Project 3 – Web Security Part 1

CS155 – Indrajit "Indy" Khare

Outline

Quick Overview of the Technologies
- HTML (and a bit of CSS)
- Javascript
- PHP

Assignment
- Assignment Overview
- Example Attack

New to web programming?

- There are 4 different languages you need to know for this project
- Part 1 requires that you write code in at least 2 of them
- Read the FAQ!

References

- HTML, JS, and PHP
  - http://www.w3schools.com
- If you end up using AJAX (only worry about XHMLHttpRequest):
  - http://www.openjs.com/articles/ajax_xmlhttp_using_post.php
- PHP
  - http://php.net

HTML

- Hypertext Markup Language
- Used to tell the browser the structure of a document

HTML - Tags

- Surrounded by **angle brackets <>**
  - <html> - wraps entire page
  - <head> - header information, not displayed
  - <title> - displayed at top of browser
  - <body> - content that will be displayed
- Normally **come in pairs**
  - Optional for some older tags, e.g.
 (in valid XHTML that should be <br/>)
  - Can also be self-closed: <td/> = <td></td>
- **Not case sensitive**, but usually lower case

Adapted from Collin Jackson (2007)

HTML - Tags

- Headings: <h1>This text will be large</h1>
- Paragraphs: This paragraph has a margin
- Page regions: Stop it!
- Comments: <!-- This text is ignored -->
- Tables:

```
<table>
 <tr><th>Header1</th><th>Header2</th></tr>
 <tr><td>Data</td><td><Data2</td></tr>
</table>
```

Header1	Header2
Data1	Data2

Adapted from Collin Jackson (2007)

HTML - Properties

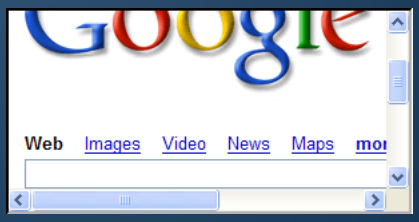
- Key=value pairs inside a tag
  -
- Can represent URLs in different ways
  - **relative:** images/hamster.gif
  - **server-relative:** /images/hamster.gif
  - **absolute:** http://animals.com/images/hamster.gif
  - Use relative where possible on Project 2
- Can delimit value with ' or " or nothing
  - (In valid XHML it must be surrounded by double quotes)

Adapted from Collin Jackson (2007)

HTML – iFrames (Helpful!)

- Embed HTML documents in other documents

```
<iframe name="myframe"
  src="http://www.google.com/">
  This text is ignored by most browsers.
</iframe>
```



Adapted from Collin Jackson (2007)

HTML – Anchor Tags

- Hyperlink navigates the browser to a URL when clicked

```
<a href="cs155.html#officehours">link text</a>
```

- Can set an anchor for a fragment

```
<a name="officehours"></a>My office hours are…
```

- Can target a frame or new window

```
<a href="http://mysite.com" target="myframe">
<a href="http://mysite.com" target="_top">
<a href="http://mysite.com" target="_parent">
<a href="http://mysite.com" target="_blank">
```

Adapted from Collin Jackson (2007)

HTML - Forms

- Provides a way to interact with server-side scripts

```
<form action="/login.php">
  <input type="text" name="username" value="">
  <input type="password" name="password" value="">
  <input type="submit" value="Log in">
</form>
```

- Use method="GET" for queries that do not send state
- Use method="POST" for requests have side effects, or passwords
- Use target="myframe" to avoid navigating away (Useful!)

