LET'S BREAK STUFF

CONTENT SECURITY POLICIES

ABOUT ME

- Senior Software Engineer at Viva IT
(those folks in orange hoodies at some conferences & events you may have been to)
- @Brunty
- @PHPem
- mfyu.co.uk
- matt@mfyu.co.uk

THINGS I DO

- Dungeon master for D&D campaigns
- Mentor, lead & teach apprentices & junior developers
- Run & organise PHP East Midlands
- Speak at user groups and conferences
- **Break production sites with incorrectly configured content security policies**

A TALK IN 3 PARTS

- XSS
- CSP
- Break stuff

THE SCARY STUFF

WHAT IS CROSS-SITE-SCRIPTING (XSS)?

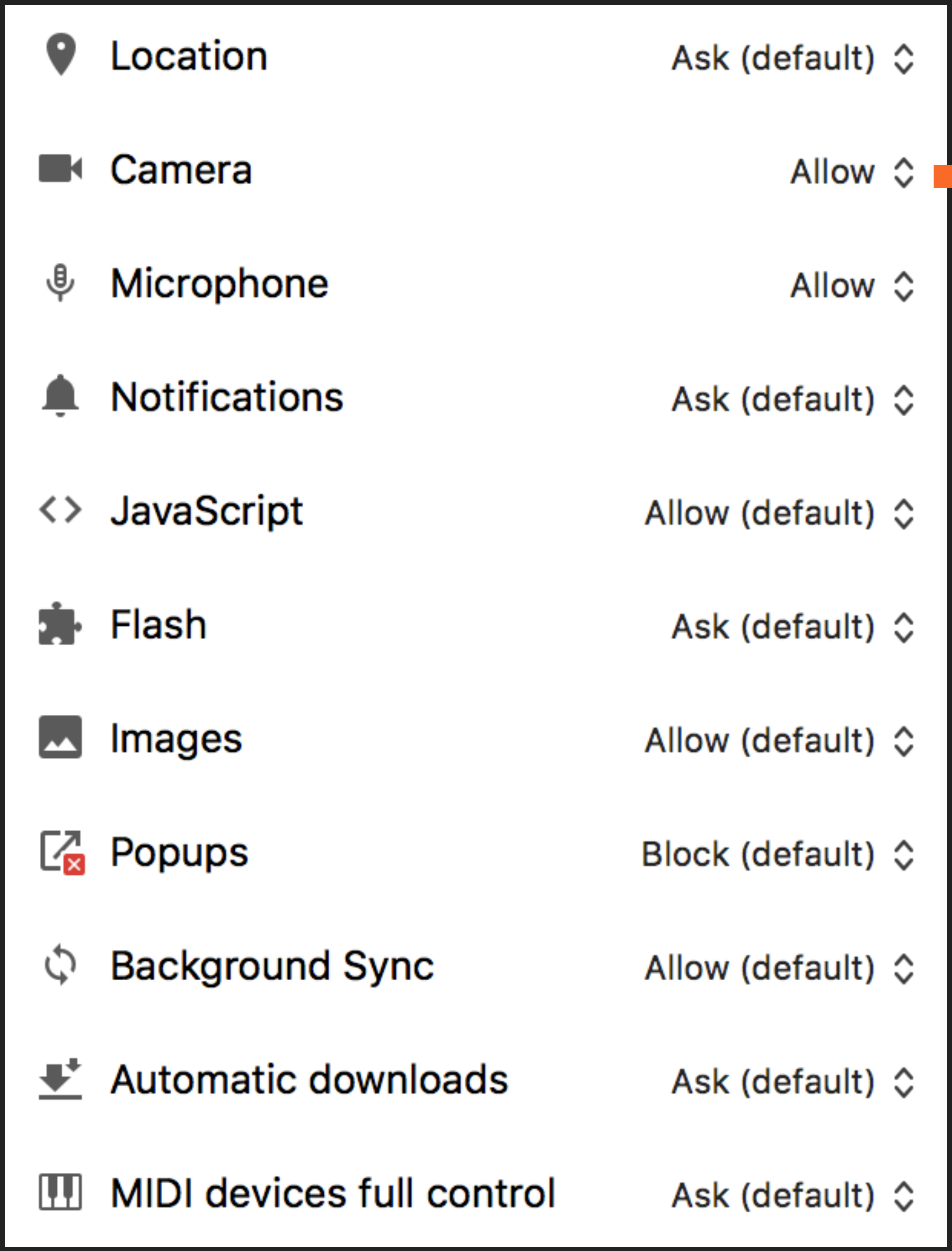
- XSS enables an attacker to inject client-side scripts into non-malicious web pages viewed by other users
- In 2016 there was a 61% likelihood of a **browser-based** vulnerability being found in a web application
- Of those browser based vulnerabilities, 86% were found to be XSS related
- That's just over 52% of all web application vulnerabilities
  https://www.edgescan.com/assets/docs/reports/2016-edgescan-stats-report.pdf

