Defending Computer Networks
Lecture 15: Web Evasions and Client Side Attacks

Stuart Staniford

Adjunct Professor of Computer Science

Logistics

- Apologies

Russian Ships Near Data Cables Are Too Close for U.S. Comfort

By **DAVID E. SANGER** and **ERIC SCHMITT** OCT. 25, 2015

WASHINGTON — Russian submarines and spy ships are aggressively operating near the vital undersea cables that carry almost all global Internet communications, raising concerns among some American military and intelligence officials that the Russians might be planning to attack those lines in times of tension or conflict.

The issue goes beyond old worries during the Cold War that the Russians would tap into the cables — a task American intelligence agencies also mastered decades ago. The alarm today is deeper: The ultimate Russian hack on the United States could involve severing the fiber-optic cables at some of their hardest-to-access locations to halt the instant communications on which the West's governments, economies and citizens have grown dependent.

Wizarding duels, muggle contest added as buzz surges about Ithaca event

MORE HEADLINES, NEWS / OCTOBER 26, 2015 / BY JOLENE ALMENDAREZ

ITHACA, N.Y. -- About four days ago, shopkeepers at Press Bay Alley thought they would put together a small Harry Potter inspired event, so families could show up for free candy and cider on Halloween.

But as of Friday afternoon, organizers say they've had to change their plans to accommodate what could be thousands of people turning up for the event.

Assigned Reading

- http://www.thegreycorner.com/2010/01/heap-spray-exploit-tutorial-internet.html

Main Goals for Today

- Micro-tour of Javascript
- Web-client attacks

HTTP Request

GET /dumprequest HTTP/1.1\r\n
Host: djce.org.uk\r\n
Connection: keep-alive\r\n
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_5) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/30.0.1599.101 Safari/537.36\r\n
DNT: 1\r\n
Referer: https://www.google.com/url?
sa=t&rct=j&q=&esrc=s&source=web&cd=3&ved=0CD4QFjAC&url=http%3A%2F
%2Fdjce.org.uk
%2Fdumprequest&ei=835lUpjEM5Xb4APEgIGoDA&usg=AFQjCNEeAn5wSZMp_y_oTmOK
onq482sS9A&sig2=pSajtDK-YYIvE4HFDqmRfA&bvm=bv.54934254,d.dmg\r\n
Accept-Language: en-US,en;q=0.8\r\n
\r\n

Try it at http://www.procato.com/my+headers/

HTTP Response

HTTP/1.1 404 Not Found\r\n
Content-Type: text/html; charset=UTF-8\r\n
X-Content-Type-Options: nosniff\r\n
Date: Mon, 21 Oct 2013 19:37:20 GMT\r\n
Server: sffe\r\n
Content-Length: 946\r\n
X-XSS-Protection: 1; mode=block\r\n
Alternate-Protocol: 80:quic\r\n
\r\n
<!DOCTYPE html>
…

Snort Example 5

alert tcp $EXTERNAL_NET any -> $HOME_NET $HTTP_PORTS (msg:"SERVER-WEBAPP D-Link DIR-300/ DIR-600 unauthenticated remote command execution attempt"; flow:to_server,established; content:"POST"; depth:4; nocase; http_method; content:"/command.php"; fast_pattern:only; http_uri; content:"cmd="; nocase; http_client_body; metadata:policy balanced-ips drop, policy security-ips drop, service http; reference:bugtraq, 57734; reference:url,exploit-db.com/exploits/24453/; reference:url,osvdb.org/show/osvdb/89861; reference:url,www.s3cur1ty.de/m1adv2013-003; classtype:attempted-admin; sid:26953; rev:1;)

89861 : D-Link Multiple Router command.php cmd Parameter Remote Command Execution

Views This Week	Views All Time	Added to OSVDB	Last Modified	Modified (since 2008)	Percent Complete
18	574	9 months ago	about 1 month ago	14 times	100%

Timeline

Disclosure Date
2013-02-04

Time to Exploit	Time to Vendor Response
52 days	7 days

Description

Multiple D-Link routers contain a flaw that is triggered when input passed via the 'cmd' parameter is not properly sanitized before being used in the command.php script. This may allow a remote attacker to execute arbitrary commands.

