Security Implications of Redirection Trail in Popular Websites Worldwide

Li Chang*
ETH Zürich
changl@ethz.ch

Hsu-Chun Hsiao
National Taiwan University
hchsiao@csie.ntu.edu.tw

Wei Jeng
University of Pittsburgh
wej9@pitt.edu

Tiffany Hyun-Jin Kim
HRL Laboratories
hjkim@hrl.com

Wei-Hsi Lin
National Taiwan University
b02902035@ntu.edu.tw

ABSTRACT

URL redirection is a popular technique that automatically navigates users to an intended destination webpage without user awareness. However, such a seemingly advantageous feature may offer inadequate protection from security vulnerabilities unless every redirection is performed over HTTPS. Even worse, as long as the final redirection to a website is performed over HTTPS, the browser's URL bar indicates that the website is secure regardless of the security of prior redirections, which may provide users with a false sense of security. This paper reports a well-rounded investigation to analyze the wellness of URL redirection security. As an initial large-scale investigation, we screened the *integrity and consistency of URL redirections* for the Alexa top one million (1M) websites, and further examined 10,000 (10K) websites with their login features. Our results suggest that 1) the majority (83.3% in the 1M dataset and 78.6% in the 10K dataset) of redirection trails among websites that support only HTTPS are vulnerable to attacks, and 2) current incoherent practices (e.g., naked domains and www subdomains being redirected to different destinations with varying security levels) undermine the security guarantees provided by HTTPS and HSTS.

1. INTRODUCTION

Web browsing often involves sending a series of URL queries. To reach one's intended destination webpage, a user enters a URL in the browser and the browser sends an HTTP request. Sometimes the initial HTTP request does not directly lead to users' intended destination, and the browser must send multiple HTTP requests sequentially. This process, known as URL redirection or HTTP redirection, indicates that a subsequent request depends on the response provided by its immediate predecessor. For example, the server may redirect a client to another URL, or the user may have to manually navigate to proceed (e.g., by clicking a login link). Such a sequence of HTTP requests/responses is referred to as a *redirection trail* in this paper. Figure 1 shows redirection trails of a fictitious `example.com` site.

It would be ideal to protect every URL that appears on a trail using HTTPS and HSTS, since HTTP provides no protection. Recent studies [15, 16, 22] have investigated the home pages of the most popular websites, and revealed a pattern of unsatisfying, inadequate adoption of HTTPS and HSTS, as well as numerous misconfigurations of HTTPS and HSTS. For example, many sites use HTTP by default, support known vulnerable protocols or ciphersuites, mix HTTP and HTTPS content, set HSTS via HTTP (which has no effect), or support no HTTPS at all—all of which can be easily exploited by an adversary who intercepts the communication.

In this work, we explore the following intriguing questions. Are there weaker links on a redirection trail compared to the security level as perceptually imposed on the home page? Would such a redirection misconfiguration further weaken communication security when it comes to HTTPS adoption? To answer these questions, we developed tools that reconstruct trails and investigate insecure links as users are redirected to reach intended destinations. We applied our tools to the Alexa top one million (1M) websites[1] and recorded the redirection trails. Not all websites provide login capabilities, but logins require secure connections. Hence, we also applied our tools to reconstruct trails to the login submissions on a subset (the top 10K) of the top 1M websites.

[1]The Alexa top 1M list was acquired on June 27^{th}, 2016.

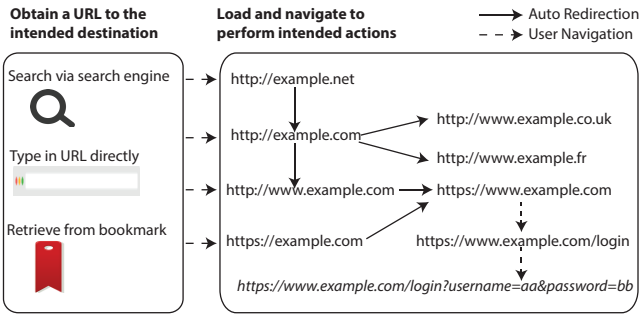


Figure 1: *An example redirection trail of a website. For the best security practice, it is ideal that every URL (from the first to the last request) is protected with HTTPS.*

*This work was done while Li Chang was at Computer Science and Information Engineering Department, National Taiwan University.

WWW 2017, April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4913-0/17/04.
http://dx.doi.org/10.1145/3038912.3052698

Table 1: *Summary of main vulnerabilities found.*

Findings	Sec	Prevalence			Security implications
		%	#	Dataset	
Most insecure redirection trail	5.1	55.0%	512,007/931,135	Top 1M	No protection at all
	5.2.2	39.1%	3,582/9,159	Top 10K	
Accessing HTTPS-only home pages via attackable redirections	5.1	83.3%	119,102/143,022	Top 1M	SSL stripping attack, malvertising, redirection sweep
	5.2.3	78.6%	1,779/2,262	Top 10K	
Optional HTTPS support on home page	5.1	29.7%	276,106/931,135	Top 1M	SSL downgrading to insecure requests
	5.2.3	27.3%	2,499/9,159	Top 10K	
Optional HTTPS support on login page	5.2.3	22.4%	1,023/4,570	Top 10K	SSL downgrading to insecure requests
Login post over HTTP	5.2.1	36.6%	953/2,604	Top 10K	Passwords in plaintext
Security level downgrade on login page	5.2.3	2.9%	42/1,470	Top 10K	SSL downgrading to insecure requests
Inconsistent security levels	6.2	20.5%	190,711/931,135	Top 1M	"Weakest link" exploitation
Inconsistent domain names	6.3	3.6%	33,234/931,135	Top 1M	Confusion with phishing attempt
Low HSTS adaption	6.4	4.5%	41,575/931,135	Top 1M	SSL stripping attack
HSTS misconfiguration	6.4	84.2%	35,008/41,575	Top 1M	SSL stripping attack

From the reconstructed trails, we observed that weak links are introduced by complex redirections and inconsistent configurations across webpages within a website. We also observed that unawareness of the security issues from redirections further jeopardizes 119,102 (83.3%) and 1,779 (78.6%) of websites with HTTPS-only home pages in the top 1M and top 10K datasets, respectively: although the webpage itself is secured by HTTPS, an insecure redirection trail exist to reach the webpage. In fact, our tools revealed that only 23,920 (2.6%) and 478 (5.2%) reachable websites in the top 1M and top 10K datasets remain secure along the entire redirection trail.

The lack of coherent secure redirect methods further complicates security, allowing different websites to practice insecure redirection in various ways. We identified 9,951 distinct redirection patterns from the Alexa top 1M websites, and the most popular 66 redirection patterns cover 85% of the websites. However, only two of these patterns are considered to be secure. Hence, a standard redirection guideline might be required to help prevent various attacks.

