Upgrading HTTPS in Mid-Air

An Empirical Study of Strict Transport Security and Key Pinning in the Wild

By: Michael Kranch and Joseph Bonneau

CITP
CENTER FOR INFORMATION TECHNOLOGY POLICY

What is HSTS and Key Pinning?

- Strict Transport Security (HSTS) is a countermeasure to HTTPS stripping through which the browser learns that specific domains must only be accessed via HTTPS by a HTTP header (dynamic) or a preset (preloaded) list.
- Key Pinning is the only currently deployed defense against a rogue certificate where the browser learns to connect to a specific HTTPS domain only if one of a designed set of keys (derived from the domain's certificate) is present.

Measurement Setup

- We utilized the OpenWPM web-measurement utility and modified the provided Selenium backbone's parsed DOM interface to extract all static resources (e.g. a tags, iframes, objects, etc.) from each site on the Chrome preload list.
- To extract dynamic resources (e.g. xmlhttprequest, scripts, etc.), we created a custom Firefox extension that implements the nsiContentPolicy interface in the Firefox extension API that is called prior to loading any resources.
- We used ZMAP to gather the complete header from every active HTTP and HTTPS IP address associated with the Alexa top million domains.
- Lastly, we created a custom crawl and used the X509 library to extract the key pins from every certificate associated with a pinned site.

Deployment of HSTS and Pinning

- HSTS was initially introduced by ForceHTTPS (Jackson and Barth) and standardized by RFC 6797 in 2012.
- HSTS is set through an HTTP header with a mandatory *max-age* (seconds) and an optional *includeSubdomains* directive.
- Google started included pre-loaded HSTS and pinning policies in Chrome in 2012 (see Figure 1 for growth over time).

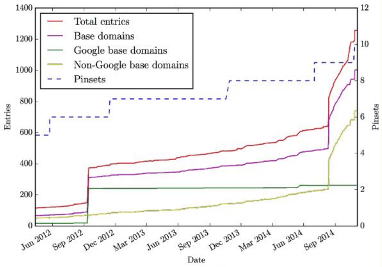

Figure 1: Growth of Preloaded List

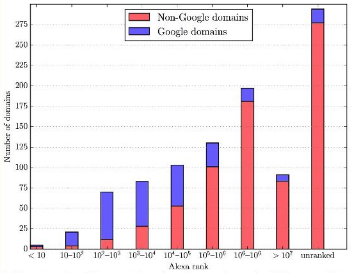

Figure 2: Alexa Rank of Preloaded Domains

- Firefox followed suit in 2014 by including a majority subset of Chrome's preload list plus several additional domains.
- Google enabled automated entry (with enforcement of additional parameters) into the preloaded list in August 2014.
- Dynamic Pinning (HPKP) was specified via draft RFC and is just now being seen in the wild.

Major Results of the Study

Error	Prevelance			Vulnerability
	%	#	Studied Domain	
Preloaded HSTS without dynamic HSTS	34.6%	349/1,008	domains with preloaded HSTS	HTTPS stripping possible on old browsers
Erroneous dynamic HSTS configuration	59.5%	7,494/12,593	top 1M domains attempting to set HSTS	HTTPS stripping possible on old browsers
Pinned site with non-pinned active content	3.0%	8/271	base domains with preloaded pins	data theft with a rogue certificate
	55.6%	5/9	non-Google base domains with preloaded pins	
Pinned site with non-pinned passive content	3.0%	8/271	base domains with preloaded pins	page modifications with a rogue certificate
	44.4%	4/9	non-Google base domains with preloaded pins	
Cookies scoped to non-pinned subdomains	1.8%	5/271	base domains with preloaded pins	cookie theft with a rogue certificate
	44.4%	4/9	non-Google base domains with preloaded pins	
Cookies scoped to non-HSTS subdomains	23.8%	182/765	base domains with preloaded HSTS	cookie theft by active network attacker
	47.8%	2,460/5,099	base domains with dynamic HSTS	

Table 1: Summary of Findings

Configuration Errors

	Alexa top 1M		Preloaded	
	#	%	#	%
Attempts to set dynamic HSTS	12,593	---	751	---
Doesn't redirect HTTP->HTTPS	5,554	44.1%	23	3.1%
Sets HTTP HSTS header only	517	4.1%	3	0.4%
Redirects to HTTP domain	774	6.1%	9	3.1%
HSTS Redirects to non-HSTS	74	0.6%	3	0.4%
Malformed HSTS header	322	2.6%	12	1.6%
max-age = 0	665	5.3%	0	0.0%
0 < max-age <= 1 day	2,213	17.6%	5	0.7%
Sets HSTS without errors	***5,099***	***40.5%***	***659***	***87.7%***

Table 2: Dynamic HSTS Errors

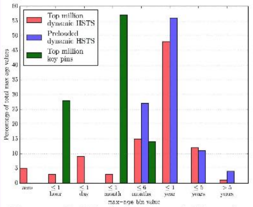

Figure 3: Histogram of Max-Age

- Configuration issues were the primary cause of HSTS errors even amongst the security leaders.
- Many sites failed to follow the specifications outlined in RFC 6797.

Mixed Content Issues

- Traditional mixed content refers to a HTTPS page loading resources from a HTTP origin, lowering the overall security to that of the HTTP site.
- HSTS and key-pinned sites similarly lower their overall security to that of the least secure loaded resource origin.
- Over half the non-Google pinned domains and just under a third of the preloaded HSTS domains include resources from traditional HTTPS sites.

	Content Type	Resource #
Active	script	15,540
	stylesheet	4,725
	link (rel="stylesheet")	2,470
	xmlhttprequest	1,515
	subdocument	170
	font	49
	total	***24,477***
Passive	image	41,702
	link (rel="shortcut icon")	146
	other passive	213
	total	***42,061***

Table 3: Types of Pinned Mixed Content Resources

Cookie Theft

- Many sites are vulnerable to cookie theft even when enabling HSTS. Since cookies by default apply to all subdomains, any site not setting HSTS to include subdomains is creating a security hole for cookies.

Condition	Preloaded	Dynamic
Domains with HSTS hole	30.1%	70.7%
Domains with vulnerable cookies	23.8%	23.8%
Cookies not marked secure	95.0%	95.0%

Table 4: Vulnerable Cookies from HSTS Domains

- More significantly, HSTS holes can leak secure cookies including authentication cookies even on pinned sites to an attacker with a rogue certificate.

Domain Hole	Auth Cookie	Insecure #	Total #
*.crypto.cat	No	3	3
*.dropbox.com	No	3	8
*.facebook.com	**Yes**	17	21
*.twitter.com	**Yes**	35	38
*.www.gmail.com	No	5	5
total		***63***	***75***

Table 5: Leakable Pinned Cookies

Conclusion

- Developers unfamiliarity with these new technologies in the leading cause of errors and many developers do not seem to fully understand same-origin policy.
- We recommend establishing defaults (*max-age* values and *includeSubdomains*) and simplifying the syntax to assist new adopters.

Future Work

- Continue to monitor the affect of automation on the growth of the preloaded list.
- Evaluate the use of new tokens (e.g. *include_subdomains_for_pinning_only*).
- Track the deployment of new technologies (HPKP).