Hey, You Have a Problem:

On the Feasibility of Large-Scale Web Vulnerability Notification

Ben Stock, Giancarlo Pellegrino, Christian Rossow, and Michael Backes

CISPA
Center for IT-Security, Privacy and Accountability

Motivation and Research Questions

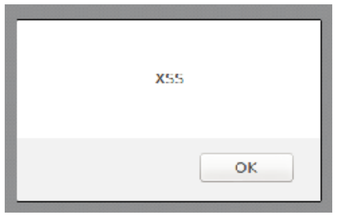
- Large-Scale (Web) Vulnerability Detection
  - Drupaggedon SQLi, Joomla! Object Deserialization, Client-Side XSS, Execute After Redirect on Ruby, …
- Focus previously on Detection, not Notification
- Our work: understand how notifications can work at scale
  - What are suitable communication channels for such a campaign?
  - Does such a campaign affect the prevalence of the notified vulnerabilities?
  - What might inhibiting factors be?

Study Setup

Types of Vulnerabilities

- Well-known vulnerabilities for WordPress (43,865 domains, Top 1M)
  - Reflected Cross-Site Scripting in PlUpload flash component (CVE-2013-0237)
  - Client-Side Cross-Site Scripting in Genericons Example Code (CVE-2015-3429)
  - XMLRPC Multicall Vulnerability
    - allows attacker to try multiple user/password combinations in a single HTTP request
  - Existing patches for all of them

- Previously-unknown Client-Side XSS vulnerabilities (925 domains, Top 10K)
  - Discovered with our methodology from CCS 2013
  - Site-specific flaws
  - No existing patches



Communication Channels

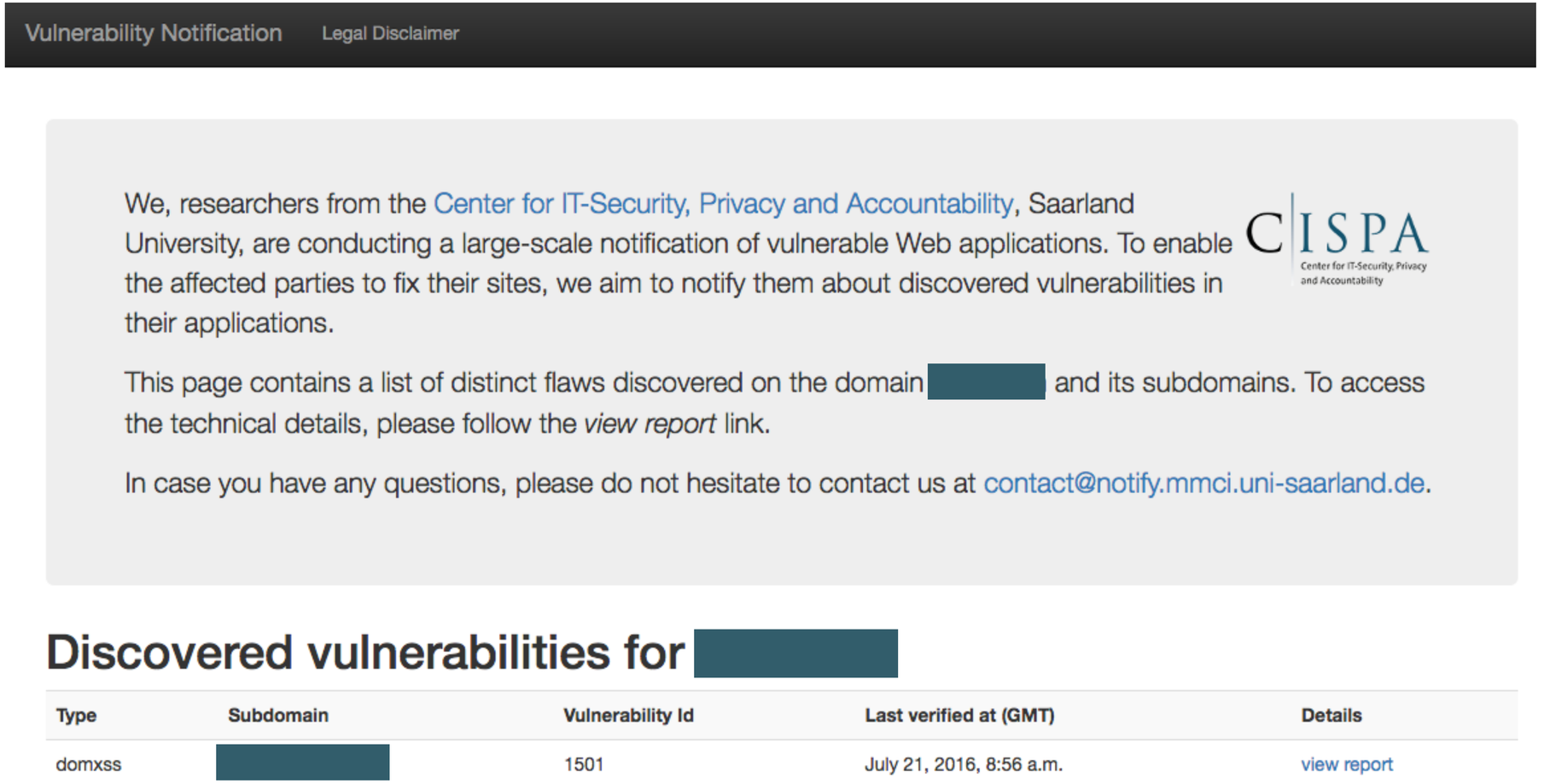
- Direct Communication Channels
  - ~~Web contact forms~~
  - Generic email addresses (info@, security@, webmaster@, abuse@)
  - Domain WHOIS information (registrant or technical contact)

- Indirect Communication Channels
  - ~~Vulnerability Reward Programs~~
  - Hosting providers (abuse contacts for the hosting IP range)
  - Trusted Third-Parties
    - regional CERTs (e.g., CERT US, CERT-Bund)
    - umbrella CERT organization FIRST
    - trusted community Ops-Trust

Notification Procedure

- Split up data set of vulnerable domains into five groups of equal size
  - Generic, WHOIS, Provider, TTP, and Control
- Notification via email with link to our Web interface
  - alternatively: access via email using token
- Aggregated Disclosure to providers and TTPs
- Bi-weekly emails
  - January 14th, January 28th, February 11th

Web Interface



Problem

From our analysis, your web site suffers from a DOM-based Cross-Site Scripting vulnerability. To the best of our knowledge, this is the first time this vulnerability is disclosed to you. For further information on this type of vulnerability and the common errors, please refer to this paper.

