



VIT[®]
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

TEAM MEMBERS

ABHISHEK KUMAR-18BIT0348(abhishek.kumar2018e@vitstudent.ac.in)
DEVANSHU RANJAN-18BIT0353(devanshu.ranjan2018@vitstudent.ac.in)
SUDHANSHU BHATIA-18BIT0391(sudhanshu.bhatia2018@vitstudent.ac.in)

FACULTY

Dr.Jeyanthi N
Associate Professor Sr.
School of Information Technology and Engineering
Vellore Institute of Technology
Vellore, India

DETECTION OF DDOS ATTACK USING WIRESHARK IN REAL TIME

ABSTRACT

One of the biggest problems of today's internet technologies is cyber attacks. In this paper whether DDoS attacks will be determined by deep packet inspection. Initially packets are captured by listening of network traffic. Packet filtering was achieved at desired number and type. These packets are recorded to database to be analyzed, daily values and average values are compared by known attack patterns and will be determined whether a DDoS attack attempts in real time systems.

Now a days many people are using Machine Learning Technique to determine whether there System is under attack or not.They are using WEKA one of the tool which is used for classification of different types of attack.

In existing world we are going to use an important tool which really makes our work faster and efficient in every aspect.Earlier we have to gather lots of ton of Datasets for visualizing attacks done on our Network ,Now we have a important tools called Wireshark which can

easily visualize the no of requests made on our network and can easily make us understand that we are under attack or not.

So in this report i will brief my work,basically we have done lots of research on Wireshark tool how its capture traffic,how we can count no of packets , how we can compare requests before and after Ddos attack.

So we have used Grabify software to trace someone IP address with the help of dummy web Link and later on with 4 lines of ping command we will do DDOS attack with our own cmd(10-12).Before doing this we will note the count of the packets on that particular network on which ddos attack have been done and compares with that data after DDOS attack.

KEYWORD

Technical Jargons related to our work are:-

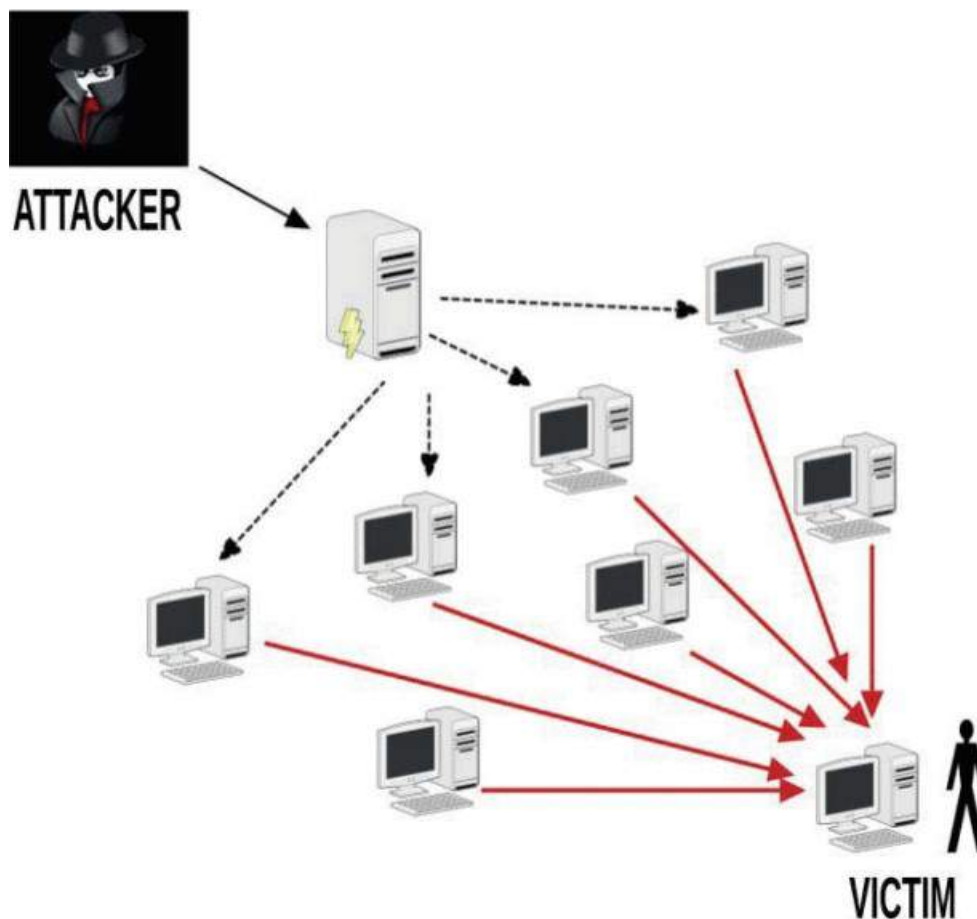
- 1.packet filtering,**
- 2.DDoS attack,**
- 3.deep packet analysis,**
- 4.cyber attacks,**
- 5.deep packet inspection,**
- 6.real time systems,**
- 7.Internet technologies**

INTRODUCTION

Through monitoring network traffic, many packets are found. Today, packets are captured mostly with the help of packet sniffers. Packet capturers gather data from the network. Packet sniffers, on the other hand, are known as the most efficient data gathering method . Packet sniffers make use of pcap library so as to capture packets. The version of Pcap library for windows is winpcap while libpcap library is used in Linux systems.Sudhanshu Bhatia employed wireshark program and conducted packet analyses.Devanshu Ranjan, in the same manner, ran wireshark program.

LITERATURE SURVEY:

DDoS attacks aims to cease the service operating on the target system. Web servers become unable to provide http service and e-mail servers can no longer send or receive emails. These attacks occur by exhausting memory, processor and band width sources or taking advantage of a weakness in the service. DDoS attacks may take place in the form of unexpectedly low network performance, slowness in access to web sites, cuts in networks connections, increase in the number of spam e-mails or inability to access certain parts of a web site. In DDoS attacks, attacker is disguised and thousands, even hundreds of thousands of fake IP addresses are created. Thus, such attacks are quite difficult to prevent. A sample attack type is given below:



Author	Title	Findings
S Sandhya, Sohini Purkayastha, Emil Joshua,Akash Deep	Assessment of website security by penetration testing using Wireshark	Wireshark tool enables the ethical hacker to reveal the flaws in the system security at the user authentication level. This approach of identifying vulnerabilities is deemed fit as the strategy involved in this

		testing is rapid and provides good success in identifying vulnerabilities.
Shaoqiang Wang, DongSheng Xu, ShiLiang Yan	Analysis and application of Wireshark in TCP/IP protocol teaching	It introduces the functions and characteristics of Wireshark, and expounds its analysis and application in TCP/IP protocol teaching by some specific examples.
Apri Siswanto, Abdul Syukur, Evizal Abdul Kadir, Suratin	Network Traffic Monitoring and Analysis Using Packet Sniffer	This study aims to obtain data about the results of traffic in a graphical form so that it can find out the number of users who access the internet and use bandwidth.
Sudhakar, R. K. Aggarwal	A survey on comparative analysis of tools for the detection of ARP poisoning	Different dangers like sniffing, phishing, caricaturing exist. This paper exhibits an outline of different system dangers and assaults to the system. There are so many apparatuses in the amenable source and many mitigation techniques which are produced as a way to tackle to these assaults. Devices like Wireshark, arpswatch, firewall has been talked about in this paper.
S. Pavithirakini, Senevirathna Bandara, D. Dhammearatchi	Improve the Capabilities of Wireshark as a tool for Intrusion Detection in DOS Attacks	Flooding is a kind of attack, in which the attacker sends several floods of packets to the victim or associated service in an effort to bring down the system. There are unlike types of flooding attacks like ping flood, Syn floods, UDP (User Datagram Protocols) floods etc.
JisaDavid CizaThomas	Efficient DDoS flood attack detection using dynamic thresholding on flow-based network traffic	The interconnected computer systems and networks are vulnerable to very large number of attacks; Distributed Denial of Service being a major one. This paper analyses the features of network traffic and the existing algorithms to detect Distributed Denial of Service attacks and proposes an efficient statistical approach to detect the attacks based on traffic features and dynamic threshold detection algorithm.
Muhammad Ahmad, Kareem Ullah	SNORT IMPLEMENTATION WITH WIRESHARK HELPING AVOIDING DDOS	Cloud computing provide different resources on demand where the users can access these

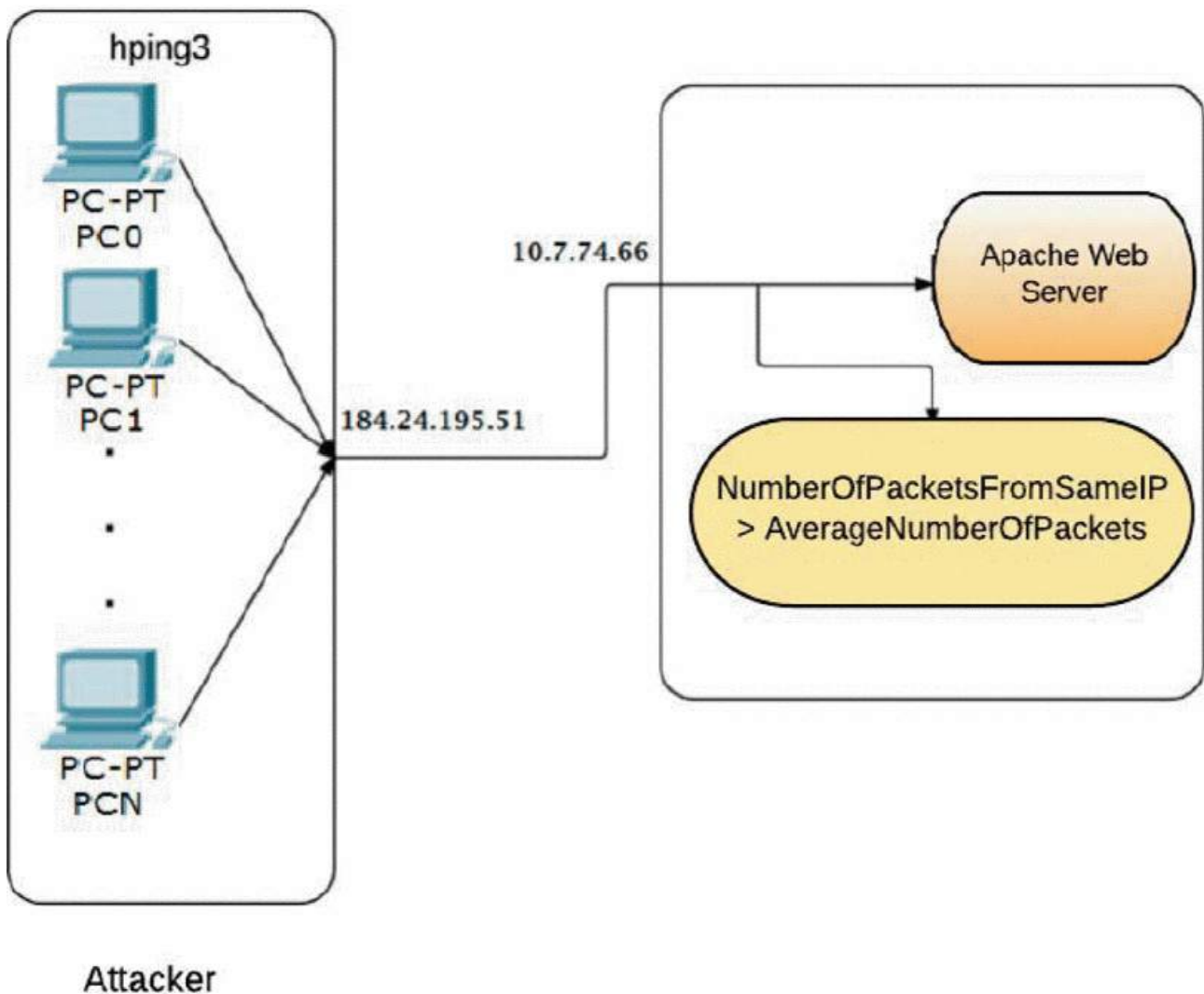
	ATTACK IN THE CLOUD COMPUTING	resources through the internet easily. Due to distributed nature of cloud computing technology remained untouched from the attackers. Attackers can attack easily on cloud computing and decrease the effectiveness and resources availability to the users in the cloud. There exist many cybercrime attacks can easily hit the security of the cloud computing
Dmitri Bekerman, Bracha Shapira, Lior Rokach, Ariel Bar	Unknown malware detection using network traffic classification	Detecting malware by analyzing network traffic. The proposed method extracts 972 behavioral features across different protocols and network layers, and refers to different observation resolutions (transaction, session, flow and conversation windows). A feature selection method is then used to identify the most meaningful features and to reduce the data dimensionality to a tractable size.
Mehedee Zaman, Tazrian Siddiqui, Mohammad Rakib Amin, Md. Shohrab Hossain	Malware detection in Android by network traffic analysis	A common behavior of mobile malware is transferring sensitive information of the cell phone user to malicious remote servers. In this paper, we describe and demonstrate in full detail, a method for detecting malware based on this behavior.

Sudhanshu detected malicious flooding traffic with the help of Wireshark tool. The system they employed includes asymmetric demonstration of packets. In this system, if flow rate is higher than normal band width, the attack in the system is detected. It is checked if there are too many packets in the line. Devanshu Ranjan observed arrivals of packets and found solutions with behaviors on network traffic. This technique makes it possible to differentiate between DDoS attack sources and real users. It can be detected whether there is an attack or not with the aid of rate of packet arrival. I Abhishek Kumar detected the presence of a possible DDoS attack by using Statistics of Wireshark. Multiple DDoS attackers can be tracked simultaneously thanks to this method. Sudhanshu detected DDoS attacks by using projected UDP technique.

The main difference between the study introduced in this article and others in the literature is that a user is able to determine the average number of packets to be

carried by its own server system and is able to perform real time detection of a possible attack if the number of packets from the same IP is higher than that.

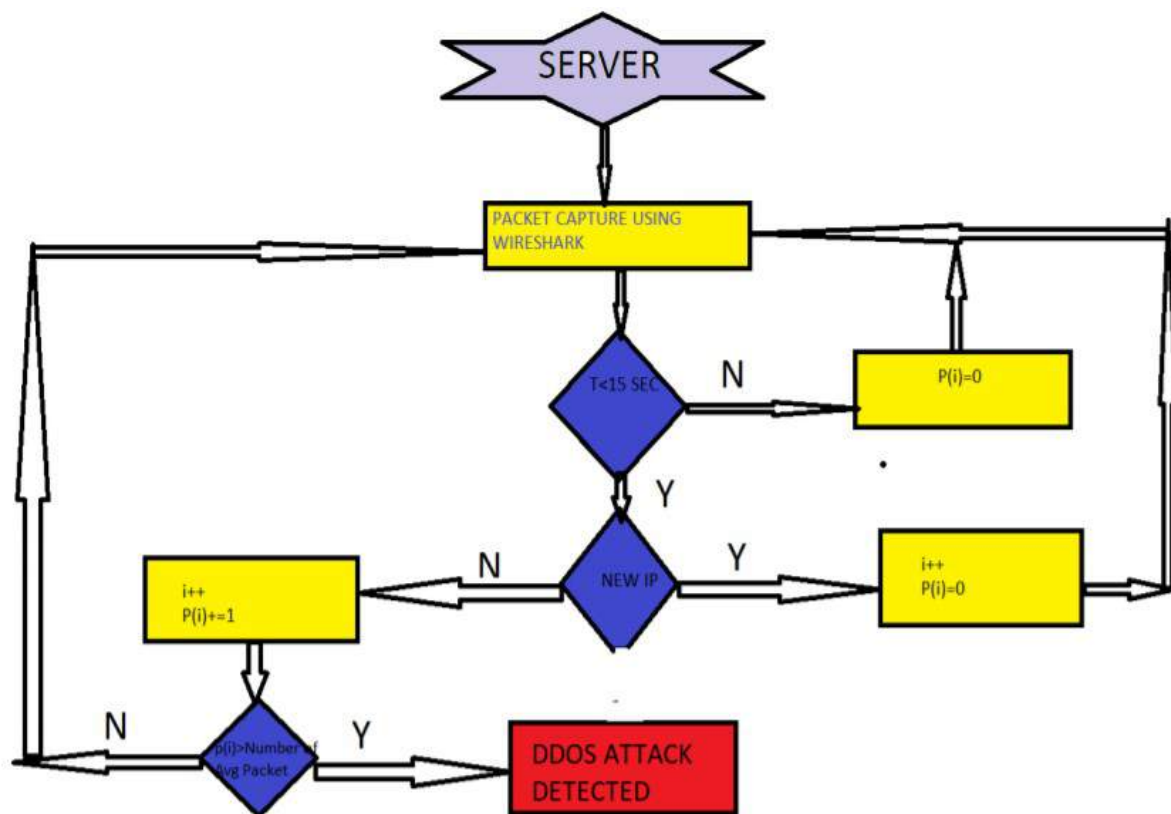
The topology of the DDoS application developed in this study . 'hping3', one of the DDoS attack tools, was used so as to generate packets. Hping3 is a tool used to generate tcp/ip packets of desired kind.



OUR PROPOSAL

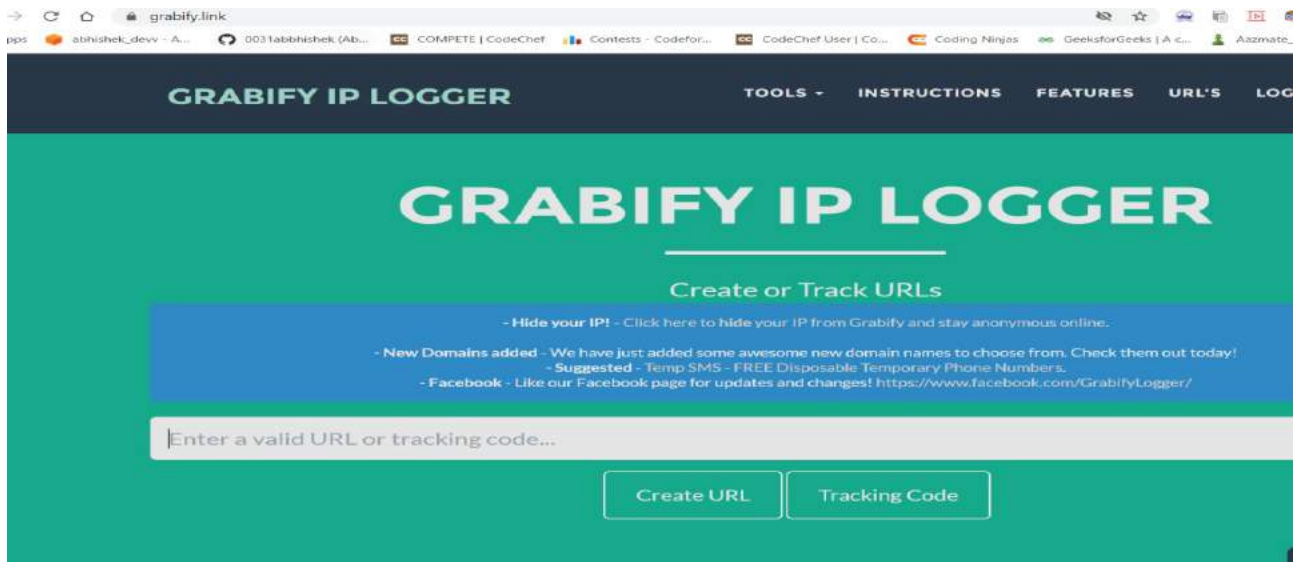
Packets are recorded instantly. In the application, the average packet number is monitored and that is how attacks are detected. Average packet number is calculated through the data from NumberOfPacket.txt file. Packets captured during attack detection are compared with the previous packet. A counter was assigned for the system. In DoS attack detection, special attention was paid on whether source address (From) and target address (To) were the same or not. In cases when two addresses were the same, the counter was increased one by one. When the counter was higher than the average, the system was halted and the statement "DoS attack was detected" was shown on the system. In DDoS attack detection, it was

especially observed if the source address (From) and target address (To) were the same or different. In cases when these addresses were identical, the counter was increased one by one. The system was halted in 2 seconds providing that the counter number was higher than the number in NumberOfPacket.txt data and statement “DDoS attack was detected” was displayed on the system. Packet number value was set as 100. This value, in fact, must be calculated via observing previous records.

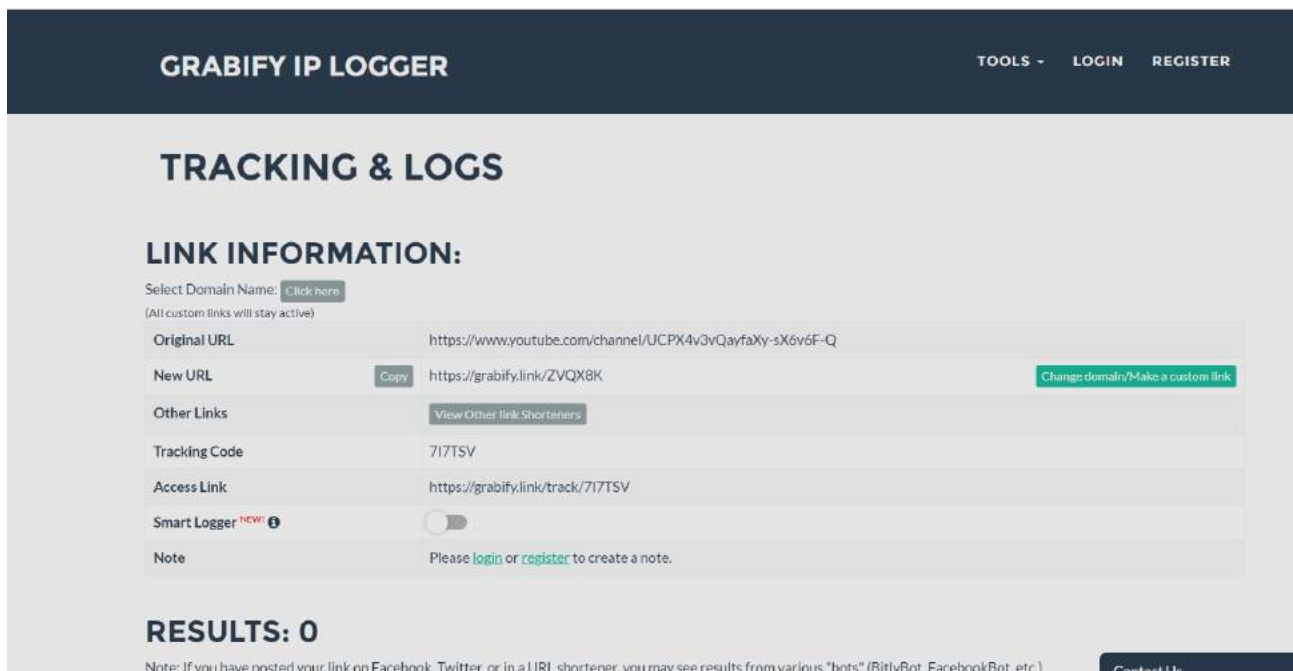


So now we all see a dry run how we all going to do this through a digram.

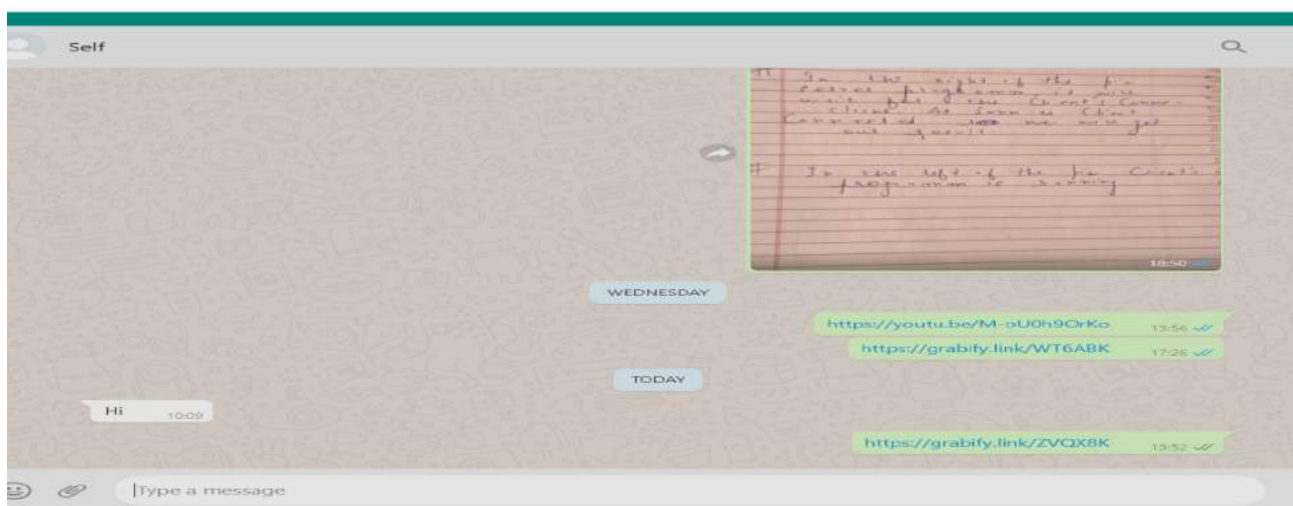
Step 1- First we will trace Some one Ip address with the help of application(Grabify Ip Logger).



