Abusing Bleeding Edge Web Standards for AppSec Glory

Bryant Zadegan
Advisor/Mentor
Mach37

keybase.io/bryant
@eganist

Ryan Lester
CEO, Co-Founder
Cyph

hacker@linux.com
@TheRyanLester
@eganist

• Does AppSec stuff, usually.
• Mentors security startups, sometimes.
• “Mentors” others on AppSec, occasionally.
• Paid a buck to make Steve Ballmer dance, but just once.

@TheRyanLester

• Runs an E2EE communication startup
• Codes for an E2EE communication startup
• Ran QA automation at a rocket factory
• Got sued by Napster (and not for piracy)
Bleeding Edge Web Standards

For Your (Ab)use, we'll talk about these:

- SubResource Integrity

  SRI Fallback

- Content Security Policy

  CSP Meta-Hardening

- HTTP Public Key Pinning

  HPKP Suicide
But Why?

• New standards are frequently drafted.
• Many introduce unforeseen complications.
• Novel uses encourage future tweaks.

Source: Harold & Kumar Go to White Castle
SubResource Integrity

• Validate resources beyond your trust (e.g. CDNs)
  `<script
     src="https://code.jquery.com/jquery.min.js"
     integrity="sha256-[hash] sha256-[hash2]"
     crossorigin="anonymous"
     fallback-src="jquery.min.js">
  </script>

• caniuse.com/subresource-integrity
SRI Fallback

Per the SRI Spec:

NOTE

On a failed integrity check, an error event is fired. Developers wishing to provide a canonical fallback resource (e.g., a resource not served from a CDN, perhaps from a secondary, trusted, but slower source) can catch this error event and provide an appropriate handler to replace the failed resource with a different one.

...so we implemented it for you.
SubResource Integrity

- Validate resources beyond your trust (e.g. CDNs)
  
  ```html
  <script
  src="https://code.jquery.com/jquery.min.js"
  integrity="sha256-[hash] sha256-[hash2]"
  crossorigin="anonymous"
  x-sri-fallback="jquery.min.js">
  </script>
  ```

- caniuse.com/subresource-integrity
heisenberg.co/srifallbackdemo/

Kneel to the demo gods
Do source gods even exist?

[source](github.com/cyph/sri-fallback)
CVE-2016-1636 Demo

heisenberg.co/sridemo/sameorigin

(° 5°)
Content Security Policy:
script-src 'self' 'unsafe-inline'

Test <script> src with valid hash  Test <script> src with invalid hash
Content Security Policy:
script-src 'self' 'unsafe-inline'

Test <script> src with valid hash  Test <script> src with invalid hash
Content Security Policy:
script-src 'self' 'unsafe-inline'

Execution of remote code with hash BADHASH00000000000000000000000000000000 at 7/25/2016, 4:51:43 PM was a success!

Test <script> src with valid hash  Test <script> src with invalid hash
CSP Meta-Hardening

• Combines semi-strict header with strict `<meta>`.

• Allows for pre-loading of trusted complex logic.

• Does not work for the verbs `frame-ancestors`, `report-uri`, or `sandbox`. 
BUiLDeR DE MO

heisenberg.co/metacspdemo/

Fall on thy sword for the demo gods.
Content Security Policy:
script-src 'self' 'unsafe-inline'
Content Security Policy:

```
script-src 'self' 'unsafe-inline'
```

Execution of inline code at 7/25/2016, 4:59:46 PM was a success!
Content Security Policy:
script-src 'self' 'unsafe-inline'

Execution of non-inline code at 7/25/2016, 4:59:54 PM was a success!
Content Security Policy:
script-src 'self'

Execution of non-inline code at 7/25/2016, 4:59:54 PM was a success!
Content Security Policy:
script-src 'self'

Execution of non-inline code at 7/25/2016, 4:59:54 PM was a success!
CSP Meta-Hardening

Considerations

• *Static content only* in initial response!

