Hey, You Have a Problem: On the Feasibility of Large-Scale Web Vulnerability Notification

Ben Stock, Giancarlo Pellegrino, and Christian Rossow, ***Saarland University;***
Martin Johns, ***SAP SE;*** **Michael Backes,** ***Saarland University and Max Planck Institute for Software Systems (MPI-SWS)***

https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/stock

This paper is included in the Proceedings of the 25th USENIX Security Symposium

August 10–12, 2016 • Austin, TX

ISBN 978-1-931971-32-4

Open access to the Proceedings of the 25th USENIX Security Symposium is sponsored by USENIX

Hey, You Have a Problem: On the Feasibility of Large-Scale Web Vulnerability Notification

Ben Stock
stock@cs.uni-saarland.de
CISPA, Saarland University
Saarland Informatics Campus

Giancarlo Pellegrino
gpellegrino@mmci.uni-saarland.de
CISPA, Saarland University
Saarland Informatics Campus

Christian Rossow
crossow@mmci.uni-saarland.de
CISPA, Saarland University
Saarland Informatics Campus

Martin Johns
martin.johns@sap.com
SAP SE

Michael Backes
backes@cs.uni-saarland.de
CISPA, Saarland University & MPI-SWS
Saarland Informatics Campus

Abstract

Large-scale discovery of thousands of vulnerable Web sites has become a frequent event, thanks to recent advances in security research and the rise in maturity of Internet-wide scanning tools. The issues related to disclosing the vulnerability information to the affected parties, however, have only been treated as a side note in prior research.

In this paper, we systematically examine the feasibility and efficacy of large-scale notification campaigns. For this, we comprehensively survey existing communication channels and evaluate their usability in an automated notification process. Using a data set of over 44,000 vulnerable Web sites, we measure success rates, both with respect to the total number of fixed vulnerabilities and to reaching responsible parties, with the following high-level results: Although our campaign had a statistically significant impact compared to a control group, the increase in the fix rate of notified domains is marginal.

If a notification report is read by the owner of the vulnerable application, the likelihood of a subsequent resolution of the issues is sufficiently high: about 40%. But, out of 35,832 transmitted vulnerability reports, only 2,064 (5.8%) were actually received successfully, resulting in an unsatisfactory overall fix rate, leaving 74.5% of Web applications exploitable after our month-long experiment. Thus, we conclude that currently no reliable notification channels exist, which significantly inhibits the success and impact of large-scale notification.

1 Introduction

The large-scale detection of vulnerabilities in Web applications has become significantly more common over the course of the last years. This can be attributed to two concurrent developments: The ever-growing adoption of open-source Web frameworks and the recent advances in automated vulnerability detection.

Open-source Web application frameworks, such as Joomla, Drupal, or WordPress, nowadays constitute the technological basis for a vast number of Web sites. Thus, a single vulnerability in any of these frameworks makes tens of thousands of Web applications attackable at once. Previously disclosed vulnerabilities range from Cross-Site Scripting attacks [20], to more severe attacks like SQL injections [10] or object deserialization flaws [19]. Given the rise in maturity and efficiency of Internet-wides scanning tools, such as ZMap [12], such flaws can effectively be identified on affected Web sites.

Furthermore, a recent stream of security research has demonstrated automated approaches that are capable of discovering Web vulnerabilities on a large scale. Examples of such research results include the discovery of high numbers of Client-Side Cross-Site Scripting problems [22], server-side application logic flaws [9], and instances of vulnerable server-side infrastructure [13].

However, as soon as a researcher has discovered security-critical vulnerabilities that affect thousands of Web sites, she has an ethical dilemma: On the one hand, with the awareness of a vulnerability comes the implicit responsibility of disclosing it to the affected parties. On the other hand, large-scale Web vulnerability disclosure is non-trivial, has not been well-studied, and there are no guidelines or suggestions on how to proceed.

Thus, in this paper, we explore the following questions: *Is it actually feasible to disseminate vulnerability information on a large scale? What are suitable communication channels to reach out to the affected parties? And does such a large-scale notification campaign affect the prevalence of vulnerabilities in the wild?*

To answer these questions, we conduct the first in-depth study on the feasibility and efficacy of large-scale Web vulnerability notification campaigns. To this end, we establish a large body of vulnerable Web sites, suffering from different types of vulnerabilities such as Reflected or Client-Side Cross-Site Scripting bugs. We then review communication channels to notify affected par-

ties at scale, including direct contacts (such as generic email aliases and WHOIS contacts) and indirect channels (such as hosting providers or CERTs). We notify the Web site administrators via these communication channels and compare how the vulnerabilities in the monitored Web sites evolve over time compared to a control group of vulnerable Web sites that we do not notify.

Our large-scale experiments reveal important takeaway messages for fellow researchers. First, we show that while large-scale notifications can have a statistically significant impact on the fix rate of vulnerable Web applications, the long-term impact is marginal. Second, we therefore analyze the efficacy of direct and indirect communication channels, both with respect to reaching the (initial) recipient of the notification as well as the impact of the channel on the vulnerability landscape. Third, from our results, we derive the core challenges of notification campaigns and find that failures to reach out to contact persons cause the most severe degradation of success rates. Finally, we cover the lessons we learned during our pioneering efforts in order to assist researchers in future large-scale notification campaigns.

To sum up, we make the following contributions:

- We systematically evaluate the suitability of different communication channels for large-scale vulnerability disclosure and present a methodology to measure the feasibility and efficacy of such a disclosure process (§3).
- We document the results of a large-scale notification campaign, discussing the data set of vulnerable domains (§4) and the impact our campaign of notifying 35,832 vulnerable domains had (§5).
- Based on the observed (low) impact on the global landscape, we analyze factors inhibiting the success of large-scale disclosure campaigns (§6).
- We highlight key insights gathered in our study and present directions for future work in this space (§7).

2 Problem Statement

In this section, we outline the research questions we aim to answer. Secondly, we follow up with information on the actual vulnerabilities we consider for our work.

2.1 Research Questions

The ad-hoc process of reporting individual Web-related vulnerabilities to a single vendor is fairly well understood. Given its small scale, such a notification can be conducted with some manual, site-specific effort. However, detecting large numbers of vulnerabilities spanning several parties has become the norm rather than the exception. Examples in the recent past include the identification of high numbers of vulnerable SSL implementations [13], Client-Side XSS vulnerabilities [22], or execute-after-redirect flaws in Ruby on Rails applications [9]. With higher numbers of affected parties, manual effort becomes unfeasible, and hence responsible disclosure transforms into a distinctly different problem. Therefore, it is necessary to investigate notification processes that function with little to no human involvement.

Taking this overarching motivation into consideration, we reach a set of distinct research questions: For one, given a large number of vulnerabilities that affect a similarly large set of disjoint site owners: *How can a scalable responsible vulnerability notification process be conducted?* As manual effort is not a realistic option, an automated vulnerability disclosure process is required.

Furthermore: *What are suitable communication channels to report vulnerabilities?* Not all methods to communicate vulnerability information can be easily used in automated processes. Once suitable channels are selected, it is necessary to examine *how successful a large-scale Web vulnerability disclosure process can be.* Even in an automated fashion, a large-scale notification campaign results in considerable effort: The notification process has to be set up, executed, and monitored. Moreover, even automated initial notification will in many cases lead to personal, non-automated communication with a subset of recipients of the vulnerability reports. Thus, it is reasonable to examine how significant the positive effect of our campaign was. Based on the results of such a study, another question arises: *Which methods were most successful in delivering notification, and what might inhibiting factors be?*

An increasing number of security researchers (professional and academic) spend significant time and effort to discover and fix security-critical software bugs, often wondering about how to proceed with large-scale disclosure processes. With the aforementioned research questions, we aim to shed light on the under-explored issue of *whether*, *how* and with *what chance of success* researchers can notify affected parties.

2.2 Vulnerability Information

To answer these research questions, we leverage a data set with concrete instances of Web sites that contain security-critical flaws. We can use this data set to notify affected parties and monitor their reactions. In the following, we outline the Web-related vulnerabilities we consider for our work, separated into *well-known* and *previously-unknown* vulnerabilities.

We establish the data set on well-known vulnerabilities by inspecting WordPress-based Web sites. We selected WordPress since it is the most frequently used

PHP-based web application, deployed by about 25% of the most popular Web sites [35]. To find WordPress vulnerabilities, we systematically searched the CVE (Common Vulnerabilities and Exposures) database [27] for vulnerabilities which could be verified with (i) non-intrusive proof-of-concept (*PoC*) tests and (ii) without requiring valid user credentials for the tested site. We chose two vulnerabilities: one reflected XSS from 2013 (CVE-2013-0237) and one Client-Side XSS discovered in 2015 (CVE-2015-3429).

In addition, we selected a recent vulnerability discovered by security researchers at Sucuri. This third vulnerability, which targets the XML-RPC service of WordPress prior to version 4.4, allows an attacker to perform brute-force amplification attacks [6]. The flaw is in the behavior of the XML-RPC service which accepts *multiple* remote procedure calls (RPCs) with *different* user credentials within a single HTTP request. As a result, an attacker can forge a request in which she tries several user passwords at once. In the remainder of this paper, we refer to this vulnerability as Multicall. All these vulnerabilities were already known and patched in current versions of WordPress when we started our experiments. In the following, we refer to these domains as D_{WP}.

