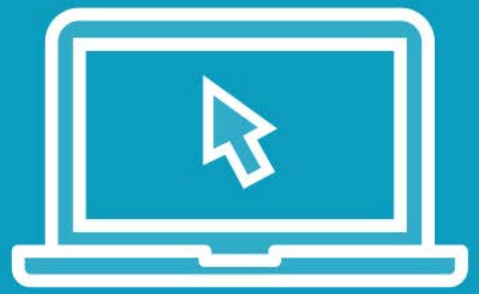
Cross-site Scripting (XSS)

Christian Wenz

@chwenz

Demo



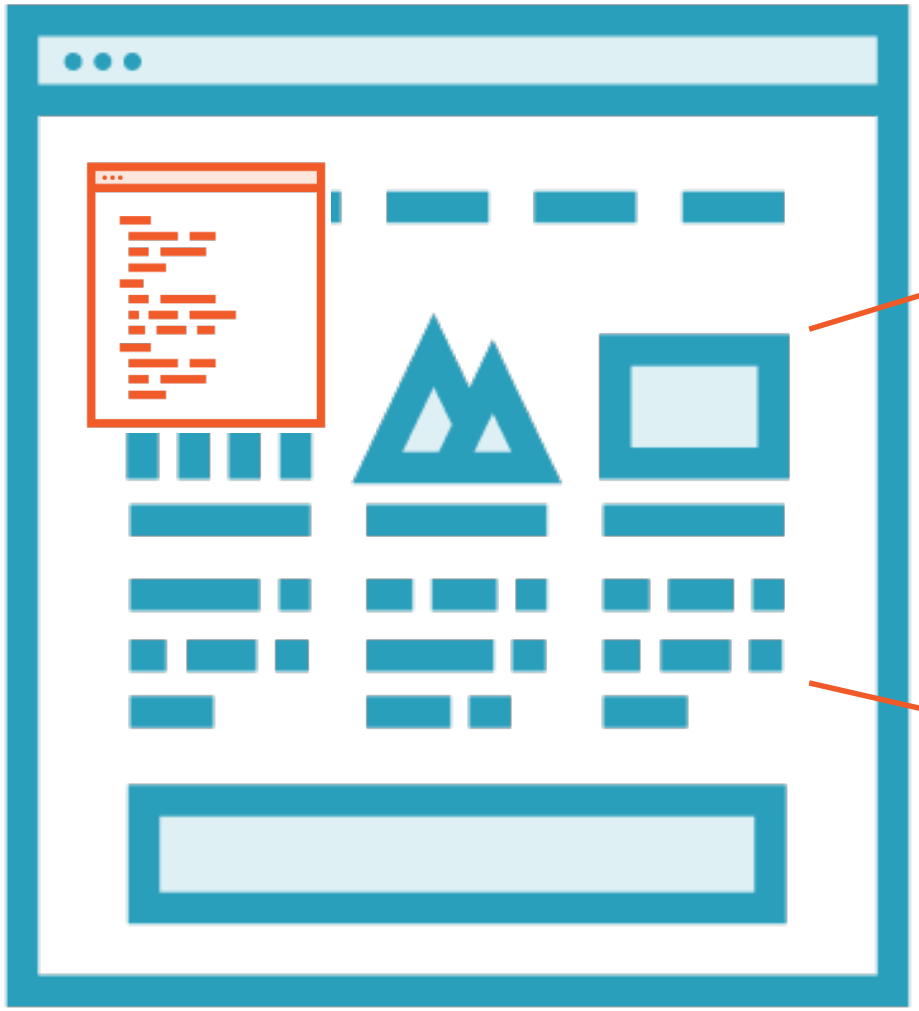
Stealing cookies

```
You searched for: <?= $_GET['q'] ?>
```

```
<input type="text" name="q"
       value="<?= $_GET['q'] ?>">
```

Anatomy of XSS

Same-origin Policy



http://example.com:80/page.php

http://example.com/script.js

Protocol

Domain

https://code.jquery.com/
jquery-x.y.z.min.js

Port

Cross-site Scripting Consequences

Cookie theft

DOM manipulation

Keylogger

Browser exploits

Anything JavaScript can do ☹

Cross-site Scripting Types

Type 1:

Stored XSS

Type 2:

Reflected XSS

Type 0:

DOM-based XSS

Filtering Input

```php
strip_tags($s)
```

```php
filter_var(
  $s, FILTER_SANITIZE_STRING)
```

```php
preg_replace($s, '/…/', '')
```

- Remove HTML tags
- Remove HTML tags with filter extension
- Home-made filtering with regular expressions