Classification

Location: Remote / Network Access
Attack Type: Input Manipulation
Impact: Loss of Integrity
Solution: Solution Unknown
Exploit: Exploit Public
Disclosure: Vendor Disputed, Third-party Verified
OSVDB: Web Related

Solution

OSVDB is not currently aware of a solution for this vulnerability.

Products

D-Link Corporation/D-Link Systems, Inc.	DIR-300	2.12b02
		2.13b01
		2.14b01
	DIR-600	2.12
		2.13

HTTP Level Evasions

- HTTP is a very complex protocol
  - Many important sub-protocols/formats
    - URIs
    - Character sets
    - Media types of entities
- As a result
  - Hard to inspect
  - Very evasion prone
  - Extensive work required in IDS to deal with issues
- We will start to work on URI issues…

Obscure HTTP Methods

- “HEAD” instead of “GET”.
- RFC 2616:

9.4 HEAD

The HEAD method is identical to GET except that the server MUST NOT return a message-body in the response. The metainformation contained in the HTTP headers in response to a HEAD request SHOULD be identical to the information sent in response to a GET request. This method can be used for obtaining metainformation about the entity implied by the request without transferring the entity-body itself. This method is often used for testing hypertext links for validity, accessibility, and recent modification.

Pipelining of Requests

- If IDS doesn't properly reassemble TCP and parse protocol:

GET foo.html HTTP/1.1\r\n\r\nGET bar.html HTTP/1.1\r\n\r\n

  - Could miss the "bar.html"
  - Have seen commercial products with this issue recently…

Directory Type Evasions

- Suppose IDS looking for "/servlet/command.php" in URL
- So attackers might try:
  - /servlet//command.php
  - /servlet///command.php
  - /servlet/./command.php
  - /servlet/./././command.php
  - /servlet/subdir/../command.php
- On Windows based web servers:
  - /servlet\command.php

URL Encoding

- RFC 2396 specifies URL format:

```
2.4.1. Escaped Encoding

   An escaped octet is encoded as a character triplet, consisting of the
   percent character "%" followed by the two hexadecimal digits
   representing the octet code. For example, "%20" is the escaped
   encoding for the US-ASCII space character.

      escaped     = "%" hex hex
      hex         = digit | "A" | "B" | "C" | "D" | "E" | "F" |
                            "a" | "b" | "c" | "d" | "e" | "f"
```

- And RFC 2616 says:

The Request-URI is transmitted in the format specified in section 3.2.1. If the Request-URI is encoded using the "% HEX HEX" encoding [42], the origin server MUST decode the Request-URI in order to properly interpret the request. Servers SHOULD respond to invalid Request-URIs with an appropriate status code.

- So IDS must do the same…

Double Percent Encoding

- %25 is '%' in ASCII
- %41 is 'A'
- So if you write %2541 and decode once
  - you get %41
- Decode again
  - you get 'A'
- Unbelievably, IIS did this…
  - IDS must follow…

Double Nibble Hex Encoding

- %%34%31
- On first decoding goes to %41
- On second decoding goes to A
- Again, Microsoft IIS supported this encoding
- Also variations like %341 and %4%31
  - Also get correctly transformed to A

– Not a current issue by default

Loose Implementations

- RFC says:
  - Method <space> URI <space> HTTP/ Version CRLF CRLF
- But some Apache versions allow
  - Method <tab> URI <tab> HTTP/ Version CRLF CRLF
- IDS must follow implementations exactly, or attacker can fool

Case Insensitivity of Windows

- /SerVLeT/ComMaNd.Php
- May well work fine if underlying OS is case insensitive
- IDS must match behavior of target

Web Drive-By Download Attacks

- Became big around 2005-2007.
- Have been the main action since then.
- Attacker response to firewall/IPS technology.
  - Largely circumvents those defenses.
  - Took a while to develop useful defenses.
    - Still a very active arms-race.