Contributions. This paper is the first large-scale study to uncover the current security practices of URL redirection. Our tools can retrieve the URL redirection for a website and construct its redirection trails to identify potential misconfiguration. We analyzed common redirection patterns, identified the most secure and insecure patterns that we observed, and analyzed security misconfigurations along the trails across the Alexa top 1M websites. The main vulnerability findings are summarized in Table 1.

2. BACKGROUND

In this section, we review the main security mechanisms studied in this paper and how URL redirections work.

HTTP over SSL/TLS (HTTPS). HTTPS is a secure version of the HTTP protocol, sending HTTP requests over Transport Layer Security (TLS) or Secure Sockets Layer (SSL). SSL/TLS establishes an encrypted and authenticated end-to-end channel between the web client and server, protecting communication against passive eavesdroppers and active adversaries who can insert, drop, or modify the communication. HTTPS connections are oftentimes visualized in the modern browser's address bar with a lock icon or the color green. In HTTPS, a client authenticates a server by validating the server's certificate, which is a signed document that binds the server's domain name and public key. An invalid certificate (e.g., one signed by an untrusted certification authority, one that has expired, etc.) will trigger a certificate warning displayed in the client browser, which may require explicit user consent to proceed to the site.

HTTP Strict Transport Security (HSTS). Many websites support both HTTPS and HTTP for backward compatibility. Even if a highly-secure website chooses to support only HTTPS, its users may be unaware of this strict security policy, leaving a room for an active man-in-the-middle (MitM) adversary to intercept traffic and launch an *SSL-Stripping* attack [26] that downgrades the client-side communication to HTTP.

HTTP Strict Transport Security (HSTS) [21] is a security policy to prevent SSL-Stripping and is supported by major browsers. When a client sends an HTTP request to an HSTS-enabled domain, the browser will internally upgrade the request to HTTPS before sending it. In addition, users are unable to click through certificate warnings.

HSTS can be enabled either dynamically or statically. For dynamic HSTS, the server sets the `Strict-Transport-Security` header with a time limit `max-age`, which alerts the client that the server wishes to adhere to HTTPS-only communication within the time limit. To prevent misuse, this header is only effective when sent over HTTPS. Thus, dynamic HSTS assumes Trust-On-First-Use—the very first request is secured by HTTPS. On the other hand, static HSTS removes the trust assumption by having browsers ship with a preloaded list of HSTS-enforced websites.

URL Redirection. URL redirection allows a webpage to be accessible via multiple URLs. For example, a website redirects from its naked domain to www subdomain, or from HTTP to HTTPS. Redirections are also used to hide referrers, personalization, moved pages, etc.

Several techniques exist for URL redirections. The server can respond with HTTP status codes (i.e., 300-308), and then the client's user agent will follow the URL indicated in the location field in the HTTP response. The server can also set a meta refresh tag in the returned HTML file or set the `window.location` attribute using JavaScript. Unfortunately, these techniques may be abused to allow invalidated redirects and forwards. More specifically, without any validation or additional methods to verify the certainty of a redirected URL, users may be redirected to malicious sites for phishing attacks, malvertising [24], redirect sweep attacks [27], and URL open redirect [1].

3. PROBLEM MODEL

The core problem of this study is to analyze the *integrity* and *consistency* of redirection trails of websites.

Integrity of redirection trail. When URL redirections occur, securely reaching the requested website depends on the security of *all* connections on the entire trail. In other words, a redirection trail is considered unsafe if a request/response is sent over HTTP. The most ideal redirection is one that correctly uses the HSTS preload list in the browser, ensuring that every packet is sent over HTTPS. Our focus is to assess websites' vulnerable links on their redirection trails. A vulnerable link allows a MitM attacker (e.g., a rouge access point) to violate the *integrity of the redirection trails* as follows. The attacker can prevent the use of more secure links towards the end of the trail, thus intercepting private information such as passwords and cookies [28]. Subtle attackers can even hijack the trail and send the user through adversary-controlled URLs for malvertising [24] or redirect sweep [27] before redirecting the user back to the intended destination. Particularly, redirect sweep attacks leverage the autofill feature of password managers, such that a MitM attacker stealthily redirects the client browser through victim sites to harvest the autofilled passwords.

Since redirections are processed by the client browser in the background and only the last URL and its security status will be displayed in the browser's address bar, it is likely that users are unaware of potential vulnerabilities. For example, when a website's redirection trails are attacked, the users' address bar might display a different HTTPS URL (e.g., phishing), a downgraded HTTP URL (e.g., SSL Stripping), or even the same URL (e.g., redirect sweep). The user cannot detect such attacks unless (s)he knows the correct URL and its security status, and knows how to inspect the hidden redirections.

In this paper, we focus on analyzing the support of HTTPS and HSTS along the redirection trails. We are particularly interested in those websites that attempt to adopt HTTPS but fail to secure every link on its redirection trails. Our methodology can be extended to consider vulnerable links under advanced attacks such as Logjam [11] and DROWN [13] by checking the available ciphersuites and protocol versions.

Consistency of redirection trail. We consider three types of consistencies for URL redirection.

• **Trail convergence**: Multiple initial approaches exist to reach a destination webpage. For example, users may type in only the naked domain (e.g., `a.com`) or www subdomain (e.g., `www.a.com`) with/without HTTPS. Ideally, all these methods would converge to the same secure webpage. On the other hand, a website's trails diverge if different initial approaches for a particular destination take users down different trails (e.g., some arrive at the home page using HTTPS while others reach the home page using HTTP).

• **Domain name consistency**: While switching domain names on a trail might have legitimate usages (e.g., increasing entry points, moving websites, or using third-party services), users may be unable to differentiate these from phishing attacks. It is particularly confusing when the initial URL and the home/login page have different domain names.

• **Configuration consistency**: Security-related configurations should be consistently applied to every webpage on a website's trails. For example, to correctly propagate the HSTS setting to every subdomain, it is recommended [7, 9] that a website set its HSTS header on its naked domain (e.g., `https://a.com`) with `includeSubDomains` and redirect all entrances to its naked domain on the first visit, as setting HSTS solely on a subdomain leaves part of the website vulnerable to attacks.

These redirection inconsistencies not only cause confusion and inaccessibility, but also introduce vulnerabilities, as they imply that some part(s) of the redirection trails may be weaker than the rest.

Table 2: *Four types of redirection trails, their examples, and symbolic representations.*

Trail name	Initial URL example	Symbol
http-domain trail	`http://a.com`	/
http-subdomain trail	`http://www.a.com`	/w
https-domain trail	`https://a.com`	s/
https-subdomain trail	`https://www.a.com`	s/w

4. METHODOLOGY

To assess the security of redirection trails, we focused on reconstructing the trails based on users' typical browsing activities, such as visiting a website's home page or logging in for personalized services. Note that not all websites offer login capabilities (e.g., in the 10K dataset, only 4,570 of the 9,159 reachable websites provide login capabilities), and locating login pages is time-consuming. To work around these challenges, we decided to conduct two studies. The first study uses the Alexa top 1M websites for a lightweight, comprehensive analysis to get a general sense of the redirection trails when visiting a website's home page. The second study dives into a selective sample—the Alexa top 10K websites—for an extensive analysis of the behavior of *all* phases along the redirection trail, including the login process.

