record due to DNS lookup limits in the original SPF record. Table 13 shows the result of the analysis. The column **Invalid SPF records** shows that there is a slight increase in the total number of invalid SPF records due to exceeding the DNS lookup limit starting from 2015 up to 2020. However, as shown by column **Percentage Of Invalid Domains**, the relative number of invalid SPF records decrease over time. The most occurred mechanism in an invalid SPF record due to DNS lookup limits is, by a large margin, the `a` mechanism. Second is the `include` mechanism, third is the `mx` mechanism, fourth is the `ptr` mechanism and the least frequently used mechanism is the `exists` mechanism. Figures 31 and 32 show the relative usages of the mechanisms in their respective dataset over time.

Date	a	exists	include	mx	ptr	Invalid SPF records	Percentage Of Invalid Domains	Domains With SPF
01-05-2015	56,634	7	25,367	29,028	7,632	9,549	0.036%	26,397,236
01-05-2016	62,369	11	30,081	27,934	7,841	10,308	0.034%	29,919,585
01-05-2017	67,544	8	30,863	24,636	4,335	10,066	0.035%	28,836,618
01-05-2018	72,453	9	39,965	27,476	6,966	11,565	0.032%	35,930,217
01-05-2019	89,855	20	42,913	25,982	5,558	12,328	0.033%	37,709,173
01-05-2020	77,183	27	42,595	23,119	4,843	11,025	0.026%	42,144,226

Table 13: Number of mechanisms that are in invalid SPF records due to DNS lookup limits when only looking at the original SPF record in the *main* dataset

However, the results showed in Table 13 and Figures 28, 29, 31, and 32 were analyzed only taken into account the original SPF record. The reason why we included these results is that when an original SPF record contains more than ten DNS lookups, the DNS operator is able to verify the SPF record without any additional DNS requests and therefore able to inform the administrator of the domain if the record is invalid. Next, we will evaluate the SPF records when also the included mechanisms by using `include` mechanisms are taken into account.

Date	Valid Domains	Invalid Domains	Percentage Of Invalid Domains	Domains With Include Mechanism(s)	Domains With SPF
01-05-2015	26,181,519	215,717	0.82%	11,818,177	26,397,236
01-05-2016	28,286,902	1,632,683	5.46%	14,972,152	29,919,585
01-05-2017	27,130,465	1,706,153	5.92%	13,771,458	28,836,618
01-05-2018	34,254,932	1,675,285	4.66%	21,035,654	35,930,217
01-05-2019	34,033,130	3,676,043	9.75%	23,787,555	37,709,173
01-05-2020	38,570,294	3,573,932	8.48%	26,634,395	42,144,226

Table 14: Number of SPF records that are declared invalid when also examining the DNS requests of the `include` mechanism(s) in the *main* dataset

The difference in invalid domains when only counting the original mechanisms of SPF records or when also counting the mechanisms of the included SPF records is vast. On the first of May 2020, when only counting the original mechanisms of an SPF records the percentage of invalid domains is: 0.026%. Whereas when also counting the mechanisms of the included SPF records has a percentage of invalid domains of 8.48%. The numbers mentioned previously give a lower-limit of domains that have an invalid SPF record since our dataset does not contain all SPF records (see Section 5.6.1). With this improved way of counting the mechanisms that provoke DNS requests, the number of invalid domains has been increased with a factor of more than 325 in the *main* dataset on the first of May 2020. One of the main takeaways from this vast increase of invalid domains is that DNS operators probably have poor insight on the DNS requests of the `include` mechanism and DNS lookup limits in general.

Since the (in)validation graphs in Figures 28 and 29 were generated using only investigating the original SPF record, therefore these Figures are not showing the complete (in)validation percentages. Therefore we generated new graphs which are visible in Figures 33 and 34. The datasets used in these graphs are respectively, the *main* dataset and the *Alexa* dataset.

Figure 33 shows three interesting movements. The first is around February 2016, the second is around June 2018, and the last one is around January 2020. Around February 2016, a sudden rise in the number of invalid SPF records is visible. The first interesting movement is shown by the first two rows of Table 15. The first row is the month before the peak, and the second row is the month after the peak. What is interesting is that the number of invalid domains is almost tripled compared to the month before, whereas the number of domains has only been increased by almost 700 thousand.

The second interesting movement is around June 2018. Around June 2018, the number of invalid SPF records have seen a sudden increase again. The sudden increase is shown in the third and fourth row in Table 15. Noticing is that the number of invalid domains has been almost doubled during that period.

The last interesting movement is a sudden drop around January 2020, which is visible in the last three rows in Table 15. The month before the drop, the percentage of invalid domains was 10.86%, while the month of the drop, the percentage of invalid domains was 4.93%. The month after the drop, the percentage of invalid domains raised again to 10.70%.

The second vast increase around June 2018 might be related to a change in the DNS of the nameserver of Hostgator. A little over 2 million domain names which were change from valid to invalid between the first of May 2018 and the first of June 2018 are domains which use the nameserver of Hostgator. A change in the default SPF record of domains which use the nameserver of Hostgator might have caused this vast increase of invalid domains. For the other two scenarios, we are not sure why this happened. There is no nameserver which is significantly large compared to the changes in invalid domains as with the case of June 2018.

In this section, we showed that many domains are invalid due to either syntactic errors or DNS lookup limit errors. DNS operators could be more active in scanning SPF records and warning the administrators of domains which have an invalid SPF record. As a result of this, the validation rate of SPF will increase, and performing e-mail sender address forgery would be more challenging.

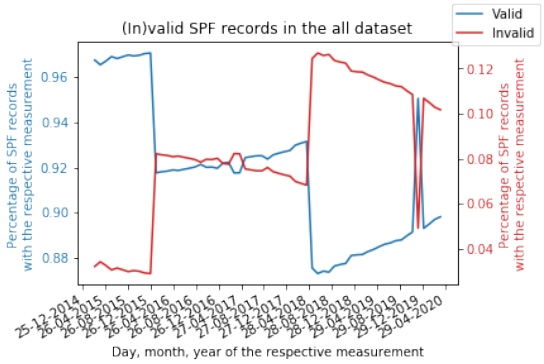

Figure 33: (In)valid percentage of SPF records using the correct way on how to count the include mechanisms in the *main* dataset

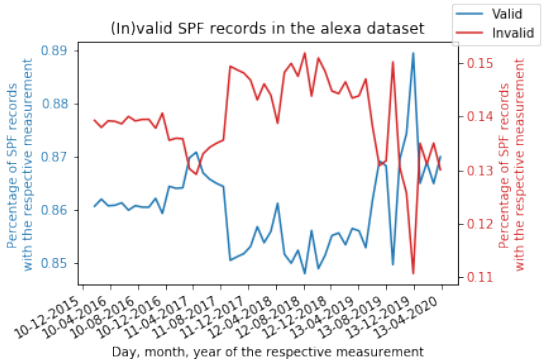

Figure 34: (In)valid percentage of SPF records using the correct way on how to count the include mechanisms in the *Alexa* dataset