Adapted from Collin Jackson (2007)

CSS

- Style information can be set as a property

```
<span style="color: red">This will be red.</span>
```

- Stylesheets set styles globally

```
<style type="text/css">
  body { margin-left: 1em; font-size: 10pt; } /* by tag */
  #par3 { font-size: 14pt; }                  /* by id */
  .conclusion { font-weight: bold; }          /* class only */
  p.conclusion { display: block }             /* by tag+class */
</style>
```

Javascript

- Browser scripting language with C-like syntax
- Sandboxed, garbage collected
- Closures

```
var x = 3; var y = function() { alert(x); };  return y;
```

- Encapsulation/objects

```
function X() { this.y = 3; } var z = new X();
alert(z.y);
```

- Can interpret data as code (eval)
- Browser-dependent

Adapted from Collin Jackson (2007)

Invoking Javascript

- Tags: `<script>alert( 'Hello world!' )</script>`
- Links: `javascript:alert( 'Hello world!' )`
  - Wrap code in "void" if it has return value
  - Beware, newlines in URIs require encoding
- Event handlers:

```
<button onclick="alert( 'Hello world!' )">
```

Adapted from Collin Jackson (2007)

Dom Manipulations

- document.**getElementByID**(id)
- document.**getElementsByTagName**(tag)
- document.**write**(htmltext)
- document.**createElement**(tagname)
- document.body.**appendChild**(node)
- document.**forms**[index].fieldname.value = …
- document.**formname**.fieldname.value = …
- frame.**contentDocument**.getElementById(id)
- someHTMLElement.innerHTML

Adapted from Collin Jackson (2007)

Other useful functions

- Navigation
  - document.**location**
  - document.formname.**submit**()
  - document.forms[0].submitfield.**click**()
- Delayed Events
  - node.**addEventListener**(eventname, handler, useCapture)
  - node.**removeEventListener**(eventname, handler, useCapture)
  - window.**setTimeout**(handler, milliseconds)

Adapted from Collin Jackson (2007)

Adding Events (Firefox)

```
<div id="foo">
  Blended Coffee
</div>

<script type="text/javascript">
var foo = document.getElementById("foo");
foo.addEventListener('click', function() {
  alert("Is freakin' awesome!");
}, false);
</script>
```

Useful CSS

var node = document.getElementById("mynodeid");

node.style.**display** = 'none'; // may not load at all

node.style.**visibility** = 'hidden'; // still takes up space

node.style.**position** = 'absolute'; // not included in flow

document.write(// can also write CSS rules to page
"<style>#mynodeid { visibility:hidden; }</style>");

Adapted from Collin Jackson (2007)

The most useful thing EVER

- Download Firebug for Firefox!
- Turn on the console for script and error logging
- Firefox is your best friend when it comes to client side development for the web

PHP

- Server scripting language with C-like syntax
- Can intermingle static HTML and code
  <input value=<?php echo $myvalue; ?>>
- Encapsulation/objects
  class X { public $y = 3; } $z = new X();
  echo $z->y;
- Can embed variables in double-quote strings
  $user = "world"; echo "Hello $user!";
  or $user = "world"; echo "Hello" . $user . "!";
- Form data in global arrays $_GET, $_POST, …

How does it work together?

OK, Let's get on with the project then!

Attack A – Cookie Theft

- Cross Site Scripting
  - Inject and execute arbitrary code on the site

Attack B

- Cross-site request forgery
  - Execute commands on another website
  - Exploits the fact that the website doesn't check who is making the request
  - Eg.

    <img src="http://bank.example/withdraw?account=bob&amount=1000000&for=mallory">

    Ref. wikipedia.org
  - In your case it's a form submission

Attack C – Reading Leaked Data

- Can't simply load in read and manipulate HTML from another domain
- What can we load directly onto our page?

Attack C - Phishing

- Making a website appear like another to steal a user's information
- Hard to defend, it's basically up to the user

Attack D

- Profile Worm
- Essentially another XSS attack
  - Read the linked description of the Myspace vulnerability

Tips

- Read the source of the web pages and all included files
  - In Firefox: Right Click > View Source
- Read the source after JS manipulation
  - In Firefox: Select a particular area
  - Then Right Click > View Selection Source
  - Or Right Click on the page > View Generated Source

Tips

- You have the PHP source! Read it to understand what's going on
- Use Firebug!

Let's do some attacking

- http://zoobar.org