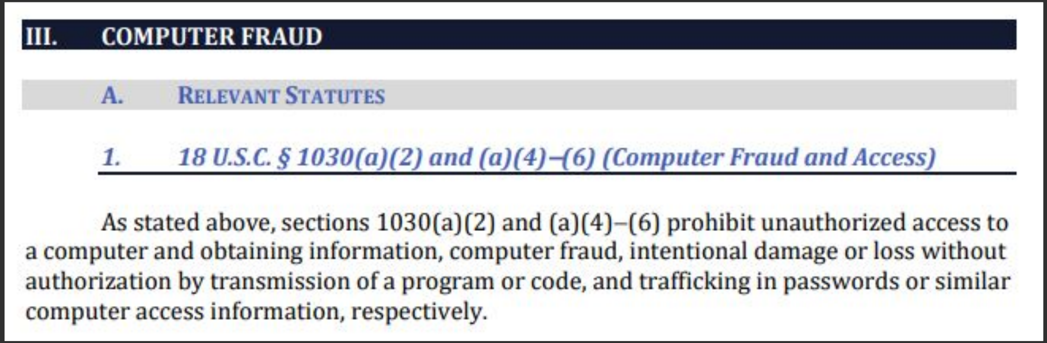
Client-Side
Web Exploitation:

Cross-Site Scripting (XSS)

UMASS CYBERSEC CLUB

Disclaimer!

Don't do this stuff without explicit permission!

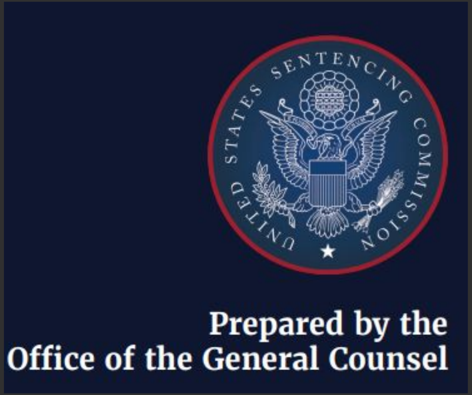


III. COMPUTER FRAUD

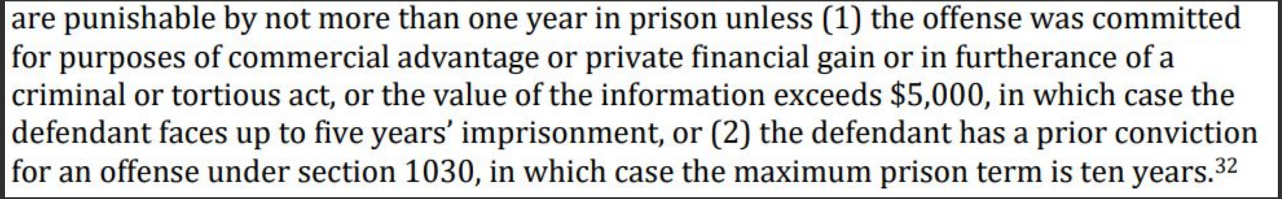
A. RELEVANT STATUTES

1. 18 U.S.C. § 1030(a)(2) and (a)(4)–(6) (Computer Fraud and Access)

As stated above, sections 1030(a)(2) and (a)(4)–(6) prohibit unauthorized access to a computer and obtaining information, computer fraud, intentional damage or loss without authorization by transmission of a program or code, and trafficking in passwords or similar computer access information, respectively.

