The National Law Institute University, Bhopal

Offers Programme

Master

Of

Cyber Law and Information Security

Project On

“Vulnerability Assessment using Wireshark and Nessus”

SUBJECT

“Cyber Operation Security”

SUBMITTED TO:
Mr. Ankur Rajput

SUBMITTED BY:
Isha Gaur
2019 MCLIS 58

Table of Contents

ABSTRACT

In the age of fast internet and global communication systems, computer security is a big challenge for any public or private organization. There exist many more threats to such organization and required some top level of security in the organization for securing company's critical information. Therefore, each individual computer system is very important to secure them because a single system is responsible to compromise whole organizations network. To verify the security checks and strengthen the organizations network, a vulnerability assessment of the whole organizations network must be performed regularly. Vulnerability scanners are useful to discover security flaws within each individual system as well as whole network also.

This paper focuses on the Wireshark and Nessus vulnerability scanners and the comparative analysis of their methods to discover various vulnerabilities available in the networks or remotely connected host system and make a comparative analysis on the bases of their ability to detect different flaws.

KEYWORDS: Threat, Vulnerability, Vulnerability Scanners, Security flaw, Port scanning.

1. INTRODUCTION

Vulnerability scanning means scanning of the systems, network devices and applications which works on front to external worlds or scanning the internally hosted system to find the security flaws on them. There are number of different approaches to understand the basic framework of Vulnerability scanners. Vulnerability scanners have a database of already exposed vulnerabilities; with reference to known vulnerability, vulnerability scanner performs the security verification on remote host. Vulnerability scanner is break down into four major modules such as user interface, scan engine, scan databases, report generation module. The Internet has grown significantly in scope, and many results are shown for the operational requirement for algorithms and protocols. It is necessary to have a good and strong protocol analyser like Wireshark and Nessus.

Wireshark (Formerly named Ethereal) is an open-source packet analyser or packet sniffer[1]. It is used to capture and analyse network traffic and tries to display the detailed Information about the collected packet over the network. Nessus is one of the most popular vulnerability scanners. It is used for both authenticated and unauthenticated vulnerability scans. It is suitable for both internal and external network scans. It is also performed the scanning of web applications.

2. STATEMENT OF PROBLEM

There are many vulnerability scanners, but Wireshark and Nessus are the first choice of the security experts. Both the tools are used for vulnerability scanning whether it is network scanning or website vulnerability. There are some functionalities which are making them different from each other. A comparison is required to identify the limitations of these tools over each other.

3. RESEARCH IMPLEMENTATIONS

3.1 LITERATURE REVIEW

For writing this paper we have gone through various books, papers, operational security incidents and articles related Wireshark and Nessus. In a paper "**Port and Vulnerability Scanning, Packet Sniffing, Intrusion Detection, and Penetration Testing**" by Ravi Kak, many tools used for entire security are described. In this paper the author mentioned the capabilities of Wireshark for information gathering and the capability of Nessus for generating vulnerability reports and also providing the solution for the detected vulnerability. The international journal "**A Comparative Analysis of Detecting Vulnerability in Network Systems**" by Sandeep Yadav and Daya Pandey, focuses on the different vulnerability scanners and their methods to discover various vulnerabilities available in the networks or remotely connected host system and make a comparative analysis on the bases of their ability to detect

[1] A packet sniffer, sometimes called a packet analyser, is composed of two main parts. First, a network adapter that connects the sniffer to the existing network. Second, software that provides a way to log, see, or analyse the data collected by the device. Available at https://www.paessler.com/it-explained/packet-sniffing accessed on 17-03-2020.

different flaws. The article "**Network forensics analysis using Wireshark**" by Zhifeng Xiao describes the role of Wireshark in information gathering regarding the network.

3.2 OBJECTIVES OF STUDY

- To understand the functionalities of Wireshark and Nessus.
- To perform a comparative study on both the tools.
- To identify the limitations of Wireshark which could be removed by Nessus.
- To analyse which tool is better for vulnerability assessment.

3.3 HYPOTHESIS

Nessus can perform additional functionalities as compared to Wireshark.

3.4 RESEARCH QUESTIONS

- What are the functionalities of Wireshark and Nessus?
- What are the performance features of both the tools?
- What are the limitations of both the tools?
- What are the additional functionalities in Nessus as compared to Wireshark?

3.5 RESEARCH METHODOLOGY

The research methodology adopted for this paper is Doctrinal.

4. WIRESHARK

Wireshark was developed by Gerald Combs in 1997 for Network analysis and troubleshooting. The Wireshark is a GUI based tool[2], there is a CUI based version of Wireshark called TShark[3]. It provides almost the same features as Wireshark but is command line based. Wireshark is the world's most popular Network Protocol Analyzer is a multipurpose tool. It can be used as a Packet Sniffer, Network Analyser, Protocol Analyser & Forensic tool. Wireshark is a tool that understands the internal working of various networking protocols. Thus, it is able to display the different fields along with their meaning.

[2] The underlying idea behind Graphic User Interface (GUI) Tools is to provide an interactive graphic user interface for GAUSS for Windows. GUI Tools allows the programmer to have the end user respond to a graphic based dialog, along with standard Windows controls, using both keyboard and mouse. GUI Tools is called from GAUSS with a minimum of programming - typically one specifies a title, prompt, and the name of the control or GUI, followed by a one line call. Available at http://www.econotron.com/guitools/guitools.htm accessed on 19-03-2020.

[3] TShark is a network protocol analyzer. It lets you capture packet data from a live network, or read packets from a previously saved capture file, either printing a decoded form of those packets to the standard output or writing the packets to a file. TShark's native capture file format is pcapng format, which is also the format used by wireshark and various other tools. Available at https://www.wireshark.org/docs/man-pages/tshark.html accessed on 20-03-2020.

4.1 FEATURES OF WIRESHARK

Various features of Wireshark are mentioned below:

1. Inspection of various networking protocols, with more being added time to time.
2. Live capture of packets.
3. Multi-Platform support: Runs on Windows, Linux, MacOS and many others.
4. It is a Graphical User Interface tool and provides many sorting, displaying and filtering options to the user.
5. Built-in colour coding for better understanding and analysis to identify a specific type of traffic.
6. Save captured packets for offline analysis.
7. Provides various statistics.
8. Search for specific packets.
9. Import packets from the various files.
10. Exporting of packets to different file formats.

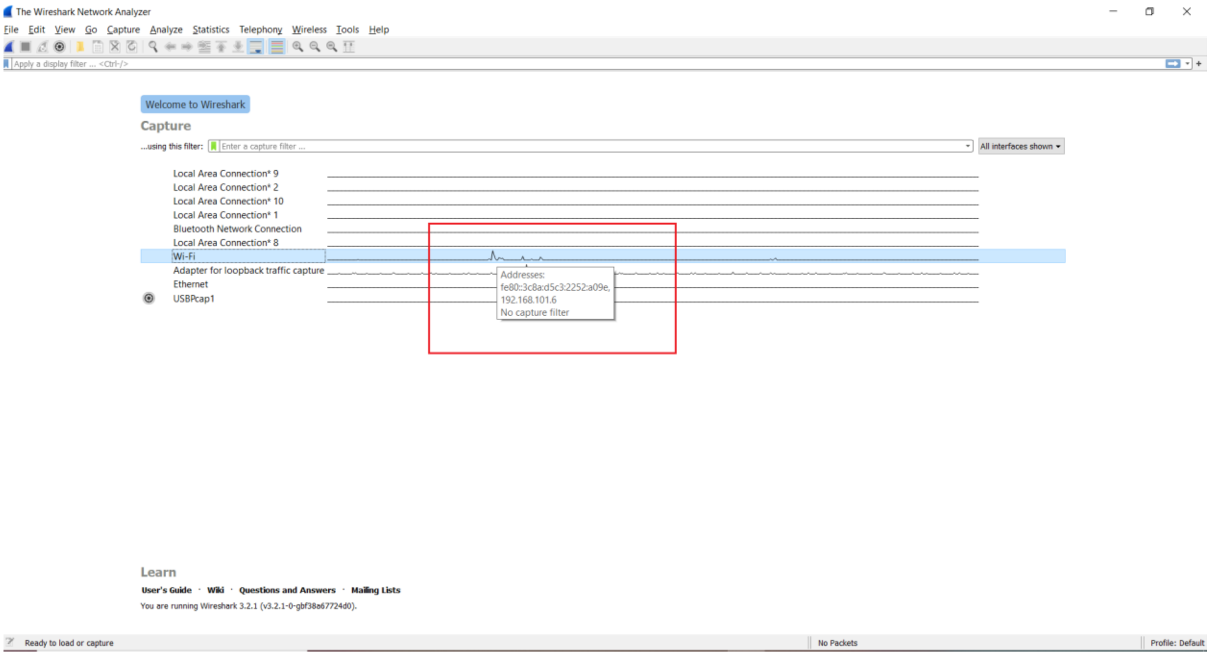
4.2 WORKING OF WIRESHARK

Wireshark sniffs packets that are sent and received by computer. It also displays the information about various networking protocols fields in the captured packets. Wireshark is passive, it analyses the communication path in which the messages are being sent by the application and protocols. Wireshark consists of two parts:

- **Pcap (packet capture library):** It takes each copy of link layer frame that is sent and received by your computer. Messages exchanged by upper-level protocols are encapsulated in the link layer and forwarded to the physical layer for transmission.
- **Packet Analyzer:** It presents the content of the fields within a message; packet analyzer knows the formation of the messages that are being transferred by the protocol and shows it to the user in the readable format.

4.3 WIRESHARK INTERFACES

When we'll run the Wireshark for the first time, this Graphical User Interface will be displayed.


The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Welcome to Wireshark
Capture
...using this filter:
Enter a capture filter ...
All interfaces shown
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists

Local Area Connection* 9
Local Area Connection* 2
Local Area Connection* 10
Local Area Connection* 1
Bluetooth Network Connection
Local Area Connection* 8
Wi-Fi
Adapter for loopback traffic capture
Ethernet
USBPcap1

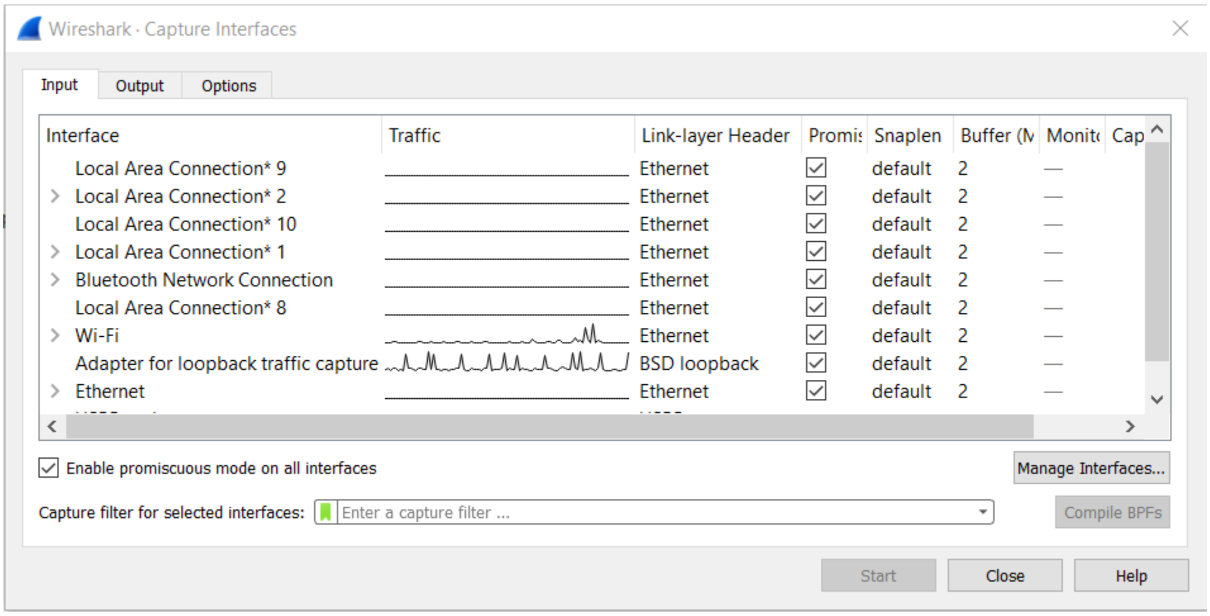
Addresses:
fe80::3c8a:d5c3:2252:a09e,
192.168.101.6
No capture filter

You are running Wireshark 3.2.1 (v3.2.1-0-gbf38a67724d0).

Ready to load or capture | No Packets | Profile: Default

Figure 1: Wireshark Initial Interface

Initially, we'll get no data related to the network, here in Figure 1 I've found Local Area Connection 1,2,8,9,and 10, Bluetooth Network Connection, Wi-Fi, Ethernet, Adapter For Loopback Traffic Capture and USBPcap1.



