You Are Hacked ☹: AJAX Security Essentials for Enterprise Java™ Technology Developers

Karthik Shyamsunder

Principal Engineer
VeriSign, Inc.

James Gould

Principal Engineer
VeriSign, Inc.

TS-6014

Speaker Qualifications

- Karthik Shyamsunder
  - Principal Engineer, VeriSign, Inc.
  - Adjunct Faculty at Johns Hopkins University
- James Gould
  - Principal Engineer, VeriSign, Inc.
  - Application Architect for VeriSign Naming Services

Overall Presentation Goal

What will you learn?

1. Understand the AJAX security model.
2. Identify the various threats to your AJAX web applications.
3. Learn what you can do to protect your application from these threats.

Agenda

Internet Threat Model

Browser Security Model

Vulnerabilities, Attacks, and Countermeasures

Secure Software-Development Process

Summary

Q&A

Agenda

Internet Threat Model

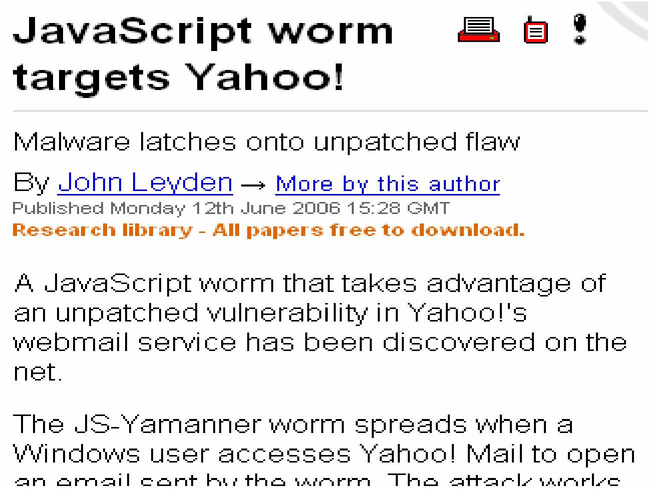
Browser Security Model

Vulnerabilities, Attacks, and Countermeasures

Secure Software-Development Process

Summary

Q&A



JavaScript worm targets Yahoo!

Malware latches onto unpatched flaw

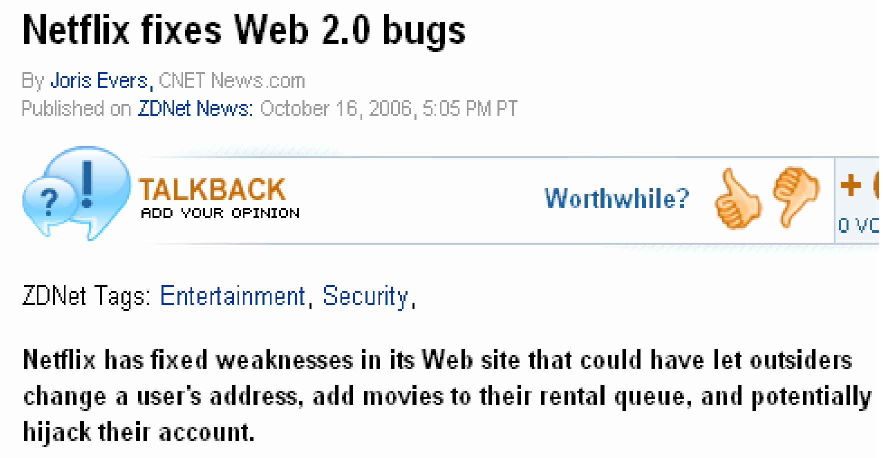
By John Leyden → More by this author

Published Monday 12th June 2006 15:28 GMT

Research library - All papers free to download.

A JavaScript worm that takes advantage of an unpatched vulnerability in Yahoo!'s webmail service has been discovered on the net.

The JS-Yamanner worm spreads when a Windows user accesses Yahoo! Mail to open an email sent by the worm. The attack works

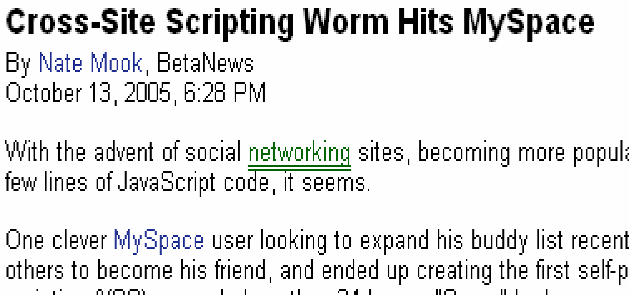


Netflix fixes Web 2.0 bugs

By Joris Evers, CNET News.com
Published on ZDNet News: October 16, 2006, 5:05 PM PT

ZDNet Tags: Entertainment, Security,

Netflix has fixed weaknesses in its Web site that could have let outsiders change a user's address, add movies to their rental queue, and potentially hijack their account.

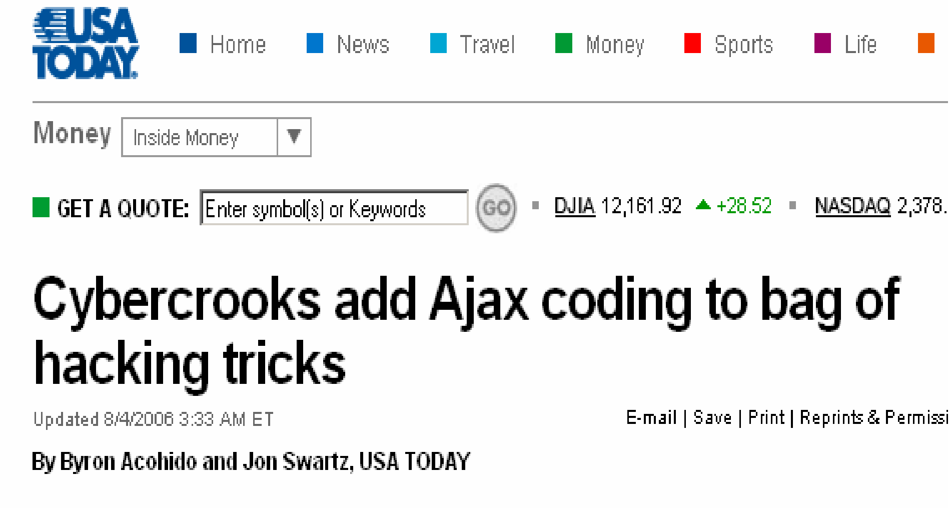


Cross-Site Scripting Worm Hits MySpace

By Nate Mook, BetaNews
October 13, 2005, 6:28 PM

With the advent of social networking sites, becoming more popul few lines of JavaScript code, it seems.

One clever MySpace user looking to expand his buddy list recent others to become his friend, and ended up creating the first self-p



Cybercrooks add Ajax coding to bag of hacking tricks

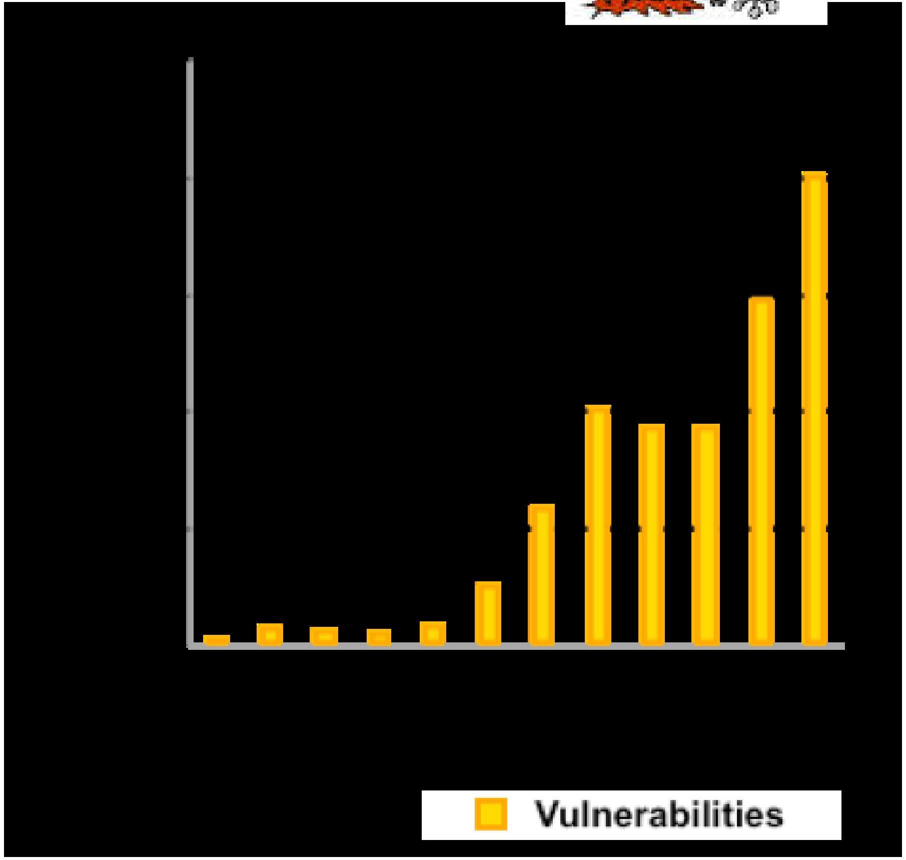
Updated 8/4/2006 3:33 AM ET

E-mail | Save | Print | Reprints & Permissi

By Byron Acohido and Jon Swartz, USA TODAY

The Problem Is Real

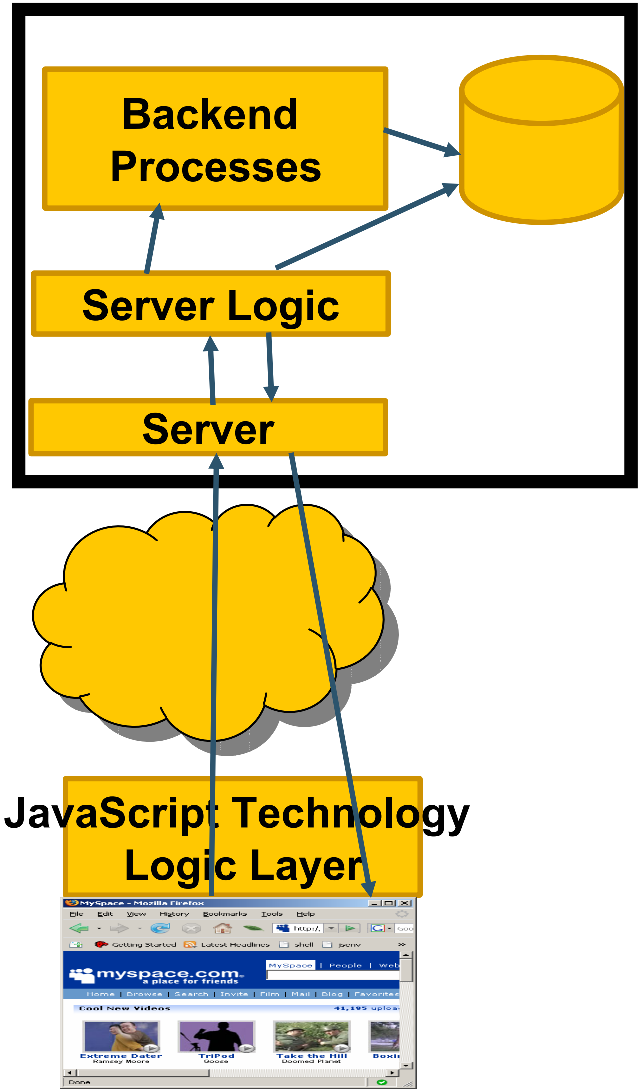
- Cyber crimes and incidents are on the rise
- 3 out of 4 business web sites are vulnerable to attack (Gartner)
- 75% of the hacks occur at the application level (Gartner)



