TCET

DEPARTMENT OF COMPUTER ENGINEERING (COMP)

(Accredited by NBA for 3 years, 3rd Cycle Accreditation w.e.f. 1st July 2019)

Choice Based Credit Grading Scheme (CBCGS)

Under TCET Autonomy

EXPERIMENT 1.A

Aim: Understand the use of network reconnaissance tools like WHOIS, dig, traceroute, ns lookup to gather information about network and domain registrars.

Theory:

1. **WHOIS:**
   Whois is a widely used Internet record listing that identifies who owns a domain and how to get in contact with them. The Internet Corporation for Assigned Names and Numbers (ICANN) regulates domain name registration and ownership. Whois records have proven to be extremely useful and have developed into an essential resource for maintaining the integrity of the domain name registration and website ownership process.
2. **Dig:**
   Dig is a network administration command-line tool for querying the Domain Name System (DNS).dig is useful for network troubleshooting and for educational purposes. It can operate based on command-line options and flag arguments or in batch mode by reading requests from an operating system file. When a specific name server is notspecified in the command invocation, it uses the operating system's default resolver, usually configured in the file resolv.conf. Without any arguments, it queries the DNS rootzone.
3. **Traceroute:**
   The traceroute command is used to determine the path between two connections. Often a connection to another device will have to go through multiple routers. The traceroute command will return the names or IP addresses of all the routers between two devices. This also allows you to see where a packet may be misguided.
4. **Nslookup:**
   Nslookup (stands for “Name Server Lookup”) is a useful command for getting information from the DNS server. It is a network administration tool for querying the Domain Name System (DNS) to obtain domain name or IP address mapping or any other specific DNS record. It is also used to troubleshoot DNS-related problems.

Conclusion:

Thus, we have successfully seen how this network commands function and their information with respect to their working and peculiarities.

For Faculty use

Correction Parameters	**Formative Assessment [40%]**	**Timely completion of Practical [40%]**	**Attendance / Learning Attitude [20%]**	
Marks Obtained				

CSS EXPERIMENT 1.B

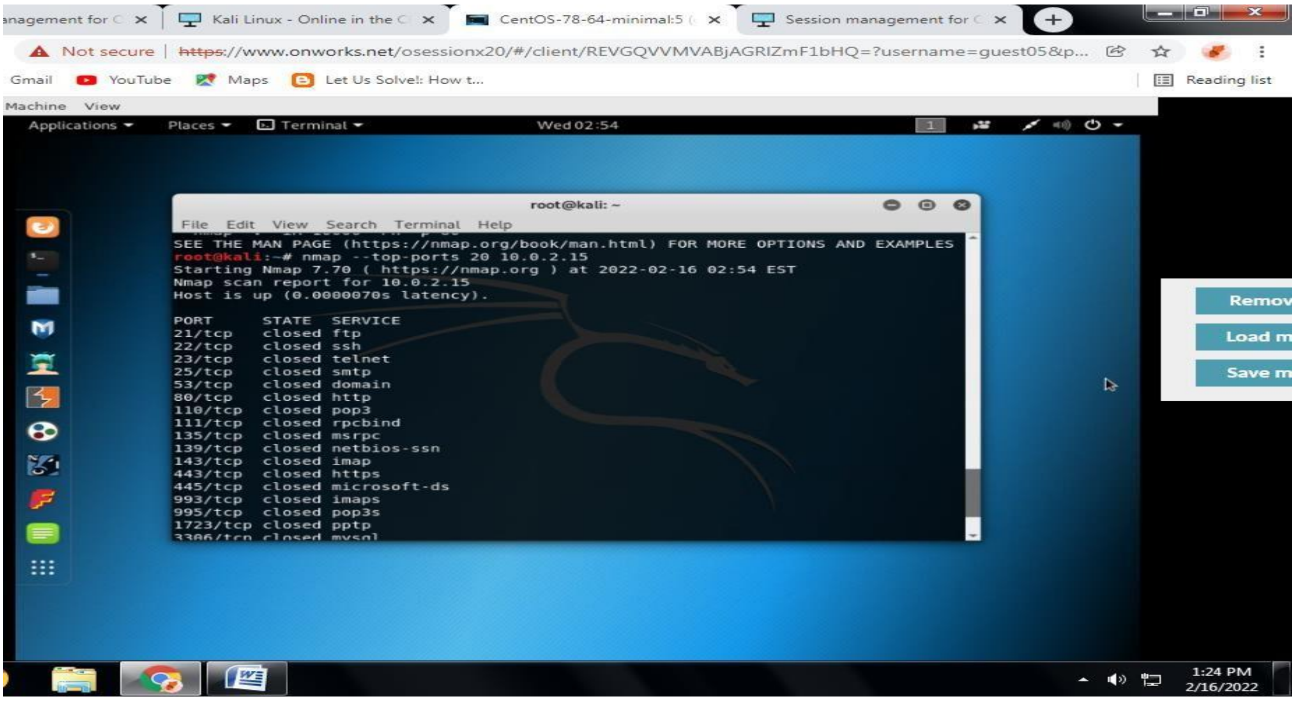
Aim: To Analyze the tool nmap and use it with different options to scan open ports, perform OS fingerprinting, do a ping scan, TCP port scan, UDP port scan, Xmas scan, etc.

Nmap

- Nmap is a network scanner
- Nmap is used to discover hosts and services on a computer network by sending packets and analyzing the responses.
- nmap provides several features for probing computer networks, including host discovery and service and operating system detection.

1. Port Scanning

It automatically scans a number of the most 'popular' ports for a host. You can run this command using:



```
SEE THE MAN PAGE (https://nmap.org/book/man.html) FOR MORE OPTIONS AND EXAMPLES
root@kali:~# nmap --top-ports 20 10.0.2.15
Starting Nmap 7.70 ( https://nmap.org ) at 2022-02-16 02:54 EST
Nmap scan report for 10.0.2.15
Host is up (0.0000070s latency).

PORT     STATE  SERVICE
21/tcp   closed ftp
22/tcp   closed ssh
23/tcp   closed telnet
25/tcp   closed smtp
53/tcp   closed domain
80/tcp   closed http
110/tcp  closed pop3
111/tcp  closed rpcbind
135/tcp  closed msrpc
139/tcp  closed netbios-ssn
143/tcp  closed imap
443/tcp  closed https
445/tcp  closed microsoft-ds
993/tcp  closed imaps
995/tcp  closed pop3s
1723/tcp closed pptp
3306/tcp closed mysql
```

nmap –top-ports <ip>

nmap --top-ports 20 <ip>

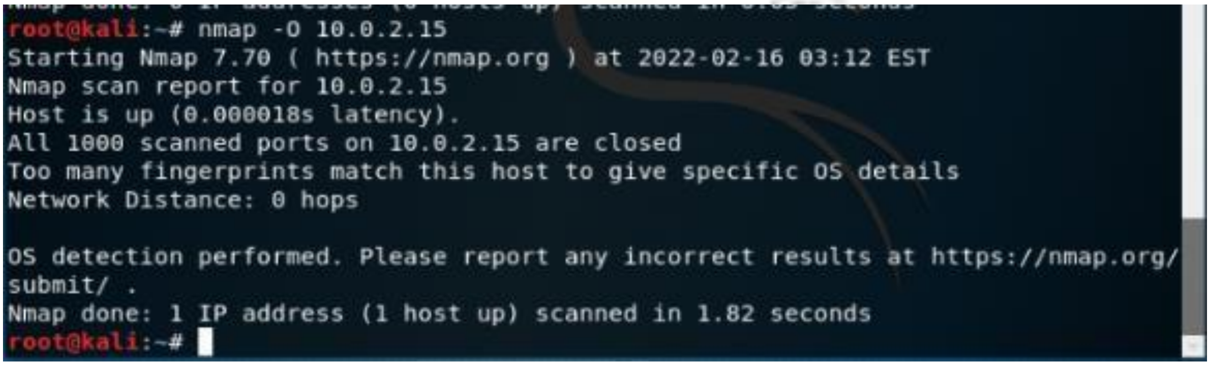
Replace the “20” with the number of ports to scan, and Nmap quickly scans that many ports. It returns a concise output that details the status of the most common ports, and this lets you quickly see whether you have any unnecessarily open ports.

2. OS fingerprinting

OS scanning is one of the most powerful features of Nmap. When using this type of scan, Nmap sends TCP and UDP packets to a particular port and then analyzes its response. It compares this response to a database of 2600 operating systems, and return information on the OS (and version) of a host.

To run an OS scan, use the following command:

# nmap -O <target IP>

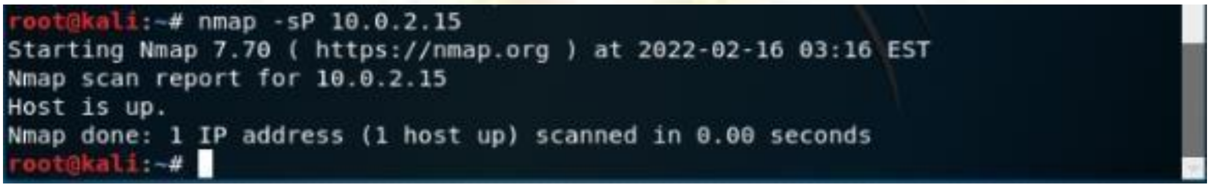


```
root@kali:~# nmap -O 10.0.2.15
Starting Nmap 7.70 ( https://nmap.org ) at 2022-02-16 03:12 EST
Nmap scan report for 10.0.2.15
Host is up (0.000018s latency).
All 1000 scanned ports on 10.0.2.15 are closed
Too many fingerprints match this host to give specific OS details
Network Distance: 0 hops

OS detection performed. Please report any incorrect results at https://nmap.org/
submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.82 seconds
root@kali:~# 
```

3. ping scan

As mentioned above, a ping scan returns information on every active IP on your network. You can execute a ping scan using this command:

nmap -sP 10.0.2.15



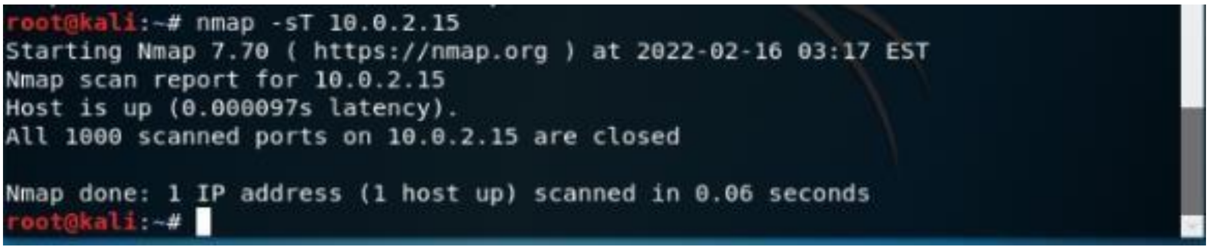
```
root@kali:~# nmap -sP 10.0.2.15
Starting Nmap 7.70 ( https://nmap.org ) at 2022-02-16 03:16 EST
Nmap scan report for 10.0.2.15
Host is up.
Nmap done: 1 IP address (1 host up) scanned in 0.00 seconds
root@kali:~# 
```

4. TCP port scan

One of the best things is the Nmap command to check open ports, and the second-best thing about Nmap is its power to work with TCP and UDP without any hiccups. Several services are limited to just TCP, but people understand the advantage of scanning UDP-based services. Here are examples of both these services that are allowed by Nmap.