WHAT CAN BE DONE WITH XSS?

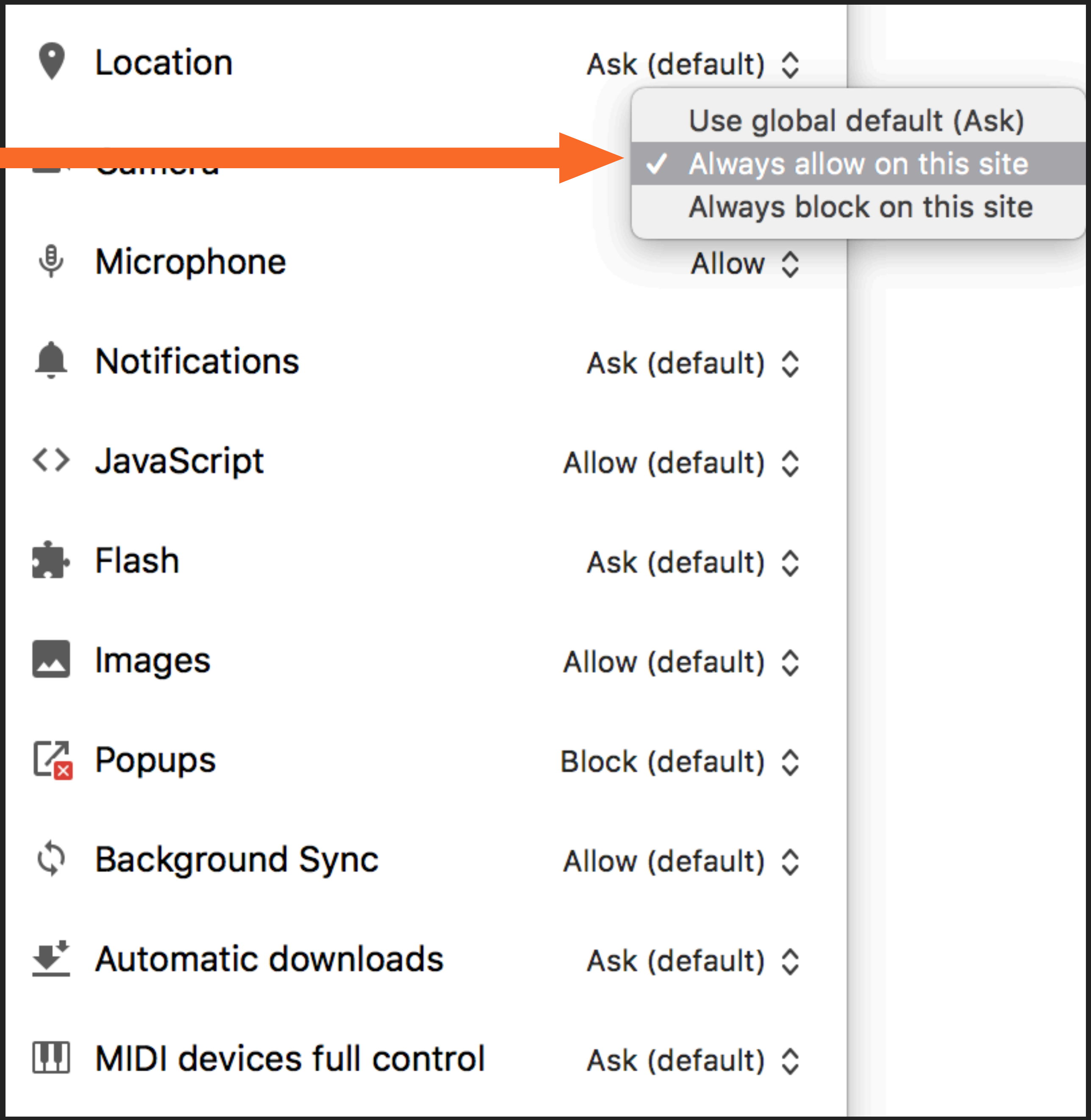
- Put pictures of ~~cats~~ dogs in web pages
- `alert('💩');`
- Rickroll a user
- Twitter self-retweeting tweet
  https://www.youtube.com/watch?v=zv0kZKC6GAM
- Samy worm
  https://en.wikipedia.org/wiki/Samy_(computer_worm)

