Didn't You Hear Me? — Towards More Successful Web Vulnerability Notifications

Ben Stock*, Giancarlo Pellegrino*‡, Frank Li†, Michael Backes*, and Christian Rossow*
*CISPA, Saarland University ‡Stanford University †UC Berkeley

***Abstract*—After treating the notification of vulnerable parties as mere side-notes in research, the security community has recently put more focus on how to conduct vulnerability disclosure at scale. The first works in this area have shown that while notifications are helpful to a significant fraction of operators, the vast majority of systems remain unpatched. In this paper, we build on these previous works, aiming to understand *why* the effects are not more significant. To that end, we report on a notification experiment targeting more than 24,000 domains, which allowed us to analyze what technical and human aspects are roadblocks to a successful campaign. As part of this experiment, we explored potential alternative notification channels beyond email, including social media and phone. In addition, we conducted an anonymous survey with the notified operators, investigating their perspectives on our notifications. We show the pitfalls of email-based communications, such as the impact of anti-spam filters, the lack of trust by recipients, and the hesitation in fixing vulnerabilities despite awareness. However, our exploration of alternative communication channels did not suggest a more promising medium. Seeing these results, we pinpoint future directions in improving security notifications.**

I. Introduction

Large-scale discovery of vulnerabilities on the Internet has become frequent over the last few years. With the widespread adoption of content management systems such as WordPress, Joomla, or Drupal, vulnerabilities in these programs result in a large number of vulnerable Web sites. In addition, insecure configurations may allow attackers to exploit systems, such as retrieving sensitive data like cryptographic key material [3]. For researchers and attackers alike, detection of such issues is often fairly straightforward: given a seed list of domains or hosts, they can efficiently scan for existing security issues.

Considering the relative ease by which miscreants can identify and exploit vulnerable systems, it is vital to quickly and effectively inform affected parties. For example, the Drupageddon flaw in 2014 [11] allowed attackers to backdoor thousands of hosts within seven hours after the vulnerability had become publicly known. Thus, the security community must focus not only on discovering vulnerabilities, but also on determining effective ways of notifying those vulnerable.

Network and Distributed Systems Security (NDSS) Symposium 2018
18-21 February 2018, San Diego, CA, USA
ISBN 1-1891562-49-5
http://dx.doi.org/10.14722/ndss.2018.23171
www.ndss-symposium.org

How to properly perform such notification is non-trivial, particularly considering the heterogeneity of Web site and system administrators. Two of our recent works [19, 25] independently studied how vulnerability notifications can be conducted at scale, i.e., for a large number of vulnerable Web sites and network services. Our results show that such large-scale notifications have promise in spurring remediation, but currently to an unsatisfactory degree. Both studies identified that a roadblock to success was the *reachability* of correct contacts. By tracking which administrators viewed detailed technical vulnerability reports, the studies also found another challenge: even after viewing such reports, a large fraction of administrators still did not take corrective steps.

We continue this line of research and aim to better understand influencing factors for the success of a large-scale notification campaign. Specifically, we investigate what impact technical measures such as spam filters have on the initial email delivery's success. Based on the insights from prior work, the delivery status remains unknown for large body of sent emails. We therefore investigate whether this unknown state is caused solely by mail servers which mark emails as spam, or if the reason is non-technical, i.e., caused by administrators who do not follow up on the provided information. To achieve this goal, we conducted an automated large-scale notification for two types of Web vulnerabilities using different types and formats of notification emails.

Although emails are a convenient way of disseminating vulnerability information at scale, it is unclear how they compare with other channels, which cannot be as easily automated as email sending. To that end, we investigate the feasibility of using channels that require additional costs (e.g., physical mail), rely on manual acquisition of contacts points (e.g., social media), or even demand high-effort interactions (e.g., calling site owners). We analyze how these alternative disclosure channels compare to a fully-automated approach via email.

Apart from the technical feasibility of conducting notifications, it is also important to understand how the *users* perceive notifications, e.g., whether they discard the messages due to a lack of trust. To understand more about the administrators' perspectives, we conducted an anonymous survey with the site owners we previously notified about security issues, and summarize the most important insights from their feedback.

Since the field of vulnerability notification at scale is still in its infancy, our work not only aims at tackling the previously described research questions, but also identifies areas in which further research should be conducted to improve the quality and success of large-scale vulnerability disclosure.

Contributions and Outline

To summarize, our paper makes the following contributions:

- After presenting our methodology in Section II, we report on the results of an automated notification campaign for more than 24,000 domains, affected by two distinctly different types of vulnerabilities (Section III).
- With the data gathered from the notification campaign, in Section IV we analyze in-depth the technical and human aspects of a notification campaign, and provide insights into parameters for the success of such a campaign.
- In Section V, we report on an experiment involving manual communication channels and how these compare to a fully automated email approach.
- By characterizing the responses to the anonymous survey we distributed to notified operators, in Section VI we discuss the administrators' perspectives on notifications and highlight ways of improving the notification process.
- Finally, based on our study as well as the prior work, we outline directions for promising research to ensure that vulnerability notifications can become more effective in the future (Section VII).

II. Methodology

The goal of our study is to illuminate what factors may influence the success of large-scale notifications on security issues. To accomplish this, we conducted controlled multivariate notification experiments. In this section, we present our experiment methodology. We first describe the security issues we distributed notifications for and then outline our procedure for conducting automated notification campaigns. Following these campaigns, we explored manual notifications along different communication channels to investigate alternative methods to automated email notifications. Finally, we distributed an anonymous survey to notified contacts to gain insight on the characteristics of respondents, their views on our notifications, and shortcomings in our process. For additional data, such as the notification messages and survey we used, please see our GitHub repository [1].

A. Targeted Security Issues

Similar to our prior work [25], we focused on security issues on Web sites that could be easily detected without interfering with normal server operations. Our first two security issues were recent (at the time of the experiments) cross-site scripting (XSS) vulnerabilities in WordPress, CVE-2016-4566 and CVE-2016-4567. Both vulnerabilities impact imported modules and reside in static files, allowing us to detect vulnerable sites by downloading the affected resources and comparing the file hashes to known vulnerable ones. As these issues were fixed in several branches of WordPress, affected webmasters merely needed to update WordPress to the latest version in their currently used branch. We refer to the data set collected for these Web sites as **WP** throughout the paper.

Our other security issue was publicly accessible Git repositories. Since Git is a distributed versioning system, every machine that has checked out a repository has a copy of its complete history. By default, Git stores both configurations and the actual repository data in the `.git` directory. If publicly accessible, an attacker can reconstruct the complete repository and inspect the source code, potentially uncovering hardcoded secrets such as database logins, API credentials, and cryptographic keys. To detect if a site contained such a repository in its public Web directory, we checked for access to the file `.git/config`. Some Web sites may have intentionally exposed a repository as public. To avoid notifying those webmasters, we inspected the `.git/refs/heads/master` file, which contains the hash of the last commit. We searched for the existence of that commit hash on Github, the most widely used Git hosting platform, and only marked a site as vulnerable if no public commit was identified. Although this approach does not eliminate all false positives, we believe that it removes a large fraction of them without requiring us to download the Git repositories and check for sensitive information. We refer to the data set collected for these sites as **Git** throughout the paper.

We initially also identified Web sites with publicly accessible core dump files, which can contain sensitive data dumped from a crashed process's memory. However, in the course of our notification campaign, we discovered that a single hosting provider was responsible for approximately 30% of affected domains. Given the limited data set of 790 domains that were affected by core dumps in total, we excluded this data set as the impact of this single hosting provider was too large to allow for a meaningful analysis.

Finally, to allow for a sound analysis, we only considered domains which were vulnerable to exactly one type of flaw, e.g., domains with WordPress vulnerabilities *and* a publicly accessible Git repository were excluded from our experiments (682 domains in total).

B. Automated Notification Experiments

The first component of our study was an automated email notification campaign. Here, we detail which email contacts we used and what notification messages we sent.

Notification Contacts — Previous notification studies [19, 25] used both direct and indirect (e.g., intermediaries such as CERTs) communication channels. Indirect channels were found to be less effective, as many intermediaries did not forward the provided information. As a result, we chose to only focus on direct channels for Web sites, such as those used in [25]. While the prior studies measured the effectiveness of each channel in isolation, we instead sent notifications for a given domain to both available channels, summarized as follows:

Domain WHOIS Information — The Domain WHOIS database contains a wide array of information about domain names, which can be queried from anyone. However, WHOIS documents are structured for human readability and do not have a consistent format. Furthermore, WHOIS providers may rate limit database queries and can employ CAPTCHAs to prevent data scraping. Instead of parsing the output, we used the WhoisXML API (`whoisxmlapi.com`) for the domains in our data set to extract contact emails in February 2017.

Generic Email Addresses — Generic email addresses are standardized email usernames to be used when contacting personnel of an organization, as defined in RFC 2142 [8]. Amongst these, we selected security or administrator oriented addresses: `security@`, `abuse@`, and `webmaster@`. We

additionally included `info@` in hopes of potentially reaching a front office for the organizations we contacted.

Notification Messages — In our analysis, we aim to understand whether different types of messages impact the success of a notification campaign. Thus, we used six different types of messages, which we describe below.

Plaintext Emails — The first group of messages were three different kinds of plaintext emails.

