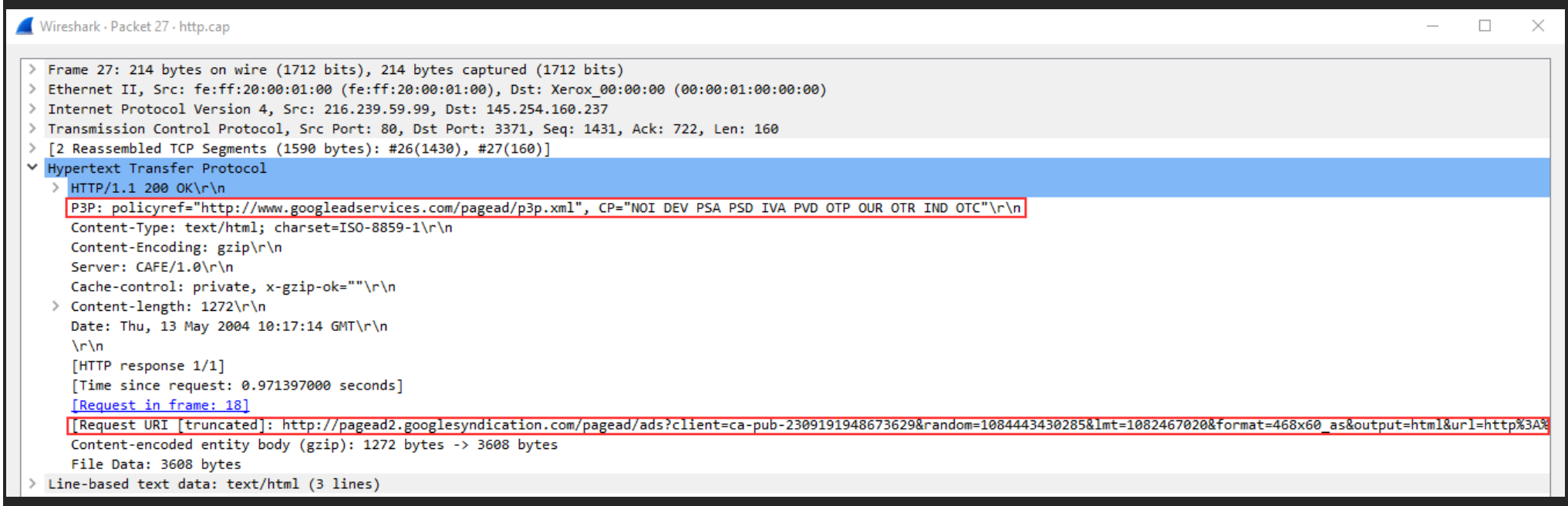
THM_Wireshark 101 [Task 11 HTTP]

HTTP or Hypertext Transfer Protocol is a commonly used port for the world wide web and used by some websites, however, its encrypted counterpart: HTTPS is more common which we will discuss in the next text. HTTP is used to send GET and POST requests to a web server in order to receive things like webpages. Knowing how to analyze HTTP can be helpful to quickly spot things like SQLi, Web Shells, and other web-related attack vectors.

HTTP Traffic Overview

You should already have a general understanding of how HTTP works before completing this room; however, if you need a refresher you can read the official paper by the IETF on HTTP methods.

HTTP is one of the most straight forward protocols for packet analysis, the protocol is straight to the point and does not include any handshakes or prerequisites before communication.