Two Main Schemas

- Scan/Compromise legit websites
  - Eg SQL Injection attacks
  - Insert <iframe>s into site
  - Iframes include content from an exploit server
- Malverts
  - Malicious ads bought through chains of middle men
  - Redirects to malicious content (often swf (Flash))

Either Way

- Exploit server runs an exploit kit
- Exploit kit tests nature of browser/plugins
  - Java
  - PDF
  - Flash
- Picks one or more exploit objects
- Takes control of browser/plugins
- Installs malware/trojans
- Command and control for instructions
- Exploit kits often have extensive management infrastructure
- Profit!

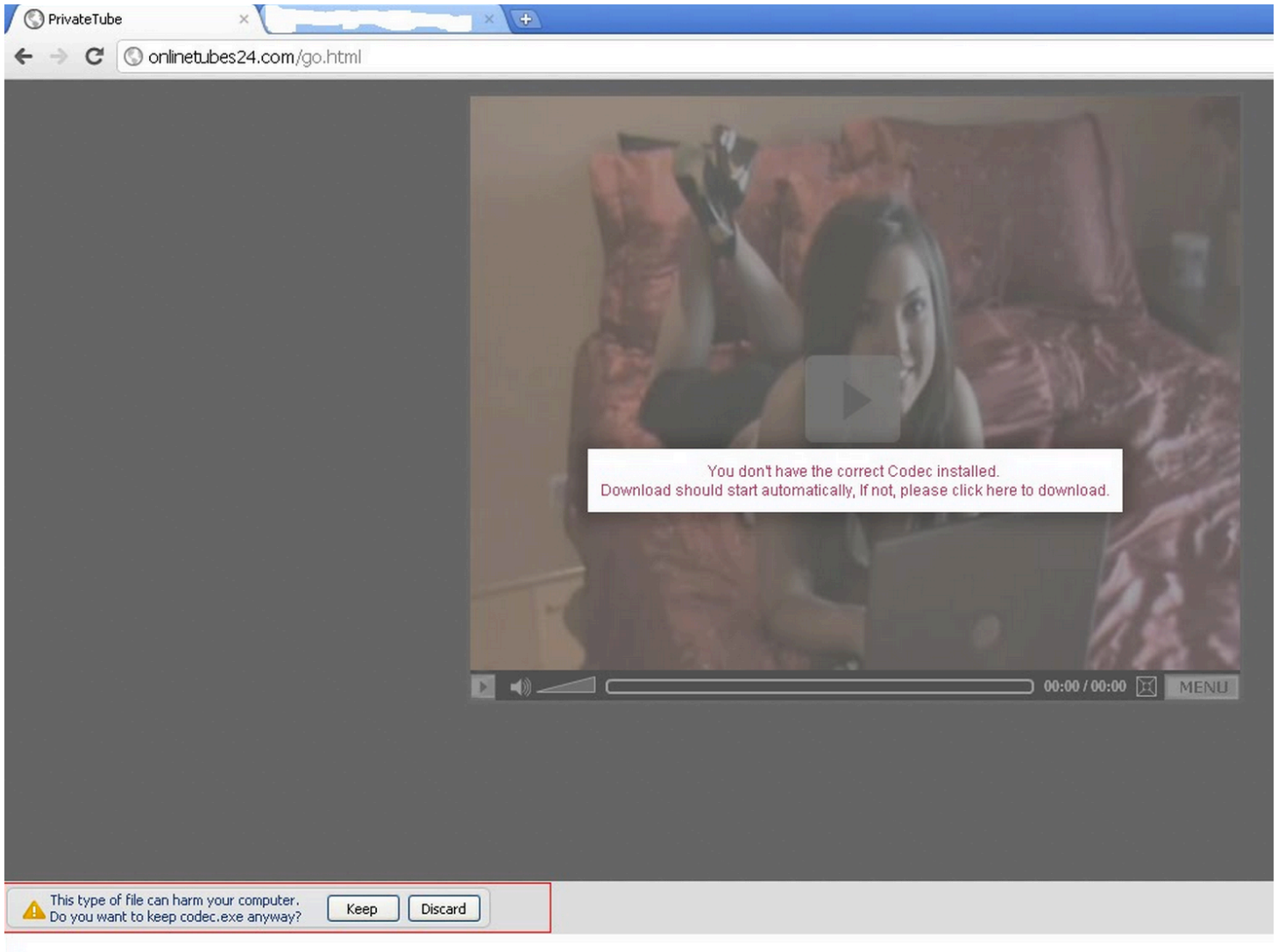
Also, social engineering

- Trick humans into installing/running malware
- Also works pretty well
- Let's look at a few examples