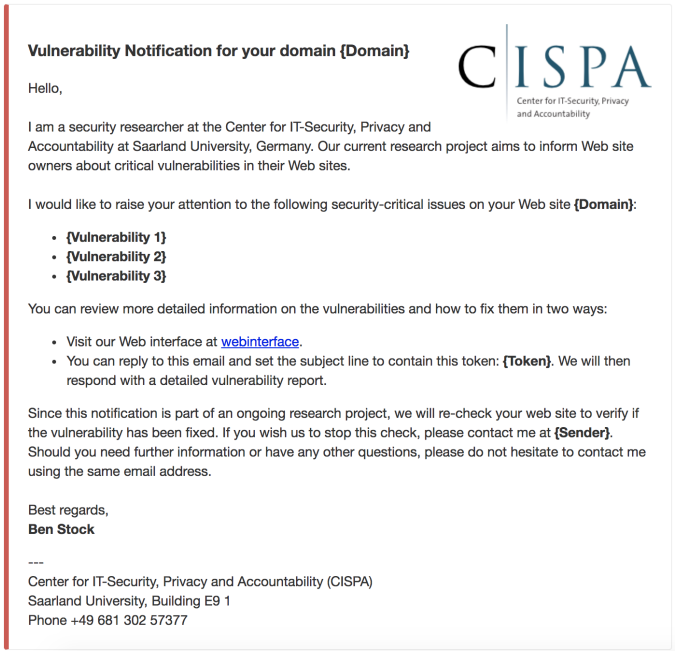
Regular Sender (Plain): The first type of message was a plaintext email according to RFC 2822 [24]. The email sender and the displayed sender name was `Ben Stock <ben.stock@notify.cispa.uni-saarland.de>`. The message contained a list describing the identified security issues and instructions to retrieve a detailed technical vulnerability report (see our GitHub repository [1]). Such reports could be retrieved either with an email reply or on our Web interface (linked to in our email). Access to the report was secured with a secret token. To retrieve the report via email, this token needed to be extracted from the message and put in the subject when replying to our mail bot. Similarly, the token was contained in the URL to our Web interface, allowing us to determine when a report was accessed.

S/MIME Email (S/MIME): Considering the potential sensitivity of vulnerability notification messages, unsigned messages may cause distrust and raise concerns about sender intentions. To determine whether a verifiable sender has any impact on the success of our campaign, we used the same email text as *Plain*, but signed with a valid S/MIME certificate [23]. To ensure that our programmatically generated signature was correct, we verified that the signature was correctly interpreted by the Thunderbird, Apple Mail, and Microsoft Outlook.

Mailbot (Mailbot): The third variant of a plaintext email contained the same messages as *Plain* and *S/MIME*. However, to investigate whether the sender name has any impact, we modified the sender email address and displayed name to be `Vulnerability Notification <notify@...>`. As in *Plain*, we did not sign emails.

HTML-based Emails — The second group of messages were two types of HTML-based emails.

HTML Email with External Resource (Tracking): Previous work [19, 25] observed that a substantial portion of notified contacts who did not take action remained in an unknown state. While no remediation was observed, it was unclear whether the message was not received by an appropriate party, or instead ignored. To better understand whether recipients receive a notification but choose to not act on it, we crafted an HTML email as depicted in Figure 1. The message content itself matched *Plain*, but we added the logo of our institution. Instead of directly attaching the image, we linked to an image hosted on our own server. The image URL was unique for each recipient, allowing us to learn that the email was indeed opened. Note that for each recipient, we only record the single bit of information on whether the image was loaded. We do not log IP addresses or any other potentially personal identifiers. However, if the resource request arrived from a blacklisted set of IP address associated with anti-virus and other services that scan URLs in emails in transit, we do not factor that request when marking a recipient as having read our message.



Vulnerability Notification for your domain {Domain}

CISPA Center for IT-Security, Privacy and Accountability

Hello,

I am a security researcher at the Center for IT-Security, Privacy and Accountability at Saarland University, Germany. Our current research project aims to inform Web site owners about critical vulnerabilities in their Web sites.

I would like to raise your attention to the following security-critical issues on your Web site **{Domain}**:

- **{Vulnerability 1}**
- **{Vulnerability 2}**
- **{Vulnerability 3}**

You can review more detailed information on the vulnerabilities and how to fix them in two ways:

- Visit our Web interface at webinterface.
- You can reply to this email and set the subject line to contain this token: **{Token}**. We will then respond with a detailed vulnerability report.

Since this notification is part of an ongoing research project, we will re-check your web site to verify if the vulnerability has been fixed. If you wish us to stop this check, please contact me at **{Sender}**. Should you need further information or have any other questions, please do not hesitate to contact me using the same email address.

Best regards,
Ben Stock

Center for IT-Security, Privacy and Accountability (CISPA)
Saarland University, Building E9 1
Phone +49 681 302 57377

Fig. 1. Template of the HTML emails

HTML Email (HTML): While the tracking logo allowed us to learn that an email was indeed opened, mail clients that do not automatically retrieve external resources actively warn users about the privacy risks. To understand if these messages would have a negative influence on our campaign, we created another HTML email template with the image as an attachment (hence it does not trigger external resource warnings).

Tone — In the final message group, we varied the message tone compared to the previous message types.

Friendly (Friendly): We selected to establish first contact with a friendlier non-technical message, excluding any links and detailed information about the security issues. Instead, we informed the recipient that we had discovered a security issue, asking them to provide an email address in charge of handling such reports. We also added our institution's default email signature with contact details to allow for recipients to vet us as a sender (see our GitHub repository [1]). When we received the correct point of contact, we manually sent the complete report information, as accessible via our Web interface.

When designing this final message type, we had no expectations about whether it would be perceived better, or if it would raise suspicion in the recipients. However, we consider it important to measure if such a change in tone can make a difference.

Measurements — To compare the effectiveness of the various message types, we collected the following measurements:

Fixed Domains Over Time — We monitored the Web sites detected as affected by a security issue, and determined if and when the site remediated. Our monitoring system checked for the security issue using the same method as used for detection, visiting each site daily. Temporary network and server outages can result in incorrect detection of remediation. Thus, after first detecting a site as fixed on a given date, we continued checking

for the following seven days and consider the domain fixed on that date if it did not appear affected again.

Report Access — Our notification recipients could access a detailed technical report on the security issue by either replying to our email messages or accessing our Web interface (with the secret token). For each domain, we recorded if and when we observed an email response or Web site visit with its associated token. For the *Friendly* group, our initial message did not include detailed information about the security issues or any links, and site operators were to request more information. We marked the report as accessed when we replied with the report (within at most 12 hours after receiving a request).

Email Delivery Status — To determine if message delivery failed to certain recipients, we checked the email inboxes of our senders for email bounces. For our *Tracking* group, we monitored for retrievals of our image resources, unique to each recipient (without logging IP addresses or other identifiers).

Experiment Parameters — On January 30th, 2017, we checked the Alexa Top 1M for the presence of one of the outlined security issues. We then randomly assigned the sites into seven groups as shown in Table I: one for each email variant and an unnotified control group. All groups except for *Friendly* contained approximately 4,000 domains in total. As manual interaction was required for recipients of *Friendly* messages, we limited the number of domains in this group to 1,000 sites. As previously mentioned, we initially also considered core dumps, but removed them during the course of our experiments as the diversity of recipients was too low. Additionally, administrators for 395 domains requested to opt out of any notification or further analysis. Table I shows the final population sizes for each group after those contacts were removed. The notification campaign started on February 3rd, 2017 and lasted six weeks. We sent two reminders on February 17th and on March 3rd.

C. Manual Notification Experiments

Automated email disclosures may be tenable, but may not be the most effective method of notification. Prior work [19, 25] observed that a significant fraction of notified contacts exhibited no responses. To explore alternative notification options, we followed up our email notifications with five different forms of manual notifications, including communication channels that may be challenging to automate delivery for.

Given the manual nature of these efforts, we chose to focus on the Web sites that did not react to our notifications (i.e., those who did not remediate or respond to our messages), as we may uncover a better method of reaching these webmasters. In particular, we reached out to email addresses, phone numbers, and postal addresses listed on affected Web sites, and submitted "contact us" web forms. We also searched the sites for links to their social media accounts on Twitter and Facebook. We describe the method behind our manual notifications in more detail in Section V. We collected the same measurements as with the automated email notifications.

Groups	**Git**	**WP**	Total
Plain	1,561	2,371	3,932
S/MIME	1,559	2,374	3,933
Mailbot	1,560	2,371	3,931
Tracking	1,548	2,370	3,918
HTML	1,565	2,371	3,936
Friendly	367	585	952
Control	1,561	2,373	3,934

TABLE I. SIZE OF OUR NOTIFICATION GROUPS

D. Recipient Survey

Our automated and manual notification experiments allowed us to observe the externally visible effects of our outreach efforts. However, to gain insights on site operator perspectives, we distributed surveys to recipients we had emailed the notifications to. Our survey extends beyond those sent in prior studies [13, 19] to notification recipients, which focused on the acceptability of the notifications. In our survey, we additionally aim to understand the demographics of our recipients, their prior experiences with security notifications and reports, reasons behind the observed remediation behaviors, and suggestions for improvements to our notification process. The details of our survey are described in Section VI.

E. Ethical Considerations

We note that our primary institution does not provide an IRB nor mandate (or enable) approval for such experiments. However, as described next, we took great care to ensure the privacy of message recipients in our experiments.

We designed our detection of security issues to minimize the impact on Web sites. To reduce load on Web sites, we only checked each site once a day, and our detection methods only required requesting a public static resource file, which does not interfere with normal server operations. We respected any opt-out requests and extensively tested our detection methods prior to their deployments.

The ethics of performing security notifications themselves are not fully settled, although a number of prior notification studies [6, 13, 17, 19, 25] have set a precedence for acceptable notifications. In particular, surveys in two prior studies [13, 19] documented the acceptability and helpfulness of these notifications in the eyes of message recipients. We likewise believe that the potential good from informing vulnerable hosts outweighs the potential risks and costs. Following the best practices outlined in these previous notification efforts, we respect requests to opt-out of our notifications. Additionally, we attempted to message all unnotified contacts at the conclusion of our study (such as those in our control groups). We offered a feedback channel through an anonymous survey for the notified organizations, which followed best practices, e.g., it was fully anonymous and optional. We note that we only collected data on organization decisions and not individuals, thus our study does not constitute human subjects research.