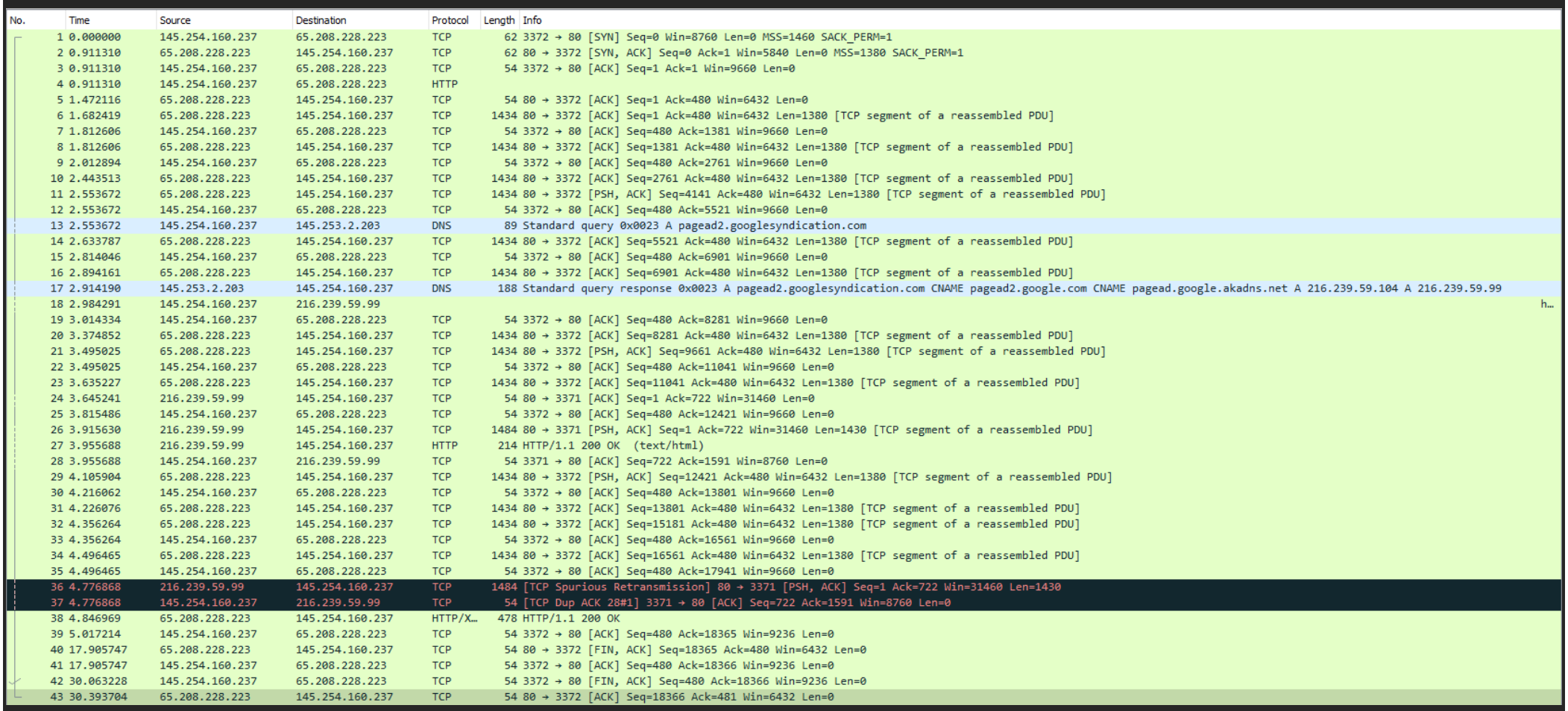


Above we can see a sample HTTP packet, looking at an HTTP packet we can easily gather information since the data stream is not encrypted like the HTTP counterpart HTTPS. Some of the important information we can gather from the packet is the Request URI, File Data, Server.

Now that we understand the basic structure of an HTTP packet we can move on to looking at a sample HTTP packet capture to get hands-on with the packets.