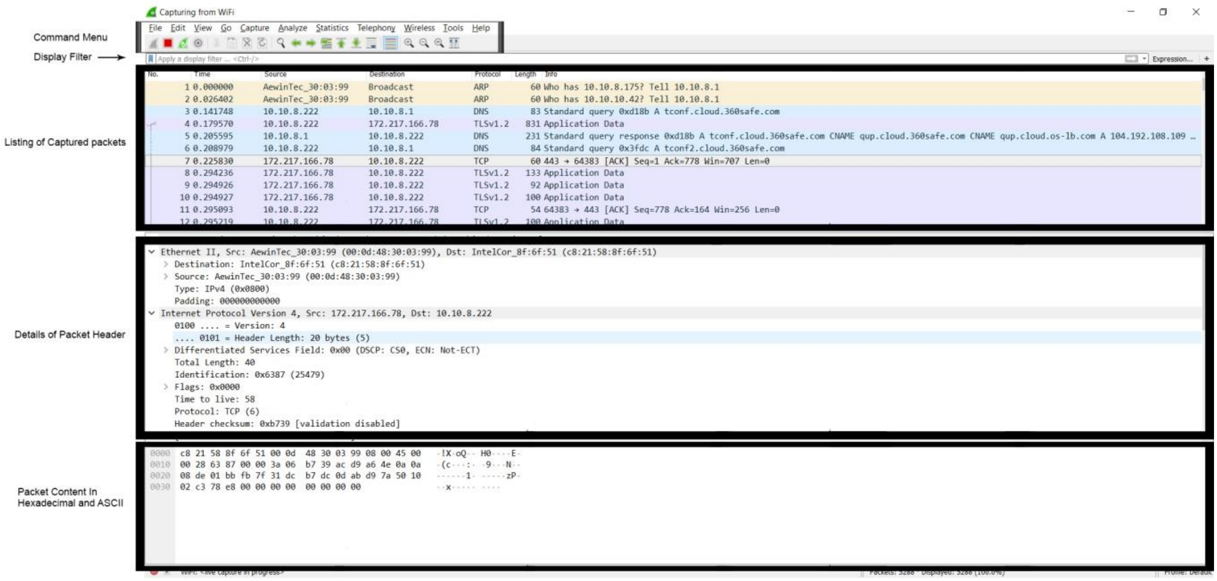
Interface	Traffic	Link-layer Header	Promis	Snaplen	Buffer (M	Monit	Cap
Local Area Connection* 9		Ethernet	☑	default	2	—	
Local Area Connection* 2		Ethernet	☑	default	2	—	
Local Area Connection* 10		Ethernet	☑	default	2	—	
Local Area Connection* 1		Ethernet	☑	default	2	—	
Bluetooth Network Connection		Ethernet	☑	default	2	—	
Local Area Connection* 8		Ethernet	☑	default	2	—	
Wi-Fi		Ethernet	☑	default	2	—	
Adapter for loopback traffic capture		BSD loopback	☑	default	2	—	
Ethernet		Ethernet	☑	default	2	—	

☑ Enable promiscuous mode on all interfaces

Capture filter for selected interfaces: Enter a capture filter ...

Figure 2: Wireshark Capture Interface

We can go to Capture Menu for detailed view; in my case I'll be selecting Wi-Fi in Figure 2.


Command Menu
Display Filter
Listing of Captured packets
Details of Packet Header
Packet Content In Hexadecimal and ASCII
Capturing from WiFi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...
WiFi: <live capture in progress>

```
No.   Time        Source             Destination      Protocol  Length Info
    1 0.000000    AewinTec_30:03:99  Broadcast        ARP        60 Who has 10.10.8.175? Tell 10.10.8.1
    2 0.026402    AewinTec_30:03:99  Broadcast        ARP        60 Who has 10.10.10.42? Tell 10.10.8.1
    3 0.141748    10.10.8.222        10.10.8.1        DNS        83 Standard query 0xd18b A tconf.cloud.360safe.com
    4 0.179570    10.10.8.222        172.217.166.78   TLSv1.2   831 Application Data
    5 0.205595    10.10.8.1          10.10.8.222      DNS       231 Standard query response 0xd18b A tconf.cloud.360safe.com CNAME qup.cloud.360safe.com CNAME qup.cloud.os-lb.com A 104.192.108.109 …
    6 0.208979    10.10.8.222        10.10.8.1        DNS        84 Standard query 0x3fdc A tconf2.cloud.360safe.com
    7 0.225830    172.217.166.78     10.10.8.222      TCP        60 443 → 64383 [ACK] Seq=1 Ack=778 Win=707 Len=0
    8 0.294236    172.217.166.78     10.10.8.222      TLSv1.2   133 Application Data
    9 0.294926    172.217.166.78     10.10.8.222      TLSv1.2    92 Application Data
   10 0.294927    172.217.166.78     10.10.8.222      TLSv1.2   100 Application Data
   11 0.295093    10.10.8.222        172.217.166.78   TCP        54 64383 → 443 [ACK] Seq=778 Ack=164 Win=256 Len=0
   12 0.295219    10.10.8.222        172.217.166.78   TLSv1.2   100 Application Data
```

```
Ethernet II, Src: AewinTec_30:03:99 (00:0d:48:30:03:99), Dst: IntelCor_8f:6f:51 (c8:21:58:8f:6f:51)
  > Destination: IntelCor_8f:6f:51 (c8:21:58:8f:6f:51)
  > Source: AewinTec_30:03:99 (00:0d:48:30:03:99)
    Type: IPv4 (0x0800)
    Padding: 000000000000
Internet Protocol Version 4, Src: 172.217.166.78, Dst: 10.10.8.222
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 40
    Identification: 0x6387 (25479)
  > Flags: 0x0000
    Time to live: 58
    Protocol: TCP (6)
    Header checksum: 0xb739 [validation disabled]
```

```
0000  c8 21 58 8f 6f 51 00 0d  48 30 03 99 08 00 45 00   ·!X·oQ·· H0····E·
0010  00 28 63 87 00 00 3a 06  b7 39 ac d9 a6 4e 0a 0a   ·(c···:· ·9···N··
0020  08 de 01 bb fb 7f 31 dc  b7 dc 0d ab d9 7a 50 10   ······1· ·····zP·
0030  02 c3 78 e8 00 00 00 00  00 00 00 00               ··x····· ····
```

Figure 3: Wireshark Interface

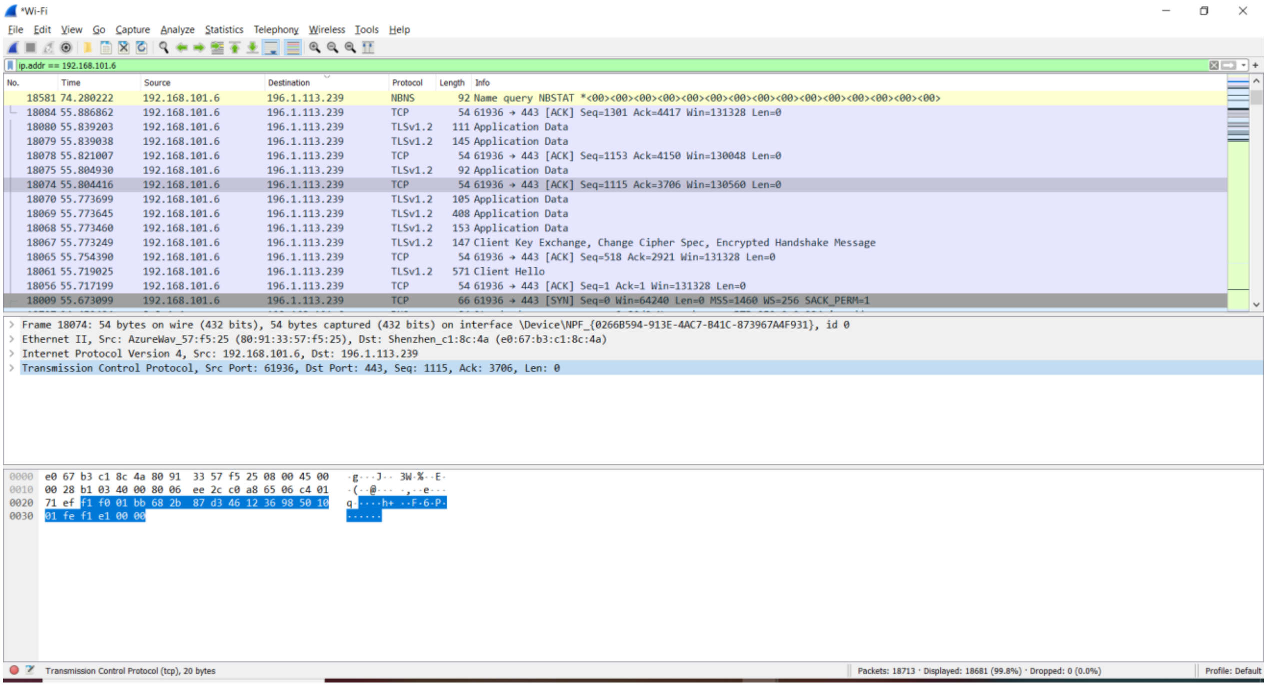
4.4 PACKET ANALYSIS OF DIFFERENT PROTOCOLS

4.4.1 IP Protocol

Internet Protocol is basically used for Transmission. It is an unreliable and connectionless Protocol. To make IP reliable we pair it with TCP which is a reliable Protocol and works at the transport layer.

- **Differentiated Services Code Point[4]:** Earlier it was known as Type of services. It allows the selection of delivery in terms of Delay, Throughput, Reliability and Cost.
- **Following steps are included for analysis of the IP packet**
  i. Inspecting Packets of IP protocols using different display filters.
  ii. Using "IP" as a Display Filter to view IP traffic.
  iii. To view traffic from specific host, say 192.168.101.6, we'll write "ip.addr == 192.168.101.6".

[4] Differentiated Services Code Point (DSCP) is a means of classifying and managing network traffic and of providing quality of service (QoS) in modern Layer 3 IP networks. It uses the 6-bit Differentiated Services (DS) field in the IP header for the purpose of packet classification. Available at https://www.dialogic.com/glossary/differentiated-services-code-point-dscp#:~:text=Differentiated%20Services%20Code%20Point%20(DSCP)%20is%20a%20means%20of%20classifying,the%20purpose%20of%20packet%20classification. Accessed on 21-03-2020.


*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

ip.addr == 192.168.101.6

```
No.   Time      Source         Destination    Protocol Length Info
18581 74.280222 192.168.101.6  196.1.113.239  NBNS      92 Name query NBSTAT *<00><00><00><00><00><00><00><00><00><00><00><00><00><00><00>
18084 55.886862 192.168.101.6  196.1.113.239  TCP       54 61936 → 443 [ACK] Seq=1301 Ack=4417 Win=131328 Len=0
18080 55.839203 192.168.101.6  196.1.113.239  TLSv1.2  111 Application Data
18079 55.839038 192.168.101.6  196.1.113.239  TLSv1.2  145 Application Data
18078 55.821007 192.168.101.6  196.1.113.239  TCP       54 61936 → 443 [ACK] Seq=1153 Ack=4150 Win=130048 Len=0
18075 55.804930 192.168.101.6  196.1.113.239  TLSv1.2   92 Application Data
18074 55.804416 192.168.101.6  196.1.113.239  TCP       54 61936 → 443 [ACK] Seq=1115 Ack=3706 Win=130560 Len=0
18070 55.773699 192.168.101.6  196.1.113.239  TLSv1.2  105 Application Data
18069 55.773645 192.168.101.6  196.1.113.239  TLSv1.2  408 Application Data
18068 55.773460 192.168.101.6  196.1.113.239  TLSv1.2  153 Application Data
18067 55.773249 192.168.101.6  196.1.113.239  TLSv1.2  147 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
18065 55.754390 192.168.101.6  196.1.113.239  TCP       54 61936 → 443 [ACK] Seq=518 Ack=2921 Win=131328 Len=0
18061 55.719025 192.168.101.6  196.1.113.239  TLSv1.2  571 Client Hello
18056 55.717199 192.168.101.6  196.1.113.239  TCP       54 61936 → 443 [ACK] Seq=1 Ack=1 Win=131328 Len=0
18009 55.673099 192.168.101.6  196.1.113.239  TCP       66 61936 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
```

```
> Frame 18074: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface \Device\NPF_{0266B594-913E-4AC7-B41C-873967A4F931}, id 0
> Ethernet II, Src: AzureWav_57:f5:25 (80:91:33:57:f5:25), Dst: Shenzhen_c1:8c:4a (e0:67:b3:c1:8c:4a)
> Internet Protocol Version 4, Src: 192.168.101.6, Dst: 196.1.113.239
> Transmission Control Protocol, Src Port: 61936, Dst Port: 443, Seq: 1115, Ack: 3706, Len: 0
```

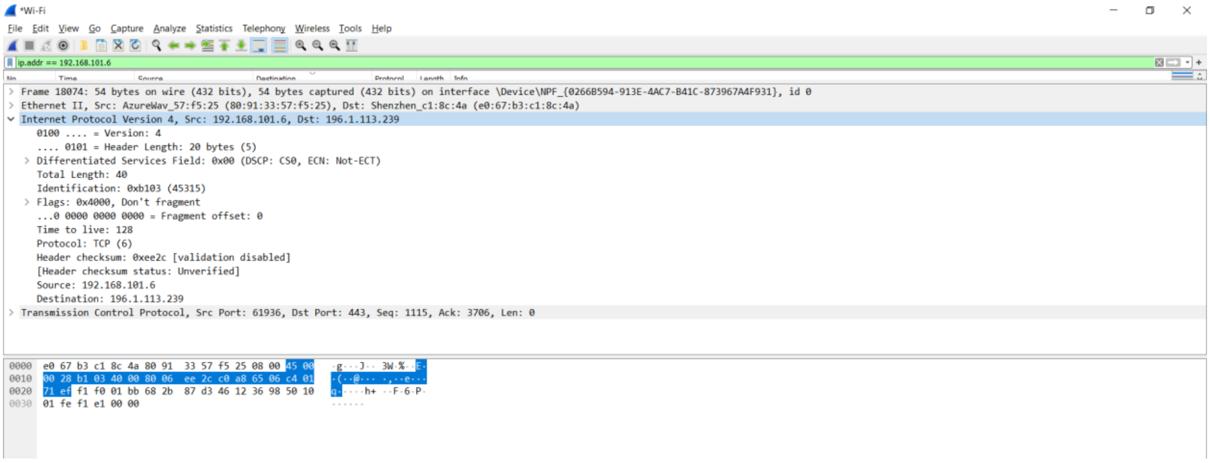
```
0000  e0 67 b3 c1 8c 4a 80 91  33 57 f5 25 08 00 45 00   ·g···J·· 3W·%··E·
0010  00 28 b1 03 40 00 80 06  ee 2c c0 a8 65 06 c4 01   ·(··@··· ·,··e···
0020  71 ef f1 f0 01 bb 68 2b  87 d3 46 12 36 98 50 10   q·····h+ ··F·6·P·
0030  01 fe f1 e1 00 00                                  ······
```

```
Transmission Control Protocol (tcp), 20 bytes    Packets: 18713 · Displayed: 18681 (99.8%) · Dropped: 0 (0.0%)    Profile: Default
```

Figure 4: Inspecting Packets of IP protocols using different display filters

Note: Highlighting the network layers for more details.