Our experiments did rely on the ability to monitor resource requests to our servers, which could potentially be used for tracking and violating the privacy of notification recipients. However, we employed safeguards to ensure that no private information was collected. We did not log IP addresses or any identifiers for any resource requests except for the random token unique to each recipient. This only allowed us to learn if that recipient or domain performed a specific action (visited our web interface or opened our email), and nothing else

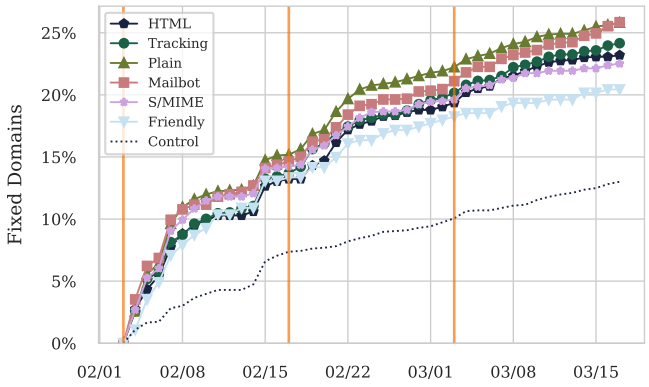


Fig. 2. Non-exploitable **Git** domains over time

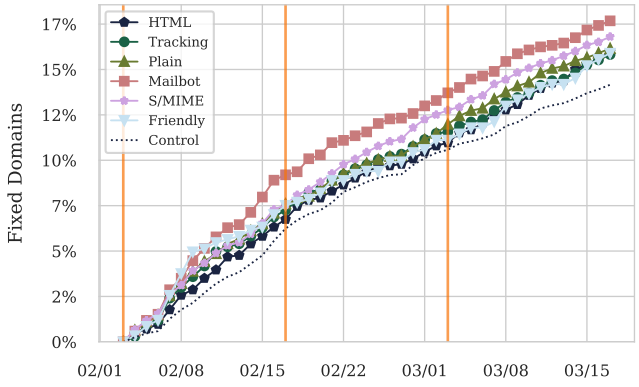


Fig. 3. Non-exploitable **WP** domains over time

about the recipient. We only report aggregate results from this monitoring, without ever revealing the outcome for a particular domain or recipient. We believe the insights we can gain from this monitoring can significantly improve our understanding of notifications at a reduced risk to message recipients.

III. Remediation Behavior

Here we describe the details of our automated email notifications and the observed remediation from affected sites.

A. Remediation Levels

Table II shows the number of remediated Web sites at the end of our campaign (March 17th). For the control group, 13% and 14% of the domains were fixed for **Git** and **WP** respectively. For **Git**, the notified domains had a fix rate of approximately 24%, whereas for **WP**, the average fix rate was merely 17%. The impact of notifications for the **WP** groups may be influenced by WordPress's automatic updates, which was deployed in late 2013 (with WordPress 3.7 [2]). We speculate that those sites that did not automatically or quickly manually update to a new version yet (which implicitly fixed our targeted XSS vulnerabilities) perhaps were less actively maintained (if at all), reducing the likelihood that our outreach efforts would spur actions at those sites.

	Git		**WP**	
Plain	403	25.8%	383	16.2%
S/MIME	351	22.5%	399	16.8%
Mailbot	403	25.8%	420	17.7%
Tracking	374	24.2%	375	15.8%
HTML	363	23.2%	379	16.0%
Friendly	75	20.4%	93	15.9%
Control	203	13.0%	336	14.2%

TABLE II. Non-exploitable domains per group and vulnerability by March 17th, 2017

Timeline of Remediation — Figures 2 and 3[1] show the ratio of fixed vulnerabilities for **Git** and **WP** over the course of our notification campaign. The vertical lines denote the two notification reminders sent.

For **Git**, we note distinct behavioral differences between the control group and the notified groups. After two weeks, an additional 7% of the notified groups had fixed on average, compared to the control group. After four weeks, this improvement rose to 11% of the notified population, indicating our first reminder messages spurred further viewed reports and fixed sites. However, no further improvements occurred after the second reminder message. This behavior contrasts slightly with the re-notification results by Li et al. [19], who found no increased remediation from a second round of messages to network operators. We observe this effect occurred but only after the second set of reports, reinforcing the notion that there is a period after which notifications are no longer effective, but that this period may be longer for webmasters than network operators. This time period is likely due to recipients deciding to not heed our messages, or never receiving our messages in the first place (e.g., bounced emails or spam filters).

Amongst the notified **Git** groups, *Plain* and *Mailbot* remediated at the highest levels, exhibiting commensurate performance. The *HTML* and *Tracking* groups are likewise similar but at a lower fix rate. Interestingly, even though its content exactly matched the other plaintext emails, *S/MIME* performed worse than HTML emails. The least effective notification group was *Friendly*. We discuss implications of our findings in the following section.

WP sites, in contrast, do not display drastic differences between the notified and control groups across our entire measurement window. We observe that the *Mailbot* group performed best, followed by *S/MIME* and *Plain*. Again *HTML* and *Tracking* show a similar fix rate. We hypothesize that this may be due to the population characteristics we discussed earlier, running older versions of WordPress without automatic updates and without actively maintaining it.

Significance of Effects — To determine if our notification efforts had a statistically significant impact on remediation, we compared the fraction of the population remediated after six weeks for each notified group with that of the control group. Our null hypothesis was that the remediation levels for a given notified group did not differ from that of the control group. We use Fisher's exact test to determine the significance of the observed improvements, with a significance threshold of $\alpha = 0.05$. Since we test multiple hypotheses,

[1]This paper contains a fixed variant of the original graphs, which were incorrect, but only noted in 2022 by Maurice Zeuner.

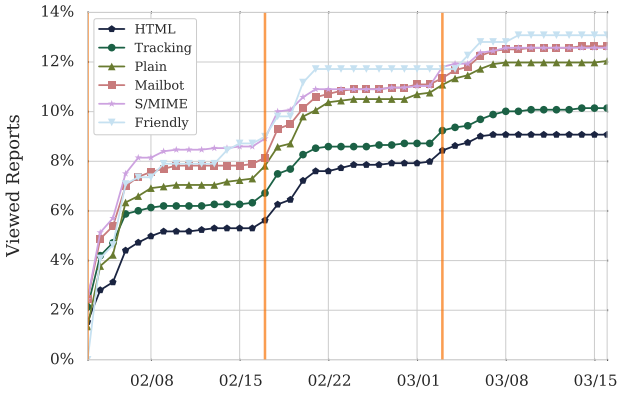


Fig. 4. Viewed reports for **Git**

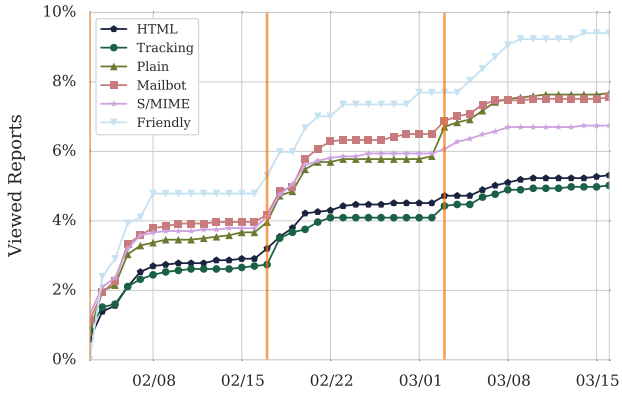


Fig. 5. Viewed reports for **WP**

we must additionally perform multi-test corrections. Thus, we apply the Holm-Bonferroni correction method [16]. Under this correction approach, all m resulting hypothesis p-values are ordered from lowest to highest. Let k be the minimal index in the ordered p-value list such that $p_k > \frac{\alpha}{m+1-k}$, where $m = 6$ and $\alpha = 0.05$. The hypotheses corresponding to the first $k-1$ p-values (in order) are rejected while the rest are not.

Table III summarizes our significance test results. For **Git**, we find that the null hypothesis is rejected for every notification group, under the Holm-Bonferroni correction. Hence, all notification variants show a significant improvement in remediation for **Git** groups. In contrast, for **WP**, only the *Mailbot* remediated at a statistically significantly higher level (again after multi-test corrections). It must be noted though that no group performed significantly better than all others.

	Git	**WP**
Plain	1.159 e-19	0.0570939
S/MIME	3.291 e-12	0.0114718
Mailbot	7.996 e-20	0.0008576
Tracking	9.971 e-16	0.1127060
HTML	1.156 e-13	0.0882801
Friendly	0.0004916	0.2948444

TABLE III. FISHER'S EXACT TEST p VALUES

B. Report Views and Conversion to Fixes Rate

Table IV shows the number (and fraction) of recipients per notification group that accessed our report, and the likelihood that they remediated conditioned on viewing the report.

For **Git**, we observe that the plaintext notification groups exhibited an approximately 12% view rate. For HTML mails, 9-10% of the reports were accessed, indicating that the mail format influenced report viewing (albeit not statistically significantly). Moreover, we find that the usage of a linked resource did not result in differing user behavior when compared to an HTML email with the image attached to it. *Friendly* had the highest overall report access rate with 13%, possibly because the initial message contained little technical information about the security issue and required a response, but also since it did not contain any links (which likely increases the spam score). Figure 4 shows how reports were accessed over time. We do observe a short burst of increased report accesses after our reminders for all notified groups (similar to what we found in prior work [25]), indicating that recipients do not wholesale ignore our reminders. However, as noted earlier, this no longer translated into additional remediation by the second reminder. Across all message types, we find that **Git** Web site operators who viewed our report were likely to remediate correctly, with at least 72% of operators in each group addressing the issue. Thus, *if* we can communicate our report information to a Web site operator, we can likely spur positive actions.

