The Threat of SSL/TLS Stripping to Online Voting

Anthony Cardillo and Aleksander Essex

Department of Electrical and Computer Engineering
Western University, London, ON, Canada
`{acardill,aessex}@uwo.ca`

Abstract. In many real-world deployments of online voting, Transport Layer Security (TLS) represents the primary (and in some cases *only*) line of defense against network based man-in-the-middle attacks that can steal voter credentials and modify ballot selections. In this paper we examine online voting in the context of *TLS stripping attacks*, which exploit the situation where a voter types or clicks a URL of the form `example.com` or `http://example.com`. Despite the widespread availability of effective protections, we present a study of voting-related websites finding the overwhelming majority are vulnerable to TLS stripping to some degree, with most offering no explicit protection at all.

1 Introduction

Recall the last time you logged in to your social network account, an online retailer, financial institution, or even online election. You should have seen a lock icon in the address bar of your browser. Was it there? Did you check? Suppose the icon was missing. Would you notice? And if you did notice, what would you attribute it's absence to? Maybe you misunderstood the security indicators. Maybe the server was mis-configured. Or maybe you were the victim of a cyber attack that prevented your browser from initiating a secure connection using transport-layer security (TLS), enabling attackers to monitor and/or modify the content you send and receive.

TLS stripping[1] [15, 4] is a network based man-in-the-middle attack which suppresses or *strips* TLS from a communication channel. The attack is made possible when a user types or clicks a non-HTTPS URL of the form `example.com` or `http://example.com` (as opposed to `https://example.com`). This instructs the browser to make an insecure request over HTTP instead of its encrypted and authenticated counterpart, HTTPS. A well configured TLS-enabled server would typically respond to such a request by directing the client to request the resource over HTTPS instead. A man-in-the-middle can intercept and suppress this response, and continue communicating with the client over HTTP. Since the client requested an HTTP connection to begin with, the missing TLS redirect

[1] More commonly known as *SSL Stripping*, it was originally named after the now-deprecated Secure Sockets Layer (SSL) protocol.

goes unnoticed, and the man-in-the-middle is free to observe and modify any data exchanged in the interaction.

Although TLS stripping is a significant and effective threat to online security generally, it has only briefly been considered in the context of elections [25, 10]. In this paper we conducted a study of the adoption of TLS stripping mitigations by election websites. We found election websites systematically lagging in the adoption of industry best-practices. We examined an international cross-section of over 100 election, vendor, and voter registration websites and found 98% were vulnerable to TLS stripping to some degree, with 84% providing no mitigation at all. We also found a number of servers with serious TLS vulnerabilities, which we disclosed to the affected organizations.

2 Motivation

Online voting is a unique use case of the web, with a confluence of factors that increase the severity of TLS stripping attacks. The factors that warrant further study of this topic are as follows.

Critical Infrastructure. Online voting websites must conform to higher cybersecurity standards. Elections are increasingly being recognized as critical infrastructure. In 2017, for example, the U.S. Department of Homeland Security designated elections systems as critical infrastructure under the Government Facilities Sector.[2] As such, it is important to understand how online voting websites are performing relative to industry security standards.

Secret Ballot, Secret Tampering. As we discuss in Section 3.1, TLS represents the main (and in some cases *only*) line of defense against network based man-in-the-middle attacks. Unlike other online settings (e.g., social media, online banking, etc.), an attack stealing voter credentials or modifying ballot selections can be more difficult to detect and correct due to ballot secrecy requirements, making the impact of a man-in-the-middle attack more severe in comparison.

Communicating URLs to Voters. The transient nature of elections often makes communicating the URL of an election website to voters a weak link in a literal and figurative sense. As discussed in Section 3, we found numerous cases of election officials and candidates explicitly directing voters to use HTTP URLs.

Systematic Lack of Best Practice Adoption. Effective mitigations for TLS stripping have long been adopted by leading websites like Google, Amazon, Facebook, etc. If elections truly are to serve as critical infrastructure, they must provide a degree of web security that is *no worse* than current industry practices. As our findings in Section 4 show, however, almost every election-related website we examined is vulnerable to TLS stripping.

[2] https://www.dhs.gov/government-facilities-sector

Barriers to Adoption. Unlike other vulnerabilities that can be resolved with software updates, mitigations to TLS stripping are opt-in, meaning the organization has to be aware of the attack, be aware of the mitigations, and take action to apply them. As we discuss in Section 5, election agencies and vendors face a variety of obstacles in this regard.

3 The Threat of TLS Stripping to Online Elections

Transport Layer Security (TLS) is a standardized group of cryptographic protocols for secure network communication [18, 19], providing confidentiality, integrity and authentication of network communications at the application layer.