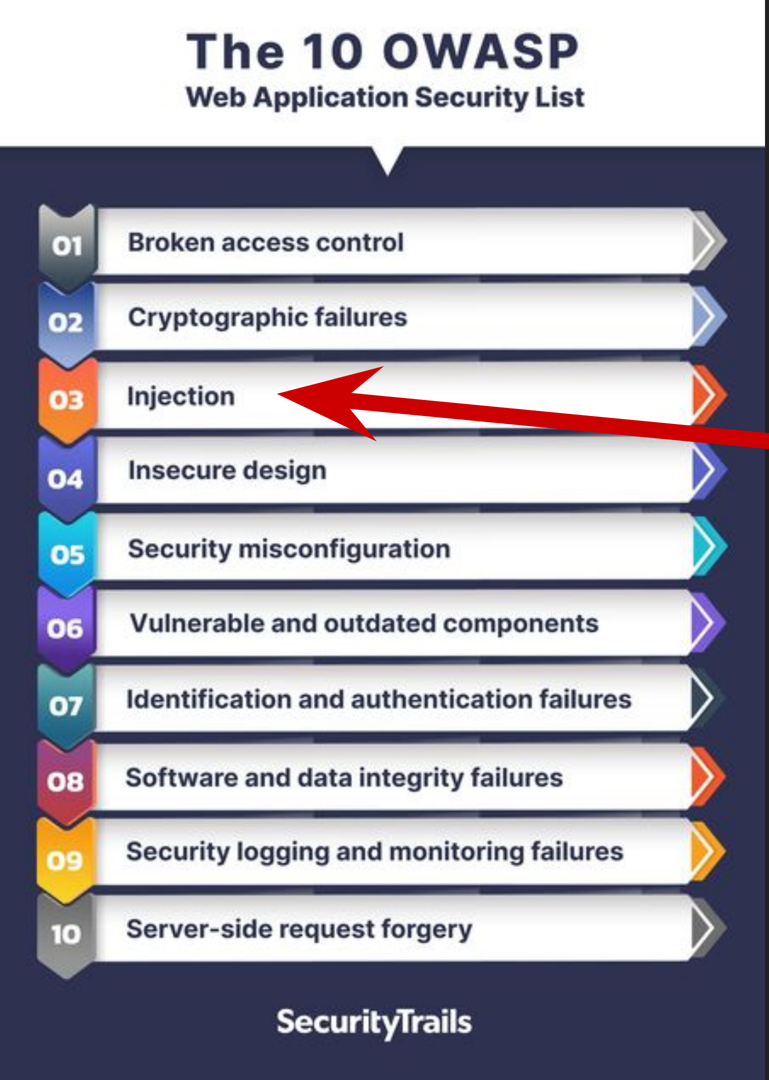


Prepared by the
Office of the General Counsel



are punishable by not more than one year in prison unless (1) the offense was committed for purposes of commercial advantage or private financial gain or in furtherance of a criminal or tortious act, or the value of the information exceeds $5,000, in which case the defendant faces up to five years' imprisonment, or (2) the defendant has a prior conviction for an offense under section 1030, in which case the maximum prison term is ten years.[32]

UMASS CYBERSEC CLUB



The 10 OWASP
Web Application Security List

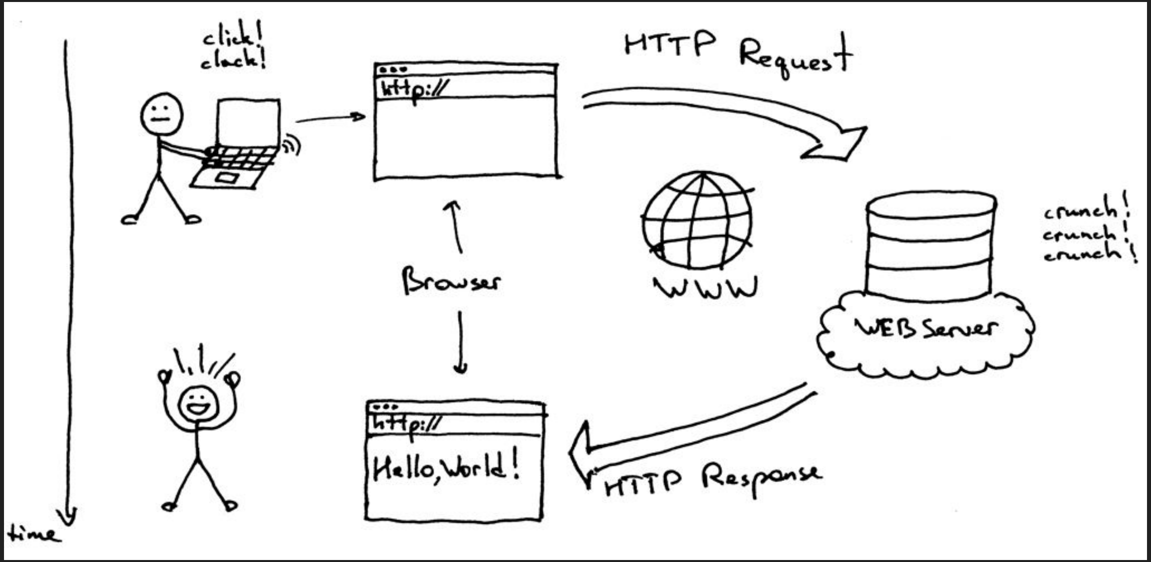
01 Broken access control
02 Cryptographic failures
03 Injection
04 Insecure design
05 Security misconfiguration
06 Vulnerable and outdated components
07 Identification and authentication failures
08 Software and data integrity failures
09 Security logging and monitoring failures
10 Server-side request forgery

