The Unintended Consequences of Email Spam Prevention

Sarah Scheffler[1], Sean Smith[1,3,*], Yossi Gilad[1,2,*], and Sharon Goldberg[1]

[1] Boston University
[2] Massachusetts Institute of Technology
[3] Amazon Technologies, Inc.

Abstract. To combat Domain Name System (DNS) cache poisoning attacks and exploitation of the DNS as amplifier in denial of service (DoS) attacks, many recursive DNS resolvers are configured as "closed" and refuse to answer queries made by hosts outside of their organization. In this work, we present a technique to induce DNS queries within an organization, using the organization's email service and the Sender Policy Framework (SPF) spam-checking mechanism. We use our technique to study closed resolvers. Our study reveals that most closed DNS resolvers have deployed common DNS poisoning defense techniques such as source port and transaction ID randomization. However, we also find that SPF is often deployed in a way that allows an external attacker to cause the organization's resolver to issue numerous DNS queries to a victim IP address by sending a single email to any address within the organization's domain, thereby providing a potential DoS vector.

1 Introduction

The Domain Name System (DNS) is one of the most fundamental Internet services. Most clients are serviced by a recursive resolver, which queries authoritative name servers until finding the IP address mapped to a domain name. The ubiquitous deployment of DNS servers, their critical nature, combined with rather limited security mechanisms embedded into the DNS protocol caused DNS to be exploited in many malicious activities on the Internet over the years, from denial of service (DoS) attacks [1, 21, 26] to cache poisoning [14]. A common best practice for recursive DNS resolvers to protect themselves from being exploited in such attacks is to be "closed," meaning that the resolver will not respond to requests for queries made by IP addresses located outside their organizations. A closed resolver forces the attacker to operate from inside the organization's network, e.g., by compromising an internal machine, and therefore provides a useful mitigation against attacks that do not target a specific organization. Since closed resolvers are widely deployed and provide a fundamental service, studying their

[*] Work conducted while at Boston University

operation is important to understanding how networks operate. Yet, closed resolvers make it difficult for researchers to measure the DNS landscape, because recursive resolvers would no longer answer remote queries. Thus performing an Internet-wide scan of DNS services becomes a challenge.

In this study, we use a method of querying closed recursive DNS resolvers by using email, taking advantage of the Sender Policy Framework [15, 25], a common anti-spam defense for email. By sending an email to a mail server within the organization, we trigger an SPF check for the sender's address, triggering an intra-organization query for a domain controlled by our own authoritative nameserver, and thus bypass the "closed" defense of the resolver. This email should be caught by the spam filter aided by the receiver's SPF system, and thus is typically not noticeable by mail server administrators.

We conducted a partial Internet scan, covering 15% of the IPv4 address space, searching for mail servers, and then sent emails to each of the mail servers we found. We then studied the induced DNS queries that our email triggered using a nameserver under our control. Our results show that many mail servers use an unsafe SPF configuration that will cause more than the maximum-recommended 10 DNS queries. We ran a test on the scanned mail servers that would induce a maximum of 10 DNS queries in the recommended SPF configuration, and would induce up to 42 DNS queries in a configuration vulnerable to abuse. We received on average 34.3 induced queries, indicating that many mail servers in the wild use this potentially-vulnerable configuration. We also used the DNS queries we received to measure the deployment of various anti-cache poisoning mechanisms across closed DNS resolvers.

We provide required background on SPF in §2. We discuss measurement methodology in §3 and analyze our results in §4. §5 discusses related work, and we conclude with recommendations in §6.

2 SMTP and Sender Policy Framework (SPF)

Emails are sent and received by Mail Transfer Agents (MTAs). In "vanilla" SMTP, any MTA is allowed to send email from any sending address. Much like there is no mechanism to stop somebody from writing a fraudulent return address on an envelope, there is no mechanism in SMTP to stop somebody from sending from an email address with a domain they are not part of. As an anti-spam defense, the Sender Policy Framework (SPF) was introduced in 2006 [25] as means of verifying email-sender identity. It was later updated in 2014 [15].