In Figure 5 , The IP version is 4, Header length is of 20 bytes, DSCP and ECN are both set to 0, the total length is 40, The Identification is also 0 which means the data is not fragmented, Inside the Flag dropdown it is set to D which means Don't Fragment. TTL is 128 which means it'll die after 128 hops; the protocol used is TCP, Header checksum is disabled, the Source and the destination address are 192.168.101.6 and 196.1.113.239 respectively.


*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

ip.addr == 192.168.101.6

```
> Frame 18074: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface \Device\NPF_{0266B594-913E-4AC7-B41C-873967A4F931}, id 0
> Ethernet II, Src: AzureWav_57:f5:25 (80:91:33:57:f5:25), Dst: Shenzhen_c1:8c:4a (e0:67:b3:c1:8c:4a)
v Internet Protocol Version 4, Src: 192.168.101.6, Dst: 196.1.113.239
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 40
    Identification: 0xb103 (45315)
  > Flags: 0x4000, Don't fragment
    ...0 0000 0000 0000 = Fragment offset: 0
    Time to live: 128
    Protocol: TCP (6)
    Header checksum: 0xee2c [validation disabled]
    [Header checksum status: Unverified]
    Source: 192.168.101.6
    Destination: 196.1.113.239
> Transmission Control Protocol, Src Port: 61936, Dst Port: 443, Seq: 1115, Ack: 3706, Len: 0
```

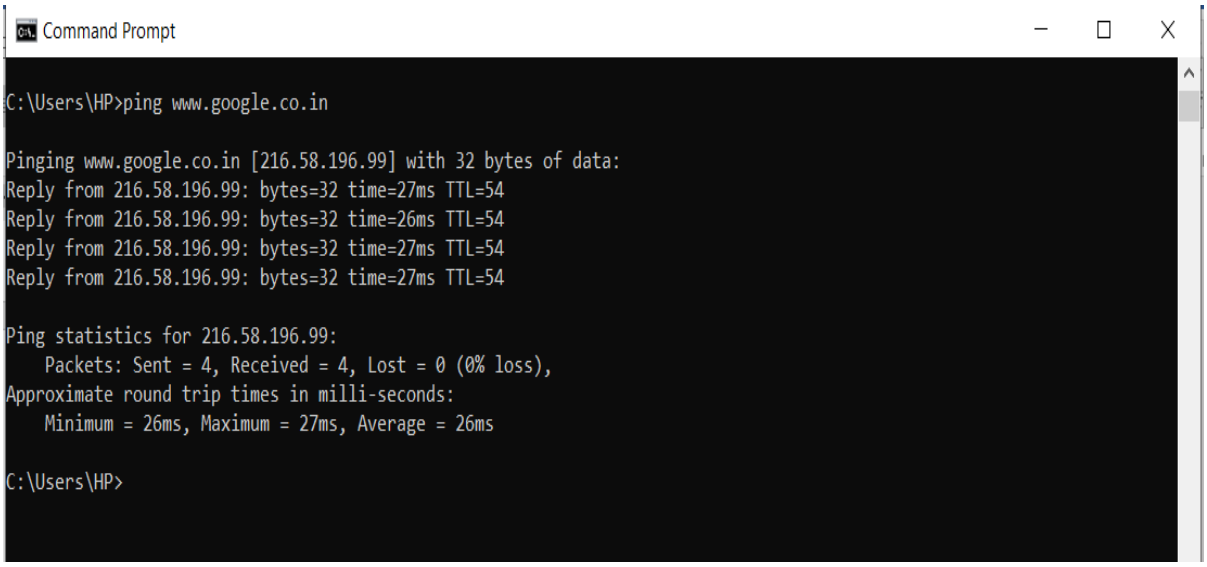
```
0000  e0 67 b3 c1 8c 4a 80 91  33 57 f5 25 08 00 45 00   ·g···J·· 3W·%··E·
0010  00 28 b1 03 40 00 80 06  ee 2c c0 a8 65 06 c4 01   ·(··@··· ·,··e···
0020  71 ef f1 f0 01 bb 68 2b  87 d3 46 12 36 98 50 10   q·····h+ ··F·6·P·
0030  01 fe f1 e1 00 00                                  ······
```

Figure 5: Header Details

4.4.2 ICMP Protocol

The Internet Control Message Protocol is used for reporting errors and various queries as IP does not have a built-in mechanism for queries and error reporting messages.

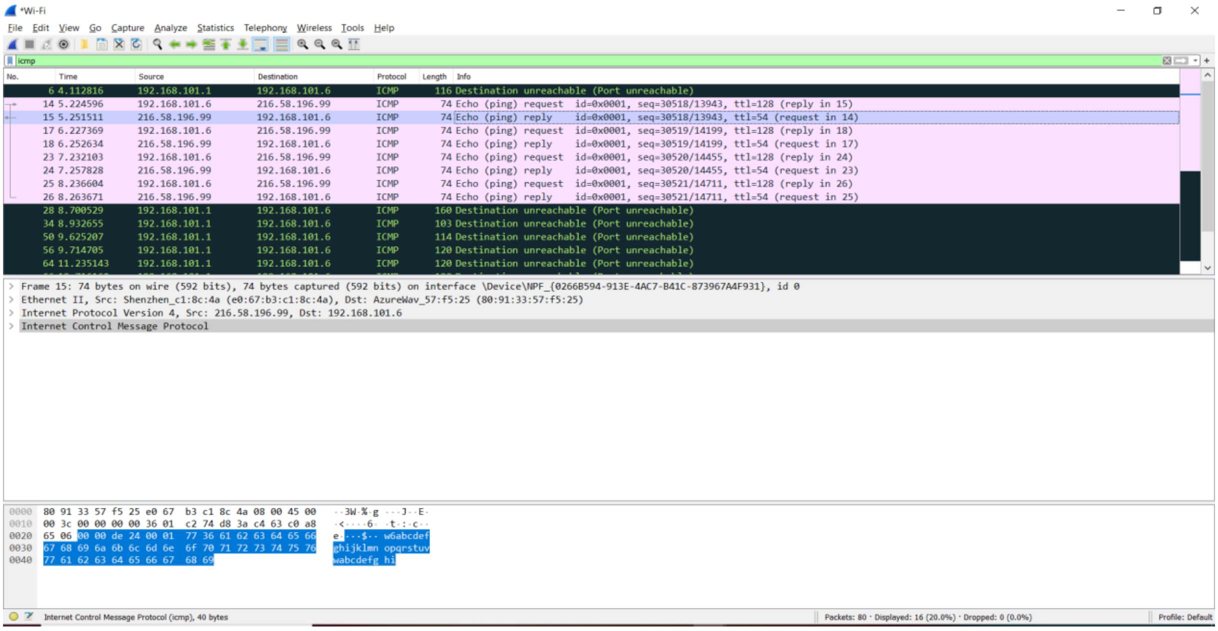
- For ICMP Echo Traffic, we'll **ping www.google.co.in** in cmd.



```
C:\Users\HP>ping www.google.co.in

Pinging www.google.co.in [216.58.196.99] with 32 bytes of data:
Reply from 216.58.196.99: bytes=32 time=27ms TTL=54
Reply from 216.58.196.99: bytes=32 time=26ms TTL=54
Reply from 216.58.196.99: bytes=32 time=27ms TTL=54
Reply from 216.58.196.99: bytes=32 time=27ms TTL=54

Ping statistics for 216.58.196.99:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 26ms, Maximum = 27ms, Average = 26ms

C:\Users\HP>
```

Figure 6: Pinging www.google.co.in in cmd.

- Simultaneously, we'll start the capture using icmp filter and stop after pinging.



*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

icmp

```
No.   Time        Source           Destination      Protocol  Length  Info
    6 4.112816    192.168.101.1    192.168.101.6    ICMP       116 Destination unreachable (Port unreachable)
   14 5.224596    192.168.101.6    216.58.196.99    ICMP        74 Echo (ping) request  id=0x0001, seq=30518/13943, ttl=128 (reply in 15)
   15 5.251511    216.58.196.99    192.168.101.6    ICMP        74 Echo (ping) reply    id=0x0001, seq=30518/13943, ttl=54 (request in 14)
   17 6.227369    192.168.101.6    216.58.196.99    ICMP        74 Echo (ping) request  id=0x0001, seq=30519/14199, ttl=128 (reply in 18)
   18 6.252634    216.58.196.99    192.168.101.6    ICMP        74 Echo (ping) reply    id=0x0001, seq=30519/14199, ttl=54 (request in 17)
   23 7.232103    192.168.101.6    216.58.196.99    ICMP        74 Echo (ping) request  id=0x0001, seq=30520/14455, ttl=128 (reply in 24)
   24 7.257828    216.58.196.99    192.168.101.6    ICMP        74 Echo (ping) reply    id=0x0001, seq=30520/14455, ttl=54 (request in 23)
   25 8.236604    192.168.101.6    216.58.196.99    ICMP        74 Echo (ping) request  id=0x0001, seq=30521/14711, ttl=128 (reply in 26)
   26 8.263671    216.58.196.99    192.168.101.6    ICMP        74 Echo (ping) reply    id=0x0001, seq=30521/14711, ttl=54 (request in 25)
   28 8.700529    192.168.101.1    192.168.101.6    ICMP       160 Destination unreachable (Port unreachable)
   34 8.932655    192.168.101.1    192.168.101.6    ICMP       103 Destination unreachable (Port unreachable)
   50 9.625207    192.168.101.1    192.168.101.6    ICMP       114 Destination unreachable (Port unreachable)
   56 9.714705    192.168.101.1    192.168.101.6    ICMP       120 Destination unreachable (Port unreachable)
   64 11.235143   192.168.101.1    192.168.101.6    ICMP       120 Destination unreachable (Port unreachable)
```

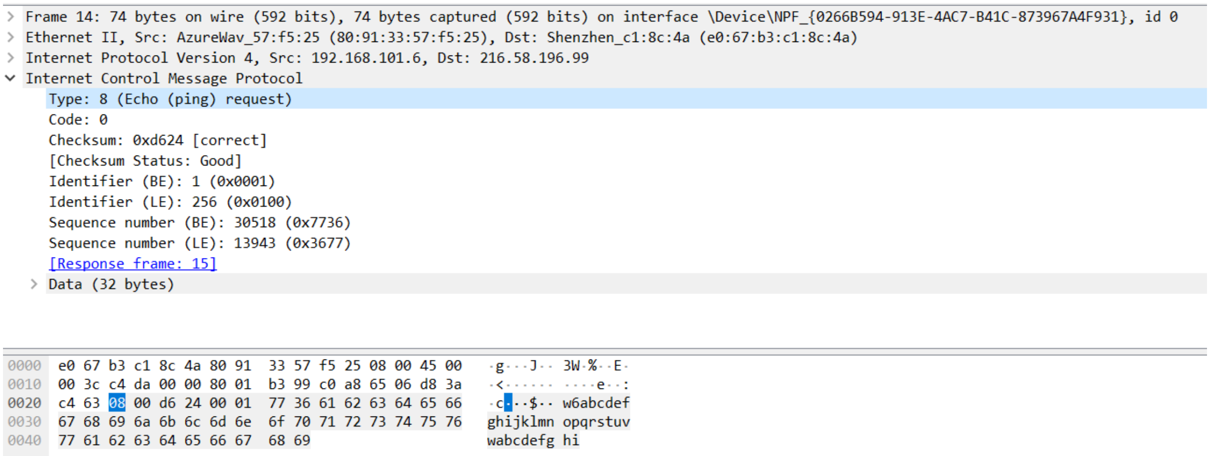
```
> Frame 15: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{0266B594-913E-4AC7-B41C-873967A4F931}, id 0
> Ethernet II, Src: Shenzhen_c1:8c:4a (e0:67:b3:c1:8c:4a), Dst: AzureWav_57:f5:25 (80:91:33:57:f5:25)
> Internet Protocol Version 4, Src: 216.58.196.99, Dst: 192.168.101.6
> Internet Control Message Protocol
```

```
0000  80 91 33 57 f5 25 e0 67  b3 c1 8c 4a 08 00 45 00   ··3W·%·g ···J··E·
0010  00 3c 00 00 00 00 36 01  c2 74 d8 3a c4 63 c0 a8   ·<····6· ·t·:·c··
0020  65 06 00 00 de 24 00 01  77 36 61 62 63 64 65 66   e····$·· w6abcdef
0030  67 68 69 6a 6b 6c 6d 6e  6f 70 71 72 73 74 75 76   ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67  68 69                     wabcdefg hi
```

Internet Control Message Protocol (icmp), 40 bytes | Packets: 80 · Displayed: 16 (20.0%) · Dropped: 0 (0.0%) | Profile: Default

