How to use CSP to stop XSS

Ken Lee

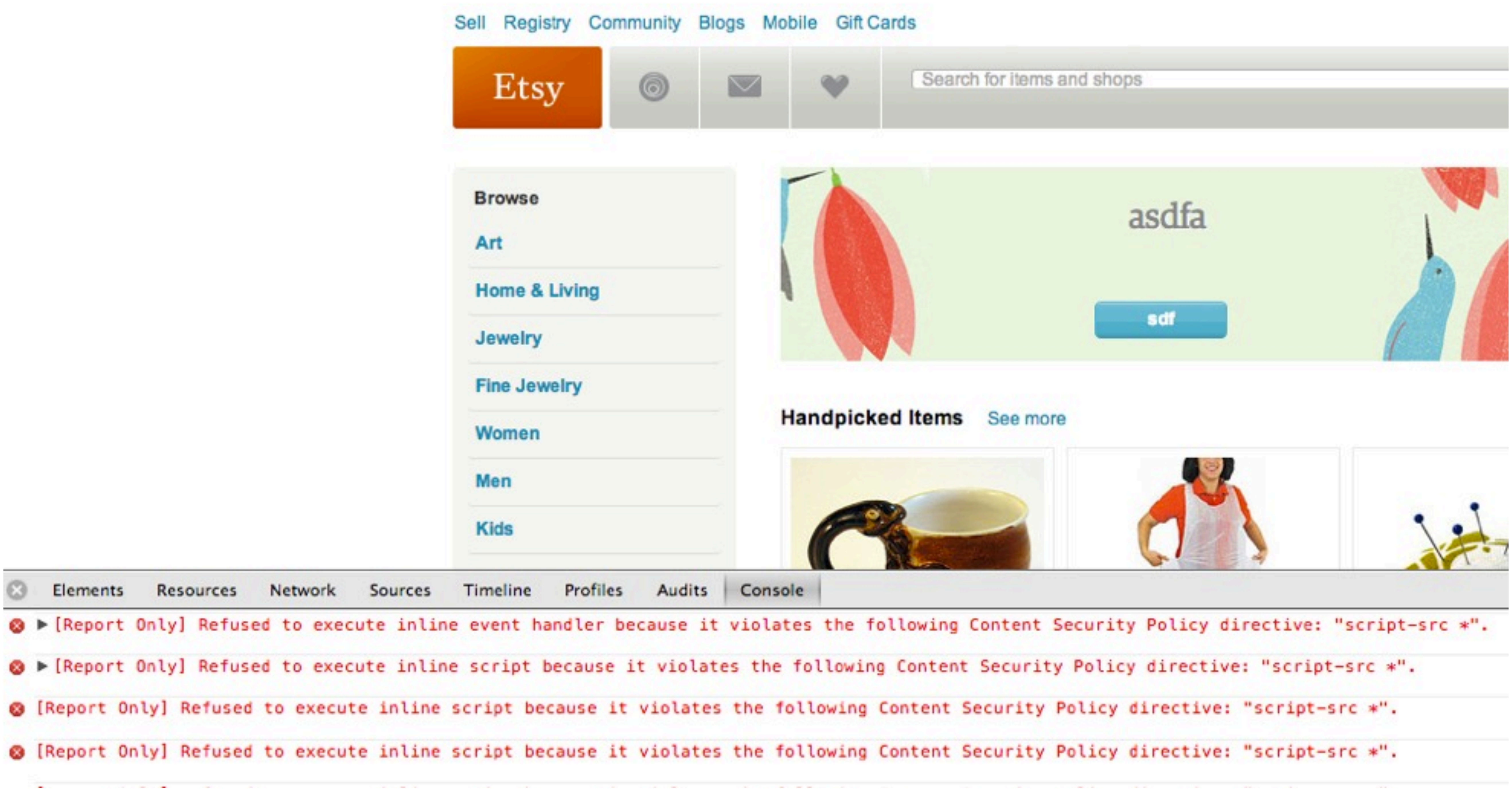
Who is this guy?

- Product Security Engineer at Etsy
- Previously worked at a financial software company
- @kennysan
- klee@etsy.com

What is CSP?

