cisco Networking Academy

Lab - Extract an Executable from a PCAP

Objectives

Part 1: Analyze Pre-Captured Logs and Traffic Captures

Part 2: Extract Downloaded Files from PCAP

Background / Scenario

Looking at logs is very important, but it is also important to understand how network transactions happen at the packet level.

In this lab, you will analyze the traffic in a previously captured pcap file and extract an executable from the file.

Required Resources

- CyberOps Workstation virtual machine

Instructions

Part 1: Analyze Pre-Captured Logs and Traffic Captures

In Part 2, you will work with the **nimda.download.pcap** file. Captured in a previous lab, **nimda.download.pcap** contains the packets related to the download of the Nimda malware. Your version of the file, if you created it in the previous lab and did not reimport your CyberOps Workstation VM, is stored in the /home/analyst directory. However, a copy of that file is also stored in the **CyberOps Workstation VM**, under the **/home/analyst/lab.support.files/pcaps** directory so that you can complete this lab. For consistency of output, the lab will use the stored version in the **pcaps** directory.

While **tcpdump** can be used to analyze captured files, **Wireshark's** graphical interface makes the task much easier. It is also important to note that **tcpdump** and **Wireshark** share the same file format for packet captures; therefore, PCAP files created by one tool can be opened by the other.

a. Change directory to the **lab.support.files/pcaps** folder, and get a listing of files using the **ls –l** command.

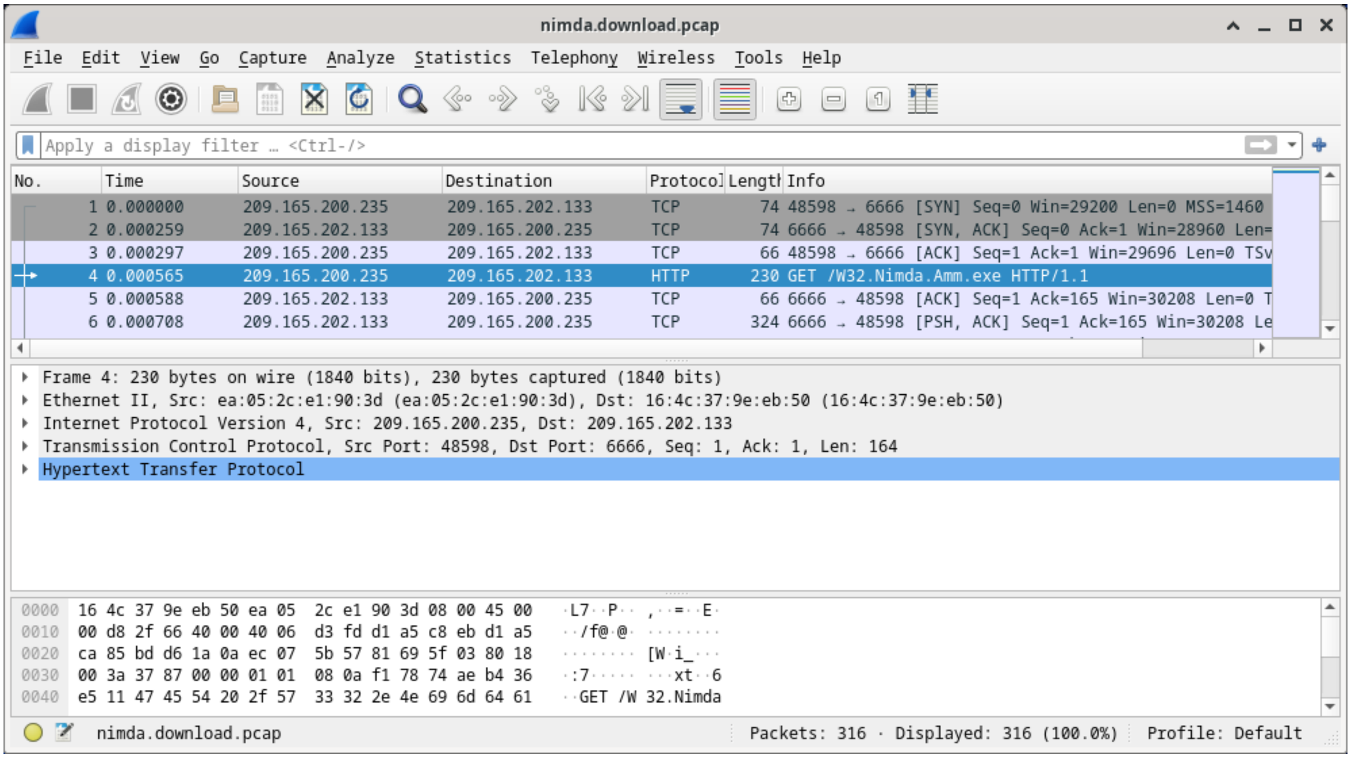
```
[analyst@secOps ~]$ cd lab.support.files/pcaps
[analyst@secOps pcaps]$ ls -l
total 7460
-rw-r--r-- 1 analyst analyst 3510551 Aug  7 15:25 lab_prep.pcap
-rw-r--r-- 1 analyst analyst  371462 Jun 22 10:47 nimda.download.pcap
-rw-r--r-- 1 analyst analyst 3750153 May 25 11:10 wannacry_download_pcap.pcap
[analyst@secOps pcaps]$
```