To conduct these studies, we developed a crawler and collected the data in October 2016 in Taiwan using Selenium 2.53.6, Firefox 47.0.1, and Python 2.7 libraries.

4.1 Six Phases of A Redirection Trail

To systematically reconstruct common redirection trails for top websites, we divide a redirection trail into six consecutive phases as follows:

Phase 1: Initial URLs. The first phase represents the beginning of a redirection trail. For each website in the Alexa top 1M list, the crawler automatically reconstructs four common redirection trails described in Table 2. (We discuss other potential initial URLs and their security implications in Section 7.1.)

Phase 2: Home Redirect. The second phase contains the automatic redirects from the initial URL. The crawler also records the protocols used and HSTS-related settings. We consider three security levels in this phase: (1) `Secure` if this phase is protected by HTTPS; (2) `Attackable` if at least one HTTP request exists in this phase; (3) `Unreachable` if the home page is unreachable due to DNS address errors, timeout settings on our crawler, and 404 errors.

Phase 3: Home Page. This third phase is the URL located at the end of the automatic redirects from the initial URL. Three security levels are considered: (1) `HTTPS` if the webpage supports only HTTPS connections; (2) `HTTP` if the webpage supports only HTTP connections; (3) `Both` if both HTTPS and HTTP connections are supported.

Phase 4: Login Redirect. The fourth phase represents the redirection from a home page to a login page. Since in practice login redirects are often triggered by user navigation, our crawler simulates this process by automatically

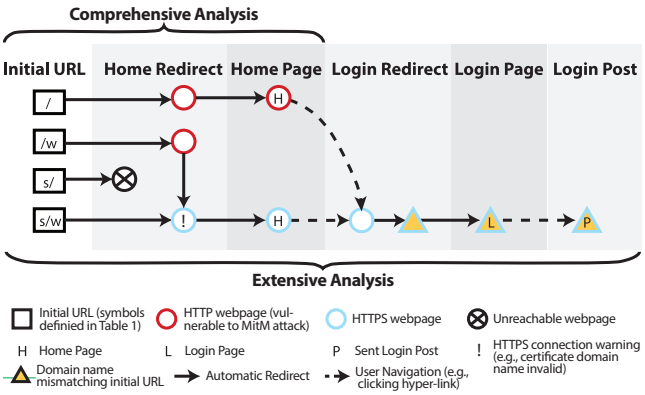


Figure 2: *An illustration of redirection trails along six phases. The first three phases and all six phases are applicable for our comprehensive and extensive analysis, respectively.*

locating login links or forms on the home page. Searching for the login page is a complex task because websites implement it in various ways. Since no standard exists for placing login capabilities, we use several heuristics for balanced performance and accuracy, as discussed in Section 7.2. In addition to the three security levels in Phase 2, we consider two additional security levels: `NoLoginFound`, if the crawler is unable to find a login link; `NoRedirect`, if there is no request sent for rendering the login page.

Phase 5: Login Page. The fifth phase is the webpage that contains a password field (i.e., `<input type="password">`). The login page is the same as the home page if the login form is on the home page. The security levels we considered for this phase are the same as those in Phase 3.

Phase 6: Login Post. This phase represents the security of the login submission request. After finding the login page, the crawler attempts to submit login credentials by automatically filling in a fictitious username and password, and either presses the enter button or clicks all possible buttons. At the same time, the crawler records the traffic sent by the browser and checks whether the username and password can be extracted from the intercepted traffic. We consider the following three security levels in this phase: (1) `HTTPS` if the login is submitted using HTTPS; (2) `HTTP` if the login is submitted using HTTP; (3) `LoginPostFailed` if the submission process is unsuccessful.

4.2 Trail Reconstruction and Visualization

Comprehensive analysis on top 1M website dataset. This analysis is a lightweight scanning process that reconstructs redirections by following the location field in the response header. The crawler also checks whether HSTS is set correctly based on the protocol used and header values. This analysis allows us to obtain the overall picture of the first three phases of the redirection trails, *from an initial URL to a home page* as illustrated in Figure 2, for each website in the Alexa top 1M websites.

Extensive analysis on top 10K website dataset. The extensive analysis extends the scope with human-in-the-loop and advanced web rendering in the login process. Specifically, the crawler reconstructs the full redirection trails, *from an initial URL to a login post* in Figure 2, by rendering webpages in Firefox and simulating user behaviors using a browser automation tool (Selenium) for the Alexa top 10K websites. Thus, the crawler in the extensive analysis can reconstruct redirections implemented via JavaScript, meta tag, etc. We discuss possible differences between the reconstructed redirection trails in these two studies in Section 7.2. To ensure that HSTS is applied, http-domain trails and http-subdomain trails are loaded twice. This analysis also tries to locate the login page and automatically submit a login credential.

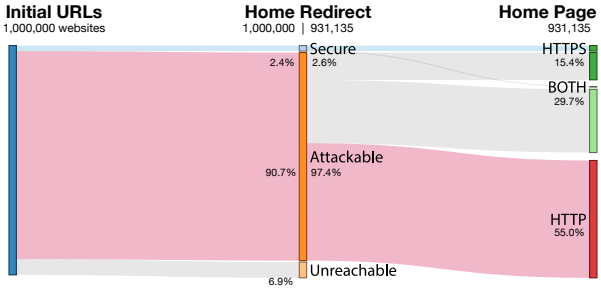


Figure 3: *Sankey diagram for top Alexa 1M dataset. It shows detailed percentages of security levels for each phase with the applicable website count. Red and blue indicate the most insecure and secure trails, respectively.*

Visualization. Figure 2 visualizes a website's four redirection trails, as defined in Table 2. In this example, two of the redirection trails are insecure (i.e., containing red-circled nodes) and thus can be hijacked by a MitM attacker. As for consistency, this website lacks trail convergence because the first trail goes to a different home page than the second and fourth trails. Moreover, the third trail hits an inaccessible URL, and the fourth trail shows domain name inconsistency because it goes through a URL under a different domain name (marked in yellow).

5. REDIRECTION TRAIL INTEGRITY

This section highlights the integrity issues of redirection trails. To examine various combinations of security levels along a redirection trail, we first discuss the comprehensive analysis process on the Alexa top 1M websites and present the redirection trail that we discovered from the Initial URL to the Home Page phase. We then discuss the extensive analysis of Alexa top 10K website dataset that revealed the redirection trail for all phases. To show an overview of websites' security levels in each phase along the trails, a Sankey diagram is used in Figures 3 and 4. For each dataset, we review the statistics of interesting phases before we dive into the trail analysis.