• `X-XSS-Protection: 1; mode=block`
CSP Meta-Hardening

- Best for adapting a semi-recent application for use with CSP.
- Application’s trusted static logic is allowed to execute on initial load.
- Meta-Hardening prevents dynamic content from potentially executing later on.
Http Public Key Pinning

• This can break **brick** sites. Use **Reporting!**
  – (Chrome 46+ only; no reporting in Firefox 😞)

  **Public-Key-Pins-Report-Only:**
  max-age=5184000; includeSubdomains;
  pin-sha256="az9AwClWuHM+fYV+d8Cv9B4sAwdcoUqj93omk180/pc=";
  pin-sha256="5UONcYAsFtYscIlFlm4+aodoL20RRHzGaOeoSNEZ+iA=";
  report-uri="https://report-uri.io/report/[id]/reportOnly"

• [caniuse.com/hpkp](http://caniuse.com/hpkp)
Deliberate self-bricking via HPKP + Rapid Key Rotation.

Let's spend 20 minutes on how we can use this:

— to enable in-browser code signing
— to control content changes and harden SRI.
— to enable nuanced web content blocking. (NetSec)
— to track users...
— to be total jerks...

...in ways we shouldn't put in print.
(Thanks Jann Horn @ Cure53 for putting us onto this!)
HPKP Suicide

Content Pinning

Browser Presentation Layer

Browser AppCache or Service Worker

Web Server

Certificate Authority

GET /

200 OK
Public-Key-Pins
<html />

POST /rekey
new.csr
200 OK
new.crt

GET /

200 OK
(from service worker)
<html />
Wait, in-browser code signing? No extensions?

In theory.

In the last slide’s content pinning scheme, code signing logic goes in the ServiceWorker.

This effectively gets us Trust On First Use for current and future code.
HPKP Suicide

for Builders

Why “In theory”? This sounds like it should work.

In fact, Cyph employs a mature, audited implementation of exactly this.

However, it was considered so novel that we had to apply for a patent on it.

But, you can come close to this for free if you...
Control local storage updates! Harden SRI!

- Set HPKP max-age to count down to your deployment date.
- Rotate routinely.
Benefits:

• Retain control of front-end content between releases.
• Mitigate risks of SRI hash tampering server-side.
• **Decent security and performance gains**
HPKP Suicide

for Builders

Considerations:

• HPKP Suicide + SRI is a design-time decision!
  – Single Page Apps (SPAs) only

• Include mitigations such as halting distribution of HPKP headers if compromised.
I don't believe in demo gods
Web Content Gateway e.g. [SomeVendor]? Lock your users out of sites even when they're not on your network!

1. For flagged domains, set HPKP headers.
2. Optionally, Rotate keys weekly at the gateway.

Done! (By us disclosing it, is this now prior art? 😊)
HPKP Suicide

for Builders

Oh...  https://crt.sh/?id=19538258

Issuer:
commonName = VeriSign Class 3 Public Primary Certification Authority - G5

Subject:
commonName = Blue Coat Public Services Intermediate CA
organizationalUnitName = Symantec Trust Network
organizationName = "Blue Coat Systems, Inc."
HPKP Suicide
for Builders

User tracking?
Well, we really shouldn't talk about this...
But since this is DEF CON...

...let's track users!
HPKP Suicide

for Builders

Pre-requisites:
1. Lots of (sub)domains to pin
2. Browsers that allow HPKP incognito
3. Rapid Key Rotation

Let’s Encrypt
(Thanks! 😊)
HPKP Suicide

HPKP SuperCookies

Server-side

• /set: Returns HPKP header

• /check: No-op — no HPKP header, status code 200

Client-side (JavaScript)

• Set new ID: Hit /set on random subset of domains

• Check ID: Hit /check on all domains; note failures
I don't believe in demo gods
Not implemented by Google.
We only ran the script in console.
HPKPSupercookie('cyph.wang').then(o => console.log(o))
HPKPSupercookie('cyph.wang').then(o => console.log(o))

Promise {[[PromiseStatus]]: "pending", [[PromiseValue]]: undefined}

2016-07-25 17:06:48.206 Object {id: 4565566, isNewUser: true}
Not implemented by Reddit.
We only ran the script in console.
ELI5: Why do some people wipe their nose when they're proud of something?

