



**KEEP
CALM
AND
CSP**

Who am I?

- Valentinas Bakaitis
- Security consultant at Aura Information Security
- @vbakaitis on twitter



POWERED
BY KORDIA

What is XSS?

- User Supplied Text: `<script>alert('xss');</script>`
- Image with user supplied title `<img title='<>' onerror='alert('xss');' />`
- User supplied URL: ``
- User input passed to eval: `<script>eval('userParam=1; alert("xxx");')</script>`

Preventing XSS:

- Escape <>
- Escape ' or " in attributes (depending on which one is used).
- Escape space if the attribute is not quoted.
- Also \ should be escaped as it can double-escape \" to \\\" which will defeat escaping.
- Check that URLs are using HTTP or HTTPS schema and not javascript:.
- Don't pass user input to eval or SetTimeout
- Don't allow users to upload html files to the same domain
- When returning any user controllable resource (e.g. json, image, files, etc) ensure that an appropriate content type is set (don't use text/html)
- OWASP describes over 80 different common XSS vectors





**WHAT IF I TOLD
YOU**

XSS COULD BE PREVENTED

CSP to the rescue!

- Content Security Policy is a security standard introduced to prevent XSS.
- It allows the browser to restrict where scripts can originate from.

Enabling CSP

- CSP is enabled by returning Content-Security Policy header.
 - `nginx.conf add_header`
 - `apache .htaccess mod_headers`
 - `IIS web.config <customHeaders>`
 - Or return it programmatically
- E.g.: `Content-Security-Policy: default-src 'none'`

Configuring CSP

- Start with default-src 'none';
 - or default-src 'self'
- Specify other rules to make your web application work: script-src, style-src, other attributes as necessary.
- CSP encourages you to avoid inline JS and eval() - unsafe-inline and unsafe-eval
- Specify report-uri for reports

Deploying CSP

- Deploy as Content-Security-Policy-Report-Only first
- Review reports, refine it, deploy as Content-Security-Policy
- Make it stricter, keeping your old Content-Security-Policy deploy the new rules under Content-Security-Policy-Report-Only to test it.

This slide is for non devs

- BAs / Prod Owners: make CSP a requirement
- Testers: suggest CSP as improvement
- DevOps: apply CSP to your staging environment and watch people flip out


CSP 2.0

- Frame-ancestors (X-Frame-Options)
- Form-action
- Plugin-types
- Nonces + Hashes

Nonces + Hashes

- CSP: `script-src 'nonce-d41d8cd98' 'sha256-1DCfk1NYWuHM8DgTq1k0ta97gzK+oBDDv4s7woGaPIY='`
- `<script nonce='d41d8cd98'>alert('1')</script>`

Browser support

Content Security Policy 1.0  - CR

Global

79.83% + 8.35% = 88.18%

Mitigate cross-site scripting attacks by whitelisting allowed sources of script, style, and other resources.

Current aligned

Usage relative

Show all

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
8			45					4.3	
9			46					4.4	
¹ 10	12	42	47			8.4		4.4.4	
¹ 11	13	43	48	9	34	9.2	8	46	47
	14	44	49	9.1	35	9.3			
		45	50		36				
		46	51						

Browser support

Content Security Policy Level 2 - CR

Global

47.21% + 8.46% = 55.66%

Mitigate cross-site scripting attacks by whitelisting allowed sources of script, style, and other resources. CSP 2 adds hash-source, nonce-source, and five new directives

Current aligned

Usage relative

Show all

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
8			45					4.3	
9			46					4.4	
10	12	³ 42	47			8.4		4.4.4	
11	13	³ 43	48	9	34	9.2	8	46	47
	14	³ 44	49	9.1	35	9.3			
		³ 45	50		36				
		³ 46	51						

Important note

- CSP is not a replacement for data validation/escaping
- It is a defence-in-depth mechanism

Questions?

Links

- <http://www.cspplayground.com/>
- https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet
- <https://www.w3.org/TR/2012/CR-CSP-20121115/>
- <https://www.w3.org/TR/CSP2/>
- <https://w3c.github.io/webappsec-csp/>
- <http://tobias.lauinger.name/papers/csp-raid2014.pdf>