For **WP** (Figure 5), we observe similar patterns in behaviors, with the *Friendly* group accessing the report at the highest rate, followed by the plaintext message groups, and finally the HTML message groups. Likewise, we observed increases in report views after reminders. The overall view rates are lower than with **Git** Web sites, as are the fix rates for those that did view the reports (averaging to 36% of recipients who opened our report). This aligns with our hypothesis that these **WP** sites may be more poorly maintained or significantly out-of-date.

IV. In-Depth Analysis

Potential challenges for an email notification campaign stem from a variety of aspects. On the one hand, emails might not get delivered in the first place, either due to non-existing mailboxes or spam filtering. On the other hand, even if a message is delivered successfully to an inbox, it is unclear whether we reached an appropriate point of contact, whether the message itself is trusted by the recipient, and whether the message conveys the proper information to spur the recipient into taking the corrective steps. In the end, the operator may make an educated decision to not fix the security issue, perhaps to avoid compatibility problems or unplanned downtime. It is important to recognize that such decision may arguably be the correct decision in certain circumstances and environments, depending on different cost-benefit considerations. Thus, 100% remediation is not necessarily the ideal goal for a notification. In this section, we investigate further these technical and human aspects that affect a notification campaign's success.

A. Technical Aspects

Bounced Emails — Unsurprisingly, widespread emailing to site owners results in a number of bounced emails. For all notified Git domains, 1,085/8,160 (13.3%) bounced *all* emails sent

	Git					WP				
Group	Notified	Viewed		Viewed → Fixed		Notified	Viewed		Viewed → Fixed	
Plain	1,561	184	(11.79%)	144	(78.26%)	2,371	182	(7.68%)	60	(32.97%)
S/MIME	1,559	188	(12.06%)	152	(80.85%)	2,374	159	(6.70%)	66	(41.51%)
Mailbot	1,560	191	(12.24%)	145	(75.92%)	2,371	178	(7.51%)	63	(35.39%)
Tracking	1,548	154	(9.95%)	112	(72.73%)	2,370	118	(4.98%)	40	(33.90%)
HTML	1,565	141	(9.01%)	103	(73.05%)	2,371	126	(5.31%)	47	(37.30%)
Friendly	367	48	(13.08%)	37	(77.08%)	585	55	(9.40%)	18	(32.73%)

TABLE IV. NOTIFIED DOMAINS, VIEWED REPORTS (AND %), FIXED AFTER VIEW (VIEWED → FIXED)

to their corresponding addresses; for WordPress 1,906/12,442 (15.3%) bounced all attempts. These bounce rates differ negligibly between different notification groups.

Spam Filtering — An email may also be stopped by a spam filter while in transit to a recipient. To explore the impact of spam filtering, we analyzed how email access rates for *Tracking* differed across mail providers with likely different spam filtering policies. Requests for the image embedded in our message (each to a unique URL) allow us to approximate the number of opened emails. In Table V, we summarize the email access rates for the **Git** *Tracking* group bucketed by the recipient mail provider, grouped by GMail, Microsoft-hosted company mail servers, and all other providers. The results are commensurate for **WP**, and elided for space.

For 477 domains, at least one email was sent to Google's mail servers. On Google's GMail web interface, external resources are retrieved by Google servers by default on email open, except if the sender is deemed suspicious [15]. We observe that the fraction of delivered emails that resulted in image loads is four times higher for other mail providers compared to GMail. Assuming the *inherent* email access levels are similar for recipients using different providers, this significant difference is likely due to either Google spam filtering or blacklisting of our email sender. Based on the timestamps of image loads for recipients using GMail, we believe our email sender was not blacklisted as image loads continued consistently for days after our messages were distributed. Thus Google's content-based spam filter appears to have filtered out a significant fraction of our messages, highlighting the significant impact spam filtering may have on security notifications. While this observation may not be surprising, it is important to note the operational similarities between security notification campaigns and those done for marketing or spam. Future coordination with mail providers may help distinguish security notifications from other less desirable messages.

B. Human Aspects

Reasons for not Fixing Issues — As previously outlined, only about one third of the operators that saw a report for their WordPress installation actually fixed the problem afterward. To better understand the reason behind this, we contacted these operators in June, excluding those sites which did eventually end up fixed. In contrast to our survey (see Section VI), instead of providing an anonymous questionnaire, we reminded operators about the unfixed vulnerabilities and asked if there was anything we could do to help. In total, we contacted 360 domain operators this way. We only received a total of 15 replies, but these nevertheless provided some insights. Six domain operators said that they did not think it was necessary to address the vulnerabilities, e.g., because they were planning to move away from WordPress altogether. Three answers indicated that the contact was not even aware of our initial message, even though the report had been accessed before. This is most likely due to larger teams handling such issues, or churn in a site's administrators. The remaining reasons given were either that the recipient was not responsible for the content (e.g., only registered the domain), upgrading would have threatened compatibility with plugins, or that the flaw had been addressed before but re-appeared (possibly through misconfiguration of the web server). Moreover, one recipient told us that he simply had forgotten about the problem and another answered that our information was not trusted, although they did follow the link to our web interface. As we report in Section VI, our more methodical survey experiment conveyed similar sentiments, particular on trust.

Provider	Delivered	Tracked	Fraction
Google	477	49	10.2%
Microsoft	103	46	44.6%
Others	1,049	392	37.4%

TABLE V. TRACKING ANALYSIS PER PROVIDER FOR GIT

Correlating Update Frequency With Remediation — WordPress by default features an RSS feed, which is typically accessible at `domain.com/feed/` and shows recent site changes such as new articles. We developed a small crawler which allowed to us to retrieve this RSS feed, which features the `lastBuildDate` field describing when the site's content was last updated. In total, we could gather this data for 660 domains in **WP** for which a report was accessed (for the rest, the feed was disabled or the site was offline when we checked). Out of these, 223 fixed the issue after having seen the report, while 427 did not. For each domain, we then extracted the number of days since the last content update relative to June 1st, 2017. Subsequently, we calculated the average time for those domains that fixed and those that did not.

We found that sites which remediated after accessing a report updated an average of 60 days priors, whereas vulnerable domains averaged 115 days since their last update. This in line with our hypothesis that webmasters of vulnerable **WP** sites who do not react skew towards those that less actively maintain their site or run such an outdated WordPress version that updating is rife with compatibility issues.

We conducted the same analysis for the control group, finding minimal differences in the time since the last update between those who ended up fixing versus those who did not (119 days for fixed domains, 109 days for still vulnerable domains). These values are similar to the domains who did not remediate

	Git			WP		
	Read	Fix	% Fix	Read	Fix	% Fix
Email Read	490	183	37.35%	642	113	17.60%
and Report Not Viewed	361	87	24.10%	546	81	14.84%
and Report Viewed	129	96	74.42%	96	32	33.33%

TABLE VI. FIX RATES OF VIEWED EMAILS IN *Tracking*

	Git		WP	
Alias	Read (*Viewed*)	Fraction R.	Read (*Viewed*)	Fraction R.
info	134 *(16)*	8.7%	189 *(18)*	8.0%
abuse	79 *(23)*	5.1%	80 *(6)*	3.4%
webmaster	87 *(23)*	5.6%	76 *(9)*	3.2%
security	26 *(3)*	1.7%	36 *(7)*	1.5%
WHOIS	254 *(42)*	16.4%	302 *(38)*	12.7%

TABLE VII. READ EMAILS AND VIEWED REPORTS THEREAFTER, GROUPED BY GENERIC ALIASES/WHOIS CONTACT

after viewing a report, suggesting similar populations.

C. Tracking Analysis

The purpose of our *Tracking* group was primarily to understand the conversion rate of email access compared to report views. Table VI summarizes the remediation behavior conditioned on email and report views. The top row *Email Read* shows the total number of domains for which we verified that the email was read. The two other rows distinguish between domains that did and did not view the report, respectively.

From Reading an Email to Viewing a Report — For **Git**, from a total of 1,548 *Tracking* notified domains, our email was opened by (at least) 490 domain operators. Note that this email reading rate of 32% represents a lower bound, as recipients might have also read the email *without* loading the external content in the email. Of these verified readers, only 129 recipients viewed our online report or requested it via email. This yields a report access rate of 26%. Apart from the 129 accesses after the hit on our tracking image, an additional 25 reports were accessed without retrieving the image. For **WP**, out of 2,370 *Tracking* domains, 642 domains had at least one email read (email reading rate of 27%), leading to 96 viewed reports (report access rate of 15%); here, we observed an additional 22 viewed reports without the tracked image.

These findings are sobering, as they show that *even if* an email was read, only 15–26% would investigate further by viewing the report. Thus, even after overcoming the technical challenges of successfully delivering a message to a recipient's inbox, there are human factors such as lack of trust or misunderstanding the information conveyed that reduce remediation rates (although again, some recipients may have made an informed decision towards inaction). Judging from the results that our previous work had with including the whole message in the body instead of a link [19, 25], we believe that the external link only played a minor role in these results.