b. Issue the command below to open the **nimda.download.pcap** file in Wireshark.

```
[analyst@secOps pcaps]$ wireshark nimda.download.pcap &
```

c. The **nimda.download.pcap** file contains the packet capture related to the malware download performed in a previous lab. The **pcap** contains all the packets sent and received while **tcpdump** was running.

Select the fourth packet in the capture and expand the Hypertext Transfer Protocol to display as shown below.


nimda.download.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
No.  Time      Source           Destination      Protocol Length Info
   1 0.000000  209.165.200.235  209.165.202.133  TCP       74 48598 → 6666 [SYN] Seq=0 Win=29200 Len=0 MSS=1460
   2 0.000259  209.165.202.133  209.165.200.235  TCP       74 6666 → 48598 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=
   3 0.000297  209.165.200.235  209.165.202.133  TCP       66 48598 → 6666 [ACK] Seq=1 Ack=1 Win=29696 Len=0 TSv
   4 0.000565  209.165.200.235  209.165.202.133  HTTP     230 GET /W32.Nimda.Amm.exe HTTP/1.1
   5 0.000588  209.165.202.133  209.165.200.235  TCP       66 6666 → 48598 [ACK] Seq=1 Ack=165 Win=30208 Len=0 T
   6 0.000708  209.165.202.133  209.165.200.235  TCP      324 6666 → 48598 [PSH, ACK] Seq=1 Ack=165 Win=30208 Le
```

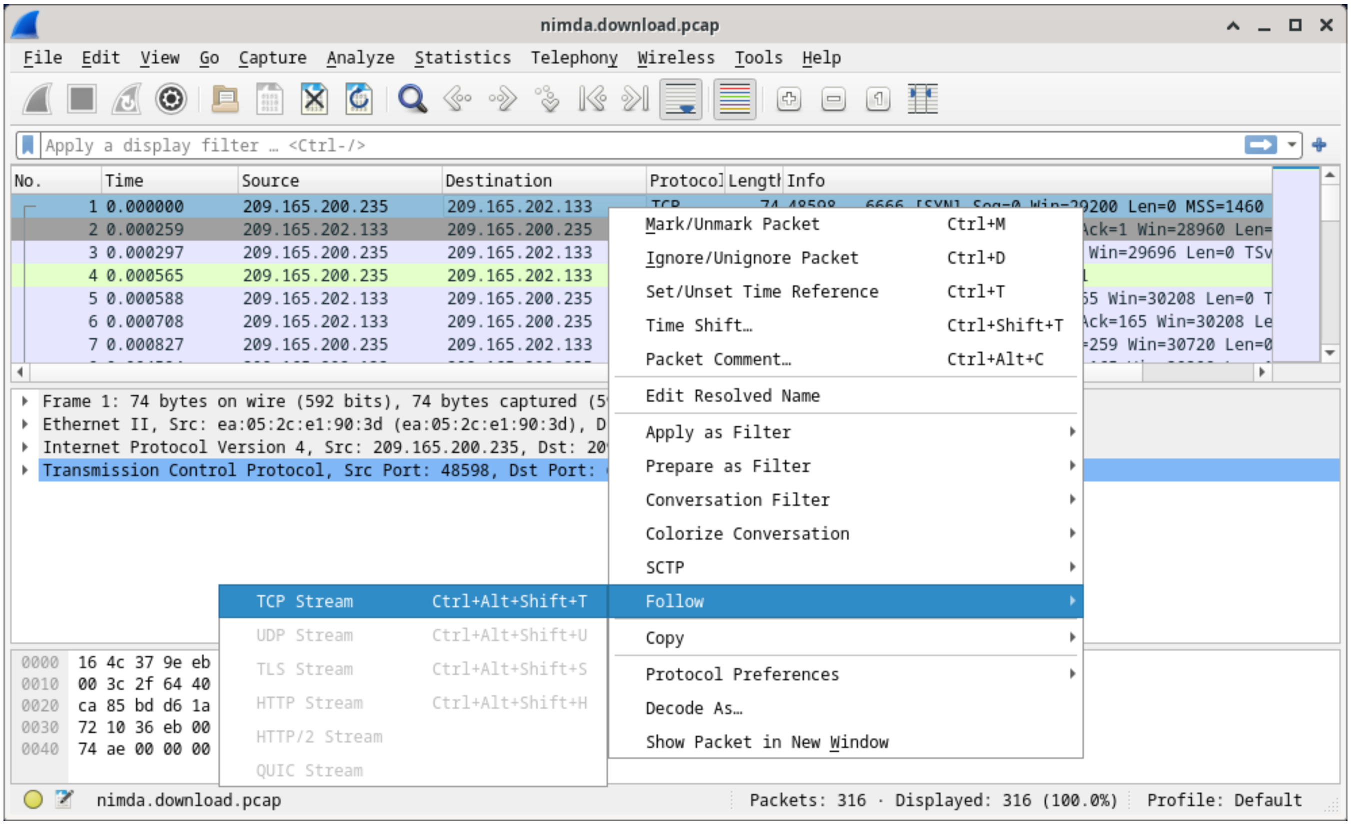
```
▸ Frame 4: 230 bytes on wire (1840 bits), 230 bytes captured (1840 bits)
▸ Ethernet II, Src: ea:05:2c:e1:90:3d (ea:05:2c:e1:90:3d), Dst: 16:4c:37:9e:eb:50 (16:4c:37:9e:eb:50)
▸ Internet Protocol Version 4, Src: 209.165.200.235, Dst: 209.165.202.133
▸ Transmission Control Protocol, Src Port: 48598, Dst Port: 6666, Seq: 1, Ack: 1, Len: 164
▸ Hypertext Transfer Protocol
```

```
0000  16 4c 37 9e eb 50 ea 05  2c e1 90 3d 08 00 45 00   ·L7··P··  ,··=··E·
0010  00 d8 2f 66 40 00 40 06  d3 fd d1 a5 c8 eb d1 a5   ··/f@·@·  ········
0020  ca 85 bd d6 1a 0a ec 07  5b 57 81 69 5f 03 80 18   ········  [W·i_···
0030  00 3a 37 87 00 00 01 01  08 0a f1 78 74 ae b4 36   ·:7·····  ···xt··6
0040  e5 11 47 45 54 20 2f 57  33 32 2e 4e 69 6d 64 61   ··GET /W  32.Nimda
```