Source: Gartner

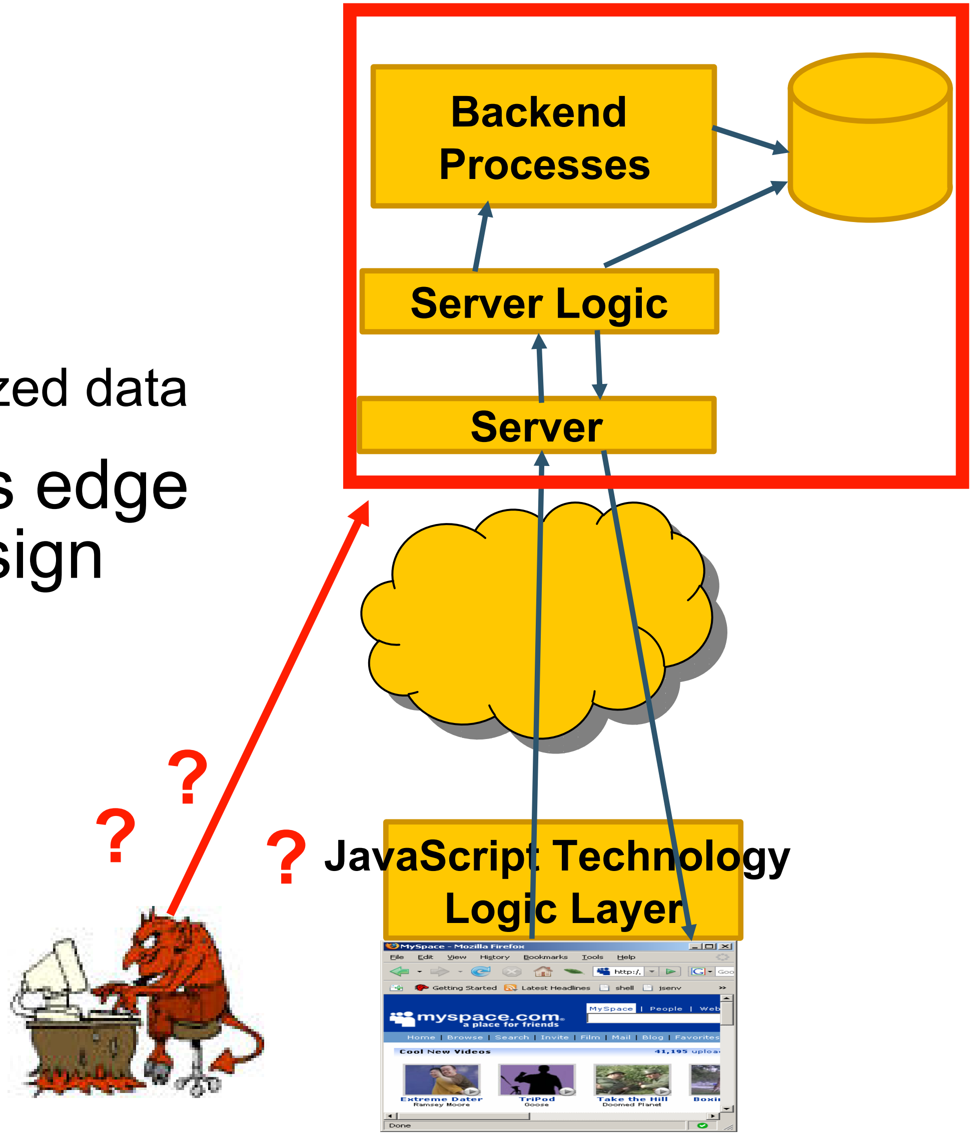
Architecture of Traditional Web Applications

- Browser—A thin client
- Most of the Application logic resides almost exclusively on server
  - Flow/business logic
  - Presentation logic
- Client acts as a dumb terminal sending actions to the server
- Server does all the processing and returns whole new page

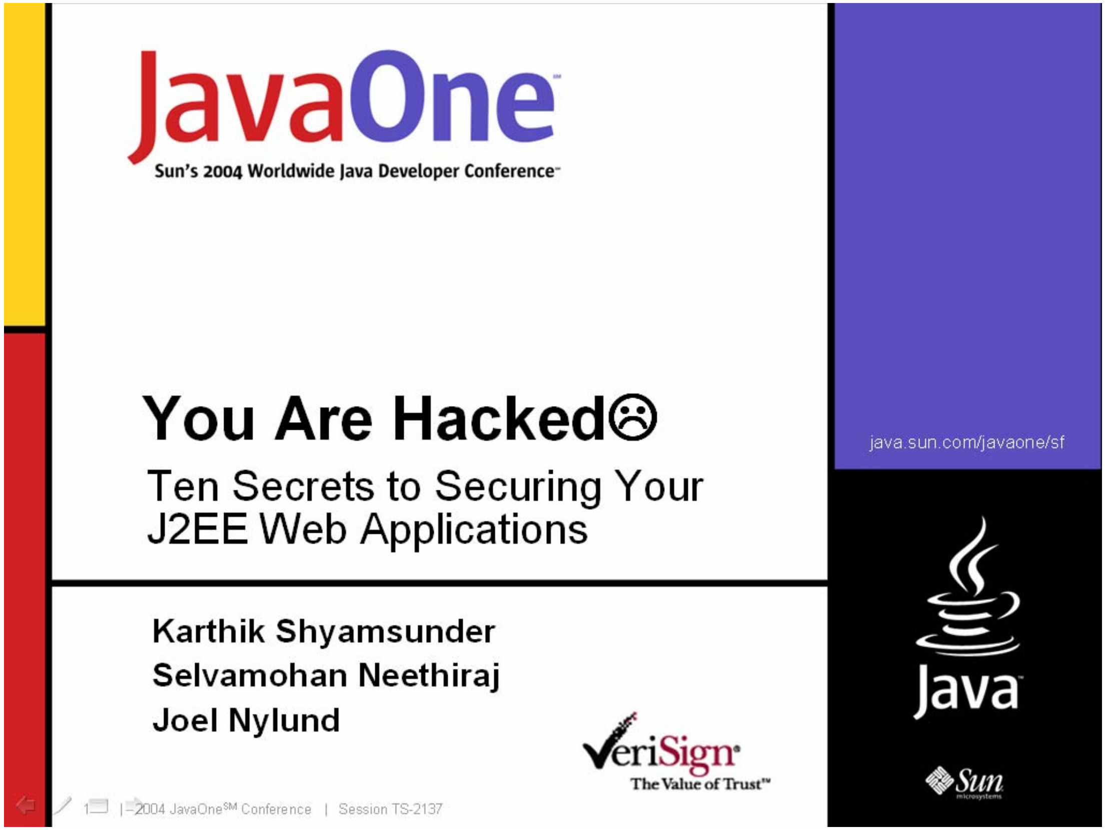


Attacks Against Traditional Web Applications

- Attacks involve:
  - Sending malicious data
  - Sending code as data
  - Trying to access unauthorized data
- Malicious command hits edge cases in application design
- What did we say?
  - Validate input parameters
  - Encode output data
  - Use proper authentication
  - Use proper authorization


Backend Processes
Server Logic
Server
?
?
?
JavaScript Technology Logic Layer
MySpace - Mozilla Firefox
myspace.com a place for friends
Cool New Videos