Date	**Valid Domains**	**Invalid Domains**	**Percentage Of Invalid Domains**	**Domains**
01-01-2016	28,297,961	849,711	3.00%	29,147,672
01-02-2016	27,363,724	2,452,907	8.96%	29,816,631
01-05-2018	34,186,117	2,508,106	7.34%	36,694,223
01-06-2018	32,496,096	4,620,618	14.22%	37,116,714
01-12-2019	36,114,091	4,396,225	12.17%	40,510,316
01-01-2020	38,498,788	1,995,798	5.18%	40,494,586
01-02-2020	36,695,037	4,391,687	11.97%	41,086,724

Table 15: Total number of SPF records that are in invalid SPF due to DNS lookup limits or due to syntactic errors in the combined *main* dataset

6.9 Include Linkage

In the preceding section, we already mentioned the effect of the `include` mechanism on DNS lookups. In this section, we further investigate the depth of include linkage, which we define as when an SPF record (original) includes an SPF record (second) and the second SPF record also includes another SPF record (third). Thereby the first/original domain is also including the SPF record of the third domain. This linkage of inclusion could be repeated over and over again and thereby including many SPF records. This linkage could be done with the knowledge of the administrator of the first domain, but it is more problematic when this happens without the knowledge of the administrator of the first domain. The chain of included SPF records that the `include` mechanisms provoke, cause an increase in DNS lookups outside the view of the original domain and might cause security risks.

We analysed the *main* dataset to determine how often this linkage of SPF records happens and at what kind of depth. On the first of May 2020, well over 27 million domains had an SPF record with at least one `include` mechanism. Of those 27 million domains, about 5.6 million domains also have an include depth of at least two. An include depth of two means that the SPF records of the original domains include another SPF record and the included SPF record also includes another SPF record. Including an SPF record at the second depth might have happened without the knowledge of the administrator of the original domain, thereby causing a potential security risk and an increase in DNS lookups.

The second point to notice is that there was one domain with a linkage depth of six, and there were 41 domains that had a linkage depth of five on the first of May 2020. The domain which had an include depth of six was `clarkems.org`. Figure 35 displays which domains `clarkems.org` includes at each depth and what kind of relations exists between the included domains. Multiple points can go wrong when including domains: cycles, DNS lookup limits, and authorising third parties. The next sections explain for three domains what kind or errors exist when examining the include chains.

6.9.1 clarkems.org (authorizing third parties, cycle, and DNS lookup limit)

Figure 35 displays which domains `clarkems.org` includes at each depth and what kind of relations exists between the included domains. A point we noted in Section 5.6.1 is that our dataset is incomplete since it does not contain all domains and contains no SPF (i.e. TXT) measurement data for subdomains. Therefore, the generated figures contain the minimum number of domains that are included by the original domain (in this case, `clarkems.org`). The incomplete dataset might, therefore, establish blindspots in our analysis. A blindspot is established when an included domain is not available in our dataset. Therefore, we are unable to automatically determine which domains the included domain includes. This means that our analysis displays the minimum number of included domains, we might miss cycles that are caused due to these so-called blindspots, and we may not be able to count all DNS lookups required to fully resolve a policy.

The original domain (`clarkems.org`) includes two domains, which represent the first include depth in Figure 35. The two included domains are: `websitewelcome.com` and `hilltophosting.com`. The domain `websitewelcome.com` includes four domains: `spf.websitewelcome.com`, `spf1.websitewelcome.com`, `spfgwp.websitewelcome.com`, and `_spf.google.com`. In our dataset, we do not have SPF measurement data for these domain names, and as such, we cannot automatically determine the domains they include. However, we manually analysed the domains, and we saw that the domains `spf.websitewelcome.com` and `_spf.google.com` include other domains. This means that these two domains are so-called blindspots. Therefore, we are unable to determine if they cause cycles and how many DNS requests they provoke. The second included domain in the first include depth (`hilltophosting.com`) is a special one because it includes the already included domain `websitewelcome.com`, and `hilltophosting.com` includes itself again and therefore creating a cycle (which will be explained in more detail later in this section). The domain `hilltophosting.com` also includes `gmail.com` and `summitems.org`. In our dataset, both domains do not include other domains. However, manually analysing the two domains displayed that `summitems.org` includes other domains. Monitoring the included domains continues until the sixth depth and at the sixth depth the domains `spf-0024a201.pphosted.com`,

`spf.verizonwireless.com`, and `spf.intl.verizon.com` are included. What this means is that all the IP addresses that match one of the SPF policies between the original domain and the sixth depth are allowed to send e-mail from the domain `clarkems.org`. Allowing all these IP addresses is most often not the intention of the administrator of the original domain. Therefore, which parties to include using the `include` mechanism should be carefully thought about and only used with trusted parties.

The second problem visible by analysing the `include` chain of `clarkems.org` is that the DNS lookup limits are exceeded. The original domain is limited to ten DNS lookups in total (see Section 2.5.1). Each `include` mechanism is at least causing one DNS query, therefore if there are more than ten included domains, the DNS lookup limit will be exceeded. The domain `clarkems.org` includes a total of twenty domains, which means that at least twenty DNS requests are executed. This results in an invalid SPF record since it queries more than ten DNS requests. As we just explained, each `include` mechanism is causing at least one DNS requests. However, the DNS requests the included SPF record performs counts to the global limit of ten DNS requests. Therefore, many domains are already declared invalid after including a few domains.

Previously, we already mentioned that `hilltophosting.com` (see first include depth in Figure 35) is establishing a cycle by including itself. Including itself, or creating in another way a cycle causes the DNS lookup limit to be reached eventually since the cycle will keep using the `include` mechanism and each `include` mechanism causes at least one DNS request. Eventually, ten DNS request will be reached, and the original SPF record (`clarkems.org`) will be declared invalid. In the case of `clarkems.org`, even if we assume that `websitewelcome.com` does not include any other domains and `hilltophosting.com` includes only itself (so we only have an include depth of one), the SPF record of `clarkems.org` will be declared invalid since the cycle causes the SPF record to exceed the ten DNS lookups limit eventually. The cycle will run infinitely, and at least one DNS lookup will be executed each run. The ten DNS lookups limit will be reached eventually, and the SPF record will be declared invalid. Therefore, when using `include` mechanisms one should verify if there are no cycles in the `include` chain.