SecurityTrails

XSS
Cross-Site Scripting

Some Review...

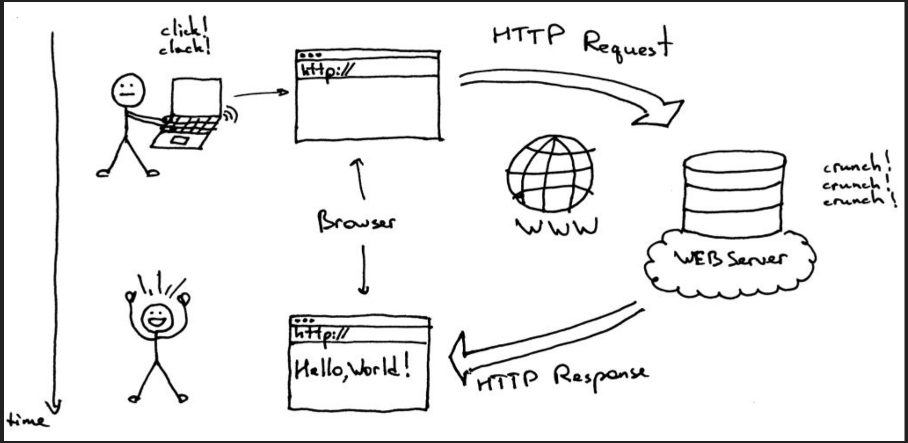
- The web (generally) functions off of a Client Server Model

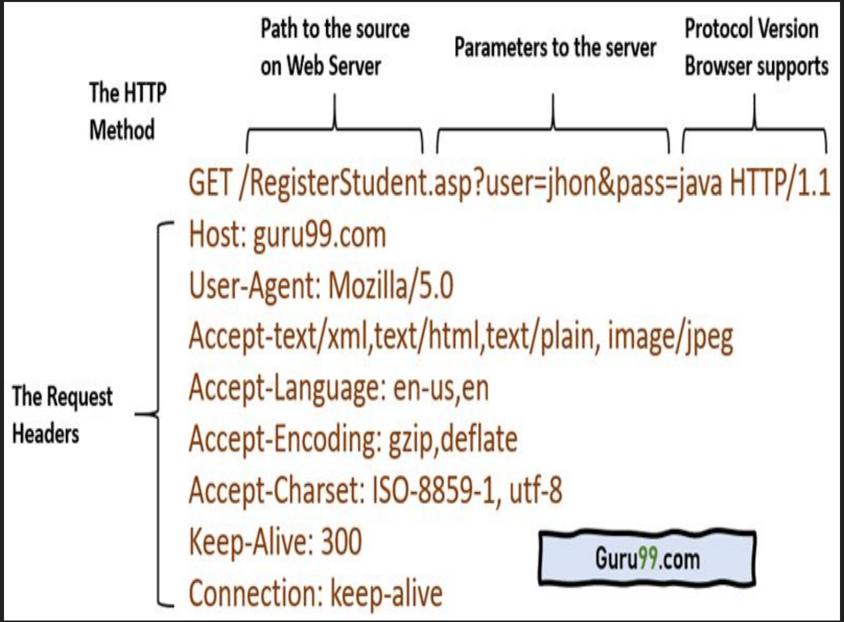


- We will be talking in terms of our **browser** being our client

Some Review...

