Large Scale Evaluation of Secure Headers in Wild

20214634 Sanggu Han
20213705 Yeongbin Hwang

Contents

- Motivation
- Goal
- Background
- Methodology
- Evaluation
- Limitations and Future Works
- Conclusion

Motivation

- Cross-site scripting (XSS) still exists!

More than 1,500 CVEs in this year!

Motivation

- What should we do?
- Input Sanitization

OWASP Java Html Sanitizer

Removed code attributes and elements that were not explicitly allow-listed in the policy.

Untrusted Input

```
<p>
        Check out
        <a href="https://www.bennadel.com" target="_blank" onmousedown="alert( 'XSS!' )">my site</a>.
</p>

<marquee loop="-1" width="100%">
        I am very trustable! You can totes trust me!
</marquee>

<p>
        <strong>Thanks for stopping by!</strong> <em>You Rock!</em> &amp;
        <blink>Woot!</blink>
</p>
```

Sanitized Input

Added link-spam and opener attack protection.

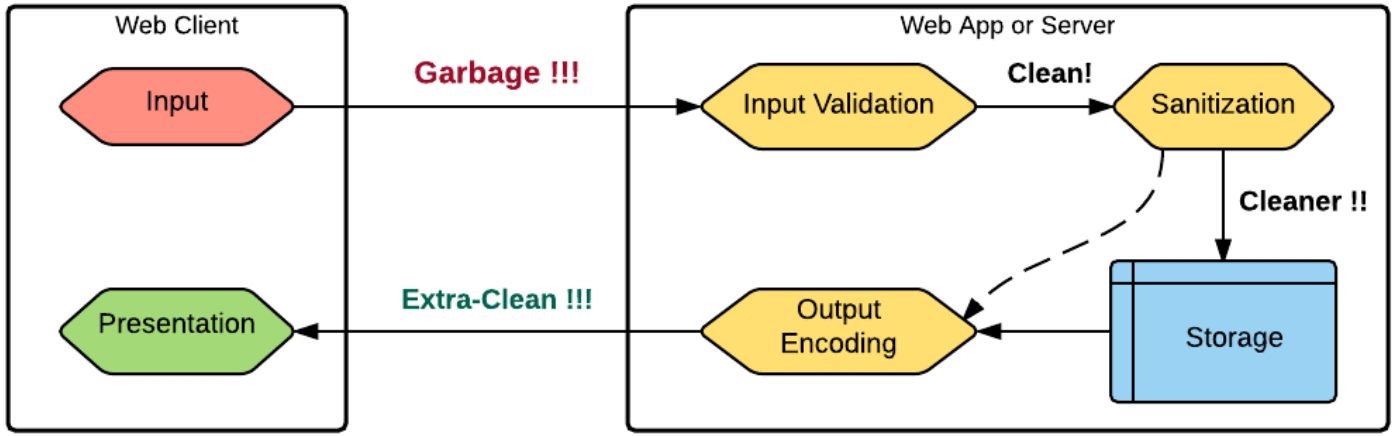
```
<p>
        Check out
        <a href="https://www.bennadel.com" target="_blank" rel="nofollow noopener noreferrer">my site</a>.
</p>


        I am very trustable! You can totes trust me!


<p>
        <strong>Thanks for stopping by!</strong> <em>You Rock!</em> &amp;
        Woot!
</p>
```

Motivation

- What should we do?
- Encode data




```
<script><alert>Hello World!</alert> </script>
```

```
&lt;script&gt;&lt;alert&gt;Hello World!&lt;/alert&gt;&lt;/script&gt;
```

Motivation

- What should we do?