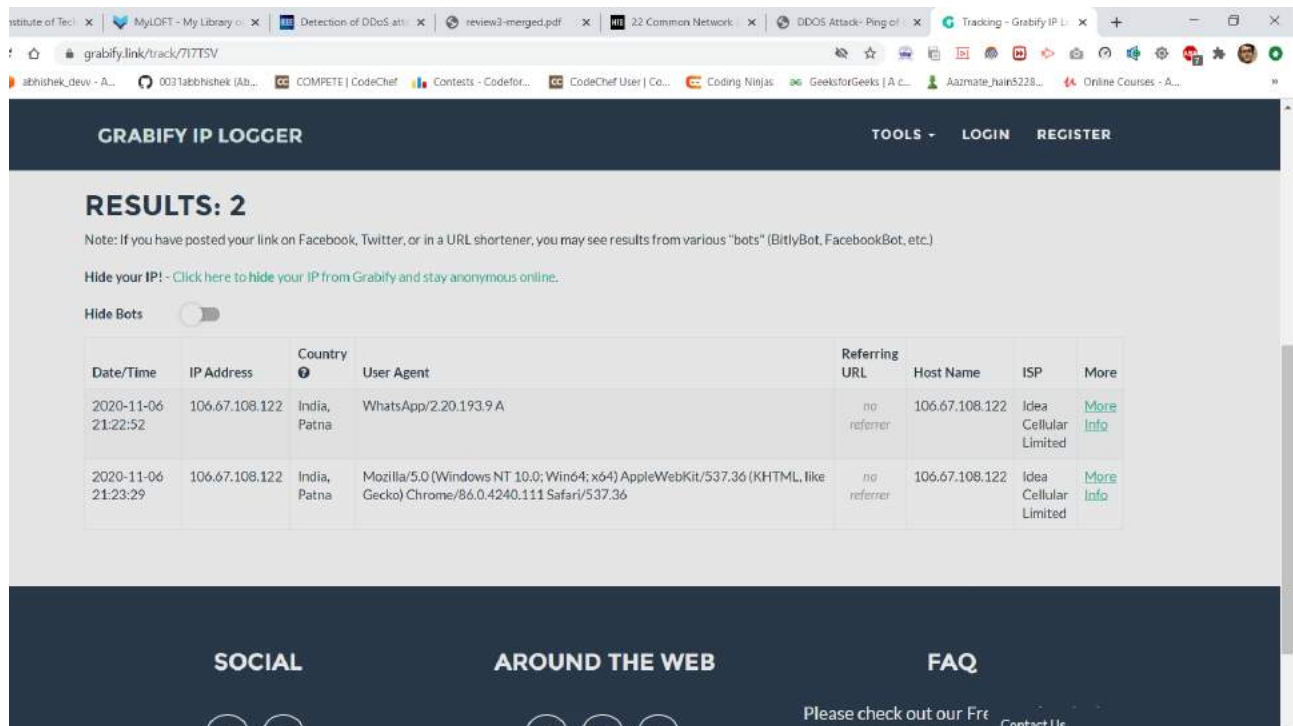
Creates a dummy weblink and send it through any messaging media and if he/she opens it then ultimately i will get there Ip address with there Location.



Copy this Link and will send it through whatsapp.



Suppose we want to do ddos attack on this guy.



The screenshot shows the 'GRABIFY IP LOGGER' website. The main heading is 'RESULTS: 2'. Below it, a note states: 'Note: If you have posted your link on Facebook, Twitter, or in a URL shortener, you may see results from various "bots" (BitlyBot, FacebookBot, etc.)'. A link is provided to 'Hide your IP! - Click here to hide your IP from Grabify and stay anonymous online.' There is a 'Hide Bots' toggle switch. A table displays the tracking results:

Date/Time	IP Address	Country	User Agent	Referring URL	Host Name	ISP	More
2020-11-06 21:22:52	106.67.108.122	India, Patna	WhatsApp/2.20.193.9 A	no referrer	106.67.108.122	Idea Cellular Limited	More Info
2020-11-06 21:23:29	106.67.108.122	India, Patna	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0.4240.111 Safari/537.36	no referrer	106.67.108.122	Idea Cellular Limited	More Info

The footer contains links for 'SOCIAL', 'AROUND THE WEB', and 'FAQ'. A footer message says 'Please check out our Free Contact Us'.

So here i got his Ip address now i can easily do ddos attack by pinging his Ip address.

Step-2- Then we will ping that Ip address with the Known Command using Cmd.

PC > Documents

```
Name      Date modified  Type  Size
pod - Notepad
File Edit Format View Help
:loop
ping 103.133.121.28 -l 800000 -w 1 -n 1
goto:loop
```

The screenshot shows a Windows desktop with several open Command Prompt windows. The main window in the foreground displays the usage and options for the 'ping' command. Other windows in the background show the execution of the 'tracert' command, displaying the path from the user's machine to the destination IP address (106.67.100.122).

Command Prompt 1 (Foreground):

```
C:\WINDOWS\system32\cmd.exe

C:\Users\Vivek kumar\Documents>goto:loop

C:\Users>C:\WINDOWS\system32\cmd.exe
Bad option -l.

Usage: ping [-t] [-a] [-n count] [-l size] [-f] [-i TTL] [-v TOS]
           [-r count] [-s count] [-j host-list] [-k host-list]
           [-w timeout] [-R] [-S srcaddr] [-c compartment] [-p]
           [-4] [-6] target_name

Options:
-t          Ping the specified host until stopped.
-a          To see statistics and continue - type Control-Break;
           To stop - type Control-C;
           Resolve addresses to hostnames.
-n count    Number of echo requests to send.
-l size     Send buffer size.
-f          Set Don't Fragment flag in packet (IPv4-only).
-i TTL      Time to Live.
-v TOS      Type of Service (IPv4-only. This setting has been deprecated
           and has no effect on the type of service field in the IP
           Header).
-r count    Record route for count hops (IPv4-only).
-s count    Timestamp for count hops (IPv4-only).
-j host-list Loose source route along host-list (IPv4-only).
-k host-list Strict source route along host-list (IPv4-only).
-w timeout  Timeout in milliseconds to wait for each reply.
-R          Use routing header to test reverse route also (IPv6-only).
           Per RFC 5895 the use of this routing header has been
           deprecated. Some systems may drop echo requests if
           this header is used.
-S srcaddr  Source address to use.
```

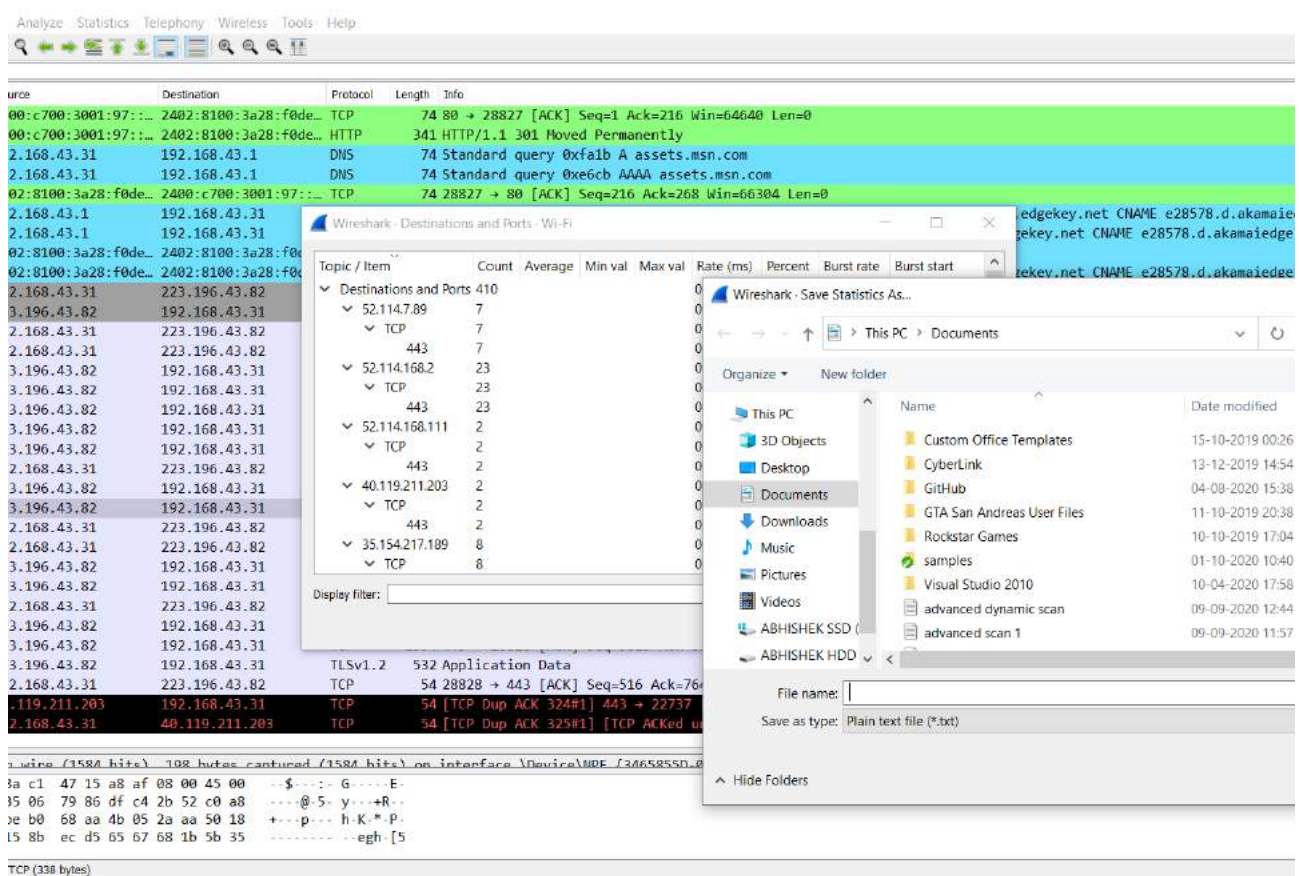
Command Prompt 2 (Background):

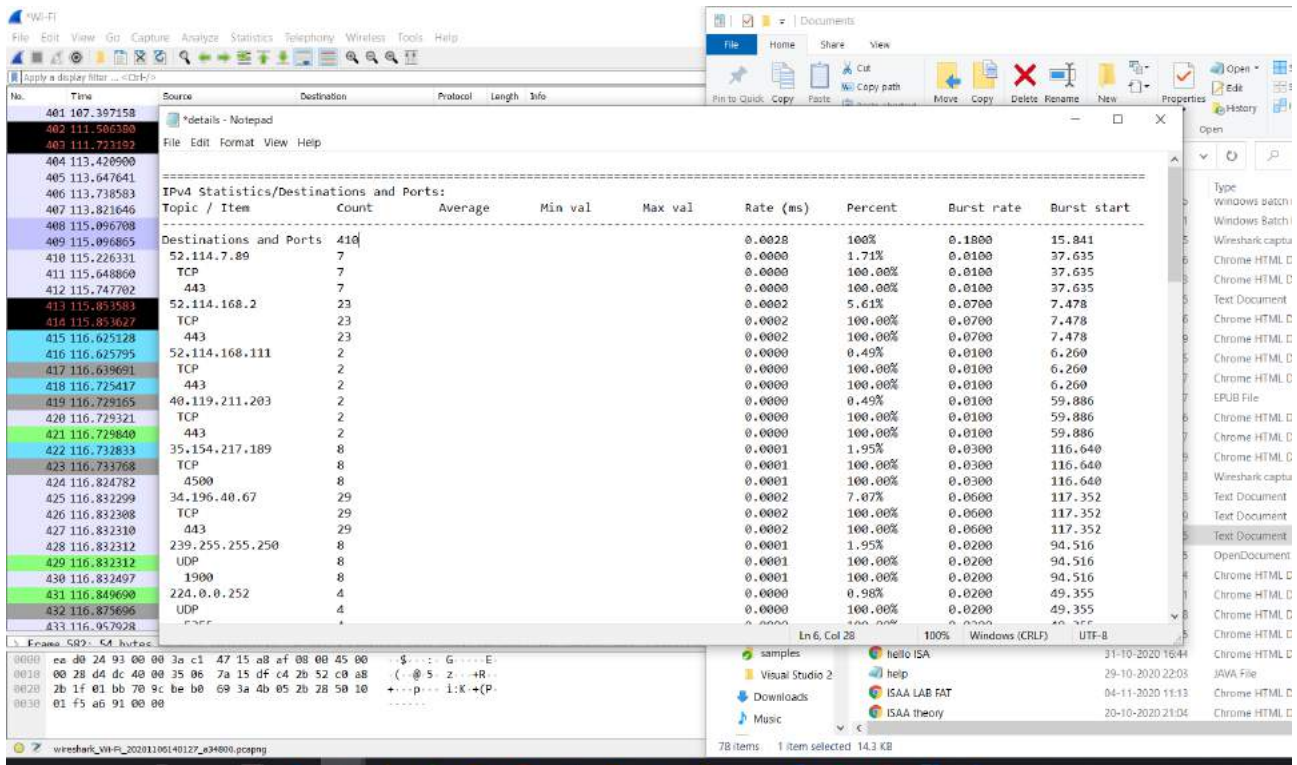
```
C:\WINDOWS\system32\cmd.exe

C:\Users>tracert 106.67.100.122

Tracing route to 106.67.100.122 over a maximum of 30 hops:
  0  <--> 0 ms  0 ms  0 ms  10.0.2.15
  1  <--> 1 ms  1 ms  1 ms  10.0.2.1
  2  <--> 1 ms  1 ms  1 ms  10.0.2.1
  3  <--> 1 ms  1 ms  1 ms  10.0.2.1
  4  <--> 1 ms  1 ms  1 ms  10.0.2.1
  5  <--> 1 ms  1 ms  1 ms  10.0.2.1
  6  <--> 1 ms  1 ms  1 ms  10.0.2.1
  7  <--> 1 ms  1 ms  1 ms  10.0.2.1
  8  <--> 1 ms  1 ms  1 ms  10.0.2.1
  9  <--> 1 ms  1 ms  1 ms  10.0.2.1
 10  <--> 1 ms  1 ms  1 ms  10.0.2.1
 11  <--> 1 ms  1 ms  1 ms  10.0.2.1
 12  <--> 1 ms  1 ms  1 ms  10.0.2.1
 13  <--> 1 ms  1 ms  1 ms  10.0.2.1
 14  <--> 1 ms  1 ms  1 ms  10.0.2.1
 15  <--> 1 ms  1 ms  1 ms  10.0.2.1
 16  <--> 1 ms  1 ms  1 ms  10.0.2.1
 17  <--> 1 ms  1 ms  1 ms  10.0.2.1
 18  <--> 1 ms  1 ms  1 ms  10.0.2.1
 19  <--> 1 ms  1 ms  1 ms  10.0.2.1
 20  <--> 1 ms  1 ms  1 ms  10.0.2.1
 21  <--> 1 ms  1 ms  1 ms  10.0.2.1
 22  <--> 1 ms  1 ms  1 ms  10.0.2.1
 23  <--> 1 ms  1 ms  1 ms  10.0.2.1
 24  <--> 1 ms  1 ms  1 ms  10.0.2.1
 25  <--> 1 ms  1 ms  1 ms  10.0.2.1
 26  <--> 1 ms  1 ms  1 ms  10.0.2.1
 27  <--> 1 ms  1 ms  1 ms  10.0.2.1
 28  <--> 1 ms  1 ms  1 ms  10.0.2.1
 29  <--> 1 ms  1 ms  1 ms  10.0.2.1
 30  <--> 1 ms  1 ms  1 ms  10.0.2.1
 31  <--> 1 ms  1 ms  1 ms  10.0.2.1
 32  <--> 1 ms  1 ms  1 ms  10.0.2.1
 33  <--> 1 ms  1 ms  1 ms  10.0.2.1
 34  <--> 1 ms  1 ms  1 ms  10.0.2.1
 35  <--> 1 ms  1 ms  1 ms  10.0.2.1
 36  <--> 1 ms  1 ms  1 ms  10.0.2.1
 37  <--> 1 ms  1 ms  1 ms  10.0.2.1
 38  <--> 1 ms  1 ms  1 ms  10.0.2.1
 39  <--> 1 ms  1 ms  1 ms  10.0.2.1
 40  <--> 1 ms  1 ms  1 ms  10.0.2.1
 41  <--> 1 ms  1 ms  1 ms  10.0.2.1
 42  <--> 1 ms  1 ms  1 ms  10.0.2.1
 43  <--> 1 ms  1 ms  1 ms  10.0.2.1
 44  <--> 1 ms  1 ms  1 ms  10.0.2.1
 45  <--> 1 ms  1 ms  1 ms  10.0.2.1
 46  <--> 1 ms  1 ms  1 ms  10.0.2.1
 47  <--> 1 ms  1 ms  1 ms  10.0.2.1
 48  <--> 1 ms  1 ms  1 ms  10.0.2.1
 49  <--> 1 ms  1 ms  1 ms  10.0.2.1
 50  <--> 1 ms  1 ms  1 ms  10.0.2.1
 51  <--> 1 ms  1 ms  1 ms  10.0.2.1
 52  <--> 1 ms  1 ms  1 ms  10.0.2.1
 53  <--> 1 ms  1 ms  1 ms  10.0.2.1
 54  <--> 1 ms  1 ms  1 ms  10.0.2.1
 55  <--> 1 ms  1 ms  1 ms  10.0.2.1
 56  <--> 1 ms  1 ms  1 ms  10.0.2.1
 57  <--> 1 ms  1 ms  1 ms  10.0.2.1
 58  <--> 1 ms  1 ms  1 ms  10.0.2.1
 59  <--> 1 ms  1 ms  1 ms  10.0.2.1
 60  <--> 1 ms  1 ms  1 ms  10.0.2.1
 61  <--> 1 ms  1 ms  1 ms  10.0.2.1
 62  <--> 1 ms  1 ms  1 ms  10.0.2.1
 63  <--> 1 ms  1 ms  1 ms  10.0.2.1
 64  <--> 1 ms  1 ms  1 ms  10.0.2.1
 65  <--> 1 ms  1 ms  1 ms  10.0.2.1
 66  <--> 1 ms  1 ms  1 ms  10.0.2.1
 67  <--> 1 ms  1 ms  1 ms  10.0.2.1
 68  <--> 1 ms  1 ms  1 ms  10.0.2.1
 69  <--> 1 ms  1 ms  1 ms  10.0.2.1
 70  <--> 1 ms  1 ms  1 ms  10.0.2.1
 71  <--> 1 ms  1 ms  1 ms  10.0.2.1
 72  <--> 1 ms  1 ms  1 ms  10.0.2.1
 73  <--> 1 ms  1 ms  1 ms  10.0.2.1
 74  <--> 1 ms  1 ms  1 ms  10.0.2.1
 75  <--> 1 ms  1 ms  1 ms  10.0.2.1
 76  <--> 1 ms  1 ms  1 ms  10.0.2.1
 77  <--> 1 ms  1 ms  1 ms  10.0.2.1
 78  <--> 1 ms  1 ms  1 ms  10.0.2.1
 79  <--> 1 ms  1 ms  1 ms  10.0.2.1
 80  <--> 1 ms  1 ms  1 ms  10.0.2.1
 81  <--> 1 ms  1 ms  1 ms  10.0.2.1
 82  <--> 1 ms  1 ms  1 ms  10.0.2.1
 83  <--> 1 ms  1 ms  1 ms  10.0.2.1
 84  <--> 1 ms  1 ms  1 ms  10.0.2.1
 85  <--> 1 ms  1 ms  1 ms  10.0.2.1
 86  <--> 1 ms  1 ms  1 ms  10.0.2.1
 87  <--> 1 ms  1 ms  1 ms  10.0.2.1
 88  <--> 1 ms  1 ms  1 ms  10.0.2.1
 89  <--> 1 ms  1 ms  1 ms  10.0.2.1
 90  <--> 1 ms  1 ms  1 ms  10.0.2.1
 91  <--> 1 ms  1 ms  1 ms  10.0.2.1
 92  <--> 1 ms  1 ms  1 ms  10.0.2.1
 93  <--> 1 ms  1 ms  1 ms  10.0.2.1
 94  <--> 1 ms  1 ms  1 ms  10.0.2.1
 95  <--> 1 ms  1 ms  1 ms  10.0.2.1
 96  <--> 1 ms  1 ms  1 ms  10.0.2.1
 97  <--> 1 ms  1 ms  1 ms  10.0.2.1
 98  <--> 1 ms  1 ms  1 ms  10.0.2.1
 99  <--> 1 ms  1 ms  1 ms  10.0.2.1
100  <--> 1 ms  1 ms  1 ms  10.0.2.1
101  <--> 1 ms  1 ms  1 ms  10.0.2.1
102  <--> 1 ms  1 ms  1 ms  10.0.2.1
103  <--> 1 ms  1 ms  1 ms  10.0.2.1
104  <--> 1 ms  1 ms  1 ms  10.0.2.1
1
```

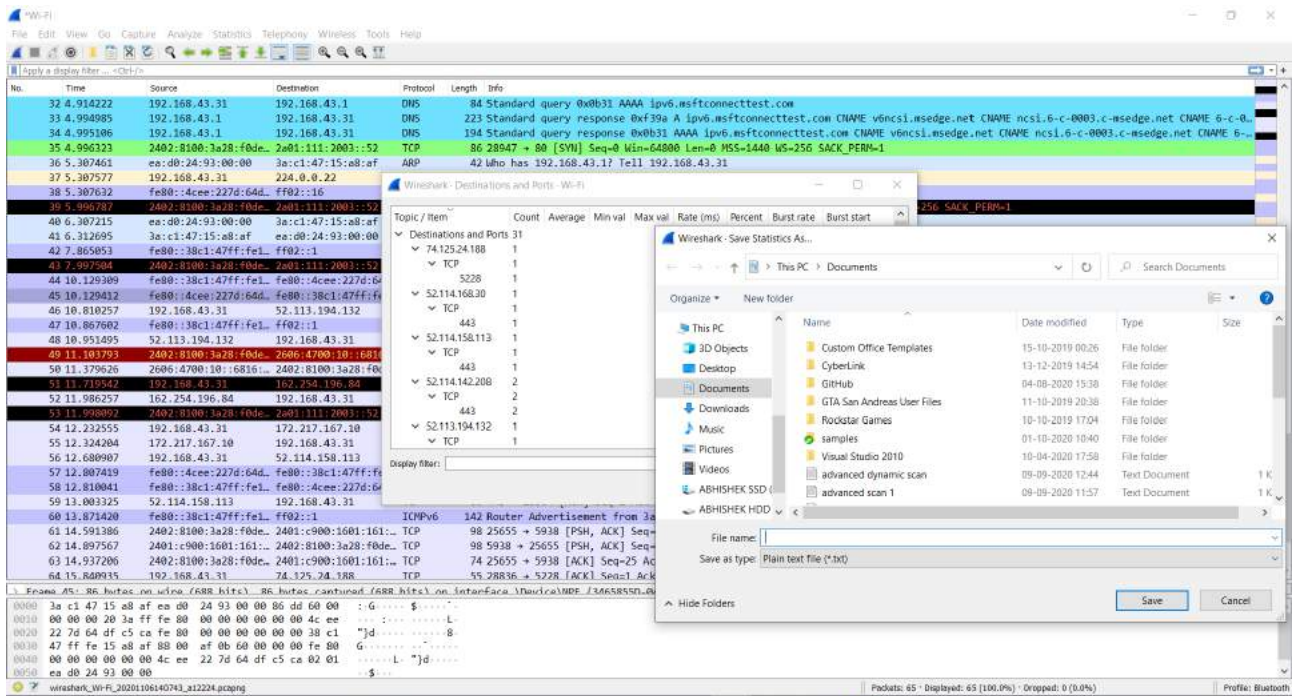
Step-3- Suppose this attack is done on me So ill Store my Average Packet details in my database in which it contains the count of the DNS and UDP Packets and save that file with XYZ.txt.



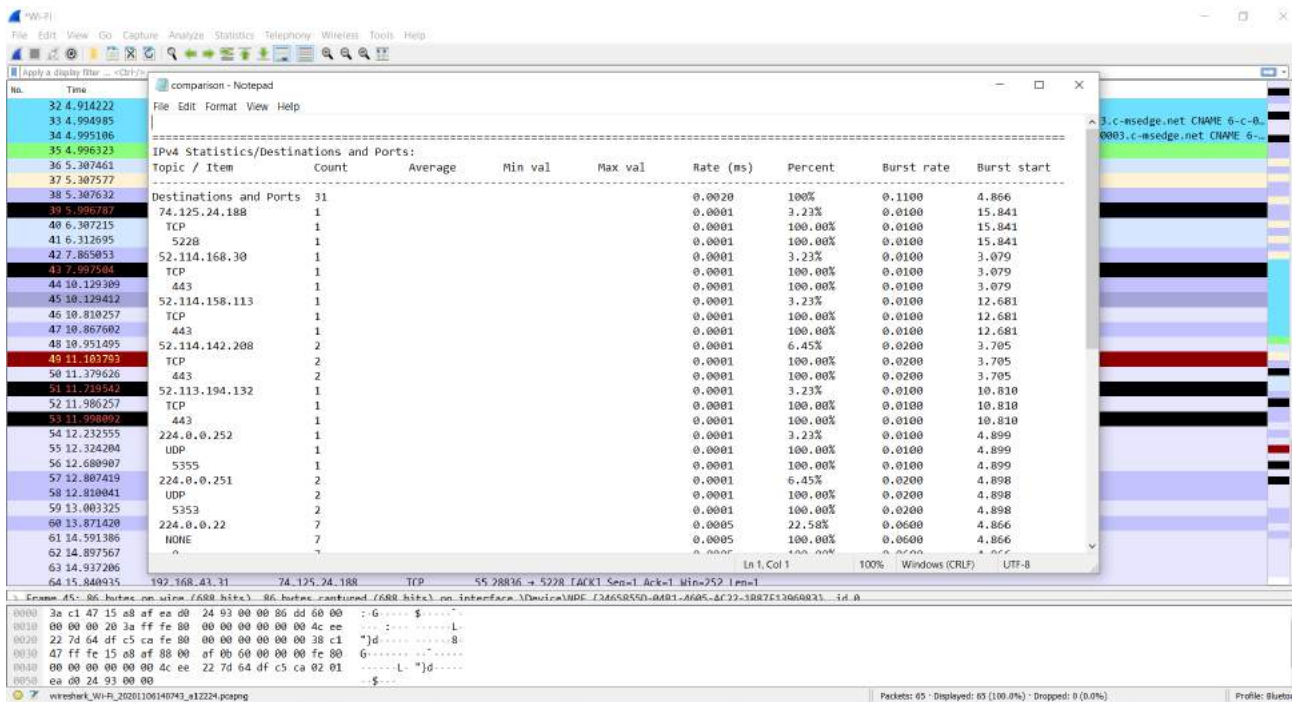


Details of the count of the Packets got saved in the desktop with .txt file.

Step-4- So now when we will use Wireshark and try to count the Packets that are coming on our network in a limited time suppose for 10 15 sec.