Attacks against TLS typically involve directly attacking components, typically by exploiting vulnerabilities in the software implementation, the cryptographic primitives, or the protocol itself. Recent examples include key-recovery attacks on export strength ciphersuites [23, 1], buffer over-read vulnerabilities [8], insufficient public-key validation [7, 22], and eavesdropping attacks exploiting either TLS compression (BEAST and CRIME) and padding oracles [11]. We refer the reader to the surveys of Clark and van Oorschot [4] and Sheffer et al. [20] for a systematic study of known attacks.

TLS stripping differs from these approaches. It does not exploit a concrete TLS vulnerability, but rather prevents a TLS connection from being established in the first place.

3.1 Role of TLS in Online Elections

Suppose a voter types the URL of an election website, `voting-site.com`. The server responds with the login page, and the voter enters their user id and password. The browser sends an HTTP `POST` over TLS to the login page, e.g., `https://voting-site.com/login.php` with the following contents:

```
auth_id=1234&auth_pwd=123456&submit=Login.
```

Most login pages do not use encryption or authentication inside of the TLS connection. Without it, or with a vulnerable implementation, a man-in-the-middle can directly recover the voter's credentials from the contents of the login `POST`. Now suppose the voter marks a vote for a candidate, Alice, and clicks on the *Cast* button. The browser makes an HTTP `POST` to the cast ballot page `https://voting-site.com/cast.php` with the contents:

```
president=Alice&election_id=US_2020&submit=Confirm
```

Since no other encryption or authentication exists on the *POST* contents, a man-in-the-middle can arbitrarily change the voter's selections, e.g., by setting `president=Bob`, or perhaps more simply by swapping candidate names in the ballot HTML.

Application Layer Encryption. Some online voting designers have attempted to mitigate the consequence of TLS based attacks by employing additional cryptographic protections at the application layer. For example, the iVote system in Australia[3] uses client-side Javascript to encrypt `POST` data. The problem remains, however, that the Javascript is delivered to the client over TLS. Teague and Halderman [12] demonstrated the ability to inject vote-stealing Javascript as a result of the use of weak crypographic parameters [23] in the state election of New South Wales in 2014. Even without actively injecting Javascript, Culnane et al. [5] demonstrated the feasibility of passively recovering voter credentials in iVote using brute force methods.

Multi-factor Authentication. Other online voting designers mitigate TLS attacks by employing additional voter authentication factors. Online voting in the Estonian system [21], for example, is done through a special election-specific software application. The approach removes much of the user from the equation by forcing TLS and using a pre-loaded (pinned) certificate to authenticate the election server. Additionally, the Estonian system uses national Electronic ID cards to digitally sign ballots (although Nemec et al. [17] recently demonstrated a major signature forgery vulnerability in the Estonian PKI). While these additional factors add defense in depth with regard to TLS stripping attacks, both the voting application[4] and the electronic ID software[5] are still initially downloaded by the voter in a web browser over TLS.

3.2 TLS Stripping

TLS stripping (originally *SSL* stripping) was introduced Moxie Marlinspike [15] on the observation that most TLS connections occurred only when a user either clicked on an explicit HTTPS link, or when the server sent an HTTPS redirect. As he noted: “Nobody types ‘https://’, or even ‘http://’ for that matter.” And when a user either types or clicks on a URL of the form `example.com` (or `http://example.com`), a TLS-enabled server typically responds by directing the client to request the resource securely over HTTPS instead. The client would then initiate a TLS handshake, after which time the interaction would continue over an encrypted and authenticated connection (See Figure 1). A man-in-the-middle can intercept and suppress this redirect, and continue the interaction with the client over HTTP. The man-in-the-middle can then initiate and maintain its own TLS connection with the server.

Because the client requested an HTTP connection to begin with, the missing TLS redirect, and continued interaction over the insecure channel goes unnoticed, and the man-in-the-middle is free to observe or modify any messages between the client and server (See Figure 2).

[3] `https://ivote.nsw.gov.au`
[4] `https://valimised.ee`
[5] `https://www.id.ee`

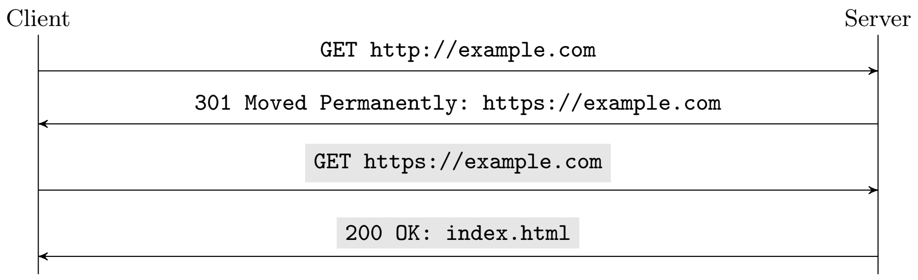


Fig. 1. Normal HTTPS Redirect. When a user types or clicks an HTTP URL, a well configured TLS-enabled server would direct the client to use HTTPS instead, and the interaction would continue over a secure TLS connection (shown in grey).

