SPFail: Discovering, Measuring, and Remediating Vulnerabilities in Email Sender Validation

Nathaniel Bennett
Brigham Young University
Provo, Utah, United States
bntnate5@byu.edu

Rebekah Sowards
Brigham Young University
Provo, Utah, United States
jbekah1@byu.edu

Casey Deccio
Brigham Young University
Provo, Utah, United States
casey@byu.edu

ABSTRACT

Email is an important medium for Internet communication. Secure email infrastructure is therefore of utmost importance. In this paper we discuss two software vulnerabilities discovered in libSPF2, a library used by mail servers across the Internet for email sender validation with the Sender Policy Framework (SPF). We describe a technique to remotely detect the vulnerabilities in a production mail server, and we use that technique to quantify the vulnerability of Internet mail servers. We also monitor the patch rate of affected servers by performing continuous measurement over a period of roughly four months. We identify thousands of vulnerable mail servers, some associated with high-profile mail providers. Even after private notifications and public disclosure of the vulnerabilities roughly 80% of the vulnerable servers remain vulnerable.

CCS CONCEPTS

• **Information systems** → **Email**; • **Security and privacy** → **Vulnerability management**; • **Networks** → Application layer protocols; Naming and addressing.

KEYWORDS

Email, SPF, Security, Measurement, DNS

ACM Reference Format:
Nathaniel Bennett, Rebekah Sowards, and Casey Deccio. 2022. SPFail: Discovering, Measuring, and Remediating Vulnerabilities in Email Sender Validation. In *ACM Internet Measurement Conference (IMC '22), October 25–27, 2022, Nice, France.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3517745.3561468

1 INTRODUCTION

The use cases of electronic mail (email) have grown significantly since its inception over forty years ago. While its original purpose was to provide a simple means of one-to-one communication over interconnected computers, email now acts as a trusted identity for countless online services, as many services employ password recovery and/or multi-factor authentication through one's email account. As such, the security of email accounts—and therefore email services—has become of great importance to the existing trust structure of the Internet.

In summer of 2021, we discovered two vulnerabilities in a software library used by many mail servers across the Internet. These vulnerabilities could potentially allow an attacker to remotely crash or gain unauthorized access to a mail server by sending the server carefully crafted messages using the Simple Mail Transfer Protocol (SMTP)—messages that lead the server to pull down an exploitative payload from the Domain Name System (DNS). Both vulnerabilities received a CVSS score of 9.8 out of 10 and were labeled as "critical". While we know of no active exploits of these vulnerabilities, they nonetheless make mail servers targets for compromise. In this paper, we: 1) detail the vulnerabilities themselves; 2) describe our methodology for remotely detecting their presence; 3) assess and attempt various wide-scale efforts to communicate the vulnerabilities to affected parties; 4) quantify patching over time; and 5) observe what effects our notification efforts had on patching.

The vulnerabilities herein discussed were found in the libSPF2 library. Mail servers use this library to validate the origin of an email message using the Sender Policy Framework (SPF) [12]. To validate an email message claiming to be from `user@example.com`, libSPF2 issues a DNS query of type `TXT` (text) for `example.com`, following the SPF specification. The content of the response is then parsed by the libSPF2 code. A carefully crafted `TXT` record can trigger a buffer overflow in vulnerable libSPF2 code. This is detailed further in Section 4.1.

After discovering the vulnerabilities, we developed a technique to remotely detect them in such a way that was minimally intrusive (e.g., minimized unsolicited email) and caused no harm or degraded service to the mail server. Section 4.2 further describes the remote detection technique. This technique allowed us to perform a large-scale measurement on thousands of Internet mail servers from two different data sets. Section 5 describes the data sets and the measurement methodology.

In each notification, we not only pointed out the problem but also provided a timeline on which we would be issuing a public disclosure. The public disclosure took the form of a Common Vulnerabilities and Exposures (CVE) published 60 days after the last private notifications. We describe the communications and methodology associated with our private and public disclosures in Section 6.4.

Beginning shortly after the discovery of the vulnerabilities and continuing through the date of the public disclosure, we routinely measured the set of servers initially identified as vulnerable. This allowed us to understand trends in patching, including triggers such as private notification or public disclosure or distribution uptake. Section 7 shows the results of our longitudinal measurements.

IMC '22, October 25–27, 2022, Nice, France
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9259-4/22/10…$15.00
https://doi.org/10.1145/3517745.3561468

Our measurements provide a unique view into how individual server maintainers respond to private vulnerability disclosure, what effect the assignment of a CVE and CVSS score may have during public disclosure, how various package managers respond to such disclosures, and what correlation, if any, these events have on the overall patch rate of a given vulnerability. The results of our work additionally motivate a more comprehensive measurement and analysis of package manager responses to CVE publications to better understand how various factors (e.g. NVDD score, package use, disclosure timing) influence the rate at which a given vulnerability is patched.

The major contributions of this paper include the following:

- a description of the libSPF2 buffer overflow discovered;
- a technique for remotely detecting the vulnerabilities in a benign way; and
- a large-scale, longitudinal measurement of vulnerability patching.

Through the measurements we carry out, we identify thousands of vulnerable mail servers, comprising 17% of the total servers we tested. We observed a higher correlation of patching associated with the public disclosure than with private notifications issued earlier. Even after all the notifications, roughly 80% of the servers we found to be vulnerable remained vulnerable.

2 BACKGROUND

2.1 Simple Mail Transfer Protocol

We briefly summarize the Simple Mail Transfer Protocol (SMTP), as it relates to our research. A sending mail transfer agent (MTA)—the *client*—opens a TCP connection to a receiving MTA—the *server*—and issues a series of commands to the server. In a typical SMTP transaction, the commands are as follows:

1. `HELO` (or newer `EHLO`). An initial greeting.
2. `MAIL FROM`. Specifies the email address from which the message will be sent.
3. `RCPT TO`. Specifies the email address to which the message is destined.
4. `DATA`. Indicates that the client is ready to send the message itself, including headers and body.

After receiving a successful response code for the `DATA` command, the client sends the email message, ending with a period on a line by itself, i.e., "`<cr><lf>.<cr><lf>`".

2.2 Sender Policy Framework

One important shortcoming of SMTP is that the sending MTA is not restricted in what sender address it passes to the server in the `MAIL FROM` command. Thus, any client could send email claiming to be from any email address. In order to prevent an SMTP `From` address being so easily spoofed, the Sender Policy Framework (SPF) was introduced [12].

SPF provides receiving servers with a mechanism to verify that a client is authorized to send email from a particular domain. It utilizes the Domain Name System (DNS)—the system best known for mapping domain names to IP addresses (e.g., `example.com` → `192.0.2.1`) [15, 16]—as a trusted source for domain owners to publish information on what IP addresses a recipient may expect

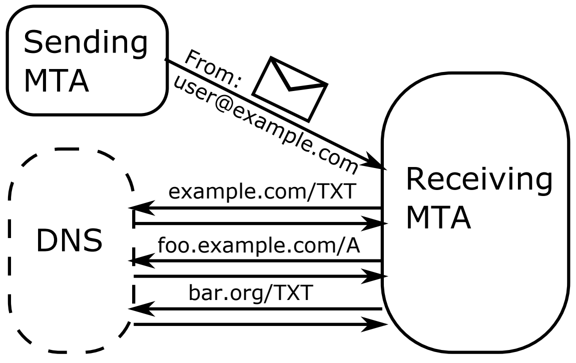


Figure 1: Flow related to SMTP and SPF policy lookup. An MTA receiving an email message from sender `user@example.com` first issues a DNS lookup for `example.com/TXT` to find its SPF policy. It then issues additional requests to flesh out that policy.

email from for that domain. This set of IP addresses, referred to as an SPF *policy*, is published in a record of type `TXT` (text).

Whenever an email server receives email claiming to be from a given domain, it queries the DNS for the domain's SPF policy and uses the response to check the IP address from which it received the email. If the IP address is not considered a legitimate sender, the email server may take appropriate action, such as rejecting the email or labeling the email as spam.

The SPF policy itself is a human-readable ASCII string that contains zero or more indicators of valid sender addresses. An indicator may be an IP address, a domain that might resolve to an IP address, or even a reference to another SPF record that contains additional indicators. These indicators each have special labels, including the following: `a` for domains whose IP addresses are allowed; `ip4` and `ip6` for specific IPv4 and IPv6 addresses that are allowed; and `include` and `redirect` for references to other SPF policies that are to be retrieved and interpreted, for inclusion with or replacement of the current policy, respectively. For example, the following might represent an SPF policy used by the domain `example.com`:

```
v=spf1 a:foo.example.com ip4:192.0.2.1
       include:bar.org -all
```

This means that the IPv4 and IPv6 addresses corresponding to `foo.example.com` are allowed senders, as is `192.0.2.1`. Any legitimate senders for `bar.org` are also considered legitimate for `example.com`. The `-all` mechanism indicates that anything else is fraudulent. The SPF lookup of this policy, in connection with an SMTP communication, is shown in Figure 1.