In SPF, a `TXT` record is set in the sender's domain to specify which IP addresses are approved to serve as the domain's MTAs (i.e., send emails on behalf of senders in that domain). When an email is sent, the receiving MTA retrieves this record using a DNS query to determine whether the sending MTA's IP address was valid.

SPF allows for more complicated validation procedure than just querying for whitelisted MTA addresses. The `TXT` record contains a list of terms, which the recipient's MTA uses to check for matches with the sender's IP address. If the

IP address matches a term, then the qualifier on that term determines whether the email should be delivered or rejected. It is common practice to end an SPF record with a "`-all`" term, which rejects any email address that did not match any other term. Some SPF terms can cause additional DNS queries, such as for the IP address matching a given `A` or `MX` record. One important feature of SPF we use in future sections is support for `include` terms, which allow an SPF record to tell an MTA to recursively evaluate another SPF record and use the result in the evaluation of the "top-level" record.

Limiting SPF's overhead. In order to avoid unreasonable load on the DNS, the SPF standard [15] requires limiting the number of DNS query-causing terms to 10, and if any more are found, an error must be returned. However, as we see in the next section, the `include` term allows a malicious email sender to circumvent this limit.

3 Measurement Methodology

In our experiment, we registered the domain name `emaildns.net`. We control its authoritative nameserver. The `emaildns.net` domain contains several SPF TXT records for subdomains that correspond to three different SPF configurations, which we explain in §3.3. We send emails from these subdomains to MTAs that we find by using `zmap` [5]. These emails trigger an SPF check that causes the MTA's DNS resolver to query our authoritative nameserver. We can observe the queries made to our nameserver, and therefore study the behavior of both the DNS resolvers and the MTAs they are querying for, even if those resolvers are closed.

3.1 Ethical Considerations

This study used a remote port scan, which relied on information publicly available by trying to communicate with MTAs legitimately. We sent at most three emails to each MTA we found, which would cause a maximum of 60 DNS queries per email (5 for `goodspf`, 10 for `badspf`, and 42 for `treespf` configuration, plus 1 query for the original SPF record for each configuration). We believe that this should not cause considerable load on any MTA or recursive resolver.

We did our best to ensure that as few humans as possible would receive email, by attempting to find email addresses that would not be delivered to a human. We also wrote the SPF record in the `emaildns.net` zone to deliberately fail after checking all our recursive statements, so that any correctly-configured SPF system would ultimately reject our email.

Our measurement emails referred readers to our project website `https://emaildns.net` which has information about our study, and contains a form that allows adding email addresses to a blacklist, if anyone wanted to opt-out of the remaining part of the study. We attempted to contact 190597 MTAs found by our `zmap` scan, and in total we sent 38720 emails successfully. During this process, 23 email addresses were added to our blacklist via our web form. We also received

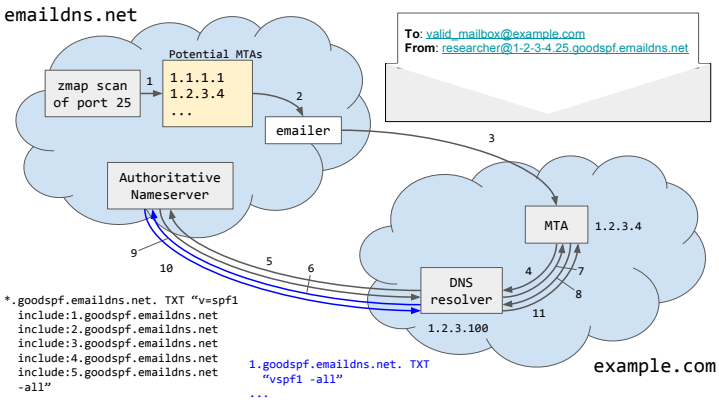

Fig. 1. Measurement technique. The example emails an MTA at IP address `1.2.3.4` using the "goodspf" configuration. The steps are detailed in the text.

three emailed requests to opt-out of the study and one emailed request to send the owner of the MTA the results of our study when we completed it.