Our data set on previously-unknown vulnerabilities contains Web sites from the Alexa Top 10,000 suffering from one or more Client-Side Cross-Site Scripting (*Client-Side XSS*) vulnerabilities. To detect these flaws, we used a methodology presented in our previous work (see Lekies et al. [22]) which is based on a taint-aware browsing engine and an exploit generator to gather verified exploitable vulnerabilities. To the best of our knowledge, these flaws were not shared with the site operators before our notification. The second data set thus represents a situation researchers face when discovering a previously unknown type or instance of Web flaws. We denote these domains as D_{CXSS}.

3 Methodology

In this section we cover fundamental aspects of our methodology on large-scale notifications. We first discuss which communication channels can be used to reach out to affected parties. Subsequently, we outline how we prepared the notification messages. Third, we present the metrics that help to answer our research questions. Finally, we discuss ethical aspects of our methodology.

3.1 Communication Channels

The first key challenge when disclosing a Web vulnerability is to *reach out to an appropriate contact person*, e.g., to the administrator of a vulnerable Web site. In the following, we review potential communication channels and discuss which of them are suitable in our context.

3.1.1 Direct Channels

We first consider and assess direct communication channels that lead to the responsible contact.

Web contacts (*discarded*) — One option would be to browse the Web site and search for contact email addresses, phone numbers, or contact forms. Naturally, calling affected parties or interacting with custom Web forms is not a viable option in a large-scale scenario. Alternatively, we could crawl the domain to extract email addresses. This process, however, is an unreliable and error-prone undertaking, and is complicated by anti-scraping mechanisms such as CAPTCHAs or obfuscated email addresses. Hence, such contacts are not viable for large-scale notifications and we do not consider them.

Generic email aliases — Standardized email aliases for each domain should ideally redirect to appropriate mailboxes and are *"to be used when contacting personnel of an organization"* (RFC 2142 [7]). The RFC also proposes alias categories suitable for our goal. Besides the security-related aliases, `security@` and `abuse@`, we chose to contact the support mailbox for the HTTP service, i.e., `webmaster@`. In addition, we include the generic `info@` alias.

Domain WHOIS information — The Domain WHOIS protocol can be used to query information about registered domain names. Depending on the providing server, however, the structure and content of the provided information varies. WHOIS data is optimized for readability to humans [8] and thus does not have a consistent document format [25]. While a human can easily use WHOIS to retrieve contact information for a *single* domain, it does not scale to large-scale disclosure. WHOIS providers also rate-limit requests and partially employ CAPTCHAs to protect contact addresses [17]. Hence, querying WHOIS at large scale is not feasible. Instead, to retrieve the WHOIS domain contact, we purchased a machine-readable WHOIS data set for the Alexa top one million Web sites as of September, 29th, 2015 [1] and augmented the data with additional, on-demand queries. To select the contact person, first priority was given to the registrant's email address. In cases where this did not exist, we selected the technical contact address instead.

3.1.2 Indirect Channels

Apart from direct channels, we can ask intermediaries to forward information about vulnerabilities on our behalf:

[1] We bought the data set from `http://whoisxmlapi.com`

VRPs (*discarded*) — In recent years, vulnerability reward programs (VRPs) have gained traction and their success has been studied by researchers [14]. VRPs incentivize researchers to responsibly disclose flaws either directly to the vendor or to a VRP organization. Companies such as Google run their own in-house programs, whereas others outsource the coordination to organizations like HackerOne or BugCrowd. Naturally, in-house programs focus on vulnerabilities that are specific to the company, and hence, cannot be used for large-scale disclosure. Moreover, VRP organizations usually only accept and forward reports for their customers. Given a large body of vulnerable sites from several domains, VRPs are not a viable option for large-scale notifications. We thus exclude VRPs from our study.

Hosting providers — Hosting providers may already have an infrastructure in place to receive and react to security complaints. The provider likely has an incentive to use its direct customer contacts to forward the vulnerability information. While there is no specific security-related contact for providers, each provider typically has an abuse mailbox, which is used to notify operators about malicious activities originating from their networks. Abuse contacts can be found in Regional Internet Registries (RIRs) contact databases or queried via the IP WHOIS protocol. Both systems rate-limit requests and return proprietary formats. To work around this limitation, we query the Abusix Contact IP WHOIS proxy service [1] to obtain contacts of providers hosting the vulnerable Web sites.

TTPs — Trusted Third-Party Coordinators (hereafter TTPs) such as CERTs (Computer Emergency Response Teams) can act as intermediaries to report software vulnerabilities to operators. TTPs either operate on a regional level, i.e., within countries, or on a global scale, e.g., FIRST (Forum of Incident Response and Security Teams) [15]. Typically, TTPs already have technical infrastructure and procedures to forward vulnerability information to the administrators within their authority. To select CERTs, we determined the countries in which the vulnerable Web applications are hosted. We selected the top-20 countries in our data set and looked up their national CERTs. As global coordinators, we chose FIRST [15], and Ops-T [29] (Operations Security Trust), a closed community of security professionals.

3.2 Notification Procedure

In this section, we outline our notification procedure. Our campaign consisted of sending emails to the four notification groups on a bi-weekly basis, i.e., an initial notification and subsequent reminders every two weeks. In each round, we only notified Web sites which were exploitable at least once in the previous 72 hours[2].

We split our overall data set of vulnerable Web sites into five disjoint groups of equal size, i.e., each domain is part of exactly one group. We use four notification groups and assign 1/5th of the domains to each of them: (i) generic email aliases, (ii) domain WHOIS contacts, (iii) abuse contacts of network providers, and (iv) TTPs. To reduce possible bias, we did not inform TTPs and network providers that they received only an excerpt of all affected sites in their constituency (since the rest were in the other groups). In addition, to set a baseline, we assign the fifth group to the *control group*, to which we did not send notifications.

3.2.1 Notification Types

To notify contacts, we carefully aggregated information to avoid a single contact receiving multiple notification emails, as discussed in the following.

Individual Disclosure — For the Generic and the WHOIS contact groups, we sent individual emails that contained a list of discovered flaws for their domain, as well as instructions to retrieve the technical report. We left it to the recipient to either view the technical report on our vulnerability disclosure Web interface, or to use our mailbot to retrieve the reports. Each domain therefore had a unique token assigned to it which could be used to retrieve the report. Additionally, the email informed the recipient that they could opt out of the experiments or get in touch with us via a dedicated mailbox. An example of the email is shown in Appendix A.

Aggregated Disclosure — For the network provider and TTP contact groups, the email contained a message similar to the one used for the individual disclosure, kindly asking the recipient to forward the information to the responsible domain admin. However, we aggregated vulnerability information per authority and prepared a single message to disclose multiple vulnerabilities affecting Web sites within the authority of TTPs and network providers. Specifically, we attached a CSV file specifying the domain, IP address, vulnerability types, Web interface link, and unique token for each site. The email also contained a note regarding opt-out and reaching us.

3.2.2 Mail Delivery and Anti-Spam Filters

To send out a large number of such notification emails, we opted to set up our own email server. Operating our own email infrastructure prevents side-effects that may have arisen when using existing email infrastructure of the university, e.g., in case an IP-based blacklist starts blocking our server due to the notification emails.

[2] See Section 3.3.1 for details on these exploitability checks

One of the key challenges for benign email campaigns is to avoid the emails being flagged as spam, based either on a bad sender reputation, or the message content. To minimize this risk, we implemented both Sender Policy Framework (SPF) [31] and, starting from the first reminder, DomainKeys Identified Mail (DKIM) [28]. For each email template, we used SpamAssassin to ensure that the message content would not be flagged as spam. Finally, throughout the mail sending process, we periodically monitored the reputation of our mail server by repeatedly querying IP-based blacklists, such as Spamhaus.org and SpamCop. In addition, as email providers may implement custom spam filters, we also tested our messages against the filters of the two most popular mail providers, i.e., Google Mail and Outlook, registering new email accounts on both services. In addition, we signed up for Microsoft's Junk Mail Reporting Program (*JMRP*), which provides feedback on the spam check for mail from a given host [26].

3.2.3 Report Interface

During the experiment design time, one of the main concerns was the manual effort of our staff to address the quantity of possible questions that Web site owners might have. To help us with this type of activity, we built a web-based system, composed of a back end component for our staff and a front end for the Web site owner.

We wanted to provide users with as much detail as possible on the vulnerability, its impact, and ways of fixing it. Therefore, we created report templates for each vulnerability. For WordPress, we provided the details and hints on updating the installation. For Client-Side XSS, we provided the proof-of-concept URL which would open an alert box, as well as information on all the files which were involved in the exploitable data flow. Depending on the type of flaw, one of the customized templates would be presented to users of the front end.

The back end allowed us to retrieve current Web site statuses and statistics. Additionally, we automatically assigned emails to each domain. This allowed us to easily find all emails associated with a domain to give the best possible information on questions from domain admins.

3.3 Measurements

We broke down our research questions into a number of measurements that we perform throughout our notification campaign. These measurements are:

Global and Per-Group Vulnerable Web Sites — To measure the number of Web sites that are still exploitable, we set up a monitoring system which periodically verifies the exploitability of flaws using PoC tests. In addition, as groups are mutually disjoint, our monitor naturally provides per-group results. The monitoring system is presented in Section 3.3.1.

Reachability of Recipients, Viewed Reports, and Time to Fix — We can directly measure the success of mail delivery by looking at both email responses and report interface access logs. The logs help to infer that a message has reached the recipient. We present this reachability analysis in Section 3.3.2.

3.3.1 Web Site Monitoring

Throughout our experiments, we periodically monitored Web sites to establish the point in time when they were no longer vulnerable. In the following, we discuss the different types and frequency of tests, as well as the approach used to determine that a site was fixed.