- Content Security Policy
- Browser-based XSS Defense
- http://www.w3.org/TR/CSP/

I throw this into the page's template/html: <script>alert('XSS')</script>


Sell Registry Community Blogs Mobile Gift Cards
Etsy
Search for items and shops
Browse
Art
Home & Living
Jewelry
Fine Jewelry
Women
Men
Kids
asdfa
sdf
Handpicked Items
See more
Elements Resources Network Sources Timeline Profiles Audits Console

```
▶ [Report Only] Refused to execute inline event handler because it violates the following Content Security Policy directive: "script-src *".
▶ [Report Only] Refused to execute inline script because it violates the following Content Security Policy directive: "script-src *".
[Report Only] Refused to execute inline script because it violates the following Content Security Policy directive: "script-src *".
[Report Only] Refused to execute inline script because it violates the following Content Security Policy directive: "script-src *".
```

How does it work?

- By default, browsers obeying a CSP do not execute javascript that is inline on the page
- In addition, it disallows the eval and similar functions like window.setTimeout

A sample CSP

```
Content-Security-Policy-Report-Only:default-src *; style-src * 'unsafe-inline';
script-src 'unsafe-inline' 'unsafe-eval' *.googleapis.com *.googleapis.com
*.pinterest.com *.etsystatic.com lognormal.net *.google.com *.google-analytics.com
*.etsystatic.com *.etsy.com *.etsysecure.com *.truste.com *.thinglink.me
*.thebrighttag.com *.facebook.net *.facebook.com *.thinglink.com *.tumblr.com
*.btstatic.com *.google-analytics.com *.twitter.com *.atdmt.com
*.googleadservices.com *.doubleclick.net *.flickr.com *.iesnare.com *.gstatic.com
nxtck.com *.akamaihd.net; report-uri /beacon/csp.php
```