The SPF specification additionally allows for certain *macros* to be used within domains. These macros are intended to give greater flexibility for SPF records to be defined based on information that can only be determined at the time of SMTP communication. Options for these macros include the username or domain part of a sender's email address (macros `l` and `d`, respectively) or the IP address of the sender (macro `i`). The output of a macro can optionally be URL-encoded by using the uppercase variant of a given macro (such as `L` or `D` for the aforementioned macros). Certain options can be set to split a macro's output into labels based on a delimiter, reverse delimited, and truncate the number of those labels used for the macro expansion. This is done by appending a numerical value to the macro designating the number of labels to use, optionally followed by the `r` (reverse) identifier. For example, given the sender `user@example.com` from the `MAIL FROM` command, the following strings would be translated accordingly in an SPF policy:

- `%l` (username of sender email address) → `user`
- `%d` (domain of sender email address) → `example.com`
- `%d2` (two highest labels only) → `example.com`
- `%d1` (highest label only) → `com`
- `%dr` (labels in reverse order) → `com.example`
- `%d1r` (labels in reverse order, highest only) → `example`

Consider the following SPF policy:

```
v=spf1 a:foo.example.com ip4:192.0.2.1
       include:bar.org a:%{d1r}.foo.com -all
```

In this case, an email from `user@example.com` would trigger the same DNS lookups as those in Figure 1 but with an additional lookup for `example.foo.com`—i.e., the last (`1`) of the reversed (`r`) set of labels of the email domain (`d`), prepended to `foo.com`.

3 PREVIOUS WORK

In work related to our own research, others have carried out studies involving the discovery and measurement of vulnerabilities and patching, as well as general measurement of email sender validation.

In 2014, Durumeric et al. measured the prevalence of the Heartbleed vulnerability in the OpenSSL library [5]. The authors launched an initial scan for vulnerable HTTPS servers within two days of the public vulnerability disclosure and found that roughly 11% of the Alexa Top 1 Million and 6% of HTTPS servers identified in a scan of the IPv4 address space were vulnerable to Heartbleed. They monitored patching over time by scanning these two sets of HTTPS servers every 8 hours for three months. In that time the authors observed the rate of vulnerable HTTPS servers shrink to 3.1% and 1.9%, respectively.

Shen et al. introduce multiple attacks to bypass email validation mechanisms [26], including SPF [12], DomainKeys Identified Mail (DKIM) [8], and Domain-based Message Authentication, Reporting, and Conformance (DMARC) [14]. Their methodology does not include software vulnerabilities, per se, but rather ways to abuse the protocols themselves to make them ineffective. They applied their techniques on 30 popular mail services and 23 mail clients to show their effectiveness.

Jeitner et al. explore the problem of DNS response sanitation [10]. Their work reveals that many software and services that rely on the DNS do not properly check the responses, making them vulnerable to attack through a DNS side channel. Among the vulnerabilities they identify is a stack buffer overflow in the libSPF2 library—a vulnerability similar to those we present herein, in terms of software library, category, and even timing (their finding was within days of ours), even though they were discovered independently. They tested 3 million open DNS resolvers for some of the vulnerabilities; other vulnerabilities could not be scanned for in a benign way and were tested instead in a lab environment. The libSPF2 vulnerabilities that we detail in this paper differ in that they can be remotely detected without causing any harm to the remote system. This allowed us not only to measure, but also to notify.

In addition to carrying out measurements specific to vulnerabilities, whether in protocol or software, more general measurements have been performed to quantify MTAs that perform email sender validation with SPF, DKIM, and/or DMARC, as well as domains that make themselves available for validation. Foster et al. investigated the prevalence of SPF, DKIM, and DMARC validation in 22 popular mail providers [6]. Scheffler et al. performed several different SPF validation tests on approximately 8,000 mail servers identified in a TCP SYN scan [23]. Deccio et al. evaluated SPF, DKIM, and DMARC validation on a large set of MTAs, finding that 85% perform SPF validation, and just over half use all three mechanisms (SPF, DKIM, and DMARC) for email sender validation [3].

Durumeric et al. performed a study of incoming mail received by Google mail servers and found that 92% of those messages could be SPF-validated—because they were sent from a domain with policy that could be properly retrieved and validated [4].

Our work captures the spirit of all of the aforementioned research. We discover and disclose vulnerabilities, similar to the work of Shen [26] and Jeitner [10]; we measure patching of vulnerabilities, similar to the work of Durumeric [5]; and we use techniques similar to that of Deccio [3] and Scheffler [23] to carry out our measurements, eliciting SPF activity of Internet MTAs to detect the presence of the libSPF2 vulnerabilities.

4 VULNERABILITY DISCOVERY AND DETECTION

4.1 Vulnerability Discovery

Both vulnerabilities were discovered by means of manual code inspection, rather than more common static analysis or fuzzing techniques. A cursory inspection of the code repository revealed several warning signs that vulnerabilities could be present in the library:

- The last commit to the repository was over 4 years ago, and steady work on the code had ceased around 2013.
- Portions of the library had complex control flow—nested loops spanning hundreds of lines of code, for instance.
- The library had a history of remotely exploitable vulnerabilities, such as those discovered by Dan Kaminsky in 2008 [11].
- Comments within the code such as: "The following code confuses both me and Coverity"[28]. (Coverity is a commercial static code analysis tool managed by Synopsys[31].)

4.1.1 *Vulnerability 1: URL Encoding sprintf Overflow.*

During the process of more closely inspecting the code, obsolete functions (such as `scanf` or `gets`) were searched for to determine whether their use could lead to memory corruption. One such instance was discovered:

```
sprintf(p_write, "%%%02x", *p_read);
```

The above code snippet is meant to encode a supplied non-ASCII character into its URL-ready equivalent. Both `p_write` and `p_read` in the above code are of type `char*`. Under normal conditions, this code was expected to take the 8-bit value stored at the location `p_read` and write it to location `p_write` in hexadecimal form. For example, a `char` with a value of 15 would be copied as "`%0F`" (plus a null-terminating byte) to location specified by `p_write`. The code's author assumes the output will be a constant length, as evidenced by a comment in the code: "No point doing snprintf... we know we're going to get 4 characters anyway" [27]. Unfortunately, this is not always the case.

The ISO C standard [19] stipulates that the `sprintf` function receive an *unsigned* integer as input when printing hexadecimal values in the manner shown above. Thus, any negative-valued signed `char` passed to the function would be converted into its 32-bit signed representation and then cast to an unsigned value. As an example, the value `-2` (represented as `0xFE` in two's complement) would be converted into `0xFFFFFFFE` before being used by `sprintf`, resulting in a 10-byte output instead of the expected 4. In the case of libSPF2, the above code snippet is executed when an SPF policy 1) contains a macro that indicates URL encoding should be performed *and* 2) the macro expansion contains characters between `0x80` and `0xFF`. In such instances, the unexpected six extra characters will overflow into allocated heap space.

4.1.2 Vulnerability 2: URL Encoding Buffer Length Reassignment. While devising a proof-of-concept that would demonstrate this vulnerability, a second heap overflow was discovered within the same block of code. In the event that an SPF macro specifies label reversal, a variable used to track the intended length of a buffer will be overwritten with a much smaller value. If the macro specifies URL encoding as well, a buffer will be allocated based on the erroneous length field and filled with data specified by the SPF record, leading to overflow. This memory corruption was much more flexible than the first, as it allows up to 100 arbitrary characters to be be written past the end of the allocated buffer.

4.2 Remote Detection

Under normal circumstances, memory corruption vulnerabilities such as buffer overruns are difficult to directly measure without leading to a corruption or crash on the server's end; past studies of such have mostly used indirect methods to fingerprint a particular vulnerable version of software rather than probing the vulnerability directly [5, 22]. In contrast to this, one of our discovered heap overflow bugs happened to be very well-suited for non-intrusive remote detection with a high degree of certainty. The bug would uniquely modify outgoing DNS queries that were part of validation for certain SPF records, but would not corrupt memory unless URL encoding was additionally specified in the record. This behavior provided an opportunity to precisely measure vulnerable hosts in a benign manner. We explain the specifics of how this is done subsequently.