Practical HTTP Packet Analysis

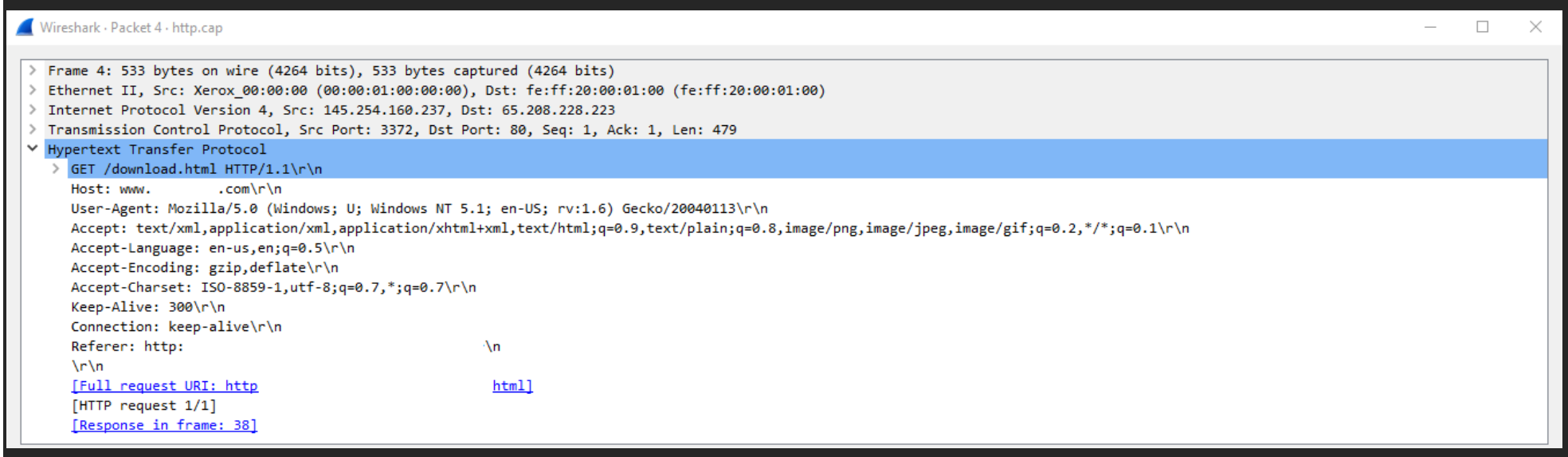
To get an understanding of the flow of HTTP packets and get hands-on with the packets, we can analyze the http.cap file. Go to the folder /root/Rooms/Wireshark101 on the AttackBox and double click the task11.pcap file to open it in Wireshark; you can also download the pcap on this task.



