Chih-Chao, Chang

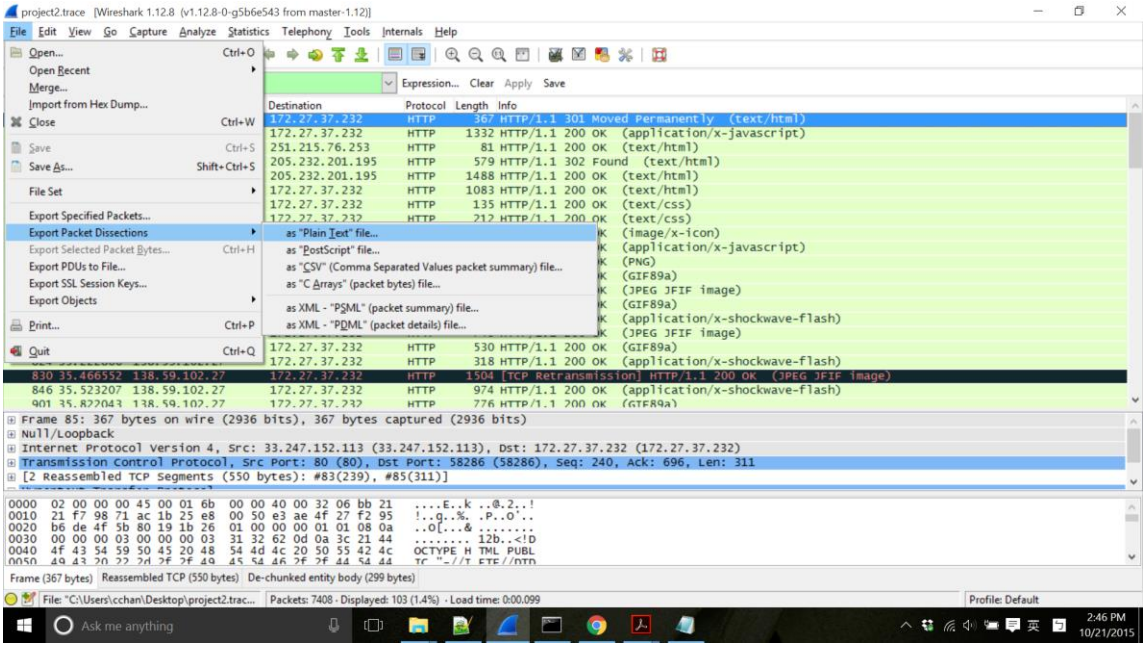
1.

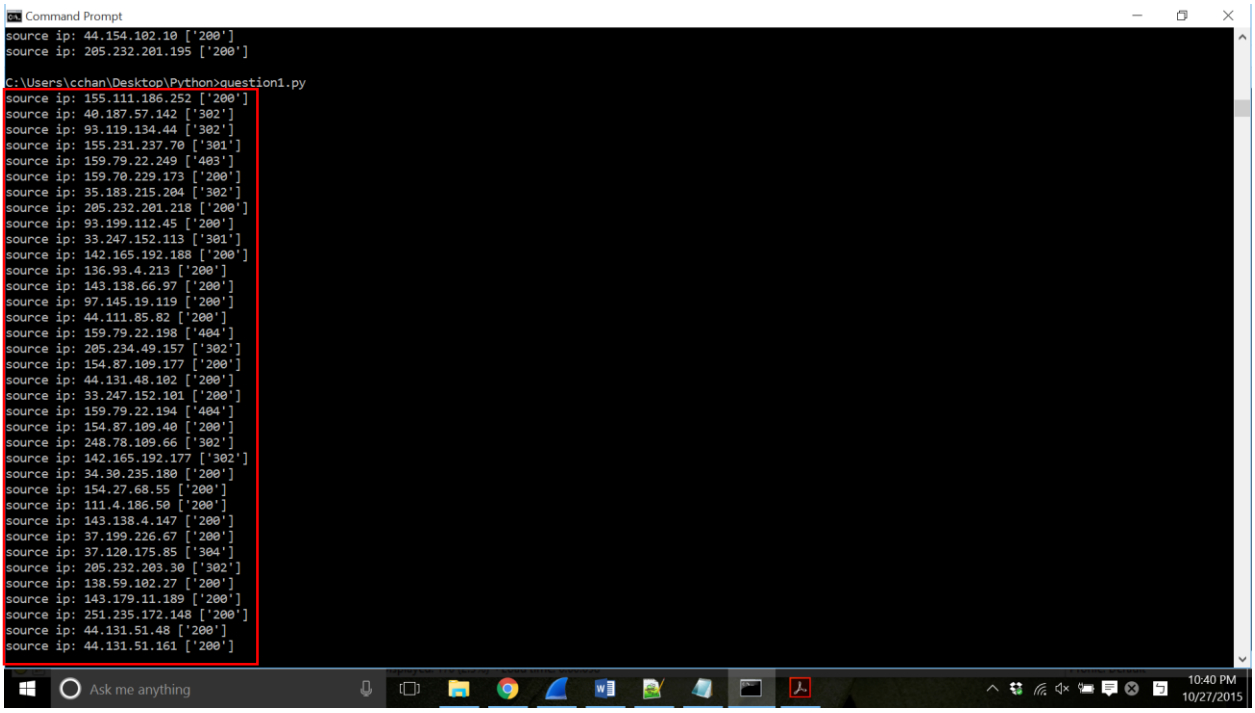
Filter: http.response

I examine the IP address only for http response which includes the servers that engaged in a valid instance of the HTTP protocol, for the status code returned were 404, 200 and other types. I also export the information into txt file (q1.txt)and then write a Python script (question1.py)to find the IP address.

The IP list is following (in Wireshark shown order not script order but they are the same):

1) 143.138.4.147	2) 33.247.152.113	3) 33.247.152.101	4) 44.131.48.102
5) 34.30.235.180	6) 97.145.19.119	7) 37.199.226.67	8) 143.138.66.97
9) 37.120.175.85	10) 44.111.85.82	11) 44.131.51.48	12)248.78.109.66
13)138.59.102.27	14) 154.87.109.40	15)154.87.109.177	16)93.119.134.44
17)155.231.237.70	18)35.183.215.204	19)159.70.229.173	20)251.235.172.148
21) 205.232.201.218	22) 44.131.51.161	23)155.111.186.252	24) 136.93.4.213
25)93.199.112.45	26) 143.179.11.189	27) 154.27.68.55	28) 111.4.186.50
29)40.187.57.142	30) 142.165.192.177	31) 142.165.192.188	32) 205.234.49.157
33) 159.79.22.194	34)159.79.22.198	35) 205.232.203.30	36)159.79.22.249