- Typically, **HTTP** is used as the protocol to communicate between servers and clients



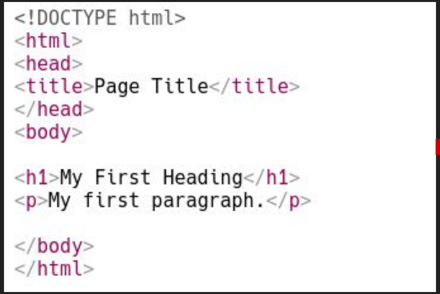


```
GET /RegisterStudent.asp?user=jhon&pass=java HTTP/1.1
Host: guru99.com
User-Agent: Mozilla/5.0
Accept-text/xml,text/html,text/plain, image/jpeg
Accept-Language: en-us,en
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1, utf-8
Keep-Alive: 300
Connection: keep-alive
```

UMASS CYBERSEC CLUB

Some Background...

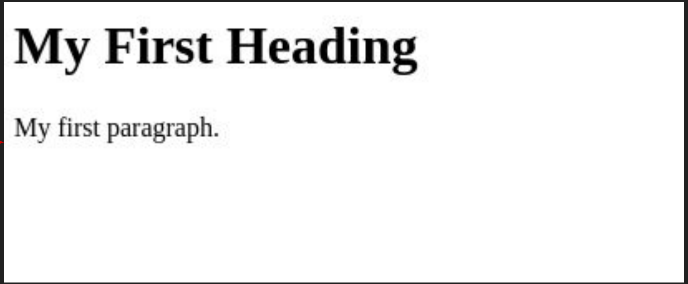
- What happens when you request a webpage?
- (Typically) server sends a **HTML** page as a response, which is **rendered** by your browser



```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```



My First Heading

My first paragraph.

Browser Requests Webpage

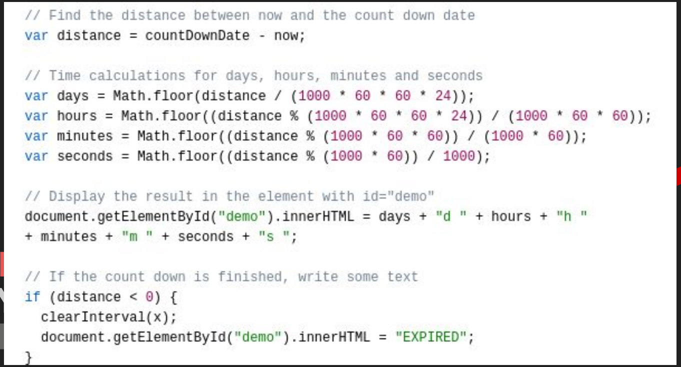
Server Sends HTML to Browser

Browser Renders HTML

UMASS CYBERSEC CLUB

Your Browser

- Client side code may also make additional requests to retrieve content or add interactive functionality
  - Mostly Javascript, but other languages are used (e.g. WASM).

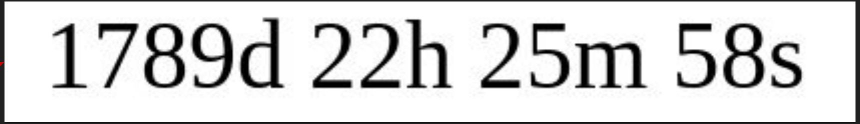


```
// Find the distance between now and the count down date
var distance = countDownDate - now;

// Time calculations for days, hours, minutes and seconds
var days = Math.floor(distance / (1000 * 60 * 60 * 24));
var hours = Math.floor((distance % (1000 * 60 * 60 * 24)) / (1000 * 60 * 60));
var minutes = Math.floor((distance % (1000 * 60 * 60)) / (1000 * 60));
var seconds = Math.floor((distance % (1000 * 60)) / 1000);

// Display the result in the element with id="demo"
document.getElementById("demo").innerHTML = days + "d " + hours + "h "
+ minutes + "m " + seconds + "s ";