When an MTA retrieves an SPF record, the server itself is responsible for expanding out any macros that may be contained within the record, as described in Section 2.2. Once macros are expanded out, additional DNS queries are issued for domains contained within the policy. An RFC-compliant SPF implementation would normally replace the macro with the data represented by it, and then reverse and truncate the segments of that data depending on the options specified within the macro. Some non-RFC-compliant implementations may fail to fully implement these options, leading to erroneous DNS queries. For example, non-compliant implementations may fail to reverse labels, fail to truncate, or even fail to perform any macro expansion at all. The behavior in the vulnerable version of libSPF2 is uniquely different to all of these, overwriting part of the expanded label with erroneous characters that would not normally occur in any other SPF implementation.

For example, consider the following mechanism, with macro, contained within an SPF policy: `a:%d1r.foo.com`. When the library processes this mechanism with a sender email address of `user@example.com`, an SPF validator would initiate a DNS request for one of the following domains:

- `example.foo.com`: RFC-compliant behavior
- `com.example.foo.com`: Non-compliant behavior (missing truncation)
- `com.com.example.foo.com`: Vulnerable libSPF2 behavior

Thus, a vulnerable implementation can be detected based on the queries observed at the authoritative DNS server.

5 VULNERABILITY MEASUREMENT

5.1 Methodology

Our measurement methodology involves interacting with Internet MTAs in the following way to detect their vulnerability remotely:

(1) Our email client establishes an SMTP connection with the MTA.
(2) Our client advertises a `MAIL FROM` address with a domain that is unique to the MTA. The domain used is a subdomain of `spf-test.dns-lab.org`, which is under our administrative control.
(3) Our client terminates the SMTP connection before/during the transmission of email.
(4) Our DNS server logs queries related to SPF lookup that contain the unique `MAIL FROM` address domain.
(5) The email server's SPF behavior is then determined based on the sequence of DNS queries it made.

Two approaches were taken in terminating the SMTP connection, designated as **NoMsg** and **BlankMsg**. For **NoMsg**, our email client would send up to the `DATA` command (see Section 2.1) and then terminate the connection before sending any email message. This approach was attempted first; if it failed to trigger an SPF query, the **BlankMsg** approach would be tried. For **BlankMsg**, our email client would send up to the `DATA` command and then send the end-of-data code, effectively transmitting an entirely empty email. Individual testing on email accounts under our control revealed that these blank messages were most often rejected or filtered rather than being delivered to inboxes. Any subsequent measurements would then use whichever approach was successful at triggering an SPF lookup. Both of these minimized unsolicited email being delivered to inboxes.

For each tested server, a unique 4- or 5-digit alphanumeric label was inserted into the advertised `MAIL FROM` domain. An unique label was also generated for each test suite and added to the domain. Together, these labels guaranteed that each server could be associated with the DNS queries it performed for a given experiment. The inclusion of these unique labels additionally guaranteed that every lookup would be received by our DNS servers rather than being cached and served by a recursive resolver or within a local system.

Domain Set	# 2-Week MX	Alexa 1000	Alexa Top List
2-Week MX	22,911 (100%)	135 (0.5%)	2,922 (12.7%)
Alexa 1000	135 (13.5%)	1,000 (100%)	1,000 (100%)
Alexa Top List	2,922 (0.7%)	1,000 (0.2%)	418,842 (100%)

Table 1: Overlap in domain measurement sets. Each cell represents the number/percent of domains in the row's domain set that also exist in the column's domain set.

Our DNS servers were configured to accept these arbitrary domain labels sent by our email client and return an SPF TXT record similar to the following:

```
v=spf1 a:%{d1r}.<id>.<suite>.spf-test.dns-lab.org
a:b.<id>.<suite>.spf-test.dns-lab.org -all
```

Note that `<id>` and `<suite>` are the alphanumeric labels mentioned previously, for unique identification; the DNS server would automatically populate these fields with the ID and test suite labels contained in the DNS request domain.

Once a given email server retrieved this SPF record, its subsequent A/AAAA record queries would then reveal how that server expands the `%{d1r}` macro. If the server uses a vulnerable version of libSPF2, the macro would expand to form a uniquely erroneous domain and the next DNS lookup would reflect that error.

For our measurements, we classify a domain as vulnerable if any one of its associated IP addresses is measured as vulnerable in either of the **NoMsg** or **BlankMsg** tests. For subsequent tests, we consider a vulnerable domain to be not vulnerable or patched once all of its initially vulnerable IP addresses measure as RFC-compliant rather than vulnerable. We consider a domain's measurement to be uncertain if any of its vulnerable IP addresses fail to return a conclusive result.

IP addresses for each domain were obtained through standard MX record queries, followed by A/AAAA queries for those MX records returned; if a given domain had no MX record assigned, its A record would instead be used, per RFC 5321 [13]. This mapping of domain to mail server addresses was obtained immediately prior to our first measurement and used throughout the 4-month longitudinal tests. As a result, there could be some variance in accuracy for the small subset of domains that changed MX records during the course of our measurements. To help account for this, we performed an additional "snapshot" measurement at the end of our longitudinal tests with updated MX records; findings specific to this snapshot are discussed in Section 7.2.

5.2 Measurement Scope

Two sets of domains were selected as sources of measurable email servers. The first set was two weeks of incoming/outgoing email address domains observed within traffic at a medium-sized university, measured February 2021. We will hereafter refer to this set as 2-Week MX. This set was chosen to ensure that our tests were focused on email domains that are commonly used. The second set was a subset of the Alexa top domains list, downloaded on October 4, 2021 [1]. We will refer to this as the Alexa Top List data set. This set was selected to provide a wide coverage of measurement on Internet domains that may not be observed in everyday email traffic, but that would nonetheless be significantly affected by an email server vulnerability. In our analysis, the top 1000 domains in the Alexa list were additionally considered in a separate group to observe trends among high-ranking domains. We will hereafter refer to set as the Alexa Top 1000. Table 1 shows the relative overlap in domains for these groups.

Alexa Top List		2-Week MX	
TLD	**Count**	**TLD**	**Count**
com	230,801	com	11,182
ru	19,844	org	3,946
ir	17,207	edu	2,108
net	16,672	net	1,441
org	14,427	us	828
in	7,856	gov	255
io	5,122	uk	241
au	4,685	cam	232
vn	4,326	ca	172
co	4,250	de	149
ua	4,139	work	142
tr	4,117	cn	99
uk	3,429	au	92
id	2,997	it	90
ca	2,835	top	86

Table 2: Most common TLDs for the Alexa Top List and 2-Week MX domain sets. This includes domains that did not accept SMTP connections.

The distribution of top-level domains (TLDs) for each set can be seen in Table 2. Domains under `com` were the most measured of any TLD for both sets by a wide margin; other than that, the two sets of domains appear to represent two unique subsections of Internet email domains, at least by frequency of TLDs.

5.3 Measurement Timeline

Measurements were taken according to the following timeline:

- **October 11, 2021**: Initial measurement of all Alexa Top List/2-Week MX domains
- **October 26, 2021**: Measurements of vulnerable domains begin; taken every 2 days
- **November 15, 2021**: Private notification sent to vulnerable email servers
- **November 30, 2021**: Measurements paused
- **January 15, 2021**: Measurements resumed every 2 days
- **January 19, 2021**: Public disclosure of CVE-2021-33912 and CVE-2021-33913
- **February 14, 2022**: Last measurement taken

An initial measurement was performed on the combined IP addresses of both sets of domains to determine what proportion of active email servers used a vulnerable version of libSPF2. Following analysis of this initial measurement, a series of evenly-spaced measurements were taken every 2 days for servers found to be vulnerable. The first such series was to measure the effect of private notifications on the rate of vulnerable servers; the second, to measure the effect of public disclosure and any associated package manager updates. By maintaining regular and even measurements over time, we were able to accurately monitor the rate at which servers were patched following various events and methods of disclosure.

6 ETHICAL CONSIDERATIONS

Our research involved multiple large-scale measurements on email systems being used by various companies, individuals, organizations, and governments. As with any large-scale measurement, the impact of experiments on network systems is an important ethical concern. We employed several methods to minimize that impact, which we will discuss below.

Our measurements also involved software vulnerabilities that exist in the wild and that affect network systems. Because of the potential these vulnerabilities have to disrupt such systems, we sought to notify relevant individuals and organizations about both vulnerabilities in a responsible and timely manner.

6.1 Minimized Network Load for MTAs

Several precautions were put in place for our measurements to ensure that the load on individual email servers would be negligible. First, duplicate IP addresses were only tested once so that any email server hosting multiple domains would not be repeatedly queried. The remaining domain/IP address pairs were then grouped by common email domain and tested sequentially so that only one SMTP connection would be established with a given email domain at any given time. Last, a hard limit of 250 concurrent outgoing SMTP connections was set to limit any potential for impact caused by increased network traffic.