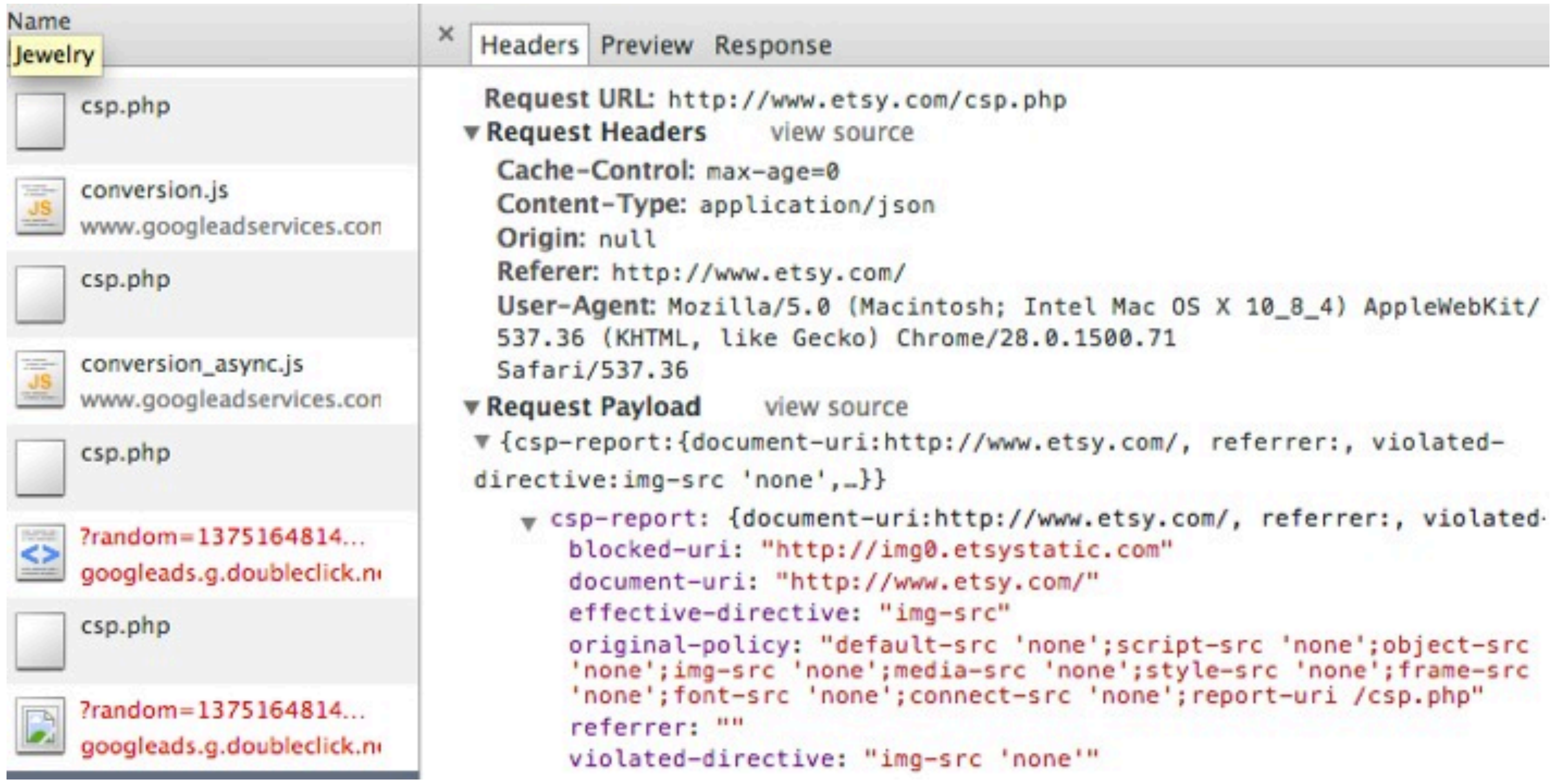
CSP directives

- connect-src
- font-src
- frame-src
- img-src
- media-src
- object-src
- style-src

- none
- self
- unsafe-inline
- unsafe-eval

report-only mode

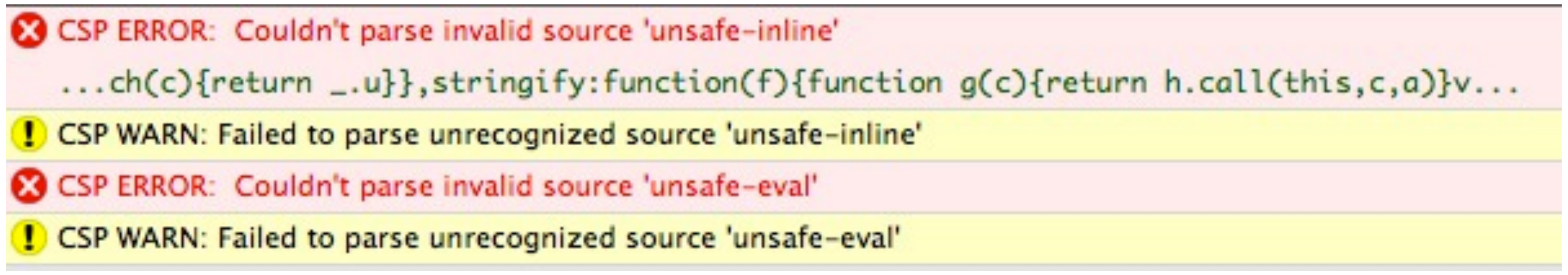
- report-uri specifies URI to POST CSP issues



```
Request URL: http://www.etsy.com/csp.php
▼Request Headers    view source
  Cache-Control: max-age=0
  Content-Type: application/json
  Origin: null
  Referer: http://www.etsy.com/
  User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_4) AppleWebKit/
  537.36 (KHTML, like Gecko) Chrome/28.0.1500.71
  Safari/537.36
▼Request Payload    view source
 ▼{csp-report:{document-uri:http://www.etsy.com/, referrer:, violated-
 directive:img-src 'none',…}}
   ▼ csp-report: {document-uri:http://www.etsy.com/, referrer:, violated
      blocked-uri: "http://img0.etsystatic.com"
      document-uri: "http://www.etsy.com/"
      effective-directive: "img-src"
      original-policy: "default-src 'none';script-src 'none';object-src
      'none';img-src 'none';media-src 'none';style-src 'none';frame-src
      'none';font-src 'none';connect-src 'none';report-uri /csp.php"
      referrer: ""
      violated-directive: "img-src 'none'"
```