Impact

Cross-Site Scripting is a vulnerability which allows an attacker to inject malicious client-side script in the web application. An attacker can exploit a Cross-Site Scripting vulnerability by luring his victim to a page with a crafted URL. The injected script is executed within the victim's browser and can be abused to steal confidential information and impersonate the victim.

For further information, please refer to Wikipedia.

Proof of Concept

We verified that your web site suffers from this vulnerability, allowing an attacker to inject malicious code. To verify this yourself, follow the link below, which will open an alert box showing your domain name.

[redacted]nform=19910#"></SCRIPT><script>alert(document.domain)</script>

If you do not trust to click on the above link, you can reproduce the attack by copying the string

[redacted]nform=19910#"></SCRIPT>
<script>alert(document.domain)</script>

to the address bar of your browser.

Please note: This proof of concept might only work in Chrome and Internet Explorer due to browser-specific encoding behaviour. Also, since both browsers use a Cross-Site Scripting filter, the filter might be blocking the execution.

Remediation

Since this vulnerability specific for your web site, there is no generic way of fixing it. To assist you in spotting the flaw, we provide the source, the sink, and all involved HTML and JavaScript files hosting the vulnerable code in the following.

Source: location.href

Sink: document.write

Files: [redacted]/common/js/dojo/www.js

Measurements

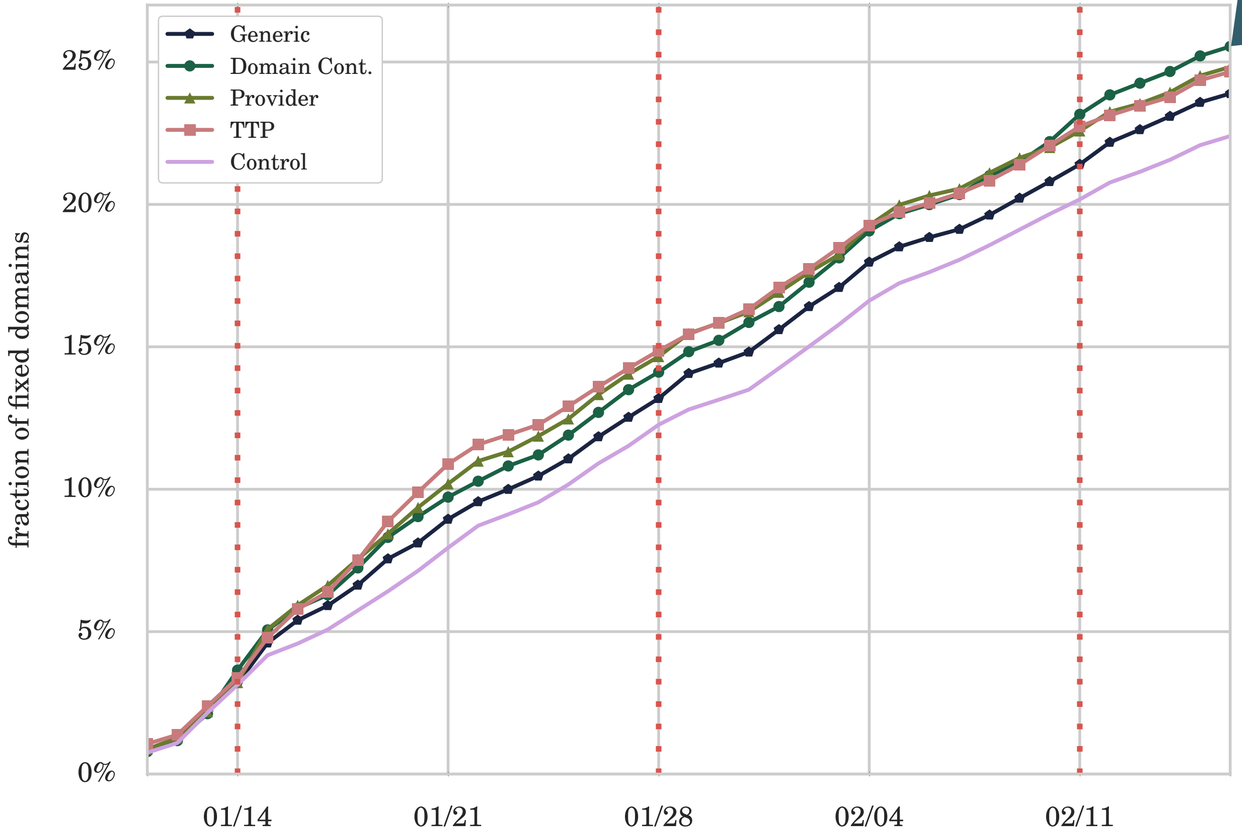
- Vulnerability monitoring
  - WordPress flaws: one PoC test per day
  - Client-Side XSS: one test every three hours

- Mailbox and accessed reports to classify domains
  - reached: report viewed or email acknowledged
  - bounced: all emails for this domain bounced
  - unreachable: no WHOIS contact, no provider abuse mail, or redirect to Web interface
  - unknown: all others
  - indirect channels: first step of the chain measured