In some cases, more than one connection would need to be made with an IP address (e.g., to try a new recipient username). In these instances, a minimum 90-second waiting period was applied between connections. This waiting period was also applied between tests of multiple IP addresses with a common email domain. In the event that a given server requested an email be delivered after a short period (i.e., grey-listing), our server would wait for eight minutes before attempting another SMTP connection.

As a final measure to reduce network load, the full set of MTAs was used only for the initial measurement. All other tests were restricted to addresses that had been found to be either vulnerable (7,212 IP addresses) or inconclusive but potentially re-measurable (721 IP addresses).

6.2 Minimized Email Delivery

The **NoMsg** and **BlankMsg** approaches to triggering SPF lookup (see Section 5.1) are both designed to minimize the potential for unsolicited email being delivered. In our approach, the **NoMsg** approach is attempted first, as it guarantees that no email will be delivered in the SMTP transaction. Following this, the **BlankMsg** approach is used on servers that accepted our SMTP connection but didn't perform any SPF validation (likely deferring SPF queries until after email data is received). The headers, subject line and body of the **BlankMsg** data are all intentionally left blank to maximize the likelihood that the received blank email will be discarded by the recipient's mail server.

To add to the emailing approaches used, the actual SPF record served for each source domain was configured so that our servers' addresses would fail checks. DMARC records were likewise published for our source domains indicating that the emails should be rejected outright rather than being delivered.

6.3 Email Username Selection

A curated list of fourteen potential email usernames was used for each of our experiments. The list of usernames were tried in the following order:

(1) `mmj7yzdm0tbk`
(2) `noreply`
(3) `donotreply`
(4) `no-reply`
(5) `postmaster`
(6) `abuse`
(7) `admin`
(8) `administrator`
(9) `newsletters`
(10) `alerts`
(11) `info`
(12) `auto-confirm`
(13) `appointments`
(14) `service`

The username `mmj7yzdm0tbk` is a randomly generated alphanumeric string; it was tested first, along with variants of `noreply`, to maximize the chance that any email that made it to an inbox would be delivered to a non-existent or unmonitored account. More than half of all **BlankMsg** tests were performed using one of these usernames. The remaining usernames were accounts likely to be owned or controlled by an administrator rather than a normal user.

6.4 Disclosure and Remediation Plan

With regards to our schedule for disclosing both vulnerabilities, we wanted to ensure that our experiments in measuring various disclosure methods did not undermine the main purpose of such disclosures—to help as many email servers get patched as possible. As such, we responsibly disclosed both vulnerabilities in the following order:

- **October 2021**: Notified package managers and the intended private/public disclosure schedule
- **November 15, 2021**: Sent a private email notification to email servers that were measured to be vulnerable
- **January 19, 2022**: Publicly disclosed CVE-2021-33912 and CVE-2021-33913

6.5 Ethics Committee Approval

Institutional Review Boards (IRBs) are often used for formal evaluation and approval of human subjects. While our institution has an IRB, our research did not involve human subjects, so it did not fall under the purview the IRB. In lieu of formal IRB approval, we consulted an ad-hoc ethics committee consisting of subject-matter experts within our same department. They approved our experiment plan as described herein.

Table 3: NoMsg/BlankMsg Test Outcomes By Domain Set

	Alexa Top List				2-Week MX				Top Email Providers[a]	
	Domains		Addresses		Domains		Addresses		Domains	
	Count	%	Count	%	Count	%	Count	%	Count	%
Total Tested	**418,840**	**100%**	**174,679**	**100%**	**22,911**	**100%**	**11,203**	**100%**	**20**	**100%**
Connection Refused	109,559	26%	81,515	47%	2,281	10%	2,773	25%	0	0%
NoMsg Test	**309,281**	**74%**	**93,164**	**53%**	**20,630**	**90%**	**8,430**	**75%**	**20**	**100%**
SMTP Failure	62,466	20%	34,167	37%	1,187	5.7%	2,032	24%	2	10%
SPF Measured	48,205	16%	12,528	13%	2,399	12%	1,953	23%	5	25%
SPF Not Measured	198,610	64%	46,469	50%	17,044	83%	4,445	53%	13	65%
BlankMsg Test	**198,610**	**47%**	**46,469**	**27%**	**17,044**	**74%**	**4,445**	**40%**	**13**	**65%**
SMTP Failure	6,512	3.3%	2,209	4.8%	440	2.6%	352	7.9%	4	30%
SPF Measured	151,753	76%	27,139	58%	14,204	83%	2,337	53%	8	62%
SPF Not Measured	40,345	20%	17,121	37%	2,400	14%	1,756	39%	1	7.7%
Total SPF Measured	**199,958**	**48%**	**39,667**	**23%**	**16,603**	**73%**	**4,290**	**38%**	**13**	**65%**

[a]Email domains most commonly seen in use as measured by Foster, et al. [6]

7 RESULTS

7.1 Initial Measurement Findings

Over 400K domains were tested in our measurement, amounting to around 180K unique IPv4/IPv6 addresses. Of this combined set of IP addresses, roughly half of the addresses did not accept any connection. The latter mostly came from domains that did not have MX records set, indicating that there may be no intended email delivery method for those domains (or that if there were, we were unable to connect to it at the time of measurement).

The subsequent results of both the **NoMsg** and **BlankMsg** tests can be seen in Table 3. Because both the **NoMsg** and **BlankMsg** test results are equally valid in determining server vulnerability, we will consider them together in subsequent analysis. Of the remaining 97K addresses, about 55K accepted our SMTP messages up to the point of termination for our **NoMsg** test, which resulted in just over 13.5K conclusive SPF behavior measurements. Following this measurement, 48K IP addresses were run using the **BlankMsg** test, which yielded an additional 28K conclusive SPF behavior measurements. Addresses were only used for the **BlankMsg** test if they succeeded the **NoMsg** test but failed to elicit any SPF lookup. Note that many of the conclusive measurements from the **NoMsg** test came from connections that were actually rejected at some point before completing; this explains why all but 7K of the **NoMsg** tests that passed were retried using the **BlankMsg** approach, despite there being over 13.5K conclusive SPF results in the **NoMsg** test.

A total of 41,692 IP addresses sent conclusive SPF queries during the course of this initial measurement, amounting to 43% of reachable SMTP servers. When mapped back to domains, however, this amounted to 66% of the reachable measurement set, or 214,802 domains. For comparison, recent measurements performed in 2020 have shown SPF validation in email servers to be at about an 85% adoption rate [3].

The findings of this initial measurement are summarized in Table 4. Around 1 in every 6 IP addresses that performed SPF validation were found to be using a vulnerable version of libSPF2, and close to a quarter of addresses incorrectly expanded SPF Macro strings either in the vulnerable pattern or some other non-RFC compliant way (see Section 4.2). These rates were somewhat lower for the domain data set gathered from email traffic—1 in 10 IP addresses were vulnerable and 1 in 6 incorrectly expand macros—but they nonetheless indicate that libSPF2 remains a relevant part of email infrastructure across many production mail servers.

Table 4: SPF Initial Results Breakdown

Domain Set	# Measured	Vulnerable	Non-RFC[a]
Domains			
Alexa Top 1000	655	28 (4.3%)	36 (5.5%)
2-Week MX	16,603	512 (3.1%)	995 (6.0%)
Alexa Top List	199,958	18,210 (9.1%)	13,704 (6.9%)
Combined	214,271	18,660 (8.7%)	14,509 (6.8%)
IP Addresses			
Alexa Top 1000	570	87 (15%)	48 (8%)
2-Week MX	4,290	429 (10.0%)	323 (7.5%)
Alexa Top List	39,667	7,077 (17.8%)	2,551 (6.4%)
Combined	41,692	7,212 (17.3%)	2,660 (6.4%)

[a]Erroneous macro expansion, but not vulnerable

7.2 Overarching Trends in Patching

Figure 2 shows the final vulnerability distribution of the domains that were initially vulnerable in October 2021. This final snapshot measurement was taken in February 2022. It highlights the cumulative effect of several vulnerability disclosure methods—private disclosure, early notification of package managers to assist preparation of patches, and public disclosure with CVE identifiers assigned to the vulnerabilities.

Patching across all domains stood at about 15% as of February 2022, with most of that patching occurring in the Alexa Top List data set. Of all groups, the Alexa Top 1000 patched the least by far, with less than 10% of domains patched. Because the Alexa Top 1000 contains some of the most commonly visited domains across the Internet, this is one of the most concerning results immediately visible in this data.