5.1 Comprehensive Analysis

For the top 1M dataset, we concentrated on analyzing the integrity of the first three phases: Initial URL, Home Redirect, and Home Page. After removing unreachable sites, the final sample of our comprehensive analysis is $N = 931,135$. Figure 3 shows the composition of security levels (in percentage) in the first three phases from left to right with the number of applicable websites for each phase.

Phase analysis. As Figure 3 shows, only 24,063 (2.6%) out of the 931,135 reachable websites are categorized as `Secure`; the majority (97.4%) are `Attackable` in the Home Redirect phase. Moreover, in the Home Page phase, more than half (55.0%) of the reachable websites still support HTTP-only connections. Although RFC 2068 recommends that there should be at most five automatic redirections,

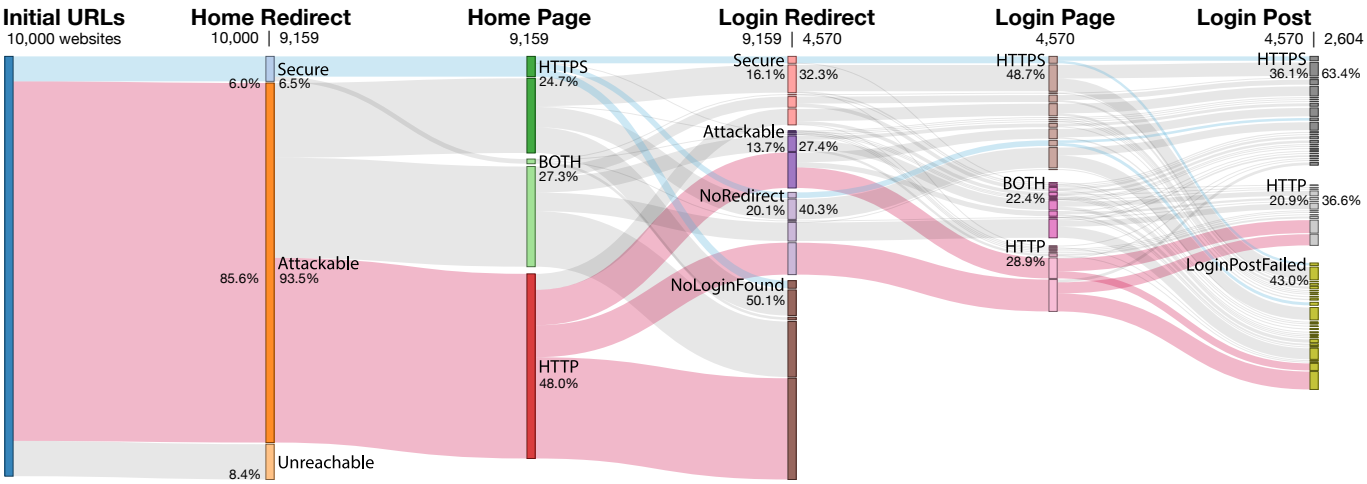


Figure 4: *Sankey diagram of redirection trails among Alexa top 10K websites worldwide. It also shows detailed percentages of security levels for each phase. Red and blue trails indicate the most insecure and secure combinations, respectively.*

3,477 (0.4%) of the reachable home pages on the http-domain trail[2] violate this recommended policy.

Trail analysis. The most secure trail consists of `Secure` in the Home Redirect phase and `HTTPS` in the Home Page phase, and our analysis revealed that only 23,920 (2.6%) of the reachable websites fall into this category. The most insecure trail is `Attackable` (Home Redirect) → `HTTP` (Home Page), and 512,007 (55.0%) of the reachable websites follow this insecure trail. Unfortunately, the most insecure trail among the Alexa top 1M websites is the most common combination in our investigation.

Mismatch between the intended and actual security levels among top 1M websites. Interestingly, 119,102 (83.3%) of the 143,022 websites that support HTTPS-only connections on their home pages allow HTTP requests during redirections, leaving their redirections attackable. This indicates a clear mismatch between the intended security level on the home page versus the actual, hidden security-relevant executions that developers must pay closer attention to. Such an inconsistency reiterates the lack of security guidelines for developers.

5.2 Extensive Analysis

Figure 4 summarizes the redirection trails of the top 10K websites using a Sankey diagram using six phases of a redirection trail. After eliminating websites that were inaccessible by our crawler, we obtained a final sample of $N = 9,159$ for the extensive analysis.

5.2.1 *Phase Analysis*

In the Home Redirect phase, only 595 (6.5%) of the 9,159 reachable home pages are categorized as `Secure`: the majority (93.5%) still transfer their data over HTTP at least once in this phase. In the Home Page phase, almost half (48.0%) of the reachable home pages ($N = 9,159$) do not support HTTPS at all even though this analysis is based on the top 10K popular websites; only 24.7% provide secure connections using HTTPS.

Among the 4,570 websites that provide login pages in the Login Redirect phase, only 1,475 (32.3%) login redirects are considered `Secure`, and 1,254 (27.4%) are `Attackable`. Interestingly, 1,841 (40.3%) of the 4,570 the login pages are the same as the home page (i.e., no redirection). In the Login Page phase, the majority (48.7%) out of 4,570 login pages support only HTTPS, which is higher than the percentage of HTTPS-only home pages. Unfortunately, 1,322 (28.9%) of the login pages support HTTP-only connections, leaving user data unprotected. Among the 2,604 successful login submissions in the Login Post phase, 953 (36.6%) send login credentials (i.e., usernames and passwords) over HTTP in plaintext. Thus, login credentials can be acquired by simply sniffing Internet traffic.

5.2.2 *Trail Analysis*

The most secure trail. Ideally, the most secure trails are highlighted in blue in Figure 4. According to our analysis results, only 478 (5.2%) of the reachable websites fit in this highest standard.

The most insecure trail. The most insecure trails are highlighted in red in Figure 4. According to our analysis, 3,582 (39.1%) of reachable websites provide the most insecure trails: 1,170 (12.8%) for the websites with login capabilities and 2,412 (26.3%) for the websites without login capabilities. Similar to the results in comprehensive analysis, the most insecure trails in the top 10K websites are also the most common, implying a troubling security risk for users.

