Defending Computer Networks
Lecture 15: HTTP & NIDS

Stuart Staniford

Adjunct Professor of Computer Science

Logistics

- HW3.
  - Due tonight.

Assigned Reading

- Rain Forest Puppy. A Look at Whisker's Anti-IDS Tactics. http://www.ussrback.com/docs/papers/IDS/whiskerids.html

Latest News

Malware infected IAEA computers, but no data was compromised: UN

Reuters, October 22, 2013

Malicious software infected some U.N. nuclear agency computers in recent months but no data in its network is believed to have been compromised, the agency said in a confidential note to member states.

The U.N.'s International Atomic Energy Agency (IAEA) plays a key role in global efforts to prevent the spread of nuclear weapons.

Among other politically sensitive tasks, it is investigating Iran's disputed atomic activities.

The IAEA, in a brief note distributed on Monday evening and seen by Reuters on Tuesday, said an internal investigation had concluded that during past months some computers operated by the agency were infected by malware.

The computers were located in common areas of the agency's Vienna headquarters, known as the Vienna International Centre (VIC), it said.

"Data from some VIC visitors' USB drives is believed to have been compromised, including during the September 2013 Board of Governors meeting and General Conference," it said, referring to two meetings of IAEA member states during that month.

"The Secretariat does not believe that the USB devices themselves were infected or that they could spread the malware further," the U.N. agency added. "The investigation indicates that no data from the IAEA network has been compromised."

Main Goals for Today

- Introduce basics of HTTP
- NIDS in Context of HTTP Server Attacks

HTTP 1.1

- Main protocol that web runs over
- By far most important protocol on Internet
- RFC 2616 (1999)
  - Obsoletes RFC 1945 for HTTP 1.0
  - HTTP originally dates back to Tim Berners Lee/ CERT in 1991 (v 0.9)
- Text based request/response protocol
  - Originally primarily to identify/download files
  - Also provides for web applications

Protocol Layering

- HTTP runs over TCP
- In HTTP 1.1, one TCP connection can have a series of HTTP requests
  - Reverse direction carries responses
- NB: connection structure is not that meaningful to HTTP
  - Different browser may use one or multiple TCP connections
    - And spread requests between them differently.
  - Proxies can rearrange requests/responses to different connections.

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://djce.org.uk/dumprequest

HTTP Header Basics

- Text lines separated by \r\n
- Header is terminated by a blank line (\r\n\r\n)
- Initial request line
  - GET /dumprequest HTTP/1.1\r\n
    - Other methods include POST, CONNECT, HEAD, DELETE, etc.
    - Focus on GET for now
- Followed by headers of form
  - Header: Value…\r\n
  - No request headers are actually required

Let’s try it

- Telnet to www.google.com 80 and try a manually entered request for nosuchpage.html

A Few Popular Request Headers

- Host:
  - Used to specify domain (server might have several).
- User-Agent:
  - Gives browser specifics (allows server to customize responses to browser)
- Referer:
  - What page (etc) sent us here
- Accept-Language:
  - We speak English, or…
- Accept:
  - media formats we accept (eg text/html)

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>
…

HTTP Response Basics

- Text lines separated by \r\n
- Header is terminated by a blank line (\r\n\r\n)
- Initial response line
  - HTTP/1.1 404 Not Found\r\n
    - Indicates status of request.
- Followed by headers of form
  - Header: Value…\r\n
  - No response headers are actually required
    - Though hard to get much done without them…

Important Response Codes

- 200 OK
- 301 Moved Permanently
- 304 Not Modified
- 400 Bad Request
- 404 Not Found
- 500 Internal Server Error

A Few Popular Response Headers

- Content-Type:
  - Media-type of entity attached after header
- Content-Length:
  - Length of same (in bytes)
- Content-encoding:
  - ‘gzip’ means compression applied
- Date:
- Server: software being run on the server

Entity Body

- Follows header
  - either request or response, but more consistently in response direction
  - Can be any media type:
    - text/html, text/plain, image/jpeg, audio/mpeg
    - http://www.iana.org/assignments/media-types
  - Three methods to delineate length:
    - Content-length
    - Transfer-encoding: chunked
    - Connection: close

Detecting Attacks on Web Servers

- Has been a major industry for 15+ years
- Exploits on the servers themselves
- Exploits on cgi scripts,
  - other server-side plugins
- SQL Injection
- Cross-site scripting
- Also HTTP command-and-control
  - Similar issues of detecting bad HTTP requests

Top Snort Rule Files

```
Stuarts-MacBook-Pro:rules stuarts$ du -s -k *.rules |sort -n
-r |head -10
6152    deleted.rules
1216    browser-plugins.rules
792     malware-cnc.rules
688     blacklist.rules
568     server-webapp.rules
392     file-identify.rules
348     file-office.rules
344     server-other.rules
328     pua-adware.rules
316     browser-ie.rules
```

Snort Example 4

alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (msg:"MALWARE-CNC Win.Trojan.Zbot variant in.php outbound connection"; flow:to_server,established; urilen:7; content:"/in.php"; http_uri; content:".ru|0D 0A|User-Agent|3A 20|Mozilla/4.0|0D 0A|"; fast_pattern:only; http_header; content:"|0A|Content-Length|3A 20|"; http_header; metadata:policy balanced-ips drop, policy security-ips drop, ruleset community, service http; reference:url,zeustracker.abuse.ch/monitor.php?ipaddress=195.22.26.231; classtype:trojan-activity; sid:26023; rev:3;)

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

5 Minute Break

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