Headers and Cookies
Who Am I?
Robin Wood
https://digi.ninja
@digininja
Background

- Started work as desktop app developer in 1996
- Moved to web app in 2003
- Moved to security testing in 2009
- Freelance tester and consultant
- Still do bits of web dev on the side
- Published over 50 security tools
HTTP Headers
<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strict-Transport-Security</td>
<td>max-age=63072000</td>
</tr>
<tr>
<td>Vary</td>
<td>Accept-Encoding</td>
</tr>
<tr>
<td>X-Content-Type-Options</td>
<td>nosniff</td>
</tr>
<tr>
<td>X-Frame-Options</td>
<td>sameorigin</td>
</tr>
<tr>
<td>X-XSS-Protection</td>
<td>1; mode=block</td>
</tr>
<tr>
<td>Referrer-Policy</td>
<td>no-referrer-when-downgrade</td>
</tr>
<tr>
<td>Content-Security-Policy</td>
<td>default-src 'self' ; style-src 'self' ; child-src <a href="https://ap">https://ap</a>...</td>
</tr>
<tr>
<td>Access-Control-Allow-Origin</td>
<td><a href="https://digi.ninja">https://digi.ninja</a></td>
</tr>
<tr>
<td>Access-Control-Allow-Headers</td>
<td>Origin, X-Requested-With, Content-Type, Access-Control-Allow...</td>
</tr>
<tr>
<td>Content-Length</td>
<td>7759</td>
</tr>
<tr>
<td>Cache-Control</td>
<td>max-age=600</td>
</tr>
<tr>
<td>Content-Type</td>
<td>text/html; charset=UTF-8</td>
</tr>
<tr>
<td>Date</td>
<td>Tue, 06 Feb 2018 10:24:21 GMT</td>
</tr>
<tr>
<td>Expires</td>
<td>Tue, 06 Feb 2018 10:34:21 GMT</td>
</tr>
<tr>
<td>Server</td>
<td>Apache</td>
</tr>
<tr>
<td>X-Powered-By</td>
<td>Rainbows</td>
</tr>
</tbody>
</table>
Main Headers

- X-Content-Type-Options
- X-Frame-Options
- X-XSS-Protection
- Referrer-Policy
- Strict-Transport-Security
- Content-Security-Policy
- Public-Key-Pins
- Expect-CT

https://digi.ninja
X-Content-Type-Options

Prevents a browser from trying to guess the file type of content

Protects against download attacks where browser makes bad choices

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Content-Type-Options
X-Content-Type-Options

No header – browser can sniff

One options:
• nosniff – honour the type specified*

* Recommended
X-Content-Type-Options

Example header:

x-content-type-options: nosniff
X-Frame-Options

Specifies how a site can be (or cannot be) used in frames and iframes

Protects against Clickjacking*

Demo https://vuln-demo.com/clickjack/

* https://www.owasp.org/index.php/Clickjacking
X-Frame-Options

No header – any site can frame this one

Three options:
• ALLOW-FROM – specify domains which can frame this one
• SAMEORIGIN – this site can frame itself
• DENY – nothing can frame this site*

* Recommended
X-Frame-Options

Example headers:

x-frame-options: sameorigin
x-frame-options: deny
X-XSS-Protection

Enable or disable a browsers built in Cross-Site Scripting protections

Affects Chrome and IE/Edge

X-XSS-Protection

No header – default browser behaviour

Four options:
• 0 – disable protections
• 1 – enable protections and sanitize output
• 1; report=<reporting-uri>*
• 1; mode=block – enable protections and block malicious content*

* Recommended
X-XSS-Protection

Example headers:

x-xss-protection: 0
x-xss-protection: 1; mode=block
Referrer-Policy

Newest header on the block

Specifies when a browser should pass a referer header

Useful when you have sensitive data in querystrings

Referrer-Policy

No header – default browser behaviour, usually just pass the header

Eight options:

• no-referrer
• no-referrer-when-downgrade
• origin
• origin-when-cross-origin
• same-origin
• strict-origin
• strict-origin-when-cross-origin*
• unsafe-url

* Recommended

https://www.w3.org/TR/referrer-policy/

https://digi.ninja
Referrer-Policy

Example headers:

referrer-policy: strict-origin-when-cross-origin
referrer-policy: origin
Referrer-Policy

Can break tracking/logging software

Obviously breaks referrer programs if not done right
Strict-Transport-Security

Also known as HSTS

Enforces HTTPS on all requests

Strict-Transport-Security

No header – traffic can use HTTP or HTTPS

Three options:

• `max-age=<expire-time>`
• `max-age=<expire-time>; includeSubDomains`*
• `max-age=<expire-time>; preload`