Web Security—
2004 JavaOne^{SM} Conference



JavaOne

Sun's 2004 Worldwide Java Developer Conference

You Are Hacked☹

Ten Secrets to Securing Your J2EE Web Applications

Karthik Shyamsunder
Selvamohan Neethiraj
Joel Nylund

VeriSign
The Value of Trust™

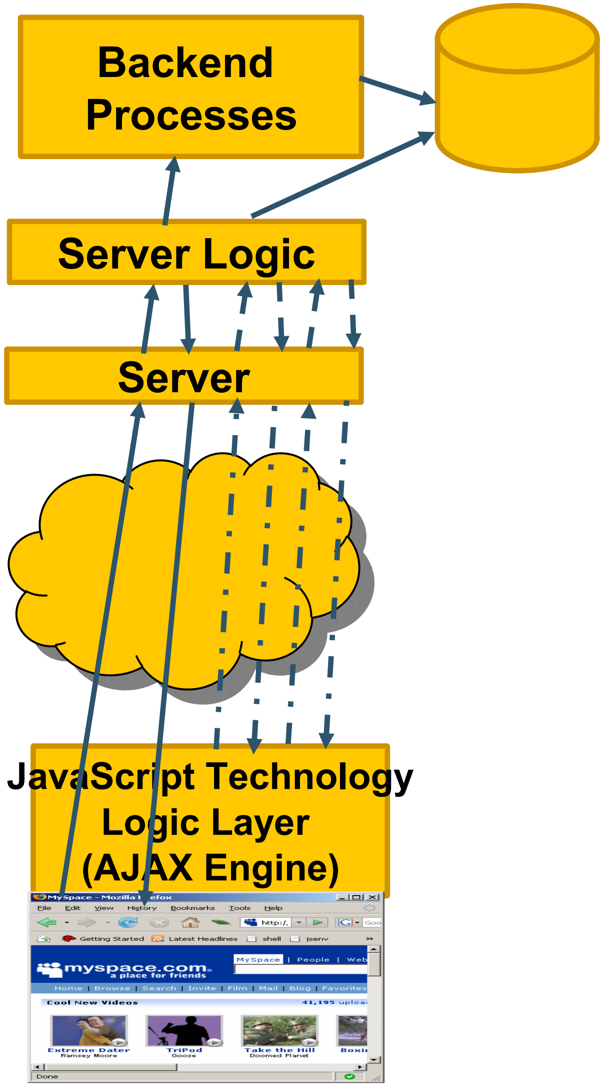
java.sun.com/javaone/sf

2004 JavaOne^{SM} Conference | Session TS-2137

Download presentation from **www.youarehacked.com**

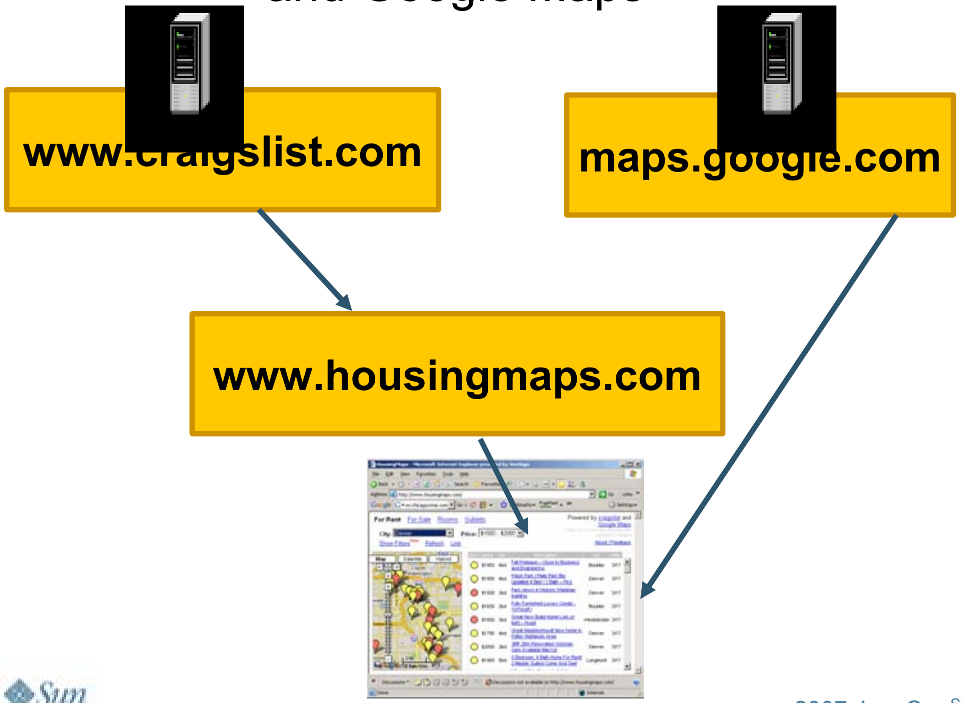
Architecture of an AJAX Application

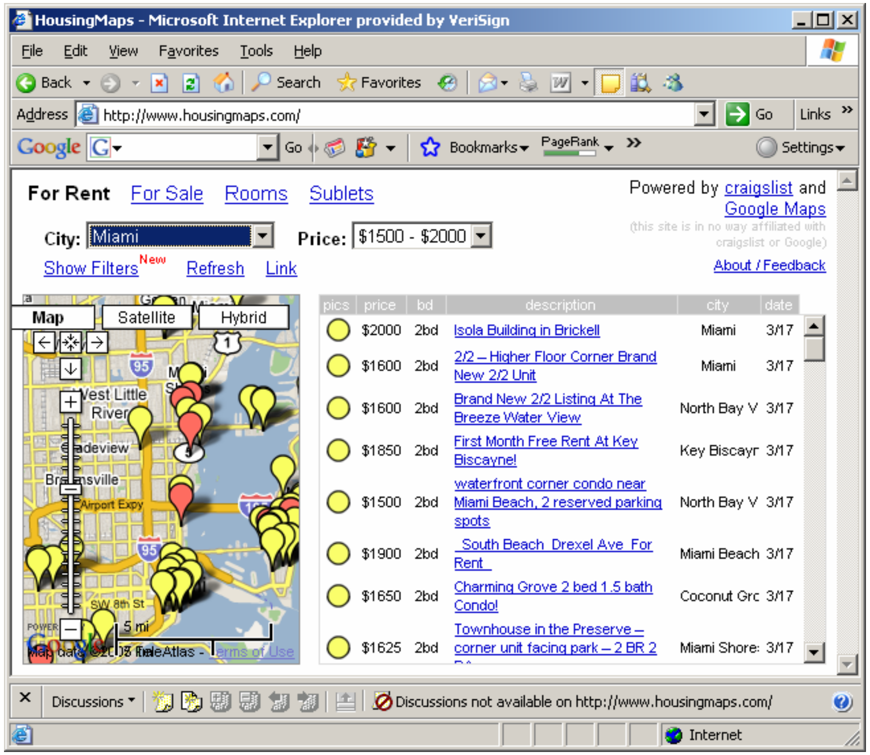
- Browser—Rich/thick-client application
- Application logic resides both on client and server
- JavaScript™ technology takes on a bigger role
- Uses XmlHttpRequest object
- Fetch any kind of resource
  - HTML, GIF (view centric)
  - XML, JSON (data centric)
  - JavaScript technology (code centric)
- Client DOM tree is being manipulated


Backend Processes
Server Logic
Server
JavaScript Technology Logic Layer (AJAX Engine)
myspace.com a place for friends

Web 2.0 Mashup Applications

- Aggregates services offered by other 3rd-party applications to form a new application
- www.housingmaps.com
  - Mashup of craigslist and Google Maps





HousingMaps - Microsoft Internet Explorer provided by VeriSign
File Edit View Favorites Tools Help
Back Search Favorites
Address
Go Links
Google Go Bookmarks PageRank Settings
Map Satellite Hybrid
West Little River
Brownsville
Airport Expy
SW 8th St
5 mi
Map data ©2007 TeleAtlas - Terms of Use
Discussions
Internet

http://www.housingmaps.com/

For Rent For Sale Rooms Sublets

Powered by craigslist and Google Maps

(this site is in no way affiliated with craigslist or Google)

City: Miami Price: $1500 - $2000

Show Filters New Refresh Link

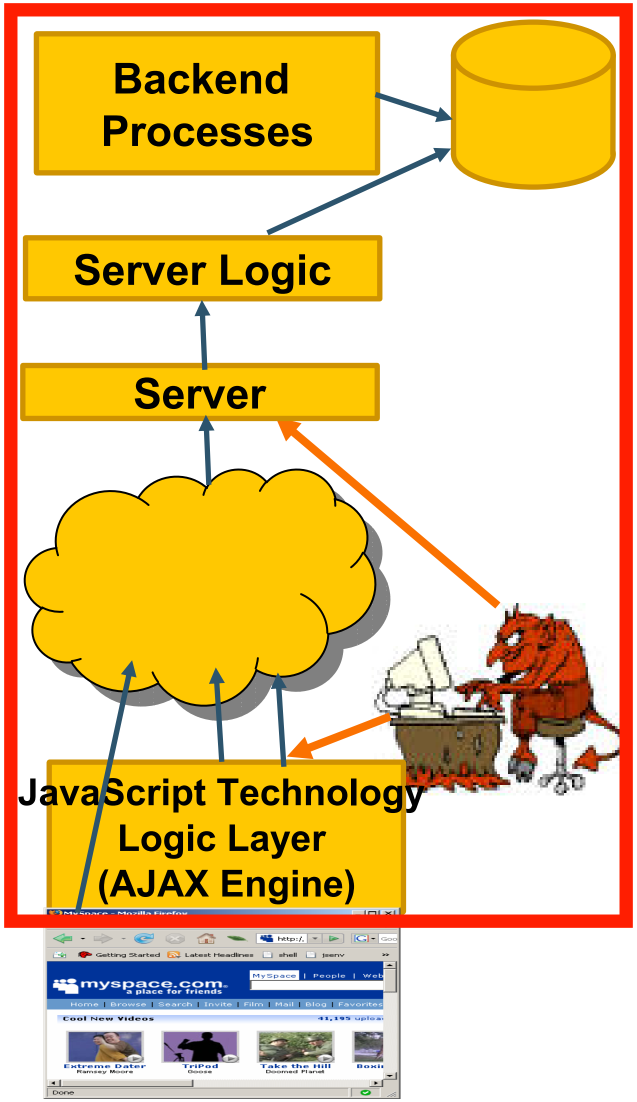
About / Feedback

pics	price	bd	description	city	date
○	$2000	2bd	Isola Building in Brickell	Miami	3/17
○	$1600	2bd	2/2 – Higher Floor Corner Brand New 2/2 Unit	Miami	3/17
○	$1600	2bd	Brand New 2/2 Listing At The Breeze Water View	North Bay V	3/17
○	$1850	2bd	First Month Free Rent At Key Biscayne!	Key Biscayr	3/17
○	$1500	2bd	waterfront corner condo near Miami Beach, 2 reserved parking spots	North Bay V	3/17
○	$1900	2bd	South Beach Drexel Ave For Rent	Miami Beach	3/17
○	$1650	2bd	Charming Grove 2 bed 1.5 bath Condo!	Coconut Grc	3/17
○	$1625	2bd	Townhouse in the Preserve – corner unit facing park – 2 BR 2	Miami Shore:	3/17

Discussions not available on http://www.housingmaps.com/

Attacks Against AJAX Applications

- Traditional web application attacks still apply
- Attacker is inside your application
  - Knowledge increases
  - Larger attack surface
  - Data serialization from unknown/untrusted sources
  - Companies migrate to AJAX without much thought to security
- In the case of mashups, attacking 3rd-party servers



Agenda

Internet Threat Model

Browser Security Model

Vulnerabilities, Attacks, and Countermeasures

Secure Software-Development Process

Summary

Q&A

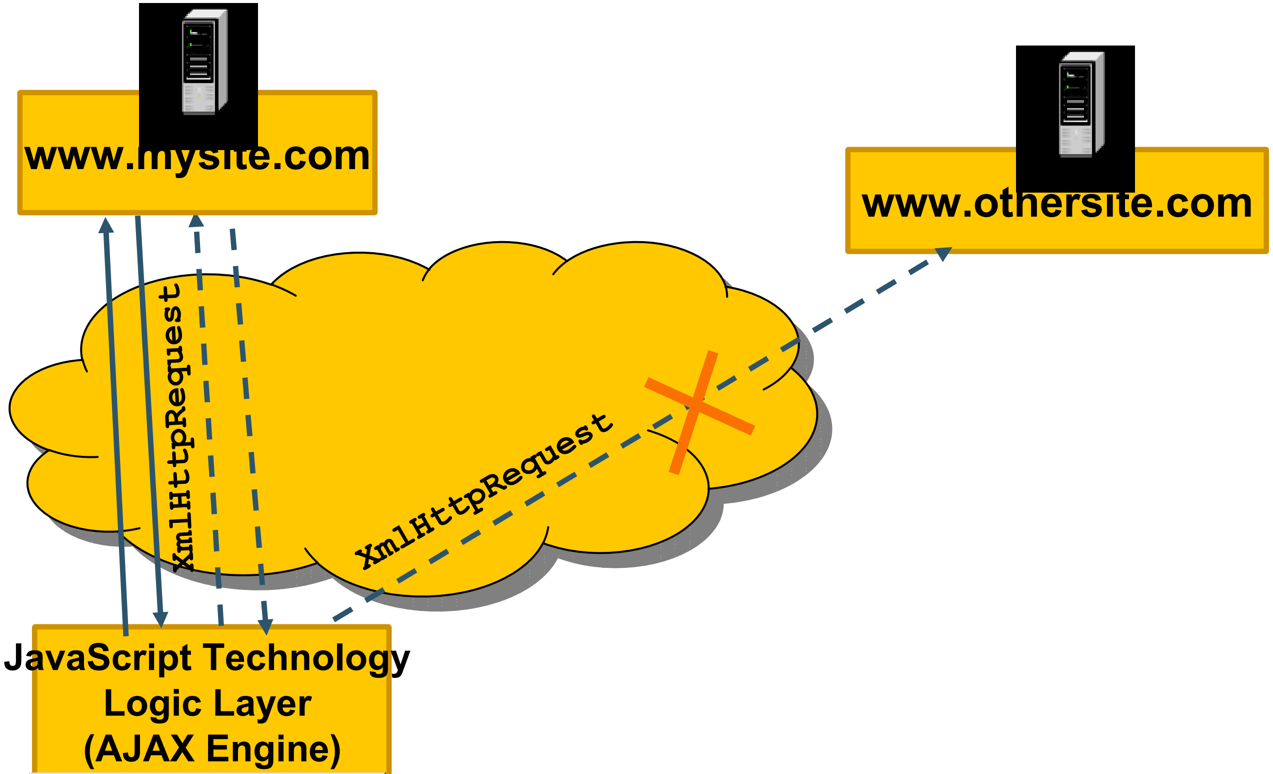
JavaScript Security in the Browser

- “Mobile code” = potential security risk
- Browsers execute JavaScript code in a sandbox
- Restrictions on JavaScript code in the sandbox
  - Cannot read/write files from/to the local system
  - Cannot execute any other programs
  - Cannot read the history of the browser
  - Cannot close a window that mobile code did not open
  - Cannot open a window that is too small

