Introduction to Information Security

Web Vulnerabilities
It's Everywhere

• So obviously, we want to examine it from an information security perspective

• Outline
  • HTTP
  • HTML, CSS, JS
  • PHP, SQL
Overview

• We have a server machine, with some OS, running a web server
  • That's just a program that listens on port 80/443 for requests, and generates responses
• We have a client machine, with some OS, running a web browser
  • That's just a program that sends requests to web servers, and renders their responses nicely
• The request/response protocol is called HTTP (HyperText Transfer Protocol)
• The client side (frontend) usually uses HTML (HyperText Markup Language), CSS (Cascading Style Sheets) and JS (JavaScript)
• The server side (backend) usually uses PHP (PHP: Hypertext Preprocessor) and SQL (Structured Query Language) – although a myriad of new technologies emerge (Django, Ruby on Rails, Node.js, Mongo, Redis, Elastic, Hadoop)
HTTP Request

[method] [URI] HTTP/1.1\r\n[header]\r\n\r\n[content]

- **Method**: **what to do** – mainly GET and POST (but also PUT, DELETE and some others)
- **URI** (Unique Resource Identifier): **what to do it on** – a path, like / or /index.html
- **Headers**: Host, User-Agent, Accept-Language, ...
HTTP Response

HTTP/1.1 [status code] [status message]\r\n[header\r\n]*
\r\n[content]

- Status: **success indicator** (200 OK, 500 Server Error, 404 Not Found, 302 Redirect, ...)
- Headers: Content-Type, Content-Length, Expires, ...
More on HTTP

• Cookies
  • A way for the web server to store something on the client side
  • Has a domain, a key, a value, and an expiration date
  • For example, given a username and a password, the server validates them and generates a cookie, which preserves the user's identity though the site
    • If I steal your cookie, I become you

• HTTPS
  • SSL + HTTP
**HTML**

```html
<html>
  <head>
    <title>My First Web Page</title>
  </head>
  <body>
    <p>Hello, world!</p>
  </body>
</html>
```

- `<tag>` / `tag>` or `<tag /></tag>`
- `<tag key="value" key="value" ...>`
- `<tag>content</tag>`
- In other words, XML that represents the structure of the Document Object Model (DOM)
CSS

```html
<head>
  <style>
    p  { color: #ff0000 }
    #foo { text-weight: bold }
    .bar { width: 100px; height: 100px; border-style: solid }
  </style>
  <link href="style.css" rel="stylesheet"/>
</head>

<body>
  <p>This is red</p>
  <div id="foo">This is bold</div>
  <div class="bar">This is a bordered box</div>
</body>
```

- selector { property: value }
JS

```html
<script>
    function foo() {
        if (1 + 1 >= 2) {
            alert('Hello, world!')
        }
    }
    foo()
</script>

• A full fledges (albeit, pretty weird) programming language
• JQUERY – a powerful JS library
  • `$('#foo').append('<p></p>').css('backgroundColor', 'red')`
  • `$.post('/foo', {'x': 1}, function(data) { alert(data) })`
Chrome Developer Tools

- Settings > More Tools > Developer Tools (ctrl + shift + i)
Exploits

- Cookies
  - CSRF (Cross-Site Request Forgery)
    - Login CSRF
    - Solution: SOP (Same Origin Policy)
  - Sniffing Cookies (HTTPS)
- Phishing (HTTPS?)
  - UI Redressing (Imageshack, favicon)
- XSS (Escaping?)
- Social Engineering
  - Guessing...
  - Daisy Chaining