Tcp port scan

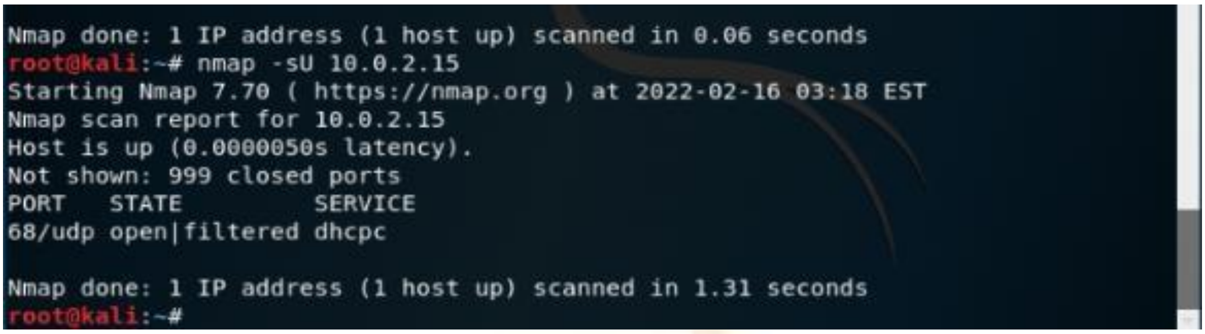
nmap -sT 10.0.2.15



```
root@kali:~# nmap -sT 10.0.2.15
Starting Nmap 7.70 ( https://nmap.org ) at 2022-02-16 03:17 EST
Nmap scan report for 10.0.2.15
Host is up (0.000097s latency).
All 1000 scanned ports on 10.0.2.15 are closed

Nmap done: 1 IP address (1 host up) scanned in 0.06 seconds
root@kali:~#
```

5. UDP port scan

nmap -sU 10.0.2.15

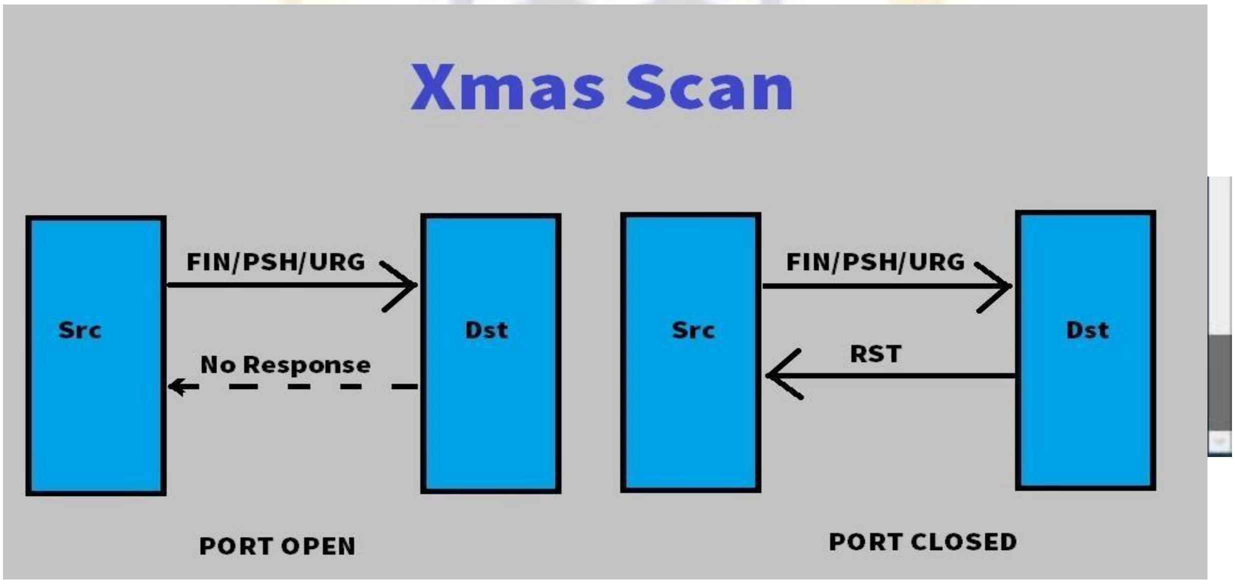


```
Nmap done: 1 IP address (1 host up) scanned in 0.06 seconds
root@kali:~# nmap -sU 10.0.2.15
Starting Nmap 7.70 ( https://nmap.org ) at 2022-02-16 03:18 EST
Nmap scan report for 10.0.2.15
Host is up (0.0000050s latency).
Not shown: 999 closed ports
PORT     STATE         SERVICE
68/udp open|filtered dhcpc

Nmap done: 1 IP address (1 host up) scanned in 1.31 seconds
root@kali:~#
```

6. Xmas scan

It Sets the FIN, PSH, and URG flags, lighting the packet up like a Christmas tree.

nmap -sX 10.0.2.15



As we can see in the above picture when we send a packet with FIN/PSH/URG flag to set and send it to the destination if we don’t get any response from Destination we will know that Port is OPEN. if we get RST in return then we know that Port is Closed.

Conclusion:

Successfully studied and implemented various nmap tool commands in kalli Linux. Also, we have seen how these tools can give us some vital security information about wired/wireless network connections.

For Faculty use

Correction Parameters	**Formative Assessment [40%]**	**Timely completion of Practical [40%]**	**Attendance / Learning Attitude [20%]**	
Marks Obtained				

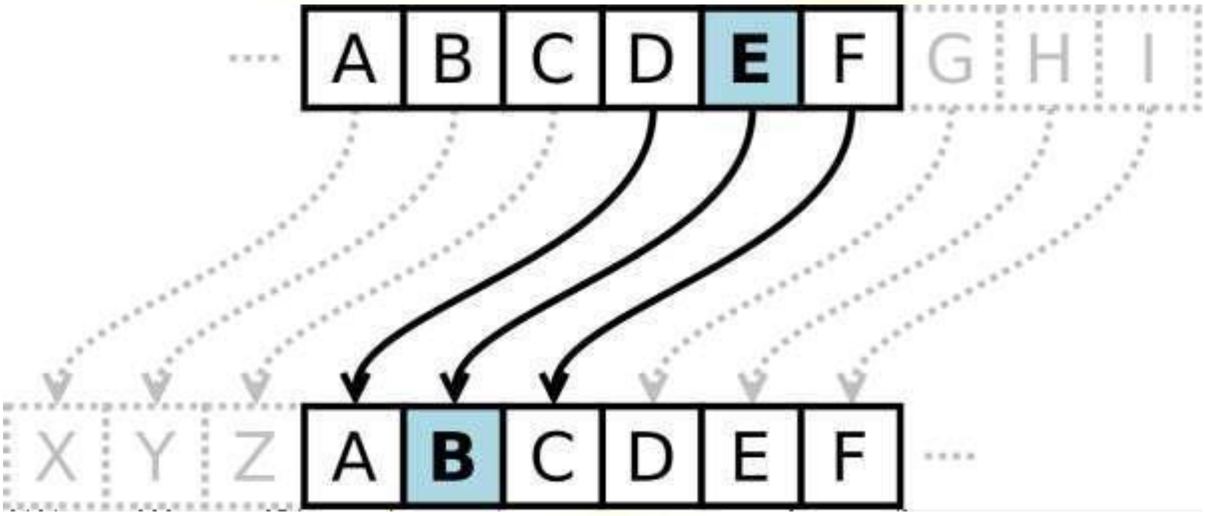
EXPERIMENT 2

Aim: Implement and design the product cipher using Substitution and Transposition ciphers.

Theory:

A Caesar cipher is a simple method of encoding messages. Caesar ciphers use a substitution method where letters in the alphabet are shifted by some fixed number of spaces to yield an encoding alphabet. A Caesar cipher with a shift of 1 would encode an A as a B, an M as an N, and a Z as an A, and so on.

It is a type of substitution cipher in which each letter in the plaintext is replaced by a letter some fixed number of positions down the alphabet. For example, with a left shift of 3, D would be replaced by A, E would become B, and so on.



To encrypt a message, enter the message in the Plaintext textbox, specify the shift, and click Encrypt. To decrypt a message, enter the message in the Ciphertext textbox, specify the shift, and click Decrypt.

Transposition cipher:

In cryptography, a transposition cipher is a method of encryption by which the positions held by units of plaintext (which are commonly characters or groups of characters) are shifted according to a regular system, so that the ciphertext constitutes a permutation of the plaintext.

Conclusion:

Successfully implemented and designed the product cipher using Substitution and Transposition ciphers.

For Faculty Use

Correction Parameters	**Formative Assessment[40%]**	**Timely completion of Practical [40%]**	**Attendance / Learning Attitude [20%]**	
Marks Obtained				

EXPERIMENT 3

Aim: To Implement RSA Algorithm

Theory: RSA algorithm is asymmetric cryptography algorithm. Asymmetric actually means that it works on two different keys i.e., **Public Key** and **Private Key.** As the name describes that the Public Key is given to everyone and Private key is kept private.