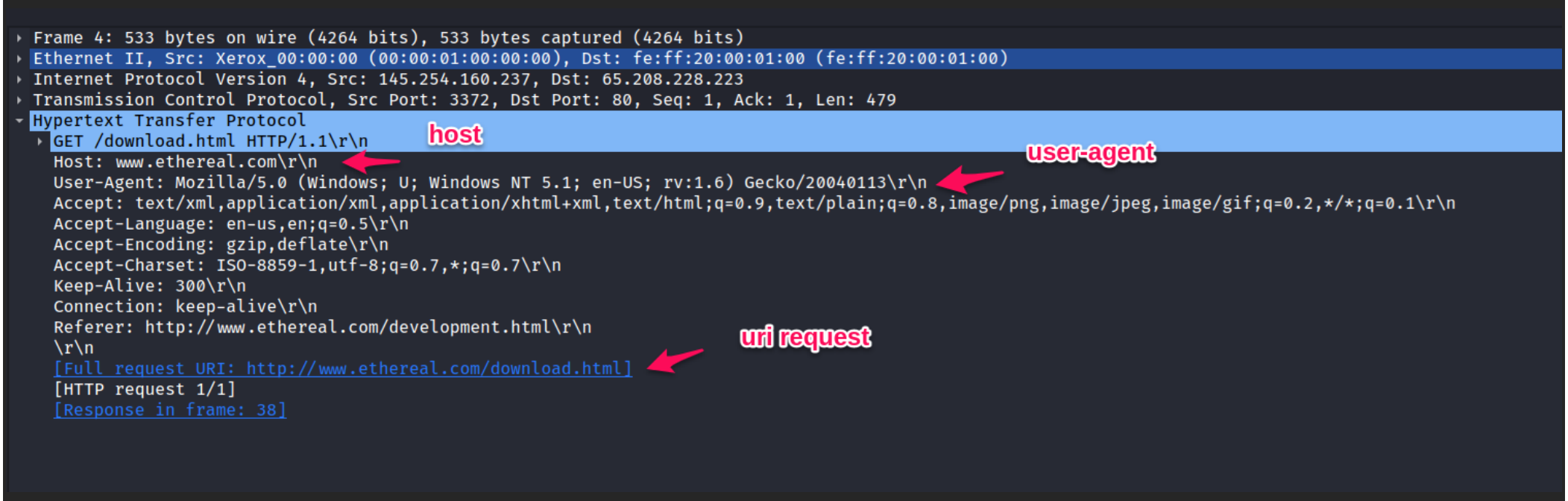
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	145.254.160.237	65.208.228.223	TCP	62	3372 → 80 [SYN] Seq=0 Win=8760 Len=0 MSS=1460 SACK_PERM=1
2	0.911310	65.208.228.223	145.254.160.237	TCP	62	80 → 3372 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1380 SACK_PERM=1
3	0.911310	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=1 Ack=1 Win=9660 Len=0
4	0.911310	145.254.160.237	65.208.228.223	HTTP		
5	1.472116	65.208.228.223	145.254.160.237	TCP	54	80 → 3372 [ACK] Seq=1 Ack=480 Win=6432 Len=0
6	1.682419	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=1 Ack=480 Win=6432 Len=1380 [TCP segment of a reassembled PDU]
7	1.812606	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=1381 Win=9660 Len=0
8	1.812606	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=1381 Ack=480 Win=6432 Len=1380 [TCP segment of a reassembled PDU]
9	2.012894	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=2761 Win=9660 Len=0
10	2.443513	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=2761 Ack=480 Win=6432 Len=1380 [TCP segment of a reassembled PDU]
11	2.553672	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [PSH, ACK] Seq=4141 Ack=480 Win=6432 Len=1380 [TCP segment of a reassembled PDU]
12	2.553672	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=5521 Win=9660 Len=0
13	2.553672	145.254.160.237	145.253.2.203	DNS	89	Standard query 0x0023 A pagead2.googlesyndication.com
14	2.633787	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=5521 Ack=480 Win=6432 Len=1380 [TCP segment of a reassembled PDU]
15	2.814046	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=6901 Win=9660 Len=0
16	2.894161	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=6901 Ack=480 Win=6432 Len=1380 [TCP segment of a reassembled PDU]
17	2.914190	145.253.2.203	145.254.160.237	DNS	188	Standard query response 0x0023 A pagead2.googlesyndication.com CNAME pagead2.google.com CNAME pagead.google.akadns.net A 216.239.59.104 A 216.239.59.99
18	2.984291	145.254.160.237	216.239.59.99			h…
19	3.014334	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=8281 Win=9660 Len=0
20	3.374852	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=8281 Ack=480 Win=6432 Len=1380 [TCP segment of a reassembled PDU]
21	3.495025	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [PSH, ACK] Seq=9661 Ack=480 Win=6432 Len=1380 [TCP segment of a reassembled PDU]
22	3.495025	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=11041 Win=9660 Len=0
23	3.635227	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=11041 Ack=480 Win=6432 Len=1380 [TCP segment of a reassembled PDU]
24	3.645241	216.239.59.99	145.254.160.237	TCP	54	80 → 3371 [ACK] Seq=1 Ack=722 Win=31460 Len=0
25	3.815486	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=12421 Win=9660 Len=0
26	3.915630	216.239.59.99	145.254.160.237	TCP	1484	80 → 3371 [PSH, ACK] Seq=1 Ack=722 Win=31460 Len=1430 [TCP segment of a reassembled PDU]
27	3.955688	216.239.59.99	145.254.160.237	HTTP	214	HTTP/1.1 200 OK (text/html)
28	3.955688	145.254.160.237	216.239.59.99	TCP	54	3371 → 80 [ACK] Seq=722 Ack=1591 Win=8760 Len=0
29	4.105904	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [PSH, ACK] Seq=12421 Ack=480 Win=6432 Len=1380 [TCP segment of a reassembled PDU]
30	4.216062	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=13801 Win=9660 Len=0
31	4.226076	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=13801 Ack=480 Win=6432 Len=1380 [TCP segment of a reassembled PDU]
32	4.356264	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=15181 Ack=480 Win=6432 Len=1380 [TCP segment of a reassembled PDU]
33	4.356264	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=16561 Win=9660 Len=0
34	4.496465	65.208.228.223	145.254.160.237	TCP	1434	80 → 3372 [ACK] Seq=16561 Ack=480 Win=6432 Len=1380 [TCP segment of a reassembled PDU]
35	4.496465	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=17941 Win=9660 Len=0
36	4.776868	216.239.59.99	145.254.160.237	TCP	1484	[TCP Spurious Retransmission] 80 → 3371 [PSH, ACK] Seq=1 Ack=722 Win=31460 Len=1430
37	4.776868	145.254.160.237	216.239.59.99	TCP	54	[TCP Dup ACK 28#1] 3371 → 80 [ACK] Seq=722 Ack=1591 Win=8760 Len=0
38	4.846969	65.208.228.223	145.254.160.237	HTTP/X…	478	HTTP/1.1 200 OK
39	5.017214	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=18365 Win=9236 Len=0
40	17.905747	65.208.228.223	145.254.160.237	TCP	54	80 → 3372 [FIN, ACK] Seq=18365 Ack=480 Win=6432 Len=0
41	17.905747	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=480 Ack=18366 Win=9236 Len=0
42	30.063228	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [FIN, ACK] Seq=480 Ack=18366 Win=9236 Len=0
43	30.393704	65.208.228.223	145.254.160.237	TCP	54	80 → 3372 [ACK] Seq=18366 Ack=481 Win=6432 Len=0