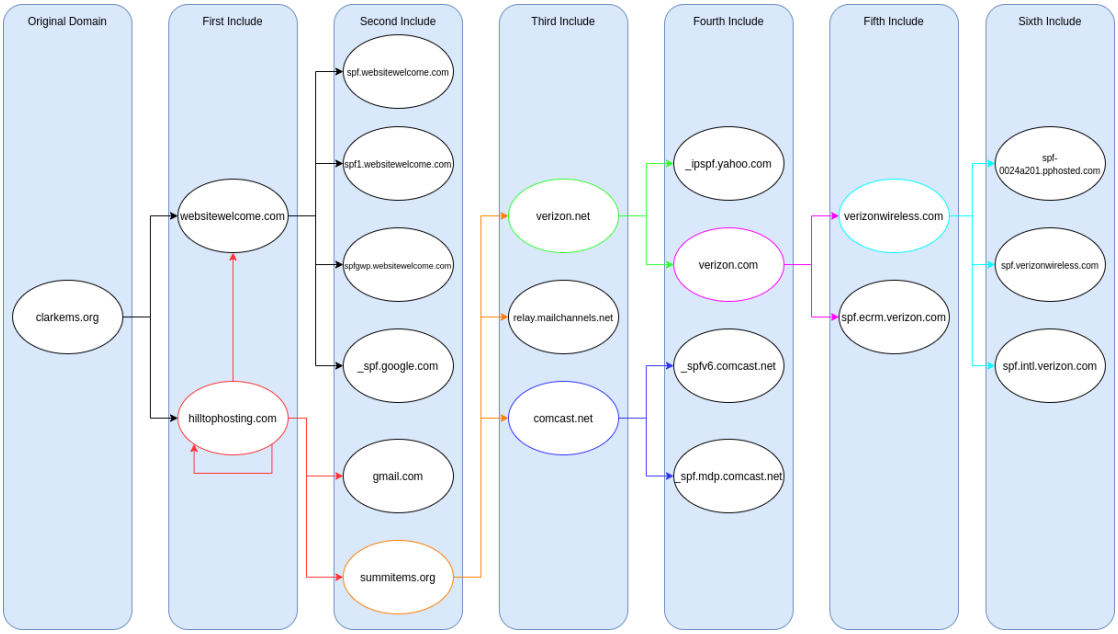

Figure 35: Include analysis of clarkems.org

6.9.2 workersunitednynj.org (authorizing third parties and DNS lookup limit)

As a second example, we discuss `workersunitednynj.org`, which we selected because an included domain includes an already included domain while not creating a cycle. The domain `workersunitednynj.org` has an `include` chain of depth five (see Figure 36) instead of six as the previously analysed domain `clarkems.org`. The domain `workersunitednynj.org` includes two domains: `seiu.org` and `seidev.org`. The domain `seiu.org` includes four domains and `seiudev.org` includes two domains: the already included `seiu.org` and `seiuaws.org`. The four domains that `seiu.org` includes do not include other domains in our dataset. In the second include depth, the domain `seiuaws.org` includes `amazonaws.com`, `amazonses.com`, and the already included domain `seiu.org` (see first include depth). In this scenario the link between `seiuaws.org` and `seiu.org` does not create a cycle, since `seiu.org` does not include any of the domains that lead to previously included domains `seiudev.org` or `seiuaws.org`. However, the four domains which are included by `seiu.org` will be evaluated twice due to the include from `seuaws.org` (second include depth) to `seiu.org` (first include depth). The third, fourth, and fifth include depths contain domains which are related to Amazon, such as `amazonaws.com`, `amazon.com`, and `spf1.amazon.com`. The `workersunitednynj.org` domain displays an example of a domain that not only allows one cloud e-mail provider, but three. The three cloud e-mail providers are: Mailchimp, Google, and Amazon. However, since there are twelve included domains, the SPF record of `workersunitednynj.org` will be declared invalid since there are at least twelve DNS requests by invoking the `include` mechanisms.

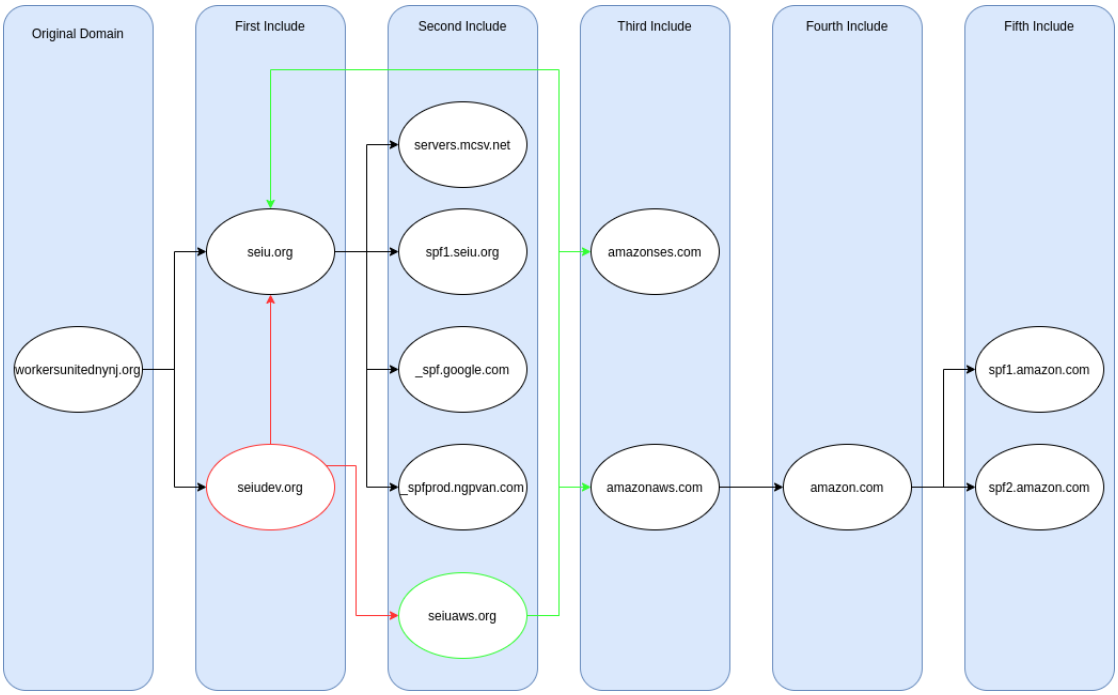

Figure 36: Include analysis of workersunitednynj.org

6.9.3 uhrig.org (authorizing third parties and DNS lookup limit)

The domain `uhrig.org` (see Figure 37) follows roughly the same trend as `workersunitednynj.org`: the original domains include two domains, the second include depth consists of five domains, the third, fourth, and fifth include depths mostly contain domains which are related to Amazon. In total `uhrig.org` includes fourteen domains. Therefore, `uhrig.org` has at least fourteen DNS request and that results in an invalid SPF record. The previous two examples displayed that some domains include an already included domain at one of the previous depths or same depth. Including an already included domain is not a big issue since it will only cause additional DNS requests. However, as we saw in `clarkems.org`, a cycle causes the original SPF record to be declared invalid due to exceeding the DNS lookup limit. The domain `uhrig.org` has a domain in the second include depth (`digitalfirstmedia.com`) which includes `_spf.google.com` which was already included by `readingeagle.com` in the first include depth. This is not a big issue since no cycle is created, only extra DNS requests are invoked by including `_spf.google.com` again. However, if these extra DNS requests induce the global DNS requests to exceed ten DNS requests, the SPF record should be declared invalid.