Browser’s “Same Origin” Policy

- Also called “Server of Origin” Policy
- “Origin” = (protocol + host + port) parts of the URL
- Restriction was put to limit interaction between frames, iframes, and script tags from different origins
- Restriction extended to include XMLHttpRequest
    - Prevents client-side JavaScript from making requests to any server other than the server from which it was downloaded
    - Different browser vendors implement this security somewhat differently

“Same Origin” Policy for AJAX



More “Same Origin” Policy Cases

	URLs	ALLOWED?	REASON
1	http://www.mysite.com/webapp1/action1	Yes	Although paths come from 2 different applications, the protocol, host and port is the same
	http://www.mysite.com/webapp2/action2		
2	http://www.mysite.com:8080/action1	No	Port numbers don’t match
	http://www.mysite.com/action2		
3	http://www.mysite.com/action1	No	Protocols don’t match
	https://www.mysite.com/actions2		
4	http://www.mysite.com/action1	No	Although www.mysite.com resolved to 128.220.101.100, but the browser does not work this out
	http://128.220.101.100/action2		
5	http://www.mysite.com/action1	No	Sub-domains are treated as separate domains
	http://scripts.mysite.com/action2		

Dancing Around the "Same Origin" Policy

1. Manipulate Browser Security Policy

2. Proxy Remote Services

3. Dynamic <SCRIPT> Tag

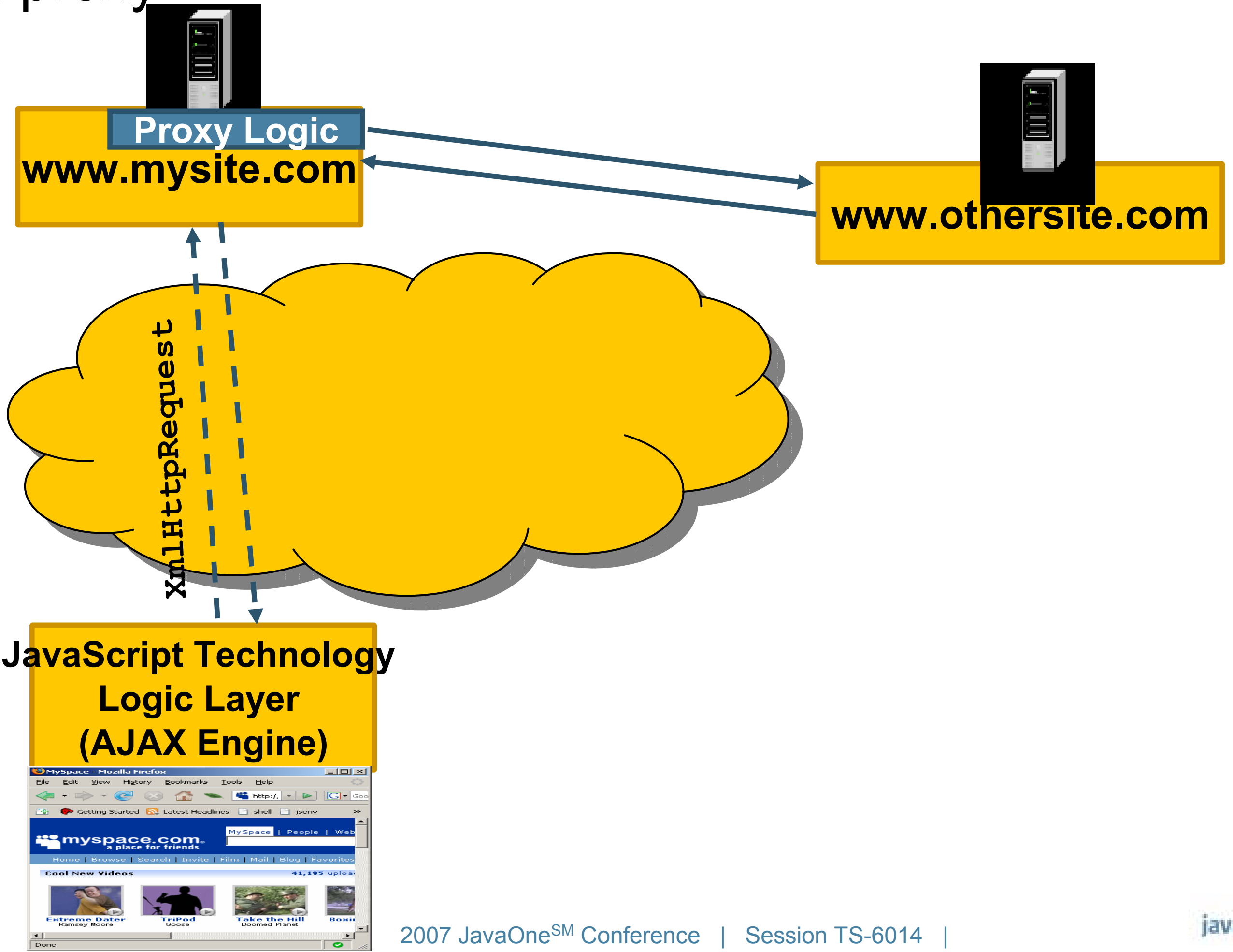
1. Manipulate Browser Security Policy

- Internet Explorer
  - IE security system is based on the concept of “zones”
  - Contacting external sites makes IE popup warning window
  - IE trusts AJAX applications running from the local file system

- Mozilla
  - Mozilla security system is based on the concept of privileges
  - Application needs to request for privilege
  - Privileges handled by netscape.security.PrivilegeManager
  - To request privilege programmatically call enablePrivilege
  - Firefox can be configured to not listen to privilege manager

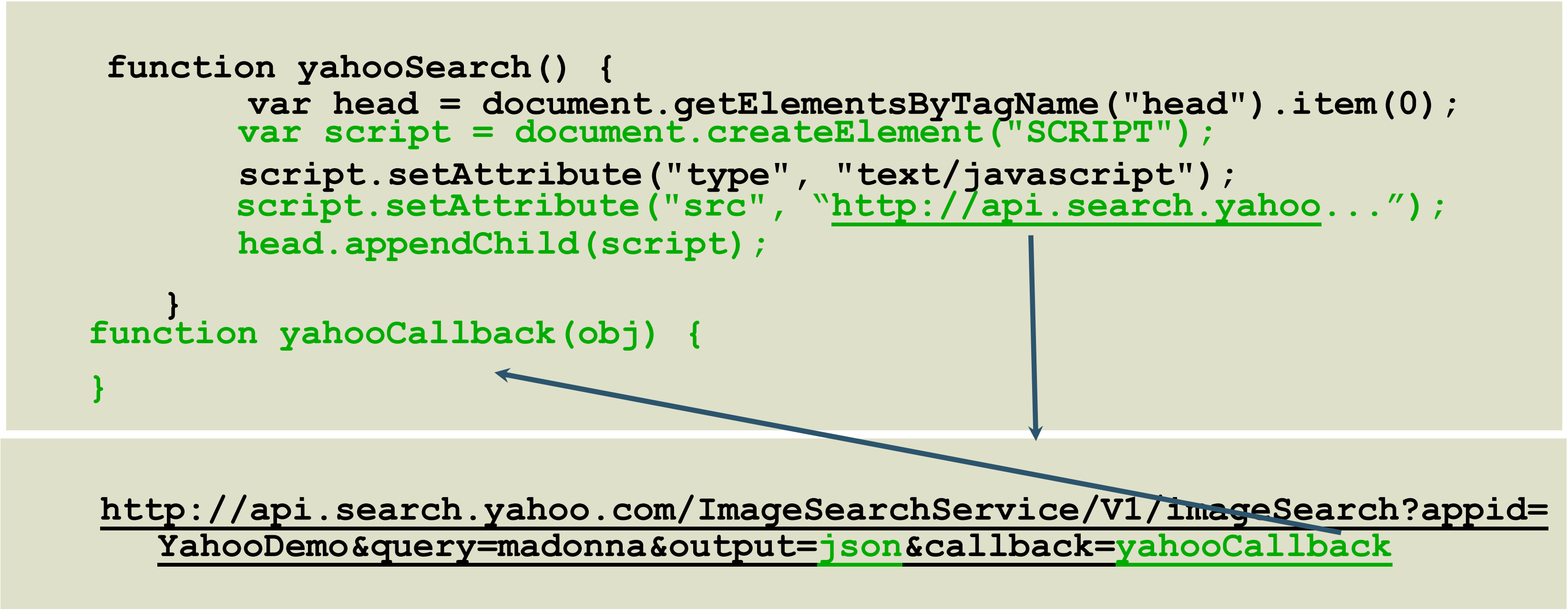
2. Proxy Remote Services

- Also called “bridge” or “server-side proxy”
- 3rd-party proxy such as Apache mod proxy
- Custom proxy



3. Dynamic <SCRIPT> tag

- Create dynamic **`<SCRIPT>`** HTML tag instead of XMLHTTPRequest
- Assign the **src** attribute the URL of the web service
- Append the **`<SCRIPT>`** to the page, which triggers the request
- Server returns JavaScript (or JSON object) executed in the browser



```
 function yahooSearch() {
        var head = document.getElementsByTagName("head").item(0);
        var script = document.createElement("SCRIPT");
        script.setAttribute("type", "text/javascript");
        script.setAttribute("src", “http://api.search.yahoo...”);
        head.appendChild(script);

    }
function yahooCallback(obj) {
}
```

```
http://api.search.yahoo.com/ImageSearchService/V1/imageSearch?appid=
   YahooDemo&query=madonna&output=json&callback=yahooCallback
```