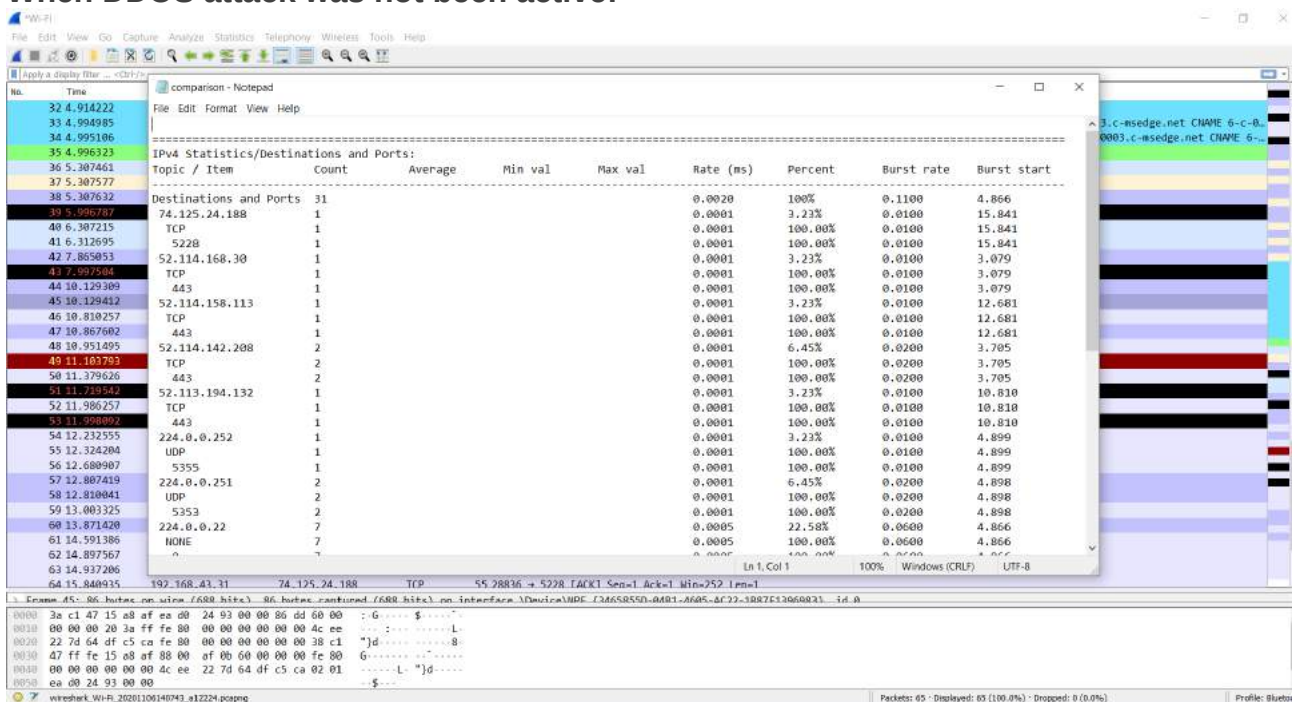


This is when Ddos attack was not been active. We are saving this for further comparison.

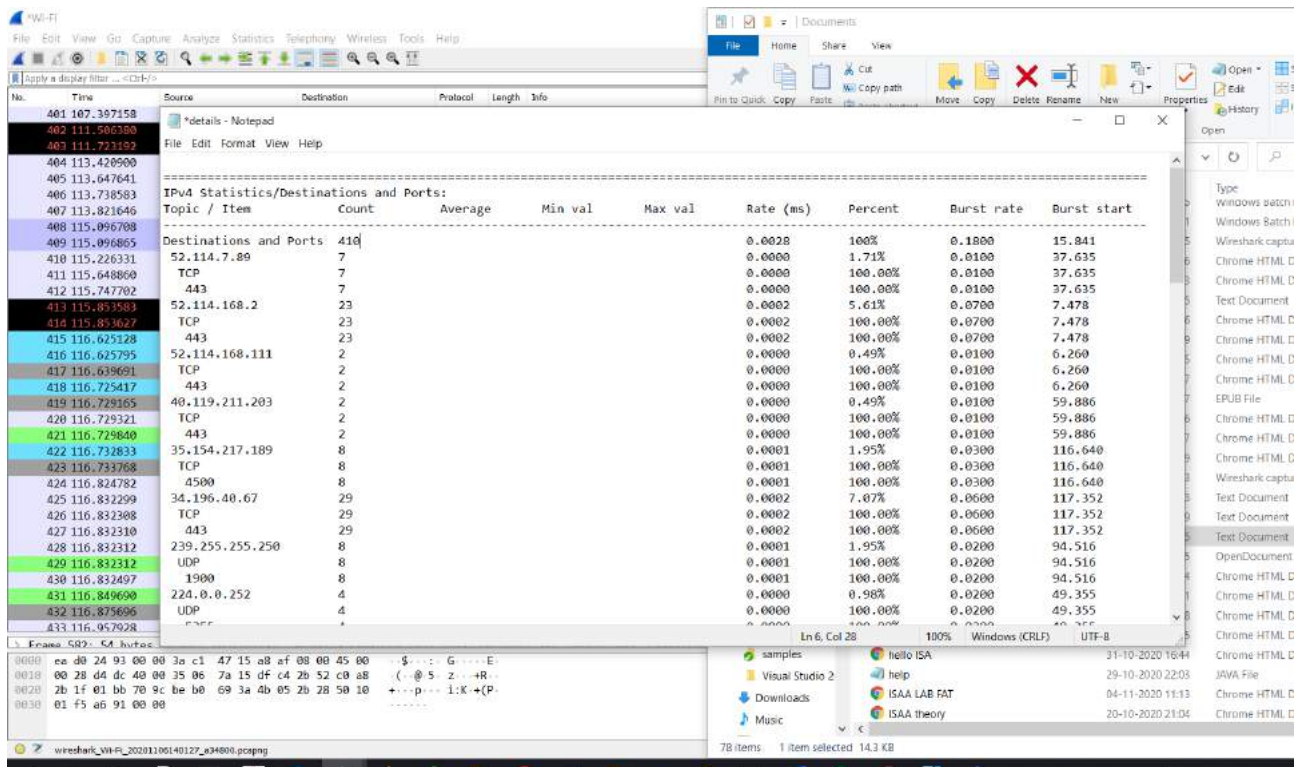


Step-5- Now we will Compare that count With Our Previous Packet details that was stored in xyz.txt (suppose threshold was set to 500 Packets (DNS)) and if it will exceed this no then we can say that we are under ddos attack.

When DDOS attack was not been active:-



When DDOS attack was active:-



The application was designed with the help of winpcap library on Windows system. and it is a library that enables actions like packet capturing and sending on our network card with ease. In the application part, network traffic is monitored via the program created with C programming language and packets are captured. As it can be gathered from filtering was made according to the number of packets to be captured and the type of packets. In the application run here, it was aimed to capture 100 packets and filter them in Tcp type.

The way any of the packets captured looks in the terminal was added to. As it is clear from the figure, packets were captured according to packet number, from address, to address, protocol type, source port, target port and payload (data about packet).

Completed attack detections are recorded onto the database. Attacks gathered from the database are given on web interface. Attacks are displayed with day and time of occurrence. While attacks are being displayed, data is derived from the database. Once in every two seconds, it is detected whether there is an attack in the system.

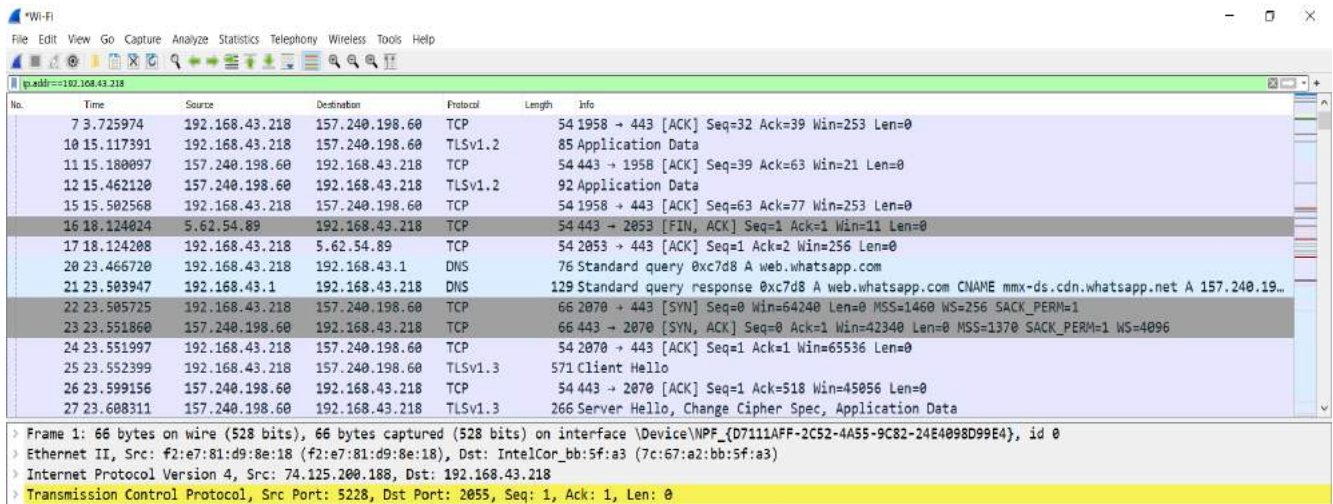
ANALYSIS OF WIRESHARK:-

LIST OF USEFULL WIRESHARK FILTERS

1. Filter traffic on specific IP address

This will display all traffic for the IP entered, source or destination.

ip.addr==192.168.43.218



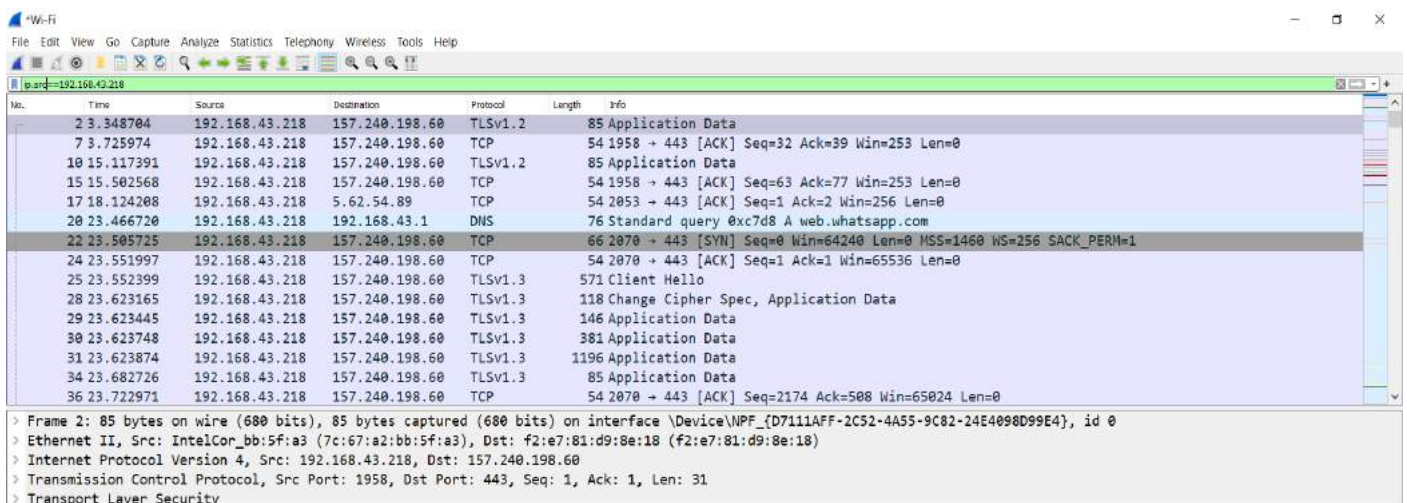
No.	Time	Source	Destination	Protocol	Length	Info
7	3.725974	192.168.43.218	157.240.198.60	TCP	54	1958 → 443 [ACK] Seq=32 Ack=39 Win=253 Len=0
10	15.117391	192.168.43.218	157.240.198.60	TLSv1.2	85	Application Data
11	15.180097	157.240.198.60	192.168.43.218	TCP	54	443 → 1958 [ACK] Seq=39 Ack=63 Win=21 Len=0
12	15.462120	157.240.198.60	192.168.43.218	TLSv1.2	92	Application Data
15	15.502568	192.168.43.218	157.240.198.60	TCP	54	1958 → 443 [ACK] Seq=63 Ack=77 Win=253 Len=0
16	18.124024	5.62.54.89	192.168.43.218	TCP	54	443 → 2053 [FIN, ACK] Seq=1 Ack=1 Win=11 Len=0
17	18.124208	192.168.43.218	5.62.54.89	TCP	54	2053 → 443 [ACK] Seq=1 Ack=2 Win=256 Len=0
20	23.466720	192.168.43.218	192.168.43.1	DNS	76	Standard query 0xc7d8 A web.whatsapp.com
21	23.503947	192.168.43.1	192.168.43.218	DNS	129	Standard query response 0xc7d8 A web.whatsapp.com CNAME mmx-ds.cdn.whatsapp.net A 157.240.19...
22	23.505725	192.168.43.218	157.240.198.60	TCP	66	2070 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
23	23.551860	157.240.198.60	192.168.43.218	TCP	66	443 → 2070 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1370 SACK_PERM=1 WS=4096
24	23.551997	192.168.43.218	157.240.198.60	TCP	54	2070 → 443 [ACK] Seq=1 Ack=1 Win=65536 Len=0
25	23.552399	192.168.43.218	157.240.198.60	TLSv1.3	571	Client Hello
26	23.599156	157.240.198.60	192.168.43.218	TCP	54	443 → 2070 [ACK] Seq=1 Ack=518 Win=45056 Len=0
27	23.608311	157.240.198.60	192.168.43.218	TLSv1.3	266	Server Hello, Change Cipher Spec, Application Data

> Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{D7111AFF-2C52-4A55-9C82-24E4098D99E4}, id 0
> Ethernet II, Src: f2:e7:81:d9:8e:18 (f2:e7:81:d9:8e:18), Dst: IntelCor_bb:5f:a3 (7c:67:a2:bb:5f:a3)
> Internet Protocol Version 4, Src: 74.125.200.188, Dst: 192.168.43.218
> Transmission Control Protocol, Src Port: 5228, Dst Port: 2055, Seq: 1, Ack: 1, Len: 0

2. Filter by source address

This will only show traffic where the source IP address is 192.168.43.218

ip.src==192.168.43.218



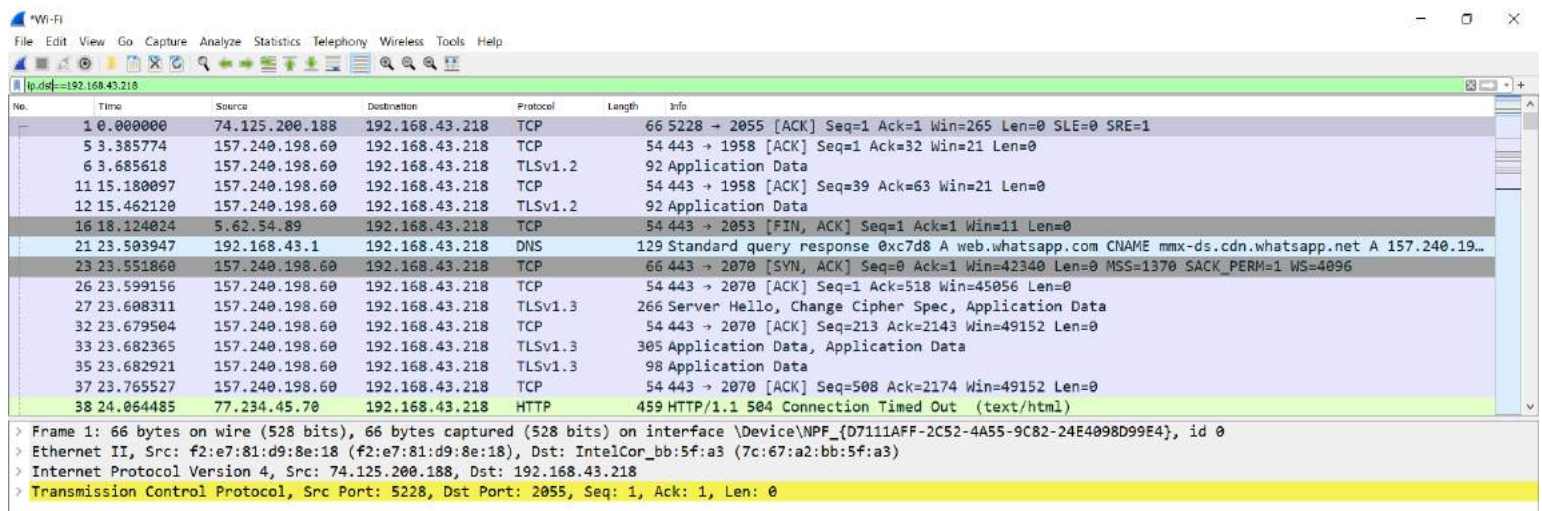
No.	Time	Source	Destination	Protocol	Length	Info
23	3.348704	192.168.43.218	157.240.198.60	TLSv1.2	85	Application Data
7	3.725974	192.168.43.218	157.240.198.60	TCP	54	1958 → 443 [ACK] Seq=32 Ack=39 Win=253 Len=0
10	15.117391	192.168.43.218	157.240.198.60	TLSv1.2	85	Application Data
15	15.502568	192.168.43.218	157.240.198.60	TCP	54	1958 → 443 [ACK] Seq=63 Ack=77 Win=253 Len=0
17	18.124208	192.168.43.218	5.62.54.89	TCP	54	2053 → 443 [ACK] Seq=1 Ack=2 Win=256 Len=0
20	23.466720	192.168.43.218	192.168.43.1	DNS	76	Standard query 0xc7d8 A web.whatsapp.com
22	23.505725	192.168.43.218	157.240.198.60	TCP	66	2070 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
24	23.551997	192.168.43.218	157.240.198.60	TCP	54	2070 → 443 [ACK] Seq=1 Ack=1 Win=65536 Len=0
25	23.552399	192.168.43.218	157.240.198.60	TLSv1.3	571	Client Hello
28	23.623165	192.168.43.218	157.240.198.60	TLSv1.3	118	Change Cipher Spec, Application Data
29	23.623445	192.168.43.218	157.240.198.60	TLSv1.3	146	Application Data
30	23.623748	192.168.43.218	157.240.198.60	TLSv1.3	381	Application Data
31	23.623874	192.168.43.218	157.240.198.60	TLSv1.3	1196	Application Data
34	23.682726	192.168.43.218	157.240.198.60	TLSv1.3	85	Application Data
36	23.722971	192.168.43.218	157.240.198.60	TCP	54	2070 → 443 [ACK] Seq=2174 Ack=508 Win=65024 Len=0

> Frame 2: 85 bytes on wire (680 bits), 85 bytes captured (680 bits) on interface \Device\NPF_{D7111AFF-2C52-4A55-9C82-24E4098D99E4}, id 0
> Ethernet II, Src: IntelCor_bb:5f:a3 (7c:67:a2:bb:5f:a3), Dst: f2:e7:81:d9:8e:18 (f2:e7:81:d9:8e:18)
> Internet Protocol Version 4, Src: 192.168.43.218, Dst: 157.240.198.60
> Transmission Control Protocol, Src Port: 1958, Dst Port: 443, Seq: 1, Ack: 1, Len: 31
> Transport Layer Security

3. Filter by destination address

Displays only traffic for the matching destination IP.

ip.dst==192.168.43.218



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	74.125.200.188	192.168.43.218	TCP	66	5228 → 2055 [ACK] Seq=1 Ack=1 Win=265 Len=0 SLE=0 SRE=1
5	3.385774	157.240.198.60	192.168.43.218	TCP	54	443 → 1958 [ACK] Seq=1 Ack=32 Win=21 Len=0
6	3.685618	157.240.198.60	192.168.43.218	TLSv1.2	92	Application Data
11	15.180097	157.240.198.60	192.168.43.218	TCP	54	443 → 1958 [ACK] Seq=39 Ack=63 Win=21 Len=0
12	15.462120	157.240.198.60	192.168.43.218	TLSv1.2	92	Application Data
16	18.124024	5.62.54.89	192.168.43.218	TCP	54	443 → 2053 [FIN, ACK] Seq=1 Ack=1 Win=11 Len=0
21	23.503947	192.168.43.1	192.168.43.218	DNS	129	Standard query response 0xc7d8 A web.whatsapp.com CNAME mmx-ds.cdn.whatsapp.net A 157.240.19...
23	23.551860	157.240.198.60	192.168.43.218	TCP	66	443 → 2070 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1370 SACK_PERM=1 WS=4096
26	23.599156	157.240.198.60	192.168.43.218	TCP	54	443 → 2070 [ACK] Seq=1 Ack=518 Win=45056 Len=0
27	23.608311	157.240.198.60	192.168.43.218	TLSv1.3	266	Server Hello, Change Cipher Spec, Application Data
32	23.679504	157.240.198.60	192.168.43.218	TCP	54	443 → 2070 [ACK] Seq=213 Ack=2143 Win=49152 Len=0
33	23.682365	157.240.198.60	192.168.43.218	TLSv1.3	305	Application Data, Application Data
35	23.682921	157.240.198.60	192.168.43.218	TLSv1.3	98	Application Data
37	23.765527	157.240.198.60	192.168.43.218	TCP	54	443 → 2070 [ACK] Seq=508 Ack=2174 Win=49152 Len=0
38	24.064485	77.234.45.70	192.168.43.218	HTTP	459	HTTP/1.1 504 Connection Timed Out (text/html)

> Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{D7111AFF-2C52-4A55-9C82-24E4098D99E4}, id 0
> Ethernet II, Src: f2:e7:81:d9:8e:18 (f2:e7:81:d9:8e:18), Dst: IntelCor_bb:5f:a3 (7c:67:a2:bb:5f:a3)
> Internet Protocol Version 4, Src: 74.125.200.188, Dst: 192.168.43.218
> Transmission Control Protocol, Src Port: 5228, Dst Port: 2055, Seq: 1, Ack: 1, Len: 0

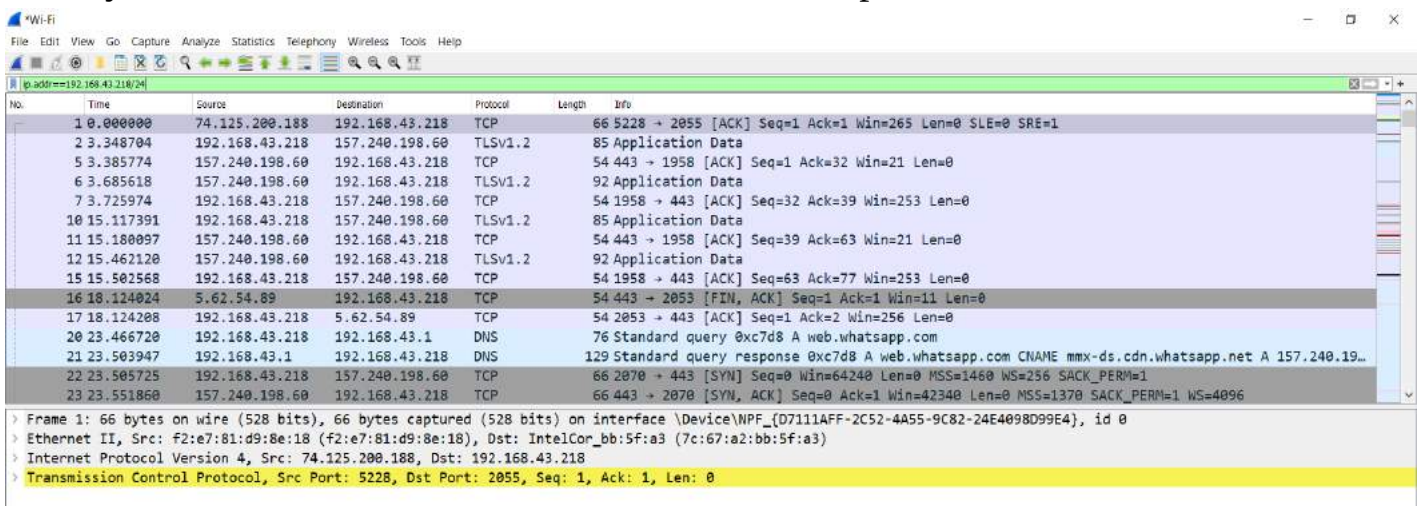
4. Filter by IP subnet

Displays all traffic for the entered subnet, this will match on source or destination. Use CIDR format for subnet display filter.

ip.addr==192.168.43.218/24

If you want to filter on a IP source subnet use **ip.src==subnet**

If you want to filter on IP destination subnet use **ip.dst==subnet**



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	74.125.200.188	192.168.43.218	TCP	66	5228 → 2055 [ACK] Seq=1 Ack=1 Win=265 Len=0 SLE=0 SRE=1
2	3.348704	192.168.43.218	157.240.198.60	TLSv1.2	85	Application Data
5	3.385774	157.240.198.60	192.168.43.218	TCP	54	443 → 1958 [ACK] Seq=1 Ack=32 Win=21 Len=0
6	3.685618	157.240.198.60	192.168.43.218	TLSv1.2	92	Application Data
7	3.725974	192.168.43.218	157.240.198.60	TCP	54	1958 → 443 [ACK] Seq=32 Ack=39 Win=253 Len=0
10	15.117391	192.168.43.218	157.240.198.60	TLSv1.2	85	Application Data
11	15.180097	157.240.198.60	192.168.43.218	TCP	54	443 → 1958 [ACK] Seq=39 Ack=63 Win=21 Len=0
12	15.462120	157.240.198.60	192.168.43.218	TLSv1.2	92	Application Data
15	15.502568	192.168.43.218	157.240.198.60	TCP	54	1958 → 443 [ACK] Seq=63 Ack=77 Win=253 Len=0
16	18.124024	5.62.54.89	192.168.43.218	TCP	54	443 → 2053 [FIN, ACK] Seq=1 Ack=1 Win=11 Len=0
17	18.124208	192.168.43.218	5.62.54.89	TCP	54	2053 → 443 [ACK] Seq=1 Ack=2 Win=256 Len=0
20	23.466720	192.168.43.218	192.168.43.1	DNS	76	Standard query 0xc7d8 A web.whatsapp.com
21	23.503947	192.168.43.1	192.168.43.218	DNS	129	Standard query response 0xc7d8 A web.whatsapp.com CNAME mmx-ds.cdn.whatsapp.net A 157.240.19...
22	23.505725	192.168.43.218	157.240.198.60	TCP	66	2070 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
23	23.551860	157.240.198.60	192.168.43.218	TCP	66	443 → 2070 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1370 SACK_PERM=1 WS=4096

> Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{D7111AFF-2C52-4A55-9C82-24E4098D99E4}, id 0
> Ethernet II, Src: f2:e7:81:d9:8e:18 (f2:e7:81:d9:8e:18), Dst: IntelCor_bb:5f:a3 (7c:67:a2:bb:5f:a3)
> Internet Protocol Version 4, Src: 74.125.200.188, Dst: 192.168.43.218
> Transmission Control Protocol, Src Port: 5228, Dst Port: 2055, Seq: 1, Ack: 1, Len: 0

5. Filter traffic based on protocol

To filter for a specific protocol just type in the name of the protocol. For example to display all DNS traffic just type DNS in the filter box.

dns

Some other common protocols you could filter on: arp, http, ftp, smtp, ssh, telnet, bootp, icmp.

Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Filter: dns

No.	Time	Source	Destination	Protocol	Length	Info
20	23.466720	192.168.43.218	192.168.43.1	DNS	76	Standard query 0xc7d8 A web.whatsapp.com
21	23.503947	192.168.43.1	192.168.43.218	DNS	129	Standard query response 0xc7d8 A web.whatsapp.com CNAME mmx-ds.cdn.whatsapp.net A 157.240.19..
75	54.878040	192.168.43.218	192.168.43.1	DNS	74	Standard query 0xd493 A api.myloft.xyz
76	54.880217	192.168.43.218	192.168.43.1	DNS	87	Standard query 0xe029 A googleads.g.doubleclick.net
77	54.925724	192.168.43.1	192.168.43.218	DNS	128	Standard query response 0xe029 A googleads.g.doubleclick.net CNAME pagead46.l.doubleclick.ne..
78	54.940604	192.168.43.218	192.168.43.1	DNS	74	Standard query 0xd493 A api.myloft.xyz
79	54.946034	192.168.43.1	192.168.43.218	DNS	106	Standard query response 0xd493 A api.myloft.xyz A 34.196.40.67 A 34.199.143.64
130	57.866721	192.168.43.218	192.168.43.1	DNS	74	Standard query 0x7b84 A vit.myloft.xyz
132	57.928703	192.168.43.218	192.168.43.1	DNS	74	Standard query 0x7b84 A vit.myloft.xyz
133	57.939199	192.168.43.1	192.168.43.218	DNS	90	Standard query response 0x7b84 A vit.myloft.xyz A 35.154.217.189
170	75.127477	192.168.43.218	192.168.43.1	DNS	80	Standard query 0xcc28 A nos.ns1.ff.avast.com
172	75.183197	192.168.43.1	192.168.43.218	DNS	144	Standard query response 0xcc28 A nos.ns1.ff.avast.com A 5.62.54.29 A 5.62.54.31 A 5.62.54.35..
231	130.868606	192.168.43.218	192.168.43.1	DNS	74	Standard query 0x8f51 A api.myloft.xyz
235	130.930370	192.168.43.218	192.168.43.1	DNS	74	Standard query 0x8f51 A api.myloft.xyz
236	130.947811	192.168.43.1	192.168.43.218	DNS	106	Standard query response 0x8f51 A api.myloft.xyz A 34.199.143.64 A 34.196.40.67

> Frame 20: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface \Device\NPF_{D7111AFF-2C52-4A55-9C82-24E4098D99E4}, id 0
 > Ethernet II, Src: IntelCor_bb:5f:a3 (7c:67:a2:bb:5f:a3), Dst: f2:e7:81:d9:8e:18 (f2:e7:81:d9:8e:18)
 > Internet Protocol Version 4, Src: 192.168.43.218, Dst: 192.168.43.1
 > User Datagram Protocol, Src Port: 64862, Dst Port: 53
 > Domain Name System (query)

6. Exclude IP address

If you want to filter out an IP address so it's not displayed use this filter.

!ip.addr==192.168.43.218

Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Filter: !ip.addr==192.168.43.218

No.	Time	Source	Destination	Protocol	Length	Info
3	3.360735	f2:e7:81:d9:8e:...	IntelCor_bb:5f:...	ARP	42	Who has 192.168.43.218? Tell 192.168.43.1
4	3.360768	IntelCor_bb:5f:...	f2:e7:81:d9:8e:...	ARP	42	192.168.43.218 is at 7c:67:a2:bb:5f:a3
8	5.438828	IntelCor_bb:5f:...	Broadcast	ARP	42	Who has 192.168.43.1? Tell 192.168.43.218
9	5.441120	f2:e7:81:d9:8e:...	IntelCor_bb:5f:...	ARP	42	192.168.43.1 is at f2:e7:81:d9:8e:18
13	15.485544	IntelCor_bb:5f:...	Broadcast	ARP	42	Who has 192.168.43.1? Tell 192.168.43.218
14	15.487402	f2:e7:81:d9:8e:...	IntelCor_bb:5f:...	ARP	42	192.168.43.1 is at f2:e7:81:d9:8e:18
18	23.141639	f2:e7:81:d9:8e:...	IntelCor_bb:5f:...	ARP	42	Who has 192.168.43.218? Tell 192.168.43.1
19	23.141677	IntelCor_bb:5f:...	f2:e7:81:d9:8e:...	ARP	42	192.168.43.218 is at 7c:67:a2:bb:5f:a3
49	25.532076	IntelCor_bb:5f:...	Broadcast	ARP	42	Who has 192.168.43.1? Tell 192.168.43.218
50	25.533747	f2:e7:81:d9:8e:...	IntelCor_bb:5f:...	ARP	42	192.168.43.1 is at f2:e7:81:d9:8e:18
55	27.771457	IntelCor_bb:5f:...	Broadcast	ARP	42	Who has 192.168.43.1? Tell 192.168.43.218
56	27.778325	f2:e7:81:d9:8e:...	IntelCor_bb:5f:...	ARP	42	192.168.43.1 is at f2:e7:81:d9:8e:18
63	32.990627	IntelCor_bb:5f:...	Broadcast	ARP	42	Who has 192.168.43.1? Tell 192.168.43.218
64	32.993420	f2:e7:81:d9:8e:...	IntelCor_bb:5f:...	ARP	42	192.168.43.1 is at f2:e7:81:d9:8e:18
65	35.584146	IntelCor_bb:5f:...	Broadcast	ARP	42	Who has 192.168.43.1? Tell 192.168.43.218

> Frame 9: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface \Device\NPF_{D7111AFF-2C52-4A55-9C82-24E4098D99E4}, id 0
 > Ethernet II, Src: f2:e7:81:d9:8e:18 (f2:e7:81:d9:8e:18), Dst: IntelCor_bb:5f:a3 (7c:67:a2:bb:5f:a3)
 > Address Resolution Protocol (reply)

7. Show traffic between two workstations or subnet

This first one will show only traffic between the two subnets

ip.addr==192.168.43.200/24 and ip.addr==192.168.43.218/24

Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

ip.addr==192.168.43.218 and ip.addr==192.168.43.218/24

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	74.125.200.188	192.168.43.218	TCP	66	5228 → 2055 [ACK] Seq=1 Ack=1 Win=265 Len=0 SLE=0 SRE=1
2	3.348704	192.168.43.218	157.240.198.60	TLSv1.2	85	Application Data
3	3.385774	157.240.198.60	192.168.43.218	TCP	54	443 → 1958 [ACK] Seq=1 Ack=32 Win=21 Len=0
6	3.685618	157.240.198.60	192.168.43.218	TLSv1.2	92	Application Data
7	3.725974	192.168.43.218	157.240.198.60	TCP	54	1958 → 443 [ACK] Seq=32 Ack=39 Win=253 Len=0
10	15.117391	192.168.43.218	157.240.198.60	TLSv1.2	85	Application Data
11	15.180097	157.240.198.60	192.168.43.218	TCP	54	443 → 1958 [ACK] Seq=39 Ack=63 Win=21 Len=0
12	15.462120	157.240.198.60	192.168.43.218	TLSv1.2	92	Application Data
15	15.502568	192.168.43.218	157.240.198.60	TCP	54	1958 → 443 [ACK] Seq=63 Ack=77 Win=253 Len=0
16	18.124024	5.62.54.89	192.168.43.218	TCP	54	443 → 2053 [FIN, ACK] Seq=1 Ack=1 Win=11 Len=0
17	18.124208	192.168.43.218	5.62.54.89	TCP	54	2053 → 443 [ACK] Seq=1 Ack=2 Win=256 Len=0
20	23.466720	192.168.43.218	192.168.43.1	DNS	76	Standard query 0xc7d8 A web.whatsapp.com
21	23.503947	192.168.43.1	192.168.43.218	DNS	129	Standard query response 0xc7d8 A web.whatsapp.com CNAME mmx-ds.cdn.whatsapp.net A 157.240.19...
22	23.505725	192.168.43.218	157.240.198.60	TCP	66	2070 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
23	23.551860	157.240.198.60	192.168.43.218	TCP	66	443 → 2070 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1370 SACK_PERM=1 WS=4096

> Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{D7111AFF-2C52-4A55-9C82-24E4098D99E4}, id 0

> Ethernet II, Src: f2:e7:81:d9:8e:18 (f2:e7:81:d9:8e:18), Dst: IntelCor_bb:5f:a3 (7c:67:a2:bb:5f:a3)

> Internet Protocol Version 4, Src: 74.125.200.188, Dst: 192.168.43.218

> Transmission Control Protocol, Src Port: 5228, Dst Port: 2055, Seq: 1, Ack: 1, Len: 0

This will show only traffic between the two specific IP address

ip.addr==192.168.43.1 and ip.addr==192.168.43.218

Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

ip.addr==192.168.43.1 and ip.addr==192.168.43.218

No.	Time	Source	Destination	Protocol	Length	Info
20	23.466720	192.168.43.218	192.168.43.1	DNS	76	Standard query 0xc7d8 A web.whatsapp.com
21	23.503947	192.168.43.1	192.168.43.218	DNS	129	Standard query response 0xc7d8 A web.whatsapp.com CNAME mmx-ds.cdn.whatsapp.net A 157.240.19...
75	54.878040	192.168.43.218	192.168.43.1	DNS	74	Standard query 0xd493 A api.myloft.xyz
76	54.880217	192.168.43.218	192.168.43.1	DNS	87	Standard query 0xe029 A googleads.g.doubleclick.net
77	54.925724	192.168.43.1	192.168.43.218	DNS	128	Standard query response 0xe029 A googleads.g.doubleclick.net CNAME pagead46.l.doubleclick.ne...
78	54.940604	192.168.43.218	192.168.43.1	DNS	74	Standard query 0xd493 A api.myloft.xyz
79	54.946034	192.168.43.1	192.168.43.218	DNS	106	Standard query response 0xd493 A api.myloft.xyz A 34.196.40.67 A 34.199.143.64
130	57.866721	192.168.43.218	192.168.43.1	DNS	74	Standard query 0x7b84 A vit.myloft.xyz
132	57.928703	192.168.43.218	192.168.43.1	DNS	74	Standard query 0x7b84 A vit.myloft.xyz
133	57.939199	192.168.43.1	192.168.43.218	DNS	90	Standard query response 0x7b84 A vit.myloft.xyz A 35.154.217.189
170	75.127477	192.168.43.218	192.168.43.1	DNS	80	Standard query 0xcc28 A nos.nsl.ff.avast.com
172	75.183197	192.168.43.1	192.168.43.218	DNS	144	Standard query response 0xcc28 A nos.nsl.ff.avast.com A 5.62.54.29 A 5.62.54.31 A 5.62.54.35...
231	130.868606	192.168.43.218	192.168.43.1	DNS	74	Standard query 0x8f51 A api.myloft.xyz
235	130.930370	192.168.43.218	192.168.43.1	DNS	74	Standard query 0x8f51 A api.myloft.xyz
236	130.947811	192.168.43.1	192.168.43.218	DNS	106	Standard query response 0x8f51 A api.myloft.xyz A 34.199.143.64 A 34.196.40.67

> Frame 20: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface \Device\NPF_{D7111AFF-2C52-4A55-9C82-24E4098D99E4}, id 0

> Ethernet II, Src: IntelCor_bb:5f:a3 (7c:67:a2:bb:5f:a3), Dst: f2:e7:81:d9:8e:18 (f2:e7:81:d9:8e:18)

> Internet Protocol Version 4, Src: 192.168.43.218, Dst: 192.168.43.1

> User Datagram Protocol, Src Port: 64862, Dst Port: 53

> Domain Name System (query)

8. Filter by MAC address

If you only want to see traffic for a specific MAC address use this filter.

eth.addr == 00:60:e0:53:13:d5

Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

eth.addr == 7c:67:a2:bb:5f:a3

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	74.125.200.188	192.168.43.218	TCP	66	5228 → 2055 [ACK] Seq=1 Ack=1 Win=265 Len=0 SLE=0 SRE=1
2	3.348704	192.168.43.218	157.240.198.60	TLSv1.2	85	Application Data
3	3.360735	f2:e7:81:d9:8e:18	IntelCor_bb:5f:a3	ARP	42	Who has 192.168.43.218? Tell 192.168.43.1
4	3.360768	IntelCor_bb:5f:a3	f2:e7:81:d9:8e:18	ARP	42	192.168.43.218 is at 7c:67:a2:bb:5f:a3
5	3.385774	157.240.198.60	192.168.43.218	TCP	54	443 → 1958 [ACK] Seq=1 Ack=32 Win=21 Len=0
6	3.685618	157.240.198.60	192.168.43.218	TLSv1.2	92	Application Data
7	3.725974	192.168.43.218	157.240.198.60	TCP	54	1958 → 443 [ACK] Seq=32 Ack=39 Win=253 Len=0
8	4.388828	IntelCor_bb:5f:a3	Broadcast	ARP	42	Who has 192.168.43.1? Tell 192.168.43.218
9	5.441120	f2:e7:81:d9:8e:18	IntelCor_bb:5f:a3	ARP	42	192.168.43.1 is at f2:e7:81:d9:8e:18
10	15.117391	192.168.43.218	157.240.198.60	TLSv1.2	85	Application Data
11	15.180097	157.240.198.60	192.168.43.218	TCP	54	443 → 1958 [ACK] Seq=39 Ack=63 Win=21 Len=0
12	15.462120	157.240.198.60	192.168.43.218	TLSv1.2	92	Application Data
13	15.485544	IntelCor_bb:5f:a3	Broadcast	ARP	42	Who has 192.168.43.1? Tell 192.168.43.218
14	15.487402	f2:e7:81:d9:8e:18	IntelCor_bb:5f:a3	ARP	42	192.168.43.1 is at f2:e7:81:d9:8e:18

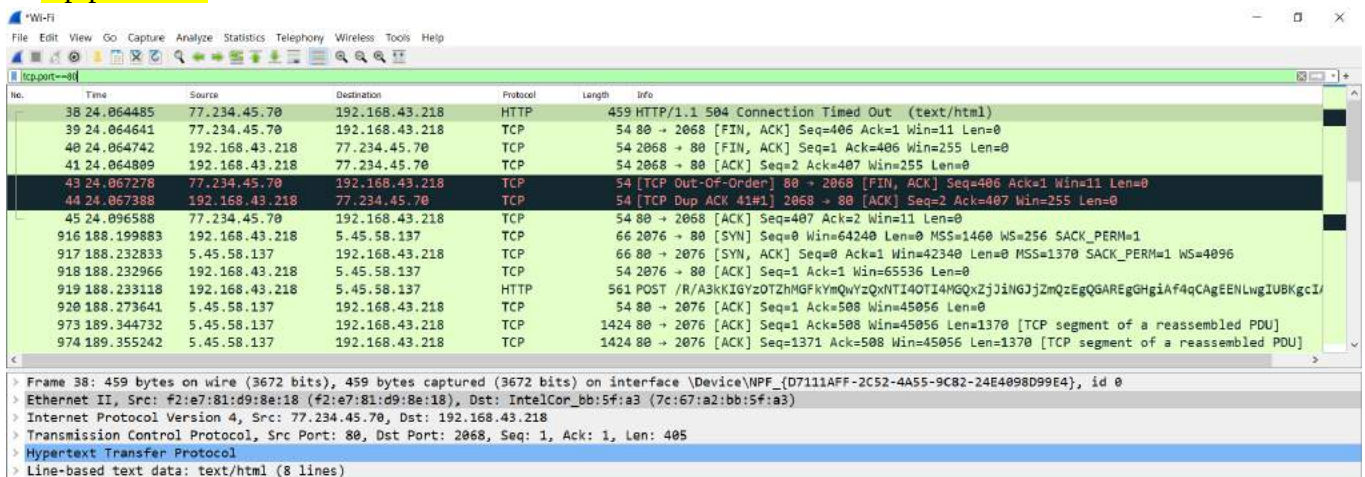
> Frame 3: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface \Device\NPF_{D7111AFF-2C52-4A55-9C82-24E4098D99E4}, id 0

> Ethernet II, Src: f2:e7:81:d9:8e:18 (f2:e7:81:d9:8e:18), Dst: IntelCor_bb:5f:a3 (7c:67:a2:bb:5f:a3)

> Address Resolution Protocol (request)

9. Filter on TCP port

tcp.port==80

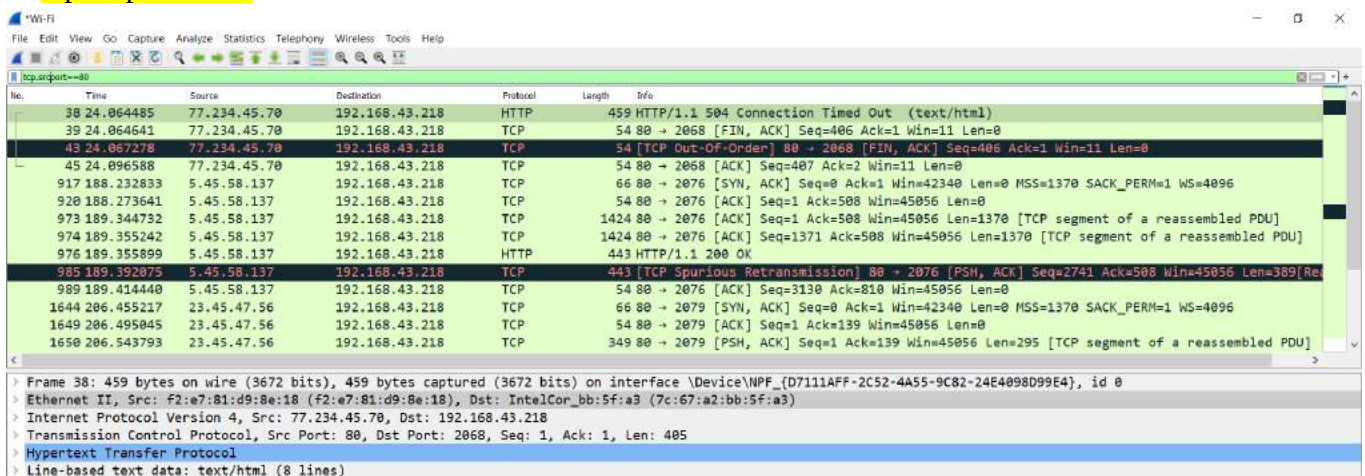


The screenshot shows the Wireshark interface with the filter 'tcp.port==80' applied. The packet list displays various TCP and HTTP packets. The selected packet (No. 43) is a TCP 'Out-Of-Order' segment from 77.234.45.70 to 192.168.43.218, port 80. The packet details pane shows the Ethernet II, Internet Protocol Version 4, and Transmission Control Protocol (TCP) layers. The TCP layer shows 'Out-Of-Order' with sequence number 80 and acknowledgment number 2068.

No.	Time	Source	Destination	Protocol	Length	Info
38	24.064485	77.234.45.70	192.168.43.218	HTTP	459	HTTP/1.1 504 Connection Timed Out (text/html)
39	24.064641	77.234.45.70	192.168.43.218	TCP	54	80 → 2068 [FIN, ACK] Seq=406 Ack=1 Win=11 Len=0
40	24.064742	192.168.43.218	77.234.45.70	TCP	54	2068 → 80 [FIN, ACK] Seq=1 Ack=406 Win=255 Len=0
41	24.064809	192.168.43.218	77.234.45.70	TCP	54	2068 → 80 [ACK] Seq=2 Ack=407 Win=255 Len=0
43	24.067278	77.234.45.70	192.168.43.218	TCP	54	[TCP Out-Of-Order] 80 → 2068 [FIN, ACK] Seq=406 Ack=1 Win=11 Len=0
44	24.067388	192.168.43.218	77.234.45.70	TCP	54	[TCP Dup ACK 41#1] 2068 → 80 [ACK] Seq=2 Ack=407 Win=255 Len=0
45	24.096588	77.234.45.70	192.168.43.218	TCP	54	80 → 2068 [ACK] Seq=407 Ack=2 Win=11 Len=0
916	188.199883	192.168.43.218	5.45.58.137	TCP	66	2076 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
917	188.232833	5.45.58.137	192.168.43.218	TCP	66	80 → 2076 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1370 SACK_PERM=1 WS=4096
918	188.232966	192.168.43.218	5.45.58.137	TCP	54	2076 → 80 [ACK] Seq=1 Ack=1 Win=65536 Len=0
919	188.233118	192.168.43.218	5.45.58.137	HTTP	561	POST /R/A3kKIGYzOTZhmGfKymQwYzQxNTI4OTI4MGQxZjJiNGJjZmQzEgQsAREgGHgIAf4qCAgEENLwgIUBKgcIA/
920	188.273641	5.45.58.137	192.168.43.218	TCP	54	80 → 2076 [ACK] Seq=1 Ack=508 Win=45056 Len=0
973	189.344732	5.45.58.137	192.168.43.218	TCP	1424	80 → 2076 [ACK] Seq=1 Ack=508 Win=45056 Len=1370 [TCP segment of a reassembled PDU]
974	189.355242	5.45.58.137	192.168.43.218	TCP	1424	80 → 2076 [ACK] Seq=1371 Ack=508 Win=45056 Len=1370 [TCP segment of a reassembled PDU]

Filter on TCP port source

tcp.srcport==80

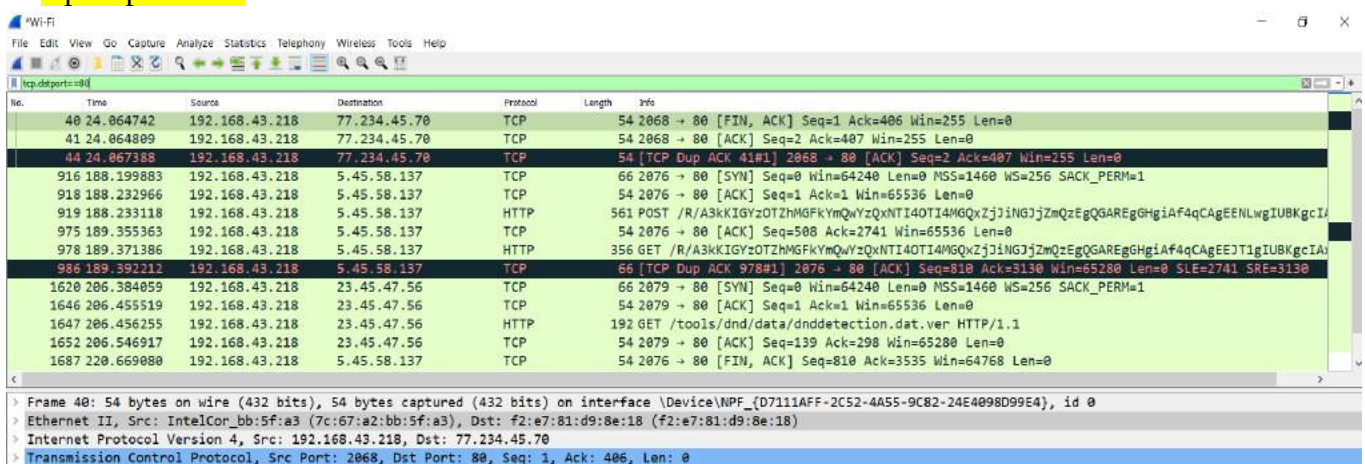


The screenshot shows the Wireshark interface with the filter 'tcp.srcport==80' applied. The packet list displays various TCP and HTTP packets. The selected packet (No. 43) is a TCP 'Out-Of-Order' segment from 77.234.45.70 to 192.168.43.218, port 80. The packet details pane shows the Ethernet II, Internet Protocol Version 4, and Transmission Control Protocol (TCP) layers. The TCP layer shows 'Out-Of-Order' with sequence number 80 and acknowledgment number 2068.

No.	Time	Source	Destination	Protocol	Length	Info
38	24.064485	77.234.45.70	192.168.43.218	HTTP	459	HTTP/1.1 504 Connection Timed Out (text/html)
39	24.064641	77.234.45.70	192.168.43.218	TCP	54	80 → 2068 [FIN, ACK] Seq=406 Ack=1 Win=11 Len=0
43	24.067278	77.234.45.70	192.168.43.218	TCP	54	[TCP Out-Of-Order] 80 → 2068 [FIN, ACK] Seq=406 Ack=1 Win=11 Len=0
45	24.096588	77.234.45.70	192.168.43.218	TCP	54	80 → 2068 [ACK] Seq=407 Ack=2 Win=11 Len=0
917	188.232833	5.45.58.137	192.168.43.218	TCP	66	80 → 2076 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1370 SACK_PERM=1 WS=4096
920	188.273641	5.45.58.137	192.168.43.218	TCP	54	80 → 2076 [ACK] Seq=1 Ack=508 Win=45056 Len=0
973	189.344732	5.45.58.137	192.168.43.218	TCP	1424	80 → 2076 [ACK] Seq=1 Ack=508 Win=45056 Len=1370 [TCP segment of a reassembled PDU]
974	189.355242	5.45.58.137	192.168.43.218	TCP	1424	80 → 2076 [ACK] Seq=1371 Ack=508 Win=45056 Len=1370 [TCP segment of a reassembled PDU]
985	189.392075	5.45.58.137	192.168.43.218	TCP	443	[TCP Spurious Retransmission] 80 → 2076 [PSH, ACK] Seq=2741 Ack=508 Win=45056 Len=389[Re
989	189.414440	5.45.58.137	192.168.43.218	TCP	54	80 → 2076 [ACK] Seq=3130 Ack=810 Win=45056 Len=0
1644	206.455217	23.45.47.56	192.168.43.218	TCP	66	80 → 2079 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1370 SACK_PERM=1 WS=4096
1649	206.495045	23.45.47.56	192.168.43.218	TCP	54	80 → 2079 [ACK] Seq=1 Ack=139 Win=45056 Len=0
1650	206.543793	23.45.47.56	192.168.43.218	TCP	349	80 → 2079 [PSH, ACK] Seq=1 Ack=139 Win=45056 Len=295 [TCP segment of a reassembled PDU]

or destination port

tcp.dstport==80



The screenshot shows the Wireshark interface with the filter 'tcp.dstport==80' applied. The packet list displays various TCP and HTTP packets. The selected packet (No. 40) is a TCP segment from 192.168.43.218 to 77.234.45.70, port 80. The packet details pane shows the Ethernet II, Internet Protocol Version 4, and Transmission Control Protocol (TCP) layers. The TCP layer shows a segment with sequence number 2068 and acknowledgment number 406.

