Security in Web 2.0, AJAX, SOA

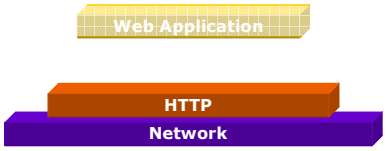
Dennis Hurst
Application Security Center, HP Software

Agenda

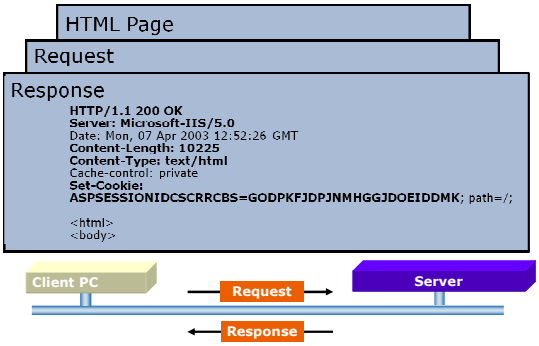
- HTTP – How web sites work
- Fundamentals of AJAX, WebServices & SOA
- Fundamentals of web hacks (SQL Injection)
- Hacking AJAX
- Exploiting WebServices & Bridges
- Testing Security in Web 2.0

HTTP

What is HTTP?



What is HTTP?



How Does Your Application Work?

- GET – Simple query string based request
- POST – Contains POST data in the body of the request

Slide 2

p1 Bei Quality Center sollte als Punkt auch FT aufgeführt sein
Juergen Pilz, 21-Mar-08

HTTP – GET With a Query String

Request

Response

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/5.0
Date: Fri, 04 Apr 2003 15:17:50 GMT
Content-Length: 4183
Content-Type: text/html
Cache-control: private
Set-Cookie: sessionid=25; path=/;
Set-Cookie: state=GA; path=/;
Set-Cookie: username=MrUser; path=/;
Set-Cookie: userid=1538; path=/;

<HTML>
<HEAD>
<TITLE></TITLE>
</HEAD>
<BODY>
```

HTTP – POST With POST Data

Form

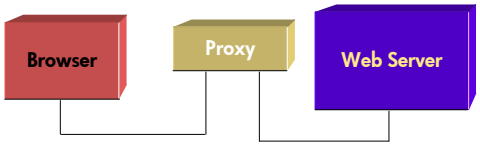
Request

Response

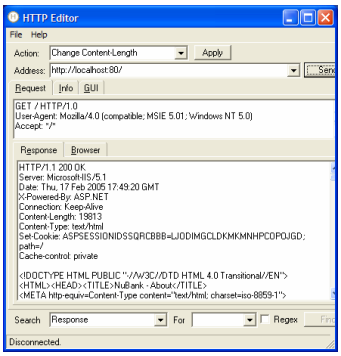
```
HTTP/1.1 200 OK
Server: Microsoft-IIS/5.0
Date: Fri, 04 Apr 2003 15:35:00 GMT
Content-Length: 80
Content-Type: text/html
Cache-control: private