// If the count down is finished, write some text
if (distance < 0) {
  clearInterval(x);
  document.getElementById("demo").innerHTML = "EXPIRED";
}
```



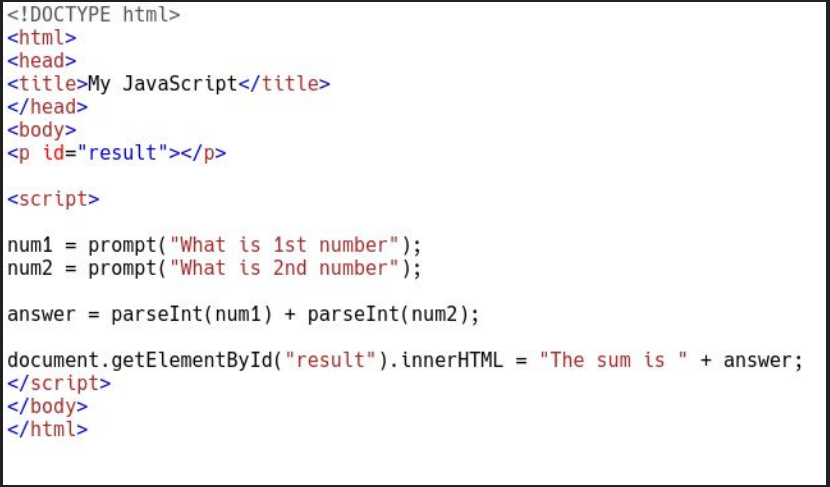
```
1789d 22h 25m 58s
```

(Pretend this is a gif)

Learn More: WASM

What is Javascript? How does our Browser use it?

- The core language of websites and the browser (what your console uses)
  - Make extra web requests, add behavior to HTML elements, & more!
- Javascript can be run in a browser with HTML script tags or event handlers
- Interpreted language
- Can also load via files



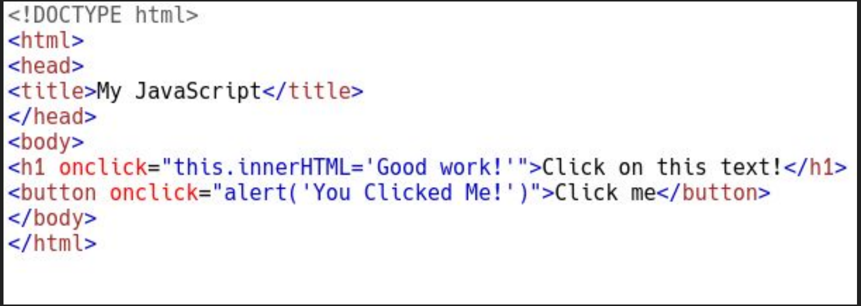
```html
<!DOCTYPE html>
<html>
<head>
<title>My JavaScript</title>
</head>
<body>
<p id="result"></p>

<script>

num1 = prompt("What is 1st number");
num2 = prompt("What is 2nd number");

answer = parseInt(num1) + parseInt(num2);

document.getElementById("result").innerHTML = "The sum is " + answer;
</script>
</body>
</html>
```



```html
<!DOCTYPE html>
<html>
<head>
<title>My JavaScript</title>
</head>
<body>
<h1 onclick="this.innerHTML='Good work!'">Click on this text!</h1>
<button onclick="alert('You Clicked Me!')">Click me</button>
</body>
</html>
```

HTML Demo

UMASS CYBERSEC CLUB

What is XSS?

Attacker makes a user run malicious browser-side code, typically scripts injected into trusted website

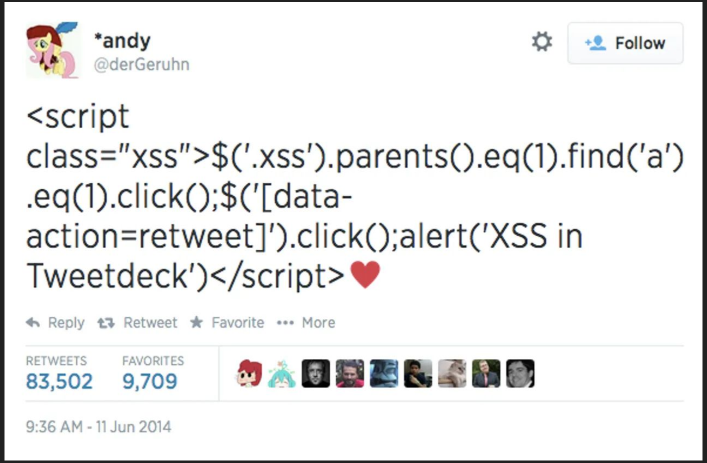
- Most often **Javascript**

e.g. Self-Retweeting Tweet

<script>alert("You've been hacked!")</script>

<svg onload=alert("You've been hacked!")>

<img src=1 onerror=alert("You've been hacked!">


*andy
@derGeruhn