After opening the PCAP we can see that this is just a simple HTTPpacket capture with a few requests.

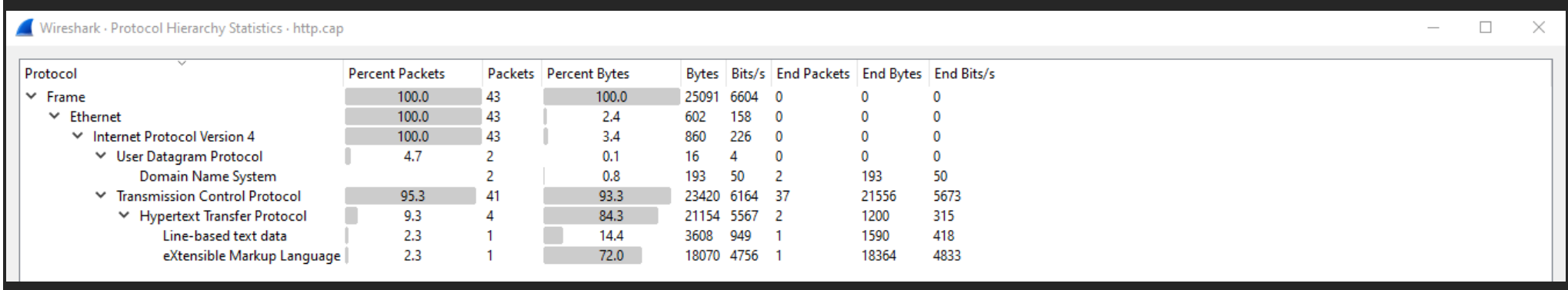
Navigating deeper into the packet capture we can look at the details of one of the HTTP requests for example packet 4.