New York City just witnessed an absolutely massive lightning strike completely spanning the Hudson River. I happened to have my camera set up to capture it. (i.imgur.com)
Considerations:

Risk: DoSing tracker domains as a public service

1. Domain whitelist for your own tracker, or
2. App-issued and tracker-verified nonce if analytics is your business model.

The pattern described is similar to others here:
SOURCE (New BSD)

github.com/cyph/hpkp-supercookie

Do source gods even exist?
HPKP Suicide

for Builders

...to be total jerks?

we *really* shouldn't talk about this...
Who are we kidding?
This is DEF CON.
HPKP Suicide
for Breakers

Pre-requisites:
1. A high-traffic target
2. A way to shell the box
3. A free certificate authority

Let’s Encrypt
(Sorry 😞)
HPKP Suicide

RansomPKP

1. Determine target
2. Generate ransom keypair (the recovery key)
3. Pwn target webserver.
4. Generate new lockout keypair + CSR
5. ?
6. Profit!
While owned users < $n$

1. "public-key-pins:
   max-age=31536000; includeSubdomains;
   pin-sha256= LOCKOUT_KEY;
   pin-sha256= RANSOM_KEY"

2. If owned users = $n$,
   1. Generate new lockout keypair + CSR
   2. Blow old lockout keypair. This locks out $n$ users.
   3. $n = 0$
Breaker Demo

isis.io
"You tweeachewous miscweant!"
-- Elmer Fudd
DO YOU WANT TO GET ON A WATCH LIST?!

BECAUSE THAT'S HOW YOU GET ON A WATCH LIST.
Your connection is not private

Attackers might be trying to steal your information from isis.io (for example, passwords, messages, or credit cards). NET::ERR_SSL_PINNED_KEY_NOT_IN_CERT_CHAIN

☑ Automatically report details of possible security incidents to Google. Privacy policy

ADVANCED

Reload
Considerations (i.e. why this is not a High):

1. Let's Encrypt rate limit: 20 certs weekly.
2. Chrome + Firefox have HPKP lockout mitigations
3. You still need to pop the box.
HPKP Suicide
RansomPKP

Programmatic Mitigations:

Chromium: WontFix Pre-Mitigated
Firefox: Match Chrome HPKP Max Age
Let's Encrypt: WontFix Out of Scope
HPKP Suicide

RansomPKP

Partial Host Mitigations

1. Use DNS Certification Authority Authorization (CAA) – RFC 6844.
2. Use HPKP; monitor headers for changes.
3. Try not to get popped.
End User Mitigations (Clearing key pins):

1. **Chrome**: chrome://net-internals/#hsts
2. **Chrome** (alt): clear any browsing data. "due to a curly brace mishap, we've been clearing it over-aggressively for years." (CVE-2016-1694)
   Clear your cache 😊
3. **Firefox**: about:config >>
   security.cert_pinning.enforcement_level = 0,
   visit site to take new header, re-enable.
SOURCE (New BSD)

github.com/cyph/ransompkp

Do source gods even exist?
Hat Tip

To Geller Bedoya, DigiCert, @el_d33, Jonn Callahan, Jann Horn and all of Cure53, Samy Kamkar, Jim Manico, Mike McBryde, Jim Rennie and his superb legal skill, Garrett Robinson, John Wilander, Doug Wilson, as well as the Chrome, Firefox, and Let's Encrypt security teams for their contributions.
Bryant Zadegan
Advisor/Mentor
Mach37

keybase.io/bryant
@eganist

Ryan Lester
CEO, Co-Founder
Cyph

hacker@linux.com
@TheRyanLester