Security relationship between top 10K and 1M websites. We further examined whether the relative security levels are associated with the popularity of websites. Using 1M websites as the population, we should sample at least 1,534 websites in order to satisfy a confidence level of 95% within the margin of error of 2.5%. We then selected a randomized sample of 2,000 websites (margin of error = 2.19%) to represent the 1M sample, and we used the represented sample to compare with the 10K websites. A Chi-square test of independence was calculated comparing 1M (achieved by 2,000 random websites) and 10K websites in terms of the distribution of websites in (1) the most secure trails (478 websites in 10K, and 66 websites in 1M), (2) neither the most secure nor insecure (5,099 websites in 10K; 942 in 1M), and (3) the most insecure trails (3,582 websites in 10K; 794 in 1M). The test result confirms that the 1M dataset (repre-

[2]Among 3,477 reachable home pages, 2,191, 2,840, and 776 redirect more than five times on the http-subdomain, https-domain, and https-subdomain URLs, respectively.

sented by the 2,000 samples) is more likely to have a higher percentage of insecure trails ($\chi^2(2, N = 10,961) = 19.75$, $p = .00005$), which indicates that the relative security levels in our samples are associated with website popularity.

5.2.3 *Mismatch Between Intended and Actual Security Levels Among Top 10K Sites*

Secure webpages but insecure redirections. Similar to the observations in the comprehensive analysis, we observed insecure redirections compromising secure webpages from the top 10K websites. Among the 2,262 websites that support HTTPS-only connection on their home pages, 1,779 (78.6%) still leave the Home Redirect phase attackable. Specifically, 348 (15.6%) send requests over HTTP while their login pages support HTTPS-only connection. Furthermore, of the 1,428 websites that support HTTPS-only connection on *both* home and login pages, 20 (1.4%) use HTTP to redirect from the home page to the login page. Without checking through every single request and respond step, these vulnerabilities are currently invisible to web users.

Optional HTTPS support on Home Page and Login Page. Although some webpages optionally support HTTPS for performance reasons, such a decision provides limited advantage because users rarely type "HTTPS" explicitly. In our analysis, of the 9,159 reachable home pages, 2,499 (27.3%) support both HTTPS and HTTP. Of the 4,570 reachable login pages, 1,023 (22.4%) support both connection types. These results imply that about a quarter of web developers are *aware* of available security options, but fail to fully secure their websites, leaving them vulnerable to downgrading attacks.

Security downgrade on Login Page. In general, a login page contains sensitive information and must be more secure than other webpages in a website. However, we identified that some websites actually have lower security on login pages than home pages. For example, of 1,470 websites with login pages and supporting HTTPS-only connections on their home pages, a surprising four (0.3%) support HTTP-only connections on their login pages. In addition, 38 (2.6%) provide both HTTP and HTTPS connections for logging in. Such trails leave the door open for attackers to downgrade security levels and acquire credentials.

Supporting HTTPS connections on both home and login pages does not necessarily prevent downgrading attacks unless all the webpages are carefully configured. For example, of the 1,023 login pages supporting both HTTP and HTTPS, 231 (22.6%) have home pages containing an HTTPS link to securely redirect to the HTTPS login pages. However, since their login pages also support HTTP, users who directly access the HTTP login page are left unprotected. These insecure trails to the login page might be accidentally taken by users or exploited by an attacker.

Silent security upgrade on credentials submission. We successfully submitted login credentials on 2,604 of the reachable websites. Eighty (3.1%) of them support an HTTP-only connection on login pages while sending login information through HTTPS. This implementation is vulnerable because an attacker can manipulate the HTTP login page and replace the HTTPS login request with HTTP. Also, it is difficult for the average user to determine whether the login data is encrypted before transmission. Protocol inconsistency on webpages and login submissions also confuses users when determining the behavior of a webpage.

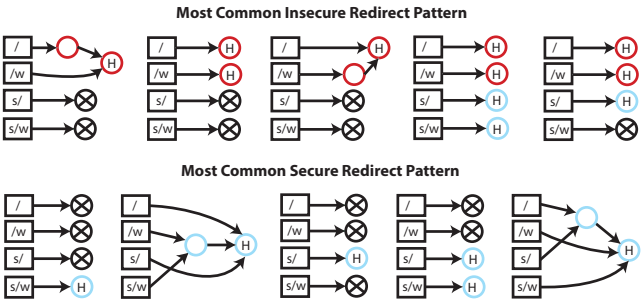
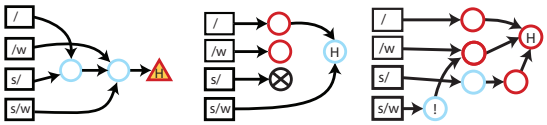


Figure 5: *Most common redirection patterns among Alexa's top 1M websites*



(a) laposte.net **(b)** delta.com **(c)** hilton.com

Figure 6: *Real-world cases of unusual redirections.*

6. REDIRECTION TRAIL CONSISTENCY

Inconsistent redirections not only confuse users but also increase the attack surface. In this section, we present popular redirection patterns among the top 1M websites. We also examine several unusual redirection patterns exhibited by popular websites, and investigate the three types of consistencies for URL redirection (as defined in §3).

6.1 Redirection Pattern

Multiple redirection patterns may exist for a website, based on how it redirects users from different initial URLs to a home page. To simplify this problem, the four common redirection trails as defined in Table 2 are used to construct redirection patterns.

Popular redirect patterns. Among the top 1M websites, 931,135 are reachable through at least one trail and can be classified into 9,951 different redirection patterns. A redirection pattern is a directed graph consisting of the four common redirection trail types as defined in Table 2. Each node represents a unique webpage. If a webpage redirects to another, there is a directed edge between the two nodes. Two redirection trails merge into one when they are redirected to the same webpage. The most popular 66 redirection patterns cover 85% of the 931,135 websites, but only two of them are considered to be secure by using HTTPS-only requests on all four trail types.

Figure 5 shows the most common insecure and secure redirection patterns. The top five insecure patterns (14.7%, 14.6%, 8.6%, 4.9%, and 2.9%) account for almost half of the reachable websites. The top five secure patterns (0.3%, 0.3%, 0.2%, 0.1%, and 0.1%) are observed in 1% of the reachable websites.

Examples of unusual redirections. Figure 6 illustrates three real-world cases. `Laposte.net` is included in the HSTS preload list and sets its HSTS header correctly using the `includeSubDomains` flag. However, as Figure 6a shows, the redirection trail bounces back to HTTP at the end be-

Table 3: *A cross-tabulation of trail inconsistencies with 4 types of the initial URL (highlighted in gray) for 1M websites.*

Case	Initial URL		Relevant websites
	http-domain	http-subdomain	
1	HTTP	HTTPS	1,759
2	HTTPS	HTTP	3,342
3	Unreachable	HTTPS	4,782
4	HTTPS	Unreachable	4,527
	https-domain	https-subdomain	
5	HTTP	HTTPS	27,151
6	HTTPS	HTTP	9,582
7	Unreachable	HTTPS	46,069
8	HTTPS	Unreachable	102,079

cause the last redirection goes to `http://www.laposte.fr`, which is under a different domain.

Another inconsistency is shown in Figure 6b. The URL `http://delta.com` redirects users to `https://www.delta.com` using HTTPS, but `http://www.delta.com` continues using HTTP. Although both initial URLs bring users to the same content, users who start with `http://www.delta.com` are left unprotected and must rely on their own effort to check if the website supports secure connection.