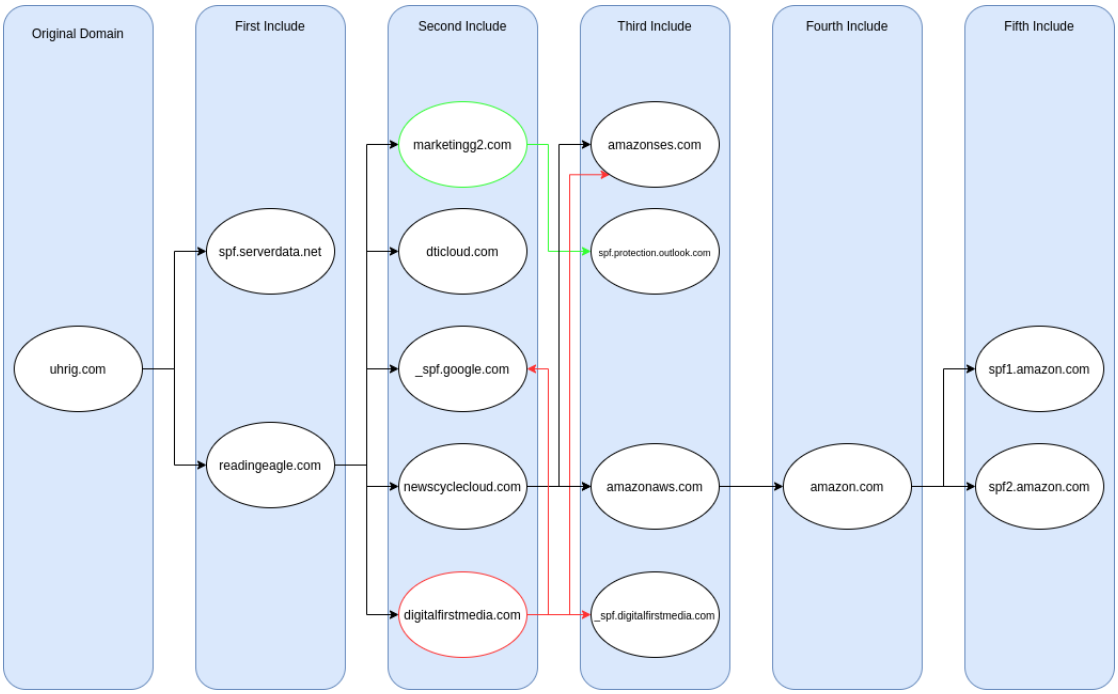

Figure 37: Include analysis of uhrig.com

This section summarises the linkage depth analysis. As shown in this section, many domains use the `include` mechanism in a linked way and thereby including SPF records of other domains. Including these domains can be exercised without the knowledge of the administrator of the domain and therefore causing more DNS requests and cause a potential security risk. Another point we noticed are cycles in SPF records. Cycles cause the SPF record to be declared invalid (due to exceeding DNS lookups limit) and cause potential security risks. The key takeaway of this section is that administrators should carefully consider which SPF records to include.

7 Conclusions

As we endeavoured to advance our understanding of SPF usage on the Internet, we defined the following main research question: *How are SPF records configured in practice?* We divided this main question into three subquestions:

- How do administrators typically configure SPF? (RQ1)
- Can we identify a change in the use of SPF over time? If yes: How do SPF records change over time? (RQ2)
- Can we recognise problematic trends of SPF use? If yes: What are the characteristics of these problematic trends? (RQ3)

Our analysis (see Section 6) addressed each of these questions. In the following sections, we will revisit and draw our conclusions for each research question. Finally, we draw our overall conclusions.

7.1 How do administrators typically configure SPF? (RQ1)

To determine how administrators typically configure SPF records, we first looked at the adoption of SPF. The percentage of domains among the *main* dataset that have an SPF policy is around 25%. This number is very low compared to the percentage of domains among the *Alexa* dataset that have an SPF policy (74%). We think that this vast difference can be explained since domains providing services to large user bases, the domains in the *Alexa* dataset, are more likely to be operated by entities that make security considerations than the majority of domain names.

We first looked at the adoption of SPF, broadly speaking. We then looked at the typical use of SPF. To determine how administrators typically configure SPF records, we have taken a look at the (relative) frequencies of qualifiers, mechanisms, modifiers, and the combination of qualifiers and mechanisms in Section 6.2 and 6.6. We described for each of the available mechanisms the relative use among domain name SPF policies in the respective dataset. We saw that the `a` mechanism is the most frequently used mechanism (1.44 per SPF record on average), second is the `all` mechanism (0.97), third is the `include` mechanism (0.73), fourth is the `ip4` mechanism (0.48), and fifth is the `mx` mechanism (0.42). The other mechanisms are used on average in less than 5% of the SPF records. From these values, we can conclude that administrators typically use a combination of the five most popular mechanisms. By which the `a` mechanism is used on average more than once in an SPF record. Another conclusion we can draw from these results is that there are SPF records without an `all` mechanisms or a `redirect` modifier. Summing the number of `all` mechanisms and `redirect` modifier results in a value of 0.986 which is less than 1 (see Table 4). When there is no mechanism that matches the sender's IP address and no `redirect` modifier, then the result of the SPF record will be a `NEUTRAL` results as if there was a `?all` mechanism. This creates a potential security risk and should be avoided.

Besides the mechanisms and modifiers are the qualifiers. The qualifiers determine how to interpret the result when a mechanism matches the sender's IP address. The most used qualifier is the `+` qualifier (0.60 per SPF record on average), closely followed by the $\sim$ qualifier (0.55). The third most used qualifier is the `-` qualifier (0.30), and the least frequently used qualifier is the `?` qualifier (0.12). These results are calculated irregardless of the mechanisms with which they are combined (e.g. `+all` and `+ip` both increase the `+` qualifier count). However, most qualifiers are used in combination with the `+` qualifier, the exception is the `all` mechanism. The `all` mechanism is most often used in combination with the `-`, $\sim$, and `?` qualifier (see Section 6.6.8).

Therefore, we further investigated the usage of the qualifiers without the occurrences of the qualifier which are attached to the `all` mechanism. The number of occurrences of the `-`, $\sim$, and `?` qualifier are cut in half when removing the qualifiers attached to the `all` mechanisms. From these results, we can conclude that most SPF policies are set up to allow certain IP addresses, otherwise drop or tag the incoming e-mail.

Next to the mechanisms and qualifiers is a third category: modifiers. The two modifiers have vast differences in (relative) usage. The `redirect` modifier was used 694,899 times in the *main* dataset on the first of May 2020, while the `exp` modifier was used only 2,158 times. The low usage of the `exp` modifier can be related to the fact that the `exp` modifier is often used for

debugging the SPF record when adjusting/implementing the SPF record and afterwards the `exp` modifier will be removed. In general, we can conclude that modifiers are not that often used, and when they are used, it is most often the `redirect` modifier to redirect the SPF record to another SPF record.

Another characteristic of how administrators typically configure SPF records is that the domains often make use of a SaaS (Software as a Service) product. The SaaS product requires the administrator to include the SPF record of the SaaS provider to let the SaaS work properly. In May 2020, we saw that around 11% of the SPF records include Microsoft's (Office 365) SPF record in the *main* dataset as shown in Table 6. The top ten most included domains can be placed in two categories of companies: *Software as a Service (SaaS) providers* and *hosting providers*. From these results, we can conclude that the majority of the `include` mechanisms are used to include the SPF policy of a company in one of the two categories.