- Secure headers
  - Content-Security-Policy
  - X-XSS-Protection
  - X-Frame-Options
  - X-Content-Type-Options

Motivation

- Website
  - There are a lot of websites and header fields
  - Different headers are used in each cases

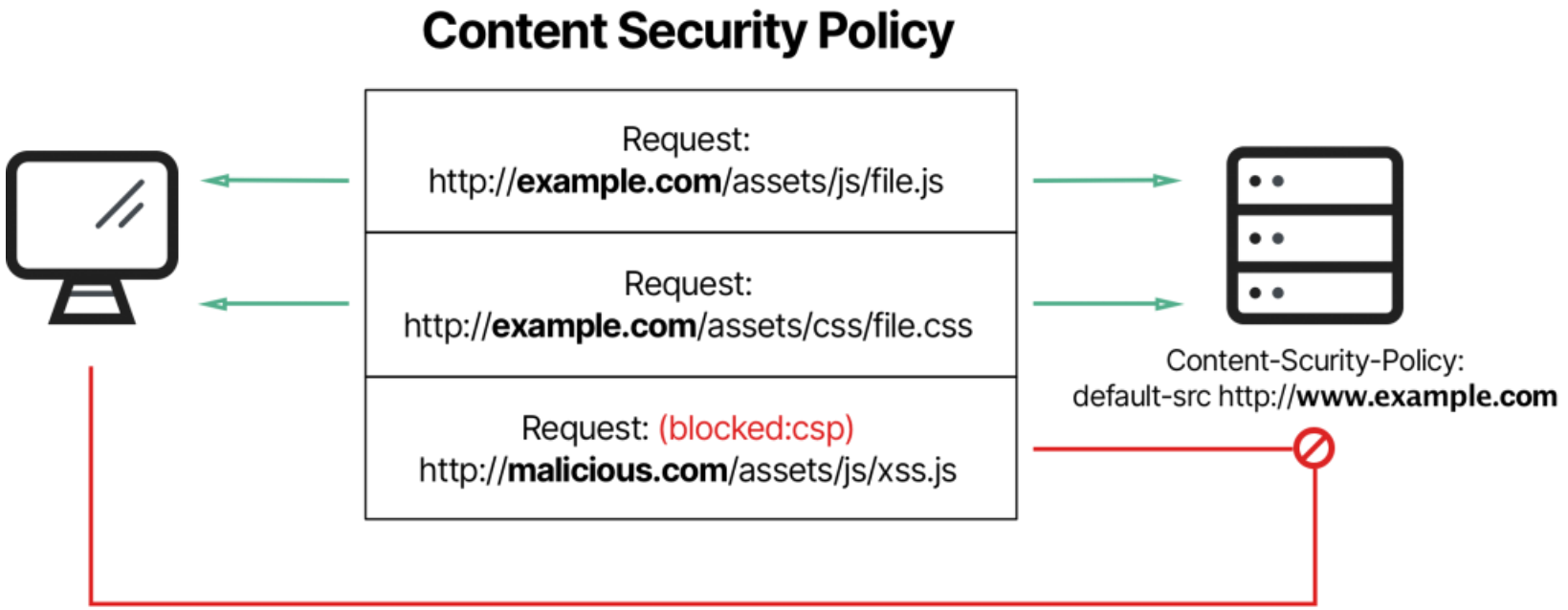
- Browser
  - Secure header supported by broswer is different.
  - Ex) Content Security Policy

Goal

- Large-scale evaluation of secure headers
  - Compare the secure headers collected in different days (several month apart)
  - Identify which fields and websites are vulnerable based on secure headers
  - Analyze the websites that can be vulnerable due to a mismatch, which is the difference between a website's request and browser support for a security header for each browser.

Background

- Content Security Policy
  - Policy that helps to detect and mitigate certain types of attacks, including Cross-Site Scripting



Background

- X-Content-Type-Options
  - Disables MIME sniffing and forces browser to use the type given in Content-Type.
- X-Frame-Options
  - Indicates whether a browser should be allowed to render a page in a `<frame>`, `<iframe>`, `<embed>` or `<object>`.
- X-XSS-Protection
  - Enables cross-site scripting filtering.