```
nimda.download.pcap    Packets: 316 · Displayed: 316 (100.0%)    Profile: Default
```

d. Packets one through three are the TCP handshake. The fourth packet shows the request for the malware file. Confirming what was already known, the request was done over HTTP, sent as a GET request.

e. Because HTTP runs over TCP, it is possible to use **Wireshark**’s **Follow TCP Stream** feature to rebuild the TCP transaction. Select the first TCP packet in the capture, a SYN packet. Right-click it and choose **Follow** > **TCP Stream**.


nimda.download.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Mark/Unmark Packet Ctrl+M
Ignore/Unignore Packet Ctrl+D
Set/Unset Time Reference Ctrl+T
Time Shift… Ctrl+Shift+T
Packet Comment… Ctrl+Alt+C
Edit Resolved Name
Apply as Filter
Prepare as Filter
Conversation Filter
Colorize Conversation
SCTP
Follow
Copy
Protocol Preferences
Decode As…
Show Packet in New Window
TCP Stream Ctrl+Alt+Shift+T
UDP Stream Ctrl+Alt+Shift+U
TLS Stream Ctrl+Alt+Shift+S
HTTP Stream Ctrl+Alt+Shift+H
HTTP/2 Stream
QUIC Stream

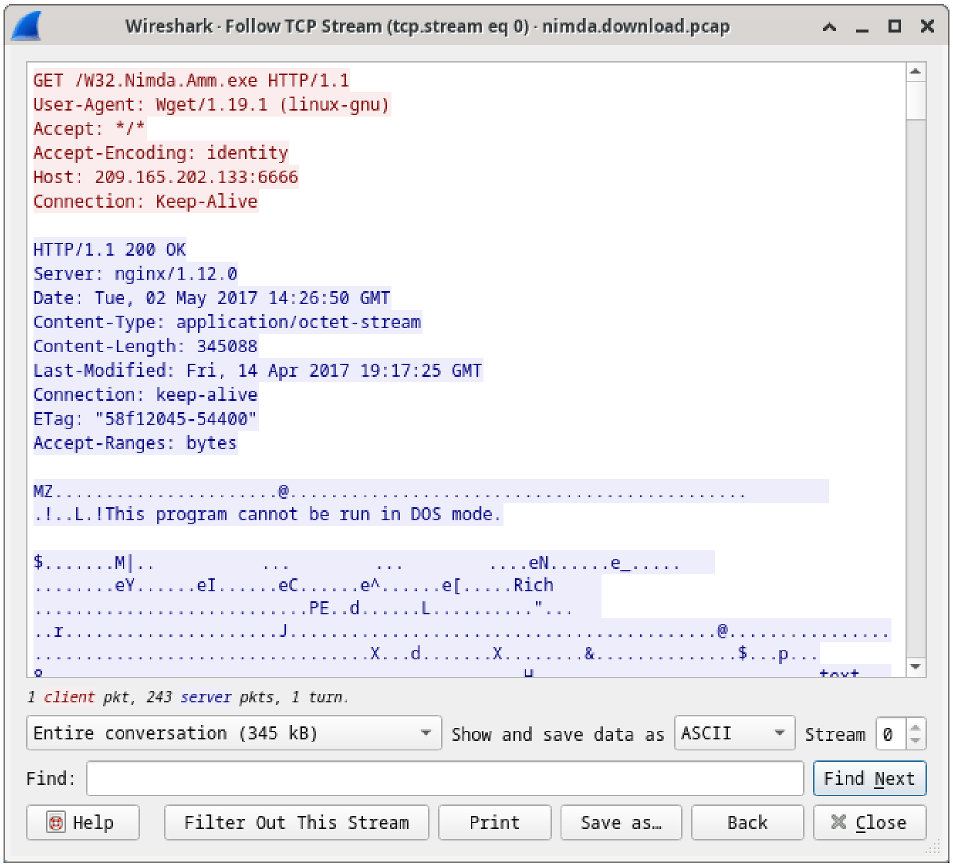
```
No.  Time      Source           Destination      Protocol Length Info
  1 0.000000  209.165.200.235  209.165.202.133  TCP      74 48598 → 6666 [SYN] Seq=0 Win=29200 Len=0 MSS=1460
  2 0.000259  209.165.202.133  209.165.200.235                   Ack=1 Win=28960 Len=
  3 0.000297  209.165.200.235  209.165.202.133                   Win=29696 Len=0 TSv
  4 0.000565  209.165.200.235  209.165.202.133
  5 0.000588  209.165.202.133  209.165.200.235                   55 Win=30208 Len=0 T
  6 0.000708  209.165.202.133  209.165.200.235                   Ack=165 Win=30208 Le
  7 0.000827  209.165.200.235  209.165.202.133                   =259 Win=30720 Len=0