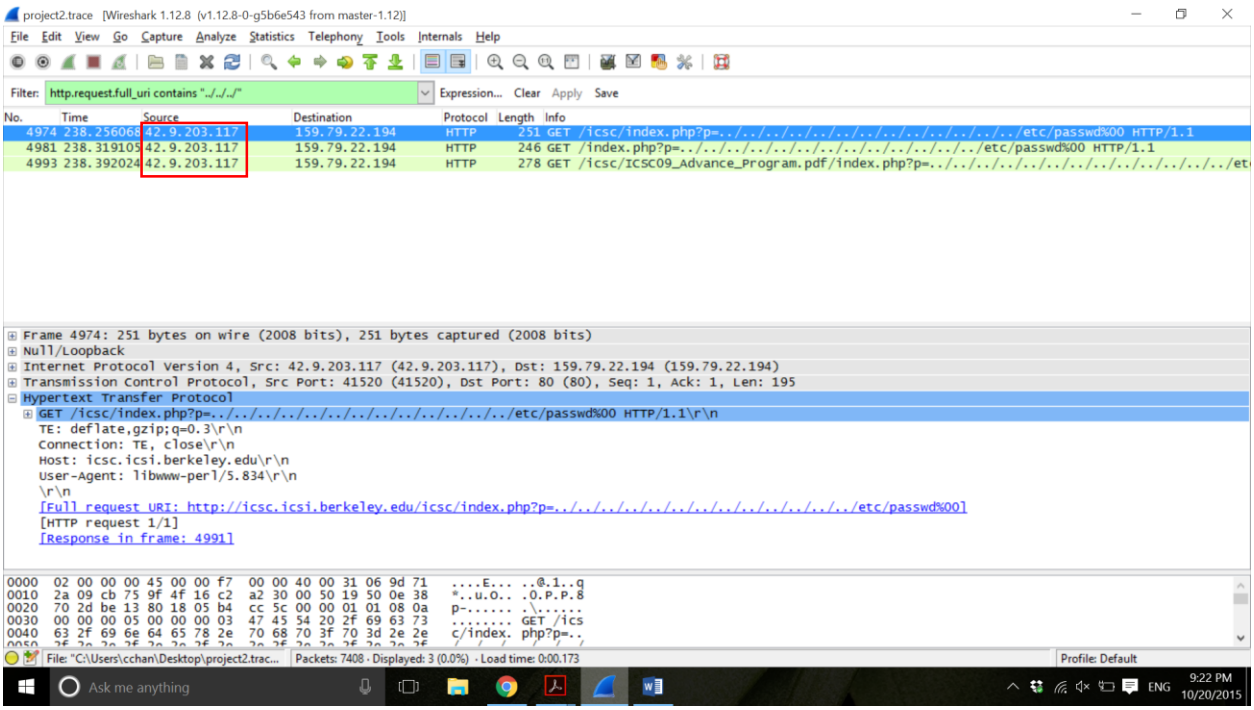



Command Prompt
source ip: 44.154.102.10 ['200']
source ip: 205.232.201.195 ['200']
C:\Users\cchan\Desktop\Python>question1.py
source ip: 155.111.186.252 ['200']
source ip: 40.187.57.142 ['302']
source ip: 93.119.134.44 ['302']
source ip: 155.231.237.70 ['301']
source ip: 159.79.22.249 ['403']
source ip: 159.70.229.173 ['200']
source ip: 35.183.215.204 ['302']
source ip: 205.232.201.218 ['200']
source ip: 93.199.112.45 ['200']
source ip: 33.247.152.113 ['301']
source ip: 142.165.192.188 ['200']
source ip: 136.93.4.213 ['200']
source ip: 143.138.66.97 ['200']
source ip: 97.145.19.119 ['200']
source ip: 44.111.85.82 ['200']
source ip: 159.79.22.198 ['404']
source ip: 205.234.49.157 ['302']
source ip: 154.87.109.177 ['200']
source ip: 44.131.48.102 ['200']
source ip: 33.247.152.101 ['200']
source ip: 159.79.22.194 ['404']
source ip: 154.87.109.40 ['200']
source ip: 248.78.109.66 ['302']
source ip: 142.165.192.177 ['302']
source ip: 34.30.235.180 ['200']
source ip: 154.27.68.55 ['200']
source ip: 111.4.186.50 ['200']
source ip: 143.138.4.147 ['200']
source ip: 37.199.226.67 ['200']
source ip: 37.120.175.85 ['304']
source ip: 205.232.203.30 ['302']
source ip: 138.59.102.27 ['200']
source ip: 143.179.11.189 ['200']
source ip: 251.235.172.148 ['200']
source ip: 44.131.51.48 ['200']
source ip: 44.131.51.161 ['200']
Ask me anything
10:40 PM
10/27/2015

2.

Filter: http.request.full_uri contains "../../../"

I look at the request contains the "../../../", we can see the IP address is the 42.9.203.117 which is trying to exploit the web server.

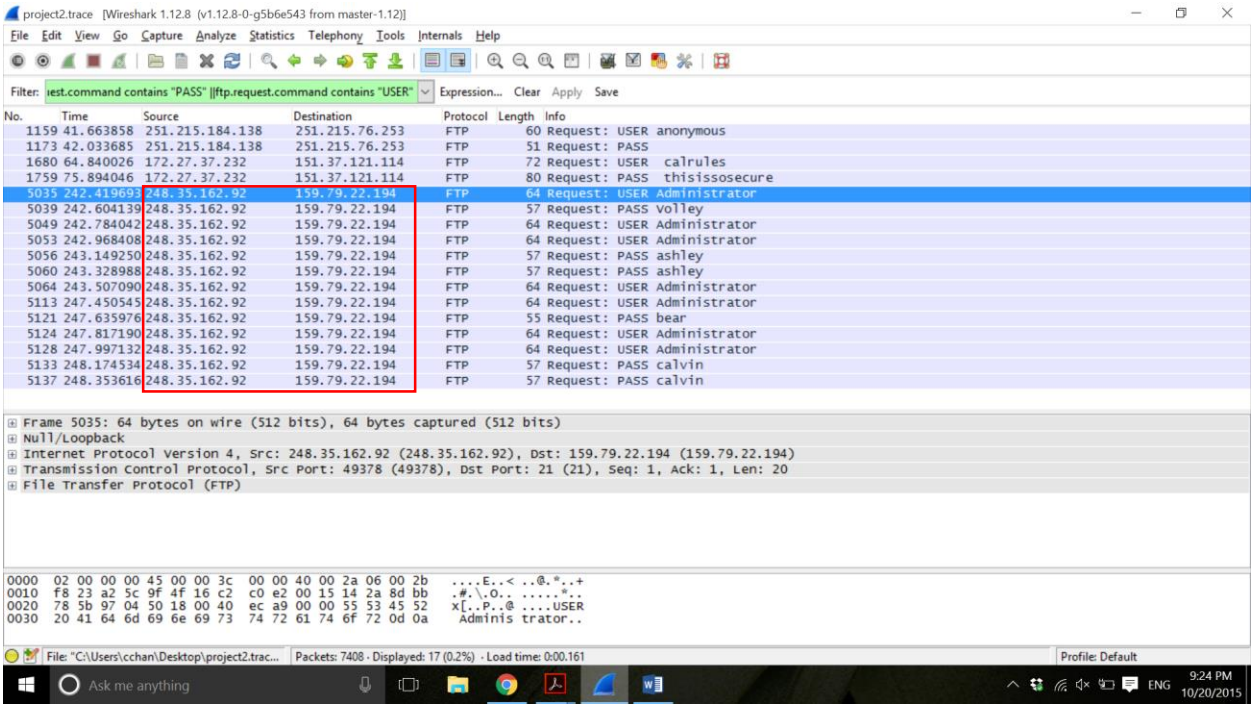


3.

Filter: ftp.request.command contains "PASS" || ftp.request.command contains "USER"

We can see that the IP address: 248.35.162.92 tried many times I assume it is the brutal force attack.

The 172.27.37.232, 251.215.184.138 also have log in record but they didn't try and try again.



4.