Methodology

- Define the target secure header
- Crawl secure headers in response headers
- Compare the headers between old ones and newly obtained ones
- Investigate supported headers for each browser

Define the Target Secure Header

- We considered secure headers based on 2 groups
  - Mozilla
  - OWASP Secure Headers Project

- We focused on fundamental cause
  - Javascript execution

Security headers
Cross-Origin-Embedder-Policy
Cross-Origin-Opener-Policy
Cross-Origin-Resource-Policy
Content-Security-Policy
Content-Security-Policy-Report-Only
Expect-CT
Feature-Policy
Referrer-Policy
Clear-Site-Data
Origin-Isolation
Strict-Transport-Security
Upgrade-Insecure-Requests
X-Content-Type-Options
X-Frame-Options
X-Permitted-Cross-Domain-Policies
X-XSS-Protection

Define the Target Secure Header

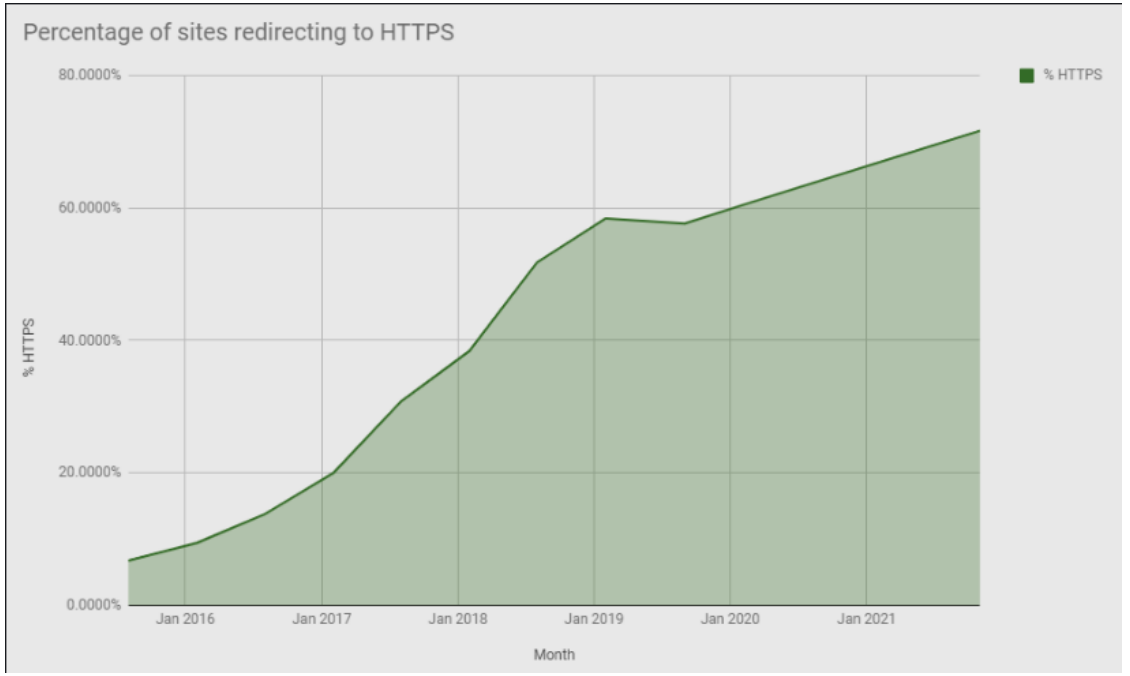
- Filter out headers related to script execution

Security headers
Cross-Origin-Embedder-Policy
Cross-Origin-Opener-Policy
Cross-Origin-Resource-Policy
Content-Security-Policy
Content-Security-Policy-Report-Only
Expect-CT
Feature-Policy
Referrer-Policy
Clear-Site-Data
Origin-Isolation
Strict-Transport-Security
Upgrade-Insecure-Requests
X-Content-Type-Options
X-Frame-Options
X-Permitted-Cross-Domain-Policies
X-XSS-Protection

Security headers (related to script)
Cross-Origin-Embedder-Policy
Cross-Origin-Opener-Policy
Cross-Origin-Resource-Policy
Content-Security-Policy
Feature-Policy
X-Content-Type-Options
X-Frame-Options
X-XSS-Protection