In Figure 6c, `hilton.com` does not automatically adopt HTTPS. The exclamation mark on the first request of the https-subdomain trail indicates that its HTTPS attempt fails due to a "domain name invalid" certificate error because the certificate is under akamai.net. However, no error is encountered on the https-domain trail. This might be attributed to a misconfiguration of the Content Distribution Network (CDN) hosting.

Although all three websites attempt to adopt various degrees of protection, inconsistent configurations across different redirection patterns compromise their security. These diverse redirection patterns and incorrect configurations demonstrate that a standard redirection guideline is needed to prevent potential misuse.

6.2 Trail Convergence

Many websites we examined show inconsistent security practices on the four types of redirection trails. Of the 931,135 websites reachable on at least one trail, 190,711 (20.5%) contain at least one inconsistency mentioned below.

Among the 1M websites, 858,035 websites were reachable on both http-domain and http-subdomain trails, and 1,759 (0.2%) of those reachable websites redirect HTTP to HTTPS on the http-subdomain trail but continue using HTTP on the http-domain one (Case 1 in Table 3); 3,342 (0.4%) redirect HTTP to HTTPS on the http-domain trail but not on the http-subdomain trail (Case 2).

Compared to HTTP trails, HTTPS trails exhibit a higher level of inconsistency. Among the 306,083 websites that are reachable via both https-domain trails and https-subdomain trails in the top 1M websites, 27,151 (8.9%) redirect HTTPS to HTTP on the https-domain trail but continue using HTTPS on the https-subdomain trail (Case 5), while 9,582 (3.1%) show the opposite behavior (Case 6). In addition, 46,049 (9.4%) are accessible on the https-subdomain trail but inaccessible on the https-domain trail (Case 7), and 102,079 (20.8%) of the HTTPS home pages are accessible only through the https-domain trail but are inaccessible on the https-subdomain trail (Case 8).

6.3 Domain Name Consistency

Checking the domain name is a common way for users to identify potential phishing attacks. If a website redirects users to different domain names, users may find it difficult to determine whether changes in domain names are indeed phishing attempts. Among the examined 931,135 reachable websites, 33,234 (3.6%) redirected to a different domain name on at least one trail. Among the 858,035 websites reachable on both http-domain and http-subdomain trails, 3,673 (0.4%) changed domain names on one trail but kept the same domain name on another. Similar situations happened for the 306,083 websites reachable on both https-domain and https-subdomain trails, where 1,770 (0.6%) exhibited domain name inconsistency.

Moreover, prior studies show that a phishing site often puts the domain name that it wants to impersonate at the front of the URL (e.g., as a subdomain of the attacker-controlled domain) [18]. However, we observed an increasing number of third-party services that allow customers to select custom subdomains, and it is common for websites to have legitimate redirections that go to external sites with similar URLs. For example, `store.imgur.com` redirects to `imgur-store.myshopify.com`, and GitHub manages their online store on `github.myshopify.com`. Since no approaches exist to validate the authenticity of these redirections, it may be insufficient for users to simply check the URLs, and additional information may be needed to verify whether users land at the intended website. In addition, websites that change domains along a redirection trail (e.g., Figure 6a) must be cautious when applying HSTS; both domains should be set correctly to ensure the validity of HSTS.

Table 4: *Further investigation of HSTS practices on 41,575 websites that set HSTS. Each numerical entry indicates the percentage of the websites that satisfy the HSTS practice condition as indicated in the first column. The symbols on the first row follow the definition from Table 2, and* ALL *indicates the overall percentage of the websites that satisfy the condition from any of the four trail types.*

	/	/w	s/	s/w	*ALL*
HSTS found	78.5%	74.6%	83.5%	70.8%	100%
HSTS found but inoperative	41.0%	31.1%	17.2%	9.4%	60.1%
`incSubDomains` not set	63.3%	59.4%	67.7%	55.3%	80.3%
Set HSTS over HTTP	9.9%	10.4%	0.6%	0.3%	11.1%
HSTS not on HTTPS domain	40.3%	51.9%	13.9%	54.3%	67.7%

6.4 Configuration Consistency

Among the reachable websites in the 1M dataset, only 41,575 (4.5%) used Strict-Transport-Security in their headers. Even worse, among the 41,575 websites that set HSTS on at least one trail, only 6,567 (15.8%) correctly implemented HSTS on all four trails that we checked.

We also observed inconsistent HSTS settings across redirection trails, as Table 4 summarizes: (1) *HSTS found* indicates that the `Strict-Transport-Security` header field is found in at least one response along the redirection; (2) *HSTS found but inoperative* indicates that HSTS is found but is invalid due to misconfigurations; (3) *incSubDomains not set* indicates that the `includeSubDomains` flag is missing in the HSTS header field; (4) *Set HSTS over HTTP* indicates that all of the HSTS header fields are set on HTTP responses; (5) *HSTS not on HTTPS naked domain* indicates that HSTS is not set on an HTTPS naked domain response.

An interesting observation is that more websites failed to set HSTS on http(s)-domain trails compared to http(s)-subdomain trails, as shown in *HSTS found but not working*. This type of failure is mainly cased by redirecting `http(s)://a.com` to `https://www.a.com` while only setting HSTS on the latter. Even worse, 4,615 (11.1%) of 41,575

websites set HSTS over HTTP connections, which indicates that HSTS will not function at all.

7. DISCUSSION

In light of the findings presented in Sections 5 and 6, we discuss possible approaches and challenges to ensure integrity and consistency of redirection trails.

7.1 Potential Mitigation Approaches

Encouraging HTTPS adoption. Some Internet entities, including Google [3] and Let's Encrypt [5], have been actively promoting HTTPS. Also, most browsers support HTTP/2 (the latest revision of the HTTP protocol) *only* over TLS, and all publicly-accessible US federal sites are required to support HTTPS-only access with HSTS by the end of 2016. However, as of February 2017, the HTTPS adoption rate among US federal sites is 73% [6], and the overall HTTPS adoption rate on the Internet is 52.8% [8]. Further research is required to incentivize HTTPS adoption.

Strict redirection policy. Similar to HSTS, a strict redirection policy can be employed with support from both clients and servers. For example, a server can indicate its policy to decline any redirection to a non-HTTPS domain or to a different domain name. The browser can then display warnings when it detects any redirection that attempts to break integrity or consistency. However, one major concern of introducing any new security policy is that the attacker can still prey on its low adoption rate and potential misconfigurations, similar to how HTTPS and HSTS are currently.

Proactive replacement. Browsers can actively suggest (e.g., using the autocomplete feature) the most secure trail when the site is available under multiple URLs. For example, the HTTPS Everywhere browser extension [4] can replace an HTTP request URL with the HTTPS request based on crowd-sourced mapping rules.