All together, we can conclude that many variations of SPF records are possible. However, there are some similarities between SPF records: most of the SPF records use a ~`all` mechanism at the end, which means that administrators typically do not like to drop e-mails. Another similarity is the high relative usage of the `a` mechanism, which means that SPF records often allow the `A` and `AAAA` resource records to be part of the SPF policy. The last similarity is the use of the `include` mechanism. Many SPF policies make use of the `include` mechanism, therefore including another domain's SPF policy and thereby creating a link between the two SPF records.

Up to now, we mentioned some characteristics of how administrators typically configure SPF. However, we did not directly answer the research question. We described in Section 6.6.9 that many domains allow specific IP addresses in their SPF policy and otherwise drop or tag the e-mail. This observation is the answer to our first research question.

7.2 Can we identify a change in the use of SPF over time? (RQ2)

We started this Thesis by explaining that the United States Department of Homeland Security (DHS) released the Binding Operational Directive (BOD) 18-1 related to enhancing e-mail and web security on the seventeenth of October 2017. Section 6.5.2 explained that due to the release of BOD 18-1, the adoption rate of SPF among `.gov` domains increased from 51.06% to 74.88% in around three and a half months. From this vast increase, we can conclude that releasing a BOD causes that most of the government instances follow the requirements written in the respective BOD. Therefore, we can conclude that government requirements can be a motivation to adopt SPF (or any other mechanism).

To identify if there was a change in the use of SPF over time, we looked at the adoption rate of SPF. The *main* and *Alexa* datasets showed an increase of adoption of SPF over time (see Figures 1 and 2). The *main* dataset went from an adoption rate of around 19% on the first of March 2015 to an adoption rate of around 25% on the first of May 2020. The *Alexa* dataset went from an adoption rate of around 54% on the first of February 2016 to an adoption rate of around 74% on the first of May 2020. From these results, we can conclude that the adoption of SPF among the domains in the *Alexa* dataset has seen a larger increase in percentage points than the domains in the *main* dataset. However, in general, we see a positive trend with the adoption of SPF: the adoption of SPF is rising.

The relative usage of the qualifiers and mechanisms also have changed over time. What notices is that three (`a`, `ip4` & `mx`) of the five most frequently used mechanisms have had a decrease in relative usage over time. As shown in Table 4, the relative usage of the `a` mechanism has been decreased by 9%, of the `ip4` mechanism 14%, and of the `mx` mechanism 9% in a five year time period. However, the `include` mechanism has shown a substantial increase in relative usage over time. The `include` mechanism has seen an increase of 25% in five years. This vast increase in relative usage of the `include` mechanism can be related to the work of van Rijswijk-Deij et al. [35]. Van Rijswijk-Deij et al. showed that the use of cloud e-mail providers had been vastly increased over time. The increase of cloud e-mail providers is also our conclusion from the analysis done in Section 6.3. Another positive trend among the mechanisms is that usage of the `+all` mechanism decreased over time. This decrease is a positive trend since the `+all` mechanism allows all IP addresses to match and returns a `PASS` result which might be a potential security risk.

Related to cloud e-mail providers is the change in requirement on how to include Google's SPF record for their G Suite (SaaS) service to be enabled and function properly. Section 6.6.14 described how Google had multiple pages on what SPF record to include to enable and function the G Suite service properly. One of the pages required to include `aspmx.googlemail.com`, while the other required to include `_spf.google.com`. The latter SPF record has replaced the former SPF record. However, this transition took quite some time and is still not completely adopted. The transition started at least in August 2010 and probably sooner. Therefore, we can conclude that administrators do not change SPF records often unless it is required.

The relative usage of the qualifiers have also changed over time. The `+` qualifier (+17%), the `-` qualifier (+4%) and the $\sim$ qualifier (+12%) have seen in increase of relative usage in the *main* dataset over the five year time period. The only qualifier which has had a decrease of relative usage over time is the `?` qualifier. The `?` qualifier has seen a decrease of 15% over time. Therefore, we can conclude that more administrators are more certain about their SPF policies since using a `?` mechanism results in that the receiving e-mail server has to handle this e-mail as if there was no SPF policy.

Another change over time is the change in the use of no qualifier attached to a mechanism. The use of a mechanism without a qualifier has decreased by around 18% over the five years. The decrease in the use of a mechanism without a qualifier means that more administrators are explicitly using the `+` mechanism more than before. The reason for explicitly using the `+` qualifier can be to clarify the SPF record. Using no qualifier with a mechanism might confuse and explicitly using the `+` qualifier solves the confusion.

Overall, we can conclude that SPF has seen quite a few (positive) changes over time: more domains are adopting SPF, government requirements are a stimulation to adopt SPF, the use of the `+all` mechanism has decreased over time, and many more positive changes have happened over time.

7.3 Can we recognise problematic trends in SPF use? (RQ3)

Our hypothesis (see Section 4.3) of the third research question was that SPF policies are set up manually by the domain name administrators. Manually defining an SPF policy means that human error exists, which might cause invalid SPF records. One of the problematic trend we found in the results section (see Section 6.9) is the linkage of `include` mechanisms. The *main* dataset on the first of May 2020, consisted of well over 27 million domains with at least one `include` mechanism in their SPF records. Of those 27 million domains, also about 5.6 million domains had an include depth of two or more.

This include linkage concludes that over 20% of the domains with an `include` mechanism in their SPF record have an include depth of at least two. We believe that as a result of this, the `include` mechanism is a dangerous mechanism to use and should only be used when including a trusted party. Otherwise, the party might include malicious domains and therefore endangering the original domain's e-mail security.

The second problematic trend which we found is the use of the combination of the ∼ qualifier and the `all` mechanism. The combination of this qualifier and mechanism causes that whenever an incoming e-mail does not match one of the mechanisms in the SPF record, the e-mail should be accepted and tagged but not discarded. Figure 17 displays that the ∼`all` mechanism has seen an increase from 43% to 55% in a period of five years. From our point of view we think that using this combination of qualifier and mechanism can be related to the following reasons:

- Unawareness
- Laziness
- Move responsibility

These three reasons are somehow linked to each other. The first reason is unawareness: an administrator might be unaware that using the ∼`all` mechanism is causing the behaviour of not dropping the e-mail but tagging it. This might sound like a kind of a spam filter in their ears if they are not experienced with the function of SPF. The opposite is true, instead of giving the receiver an easy protocol to verify if the sender was authorised to send the e-mail, the administrator is switching the responsibility to the receiver of the e-mail. The receiver will check if the sender of the e-mail matches any of the mechanisms. If that is not the case, the ∼`all` mechanism will be matched at the end. Matching the ∼`all` mechanism means that the receiver of the e-mail is now responsible for how to handle the incoming e-mail. Should the receiver accept the e-mail or drop it? The receiver is not sure if the sender of the e-mail was authorised to send the e-mail, because a ∼`all` mechanism was used. In these cases, SPF did not help the receiver of the e-mail, but made the receiver of the e-mail with a difficult question to answer: how to handle the e-mail?