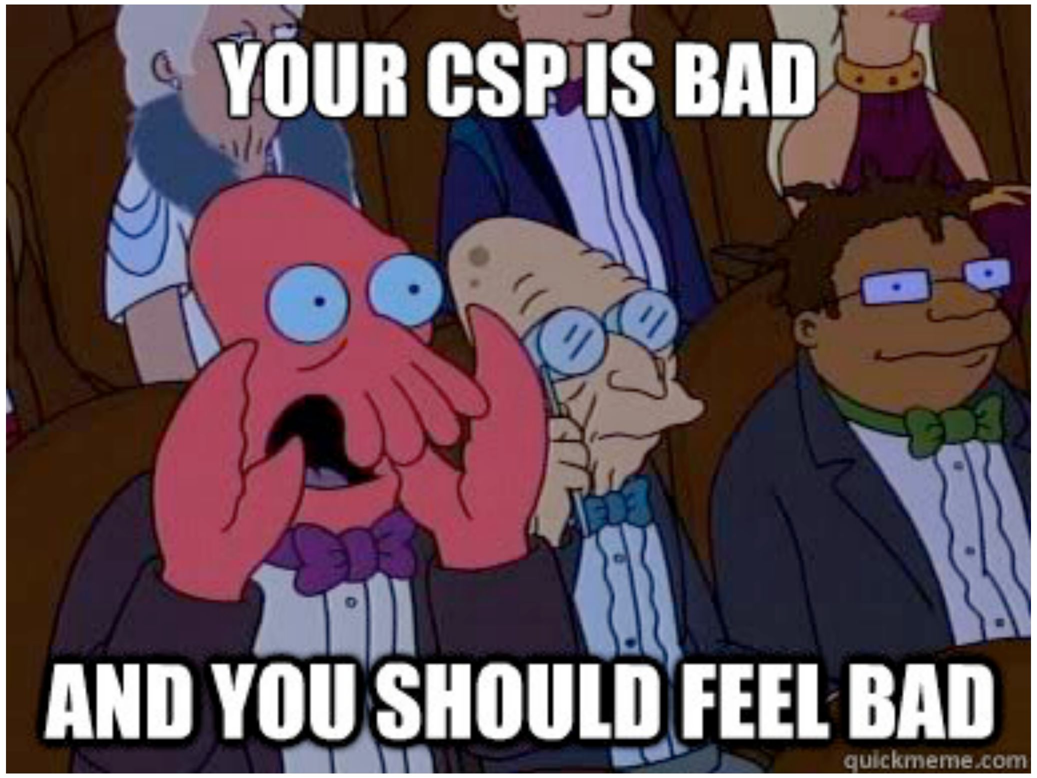
- Doesn't actually block content from loading

CSP is still evolving...



```
CSP ERROR:  Couldn't parse invalid source 'unsafe-inline'
  ...ch(c){return _.u}},stringify:function(f){function g(c){return h.call(this,c,a)}v...
CSP WARN: Failed to parse unrecognized source 'unsafe-inline'
CSP ERROR:  Couldn't parse invalid source 'unsafe-eval'
CSP WARN: Failed to parse unrecognized source 'unsafe-eval'
```

Browsers are mostly CSP 1.0 compliant these days



What about Inline JS?

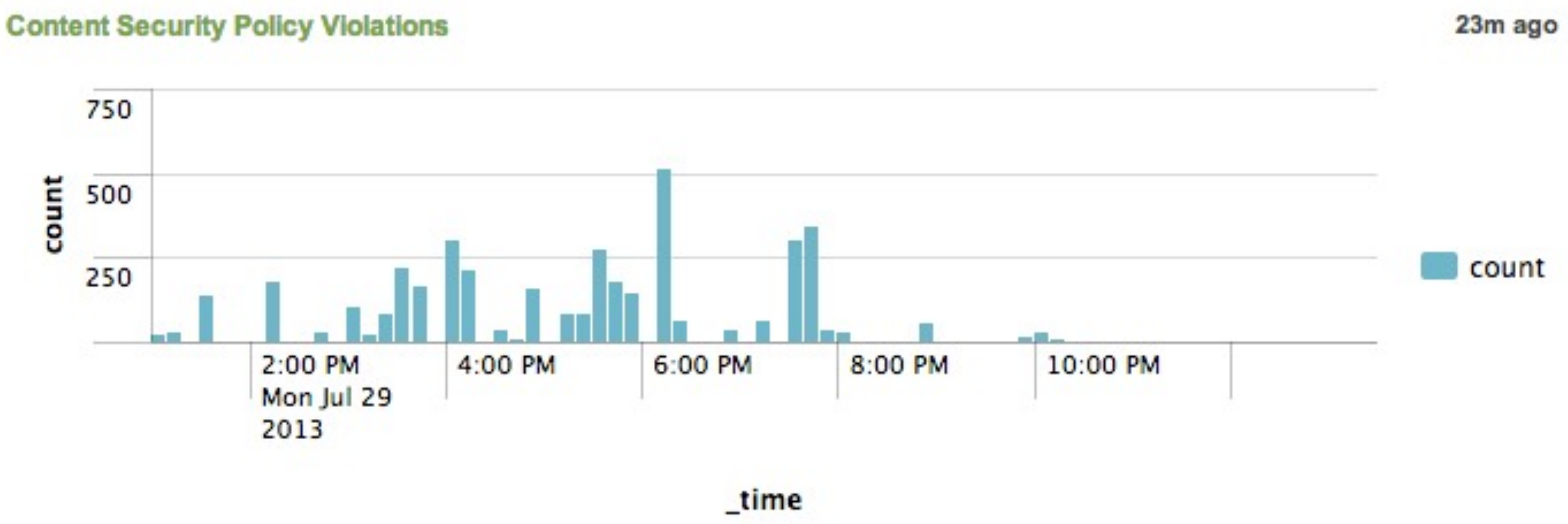
- CSP 1.0 says: create external scripts out of your inline js
- Or you can have unsafe-inline as a directive
- If you use require.js or any other async javascript library, gl/hf;
- CSP 1.1 to the rescue
- ...some day?