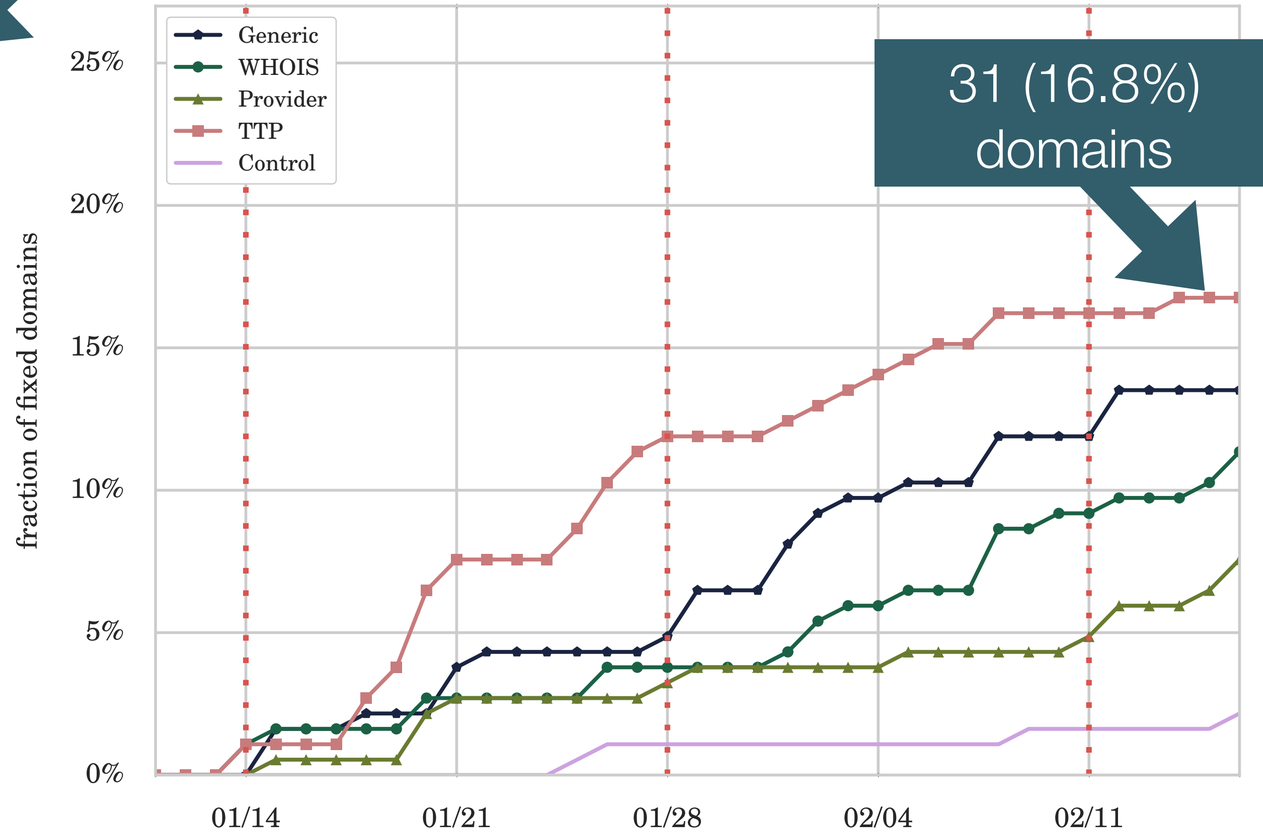
Global Impact of Notification

Fixed Sites over Time

WordPress



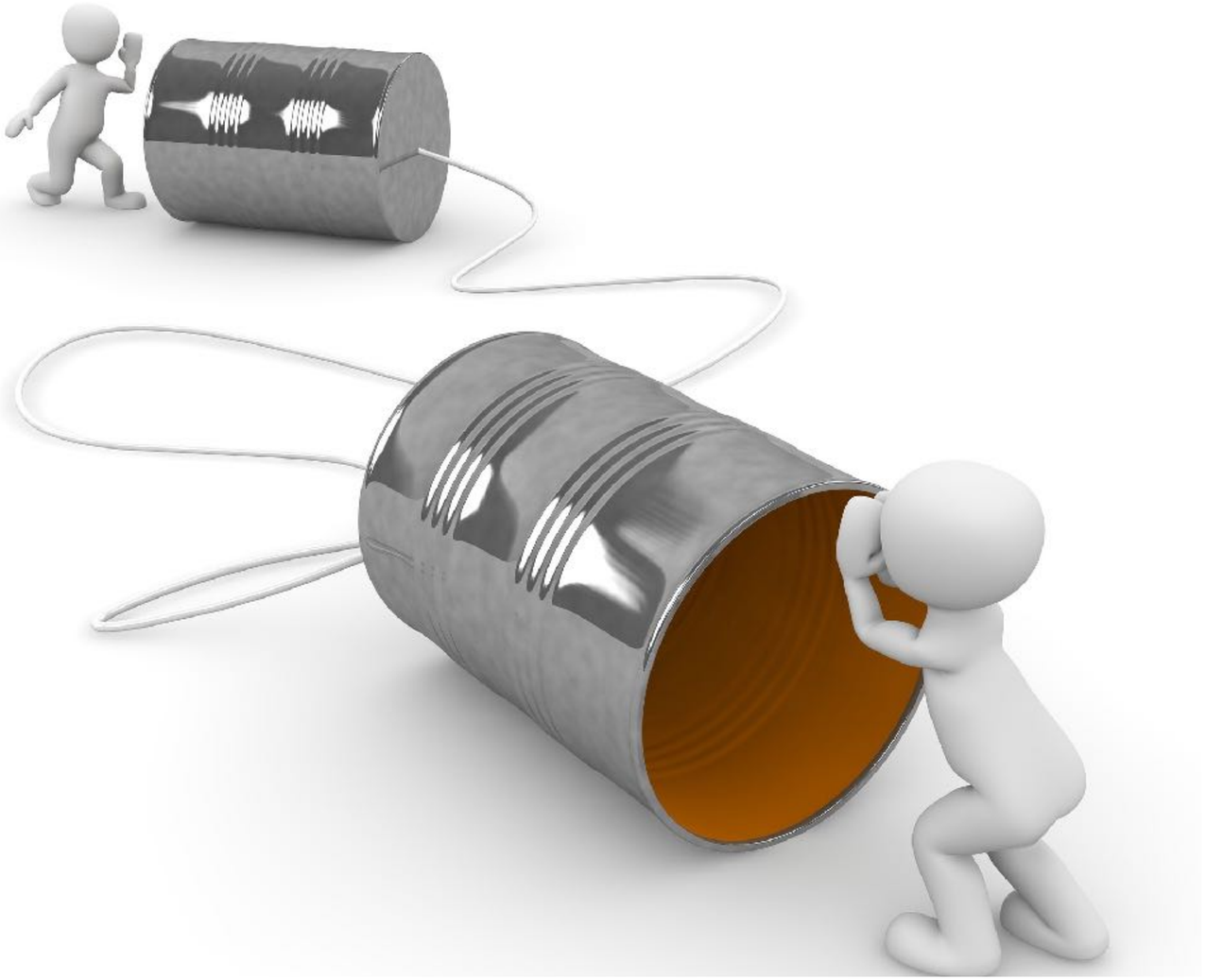
Client-Side XSS



CISPA
Center for IT-Security, Privacy and Accountability

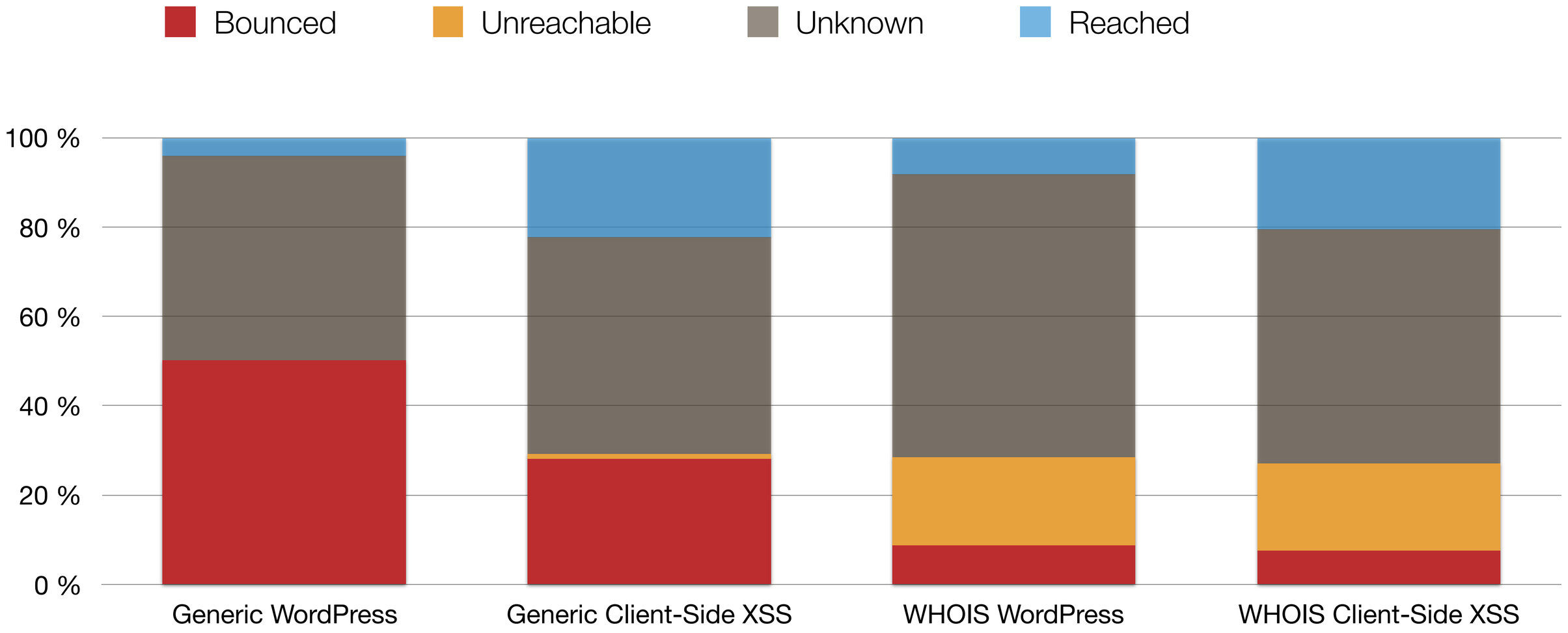
„Although the notifications for both WordPress and Client-Side XSS showed **significant improvements** over the **control group**, the number of domains which were fixed is **unsatisfactory** (25.8% and 12.6%, respectively).“