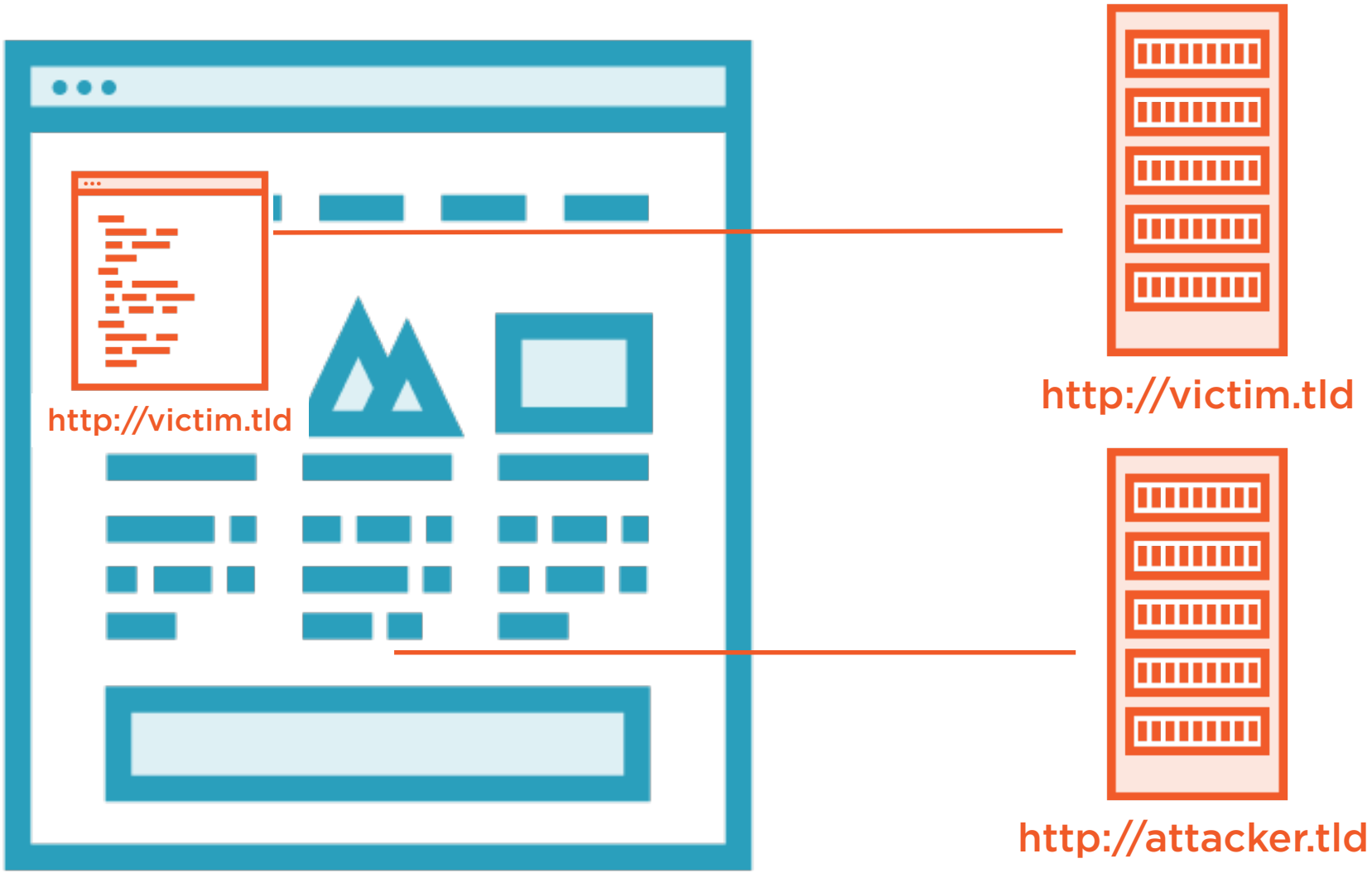
Preventing Cross-site Scripting

```
$s = '<p class="c">Let\'s go!™</p>';


echo htmlspecialchars($s);
// &lt;p class=&quot;c&quot;&gt;Let's go!™&lt;/p&gt;

echo htmlspecialchars($s, ENT_QUOTES);
// &lt;p class=&quot;c&quot;&gt;Let&#039;s go!™&lt;/p&gt;

echo htmlentities($s, ENT_QUOTES);
// &lt;p class=&quot;c&quot;&gt;Let&#039;s
   go!&trade;&lt;/p&gt;

echo filter_var($s, FILTER_SANITIZE_FULL_SPECIAL_CHARS);
```

Preventing Cross-site Scripting in JSON

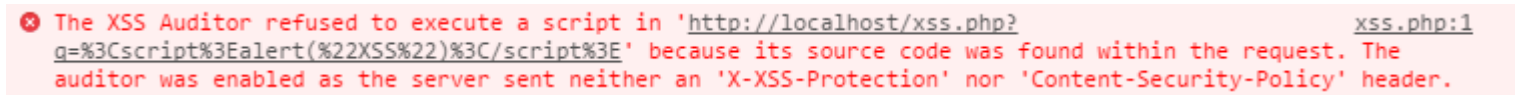
```php
$data = [
  'term' => $_GET['q']
];


header('Content-type: application/json');


echo json_encode(
  $data,
  JSON_HEX_TAG | JSON_HEX_APOS | JSON_HEX_QUOT |
  JSON_HEX_AMP);
```