We also contacted the authors of a paper that used a similar IPv4 scan [10] and used their blacklist of CIDR ranges, under the logic that anyone that opted out of that study probably would not want to be part of this one either. We noted on our website that we would add CIDR ranges to the blacklist if we received an email request to do so. We received no such requests during the scan.

3.2 Operation

Our measurement has several steps, which we summarize in Figure 1 and describe in more detail in section 3.4. First, we test whether an IP address is a potential MTA, by using `zmap` [5] to check if that IP address hosts a server listening on port 25, the default SMTP port. Second, because SPF is only used when an email is sent to a valid address, for each MTA IP address we attempt to find a valid email address served by that MTA, following a procedure discussed in section 3.4. Third, we send emails to addresses served by the MTA to invoke the SPF check. The sending address of this email is within `emaildns.net`, so our authoritative nameserver for `emaildns.net` will be queried during the SPF check of this email. To allow measuring different MTAs/DNS resolvers in parallel, we encode information about the recipient MTA in the subdomain of the sender's address in our email. The sending address is in the form `<dashed-ip4-mta-address>.<portnum>.spf-config.emaildns.net`. For example, if sending to an MTA at `1.2.3.4` using the "treespf" config, the domain of the sending email address, which will be checked via the receiver's SPF, is `1-2-3-4.25.treespf.emaildns.net`.

Steps 4-11 in Figure 1 illustrate what happens after an email was sent and SPF checks begin. In step 4 the recipient's MTA issues a DNS query for the sending domain name's SPF record. In step 5 the DNS resolver at the recipient

MTA side (possibly a closed resolver) sends this `TXT` query for the sender's domain name to the authoritative nameserver at `emaildns.net`, where it is logged. In steps 6-7 the response, which contains a number of `include` statements depending on the SPF configuration, is sent to the recipient's DNS resolver and then its MTA. Steps 8-11 are *induced* DNS queries for the names specified by the `include` statements in the original SPF record. Steps 8-11 repeat a number of times dependant on the SPF configuration, as discussed in §3.3.

3.3 SPF Configurations

We use three sets of SPF records, which we call goodspf, badspf, and treespf. As discussed in §2, SPF records are restricted to containing no more than 10 DNS query-causing terms. Our three SPF implementations are designed to study how this restriction works in practice. goodspf contains 5 DNS query-causing terms and is a valid SPF record. badspf contains 20 DNS query-causing terms, and is therefore invalid. treespf contains 6 DNS query-causing terms, each of which causes an *additional* 6 DNS queries, for a total of 42 induced queries, but only 6 DNS query-causing terms. We describe each of these configurations in more detail below. All three SPF configurations ultimately evaluate to `fail` so that our email will not be delivered, but only after the entire SPF check finishes.

goodspf. This record, shown in the lower left of Figure 1, contains five `include` statements, that redirect to `i.goodspf.emaildns.net` for i from 1 through 5. The SPF for record for all `i.goodspf.emaildns.net` is `"v=spf1 -all"`. An `include` does not result in an immediate `fail` for the main query if the included SPF check fails [15][§5.2]. These `include` terms can be thought of as SPF "dead ends." Each included check fails, but the main SPF check continues, checking all `include` terms before making its final decision.