Fake AV

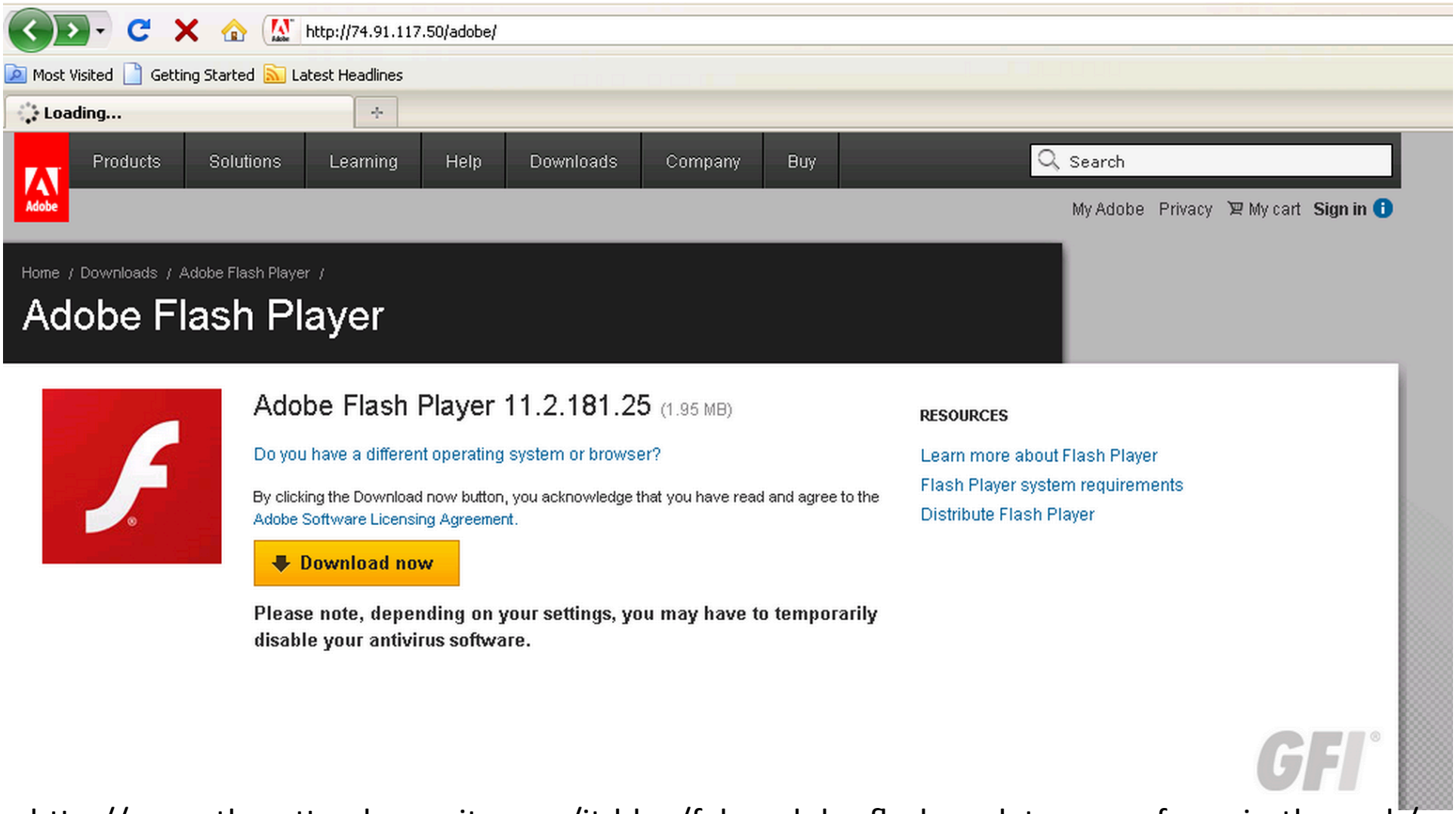

Microsoft Security Warning
Antivirus 360 Web Scanner detected dangerous spyware on your system!
Detected malicious programs can damage your computer and compromise your privacy. It is strongly recommended to remove them immediately.
Name
Type
Risk level
Spyware.IEMonster.b
Spyware
CRITICAL
Zlob.PornAdvertiser.Xplisit
Spyware
High
Trojan.InfoStealer.Banker.s
Trojan
Medium
Remove All
Ignore

More Social Engineering



http://research.zscaler.com/2011/12/fake-video-codecs-still-going-strong.html

And More



http://www.threattracksecurity.com/it-blog/fake-adobe-flash-updates-resurfaces-in-the-web/

HTML

- Main markup language of the web
- text/html is most common type over HTTP
- Tag based formatting

```
<!DOCTYPE html>
<html>
 <head>
  <title>HTML Example</title>
 </head>
 <body>
  <p><b>Hello...</b> <a href="http://www.cnn.com/">world!</a></p>
 </body>
</html>
```

Try it – load into browser

Some other tags of interest

- <table>
  - <tr><th>Column 1</th><th>Column 2</th></tr>
  - <tr><td>Data A</td><td>Data B</td></td>
- </table>
- Bullet 1
Bullet 2
- <img src = “http://foo.com/bar.jpg”>
- <iframe src = “http://foo.com/bar.html” width =20 height=40/>

JS Inclusion in HTML

- <script> js – blah - blah </script>
  - Technically should be
  - <script language = “javascript”>
- <script src = “foo.js”>
- These are interpreted/run at page load time
- In tag attributes:
  - <button type="button" onclick=”myJSFunc()">Button Name</button>
  - onmouseover, onkeypress, dozens more events that can trigger interpretation/execution of additional js