WHAT CAN BE DONE WITH XSS?

- Make modifications to the DOM - replace a form action to point to your own script to capture credentials.
- Load in additional scripts, resources, styles, images etc
- Access HTML5 APIs - webcam, microphone, geolocation
- Steal cookies (and therefore steal session cookies)

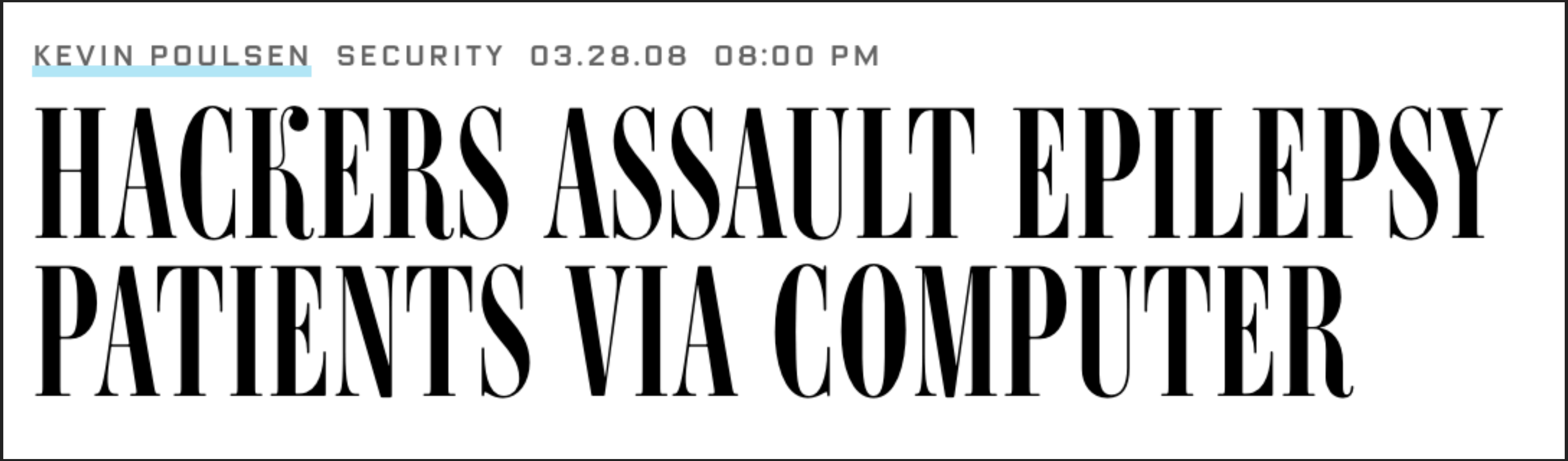


Location	Ask (default)
Camera	Allow
Microphone	Allow
Notifications	Ask (default)
JavaScript	Allow (default)
Flash	Ask (default)
Images	Allow (default)
Popups	Block (default)
Background Sync	Allow (default)
Automatic downloads	Ask (default)
MIDI devices full control	Ask (default)



Location	Ask (default)
Camera	Use global default (Ask) / [✓ Always allow on this site] / Always block on this site
Microphone	Allow
Notifications	Ask (default)
JavaScript	Allow (default)
Flash	Ask (default)
Images	Allow (default)
Popups	Block (default)
Background Sync	Allow (default)
Automatic downloads	Ask (default)
MIDI devices full control	Ask (default)

WHAT CAN BE DONE WITH XSS?