Filter: ftp && ip.src == 172.27.37.232 || ip.dst == 172.27.37.232

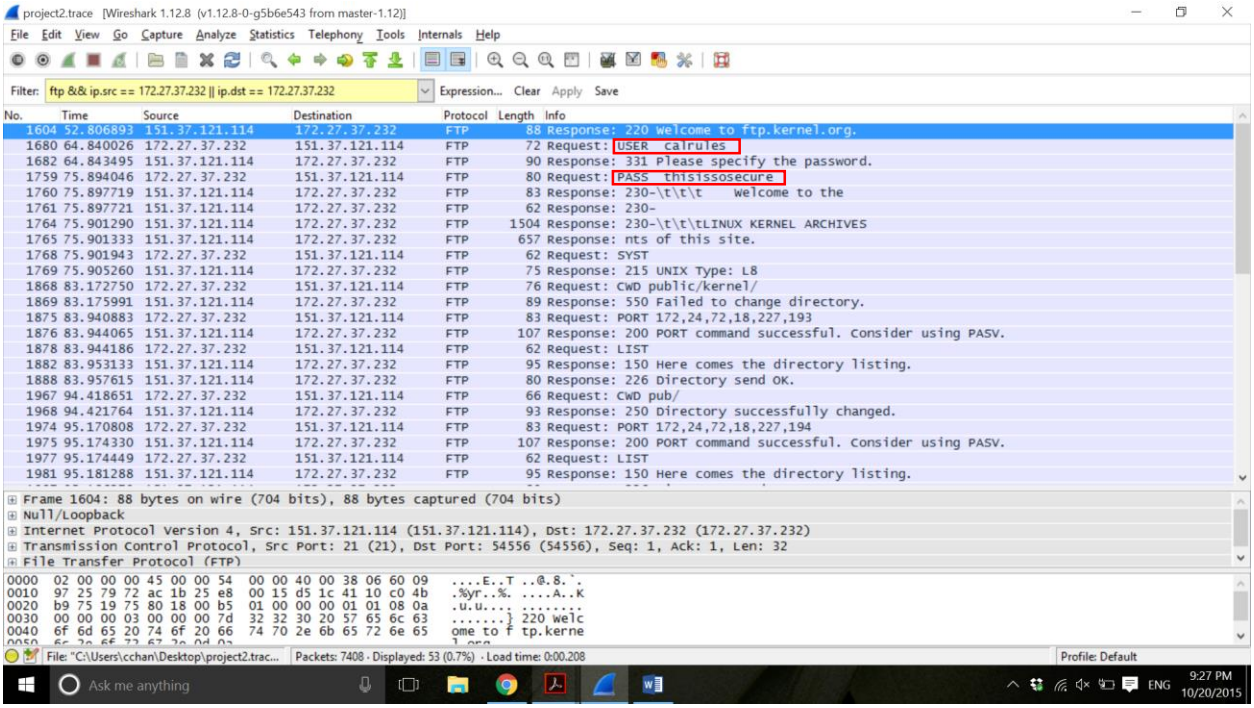
Filter: ftp && ip.src == 251.215.184.138 || ip.dst == 251.215.184.138

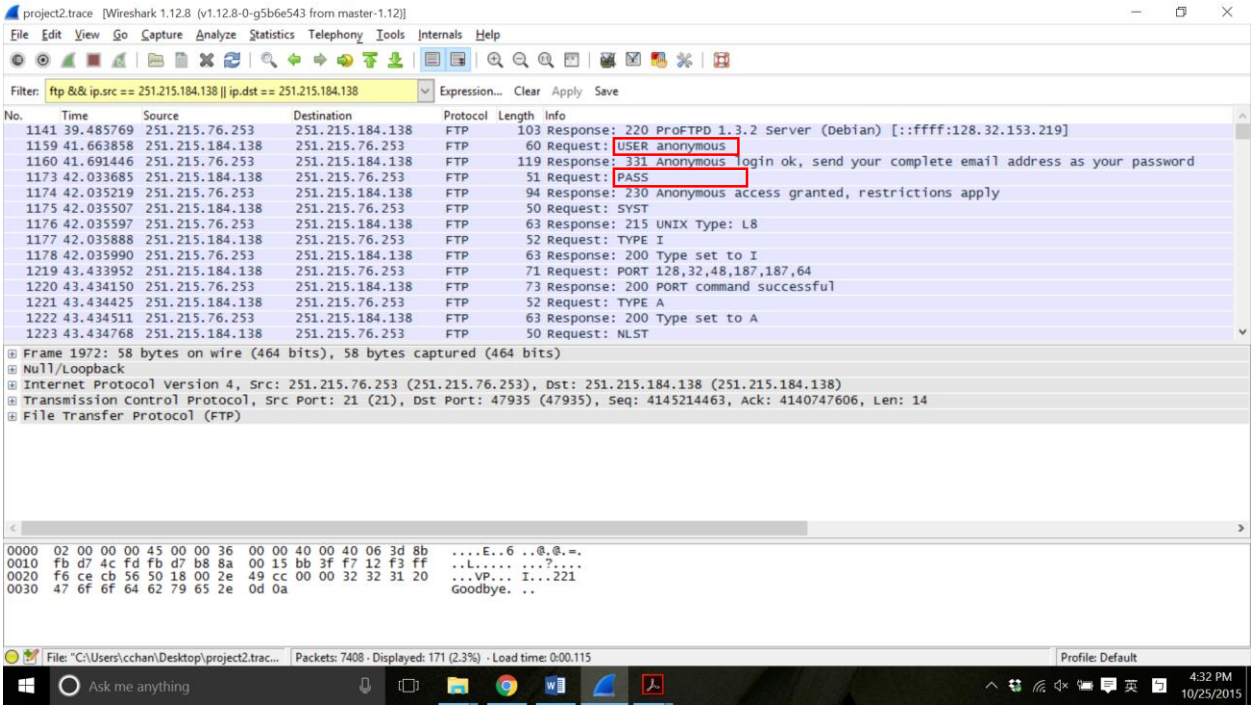
I use the last question search result to set the filter.

The user of 172.27.37.232 is “calrules” and password is “thisisosecure”.

The user of 251.215.184.138 is “anonymous” and password is “ ”(none).

When enter the other protocol, we could not find any other username and password.