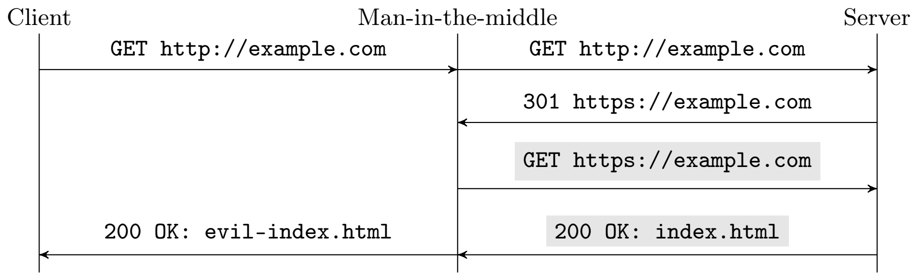


Fig. 2. TLS Stripping Attack. A man-in-the-middle can intercept a request made by a client over HTTP and proxy it to the server via its own separate TLS connection (shown in grey). Since the client never saw the HTTPS redirect, the man-in-the-middle is free to continue the connection over HTTP.

Detecting TLS Stripping. TLS stripping attacks cannot not be reliably detected by users in practice, as they do not generate browser errors or warnings. Instead, they require the user to notice the *absence* of security indicators such as the `https://` in the URL, and the padlock icon in the address bar [6, 2]. Inconsistent and confusing security indicators add to the challenge [3]. Some browsers such as Safari and Edge do not even display the `HTTPS` indicator, and employ a subtle padlock icon (see Figure 3). Due to limited screen resolution, many mobile browsers hide the address bar when scrolling. Even if a user consciously registers the unexpected behavior, research has suggested that voters are not likely attribute it to a malicious cause [16].

3.3 HTTP Strict Transport Security (HSTS)

When TLS stripping was first introduced, there was no mechanism to definitively declare preference for HTTPS, which led to the development of the HTTP Strict Transport Security (HSTS) standard [13]. An HSTS directive can be placed in

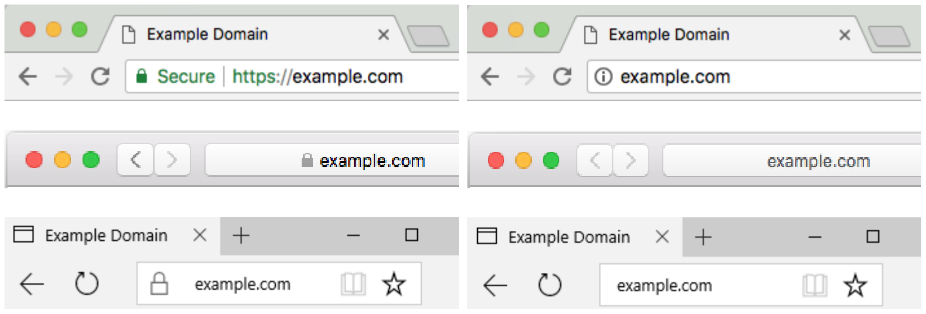


Fig. 3. Attack indicators Across Browsers. Secure connections (left) versus TLS stripped connections (right) in the desktop versions of Chrome (top), Safari (middle), and Edge (bottom).

an HTTP response header allowing a server to advertise its intention to only communicate over HTTPS for all future connections. This includes an expiry period (expressed in seconds), and optional fields to include subdomains, and express consent to be added to the preload list. For example, an HTTP response header containing

```
Strict-Transport-Security: max-age=31536000; includeSubDomains;
                            preload;
```

would direct the browser to store the preference in cache for 1 year, for all subdomains, and would direct browser developers to add the site to the preload list. From this point forward until expiry (or until the user clears the browser's cache), the browser will make all requests to that domain (and any subdomains) over HTTPS—even if the user types or clicks an HTTP link.

Adoption of the HSTS standard has been slow but steady. At the time of writing, 16.8% of the top 150,000 TLS-enabled sites supported HSTS.[6] As of 2015, the US government instituted a policy requiring all executive (i.e., `.gov`) departments and agencies to use HTTPS and enable HSTS.[7]

3.4 HSTS Preload list

HSTS uses a trust-on-first-use model. To be effective in preventing a TLS stripping attack, a user must have previously visited the website in order to have received and stored the server's preference. There are, however, a number of situations where this requirement would not be met, such as when: the user has never visited the site before; the browser, operating system or device was recently upgraded or switched; the user recently cleared their browser's cache; or, the user had not visited the site for an amount of time exceeding the `max-age`.

[6] https://www.ssllabs.com/ssl-pulse
[7] US Office of Management and Budget. Memorandum M-15-13, 2015. https://https.cio.gov

For many websites, and especially election websites, it may not be reasonable to assume a previous visit has recently occurred. In addition, some URLs may never be visited. For example, a visit to `example.com` might redirect to `https://www.example.com`. Since the client never visited `https://example.com`, it will not receive an `includeSubDomains` directive that applies to the entire zone, and any subdomains (e.g., `sub.example.com`) will be left unprotected.

