Mitigating XSS Risk

Security

Otago Polytechnic
Dunedin, New Zealand

The problem

- We saw last week that XSS vulnerabilities happen when:
  1. We trust user input to be valid and safe;
  2. We output unsafe data to the browser without processing it.
- We can also make the vulnerability worse by being too accepting of various types of HTTP requests.

It's important to recognise these patterns so that you know when to be on the alert for XSS vulnerabilities.

Vulnerable code

```
<p> Hi, <?php echo($_COOKIE[’user’]); ?> </p>
<p><img src="picard.jpg" /></p>
<p> <?php echo($_REQUEST[’secure’]); ?> </p>‘
```

Step one: processing output

- Much of the problem comes from outputting things like unwanted“`<script>`” tags to the browser.
- Usually, we don’t want to pass any HTML from user input back to the browser.
- It turns out that characters like “<” and “>” have alternative representations.
- For eample, “<” can be represented as “`&gt;`”.
- These representations are safer to send to the users’ browser.

In PHP:

Instead of

```
<?php echo($unsafe-data); ?>
```

Use

```
<?php echo(htmlentities($unsafe-data)); ?>
```