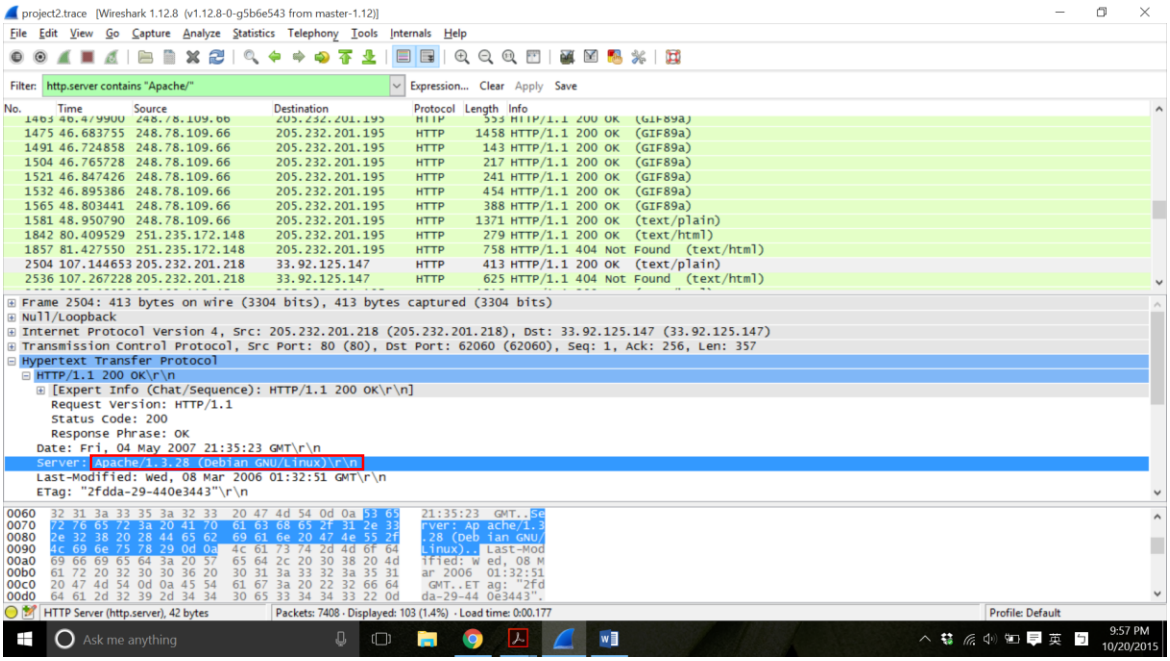


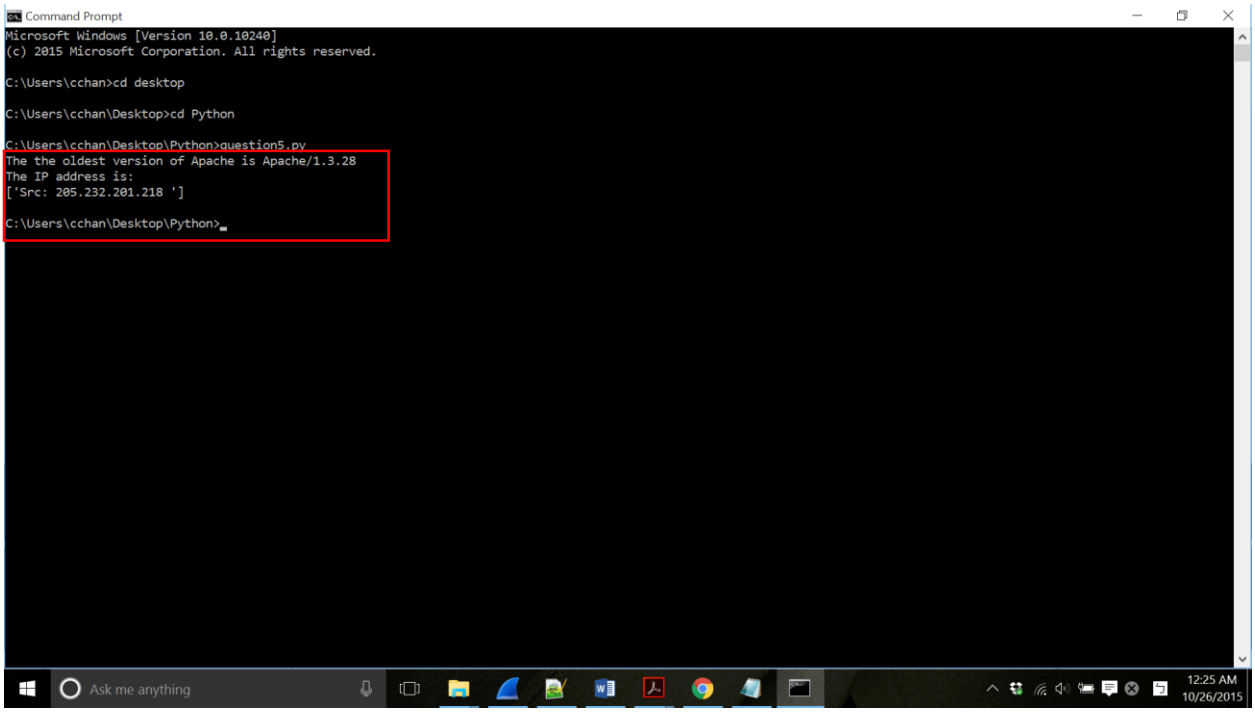

project2.trace [Wireshark 1.12.8 (v1.12.8-0-g5b6e543 from master-1.12)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: ftp && ip.src == 251.215.184.138 || ip.dst == 251.215.184.138
Expression... Clear Apply Save
No. Time Source Destination Protocol Length Info
1141 39.485769 251.215.76.253 251.215.184.138 FTP 103 Response: 220 ProFTPD 1.3.2 Server (Debian) [::ffff:128.32.153.219]
1159 41.663858 251.215.184.138 251.215.76.253 FTP 60 Request: USER anonymous
1160 41.691446 251.215.76.253 251.215.184.138 FTP 119 Response: 331 Anonymous login ok, send your complete email address as your password
1173 42.033685 251.215.184.138 251.215.76.253 FTP 51 Request: PASS
1174 42.035219 251.215.76.253 251.215.184.138 FTP 94 Response: 230 Anonymous access granted, restrictions apply
1175 42.035507 251.215.184.138 251.215.76.253 FTP 50 Request: SYST
1176 42.035597 251.215.76.253 251.215.184.138 FTP 63 Response: 215 UNIX Type: L8
1177 42.035888 251.215.184.138 251.215.76.253 FTP 52 Request: TYPE I
1178 42.035990 251.215.76.253 251.215.184.138 FTP 63 Response: 200 Type set to I
1219 43.433952 251.215.184.138 251.215.76.253 FTP 71 Request: PORT 128,32,48,187,187,64
1220 43.434150 251.215.76.253 251.215.184.138 FTP 73 Response: 200 PORT command successful
1221 43.434425 251.215.184.138 251.215.76.253 FTP 52 Request: TYPE A
1222 43.434511 251.215.76.253 251.215.184.138 FTP 63 Response: 200 Type set to A
1223 43.434768 251.215.184.138 251.215.76.253 FTP 50 Request: NLST
Frame 1972: 58 bytes on wire (464 bits), 58 bytes captured (464 bits)
Null/Loopback
Internet Protocol Version 4, Src: 251.215.76.253 (251.215.76.253), Dst: 251.215.184.138 (251.215.184.138)
Transmission Control Protocol, Src Port: 21 (21), Dst Port: 47935 (47935), Seq: 4145214463, Ack: 4140747606, Len: 14
File Transfer Protocol (FTP)
0000 02 00 00 00 45 00 00 36 00 00 40 00 40 06 3d 8b ....E..6 ..@.@.=.
0010 fb d7 4c fd fb d7 b8 8a 00 15 bb 3f f7 12 f3 ff ..L..... ...?....
0020 f6 ce cb 56 50 18 00 2e 49 cc 00 00 32 32 31 20 ...VP... I...221
0030 47 6f 6f 64 62 79 65 2e 0d 0a Goodbye. ..
File: "C:\Users\cchan\Desktop\project2.trac...
Packets: 7408 · Displayed: 171 (2.3%) · Load time: 0:00.115
Profile: Default
Ask me anything
4:32 PM
10/25/2015

5.

Filter: http.server contains "Apache/"

After enter the filter, I export the information into a plain txt file (q5.txt) and use a python script to print out all the frame and sever type. Then I sort the server type. The 1.3.28 is the oldest. We could see there are 4 different frame used the 1.3.28 server. They all are the same IP address: 205.232.201.218. I use the Python script (question5.py) to search. I first put the server version into the dictionary as a key and then put the IP address as the value. On the other hand, I stored the server version into an array, then sort it. Use the minimum number as the key to print out its source IP.