KEVIN POULSEN SECURITY 03.28.08 08:00 PM

HACKERS ASSAULT EPILEPSY PATIENTS VIA COMPUTER

https://www.wired.com/2008/03/hackers-assault-epilepsy-patients-via-computer/

STORED XSS (AKA PERSISTENT OR TYPE I)

- Occurs when input is stored - generally in a server-side database, but not always
- This could also be within a HTML5 database, thus never being sent to the server at all
- who.is was a site Rickrolled by a TXT record in the DNS of a website **(yes, really)**

REFLECTED XSS (AKA NON-PERSISTENT OR TYPE II)

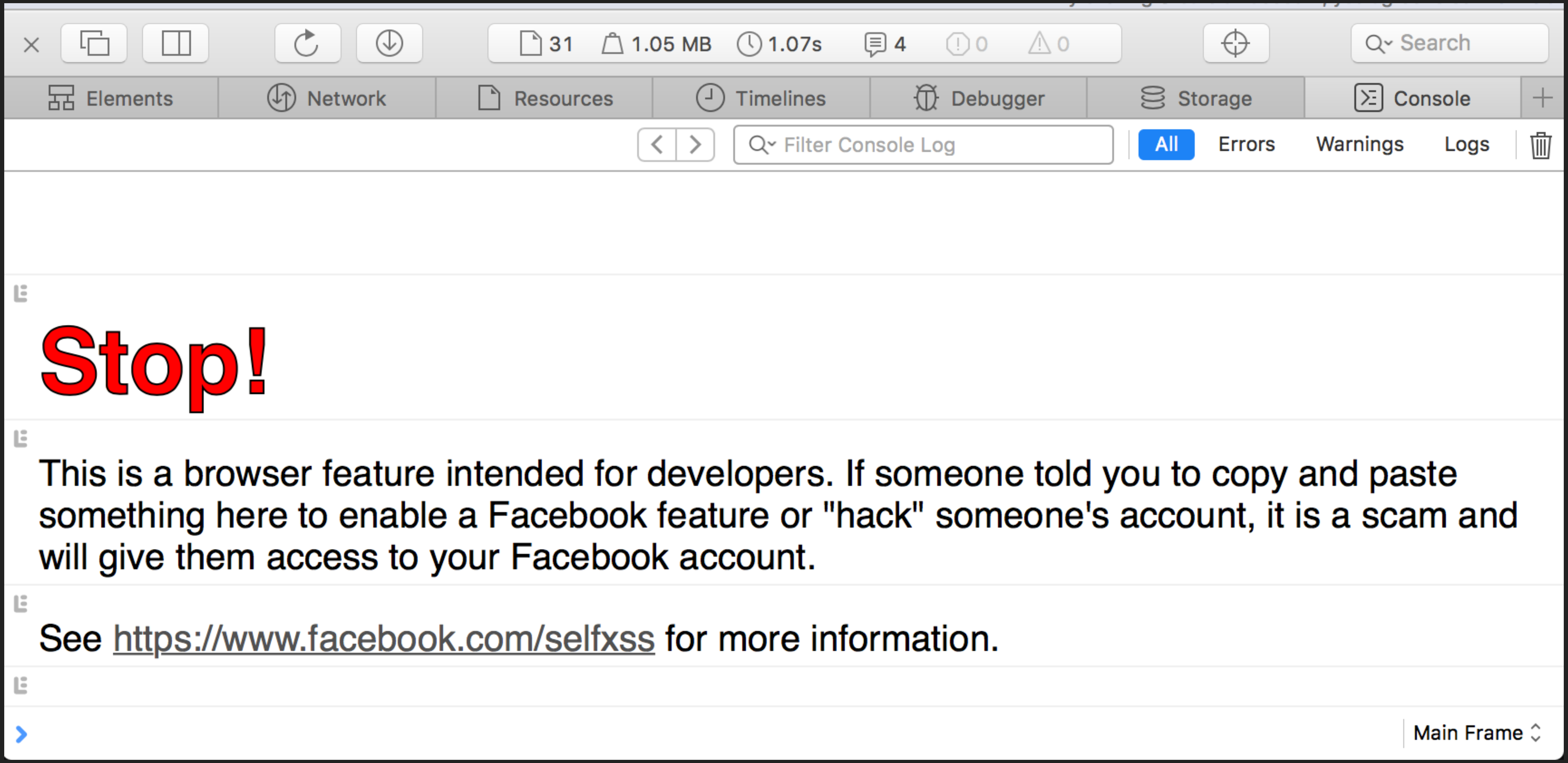
- Occurs when user input provided in the request is immediately returned - such as in an error message, search string etc
- Data is not stored, and in some instances, may not even reach the server (see the next type of XSS)