The goodspf configuration is meant to establish a baseline for SPF behavior, ensuring that the record containing fewer than 10 DNS query-causing terms is processed as we expect it to be: one DNS query for the main SPF record, and one induced query for each of the five included SPF records.

badspf. This record has 20 include statements that each cause a single DNS query (to `i.badspf.emaildns.net` for i from 1 through 20). Each of these "sub-records" is `" v=spf1 -all"`. It is therefore non-compliant with the SPF specification, which restricts the number of DNS query-causing terms to 10. We use the badspf record to check whether this limit of 10 queries is enforced at all — if it is, we would expect to get 10 induced queries per badspf email sent, and if it is not, then we would expect to get 20.

treespf. This record, shown graphically in Figure 2, allows us to gain insights on how the 10 DNS-query-causing statements is enforced. While the standard limits the number of *query-causing terms* in an SPF record to 10, it does not limit the *total number* of SPF DNS queries made, and therefore treespf is RFC-compliant. The treespf configuration is compliant the standard's limitation of 10 query-causing terms, but causes many more than 10 queries, by using recursively-called nested includes. This record, shown graphically in Figure 2, allows us to gain

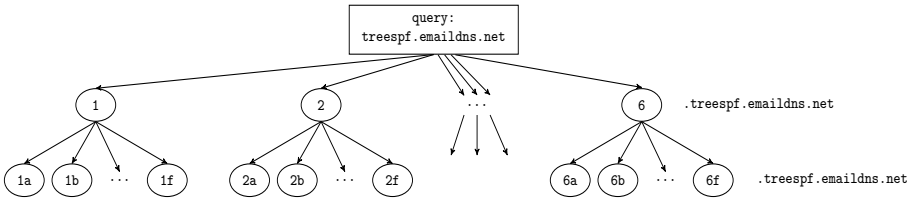

Fig. 2. In treespf, one main query directly induces 6 subqueries, each of which induces 6 more subqueries, for a total of 42 induced subqueries.

insight on how the limit of 10 DNS query-causing statements is enforced. While the standard limits the number of *query-causing terms* in an SPF record to 10, it does not limit the *total number* of DNS queries made. The treespf configuration is RFC-compliant, as it contains fewer than 10 DNS query-causing terms. However, each of these terms causes more than one DNS query, by using an "include" statement to cause a recursive evaluation of an entirely new SPF query. The top level `*.treespf.emaildns.net` record contains includes to six other SPF records: `i.treespf.emaildns.net` for i from 1 through 6. Then each of those records contains includes to six additional records: `1a.treespf.emaildns.net` through `1f.treespf.emaildns.net`, for example. A total of 42 DNS queries are made within the SPF record, however, each individual record contains exactly six (fewer than 10) query-causing statements.

3.4 Experimental Procedure

We next describe how we performed our measurement.

Scanning for MTAs. We used `zmap` [5] to perform a TCP SYN scan to find services listening on port 25 over IPv4. For each shard of 2^{24} IP addresses, we first scanned the entire shard, collecting all IP addresses that responded to our TCP SYN on port 25 and saving them to a file. This process took roughly five minutes per shard. Note that we did not complete our entire IPv4 scan. Our results are an initial finding that indicate how recursive SPF check works. (We discuss our results and their limitations in §4.)

Finding valid recipient email addresses. This step uses a heuristic to get email addresses that are likely serviced by the MTA. We used usernames such as "noreply" or "postmaster," and we learned potential domains through a `whois` lookup and using a reverse DNS lookup. We also removed subdomains from each of these domain guesses as additional possible options. We attempt to begin delivery to each combination of these usernames and domains until we get one that the MTA recognizes.

Sending emails and logging induced DNS queries. We encode the information about the recipient MTA in our sender email address. For instance, if sending to an MTA at IP address `1.2.3.4` and using the goodspf configuration, we would send our email from `researcher@1-2-3-4.25.goodspf.emaildns.net`. (The "25" in the address represents the port number, and was included to allow

expanding our study to more ports.) We then send an email to addresses we guessed in that domain. If we do not receive a `250 OK` SMTP status response, then we try the next email address that our heuristic provides. Our email sender handles in parallel addresses for 300 MTAs.

Once we found a working email address for the MTA, we sent two additional emails, so that in total an MTA receives three emails, one for each SPF configuration. Throughout this whole process, we log all DNS queries received on `emaildns.net`'s authoritative name, and we we retained all of our SMTP logs.