Let’s have a look

- Developer view of www.cnn.com

Some basics of syntax

- Variable declarations
  - var x; // Now x is undefined
  - var x = 5; // Now x is a Number
  - var x = "John"; // Now x is a String
- Loose dynamic typing a la Perl etc
- All the usual C operators: +, -, ++, &&, …
- + on strings is concatenation
  - “foo” + “bar” == “foobar”
  - “foo”+5 == “foo5”

Scoping Example

```
<!DOCTYPE html>
<html>
  <head>
    <title>JavaScript Example</title>
  </head>
  <body>
  <script>var foo = "Hello world";</script>
    <p><b>Hello...</b> <a href=
“http://www.cnn.com/” onmouseover="alert(foo)">world!</a></p>
  </body>
</html>
```

JavaScript Arrays

- var cars=["Saab","Volvo","BMW"];
  - cars[0] == “Saab”
  - cars.length == 3
- Arrays can be returned from functions and passed to functions

Control Structures

- if(i<5) {foo code} else {bar code}
- for (var i=0;i<N;i++) { blah; blah;}
- while (i < 5) {blah; blah;}
- switch(n) {
  - case 1: blah;break;
  - case 2: blah; break;
  - default: blah}

Object Orientation in JS

- Objects are like hashes/dictionaries
- var person={firstname:"John”, lastname:"Doe", id:5566};
  - person.id==5566
- Everything is an object, and many standard methods available
  - var foo = “bar”;
  - foo.length == 3
  - foo.substring(0,1) == “ba”

Accessing the DOM from JS

- Given Hello world.
  - var x=document.getElementById("intro");
  - var y = document. getElementsByTagName("p")
    - y is now an array of all the elements
    - for(var i=0; i<y.length; i++)…
  - x.innerHTML = "Goodbye."
    - Will replace "Hello world" with "Goodbye"
  - document.createElement("p");