Fix Rates — Considering only those domains for which a report was viewed (and the email was opened), we observe that 74.42% and 33.33% of the issues were addressed for **Git** and **WP**, respectively. This is in line with the results for all other groups. The fix rate for **WP** domains which did receive our email, but did not open the linked report, are in line with the results for the control group (see Table II). Notably, the outcome is different for the **Git** domains who opened the email but did not view the report: 24.7% of these domains remediated. This is significantly higher than the 13.0% fix ratio exhibited by the **Git** control group, which may suggest that recipients already had enough information to handle the case even without accessing our detailed reports.

The difference in the fix rates may be due to the varying nature of the security issues. While the Git version control system requires a certain level of technical expertise to use, WordPress can be installed by a non-technical user. Thus, site operators for **Git** domains may skew towards a better understanding of the technical details about the reported issue compared to **WP** webmasters. This may also explain the higher **Git** fix rate even when viewing only the email. Additionally, our hypothesis that the vulnerable **WP** sites are less actively maintained and run outdated software may also be a contributing factor.

Usefulness of Generic Alias Addresses — RFC-specified generic email aliases (such as `security@`) may not be consistently available. To evaluate their effectiveness, we compare the fraction of address recipients that accessed our emails and our reports for WHOIS contacts versus generic aliases.

Our findings for email accesses are shown in Table VII. We note that the total number of viewed reports from generic aliases exceeds group sizes in Table VI, because a site might access the report from messages received at multiple aliases. Notably, the least effective alias appeared to be the `security` alias, whereas `info` was most effective. This can be explained in part by the fact that for the first round of notifications, 85% of the emails to `security` bounced, while for `info`, the bounce rate was only at 62%. Nevertheless, this highlights that a dedicated security email address is hard to reach. Moreover, it seems like the WHOIS contact information is more useful, allowing us to deliver the initial notification to 12.7% and 16.4% of the **WP** and **Git** sites, respectively.

The table also shows the number of domains for which a report was accessed *after* the email was opened. While the overall numbers are generally low, it is worth noting that for **Git**, `abuse` and `webmaster` had the highest number of viewed reports, even though they did not have the highest number of emails opened. Hence, while it appears that these aliases are not as easy to reach as `info`, recipients of the technical email addresses might have a better understanding of the issue described, prompting them to view the report.

D. Parameters for the Success of a Notification Campaign

The dimensions of potential biases introduced by tracking users are unclear. One could argue that security-aware recipients would have external resource retrieval disabled in their mail client. Yet we discovered that several mail clients, such as Apple Mail and GMail, enable such external resources by default. Thus, under the assumption that the results we gathered at least *approximate* the whole population of vulnerable sites we notified, the analysis shows that the success of a notification campaign depends largely on three key factors:

1) The **email reading rate** heavily depends on how many emails are successfully delivered to the inbox of the recipient and sets the basis for the campaign's success.

2) The **awareness raising factor** of the campaign sets the baseline for people being aware of the security problem, largely influenced by the recipient's trust in the email and, consequently, if she considers the reported vulnerabilities as serious (e.g., by viewing the detailed report).
3) The **aware-to-fix rate**, which was high (above 75%) for **Git**, can be influenced by the type of report and type of vulnerability and represents the chance that an issue is fixed after the report was viewed (or the target became aware of the issue implicitly by seeing our email).

Regarding the email reading rate, our observations show that it is independent of the type of vulnerability. We found the bounce rates and tracking rates are comparable between **Git** and **WP**. We note however that Stock et al. [25] also considered Client-Side XSS issues focusing on highly-popular sites (i.e., the Alexa Top 10,000). For them, the bounce rates varied between WordPress (Alexa Top 1m) and Client-Side XSS bugs (Alexa Top 10k). We believe that the email reading rate is independent of the type of vulnerability, but is influenced by the popularity of a Web site. Moreover, we showed that Google Mail in particular exhibited a much lower tracking rate than other mail servers. This is likely caused by more restrictive filtering rules employed by Google, especially compared to Microsoft mail services. At the same time, of the 103,189 emails we sent for our first round of notifications, 21,380 (21%) were delivered to Google. Considering only those emails which did not bounce due to a non-existing mailbox, 12,336/40,002 (31%) of successful deliveries went to a Google mail exchange. Hence, if our messages were flagged as possible spam by Google, this single service had a severe impact on the success of our campaign. To solve such an issue, large-scale notification campaigns can aim to collaborate with larger email providers to achieve a higher coverage of mail servers flagging the emails as benign.

The second parameter, i.e., the ability to raise awareness of a vulnerability issue, is hard to measure. One way of doing so is looking at the likelihood of a recipient actually viewing the report. However, from our experience, this view rate depends on the type of vulnerability (and implicitly also the type of administrator). The fraction of operators who opened our emails and subsequently visited the contained link varied from 15% to 26%. A possible explanation for the difference is the audience of the notifications: for **Git**, the expected tech-savviness is higher than for an operator of a **WP** domain. In fact, although it contained important technical information on how to fix the vulnerability, only about one in six recipients who opened an email in our **WP** data set also visited the Web site. Second, after reading our initial notification email, it might not be necessary to fetch further information by reading a report, which also influences the view rate.

Finally, the aware-to-fix rate is significantly lower than the optimum rate of near 100%. We showed (similar to previous work [19, 25]) that even viewing the detailed report about a security issue does not always result in a fixed site. Especially for WordPress, fix rates of sites that viewed the report were around 40%, showing that 60% of the sites do not fix the issue even when aware of it. For **Git**, the fix rate was much higher at around 80%. It appears that there are two major factors for the low fix rate on WordPress. First, notified site operators might not have been able to understand the implications of the disclosed vulnerabilities. Arguably, researchers should try to better understand what level of detail is needed for the expected type of recipient. Second, they might not have considered these a risk for their sites. This is partially underlined by the responses we received to the emails we sent to operators who had not fixed their sites despite viewing a report. Six out of 15 respondents stated that they were not going to address the vulnerability due to the limited risk they associated with the issues. Hence, it is reasonable to assume that the perceived risk associated with the type of disclosed issue influences the fraction of fixed sites and our results should be re-evaluated with other types of vulnerabilities.

V. Notifying Manually Acquired Contacts

So far we used contact points which could be automatically deduced (e.g., generic aliases) or retrieved from a database (e.g., WHOIS data). Furthermore, with email, we used a fully-automated communication channel. However, following this procedure did not perform in a satisfactory manner when considering the fraction of contacts that were successfully reached. In this section, we study how alternative communication channels can help to improve this situation.

A. Experiment Parameters

As outlined in the following, the alternative communication channels we chose require significant manual effort to collect contact points or to perform the notification. We therefore randomly sampled 1,000 vulnerable domains which were in an *unknown* state by April 10th, 2017, i.e., had not accessed a report, had not retrieved the external resource linked in our email and had not bounced all emails. We selected these domains before excluding core dumps from our experiments, which slightly reduces the final set to 970 domains.

B. Manual Communication Channels

In the following, we briefly outline each of the manual communication channels we selected. To find such a channel, we manually analyzed the vulnerable Web sites to extract as many of the available contact points as possible. In total, this task required about 40 person hours to complete.

Postal — The first manual channel is postal snail mail. We, therefore, searched either for a contact page or an imprint to retrieve the postal address associated with a Web site. In case more than one address could be found (e.g., because a company had multiple offices), we prioritized locations in English-speaking countries. For the letters, we used the address format of the destination country, most importantly also the character set (e.g., for China or Russia). Since the recipient had to manually type the URL to the vulnerability report, we set up shortened URLs with site-unique six-character tokens. Apart from this shortened URL, we used the same template as for the automated disclosure via email. For all manuals channels described in the following, we also used the shortened URLs.

Email — Another manual channel was again email, but this time contacting email addresses contained on the Web site that had to be searched manually. Whenever we discovered more than one email address on a site, we used the one which most closely resembled an administrator. In many cases, however, we had to resort to a general `contact@` alias. Once an email

address was retrieved, we followed the exact procedure of the *Plain* group to send an email.

Web forms — Some Web sites offer HTML-based forms to get into contact. In such cases, we either used general purpose forms or if the form was guided (i.e., we had to select a reason for our contact request) we used the topic which most closely resembled security. Whenever possible, we provided our full names, affiliation, postal address, phone number, and email to allow for the notified party to get back to us.

Social Media — We tried to find the social media accounts on Facebook and Twitter that belong to the Web site. While on Facebook, directly messaging a person or company is possible, Twitter does not allow sending direct messages to users that are not following the sender. Therefore, on Twitter we send an @ message to the account, asking them to contact us via email so that we could disclose the vulnerability. In both cases, instead of setting up a new account without any friends or followers, we used two of the authors' well-established real accounts to convey trust in the sender.

Phone Numbers — Finally, we chose to call vulnerable sites to disclose the issues. Similar to postal mail, we selected the phone number which was most likely to reach an English-speaking person (e.g., a US or UK-based office). We then determined the best time to call the remote party based on their timezone. When calling, we first established whether the person on the other end understood English. If not, we asked to speak to somebody who could, or ended the call if no such person was available. Once we reached someone who could speak English, we told them our affiliation and that we had discovered a vulnerability on their site. We inquired whether they would be the right person to speak to about this and if not, asked to be forwarded. If the correct person was not available via phone, we asked them to either give us their email address or to send us an email. Unless specifically asked, we did not disclose any more information on the actual vulnerability over the phone but rather sent the report via email.

C. Availability of Manual Channels and Group Assignment

Table VIII shows the fraction of domains for which we could find any of the aforementioned contact channels, split for Git and WordPress. This table reveals a notable insight: approximately half of all the domains do not provide an email address. Previous work on notification has, however, always focused on using emails as the main channel to reach administrators. Yet, it appears that at least for the snapshot of our data used for this experiment, site admins prefer not to be contacted via email.