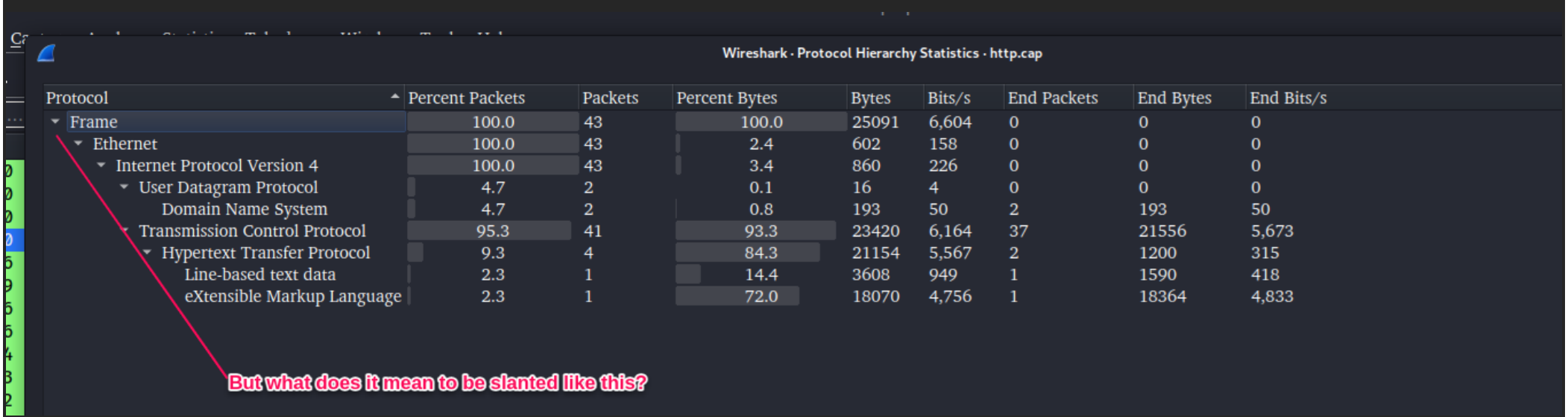


From this packet we can identify some very important information like the host, user-agent, requested URI, and response.



We can use some of Wireshark's built-in features to help digest all of this data and organize it for further future analysis. We can begin by looking at a very useful feature in Wireshark to organize the protocols present in a capture the Protocol Hierarchy. **Navigate to Statistics > Protocol Hierarchy.**





This information can be very useful in practical applications like threat hunting to identify discrepancies in packet captures.

The next feature in Wireshark we will look at is the Export HTTP Object. This feature will allow us to organize all requested URIs in the capture. **To use Export HTTP Object navigate to file > Export Objects > HTTP.**

I've done this quite a few times before so I'll skip it

Wireshark · Export · HTTP object list

Packet	Hostname	Content Type	Size	Filename
27	pagead2.googlesyndication.com	text/html	3608 bytes	ads?client=ca-pub-2309191948673629&random=1084443430285&lr
38		text/html	18 kB	.html

Similar to the Protocol Hierarchy this can be useful to quickly identify possible discrepancies in captures.

The last feature we will cover in this section of this room is Endpoints. This feature allows the user to organize all endpoints and IPs found within a specific capture. Just like the other features, this can be useful to identify where a discrepancy is originating from. **To use the Endpoints feature navigate to Statistics > Endpoints.**

Wireshark · Endpoints · http.cap

Ethernet · 2 | IPv4 · 4 | IPv6 | TCP · 4 | UDP · 2

Address	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes	Country	City	AS Number	AS Organization
65.208.228.223	34	20 k	18	19 k	16	1351	—	—	—	—
145.253.2.203	2	277	1	188	1	89	—	—	—	—
	43	25 k	20	2323	23	22 k	—	—	—	—
216.239.59.99	7	4119	4	3236	3	883	—	—	—	—

Ethernet · 2 | IPv4 · 4 | IPv6 | TCP · 4 | UDP · 2

Address	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes
00:00:01:00:00:00	43	25 k	20	2,323	23	
fe:ff:20:00:01:00	43	25 k	23	22 k	20	

I can see two MAC addresses, 4 IP addresses, all of which recieved TCP packets, but only two recieved UDP

Wireshark · Endpoints · http.cap

Ethernet · 2 | IPv4 · 4 | IPv6 | TCP · 4 | UDP · 2

Address	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes	Country	City	AS Number	AS Organization
65.208.228.223	34	20 k	18	19 k	16	1,351	—	—	—	—
145.253.2.203	2	277	1	188	1	89	—	—	—	—
145.254.160.237	43	25 k	20	2,323	23	22 k	—	—	—	—
216.239.59.99	7	4,119	4	3,236	3	883	—	—	—	—

Ethernet · 2 | IPv4 · 4 | IPv6 | TCP · 4 | UDP · 2

Address	Port	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes
65.208.228.223	80	34	20 k	18	19 k	16	
145.254.160.237	3372	34	20 k	16	1,351	18	
145.254.160.237	3371	7	4,119	3	883	4	
216.239.59.99	80	7	4,119	4	3,236	3	

Ethernet · 2 | IPv4 · 4 | IPv6 | TCP · 4 | UDP · 2

Address	Port	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Byt
145.253.2.203	53	2	277	1	188	1	
145.254.160.237	3009	2	277	1	89	1	

HTTP is not a common protocol to see too much as HTTPS is now more commonly used; however, HTTP is still used often and can be very easy to analyze if given the opportunity.