```
<script
class="xss">$('.xss').parents().eq(1).find('a')
.eq(1).click();$('[data-
action=retweet]').click();alert('XSS in
Tweetdeck')</script>♥
```

RETWEETS	FAVORITES
83,502	9,709

9:36 AM - 11 Jun 2014

More on:
[Self-Retweeting Tweet](#)
[XSS](#)

UMASS CYBERSEC CLUB

This code is running on our server. Why is this bad?

- (How can we get code execution?)

```
from flask import Flask, request, redirect
app = Flask(__name__)

@app.route("/")
def index():
    user_input = request.args.get('name')
    if (user_input is None):
        return redirect("?name=hacker", code=302)
    return f"<h1> Good Evening {user_input}</h1>"
```

Discuss with your table!

training.umasscybersec.org

UMASS CYBERSEC CLUB

FLAG: UMASS{I_L0V3_X5S}

Forms of XSS

Reflected XSS

- Data from HTTP request included in response (“Reflected” back to user)

Stored XSS

- XSS Payload stored by server and sent to user

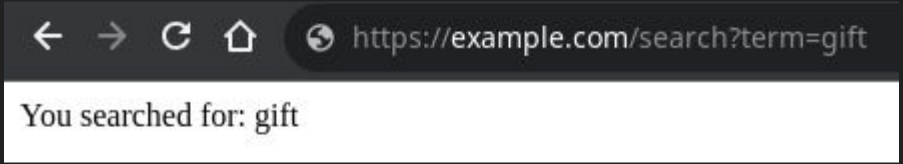
DOM XSS

- Intended Client-Side Javascript processes user input in unsafe way resulting in XSS

INSERT WITTY IMAGE HERE

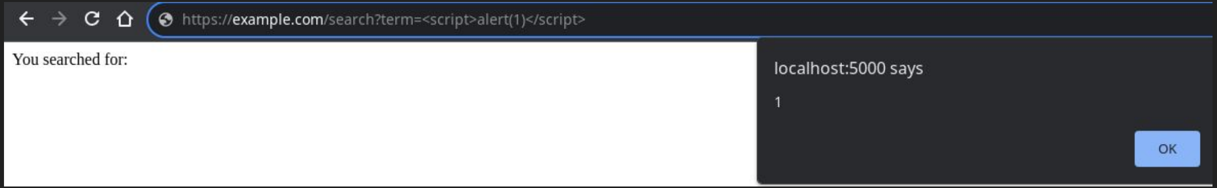
Reflected XSS