DOM-BASED XSS (AKA TYPE-0)

- The entire flow of the attack takes place within the browser
- For example, if JavaScript in the site takes input, and uses something like document.write based on that input, it can be vulnerable to a DOM-based XSS attack

SELF XSS

- Social-engineering form of XSS
- Requires the user to execute code in the browser
- Doing so via the console can't be protected by a lot of methods
- Not considered a 'true' XSS attack due to requiring the user to execute the code



Stop!

This is a browser feature intended for developers. If someone told you to copy and paste something here to enable a Facebook feature or "hack" someone's account, it is a scam and will give them access to your Facebook account.

See https://www.facebook.com/selfxss for more information.

LET'S FIGHT BACK

WHAT IS A CSP?

HTTP RESPONSE HEADER TO HELP REDUCE XSS RISKS

WHAT IS A CSP?

IT IS NOT A SILVER BULLET

WHAT IS A CSP?

IT IS AN EXTRA LAYER OF SECURITY

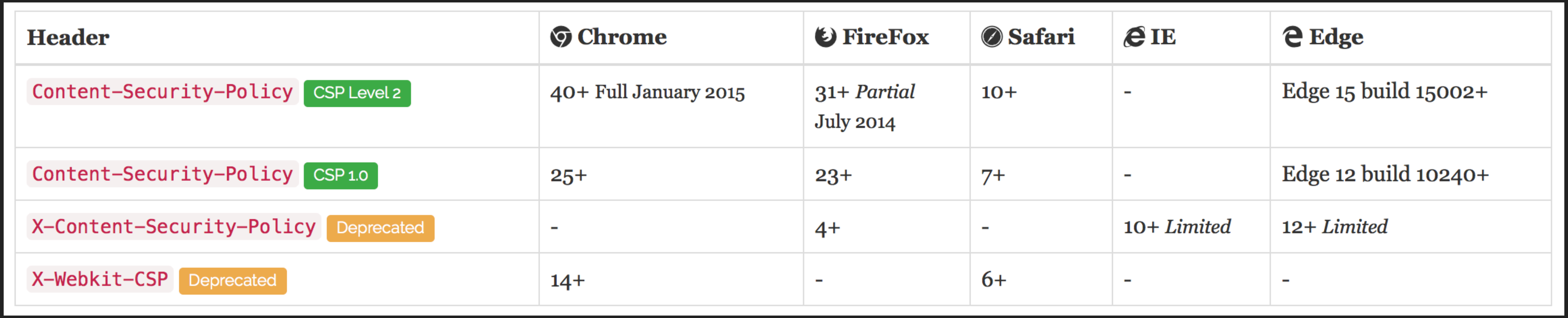
HOW DOES A CSP WORK?

DECLARES WHAT RESOURCES ARE ALLOWED TO LOAD

IT CAN EVEN HELP WITH

BLOCKING THOSE PESKY CRYPTO-MINING SCRIPTS THAT HAVE BEEN POPPING UP

BROWSER SUPPORT



Header	Chrome	FireFox	Safari	IE	Edge
`Content-Security-Policy` CSP Level 2	40+ Full January 2015	31+ *Partial* July 2014	10+	-	Edge 15 build 15002+
`Content-Security-Policy` CSP 1.0	25+	23+	7+	-	Edge 12 build 10240+
`X-Content-Security-Policy` Deprecated	-	4+	-	10+ *Limited*	12+ *Limited*
`X-Webkit-CSP` Deprecated	14+	-	6+	-	-

Meh, it's alright(ish)
Sorry IE users

WHAT CAN WE PROTECT?