The 2-Week MX data set had more inconclusive domains at the end of our measurements than the other data sets we used for our testing, though the percentage of patched domains was

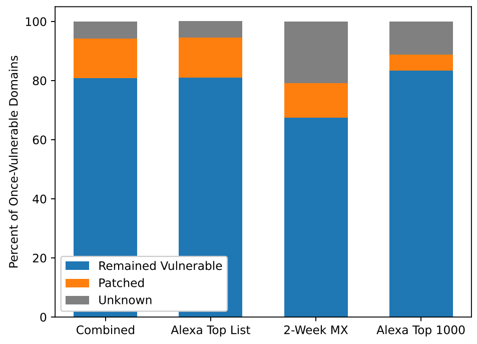


Figure 2: Overarching trends in domains patched. Domains are considered 'unknown' if they were never conclusively detected to be vulnerable/patched for measurements taken in February 2022.

comparable to the combined percentage. One possible explanation for the greater proportion of inconclusive measurements is that the 2-Week MX data set may have contained a higher proportion of domains used temporarily for email spam. A closer look at our 2-Week MX data set revealed some domains had characteristic signs of being used for email spam (e.g. domains that attempt to appear similar to well-known sites or that come from TLDs with an anecdotally high incidence of spam domains). These domains were included in the data set because they 1) sent email that elicited a DNS lookup during our two-week domain collection period and 2) had retrievable MX records when the data set was finalized. DNS lookups conducted in February for many of these domains showed no MX records where they previously existed. Although some spam domains may have been included, the majority of the domains included in the 2-Week MX data set were legitimate email domains.

7.3 Geographic Trends

Latitude/longitude coordinates were obtained for each vulnerable IP address using the DbIP geolocation database [2]. To give a more clear representation of the relative concentrations of these addresses, coordinates are aggregated into geographically distinct buckets, and the resulting frequencies of vulnerable servers are shown as a choropleth map in Figure 3a. The proportion of servers that eventually patched the vulnerability for each geographically distinct bucket is depicted in a similar manner in Figure 3b.

As a general trend, vulnerable email servers were discovered throughout most populous regions of the world, with a slightly higher concentration showing up in Europe. Almost all areas showing high frequencies of patching (over 60%) were those that contained few IP addresses, though there are a few notable exceptions that will subsequently be explored. Servers located in several regions show almost no indication of any patching, such as China and Taiwan, Central and South America, and Russia.

An inspection of the TLDs of affected servers reveals a similar trend—one or two countries with surprisingly high patch rates,

Table 5: Best/Worst Patch Rates for TLDs with 50 or more Initially Vulnerable Domains

TLD	# Patched	# Initially Vulnerable	% Patched
.za	118	150	79%
.gr	40	53	75%
.de	84	183	46%
.eu	16	56	29%
.tr	65	232	28%
...			
.ir	67	2,130	3%
.il	5	182	3%
.by	2	98	2%
.ru	37	2,030	2%
.tw	0	96	0%

several others that tend towards 20%, and a few outliers lagging behind with few or no patched domains. Table 5 highlights this discrepancy, listing the top and bottom 5 TLDs by patch rate for TLDs containing a non-trivial number of affected domains. As a reference benchmark, domains ending in com had both the largest quantities of vulnerable and patched servers (8,412 initially vulnerable and 1,266 eventually patched), with a patch rate of 15%.

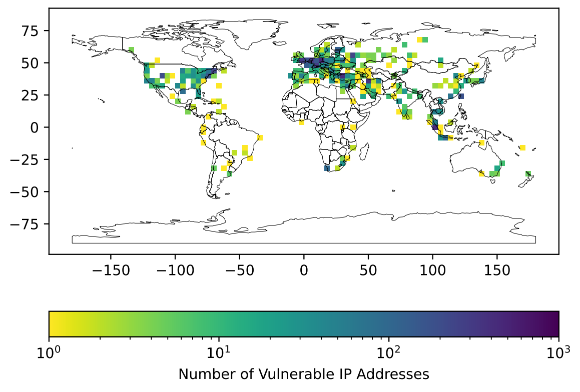
Of particular note is the TLD for South Africa (za), the only TLD to have a majority of its email servers patched in our four-month window of measurement. This trend is correlated with the high rate of patching among IP addresses in South Africa shown in Figure 3b. Further analysis of these domains revealed that 98% of those patched (all but 2 servers) were patched within the first window of measurements in October/November, before any public disclosure was released. The private notification sent out on November 15, 2021, seems to have had little effect on this trend; most servers were patched before this date, and the remaining few did not show any indication of having opened the email prior to patching.

Apart from the TLD for South Africa, most of the country-specific TLDs follow the same pattern as was seen in Figure 3b, with Taiwan, Russia, and Belarus having the lowest rates of patching and select European TLDs exhibiting higher-than-average patch rates.

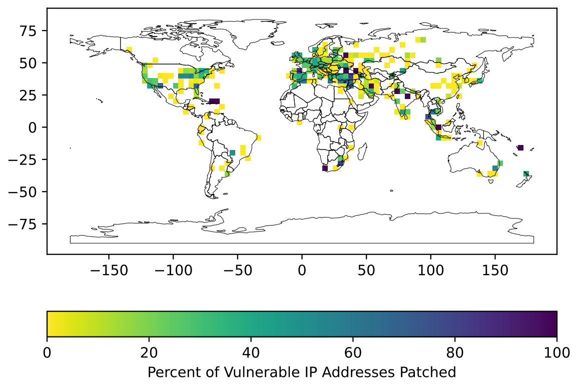
Note that disruptions in Internet services related to the 2022 Russian invasion of Ukraine were detected starting on February 17, 2022, and that the earliest reported disruptions of select Ukrainian Internet sites started on February 15, 2022 [18]. Our vulnerability measurements were completed by February 14 2022, thereby precluding the possibility of skewed measurements due to Internet outages related to the invasion.

7.4 Measurement by Site Ranking

Both the Alexa Top List and the 2-Week MX sets provided some metric for measuring the relative usage of a given domain, the former through explicit rank of each domain and the latter through the number of DNS MX queries each domain was detected in. Figure 4 shows servers measured to be vulnerable according to these relative rankings, and it partitions the full range of domain ranks into 20 distinct buckets. For comparison, it also shows the number of domains that were later patched for each bucket. As we can

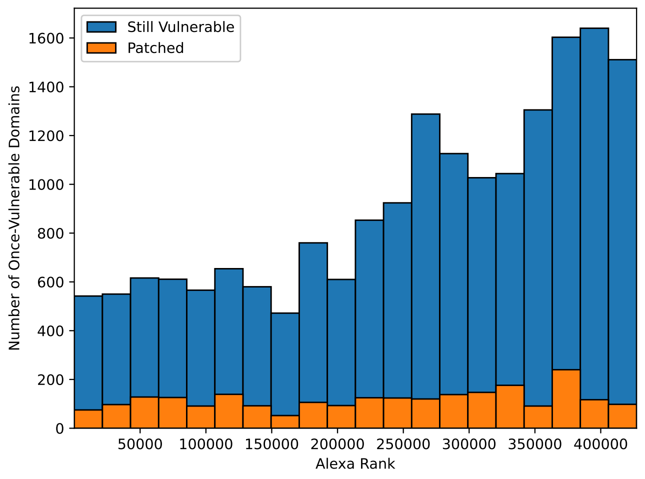


(a)

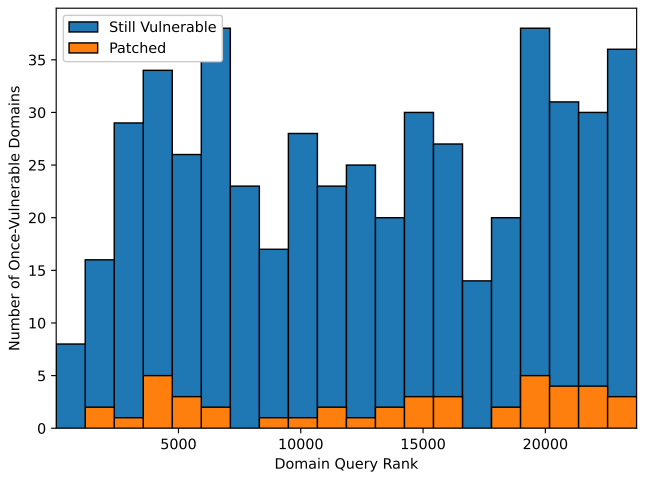


(b)

Figure 3: Geographic distribution of (a) vulnerable IP addresses and (b) patched IP addresses.