An example of asymmetric cryptography:

1. A client (for example browser) sends its public key to the server and requests for some data.
2. The server encrypts the data using client's public key and sends the encrypted data.
3. Client receives this data and decrypts it.

Since this is asymmetric, nobody else except browser can decrypt the data even if a third partyhas public key of browser.

The idea of RSA is based on the fact that it is difficult to factorize a large integer. The public key consists of two numbers where one number is multiplication of two large prime numbers. And private key is also derived from the same two prime numbers. So, if somebody can factorize the large number, the private key is compromised. Therefore, encryption strength totally lies on the key size and if we double or triple the key size, the strength of encryption increases exponentially. RSA keys can be typically 1024 or 2048 bits long, but experts believe that 1024-bit keys could be broken in the near future. But till now it seems to be an infeasible task.

Conclusion:
It is concluded that, while establishing RSA key pairs, usage keys and general-purposekeys are integrated. In usage RSA keys, two key pairs are used for encryption and signatures. In General-purpose key, one single pair is used for both encryption and signature.

For Faculty Use

Correction Parameters	**Formative Assessment [40%]**	**Timely completion of Practical [40%]**	**Attendance / Learning Attitude [20%]**	
Marks Obtained				

EXPERIMENT 4

Aim: To Implement Diffie-Hellman Algorithm

Theory:

Diffie Hellman (DH) key exchange algorithm is a method for securely exchanging cryptographic keys overa public communications channel. Keys are not actually exchanged – they are jointly derived. It is named after their inventors Whitfield Diffie and Martin Hellman.

If Alice and Bob wish to communicate with each other, they first agree between them a large prime numberp, and a generator (or base) g (where 0 < g < p).

Alice chooses a secret integer a (her private key) and then calculates g^a mod p (which is her public key).Bob chooses his private key b, and calculates his public key in the same way.

Bob knows b and g^a, so he can calculate (g^a)^b mod p = g^ab mod p. Therefore both Alice and Bob know a shared secret g^ab mod p. An eavesdropper Eve who was listening in on the communication knows p, g, Alice’s public key (g^a mod p) and Bob’s public key (g^b mod p). She is unable to calculate the shared secret from these values.

In static-static mode, both Alice and Bob retain their private/public keys over multiple communications. Therefore the resulting shared secret will be the same every time. In ephemeral-static mode one party will generate a new private/public key every time, thus a new shared secret will be generated.

Advantages:

- The sender and receiver don’t need any prior knowledge of each other.
- Once the keys are exchanged, the communication of data can be done through aninsecure channel.
- The sharing of the secret key is safe.

Disadvantages:

- The algorithm cannot be sued for any asymmetric key exchange.
- Similarly, it cannot be used for signing digital signatures.
- Since it doesn’t authenticate any party in the transmission, the Diffie Hellman key exchange is susceptible to a man-in-the-middle attack.

Conclusion:

The Diffie Hellman key Exchange has proved to be a useful key exchange system due to its advantages. While it is really tough for someone snooping the network to decrypt the data and get the keys, it is still possible if the numbers generated are not entirely random.

For Faculty use

Correction Parameters	**Formative Assessment [40%]**	**Timely completion of Practical [40%]**	**Attendance / Learning Attitude [20%]**	
Marks Obtained				

EXPERIMENT 5

Aim: Study of packet sniffer tools: Wireshark:

1. Download and install Wireshark and capture icmp, tcp, and http packets in promiscuous mode.
2. Explore how the packets can be traced based on different filters

Theory:

Wireshark, a network analysis tool formerly known as Ethereal, captures packets in real time and display them in human-readable format. Wireshark includes filters, color-coding and other featuresthat let you dig deep into network traffic and inspect individual packets.

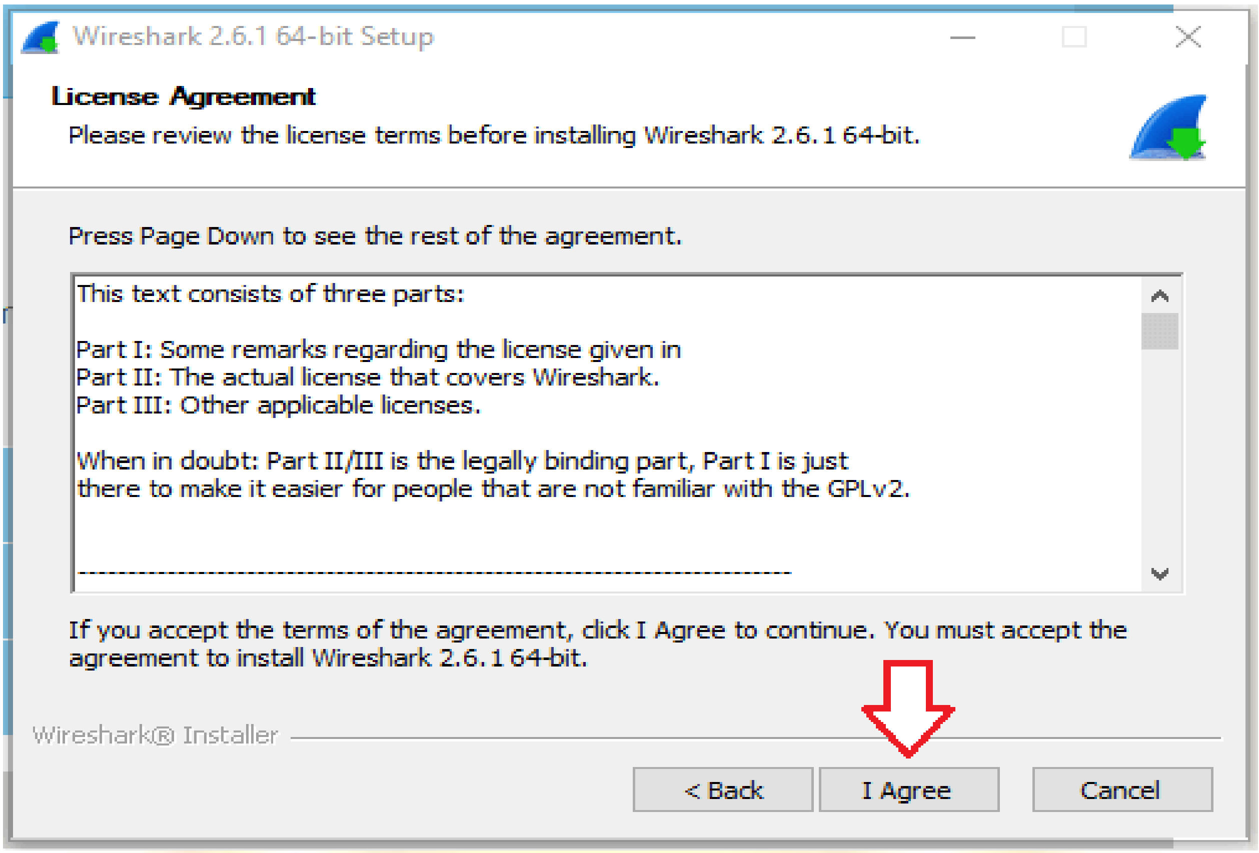
Features of Wireshark:

- Available for UNIX and Windows.
- Capture live packet data from a network interface.
- Open files containing packet data captured with tcpdump/WinDump, Wireshark, and a
- number of other packet capture programs.
- Import packets from text files containing hex dumps of packet data.
- Display packets with very detailed protocol information.
- Export some or all packets in a number of capture file formats.
- Filter packets on many criteria.
- Search for packets on many criteria.
- Colorize packet display based on filters.
- Create various statistics.

Capturing Packets

After downloading and installing wireshark, you can launch it and click the name of an interface under Interface List to start capturing packets on that interface. For example, if you want to capture traffic on the wireless network, click your wireless interface. You can configure advanced features by clicking Capture Options.

Installation of Wireshark:



Wireshark 2.6.1 64-bit Setup

License Agreement

Please review the license terms before installing Wireshark 2.6.1 64-bit.

Press Page Down to see the rest of the agreement.

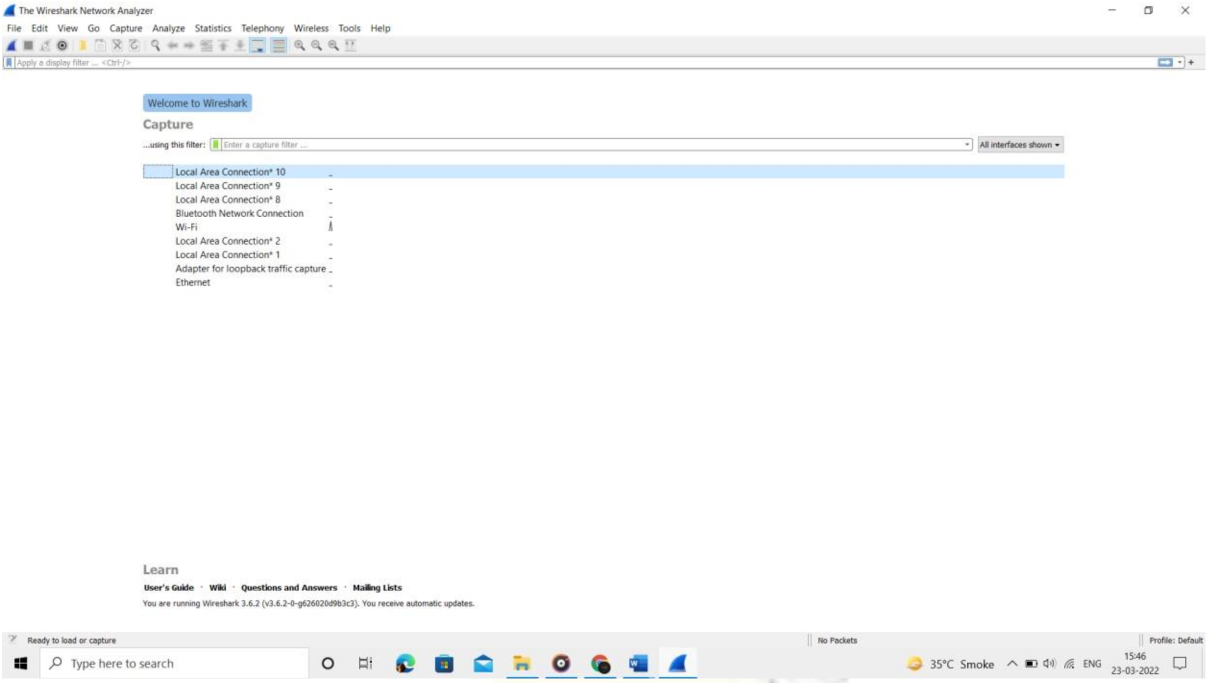
This text consists of three parts:

Part I: Some remarks regarding the license given in
Part II: The actual license that covers Wireshark.
Part III: Other applicable licenses.