- Potentially dangerous user input is directly inserted into the HTML w/out processing
- Example:



https://example.com/search?term=gift

You searched for: gift



https://example.com/search?term=<script>alert(1)</script>

You searched for:

localhost:5000 says

1

- Any time the user visits our link with the parameters they will have the Javascript run on their client
- Common for phishing-like attacks

BABY XSS

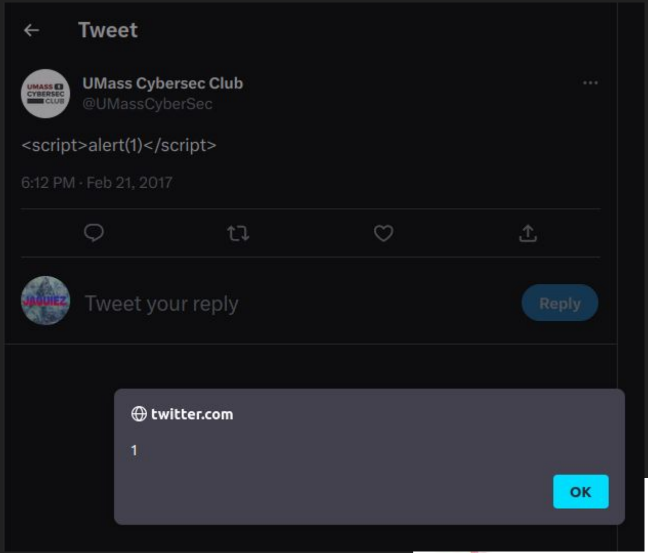
Time for your first challenge!

Try to get any popup to show!

https://training.umasscybersec.org/

Stored XSS

- User input is somehow stored on the server, which is then sent back to a victim user without need for further input
- **Example**: A post containing a payload is created on X (formerly known as Twitter) which is executed every time the attacker's posts are viewed
    - Victim's client retrieves payload stored on the server when viewing the post


Tweet

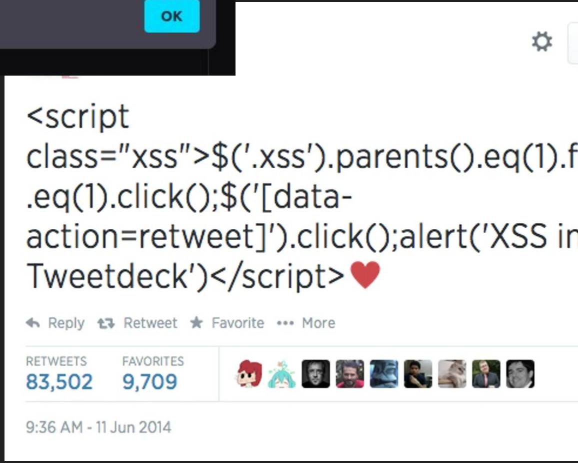
UMass Cybersec Club
@UMassCyberSec

<script>alert(1)</script>

6:12 PM · Feb 21, 2017

Tweet your reply
Reply

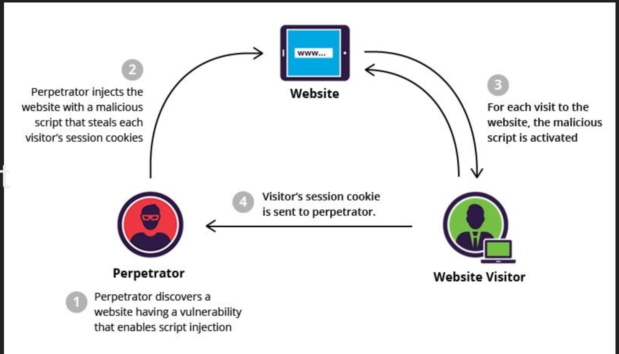
twitter.com
1
OK