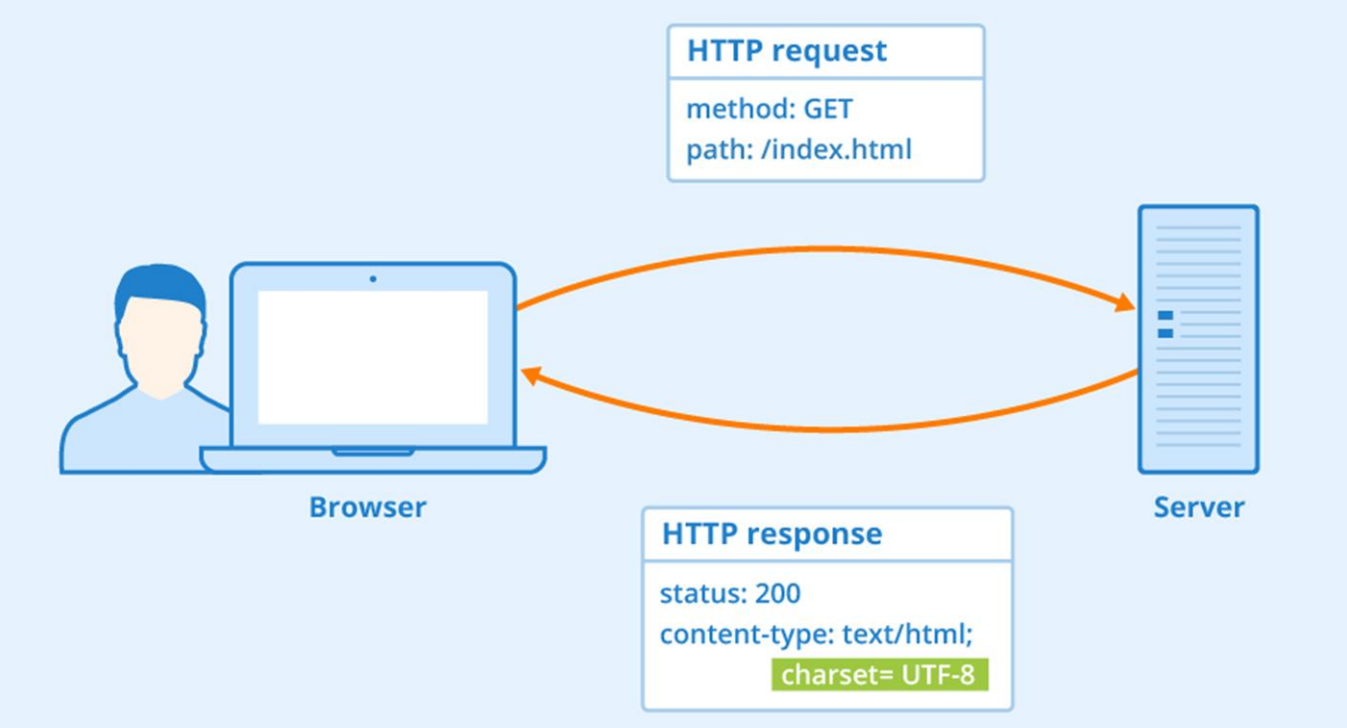
Crawl Security Headers

- Crawl the past response headers in previous crawl report.
  - https://crawler.ninja/
  - They provide raw data and statistics



Crawl Security Headers

- Implement the crawler that can extract the response headers
- Collect the response headers from alexa top 1M websites



Response Headers

accept-ranges: bytes
age: 47234
alt-svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000,h3-Q050=":443";
a=2592000,h3-Q046=":443"; ma=2592000,h3-Q043=":443"; ma=2592000,quic=
443"; ma=2592000; v="46,43"
cache-control: public, max-age=31536000
content-encoding: br
content-length: 949
content-type: text/css
cross-origin-opener-policy-report-only: same-origin; report-to="youtube"
cross-origin-resource-policy: cross-origin
date: Tue, 06 Dec 2022 15:04:36 GMT
expires: Wed, 06 Dec 2023 15:04:36 GMT

Compare the Headers

- Compare the headers between old ones and newly obtained ones
- Websites change their responses over time