WordPress Vulnerabilities Tests — To test for the three WordPress vulnerabilities, we implemented the following vulnerability-specific probes.

XSS Vulnerabilities — The CVE-2013-0237 vulnerability affects a specific version of a Flash file which is part of the PlUpload component included in default WordPress installations. Our test retrieves the Flash file and compares its checksum against the checksum of the known vulnerable version. If the checksums match, the test returns *Exploitable*. In all other cases, it returns *Non-Exploitable*. Similarly, the presence of the CVE-2015-3429 vulnerability can be verified by comparing the checksum of a specific HTML page. If the checksum values match, the Web site is considered *Exploitable*; otherwise it is *Non-Exploitable*. Both these files have the same content regardless of Web site language, i.e., we did not have to implement a checker per site language.

Multicall — The detection of the Multicall vulnerability requires further care. In a vulnerable installation, the XML-RPC API checks all user-provided credentials per request. In the patched variant, it skips all credential checks if one has failed, but still returns a list of *invalid credential* error messages. As a result, the output of the service cannot be used to deduce exploitability. However, as vulnerable services process all calls—including the ones with invalid credentials—the processing time is longer than for the patched version. Based on this observation, we developed a test which uses this timing side channel to deduce if a site is vulnerable or not. For the technical details, we refer the interested reader to Appendix B. As side channels are susceptible to false positives, we correlate the results with the deployed version of WordPress, which we extract by using the testing tool *plecost* [18]. If both timing analysis and version reflect a vulnerable service, the site is *Exploitable*. In all other cases the Web site is *Non-Exploitable*.

Web Site Time-Series Analysis — Since the exploitability relies on core components of WordPress, it can be reliably triggered. To keep server loads to the necessary minimum, we only checked for these vulnerabilities once per day. Given the variety and number of Web sites that we monitor, however, our point-wise observations are susceptible to temporal errors, such as Web applications that are temporarily inaccessible or are otherwise unresponsive at the time of our check. To decide whether a Web site is no longer exploitable, or just temporarily unavailable, we calculate the confidence of our tests. The confidence is the complement of the number of unlikely events (i.e., number of observed transitions from *Non-Exploitable* to *Exploitable*). A site is only marked as fixed if the confidence is greater than 0.99. For a precise definition of our confidence function, we refer the interested reader to Appendix C.

Client-Side XSS Test — To test for this vulnerability, we used the set of per-domain exploits discovered with our methodology from previous work. Each exploit is a URL including the XSS payload. For more details on this aspect, we refer the reader to our paper [22]. The exploits are grouped according to the vulnerability they trigger. To keep the load on the target server low, we initially only check one exploit URL. If the exploit works, we consider the site still vulnerable. If the exploit fails, we do not consider the Web site as fixed yet. In fact, as our previous work has shown, a vulnerable page may no longer exist or the flaw may be caused by rotating advertisements [32]. To rule out such volatility, we re-check the exploit every three hours. Additionally, if the page no longer exists, we check other exploits from the same exploit group and update the PoC if any of them succeeds. A flaw is only marked as fixed if it was not exploitable for at least three consecutive days.

3.3.2 Mailbox and Report Access Log Analysis

One of our core research questions is to study the effectiveness of each notification group. To measure if we actually *reached* someone, we analyzed our mailbox and the logs of the front end. Our front end allows notified parties to retrieve a detailed technical report by clicking on a link, or, if the owner distrusts URLs in emails, via our mailbot. In both cases, the recipient has to submit a unique token. We kept track of all actions by logging the tokens and access times. This allowed us to perform fine-grained analyses regarding the access to our vulnerability reports. In addition, the mailbox we used to disseminate emails received a variety of automated replies that provided insights on the status of the email delivery. We use all this information to classify each contact point into one of the following categories.

Reached — For individual disclosure, we state that we *reached* a contact point when the response message is, for example, an auto-responder message acknowledging the receipt of our email, a response by a tracking system (e.g., a new ticket), and other messages in which the recipient unequivocally states that the message was received. We say that we reached a contact point also when we observe an access to corresponding domain's technical report. In case of aggregated disclosure, if the email recipient, i.e., TTP or provider, is reached, then each of the domains within their constituency is also marked as reached. Similarly for individual disclosure, if a technical report for any domain within a constituency is accessed, we mark all associated domains as reached.

Bounced — Bounce messages are messages sent by a mail transfer agent to notify the sender that an error occurred and the message could not be delivered. Such errors might stem from the mailbox not existing or being full. If *all* emails we sent for a particular domain bounce, we classify the domain as *bounced*.

Unreachable — A contact point is unreachable when the response message indicates that no human will process our request. Examples of these cases are messages stating that the mailbox is unattended, or emails asking to contact Web site personnel only via a web form. Another example of an unreachable contact point is domains for which we could not retrieve any email address, e.g., if such information is missing in the WHOIS data.

Unknown — When we cannot establish whether a message was received, bounced, or the contact point was unreachable, we mark the contact point as *Unknown*.

While the identification of bounce messages is quite straightforward based on their content (e.g., SMTP error codes), automatically assigning emails to the aforementioned categories is prone to errors. Hence, we manually assigned incoming emails to one of the categories.

3.4 Ethical Considerations

We addressed ethical concerns from the early stages of our methodology design. In general, we design our experiments to be *unobtrusive*. This is, e.g., reflected on both the WordPress vulnerabilities selection criteria and the monitoring frequency. Despite that, our regular checks may still be undesired by network providers and Web site owners. For this reason, in our emails, we included instructions to *opt out* of our study. Moreover, we configured descriptive reverse DNS names for our infrastructure and hosted a website that described our initiative, again detailing contact information (postal address, email address, phone numbers) and an opt-out procedure.

The second ethical consideration of our experiments was fairness towards Web site administrators. As a result

of our methodology, there are administrators that were not made aware of vulnerabilities affecting their sites, i.e., were contained in the control group. After the end of our notification campaign, we informed them using the discussed direct channels and shared the list of vulnerable domains with the TTPs.

Our experiments are in part related to human operators on the receiving end of our notification emails. Our organizations, however, neither mandate nor provide an IRB approval before conducting such experiments.

4 Data Set

In this section, we explain the details of our experiments such as the time period, data sets of vulnerable applications, and composition of our notification groups.

Vulnerable Domains — In total, our data set included 44,790 Web sites of which 43,865 are WordPress-based Web sites suffering from at least one vulnerability discussed in Section 2.2 (D_{WP}). The remaining 925 Web sites were susceptible to site-specific Client-Side XSS exploits (D_{CXSS}).

Notification Groups — We randomly split the data sets of vulnerable Web sites into five equally-sized groups of 185 D_{CXSS} and 8,773 D_{WP} domains. During the lookup process, we could not retrieve contact points for several domains. For the domain group, the WHOIS database did not contain email addresses for 34 (18.4%) and 1,665 Web sites (19.0%) for D_{CXSS} and D_{WP}, respectively. For the network provider group, queries to the Abusix servers did not return a contact for 18 (9.7%) and 254 domains (2.9%) of D_{CXSS} and D_{WP}, respectively. In all these cases, we marked these Web sites as *unreachable* within their contact group.

For the TTP group, we followed a different approach. To select regional coordinators, we extracted the country code of the ASN hosting each domain, using the IP-to-ASN database of Team Cymru [33]. Finally, we selected coordinators for the 20 most frequent country codes from the list maintained by CERT-CC [4]. These regional CERTs account for 90.8% of the domains in the TTP contact group, the remaining 9.2% were hosted in a country outside of the top 20. To close this gap, we augmented the set of coordinators with the two global coordinators FIRST and Ops-T.

Notification Campaign — We notified the affected parties on Jan 14th, 2016. We sent two reminders, one after two weeks (Jan 28th), and one after four weeks (Feb 11th). For FIRST, we accidentally delayed the initial mail delivery by two days due to a misunderstanding, but made sure they received the vulnerability information as soon as the issue was resolved. In total, our server delivered 17,819 emails as the initial notification, 15,110 as primary reminders, and 13,588 as secondary reminders. In each round, the number of emails sent decreased because administrators fixed the vulnerability, they explicitly asked to be excluded from the experiments, or email addresses were invalidated due to bounces.

Unsubscribed Domains — As discussed in Section 3.4, we enabled domains to opt out of our analysis. Throughout the duration of our experiments, we received five requests to exclude a total of 187 domains, of which 149 were in a notification group and 38 in the control group. We received an email from a hosting provider on the second day of our campaign, threatening to sue our university if we did not immediately stop the analysis on this network. Additionally, we received messages from domain owners that were contacted by their hosting providers, stating that their Web sites would be taken down if the vulnerabilities were not fixed within a short timeframe. In these cases, we not only excluded the domains from any further analysis, but also reached out to the providers to clarify the obvious misunderstanding.

5 Site Vulnerability Evolution

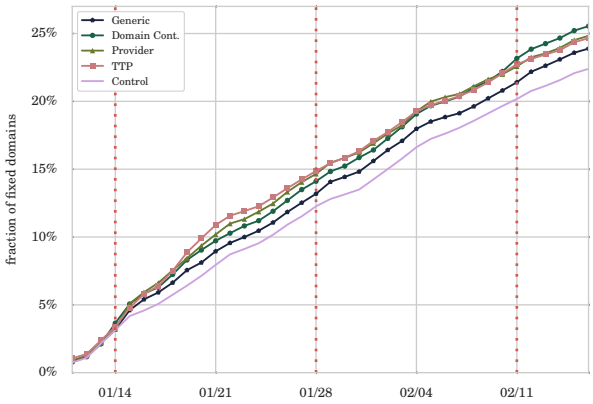
Using the methodology described before, we now instantiate our large-scale notification experiment and assess whether we can affect the prevalence of vulnerabilities in the wild. We answer this question by looking at the observed trends of fixed vulnerabilities and the significance of our campaign. Finally, we have a closer look at each data set and discuss the impact in isolation.