* Recommended
Strict-Transport-Security

Example headers:

strict-transport-security: max-age=31536000;
includeSubDomains

strict-transport-security: max-age=0;
Strict-Transport-Security

Site still vulnerable on first load
Can be mitigated with preloading
Submit at [https://hstspreload.org/](https://hstspreload.org/)
Content-Security-Policy

Hardest one on the list to implement

Locks down how and what resources can be used by a site by use of whitelisting

Two modes, enabled and report only
Content-Security-Policy

Mozilla scrapes Google’s list for Firefox
https://blog.mozilla.org/security/2012/11/01/preloading-hsts/

Mozilla’s Guide

Publishers Guide
https://content-security-policy.com/
Content-Security-Policy

Example header:

Content-Security-Policy: default-src https:
Content-Security-Policy: default-src site.com
Content-Security-Policy: default-src https://site.com
Content-Security-Policy

Example header:

Content-Security-Policy

child-src  media-src
connect-src object-src
default-src plugin-types
font-src report-uri
form-action sandbox
frame-ancestors script-src
frame-src* style-src
img-src

* frame-src is deprecated, use child-src

https://digi.ninja
<table>
<thead>
<tr>
<th>Feature</th>
<th>Edge</th>
<th>Opera</th>
<th>Firefox</th>
<th>Google Chrome</th>
<th>IE 11</th>
<th>IE 10</th>
<th>IE 9</th>
<th>IE 8</th>
<th>IE 7</th>
<th>IE 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>base-uri</td>
<td>25</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>block-all-mixed-content</td>
<td></td>
<td></td>
<td></td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>child-src</td>
<td>25</td>
<td>yes</td>
<td>yes</td>
<td>not supported</td>
<td></td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>connect-src</td>
<td>25</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>default-src</td>
<td>25</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>dns-origin</td>
<td></td>
<td>no</td>
<td>no</td>
<td>not supported</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>font-src</td>
<td>25</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td></td>
<td>yes</td>
<td></td>
<td>no</td>
</tr>
<tr>
<td>form-action</td>
<td>40</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td></td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>frame-ancestors</td>
<td>40</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td></td>
<td>yes</td>
<td></td>
<td>no</td>
</tr>
<tr>
<td>frame-src</td>
<td>25</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>img-src</td>
<td>25</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>manifest-src</td>
<td>25</td>
<td>no</td>
<td>no</td>
<td>not supported</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>media-src</td>
<td>25</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>navigation-to</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>not supported</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>object-src</td>
<td>25</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>plugin-types</td>
<td>40</td>
<td>no</td>
<td>no</td>
<td>not supported</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>referer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>yes</td>
<td></td>
<td>no</td>
</tr>
<tr>
<td>report-sample</td>
<td>32</td>
<td>no</td>
<td>no</td>
<td>not supported</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>report-to</td>
<td>59</td>
<td>no</td>
<td>no</td>
<td>not supported</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>require-safe-url</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>not supported</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>sandbox</td>
<td>25</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>script-src</td>
<td>25</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>strict-dynamic</td>
<td>52</td>
<td>no</td>
<td>no</td>
<td>not supported</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>style-src</td>
<td>25</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>upgrade-insecure-req</td>
<td>43</td>
<td>no</td>
<td>no</td>
<td>not supported</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>worker-src</td>
<td>59</td>
<td>no</td>
<td>no</td>
<td>not supported</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>

- **Full support**
- **Compatibility unknown**
- **No support**
- **Experimental. Expect behavior to change in the future.**
- **Deprecated. Not for use in new browser support.**
- **Non-standard. Expect poor cross-browser support.**
- **See Implementation notes.**
- **User must explicitly enable this feature.**
Reporting of failures can be done by adding the following to the header:

```plaintext
report-uri https://report-uri.com
```

For example:

```plaintext
```
Content-Security-Policy

Strongly recommend setting up an account with Report URI and sending reports there

https://report-uri.com/

Just remember to monitor them!
<table>
<thead>
<tr>
<th>Action</th>
<th>Date</th>
<th>URI</th>
<th>Directive</th>
<th>Blocked URI</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>07/02/2018 11:17:45</td>
<td><a href="https://digi.ninja/projects_metasploit.php">https://digi.ninja/projects_metasploit.php</a></td>
<td>font-src</td>
<td>blocked hostname</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>07/02/2018 11:17:35</td>
<td><a href="https://digi.ninja/projects_networking.php">https://digi.ninja/projects_networking.php</a></td>
<td>font-src</td>
<td>blocked path</td>
<td>1</td>
</tr>
</tbody>
</table>
Public Key Pinning