Figure 7: ICMP Filter

- **ICMP Echo request**



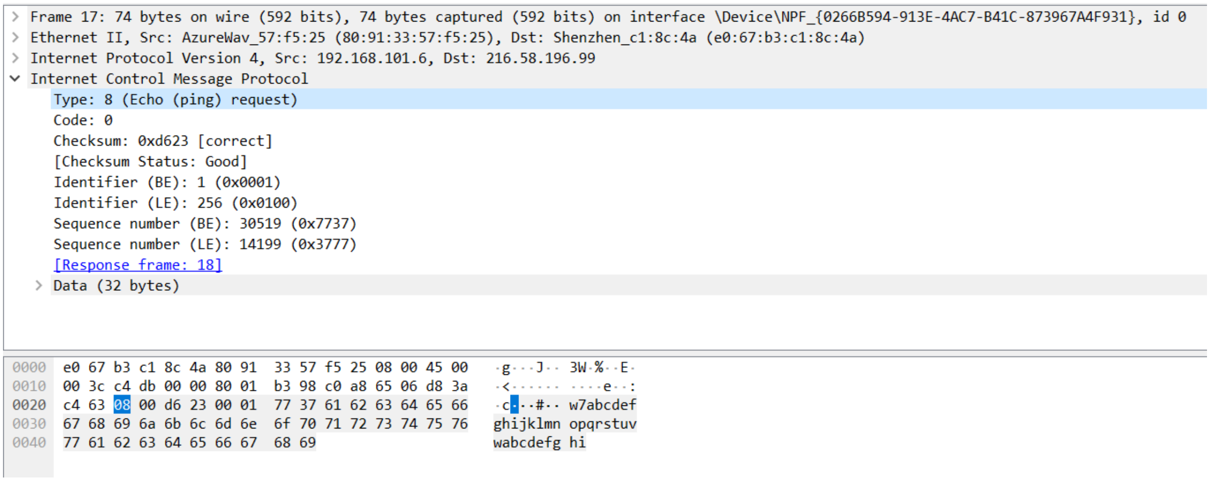
```
> Frame 14: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{0266B594-913E-4AC7-B41C-873967A4F931}, id 0
> Ethernet II, Src: AzureWav_57:f5:25 (80:91:33:57:f5:25), Dst: Shenzhen_c1:8c:4a (e0:67:b3:c1:8c:4a)
> Internet Protocol Version 4, Src: 192.168.101.6, Dst: 216.58.196.99
v Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0xd624 [correct]
    [Checksum Status: Good]
    Identifier (BE): 1 (0x0001)
    Identifier (LE): 256 (0x0100)
    Sequence number (BE): 30518 (0x7736)
    Sequence number (LE): 13943 (0x3677)
    [Response frame: 15]
  > Data (32 bytes)

0000  e0 67 b3 c1 8c 4a 80 91  33 57 f5 25 08 00 45 00   ·g···J·· 3W·%··E·
0010  00 3c c4 da 00 00 80 01  b3 99 c0 a8 65 06 d8 3a   ·<······ ····e··:
0020  c4 63 08 00 d6 24 00 01  77 36 61 62 63 64 65 66   ·c···$·· w6abcdef
0030  67 68 69 6a 6b 6c 6d 6e  6f 70 71 72 73 74 75 76   ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67  68 69                     wabcdefg hi
```

Figure 8: ICMP Echo Request

Type refers to the type of message it is 8 here means it’s an echo request and code is 0 which means it is used for ping, checksum is used for error checking and here its status is good means no errors are there. Identifier field is basically an ID value that is assigned to the message. Sequence number is the no. for each host.

- **ICMP Echo Reply**

Type 0 refers to the type of message used for echo reply, code is request, checksum status is good and correct. Identifier is value that assigned to the message which is same as request packet. Sequence number is 30519 in BE and 14199 in LE, here Request Frame is displaying for convenience.



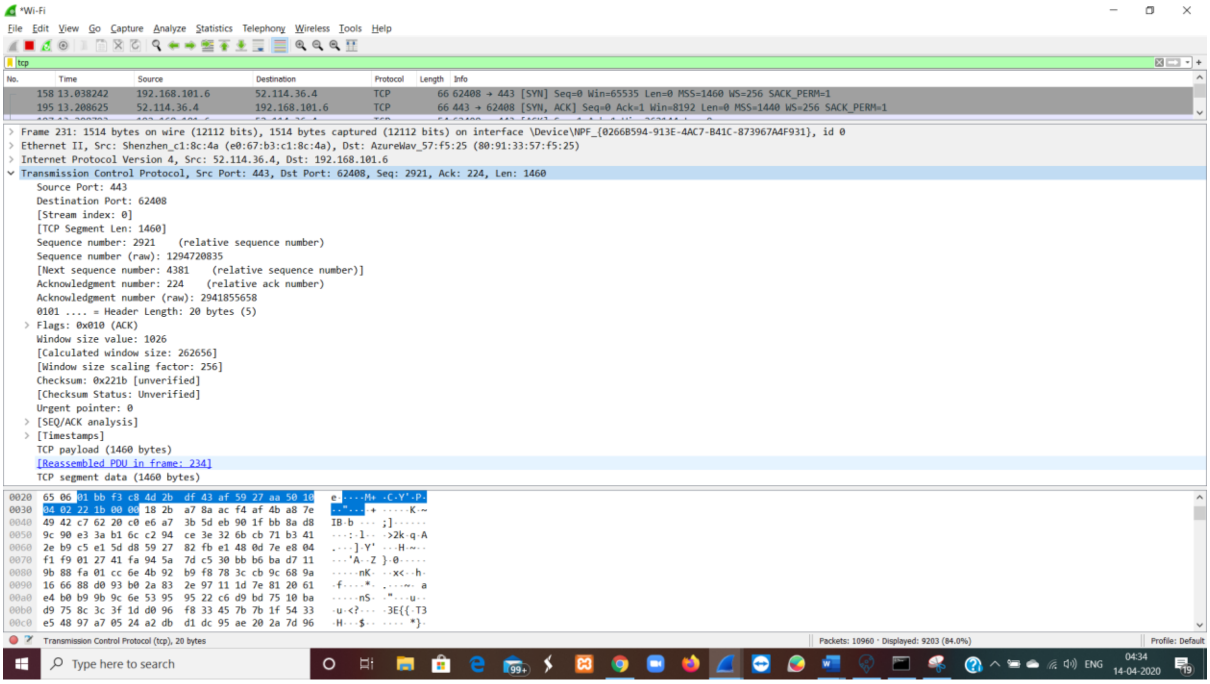
```
> Frame 17: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{0266B594-913E-4AC7-B41C-873967A4F931}, id 0
> Ethernet II, Src: AzureWav_57:f5:25 (80:91:33:57:f5:25), Dst: Shenzhen_c1:8c:4a (e0:67:b3:c1:8c:4a)
> Internet Protocol Version 4, Src: 192.168.101.6, Dst: 216.58.196.99
v Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0xd623 [correct]
    [Checksum Status: Good]
    Identifier (BE): 1 (0x0001)
    Identifier (LE): 256 (0x0100)
    Sequence number (BE): 30519 (0x7737)
    Sequence number (LE): 14199 (0x3777)
    [Response frame: 18]
  > Data (32 bytes)

0000  e0 67 b3 c1 8c 4a 80 91  33 57 f5 25 08 00 45 00   ·g···J·· 3W·%··E·
0010  00 3c c4 db 00 00 80 01  b3 98 c0 a8 65 06 d8 3a   ·<······ ····e··:
0020  c4 63 08 00 d6 23 00 01  77 37 61 62 63 64 65 66   ·c···#·· w7abcdef
0030  67 68 69 6a 6b 6c 6d 6e  6f 70 71 72 73 74 75 76   ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67  68 69                     wabcdefg hi
```

Figure 9: ICMP Echo Reply

4.4.3 Transmission Control Protocol

TCP provides reliable and error-free transmission between the two ends. It is a connection-oriented protocol. TCP divides the data into small pieces called Segments.

- Capturing TCP Packets using Display Filter **“TCP”.**



```
*Wi-Fi
tcp
No.     Time        Source          Destination     Protocol Length Info
    158 13.038242   192.168.101.6   52.114.36.4     TCP          66 62408 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
    195 13.208625   52.114.36.4     192.168.101.6   TCP          66 443 → 62408 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1440 WS=256 SACK_PERM=1

> Frame 231: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface \Device\NPF_{0266B594-913E-4AC7-B41C-873967A4F931}, id 0
> Ethernet II, Src: Shenzhen_c1:8c:4a (e0:67:b3:c1:8c:4a), Dst: AzureWav_57:f5:25 (80:91:33:57:f5:25)
> Internet Protocol Version 4, Src: 52.114.36.4, Dst: 192.168.101.6
v Transmission Control Protocol, Src Port: 443, Dst Port: 62408, Seq: 2921, Ack: 224, Len: 1460
    Source Port: 443
    Destination Port: 62408
    [Stream index: 0]
    [TCP Segment Len: 1460]
    Sequence number: 2921    (relative sequence number)
    Sequence number (raw): 1294720835
    [Next sequence number: 4381    (relative sequence number)]
    Acknowledgment number: 224    (relative ack number)
    Acknowledgment number (raw): 2941855658
    0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x010 (ACK)
    Window size value: 1026
    [Calculated window size: 262656]
    [Window size scaling factor: 256]
    Checksum: 0x221b [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  > [SEQ/ACK analysis]
  > [Timestamps]
    TCP payload (1460 bytes)
    [Reassembled PDU in frame: 234]
    TCP segment data (1460 bytes)

0020  65 06 01 bb f3 c8 4d 2b  df 43 af 59 27 aa 50 10   e·····M+ ·C·Y'·P·
0030  04 02 22 1b 00 00 18 2b  a7 8a ac f4 af 4b a8 7e   ··"····+ ·····K·~
0040  49 42 c7 62 20 c0 e6 a7  3b 5d eb 90 1f bb 8a d8   IB·b ··· ;]······
0050  9c 90 e3 3a b1 6c c2 94  ce 3e 32 6b cb 71 b3 41   ···:·l·· ·>2k·q·A
0060  2e b9 c5 e1 5d d8 59 27  82 fb e1 48 0d 7e e8 04   .···]·Y' ···H·~··
0070  f1 f9 01 27 41 fa 94 5a  7d c5 30 bb b6 ba d7 11   ···'A··Z }·0·····
0080  9b 88 fa 01 cc 6e 4b 92  b9 f8 78 3c cb 9c 68 9a   ·····nK· ··x<··h·
0090  16 66 88 d0 93 b0 2a 83  2e 97 11 1d 7e 81 20 61   ·f····*· .···~· a
00a0  e4 b0 b9 9b 9c 6e 53 95  95 22 c6 d9 bd 75 10 ba   ·····nS· ·"···u··
00b0  d9 75 8c 3c 3f 1d d0 96  f8 33 45 7b 7b 1f 54 33   ·u·<?··· ·3E{{·T3
00c0  e5 48 97 a7 05 24 a2 db  d1 dc 95 ae 20 2a 7d 96   ·H···$·· ···· *}·

Transmission Control Protocol (tcp), 20 bytes      Packets: 10960 · Displayed: 9203 (84.0%)      Profile: Default
```

Figure 10: TCP Details.

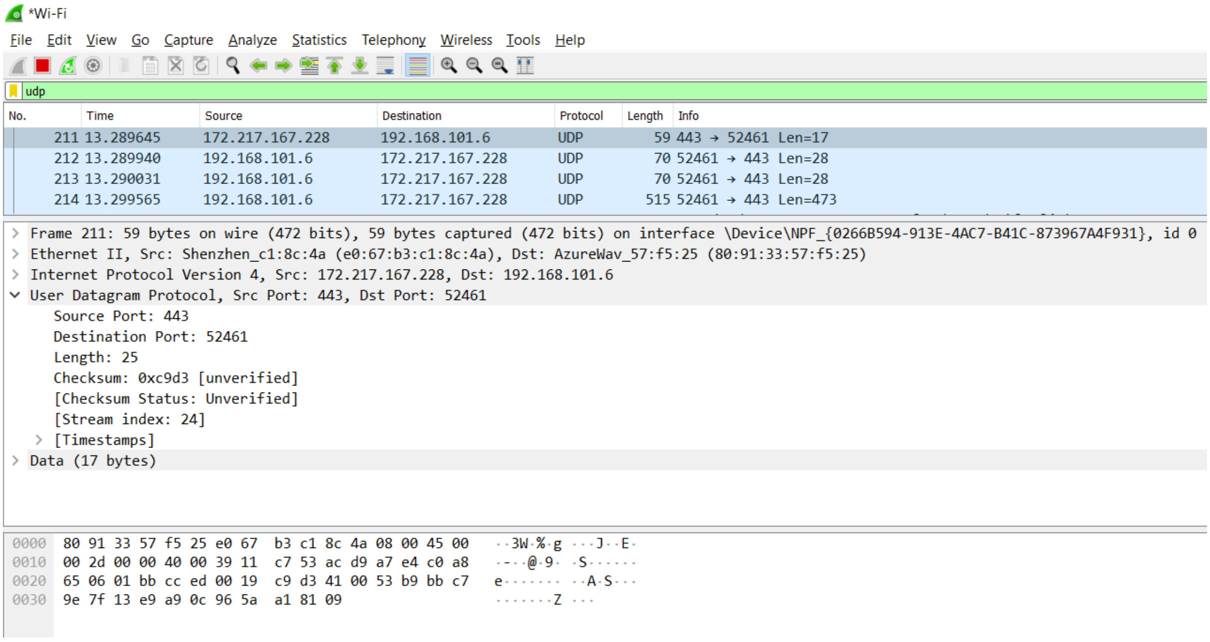
In the Figure 9, we can see that the source port is 443 which is https, and the destination port is 62408 which is TCP, Sequence number is 2921, acknowledgement number is 224, the header length is 20 bytes, the Flag is only set to SYN: 1, the window size value is 1026, checksum is not verified or not performed and the urgent pointer is set to 0.

4.4.4 User Datagram Protocol

The User Datagram Protocol (UDP) is one of the simplest protocols in the TCP/IP Protocol. It is Unreliable and Connectionless protocol[5]. So, there is no need for establishing a connection before sending the data.

- The user datagram is where application works with the use of ports. The packet is captured using display filter “UDP”.

[5] A connectionless protocol (or protocol layer) does not have a sequence number and acknowledgment scheme. That’s all. No sequence numbers mean that a protocol is ‘connectionless’. It could also be referred to as a ‘datagram’. Available at https://www.liveaction.com/docs/glossary/llc-ieee-802-2-logical-link-control/defining-connection-less-and-connection-oriented/ accessed on 22-03-2020.