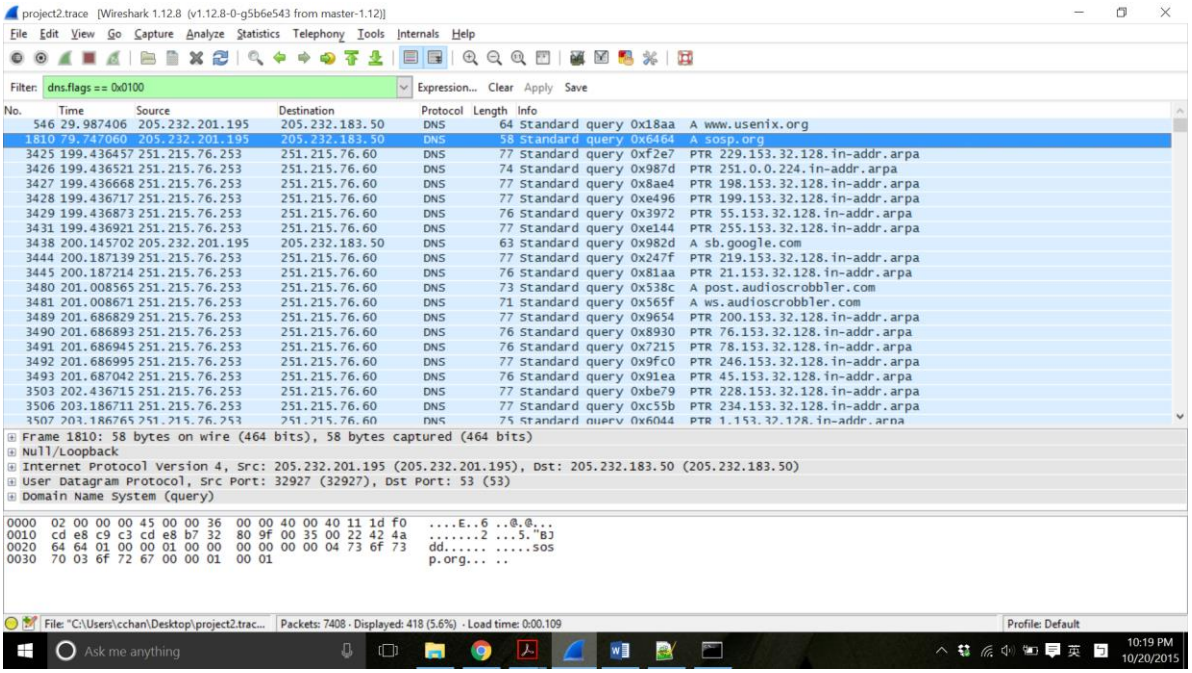



Command Prompt
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.
C:\Users\cchan>cd desktop
C:\Users\cchan\Desktop>cd Python
C:\Users\cchan\Desktop\Python>question5.py
The the oldest version of Apache is Apache/1.3.28
The IP address is:
['Src: 205.232.201.218 ']
C:\Users\cchan\Desktop\Python>
Ask me anything
12:25 AM
10/26/2015

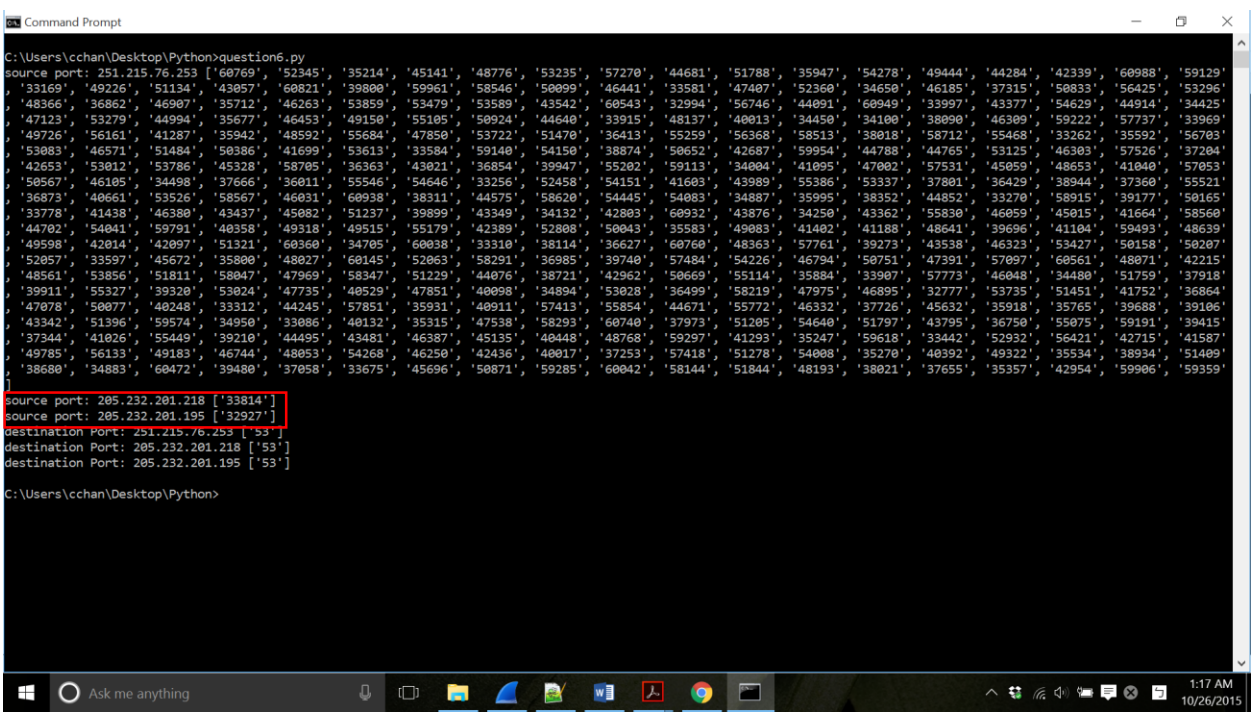
6.

Filter: dns.flags == 0x0100

I filter the packets using dns flags 0x0100. We can see all the request queries as shown below:



Then I used a Python script (question6.py) to map the different source ports used to their source IP address in the export txt file (q6.txt). We can see the first IP has many destination IP. And the 205.232.201.218 and 205.232.201.195 only have one connection.



7.