–The Authors

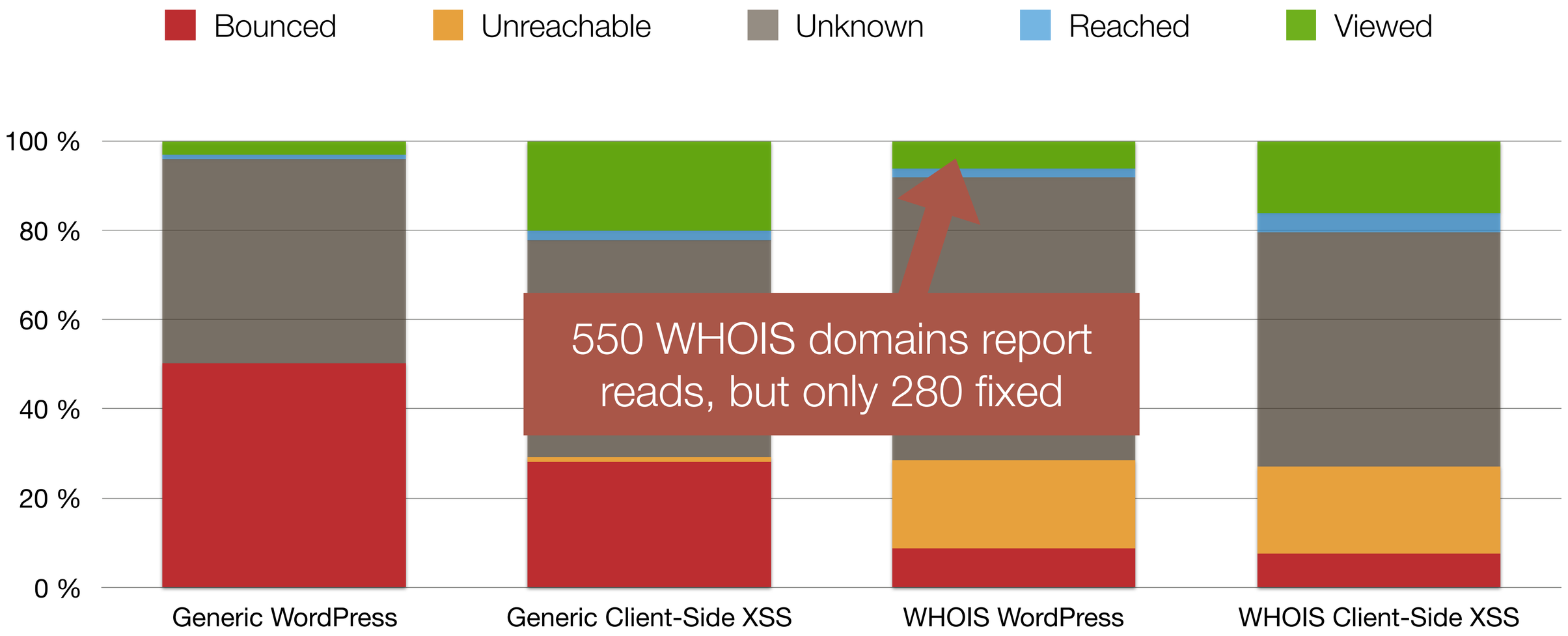


Communication Channel Analysis

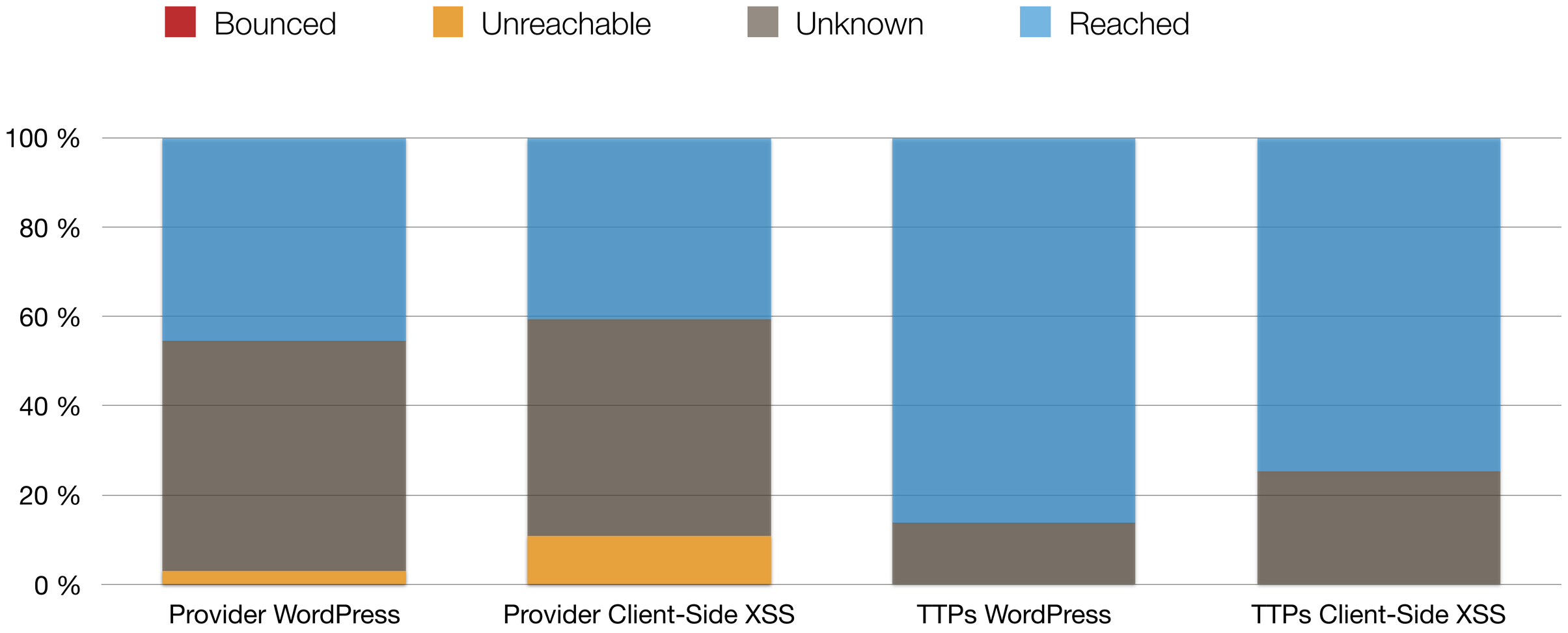
Reachability of Direct Channels



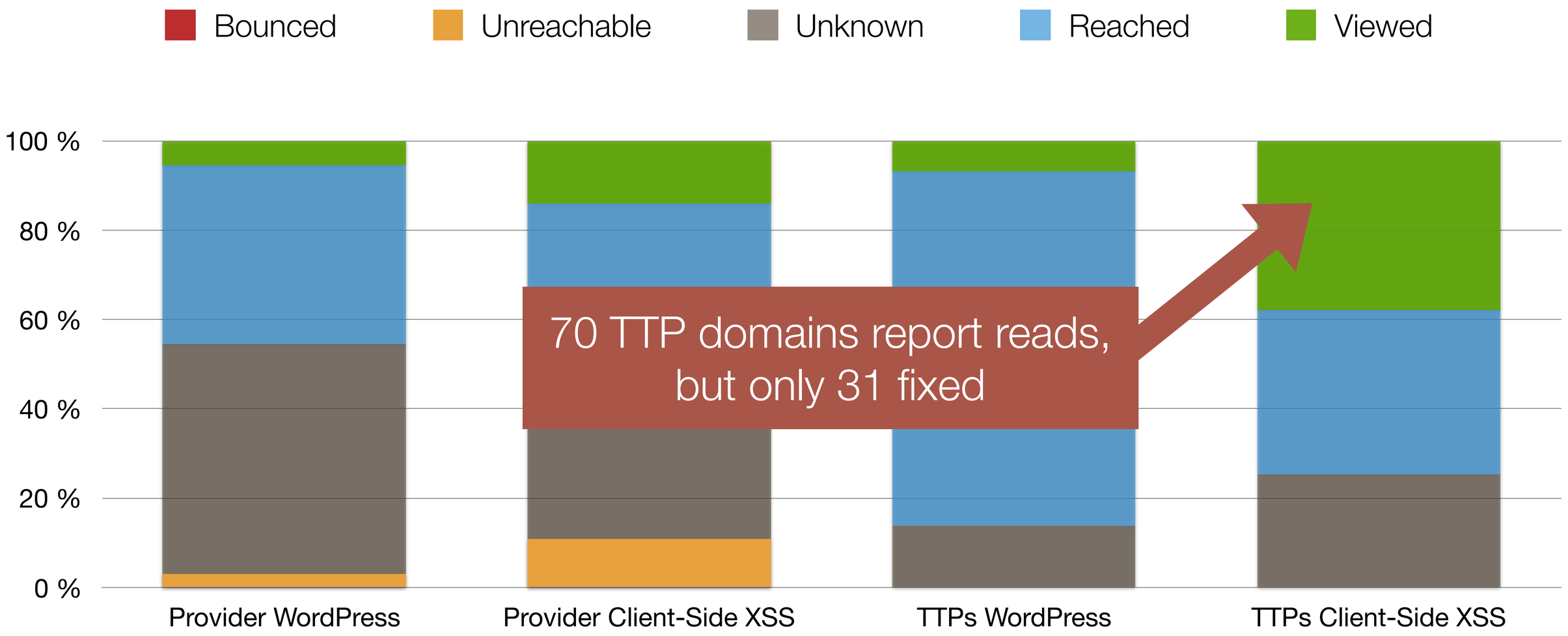
Reachability of Direct Channels



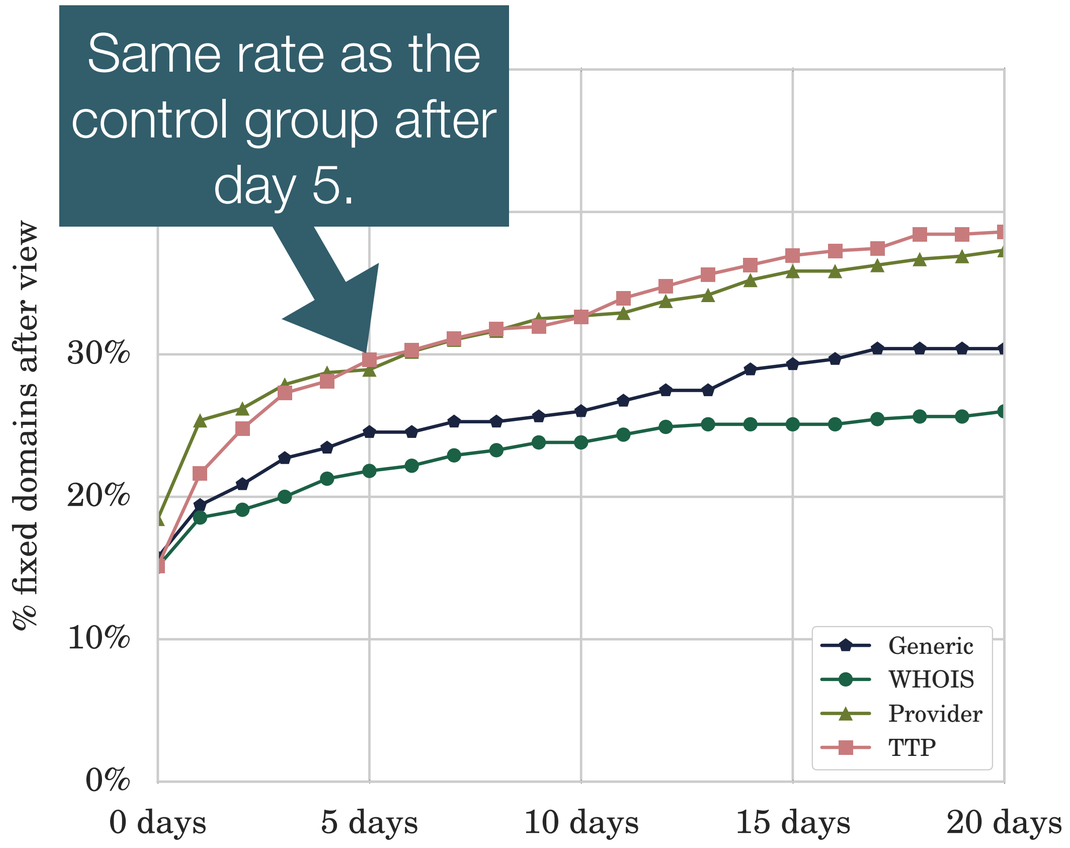
Reachability of Indirect Channels



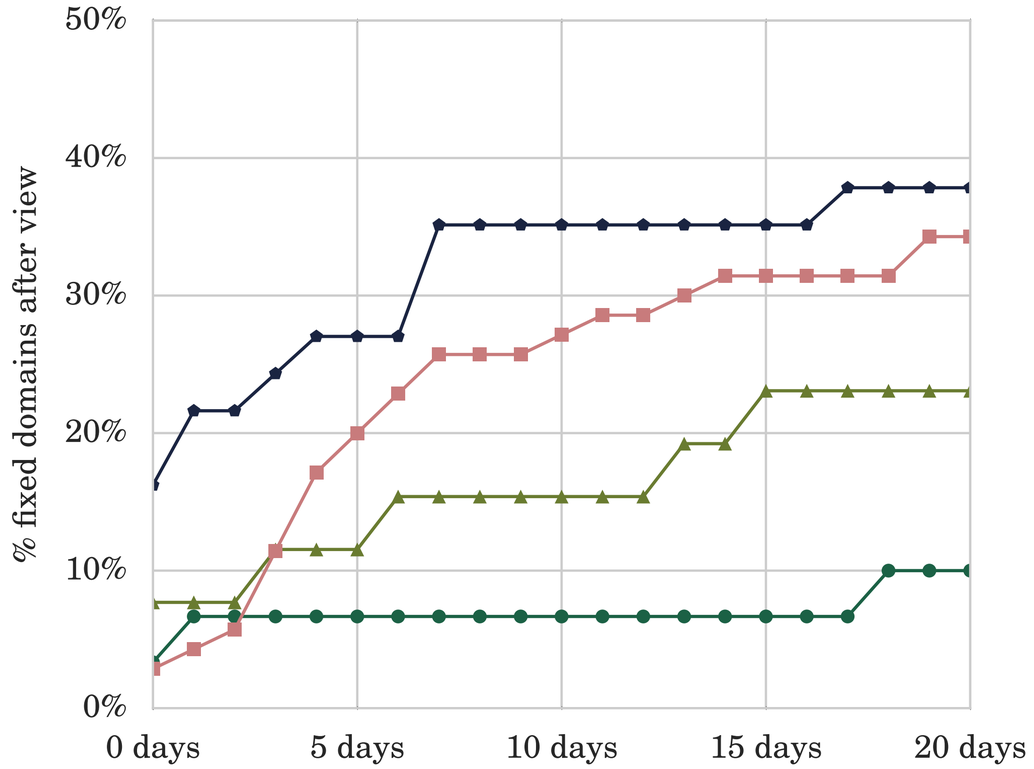
Reachability of Indirect Channels



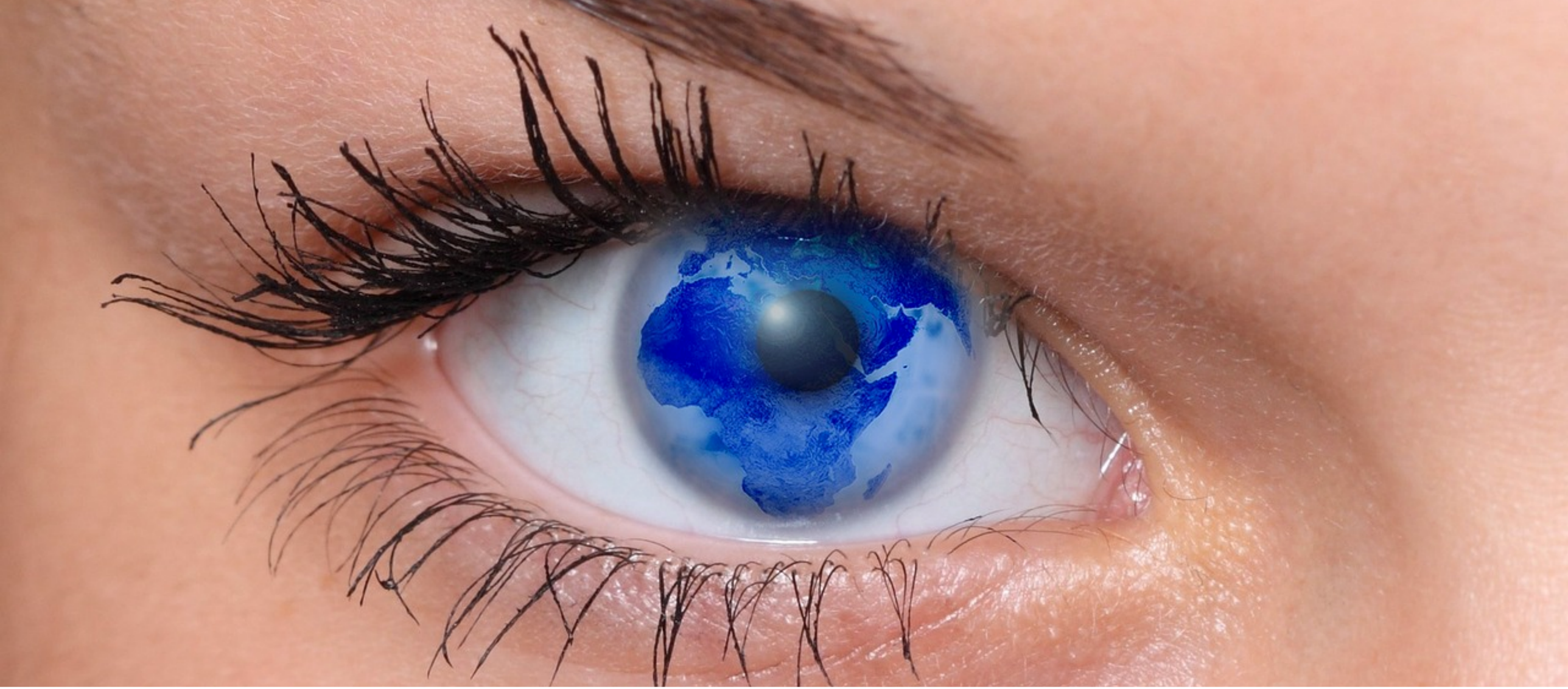
Time to Fix after Report View



WordPress



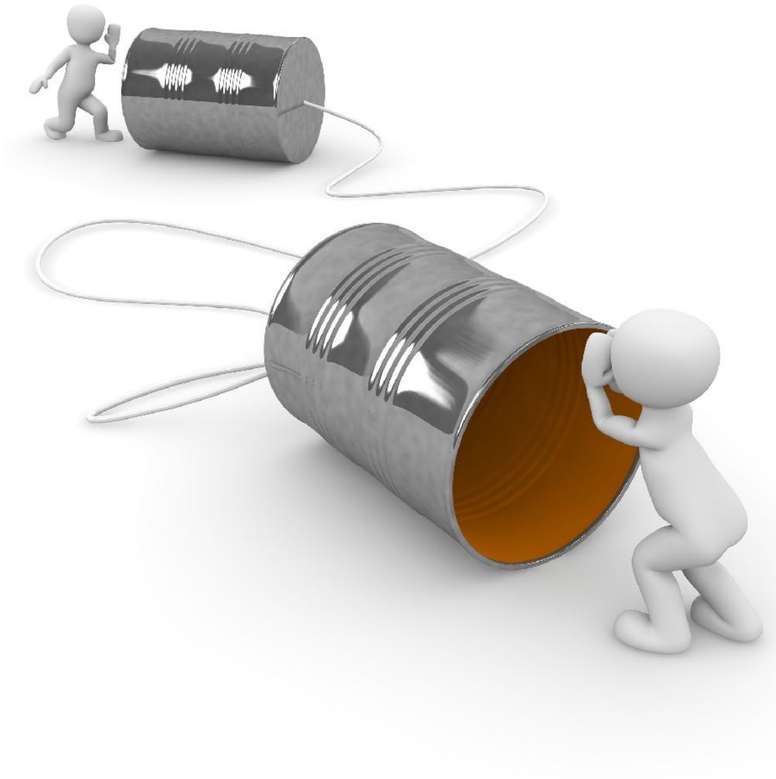
Client-Side XSS



Key Insights and *Follow-Up Questions*

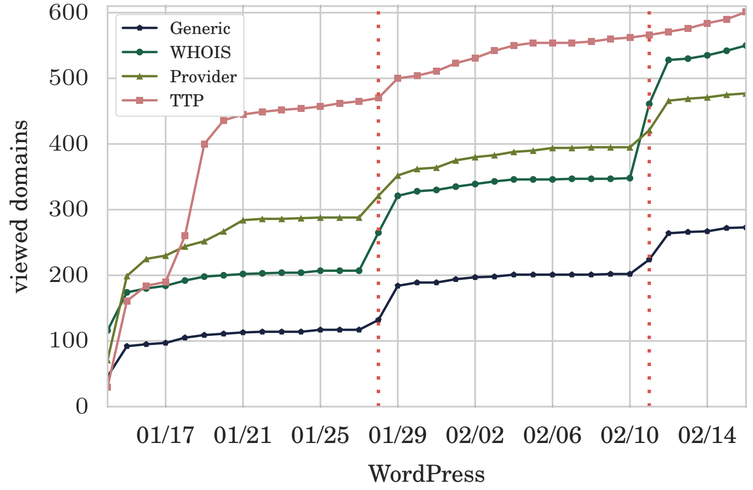
Establishing Communication Channels

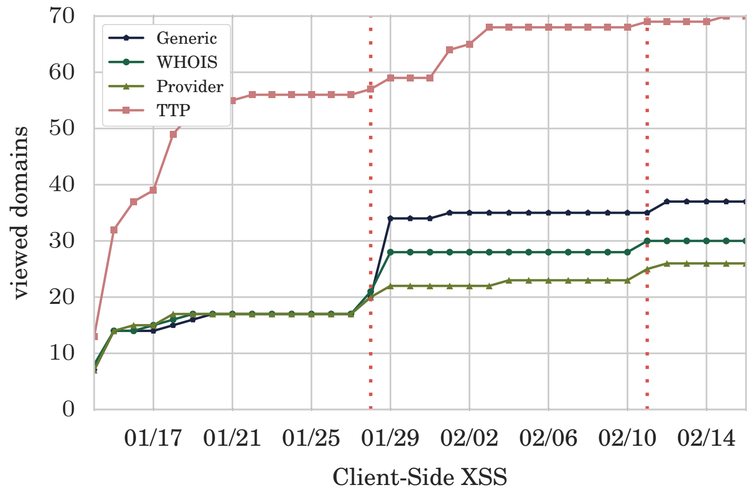
- Direct channels are hard to reach
  - generic emails perform really bad for average Web sites
  - WHOIS helps, but is incomplete (~18.5% without entry)