5.1 Trend of Fixed Vulnerabilities

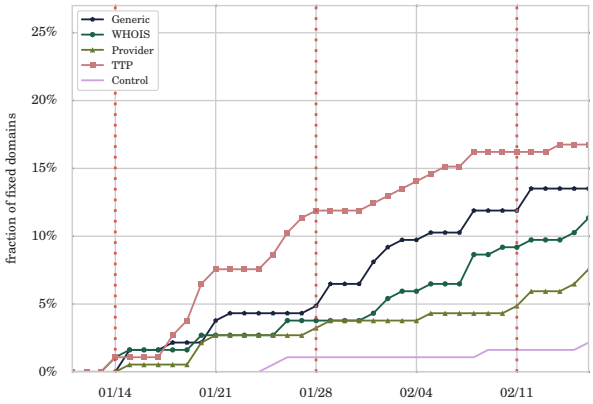
Table 1 shows the total number of non-exploitable domains at the end of our measurements (February 16th). The fraction of non-exploitable Web sites ranges from about 25.1% (*Generic*) to 26.5% (*WHOIS*) for D_{WP}. For D_{CXSS}, the fraction of domains has a greater variety and ranges from 8.6% (*Provider*) to 16.8% (*TTP*). We observe a distinctive difference in the fix rates for the control group for D_{WP} (23.3%) and D_{CXSS} (2.4%). We discuss the reasons for this in Sections 5.2.1 and 5.2.2.

	D_{WP}		D_{CXSS}	
Generic	2,201	*25.1%*	25	*13.5%*
WHOIS	2,325	*26.5%*	21	*11.4%*
Provider	2,261	*25.8%*	16	*8.6%*
TTP	2,268	*25.9%*	31	*16.8%*
Control	2,043	*23.3%*	4	*2.2%*

Table 1: Non-exploitable domains per group by 02/16



(a) WordPress



(b) Client-Side XSS

Figure 1: Fixed vulnerabilities over time

Figures 1a and 1b show the timeline and reveal when the vulnerabilities were fixed. For D_{WP}, we observe that all groups (including the control group) have a steady increase in non-exploitable domains. Each group progresses to the maximum, following a slight "wave" shape. This shape correlates to a weekly pattern. The weeks following a notification round (denoted as the horizontal lines) are characterized by a slightly steeper increase than the weeks before a notification round. For each of the groups, more than 2,000 of the 8,773 domains were no longer vulnerable at the end of our study. Overall, all notification groups performed better than the control group.

In contrast to the steady increase we observed for D_{WP}, no such pattern can be found for D_{CXSS}. Generally speaking, the difference between control and notified groups is much larger for D_{CXSS}. We discuss the results in Section 5.2.

Result Significance — During our experiments, we witnessed an increase in the fix rate for notified domains in comparison to the control group for both D_{WP} and D_{CXSS}. In order to ascertain whether this is due to chance or an effect of our campaign, we analyze the gathered data in more detail. To that end, we state the hypothesis H_0 that the difference in the results originates purely from chance and is not an effect of our notification. For both D_{CXSS} and D_{WP}, we use Pearson's χ^2 [30] test to calculate the p-values for each notification group in comparison to the control group. This test allows us to determine whether the differences between the results arose by chance. Both the resulting p-values and the raw χ^2 values for both D_{WP} and D_{CXSS} are shown in Table 2.

	D_{WP}		D_{CXSS}	
Generic	0.0053449	*7.76*	0.0000486	*16.50*
WHOIS	0.0000009	*24.24*	0.0004299	*12.40*
Provider	0.0001308	*14.63*	0.0058000	*7.61*
TTP	0.0000796	*15.57*	0.0000016	*23.00*
Overall Campaign	0.0000012	*23.51*	0.0000360	*17.07*

Table 2: p-values and *raw values* from χ^2 tests

In our case, however, we need to account for the comparison of the control group to several other groups. Hence, an error in the control group would bias all comparisons. Applying the Holm-Bonferroni method (α = 0.05) [16], we observe that every value p_i is below $\frac{\alpha}{5-i+1}$, i.e., H_0 does not hold. Hence, we find that the notified groups all differ significantly from the control group.

For the sake of completeness, in addition to the comparison to the control group, Tables 7 and 8 in the appendix show the p-values when comparing the notified groups against each other. While direct comparison between two groups sometimes reveals significant differences (e.g., D_{WP} WHOIS and Generic), no group performed significantly better than *all* other groups.

5.2 Impact Analysis

In this section, we analyze the collected data in more depth. More precisely, we discuss the changes for D_{WP} over time, showing how long-lasting the effect of our notification was. Moreover, we explain why the number of fixed domains in the control group is high, at almost one fifth. Subsequently, we discuss the impact of our notification on D_{CXSS}.

5.2.1 WordPress

Our notification campaign increased the number of domains that fixed the WordPress vulnerabilities by 11.2%. All notification groups outperformed the control group. The WHOIS group was the communication channel with the highest fix ratio (+15.4%). Next, the TTPs (+12.1%)

and providers (+10.9%) were second and third, respectively. The least effective channel was Generic, which still showed a 5.9% increase over the control group.

Besides the overall impact, we analyzed our collected data on a per-week basis, starting on the first day of our campaign. The results of this analysis are shown in Table 3. Next to the absolute number, the table also shows the fraction of domains fixed in each week. Note that this is relative to the number of domains that was still vulnerable at the beginning of that week for that group. In the first week, all channels performed better than the control group. In the second week, only providers still showed an observable increase in fixed domains. After the reminders, only the Generic and WHOIS contact groups had slightly increased fix numbers. All in all, however, the most drastic change occurred in the first week. Based on our definition of *fixed* domains (see Section 3.3.1), conclusive results can only be given at least three days back. As we stopped our experiments on January, 17th, we can only provide information on the first four weeks.

We observe that a substantial fraction of about one fifth of the control group was fixed during the duration of our study. This observation can be explained by analyzing the evolution of WordPress installations. We require version information to make a decision about the Multicall flaw, i.e., we had this information readily available for all days of the experiment. Indeed, out of the 8,773 domains in the control group, 1,134 moved from a version prior to 4.4 to an updated variant (not susceptible to Multicall) in the timeframe of our experiments. In addition, we observed that 360 domains no longer used WordPress or were offline at the end of our study. Throughout the remainder of the paper, we refer to these fixes as the *natural decay* of vulnerable domains.

It is also clear from Table 3 that in the first week of our campaign, a comparatively higher number of domains in the control group was marked first. While we cannot conclusively say why this occurred, we have anecdotal evidence from emails we received from one administrator who was responsible for multiple domains. Even though we had only notified him about one domain, he fixed several domains at once. Similarly, the WHOIS data set we purchased shows that different domains (spread across notified and control groups) contained the same contact email, i.e., had the same owner.

	Control	Generic	WHOIS	Prov.	TTP
14/1-20/1	438	521	610	631	664
	(5.1%)	(6.1%)	(7.2%)	(7.4%)	(7.8%)
21/1-27/1	387	387	397	416	395
	(4.8%)	(4.8%)	(5.1%)	(5.3%)	(5.0%)
28/1-03/2	382	416	418	380	380
	(5.0%)	(5.5%)	(5.6%)	(5.1%)	(5.1%)
04/2-10/2	386	389	402	368	365
	(5.3%)	(5.4%)	(5.7%)	(5.2%)	(5.2%)

Table 3: WordPress flaws fixed per week

5.2.2 Client-Side XSS

Contrary to what we observed for D_{WP}, there is no evidence for a natural decay of Client-Side XSS vulnerabilities. This stems from the fact that updates for the WordPress flaws are readily available and that sites are constantly being upgraded. For D_{CXSS}, however, to the best of our knowledge, developers were not aware of the flaws and no automated update existed to patch the flaws. Therefore, the impact of our campaign is much higher in comparison to D_{WP}.

In total, our campaign on average increased the number of fixed domains by a factor of almost 6, compared to the negligible number of *three* fixed domains in the control group. The highest fix rate was achieved by Trusted Third-Parties (16.8%), followed by the Generic channel with 13.5%. Additionally, 11.4% of the WHOIS and 8.7% of the provider group domains were fixed.

Important to note in this instance is one specific feature of Client-Side XSS: such issues are often caused by third-party scripts [32], which are out of the control of the administrator of the vulnerable domain. If a vulnerable third-party component is used across multiple domains, it is sufficient if *one* affected party reports this to the third-party vendor. In one particular case, we found that nine domains suffered from the same flaw. The vulnerability was fixed on February 8th, and its effect is visible in Figure 1b in the increase of fixed domains for Generic (three domains), WHOIS (four domains), and TTP (two domains) on that day. Since by chance, none of the domains was in the control group, this anomaly had a heavy impact on the overall notification campaign.

6 Communication Channel Analysis

In the previous section, we outlined the global picture on the vulnerability landscape and how much our campaign impacted it. Although the notifications for both D_{WP} and D_{CXSS} showed significant improvements over the control group, the number of domains which were fixed is unsatisfactory (25.8% and 12.6%, respectively). This raises the question whether we succeeded in reaching out to administrators. Therefore, in this section, we analyze how both direct and indirect communication channels performed in terms of successfully reporting the flaws to the responsible administrators.