(a) Alexa Top List



(b) 2-Week MX

Figure 4: Number of domains with at least one vulnerable mail server address, binned by rank in (a) the Alexa Top List and (b) the 2-Week MX data set. Domains in the 2-Week MX data set are ranked by the number of DNS queries observed for each domain during the 2-week measurement period.

see, Figure 4a reveals the general trend that high-ranked domains had fewer measurably vulnerable servers than lower-ranked domains, with the bottom 20K domains having nearly twice as many vulnerable servers as the top 20K. Because it represents a much smaller set of domains, less can be inferred of the 2-week measurement shown by Figure 4b. Taken together, both charts show that higher-ranked domains generally had a slightly higher proportion of patched domains compared to lower-ranked domains, but no trends show significantly higher patching (i.e. 40% patch rate or higher) for any particular rank group.

7.5 Top Sites Measured Vulnerable

When determining the impact of a vulnerability in email-related protocols, it is necessary to separately consider the impact on email service providers, as they represent a somewhat unique threat model relative to individual and commercial domains. Whereas the scope of an individual or commercial server being compromised is limited to the smaller subset of individuals operating under that given server's domain, the compromise of an email provider's services could lead to the unauthorized access of millions of user accounts. With most Internet services employing email as a primary option for account password reset, the consequences of compromised personal email are severe.

Our measurements revealed that none of the most prominent email service providers (Gmail, Outlook, iCloud and Yahoo, to name a few) were measured to be susceptible to the aforementioned vulnerabilities. However, several international email services ranked within the Alexa Top 1000 were discovered to be vulnerable, equating to over 100 million user accounts:

- `naver.com`, South Korea's largest search engine and online media provider with around 42 million enrolled users with email accounts [17, 34].
- `mail.ru/vk.com`, Russia's largest Internet company/email provider with 46 million active users [32, 33].
- `wp.pl`, a Polish Internet/email service of over 20 million email users [20, 21].
- `seznam.cz/email.cz`, a prominent Czech news site/email provider with upwards of 1 million active users [24, 25].

Based on an analysis of popular email services done by Foster, et al., in 2015 [6], three of the above services ranked within the top 20 most common email services: `mail.ru` (6th), `naver.com` (12th) and `wp.pl` (20th). When combined, these domains accounted for 1.72% of all email addresses observed in their analysis.

The presence of libSPF2 in major email providers across several countries suggests that the library remains a relevant and important component in global email infrastructure despite no active maintenance in its code repository. This is further supported by some of the other domains measured from the Alexa Top 1000: vulnerabilities were detected in two out of the world's five largest banks, as well as a few well-known Internet media sites with users numbering in the billions. These findings, combined with the findings on vulnerability rate by domain rank (see Section 7.4), serve to underscore the severity of impact of these vulnerabilities in email infrastructure.

Other than one of the Internet media sites, *none* of the above notable websites or email service providers patched their email servers during the four months of measurement; all were measured to still be vulnerable at some point in mid-February 2022.

7.6 Measurements over Time

We collected longitudinal vulnerability measurements to discover patching behavior for domains that were initially found to be vulnerable. These vulnerability measurements were conducted over a four-month period, divided into two windows with a gap in between, beginning one month before we sent out private disclosure notifications on October 11, 2021, and ending about one month after public disclosure on February 14, 2022.

The original set of vulnerable addresses, which we used in our continuing measurements, included 7,212 IP addresses which hosted mail services for 18,660 domains. Figure 5 shows the percent and number of those initially vulnerable domains from which we were able to get conclusive results for each measurement, or for which the measurement was successful. That number fluctuated over time but stabilized somewhat beginning in the last couple of weeks of our first window of measurement.

Because we were not able to get results from every IP address for every trial, we followed a simple set of rules to determine how

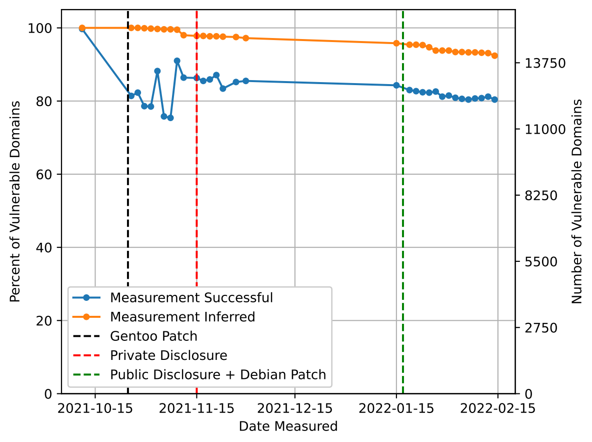


Figure 5: Measurement of conclusive vulnerability results over time, beginning on October 11, 2021, when over 14k domains hosted on 7k unique server addresses were found to be vulnerable. Each dot represents a distinct measurement. Successful measurements fluctuated over time in the beginning, but stabilized in late November. Vertical gaps between successful measurements and inferred measurements are attributed to: 1) domains from which we were only able to collect data intermittently, and 2) domains that we were able to confirm were no longer vulnerable but that we stopped being able to collect data from.

to categorize IP addresses for which we did not receive results in a given data collection trial. These rules were based on the assumption that MTAs are very unlikely to regress after they have been patched. These rules were:

1. IP addresses that are measured vulnerable at some point can be inferred to be vulnerable between that point and the beginning of measurements.
2. IP addresses that are measured patched at some point can be inferred to be patched between that point and the end of measurements.

The domains that we can infer the status of are represented in Figure 5. It should be noted that domains with inferred measurements have an inverse relationship to domains with an inconclusive status. Each change in the total number of inferred measurements can be interpreted as the number of domains that were vulnerable in the prior measurement, but for which we were never again able to get conclusive results. There are a few potential reasons that we were no longer able to get results from these domains:

- Some of the domain's MTAs may have blacklisted email from our server.
- Some MTAs may have moved to another IP address.
- Some MTAs may have gone down and not been repaired.
- Some MTAs may have stopped validating SPF.

A strong majority of these "inconclusive" MTAs still allowed initial TCP connections but would abort the SMTP connection 5XX/421 response code (where they allowed the same transactions

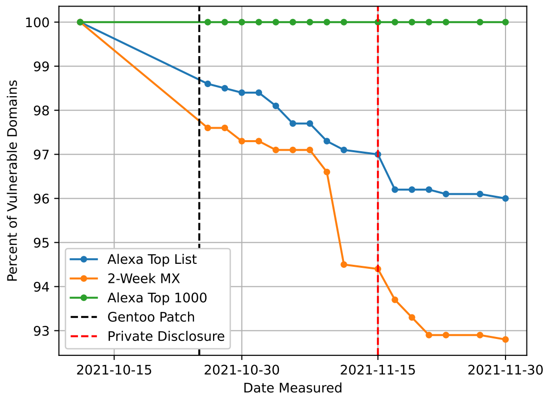


Figure 6: libSPF2 vulnerability rates for each domain list during first window of measurement (window scaled for detail). Each dot represents a distinct measurement taken.

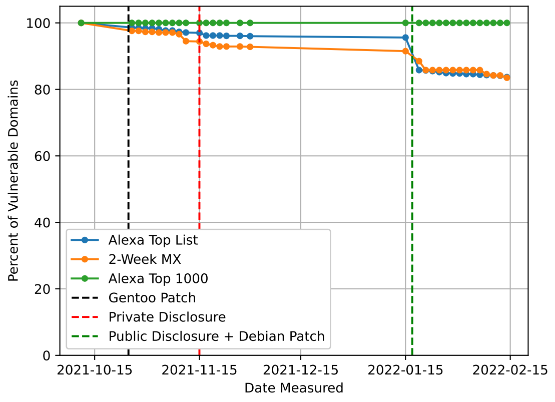


Figure 7: libSPF2 vulnerability rates for each domain list throughout the duration of measurements. Each dot represents a distinct measurement.

in prior tests), suggesting that the blacklisting hypothesis was the most common cause of unsuccessful measurement.

We collected vulnerability measurements in two distinct windows. The first window began in November and ended shortly after our private disclosure. The second began shortly before the public disclosure of the CVE and ended about one month after public disclosure. Figure 6 shows the measured vulnerability of domains in each data set during the first window. Note that only domains for which measurements can be inferred are included in the calculation of the percentages shown. Also note that all significant changes in the measurement inferred, as shown in Figure 5, were carefully inspected for interaction with these percentages and were not found to have a significant impact on the calculated percentages.