- Indirect channels are easier to „reach“
  - Often do not forward the information
  - top 5 providers (~25% of domains) did not react
- *RQ: How can the security community come up with reliable means of establishing communication channels between researchers and affected parties?*



Need for Reminders and Time to Fix

- Reminders helped especially for direct channels
- Once report was viewed, fix ratio was ~25-30%
  - after five days, WordPress fix rate equaled control group
- Future notification campaigns should make frequent use of reminders
- *RQ: How can we improve on the fix ratio?*





Sender Reputation

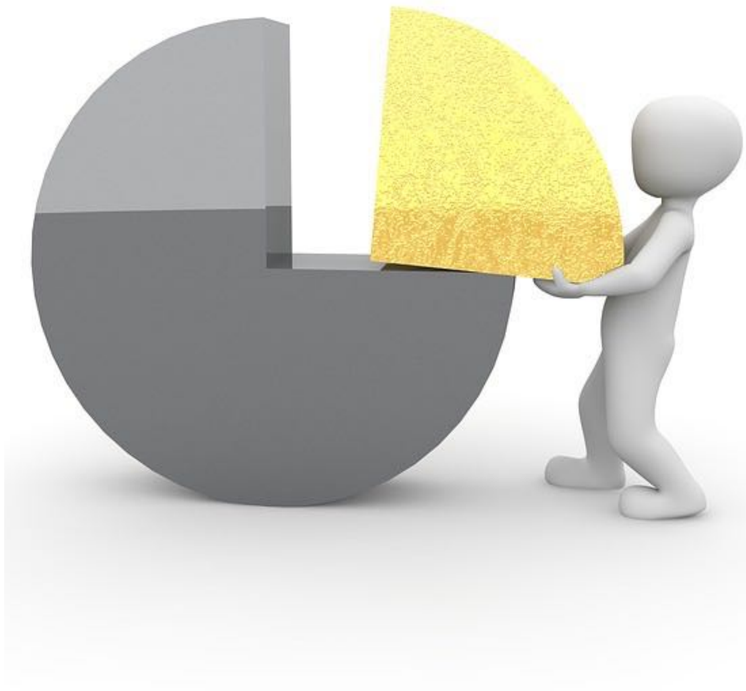
- Previous work [5] found that sender reputation does not matter
- Our work begs to differ
  - German CERT more inclined to forward information
  - Providers more inclined to act upon German CERT info