▸ Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (5
▸ Ethernet II, Src: ea:05:2c:e1:90:3d (ea:05:2c:e1:90:3d), D
▸ Internet Protocol Version 4, Src: 209.165.200.235, Dst: 20
▸ Transmission Control Protocol, Src Port: 48598, Dst Port:

0000  16 4c 37 9e eb
0010  00 3c 2f 64 40
0020  ca 85 bd d6 1a
0030  72 10 36 eb 00
0040  74 ae 00 00 00

nimda.download.pcap    Packets: 316 · Displayed: 316 (100.0%)    Profile: Default
```

f. Wireshark displays another window containing the details for the entire selected TCP flow.



Wireshark · Follow TCP Stream (tcp.stream eq 0) · nimda.download.pcap

```
GET /W32.Nimda.Amm.exe HTTP/1.1
User-Agent: Wget/1.19.1 (linux-gnu)
Accept: */*
Accept-Encoding: identity
Host: 209.165.202.133:6666
Connection: Keep-Alive

HTTP/1.1 200 OK
Server: nginx/1.12.0
Date: Tue, 02 May 2017 14:26:50 GMT
Content-Type: application/octet-stream
Content-Length: 345088
Last-Modified: Fri, 14 Apr 2017 19:17:25 GMT
Connection: keep-alive
ETag: "58f12045-54400"
Accept-Ranges: bytes

MZ......................@.........................................
.!..L.!This program cannot be run in DOS mode.