During the first window of measurement, approximately 10% of the 2-Week MX domains and about 4% of the Alexa Top List domains began validating SPF either with the safe update of libSPF2 or with a different SPF validation library. Much of this patching took place before our private disclosure notification, suggesting proactive monitoring of package updates. Figure 6 also shows a decrease in vulnerable domains in the measurements immediately following our private disclosure notifications. However, data collected regarding our private notification emails indicates that this decrease in vulnerability is unrelated to our private notifications, which were only minimally effective, as discussed in Section 7.7.

Figure 7 shows vulnerability measurements across the entire four-month measurement period. The public disclosure of the CVE on January 19, 2022, coincided with the libSPF2 package being patched in the Debian Linux distribution. Immediately following disclosure, there was a significant decrease in vulnerable domains, especially in the Alexa Top List data set, but it is not possible to determine what portion of this is due to individual server patching and what portion is due to the Debian patch.

By the end of our four-month measurement period, just over 80% of domains that we were able to infer results for were still vulnerable. This means that around 20% of domains began to validate SPF with either an updated version of libSPF2 or a different SPF validation library within or before one month after the CVE was publicly disclosed.

This data shows a slightly different rate of patching than the final data represented in Figure 2; for instance, a small number of domains in the Alexa 1000 set are designated as "patched" in the final snapshot data, even though the longitudinal measurement does not show any sign of patching. This, to some extent, is because of domains that could not be conclusively measured in the longitudinal measurements (see Figure 5) but that were found to be patched in the final snapshot measurement. These tests resolved updated IP addresses for domains from the DNS and therefore had a higher portion of conclusive test results.

The most evident outlier in the data that we collected was the Alexa Top 1000 data set, which is a subset of the Alexa Top List data set. In our initial measurements, 28 of these 1,000 domains (hosted on 87 servers overall) exhibited vulnerable behavior indicating that they were using the vulnerable version of libSPF2.

We found no indication that any of these domains had patched by the end of our four-month data collection period; however, we had stopped receiving conclusive results for a considerable number of these domains around mid-November, as shown in Figure 8. After resolving updated IP addresses from the DNS, we were later able to get conclusive measurements for almost all of these high-profile domains and detect a handful that had patched implementations; more discussion on these can be found in Section 7.5.

7.7 Response to Private Notification

For each vulnerable domain, a private email was drafted providing details on the vulnerability and information on how to patch or change libraries. These emails were delivered to the `postmaster` address (e.g., `postmaster@example.com`) to maximize the chances of receipt, pursuant to the SMTP specification [13]. Care was taken to minimize the number of emails sent: any domain with multiple vulnerable IP addresses would only have one email delivered, and in the event that multiple vulnerable domains mapped to the same

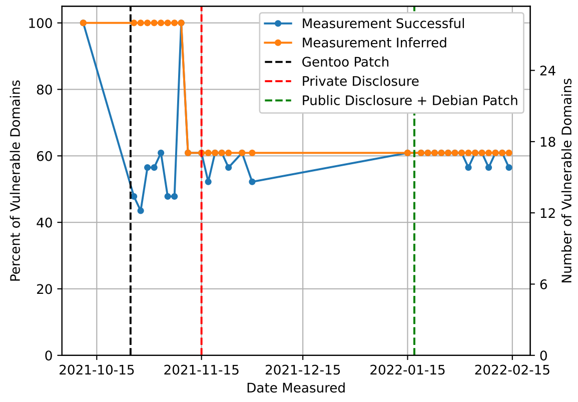


Figure 8: Measurement of conclusive vulnerability results over time for the Alexa Top 1000 data set. Each dot represents an individual measurement, which began on October 11, 2021, when 28 domains were found to be vulnerable.

MX records/IP addresses email would be delivered to the inbox of one of the domains, not all of them. We sent notification emails from different domains and IP addresses than were used in our wide-scale measurements to ensure that email delivery would not be affected by spam filtering.

A total of 6,488 "private notification" emails were sent. 2,054 of these emails (31.6%) were returned as undelivered.

To assess the practical impact of sending out private notification emails, we provisioned a means within the email to measure the number of emails that had been opened by recipients. A link to an image that contained a unique identifier within its URL was embedded in the disclosure email; each time a recipient opened the email and loaded the image, a request would be sent containing that unique identifier to our web servers hosting that image. While our method of tracking is incapable of measuring emails opened in clients that opt not to load images and therefore represents a lower bound estimate, it is nevertheless useful in providing some benchmark for the number of emails that were opened.

The effectiveness of notifications involving HTML emails and tracking images might be questioned, because some email software categorizes such messages as suspicious or because end users refuse to open them. However, there is some precedent for using tracking images in vulnerability notifications [9, 30]. Additionally, Stock, et al., found only marginal differences in patch rates when an HTML email (with or without tracking) was used for notification, instead of a plain-text email [30]. We also note that our email messages included a plain-text version of the notification in the hope that even users of email software that does not render HTML email content might still see our message and take appropriate action.

Out of the 4,434 domains that successfully received a notification email, 512 (12%) opened the email at some time between the private notification and public disclosure. Of those that opened the email, 177 (4%) eventually patched their email servers at some point during the course of measurements, but only 9 (<1%) were patched in between private and public disclosure.

Table 6: Patch Timeline for Package Managers Ordered By Days Between Disclosure and Patch

Package Manager	Time From Disclosure To Patch (Date)	
	CVE-2021-20314	*CVE-2021-33912/13*
Debian	0 (2021-08-11)	0 (2021-01-20)
Alpine	0 (2021-08-11)	50 (2022-03-11)
RedHat	42 (2021-09-22)	0* (2021-09-22)
Gentoo	75 (2021-10-25)	0* (2021-10-25)
Arch Linux	103 (2021-11-22)	0* (2021-11-22)
Ubuntu	230+ (Unpatched)	70+ (Unpatched)
FreeBSD Ports	230+ (Unpatched)	70+ (Unpatched)
NetBSD	230+ (Unpatched)	70+ (Unpatched)
SUSE Hub	230+ (Unpatched)	70+ (Unpatched)

* Patches included in CVE-2021-20314 fix

Of those domains that failed to receive a notification email, 37 (2%) patched after the private notification but before public disclosure. These are most likely the result of package manager patches that were released prior to public disclosure; this is discussed in Section 7.8

These trends, combined with the results from Section 7.6, seem to indicate that private disclosure is relatively ineffective at remediating vulnerabilities, at least when such disclosures are performed at scale. Even in instances where there was an easily discoverable communication channel to communicate the vulnerability to affected parties (such as is the case with our email-related vulnerabilities), the measured response was minimal at best.

7.8 Package Manager Trends

We summarize and note package manager responses to the aforementioned vulnerabilities, as well as to CVE-2021-20314, a remote stack-based buffer overflow attack in libSPF2 discovered by Jeitner et al. [10], in Table 6. The latter is included because several package managers opted to include the fixes we had added to the libSPF2 code repository when addressing the earlier CVE. In addition, the inclusion of the earlier CVE gives a second point of reference to evaluate the timeliness of package manager responses to vulnerabilities.

In most of the package managers we observed, the libSPF2 package was "orphaned"—no specific maintainer was assigned to the package. The absence of any active maintainer in either the code base or the package manager is likely an important contributing factor to why several package managers never ended up patching either vulnerability despite having the package managers having CVE trackers indicating that they were open issues.

The timeline shown in Table 6 can help give us context for patch measurements over time and for response to the vulnerabilities listed, though the time taken to patch in these instances do not necessarily predict general vulnerability response times for the above package managers. Future work may need to be done to explore historical and current trends of patching in package managers for more concrete conclusions to be drawn on this topic.

Table 7: Behaviors in SPF Macro Expansion by IP Address

Macro behavior for %{d1r}	# Affected	% of Measured
RFC-compliant	31,511	76%
Labels not truncated	1,039	2.5%
Labels not reversed	920	2.2%
No truncation or reversal	429	1.0%
Macro not expanded	300	0.7%

7.9 Other observed errors in SPF Validation

As mentioned in Section 7.1, about 6% of conclusively measured servers exhibited SPF macro expansion errors that were distinct from the libSPF2 vulnerability expansion fingerprint. In some cases, the server's SPF implementation would fail to perform any macro expansion, i.e. the server would send a DNS query of the form: `%{d1r}.<id>.<suite>.spf-test.dns-lab.org`. In other servers, macro expansion rules were only partially adhered to, such as by successfully reversing the expanded domain's labels but failing to appropriately truncate them (or vice versa). The relative frequencies of these erroneous expansions are summarized in Table 7.