For these reasons, the Chromium security team created a list[8] of HSTS-enabled domains *preloaded* into Chrome, and each user receives this list automatically whenever Chrome is installed or updated. Visiting any domain on this list is done over HTTPS only (even on a first visit), meaning no HTTP to HTTPS redirects are ever made, and therefore no opportunity for TLS stripping exists. Other browsers like Firefox, Opera, Safari and IE/Edge use preload lists based on the Chrome list.

In 2014 the list contained only 233 non-Google domains [14], and additions were handled manually over email. At the time of writing there were 50,000 domains on the list, spanning a wide variety of sites ranging from large companies and government agencies, to small businesses and personal websites, and addition is handled by an automated submission site.[9] Although the `.gov` domains account for less then 1% of the preload list, the DotGov registrar has begun automatically preloading all newly registered `.gov` domains.

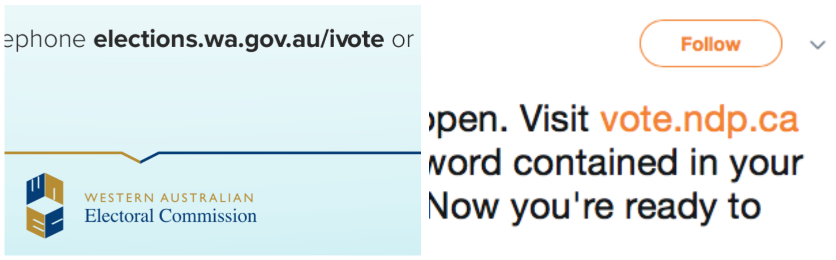
3.5 Communicating Voting Website URLs

One of the major challenges of hosting an online election is communicating authentic URLs to voters. In an effort to increase voter turnout, election officials and campaigns may employ a variety of modes of communication. A vulnerability arises, however, if the voting website does not employ HSTS preloading *and* the user types or clicks on an HTTP URL. As we will see in the following section, this first circumstance occurs in the vast majority of election websites. But how likely is a user to initially visit the voting site over HTTP? User testing is beyond the scope of this study, however we observed numerous situations in which the voter was explicitly directed to the voting site over HTTP (see Figure 4).

For example, during the New Democratic Party of Canada's leadership election in October 2017, we observed numerous HTTP links on social media originating from the accounts of candidates, riding associations, supporters, and even the party itself. In the mailer to voters, the instructions explicitly directed voters to the website via HTTP: "How can I vote online? Type vote.ndp.ca into the address bar of your web browser."

In another example, the Ontario Labour Relations Board held an online strike vote for college teachers in November 2017. Not only was the landing page to the voting site emailed as an HTTP link to voters (`olrb-crto.isivote.com`), the site was not available over HTTPS. Only when voters selected their preferred

[8] https://www.chromium.org/hsts
[9] https://hstspreload.org
[11] https://twitter.com/NDP/status/910253791718645765


ephone **elections.wa.gov.au/ivote** or

pen. Visit vote.ndp.ca
word contained in your
Now you're ready to

(a) Western Australia Election Commission, Poster, 2017. (b) New Democratic Party of Canada, Twitter, 2017

Fig. 4. The Weakest Link. Examples from recent online elections of voters being directed to type [24] or click[11] insecure HTTP URLs.

language were they redirected to a TLS enabled site (on a different domain) to log in.

4 Study of HSTS Preload Adoption in Election-related Websites

This section presents a study of HSTS pre-loading among websites with an election focus. We examined a number of websites, including the online voting sites of specific elections, government agencies responsible for election administration and voter registration, and the websites of companies offering online voting solutions. In total we examined 103 election sites, and chose to focus on Australia, Canada, Switzerland and the United States as countries that have a high concentration of websites pertaining to elections.

Each site was evaluated for the availability of a well-configured TLS configuration, the presence of HSTS headers with a non-trivial max-age, and membership in the HSTS preload list. We comment on any vulnerabilities in the TLS configuration for sites that scored a grade of C or lower on Qualsys SSL Test.[12]

4.1 Election Websites

Only a handful of online elections occur at any time, and many sites stay online only during the polling period. For this study, our goal was to present a diverse (not necessarily complete) sample of voting sites. We examined the TLS configuration of voting sites with active login pages during mid May 2018, and present our findings in Table 1, which revealed minimal protection against TLS stripping attacks.