$.......M|..          ...          ...         ....eN......e_.....
........eY.......eI.......eC.......e^.......e[.....Rich
...........................PE..d......L..........."...
..r....................J..........................................@................
.................................X...d.......X........&..............$...p...
```

1 client pkt, 243 server pkts, 1 turn.

Entire conversation (345 kB) Show and save data as ASCII Stream 0

Find:

What are all those symbols shown in the **Follow TCP Stream** window? Are they connection noise? Data? Explain.

Those characters are actual contents of the downloaded malware but since its binary data it cannot be shown in clear text.

There are a few readable words spread among the symbols. Why are they there?

Those are the strings in the malware executable.

Challenge Question: Despite the **W32.Nimda.Amm.exe** name, this executable is not the famous worm. For security reasons, this is another executable file that was renamed as **W32.Nimda.Amm.exe**. Using the word fragments displayed by **Wireshark**'s **Follow TCP Stream** window, can you tell what executable this really is?

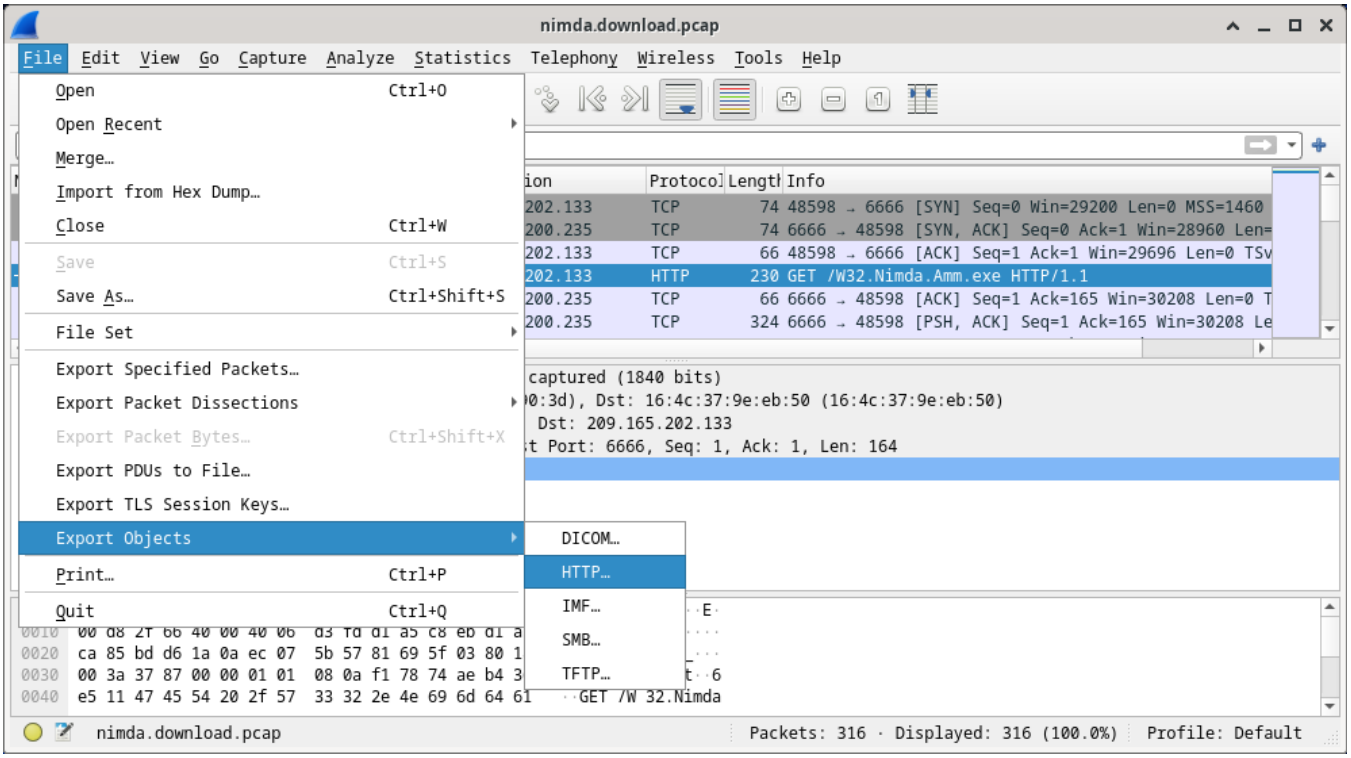