Answer the questions below

What percent of packets originate from Domain Name System?

4.7

Wireshark · Protocol Hierarchy Statistics · http.cap

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bi
Frame	100.0	43	100.0	25091	6,604	0	0	0
Ethernet	100.0	43	2.4	602	158	0	0	0
Internet Protocol Version 4	100.0	43	3.4	860	226	0	0	0
User Datagram Protocol	4.7	2	0.1	16	4	0	0	0
Domain Name System	4.7	2	0.8	193	50	2	193	50
Transmission Control Protocol	95.3	41	93.3	23420	6,164	37	21556	5,673
Hypertext Transfer Protocol	9.3	4	84.3	21154	5,567	2	1200	315
Line-based text data	2.3	1	14.4	3608	949	1	1590	418
eXtensible Markup Language	2.3	1	72.0	18070	4,756	1	18364	4,833

What endpoint ends in .237?

145.254.160.237

Ethernet · 2 | IPv4 · 4 | IPv6 | TCP · 4 | UDP · 2

Address	Port	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes
145.253.2.203	53	2	277	1	188	1	
145.254.160.237	3009	2	277	1	89	1	

What is the user-agent listed in packet 4?

Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.6) Gecko/20040113

```
▸ Frame 4: 533 bytes on wire (4264 bits), 533 bytes captured (4264 bits)
▸ Ethernet II, Src: Xerox_00:00:00 (00:00:01:00:00:00), Dst: fe:ff:20:00:01:00 (fe:ff:20:00:01:00)
▸ Internet Protocol Version 4, Src: 145.254.160.237, Dst: 65.208.228.223
▸ Transmission Control Protocol, Src Port: 3372, Dst Port: 80, Seq: 1, Ack: 1, Len: 479
▾ Hypertext Transfer Protocol
  ▸ GET /download.html HTTP/1.1\r\n
    Host: www.ethereal.com\r\n
    User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.6) Gecko/20040113\r\n
    Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,ima
    Accept-Language: en-us,en;q=0.5\r\n
    Accept-Encoding: gzip,deflate\r\n
    Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7\r\n
    Keep-Alive: 300\r\n
    Connection: keep-alive\r\n
    Referer: http://www.ethereal.com/development.html\r\n
    \r\n
    [Full request URI: http://www.ethereal.com/download.html]
    [HTTP request 1/1]
    [Response in frame: 38]
```

Looking at the data stream what is the full request URI from packet 18?

http://pagead2.googlesyndication.com/pagead/ads?client=ca-pub-2309191948673629&random=1084443430285&lmt=1082467020&format=468x60_as&output=html&url=http%3A%2F%2Fwww.ethereal.com%2Fdownload.html&color_bg=FFFFFF&color_text=333333&color_link=000000&color_url=666633&color_border=666633

```
    Keep-Alive: 300\r\n
    Connection: keep-alive\r\n
    Referer: http://www.ethereal.com/download.html\r\n
    \r\n
    [Full request URI [truncated]: http://pagead2.googlesyndication.com/pagead/ads?client=ca-pub-23091919
    [HTTP request 1/1]
    [Response in frame: 27]
```

What domain name was requested from packet 38?

http://www.ethereal.com

```
Hypertext Transfer Protocol
  ▸ HTTP/1.1 200 OK\r\n
    Date: Thu, 13 May 2004 10:17:12 GMT\r\n
    Server: Apache\r\n
    Last-Modified: Tue, 20 Apr 2004 13:17:00 GMT\r\n
    ETag: "9a01a-4696-7e354b00"\r\n
    Accept-Ranges: bytes\r\n
  ▸ Content-Length: 18070\r\n
    Keep-Alive: timeout=15, max=100\r\n
    Connection: Keep-Alive\r\n
    Content-Type: text/html; charset=ISO-8859-1\r\n
    \r\n
    [HTTP response 1/1]
    [Time since request: 3.935659000 seconds]
    [Request in frame: 4]
    [Request URI: http://www.ethereal.com/download.html]
    File Data: 18070 bytes
▸ eXtensible Markup Language
```

Looking at the data stream what is the full request URI from packet 38?

http://www.ethereal.com/download.html