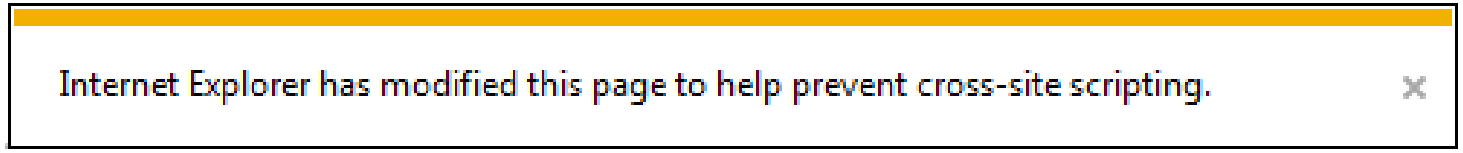
XSSI


http://victim.tld
http://victim.tld
http://attacker.tld
http://attacker.tld

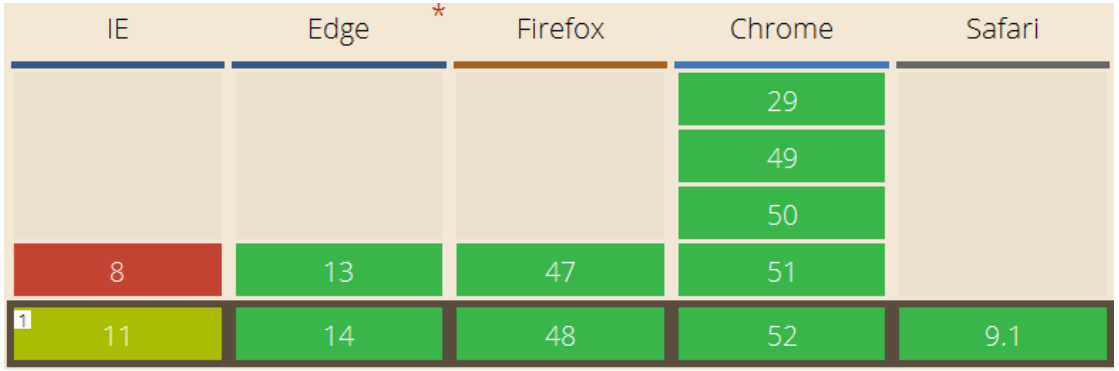
Browser XSS Protection

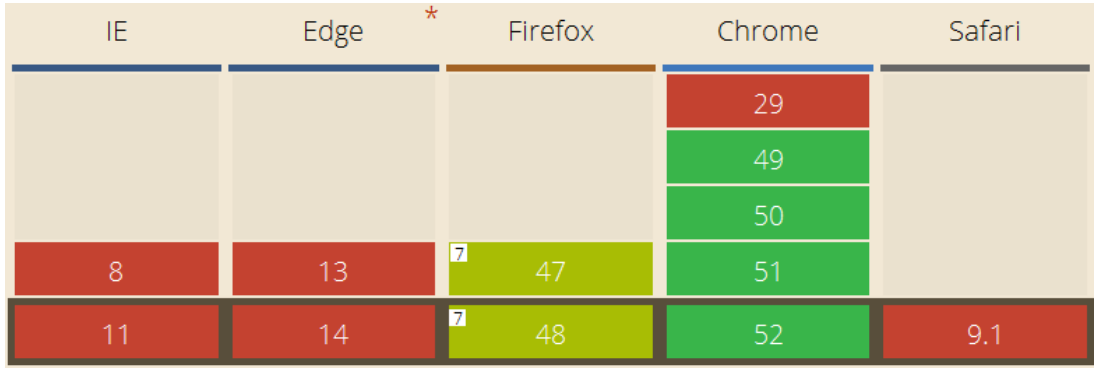
X-XSS-Protection: `1; mode=block`





Content Security Policy (CSP)





CSP 1

https://www.w3.org/TR/2012/CR-CSP-20121115/

CSP 2

https://www.w3.org/TR/CSP2/

(Some) Content Security Policy Directives

default-src	**connect-src**	**img-src**
media-src	**script-src**	**style-src**

New Content Security Policy 2 Directives

base-uri

child-src

form-action

frame-ancestors

plugin-types

Allowing Inline Code in Content Security Policy

```
Content-Security-Policy:
  script-src
    'self' 'unsafe-inline';
```

- Allow inline code (not recommended)

```
Content-Security-Policy:
  script-src
    'self' 'nonce-abc123';

<script nonce="abc123">
```

- Use a nonce (CSP 2)

```
Content-Security-Policy:
  script-src
    'self' 'sha256-AbC/D2E=';
```

- Use a hash (CSP 2)

Content Security Policy Test Run

```
Content-Security-Policy-Report-Only

  report-uri /path/to/script.php


{"csp-report":
  {"document-uri":"http://localhost/xss.php",
  "referrer":"",
  "violated-directive":"script-src 'self'",
  "effective-directive":"script-src",
  "original-policy":"script-src 'self'; …",
  "blocked-uri":"inline",
  "status-code":200}
}
```

Summary

Escape output

Make your application CSP-ready