4 Analysis of Induced DNS Queries

We broke our scan into 256 shards, based on the 8 most significant bits of the IP address. The scan ran from September 11th, 2017 through September 22, 2017 and covered IP addresses from 0.0.0.0 through 34.255.255.255. Although our results are partial, we believe that there are valuable insights to be gained from the portion of the scanned address space.

4.1 SPF deployment

We categorize the queries that our authoritative name server receives into three bins: (1) "main queries," which are `TXT` queries for a domain that we sent an email from, for example, `1-2-3-4.25.badspf.emaildns.net`; (2) "induced queries," which are queries induced by `include` statements within our main SPF record, such as `16.badspf.emaildns.net`; and (3) other, miscellaneous queries.

SPF as DoS vector. Checking an SPF record with many nested includes could cause undue load on the SPF-validating MTA, causing it to make far more DNS queries per email than it should. This could result in degraded performance or complete denial of service. Since SPF places a limit on the number of *query-causing terms* in an SPF record, rather than the total number of queries made, an SPF record with many `include` statements can recursively cause queries to any `include` statements in *those* records, and so on. `treespf` is a proof of concept of this: it induces 42 queries even though it contains only 10 query-causing terms.

A more dangerous version of this would involve mutually recursive SPF records, shown in Figure 3. Attackers could "bounce between" recursive calls to look up each SPF record, and each call would cause 9 additional `include` queries to an unrelated victim nameserver. However, this was not tested as part of this work.

SPF Configuration Results. Our results are summarized in Table 1.

As our baseline measurement to ensure the SPF check does what we expect when it sees a well-formed, typical SPF record, our `goodspf` record should induce 5 queries. We received 4.87 induced queries per main `goodspf` query (39583/8125), which is about what we expected.

To ensure that the SPF check correctly halts at 10 DNS-query causing terms in a single record, our `badspf` record attempted to induce 20 queries, only 10 of

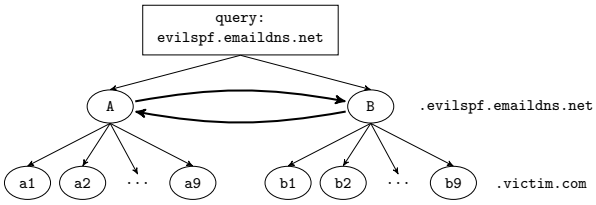

Fig. 3. A mutually recursive chain of SPF `include` statements could cause an infinite number of queries to `victim.com`.

SPF configuration	Query response ratio	Expected response ratio
goodspf	4.87 (39583/8125)	5.0
badspf	7.79 (136881/17562)	10.0
treespf	34.3 (280734/8192)	either 42.0 or 10.0

Table 1. Expected and actual ratios for the three different SPF configurations used.

which should actually occur. Our badspf record on average induced 7.79 queries per main query (136881/17562). We received approximately the same number of goodspf and treespf queries, but double that number of badspf queries. We believe that the error caused by querying badspf causes some MTA software to query the record again after a few minutes, indicating that an SPF `permerror` may be incorrectly treated as a `temperror` by some implementations.

Our main result is the ratio of induced treespf queries. As described in section 3.3, the treespf configuration contains only 6 DNS query-causing terms (below the RFC-mandated limit of 10), but each of those terms is an include statement that causes 6 additional DNS queries. If SPF performed a once-per-email-address check that halted DNS queries after 10 total queries, then we would expect to receive the same number of induced queries as in badspf However, if the check is performed separately on all recursively-checked include statements rather than once per email address, we would expect to get 42 induced queries per main query. This conforms to the SPF RFC, but goes against the RFC's stated goal of avoiding unreasonable load on the DNS. We received 34.3 induced queries per main treespf query (280734/8192), clearly indicating that many SPF configurations perform the check per-include rather than once per email address.