When in doubt: Part II/III is the legally binding part, Part I is just
there to make it easier for people that are not familiar with the GPLv2.

If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Wireshark 2.6.1 64-bit.

After downloading and installing wireshark, you can launch it and click the name of an interface under Interface List to start capturing packets on that interface. For example, if you want to capture traffic on the wireless network, click your wireless interface. You can configure advanced features by clicking Capture Options.



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

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

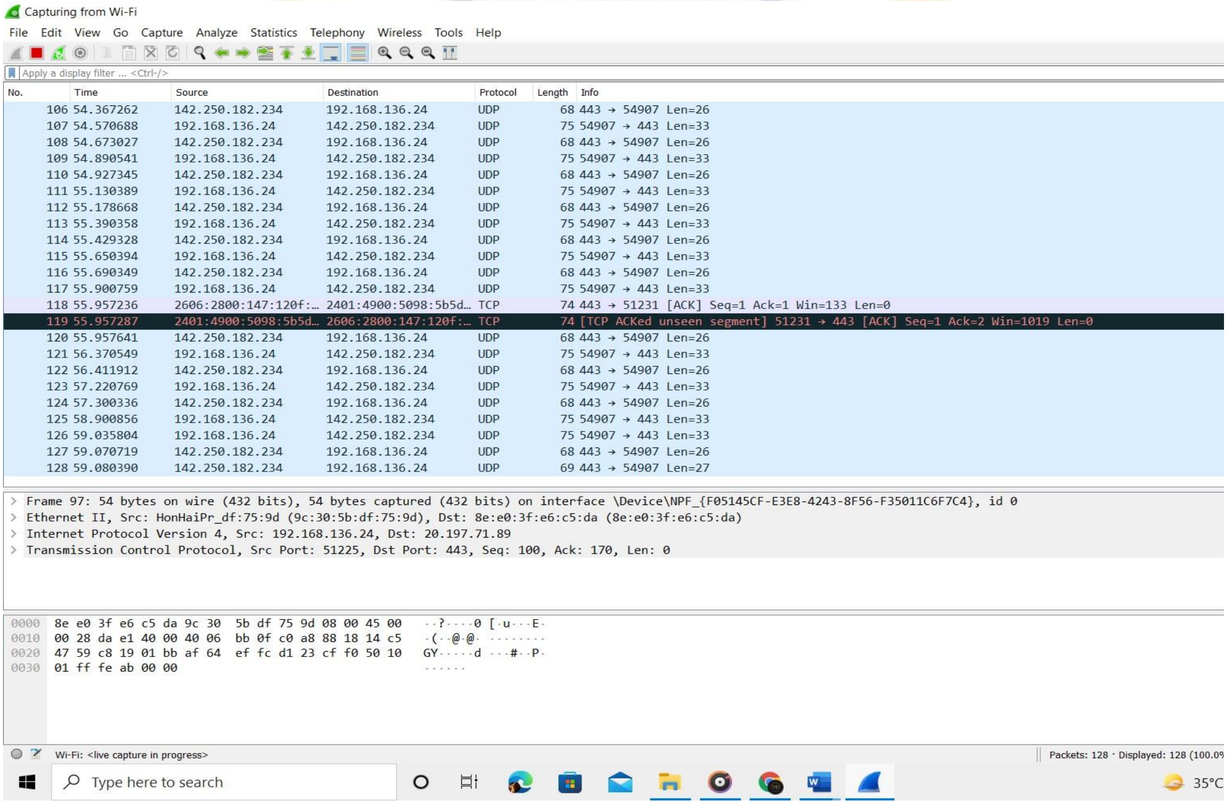
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists

You are running Wireshark 3.6.2 (v3.6.2-0-g626020d9b3c3). You receive automatic updates.

Ready to load or capture
No Packets
Profile: Default
Type here to search
35°C Smoke
ENG
15:46
23-03-2022

As soon as you click the interface's name, you'll see the packets start to appear in real time.

Wireshark captures each packet sent to or from your system. If you're capturing on a wireless interface and have promiscuous mode enabled in your capture options, you'll also see other the other packets on the network



Capturing from Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

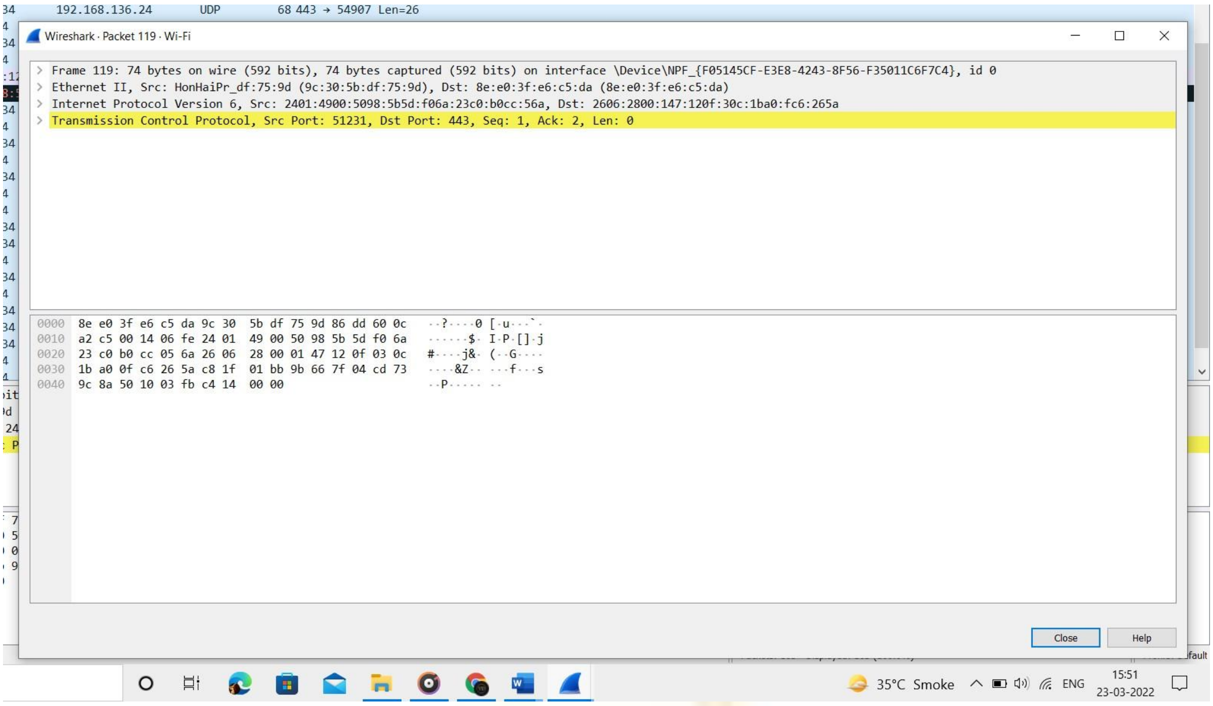
```
No.    Time          Source                Destination           Protocol  Length  Info
   106 54.367262     142.250.182.234       192.168.136.24        UDP          68 443 → 54907 Len=26
   107 54.570688     192.168.136.24        142.250.182.234       UDP          75 54907 → 443 Len=33
   108 54.673027     142.250.182.234       192.168.136.24        UDP          68 443 → 54907 Len=26
   109 54.890541     192.168.136.24        142.250.182.234       UDP          75 54907 → 443 Len=33
   110 54.927345     142.250.182.234       192.168.136.24        UDP          68 443 → 54907 Len=26
   111 55.130389     192.168.136.24        142.250.182.234       UDP          75 54907 → 443 Len=33
   112 55.178668     142.250.182.234       192.168.136.24        UDP          68 443 → 54907 Len=26
   113 55.390358     192.168.136.24        142.250.182.234       UDP          75 54907 → 443 Len=33
   114 55.429328     142.250.182.234       192.168.136.24        UDP          68 443 → 54907 Len=26
   115 55.650394     192.168.136.24        142.250.182.234       UDP          75 54907 → 443 Len=33
   116 55.690349     142.250.182.234       192.168.136.24        UDP          68 443 → 54907 Len=26
   117 55.900759     192.168.136.24        142.250.182.234       UDP          75 54907 → 443 Len=33
   118 55.957236     2606:2800:147:120f:…  2401:4900:5098:5b5d…  TCP          74 443 → 51231 [ACK] Seq=1 Ack=1 Win=133 Len=0
   119 55.957287     2401:4900:5098:5b5d…  2606:2800:147:120f:…  TCP          74 [TCP ACKed unseen segment] 51231 → 443 [ACK] Seq=1 Ack=2 Win=1019 Len=0
   120 55.957641     142.250.182.234       192.168.136.24        UDP          68 443 → 54907 Len=26
   121 56.370549     192.168.136.24        142.250.182.234       UDP          75 54907 → 443 Len=33
   122 56.411912     142.250.182.234       192.168.136.24        UDP          68 443 → 54907 Len=26
   123 57.220769     192.168.136.24        142.250.182.234       UDP          75 54907 → 443 Len=33
   124 57.300336     142.250.182.234       192.168.136.24        UDP          68 443 → 54907 Len=26
   125 58.900856     192.168.136.24        142.250.182.234       UDP          75 54907 → 443 Len=33
   126 59.035804     192.168.136.24        142.250.182.234       UDP          75 54907 → 443 Len=33
   127 59.070719     142.250.182.234       192.168.136.24        UDP          68 443 → 54907 Len=26
   128 59.080390     142.250.182.234       192.168.136.24        UDP          69 443 → 54907 Len=27
```

```
> Frame 97: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface \Device\NPF_{F05145CF-E3E8-4243-8F56-F35011C6F7C4}, id 0
> Ethernet II, Src: HonHaiPr_df:75:9d (9c:30:5b:df:75:9d), Dst: 8e:e0:3f:e6:c5:da (8e:e0:3f:e6:c5:da)
> Internet Protocol Version 4, Src: 192.168.136.24, Dst: 20.197.71.89
> Transmission Control Protocol, Src Port: 51225, Dst Port: 443, Seq: 100, Ack: 170, Len: 0
```

```
0000  8e e0 3f e6 c5 da 9c 30  5b df 75 9d 08 00 45 00   ··?····0 [·u···E·
0010  00 28 da e1 40 00 40 06  bb 0f c0 a8 88 18 14 c5   ·(··@·@· ········
0020  47 59 c8 19 01 bb af 64  ef fc d1 23 cf f0 50 10   GY·····d ···#··P·
0030  01 ff fe ab 00 00                                  ······
```