During the explanation of the unawareness reason of why to use a ∼`all` mechanism, we already discussed the part of moving the responsibility from the domain by which the e-mail might have been sent from to the receiver of the e-mail. When a `-all` mechanism would have been used, the responsibility would still be at the domain by which the e-mail might have been sent from and not at the receiver of the e-mail. This means that when a `-all` mechanism was used, the receiver of the e-mail knows that when this mechanism has been matched, the e-mail should be dropped.

The last reason to discuss is the laziness reason. What is meant by laziness is that when a company has many places where e-mail might be sent from and by which increases the risks of forgetting to include a mechanism to match on it. In these cases, it might be more comfortable for the company/administrators to set up a ∼`all` mechanism at the end of the SPF record instead of a `-all` mechanism. When a `-all` mechanism would have been used, valid e-mail could have been dropped, and by which valuable information could have been lost in transit. When they change the mechanism to ∼`all`, e-mails are not dropped directly but tagged, and then the receiver of the e-mail is responsible for how to handle the e-mail. In this case, the responsibility on how to handle the e-mail will be transferred from the domain (which might have sent the e-mail) to the receiver of the incoming e-mail.

Another problematic trend of SPF records we recognised is the high number of invalid SPF records. On the first of May 2020, almost 10% of the SPF records in the *main* dataset are

declared as invalid. We examined two possible causes: syntactic errors and DNS lookup limits. The latter causes for most of the invalid SPF records (8.42% vs 1.42%). The used dataset is not complete (missing subdomains), and therefore the percentage is potentially even higher when a complete dataset would have been used. However, DNS operators are able to scan the original SPF record and verify the SPF records. When a DNS operator finds an invalid SPF record, it can warn the administrator of the domain.

The next problematic trend of SPF records is the linkage of the `include` mechanisms. Section 6.9 explained how domains are linked with each other due to the use of `include` mechanisms. In most cases, this is fine and safe. However, when an administrator is unaware of the fact that the included domain also includes another domain. The IP addresses that the latter domain allows are also authorised to send e-mail from the original domain, which might be unaware to the administrator of that domain and causing a potential security risk. Not only the authorisation of sending e-mail is the problem caused by linked SPF records. Linked SPF records might also cause cycles, and therefore the SPF record will be declared invalid due to too many DNS lookups. Therefore, we can conclude that using `include` mechanisms should be done carefully and only with trusted third-parties to reduce security risks.

Overall, we analysed two problematic trends in SPF use: syntactic errors and DNS lookup limit errors. Both problematic trends cause SPF records to be declared invalid. The syntactic errors are easily repairable by verifying the SPF record using an (online) analyser tool. The DNS lookup limit errors are more difficult to repair since the administrators might be required to include specific mechanisms which cause DNS lookups. We also encountered some limitations in our analyses. The first limitation is that the code that validates the SPF records on syntactic errors is not performing a complete check. The code only verifies the first part of the SPF record (i.e. before the `:` sign). Therefore, some SPF records are declared valid while they should have been declared invalid. The second limitation is that the used dataset is incomplete since it does not contain all domains and (almost) no subdomains. This incomplete dataset causes blindspots in our include chain analysis (see Section 6.9).

7.4 General Conclusion

The previous three sections described the conclusions for all three research questions. Overall, we can conclude that administrators typically set up an SPF policy once, and the administrators do not often change the SPF policy afterwards. The SPF policies are mostly set up using a combination of the five most-used mechanisms (i.e. `a`, `all`, `include`, `ip4` & `mx`). Most SPF policies are set up by allowing certain IP addresses and otherwise drop or tag the incoming e-mail policy.

SPF records have seen a change in configurations over time. The most noticeable is the vast increase of the `include` mechanism. Many Software as a Service (SaaS) providers require the clients to include the SPF policy of the SaaS to let the SaaS work properly. A positive trend is the decrease in usage of the `+all` mechanism. The `+all` mechanism is allowing all IP addresses and might create a security risk. The last change we discuss is the decrease of the mechanisms without a qualifier. This decrease might be because using a qualifier is establishing less confusion about how to interpret the mechanism.

We saw some problematic trends in SPF use. Overall, around 10% of the SPF policies were declared invalid by our analysis. Most often this was because the DNS lookup limit was exceeded. The DNS lookup limit is often exceeded because domains use `include` mechanisms which cause at least one DNS lookup and probably in most cases more. Another reason why SPF policies were declared invalid by our analysis is due to syntactic errors. Syntactic errors can vary from a simple typo to a qualifier that has no mechanism attached to it.

Working with *big data* caused some troubles. One of them was that performing actions on the data should be carefully thought about since *expensive* actions can take a long time to execute. Example of an *expensive* action, is the use of a *join* action. A *join* action combines two datasets given a certain condition. Another action which was an *expensive* action is the use of regular expressions. We used regular expressions to filter some data, but we noticed that it is most often much quicker to write our own function, which replicates the result of the regular expression.

Working with Spark to handle operations on our *big data* also caused some issues. We once created a piece of code that analysed the DNS lookups of the SPF records. Running this piece of code on the *small Alexa* dataset was successful. However, running this piece of code on the *main* dataset was unsuccessful since after around five minutes, a time-out error was returned. To fix this issue, we added prints into the code, and these prints cause the time-out error delay to be extended, by which the piece of code was also running correctly on the *main* dataset.

Overall, we can conclude that SPF has seen a growth in adoption over time. This growth is a positive matter since e-mail address sender forgery will be more challenging to execute. Therefore, receiving e-mails would be safer. However, as we showed in our Thesis, almost 10% of all SPF records were declared invalid by our analysis. Invalid SPF records mean that the incoming e-mail should be treated as if there was no SPF record and a false sense of security could be established.

8 Future Work

We believe that further research on SPF records can be done in multiple ways. The first way is to get rid of the limitations of our dataset when performing the include depth analysis as described in Section 5.6.1. To perform a more accurate linkage depth analysis, we need to have access to all the domains which are linked to each other. In most cases, this means that we need DNS data of the subdomains of domains. Currently, OpenINTEL is not gathering data of subdomains, except the `www` subdomain, and therefore we are not able to perform a complete analysis. Creating a complete dataset is doable since subdomains are registered in the DNS zone of a domain using a `A`, `ALIAS`, or `CNAME` record and our dataset contains these records.

In section 6.8, we mentioned that almost 10% of the SPF records are declared invalid due to syntactic errors or exceeded DNS lookup limits. We think that most of the administrators of domains with an invalid record are unaware that their SPF record has been declared invalid, which can cause security risks. To remove the unawareness aspect of the invalid SPF records, one could notify the administrators by e-mail (how ironic) for example. A way to implement this is to retrieve all the DNS data of the domains and verify the SPF records daily. Whenever an SPF record has been declared invalid, an e-mail should be sent to the abuse contact of the domain.