Agenda

The Internet Threat Model

Browser Security Model

Vulnerabilities, Attacks, Countermeasures

Secure Software-Development Process

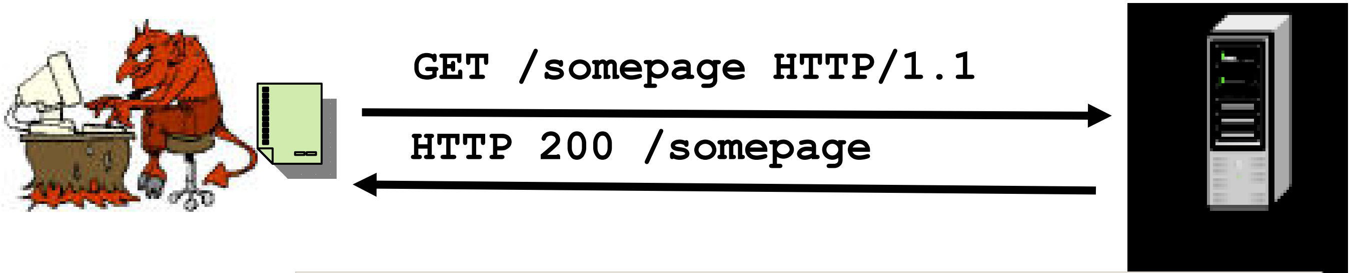
Summary

Q&A

Exposure of Internal Details

Vulnerabilities

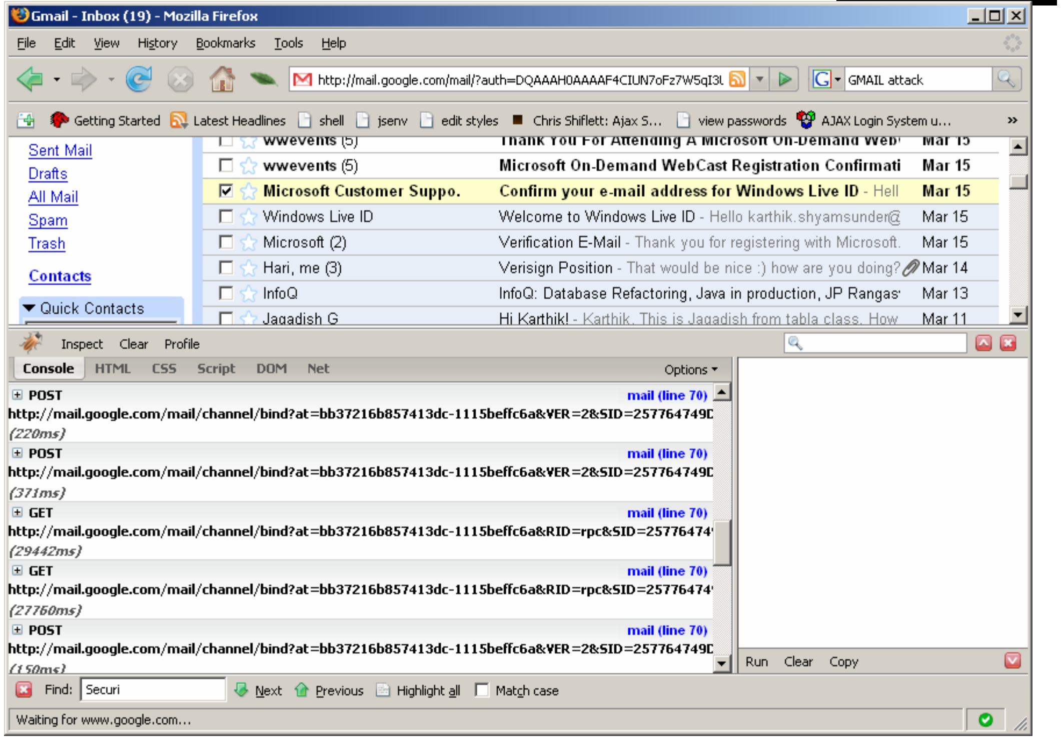
- Profiling



GET /somepage HTTP/1.1

HTTP 200 /somepage

- What's new in Web 2.0?
  - Better Tools
  - Firebug
    - View DOM tree
    - Put breakpoints
    - Alter values
  - Watir
    - Ruby-based tool
  - Selenium
    - Java technology-based Tool



□	wwevents (5)	Thank You For Attending A Microsoft On-Demand Web	Mar 15
□	wwevents (5)	Microsoft On-Demand WebCast Registration Confirmati	Mar 15
☑	Microsoft Customer Suppo.	Confirm your e-mail address for Windows Live ID - Hell	Mar 15
□	Windows Live ID	Welcome to Windows Live ID - Hello karthik.shyamsunder@	Mar 15
□	Microsoft (2)	Verification E-Mail - Thank you for registering with Microsoft.	Mar 15
□	Hari, me (3)	Verisign Position - That would be nice :) how are you doing?	Mar 14
□	InfoQ	InfoQ: Database Refactoring, Java in production, JP Rangas	Mar 13
□	Jagadish G	Hi Karthik! - Karthik. This is Jagadish from tabla class. How	Mar 11

```
+ POST                                                                  mail (line 70)
http://mail.google.com/mail/channel/bind?at=bb37216b857413dc-1115beffc6a&VER=2&SID=257764749D
(220ms)
+ POST                                                                  mail (line 70)
http://mail.google.com/mail/channel/bind?at=bb37216b857413dc-1115beffc6a&VER=2&SID=257764749D
(371ms)
+ GET                                                                   mail (line 70)
http://mail.google.com/mail/channel/bind?at=bb37216b857413dc-1115beffc6a&RID=rpc&SID=25776474
(29442ms)
+ GET                                                                   mail (line 70)
http://mail.google.com/mail/channel/bind?at=bb37216b857413dc-1115beffc6a&RID=rpc&SID=25776474
(27760ms)
+ POST                                                                  mail (line 70)
http://mail.google.com/mail/channel/bind?at=bb37216b857413dc-1115beffc6a&VER=2&SID=257764749D
(150ms)
```

Find: Securi

Waiting for www.google.com...

Exposure of Internal Details

Vulnerabilities

- What’s new in Web 2.0?
  - Much more client-side code for hacker to view and dissect
  - Potentially more client-side comments for hacker to view
  - Better social community (blogs, newsgroups, forums)
  - Hackers’ knowledge has increased
    - Application architecture/design details
    - Program business/logic flow details
    - Function names, variable names, return types
    - Helps build a footprint of the web application
  - Direct API access
    - Developers encouraged to expose more web services
    - Attacker calls your backend functions directly
    - Bypasses logic in the client side
    - Calls functions out of order

Exposure of Internal Details

Countermeasures

- Do not give out unnecessary information
- Remove comments from HTML/JavaScript technology code
  - Developer names, design details, notes, build numbers
  - Use build-time tools to remove comments
- Turn off WSDL for your web services
  - Many tools auto generate WSDLs—turn them off
  - No need to expose all services, inputs, and types to users

Exposure of Internal Details

Countermeasures

- Is AJAX the appropriate technology?
  - Use traditional web-application technology where security is a high priority
- Obfuscate your JavaScript technology code

```
function helloWorld(name) {
    alert("Hello World " + name + "!");
}
```

```
eval(function(p,a,c,k,e,d){while(c--){if(k[c]){p=p.replace(new
    RegExp('\\b'+c+'\\b','g'),k[c])}}return p}('5 3(0){2("4 1
"+0+"!")}',6,6,'name|World|alert|helloWorld|Hello|function'.split('
                    |')))
```

- Note: obfuscation is not fool-proof

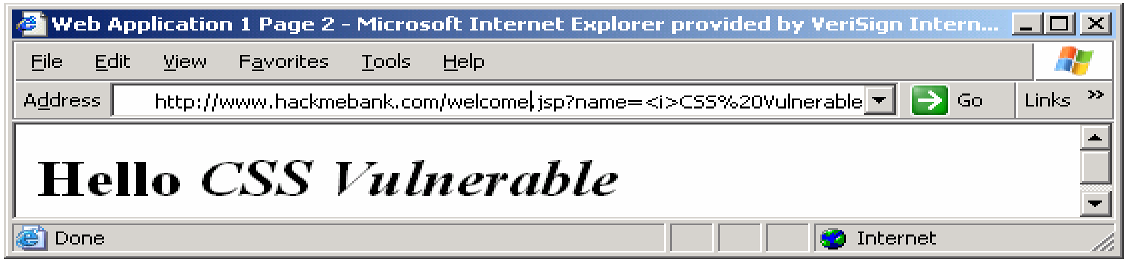
Cross-Site Scripting

Vulnerabilities

- Accomplished by code injection (HTML, JavaScript technology)

```
http://www.hackmebank.com/welcome.jsp?name=john
...
<h1>Hello <%= request.getParameter("name") %></h1>
...
```

```
http://www.hackmebank.com/welcome.jsp?name=<i>CSS%20Vulnerable</i>
```



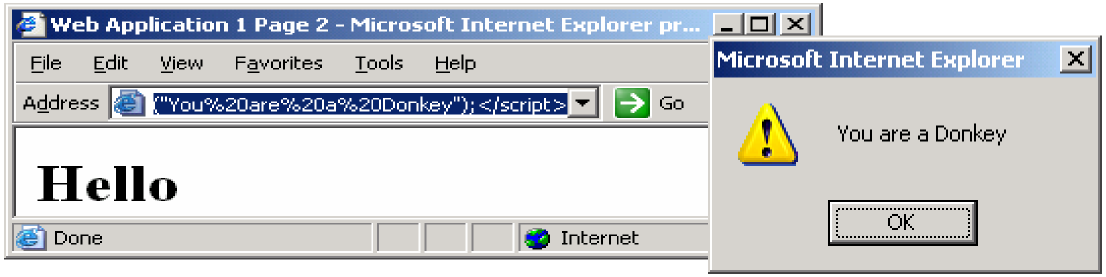
Web Application 1 Page 2 - Microsoft Internet Explorer provided by VeriSign Intern...

Address http://www.hackmebank.com/welcome.jsp?name=<i>CSS%20Vulnerable

Hello *CSS Vulnerable*

Done

```
http://www.hackmebank.com/welcome.jsp
?name=<script>alert("You%20are%20a%20Donkey");</script>
```



Web Application 1 Page 2 - Microsoft Internet Explorer pr...

Address ("You%20are%20a%20Donkey");</script>

Hello

Done

Microsoft Internet Explorer

You are a Donkey

OK

Cross-Site Scripting

Vulnerabilities

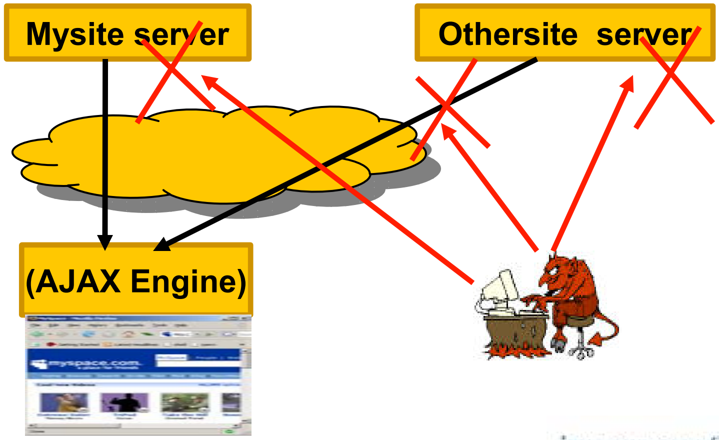
- What’s new in Web 2.0?
  - JavaScript Technology Object Poisoning
    - Manipulate the fields
    - Manipulate the functions
    - Same applies for Arrays

```
acct = {
   number : 1234,
   balance : 99.99,
   name : "John Doe",
   update : function(){ … },
   delete : function(){ … }
   };
```

```
acct.update = function() {// malicious
   code }
```

```
temp = acct.delete;
acct.delete = acct.update;
acct.update = acct.delete;
```

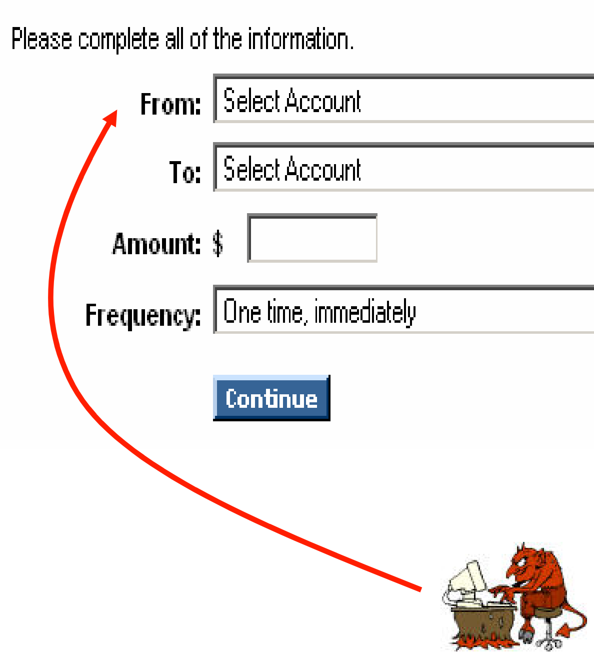
- JSON Poisoning
  - Poison data in server
  - Poison data in other server
  - Man in the middle attack can inject poison data



Cross-Site Scripting

Vulnerabilities

- Presentation/View Poisoning
  - Attacker does not attack the logic
  - Manipulates the CSS objects
  - Changes labels, re-skinning and repositioning UI components



Please complete all of the information.