*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
udp

```
No.   Time        Source           Destination      Protocol  Length  Info
  211 13.289645   172.217.167.228  192.168.101.6    UDP           59  443 → 52461 Len=17
  212 13.289940   192.168.101.6    172.217.167.228  UDP           70  52461 → 443 Len=28
  213 13.290031   192.168.101.6    172.217.167.228  UDP           70  52461 → 443 Len=28
  214 13.299565   192.168.101.6    172.217.167.228  UDP          515  52461 → 443 Len=473
```

```
> Frame 211: 59 bytes on wire (472 bits), 59 bytes captured (472 bits) on interface \Device\NPF_{0266B594-913E-4AC7-B41C-873967A4F931}, id 0
> Ethernet II, Src: Shenzhen_c1:8c:4a (e0:67:b3:c1:8c:4a), Dst: AzureWav_57:f5:25 (80:91:33:57:f5:25)
> Internet Protocol Version 4, Src: 172.217.167.228, Dst: 192.168.101.6
v User Datagram Protocol, Src Port: 443, Dst Port: 52461
    Source Port: 443
    Destination Port: 52461
    Length: 25
    Checksum: 0xc9d3 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 24]
  > [Timestamps]
> Data (17 bytes)
```

```
0000  80 91 33 57 f5 25 e0 67  b3 c1 8c 4a 08 00 45 00   ··3W·%·g ···J··E·
0010  00 2d 00 00 40 00 39 11  c7 53 ac d9 a7 e4 c0 a8   ·-··@·9· ·S······
0020  65 06 01 bb cc ed 00 19  c9 d3 41 00 53 b9 bb c7   e······· ··A·S···
0030  9e 7f 13 e9 a9 0c 96 5a  a1 81 09                 ·······Z ···
```

Figure 11: UDP Details

In Figure 10, we can see that the source port is 443 which is https and the destination port are 52461 which is UDP, the length of UDP is 25 bytes which is the total of header information and the data. The UDP checksum is optional and in the case of the UDP Inspection it is unverified, and its status is unverified as well. The length of the data is 17 bytes.

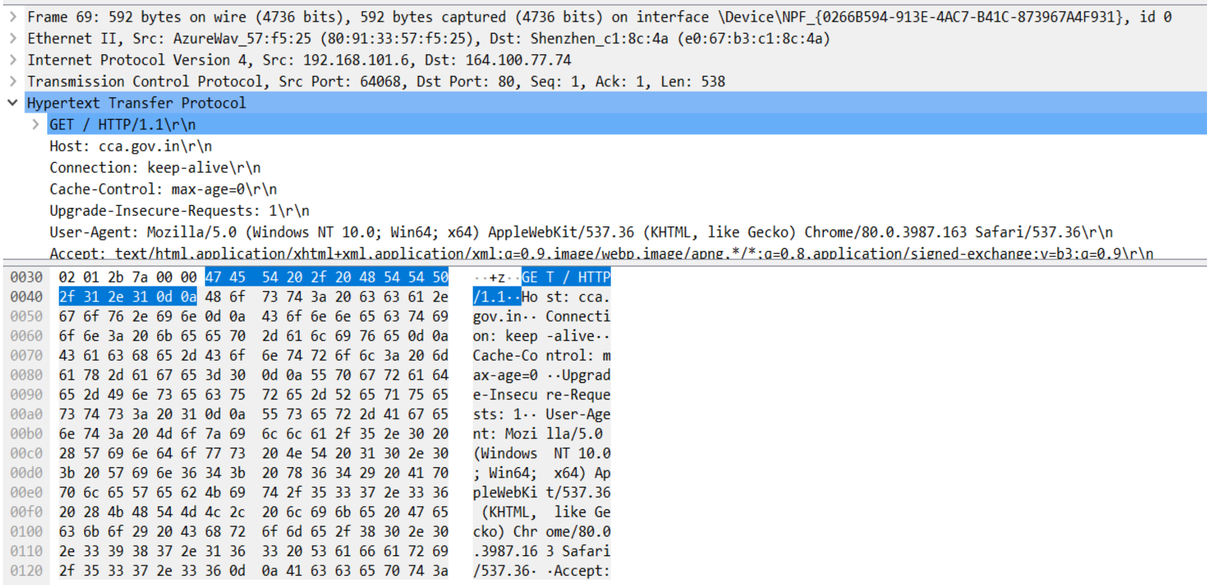
4.4.5 Hyper Text Transfer Protocol (HTTP)

The Hyper Text Transfer Protocol (HTTP) is a text-based request-response protocol that works on the Application layer of TCP/IP.

- **HTTP Request**

HTTP Request: A message request sent by the client.

- ➢ We'll capture packets using display filter**"http"**.
- ➢ In the Wireshark, we're going to look for get request from the info field.
- ➢ After selecting the particular packet, we'll look for the particular header details.



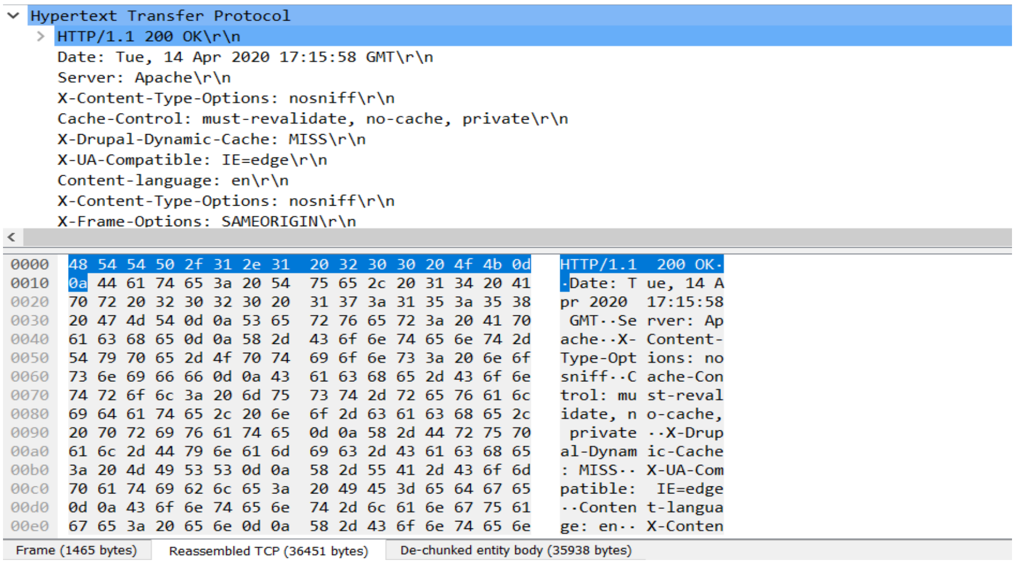
```
> Frame 69: 592 bytes on wire (4736 bits), 592 bytes captured (4736 bits) on interface \Device\NPF_{0266B594-913E-4AC7-B41C-873967A4F931}, id 0
> Ethernet II, Src: AzureWav_57:f5:25 (80:91:33:57:f5:25), Dst: Shenzhen_c1:8c:4a (e0:67:b3:c1:8c:4a)
> Internet Protocol Version 4, Src: 192.168.101.6, Dst: 164.100.77.74
> Transmission Control Protocol, Src Port: 64068, Dst Port: 80, Seq: 1, Ack: 1, Len: 538
v Hypertext Transfer Protocol
  > GET / HTTP/1.1\r\n
    Host: cca.gov.in\r\n
    Connection: keep-alive\r\n
    Cache-Control: max-age=0\r\n
    Upgrade-Insecure-Requests: 1\r\n
    User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.163 Safari/537.36\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9\r\n

0030  02 01 2b 7a 00 00 47 45  54 20 2f 20 48 54 54 50   ··+z··GE T / HTTP
0040  2f 31 2e 31 0d 0a 48 6f  73 74 3a 20 63 63 61 2e   /1.1··Ho st: cca.
0050  67 6f 76 2e 69 6e 0d 0a  43 6f 6e 6e 65 63 74 69   gov.in·· Connecti
0060  6f 6e 3a 20 6b 65 65 70  2d 61 6c 69 76 65 0d 0a   on: keep -alive··
0070  43 61 63 68 65 2d 43 6f  6e 74 72 6f 6c 3a 20 6d   Cache-Co ntrol: m
0080  61 78 2d 61 67 65 3d 30  0d 0a 55 70 67 72 61 64   ax-age=0 ··Upgrad
0090  65 2d 49 6e 73 65 63 75  72 65 2d 52 65 71 75 65   e-Insecu re-Reque
00a0  73 74 73 3a 20 31 0d 0a  55 73 65 72 2d 41 67 65   sts: 1·· User-Age
00b0  6e 74 3a 20 4d 6f 7a 69  6c 6c 61 2f 35 2e 30 20   nt: Mozi lla/5.0 
00c0  28 57 69 6e 64 6f 77 73  20 4e 54 20 31 30 2e 30   (Windows  NT 10.0
00d0  3b 20 57 69 6e 36 34 3b  20 78 36 34 29 20 41 70   ; Win64;  x64) Ap
00e0  70 6c 65 57 65 62 4b 69  74 2f 35 33 37 2e 33 36   pleWebKi t/537.36
00f0  20 28 4b 48 54 4d 4c 2c  20 6c 69 6b 65 20 47 65    (KHTML,  like Ge
0100  63 6b 6f 29 20 43 68 72  6f 6d 65 2f 38 30 2e 30   cko) Chr ome/80.0
0110  2e 33 39 38 37 2e 31 36  33 20 53 61 66 61 72 69   .3987.16 3 Safari
0120  2f 35 33 37 2e 33 36 0d  0a 41 63 63 65 70 74 3a   /537.36· ·Accept:
```

Figure 12: HTTP Request

GET request is made to the server, Host specifies the internet host which is www.cca.gov.in\r\n, Connection: Keep-alive, which means the connection is persistent and not closed, allowing for subsequent requests to the same server to be done. Cache Control: max-age=0 indicates that clients can cache a resource and must revalidate each time before using it, User-agent: shows it is Windows 10 Winx64, and the browser which is Chrome 80.0.3987.163. Upgrade-Insecure-Requests request header sends a signal to the server expressing the client's preference for an encrypted and authenticated response.

- **HTTP Response**

  HTTP Response: A response message from the server.



```
v Hypertext Transfer Protocol
  > HTTP/1.1 200 OK\r\n
    Date: Tue, 14 Apr 2020 17:15:58 GMT\r\n
    Server: Apache\r\n
    X-Content-Type-Options: nosniff\r\n
    Cache-Control: must-revalidate, no-cache, private\r\n
    X-Drupal-Dynamic-Cache: MISS\r\n
    X-UA-Compatible: IE=edge\r\n
    Content-language: en\r\n
    X-Content-Type-Options: nosniff\r\n
    X-Frame-Options: SAMEORIGIN\r\n

0000  48 54 54 50 2f 31 2e 31  20 32 30 30 20 4f 4b 0d   HTTP/1.1  200 OK·
0010  0a 44 61 74 65 3a 20 54  75 65 2c 20 31 34 20 41   ·Date: T ue, 14 A
0020  70 72 20 32 30 32 30 20  31 37 3a 31 35 3a 35 38   pr 2020  17:15:58
0030  20 47 4d 54 0d 0a 53 65  72 76 65 72 3a 20 41 70    GMT··Se rver: Ap
0040  61 63 68 65 0d 0a 58 2d  43 6f 6e 74 65 6e 74 2d   ache··X- Content-
0050  54 79 70 65 2d 4f 70 74  69 6f 6e 73 3a 20 6e 6f   Type-Opt ions: no
0060  73 6e 69 66 66 0d 0a 43  61 63 68 65 2d 43 6f 6e   sniff··C ache-Con
0070  74 72 6f 6c 3a 20 6d 75  73 74 2d 72 65 76 61 6c   trol: mu st-reval
0080  69 64 61 74 65 2c 20 6e  6f 2d 63 61 63 68 65 2c   idate, n o-cache,
0090  20 70 72 69 76 61 74 65  0d 0a 58 2d 44 72 75 70    private ··X-Drup
00a0  61 6c 2d 44 79 6e 61 6d  69 63 2d 43 61 63 68 65   al-Dynam ic-Cache
00b0  3a 20 4d 49 53 53 0d 0a  58 2d 55 41 2d 43 6f 6d   : MISS·· X-UA-Com
00c0  70 61 74 69 62 6c 65 3a  20 49 45 3d 65 64 67 65   patible:  IE=edge
00d0  0d 0a 43 6f 6e 74 65 6e  74 2d 6c 61 6e 67 75 61   ··Conten t-langua
00e0  67 65 3a 20 65 6e 0d 0a  58 2d 43 6f 6e 74 65 6e   ge: en·· X-Conten
```

Frame (1465 bytes)
Reassembled TCP (36451 bytes)
De-chunked entity body (35938 bytes)

Figure 13: HTTP Response

- **In Figure 13, HTTP/1.1 200 OK\r\n** means server agrees for the communication, the **latency** is the amount of time it takes for the **Date** is the date and time when the response received by the host which is 14 April 2020 and time is 17:15:58.

5. NESSUS

Nessus is a free open source security scanner. The Nessus was started in 1998 and the scanner Nessus was released in April 1998. Nessus have all the features described, port scanning, services recognition, information gathering and check for flaws. The advantage of this is that you can combine the best programs from different areas in a Nessus scan. One thing that makes Nessus so powerful is its client server technology. With this technology servers can be placed in many different places in a network and the network can be tested from various points of view. There is the server that performs the actual testing and the client provides configuration and reporting functionality. You can have one central client or multiple distributed clients controlling all the servers. With this feature there comes a two-great deal of flexibility for the vulnerability tester.