Wi-Fi: <live capture in progress>
Packets: 128 · Displayed: 128 (100.0%
Type here to search
35°C

Capturing TCP

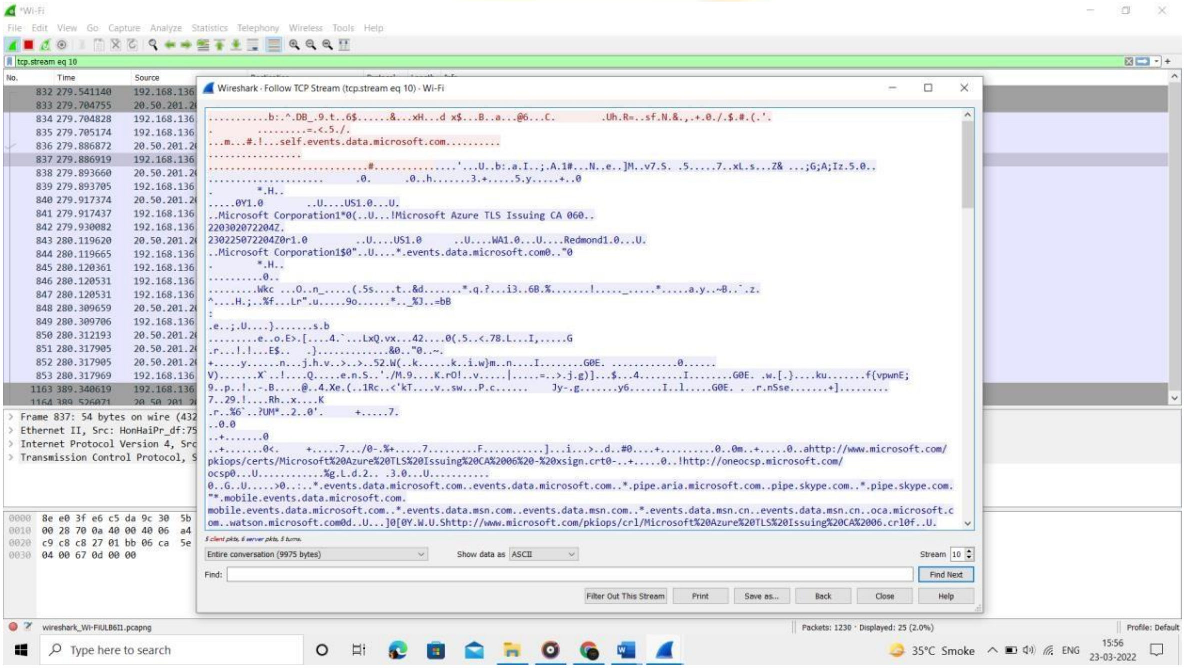


Wireshark · Packet 119 · Wi-Fi

```
> Frame 119: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{F05145CF-E3E8-4243-8F56-F35011C6F7C4}, id 0
> Ethernet II, Src: HonHaiPr_df:75:9d (9c:30:5b:df:75:9d), Dst: 8e:e0:3f:e6:c5:da (8e:e0:3f:e6:c5:da)
> Internet Protocol Version 6, Src: 2401:4900:5098:5b5d:f06a:23c0:b0cc:56a, Dst: 2606:2800:147:120f:30c:1ba0:fc6:265a
> Transmission Control Protocol, Src Port: 51231, Dst Port: 443, Seq: 1, Ack: 2, Len: 0
```

```
0000  8e e0 3f e6 c5 da 9c 30  5b df 75 9d 86 dd 60 0c   ··?····0 [·u···`·
0010  a2 c5 00 14 06 fe 24 01  49 00 50 98 5b 5d f0 6a   ······$· I·P·[]·j
0020  23 c0 b0 cc 05 6a 26 06  28 00 01 47 12 0f 03 0c   #····j&· (··G····
0030  1b a0 0f c6 26 5a c8 1f  01 bb 9b 66 7f 04 cd 73   ····&Z·· ···f···s
0040  9c 8a 50 10 03 fb c4 14  00 00                     ··P····· ··
```

Close
Help
35°C Smoke
ENG
15:51
23-03-2022



*Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp.stream eq 10

```
No.   Time        Source
 832 279.541140  192.168.136
 833 279.704755  20.50.201.2
 834 279.704828  192.168.136
 835 279.705174  192.168.136
 836 279.886872  20.50.201.2
 837 279.886919  192.168.136
 838 279.893660  20.50.201.2
 839 279.893705  192.168.136
 840 279.917374  20.50.201.2
 841 279.917437  192.168.136
 842 279.930082  192.168.136
 843 280.119620  20.50.201.2
 844 280.119665  192.168.136
 845 280.120361  192.168.136
 846 280.120531  192.168.136
 847 280.120531  192.168.136
 848 280.309659  20.50.201.2
 849 280.309706  192.168.136
 850 280.312193  20.50.201.2
 851 280.317905  20.50.201.2
 852 280.317905  20.50.201.2
 853 280.317969  192.168.136
1163 389.340619  192.168.136
```

```
> Frame 837: 54 bytes on wire (432
> Ethernet II, Src: HonHaiPr_df:75
> Internet Protocol Version 4, Src
> Transmission Control Protocol, S
```

```
0000  8e e0 3f e6 c5 da 9c 30  5b
0010  00 28 70 0a 40 00 40 06  a4
0020  c9 c8 c8 27 01 bb 06 ca  5e
0030  04 00 67 0d 00 00
```

Wireshark · Follow TCP Stream (tcp.stream eq 10) · Wi-Fi

```
...........b:.^.DB_.9.t..6$......&...xH...d x$...B..a...@6...C.        .Uh.R=..sf.N.&.,.+.0./.$.#.(.'.
.        .........=.<.5./.
...m...#.!...self.events.data.microsoft.com..........
.................
............................#.................'...U..b:.a.I..;.A.1#...N..e..]M..v7.S. .5.....7..xL.s...Z& ...;G;A;Iz.5.0..
......................      .0.      .0..h.......3.+.....5.y.....+..0
.        *.H..
.....0Y1.0        ..U....US1.0...U.
..Microsoft Corporation1*0(..U...!Microsoft Azure TLS Issuing CA 060..
220302072204Z.
230225072204Z0r1.0         ..U....US1.0      ..U....WA1.0...U....Redmond1.0...U.
..Microsoft Corporation1$0"..U....*.events.data.microsoft.com0.."0
.        *.H..
..........0..
.........Wkc ...O..n_.....(.5s....t..&d.......*.q.?...i3..6B.%.......!....._.....*.....a.y..~B..`.z.
^....H.;..%f...Lr".u.....9o......*.._%J..=bB
:
.e..;.U....}.......s.b
.........e..o.E>.[....4.`...LxQ.vx...42....0(.5..<.78.L...I,.....G
.r...!.!...E$..   .}.............&0.."0..~.
+.....y......n...j.h.v..>..>..52.W(..k......k..i.w)m..n....I........G0E. ............0......
V).......X`..!....Q.....e.n.S..'./M.9....K.rO!..v.....|.....=..>.j.g)]...$...4........I........G0E. .w.[.}....ku.......f{vpwnE;
9..p..!..-.B.....@..4.Xe.(..1Rc..<'kT....v..sw...P.c......    Jy-.g.......y6......I..l.....G0E. . .r.nSse.......+].........
7..29.!....Rh..x....K
.r..%6`..?UM*..2..0'.     +.....7.
..0.0
..+.......0
..+.......0<.    +.....7.../0-.%+.....7.........F...........]...i...>..d..#0....+..........0..0m..+.....0..ahttp://www.microsoft.com/
pkiops/certs/Microsoft%20Azure%20TLS%20Issuing%20CA%2006%20-%20xsign.crt0-..+.....0..!http://oneocsp.microsoft.com/
ocsp0...U...........%g.L.d.2.. .3.0...U..........
0..G..U.....>0..:..*.events.data.microsoft.com..events.data.microsoft.com..*.pipe.aria.microsoft.com..pipe.skype.com..*.pipe.skype.com.
"*.mobile.events.data.microsoft.com.
mobile.events.data.microsoft.com..*.events.data.msn.com..events.data.msn.com..*.events.data.msn.cn..events.data.msn.cn..oca.microsoft.c
om..watson.microsoft.com0d..U...]0[0Y.W.U.Shttp://www.microsoft.com/pkiops/crl/Microsoft%20Azure%20TLS%20Issuing%20CA%2006.crl0f..U.
```

5 client pkts, 6 server pkts, 5 turns.