Filter: tcp.flags == 0x02

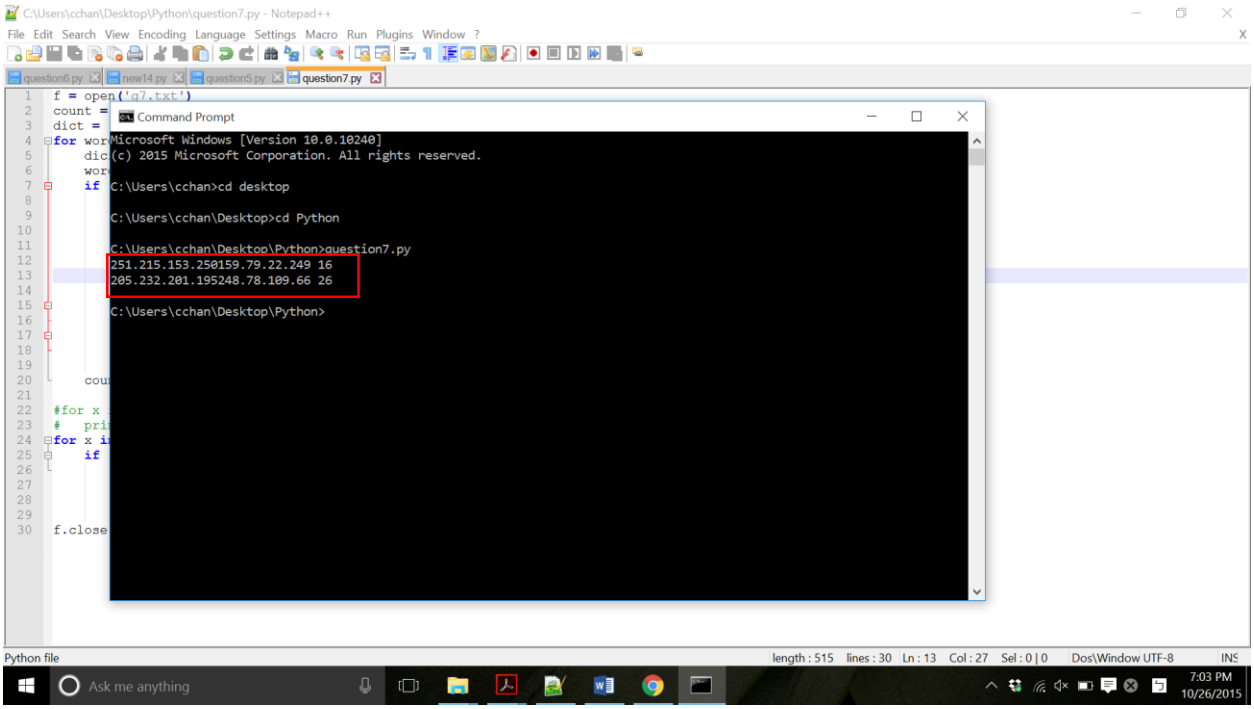
After using the script (question7.py) to search the IP endpoint in q7.txt more than 5 times, I examine the seq number to see which IP endpoints connection provide the broadest 32-bit coverage in their ISNs.

Filter: tcp.flags == 0x02 && ip.src == 205.232.201.195 && ip.dst == 248.78.109.66

205.232.201.195 & 248.78.109.66 with a range 3,152,737,232 – 3,122,016,130= 30,721,102

Filter: tcp.flags == 0x02 && ip.src ==251.215.153.250 && ip.dst == 159.79.22.249

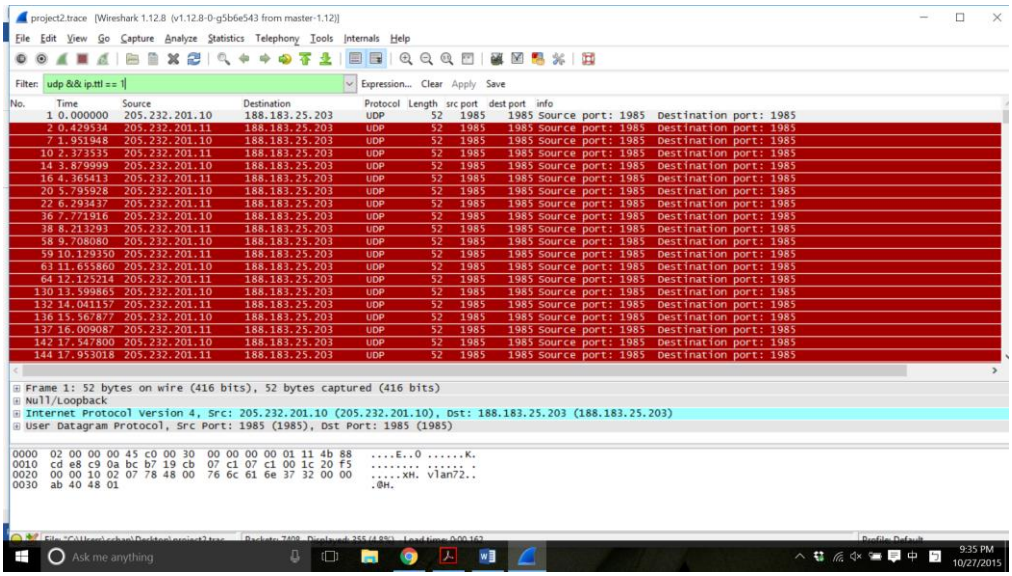
251.215.153.250 & 159.79.22.249 range from 3,783,016,653 to 42,319,747. The range is 3,740,696,906. Obviously, the range of 251.215.153.250 & 159.79.22.249 is bigger. That's what we need.



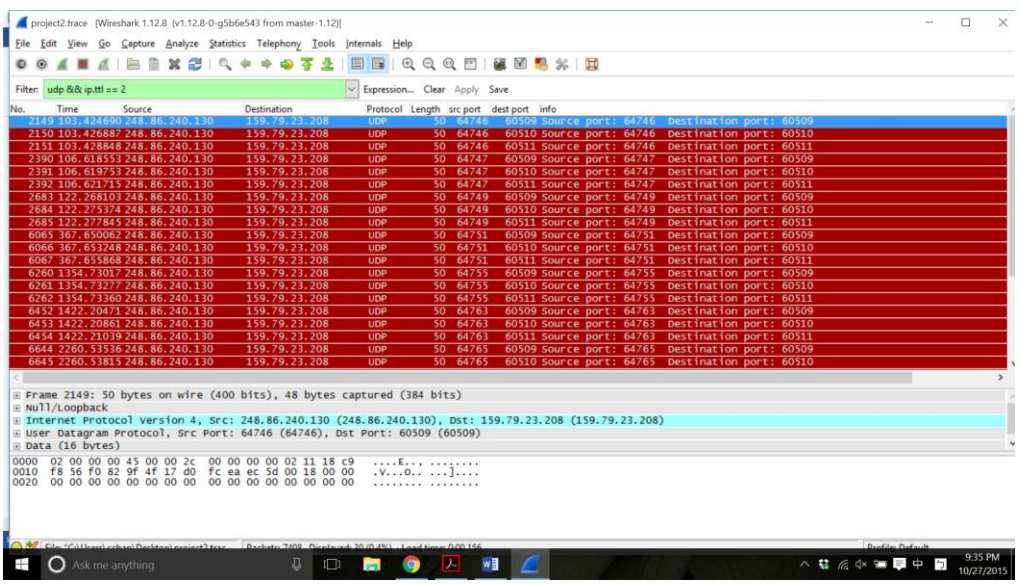
8.

I tried to look for a host performing trace route. The trace route scanning will send UDP packets with incrementing ttl and receives ICMP packets in return. So I enter the filter shown as following.

