Does Pushing Security on Clients Make Them Safer?

Matthew Traudt
U.S. Naval Research Laboratory
firstname.lastname@nrl.navy.mil

Paul Syverson
U.S. Naval Research Laboratory
firstname.lastname@nrl.navy.mil

1. INTRODUCTION

In this talk we will describe attacks stemming from pushing upgraded security settings for a visited site to clients connecting to that site. We specifically discuss the IETF standards HTTP Strict Transport Security (HSTS) [3] and HTTP Alternative Services (alt-svc) [4]. We will especially be concerned with users of Tor Browser, since they have chosen a browser whose most salient property is a greater emphasis on user privacy, security, and censorship resistance [5]. We will also consider Chrome, Firefox, and Safari users, however. Potential for user tracking was acknowledged in the RFCs for these IETF standards and have received some subequent attention. But the ability to track is way more significant than generally recognized. (For example, our HSTS tracking attacks are not prevented by countermeasures deployed by Apple last year after HSTS tracking of Safari users was detected in the wild [2].) Furthermore, censoring of offered content and user tracking by third parties have not been discussed by others. We will provide demonstrations of our attacks at example websites as well as all relevant code.

2. BASICS OF HSTS AND ONION ALT SERVICES

HSTS prevents a user from inadvertently using an unencrypted HTTP connection to a particular website by setting state in the browser that forces all subsequent connections to the host domain to use TLS [3]. State is set via a preloaded list or dynamically via an HTTP header from the host (which must be securely received over a TLS connection). Alternative Services allow a host to send a header providing an alternative domain, port, and even protocol for subsequent connections by a client for improved load balancing, locality optimization, or security. Though the client should subsequently use the alt service, it is not required. The displayed URL for an alt service will stay the same as in the original connection, and alt services "do not replace or change the origin for any given resource; in general, they are not visible to the software 'above' the access mechanism" [4].

Starting in September 2018, Tor users visiting Cloudflare-backed websites (hereafter all simplified to `cloudflare.com`) could be rerouted via `alt-svc` header to one of ten .onion addresses [6]. Since clients are expected to use an alt service they have been given, this would seem to offer self-authentication, routing, and address lookup protections provided by onion services [7]. And since the URL leading to these alt-service connections and displayed in the browser URL bar would simply be `cloudflare.com` (or the basic domain that cloudflare is backing), the user sees such connections to these self-authenticating domains as to a recognizable, traditional domain name. This would seem to address the long-recognized problem that .onion addresses are generally random strings encoding a public key and not human meaningful [10]. Further, according to RFC 7686, Certificate Authorities are only allowed to issue Extended Validation (EV) TLS certificate for onion addresses [1], making them more expensive and harder to obtain. But, since the TLS certificate supporting the authentication is for the displayed URL, such alt-service connections can use a much cheaper and easily obtained Domain Validation (DV) certificate. Thus for Tor users, onion alt services seem like a strong usable security win. And Cloudflare is not alone; Facebook similarly started offering onion alternative services to Tor users in 2018.

Because routing to an alt service is recommended but not required for browsers and is imperceptible to users or "software above the access mechanism", users cannot typically know if they are actually connecting to an onion alt service or not. Onion addresses are not human meaningful but they are human verifiable, e.g., they can be stored in bookmarks—but this does not help when they are invisible to both the user and the bookmark software. Similarly, Tor's onion service DHT guarantees address lookup confidentiality and integrity—but not when it is not known whether DNS or the onion service DHT is being used for the lookup. On the flip side, an adversary that obtains a fraudulent TLS certificate for a domain and gets a client to accept an `alt-svc` header for an onion address or other domain it controls can hijack future connections to that domain without any subsequent action or hijack visible in DNS. At least as bad, however, is that alt services allow tracking and censorship of Tor Browser and Firefox users.

3. TRACKING & CENSORING USERS VIA HSTS OR ALTERNATIVE SERVICES

Our HSTS attacks were published and presented last summer [9]. Our primary goals of presenting them here are

- Given their significance to privacy, censorship, and user security (especially of Tor users), we wanted to update the extent to which things have changed in the last year (spoiler: not at all), and to encourage discussion of what to do about this.
- To raise them to the PETS community as another example alongside onion alt services of pushing state to clients that might appear to improve security but also significantly undermines security. We hope to encourage in this talk a dialogue about this problem.

We will demonstrate and review our results on HSTS first and third-party tracking as well as censorship of users. A website demonstrating proof-of-concept of our HSTS attacks is at `https://hsts.satis.system33.pw/`. Code and demonstration videos are available at `https://github.com/pastly/satis-hsts-tracking`.

The onion alt service attacks we will present have not been previously published. We will provide links to our `alt-svc` demonstrations and code during the presentation as well. Our analysis indicates that neither Chrome nor Safari have yet significantly deployed `alt-svc` support in general. Firefox is, however, vulnerable to both first and third-party tracking and censorship. We will demonstrate how a host can send each first-time visitor a unique `alt-svc` header that can be used for tracking upon subsequent visits. (Or the host can do so to classify visitors arriving at a particular time, or from a given referer, etc.) In Tor Browser, this works whether the `alt-svc` header reroutes to a .onion domain, another domain resolvable in DNS, or even to another unique port at the same domain.