Of the 10 sites examined, only two were on the preload list. The first is Helios, an end-to-end verifiable internet voting scheme (E2E-VIV) [9], which has been

[12] https://ssllabs.com/ssltest

Table 1. Snapshot of HSTS in Online Voting Websites

Domain	Election	TLS	HSTS	Preload
`ivote-cvs.elections.wa.gov.au`	Western Australia State 2017	●	○	○
`esc-vote.com/acm2018`	ACM General Election 2018	●	○	○
`evoting.ch`	Swiss Post E-Voting	●	●	●
`heliosvoting.org`	Helios	●	●	●
`intvoting.com/OntarioPC`	Ontario PC Leadership 2018	●	○	○
`innskraning.island.is`	Iceland Citizen E-referendum	●	○	○
`ivote.nsw.gov.au`	New South Wales State 2015	●	●	○
`olrb-crto.isivote.com`	Ontario Labour Board 2018	○	○	○
`valimised.ee`	Elections Estonia	●	○	○
`vote.ndp.ca`	NDP Canada Leadership 2017	●	○	○

used to conduct elections for organizations such as the International Association of Cryptologic Researchers (IACR) and the Princeton Graduate Students' Association. We contacted the Helios maintainers in October 2017 asking them to consider adding `heliosvoting.org` to the preload list. They did, and (to our knowledge) became the first online voting site on the preload list. The second is Swiss Post's E-Voting site, which has been used by the cantons of Fribourg and Neuchâtel, and will be used for the first time in upcoming elections in the cantons of Basel-Stadt, Glarus, and Thurgau. We observed inbound links to the E-Voting site originating from the canton sites. We examined these sites and found none were using HSTS or preloading, and some even did not offer TLS (see Table 2). Voters attempting to visit via `evoting.ch` via the canton websites would, therefore, still be vulnerable to TLS stripping.

Table 2. HSTS in Swiss Cantons Using Online Voting

Domain	Canton	TLS	HSTS	Preload
`bs.ch`	Basel-Stadt	○	○	○
`fr.ch`	Fribourg	●	○	○
`gl.ch`	Glarus	◐[a,b]	○	○
`ne.ch`	Neuchâtel	○	○	○
`tg.ch`	Thurgau	◐[c]	○	○

[a] Vulnerable to POODLE (`CVE-2014-3566`).
[b] Uses weak/obsolete ciphersuites.
[c] Vulnerable to ROBOT (`CVE-2017-13099` and others).

4.2 Online Voting Vendor Websites

We studied the corporate websites of vendors offering online voting solutions. Vendors are the implementors of online voting sites, and although their own sites are typically not directly associated with ballot casting, we would contend that they serve as an indicator of a vendor's awareness and ability to follow best practices. A selection of 27 different online voting vendors was studied (see Table 3).

At the time of inspection, 3 domains exhibited insufficient TLS protection; Dominion and Election Service Co. were improperly serving their cloud provider's certificate, which generated a browser error. Voting Place was offering SSL 3.0 which is vulnerable to `CVE-2014-3566`. Of the remaining domains, only 9 implemented HSTS, and no sites were found on the preload list.

4.3 Election Agencies

Critical election information such as dates and polling locations must be communicated to voters. While web-based attacks against election agencies providing such information would not impact the vote directly, it could serve to suppress votes, and undermine public trust. For example, an automated phone scam was used in Guelph, Canada in 2011 to direct voters to a fake polling location, and led to a criminal conviction,[13] which might be harder to achieve in the more anonymous setting of the web.

We decided to study the configurations of election agency websites in two countries who have used online voting at the sub-national level: Australia (see Table 4) and Canada (see Table 5). Of the 14 federal, provincial and territorial election agencies in Canada, none used HSTS or preloading, 2 had serious TLS configuration issues, and 9 did not implement TLS protection whatsoever. Elections British Columbia was found serving a self-signed certificate, which caused a browser error. We contacted them, and they promptly responded by obtaining and installing a certificate with a valid trust path. Élections Québec acknowledged receipt of our recommendations to disable weak and obsolete ciphersuites (esp. those using RC4), but had still not done so at time of writing.

Australia's election agencies fared slightly better: all 9 domains used TLS, however HSTS was found on a single domain only, and none were on the preload list.

4.4 Voter Registration Websites

Another important online election-related activity is voter registration. Some governments allow voters to sign up to vote and access information such as registration details, contact information of elected officials, voting instructions, location of ballot drop boxes and voting centers. For this component, we decided to focus on the Unites States. Most states in the US offer online voter registration.