6.1 Direct Channels

In this section, we analyze the direct channels, first determining whether we reached the intended recipient. We then assess how many reports were accessed and how quickly flaws were fixed after report view.

6.1.1 Reachability Analysis

For the direct channels, i.e., Generic email addresses and WHOIS contacts, we sent a single email per domain. As outlined in Section 3.3.2, we then classified each domain as to its *reachability state*. The results of the classification are shown in Table 4. Based on the numbers we observe, several characteristics for the direct channels become apparent. First and foremost, more than half of all domains are marked as *unknown*, either because the emails were silently bounced, delivered to an unmonitored mailbox, or ignored by the recipient. Apart from this, we observe that the groups have distinct differences, which we discuss separately in the following.

For the Generic group, we received a large number of email bounces. More precisely, for 50% and 28% of D_{WP} and D_{CXSS} domains, respectively, *all* emails we sent bounced. The difference in number of bounces between D_{CXSS} and D_{WP} likely originates from the higher popularity of D_{CXSS} Web sites (Alexa Top 10,000) in comparison to D_{WP} (Alexa Top 1 million). As popular Web sites may have a more structured staff, they may tend to adhere to standards like RFC-2142 [7], thus reducing the number of bounces. Nevertheless, even for the high-ranked D_{CXSS} Web sites, only 41 (22.2%) were actually reached. Moreover, for 90 D_{CXSS} domains (48.6%) we neither received emails acknowledging our reports, nor saw any hits on our Web site. Similarly, we observe that more than 45% of the D_{WP} domains are *unknown*.

The situation for the WHOIS group is slightly different: the fraction of bounces is significantly lower than for the Generic channels. This appears natural based on the fact that a valid email address is typically necessary to register a domain. However, apart from the large body of unknown domains, we see that the second-biggest fraction of domains belong to the *unreachable* bucket. This is caused by the fact that for 1,699 domains (both D_{WP} and D_{CXSS}), we could not retrieve a contact address from the WHOIS information. Additionally, for 37 domains, our messages were not delivered to the domain owner because the emails from the WHOIS database are of organizations that hide email addresses. The remaining 31 domains in that bucket were marked as unreachable since we received emails redirecting us to a Web-based form.

Given the large volume of emails we sent out for direct notifications (four on Generic, one on WHOIS), anti-spam filters also interfered with reaching out to Web site administrators. Despite our careful preparation of infrastructure and email content, our messages were partially labeled as spam. Even though our mailserver was never listed in any well-known blacklist, Microsoft's JMRP reported that the emails of the first two rounds were in parts flagged as spam. Interestingly, none of the emails of the second round of reminders were labeled as spam.

Additionally, provider-specific anti-spam filters mislabeled our emails. For 562 domains, we received bounces stating that our mails were classified as spam and thus rejected. In addition, in a handful of cases, we received feedback from contact points stating that they had only received our reminders and not the initial notification. This was either caused by *silent bounces* (the mail server accepted the email, but dropped it without notifying the sender), or by our email ending up in the spam folder, and then being automatically deleted after a few days.

6.1.2 Report Access

As we have seen in the previous section, the number of reached domains for the direct channels is low, amounting to only 357 and 714 D_{WP} domains for Generic and WHOIS, respectively. The increase in fixed D_{WP} domains compared to the control group, however, is even smaller: 158 and 282, respectively (see Table 1). We observe a similar trend for D_{CXSS}. To investigate this discrepancy, in the following we analyze for how many of the *reached* domains a report was accessed on our Web interface or via the mailbot.

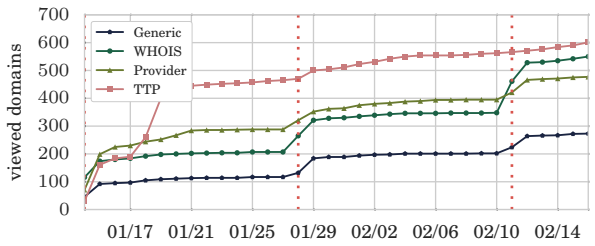
Table 5 shows the number of domains which accessed a report at least once. In addition, Figures 2a and 2b

	Generic		*WHOIS*		
	D_{WP}	D_{CXSS}	D_{WP}	D_{CXSS}	**Total**
Reached	357	41	714	38	1,150
Bounced	4,395	52	771	14	5,232
Unreach.	10	2	1,731	36	1,779
Unknown	4,011	90	5,557	97	9,755
Total	8,773	185	8,773	185	17,916

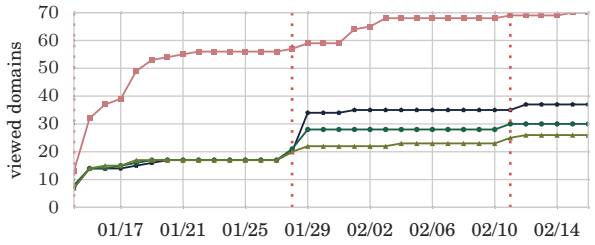
Table 4: Success of direct channels

	D_{WP}		D_{CXSS}	
Generic	273	*3.1%*	37	*20%*
WHOIS	550	*6.3%*	30	*16.2%*
sum direct	823	*4.6%*	67	*18.1%*
Provider	477	*5.4%*	26	*14.1%*
TTP	601	*6.9%*	70	*37.8%*
sum indirect	1,078	*6.1%*	96	*25.9%*

Table 5: Viewed reports for all channels up to 02/16

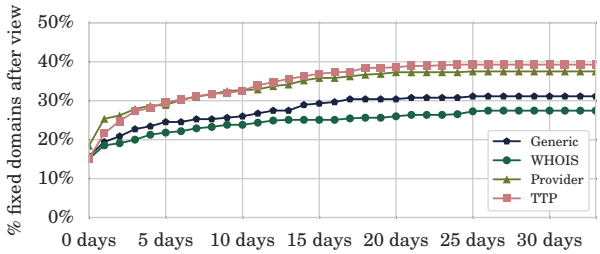


(a) WordPress

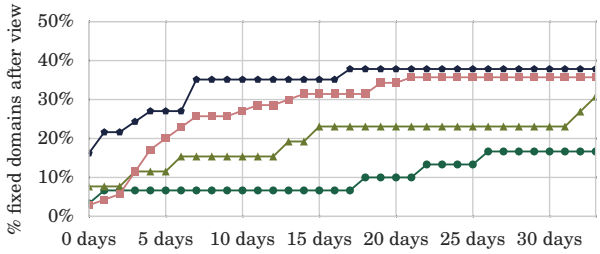


(b) Client-Side XSS

Figure 2: Accessed reports for different channels



(a) WordPress



(b) Client-Side XSS

Figure 3: Time from first report view to flaws fixed

plot the temporal evolution of the viewed reports for D_{WP} and D_{CXSS}, respectively. We observe that within a few days of the initial notification, the number of accessed domain reports stabilizes for both D_{WP} and D_{CXSS}. Both reminders increased the number of viewed domains, but interestingly the effect for D_{WP} was larger for the second reminder, whereas for D_{CXSS} this held true for the first.

In total, only 823 of the D_{WP} domains had reports viewed for the direct channels. For D_{CXSS}, the ratio of viewed reports was much higher, but still adds up to only 67 (18.1%) of the notified domains, whereas Generic was slightly better than WHOIS with 37 viewed reports.

6.1.3 From Report Access to Fix

Even though not all *reached* domains had a report view, the number of domains for which a report was viewed is still larger than the increase in fixed domains. This is due to the fact that instead of only viewing a report, the final step towards ensuring that the domain is no longer vulnerable is to understand the specific issue and patch it accordingly. In this section, we therefore analyze how many domains from the direct channels were fixed after a report was viewed and how long it took.

Figures 3a and 3b show the time between a report being viewed and the fix of the underlying flaw. For D_{WP} we observe on the Generic channel about 30% of the *viewed* domains are fixed within 5 days. Similarly, for WHOIS we observe a slightly higher rate of 32.5% within that timeframe. After this, the increase in fixed domains aligns with what we what observed for domains in the control group (roughly 0.5-0.7% per day). In essence, domains are either fixed within a very short time after initial report view or become non-exploitable just based on the natural decay of vulnerable domains.

For the D_{CXSS} domains, this pattern differs, showing significant increases in the fix rates even after more than a week. These flaws, in contrast to WordPress, are mostly site-specific and, more importantly, a fix has to be developed first. Hence, a longer timeframe for the fix is somewhat expected. Of all channels, the Generic channel had the highest rate of fixed domains after having viewed a report with about 38%. Noteworthy in this instance is the poor conversion rate for the WHOIS group. One possible explanation for this is the fact that the domain owner for high-ranked sites might be disconnected from the engineering team. Hence, the information might have been viewed by this person, but not properly forwarded to the actual administrator of the site. In contrast, for D_{WP}, we argue that such installations are mostly used by either small companies or single persons, and hence no such disconnect exists.

6.2 Indirect Communication Channels

After the discussion of direct channels, we now follow up with an in-depth analysis on the indirect channels.