From:	Select Account ▾
To:	Select Account ▾
Amount: $	
Frequency:	One time, immediately ▾

- SCRIPT Injection
  - Injects malicious <SCRIPT> tag
  - New scripts
  - Invoke back-end functions
  - Make existing functions invalid

Cross-Site Scripting

Countermeasures

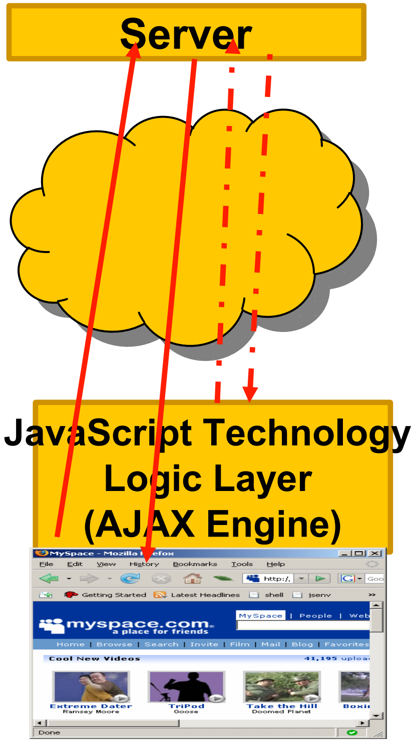
- Practice input validation!
- Practice output encoding
  - HTML encoding when sending output to browser to avoid XSS
  - Practice JavaScript technology encoding to neutralize XSS

Cross-Site Request Forgery

Vulnerabilities

- Also known as XSRF and CSRF and Cross-Site Reference Forgery
- Works by exploiting a trust that a user has in the application
  - `<img src="http://host/command">`
  - `<script src="http://host/command">`
  - `<iframe src="http://host/command">`

  -
    ```
    <script>
      var foo = new Image();
      foo.src = "http://host/command";
    </script>
    ```
- What’s new Web 2.0?
  - Use **`XMLHttpRequest`** object to perform CSRF requests
  - Exposed web services amplify this attack
  - Better control over the request that can be sent
    - Can send HTTP headers, and make GET/POST request
    - Can receive HTTP status code, headers, and response data



Server

JavaScript Technology Logic Layer (AJAX Engine)

Cross-Site Request Forgery

Countermeasures

- Common Misconceptions About Cross-Site Request Forgery
  - It is only exploitable in browser-based applications
    - Scripts embedded into Word, Flash, Movie, RSS, or Atom web feed
  - It is not exploitable in POST-based services
    - **`<FORM>`** tag can be used to submit POST requests
  - It can be prevented by implementing **`Referer`** header checking
    - **`Referer`** header can be spoofed by using **`XMLHttpRequest`**
  - It can be prevented by using the “one time token” pattern
    - Attacker can use existing XSS flaw to grab the token
- Potential Solutions
  - Implement POST-based service and **`Referer`** header checking and token approach
  - Prompt the user with PIN or strong CAPTCHA before each important action
  - Set a short time period for user sessions
  - Prevent XSS flaw

Improper Validation

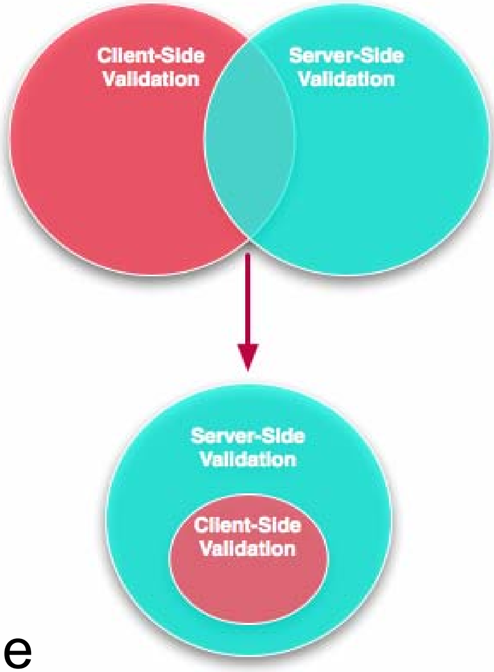
Vulnerability

- Application accepts invalid/malicious input
  - SQL Injection, XSS, Parameter Tampering
- What's new in Web 2.0?
  - Validation confusion
    - Where is the validation done (client/server/both)?
    - With Sophisticated drag and drop IDEs, validation details are hidden
  - Complexity of data has increased
    - Lack of good toolkits/regular expressions available to validate these types of input
  - What input gets validated?
    - Developers usually validate GET/POST parameters
    - Developers often forget about HTTP Headers
    - Developers forget about file input (images, audio, video)
  - Trusting data from B2B partners
    - Mashups are bringing data from non-validated sources

Improper Validation

Countermeasures

- Never trust the client!
- Validate all input data to the application
- Use strong validation techniques
  - Correctness, type, format, length, range, and context
  - Use white-listing instead of Black-listing
  - Escaping input if possible
- Always validate on the server side
  - Server-side validation = data integrity and security
- Client-side validation as a subset of server side
  - Client-side validation = usability and performance
- For mashups, never trust the external server

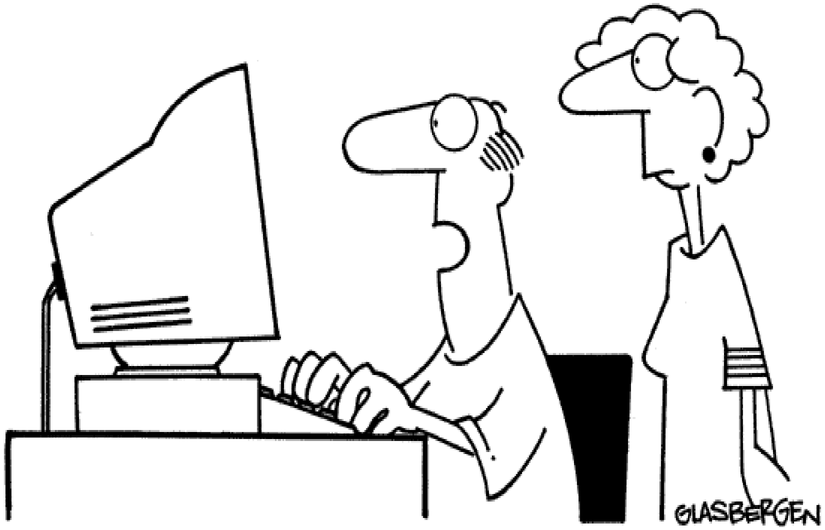


Exploit Broken Authentication

Vulnerabilities

- Authentication
  - Act of proving who you say you are
  - Methods
    - User Name and Password
    - Certificate
- Broken Authentication leads to:
  - Identity theft
  - Session hijacking
  - Loss of data
- Attack types
  - Man-in-the-middle attack
  - Replay attack

Copyright 2002 by Randy Glasbergen. www.glasbergen.com

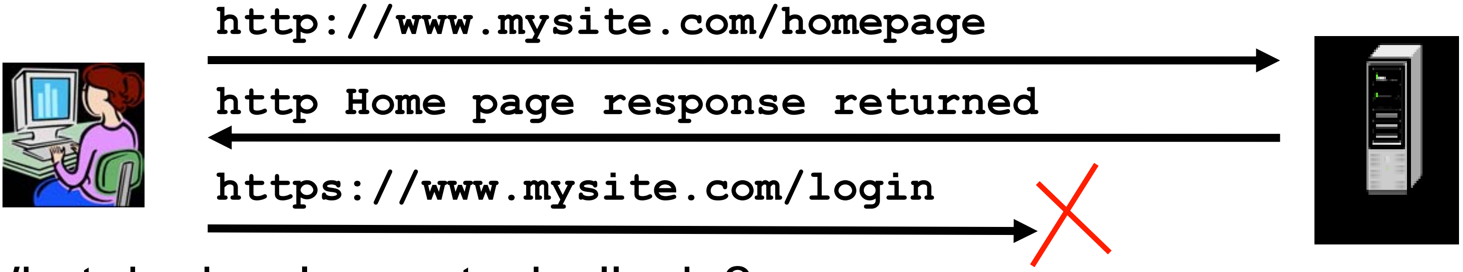


"Someone got my Social Security number off the internet and stole my identity. Thank God — *I hated being me!*"

Exploit Broken Authentication

Vulnerabilities

- What’s new in Web 2.0?
  - Most Web 2.0 applications are HTTP-based community sites
  - Scenario—HTTP AJAX application with HTTPS authentication

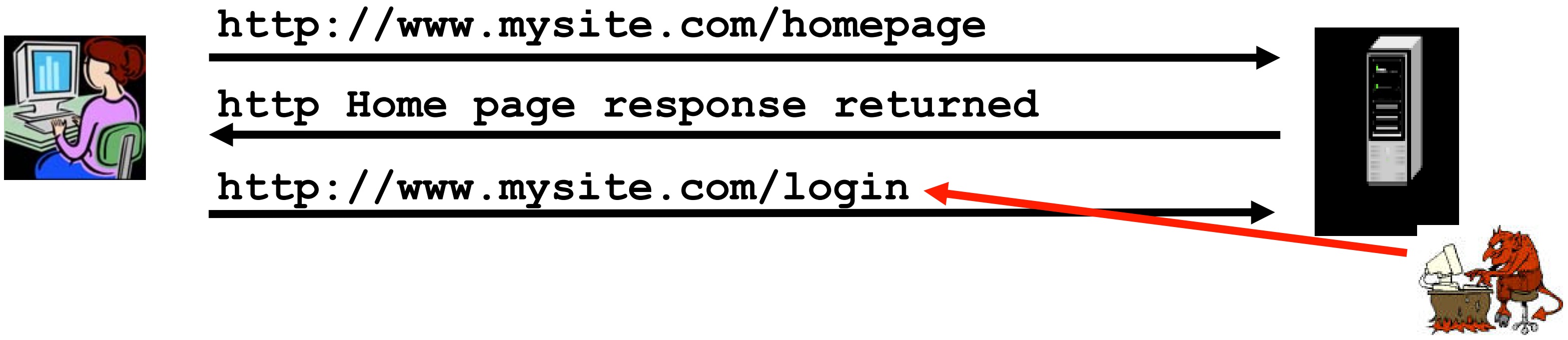


http://www.mysite.com/homepage

http Home page response returned

https://www.mysite.com/login

- What do developers typically do?
  - Use HTTP for entire AJAX application



http://www.mysite.com/homepage

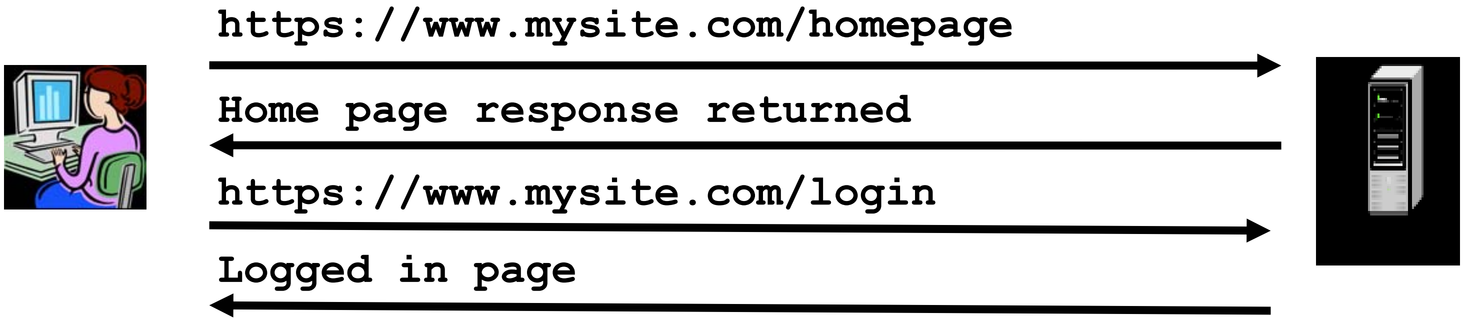
http Home page response returned

http://www.mysite.com/login

Exploit Broken Authentication

Countermeasures

- Option 1: Use HTTPS for the entire Web 2.0 application
  - Does address the “Same Origin” Policy
  - Hacker cannot sniff any packets
  - Frequent SSL handshakes is expensive
    - Full or partial SSL handshake is dependent on the browser



https://www.mysite.com/homepage

Home page response returned

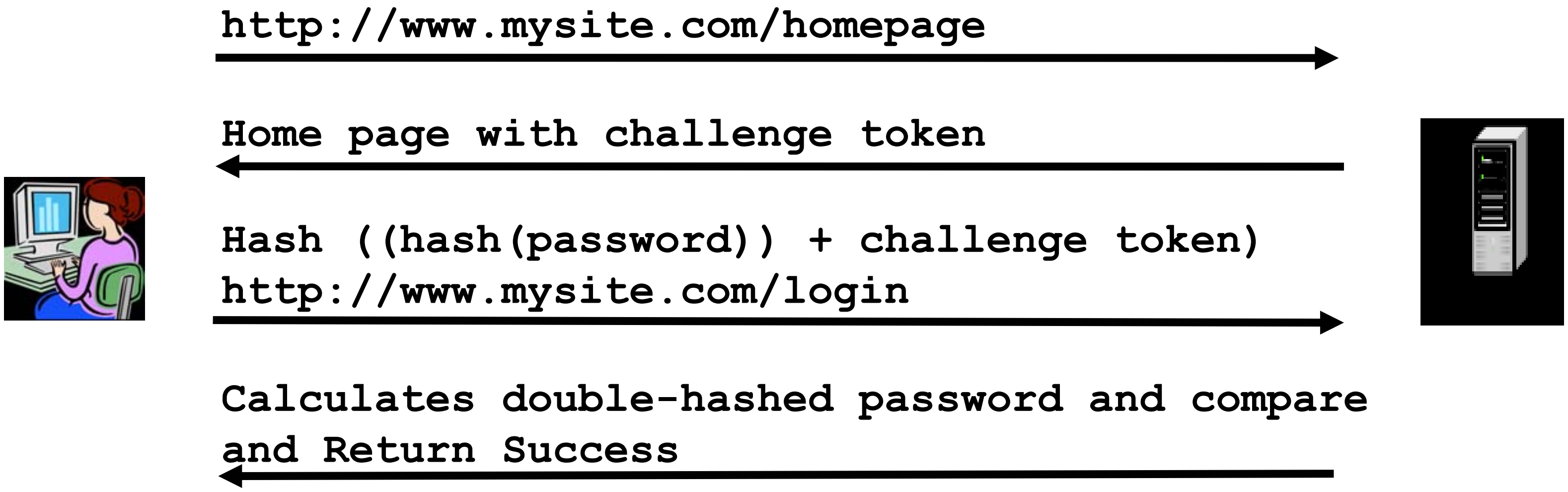
https://www.mysite.com/login

Logged in page

Authentication Issues

Countermeasures

- Option 2: Use HTTP with the “Direct Login” AJAX pattern
  - Addresses the “Same Origin” Policy
  - Does not incur the HTTPS cost
  - Requires encrypting password so it cannot be decoded or replayed
    - Use server one-time random challenge token
    - Use of double-hashed password



http://www.mysite.com/homepage

Home page with challenge token

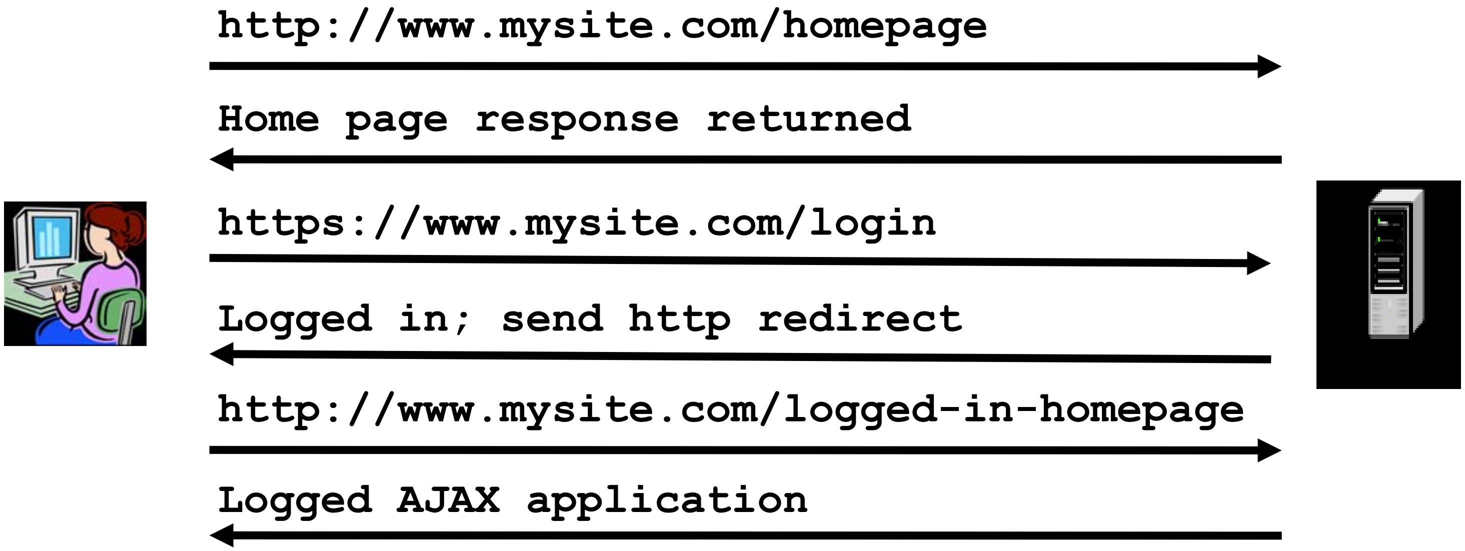
Hash ((hash(password)) + challenge token)
http://www.mysite.com/login

Calculates double-hashed password and compare
and Return Success

Authentication Issues

Countermeasures

- Option 3: Use traditional HTTPS login page with redirect to HTTP AJAX application
  - Can use plain HTTP for AJAX application
  - Can use a secure transport when passing user credentials
  - Simpler than the “Direct Login” AJAX pattern