- *RQ: What is the impact of the sender reputation, especially when using intermediaries, on the success of a notification campaign?*

User Distrust

- Our experiments required users to click a link
  - or send an email with a token
- Community trains users not to click/react
- Notified control group with full disclosure email
  - results only differed significantly for WordPress
  - BUT: performed worse than with links!
- Potential issue in the message length
- *RQ: To what extent does the message tone, content, and length influence the success of notification campaigns?*

Results Generality

- Results appear to be dependent on the domain
  - Providers worked best for Li et al. [23] and Heartbleed [13]
- Even within the same domain, results differ
  - e.g. Generic on WordPress v. Client-Side XSS
- *RQ: Are campaigns more successful if the vulnerabilities gained attention in the media (such as Heartbleed)?*
- *RQ: Does it matter who needs to fix the vulnerability, be it a network admin, Web site developer, or end-user?*



Conclusion

- Conducted first analysis into notifications for Web vulnerabilities at scale
  - two data sets: well-known (WordPress) and previously-unknown (Client-Side XSS) flaws
  - four communication channels: direct (generic emails, WHOIS) and indirect (providers, TTPs)
- Results show statistically significant improvement caused by our campaign
  - WHOIS worked best for WordPress, TTP best for Client-Side XSS
- Overall improvement was unsatisfactory
  - 74.5% of all domains in data set vulnerable at the end of our experiments
- Main problem is reaching administrators in the first place
  - 30% fix rate within five days (WordPress) / 25% (Client-Side XSS)

Thank you!
Questions?