[13] R. v. Sona, 2016 ONCA 452, Court of Appeals for Ontario, 2016. http://www.ontariocourts.ca/decisions/2016/2016ONCA0452.pdf

Table 3. HSTS in Online Voting Vendor Websites

Domain	Company	TLS	HSTS	Preload
`agora.vote`	Agora	●	○	○
`coalichain.io`	Coalichain	●	○	○
`cyber.ee`	Cybertentica	●	●	○
`democracy.earth`	Democracy Earth	●	○	○
`dominionvoting.com`	Dominion Voting	◐[a]	○	○
`eballot.com`	eBallot	●	○	○
`electionrunner.com`	Election Runner	●	○	○
`electionservicesco.com`	Election Services Co.	◐[a]	○	○
`www.essvote.com`	ES&S	●	●	○
`everyonecounts.com`	Everyone Counts	●	●	○
`followmyvote.com`	Follow My Vote	●	●	○
`id.ee`	Estonian National Identiy	●	●	○
`intelivote.com`	Intelivote	●	●	○
`polyas.com`	Polyas	●	●	○
`polys.me`	Polys	●	●	○
`scytl.com`	Scytl	●	○	○
`smartmatic.com`	Smartmatic	●	○	○
`simplesurvey.com`	Simple Survey	●	○	○
`simplyvoting.com`	Simply Voting	●	○	○
`sk.ee`	SK ID Solutions	●	○	○
`tivi.io`	TIVI	●	○	○
`voatz.com`	Voatz	◐[b]	○	○
`vogo.vote`	Vogo	●	○	○
`votebox.co`	Vote Box	●	●	○
`votem.com`	Votem	●	○	○
`votewatcher.com`	Vote Watcher	◐[a]	○	○
`votingplace.net`	Voting Place	◐[b]	○	○

[a] Certificate common-name mismatch with cloud host.
[b] Vulnerable to POODLE (`CVE-2014-3566`).

Table 4. HSTS in Australian Election Agencies

Domain	Division	TLS	HSTS	Preload
`aec.gov.au`	Australia	●	○	○
`elections.act.gov.au`	Australian Capital Territory	●	○	○
`elections.nsw.gov.au`	New South Wales	●	○	○
`ntec.nt.gov.au`	Northern Territory	●	○	○
`ecq.qld.gov.au`	Queensland	●	○	○
`ecsa.sa.gov.au`	South Australia	●	○	○
`tec.tas.gov.au`	Tasmania	●	○	○
`vec.vic.gov.au`	Victoria	●	●	○
`waec.wa.gov.au`	Western Australia	●	○	○

Table 5. HSTS in Canadian Election Agencies

Domain	Division	TLS	HSTS	Preload
`elections.ca`	Canada	○	○	○
`elections.ab.ca`	Alberta	○	○	○
`elections.bc.ca`	British Columbia	◐[a]	○	○
`electionsmanitoba.ca`	Manitoba	○	○	○
`www.electionsnb.ca`	New Brunswick	○	○	○
`www.elections.gov.nl.ca`	Newfoundland	○	○	○
`electionsnwt.ca`	Northwest Territories	○	○	○
`electionsnovascotia.ca`	Nova Scotia	●	○	○
`elections.nu.ca`	Nunavut	●	○	○
`elections.on.ca`	Ontario	●	○	○
`electionspei.ca`	Prince Edward Island	○	○	○
`electionsquebec.qc.ca`	Quebec	◐[b]	○	○
`elections.sk.ca`	Saskatchewan	○	○	○
`electionsyk.ca`	Yukon	○	○	○

[a] Using self-signed certificate.
[b] Using weak/obsolete ciphersuites.

According to the National Conference of State Legislators,[14] 37 US states plus the District of Columbia offered online voter registration as of 2018. We examined each of these states, and the results are given in Table 6. Only Maryland, Minnesota, Ohio and South Carolina implemented HSTS, and Florida's registration site[15] was the only one found in the preload list. Interestingly, this domain failed to meet all the preload eligibility requirements (including the use of the `strict-transport-security` header). Furthermore, Florida's site was present in the Chrome preload list *only*, meaning Firefox, Safari and Edge users would not be protected from TLS stripping.

Our investigation also revealed that the voter registration websites for Alaska, Louisiana, Massachusetts and Oregon had TLS configurations with serious vulnerabilities. We notified the affected states of our respective findings. Massachusetts promptly responded by disabling SSL 2 and 3, and we had a conference call with the state director of elections about our findings and their efforts to address them.

5 Barriers to Adoption

With only 50,000 entries on the HSTS preload list out of hundreds of millions of websites worldwide, adoption among voting sites was expected to be low. The question is: what barriers are faced by election management bodies and vendors?