No.	Time	Source	Destination	Protocol	Length	Info
40	24.064742	192.168.43.218	77.234.45.70	TCP	54	2068 → 80 [FIN, ACK] Seq=1 Ack=406 Win=255 Len=0
41	24.064809	192.168.43.218	77.234.45.70	TCP	54	2068 → 80 [ACK] Seq=2 Ack=407 Win=255 Len=0
44	24.067388	192.168.43.218	77.234.45.70	TCP	54	[TCP Dup ACK 41#1] 2068 → 80 [ACK] Seq=2 Ack=407 Win=255 Len=0
916	188.199883	192.168.43.218	5.45.58.137	TCP	66	2076 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
918	188.232966	192.168.43.218	5.45.58.137	TCP	54	2076 → 80 [ACK] Seq=1 Ack=1 Win=65536 Len=0
919	188.233118	192.168.43.218	5.45.58.137	HTTP	561	POST /R/A3kKIGYzOTZhmGfKymQwYzQxNTI4OTI4MGQxZjJiNGJjZmQzEgQsAREgGHgIAf4qCAgEENLwgIUBKgcIA/
975	189.355363	192.168.43.218	5.45.58.137	TCP	54	2076 → 80 [ACK] Seq=508 Ack=2741 Win=65536 Len=0
978	189.371386	192.168.43.218	5.45.58.137	HTTP	356	GET /R/A3kKIGYzOTZhmGfKymQwYzQxNTI4OTI4MGQxZjJiNGJjZmQzEgQsAREgGHgIAf4qCAgEETIgIUBKgcIA/
986	189.392112	192.168.43.218	5.45.58.137	TCP	66	[TCP Dup ACK 978#1] 2076 → 80 [ACK] Seq=810 Ack=3130 Win=65280 Len=0 SLE=2741 SRE=3130
1620	206.384059	192.168.43.218	23.45.47.56	TCP	66	2079 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
1646	206.455519	192.168.43.218	23.45.47.56	TCP	54	2079 → 80 [ACK] Seq=1 Ack=1 Win=65536 Len=0
1647	206.456255	192.168.43.218	23.45.47.56	HTTP	192	GET /tools/dnd/data/dnndetection.dat.ver HTTP/1.1
1652	206.546917	192.168.43.218	23.45.47.56	TCP	54	2079 → 80 [ACK] Seq=139 Ack=298 Win=65280 Len=0
1687	220.669880	192.168.43.218	5.45.58.137	TCP	54	2076 → 80 [FIN, ACK] Seq=810 Ack=3535 Win=64768 Len=0

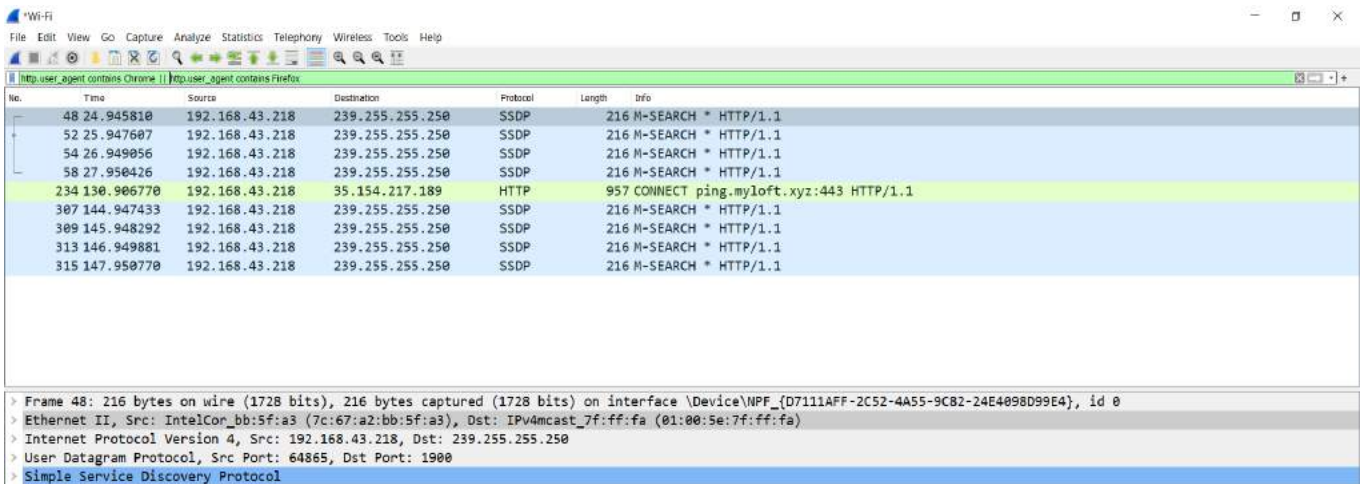
10. Find user agents

It's a good idea to understand what user agents are being used on your network, malicious traffic can often use unusual agent strings. To search for a user agent use this filter.

http.user_agent contains Firefox

Replace Firefox with the user agent string you want to search for. I like to use this one to exclude common user agents, this helps to quickly find possible malicious traffic.

http.user_agent contains Firefox || http.user_agent contains Chrome



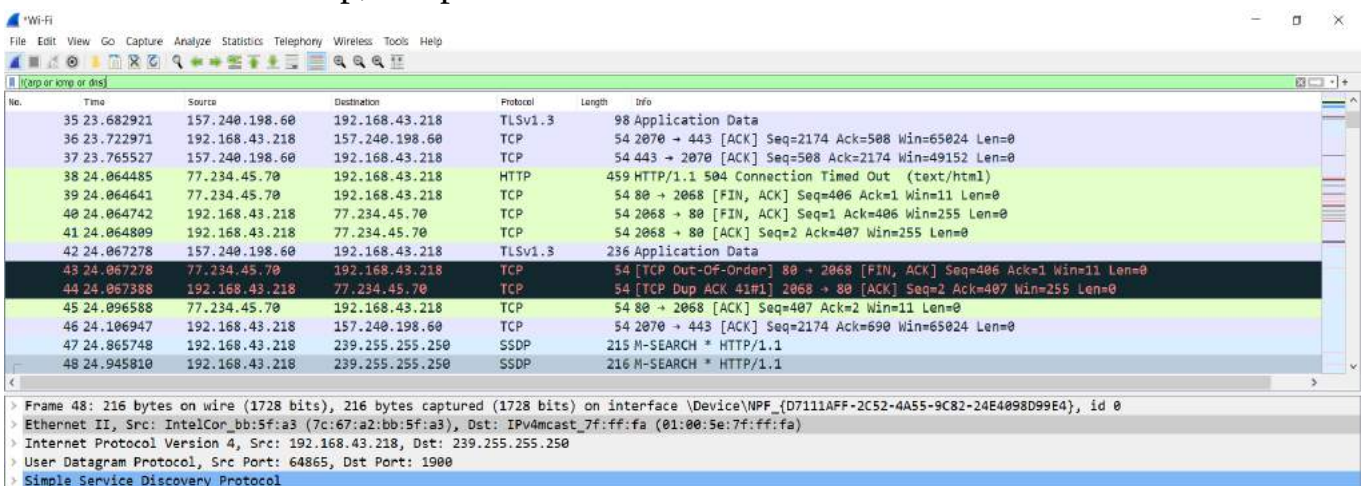
This will filter out all user agents that contain Firefox or Chrome, I can continue to add on to the filter to exclude other common user agents.

11. Filter background network noise

There are several protocols that can be very noisy, it sometimes helps to filter this out so you can focus on other traffic.

!(arp or icmp or dns)

This will filter out arp, icmp and DNS traffic.

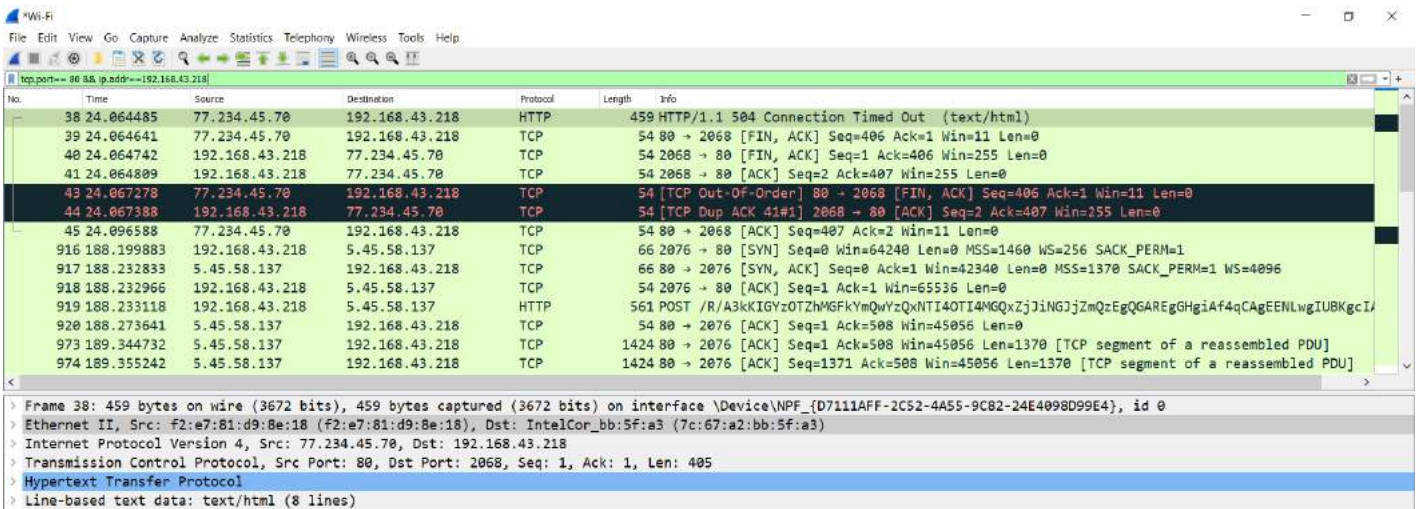


12. Filter on port and IP Address

If you want to see traffic from a certain IP on a specific port use this filter

tcp.port 80 && ip.addr == 192.168.43.218

This will show only port 80 (https) that has IP 192.168.43.218 in the source or destination.

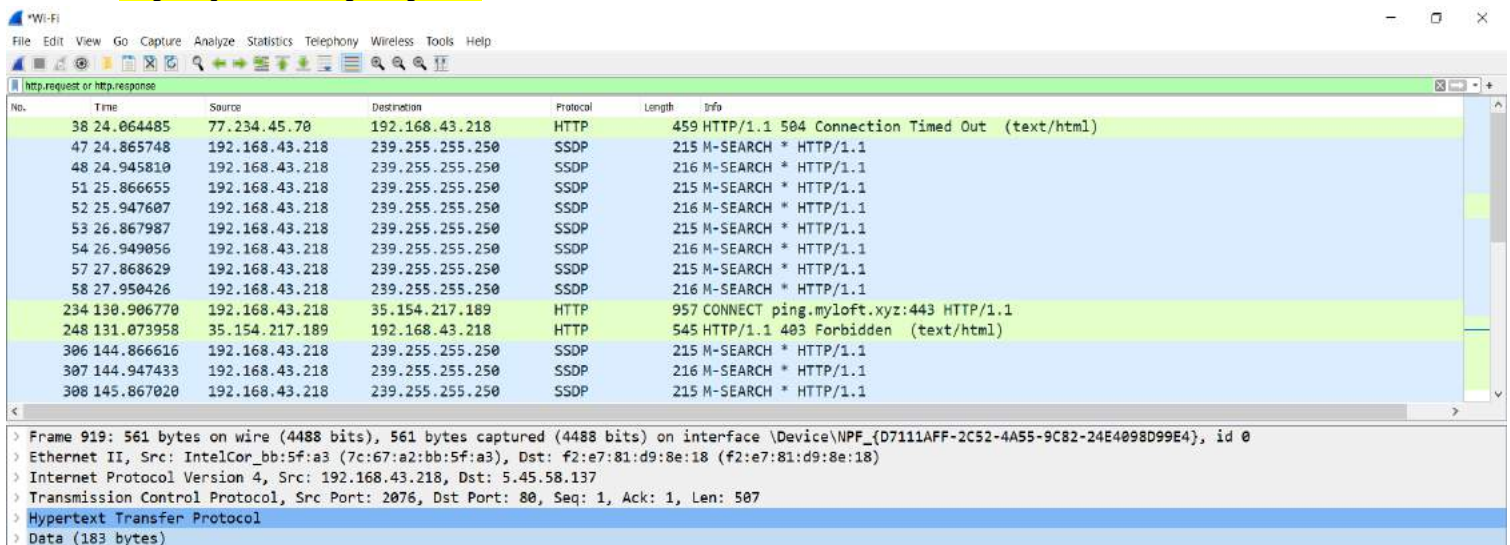


No.	Time	Source	Destination	Protocol	Length	Info
38	24.064485	77.234.45.70	192.168.43.218	HTTP	459	HTTP/1.1 504 Connection Timed Out (text/html)
39	24.064641	77.234.45.70	192.168.43.218	TCP	54	80 → 2068 [FIN, ACK] Seq=406 Ack=1 Win=11 Len=0
40	24.064742	192.168.43.218	77.234.45.70	TCP	54	2068 → 80 [FIN, ACK] Seq=1 Ack=406 Win=255 Len=0
41	24.064809	192.168.43.218	77.234.45.70	TCP	54	2068 → 80 [ACK] Seq=2 Ack=407 Win=255 Len=0
43	24.067278	77.234.45.70	192.168.43.218	TCP	54	[TCP Out-Of-Order] 80 → 2068 [FIN, ACK] Seq=406 Ack=1 Win=11 Len=0
44	24.067388	192.168.43.218	77.234.45.70	TCP	54	[TCP Dup ACK 41#1] 2068 → 80 [ACK] Seq=2 Ack=407 Win=255 Len=0
45	24.096588	77.234.45.70	192.168.43.218	TCP	54	80 → 2068 [ACK] Seq=407 Ack=2 Win=11 Len=0
916	188.199883	192.168.43.218	5.45.58.137	TCP	66	2076 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
917	188.232833	5.45.58.137	192.168.43.218	TCP	66	80 → 2076 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1370 SACK_PERM=1 WS=4096
918	188.232966	192.168.43.218	5.45.58.137	TCP	54	2076 → 80 [ACK] Seq=1 Ack=1 Win=65536 Len=0
919	188.233118	192.168.43.218	5.45.58.137	HTTP	561	POST /R/A3kKIGYzOTZHMgFKYmQwVzQxNTI4OTI4MGQxZjJlNGJjZmQzEgQGAREgGHgiAf4qCAGeENLwgIUBKgcI/
920	188.273641	5.45.58.137	192.168.43.218	TCP	54	80 → 2076 [ACK] Seq=1 Ack=508 Win=45056 Len=0
973	189.344732	5.45.58.137	192.168.43.218	TCP	1424	80 → 2076 [ACK] Seq=1 Ack=508 Win=45056 Len=1370 [TCP segment of a reassembled PDU]
974	189.355242	5.45.58.137	192.168.43.218	TCP	1424	80 → 2076 [ACK] Seq=1371 Ack=508 Win=45056 Len=1370 [TCP segment of a reassembled PDU]

> Frame 38: 459 bytes on wire (3672 bits), 459 bytes captured (3672 bits) on interface \Device\NPF_{D7111AFF-2C52-4A55-9C82-24E4098D99E4}, id 0
> Ethernet II, Src: f2:e7:81:d9:8e:18 (f2:e7:81:d9:8e:18), Dst: IntelCor_bb:5f:a3 (7c:67:a2:bb:5f:a3)
> Internet Protocol Version 4, Src: 77.234.45.70, Dst: 192.168.43.218
> Transmission Control Protocol, Src Port: 80, Dst Port: 2068, Seq: 1, Ack: 1, Len: 405
> Hypertext Transfer Protocol
> Line-based text data: text/html (8 lines)

13. Filter for http get and responses

http.request or http.response



No.	Time	Source	Destination	Protocol	Length	Info
38	24.064485	77.234.45.70	192.168.43.218	HTTP	459	HTTP/1.1 504 Connection Timed Out (text/html)
47	24.865748	192.168.43.218	239.255.255.250	SSDP	215	M-SEARCH * HTTP/1.1
48	24.945810	192.168.43.218	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
51	25.866655	192.168.43.218	239.255.255.250	SSDP	215	M-SEARCH * HTTP/1.1
52	25.947607	192.168.43.218	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
53	26.867987	192.168.43.218	239.255.255.250	SSDP	215	M-SEARCH * HTTP/1.1
54	26.949056	192.168.43.218	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
57	27.868629	192.168.43.218	239.255.255.250	SSDP	215	M-SEARCH * HTTP/1.1
58	27.950426	192.168.43.218	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
234	130.906770	192.168.43.218	35.154.217.189	HTTP	957	CONNECT ping.myloft.xyz:443 HTTP/1.1
248	131.073958	35.154.217.189	192.168.43.218	HTTP	545	HTTP/1.1 403 Forbidden (text/html)
306	144.866616	192.168.43.218	239.255.255.250	SSDP	215	M-SEARCH * HTTP/1.1
307	144.947433	192.168.43.218	239.255.255.250	SSDP	216	M-SEARCH * HTTP/1.1
308	145.867020	192.168.43.218	239.255.255.250	SSDP	215	M-SEARCH * HTTP/1.1

> Frame 919: 561 bytes on wire (4488 bits), 561 bytes captured (4488 bits) on interface \Device\NPF_{D7111AFF-2C52-4A55-9C82-24E4098D99E4}, id 0
> Ethernet II, Src: IntelCor_bb:5f:a3 (7c:67:a2:bb:5f:a3), Dst: f2:e7:81:d9:8e:18 (f2:e7:81:d9:8e:18)
> Internet Protocol Version 4, Src: 192.168.43.218, Dst: 5.45.58.137
> Transmission Control Protocol, Src Port: 2076, Dst Port: 80, Seq: 1, Ack: 1, Len: 507
> Hypertext Transfer Protocol
> Data (183 bytes)

15. Filter on three way handshake

The three way handshake is often used to calculate the network round trip time. This filter will display all the SYN, SYN ACK and SYN packets that should match the three way handshake.

tcp.flags.syn==1 or (tcp.seq==1 and tcp.ack==1 and tcp.len==0 and tcp.analysis.initial_rtt)

Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp.flags.syn==1 or (tcp.seq==1 and tcp.ack==1 and tcp.len==0 and tcp.analysis.initial_rtt)

No.	Time	Source	Destination	Protocol	Length	Info
135	57.973720	35.154.217.189	192.168.43.218	TCP	66	443 → 2072 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1370 SACK_PERM=1 WS=4096
136	57.973878	192.168.43.218	35.154.217.189	TCP	54	2072 → 443 [ACK] Seq=1 Ack=1 Win=65536 Len=0
173	75.184869	192.168.43.218	5.62.54.29	TCP	66	2073 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
174	75.224506	5.62.54.29	192.168.43.218	TCP	66	443 → 2073 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1370 SACK_PERM=1 WS=4096
175	75.224610	192.168.43.218	5.62.54.29	TCP	54	2073 → 443 [ACK] Seq=1 Ack=1 Win=65536 Len=0
230	130.867967	192.168.43.218	35.154.217.189	TCP	66	2074 → 4500 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
232	130.904202	35.154.217.189	192.168.43.218	TCP	66	4500 → 2074 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1370 SACK_PERM=1 WS=4096
233	130.904355	192.168.43.218	35.154.217.189	TCP	54	2074 → 4500 [ACK] Seq=1 Ack=1 Win=65536 Len=0
237	130.949399	192.168.43.218	34.199.143.64	TCP	66	2075 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
239	130.985999	34.199.143.64	192.168.43.218	TCP	66	443 → 2075 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1370 SACK_PERM=1 WS=4096
240	130.986121	192.168.43.218	34.199.143.64	TCP	54	2075 → 443 [ACK] Seq=1 Ack=1 Win=65536 Len=0
916	188.199883	192.168.43.218	5.45.58.137	TCP	66	2076 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
917	188.232833	5.45.58.137	192.168.43.218	TCP	66	80 → 2076 [SYN, ACK] Seq=0 Ack=1 Win=42340 Len=0 MSS=1370 SACK_PERM=1 WS=4096
918	188.232966	192.168.43.218	5.45.58.137	TCP	54	2076 → 80 [ACK] Seq=1 Ack=1 Win=65536 Len=0

> Frame 918: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface \Device\NPF_{D7111AFF-2C52-4A55-9C82-24E4098D99E4}, id 0

> Ethernet II, Src: IntelCor_bb:5f:a3 (7c:67:a2:bb:5f:a3), Dst: f2:e7:81:d9:8e:18 (f2:e7:81:d9:8e:18)

> Internet Protocol Version 4, Src: 192.168.43.218, Dst: 5.45.58.137

> Transmission Control Protocol, Src Port: 2076, Dst Port: 80, Seq: 1, Ack: 1, Len: 0

16. Find executable or other file types

Need to see if users are download .exe or other file types use this filter

frame contains jpg or exec or zip

Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

frame contains jpg or exec or zip

No.	Time	Source	Destination	Protocol	Length	Info
11..	59.380407	35.208.0.58	192.168.0.106	HTTP	783	HTTP/1.1 200 OK (text/html)
11..	59.692693	192.168.0.106	35.208.0.58	HTTP	367	GET /images/netlab-labelled-pod1.jpg HTTP/1.1
14..	84.151114	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=2825 Ack=337 Win=29696 Len=1412 [TCP segment of a reassembled PDU]
14..	84.151114	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=12709 Ack=337 Win=29696 Len=1412 [TCP segment of a reassembled PDU]
14..	84.767240	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=59305 Ack=337 Win=29696 Len=1412 [TCP segment of a reassembled PDU]
15..	84.769268	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=88957 Ack=337 Win=29696 Len=1412 [TCP segment of a reassembled PDU]
17..	86.330470	192.168.0.106	35.208.0.58	HTTP	369	GET /images/DansCoursesLogo_100x100.jpg HTTP/1.1
16..	96.832638	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=133023 Ack=1552 Win=32768 Len=1412 [TCP segment of a reassembled PDU]
16..	96.833641	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=142907 Ack=1552 Win=32768 Len=1412 [TCP segment of a reassembled PDU]
16..	96.851525	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=189583 Ack=1552 Win=32768 Len=1412 [TCP segment of a reassembled PDU]
16..	96.859256	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=219235 Ack=1552 Win=32768 Len=1412 [TCP segment of a reassembled PDU]
18..	104.740399	35.208.0.58	192.168.0.106	TCP	1466	80 → 4395 [ACK] Seq=45279 Ack=1600 Win=32768 Len=1412 [TCP segment of a reassembled PDU]
18..	104.742023	35.208.0.58	192.168.0.106	TCP	1466	80 → 4395 [ACK] Seq=55163 Ack=1600 Win=32768 Len=1412 [TCP segment of a reassembled PDU]
18..	104.744420	35.208.0.58	192.168.0.106	TCP	1466	80 → 4395 [ACK] Seq=79240 Ack=1600 Win=32768 Len=1412 [TCP segment of a reassembled PDU]

> Frame 1432: 1466 bytes on wire (11728 bits), 1466 bytes captured (11728 bits) on interface \Device\NPF_{D7111AFF-2C52-4A55-9C82-24E4098D99E4}, id 0

> Ethernet II, Src: Tp-LinkT_32:16:34 (c4:71:54:32:16:34), Dst: IntelCor_bb:5f:a3 (7c:67:a2:bb:5f:a3)

> Internet Protocol Version 4, Src: 35.208.0.58, Dst: 192.168.0.106

> Transmission Control Protocol, Src Port: 80, Dst Port: 4382, Seq: 2825, Ack: 337, Len: 1412

Source Port: 80

Destination Port: 4382

[Stream index: 43]

[TCP Segment Len: 1412]

Sequence Number: 2825 (relative sequence number)

Sequence Number (raw): 3158065434

[Next Sequence Number: 4237 (relative sequence number)]

Acknowledgment Number: 337 (relative ack number)

Just add in any other file extension you want to filter for.