To avoid mixed content blocking, every resource (e.g., image and JavaScript) on an HTTPS webpage should also be loaded via HTTPS. However, upgrading a legacy website with hundreds or thousands of HTTP resources can be troublesome. Hence, a server can add, in every response header, a Content Security Policy (CSP) directive called `upgrade-insecure-requests` [10], which instructs the client's user agent to automatically rewrite all HTTP URLs on the webpage to HTTPS. To mitigate downgrading attacks, however, HSTS is still required.

7.2 Limitations of Our Approach

In this section, we acknowledge shortcomings of this work and discuss how we plan to address them in future work.

Obtaining initial URLs. We assumed that users enter a URL with a domain name when they first visit a website. However, users might access websites in different ways (e.g., bookmarks, search engine, email links, etc.) and their browsing habits might introduce additional weak links to a redirection trail. For example, cache entries in the browser or search engines, autocomplete, and search ranks may let users select an insecure HTTP version even though the HTTPS version exists.

To investigate how search engines might prioritize search results, we conducted a preliminary study to collect the first entry returned by Google when searching for each of the top 10K websites. We found that 40% of the obtained search results have an HTTP prefix even though these websites support HTTPS. This vulnerability guides users to the insecure connection even though they can avoid it. Our next step is to expand this preliminary study and investigate users' browsing habits, emphasizing how users navigate to a website and the security implications.

Locating login. We observed that many of the possible authentication entrances contain visual instructions, mostly in text, to guide users. Developers also leave information in their code to indicate possible authentication entrances. To balance performance and accuracy, we used several heuristics to locate the login pages.

Given a link, we checked whether the home page contains a password field. If no password field existed, we removed invisible elements and collected elements with specific tag names (e.g., `<a>`).[3] After collecting the web elements, we retrieved the information from their attributes (e.g., `id`, `href`, `name`, and `text`). By focusing on elements with more attributes containing matching keywords (e.g., login, logon, log-in, 登入, iniciar sesión, etc.), we avoided single sign-on logins. To observe whether web elements would lead to a login page, our crawler clicked on possible login entrances, and searched for login components in all elements that might appear after the action (i.e., dynamically rendered Document Object Model elements, a pop-up alert, or the redirected page). To speed up the process, we used the first identified login page.

To cross-validate the results, we manually inspected 50 randomly-selected websites from the Alexa top 10K websites and considered the results as the ground truth. After excluding three that were inaccessible, we then computed the performance of our heuristic approach for locating login pages, and the validation confirmed that our approach has a high (100%) precision rate (i.e., all identified login pages are indeed login pages) and a high (88%) recall rate (i.e., this approach locates most of the login pages). To improve the accuracy and speed of our heuristics-based approach, we plan to incorporate the heuristics proposed in prior work (e.g., searching for login buttons in specific regions [30]) and perform a static analysis on JavaScript code and HTML objects before dynamically rendering a webpage.

Different trail constructions. The extensive analysis uses browsers to render the websites, and therefore is capable of handling additional details. In comprehensive analysis, however, we reconstructed redirect trails with location fields in the HTTP response headers for efficiency, neglecting redirections using JavaScript, HTML meta header, etc. This tradeoff, however, may cause the reconstructed trails of the two analyses to be different.

To measure the validity of the comprehensive analysis results, we randomly sampled 2,000 websites from the Alexa top 1M websites, and compared the redirection trails reconstructed using the extensive analysis to those reconstructed using the comprehensive analysis. The results show that our comprehensive analysis approach failed to fully reconstruct the http trails in 84 (4.6%) out of the 1,845 websites that were reachable on the http trails, due to HTTPS redirection failure, redirection path mismatch, and location service set-

[3]Other tag names (e.g., `<div>` and `<input>`) are also feasible. However, we only kept `<a>` in the latest data collection since login functions are more frequently found in `<a>` elements.

ting mismatch. Our tool also failed to fully reconstruct the https trails in 84 (4.9%) out of the 1,705 websites that were reachable on the https trails, mainly due to the additional path information (after the domain name in URL) provided in the extensive analysis. One future direction is to improve the accuracy of our lightweight crawler while minimizing the performance overhead.

The impact of CDN and web hosting. Among the 10K websites we crawled, more than 1,000 of them triggered certificate warnings. The most common warning message was "certificate common name invalid", due to mismatching domain names. Our manual inspection revealed that a large fraction of these warnings are due to CDN hosting (e.g., Akamai) or web hosting (e.g., GitHub),[4] and the error-triggering certificate is issued to the hosting domain [25]. Such false warnings have a negative impact on web security because users might adapt to ignore warnings.

Another subtle issue with CDN is that a client has no information on whether the connection between the CDN and the actual server is secure. For example, CloudFlare's Flexible SSL option only secures the connections between a client and CloudFlare, but not between CloudFlare and a server. An interesting future work is to investigate redirection integrity and consistency issues introduced by CDN and web hosting. Our current study, however, ignores certificate warnings from HTTPS, and therefore our result may be marginally more optimistic than the reality.

8. RELATED WORK

This work investigates the integrity and consistency of redirection trails of the most popular websites. In this section, we review attacks related to URL redirection and known issues for deploying HTTPS.

Attacks related to URL redirection. Automatic redirection can facilitate attacks like URL open redirect [1], malvertising [24], and redirect sweep [27]. URL open redirect [1] is a web application vulnerability that allows an attacker to redirect a vulnerable webpage to another of the attacker's choice. URL open redirect can be used with phishing attacks to make the phishing URL appear to be originating from a legitimate website. Malicious advertising (malvertising) heavily leverages automatic redirects to trick users into downloading malware or to generate click fraud in pay-per-click advertising campaigns. Li et al. [24] studied malvertising redirection chains and identify features that can help distinguish malicious ads from good ones. In a redirect sweep attack [27], the attacker aims to harvest autofilled passwords by stealthily redirecting the client browser through victim sites.

Our work focuses on investigating the redirection trails in popular websites and reveals that many redirection trails can be hijacked by a MitM attacker or may confuse users when making security critical decisions (e.g., phishing attacks, whether the certificate error is caused by real attacks or misconfigurations) due to inconsistent configurations.

Known issues when deploying HTTPS. One major obstacle to HTTPS deployment is its unsatisfactory adoption rate [22, 27]. HTTPArchive [2] shows a slightly upward trend of HTTPS requests among the top 1M sites on Alexa, but the fraction of HTTPS requests is only 34% as of October 2016. Clark and van Oorschot [14] summarize the common security issues in TLS and HTTPS. Kranch and Bonneau [22] studied HSTS adoption of the top domains and found several subtle HSTS configuration errors, such as setting HSTS over HTTP. Although using HTTPS everywhere can secure the redirection trails, the low adoption rate suggests the need to explore alternative solutions.