<html>
<body>
Welcome John.
.....</body></html>
```

Tools – Local Host Proxies



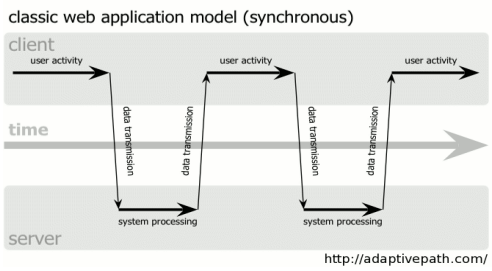
Tools – HTTP Editors



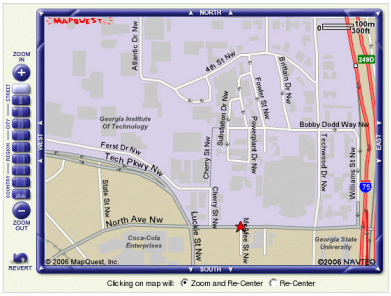
Demo – Localhost Proxies & HTTP Editors

Fundamentals of AJAX, Web Services & SOA

The Long Wait of a Page Refresh



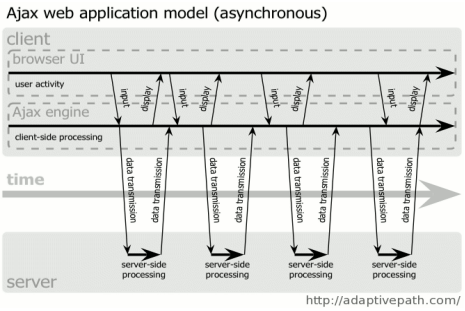
MapQuest circa 2000



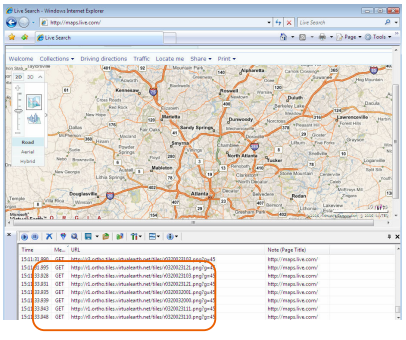
Web 2.0 Style Web Application

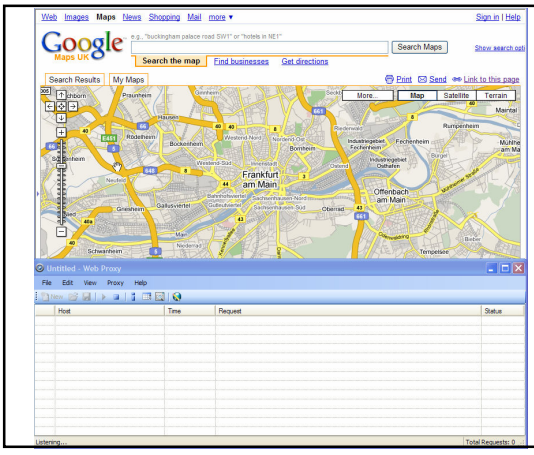
- JavaScript traps user events
- Sends HTTP request in background
- Application stays responsive
- Server returns requested data
- JavaScript processes data, dynamically changes page as needed

Providing a Rich User Experience



MapQuest circa 2000





Comparison

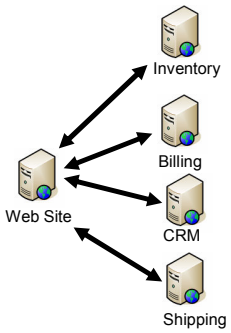
	Web 1.0	Web 2.0
New content retrieved with	Full page refresh	XmlHttpRequest
During content retrieval	Application in undefined state	Application fully usable
Page layout	On server	On client
Actions that change content	Hyperlink Form submission	Any user event
Atomic unit for content	HyperText page Some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

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

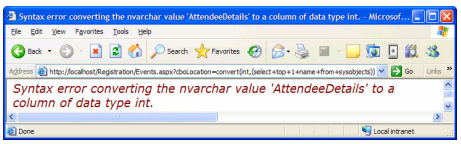
1. Surf the Web site
2. Find dynamic pages
3. Change parameters to locate SQL Error
4. Exploit SQL Injection

SQL Injection – Vulnerable Code

Vulnerable code

```
sSql = sSql + " where LocationID = " + Request["cboLocation"] + "";
oCmd.CommandText = sSql;
```

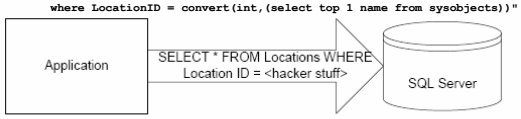
URL



SQL Injection – Vulnerable Code

Debug View

```
? oCmd.CommandText
"SELECT EventName, EndDate, [Description], [Location], …….
   from Events
   where LocationID = convert(int,(select top 1 name from sysobjects))"
```



Demo – SQL Injection

Hacking AJAX

Demo – SQL Injection against AJAX and Web Services

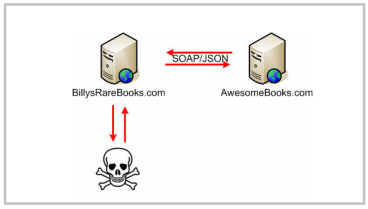
Exploiting WebServices & Bridges (SOA)

Data Theft Through a Bridget

- Direct access hits limitations
- Exploit trust to steal more data
- Performance enhancements only help attacker

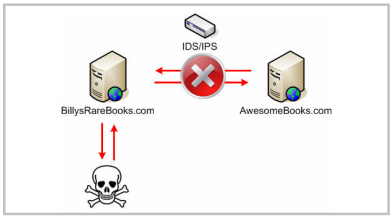
Attacking 3rd Parties Through Bridges

- AwesomeBooks detects the XSS or SQL Injection attacks
  - AwesomeBooks: *Why is BillysRareBooks SQL injecting me?*
  - Another layer to hide behind



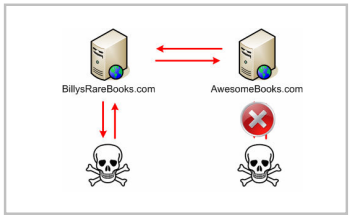
Attacking 3rd Parties Through Bridges

- 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



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



Testing for security in Web 2.0

Similarities with traditional web sites

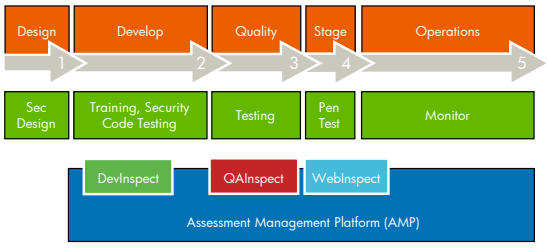
- Exploits are the same technique
  - SQL Injection
  - Cross Site Scripting
  - Cross Site Request Forgery
  - Authentication, Authorization, Forceful Browsing
- Must test (manipulate) request at a low (HTTP) level to see the "true" nature of the application
- GET / POST rules still apply
- At it's 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 date
  - 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 end 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



effective, efficient, and repeatable