17. Search traffic based on a keyword

tcp contains danscourses.com

*Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp contains danscourses.com

No.	Time	Source	Destination	Protocol	Length	Info
14...	82.978941	192.168.0.106	35.208.0.58	HTTP	390	GET / HTTP/1.1
14...	84.151114	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=1 Ack=337 Win=29696 Len=1412 [TCP segment of a reassembled PDU]
14...	84.151114	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=1413 Ack=337 Win=29696 Len=1412 [TCP segment of a reassembled PDU]
14...	84.151114	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=2825 Ack=337 Win=29696 Len=1412 [TCP segment of a reassembled PDU]
14...	84.151114	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=5649 Ack=337 Win=29696 Len=1412 [TCP segment of a reassembled PDU]
14...	84.151114	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=8473 Ack=337 Win=29696 Len=1412 [TCP segment of a reassembled PDU]
14...	84.151114	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=9885 Ack=337 Win=29696 Len=1412 [TCP segment of a reassembled PDU]
14...	84.151114	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=11297 Ack=337 Win=29696 Len=1412 [TCP segment of a reassembled PDU]
14...	84.151114	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=12709 Ack=337 Win=29696 Len=1412 [TCP segment of a reassembled PDU]
14...	84.409407	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=14121 Ack=337 Win=29696 Len=1412 [TCP segment of a reassembled PDU]
14...	84.409407	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=15533 Ack=337 Win=29696 Len=1412 [TCP segment of a reassembled PDU]
14...	84.409407	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=16945 Ack=337 Win=29696 Len=1412 [TCP segment of a reassembled PDU]
14...	84.410022	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=18357 Ack=337 Win=29696 Len=1412 [TCP segment of a reassembled PDU]
14...	84.410022	35.208.0.58	192.168.0.106	TCP	1466	80 → 4382 [ACK] Seq=19769 Ack=337 Win=29696 Len=1412 [TCP segment of a reassembled PDU]

> Frame 1432: 1466 bytes on wire (11728 bits), 1466 bytes captured (11728 bits) on interface \Device\NPF_{D7111AFF-2C52-4A55-9C82-24E4098D99E4}, id 0

> Ethernet II, Src: Tp-LinkT_32:16:34 (c4:71:54:32:16:34), Dst: IntelCor_bb:5f:a3 (7c:67:a2:bb:5f:a3)

> Internet Protocol Version 4, Src: 35.208.0.58, Dst: 192.168.0.106

> Transmission Control Protocol, Src Port: 80, Dst Port: 4382, Seq: 2825, Ack: 337, Len: 1412

Source Port: 80

Destination Port: 4382

[Stream index: 43]

[TCP Segment Len: 1412]

Sequence Number: 2825 (relative sequence number)

Sequence Number (raw): 3158065434

[Next Sequence Number: 4237 (relative sequence number)]

Acknowledgment Number: 337 (relative ack number)

This displays all TCP packets that contain the word danscourse. Just replace the word with want you want to search for. The only problem with this filter is it's limited to TCP packets only. To include all protocols use this filter.

frame contains danscourses.com

*Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

frame contains danscourses.com

No.	Time	Source	Destination	Protocol	Length	Info
15...	85.029744	35.208.0.58	192.168.0.106	HTTP	1203	HTTP/1.1 200 OK (text/html)
15...	85.070735	192.168.0.106	35.208.0.58	HTTP	419	GET /wp-content/plugins/jetpack/modules/theme-tools/compat/twentyseventeen.css?ver=9.0.2 HTTP/1.1
15...	85.112569	192.168.0.106	35.208.0.58	HTTP	396	GET /wp-includes/css/dist/block-library/style.min.css?ver=5.5.3 HTTP/1.1
15...	85.114559	192.168.0.106	35.208.0.58	HTTP	396	GET /wp-includes/css/dist/block-library/theme.min.css?ver=5.5.3 HTTP/1.1
16...	85.371410	192.168.0.106	35.208.0.58	HTTP	403	GET /wp-content/plugins/contact-form-7/includes/css/styles.css?ver=5.3 HTTP/1.1
16...	85.412688	192.168.0.106	35.208.0.58	HTTP	401	GET /wp-content/plugins/widget-options/assets/css/widget-options.css HTTP/1.1
16...	85.412879	192.168.0.106	35.208.0.58	HTTP	414	GET /wp-content/plugins/jetpack/_inc/genericons/genericons/genericons.css?ver=3.1 HTTP/1.1
16...	85.412994	192.168.0.106	35.208.0.58	HTTP	398	GET /wp-content/themes/twentyseventeen-child/style.css?ver=20190507 HTTP/1.1
16...	85.420204	192.168.0.106	35.208.0.58	HTTP	397	GET /wp-content/themes/twentyseventeen/css/blocks.css?ver=20190102 HTTP/1.1
16...	85.437017	192.168.0.106	35.208.0.58	HTTP	390	GET /wp-content/plugins/jetpack/css/jetpack.css?ver=9.0.2 HTTP/1.1
16...	85.655168	192.168.0.106	35.208.0.58	HTTP	409	GET /wp-content/plugins/google-analytics-for-wordpress/assets/js/frontend.min.js?ver=7.12.3 HTTP/1.1
16...	85.686779	192.168.0.106	35.208.0.58	HTTP	368	GET /wp-includes/js/jquery/jquery.js?ver=1.12.4-wp HTTP/1.1
17...	86.330470	192.168.0.106	35.208.0.58	HTTP	369	GET /images/DansCoursesLogo_100x100.jpg HTTP/1.1
18...	87.984947	192.168.0.106	104.16.221.29	HTTP	330	GET /js HTTP/1.1

> Frame 1432: 1466 bytes on wire (11728 bits), 1466 bytes captured (11728 bits) on interface \Device\NPF_{D7111AFF-2C52-4A55-9C82-24E4098D99E4}, id 0

> Ethernet II, Src: Tp-LinkT_32:16:34 (c4:71:54:32:16:34), Dst: IntelCor_bb:5f:a3 (7c:67:a2:bb:5f:a3)

> Internet Protocol Version 4, Src: 35.208.0.58, Dst: 192.168.0.106

> Transmission Control Protocol, Src Port: 80, Dst Port: 4382, Seq: 2825, Ack: 337, Len: 1412

Source Port: 80

Destination Port: 4382

[Stream index: 43]

[TCP Segment Len: 1412]

Sequence Number: 2825 (relative sequence number)

Sequence Number (raw): 3158065434

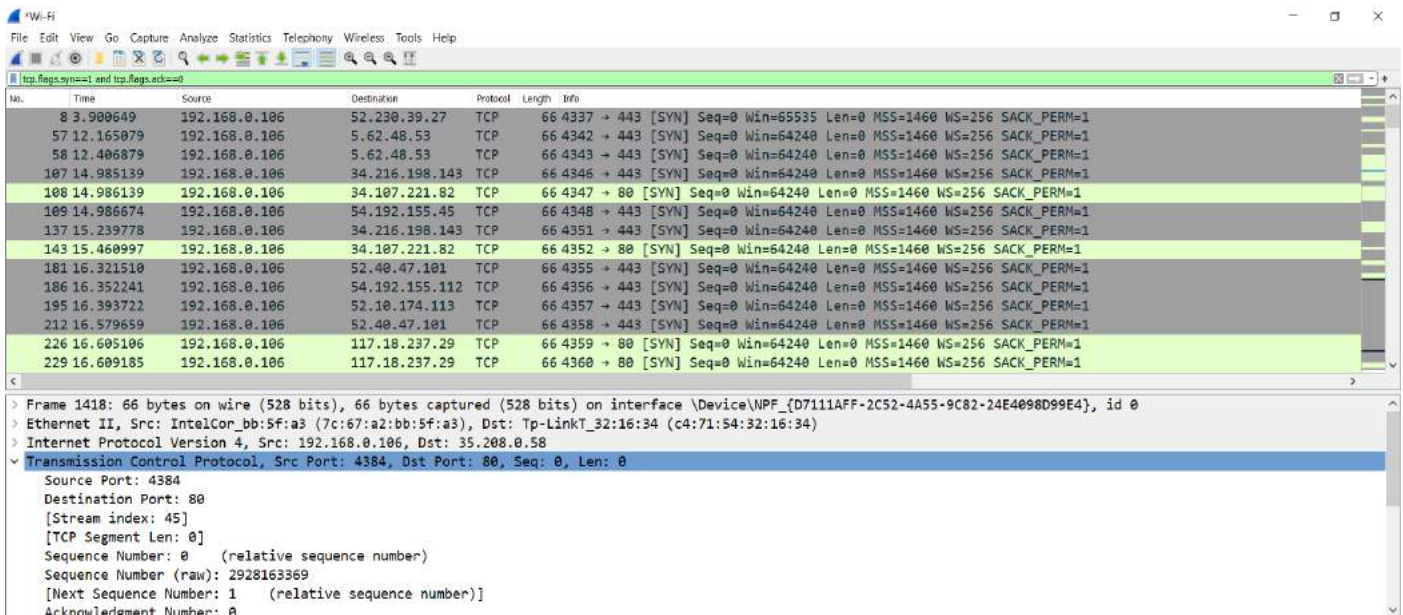
[Next Sequence Number: 4237 (relative sequence number)]

Acknowledgment Number: 337 (relative ack number)

18. Detecting SYN Floods (Possible DDoS attacks)

DDos attacks can be done in a variety of ways, a large number of TCP connections is one of them. To look for a large number of tcp connection attempts use this filter.

tcp.flags.syn == 1 and tcp.flags.ack == 0



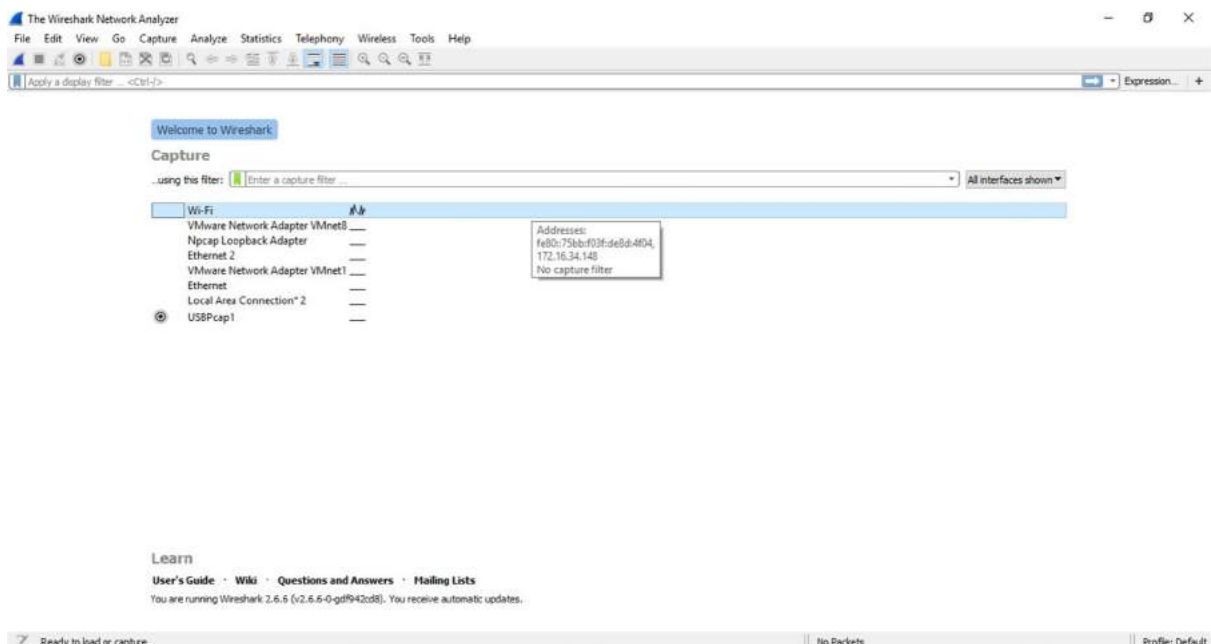
Attack on insecure website's images:

If someone was eavesdropping on your network and sniffing packets, they could see the web pages that you were looking at. More specifically they can extract the packets of information containing the images that you were browsing through on your network.

The following steps are performed to extract the images that the user was looking at:

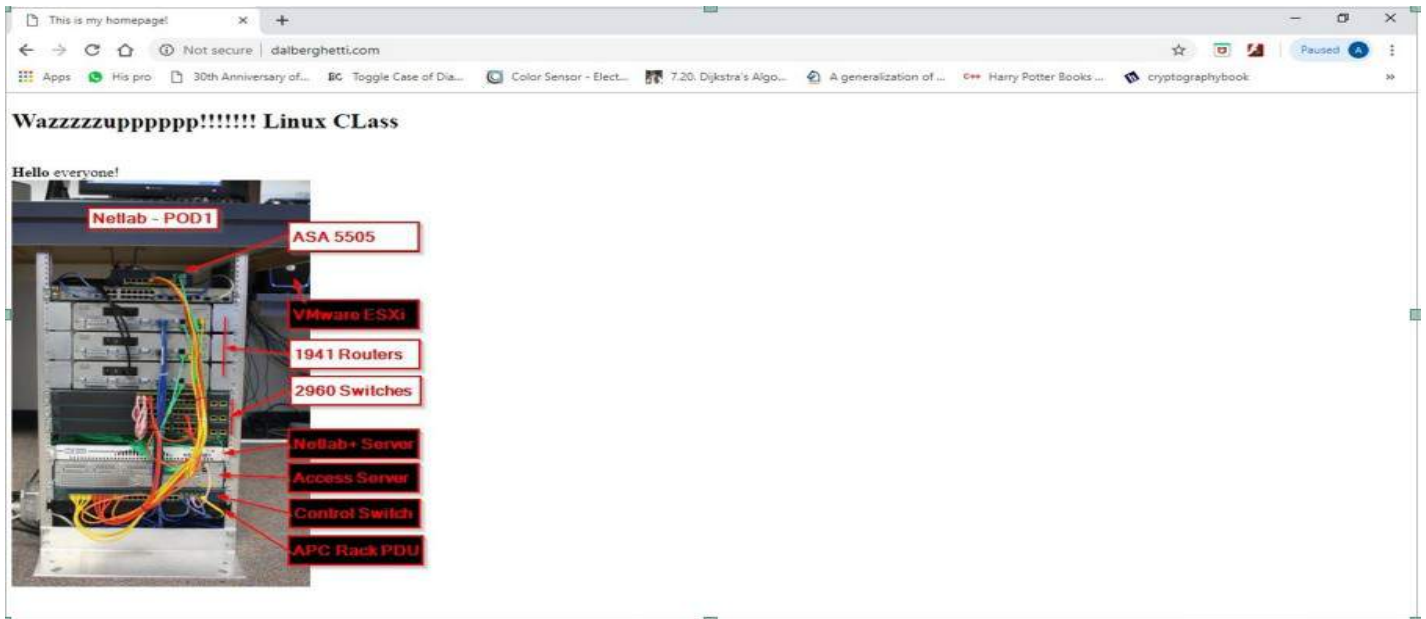
Step 1:

Open the Wireshark application and start capturing interfaces (the wireless network connection in this case).



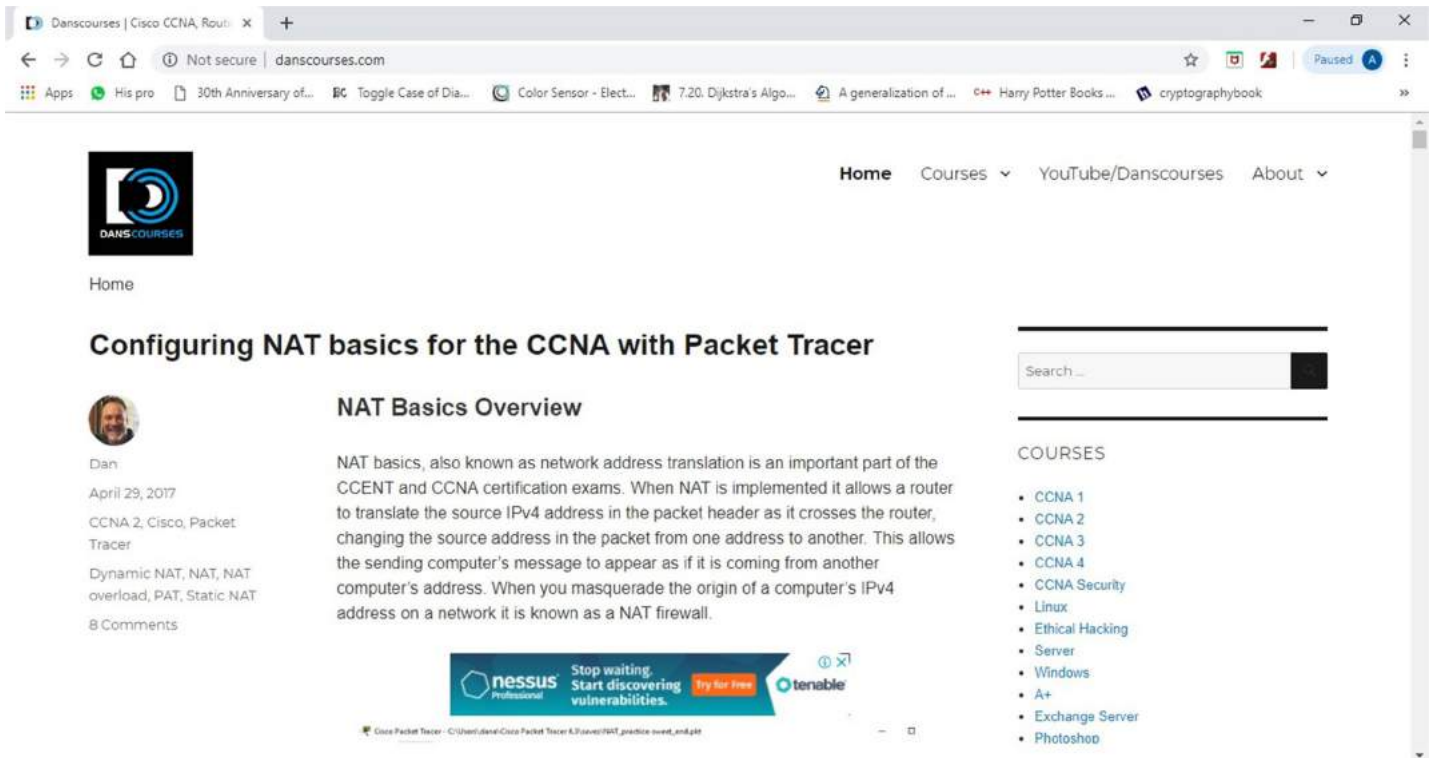
Step 2:

Open up a web browser. In our case open up dalberghetti.com. This website shows us a simple image. Note that this is an insecure web page.



Step 3:

Open up another web page called danscourses.com which is again an insecure website. Browse through this website.



CCNA 1 Archives | Danscourses

Not secure | danscourses.com/category/courses/ccna-1/

Apps His pro 30th Anniversary of... Toggle Case of Dia... Color Sensor - Elect... 7.20, Dijkstra's Algo... A generalization of ... Harry Potter Books ... cryptographybook

Network Troubleshooting PT Activity

Learn More

Step 4:

Stop capturing traffic on Wireshark.

Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http

No.	Time	Source	Destination	Protocol	Length	Info
1002	63.846939	35.208.0.58	192.168.0.107	HTTP	1466	[TCP Previous segment not captured] Continuation
1003	63.846939	35.208.0.58	192.168.0.107	HTTP	1466	Continuation
1006	63.847836	35.208.0.58	192.168.0.107	HTTP	1466	Continuation
1007	63.847836	35.208.0.58	192.168.0.107	HTTP	1466	Continuation
1008	63.847836	35.208.0.58	192.168.0.107	HTTP	1466	Continuation
1017	63.863907	192.0.73.2	192.168.0.107	HTTP	1347	HTTP/1.1 200 OK (JPEG JFIF image)
1024	63.886004	35.208.0.58	192.168.0.107	HTTP	902	HTTP/1.1 200 OK (application/javascript)
1241	63.994540	35.208.0.58	192.168.0.107	HTTP	1466	Continuation
1386	64.156460	35.208.0.58	192.168.0.107	HTTP	1466	Continuation
1387	64.156460	35.208.0.58	192.168.0.107	HTTP	1466	Continuation
1388	64.156460	35.208.0.58	192.168.0.107	HTTP	1466	Continuation
1392	64.156752	35.208.0.58	192.168.0.107	HTTP	1466	Continuation
1393	64.156752	35.208.0.58	192.168.0.107	HTTP	1466	Continuation
1404	64.282351	35.208.0.58	192.168.0.107	HTTP	1466	Continuation
1409	64.461560	35.208.0.58	192.168.0.107	HTTP	1466	Continuation

> Frame 1017: 1347 bytes on wire (10776 bits), 1347 bytes captured (10776 bits) on interface \Device\NPF_{D7111AFF-2C52-4A55-9C82-24E4098D99E4}, id 0

> Ethernet II, Src: Tp-Link_T_32:16:34 (c4:71:54:32:16:34), Dst: IntelCor_bb:5f:a3 (7c:67:a2:bb:5f:a3)

> Internet Protocol Version 4, Src: 192.0.73.2, Dst: 192.168.0.107

> Transmission Control Protocol, Src Port: 80, Dst Port: 51413, Seq: 1413, Ack: 418, Len: 1293

> [2 Reassembled TCP Segments (2705 bytes): #1016(1412), #1017(1293)]

> Hypertext Transfer Protocol

> JPEG File Interchange Format

0200 11111111 1101000 11111111 11100000 00000000 00010000 01001010 01000110JF

0208 01001001 01000110 00000000 00000001 00000001 00000001 00000000 01100000 IF.....

0210 00000000 01100000 00000000 00000000 11111111 11111110 00000000 00111011

0218 01000011 01010010 01000101 01000001 01010100 01001111 01010010 00111010 CREATOR:

0220 00100000 01100111 01100100 00101101 01101010 01110000 01100101 01100111 gd-jpeg

0228 00100000 01110110 00110001 00101110 00110000 00100000 00101000 01110101 v1.0 (u

0230 01110011 01101001 01101110 01100111 00100000 01001001 01001010 01000111 sing IJG

Frame (1347 bytes) Reassembled TCP (2705 bytes)

JPEG File Interchange Format (image-jpeg), 2,193 bytes

Packets: 5128 · Displayed: 132 (2.6%) · Dropped: 0 (0.0%) Profile: Default

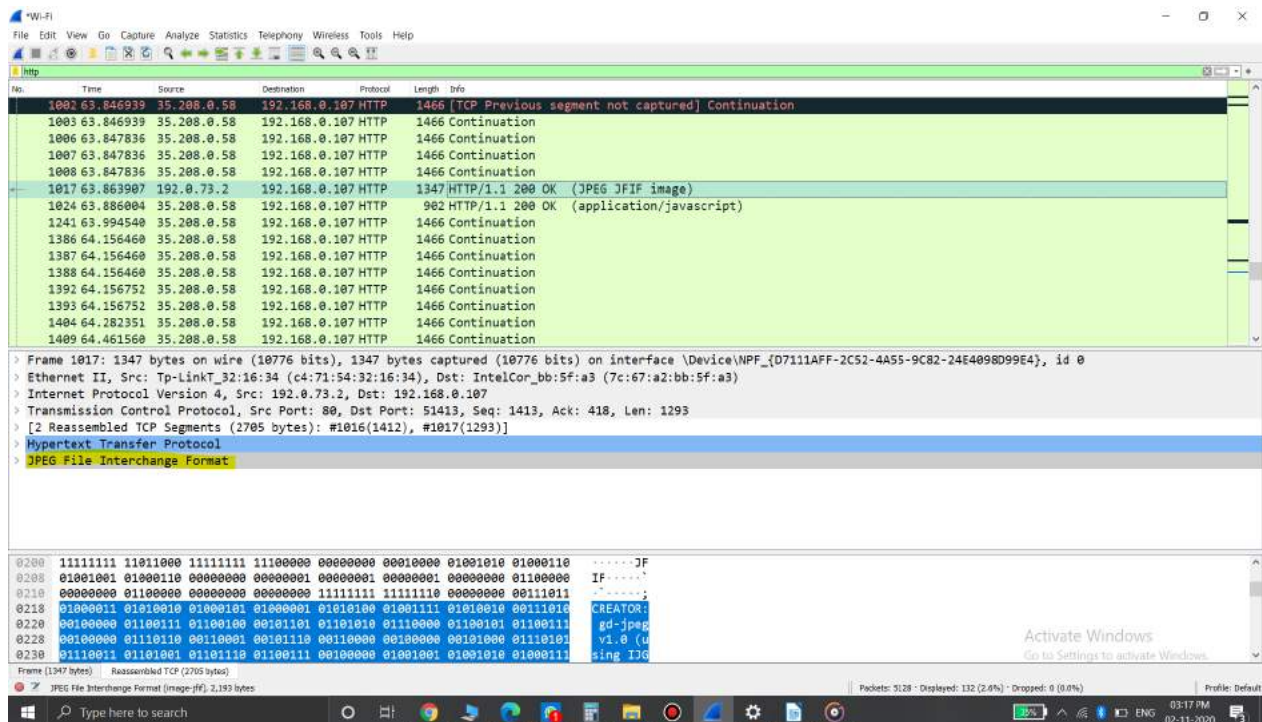
Activate Windows
Go to Settings to activate Windows.

Type here to search

03:17 PM
02-11-2020

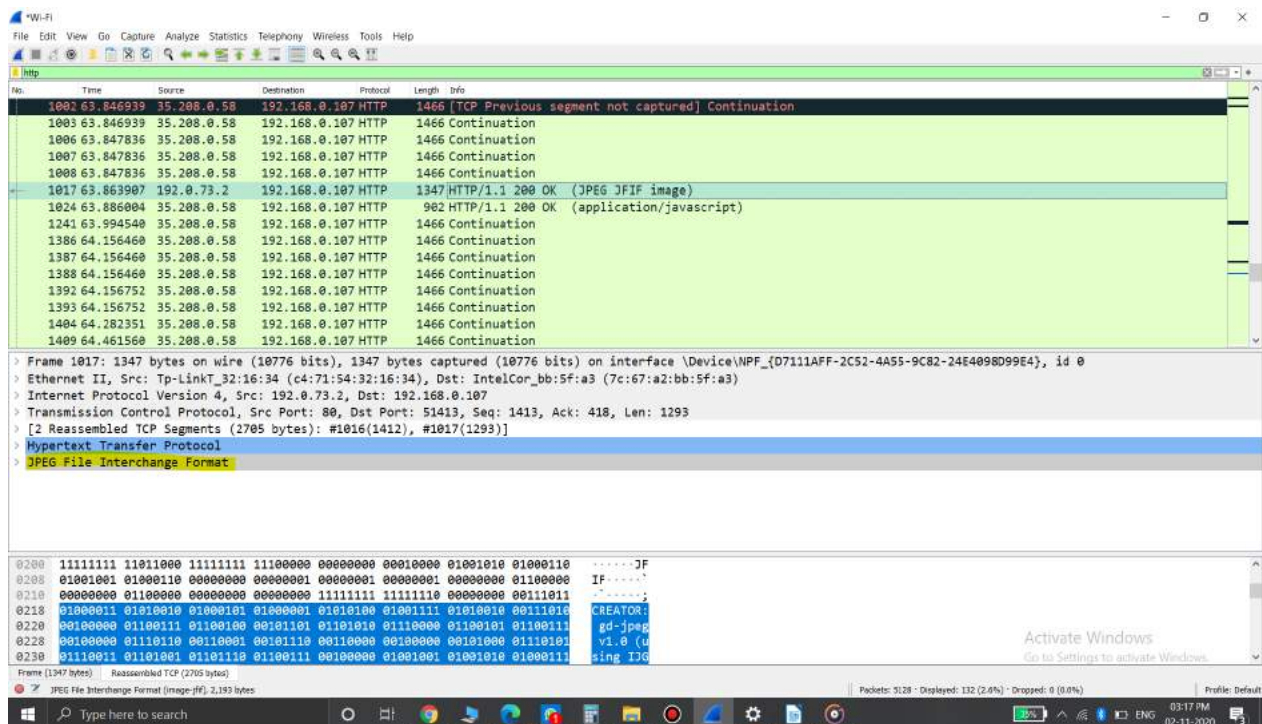
Step 5:

Filter the captured data, i.e. all the different protocols and packets that have been captured by typing “http” in the **Filter** tab and press Enter



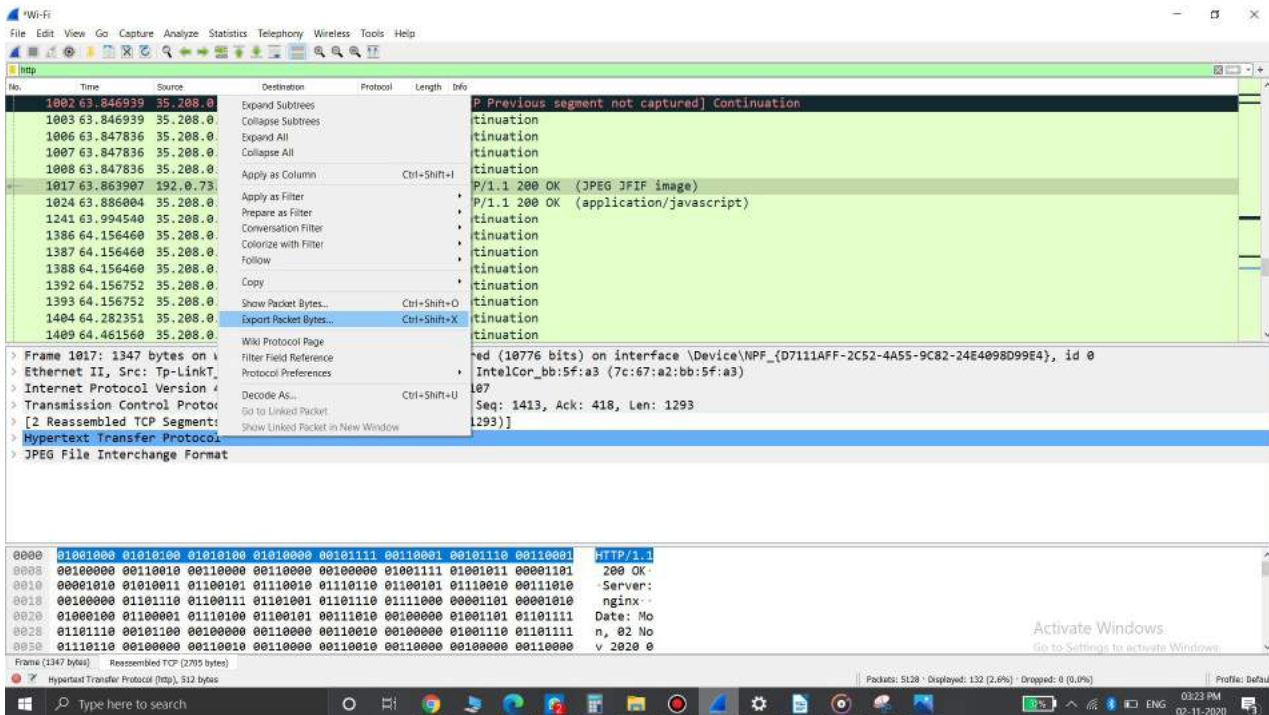
Step 6:

We are interested in capturing the images seen by the user so scroll down the list and look for types of files like JPEG, GIF and PNG files. Once we find such request, we click on it (in this case a JPEG file) and go to the second window area.