We previously mentioned that almost 10% of the SPF records are declared invalid due to syntactic errors or exceeded DNS lookup limits. However, the piece of code that validates is performing an incomplete check (see Section 5.5). Only the first part of a mechanism and modifier is verified. The part after the `:` sign is not verified. What this means is that SPF records that contain a mechanism with an invalid domain or IP address behind the `:` sign are in our analysis declared as valid. However, these SPF records should have been declared invalid. To improve our work, one could extend our code to verify the whole mechanism or modifier. The extended code will result in more SPF records that are declared invalid due to syntactic errors.

A way to display the results of this Thesis is to build a website, whereby the include linkage of domains is visible. One could enter a URL of a domain, and a figure would be automatically generated like Figure 35. Another way to display interactive information to the users is to build a graph. A graph could be used to show linkages between the domains in a graph structure. Figure 38 shows a graph with two *main* center domains (i.e. `_spf.protection.netiyi.com` and `alpha-mail.com`). Both *main* domains are included by many other domains as the links in the graph show. This example can be extended by all the domains in our dataset, to create a large interactive graph by which users can search for a domain and see the links.

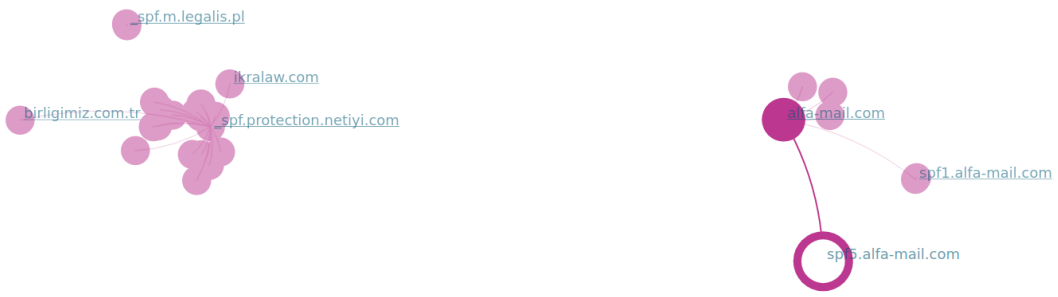

Figure 38: Example of an (interactive) graph

On the first of May, we saw that the adoption rate of SPF was around 25% for the *main* dataset. One way to improve this number is to notify hosting providers to adjust their nameservers such that they contain SPF records. When a domain uses the nameservers of a hosting provider, the DNS values will be set according to the nameservers. When the nameservers contain an SPF record, the domains which use the nameservers also have an SPF record. By which the number of domains that adopt SPF will increase.

During this Thesis, we mostly looked at the following datasets: *main* and *Alexa* datasets. However, the OpenINTEL project contains many more datasets. One type of dataset is the

datasets which contain country-code TLDs (ccTLDs). The OpenINTEL project consists of 16 ccTLDs, which are currently not (fully) analysed. One could mutually compare the results of TLDs to determine which ccTLDs are adapting SPF more than others, for example. The same as we did by comparing the adoption rates between the `.se` dataset and the `.ee` dataset.

In Section 2.5.1, we mentioned that we tested a few domains which had a cycle in their SPF policy using two online SPF analysers. Both SPF analysers declared the SPF records invalid due to the cycles. However, we do not know how *real* e-mail servers handle such kind of cycles. The e-mail server might detect the cycle, stop cycling, and continue with the remaining part to examine. On the other hand, the e-mail server might continue cycling until ten DNS lookups are reached, and then the SPF record is declared invalid. We have not analysed cycles in-depth, and therefore one could analyse this further by analysing how many domains have a cycle and how to prevent cycles of happening.

In Section 6.6.10, we mentioned that there are modifiers that have a qualifier attached to them. It is not allowed to have a qualifier attached to a modifier, and therefore these SPF records should be declared invalid. However, in our invalidation analysis, we did not include this kind of check. To improve this Thesis, one could extend our invalidation analysis by adding a check for if a qualifier is attached to a modifier. If that is the case, the SPF record should be declared invalid.

9 Acknowledgments

13th of March 2020: sitting in a room of eight by four meters with four other smart humans as usual.
16th of March 2020: official COVID-19 lockdown at the university. No students are allowed at the University of Twente. Humans need to keep one and a half meter distance.
19th of August 2020: students are still not allowed to go to the university.

Three regular dates are written above, and every year they would not have been special. However, it is 2020, COVID-19 has taken over the world, and students are not allowed to go to the university. Only the first month of my Thesis I was allowed to go to university. The other four months I have been working from home.

During the so-called lockdown, I have had multiple video and audio calls with my supervisor Mattijs Jonker. In the meantime, we have also sent more than 100 e-mails to each other. Therefore, I would thank Mattijs Jonker for guiding me through this challenging and exciting process of writing my Thesis.

Next, I would like to thank my grandparents, parents, siblings, uncles, aunts, cousins, and friends for always being supportive.

Since I am a Christian, I would like to include my favourite Syriac-Orthodox hymn in my Thesis (see Figure 39).

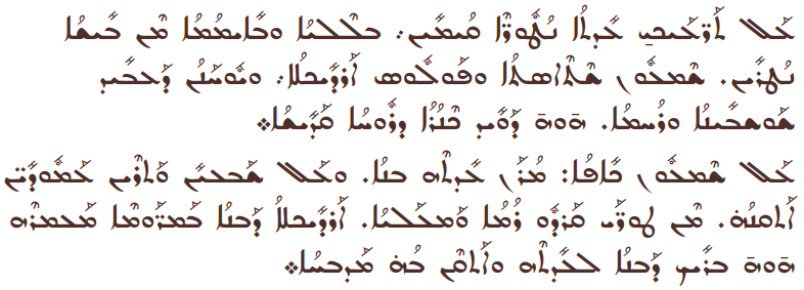

Figure 39: Syriac-Orthodox hymn: 'Al tar'ayk 'ito (`https://www.kolesuryoye.nl/altaraykito.html`)

References

[1] 99Firms.com. How many email users are there? https://99firms.com/blog/how-many-email-users-are-there/#gref, May 2019. Accessed: 2020-02-27.

[2] T. Alade. Tutorial: How to determine the optimal number of clusters for k-means clustering. https://blog.cambridgespark.com/how-to-determine-the-optimal-number-of-clusters-for-k-means-clustering-14f27070048f, May 2018. Accessed: 2020-06-19.

[3] N. T. Anh, T. Q. Anh, and N. X. Thang. Spam filter based on dynamic sender policy framework. In *2010 Second International Conference on Knowledge and Systems Engineering*, pages 224–228, Oct 2010.

[4] Apache Spark. Apache spark - unified analytics engine for big data. https://spark.apache.org/. Accessed: 2019-12-16.

[5] Apache Spark. Clustering - spark 3.0.0 documentation. http://spark.apache.org/docs/latest/ml-clustering.html. Accessed: 2020-07-14.

[6] L. Bernheim. 20+ largest web hosting companies in 2020: World & us markets. https://www.hostingadvice.com/how-to/largest-web-hosting-companies/, November 2019. Accessed: 2020-05-18.