Regarding the security of login pages, Silver et al. [27] manually examined the Alexa's top 500 sites in 2013 and found that among 408 sites with login forms, 48% loaded the login page using HTTP, and 30.15% loaded and submitted the login form using HTTP. Stebila [29] also found that among 125 sites providing the login functionality, 19 of them loaded the login page using HTTP but posted the login form to HTTPS. However, there is no indication for the users as to whether the form will be submitted via HTTP or HTTPS. Our study also observes a large-scale problem of insecure logins. Moreover, the security level of login pages can be worsened due to insecure redirection trails.

Websites that support HTTPS may still be vulnerable if they use weak cryptographic suites [8, 19, 23]. Lee et al. [23] evaluated cryptographic strengths (e.g., key size, protocol versions, and ciphers) over 19,000 SSL/TLS servers and found a large fraction of surveyed sites supporting weak or even broken cryptography. Since the security of TLS depends on certificate validity, increasing attention has been paid to the certificate authority model and the certificate verification process [15, 17, 20]. Huang et al. [20] found that 0.2% of SSL connections to Facebook are susceptible to MitM attacks. Akhawe et al. [12] studied the click-through problem of certificate warnings and suggest that browsers should minimize false or low-risk certificate warnings to conserve users' attention. Our methodology can be extended to identify vulnerable links under advanced attacks such as Logjam [11] and DROWN [13] by checking the available ciphersuites and protocol versions.

9. CONCLUSION

A website's security level should not be evaluated based on the final stage of its redirection trail; instead, the entire redirection trail must be considered. In this study, we presented the surprisingly insecure statuses of websites during the URL redirection process; only a very small portion of websites met the ideal security level where every redirection is performed over HTTPS. To prevent security misconfigurations during this process, coherent redirection guidelines or protocols may be needed to help implementers avoid misusing the security services and minimize (if not remove) the attack surface. We hope our findings can motivate further research and best practices to improve the security of redirection trails.

Acknowledgments

This research was supported in part by the Ministry of Science and Technology of Taiwan (MOST 105-2633-E-002-001), National Taiwan University (NTU-105R104045), and Intel Corporation.

[4]Some web hosting sites, such as GitHub and WordPress, allow users to use a custom domain name for the website (e.g., `a.com` instead of `a.github.io`). However, GitHub does not support HTTPS for custom domain as of February 2017.

10. REFERENCES

[1] CWE-601: URL Redirection to Untrusted Site ('Open Redirect'). https://cwe.mitre.org/data/definitions/601.html.

[2] Http archive. http://httparchive.org/.

[3] HTTPS at Google. https://www.google.com/transparencyreport/https/.

[4] Https everywhere. https://www.eff.org/Https-everywhere.

[5] Let's Encrypt. https://letsencrypt.org/.

[6] Pulse: How federal government domains are meeting best practices on the web. https://pulse.cio.gov/.

[7] RFC6797 HTTP Strict Transport Security (HSTS). https://tools.ietf.org/html/rfc6797.

[8] Survey of the SSL implementation of the most popular web sites. https://www.trustworthyinternet.org/ssl-pulse/.

[9] The HTTPS-Only Standard. https://https.cio.gov/hsts/.

[10] Upgrade Insecure Requests (W3C Candidate Recommendation). https://www.w3.org/TR/upgrade-insecure-requests.

[11] D. Adrian, K. Bhargavan, Z. Durumeric, P. Gaudry, M. Green, J. A. Halderman, N. Heninger, D. Springall, E. Thomé, L. Valenta, B. VanderSloot, E. Wustrow, S. Zanella-Béguelin, and P. Zimmermann. Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice. In *ACM CCS*, 2015.

[12] D. Akhawe, B. Amann, M. Vallentin, and R. Sommer. Here's My Cert, So Trust Me, Maybe? Understanding TLS Errors on the Web. In *WWW*, 2013.

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

[14] J. Clark and P. C. van Oorschot. SoK: SSL and HTTPS: Revisiting past challenges and evaluating certificate trust model enhancements. In *IEEE Symposium on Security and Privacy*, 2013.

[15] Z. Durumeric, J. Kasten, M. Bailey, and J. A. Halderman. Analysis of the HTTPS Certificate Ecosystem. In *ACM Internet Measurement Conference*, 2013.

[16] Z. Durumeric, E. Wustrow, and J. A. Halderman. ZMap: Fast Internet-wide Scanning and Its Security Applications. In *USENIX Security Symposium*, 2013.

[17] R. Holz, L. Braun, N. Kammenhuber, and G. Carle. The SSL Landscape: a Thorough Analysis of the x.509 PKI Using Active and Passive Measurements. In *ACM IMC*, 2011.

[18] J. Hong. The Current State of Phishing Attacks. *Communications of the ACM*, 55(1):74–81, 2012.

[19] L.-S. Huang, S. Adhikarla, D. Boneh, and C. Jackson. An Experimental Study of TLS Forward Secrecy Deployments. *IEEE Internet Computing*, 18(6):43–51, 2014.

[20] L.-S. Huang, A. Rice, E. Ellingsen, and C. Jackson. Analyzing Forged SSL Certificates in the Wild. In *IEEE Symposium on Security and Privacy*, 2014.

[21] C. Jackson and A. Barth. ForceHTTPS: Protecting High-Security Web Sites from Network Attacks. In *WWW*, 2008.

[22] M. Kranch and J. Bonneau. Upgrading HTTPS in Mid-Air: An Empirical Study of Strict Transport Security and Key Pinning. In *NDSS*, 2015.

[23] H. Lee, T. Malkin, and E. Nahum. Cryptographic Strength of SSL/TLS Servers: Current and Recent Practices. In *ACM IMC*, 2007.

[24] Z. Li, K. Zhang, Y. Xie, F. Yu, and X. Wang. Knowing Your Enemy: Understanding and Detecting Malicious Web Advertising. In *ACM CCS*, 2012.

[25] J. Liang, J. Jiang, H. Duan, K. Li, T. Wan, and J. Wu. When HTTPS Meets CDN: A Case of Authentication in Delegated Service. In *IEEE Symposium on Security and Privacy*, 2014.

[26] M. Marlinspike. New Tricks for Defeating SSL in Practice. In *BlackHat*, 2009.

[27] D. Silver, S. Jana, E. Chen, C. Jackson, and D. Boneh. Password Managers: Attacks and Defenses. In *USENIX Security*, 2014.

[28] S. Sivakorn, I. Polakis, and A. D. Keromytis. The Cracked Cookie Jar: HTTP Cookie Hijacking and the Exposure of Private Information. In *IEEE Symposium on Security and Privacy*, 2016.

[29] D. Stebila. Reinforcing Bad Behaviour: the Misuse of Security Indicators on Popular Websites. In *ACM OZCHI*, 2010.

[30] Y. Zhou and D. Evans. SSOScan: Automated Testing of Web Applications for Single Sign-On Vulnerabilities. In *USENIX Security*, 2014.