Filter: udp && ip.ttl == 1

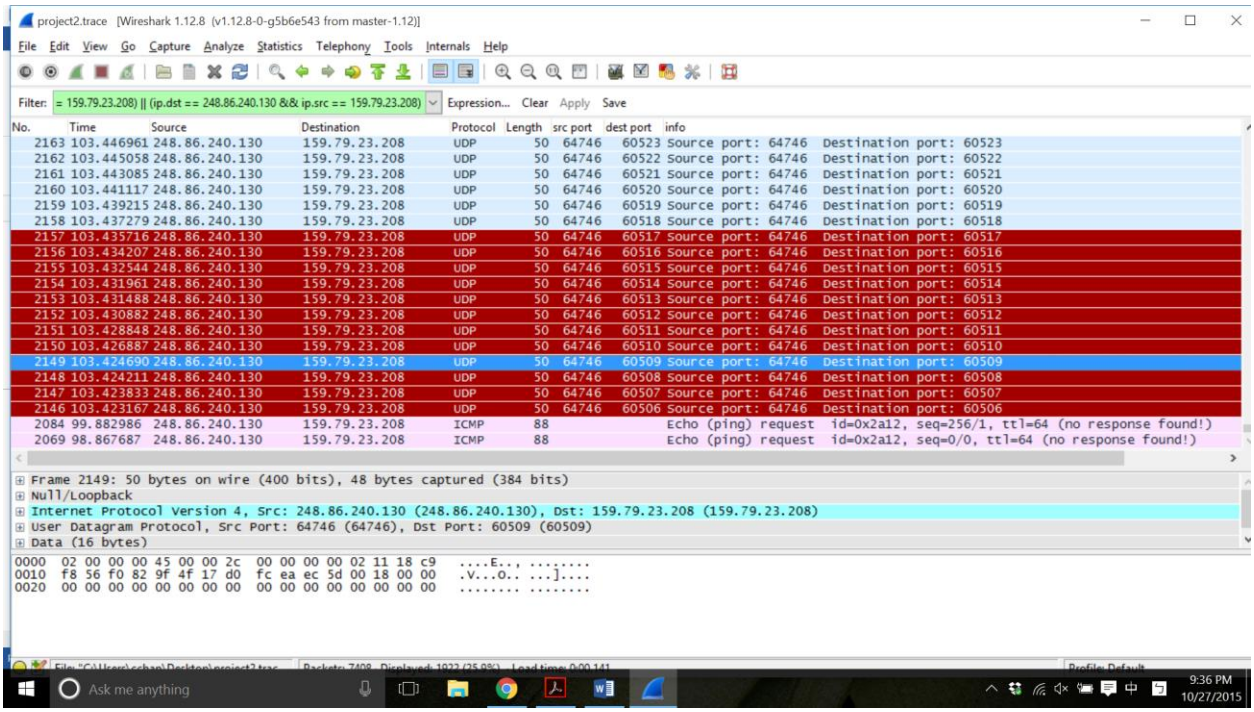


Filter: udp && ip.ttl == 2



We could see that IP: 248.86.240.130 and 159.79.23.208 are running traceroute for detecting routers on a path.

Filter: (ip.src == 248.86.240.130 && ip.dst == 159.79.23.208) || (ip.dst == 248.86.240.130 && ip.src == 159.79.23.208)

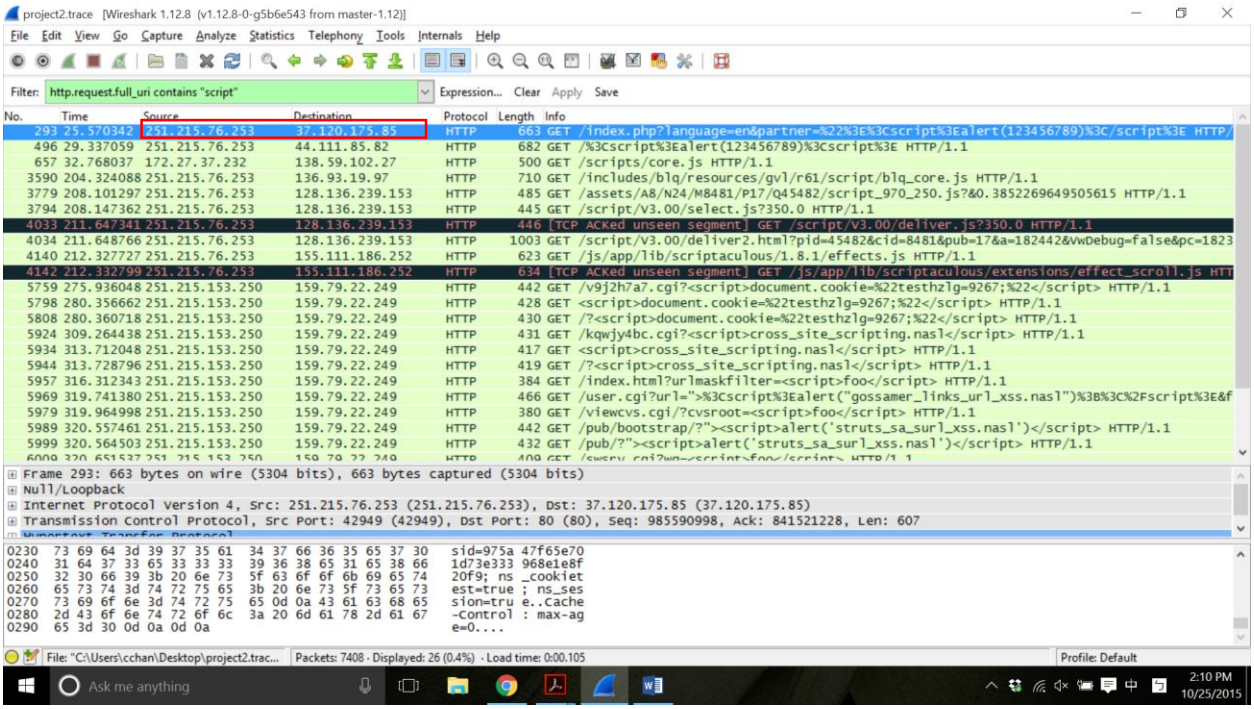


I find a host that incrementing its ttl: source IP 248.86.240.130 and destination IP 159.79.23.208.

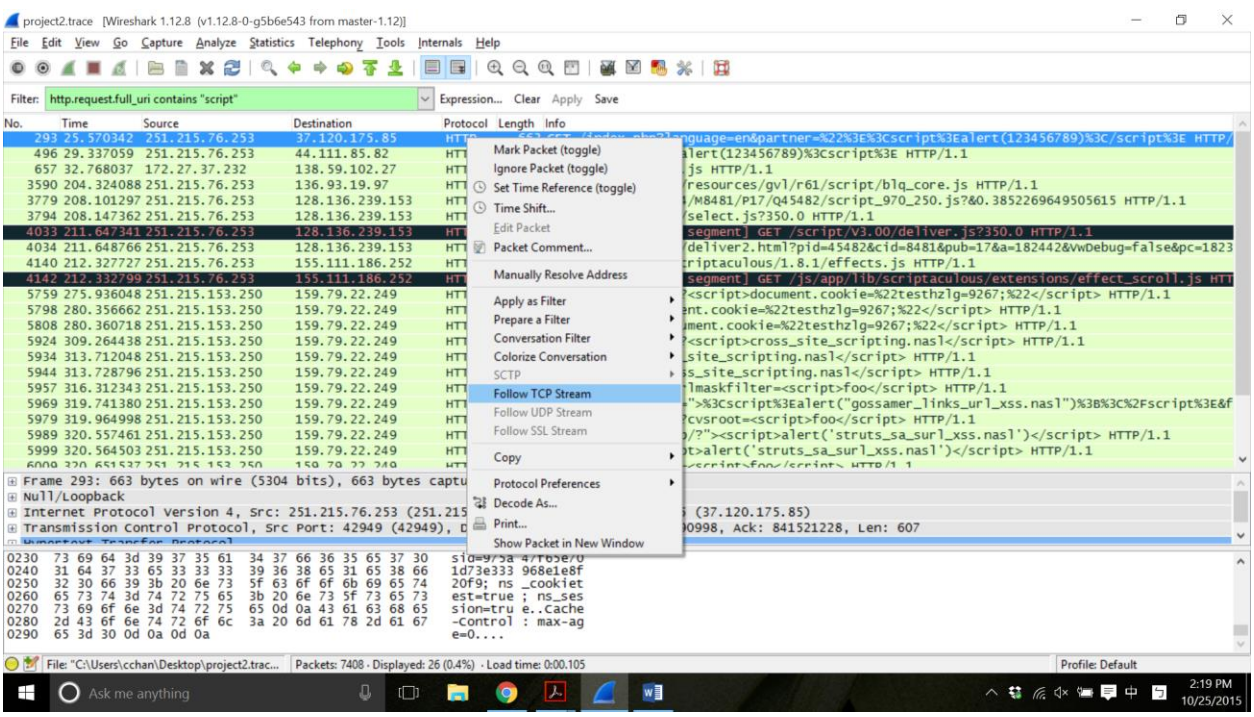
9.