Entire conversation (9975 bytes)

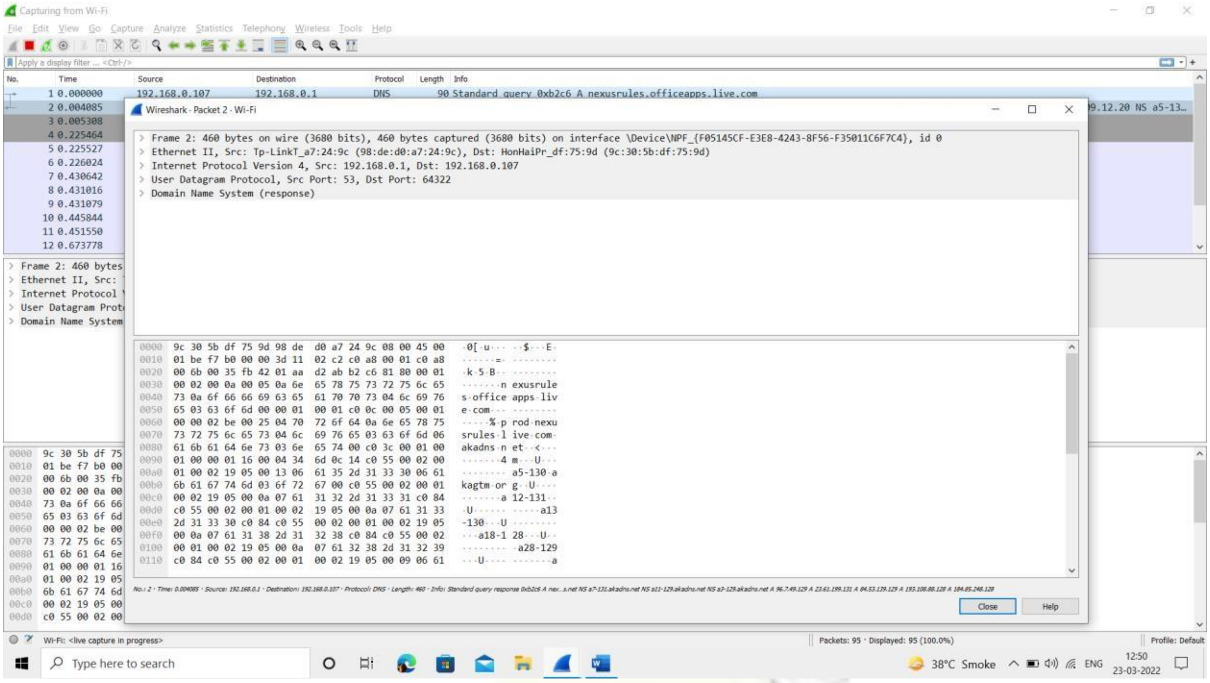
Show data as ASCII

Stream 10

Find:

Packets: 1230 · Displayed: 25 (2.0%)

wireshark_Wi-FiULB6I1.pcapng



Capturing from Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Wireshark · Packet 2 · Wi-Fi
Close
Help
Type here to search
38°C Smoke
ENG
12:50
23-03-2022

```
No.  Time        Source          Destination   Protocol  Length  Info
   1 0.000000    192.168.0.107   192.168.0.1   DNS           90  Standard query 0xb2c6 A nexusrules.officeapps.live.com
   2 0.004085
   3 0.005308
   4 0.225464
   5 0.225527
   6 0.226024
   7 0.430642
   8 0.431016
   9 0.431079
  10 0.445844
  11 0.451550
  12 0.673778
```

```
> Frame 2: 460 bytes on wire (3680 bits), 460 bytes captured (3680 bits) on interface \Device\NPF_{F05145CF-E3E8-4243-8F56-F35011C6F7C4}, id 0
> Ethernet II, Src: Tp-LinkT_a7:24:9c (98:de:d0:a7:24:9c), Dst: HonHaiPr_df:75:9d (9c:30:5b:df:75:9d)
> Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.107
> User Datagram Protocol, Src Port: 53, Dst Port: 64322
> Domain Name System (response)
```

```
0000  9c 30 5b df 75 9d 98 de  d0 a7 24 9c 08 00 45 00   ·0[·u··· ··$···E·
0010  01 be f7 b0 00 00 3d 11  02 c2 c0 a8 00 01 c0 a8   ······=· ········
0020  00 6b 00 35 fb 42 01 aa  d2 ab b2 c6 81 80 00 01   ·k·5·B·· ········
0030  00 02 00 0a 00 05 0a 6e  65 78 75 73 72 75 6c 65   ·······n exusrule
0040  73 0a 6f 66 66 69 63 65  61 70 70 73 04 6c 69 76   s·office apps·liv
0050  65 03 63 6f 6d 00 00 01  00 01 c0 0c 00 05 00 01   e·com··· ········
0060  00 00 02 be 00 25 04 70  72 6f 64 0a 6e 65 78 75   ·····%·p rod·nexu
0070  73 72 75 6c 65 73 04 6c  69 76 65 03 63 6f 6d 06   srules·l ive·com·
0080  61 6b 61 64 6e 73 03 6e  65 74 00 c0 3c 00 01 00   akadns·n et··<···
0090  01 00 00 01 16 00 04 34  6d 0c 14 c0 55 00 02 00   ·······4 m···U···
00a0  01 00 02 19 05 00 13 06  61 35 2d 31 33 30 06 61   ········ a5-130·a
00b0  6b 61 67 74 6d 03 6f 72  67 00 c0 55 00 02 00 01   kagtm·or g··U····
00c0  00 02 19 05 00 0a 07 61  31 32 2d 31 33 31 c0 84   ·······a 12-131··
00d0  c0 55 00 02 00 01 00 02  19 05 00 0a 07 61 31 33   ·U······ ·····a13
00e0  2d 31 33 30 c0 84 c0 55  00 02 00 01 00 02 19 05   -130···U ········
00f0  00 0a 07 61 31 38 2d 31  32 38 c0 84 c0 55 00 02   ···a18-1 28···U··
0100  00 01 00 02 19 05 00 0a  07 61 32 38 2d 31 32 39   ········ ·a28-129
0110  c0 84 c0 55 00 02 00 01  00 02 19 05 00 09 06 61   ···U···· ·······a
```

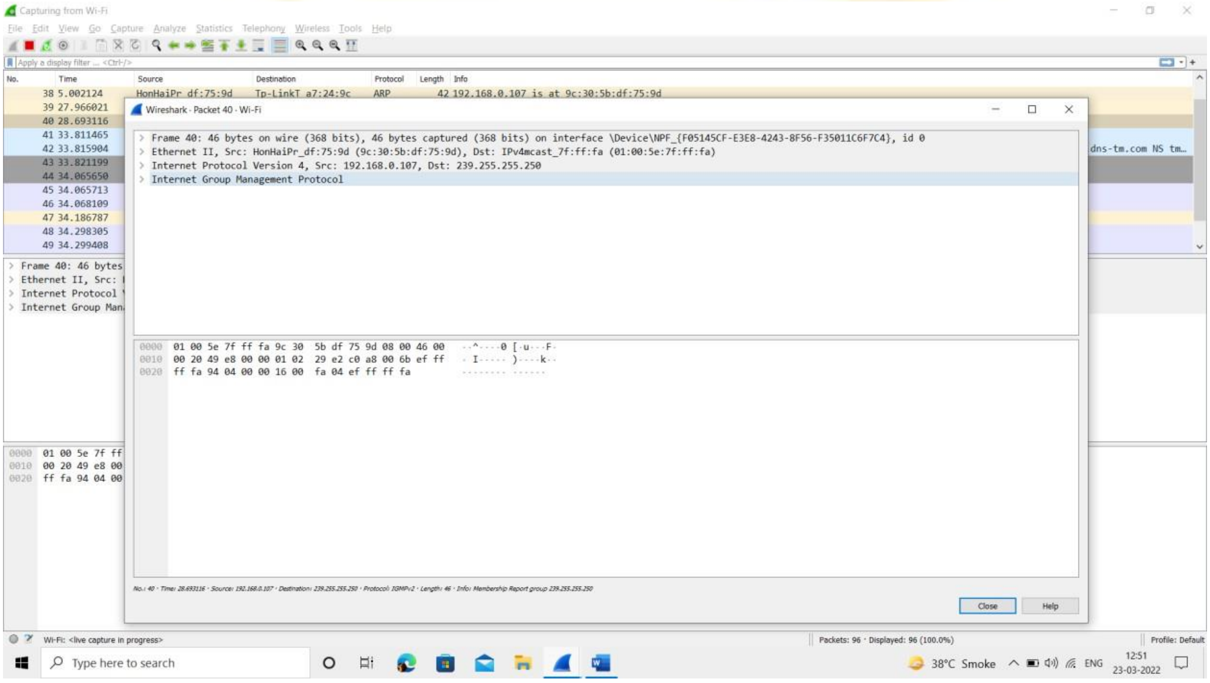
No.: 2 · Time: 0.004085 · Source: 192.168.0.1 · Destination: 192.168.0.107 · Protocol: DNS · Length: 460 · Info: Standard query response 0xb2c6 A nex...s.net NS a7-131.akadns.net NS a11-129.akadns.net NS a3-129.akadns.net A 96.7.49.129 A 23.61.199.131 A 84.53.139.129 A 193.108.88.128 A 184.85.248.128

Wi-Fi: <live capture in progress>

Packets: 95 · Displayed: 95 (100.0%)

Profile: Default

ICMP



Capturing from Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
Wireshark · Packet 40 · Wi-Fi
Close
Help
Type here to search
38°C Smoke
ENG
12:51
23-03-2022

```
No.  Time         Source              Destination         Protocol  Length  Info
  38 5.002124     HonHaiPr_df:75:9d   Tp-LinkT_a7:24:9c   ARP           42  192.168.0.107 is at 9c:30:5b:df:75:9d
  39 27.966021
  40 28.693116
  41 33.811465
  42 33.815904
  43 33.821199
  44 34.065650
  45 34.065713
  46 34.068109
  47 34.186787
  48 34.298305
  49 34.299408
```

```
> Frame 40: 46 bytes on wire (368 bits), 46 bytes captured (368 bits) on interface \Device\NPF_{F05145CF-E3E8-4243-8F56-F35011C6F7C4}, id 0
> Ethernet II, Src: HonHaiPr_df:75:9d (9c:30:5b:df:75:9d), Dst: IPv4mcast_7f:ff:fa (01:00:5e:7f:ff:fa)
> Internet Protocol Version 4, Src: 192.168.0.107, Dst: 239.255.255.250
> Internet Group Management Protocol
```