SYSSEC KAIST

Investigate each Browser

- Use caniuse api that implemented in npm
- Can check browser compatibility and supported version
- We make up supported version lists
  - Based on all security headers

```
caniuse.getSupport('contentsecuritypolicy', true)
{
  and_chr: { y: 107 },
  and_ff: { y: 106 },
  and_qq: { y: 13.1 },
  and_uc: { y: 13.4 },
  android: { n: 4.2, y: 4.4 },
  chrome: { n: 13, y: 14 },
  edge: { y: 12 },
  firefox: { n: 3.6, y: 4 },
  ios_saf: { y: 6, n: 4.2, a: 5 },
  kaios: { y: 2.5 },
  op_mini: {},
  op_mob: { n: 12.1, y: 72 },
  opera: { n: 12.1, y: 15 },
  safari: { n: 5, y: 6, a: 5.1 },
  samsung: { y: 4 }
}
```

Evaluation

- Overall adoption trend
- Content-Security-Policy & X-Frame-Options
- X-XSS-Protection
- X-Content-Type-Options

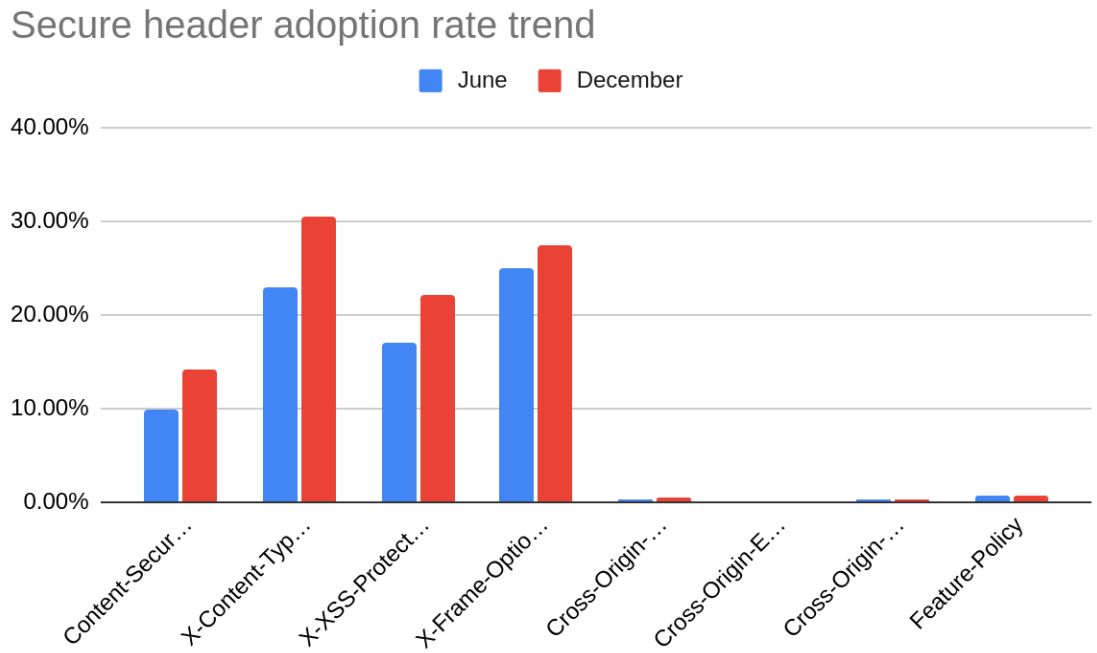
Evaluation

- Target website
  - Alexa top 1M websites
    - ≈ 820k Response header data In June (from crawl.ninja)
    - ≈ 650k Response header domains in December (by our team)

- Target browser
  - Top 15 browsers in mobile and desktop.