g. Click **Close** in the Follow TCP Stream window to return to the Wireshark nimda.download.pcap file.

Part 2: Extract Downloaded Files from PCAP

Because capture files contain all packets related to traffic, a PCAP of a download can be used to retrieve a previously downloaded file. Follow the steps below to use **Wireshark** to retrieve the Nimda malware.

a. In that fourth packet in the **nimda.download.pcap** file, notice that the **HTTP GET** request was generated from **209.165.200.235** to **209.165.202.133**. The Info column also shows this is in fact the GET request for the file.

b. With the GET request packet selected, navigate to **File > Export Objects > HTTP**, from **Wireshark**'s menu.



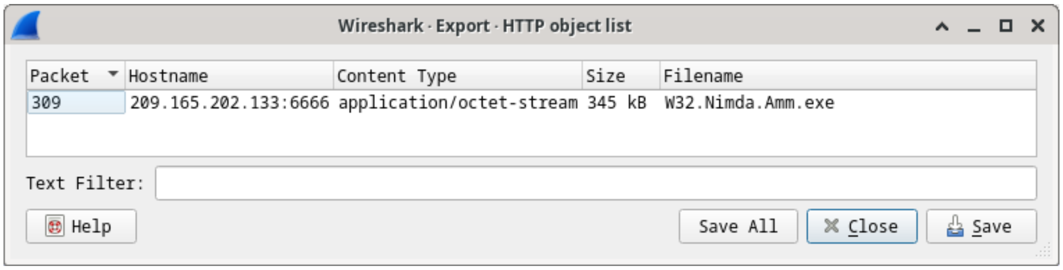
```
ion      Protocol Length Info
202.133  TCP        74 48598 → 6666 [SYN] Seq=0 Win=29200 Len=0 MSS=1460
200.235  TCP        74 6666 → 48598 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=
202.133  TCP        66 48598 → 6666 [ACK] Seq=1 Ack=1 Win=29696 Len=0 TSv
202.133  HTTP      230 GET /W32.Nimda.Amm.exe HTTP/1.1
200.235  TCP        66 6666 → 48598 [ACK] Seq=1 Ack=165 Win=30208 Len=0 T
200.235  TCP       324 6666 → 48598 [PSH, ACK] Seq=1 Ack=165 Win=30208 Le
```