- default-src
- script-src
- style-src
- img-src
- form-action
- update-insecure-requests
- and so much more…

FULL REFERENCE:

https://content-security-policy.com
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Security-Policy

```
img-src *
```

ALLOWS ANY URL EXCEPT DATA: BLOB: FILESYSTEM: SCHEMES.

```
object-src 'none'
```

DON'T LOAD RESOURCES FROM ANY SOURCE

```
style-src 'self'
```

ALLOW LOADING FROM SAME SCHEME, HOST AND PORT

`script-src 'unsafe-inline'`

ALLOWS USE OF INLINE SOURCE ELEMENTS SUCH AS STYLE ATTRIBUTE, ONCLICK, OR SCRIPT TAG BODIES

DON'T USE UNSAFE-INLINE

```
script-src 'self' 'nonce-$RANDOM'
```

```
<script nonce="$RANDOM">...</script>
```

```
Content-Security-Policy: default-src 'none'; script-
src 'self' https://*.google.com 'nonce-random123';
style-src 'self'; img-src 'self'; upgrade-insecure-
requests; form-action 'self';
```

LEARN FROM MY MISTAKES

I BROKE PRODUCTION WITH A BAD CSP

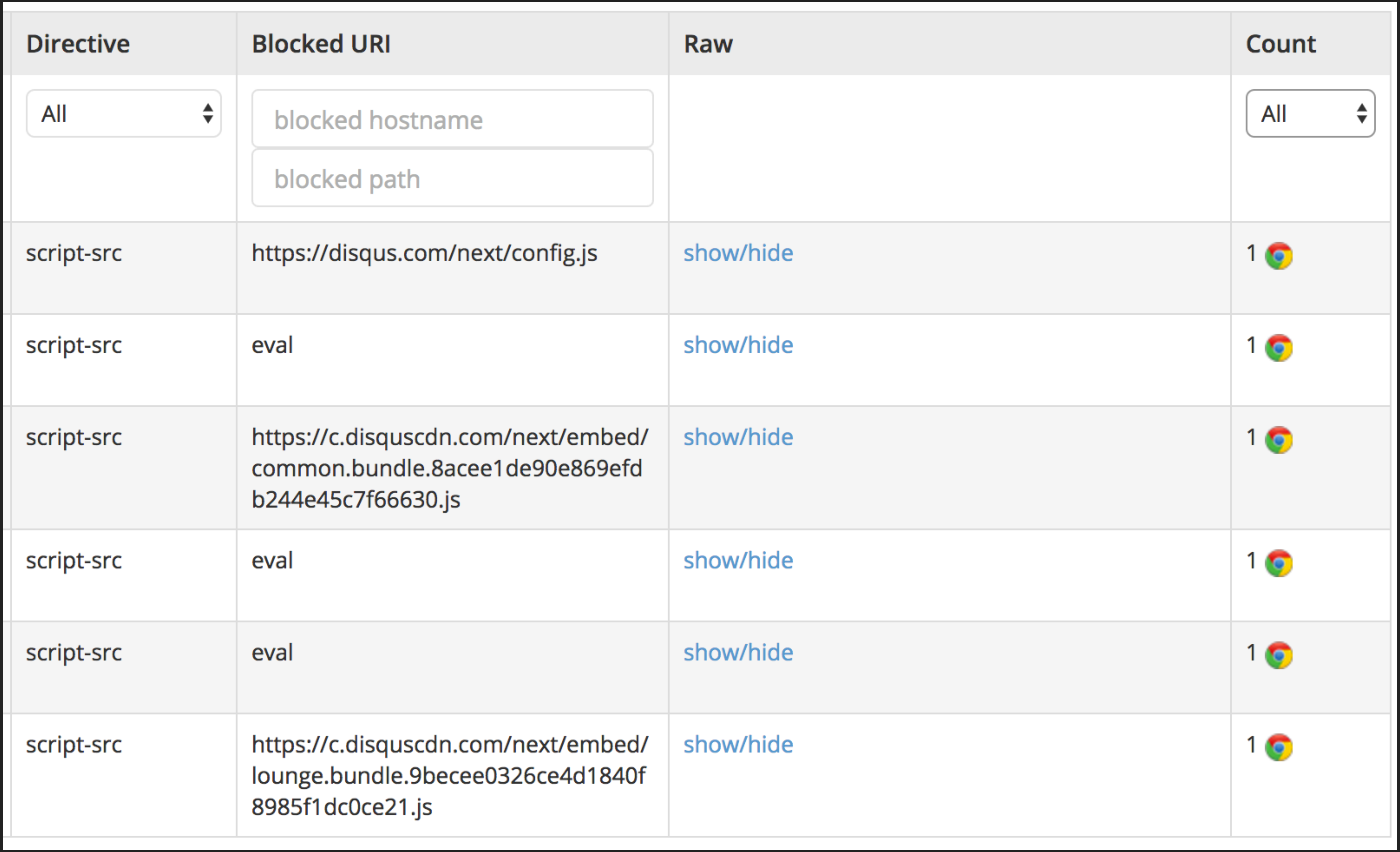
DON'T DO WHAT I DID

REPORT-URI

WHEN A POLICY FAILURE OCCURS, THE BROWSER SENDS A JSON PAYLOAD TO THAT URL

```
{
    "csp-report": {
        "blocked-uri": "self",
        "document-uri": "https://mysite.com",
        "line-number": 1,
        "original-policy": "script-src 'self'",
        "script-sample": "try {  for(var lastpass_iter=0; lastpass...",
        "source-file": "https://mysite.com",
        "violated-directive": "script-src 'self'"
    }
}
```

REPORT-URI.IO



Directive	Blocked URI	Raw	Count
All	blocked hostname blocked path		All
script-src	https://disqus.com/next/config.js	show/hide	1
script-src	eval	show/hide	1
script-src	https://c.disquscdn.com/next/embed/common.bundle.8acee1de90e869efdb244e45c7f66630.js	show/hide	1
script-src	eval	show/hide	1
script-src	eval	show/hide	1
script-src	https://c.disquscdn.com/next/embed/lounge.bundle.9becee0326ce4d1840f8985f1dc0ce21.js	show/hide	1