5.1 FEATURES OF NESSUS

Various features of Nessus are mentioned below:

i. Nessus is perhaps the only security scanner that has the capability to detect not only the remote flaws of the hosts that are on a network, but their missing patches and local flaws as well.

ii. By using the command Nessus-update-plugins, The Nessus security checks database (which is updated on a daily basis) can be retrieved.

iii. Updating Nessus does not involve downloading potentially threatening binaries from the internet.

iv. Nessus will quickly exploit the systems strengths, so it can increase its scanning efficiency.

v. Nessus includes NASL, (Nessus Attack Scripting Language) a language designed to rapidly write security test.

vi. Nessus can identify a FTP server running on a non-standard port, or a web server running on port 8080.

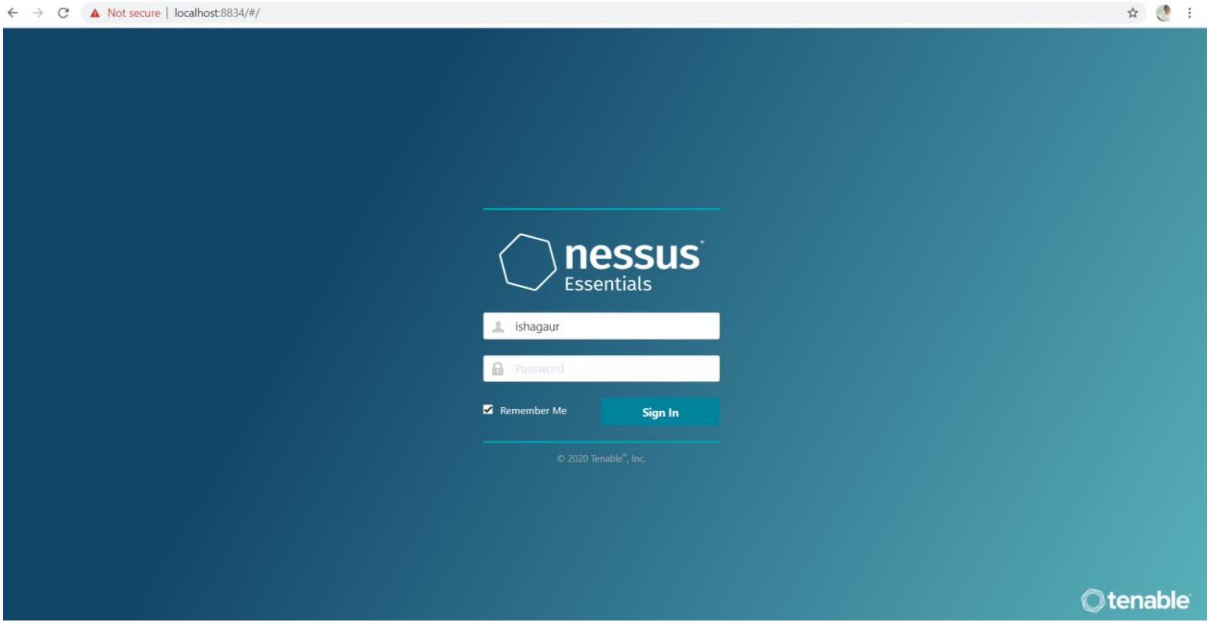
vii. Nessus can test all of the services that are run twice or more by a host run.

viii. Nessus has the capability to test SSLized services such as https, smtps, imaps, and can even be supplied with a certificate so that it can be integrated into a PKI type environment.

ix. Nessus gives you the option to either perform a regular non-destructive security audit on a daily basis, or to throw everything you can at a remote host to test its mettle, and see how it will withstand attacks from intruders.

5.2 NESSUS INTERFACE

After installing Nessus, we have to create an account with user id and password for using Nessus. Below is the image of login page of Nessus:

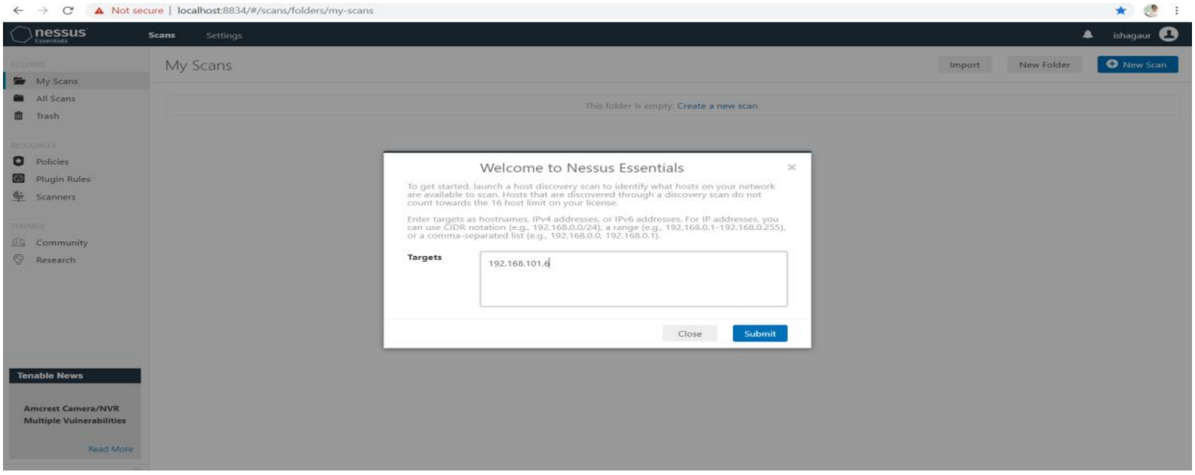


ishagaur

☑ Remember Me

Figure 14: Login page of Nessus

After login it asks for the host address that you target to scan for vulnerabilities. This option is optional, you can skip and go further for other templates of Nessus. In this project we are scanning network So either we can opt for direct scanning or we can select from the templates.



This folder is empty. Create a new scan.

Welcome to Nessus Essentials

To get started, launch a host discovery scan to identify what hosts on your network are available to scan. Hosts that are discovered through a discovery scan do not count towards the 16 host limit on your license.

Enter targets as hostnames, IPv4 addresses, or IPv6 addresses. For IP addresses, you can use CIDR notation (e.g., 192.168.0.0/24), a range (e.g., 192.168.0.1-192.168.0.255), or a comma-separated list (e.g., 192.168.0.0, 192.168.0.1).

Targets: 192.168.101.6

Figure 15: Direct host scanning feature of Nessus.

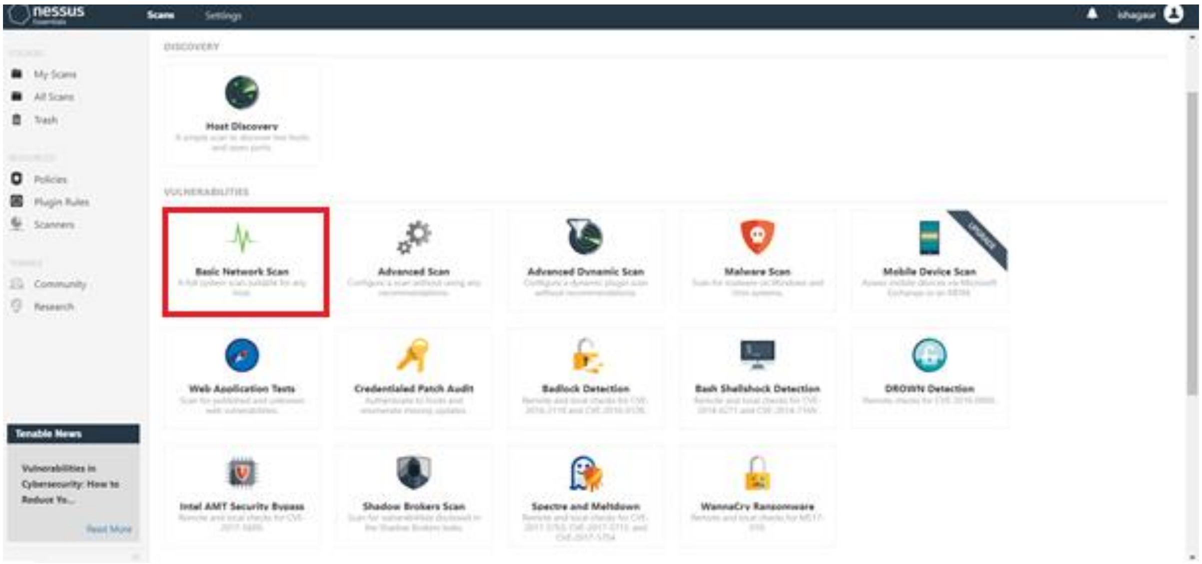
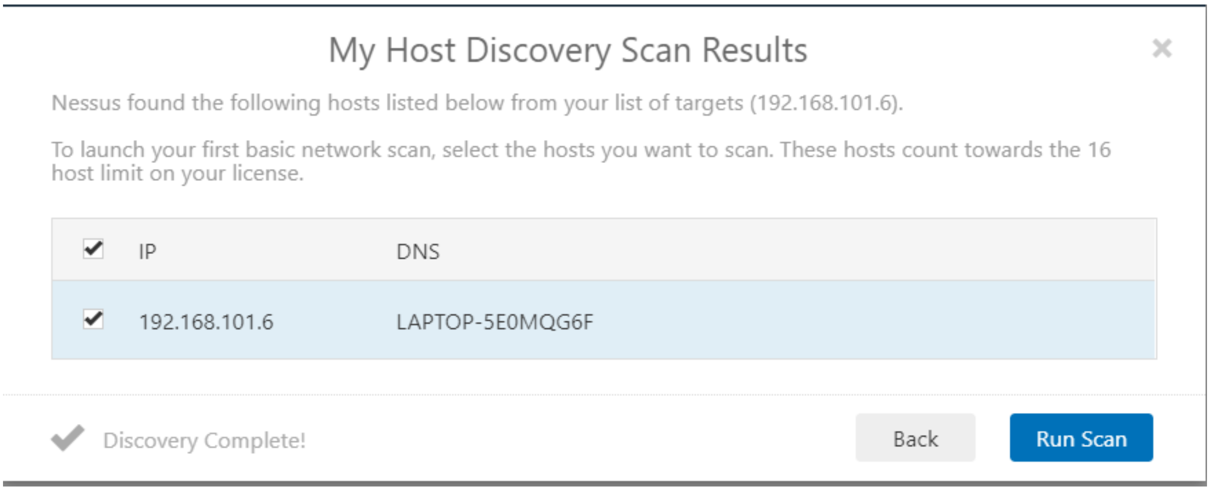


Figure 16: Templates of Nessus



My Host Discovery Scan Results

Nessus found the following hosts listed below from your list of targets (192.168.101.6).

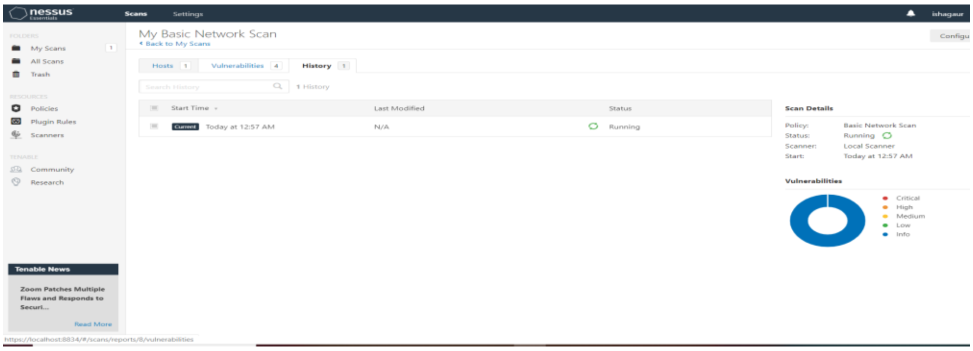
To launch your first basic network scan, select the hosts you want to scan. These hosts count towards the 16 host limit on your license.

☑	IP	DNS
☑	192.168.101.6	LAPTOP-5E0MQG6F

✓ Discovery Complete!

Figure 17: Detection of targeted host for scanning.

Depending on what and how many devices you have on your network, the scan takes a while.



My Basic Network Scan

‹ Back to My Scans

Start Time	Last Modified	Status
Current Today at 12:57 AM	N/A	Running

Scan Details

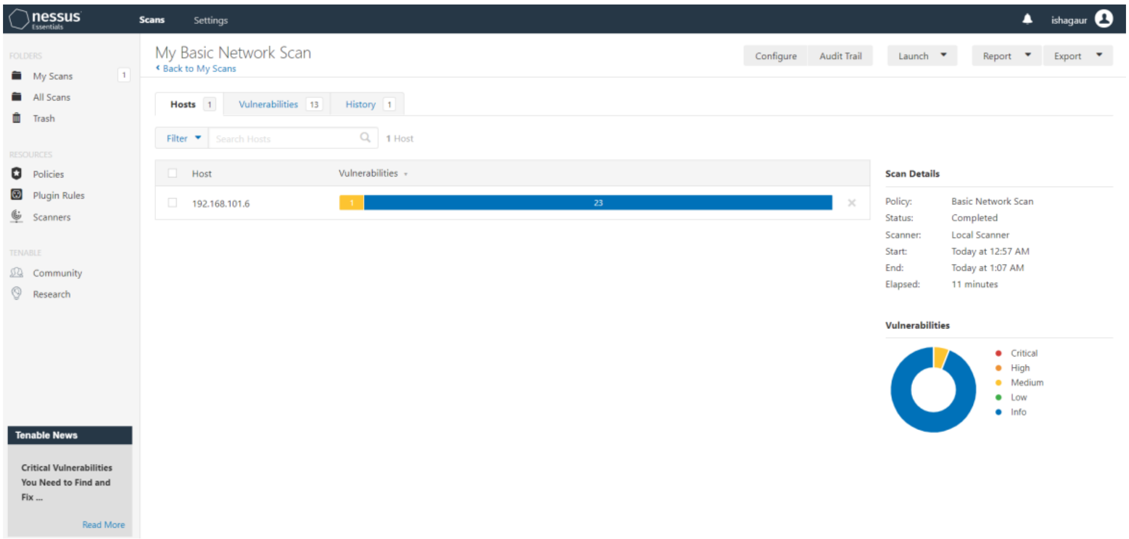
Policy: Basic Network Scan
Status: Running
Scanner: Local Scanner
Start: Today at 12:57 AM

Vulnerabilities

Figure 18: Running status of scan.