It was very common for the different induced queries caused by the same main query to come from different resolvers. Because of this, it was infeasible to tie induced queries (e.g., `3b.treespf.emaildns.net`) with the exact main query that caused it (e.g. `1-2-3-4.25.treespf.emaildns.net`). Attempts at approximating the relationship between induced queries and main queries based on the timestamp and querying DNS server were very imprecise. We limit our analysis here to an aggregate: the ratio of all induced queries to all main queries.

These results show that although the SPF's standard intends to "avoid unreasonable load on the DNS" by limiting the number of DNS-querying terms

in an SPF record, a malicious SPF record can use nested include statements to circumvent this limitation using recursion.

4.2 Security of DNS Resolvers

Having received DNS queries from domain names within organizations, we can study the deployment of suggested DNS security mechanisms on DNS resolvers.

Closed vs open. A fundamental defense against DNS-based attacks is to deploy recursive DNS resolvers as closed. And even among open DNS infrastructures, DNS queries are often passed from an open to a closed resolver before recursion occurs [23]. Our measurement technique gives us the IP address of the recursive resolver that queries the authoritative nameserver, regardless of whether it is open or closed, which is the one that matters for determining how SPF checks are handled. To check whether the resolver is closed, we send it a DNS query from our machine using `dig`. If it responds to our query, then we assume it is open. If it does not, we assume it is closed.

Port Randomization and TXID Randomization. It is widely accepted that DNS resolvers should randomize the source port and transaction ID in their queries, and then validate these fields as echoed in the DNS responses in order to defend against DNS poisoning [11].

To check whether or not port randomization and TXID randomization were enabled, we looked at the queries we received from each DNS server in chronological order. We used two thresholds to test whether the query transaction IDs were feasibly close to nonrandom order: first, we checked whether 70% of query TXID numbers (resp. port numbers) we received from each DNS server were within 500 of the previous one. The second check is for whether 50% of query TXID numbers (resp. port numbers) were with 1000 of the previous one.

0x20 Randomization. Dagon et al. proposed that resolvers would randomize the latter-cases in domain names queries as a cache poisoning defense [4]. Since the queried domain name is echoed in the response, this provides additional entropy to DNS queries that the resolver could validate.

The capitalization patterns of the queries we receive inform us whether or not the querying server utilizes 0x20 randomization. If the server is using 0x20 randomization, we would expect there to be roughly 50% uppercase and lowercase letters in the queries across all queries received from this nameserver. We check whether the ratio of uppercase letters is between 30% and 70%, and if it does, we determine that this querier has 0x20 randomization.

DNS Configuration Results. We received queries for our SPF records from 8889 total nameserver IP addresses. For each of these, we measured whether the server is open or closed. We required at least 4 queries received from a nameserver in order to determine whether it used randomization. 5718 nameservers sent us at least 4 queries; for these servers we measured whether source port randomization, transaction ID randomization, and 0x20 randomization were used.

Of 1160 open nameservers that sent us at least 4 queries, 1153 (99%) used both transaction ID randomization and port randomization, and the remaining 7 (1%) used only transaction ID randomization. No open nameservers used 0x20 randomization. Of 4558 closed nameservers that sent us at least 4 queries, 4547 (99%) of them used both transaction ID randomization and port randomization. 10 closed nameservers used only transaction ID randomization, and and 1 used only port randomization. Only one closed nameserver used 0x20 randomization; it was among the 4547 that also used both other defenses.

We find that DNS defenses are nearly ubiquitous - 99% of all nameservers used both transaction ID randomization and port randomization, and the defenses of closed resolvers only barely differ from the defensees of open resolvers. Furthermore, most nameservers have taken the extra precaution of being closed. Of 8889 nameservers that queried us, 7303 (82%) did not respond to DNS query from outside the organization. One implication of this is that open resolvers are only a small part of the DNS ecosystem, and so DNS measurements conducted only on open resolvers may not be representative.