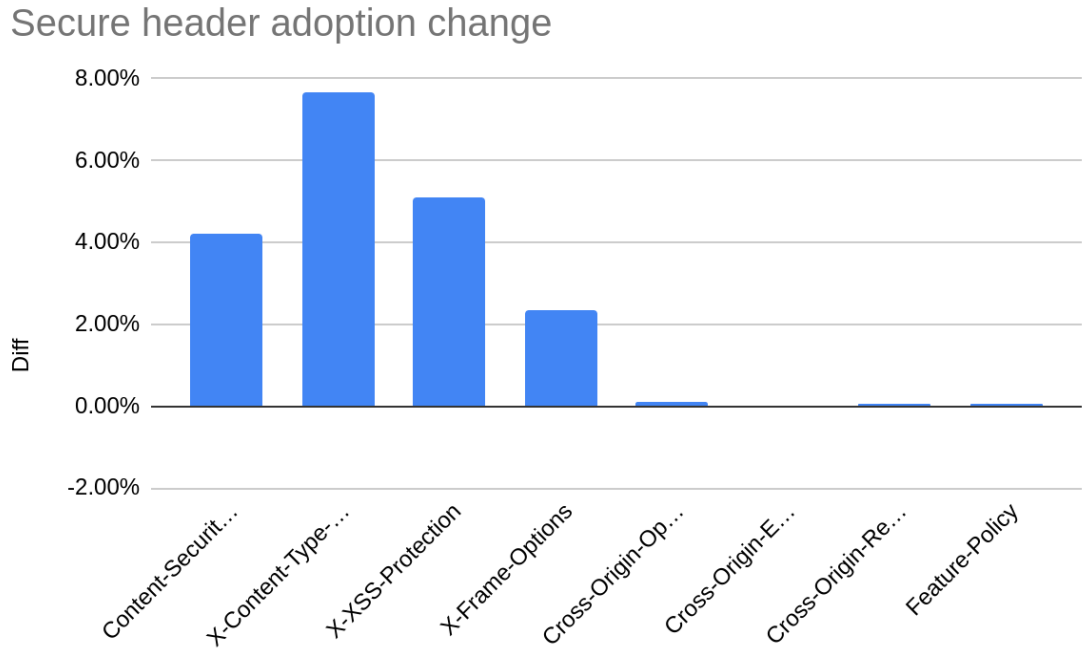
Overall Trends

- # of secure header / # of response
  - On June, December
- Secure header adoption trend
  - Increases as time goes
  - New secure headers got no spotlight



Overall Trends

- # of newly added secure header
  - On December w.r.t June
- Newly-Added secure headers barely increased

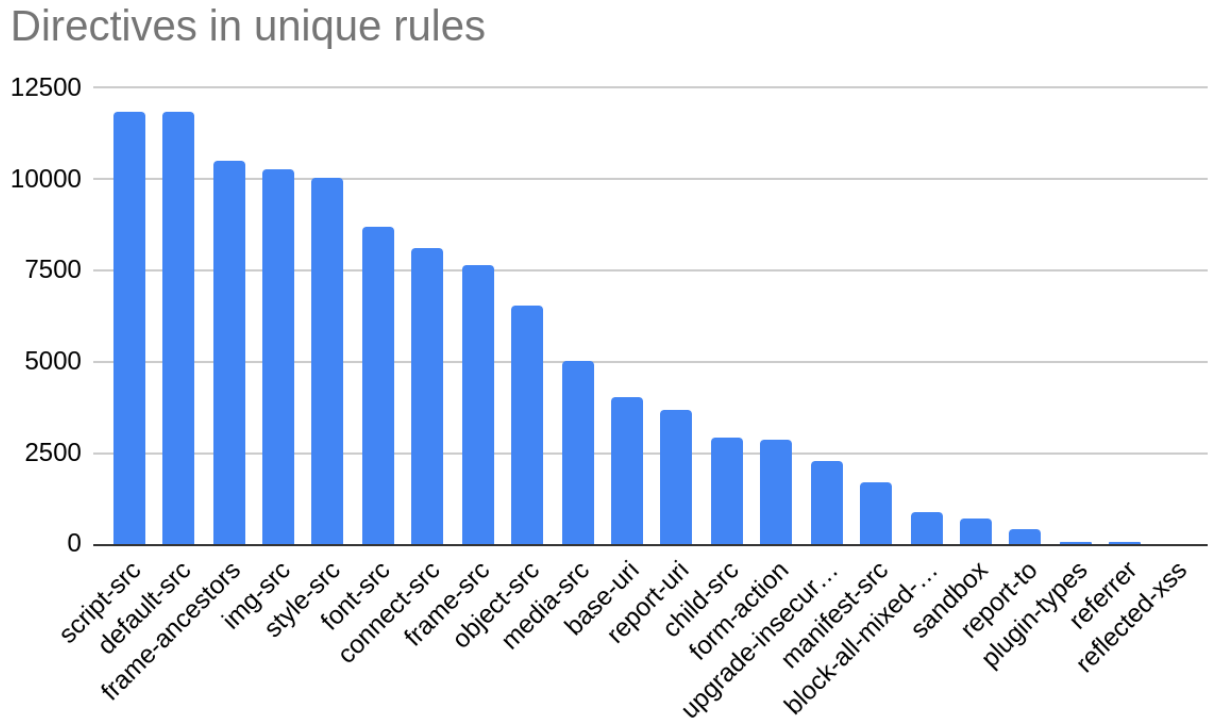


Content-Security-Policy (CSP)