[14] https://ncsl.org
[15] https://registertovoteflorida.gov

Table 6. HSTS in US Voter Registration Websites

Domain	State	TLS	HSTS	Preload
`alabamavotes.gov`	Alabama	●	○	○
`voterregistration.alaska.gov`	Alaska	◐[a,b,c]	○	○
`servicearizona.com`	Arizona	●	○	○
`registertovote.ca.gov`	California	●	○	○
`sos.state.co.us`	Colorado	●	○	○
`voterregistration.ct.gov`	Connecticut	●	○	○
`ivote.de.gov`	Delaware	●	○	○
`vote4dc.com`	D.C.	●	○	○
`registertovoteflorida.gov`	Florida	●	○	◐[f]
`registertovote.sos.ga.gov`	Georgia	●	○	○
`olvr.hawaii.gov`	Hawaii	●	○	○
`apps.idahovotes.gov`	Idaho	●	○	○
`ova.elections.il.gov`	Illinois	●	○	○
`indianavoters.in.gov`	Indiana	●	○	○
`sos.iowa.gov`	Iowa	●	○	○
`www.kdor.ks.gov`	Kansas	●	○	○
`vrsws.sos.ky.gov`	Kentucky	●	○	○
`sos.la.gov`	Louisiana	◐[c]	○	○
`voterservices.elections.state.md.us`	Maryland	●	●	○
`www.sec.state.ma.us`	Massachusetts	◐[c,d]	○	○
`mnvotes.sos.state.mn.us`	Minnesota	●	●	○
`sos.mo.gov`	Missouri	●	○	○
`nebraska.gov`	Nebraska	●	○	○
`nvsos.gov`	Nevada	●	○	○
`portal.sos.state.nm.us`	New Mexico	●	○	○
`dmv.ny.gov`	New York	●	○	○
`olvr.sos.state.oh.us`	Ohio	●	●	○
`secure.sos.state.or.us`	Oregon	◐[e]	○	○
`www.pavoterservices.state.pa.us`	Pennsylvania	●	○	○
`vote.sos.ri.gov`	Rhode Island	●	○	○
`info.scvotes.sc.gov`	South Carolina	●	●	○
`ovr.govote.tn.gov`	Tennessee	●	○	○
`secure.utah.gov`	Utah	●	○	○
`olvr.sec.state.vt.us`	Vermont	●	○	○
`vote.virginia.gov`	Virginia	●	○	○
`wei.sos.wa.gov`	Washington	●	○	○
`ovr.sos.wv.gov`	West Virginia	●	○	○
`myvote.wi.gov`	Wisconsin	●	○	○

[a] Vulnerable to `CVE-2014-0224`.
[b] Vulnerable to Logjam (`CVE-2015-4000`).
[c] Vulnerable to POODLE (`CVE-2014-3566`).
[d] Vulnerable to DROWN (`CVE-2016-0800`).
[e] Vulnerable to ROBOT (`CVE-2017-13099` and others).
[f] In Chrome preload list *only*.

Education and Awareness. The first barrier predominately appears to be a lack of awareness of TLS stripping, its implications, and the action required to protect against it. Increasing awareness begins by identifying at-risk websites and engaging with them. Eventually we hope to encourage the Qualsys SSL Test to include HSTS preloading in their grade scoring.

Technical Barriers. Membership in the preload list nominally requires: a valid TLS configuration; HTTPS redirects on the same domain; HSTS header with a sufficient `max-age`; the `include-subdomains` field; and, the `preload` field. Meeting these requirements can be non-trivial within the given web environment. Many web application frameworks (e.g., ASP .NET, Django, etc.) support HSTS via 3rd party plugins. The Helios maintainers observed that while Django offered an HSTS module, it did not support the preload field. In other cases (e.g., Apache, Nginx), preloading is a non-standard option which can be included only through a custom user-defined configuration. Microsoft's IIS server explicitly supports a preload attribute as of version 10, however all the US voter registration sites we observed running IIS used versions 8.5 or below.

Institutional Barriers. It is the root domain (e.g., `example.com`) that must be added, even if it is only a subdomain requiring preloading. This can be problematic in large institutions with numerous subdomains where applying a blanket HSTS policy would cause functional issues to, e.g., development severs. For example, Culnane et al. [5] observed that the online voting website of the 2017 Western Australian state (`ivote-cvs.elections.wa.gov.au`) was being hosted on the New South Wales iVote server, but would require action from the Western Australian IT staff (`wa.gov.au`) to be included in the preload list.

Scalability. Finally, there are scalability considerations to the preload list itself. Many election URLs are active only during polling period, and some domains have a clear one-time use (e.g., `election2020.org`). Adding all election websites to the preload list would, over time, risk filling it with stale domains.

We contacted the maintainers of the Chrome preload list and they were clear that, for the time being at least, all election sites are recommended for HSTS preloading, and that they be added at least 3 months in advance of the polling period to ensure time to be pushed out to voters via browser updates.

Conclusion

If you plan to deliver election and voting services online, the best practice to prevent TLS stripping attacks is to add your domain to the HSTS preload list. All voters using an updated browser will then be directed to your site securely over HTTPS, even if someone in your organization directs them to type or click an insecure HTTP link. This paper presented a study of over one hundred websites related to online elections and found almost none are presently doing this. The reasons are varied, but predominantly seem to be a matter of a lack of awareness of this issue, and we hope this paper will aid in this regard.

Acknowledgements

Thanks to Ben Adida, Kirsten Dorey, Lucas Garron and the anonymous reviewers.

References

[1] D. Adrian, K. Bhargavan, Z. Durumeric, P. Gaudry, M. Green, J. A. Halderman, N. Heninger, D. Springall, E. Thomé, L. Valenta, et al. Imperfect forward secrecy: How diffie-hellman fails in practice. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 5–17. ACM, 2015.