5.3 TEST RESULTS OF NETWORK SCAN:

- Result 1:



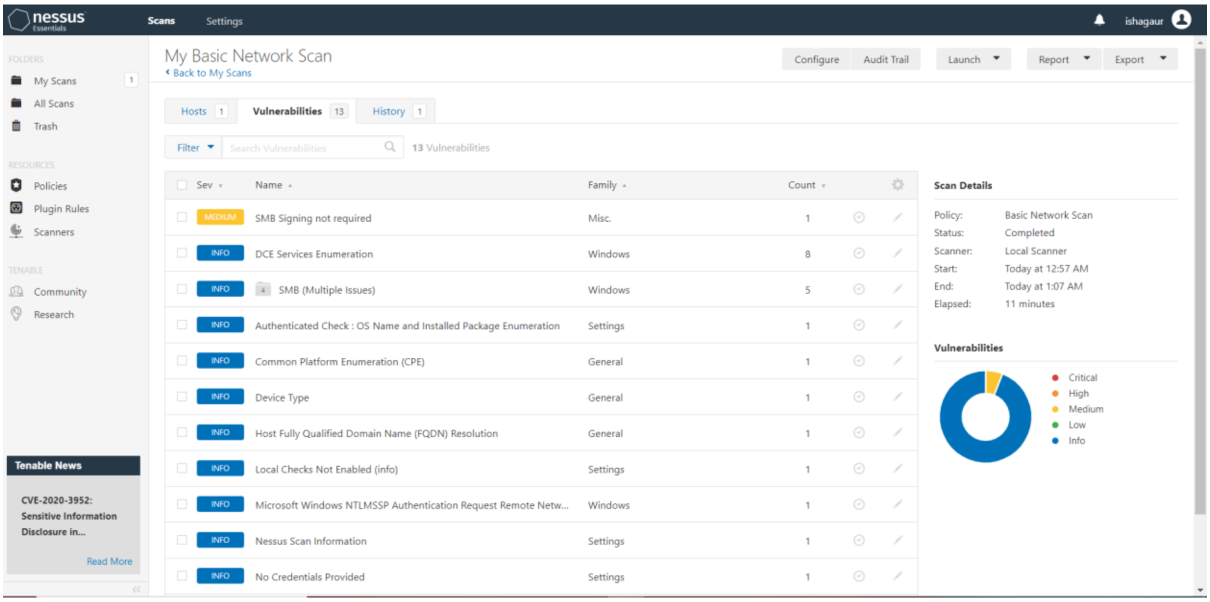
Scan Details

Policy:	Basic Network Scan
Status:	Completed
Scanner:	Local Scanner
Start:	Today at 12:57 AM
End:	Today at 1:07 AM
Elapsed:	11 minutes

Figure 19: Targeted Host scanning report

It took 11 minutes to scan. It resulted in 4.17% medium (Yellow) and 95.83% of information (Blue) vulnerabilities for host 192.168.101.6.

- Result 2:



Sev	Name	Family	Count
MEDIUM	SMB Signing not required	Misc.	1
INFO	DCE Services Enumeration	Windows	8
INFO	4 SMB (Multiple Issues)	Windows	5
INFO	Authenticated Check : OS Name and Installed Package Enumeration	Settings	1
INFO	Common Platform Enumeration (CPE)	General	1
INFO	Device Type	General	1
INFO	Host Fully Qualified Domain Name (FQDN) Resolution	General	1
INFO	Local Checks Not Enabled (info)	Settings	1
INFO	Microsoft Windows NTLMSSP Authentication Request Remote Netw...	Windows	1
INFO	Nessus Scan Information	Settings	1
INFO	No Credentials Provided	Settings	1

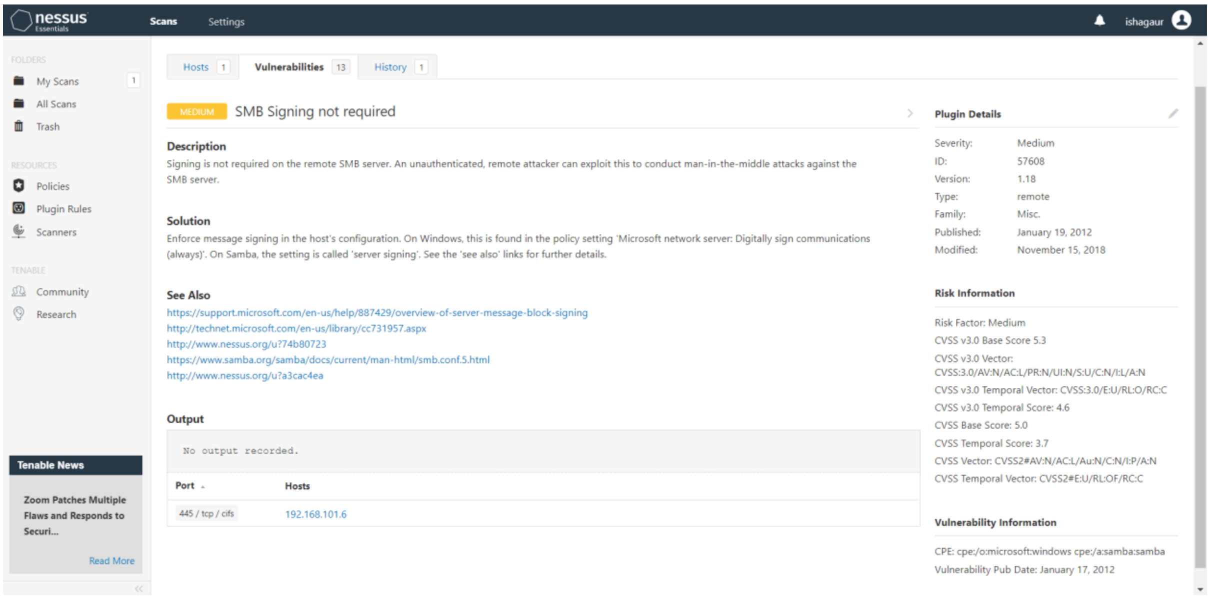
Scan Details

Policy:	Basic Network Scan
Status:	Completed
Scanner:	Local Scanner
Start:	Today at 12:57 AM
End:	Today at 1:07 AM
Elapsed:	11 minutes

Figure 20: Vulnerabilities report of scanning.

Here we found 13 vulnerabilities report. Out of thirteen, one vulnerability was of medium level which belong to miscellaneous family in plugin id 57608. Other vulnerabilities was of info level.

Nessus provides the facility to analyse the vulnerability with description and solution.



MEDIUM SMB Signing not required

Description

Signing is not required on the remote SMB server. An unauthenticated, remote attacker can exploit this to conduct man-in-the-middle attacks against the SMB server.

Solution

Enforce message signing in the host's configuration. On Windows, this is found in the policy setting 'Microsoft network server: Digitally sign communications (always)'. On Samba, the setting is called 'server signing'. See the 'see also' links for further details.

See Also

https://support.microsoft.com/en-us/help/887429/overview-of-server-message-block-signing
http://technet.microsoft.com/en-us/library/cc731957.aspx
http://www.nessus.org/u?74b80723
https://www.samba.org/samba/docs/current/man-html/smb.conf.5.html
http://www.nessus.org/u?a3cac4ea

Output

```
No output recorded.
```

Port	Hosts
445 / tcp / cifs	192.168.101.6

Plugin Details

Severity:	Medium
ID:	57608
Version:	1.18
Type:	remote
Family:	Misc.
Published:	January 19, 2012
Modified:	November 15, 2018

Risk Information

Risk Factor: Medium
CVSS v3.0 Base Score 5.3
CVSS v3.0 Vector: CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:L/A:N
CVSS v3.0 Temporal Vector: CVSS:3.0/E:U/RL:O/RC:C
CVSS v3.0 Temporal Score: 4.6
CVSS Base Score: 5.0
CVSS Temporal Score: 3.7
CVSS Vector: CVSS2#AV:N/AC:L/Au:N/C:N/I:P/A:N
CVSS Temporal Vector: CVSS2#E:U/RL:OF/RC:C

Vulnerability Information

CPE: cpe:/o:microsoft:windows cpe:/a:samba:samba
Vulnerability Pub Date: January 17, 2012

Figure 21: Vulnerability analysis in Nessus.

Here we can get the description of the vulnerability, related solution of the vulnerability. We can also get the information related to Plugin details, Risk information, output and vulnerability information. It also provides the more references for helping in resolving the vulnerability.

6. COMPARING NESSUS AND WIRESHARK

a. Scanning and Policy:

- **Nessus:**

1. User Defined Policy: Nessus has pre-built as well as custom/user-defined template feature which allows us to scan and discover live hosts and various open ports.
2. Scans IPv4/IPv6/hybrid networks and scans for system hardening and missing patches.
3. Network devices like firewalls, routers, switches, printers etc is covered also offline configuration auditing of network devices can be done.

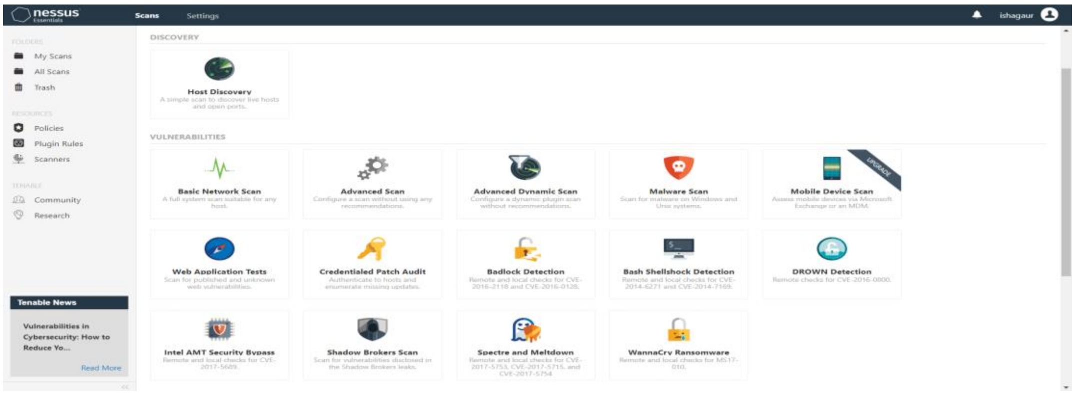
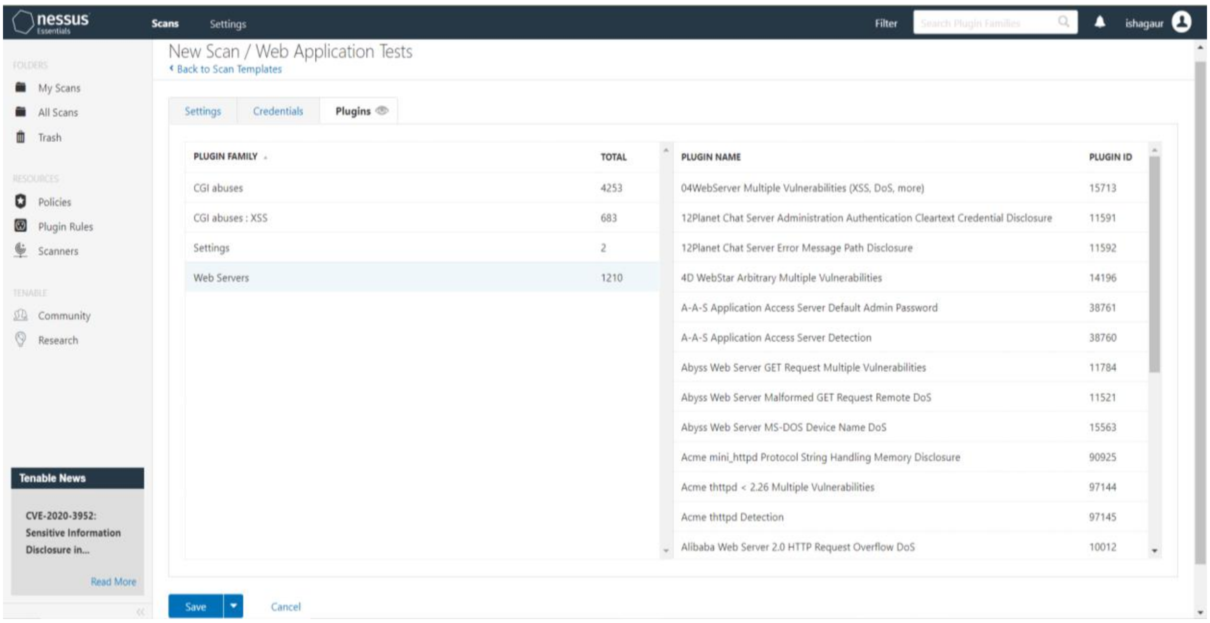

nessus Essentials
Scans
Settings
ishagaur
FOLDERS
My Scans
All Scans
Trash
RESOURCES
Policies
Plugin Rules
Scanners
TENABLE
Community
Research
Tenable News
Vulnerabilities in Cybersecurity: How to Reduce Yo...
Read More
DISCOVERY
Host Discovery
A simple scan to discover live hosts and open ports.
VULNERABILITIES
Basic Network Scan
A full system scan suitable for any host.
Advanced Scan
Configure a scan without using any recommendations.
Advanced Dynamic Scan
Configure a dynamic plugin scan without recommendations.
Malware Scan
Scan for malware on Windows and Unix systems.
Mobile Device Scan
Assess mobile devices via Microsoft Exchange or an MDM.
UPGRADE
Web Application Tests
Scan for published and unknown web vulnerabilities.
Credentialed Patch Audit
Authenticate to hosts and enumerate missing updates.
Badlock Detection
Remote and local checks for CVE-2016-2118 and CVE-2016-0128.
Bash Shellshock Detection
Remote and local checks for CVE-2014-6271 and CVE-2014-7169.
DROWN Detection
Remote checks for CVE-2016-0800.
Intel AMT Security Bypass
Remote and local checks for CVE-2017-5689.
Shadow Brokers Scan
Scan for vulnerabilities disclosed in the Shadow Brokers leaks.
Spectre and Meltdown
Remote and local checks for CVE-2017-5753, CVE-2017-5715, and CVE-2017-5754
WannaCry Ransomware
Remote and local checks for MS17-010.