```
<script
class="xss">$('.xss').parents().eq(1).f
.eq(1).click();$('[data-
action=retweet]').click();alert('XSS i
Tweetdeck')</script>
```

Reply Retweet Favorite More

RETWEETS	FAVORITES
83,502	9,709

9:36 AM - 11 Jun 2014



Website
www...
2 Perpetrator injects the website with a malicious script that steals each visitor's session cookies
3 For each visit to the website, the malicious script is activated
4 Visitor's session cookie is sent to perpetrator.
Perpetrator
Website Visitor
1 Perpetrator discovers a website having a vulnerability that enables script injection

UMASS CYBERSEC CLUB

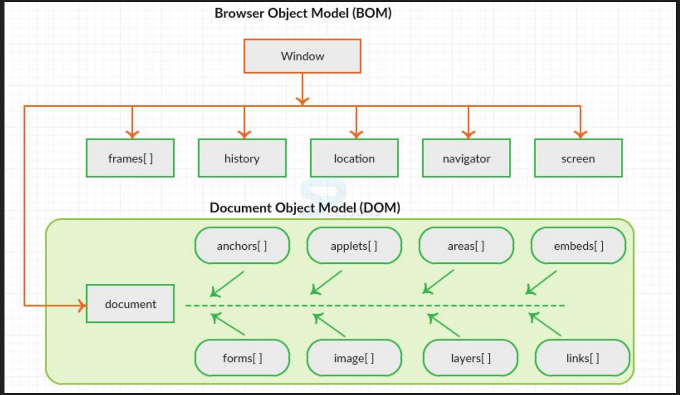
Document and Window - Javascript Objects!

document - Object that holds all other objects loaded on web page - basically the contents of your tab.

- e.g. cookies, images, forms, & other elements
- We used document.getElementById() earlier

window - Object that represents an open window in a browser - basically your tab itself.

- Is the root of your browser - the global object.
    - Document is a property of window.
    - Other properties like location (i.e. webpage's URL)
- Javascript variables are stored in the window object.
    - Don't need to write *"window.[property]"* to access properties
        - i.e. console.log(foo) = console.log(window.foo)



UMASS CYBERSEC CLUB

DOM XSS

Document Object Model (The Document, basically)

- DOM XSS is when user input is processed incorrectly, resulting in XSS.
  - Often results in payload being written to the DOM
- Javascript has unsafe functions that you should never use!
  - These are known as XSS sinks
- Example:
  - A website has code that calls **document.write(user_input)**
  - The user input will be written to the page as HTML code!

Some Bad Javascript Functions

document.write(message) - Text will directly written to the document meaning it store anything you put as HTML

document.writeln(message) - Same as above

element.innerHTML - Sets the html of an element to whatever the user supplies!

element.outerHTML - Same issue as innerHTML

element.onEvent - Will execute javascript code depending on the event. Event handlers can also be passed through HTML

UMASS CYBERSEC CLUB

XSS Cheat Sheet

alert(message) - create a popup to the user with whatever value you put as message

fetch(URL) - makes a request to the URL and returns the response

document - a Javascript object that holds all of the other objects loaded on the web page

document.cookie - an object that lets us read and write cookies to the document

https://webhook.site/ - Free website you can send requests to in order to monitor them

Include a URL Parameter to Exfiltrate Info!

Extra Tip: Script tags are only run the first time a web page is loaded. Use event handlers if your payload is only retroactively added.

UMASS CYBERSEC CLUB

BAD-SEARCHER & NOTETAKER

More Challenges!

Try Bad-Searcher first, then Notetaker

You have source this time.

https://training.umasscybersec.org/

UMASS CYBERSEC CLUB

Mitigations

Sink Definition:

- A point in a program where user-supplied data is processed

Avoid Sinks

- Avoid displaying, processing user input wherever possible!

Safe Sinks

- eg. elem.textContent = userval; document.createElement(userval);
- Wrap variables in quotes when passing to functions, e.g. <script>alert('$userval')</script>

Sanitization

- <script type="text/javascript" src="src/purify.js"></script>
- elem.innerHTML = DOMPurify.sanitize(userval);

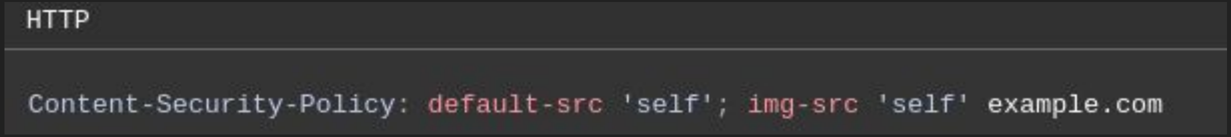
UMASS CYBERSEC CLUB

Learn More: XSS Prevention

Mitigations

CSP

- Content-Security-Policy
- Controls what resources a document is allowed to load. Commonly used to restrict Javascript resources.



```
HTTP
```

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

Only allows resources from the same origin as the document AND disallows inline execution. (No user-supplied code). Images can also come from example.com.

- Considered defense-in-depth approach, bypassable case-by-case, e.g. through DOM XSS

UMASS CYBERSEC CLUB

Learn More: CSP

One Day You will be Employed

Where might you see XSS in the real world?

APIs

oh my god the APIs

Here is a (recreation of a) reflected XSS vulnerability I saw in the real world.

https://training.umasscybersec.org/

UMASS CYBERSEC CLUB

Where can you practice more?

Portswigger Labs

CTFs

- Our CTFs, MinutemanCTF & UMassCTF
- Year-Round CTFs like PicoCTF

Last week's LACTF Mav Fan and Purell challenges are good advanced XSS challenges!

Questions?

How do I learn more?

How can I get involved?

When are you guys available?

Come Up & Ask!

Resources Posted in Discord

Newsletter

Discord

Twitter

Website