5 Related Work

The Domain Name System has been the main focus of many denial of service attacks for many years [21], and many methods for detection and mitigation have been proposed [19, 13, 1, 17, 26]. Prior surveys of the Domain Name System that measure both DoS mitigation and defenses against cache poisoning [2, 24, 23] have focused on open resolvers.

Recent work by Klein et. al. [16] also measures the responses of closed DNS servers by probing them using email. If an MTA receives an email sent to a nonexistent user, it will query the `MX` record of the sender's domain name in order to determine where to send a bounceback email. Sending emails to nonexistant users does not allow studying the deployment of SPF since the recipient MTA would discard the email before checking the sender's validity. Huston [12] measures the behavior of closed DNS resolvers in IPv6. The method works similarly to our own, by causing a remote server to query its own closed DNS server, but it uses targeted advertisements rather than email spam prevention.

Several works have evaluated the deployment of Sender Policy Framework in the context of email security [6, 7, 8, 18, 10], and several mention the risk of utilizing SPF in DoS attacks [9, 20]. The updated SPF standard [15] took this into account in the new version, and made a recommendation to limit the number of DNS query-causing terms checked. However, we have shown that this defense can be circumvented using `include` statements.

6 Recommendations

Standard update. The most recent version of the SPF standard (2014)[15][§11.1] discusses the possibility of malicious SPF terms and proposes to limit the number of "void lookups" (lookups that result in a response with 0 answers, or that

cause a name error) to 2 per SPF record, after which an error is returned. This is in addition to the maximum of 10 DNS query-causing terms limit. We recommend that both of these limits be global, rather than "resetting" when recursion occurs in `include` statements.

Implementations. We recommend that new versions of the SPF library [22, 3] follow our previous suggestion to using global counts of DNS queries and void lookups per email, rather than resetting these to 0 when recursion occurs and a new SPF record is fetched. We envision this being the default option.

Acknowledgements

We thank Jared Mauch for contributing the machines we used to scan the Internet address space for MTAs and store our results. Sharon Goldberg thanks Haya Shulman for useful discussions about DNS resolvers and email. This research was supported, in part, by NSF grants 414119 and 1350733.

Bibliography

[1] Hitesh Ballani and Paul Francis. Mitigating DNS DoS Attacks. In *Proceedings of Computer and Communications Security*, pages 189–198. ACM, 2008.

[2] Andreas Borgwart, Haya Shulman, and Michael Waidner. Towards Automated Measurements of Internet's Naming Infrastructure. In *Software Science, Technology and Engineering (SWSTE)*, pages 117–124. IEEE, 2016.

[3] The SPF Council. Sender Policy Framework, April 2014. http://www.openspf.org/.

[4] David Dagon, Manos Antonakakis, Paul Vixie, Tatuya Jinmei, and Wenke Lee. Increased DNS Forgery Resistance through 0x20-bit Encoding: Security via Leet queries. In *Proceedings of Computer and Communications Security*, pages 211–222. ACM, 2008.

[5] Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. ZMap: Fast Internet-wide Scanning and Its Security Applications. In Samuel T. King, editor, *USENIX Security Symposium*, pages 605–620. USENIX Association, 2013. ISBN 978-1-931971-03-4.

[6] Zakir Durumeric, David Adrian, Ariana Mirian, James Kasten, Elie Bursztein, Nicolas Lidzborski, Kurt Thomas, Vijay Eranti, Michael Bailey, and J. Alex Halderman. Neither Snow Nor Rain Nor MITM: An Empirical Analysis of Email Delivery Security. In *Internet Measurement Conference*, pages 27–39. ACM, 2015. ISBN 978-1-4503-3848-6. URL http://dl.acm.org/citation.cfm?id=2815675.

[7] Ian D Foster, Jon Larson, Max Masich, Alex C Snoeren, Stefan Savage, and Kirill Levchenko. Security by any other Name: On the Effectiveness of Provider based Email Security. In *Proceedings of Computer and Communications Security*, pages 450–464. ACM, 2015.