Figure 22: Scanning features and Policy template of Nessus.

4.) Web Application scanning with add-ons plug-in. It checks from various web app vulnerabilities to owasp top 10. It also has Mobile Device Scans and other vulnerabilities scanner. It also has Plugins that is mapped with CVE and NVD and is updated frequently.



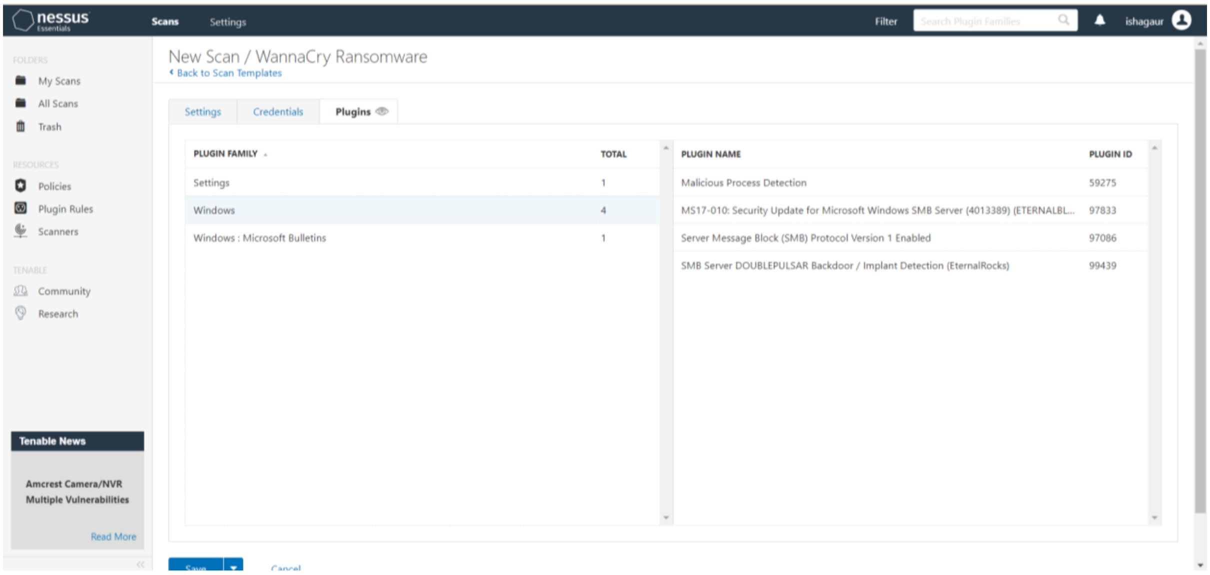
New Scan / Web Application Tests
‹ Back to Scan Templates

PLUGIN FAMILY	TOTAL
CGI abuses	4253
CGI abuses : XSS	683
Settings	2
Web Servers	1210

PLUGIN NAME	PLUGIN ID
04WebServer Multiple Vulnerabilities (XSS, DoS, more)	15713
12Planet Chat Server Administration Authentication Cleartext Credential Disclosure	11591
12Planet Chat Server Error Message Path Disclosure	11592
4D WebStar Arbitrary Multiple Vulnerabilities	14196
A-A-S Application Access Server Default Admin Password	38761
A-A-S Application Access Server Detection	38760
Abyss Web Server GET Request Multiple Vulnerabilities	11784
Abyss Web Server Malformed GET Request Remote DoS	11521
Abyss Web Server MS-DOS Device Name DoS	15563
Acme mini_httpd Protocol String Handling Memory Disclosure	90925
Acme thttpd < 2.26 Multiple Vulnerabilities	97144
Acme thttpd Detection	97145
Alibaba Web Server 2.0 HTTP Request Overflow DoS	10012

Figure 23: Web Application Scanning in Nessus

5.) Remote and Local Check for WannaCry Ransomware: it checks for the latest security builds from the Microsoft is available on the active hosts or not.



New Scan / WannaCry Ransomware
‹ Back to Scan Templates

PLUGIN FAMILY	TOTAL
Settings	1
Windows	4
Windows : Microsoft Bulletins	1

PLUGIN NAME	PLUGIN ID
Malicious Process Detection	59275
MS17-010: Security Update for Microsoft Windows SMB Server (4013389) (ETERNALBL...	97833
Server Message Block (SMB) Protocol Version 1 Enabled	97086
SMB Server DOUBLEPULSAR Backdoor / Implant Detection (EternalRocks)	99439

Figure 24: WannaCry Ransomware Check in Nessus

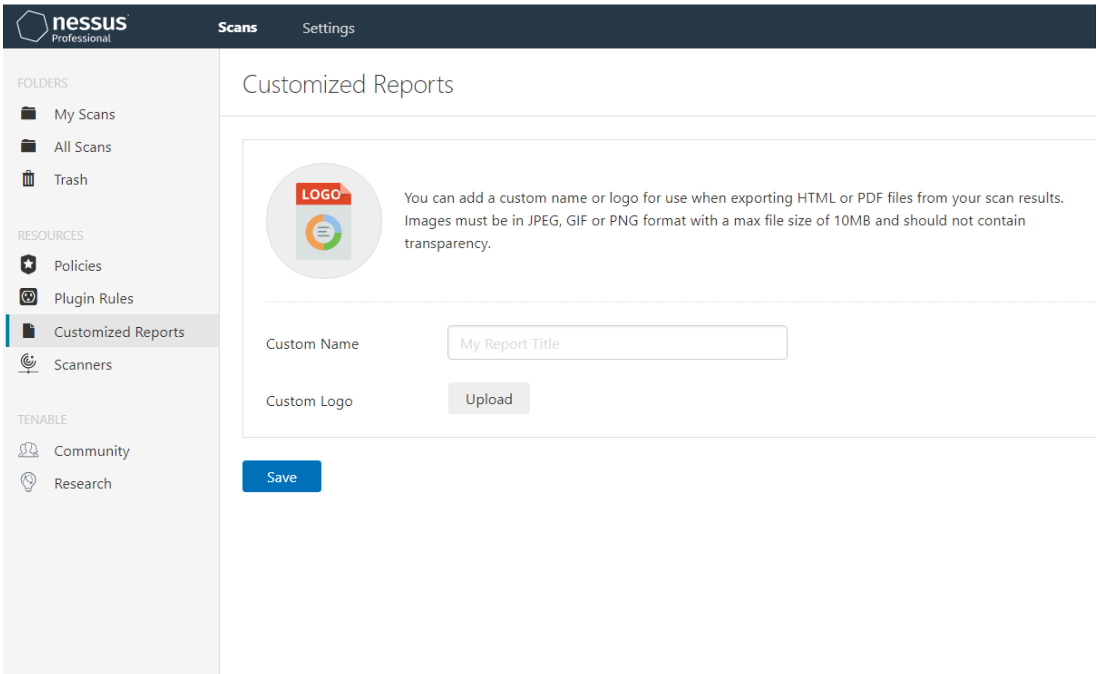
- **Wireshark:**

  The above scanning features and policies are not available in Wireshark. It just monitors the network and does deep packet inspection, nothing else.

b. Reporting and Monitoring

- **Nessus:**

  1. Customized Reporting: Customization and sorting can be done by vulnerability and host, as well as report comparison can be done.
  2. Exporting can be done in XML, PDF, HTML and CSV Formats.



Customized Reports

You can add a custom name or logo for use when exporting HTML or PDF files from your scan results. Images must be in JPEG, GIF or PNG format with a max file size of 10MB and should not contain transparency.

Custom Name: My Report Title

Custom Logo: Upload

Figure 25: Customize Reporting in Nessus

  3. Scan Results can be automatically notified via emails also report sharing can be done through Nessus Manager.

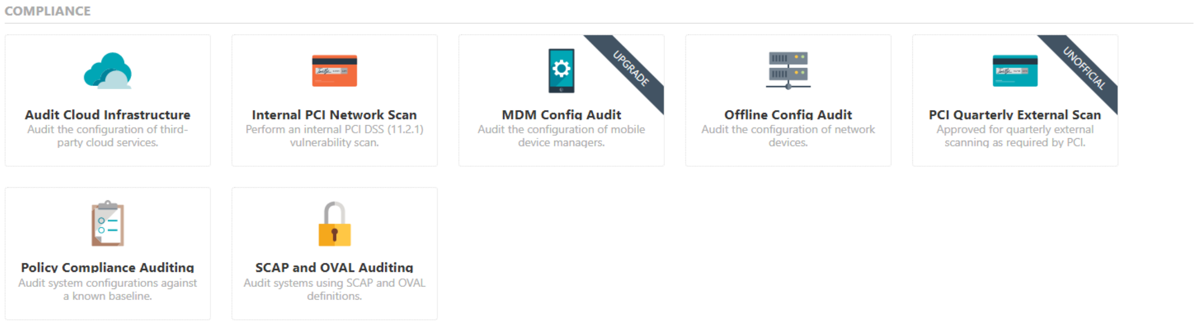
- **Wireshark:**

  Scanned files are saved in PCAP or PCAPNG in Wireshark. User interface also does not provide the full understandable information related to captured file in Wireshark as comparing to Nessus. We can export the designed files efficiently in Nessus where as files save by Wireshark can be opened in Wireshark only.

c. Compliance

- **Nessus:**

1. Nessus adhere the compliance and various policies; it scans network devices against the compliance standard. These scans results can be used by the auditors to fix the problems in larger scale.



COMPLIANCE

Audit Cloud Infrastructure
Audit the configuration of third-party cloud services.

Internal PCI Network Scan
Perform an internal PCI DSS (11.2.1) vulnerability scan.

MDM Config Audit
Audit the configuration of mobile device managers.

Offline Config Audit
Audit the configuration of network devices.

PCI Quarterly External Scan
Approved for quarterly external scanning as required by PCI.

Policy Compliance Auditing
Audit system configurations against a known baseline.

SCAP and OVAL Auditing
Audit systems using SCAP and OVAL definitions.

Figure 26: Compliance in Nessus

2. Nessus offers the ability to audit the Microsoft Azure Cloud environment to detect misconfigurations in the cloud environment and account settings. It performs an internal PCI DSS (11.2.1) vulnerability scan.
3. It audits the configuration of mobile device managers (MDM). It also audits the configuration of network devices.
4. It audits system configurations against a known baseline. It audits systems using SCAP and OVAL definitions.

- **Wireshark:**

Wireshark is not used for auditing the compliance but it can be used to make the organisation compliant to various compliance including PCI. We can put sniffer like Wireshark to secure the network k and see what sort of authentication credentials you can pick up. If these applications and servers are on the same network where cardholder data is stored, processed, or transmitted, then you have a significant problem to address.

7. CONCLUSION

Nessus and Wireshark are excellent tools that will greatly aid the ability to test and discover known security problems. The cases of packet analysis demonstrated in this paper help us to realise packet analysers and network scanning especially Wireshark and Nessus are crucial to network forensics. The need for network packet analysis is raised by the fact that the methods currently used by most users for network security are not effective enough to detect all the computer attacks, especially the latest ones. Vulnerability scanning in Wireshark and Nessus can discover a broad range of security threats and attacks against networked computer systems.

8. REFERENCES

i. Henrik A. and Martin A, “A short review of the Nessus computer network vulnerability analysing too” Available at < http://static.veracomp.pl/cdn/articles/pdf/Data%20Sheet-%20Nessus%20Professional.pdf > Accessed on 22-03-2020.

ii. Avi K., “Port and Vulnerability Scanning, Packet Sniffing, Intrusion Detection, and Penetration Testing”, Available at <https://engineering.purdue.edu/kak/compsec/NewLectures/Lecture23.pdf > Accessed on 25-03-2020.

iii. Sandeep Y., Daya P. and Srikant L., “A Comparative Analysis of Detecting Vulnerability in Network Systems”, Available at <http://ijarcsse.com/Before_August_2017/docs/papers/Volume_7/5_May2017/SV7I5-0261.pdf > Accessed on 28-03-2020.

iv. https://www.vskills.in/certification/certified-linux-administrator-nmap-snort-nessus-and-wireshark Accessed on 30-03-2020.

v. Santosh K., “Detect/Analyze Scanning Traffic Using Wireshark” Available at <https://www.koenig-solutions.com/documents/PenTestExtra-06-2013.pdf> Accessed on 02-04-2020.

vi. Zhifeng X., Vivens N and Yang X, “Network forensics analysis using Wireshark” Available at <https://www.researchgate.net/publication/281573989_Network_forensics_analysis_using_Wireshark > Accessed on 04-04-2020.

vii. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Connection Accessed on 04-04-2020.

viii. https://www.w3.org/TR/upgrade-insecure-requests/ Accessed on 05-04-2020.

ix. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Upgrade-Insecure-Requests Accessed on 05-04-2020.

x. https://www.uniassignment.com/essay-samples/information-technology/what-are-the-main-features-of-nessus-information-technology-essay.php Accessed on 06-04-2020.

xi. https://static.veracomp.pl/cdn/articles/pdf/Data%20Sheet-%20Nessus%20Professional.pdf Accessed on 07-04-2020.

xii. https://docs.tenable.com/nessus/Content/ScanAndPolicyTemplates.htm Accessed on 09-04-2020.