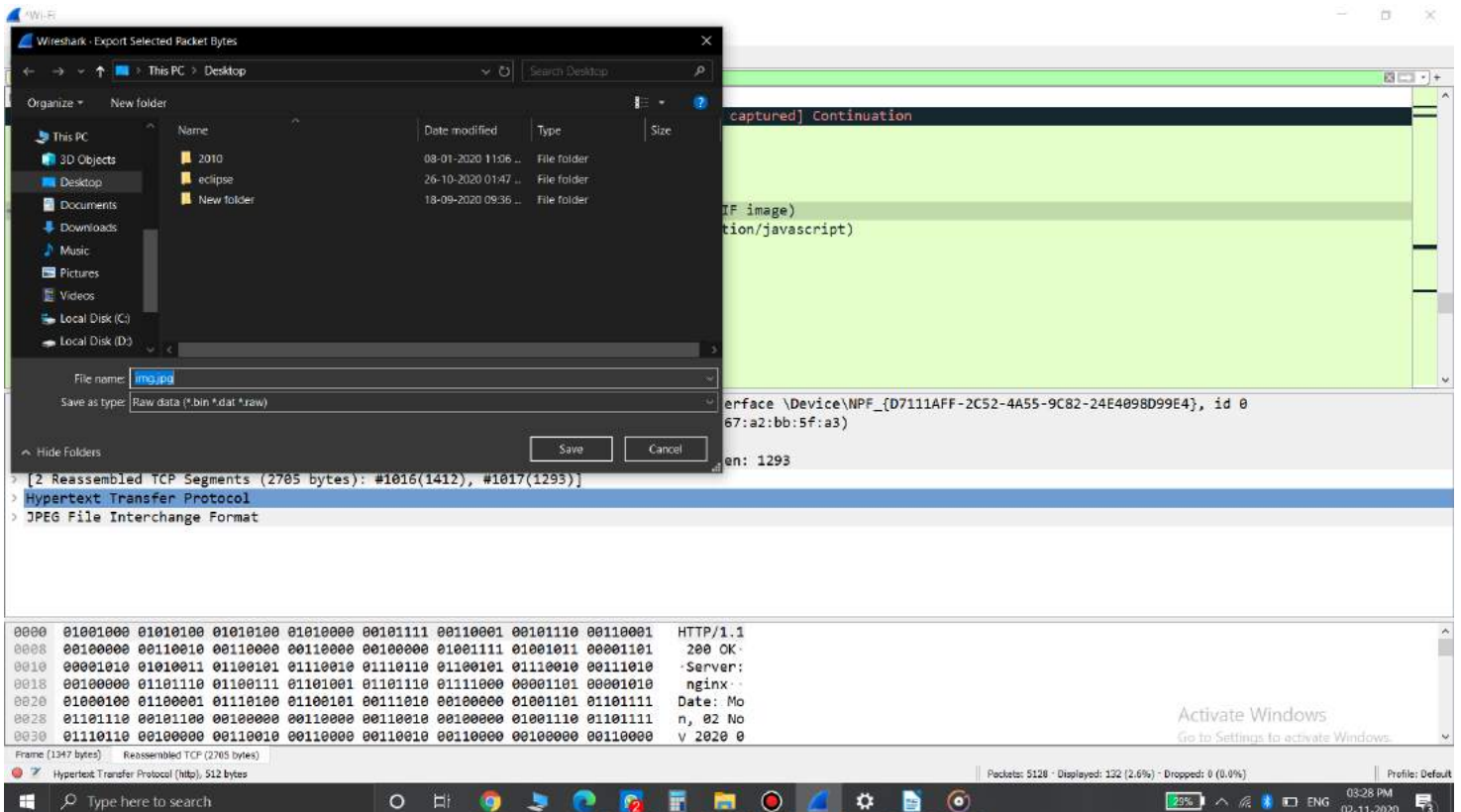
Step 7:

Right click on the “JPEG File Interchange Format” and click on the “Export Packet Bytes” option.



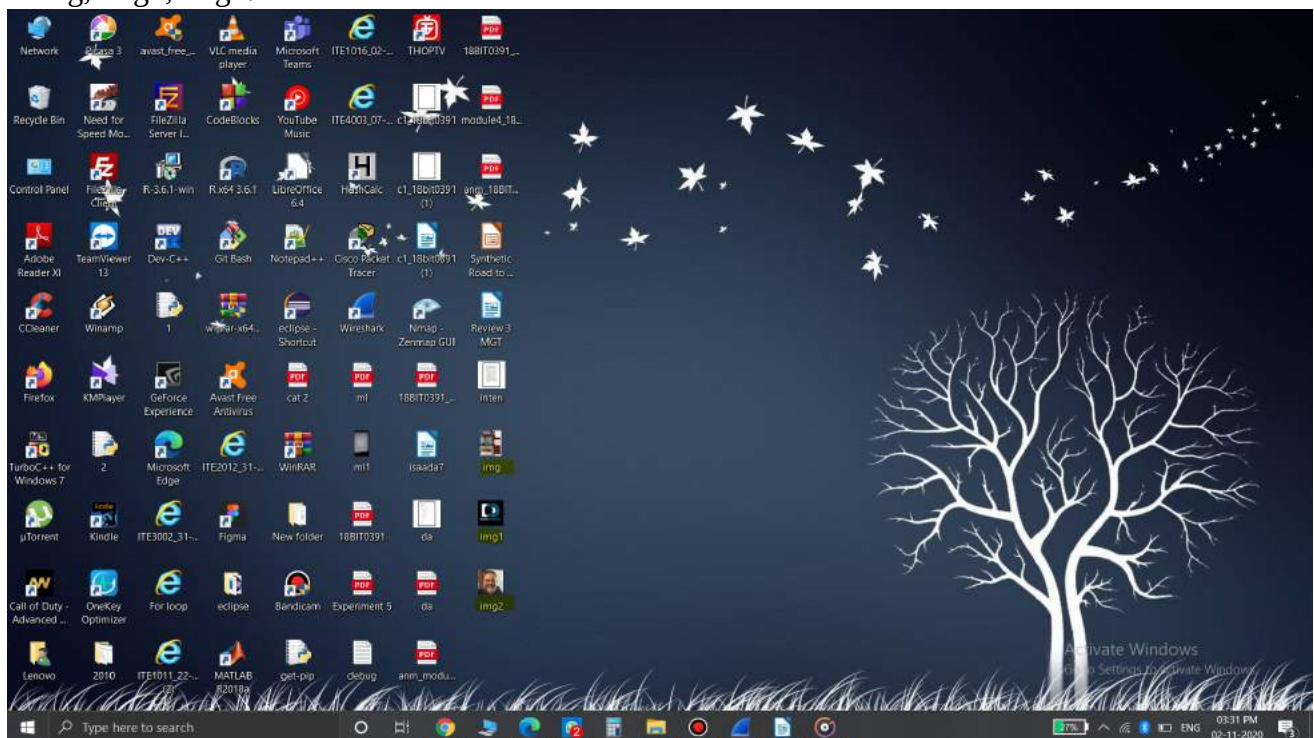
- Step 8:

Save the packet as img.jpg on the desktop.



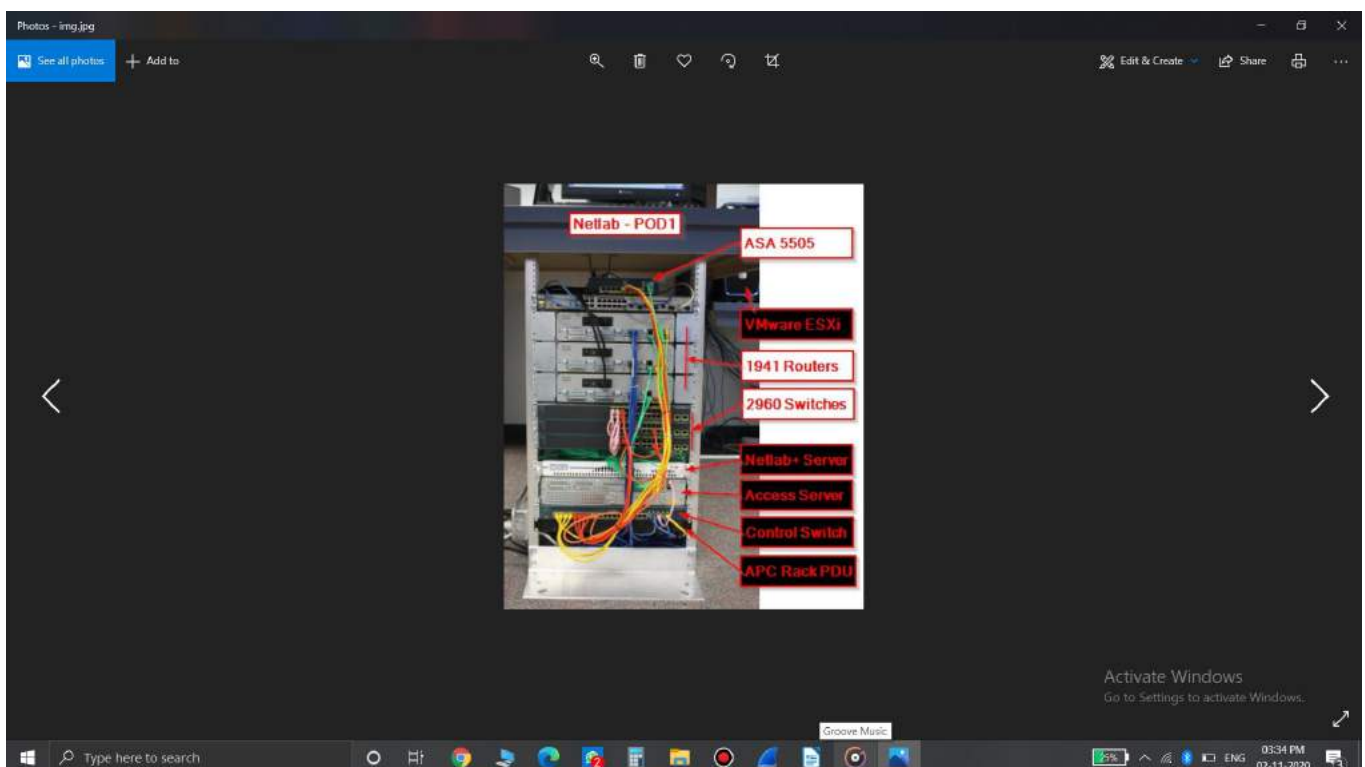
Step 9:

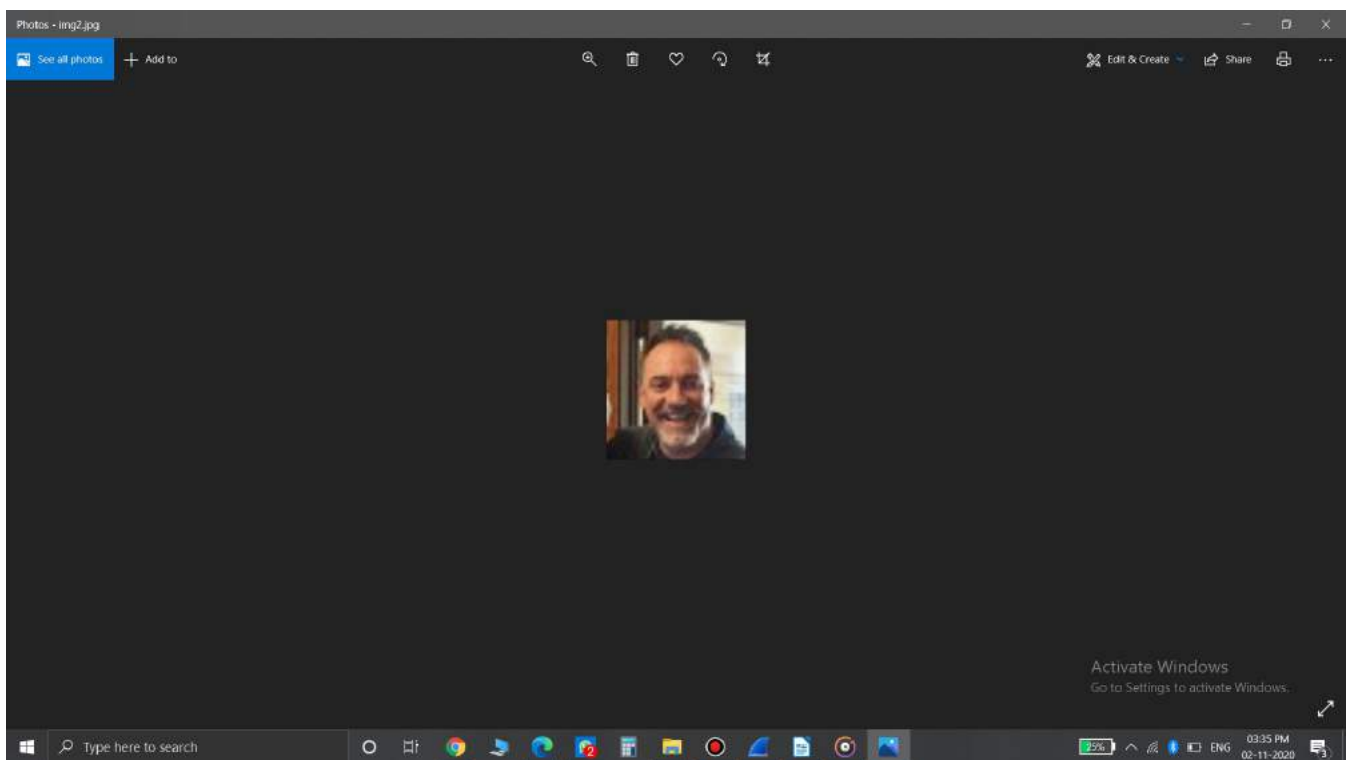
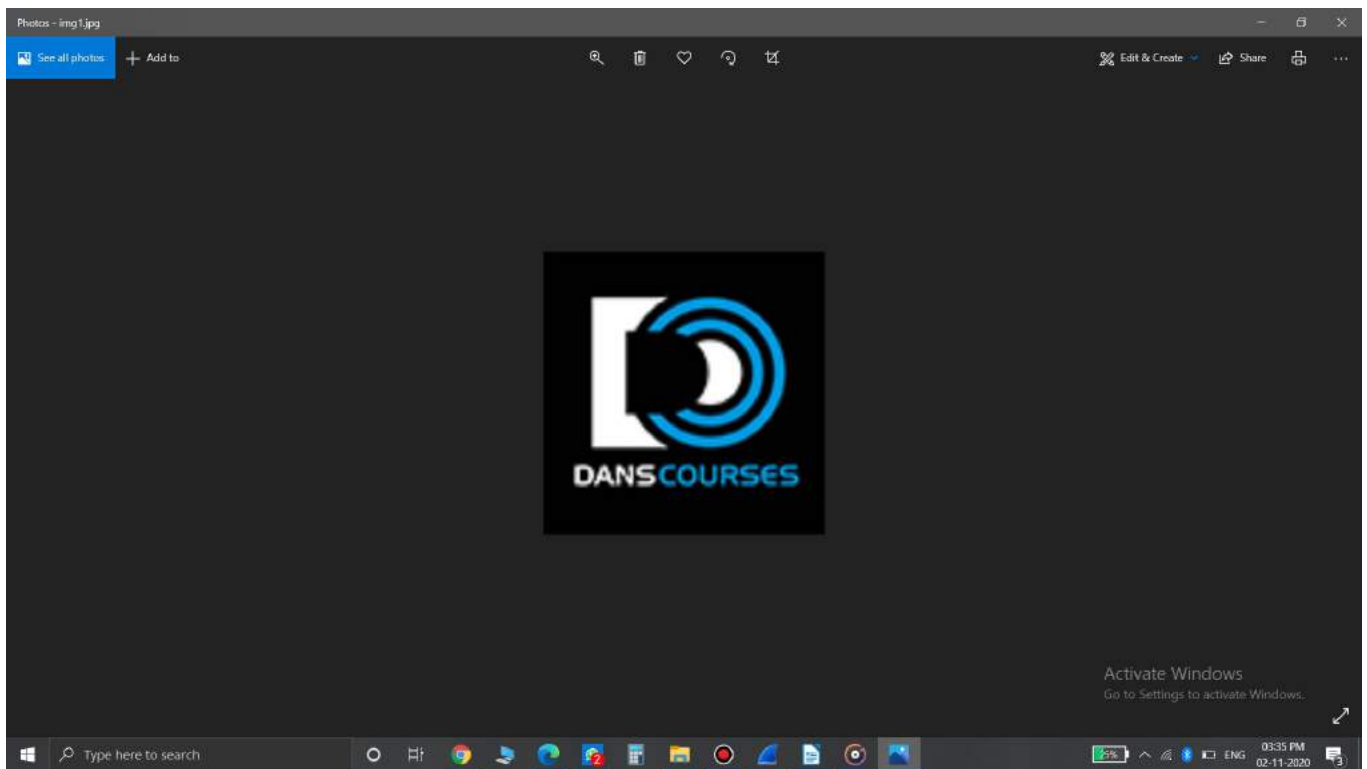
Scroll down the packets list and do the same process for JPEG, PNG, GIF files. After doing these steps just look at the desktop where you can now see the images named as img, img1, img2.



Step 10:

Open these images and you can now see the images that the user browsed through on the internet. Here is one image that we have rebuilt from the packet capture:





So, we learnt that if somebody was listening in on the network, they could pull the images and the type of information you are viewing on the internet if they were using a packet sniffing program like Wireshark. Hence we must always use secure websites that are encrypted.

Malware Traffic Analysis Using Wireshark

So at first in this we are going to learn about IOC's?

Ioc stands for Indication of Compromise. It is like a piece of forensic data.

As example:

Ip address

Domain name

User agent

Host name

File hashes

Specific pattern

So collection of Ioc can help organisation to detect and prevent attack

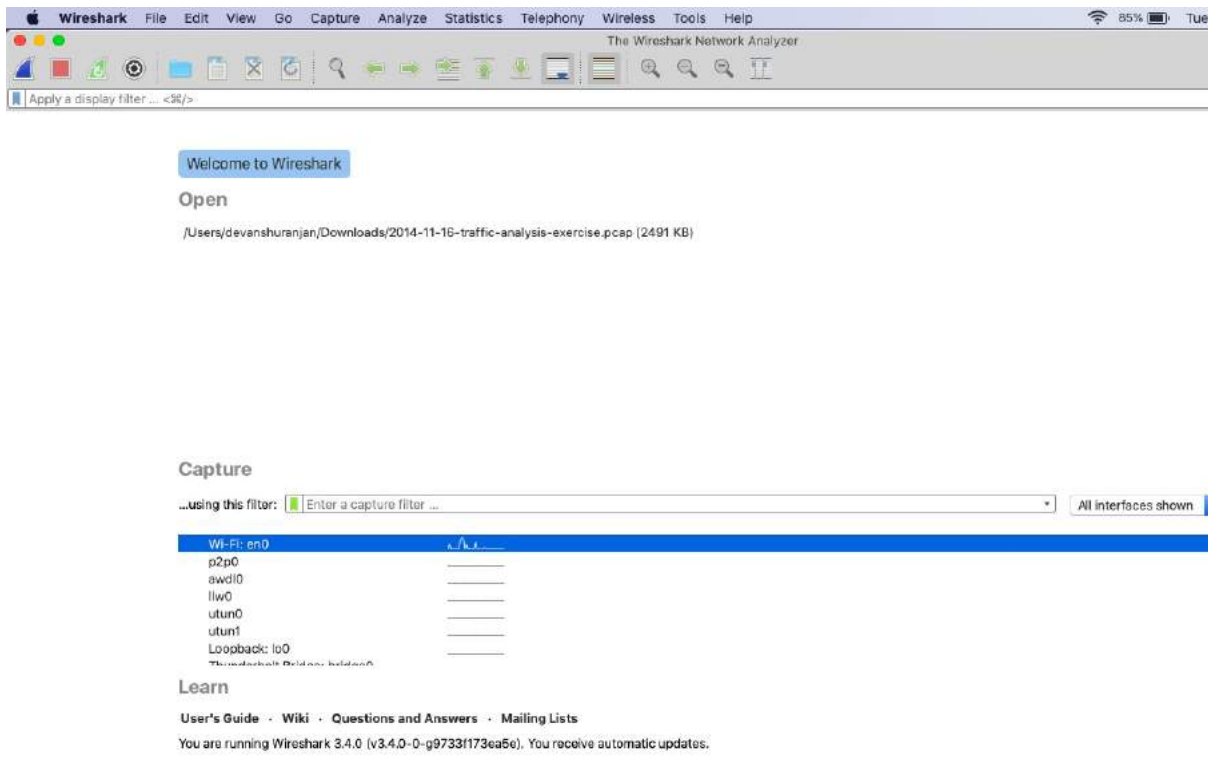
And after this a question got arise that why are we looking for IOC's then it is being answered as

What are we looking for – IOC's

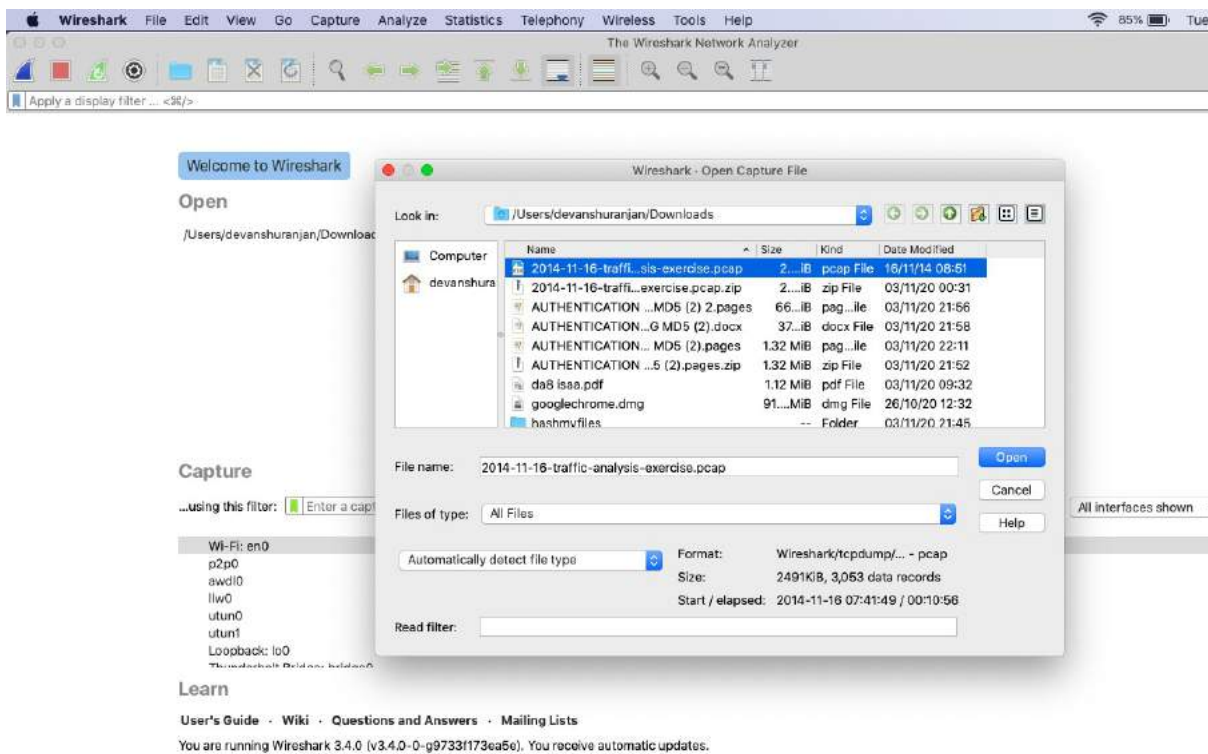
1. What are the infected file(s) downloaded and their hashes?
2. What is URL/ Domain of the infected site?
3. What is the IP address of the infected machine
4. What is the hostname of the infected Machine
5. What is the mac address of the infected machine

So now we are going to implement all the things

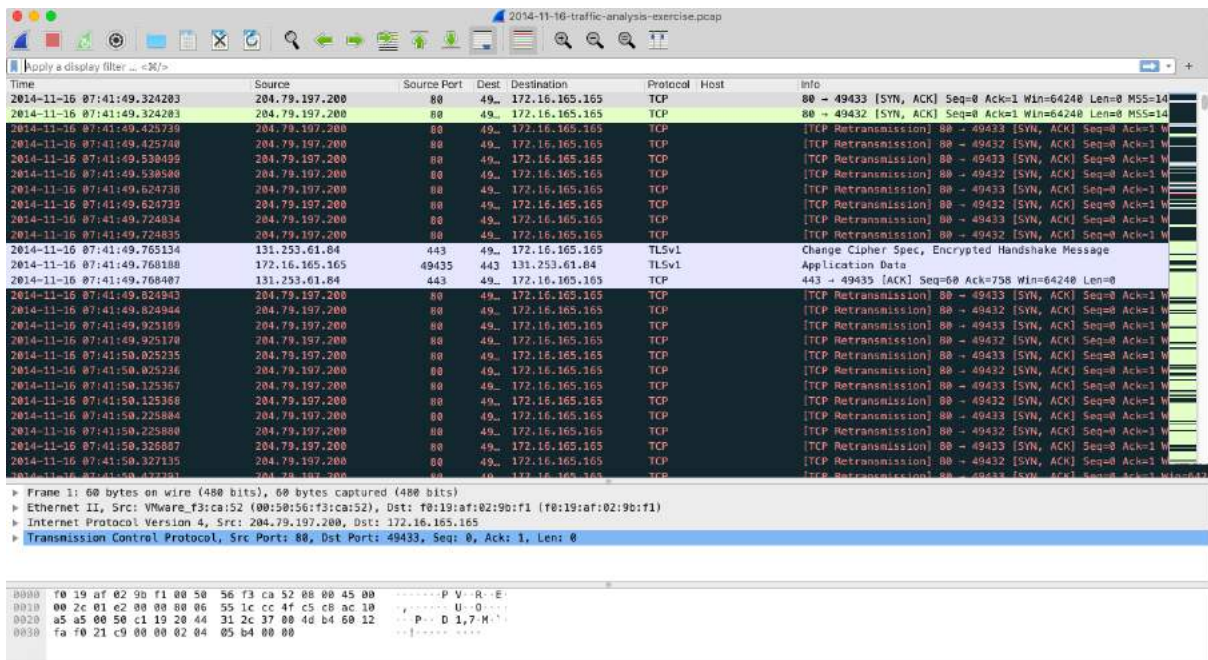
Step 1: we have to open wireshark.