[8] I. Gojmerac, P. Zwickl, G. Kovacs, and C. Steindl. Large-Scale Active Measurements of DNS Entries Related to E-mail System Security. In *International Conference on Communications*, pages 7426–7432, June 2015. doi: 10.1109/ICC.2015.7249513.

[9] Amir Herzberg. DNS-based Email Sender Authentication Mechanisms: A Critical Review. *Computers & security*, 28(8):731–742, 2009.

[10] Ralph Holz, Johanna Amann, Olivier Mehani, Matthias Wachs, and Mohamed Ali Kâafar. TLS in the Wild: An Internet-wide Analysis of TLS-based Protocols for Electronic Communication. *CoRR*, abs/1511.00341, 2015. URL http://arxiv.org/abs/1511.00341.

[11] A. Hubert and R. van Mook. Measures for Making DNS More Resilient against Forged Answers. RFC 5452 (Proposed Standard), January 2009. URL http://www.ietf.org/rfc/rfc5452.txt.

[12] Geoff Huston. IPv6 and the DNS, October 2016. https://blog.apnic.net/2016/10/20/ipv6-and-the-dns/.

[13] Georgios Kambourakis, Tassos Moschos, Dimitris Geneiatakis, and Stefanos Gritzalis. Detecting DNS Amplification Attacks. In *International Workshop on Critical Information Infrastructures Security*, pages 185–196. Springer, 2007.

[14] Dan Kaminsky. Its the End of the Cache as we Know It. *Black-Hat USA*, 2008.

[15] S. Kitterman. Sender Policy Framework (SPF) for Authorizing Use of Domains in Email, Version 1. RFC 7208 (Proposed Standard), April 2014. URL http://www.ietf.org/rfc/rfc7208.txt. Updated by RFC 7372.

[16] Amit Klein, Haya Shulman, and Michael Waidner. Internet-wide Study of DNS Cache Injections. In *INFOCOM*, pages 1–9. IEEE, 2017.

[17] Marc Kührer, Thomas Hupperich, Christian Rossow, and Thorsten Holz. Exit from Hell? Reducing the Impact of Amplification DDoS Attacks. In *USENIX Security Symposium*, pages 111–125, 2014.

[18] A Malatras, I Coisel, and I Sanchez. Technical Recommendations for Improving Security of Email Communications. In *Information and Communication Technology, Electronics and Microelectronics*, pages 1381–1386. IEEE, 2016.

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

[20] Tatsuya Mori, Kazumichi Sato, Yousuke Takahashi, and Keisuke 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, pages 31–37, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0788-8. doi: 10.1145/2030376.2030380. URL http://doi.acm.org/10.1145/2030376.2030380.

[21] Vern Paxson. An Analysis of using Reflectors for Distributed Denial-of-Service Attacks. *ACM SIGCOMM Computer Communication Review*, 31(3):38–47, 2001.

[22] Wayne Schlitt. libspf2 - SPF Library. https://www.libspf2.org/.

[23] Kyle Schomp, Tom Callahan, Michael Rabinovich, and Mark Allman. On Measuring the Client-side DNS Infrastructure. In *Proceedings of Internet Measurement Conference*, pages 77–90, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-1953-9. doi: 10.1145/2504730.2504734. URL http://doi.acm.org/10.1145/2504730.2504734.

[24] Geoffrey Sisson. DNS Survey. http://dns.measurement-factory.com/surveys/201010/dns_survey_2010.pdf, November 2010. The Measurement Factory.

[25] M. Wong and W. Schlitt. Sender Policy Framework (SPF) for Authorizing Use of Domains in E-Mail, Version 1. RFC 4408 (Experimental), April 2006. URL http://www.ietf.org/rfc/rfc4408.txt. Obsoleted by RFC 7208, updated by RFC 6652.

[26] Saman Taghavi Zargar, James Joshi, and David Tipper. A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks. *IEEE communications surveys & tutorials*, 15(4):2046–2069, 2013.