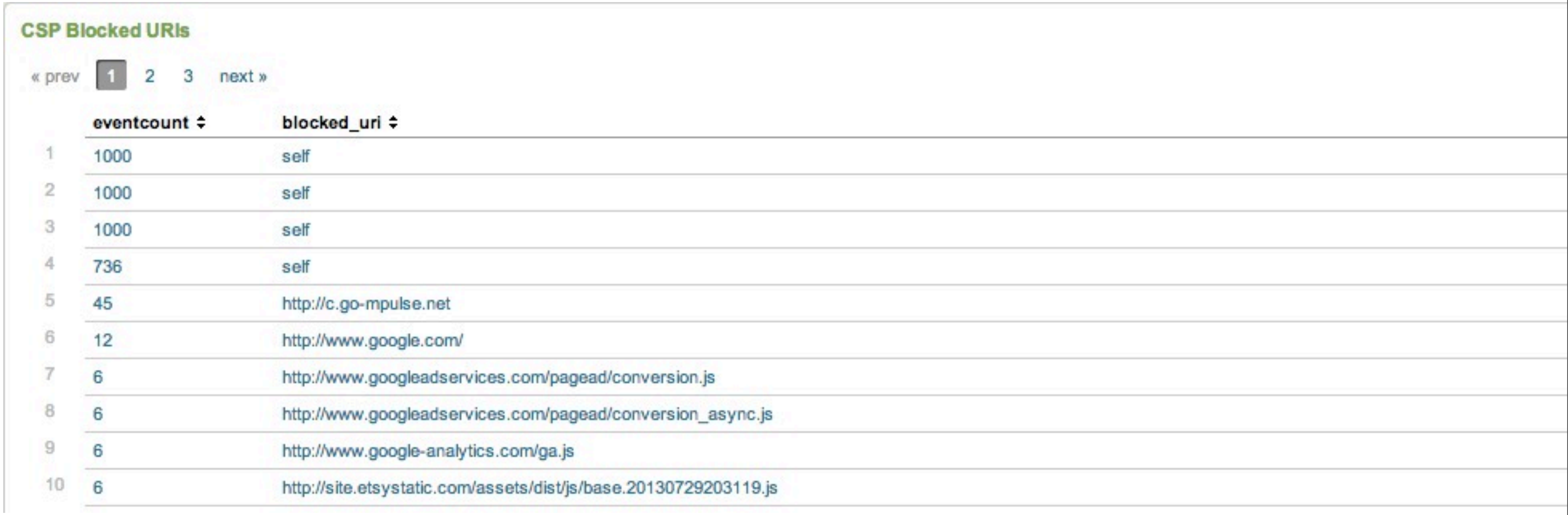
http://www.etsy.com/listing/157723652/keep-calm-and-hold-my-beer-poster-117-x

Rolling Out CSP

- How should you approach deploying CSP?
- Most sites have focused on deploying CSP to specific functionality
- Why does this make sense?

Monitor All The Things!