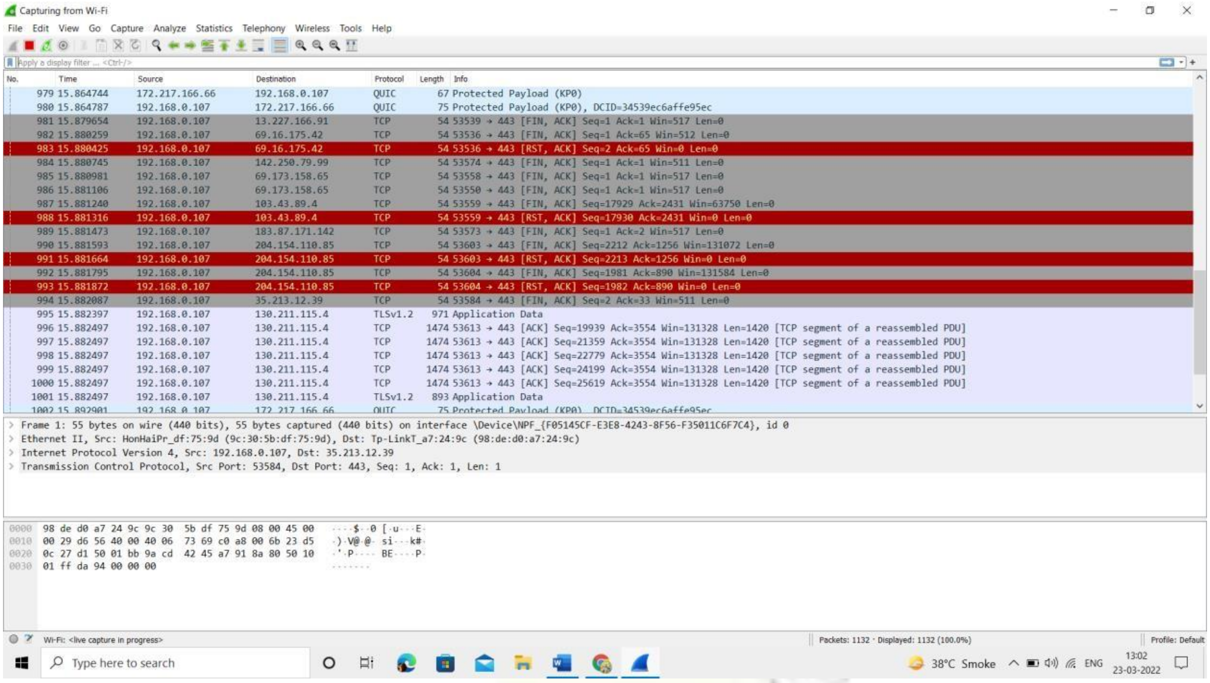
```
0000  01 00 5e 7f ff fa 9c 30  5b df 75 9d 08 00 46 00   ··^····0 [·u···F·
0010  00 20 49 e8 00 00 01 02  29 e2 c0 a8 00 6b ef ff   · I····· )····k··
0020  ff fa 94 04 00 00 16 00  fa 04 ef ff ff fa         ········ ······
```

No.: 40 · Time: 28.693116 · Source: 192.168.0.107 · Destination: 239.255.255.250 · Protocol: IGMPv2 · Length: 46 · Info: Membership Report group 239.255.255.250

Wi-Fi: <live capture in progress>

Packets: 96 · Displayed: 96 (100.0%)

Profile: Default


Capturing from Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

```
No.   Time       Source          Destination      Protocol Length Info
  979 15.864744  172.217.166.66  192.168.0.107    QUIC       67 Protected Payload (KP0)
  980 15.864787  192.168.0.107   172.217.166.66   QUIC       75 Protected Payload (KP0), DCID=34539ec6affe95ec
  981 15.879654  192.168.0.107   13.227.166.91    TCP        54 53539 → 443 [FIN, ACK] Seq=1 Ack=1 Win=517 Len=0
  982 15.880259  192.168.0.107   69.16.175.42     TCP        54 53536 → 443 [FIN, ACK] Seq=1 Ack=65 Win=512 Len=0
  983 15.880425  192.168.0.107   69.16.175.42     TCP        54 53536 → 443 [RST, ACK] Seq=2 Ack=65 Win=0 Len=0
  984 15.880745  192.168.0.107   142.250.79.99    TCP        54 53574 → 443 [FIN, ACK] Seq=1 Ack=1 Win=511 Len=0
  985 15.880981  192.168.0.107   69.173.158.65    TCP        54 53558 → 443 [FIN, ACK] Seq=1 Ack=1 Win=517 Len=0
  986 15.881106  192.168.0.107   69.173.158.65    TCP        54 53550 → 443 [FIN, ACK] Seq=1 Ack=1 Win=517 Len=0
  987 15.881240  192.168.0.107   103.43.89.4      TCP        54 53559 → 443 [FIN, ACK] Seq=17929 Ack=2431 Win=63750 Len=0
  988 15.881316  192.168.0.107   103.43.89.4      TCP        54 53559 → 443 [RST, ACK] Seq=17930 Ack=2431 Win=0 Len=0
  989 15.881473  192.168.0.107   183.87.171.142   TCP        54 53573 → 443 [FIN, ACK] Seq=1 Ack=2 Win=517 Len=0
  990 15.881593  192.168.0.107   204.154.110.85   TCP        54 53603 → 443 [FIN, ACK] Seq=2212 Ack=1256 Win=131072 Len=0
  991 15.881664  192.168.0.107   204.154.110.85   TCP        54 53603 → 443 [RST, ACK] Seq=2213 Ack=1256 Win=0 Len=0
  992 15.881795  192.168.0.107   204.154.110.85   TCP        54 53604 → 443 [FIN, ACK] Seq=1981 Ack=890 Win=131584 Len=0
  993 15.881872  192.168.0.107   204.154.110.85   TCP        54 53604 → 443 [RST, ACK] Seq=1982 Ack=890 Win=0 Len=0
  994 15.882087  192.168.0.107   35.213.12.39     TCP        54 53584 → 443 [FIN, ACK] Seq=2 Ack=33 Win=511 Len=0
  995 15.882397  192.168.0.107   130.211.115.4    TLSv1.2   971 Application Data
  996 15.882497  192.168.0.107   130.211.115.4    TCP      1474 53613 → 443 [ACK] Seq=19939 Ack=3554 Win=131328 Len=1420 [TCP segment of a reassembled PDU]
  997 15.882497  192.168.0.107   130.211.115.4    TCP      1474 53613 → 443 [ACK] Seq=21359 Ack=3554 Win=131328 Len=1420 [TCP segment of a reassembled PDU]
  998 15.882497  192.168.0.107   130.211.115.4    TCP      1474 53613 → 443 [ACK] Seq=22779 Ack=3554 Win=131328 Len=1420 [TCP segment of a reassembled PDU]
  999 15.882497  192.168.0.107   130.211.115.4    TCP      1474 53613 → 443 [ACK] Seq=24199 Ack=3554 Win=131328 Len=1420 [TCP segment of a reassembled PDU]
 1000 15.882497  192.168.0.107   130.211.115.4    TCP      1474 53613 → 443 [ACK] Seq=25619 Ack=3554 Win=131328 Len=1420 [TCP segment of a reassembled PDU]
 1001 15.882497  192.168.0.107   130.211.115.4    TLSv1.2   893 Application Data
 1002 15.892901  192.168.0.107   172.217.166.66   QUIC       75 Protected Payload (KP0), DCID=34539ec6affe95ec
```

```
> Frame 1: 55 bytes on wire (440 bits), 55 bytes captured (440 bits) on interface \Device\NPF_{F05145CF-E3E8-4243-8F56-F35011C6F7C4}, id 0
> Ethernet II, Src: HonHaiPr_df:75:9d (9c:30:5b:df:75:9d), Dst: Tp-LinkT_a7:24:9c (98:de:d0:a7:24:9c)
> Internet Protocol Version 4, Src: 192.168.0.107, Dst: 35.213.12.39
> Transmission Control Protocol, Src Port: 53584, Dst Port: 443, Seq: 1, Ack: 1, Len: 1
```

```
0000  98 de d0 a7 24 9c 9c 30  5b df 75 9d 08 00 45 00   ····$··0 [·u···E·
0010  00 29 d6 56 40 00 40 06  73 69 c0 a8 00 6b 23 d5   ·)·V@·@· si···k#·
0020  0c 27 d1 50 01 bb 9a cd  42 45 a7 91 8a 80 50 10   ·'·P···· BE····P·
0030  01 ff da 94 00 00 00                               ·······
```

Wi-Fi: <live capture in progress>
Packets: 1132 · Displayed: 1132 (100.0%)
Profile: Default
Type here to search
38°C Smoke
ENG
13:02
23-03-2022

Conclusion:

Wireshark installation and network traffic analysis using packet sniffing is done. Detailed information about packets are explored by applying filters**.**

For Faculty use

Correction Parameters	**Formative Assessment [40%]**	**Timely completion of Practical [40%]**	**Attendance / Learning Attitude [20%]**	
Marks Obtained				

EXPERIMENT 6

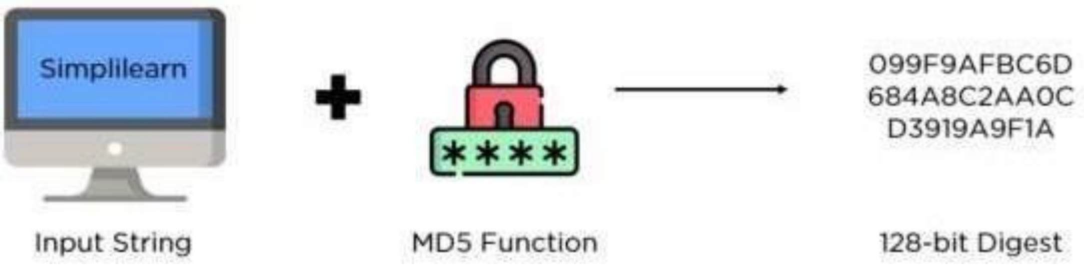
Aim: Analyze and implement MD5 and SHA1 cryptographic algorithm.

Theory:

Hashing consists of converting a general string of information into an intricate piece of data.This is done to scramble the data so that it completely transforms the original value, makingthe hashed value utterly different from the original. Hashing uses a hash function to convert standard data into an unrecognizable format. These hash functions are a set of mathematical calculations that transform the original information into their hashed values, known as the hash digest or digest in general. The digest size is always the same for a particular hash function like MD5 or SHA1, irrespective of input size.

MD5 (Message Digest Method 5) is a cryptographic hash algorithm used to generate a 128 - bit digest from a string of any length. It represents the digests as 32 digit hexadecimal numbers. The digest size is always 128 bits, and thanks to hashing function guidelines, a minor change in the input string generate a drastically different digest. This is essential to prevent similar hash generation as much as possible, also known as a hash collision.