```
http://www.mysite.com/homepage
Home page response returned
https://www.mysite.com/login
Logged in; send http redirect
http://www.mysite.com/logged-in-homepage
Logged AJAX application
```

Exploit Broken Access Control

Vulnerabilities

- Access Control is also called Authorization
- Implementing correct access control is not trivial
- Broken Access Control leads to:
  - Unauthorized access to sensitive data
  - Unauthorized users executing illegal transactions
- What's new in Web 2.0?
  - Authorization logic can be exposed in the client
    - Exposure of Role names
    - Exposure of unauthorized functions
    - Hacker can by-pass client-side authorization logic
  - Increased number of services that need to be protected
    - Hackers can obtain the services from the client-side code
    - Hackers can guess the names of services from the client-side code
    - Hackers can obtain the services from published WSDLs
    - Hackers can call services out-of-order

Exploit Broken Access Control

Countermeasures

- Minimize exposure of authorization logic in the AJAX client!
- Always check authorization on the server
- Java Platform security model still applies (declarative and programmatic)

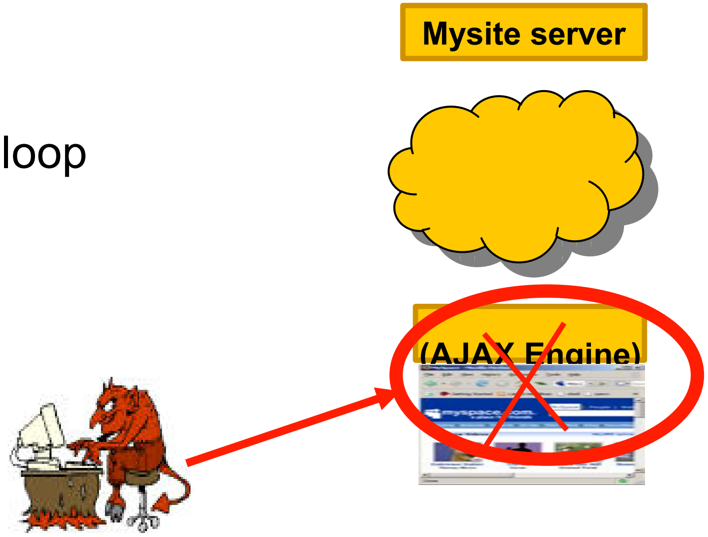
```
<security-constraint>
  <web-resource-collection>
       …
     <url-pattern>services/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
     <role-name>manager</role-name>
  </auth-constraint>
</security-constraint>
```

