Security in Web 2.0, AJAX, SOA

Dennis Hurst
Application Security Center, HP Software

October 2003 Copyright © 2006 HP corporate presentation. All rights reserved.

Agenda
- HTTP - How web sites work
- Fundamentals of AJAX, WebServices & SOA
- Fundamentals of web hacks (SQL Injection)
- Hacking AJAX
- Exploiting WebServices & Bridges
- Testing Security in Web 2.0

HTTP

What is HTTP?

- HTML Page
- Request
- Response

HTTP/1.1 200 OK
Date: Wed, 30 Jan 2002 02:30:17 GMT
Server: Apache/1.3.1 (Unix) (Fedora Core 3)
Content-type: text/html
Content-length: 11

<html><head></head><body><h1>Hello, world!</h1></body></html>

Client PC  Request  Server  Response

How Does Your Application Work?

- GET – Simple query string based request
- POST – Contains POST data in the body of the request
Bei Quality Center sollte als Punkt auch FT aufgeführt sein

Juergen Pilz, 21-Mar-08
The Long Wait of a Page Refresh

Web 2.0 Style Web Application

Providing a Rich User Experience
Comparison

<table>
<thead>
<tr>
<th></th>
<th>Web 1.0</th>
<th>Web 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>New content retrieved with</td>
<td>Full page refresh</td>
<td>XmlHttpRequest</td>
</tr>
<tr>
<td>During content retrieval</td>
<td>Application in undefined state</td>
<td>Application fully usable</td>
</tr>
<tr>
<td>Page layout</td>
<td>On server</td>
<td>On client</td>
</tr>
<tr>
<td>Actions that change content</td>
<td>Hyperlink Form submission</td>
<td>Any user event</td>
</tr>
<tr>
<td>Atomic unit for content</td>
<td>HyperText page Some media</td>
<td>Data</td>
</tr>
</tbody>
</table>

WebServices & SOA

- HTTP for applications, not people
- Client sends HTTP/XML Request
- Server responds with HTTP/XML Response
- Allows for a “Service Oriented Architecture”

Service Oriented Architecture (SOA)
- Built by exposing functionality through Web Services
- Allows for loose coupling of systems to create complex systems
- Solves MANY compatibility issues
- Opens some security issues

Fundamentals of web hacks

SQL Injection

SQL Injection is a technique for exploiting Web applications that use client-supplied data in SQL queries without stripping potentially harmful characters first

SQL Injection
1. Surf the Web site
2. Find dynamic pages
3. Change parameters to locate SQL Error
4. Exploit SQL Injection

SQL Injection – Vulnerable Code

Vulnerable code

```sql
sSql = sSql + ' where locationId = ' + Request('bLocation') + ';
```

URL

Syntax error converting the integer value 'AttendeeDetails' to a column of data type int.
SQL Injection – Vulnerable Code

Demo – SQL Injection

Hacking AJAX

Demo – SQL Injection against AJAX and Web Services

Exploiting WebServices & Bridges (SOA)

Data Theft Through a Bridget
- Direct access hits limitations
- Exploit trust to steal more data
- Performance enhancements only help attacker
Attacking 3rd Parties Through Bridges
- AwesomeBooks detects the XSS or SQL Injection attacks
  - AwesomeBooks: Why is BillysRareBooks SQL injecting me?
  - Another layer to hide behind

Attacking 3rd Parties Through Bridges
- Maybe 3rd party doesn’t notice at all
  - Large site with lots of requests from affiliates
  - Performs less analysis, attacks only work through bridge

Attacking 3rd Parties Through Bridges
- Auto-shunning IDS/IPS notices XSS or SQL Injection attacks
  - IPS: This site is SQL injecting me! (Blocks IP)
  - Wanted SQL injection, got DoS of aggregate site

Testing for security in Web 2.0
Similarities with traditional web sites
- Exploits are the same technique
  - SQL Injection
  - Cross Site Scripting
  - Cross Site Request Forgery
  - Authentication, Authorization, Forceful Browsing
- Must test (manipulate) request at a low (HTTP) level to see the “true” nature of the application
- GET / POST rules still apply
- At it’s core it’s an HTTP based application
  - Very little has changed in the HTTP standard in 15 years

Testing for security in Web 2.0
Differences with traditional web sites
- AJAX and Web Services are harder to manipulate
  - SOAP, XML, JSON encoded data
  - Must understand the XML/JSON and manipulate it
  - Authentication is harder in Web Services
  - Tools need to understand XML, SOAP, JSON
  - Bridged attacks
    - Web site may front and MANY other applications
    - This allows for bridged attacks
    - These are harder to understand and test
    - AJAX is based on a “framework” you don’t control so it can change on you with little to no notice.

How HP is helping
- Design
- Develop
- Quality
- Stage
- Operations

- Sec
- Design
- Training
- Security
- Code
- Testing
- Testing
- Fan
- Fail
- Monitor

- Dashboard
- Qbimpact
- WebInspect

Assessment Management Platform (AMP)

effective, efficient, and repeatable
Technology for better business outcomes