- 91k of 650k domains **(≈ 14%) use CSP** in their header
- For last 6 months, **4% more domains adopted CSP**
- Collected **19786 unique rules** from crawled data
  - ≈ 12k rules were included script-src to prevent XSS

Content-Security-Policy (CSP)

- ≈ 20k unique rules
- script-src takes **≈ 60%**
  - 60% to prevent XSS



SYSSEC KAIST

Content-Security-Policy (CSP)

- CSP's *`frame-ancestor`* directive obsoletes XFO (10k cases)
- X-Frame-Options (XFO) (180k cases)
- Either one of `frame-ancestor` or XFO is set, improves protection against clickjacking
  - 186k cases observed
- So, **≈ 29% domains improved protection against clickjacking**

Content-Security-Policy (CSP)

- Most common wrong configuration case study
  - Script-src
    - Using unsafe-inline only
    - Using white-list only
  - Syntax
    - Using self, none instead of 'self', 'none'
    - Using semicolon (;) as delimiter for values

X-XSS-Protection (XXSSP)

- 143k of 650k domains **(≈ 22%) use XXSSP** in their header

- Used for protect website from XSS
  - Detect reflected XSS

X-XSS-Protection (XXSSP)

- 0 (disabled) : 5873 cases (4%)
- **1 (enabled) : 137167 cases (95.7%)**
- Wrong config: 338 cases (0.2%)
  - 0, 1
  - On
  - Sameorigin
  - …

X-Content-Type-Options (XCTO)

- 198k of 650k domains **(≈ 30%) use XCTO** in their header
- Only one option exists, *nosniff*
- Common wrong configuration
  - Values for other secure header (0, 1, script-src, …)

Browsers

- Different browser, different support

- Most of the secure headers are supported in browsers.

	CSP	XCTO	XFO	CORS	XXP	COEP	COMP	CORP
Chrome	O	O	O	O	X	O	O	O
Edge	O	O	O	O	X	O	O	O
Firefox	O	O	O	O	X	O	O	O
Safari	O	O	O	O	O	O	O	O
Safari on iOS	O	O	O	O	O	O	O	O
Android	O	O	O	O	X	O	O	O

Table - Compatibility Matrix on most recent stable version

Browsers

- **XXSSP is deprecated on every browsers except safari**

Deprecated on most browsers!

	CSP	XCTO	XFO	CORS	XXP	COEP	COMP	CORP
Chrome	O	O	O	O	X	O	O	O
Edge	O	O	O	O	X	O	O	O
Firefox	O	O	O	O	X	O	O	O
Safari	O	O	O	O	O	O	O	O
Safari on iOS	O	O	O	O	O	O	O	O
Android	O	O	O	O	X	O	O	O

Table - Compatibility Matrix on most recent stable version

XXSSP Revisited

- Case study
  - XXSSP is enabled, but CSP is not enabled (over 87k cases, **≈ 11.7%**)
  - **Vulnerable to XSS in most browsers (Chrome, Firefox, …)**
  - **11.7% of domains are potentially vulnerable to script execution!**

Limitations and Future Works

- Investigate rest of security headers
  - Other factors such as forcing HTTPS, checking certificates, … can be done with secure headers too.

- Analysis tool for response header
  - By leveraging best logics for each secure header, let devloper know what's the best for their service protection

Conclusion

- Secure headers are proposed to increase the security of web applications.
  - Prevent script execution, restrict script's permisson, …
- Deployment rates of critical secure header is insufficient
  - Especially on newly introduced ones
- There are some wrong configurations in response headers.
  - Wrong configurations still makes website vulnerable

Thank you