```
    if(!request.isUserInRole("manager")) {
        throw new UnAuthorizedException();
    }
```

Denial of Service

Vulnerabilities

- Attack by which a hacker prevents legitimate users of a service from using that service
- Misconception is DOS and DDOS are primarily network-level attacks and not application attacks
- What’s new in Web 2.0?
  - Attacks against the client:
    - Can tie up client host CPU with infinite loop
    - Can use window.setInterval() command to avoid being shutdown
    - Can take advantage of browser bugs
    - Can inject bad data to make browser crash



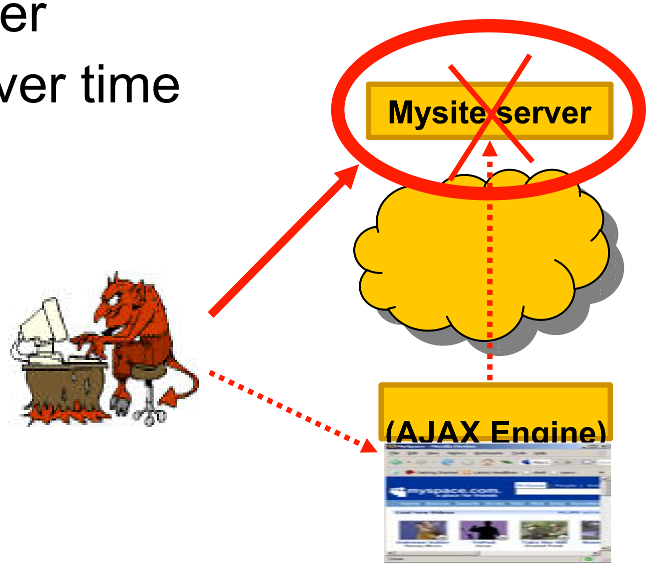
Mysite server

(AJAX Engine)

Denial of Service

Vulnerabilities

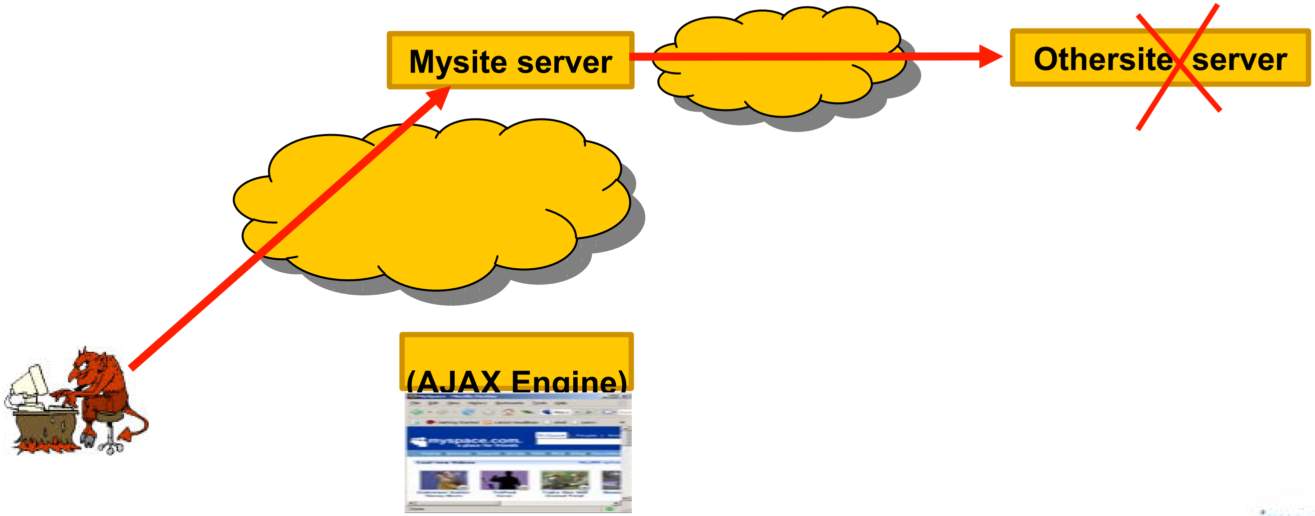
- What’s new in Web 2.0? (Cont.)
  - Attacks against the server
    - Directly call exposed web services uncontrollably
    - Insert malicious scripts to invoke services uncontrollably
    - Call exposed web services out of order
    - Insert malicious scripts that spread over time
    - For example, Samy worm



Denial of Service

Vulnerabilities

- What’s new in Web 2.0? (Cont.)
  - Attacks against the server’s server
    - Also called “smashing the mashup”
    - Attacks the proxy bridge to attack the “other site”
    - Takes advantage of trust between B2B
    - Takes advantage of the resource capabilities in B2B applications
    - Difficult for the “other site” to distinguish normal usage vs. an attack



Denial of Service

Countermeasures

- Since most of the DOS is caused by code injection, cross-site scripting bugs need to be eradicated
  - Validate input data effectively
  - Practice output encoding
- Rate-limit requests in B2C and B2B services
  - Limit Number of requests in a given period (by minute, hour, day)
  - Implement Dynamic bandwidth limitation
    - Decrease bandwidth with increased volume
- Prevent automation
  - Use CAPTCHA schemes
  - Use “one time token” design pattern
- Have good monitoring, real-time analysis, and alert systems in place
- Tune system for “at most” performance (CPU, Java Virtual Machine (JVM™), network, ...)
- Over-provision your system

The terms “Java Virtual Machine” and “JVM” mean a Virtual Machine for the Java™ platform.

Code Complexity Issues

Vulnerabilities

- Already developers have to master a lot!
  - If the technology is not well understood the risk of introducing vulnerabilities increases
- What's new in Web 2.0?
  - Now there are even more technologies to master!
  - JavaScript technology is playing a bigger role
    - Dynamic language (RTTI, dynamic types, closures)
    - Dynamic code generation and execution using eval()
    - Most errors happen at runtime
    - Not only master new concepts, but how they interact
    - Thread-safety issues

Code Complexity Issues

Countermeasures

- Education, Education, Education!
  - Learn JavaScript technology!
  - Pass this security presentation around in your group
- Practice unit testing for JavaScript technology code too
- Understand issues related to AJAX concurrency
  - Both client and server side
  - Understand that browsers launch requests simultaneously
  - Potentially synchronize on session object in servlets
- Build time in schedule for developers to ramp up

Agenda

Internet Threat Model

Browser Security Model

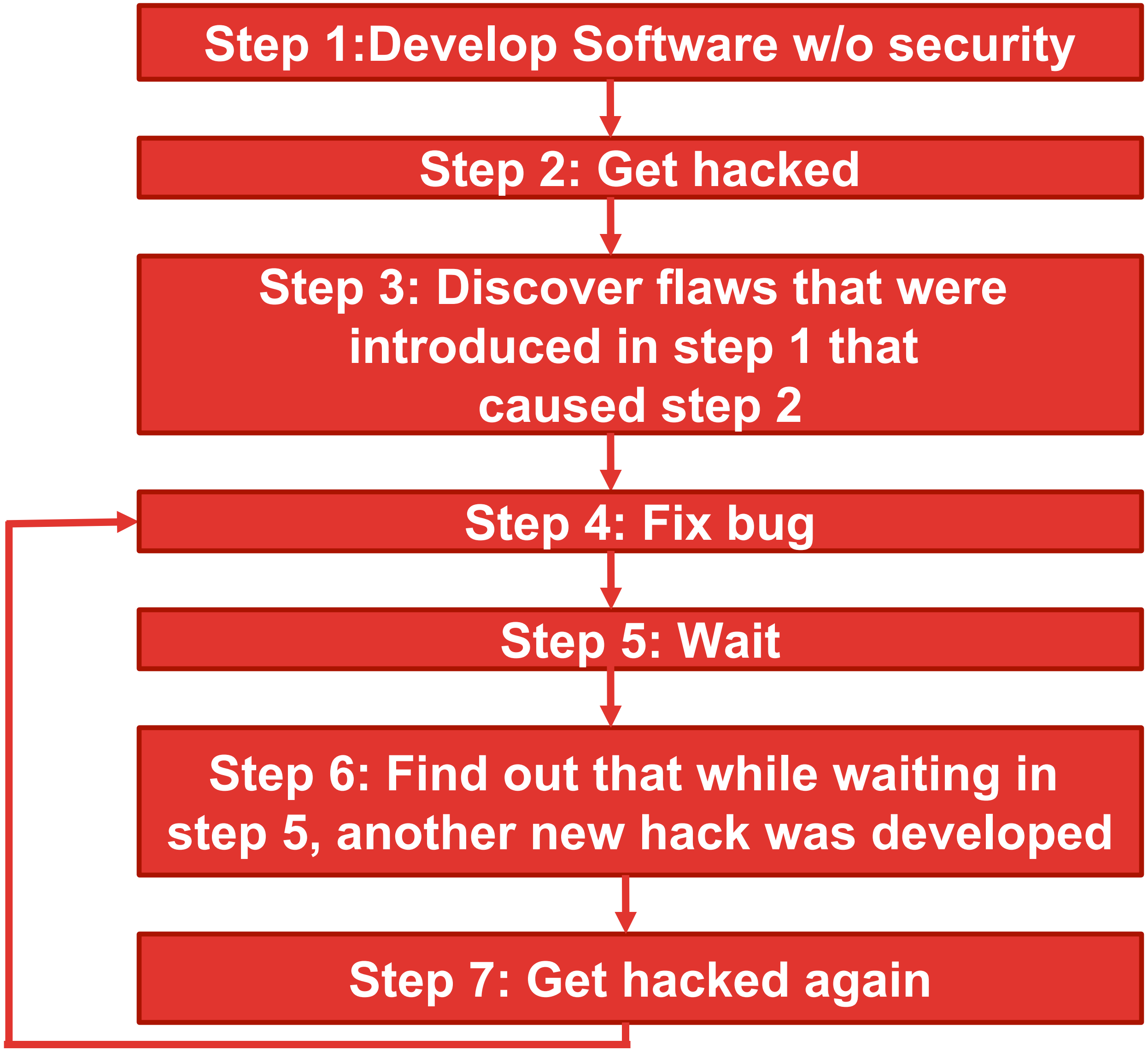
Vulnerabilities, Attacks, and Countermeasures

Secure Software-Development Process

Summary

Q&A

Seven Steps of DOOM



Step 1:Develop Software w/o security

Step 2: Get hacked

Step 3: Discover flaws that were introduced in step 1 that caused step 2

Step 4: Fix bug

Step 5: Wait

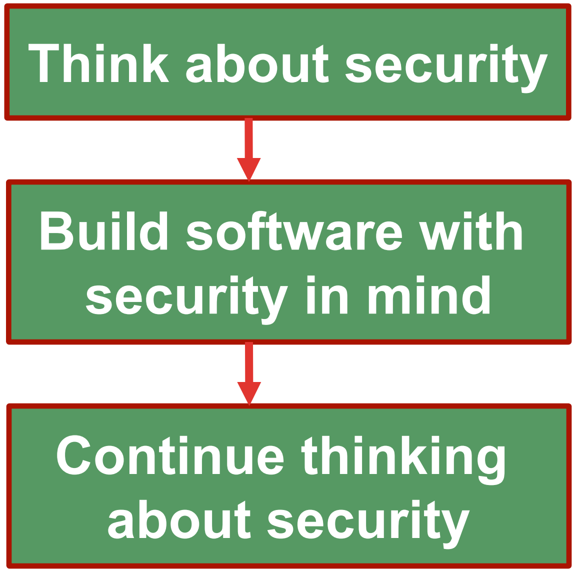
Step 6: Find out that while waiting in step 5, another new hack was developed

Step 7: Get hacked again

Steps to Success

Security in Software-Development Life Cycle

- Secure design
- Secure development
- Secure testing
- Secure deployment and operations
- Audit process



Think about security

Build software with security in mind

Continue thinking about security

Application Security Review

Security in Software-Development Life Cycle

1. Identify assets
2. Create a security architecture
3. Identify and document vulnerabilities
4. Assess your risk
5. Plan for risk mitigation

Summary

- AJAX is a powerful suite of technologies
  - AJAX can improve user experience
- But, be aware of security risks
  - Creates new ways to attack via old vulnerabilities
- Always keep security in mind when building applications

Research Contributors

- David Smith
  - Engineer—VeriSign, Inc.
  - dsmith@verisign.com
- Rajesh Badam
  - Engineer—VeriSign, Inc.
  - rbadam@verisign.com
- Satish Dandu
  - Engineer—VeriSign, Inc.
  - sdandu@verisign.com

For More Information

- http://www.youarehacked.com
- http://www.owasp.org
- http://www.cgisecurity.com/
- http://www.webhackingexposed.com/

- TS-6536
  - Enabling Identity 2.0 in Java Technology URLs

Q&A

Karthik Shyamsunder
Principal Engineer
VeriSign, Inc.

James Gould
Principal Engineer
VeriSign, Inc.

You Are Hacked ☹: AJAX Security Essentials for Enterprise Java™ Technology Developers

Karthik Shyamsunder

Principal Engineer
VeriSign, Inc.

James Gould

Principal Engineer
VeriSign, Inc.

TS-6014