We will also demonstrate that such tracking can be done via a third party: a demonstration mock-up blog site includes a link to a CSS file at a demonstration mock-up CDN. Visiting a mock-up news site at a different domain that loads CSS from the same CDN loads the same identifying alternative service. This works in Firefox but not Tor Browser, for reasons we will discuss below.

Finally, just as we demonstrated in our HSTS attacks, an adversary that can track users via alt services can block or alter content at the different alternative services provided. In fact, it is more straightforward to set up than via HSTS.

4. LIMITATIONS, MITIGATIONS, QUESTIONS

There are some limitations on the alt service tracking we describe. First, in Firefox `alt-svc` state is not shared across regular browsing mode and private browsing mode; opening in regular browsing mode a domain that attempts to track its users this way and subsequently visiting it in private browsing mode (or vice versa) will not result in successfully tracking the user. Nonetheless, as long as *a* private browsing tab is left open, first and third party tracking state is preserved within private browsing mode, even if all existing tabs relevant to the tracking are closed before a new relevant tab is opened. Tor Browser isolates `alt-svc` state according to the displayed URL. Thus third-party tracking effectively does not work.

Neither HSTS nor `alt-svc` state is easy to find for ordinary users. And simply selecting "clear private data" or similar does not typically clear HSTS state, though other actions might (for example, selecting "Forget about this site." in Firefox). Chrome at least provides GUIs allowing the user to see and remove dynamic HSTS state if "Domain Security Policy" is selected under `chrome://net-internals/#hsts`. But, (1) this GUI is not easy to find (it is not linked to under "Settings" even when selecting "Advanced"), and (2) the user must enter specific domains in the GUI that then allow her to see or remove HSTS state just for the specified domain. Whether or not an alt service is being used is even harder to determine, unless one is visiting a site designed to expose this, as in our demonstration.

On the other hand, while dynamic HSTS is by design effectively impossible to block, blocking `alt-svc` headers is relatively easy. In companion work, we have created a WebExtension that amongst other things, allows one to specify lists of alternative services trusted for respective parent domains [8]. Such tracking prevention is not the primary purpose of this extension. But with our extension installed in either Firefox or Tor Browser, only trusted alt services are passed to the browser. This does not allow a Tor user to know whether or not she is getting the usual protections of onion services if an onion alt service is allowed by the extension. But at least tracking and censorship as we have described are not possible. The WebExtension is available at `https://github.com/pastly/satis-selfauth-domains/tree/master/webext`.

Our examples show that pushing dynamic state changes to browsers in order to improve security does not unambiguously achieve that aim. Sometimes the appearance of improved security is illusory: onion alt services do not improve authentication. On the other hand preventing, for example, dynamic HSTS could also allow some Man in the Middle attacks. And making it easier for users to observe and select for themselves whether to permit, block, or clear such state does not help if the decision about which makes them safer requires a detailed understanding of the nuances of their threat environment as well as their own current and future behavior. We hope that raising these examples will lead to discussion that will itself lead to better choices in designing systems to protect users.

5. REFERENCES

[1] J. Appelbaum and A. Muffett. The .onion special-use domain name. `https://tools.ietf.org/html/rfc7686`, 2015.

[2] B. Fulgham. Protecting against HSTS abuse. `https://webkit.org/blog/8146/protecting-against-hsts-abuse/`, March 18 2018.

[3] J. Hodges, C. Jackson, and A. Barth. HTTP Strict Transport Security (HSTS). `https://tools.ietf.org/html/rfc6797`, November 2012.

[4] M. Nottingham, P. McManus, and J. Reschke. HTTP Alternative Services. `https://tools.ietf.org/html/rfc7838`, April 2016.

[5] M. Perry, E. Clark, S. Murdoch, and G. Koppen. The design and implementation of the Tor Browser [DRAFT]. `https://2019.www.torproject.org/projects/torbrowser/design/`, June 15 2018.

[6] M. Sayrafi. Introducing the Cloudflare onion service. `https://blog.cloudflare.com/cloudflare-onion-service/`, September 20 2018.

[7] P. Syverson. The once and future onion. In S. N. Foley, D. Gollmann, and E. Snekkenes, editors, *Computer Security – ESORICS 2017*, pages 18–28. Springer-Verlag, LNCS 10492, 2017.

[8] P. Syverson and M. Traudt. Self-authenticating traditional domain names. In submission.

[9] P. Syverson and M. Traudt. HSTS supports targeted surveillance. In *USENIX Workshop on Free and Open Communications on the Internet (FOCI)*, 2018.

[10] P. Winter, A. Edmundson, L. M. Roberts, A. Dutkowska-Żuk, M. Chetty, and N. Feamster. How do Tor users interact with onion services? In *27th USENIX Security Symposium (USENIX Security 18)*, Baltimore, MD, 2018. USENIX Association.