Apart from this, we observe that both types of considered Web sites, i.e., those using Git and those run on WordPress, provide at least one communication channel in about 90% of the cases. Hence, it arguably is possible to retrieve a contact point for almost any domain in need of a vulnerability notification. However, looking up such information at scale is not feasible. Moreover, considering that calling sites or sending them postal mail incurs costs, we will discuss if the benefit can outweigh these costs in the following.

To avoid any bias in the selection of a channel, we randomly assigned each domain in the data set to any of the aforementioned channels. After this, we determined whether a given domain could be contacted via that channel, i.e., whether there was such a contact point for it. If it could not be contacted, we removed that domain from any further consideration. Hence, of the 970 domains, only 364 could be contacted manually. The number of domains for each channel are shown in Table IX in the *Cont.* columns.

Channel	Git Config	WordPress
Postal	38.4%	32.0%
Email	57.1%	45.6%
Web forms	36.1%	39.2%
Social Media	52.8%	46.1%
Phone	43.4%	33.6%
At least one	89.9%	89.3%

TABLE VIII. AVAILABILITY OF COMM. CHANNELS CONTACTS

D. Success of Our Manual Notification

After the assignment of the groups, we started our manual notification on April 11th with Social Media, Web forms, and target-specific email. We received no bounces for any email we sent. The next day we called the sites in the phone group. The same evening, we prepared the letters, which were sent out on April 13th. We ended our measurements on May 23rd.

The overall results of our manual notification are depicted in Table IX, showing the number of contacted domains, reports accessed, and fixed domains (after having accessed a report) for Git and WordPress, respectively. Notably, the email addresses collected from the Web sites did not appear to be useful for notifications. Given that we contacted only such domains for which we had previously no information whether they received our message in the first place, two reasons might have come into play. First, our email triggered the same technical measures as before (e.g., spam filters). Second, our email was already received the first time we sent it, but disregarded as untrustworthy. Regardless of the reason, even using manually-vetted addresses did not increase the effectiveness of emails.

For Git, the most successful channels were postal and phone, leading to 23% and 25% accessed reports, respectively. At the same time, these channels are the most expensive ones. For WordPress, we find that Social Media performed best, followed by the phone channel. However, the generally low number of reactions do not allow for a meaningful comparison to the automated notification. Nevertheless, we observe that the respective best channels appeared to reach more administrators than our original email campaign.

	Git Config			WordPress		
Channel	Cont.	Reports	Fixed	Cont.	Reports	Fixed
Postal	30	7 (*23.3%*)	5	37	3 (*8.1%*)	1
Email	48	0 (*0.0%*)	0	43	0 (*0.0%*)	0
Web forms	29	6 (*20.7%*)	4	40	2 (*5.0%*)	1
Social Media	36	5 (*16.7%*)	1	55	7 (*12.7%*)	3
Phone	20	5 (*25.0%*)	2	26	3 (*11.5%*)	2
Total	163	23 (*14.1%*)	12	201	15 (*7.5%*)	7

TABLE IX. RESULTS OF MANUAL NOTIFICATION

E. Discussion

The goal of this experiment was to determine if alternative channels could be used to improve vulnerability notifications. For 90% of the domains, at least one such channel could be established. Seeing the effort of manual notifications, they yield a comparatively low success rate. Using the best available channels, i.e., postal and phone for Git as well as social media and phone for WordPress, we managed to reach a slightly higher fraction of reports compared to our regular notification campaign. However, it remains unclear how to determine the best alternative channel *in beforehand*.

At the same time, we must consider the cost of this notification. Apart from approximately 60 hours of work to conduct 364 notifications manually (approx. 40 hours to look up contacts for all 970 domains, 10 hours for the phone calls, 5 hours for letters and 5 hours for social network and Web forms), we spent both postage and telephone costs. Moreover, of the 67 letters we sent, 18 were returned to us because a non-existing recipient. We thus argue that such a manual notification can by no means be considered cost-efficient when trying to notify an even larger number of sites.

VI. Survey

Following our automated and manual notifications, we sent anonymous surveys (see our GitHub repository [1]) to all notification recipients who did not opt-out of our experiments, as described in Section II-D. Here, we analyze the responses and distill the insights they provide.

A. Summary of Survey Submissions

We distributed a distinct survey to each notification response group (which were based on their interactions with our notifications and their remediation status). This allowed us to track the anonymous survey answers per response group. The sets of questions for each survey were identical, except for an additional question for response groups that did not fix the vulnerability, asking for the reasons they did not address the issue. Note all survey questions were optional, typically resulting in fewer question answers than submitted surveys.

Across our six response groups, we received a total of 193 submitted surveys, and an additional 232 initiated but not submitted surveys. However, these submissions were primarily from two response groups. Notification recipients who viewed our vulnerability report Web site and also fixed a vulnerability (a group we label as REPORT_FIXED) contributed 162 submissions (and an additional 150 other views). The high submission rate for this group is unsurprising given they were most likely to have observed and responded to our prior notification efforts. More unexpected was that the response group with the second most survey answers was those who did not appear to view our email and did not address the issues (which we label as NOREAD_NOFIX). They completed 27 surveys (with 59 incomplete ones). This indicates that a number of these email contacts were active, despite the fact we could not *verify* they had read the email.

The remaining response groups provided two or fewer submissions, with 12 or fewer total views. These are surprisingly low reply rates, as some response groups include those we observed as having viewed our report Web site or opened the notification message, indicating the email addresses were at least active. Even though we randomly distributed the order of the sent out emails and regularly checked for the presence of our mail server in well-known IP blacklists, our messages containing a link to the survey page might have been detected as spam by mail servers. Since we conducted our survey in an anonymous fashion, it is impossible to determine whether specific mail servers were more likely to flag our mail as spam. However, we assume that the impact of this is similar to what we had observed for the notifications in the *Tracking* group, i.e., that likely Google's mail servers might have flagged more of our messages as spam. Given this distribution of survey submissions, we cannot assume our findings generalize to all contacts, but they can provide insights nonetheless.

B. Respondent Demographics

We asked respondents to self-describe the type of Web site their organization maintains and the nature of their organization. The most popular response was that the Web site was a corporate or business homepage. Over a third of contacts (66 out of 162) in the REPORT_FIXED group self-reported this answer, as did 6 out of 27 (22%) of those in the NOREAD_NOFIX group. This indicates that exploitation of our Web site vulnerabilities may have financial implications, potentially compromising businesses or harming their customers. Thus, our reported vulnerabilities can have a noteworthy impact. Other notable Web site categories included personal or blog pages, e-commerce Web sites, community forums, online services, and academic institution homepages, although none of these categories accounted for more than 7% of respondents.

C. Security Response Manpower

The availability of security personnel to address Web vulnerabilities impacts the effectiveness of security notifications. For example, Web sites with only one webmaster tasked with responding to vulnerability reports may struggle to properly address all issues. We asked respondents to disclose the number of personnel tasked with responding to security incidents or security notifications, as well as the number of people directly responsible for administrating or maintaining the Web site.

The survey responses indicate a Web site typically has few administrators who handle security issues, less so than the total number of webmasters. Out of the 173 answers we received regarding personnel, 92 Web sites (53%) indicated there was only one security administrator, 66 (38%) indicated there were two to five such webmasters, and 9 (5%) indicate no one was tasked with handling security incidents. Only 6 responses indicate there were five or more security personnel. In comparison, 101 Web sites (58%) had two to five webmasters, 56 (32%) had a single Web site maintainer, and two responses indicated no one managed the Web site. 14 (8%) indicated more than five people helped run the Web site.

D. Acceptability of Detection and Notification

It is vital to understand whether our vulnerability scans and subsequent notifications are acceptable to those we contact. A

negative answer would argue against the practice of notification. To assess this, we asked respondents to indicate whether they found it acceptable to detect security issues on their Web site and whether it was acceptable for us to notify them about any detected problems. We additionally asked if they would find such notifications useful in the future. Naturally, this carries a sampling bias, given that recipients who found our message helpful would be more willing to take our survey.

Our responses were almost universally positive. Over 98% of respondents (178 out of 181) indicated detection was allowed, and only 1 out of 185 survey answers (0.5%) indicated security notifications were unacceptable. These notifications were not only acceptable, they were deemed helpful, as 99% (181 out of 183) surveys showed they would find our notifications useful in the future. Our survey responses may suffer from response bias, such that those who feel strongly positive or negative are more likely to complete our survey. However, the nearly universally positive responses suggest that these notifications are well-received. In addition, we received numerous thanks in the survey's free response comments. This conclusion is in line with prior security notification studies [13, 19], which have also reported positive interactions with the vast majority of notification recipients (through surveys and notification follow-on communication).

E. Proper Points of Contact

Our notification efforts have illuminated the challenges in even establishing contact with the maintainers of a site. To gain insight on the proper points of contact, we inquired as to the best point(s) of contact for each respondent. We observe a wide range of responses, although the WHOIS technical email contact and site-listed emails appeared to be the predominantly preferred contacts. 123 respondents marked the WHOIS technical contact as appropriate, and 115 did so for email contacts listed on the site. Other popular options included the WHOIS abuse contact (59 replies) and Web forms (55). For the remaining options we listed, including social media, as well as snail mail addresses and phone numbers (both in WHOIS records and on the site), between 15 and 25 respondents designated those as reasonable contact points.

This range of responses highlights the difficulty in reliably reaching the appropriate webmaster, as different Web sites may be best notified at different points of contact. It also provides guidance towards future notification efforts, such as utilizing emails scraped from Web sites or automating Web site form submissions. Also of note is the popularity of the WHOIS technical email compared to the WHOIS abuse contact. The difference is drastic, even when considering the possibility that some respondents may have mistaken (or did not distinguish) between the two. This counters the intuition used in prior notification studies [6, 13, 19, 25, 26] which have utilized the abuse contact, considering it more security-relevant.