	Provider		*TTP*		
	D_{WP}	D_{CXSS}	D_{WP}	D_{CXSS}	**total**
Reached	3,992	75	7,567	138	11,772
Bounced	2	0	0	0	2
Unreach.	271	20	0	0	291
Unknown	4,508	90	1,206	47	5,851
Total	8,773	185	8,773	185	17,916

Table 6: Success of indirect channels

6.2.1 Reachability Analysis

Similarly to direct channels, we also classified all domains based on their reachability. In this case, however, reaching the end point does not entail that we reached the administrator of the site. Instead, since we rely on intermediaries, we can only measure whether they received the email and not whether they forwarded it to the responsible party. This is also reflected in Table 6 in the large number of *reached* domains. As discussed in Section 3.3.2, a domain is marked as *reached* once the intermediary was reached, i.e., either confirmed our email or viewed at least one report disclosed to them.

In total, we reached 592 network providers responsible for 3,992 and 75 domains, respectively. Unreachable domains for the indirect channels were providers for which no contact existed in the Abusix database. In contrast to the high number of reached providers, about 50% for both D_{WP} and D_{CXSS} remain unknown.

For TTPs, such as CERTs or Ops-T, the numbers seem even more promising. Here, the *unknown* domains correspond to such domains for which we received no feedback from the hosting country's CERT, or those which were not located in any of the top 20 countries. Since a relatively small number of TTPs is responsible for a large body of vulnerable domains, the fraction of *reached* domains is comparatively high.

6.2.2 Report Access

The large number of reached domains appears to be a positive sign for a successful vulnerability notification. However, looking at the number of accessed reports shown in Table 5, we find that the improvement over the direct channels is less significant.

In general, the report access pattern for the provider group (depicted in Figures 2a and 2b) is similar to what we observed for direct channels: an initial increment of access to our reports after each notification round followed by long intervals of a quasi-constant number of accesses. For the provider group, the percentage of viewed reports remains low at 5.4% and 14.1% for D_{WP} and D_{CXSS}, respectively. For the most part, this is caused by unhelpful providers: the top 5 providers accounted for 2,082 domains, but none of them reviewed any report, thus effectively stopping the notification process dead in its tracks for more than half of the *reached* domains.

Compared to the providers, TTPs show a significantly different access pattern. First of all, after the first notification round, we observe an increase of access after four days. This can in part be attributed to FIRST receiving our initial email two days late. Additionally, from feedback received from the third-largest active regional CERT, we learned that they follow their own dates to distribute notifications. While we sent notifications every other Thursday, this CERT sent their notifications on Mondays. In addition, we argue that the CERTs did not simply forward our messages, but rather vetted them first. Therefore, it is highly likely that the information was vetted on Friday, and only forwarded to responsible parties on Monday. This can be observed in Figures 2a and 2b as the steep increase starting from the fourth day of our campaign (i.e., Monday, January 18th).

For TTPs, we have a large number of reached domains, but cannot observe an analogous increase in number of viewed reports. The fractions of viewed reports are 37.8% for D_{CXSS} and 6.9% for D_{WP}. In contrast to the direct channels, our measurements could not reveal direct causes for these low numbers such as bounces or unreachable contacts. An explanation may be that TTPs did not forward our notification emails to Web site administrators. Among the 20 regional CERTs, 18 reacted to our email. Since we did not receive bounces for the two non-reactive CERTs, we marked the domains in their constituency as *unknown*. 10 CERTs that reacted to our email did not view a single report. This could happen for two reasons. First, rather than vetting the information, these TTPs could have directly forwarded the information, but the Web site administrators did not receive them, or did not act upon them. Second, the CERT might not have forwarded the information at all. Given the assumption that a TTP would first vet the information originating from an untrusted source, the lack of accesses to the report favors the second explanation, i.e., our notification messages were not forwarded by half of the CERTs.

In total, the combination of indirect channels performed better in terms of accessed reports than the combination of direct channels. However, providers performed worst of all channels for D_{CXSS} and ranked third for D_{WP}. In contrast, TTPs were most successful for report access on both types of vulnerable domains. This result, however, is greatly influenced by Ops-T: this TTP alone was exclusively responsible for 135 report accesses for D_{WP} and 36 for D_{CXSS}. Let us consider a scenario in which we could have relied only on regional CERTs and FIRST. In this case, the number of viewed reports for TTPs would have gone down to 466 and 34 for D_{WP} and D_{CXSS}, respectively. These numbers are similar to

or even lower than the other groups. Hence, although the TTPs were of great help in our campaign, a researcher without access to the vetted Ops-T community could not have achieved a comparable report access rate.

6.2.3 From Report Access to Fix

For the indirect channels, we also measured the time between first access of a report and fix for the disclosed vulnerability. The results are depicted in Figures 3a and 3b. For D_{WP}, although TTPs perform similar to the direct channels, they have a distinct lag. More precisely, only 15.4% of domains were fixed within one day of report access, whereas this number ranged between 20.8% and 22.4% for direct channels. This underlines our hypothesis that TTPs would first vet the information we presented them. Hence, the first access to a domain report would have originated from the TTP, which subsequently forwarded the information to the responsible party.

As discussed before, there is less of a natural decay for D_{CXSS} vulnerabilities, hence any fix is more likely to be caused by our notifications. However, due to the specifics of Client-Side XSS, where a single third-party script may be the cause for multiple flaws, it is hard to detect a distinct trend for the vulnerabilities. Similar to the D_{WP} flaws, we observe a lag for both providers and TTPs between the initial view of the report and the time to fix, especially compared to the Generic group. This again highlights the vetting process of the intermediaries.

6.3 Discussion

As this section highlighted, the most significant issue we were faced with during our notification campaign was reaching the administrators in the first place. The issues depend on both communication channels and characteristics for the vulnerable domains. Naturally, reaching the intermediaries was quite straightforward: the email addresses were well-known and we only needed to send a comparatively small number of emails to them. In contrast, on the direct communication channels, especially for domains from D_{WP}, we had a large number of bounces or unreachable contacts to begin with.

While the results for reachability suggest that using intermediaries greatly improves the chances of successful notifications, the benefit was not carried on to the number of viewed reports. Although TTPs performed best for both D_{WP} and D_{CXSS} report views, this was mostly caused by the closed Ops-T community. Additionally, our reminders improved on the number of accessed reports, especially for the direct channels.

Once a report was viewed, fix rates for D_{WP} were similar across all groups, while the providers had a slightly higher fix rate. Moreover, we found that with a chance of approximately 30%, domains were fixed within 5 days of a report view. After this period, the fraction of fixed domains only increased by what we observed to be the natural decay of flaws. For D_{CXSS}, differences between communication channels were more drastic, with a much lower performance by the WHOIS channel. Although the number of fixes is subject to side-effects from vulnerabilities caused by third-party scripts, we note that the Generic channel worked best when considering the fix rate within the first 5 days.

7 Key Insights and Follow-Up Questions

In our work, we not only wanted to measure how successful a large-scale vulnerability notification campaign could be, but also aimed at determining what issues researchers would face. In the following, we discuss the key insights gained in our efforts, and present a number of follow-up questions which arise out of our findings.

Establishing Communication Channels — First and foremost, establishing a direct communication channel is remarkably challenging. For direct channels, we observe three main problems: (i) standardized addresses perform poorly with less popular Web sites; (ii) WHOIS contacts are a valid alternative for less popular Web sites, but the WHOIS database is not complete (about 20% domains lack contact points); and (iii) sending a large number of emails can be considered a spam campaign. All these reasons contribute to the low fraction of viewed reports.

Relying on indirect channels may reduce the workload for disclosure, as it *de facto* outsources the effort to an external organization. We found that for a large fraction of the domains such an intermediary could be reached. Although judging from the number of viewed reports, TTPs have the highest success rate, the overall results are still unsatisfactory. Moreover, we cannot ascertain how many TTPs forwarded the information or simply discarded them. We also found that reminders do not cause significant changes for TTPs, but have a slight impact on providers. Given all these facts, we find that establishing a communication channel to Web site administrators remains a hard problem even in the presence of intermediaries. Thus, the first question that arises for future work is: *How can the security community come up with reliable means of establishing communication channels between researchers and affected parties?*

User Distrust — For our experiments it was imperative that we were able to keep track of delivered and viewed reports. Therefore, we embedded a link to our web interface into the email. This naturally increases the risk of improper spam classification. Moreover, the security community trains users not to click on untrusted links or respond to suspicious emails, i.e., a significant fraction

of all reached administrators might not have followed our link or accessed the report via email. To investigate to what extent such behavior might have influenced our findings, we sent emails containing the full vulnerability details to all domains of the control group that were still vulnerable at the end of our experiments, using only the direct notification channels. While for D_{CXSS}, no significant difference could be observed, D_{WP} domains notified this way show significant differences. Contrary to the intuition that a report only accessible via a link would hinder the campaign, however, these domains performed *worse* in terms of fix rates. This curious fact might be caused by the fact that the emails contained multiple links (e.g., to the vulnerable site, the description on mitre.org, and the information on updating WordPress), hence triggering more spam filters. Alternatively, such long emails might have aroused more suspicions by the recipients. Hence, this rises another question: *To what extent does the message tone, content, and length influence the success of notification campaigns?*

Sender Reputation — When looking at the results for TTPs in more detail, we find that the trusted Ops-T community was responsible for a large portion of successful notification deliveries, even up to 50% of D_{CXSS} viewed reports for the TTPs. Thus, we argue that although studies have shown otherwise [5], trust in the sender of a notification message may be important factor to a campaign's success. This also holds true for the German CERT, which (according to our data) was more inclined to forward our messages. In this case, the cause is most likely the fact that we originate from a German university and have had interactions with the CERT before. This brings up another question for future work: *What is the impact of the sender reputation, especially when using intermediaries, on the success of a notification campaign?*

Time to Fix and Need for Reminders — Once a report was viewed, the fix rate for D_{WP} was around 30% within five days, regardless of the channel that was used to originally transmit the report. After that, the fix rate approximates what we observe for the control group, i.e., is most likely not an effect of our notification. For D_{CXSS}, the fix ratio for Generic, Provider, and TTP channel was between 30% and 40%, while the WHOIS group achieved a fix rate of less than 20%. In total, fixing these vulnerabilities typically took longer, which stems from the lack of a readily available patch for the custom flaws.