[2] M. Alsharnouby, F. Alaca, and S. Chiasson. Why phishing still works: User strategies for combating phishing attacks. *International Journal of Human-Computer Studies*, 82:69–82, 2015.

[3] C. Amrutkar, P. Traynor, and P. C. Van Oorschot. An empirical evaluation of security indicators in mobile web browsers. *IEEE Transactions on Mobile Computing*, 14(5):889–903, 2015.

[4] J. Clark and P. C. van Oorschot. Sok: Ssl and https: Revisiting past challenges and evaluating certificate trust model enhancements. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 511–525. IEEE, 2013.

[5] C. Culnane, M. Eldridge, A. Essex, and V. Teague. Trust implications of ddos protection in online elections. In *International Joint Conference on Electronic Voting*, pages 127–145. Springer, 2017.

[6] R. Dhamija, J. D. Tygar, and M. Hearst. Why phishing works. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 581–590. ACM, 2006.

[7] K. Dorey, N. Chang-Fong, and A. Essex. Indiscreet logs: Diffie-hellman backdoors in tls. In *Proceedings of the 24th Annual Network and Distributed System Security Symposium (NDSS 2017). The Internet Society*, 2017.

[8] Z. Durumeric, J. Kasten, D. Adrian, J. A. Halderman, M. Bailey, F. Li, N. Weaver, J. Amann, J. Beekman, M. Payer, et al. The matter of heartbleed. In *Proceedings of the 2014 Conference on Internet Measurement Conference*, pages 475–488. ACM, 2014.

[9] S. Dzieduszycka-Suinat, J. Murray, J. Kiniry, D. Zimmerman, D. Wagner, P. Robinson, A. Foltzer, and S. Morina. The Future of Voting: End-to-End Verifiable Internet Voting - Specification and Feasibility Study. US Vote Foundation, 2015.

[10] A. Essex. Detecting the detectable: Unintended consequences of cryptographic election verification. *IEEE Security & Privacy*, 15(3):30–38, 2017.

[11] B. Fogel, S. Farmer, H. Alkofahi, A. Skjellum, and M. Hafiz. Poodles, more poodles, freak attacks too: how server administrators responded to three serious web vulnerabilities. In *International Symposium on Engineering Secure Software and Systems*, pages 122–137. Springer, 2016.

[12] J. A. Halderman and V. Teague. The new south wales ivote system: Security failures and verification flaws in a live online election. In *International Conference on E-Voting and Identity*, pages 35–53. Springer, 2015.

[13] J. Hodges, C. Jackson, and A. Barth. HTTP Strict Transport Security (HSTS), RFC 6797, 2012.

[14] M. Kranch and J. Bonneau. Upgrading https in mid-air: An empirical study of strict transport security and key pinning. In *NDSS*, 2015.

[15] M. Marlinspike. More tricks for defeating ssl in practice. *Black Hat USA*, 2009.

[16] E. Moher, J. Clark, and A. Essex. Diffusion of voter responsibility: Potential failings in e2e voter receipt checking. In *USENIX Journal of Election Systems and Technology*, 2015.

[17] M. Nemec, M. Sys, P. Svenda, D. Klinec, and V. Matyas. The return of coppersmith's attack: Practical factorization of widely used rsa moduli. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1631–1648. ACM, 2017.

[18] E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2, RFC 5246, 2008.

[19] E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3 (Draft 28), 2018.

[20] Y. Sheffer, R. Holz, and P. Saint-Andre. Summarizing known attacks on transport layer security (tls) and datagram tls (dtls). *RFC 7457*, 2015.

[21] D. Springall, T. Finkenauer, Z. Durumeric, J. Kitcat, H. Hursti, M. MacAlpine, and J. A. Halderman. Security analysis of the estonian internet voting system. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 703–715. ACM, 2014.

[22] L. Valenta, D. Adrian, A. Sanso, S. Cohney, J. Fried, M. Hastings, J. A. Halderman, and N. Heninger. Measuring small subgroup attacks against diffie-hellman (eprint). In *Proceedings of the 24th Annual Network and Distributed System Security Symposium (NDSS 2017). The Internet Society*, 2017.

[23] L. Valenta, S. Cohney, A. Liao, J. Fried, S. Bodduluri, and N. Heninger. Factoring as a service. In *International Conference on Financial Cryptography and Data Security*, pages 321–338. Springer, 2016.

[24] Western Australian Electoral Commission. 2017 State General Election Election Report, 2017.

[25] F. Zagórski, R. T. Carback, D. Chaum, J. Clark, A. Essex, and P. L. Vora. Remotegrity: Design and use of an end-to-end verifiable remote voting system. In *International Conference on Applied Cryptography and Network Security*, pages 441–457. Springer, 2013.