F. Trust as a Factor

Our remaining survey questions attempted to assess the qualities and characteristics of our notification messages. While there was a diverse set of feedback, a common theme emerged of trust in the notification. A number of notification recipients did not immediately react or dismissed our message as initially untrustworthy, and those who eventually did heed our notification indicated it was after some investigation to confirm its validity. Such negative perception of a vulnerability report naturally can significantly limit its effectiveness.

There were three common characteristics of our notification that were most frequently discussed as impacting its trustworthiness. First was the message sender, which 14 survey submissions explicitly stated was either unfamiliar or lacked an established trustworthy reputation. One respondent said "As an American, Saarland University was not a school I had heard of before." Others stated that the notifications could be improved if we were to "build up a reputation to make them seem less dodgy and more trustworthy," or "use a trusted brand to rely on." Several of those that noted they did not recognize the sender also discussed needing to investigate to confirm the legitimacy of the notification source. For example, one respondent recalled that they "had to google you to know it was not phishing" while another suggested providing "instructions on how to verify the authenticity of the sender." These replies indicate that the sender may impact a notification's effectiveness, a conclusion that conflicts with the study from Cetin et al. [6], which experimented with several different notification senders and found that sender reputation did not appear to affect remediation rates. Further investigation into this factor is needed, as our survey responses suggest that trust in the sender is an important consideration.

The second characteristic of our notification that survey recipients noted was the use of an external link. Email links, particularly from unknown senders, can appear suspicious as they can lead to phishing or malware domains. As our notification's report link was hosted at a university's domain, those who were unfamiliar with the institution or its domain could be wary of clicking on it. One survey submission advised "Don't try to link me to an external site. This makes me immediately think it may be a phishing attempt." Another expressed similar sentiments: "I wouldn't have acted if there wasn't something verifiable inside of the e-mail, or if it required following anything from the e-mail or accessing any attachments." These replies suggest that a notification message should encompass the entire vulnerability report. While perhaps external links to information pages (or sites that validate the sender) could still be provided, recipients should not have to visit these to obtain the necessary information. This corresponds with the findings of Li et al. [19], who found that more detailed notification messages resulted in improved remediation behavior.

A final quality widely mentioned was the similarity to phishing or spam emails. In fact, 10 notification recipients indicated they found our notification originally in their spam folders, which likely grossly under-represents the true fraction as those who never observed our initial notification were unlikely to respond to our survey (or perhaps our survey message itself was relegated to spam). Numerous comments touched on their suspicions, such as:

- "At first this email was considered rather suspicious (spam / phishing / malware). We checked it with extreme caution."
- "I ignored first one as spam/phishing but reacted to the friendly reminder followup."
- "My initial instinct was your email was spam and/or some kind of attack itself, but I quickly realized it was real."

- "Risk to read it as spam due to lots of false security mails from no-trust-senders."

Also, several replies indicated they had poor past experiences with some less benevolent parties claiming to have relevant security information. One person recounted "I have been contacted in the past by bug bounty hunters who hounded me for payment for vulnerabilities they found in another site I run." Another explained "Lots and lots of emails I receive claim to be attempts to alert me about something, and it is usually a veiled sales pitch for something like SEO services." This illuminates a challenge for these administrators in filtering out legitimate notifications.

From this, it is apparent that the appearance of the notification message is also a vital consideration, as it needs to distinguish itself from spam and phishing messages. One potential solution was proposed by five responders, who suggested using more professionally designed or HTML-based messages. Another might be to switch to a subscription model, where webmasters subscribe to vulnerability notifications. This suggestion was proposed in seven comments, which explained that such a model would result in increased trust in both the sender and the received messages. Overall, future work on notifications should investigate not just the method of conducting notifications, but the design of the notification message itself, with an emphasis on establishing trust.

VII. Quo vadis, vulnerability notifications?

While the majority of domains we notified about remained unfixed, a significant fraction did correct the issues, showing the promise of notifications. This aligns with the results from Stock et al. [25] and Li et al. [19]. Moreover, our survey responses indicated strongly positive reception to our efforts. The research into large-scale security notifications is still in its infancy though. In this section, we outline challenges we identified from our study as well as previous work, laying a roadmap for what future work could focus on.

A. Better Delivery Mediums

In nearly all works related to large-scale notifications, the authors faced similar technical hurdles. One of the main challenges was in identifying an appropriate email address to contact (in an automated manner). We likewise faced these issues, resulting in a large number of bounces for generic aliases. Particularly sobering was the 85% bounce rate for the `security@` alias. Even though we could extract contact information from the WHOIS entries for some of the affected domains, these at times pointed to an Internet registrar, hosting provider, or WHOIS privacy proxy. Yet, the respondents to our survey stated that the WHOIS technical contact should prove most useful in trying to reach the administrator. In addition, our experiences with alternative communication channels have not found one superior to email.

Relying on large volumes of emails also increases the risk that the sending mail server becomes blacklisted. We witnessed that the most popular mail service, i.e., Google, likely flagged a substantial fraction of our messages as spam, although it is unclear precisely how the volume of sent emails affected the spam filtering decisions. Hence, having dedicated security contacts, possibly accessible via WHOIS or RDAP [22], could prove helpful in successfully delivering notifications.

Apart from email-based notifications, the use of vulnerability reward programs (VRPs) has been studied by previous works, showing that they are very effective [14]. From the responses to our survey, we also learned that site operators could be reached more easily if they could opt-in to a service, instead of receiving unsolicited notifications. While effective, VRPs incur additional costs for domain operators. Thus, it seems unlikely that a large enough number of domains will register for already existing VRPs to provide a complete solution. However, future efforts could attempt to run low effort and cheap services that provide vulnerability information.

B. Increasing Trust in Notifications

An over-arching issue faced by security notifications is a lack of trust in the message or the sender. While Cetin et al. [6] concluded that the sender does not impact the effectiveness of a notification campaign, our work paints a different picture. From the messages we received throughout our campaign as well as the responses to our survey, we repeatedly saw cases where trust was a vital concern. Domain operators explained that they were often contacted by malicious parties, either trying to sell services or products, or attempting to phish the recipients. Our security notifications appeared similar in nature, as they too were unsolicited. In addition, we received feedback that our sending institution was not well-known to the recipients and from their perspective, our email lacked means to verify our identity (even with our signed email variant). Thus, technical means to verify identities (such as digital signatures) are themselves insufficient in establishing trust.

From these sentiments, we suggest that future notification efforts should consider partnering with well-known organizations, such as Google. As a case in point, Li et al. [20] used Google's Search Console program to notify hijacked webmasters, observing some of the highest remediation levels from prior notification works (about 80% of webmasters took action). From a research perspective though, an interesting direction is exploring how to establish this trust independently. Some survey participants mentioned that the external links in our messages decreased its trustworthiness. However, Stock et al. [25] experimented with removing linked resources and disclosing all technical details in the notification email itself, observing lower fix rates. Hence, there is room for future work on establishing more trust in the notification messages.

C. Tailoring Notification Content to the Recipients

In our work, we considered two types of issues that affected different types of operators. On the one hand, WordPress can be easily installed by almost anyone, as it merely requires downloading the packed source and following the setup instructions. On the other hand, use of Git implies that the site operates on source code which requires versioning. Intuitively, Git usage requires a higher degree of technical expertise than setting up a WordPress installation. Thus, it is possible that WordPress users might benefit most from different information than more technical Git users. Additionally, different tones or message presentations (perhaps to catch the attention of, establish trust with, or further incentivize recipients) may have varying success depending on the notified population.

D. Cost Effectiveness

Setting up the infrastructure to automatically detect and notify on security issues at scale is a non-trivial but front-loaded cost, as each subsequent notification campaign can rely on the existing infrastructure. However, our work (as well as prior studies) have shown that these fully automated approaches yield suboptimal results. Thus in our study, we investigated the feasibility of utilizing other means of communication, which incur additional costs (either financial or manual effort) that scale linearly with the number of notified sites.

In our experiment, the *Friendly* channel had the highest fraction of viewed reports for both types of security issues. However, the definition of *viewed* is different from the other treatment groups, as this merely means that we sent an email with the report, and not that somebody actually saw the details. This manual approach did not exhibit the highest fix rates for the issues we disclosed, thus was not worth the effort. Similarly, our manual notifications showed comparable results in helping administrators in fixing their sites but incurred even higher financial or manpower costs. From the cost-benefit tradeoff, automated email notifications are still our best option.

In general, investigating the economics of notifications is an interesting future research direction. Are there incentives that can drive up remediation rates, such as through policy or legal changes? Alternatively, certain parties may be more receptive to notifications and could be prioritized for them. For example, we observed that WordPress sites that updated more frequently were more likely to fix in response to our notification. By considering the costs and benefits of various notification parameters, future research could identify areas of improvement for security notifications.

VIII. Related Work

For decades, the research community has extensively focused on discovering various issues, such as in Web applications (e.g., [4, 10, 18, 27]) and Internet servers (e.g., [3, 9, 12]). However, only relatively recently has the security community begun exploring the effectiveness of outreach efforts to inform affected parties and spur remediation.