Additionally, as indicated by the increase in viewed reports right after our reminders, we find that they are necessary and useful. Moreover, considering that a patch typically only occurred within the first five days of a report being viewed, *future work should select a shorter interval for such reminders.*

Results Generality — Our work was a first glimpse into the landscape of vulnerability notifications. We explicitly studied the effects of such a campaign in the context of Web vulnerabilities which could be verified without interfering with the server's normal operation, i.e., we did not consider high-impact flaws such as SQL injections or remote code execution. The impact of our notification campaign was statistically significant, but smaller than in other related experiments (e.g., [13, 21]). Judging from their results, the criticality of the discovered flaws may also effect the impact of a notification campaign.

While in principle we could have applied our methodology to other types of flaws, we limited our study specifically to Web vulnerabilities. Contrary to other works in this space, our methodology to find Client-Side XSS gave us the opportunity to notify domain owners of site-specific flaws rather than vulnerabilities which are the same across multiple installations. Also, to the best of our knowledge, no other parties had access to such a data set, and hence, the site administrators were not aware of the flaws before our notification. This data set gave us the opportunity to study the behavior of administrators when notified of previously-unknown vulnerabilities, and to perform a comparative analysis with a data set of known vulnerabilities.

Web vulnerabilities can be attributed to a *domain*, therefore allowing us to use additional anchors to reach administrators, e.g., in comparison to physical devices such as home routers. We nevertheless believe the methodology can be applied to other types of flaws, to determine whether the vulnerability type influences the impact of notifications. Hence, comparing different means of notifying parties for different types of vulnerabilities is an interesting direction for future research.

Even though the WordPress flaws were publicly known beforehand, they had not received media attention such as, e.g., Heartbleed or NTP amplification attacks. The vulnerabilities we disclosed also concerned the application layer rather than the network layer, i.e., needed to be fixed by a large number of disjunct site owners rather than significantly fewer network providers. Investigating these factors therefore is an interesting direction for future research. This opens new research questions such as: *Are campaigns more successful if the vulnerabilities gained attention in the media (such as Heartbleed)? Does it matter* who needs to fix the vulnerability*, be it a Web site developer, network admins, or end-users?*

8 Related Work

In this section, we relate our study to prior work, reviewing works on vulnerability notifications, large-scale security analysis and an emerging area of disclosing security-relevant information.

Large-Scale Vulnerability Notification — In concurrent work, Li et al. [23] investigated the feasibility of vulnerability notifications for networked systems, i.e., industry control systems, misconfigured IPv6 firewalls, and DDoS amplifiers. Similarly to our work, they discovered that notifications have a positive impact, but the global effect is low and thus unsatisfactory. Contrary to our work, they did not use links to track the reachability of recipients. They did, however, gain insights into how message content influences fix rates. Moreover, from messages received in a survey they handed out to the notified parties, they found that such notifications generally are welcomed by affected parties, underlining the need for future work in this problem space.

Prior to this work, Li et al. [24] investigated how notification of compromised Web sites can improve the time to clean-up from malware infestation. They find that directly communicating with administrators via the Google Webmaster Console increases likelihood of clean-up by 50% and decreases infection lengths by 62%.

Prior to these closely related works, Durumeric et al. [13] conducted a large-scale analysis of the Heartbleed bug. This work showed that large-scale notification may increase the number of patched servers by about 50%. The main differences between this work and our study are in the composition of the data set. The first, important one is the type of flaw: the Heartbleed bug was a very popular, high-impact flaw with outstanding media coverage and its own website. Our data set does not contain flaws of this type, however it contains previously undisclosed XSS vulnerabilities that, to the best of our knowledge, were unknown to the Web site administrator prior to our disclosure. As a result, our data set allows us to study the problem without bias due to popularity. Second, the authors used the network operator abuse contact (retrieved from the IP WHOIS), whereas we use multiple channels. Finally, our study focuses on Web vulnerabilities, and does not target flaws in the Internet infrastructure, e.g., SSL/TLS.

Similarly to the previous work, Kührer et al. [21] reported on the vulnerability disclosure process on a data set of 9 million servers susceptible to becoming unwitting attackers in NTP amplification attacks. The authors reported all flaws using *en masse* well-established channels. This allowed them to remove 90% of the vulnerable servers within 7 weeks. As opposed to our paper, they relied only on two channels, i.e., TTPs and vendors, and they did not consider other possible ones such as domain WHOIS. More importantly, the authors did not perform a comparative analysis between channels, leaving the research questions of our paper unanswered.

Large-Scale Security Analyses — Recently, we have witnessed an increasing number of large-scale security analyses ranging from validation of security testing techniques (e.g., Balduzzi et al. [2], Lekies et al. [22], Doupé et al. [9]) to Internet-wide analyses of insecure behavior (e.g., Durumeric et al. [11], Kührer et al. [21]), which can spot a large number of security issues. While most of the efforts have been expended on tools and analysis techniques, little has been done to address the problem of reporting the discovered issues. For example, Balduzzi et al. [2] tested 5,000 Web sites for HTTP parameter pollution (HPP), discovering a vulnerability in 30% of them. With reference to the disclosure, the authors left the problem unaddressed, only mentioning that they could not reach all Web site owners. (For other examples in the Web domain, please refer to Doupé et al. [9]).

To better support large-scale analysis, new tools have been developed. For example, ZMap [12] can scan the entire IPv4 address space in 45 minutes. ZMap has already been used for Internet-wide analysis. For example, Durumeric et al. [11] used ZMap to study the HTTPS ecosystem, uncovering a variety of issues including certificates with invalid domains and certificate misuse. Similarly, this paper does not address the problem of reaching operators to solve the problem.

Notification of Security-Relevant Information — This paper can be seen as part of an emerging line of research that develops the idea of using notifications of security-relevant information as a security measure. Works in this area studied the distribution of security-relevant information from different angles and with a major focus on malware reports. For example, Cetin et al. [5] studied the role of sender reputation in abuse reports by sending 480 reports to network providers and Web site owners from senders with different reputations. Their study found no evidence that reputation improves cleanup rates, but they observed that, after accessing an online technical report, network providers performed better than Web site owners.

Vasek and Moore [34] looked at the problem from the angle of the quality of the reports, concluding that detailed reports increase the number of cleanups, while reports with minimal details perform better than not sending reports at all. Our paper builds on the results of these works: we did not consider sender reputation as a variable, and we prepare detailed reports.

Canali et al. [3] studied provider diligence by setting up compromised Web sites and notifying them about ongoing malicious activities. Their experiments showed that 64% of complaints were ignored. While this work shows an alarming attitude towards these problems, the size of their data set, 22 providers, makes it hard to generalize to a larger scale. From this point of view, our paper provides a broader view on the issue, including other notification channels and comparative analysis using a control group as a baseline.

9 Conclusion

With the increase of inter-connectivity on the Internet, the magnitude and diversity of large-scale vulnerability incidents will likely rise. We presented our experiences with a large-scale notification process to inform Web site owners about vulnerable Web apps. While our notifications have had an statistically significant impact on the vulnerability remediation, the overall fix rate is unsatisfactory, leaving 74.5% of Web sites exploitable after our month-long experiment.

This naturally begs the question for the potential reasons for the large fraction of unfixed sites. The major cause is the unsolved challenge to reach out to persons who can deploy a fix, such as developers or administrators. Of all contacts that we notified, only 5.8% viewed our vulnerability report. Out of these, 40% fixed the vulnerability within a week. This, but also the ease of fixing the vulnerabilities (in most cases just update WordPress), indicates that the main problem is actually to *disseminate* the vulnerability information.

How do we inform affected parties about vulnerabilities on large scale? Identifying contact points remains *the* main challenge that has to be addressed by the Internet society, including network providers, CERTs, and registrars. We imagine that this problem could, for example, be tackled by centralized contact databases, more efficient dissemination strategies within hosters/CERTs, or even a new notification channel or trusted party responsible for such notifications. Until we find solutions to the reachability problem, the effects of large-scale notifications are likely to remain low in the future.

Acknowledgements

The authors would like to thank Thomas Schreck and the Spanish CERT for providing insights into the inner workings of CERT organizations. Also, we would like to thank the anonymous reviewers for their helpful comments. In addition, we thank our shepherd Leyla Bilge for her support in improving the paper for the camera-ready version. This work was supported by the German Ministry for Education and Research (BMBF) through funding for the Center for IT-Security, Privacy and Accountability (CISPA).

References

[1] Abusix GmbH. Abuse contact database. `https://abusix.com/contactdb.html`, 2016.

[2] Marco Balduzzi, Carmen Torrano Gimenez, Davide Balzarotti, and Engin Kirda. Automated discovery of parameter pollution vulnerabilities in web applications. In *Proceedings of the Network and Distributed System Security Symposium, 2011*.

[3] Davide Canali, Davide Balzarotti, and Aurélien Francillon. The role of web hosting providers in detecting compromised websites. In *Proceedings of the 22nd International World Wide Web Conference*, 2013.