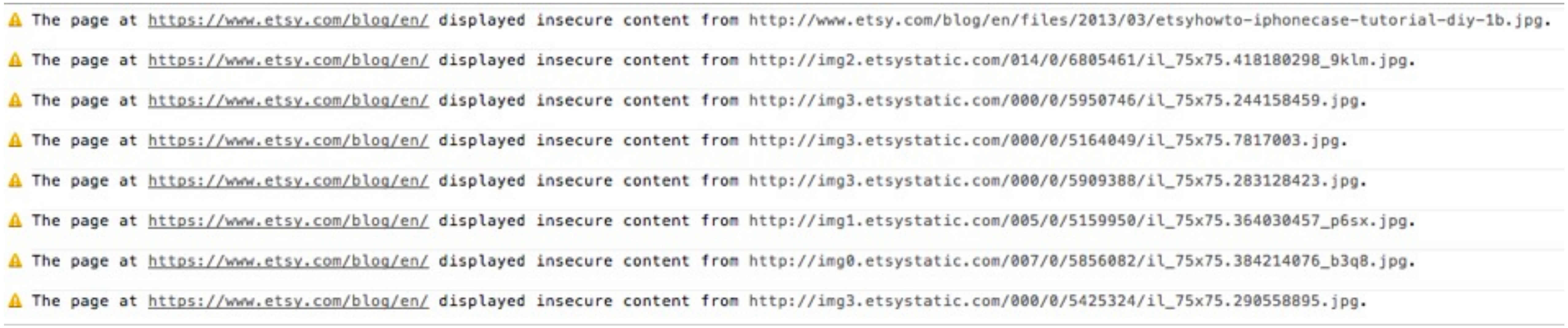


CSP Blocked URIs

« prev 1 2 3 next »

	eventcount	blocked_uri
1	1000	self
2	1000	self
3	1000	self
4	736	self
5	45	http://c.go-mpulse.net
6	12	http://www.google.com/
7	6	http://www.googleadservices.com/pagead/conversion.js
8	6	http://www.googleadservices.com/pagead/conversion_async.js
9	6	http://www.google-analytics.com/ga.js
10	6	http://site.etsystatic.com/assets/dist/js/base.20130729203119.js

Mixed Content



```
The page at https://www.etsy.com/blog/en/ displayed insecure content from http://www.etsy.com/blog/en/files/2013/03/etsyhowto-iphonecase-tutorial-diy-1b.jpg.
The page at https://www.etsy.com/blog/en/ displayed insecure content from http://img2.etsystatic.com/014/0/6805461/il_75x75.418180298_9klm.jpg.
The page at https://www.etsy.com/blog/en/ displayed insecure content from http://img3.etsystatic.com/000/0/5950746/il_75x75.244158459.jpg.
The page at https://www.etsy.com/blog/en/ displayed insecure content from http://img3.etsystatic.com/000/0/5164049/il_75x75.7817003.jpg.
The page at https://www.etsy.com/blog/en/ displayed insecure content from http://img3.etsystatic.com/000/0/5909388/il_75x75.283128423.jpg.
The page at https://www.etsy.com/blog/en/ displayed insecure content from http://img1.etsystatic.com/005/0/5159950/il_75x75.364030457_p6sx.jpg.
The page at https://www.etsy.com/blog/en/ displayed insecure content from http://img0.etsystatic.com/007/0/5856082/il_75x75.384214076_b3q8.jpg.
The page at https://www.etsy.com/blog/en/ displayed insecure content from http://img3.etsystatic.com/000/0/5425324/il_75x75.290558895.jpg.
```

- Your CSP endpoint can help you detect instances of mixed content
- HSTS can help you kill a lot of it
- ...But usually the problem won't be from your subdomains

Some Words...

- Adding unsafe-inline and unsafe-eval basically defeats CSP's ability to stop XSS.
- CSP can cause header sizes to grow very large!
- Make sure you test your policy!

CSPTools

- Content-Security-Policy ~Firefox 23, Chrome 25.
- Append Report-Only for "reporting mode"
- Add a report-uri at the end to make the browser POST a CSP violation there
- Fix all the violations, CSP all the things

CSPTools

- Want to test out a Content Security Policy, but scared to push your policy to prod?
- You hate poisoning your hosts file every time you want to test your CSP in your dev environment
- You'll love CSP Tools. I promise.

CSPTools

- Features 3 different set of tools
    - Proxy - Intercepts http, https traffic, inserts a csp header, and logs csp reports
    - Browser - auto-browse sections of your site with the proxy (can we say unit tests?)
    - Parser - Creates a csp policy based off proxy traffic

DEMO

DEMO

Get It.

- On Github: http://kennysan.github.io/CSPTools
- Found bugs? Issue a pull request!
- Hit me up on twitter! @Kennysan
- Greetz to Kai Zhong for helping me with the pythons