```
captured (1840 bits)
)0:3d), Dst: 16:4c:37:9e:eb:50 (16:4c:37:9e:eb:50)
 Dst: 209.165.202.133
t Port: 6666, Seq: 1, Ack: 1, Len: 164
```

```
0010  00 d8 2f 66 40 00 40 06  d3 fd d1 a5 c8 eb d1 a            ··E·
0020  ca 85 bd d6 1a 0a ec 07  5b 57 81 69 5f 03 80 1           ····
0030  00 3a 37 87 00 00 01 01  08 0a f1 78 74 ae b4 3           t··6
0040  e5 11 47 45 54 20 2f 57  33 32 2e 4e 69 6d 64 61    ··GET /W 32.Nimda
```

nimda.download.pcap Packets: 316 · Displayed: 316 (100.0%) Profile: Default

c. Wireshark will display all HTTP objects present in the TCP flow that contains the GET request. In this case, only the **W32.Nimda.Amm.exe** file is present in the capture. It will take a few seconds before the file is displayed.



Packet	Hostname	Content Type	Size	Filename
309	209.165.202.133:6666	application/octet-stream	345 kB	W32.Nimda.Amm.exe

Why is **W32.Nimda.Amm.exe** the only file in the capture?

because tcp dump capture was stopped after wget request ended

d. In the **HTTP object list** window, select the **W32.Nimda.Amm.exe** file and click **Save As** at the bottom of the screen.

e. Click the left arrow until you see the **Home** button. Click Home and then click the **analyst** folder (not the analyst tab). Save the file there.

f. Return to your terminal window and ensure the file was saved. Change directory to the **/home/analyst** folder and list the files in the folder using the **ls -l** command.

```
[analyst@secOps pcaps]$ cd /home/analyst
[analyst@secOps ~]$ ls –l
total 364
drwxr-xr-x 2 analyst analyst   4096 Sep 26  2014 Desktop
drwx------ 3 analyst analyst   4096 May 25 11:16 Downloads
drwxr-xr-x 2 analyst analyst   4096 May 22 08:39 extra
drwxr-xr-x 8 analyst analyst   4096 Jun 22 11:38 lab.support.files
drwxr-xr-x 2 analyst analyst   4096 Mar  3 15:56 second_drive
-rw-r--r-- 1 analyst analyst 345088 Jun 22 15:12 W32.Nimda.Amm.exe
[analyst@secOps ~]$
```

Was the file saved?

g. The **file** command gives information on the file type. Use the file command to learn a little more about the malware, as show below:

```
[analyst@secOps ~]$ file W32.Nimda.Amm.exe
W32.Nimda.Amm.exe: PE32+ executable (console) x86-64, for MS Windows
[analyst@secOps ~]$
```

As seen above, **W32.Nimda.Amm.exe** is indeed a Windows executable file.

In the malware analysis process, what would be a probable next step for a security analyst?

Run strings, run in decompiler or run in sandboxed env