REPORT-ONLY

```
Content-Security-Policy-Report-Only: [policy]; report-
uri https://app.report-uri.io/r/default/csp/reportOnly;
```

TRIAL STUFF BEFORE ENFORCING

THERE WILL BE NOISE, LOTS OF NOISE

TIPS

- Have an easy and quick way to disable the CSP in production if required
- Better yet, have a way to switch it from enforced to report only so you can get violations reported to help you debug
- Add the CSP at an application level if you need a nonce

MULTIPLE POLICIES

- They complicate things
- For a resource to be allowed, it must be allowed by all policies declared (problematic if an enforced policy)
- I tend to avoid them where possible on enforced policies
- But with report-only mode they can be very useful to deploy and test multiple policies at the same time (as nothing breaks for the user)

NONCES

- Don't generate multiple nonces in the same request (but **do** generate a new nonce on each separate request)
- If using a templating engine (such as twig) - add the nonce as a global so it's available in every template by default
- Write a helper in your template engine to generate script tags with a nonce if it's available

DEMO TIME!
LET'S BREAK STUFF

@SCOTT_HELME

HE KNOWS HIS STUFF!

@MR_GOODWIN

HE FIRST INTRODUCED ME TO WHAT A CSP IS

LINKS & FURTHER READING

- https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)
- https://content-security-policy.com
- https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Security-Policy
- https://report-uri.io
- https://scotthelme.co.uk/just-how-much-traffic-can-you-generate-using-csp/
- https://www.edgescan.com/assets/docs/reports/2016-edgescan-stats-report.pdf
- http://theharmonyguy.com/oldsite/2011/04/21/recent-facebook-xss-attacks-show-increasing-sophistication/
- https://github.com/Brunty/csp-demo

THANK YOU

QUESTIONS?

@BRUNTY

JOIND.IN/TALK/9A570

NOTI.ST/BRUNTY

MFYU.CO.UK