[7] Betaalvereniging Nederland. 3,81 miljoen euro schade door phishing bij internetbankieren in 2018. https://www.betaalvereniging.nl/actueel/nieuws/381-miljoen-euro-schade-door-phishing-bij-internetbankieren-in-2018/, March 2019. Accessed: 2019-12-13.

[8] Cisco. How can service providers face ipv4 address exhaustion? https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/enterprise-ipv6-solution/whitepaper_c11-698132.html, June 2012. Accessed: 2020-06-23.

[9] Department of Homeland Security. Binding operational directive 18-01. https://cyber.dhs.gov/bod/18-01/, October 2017. Accessed: 2019-12-13.

[10] Devon. Correct spf record using google apps. https://stackoverflow.com/questions/1082048/correct-spf-record-using-google-apps, August 2010. Accessed: 2020-05-14, see response from Devon.

[11] DMARCAnalyzer. Spf record check - dmarc analyzer. https://www.globalsign.com/en/blog/what-is-s-mime. Accessed: 2020-07-20.

[12] DocuSign. Docusign - #1 in electronic signature and agreement cloud. https://www.docusign.com/. Accessed: 2020-07-20.

[13] K. Fazzini. Email wire fraud is so simple for criminals to pull off, it's cost companies $26 billion since 2016, says fbi. https://www.cnbc.com/2019/09/11/email-wire-fraud-cost-26-billion-since-2016-says-fbi.html, September 2019. Accessed: 2019-12-18.

[14] GlobalSign. What is s/mime and how does it work? https://www.globalsign.com/en/blog/what-is-s-mime. Accessed: 2020-07-20.

[15] J. Gold. Estonia as an international cybersecurity leader. https://e-estonia.com/estonia-as-an-international-cybersecurity-leader/, August 2019. Accessed: 2020-04-23.

[16] S. Görling. An overview of the sender policy framework (spf) as an anti-phishing mechanism. *Internet Research*, 17:169–179, 04 2007.

[17] Helpnet Security. Phishing attacks at highest level in three years. https://www.helpnetsecurity.com/2019/11/07/phishing-attacks-levels-rise/, November 2019. Accessed: 2020-07-14.

[18] L. Howard. Ipv6 growth. https://www.retevia.net/ipv6-growth/, February 2019. Accessed: 2020-06-23.

[19] S. Kitterman. Sender policy framework (spf) for authorizing use of domains in email, version 1. https://tools.ietf.org/html/rfc7208, April 2014. Accessed: 2019-12-16.

[20] S. Kitterman. Sender policy framework (spf) for authorizing use of domains in email, version 1. https://tools.ietf.org/html/rfc7208#section-5.5, April 2014. Accessed: 2020-03-16.

[21] S. Kitterman. Sender policy framework (spf) for authorizing use of domains in email, version 1. https://tools.ietf.org/html/rfc7208#section-4.6.4, April 2014. Accessed: 2020-02-27.

[22] MilanK. Setting up spf record. https://support.google.com/a/forum/AAAA034zvV8Vns11blwMrM/?hl=nl, March 2009. Accessed: 2020-05-14.

[23] S. Moore and E. Keen. Gartner forecasts worldwide information security spending to exceed $124 billion in 2019. https://www.gartner.com/en/newsroom/press-releases/2018-08-15-gartner-forecasts-worldwide-information-security-spending-to-exceed-124-billion-i August 2018. Accessed: 2020-01-08.

[24] T. Mori, K. Sato, Y. Takahashi, and K. Ishibashi. How is e-mail sender authentication used and misused? In *Proceedings of the 8th Annual Collaboration, Electronic Messaging, Anti-Abuse and Spam Conference*, CEAS '11, page 31–37, New York, NY, USA, 2011. Association for Computing Machinery.

[25] G. C. M. Moura, M. Müller, M. Davids, M. Wullink, and C. Hesselman. Domain names abuse and tlds: From monetization towards mitigation. In *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pages 1077–1082, May 2017.

[26] Mxtoolbox. Spf - spf recursive loop. https://mxtoolbox.com/problem/spf/spf-recursive-loop?page=prob_spf&action=spf:clarkems.org&showlogin=1&hidepitch=0&hidetoc=1. Accessed: 2020-07-20.

[27] OpenINTEL. Openintel: Active dns measurement project. https://openintel.nl. Accessed: 2019-12-16.

[28] J. B. Postel. Simple mail transfer protocol. https://tools.ietf.org/html/rfc821, August 1982. Accessed: 2020-07-14.

[29] RTL Nieuws. Journalisten versturen nepmails uit naam van mark rutte naar kamerleden. https://www.rtlnieuws.nl/nieuws/nederland/artikel/3706711/journalisten-versturen-nepmails-uit-naam-van-mark-rutte-naar, October 2017. Accessed: 2019-12-18.

[30] D. Sipahi, G. Dalkılıç, and M. H. Özcanhan. Detecting spam through their sender policy framework records. https://onlinelibrary.wiley.com/doi/pdf/10.1002/sec.1280, May 2015. Accessed: 2019-12-18.

[31] R. Sobers. 110 must-know cybersecurity statistics for 2020. https://www.varonis.com/blog/cybersecurity-statistics/, November 2019. Accessed: 2020-01-08.

[32] Tweede Kamer. Update over valse e-mailberichten uit naam kamerleden. https://www.tweedekamer.nl/nieuws/kamernieuws/update-over-valse-e-mailberichten-uit-naam-kamerleden, October 2017. Accessed: 2019-12-13.

[33] O. van der Toorn, R. van Rijswijk-Deij, B. Geesink, and A. Sperotto. Melting the snow: Using active dns measurements to detect snowshoe spam domains. In *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, pages 1–9, April 2018.

[34] R. van Rijswijk-Deij, M. Jonker, A. Sperotto, and A. Pras. A high-performance, scalable infrastructure for large-scale active dns measurements. `https://rijswijk.github.io/files/jsac2016-34-7-vanrijswijk-preprint.pdf`, March 2016. Accessed: 2019-12-16.

[35] R. van Rijswijk-Deij, M. Jonker, A. Sperotto1, and A. Pras. The internet of names: A dns big dataset. `https://conferences.sigcomm.org/sigcomm/2015/pdf/papers/p91.pdf`, August 2015. Accessed: 2020-06-16.

[36] R. van Rijswijk-Deij, M. Jonker, A. Sperotto1, and A. Pras. A high-performance, scalable infrastructure for large-scale active dns measurements. `https://rijswijk.github.io/files/jsac2016-34-7-vanrijswijk-preprint.pdf`, April 2016. Accessed: 2020-07-14.

[37] Verizon. 2019 data breach investigations report. `https://enterprise.verizon.com/resources/reports/2019-data-breach-investigations-report.pdf`, 2019. Accessed: 2020-01-08.

[38] M. Wong and W. Schlitt. Sender policy framework (spf) for authorizing use of domains in email, version 1. `https://tools.ietf.org/html/rfc4408`, April 2006. Accessed: 2019-12-16.