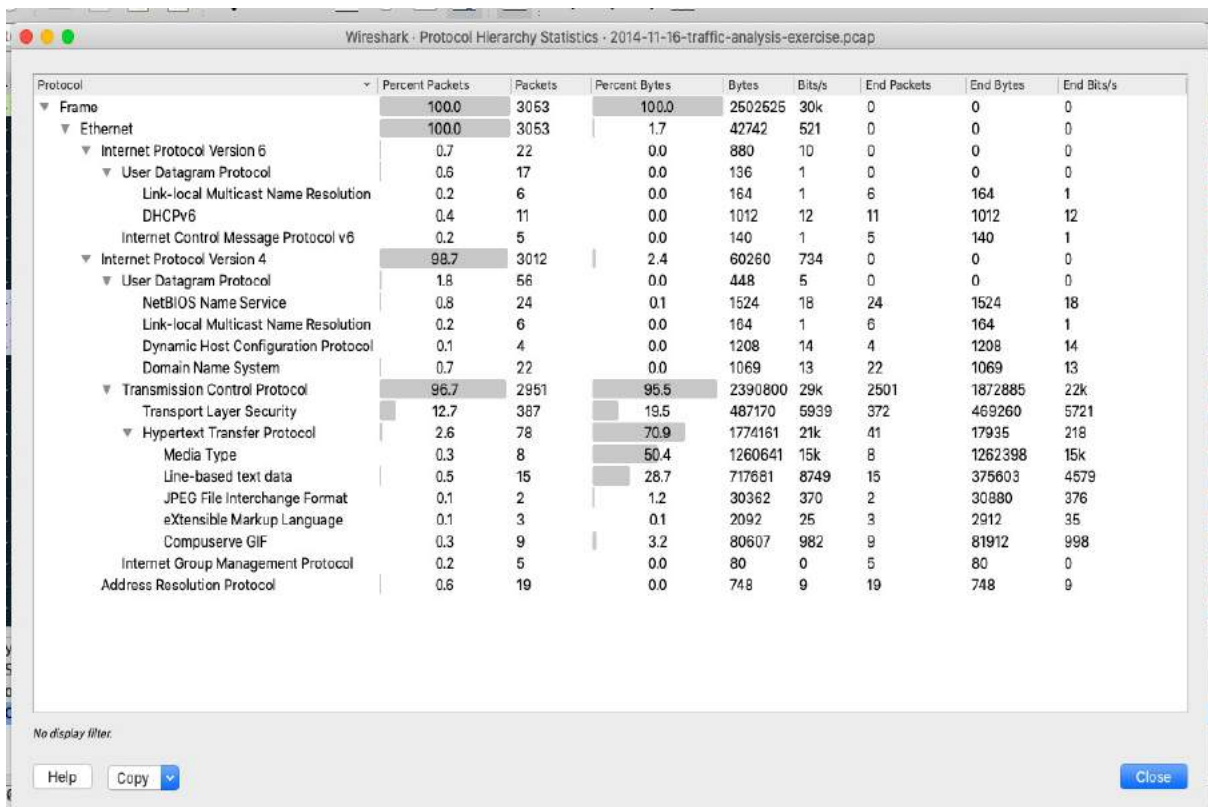
Step 2). we have to download a file of malicious website and then we can insert in into wireshark.



Step 3). After inserting it we will get this in wireshark window and we have to rearrange its column and then we can add some comoumns in it



Step 4).After that we have to learn about the protocol hierarchy statistics of the file



Step 5).Now we will filter that http.request thing and in that we can extract that http details.

2014-11-16-traffic-analysis-exercise.pcap

Time	Source	Source Port	Dest	Destination	Protocol	Host	Info
2014-11-16 07:41:51.345014	172.16.165.165	49431	80	204.79.197.200	HTTP	www.bing.com	POST /fd/ls/lsp.aspx HTTP/1.1
2014-11-16 07:41:53.562855	172.16.165.165	49429	80	204.79.197.200	HTTP	www.bing.com	GET /fd/ls/GLinkPing.aspx?IG=ae5988ea2d64991aa8b8996f61 HTTP/1.1
2014-11-16 07:41:55.397889	172.16.165.165	49437	80	82.150.140.30	HTTP	www.ciniholla.	GET / HTTP/1.1
2014-11-16 07:41:56.808775	172.16.165.165	49438	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/style.css HTTP/1.1
2014-11-16 07:41:56.819322	172.16.165.165	49439	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/plugins/contact-form-7/includes/css/styl HTTP/1.1
2014-11-16 07:41:56.819491	172.16.165.165	49440	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-includes/js/jquery/jquery-migrate.min.js?ver=1.2 HTTP/1.1
2014-11-16 07:41:56.819692	172.16.165.165	49441	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/plugins/sitenap/css/page-list.css?ver=4. HTTP/1.1
2014-11-16 07:41:56.819825	172.16.165.165	49442	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/js/functions.js HTTP/1.1
2014-11-16 07:41:57.572787	172.16.165.165	49439	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-includes/js/jquery/jquery.js?ver=1.10.2 HTTP/1.1
2014-11-16 07:41:57.572882	172.16.165.165	49441	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/plugins/contact-form-7/includes/js/jquer HTTP/1.1
2014-11-16 07:41:57.572898	172.16.165.165	49442	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/plugins/contact-form-7/includes/js/scrip HTTP/1.1
2014-11-16 07:41:57.858487	172.16.165.165	49443	80	185.53.178.9	HTTP	adultbiz.in	GET /new/jquery.php HTTP/1.1
2014-11-16 07:41:58.044582	172.16.165.165	49437	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/reset.css HTTP/1.1
2014-11-16 07:41:59.922627	172.16.165.165	49438	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/ing/br_logo.gif HTTP/1.1
2014-11-16 07:41:59.922869	172.16.165.165	49440	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/ing/donate_on.gif HTTP/1.1
2014-11-16 07:41:59.923000	172.16.165.165	49437	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/ing/newsletter_on.gif HTTP/1.1
2014-11-16 07:41:59.923184	172.16.165.165	49442	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/ing/facebook_on.gif HTTP/1.1
2014-11-16 07:41:59.923368	172.16.165.165	49441	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/ing/twitter_on.gif HTTP/1.1
2014-11-16 07:41:59.923565	172.16.165.165	49439	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/ing/youtube_logo_on.gif HTTP/1.1
2014-11-16 07:42:00.526548	172.16.165.165	49439	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/uploads/2012/01/P1268499-200x298.jpg HTTP/1.1
2014-11-16 07:42:00.526586	172.16.165.165	49442	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/uploads/2013/09/TMG-20130920-MA002-150x1 HTTP/1.1
2014-11-16 07:42:00.526799	172.16.165.165	49441	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/ing/squareorangedecor.gif HTTP/1.1
2014-11-16 07:42:01.397948	172.16.165.165	49444	80	74.125.233.96	HTTP	www.youtube.c.	GET /embed/hqg5ewj18hk HTTP/1.1
2014-11-16 07:42:09.726891	172.16.165.165	49438	80	82.150.140.30	HTTP	www.ciniholla.	GET /favicon.ico HTTP/1.1
2014-11-16 07:42:11.112064	172.16.165.165	49449	80	188.225.73.100	HTTP	24corp-shop.c.	GET / HTTP/1.1
2014-11-16 07:42:11.112167	172.16.165.165	49449	80	188.225.73.100	HTTP	24corp-shop.c.	GET / HTTP/1.1

Frame 52: 1002 bytes on wire (8016 bits), 1002 bytes captured (8016 bits) on Ethernet II, Src: f0:19:af:02:9b:f1 (f0:19:af:02:9b:f1), Dst: VMware_f3:ca:52 (00:50:56:f3:ca:52)

Internet Protocol Version 4, Src: 172.16.165.165, Dst: 204.79.197.200

Transmission Control Protocol, Src Port: 49431, Dst Port: 80, Seq: 821, Ack: 1, Len: 948

2 Reassembled TCP Segments (1768 bytes): #51(820), #52(948)

HyperText Transfer Protocol

0000 00 50 56 f3 ca 52 f0 19 af 02 9b f1 88 00 45 00 PV:R:-----E
0010 03 dc 14 20 40 00 00 06 ff 2d ac 10 a5 a5 cc 4f ---@:-----G
0020 c5 c8 c1 17 00 50 c3 e1 a5 80 e0 f0 aa 74 50 18 ---P:-----P
0030 f7 62 e8 8e 00 00 3c 43 6c 69 65 6e 74 49 6e 73 b---<C clientIns

Frame (1002 bytes) Reassembled TCP (1768 bytes)

Step 6).Here we will get the host address of some http files and add it as a column

2014-11-16-traffic-analysis-exercise.pcap

Time	Source	Source Port	Dest	Destination	Protocol	Host	Info
2014-11-16 07:41:51.345014	172.16.165.165	49431	80	204.79.197.200	HTTP	www.bing.com	POST /fd/ls/lsp.aspx HTTP/1.1
2014-11-16 07:41:53.562855	172.16.165.165	49429	80	204.79.197.200	HTTP	www.bing.com	GET /fd/ls/GLinkPing.aspx?IG=ae5988ea2d64991aa8b8996f61 HTTP/1.1
2014-11-16 07:41:55.397889	172.16.165.165	49437	80	82.150.140.30	HTTP	www.ciniholla.	GET / HTTP/1.1
2014-11-16 07:41:56.808775	172.16.165.165	49438	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/style.css HTTP/1.1
2014-11-16 07:41:56.819322	172.16.165.165	49439	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/plugins/contact-form-7/includes/css/styl HTTP/1.1
2014-11-16 07:41:56.819491	172.16.165.165	49440	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-includes/js/jquery/jquery-migrate.min.js?ver=1.2 HTTP/1.1
2014-11-16 07:41:56.819692	172.16.165.165	49441	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/plugins/sitenap/css/page-list.css?ver=4. HTTP/1.1
2014-11-16 07:41:56.819825	172.16.165.165	49442	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/js/functions.js HTTP/1.1
2014-11-16 07:41:57.572787	172.16.165.165	49439	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-includes/js/jquery/jquery.js?ver=1.10.2 HTTP/1.1
2014-11-16 07:41:57.572882	172.16.165.165	49441	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/plugins/contact-form-7/includes/js/jquer HTTP/1.1
2014-11-16 07:41:57.572898	172.16.165.165	49442	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/plugins/contact-form-7/includes/js/scrip HTTP/1.1
2014-11-16 07:41:57.858487	172.16.165.165	49443	80	185.53.178.9	HTTP	adultbiz.in	GET /new/jquery.php HTTP/1.1
2014-11-16 07:41:58.044582	172.16.165.165	49437	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/reset.css HTTP/1.1
2014-11-16 07:41:59.922627	172.16.165.165	49438	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/ing/br_logo.gif HTTP/1.1
2014-11-16 07:41:59.922869	172.16.165.165	49440	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/ing/donate_on.gif HTTP/1.1
2014-11-16 07:41:59.923000	172.16.165.165	49437	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/ing/newsletter_on.gif HTTP/1.1
2014-11-16 07:41:59.923184	172.16.165.165	49442	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/ing/facebook_on.gif HTTP/1.1
2014-11-16 07:41:59.923368	172.16.165.165	49441	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/ing/twitter_on.gif HTTP/1.1
2014-11-16 07:41:59.923565	172.16.165.165	49439	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/ing/youtube_logo_on.gif HTTP/1.1
2014-11-16 07:42:00.526548	172.16.165.165	49439	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/uploads/2012/01/P1268499-200x298.jpg HTTP/1.1
2014-11-16 07:42:00.526586	172.16.165.165	49442	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/uploads/2013/09/TMG-20130920-MA002-150x1 HTTP/1.1
2014-11-16 07:42:00.526799	172.16.165.165	49441	80	82.150.140.30	HTTP	www.ciniholla.	GET /wp-content/themes/cini/ing/squareorangedecor.gif HTTP/1.1
2014-11-16 07:42:01.397948	172.16.165.165	49444	80	74.125.233.96	HTTP	www.youtube.c.	GET /embed/hqg5ewj18hk HTTP/1.1
2014-11-16 07:42:09.726891	172.16.165.165	49438	80	82.150.140.30	HTTP	www.ciniholla.	GET /favicon.ico HTTP/1.1
2014-11-16 07:42:11.112064	172.16.165.165	49449	80	188.225.73.100	HTTP	24corp-shop.c.	GET / HTTP/1.1
2014-11-16 07:42:11.112167	172.16.165.165	49449	80	188.225.73.100	HTTP	24corp-shop.c.	GET / HTTP/1.1

Referer: http://www.bing.com/search?q=ciniholland.nl&q&sds&form=QBLH\r\n

Content-Type: text/xml\r\n

Accept-Encoding: gzip, deflate\r\n

User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0)\r\n

Host: www.bing.com\r\n

Content-Length: 948\r\n

0000 50 4f 53 54 20 2f 66 64 2f 6c 73 2f 6c 73 70 2e POST /fd/ls/lsp.
0010 61 73 70 70 20 48 54 54 50 2f 31 2e 31 0d 0a 41 aspx HTTP/1.1--A
0020 63 63 65 70 74 3a 20 2a 2f 2a 0d 0a 41 63 63 65 ccept: */*--Acce
0030 70 74 2d 4c 61 6e 67 73 61 67 65 3a 20 65 6e 2d pt-Langu age: en-

Frame (1002 bytes) Reassembled TCP (1768 bytes)

Step 7). We will export the file information and then we will use the content type as java file and save it.

2014-11-16-traffic-analysis-exercise.pcap

Wireshark · Export · HTTP object list

Text Filter: Content Type: All Content-Types

Packet	Hostname	Content Type	Size	Filename
52	www.bing.com	text/xml	948 bytes	lsp.aspx
130	www.bing.com	image/gif	42 bytes	GLinkPing.aspx?IG=aee5
311	www.ciniholland.nl	text/css	927 bytes	styles.css?ver=3.7.2
313	www.ciniholland.nl	text/javascript	237 bytes	functions.js
314	www.ciniholland.nl	text/css	702 bytes	page-list.css?ver=4.2
318	www.ciniholland.nl	text/html	61kB	/
340	www.ciniholland.nl	text/css	4807 bytes	style.css
341	www.ciniholland.nl	text/javascript	7200 bytes	jquery-migrate.min.js?ver
401	www.ciniholland.nl	text/css	1092 bytes	reset.css
432	www.ciniholland.nl	text/javascript	8913 bytes	scripts.js?ver=3.7.2
445	www.ciniholland.nl	text/javascript	16kB	jquery.form.min.js?ver=3.
495	adultbiz.in	text/html	8638 bytes	jquery.php
533	www.ciniholland.nl	text/javascript	93kB	jquery.js?ver=1.10.2
569	www.ciniholland.nl	image/gif	1270 bytes	youtubelogo_on.gif
572	www.ciniholland.nl	image/gif	577 bytes	twitter_on.gif
573	www.ciniholland.nl	image/gif	536 bytes	facebook_on.gif
595	www.ciniholland.nl	image/gif	4660 bytes	br_logo.gif
596	www.ciniholland.nl	image/gif	2476 bytes	newsletter_on.gif
597	www.ciniholland.nl	image/gif	2316 bytes	donate_on.gif
598	www.ciniholland.nl	image/gif	65 bytes	squareorangedecor.gif
654	www.ciniholland.nl	image/jpeg	19kB	P1260499-200x298.jpg

Help Preview Save All Close Save

ing.com · Connect
 ion: Kee p-Alive.
 Cookie: _HOP=;
 _EDGE_S= F=1&SID=
 142DA63B 92026172
 245CA0CF 93E5607F

Step 8). We will save it as follows :

2014-11-16-traffic-analysis-exercise.pcap

Source Port: 49431, Dest: 80, Destination: 204.79.197.200, Protocol: HTTP, Host: www.bing.com, Info: POST /fd/ls/lsp.aspx HTTP

Wireshark · Export · HTTP object list

Text Filter: Content Type: All Content-Types

Packet	Hostname	Content Type	Size	Filename
595	www.ciniholland.nl	image/gif	4660 bytes	br_logo.gif
596	www.ciniholland.nl	image/gif	2476 bytes	newsletter_on.gif
597	www.ciniholland.nl	image/gif	2316 bytes	donate_on.gif
598	www.ciniholland.nl	image/gif	65 bytes	squareorangedecor.gif
654	www.ciniholland.nl	image/jpeg	19kB	P1260499-200x298.jpg
661	www.ciniholland.nl	image/jpeg	10kB	IMG-20130928-WA002-
976	www.ciniholland.nl	image/vnd.microsoft.icon	17kB	favicon.ico
1074	24corp-shop.com	text/html	890 bytes	/
1079	24corp-shop.com	text/html	890 bytes	/
1356	24corp-shop.com	image/gif	68kB	notfound.gif
1554	stand.trustandprobaterealty.com	text/html	257kB	?PHPSESID=njrMNRuDN
1566	stand.trustandprobaterealty.com	text/html	255kB	?PHPSESID=njrMNRuDN
1991	stand.trustandprobaterealty.com	application/x-msdownload	401kB	index.php?req=mp3&nun
2379	stand.trustandprobaterealty.com	application/x-msdownload	401kB	index.php?req=mp3&nun
2394	stand.trustandprobaterealty.com	application/x-shockwave-flash	8227 bytes	index.php?req=swf&num
2415	stand.trustandprobaterealty.com	application/x-shockwave-flash	8227 bytes	index.php?req=swf&num
2469	stand.trustandprobaterealty.com	text/xml	572 bytes	index.php?req=xml&num
2475	stand.trustandprobaterealty.com	text/xml	572 bytes	index.php?req=xml&num
2489	stand.trustandprobaterealty.com	application/java-archive	10kB	index.php?req=jar&num=
2502	stand.trustandprobaterealty.com	application/java-archive	10kB	index.php?req=jar&num=
2977	stand.trustandprobaterealty.com	application/x-msdownload	401kB	index.php?req=mp3&nun


Help Preview Save All Close Save

se 6e 65 63 74 ing.com · Connect
 5c 69 76 65 0d ion: Kee p-Alive
 1f 50 3d 3b 20 Cookie: _HOP=;
 26 53 49 44 3d _EDGE_S= F=1&SID=
 32 36 31 37 32 142DA63B 92026172
 35 36 30 37 46 245CA0CF 93E5607F


Step 10).Our file details is as such



index.php%3freq=jar&nu...iOTM.jar



Screenshot 2020-11...21.19.38



index.php%3freq=jar&nu...iOTM.jar

Step 11).After that we will open online MD5 tool to convert it to hash codes

OnlineMD5

FOR ONLINE BUSINESS
BADE KAAM KA .COM

MD5 & SHA1 Hash Generator For File

Generate and verify the MD5/SHA1 checksum of a file without uploading it. Choose File no file selected

Click to select a file, or drag and drop it here(max: 4GB).

Filename: No File Selected
File size: 0 Bytes
Checksum type: ☒ MD5 ☐ SHA1 ☐ SHA-256
File checksum:
Compare with:
Process:

Compare Pause Stop

Opsgenie 200+ integrations with the best monitoring, workflow, and collaboration tools Start for free

Step 12). We will insert that file there and after that we will get hash codes

OnlineMD5

FOR ONLINE BUSINESS
BADE KAAM KA .COM

MD5 & SHA1 Hash Generator For File

Generate and verify the MD5/SHA1 checksum of a file without uploading it. Choose File index.php%3f...NmJIOTM.jar

Click to select a file, or drag and drop it here(max: 4GB).

Filename: index.php%3f...NmJIOTM.jar
File size: 10,606 Bytes
Checksum type: ☒ MD5 ☐ SHA1 ☐ SHA-256
File checksum: 1E34FDEBBF655CEBEA78B45E43520DDF
Compare with:
Process: 100.00%

Compare Pause Stop

Opsgenie 200+ integrations with the best monitoring, workflow, and collaboration tools Start for free

Step 13).After that we will open virustotal to enquire about the malware thing



Analyze suspicious files and URLs to detect types of malware, automatically share them with the security community

FILE
URL
SEARCH

URL, IP address, domain, or file hash

By submitting data above, you are agreeing to our [Terms of Service](#) and [Privacy Policy](#), and to the **sharing of your Sample submission with the security community**. Please do not submit any personal information; VirusTotal is not responsible for the contents of your submission. [Learn more](#).

Want to automate submissions? [Check our API](#), free quota grants available for new file uploads



Step 14). We will insert that hash codes in that and we will get all the file information

178be0ed83a7a9020121dee1c305fd6ca3b74d15836835cfb1684da0b44190d3

[Sign in](#)
[Sign up](#)

34
62

34 engines detected this file

178be0ed83a7a9020121dee1c305fd6ca3b74d15836835cfb1684da0b44190d3
e33

10.36 KB
Size

2020-10-24 21:18:35 UTC
9 days ago

cve-2012-0507 exploit jar

DETECTION
DETAILS
RELATIONS
BEHAVIOR
COMMUNITY

AegisLab	Hacktool.Java.Generic.3tc	Alibaba	Exploit:JAVA/CVE-2012-0507:9b7117c2
Arcabit	Java.Exploit.CVE-2012-0507.AG	Avast	Java.Malware-gen [Trj]
AVG	Java.Malware-gen [Trj]	Avira (no cloud)	EXP:JAVA.Rafold.AL.Gen
BitDefender	Java.Exploit.CVE-2012-0507.AG	CAT-QuickHeal	Exp.JAVA.Agent.DRV
ClamAV	Java.Malware.Agent-5656862-0	Comodo	Malware@#1enp2kd12fn
Cynet	Malicious (score: 85)	Cyren	Java.Agent.KR
Emsisoft	Java.Exploit.CVE-2012-0507.AG (B)	eScan	Java.Exploit.CVE-2012-0507.AG
ESET-NOD32	A Variant Of Java/Exploit.Agent.REU	F-Secure	Exploit:EXP:JAVA.Rafold.AL.Gen
FireEye	Java.Exploit.CVE-2012-0507.AG	GData	Java.Exploit.CVE-2012-0507.AG



So here we all can see that out of 62 engines 32 engines have malware and this can harm your system.

CONCLUSION:-

This article has presented the Smart Detection system, an online approach to DoS/DDoS attack detection. The software uses the Random Forest Tree algorithm to classify network traffic based on samples taken by the sFlow protocol directly from network devices. Several experiments were performed to calibrate and evaluate system performance. Results showed that the proposed method is feasible and presents improved performance when compared with some recent and relevant approaches available in the literature.

The proposed system was evaluated based on three intrusion detection benchmark datasets, namely, CIC-DoS, CICIDS2017, and CSE-CIC-IDS2018, and was able to classify various types of DoS/DDoS attacks, such as TCP flood, UDP flood, HTTP flood, and HTTP slow. Furthermore, the performance of the proposed method was compared against recent and related approaches. Based on the experimental results, the Smart Detection approach delivers improved DR, FAR, and PREC. For example, in the CIC-DoS and CSE-CIC-IDS2018 datasets, the proposed system acquired DR and PREC higher than 93% with FAR less than 1%. Although the system has achieved significant results in its scope, it needs some improvements, such as a better hit rate among attack classes and an automatic parameter calibration mechanism that maximizes the detection rate of attacks.

Future works include analysis of DDoS attacks based on the vulnerabilities of services such as Heartbleed and web brute force attack, enhancement in the multiple-class classification, self-configuration of the system, developing methods for correlating triggered alarms, and formulating protective measures.

REFERENCES:-

- [1].CV Arul Kumar, D. Manoj Kumar and P. Prithiviraj, "Privacy Policy Multiparty Access Control On Content Sharing Sites",*Imperial Journal of Interdisciplinary Research*, vol. 2.6, 2016.
- [2]. Huang Xiaoyan, "Network Protocol Analysis Based on Ethereal",*Tropical Agricultural Engineering*, vol. 33, no. 1, pp. 42-45, 2009.
- [3].M. A. Qadeer, A. Iqbal, M. Zahid and M. R. Siddiqui, "Network traffic analysis and intrusion detection using packet sniffer",*Communication Software and Networks 2010. ICCSN'10. Second International Conference on*, pp. 313-317, 2010.
- [4].N. Lizarti and W. Agustin, "Aplikasi Network Traffic Monitoring Menggunakan Simple Network Management Protocol (SNMP) pada Jaringan Virtual Private Network (VPN)",*SATIN-Sains dan Teknologi Informasi*, vol. 1, pp. 27-34, 2015.
- [5].M. Dagon, D. Luo and R. Lee, "A centralized monitoring infrastructure for improving DNS security",*Springer International Workshop on Recent Advances in Intrusion Detection*, pp. 18-37, 2010.

- [6].S. Kumar and S. Tapaswi, "A centralized and prevention technique against ARP poisoning",IEEE International Conference of Cyber Security Warfare and Digital Forensic, pp. 259-26, 2012.\
- [7].U. Banerjee, A. Vashishtha and M. Saxena, "Evaluation of the Capabilities of WireShark as a tool for Intrusion Detection", International Journal of Computer Applications, vol. 6, no. 7, pp. 1-5, 2010.
- [8].J.Mirkovic,E.Arikan,S.Wei,R.Thomas,S.Fahmy,P.Reiher Benchmarks for DDoS defense evaluation Proceedings of the IEEE military communications conference, MILCOM,IEEE(2006), pp.1-10
- [9].Tiwari, P. K and B. Mishra. Cloud Computing Security Issues, Challenges and Solution. International Journal of Emerging Technology and Advanced Engineering, 2(8), 306-310 (2012).
- [10]. Ali, W., J. Sang, H. Naeem, R. Naeem and A. Raza. Wireshark window authentication based packet captureing scheme to pervent DDoS related security issues in cloud network nodes. In Software Engineering and Service Science, 2015 6th IEEE International Conference on, 114-118(2015).
- [11]A. S. Raihana, A. F. Mohd, M. N. A. Zul, M. Z. Mohd, S. R. Siti and Y. Robiah, "Revealing the Criterion on Botnet Detection Technique",*International Journal of Computer Science (IJCSI)*, vol. 10, no. 2, pp. 208-215, March 2013.
- [13].P. Sangkatsanee, N. Wattanapongsakorn and C. Charnsripinyo, "Practical real-time intrusion detection using machine learning approaches",*Computer Communications*, vol. 34, no. 18, pp. 2227-2235, December 2011.
- [14].T. Isohara, K. Takemori and A. Kubota, "Kernel-based behavior analysis for android malware detection",*Internation Conference on Computational Intelligence and Security*, pp. 1011-1015, Dec 3-4 2011.