Short version – Specify in a header which CAs can sign your certificates

Longer version is a bit more complicated than that

https://developer.mozilla.org/en-US/docs/Web/HTTP/Public_Key_Pinning
Public Key Pinning

Current advice – Don’t do it!
Public Key Pinning

Deprecated by Google in Chrome 67

https://www.theregister.co.uk/2017/10/30/google_hpkp/
Public Key Pinning

PKP Suicide

https://www.smashingmagazine.com/be-afraid-of-public-key-pinning/
Public Key Pinning

Replaced by...
Expect-CT

Replaces Key Pinning

Tells the browser to only accept a certificate if there is an entry for it in the certificate transparency register

Expect-CT

No header – browser dependent

One options:

- enforce – Only accept the cert if CT found
- max-age – The number of seconds to honour the header
- report-uri – URI to report failures to
Expect-CT

Example headers:

Expect-CT: max-age=0, report-uri="<report URI>"

Expect-CT: enforce, max-age=60, report-uri="<report URI>"
Expect-CT

Check CT logs here
https://crt.sh

Facebook monitoring
## Expect-CT

<table>
<thead>
<tr>
<th>crt.sh ID</th>
<th>Logged At</th>
<th>Not Before</th>
<th>Issuer Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>272121594</td>
<td>2017-12-06</td>
<td>2017-12-06</td>
<td>C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3</td>
</tr>
<tr>
<td>250449514</td>
<td>2017-11-09</td>
<td>2017-11-09</td>
<td>C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3</td>
</tr>
<tr>
<td>223131904</td>
<td>2017-10-23</td>
<td>2017-10-22</td>
<td>C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3</td>
</tr>
<tr>
<td>209068152</td>
<td>2017-09-06</td>
<td>2017-09-06</td>
<td>C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3</td>
</tr>
<tr>
<td>195061250</td>
<td>2017-09-06</td>
<td>2017-09-06</td>
<td>C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3</td>
</tr>
<tr>
<td>155189032</td>
<td>2017-06-16</td>
<td>2017-06-16</td>
<td>C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3</td>
</tr>
<tr>
<td>106657775</td>
<td>2017-03-24</td>
<td>2017-03-23</td>
<td>C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3</td>
</tr>
<tr>
<td>71760150</td>
<td>2016-12-31</td>
<td>2016-12-30</td>
<td>C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3</td>
</tr>
<tr>
<td>40995830</td>
<td>2016-10-07</td>
<td>2016-10-07</td>
<td>C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3</td>
</tr>
<tr>
<td>24760643</td>
<td>2016-07-16</td>
<td>2016-07-16</td>
<td>C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3</td>
</tr>
<tr>
<td>17197415</td>
<td>2016-04-24</td>
<td>2016-04-23</td>
<td>C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3</td>
</tr>
<tr>
<td>17190771</td>
<td>2016-04-23</td>
<td>2016-04-23</td>
<td>C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3</td>
</tr>
<tr>
<td>12473468</td>
<td>2016-02-01</td>
<td>2016-01-31</td>
<td>C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3</td>
</tr>
<tr>
<td>11538949</td>
<td>2016-01-05</td>
<td>2016-01-05</td>
<td>C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3</td>
</tr>
<tr>
<td>11538948</td>
<td>2016-01-05</td>
<td>2016-01-05</td>
<td>C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3</td>
</tr>
<tr>
<td>11538959</td>
<td>2016-01-05</td>
<td>2016-01-05</td>
<td>C=US, O=Let's Encrypt, CN=Let's Encrypt Authority X3</td>
</tr>
</tbody>
</table>
Cookies
The Flags

- Secure
- HttpOnly
- SameSite
Secure

Ensures the cookie is only sent over HTTPS

Stops cookies being sniffed while in transit

Should be set on all session cookies

https://www.owasp.org/index.php/SecureFlag
Secure

Not needed if no HTTP:// site exists?

http://site.com:443/page
HttpOnly

Prevents JavaScript from accessing the cookie

Blocks session hijacking through cookie theft

Should be set on all session cookies

https://www.owasp.org/index.php/HttpOnly
SameSite

New flag from around November 2017

Chrome 62 onwards, Firefox 59 onwards

Not in IE, Edge or Safari

https://www.owasp.org/index.php/SameSite
SameSite

Cookie only sent with a request if the request comes from the same site

Designed to prevent Cross-Site Request Forgery (CSRF) attacks
SameSite

No header – no restriction on cookies

Two options:

• **strict** – never send the cookie unless the request originates same site

• **lax** – send the cookie for “Safe” methods (GET, HEAD, OPTIONS, TRACE)*

* Recommended
Any Questions?
Robin Wood
https://digi.ninja
@digininja