[4] CERT-CC. List of National CSIRTs. `http://www.cert.org/incident-management/national-csirts/national-csirts.cfm`, 2016.

[5] Orcun Cetin, Mohammad Hanif Jhaveri, Carlos Ganán, Michel van Eeten, and Tyler Moore. Understanding the role of sender reputation in abuse reporting and cleanup. In *Workshop on the Economy of Information Security (WEIS 2015)*.

[6] Daniel Cid. Brute force amplification attacks against WordPress XMLRPC. `https://blog.sucuri.net/2015/10/brute-force-amplification-attacks-against-wordpress-xmlrpc.html`.

[7] D. Crocker. Mailbox Names for Common Services, Roles and Functions. RFC 2142 (Proposed Standard), `http://www.ietf.org/rfc/rfc2142.txt`, May 1997.

[8] L. Daigle. WHOIS Protocol Specification. RFC 3912 (Draft Standard), `http://www.ietf.org/rfc/rfc3912.txt`, September 2004.

[9] Adam Doupé, Bryce Boe, Christopher Kruegel, and Giovanni Vigna. Fear the EAR: Discovering and mitigatfeaing execution after redirect vulnerabilities. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*, 2011.

[10] Drupal Security Team. Drupal core - highly critical - public service announcement - PSA-2014-003. `https://www.drupal.org/PSA-2014-003`.

[11] Zakir Durumeric, James Kasten, Michael Bailey, and J. Alex Halderman. Analysis of the HTTPS certificate ecosystem. In *Proceedings of the 2013 ACM Internet Measurement Conference*, 2013.

[12] Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. ZMap: Fast Internet-wide scanning and its security applications. In *Proceedings of the 22nd USENIX Security Symposium*, 2013.

[13] Zakir Durumeric, James Kasten, David Adrian, J. Alex Halderman, Michael Bailey, Frank Li,

Nicholas Weaver, Johanna Amann, Jethro Beekman, Mathias Payer, and Vern Paxson. The matter of Heartbleed. In *Proceedings of the 2014 ACM Internet Measurement Conference*, 2014.

[14] Matthew Finifter, Devdatta Akhawe, and David Wagner. An empirical study of vulnerability rewards programs. In *Proceedings of the 22nd USENIX Security Symposium*, 2013.

[15] FIRST.org, Inc. Forum of Incident Response and Security Teams. https://www.first.org/, 2016.

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

[17] ICANN Security and Stability Advisory Committee. SAC 023: Is the WHOIS service a source for email addresses for spammers? https://www.icann.org/en/system/files/files/sac-023-en.pdf.

[18] Iniqua. plecost. https://github.com/iniqua/plecost, 2016.

[19] Joomla! [20151206] - Core - Session Hardening. https://developer.joomla.org/security-centre/639-20151206-core-session-hardening.html.

[20] Aaron Jorbin. WordPress 4.4.1 Security and Maintenance Release. https://wordpress.org/news/2016/01/wordpress-4-4-1-security-and-maintenance-release/.

[21] Marc Kührer, Thomas Hupperich, Christian Rossow, and Thorsten Holz. Exit from hell? reducing the impact of amplification DDoS attacks. In *Proceedings of the 23rd USENIX Security Symposium*, 2014.

[22] Sebastian Lekies, Ben Stock, and Martin Johns. 25 million flows later: Large-scale detection of DOM-based XSS. In *Proceedings of the 20th ACM Conference on Computer and Communications Security*, 2013.

[23] Frank Li, Zakir Durumeric, Jakub Czyz, Mohammad Karami, Damon McCoy, Stefan Savage, Michael Bailey, and Vern Paxson. You've got vulnerability: Exploring effective vulnerability notifications. In *Proceedings of the 25th USENIX Security Symposium*, 2016.

[24] Frank Li, Grant Ho, Eric Kuan, Yuan Niu, Lucas Ballard, Kurt Thomas, Elie Bursztein, and Vern Paxson. Remedying web hijacking: Notification effectiveness and webmaster comprehension. In *Proceedings of the 25th International World Wide Web Conference*, 2016.

[25] Suqi Liu, Ian Foster, Stefan Savage, Geoffrey M. Voelker, and Lawrence K. Saul. Who is .com?: Learning to parse WHOIS records. In *Proceedings of the 2015 ACM Internet Measurement Conference*.

[26] Microsoft Corporation. Services for senders and ISPs. https://mail.live.com/mail/services.aspx.

[27] MITRE Corporation. Common Vulnerabilities and Exposures. http://cve.mitre.org/.

[28] Mutual Internet Practices Association. DomainKeys Identified Mail. http://www.dkim.org/, 2016.

[29] OpSecAdmin. Operations Security Trust. https://www.ops-trust.net/, 2016.

[30] Karl Pearson. X. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 50 (302):157–175, 1900.

[31] SPF Project. Sender Policy Framework. http://www.openspf.org/, 2016.

[32] Ben Stock, Stephan Pfistner, Bernd Kaiser, Sebastian Lekies, and Martin Johns. From facepalm to brain bender: Exploring client-side cross-site scripting. In *Proceedings of the 22nd ACM Conference on Computer and Communications Security*, 2015.

[33] Team Cymru. Team Cymru IP to ASN Mapping. http://www.team-cymru.org/IP-ASN-mapping.html, 2016.

[34] Marie Vasek and Tyler Moore. Do malware reports expedite cleanup? An experimental study. In *5th Workshop on Cyber Security Experimentation and Test, CSET*, 2012.

[35] W3Techs. Usage of content management systems for websites. http://w3techs.com/technologies/overview/content_management/all/.

A Notification Email

```
Subject: Vulnerability Notification for  your domain
...com

Hello,

Primer: All information in and attached to this
email is confidential and should be passed on to
individuals and organizations on a need-to-know
principle only.

We are security researchers from Saarland University,
Germany. In our research, we have been scanning
several web sites for critical vulnerabilities.
We would like to inform you that your website is
susceptible to the following vulnerability(ies):

- XMLRPC Multicall Vulnerability

The XMLRPC API of Wordpress can be abused to execute
numerous commands on the server-side, thereby
allowing an attacker to bruteforce passwords or
perform a Denial-of-Service attack against the
server.

You can review more detailed
information using our web interface at
https://notify.mmci.uni-saarland.de/....
Alternatively, you can retrieve more information
via email. To do so, please respond to this email
and set the subject line to *only* contain the token
72cf.... We will automatically respond with the
vulnerability report via email.

Since this notification is part of an ongoing
research project, we will re-scan your web site to
see if the vulnerability has been fixed. If you wish
us to stop scanning your web site, please contact us
at contact@notify.mmci.uni-saarland.de. Should you
need further information or have any other questions,
please do not hesitate to contact us using the same
email address.

Best Regards,
 Ben Stock, Researcher at CISPA
---
Center for IT-Security, Privacy, and Accountability
Saarland University, Building E9 1
Phone +49 681 302 57377
```

B Multicall Checker

The Multicall checker uses a timing side-channel to determine whether a WordPress XMLRPC service is vulnerable or not. First, we estimate network latency with a probe request. We use the round-trip time of the probe to rule out network transmission time from the measurement of the multicall request. The resulting value is an estimation of the CPU time. However, as this value may be sensitive to fluctuations of the network latency estimation, we increase the execution time with inefficient user credentials. These strings trigger WordPress sanitization procedures, resulting in longer execution time, and guarantee to always fail at login attempts to prevent accidental unauthorized access to a WordPress installation. Our test estimates CPU time of two multicall requests with 40 and 80 calls per request. We measured that our servers take 1,600 ms and 3,300 ms of CPU time on an Intel i7 processor for 40 and 80 multicalls, respectively, which we use as indicators for vulnerable services. Finally, we correlate the time analysis with the version of the deployed WordPress, which can be extracted from several sources. If both timing analysis and version indicate a vulnerable service, then we mark the Web site as *Exploitable*, otherwise as *Non-Exploitable*.

C Confidence Function

The monitoring system for WordPress returns a collection of time series $T^v_{ws} = s_1 \cdot s_2 \cdot \ldots \cdot s_n$ where ws is the Web site, v a vulnerability, and s_i is the result of the checker in a point of time i with $1 \leq i \leq n$. The value s_i can be E if the checker found the vulnerability *Exploitable*; N for *Non-Exploitable*. In our experiments, we need to establish whether a Web site ws has fixed v. To do that, we take into account the rate of unlikely events within a time series in order to establish a confidence level, i.e., the number of substrings $N \cdot E$ in the longer prefix of the time series. We define the confidence that the Web site is not vulnerable as the complement of this rate. We define a Web site as not vulnerable if the confidence is greater than 0.99. If no errors occur in a time series, we define a conservative error of 0.1, i.e., a domain is marked fixed if it was *Non-Exploitable* for three consecutive days.

D Comparison of Notified Groups

	Generic	WHOIS	Provider	TTP
Generic	-	0.0323810	0.2982599	0.2456719
WHOIS	0.0323810	-	0.2714901	0.3276381
Provider	0.2982599	0.2714901	-	0.9038795
TTP	0.2456719	0.3276381	0.9038795	-

Table 7: p-values from χ^2 tests for D_{WP}

	Generic	WHOIS	Provider	TTP
Generic	-	0.5285343	0.1360734	0.3841099
WHOIS	0.5285343	-	0.3862386	0.1346949
Provider	0.1360734	0.3862386	-	0.0191932
TTP	0.3841099	0.1346949	0.0191932	-

Table 8: p-values from χ^2 tests for D_{CXSS}