One area of focus by these recent notification efforts is on incidents of compromise. In 2012, Vasek and Moore [26] investigated abuse reports to malware domains and how the level of detail in the report influenced cleanup rates. They found that 55% of all contacted domains cleaned up compared to 45% of unnotified sites. However, those who received detailed reports performed the highest, cleaning up 62% of their sites. In 2013, Canali et al. [5] studied how hosting providers handle such reports. For 20 hosting providers, they hosted Web sites serving malware and self-reported each site to the corresponding hosting provider, observing that 64% of the complaints were ignored. Similarly, Nappa et al. [21] notified 19 hosting providers about malicious servers in their network; however, only 7 actually took steps to take down the affected servers. Cetin et al. [6] analyzed the role of sender reputation on the success of malware cleanup behavior, using different sender names and email aliases. They did not strong evidence that the sender influences remediation, although their total population was 480 contacts (across multiple treatment groups). Recently, Li et al. [20] showed communicating with webmasters of hijacked Web sites via Google Search Console increased the clean-up rates by 50% (to approximately 80% of sites) and reduced infection lifetimes by 60%. This finding reinforces our notion that notifications rely on reliable points of contact, as Search Console can provide a higher fidelity delivery medium.

To proactively thwart future security problems, notifications can also target parties running vulnerable or misconfigured software. Durumeric et al. [13] discovered and notified servers susceptible to the 2014 Heartbleed bug, a vulnerability that received much public attention. Conducting an A/B notification experiment two weeks after public disclosure, they found that notifications resulted in a 50% increase in patching. Similarly, Kührer et al. [17] contacted administrators running services susceptible to DDoS amplification attacks. Within 7 weeks, 90% of the 9 million discovered servers could not longer be used for amplification attacks. A contributing factor was that the authors also coordinated with large backbone Internet providers, who began filtering potential amplification-inducing packets (such as NTP *monlist* requests) before they could reach vulnerable servers, thus potentially masking the true patch rate.

In 2016, two concurrent works reported on large-scale controlled notification experiments for vulnerable Web servers (Stock et al. [25]) and network misconfigured hosts (Li et al. [19]), exploring different factors that may affect a notification campaign's success. Both observed clear improvements in remediation levels, but noted the challenges of reliably reaching an appropriate point of contact and inciting further corrective actions. Most recently in 2017, Cetin et al. [7] investigated whether providing a mechanism to verify vulnerability could encourage further remediation. They found that those who visited their demonstration Web site were more likely to fix, but only 10% of contacts visited. They observed high bounce rates indicating a significant fraction of their notifications did not reach anyone. The goal of our work is to investigate and provide insights on the challenges these prior works have observed, to provide a roadmap for improved future notification efforts.

IX. Conclusion

In this paper, we have reported on a notification experiment targeting more than 24,000 domains, with the goal of shedding light on why the success of vulnerability notifications is more limited than ideal. By exploring a variety of facets surrounding these notifications, including the channel of communication, the message presentation, and Web site operator perspectives, we have concretely identified several challenges that inhibit a notification campaign's success. Our results and insights help lay the groundwork for future work in making security notifications more successful. In summary, our main takeaways are as follows:

- Email as a communication medium suffers from several shortcomings, including technical (e.g., anti-spam filters) and non-technical hurdles (e.g., lack of sender trust). However, the minimal increases in the success of alternative and less automated notification channels (such as phone, postal mail, or web forms) do not justify their significant financial costs and time overheads.

- We empirically validated that sender trust *does* have a significant influence on success, contrasting previous work [6]. It is not only vital that recipients trust a notification message, but also that email providers distinguish and discard truly unwanted spam and phishing emails.
- We identified a large gap between being aware of a security problem and addressing it. The fraction of sites that fixed a problem after learning about it differed per group, ranging from about 33% of vulnerable WordPress sites to 81% of sites with overly public Git folders. This highlights that reaching out to affected parties is only half of the battle, and the message itself is important in *convincing* operators to take action. Moreover, future work should investigate what level of technical detail is required depending on the type of vulnerability being reported.

We hope that future research such as those outlined in Section VII can address some of these challenges to pave the way to more successful security notifications.

Acknowledgements

We would like to thank the anonymous reviewers for their valuable feedback. This work was supported by the German Federal Ministry of Education and Research (BMBF) through funding for the Center for IT-Security, Privacy and Accountability (CISPA) (FKZ: 16KIS0345, 16KIS0656) and the CISPA-Stanford Center for Cybersecurity (FKZ: 13N1S0762), as well as the National Science Foundation awards CNS-1237265 and CNS-1518921.

References

[1] https://github.com/ben-stock/notification-ndss2018.

[2] "WordPress codex version 3.7," online, https://codex.wordpress.org/Version_3.7.

[3] N. Aviram, S. Schinzel, J. Somorovsky, N. Heninger, M. Dankel, J. Steube, L. Valenta, D. Adrian, J. A. Halderman, V. Dukhovni, E. Kasper, S. Cohney, S. Engels, C. Paar, and Y. Shavitt, "DROWN: Breaking TLS Using SSLv2," in *USENIX Security Symposium*, 2016.

[4] M. Balduzzi, C. T. Gimenez, D. Balzarotti, and E. Kirda, "Automated discovery of parameter pollution vulnerabilities in web applications," in *Proceedings of the Network and Distributed System Security Symposium, 2011*.

[5] D. Canali, D. Balzarotti, and A. Francillon, "The role of web hosting providers in detecting compromised websites," in *International World Wide Web Conference*, 2013.

[6] O. Cetin, M. H. Jhaveri, C. Ganán, M. van Eeten, and T. Moore, "Understanding the role of sender reputation in abuse reporting and cleanup," in *Workshop on the Economy of Information Security*, 2015.

[7] O. Cetin, C. Ganán, M. Korczynski, and M. van Eeten, "Make notifications great again: Learning how to notify in the age of large-scale vulnerability scanning," in *Workshop on the Economy of Information Security*, 2017.

[8] D. Crocker, "Mailbox Names for Common Services, Roles and Functions," RFC 2142 (Proposed Standard), RFC Editor, Fremont, CA, USA, pp. 1–6, May 1997. [Online]. Available: https://www.rfc-editor.org/rfc/rfc2142.txt

[9] J. Czyz, M. J. Luckie, M. Allman, and M. Bailey, "Don't forget to lock the back door! A characterization of IPv6 network security policy." in *NDSS*, 2016.

[10] A. Doupé, B. Boe, C. Kruegel, and G. Vigna, "Fear the EAR: Discovering and mitigating execution after redirect vulnerabilities," in *ACM CCS*, 2011.

[11] Drupal Security Team, "Public service announcement psa-2014-003." [Online]. Available: https://www.drupal.org/PSA-2014-003

[12] Z. Durumeric, J. Kasten, M. Bailey, and J. A. Halderman, "Analysis of the HTTPS certificate ecosystem," in *ACM Internet Measurement Conference*, 2013.

[13] Z. Durumeric, F. Li, J. Kasten, J. Amann, J. Beekman, M. Payer, N. Weaver, D. Adrian, V. Paxson, M. Bailey, and J. A. Halderman, "The matter of Heartbleed," in *ACM Internet Measurement Conference*, 2014.

[14] M. Finifter, D. Akhawe, and D. Wagner, "An empirical study of vulnerability rewards programs," in *USENIX Security Symposium*, 2013.

[15] Google, "Choose whether to show images," online, https://support.google.com/mail/answer/145919.

[16] S. Holm, "A simple sequentially rejective multiple test procedure," *Scandinavian Journal of Statistics*, pp. 65–70, 1979.

[17] M. Kührer, T. Hupperich, C. Rossow, and T. Holz, "Exit from hell? Reducing the impact of amplification DDoS attacks," in *USENIX Security Symposium*, 2014.

[18] S. Lekies, B. Stock, and M. Johns, "25 million flows later: Large-scale detection of DOM-based XSS," in *ACM CCS*, 2013.

[19] F. Li, Z. Durumeric, J. Czyz, M. Karami, D. McCoy, S. Savage, M. Bailey, and V. Paxson, "You've got vulnerability: Exploring effective vulnerability notifications," in *USENIX Security Symposium*, 2016.

[20] F. Li, G. Ho, E. Kuan, Y. Niu, L. Ballard, K. Thomas, E. Bursztein, and V. Paxson, "Remedying web hijacking: Notification effectiveness and webmaster comprehension," in *International World Wide Web Conference*, 2016.

[21] A. Nappa, M. Z. Rafique, and J. Caballero, "Driving in the cloud: An analysis of drive-by download operations and abuse reporting," in *DIMVA*, 2013.

[22] A. Newton, B. Ellacott, and N. Kong, "HTTP Usage in the Registration Data Access Protocol (RDAP)," RFC 7480 (Proposed Standard), RFC Editor, Fremont, CA, USA, pp. 1–16, Mar. 2015. [Online]. Available: https://www.rfc-editor.org/rfc/rfc7480.txt

[23] B. Ramsdell and S. Turner, "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification," RFC 5751 (Proposed Standard), RFC Editor, Fremont, CA, USA, pp. 1–45, Jan. 2010. [Online]. Available: https://www.rfc-editor.org/rfc/rfc5751.txt

[24] P. Resnick, "Internet Message Format," RFC 2822 (Proposed Standard), RFC Editor, Fremont, CA, USA, pp. 1–51, Apr. 2001, obsoleted by RFC 5322, updated by RFCs 5335, 5336. [Online]. Available: https://www.rfc-editor.org/rfc/rfc2822.txt

[25] B. Stock, G. Pellegrino, C. Rossow, M. Johns, and M. Backes, "Hey, you have a problem: On the feasibility of large-scale web vulnerability notification," in *USENIX Security Symposium*, 2016.

[26] M. Vasek and T. Moore, "Do malware reports expedite

cleanup? An experimental study," in *5th Workshop on Cyber Security Experimentation and Test, CSET*, 2012.
[27] L. Weichselbaum, M. Spagnuolo, S. Lekies, and A. Janc, "CSP is dead, long live CSP! On the insecurity of whitelists and the future of content security policy," in *ACM CCS*, 2016.