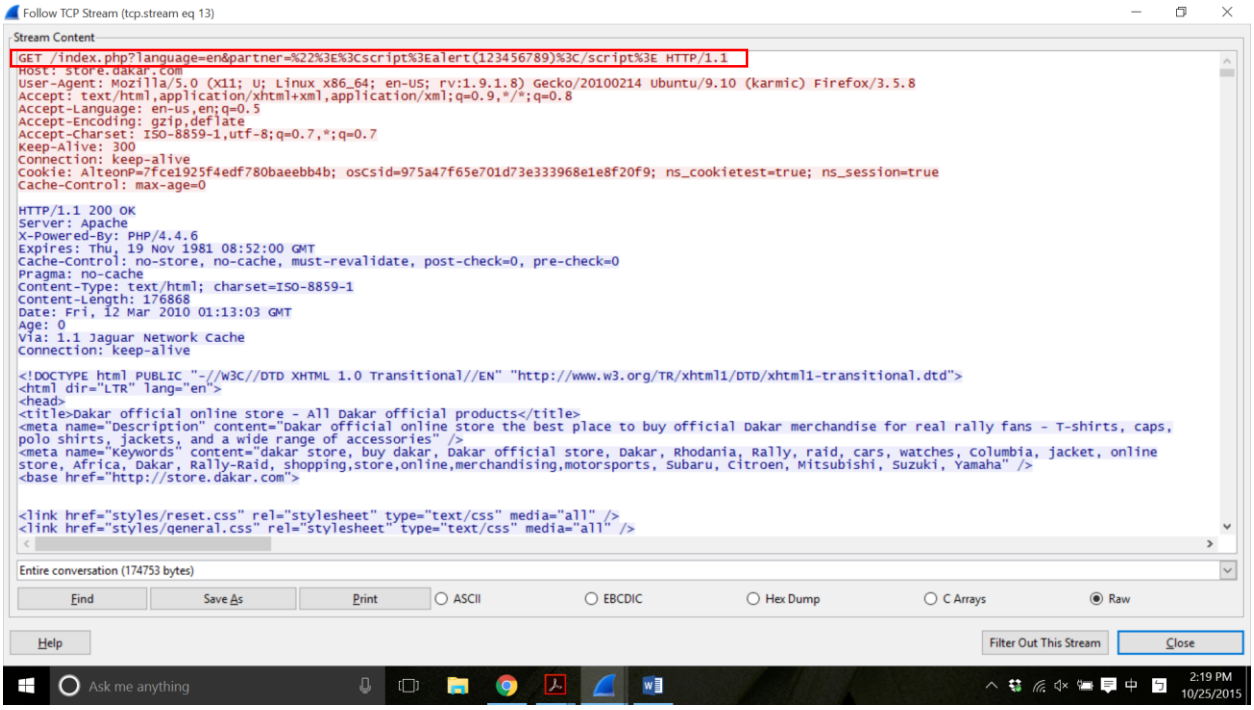
I use the filter to search the URL that contains a script inside the URL itself.
Filter: http.request.full_uri contains "script"

IP: dst=37.120.175.85 (Vulnerable Server)

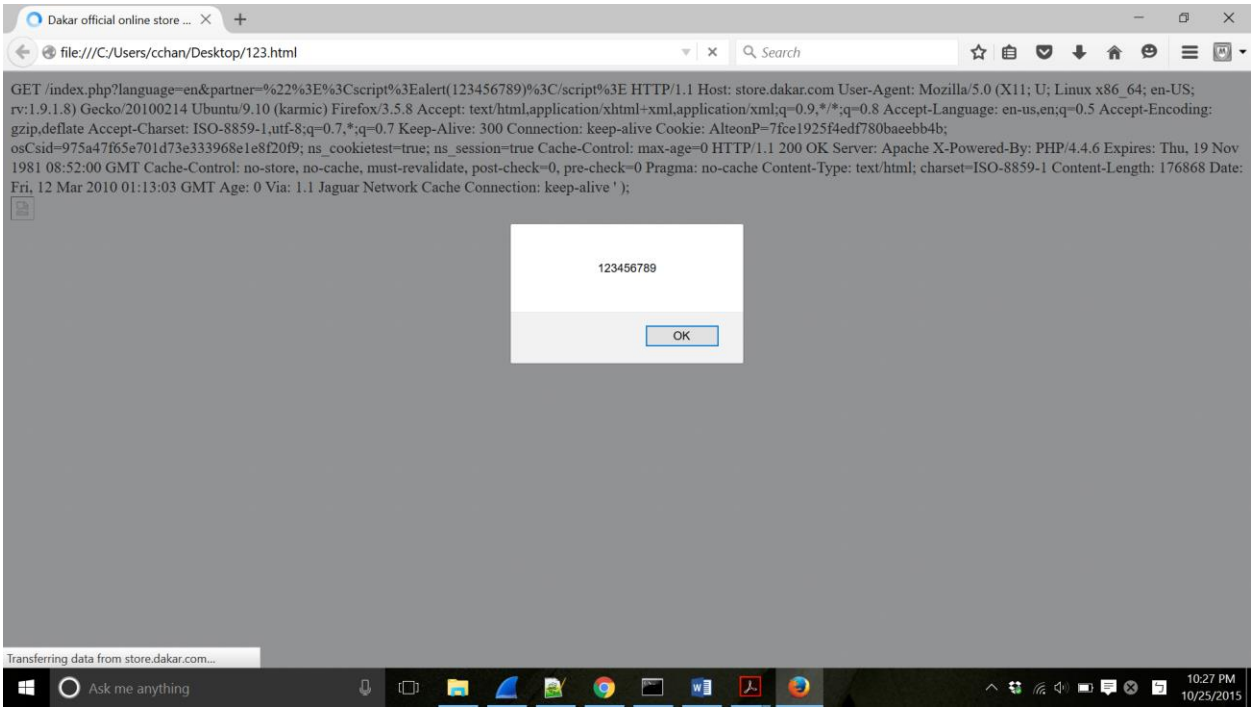


And then follow the TCP stream to get the response.





I save it as an html file. Here is the executing situation. That alert “123456789” is reflected XSS.



The real world website already fixed this situation.