In some instances, several distinct macro expansions for a given SPF record were detected for a given test of a domain, indicating that the server had multiple SPF implementations being activated when an email was received. In total 2,615 servers, or 6% of all measurable IP addresses, sent DNS queries for SPF validation containing 2 or more distinct macro expansion patterns. This is likely the result of email being passed through multiple SMTP servers using different implementations of SPF or spam filtration services such as SpamAssassin or Rspamd [7, 29]. All in all, the existence of several macro expansion errors calls into question the accuracy of other SPF implementations and suggests other SPF record-parsing libraries may yet have additional undetected vulnerabilities.

8 CONCLUSION

In this paper we have presented our research related to the identification and remote detection of two software vulnerabilities in the libSPF2 library, which is used by many mail servers for email sender validation. We detail the software flaws themselves and measure both vulnerability and patching of affected MTAs across the Internet using two data sets: the Alexa Top List; and domains captured from two weeks of DNS queries related to email traffic at a mid-sized university. From across the data sets, we identified 7,212 (17%) MTAs associated with 18,733 (8.7%) domains. Of those found to be vulnerable, we observed a patch rate of 24% and 13% for MTAs and domains, respectively. We correlated our findings with the results of a private and public disclosure of the vulnerabilities, as well as package manager adoption of the patch. While the private disclosure seemed to make little difference in the patch rate, the public disclosure that followed correlated with a much greater decrease in vulnerable MTAs. Nonetheless, even after the public disclosure, as many as 80% of MTAs found to be vulnerable *remain* vulnerable.

The unique nature of the vulnerabilities allowed them to be remotely detected without causing harm to the affected system. This capability provides additional insights not only into the presence of the software vulnerabilities in Internet systems but also on the impact of notifications—both public and private—and the role of and reliance on package managers for propagating updates. It is our hope that the observations made can help improve vulnerability detection and patching to make Internet infrastructure safer and more reliable.

REFERENCES

[1] Amazon. 2021. Alexa Top Sites. https://aws.amazon.com/alexa-top-sites/ https://aws.amazon.com/alexa-top-sites/.

[2] DB-IP. 2022. IP Geolocation API & Free Address Database. https://db-ip.com/

[3] Casey Deccio, Tarun Yadav, Nathaniel Bennett, Alden Hilton, Michael Howe, Tanner Norton, Jacob Rohde, Eunice Tan, and Bradley Taylor. 2021. Measuring Email Sender Validation in the Wild. In *Proceedings of the 17th International Conference on Emerging Networking EXperiments and Technologies* (Virtual Event, Germany) *(CoNEXT '21)*. Association for Computing Machinery, New York, NY, USA, 230–242. https://doi.org/10.1145/3485983.3494868

[4] Zakir Durumeric, David Adrian, Ariana Mirian, James Kasten, Elie Bursztein, Nicolas Lidzborski, Kurt Thomas, Vijay Eranti, Michael Bailey, and J. Alex Halderman. 2015. Neither Snow Nor Rain Nor MITM...: An Empirical Analysis of Email Delivery Security. In *IMC '15 Internet Measurement Conference*. Association for Computing Machinery.

[5] Zakir Durumeric, Frank Li, James Kasten, Johanna Amann, Jethro Beekman, Mathias Payer, Nicolas Weaver, David Adrian, Vern Paxson, Michael Bailey, and J. Alex Halderman. 2014. The Matter of Heartbleed. In *Proceedings of the 2014 Conference on Internet Measurement Conference* (Vancouver, BC, Canada) *(IMC '14)*. Association for Computing Machinery, New York, NY, USA, 475–488. https://doi.org/10.1145/2663716.2663755

[6] Ian D. Foster, Jon Larson, Max Masich, Alex C. Snoeren, Stefan Savage, and Kirill Levchenko. 2015. Security by Any Other Name: On the Effectiveness of Provider Based Email Security. In *CCS '15 Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. Association for Computing Machinery.

[7] Apache Software Foundation. 2022. SpamAssassin. https://spamassassin.apache.org/

[8] T. Hansen, D. Crocker, and P. Hallam-Baker. 2009. *DomainKeys Identified Mail (DKIM) Service Overview*. RFC 5585. RFC Editor. https://www.rfc-editor.org/rfc/rfc5585.txt

[9] Alden Hilton, Joel Hirschmann, and Casey Deccio. 2022. Beware of IPs in Sheep's Clothing: Measurement and Disclosure of IP Spoofing Vulnerabilities. *IEEE/ACM Transactions on Networking* 30, 4 (2022), 1659–1673. https://doi.org/10.1109/TNET.2022.3149011

[10] Philipp Jeitner and Haya Shulman. 2021. Injection Attacks Reloaded: Tunnelling Malicious Payloads over DNS. In *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, 3165–3182. https://www.usenix.org/conference/usenixsecurity21/presentation/jeitner

[11] Dan Kaminsky. 2008. DNS TXT Record Parsing Bug in LibSPF2. https://dankaminsky.com/dns-txt-record-parsing-bug-in-libspf2/

[12] S. Kitterman. 2014. *Sender Policy Framework (SPF) for Authorizing Use of Domains in Email, Version 1*. RFC 7208. RFC Editor. https://www.rfc-editor.org/rfc/rfc7208.txt

[13] J. Klensin. 2008. *Simple Mail Transfer Protocol*. RFC 5321. RFC Editor. https://www.rfc-editor.org/rfc/rfc5321.txt

[14] M. Kucherawy and E. Zwicky. 2015. *Domain-based Message Authentication, Reporting, and Conformance (DMARC)*. RFC 7489. RFC Editor. https://www.rfc-editor.org/rfc/rfc7489.txt

[15] P. Mockapetris. 1987. *Domain Names - Concepts and Facilities*. RFC 1034. RFC Editor. https://www.rfc-editor.org/rfc/rfc1034.txt

[16] P. Mockapetris. 1987. *Domain Names - Implementation and Specification*. RFC 1035. RFC Editor. https://www.rfc-editor.org/rfc/rfc1035.txt

[17] Naver. 2022. Naver. https://www.naver.com/

[18] NetBlocks. 2022. Ukraine banking and defense platforms knocked out amid heightened tensions with Russia. https://netblocks.org/reports/ukraine-banking-and-defence-platforms-knocked-out-russia-conflict-JBQX7mAo

[19] International Standards Organization. 2018. ISO/IEC 9899:2018: Information technology - Programming languages - C. https://www.iso.org/standard/74528.html

[20] Wirtualna Polska. 2022. Statistics. https://holding.wp.pl/en/statistics

[21] Wirtualna Polska. 2022. Wirtualna Polska - Wszystko co ważne - www.wp.pl. https://www.wp.pl/

[22] Eric Rescorla. 2003. Security Holes . . . Who Cares?. In *12th USENIX Security Symposium (USENIX Security 03)*. USENIX Association, Washington, D.C. https://www.usenix.org/conference/12th-usenix-security-symposium/security-holes-who-cares

[23] Sarah Scheffler, Sean Smith, Yossi Gilad, and Sharon Goldberg. 2018. The Unintended Consequences of Email Spam Prevention. In *Passive and Active Measurement*. Springer International Publishing.

[24] Seznam. 2022. About us - About Seznam. https://o.seznam.cz/en/about-us/

[25] Seznam. 2022. Seznam - najdu tam, co neznám. https://www.seznam.cz/

[26] Kaiwen Shen, Chuhan Wang, Minglei Guo, Xiaofeng Zheng, Chaoyi Lu, Baojun Liu, Yuxuan Zhao, Shuang Hao, Haixin Duan, Qingfeng Pan, and Min Yang. 2021. Weak Links in Authentication Chains: A Large-scale Analysis of Email Sender Spoofing Attacks. In *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, 3201–3217. https://www.usenix.org/conference/usenixsecurity21/presentation/shen-kaiwen

[27] Shevek. 2008. import libspf2 1.2.5. https://github.com/shevek/libspf2/commit/9928ce57b334c2914ccf4def8a8da61b138e4b70

[28] Shevek. 2008. start on length computation in spf_expand. https://github.com/shevek/libspf2/commit/496322ef486935e1cd52af3c09a26300cb465869

[29] Vsevolod Stakhov. 2022. Rspamd. https://www.rspamd.com/

[30] B. Stock, G. Pellegrino, F. Li, M. Backes, and C. Rossow. 2018. Didn't You Hear Me? — Towards More Successful Web Vulnerability Notifications. *Network and Distributed Systems Security (NDSS) Symposium* (2018).

[31] Inc. Synopsys. 2022. About Coverity. https://web.archive.org/web/20220316081445/https://scan.coverity.com/about

[32] VK. 2022. VK / Main. https://vk.company/

[33] VK. 2022. VK / What is VK. https://vk.company/en/company/about/

[34] Wikipedia. 2022. Naver. https://en.wikipedia.org/wiki/Naver