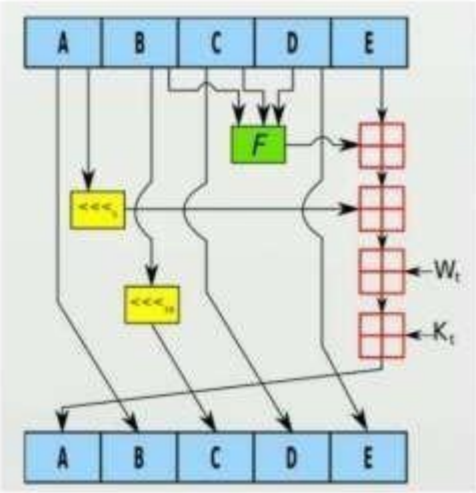
Functions associated:
encode (): Converts the string into bytes to be acceptable by hash function. digest()
: Returns the encoded data in byte format.
hexdigest () : Returns the encoded data in hexadecimal format.



SHA-1 produces a 160-bit hash value or message digests from the inputted data (data that requires encryption), which resembles the hash value of the MD5 algorithm. It uses 80 rounds of cryptographic operations to encrypt and secure a data object. Some of the protocols that use SHA-1 include:

1. Transport Layer Security (TLS)
2. Secure Sockets Layer (SSL)
3. Pretty Good Privacy (PGP)
4. Secure Shell (SSH)
5. Secure/Multipurpose Internet Mail Extensions (S/MIME)
6. Internet Protocol Security (IPSec)

SHA-1 is commonly used in cryptographic applications and environments where the need for data integrity is high. It is also used to index hash functions and identify data corruption and checksum errors.



Advantages of MD5:

- Easy to Compare: Unlike the latest hash algorithm families, a 32 digit digest is relatively easier to compare when verifying the digests.
- Storing Passwords: Passwords need not be stored in plaintext format, making them accessible for hackers and malicious actors. When using digests, the database also gets aboost since the size of all hash values will be the same.
- Low Resource: A relatively low memory footprint is necessary to integrate multiple services into the same framework without a CPU overhead.

- Integrity Check: You can monitor file corruption by comparing hash values before and after transit. Once the hashes match, file integrity checks are valid, and it avoids data corruption.

Disadvantages of MD5:

- When compared to other algorithms like the SHA algorithm, MD5 is comparatively slow.
- It is possible to construct the same hash function for two distinct inputs using MD5.
- MD5 is less secure when compared to the SHA algorithm since MD5 is more vulnerable to collision attacks.

Application:

- Password Verification
  It is common to store user credentials of websites in a hashed format to prevent third parties from reading the passwords. Since hash functions always provide the same output for the same input, comparing password hashes is much more private.
  The entire process is as follows:

  1. User signs up to the website with a new password
  2. It passes the password through a hash function and stores the digest on the server
  3. When a user tries to log in, they enter the password again
  4. It passes the entered password through the hash function again to generate a digest
  5. If the newly developed digest matches the one on the server, the login is verified.

- Integrity Verification
  Some files can be checked for data corruption using hash functions. Like the above scenario, hash functions will always give the same output for similar input, irrespective of iteration parameters. The entire process follows this order:

  1. A user uploads a file on the internet
  2. It also uploads the hash digest along with the file
  3. When a user downloads the file, they recalculate the hash digest
  4. If the digest matches the original hash value, file integrity is maintained

Algorithm:

- MD5

A 512-bit string is divided into 16 words of 32 bits each using the MD5 message- digest hashing method. MD5 generates a 128-bit message digest as a result of the operation. There are four steps involved in producing a message digest:

Appending padding bits to the original message.

1. Appending length bits.
2. Initializing MD or Message digest buffer.
3. Processing of messages in 16-word blocks to produce the final output or result.

Conclusion:

Successfully understood and implemented menu driven MD5 and Sha-1 hash algorithm using python library hashlib and found equivalent hash, hexadecimal and Byte value for a string input.

For faculty use

Correction Parameters	**Formative Assessment [40%]**	**Timely completion of Practical [40%]**	**Attendance / Learning Attitude [20%]**	
Marks Obtained				

EXPERIMENT 9

CASE STUDY

Title:

Setting up firewall using iptables in Kali Linux.

Introduction:

Kali Linux is funded and maintained by Offensive Security, an information training company. It is a Debian-based Linux distribution built with the aim of advanced penetration testing and security auditing. You may ask what the Debian standard is; it is an entire command-line system without an x11 or GUI environment. It is just a primary server. If you don't use it with a landline connection, you can only use it for learning the command line. It is equipped with a tonne of tools installed with it that makes an ethical hacker can go on a war with these tools in his arsenal.

Kali Linux is packed with essential tools for information security tasks, penetration testing, computer forensics, reverse engineering, and much more. As Kali Linux is just an operating software, it is not illegal by itself. But when someone uses it for hacking, it is considered illegal if someone uses it to learn, teach, or understand the intricacies as it is licensed for download.

Kali Linux contains several hundred tools targeted towards various information security tasks, such as Penetration Testing, Security Research, Computer Forensics and Reverse Engineering. Kali Linux is a multi-platform solution, accessible and freely available to information security professionals and hobbyists.

Advantages:

- It is free: It is free for download.
- Multi-lingual support: Although most applications and tools are primarily written in English, Kali Linux provides multi-lingual support. That opens opportunities for people to use these resources in their local language and use it for their customizable purpose.
- Customizable: The developers at Kali Linux have been very liberal while developing this and have left opportunities for customization for fellow developers to come on board and modify it as per their likings.
- Open source: As this belongs to the Linux family, it is available on an open-source platform. The entire development tree, along with codes, is known for viewing and modifying on Git.
- A plethora of tools available: Kali Linux comes equipped with more than 600 different devices.

4 Reasons Why Hackers Use Kali Linux

- Requires Minimal System Resources
- Kali Linux Is Good for Beginners
- Kali Linux Is Legal
- Kali Linux Ships With Hundreds of Testing Tools

A Firewall is a network security device that monitors and filters incoming and outgoing network traffic based on an organization’s previously established security policies. At its most basic, a firewall is essentially the barrier that sits between a private internal network and the public Internet. A firewall’s main purpose is to allow non-threatening traffic in and to keep dangerous traffic out.

Firewalls have existed since the late 1980’s and started out as packet filters, which were networks set up to examine packets, or bytes, transferred between computers. Though packet filtering firewalls are still in use today, firewalls have come a long way as technology has developed throughout the decades.

Background:

Iptables is an extremely flexible firewall utility built for Linux operating systems. Whether you’re a novice Linux geek or a system administrator, there’s probably some way that iptables can be a great use to you. Read on as we show you how to configure the most versatile Linux firewall. iptables is a command-line firewall utility that uses policy chains to allow or block traffic. When a connection tries to establish itself on your system, iptables looks for a rule in its list to match it to. If it doesn’t find one, it resorts to the default action.

- Types of chains: input, forward and output.
- Input – This chain is used to control the behavior for incoming connections. For example, if a user attempts to SSH into your PC/server, iptables will attempt to match the IP address and port to a rule in the input chain.
- Forward – This chain is used for incoming connections that aren’t actually being delivered locally. Think of a router – data is always being sent to it but rarely actually destined for the router itself; the data is just forwarded to its target. Unless you’re doing some kind of routing, NATing, or something else on your system that requires forwarding, you won’t even use this chain.
- Output – This chain is used for outgoing connections. For example, if you try to ping howtogeek.com, iptables will check its output chain to see what the rules are regarding ping and howtogeek.com before making a decision to allow or deny the connection attempt.

Connection-specific Responses - With your default chain policies configured, you can start adding rules to iptables so it knows what to do when it encounters a connection from or to a particular IP address or port. In this guide, we’re going to go over the three most basic and commonly used “responses”. Accept – Allow the connection.

Drop – Drop the connection, act like it never happened. This is best if you don’t want the source to realize your system exists.

Reject – Don’t allow the connection, but send back an error. This is best if you don’t want a

particular source to connect to your system, but you want them to know that your firewall blocked them.

Methodology:

Adding rules

By default, iptables does not have any rules defined. You can verify this yourself on a new

server by typing the following command: **iptables -L**

```
iptables -A INPUT -i lo -j ACCEPT iptables -A INPUT -m state --state
RELATED,ESTABLISHED -j ACCEPT iptables -A INPUT -p tcp -m tcp --dport 7822
-j ACCEPT iptables -A INPUT -j DROP
```

In all of these commands, the **-A** option instructs iptables to append the rule to the end of the specified chain (in this case, the **INPUT** chain). Let's step through each command:

- The first command permits all packets for the local loopback interface. Many programs use the loopback interface, so it is a good idea to accept packets on it.
- The second command uses the **-m** option to load the state module. This module determines and monitors a packet's state, which can be **NEW**, **ESTABLISHED**, or **RELATED**. In this rule, we accept incoming packets that belong to a connection that has already been established.
- The third command accepts incoming TCP connections on port 7822 (SSH).
- The last command drops (rejects) incoming packets that do not match any of the preceding rules.

Inserting rules

The set of rules we defined above is pretty limited. If SSH is the only incoming connection you want to allow, then you're all set. Most likely, though, you will need to add access to services as you configure your server.

However, if we just add a rule using the **-A** option shown above, it will be the last rule in the chain, right after our **DROP** rule. Because iptables works through rules in sequence, this means that it will never get to the new rule, because the packet will have already been dropped. Therefore, we need a way to insert new rules into the chain.

The **-I** option enables us to insert a new rule anywhere in the chain. Let's insert a rule that allows incoming TCP connections on port 80 (HTTP). We want the rule to come just before the **DROP** rule, which is currently the fourth rule in the chain:

This inserts our HTTP rule in the fourth line, and pushes the **DROP** rule down to the fifth line.

```
iptables -I INPUT 4 -p tcp -m tcp --dport 80 -j ACCEPT
```

Deleting rules

To delete a rule, use the **-D** option. You need to know the number of the rule you want to delete (just as you must know the number when you insert a rule). The following command demonstrates how to delete the fifth rule from the **INPUT** chain:

```
iptables -D INPUT 5
```

If you want to delete **all** of the rules at once, type the following command:

```
iptables -F
```

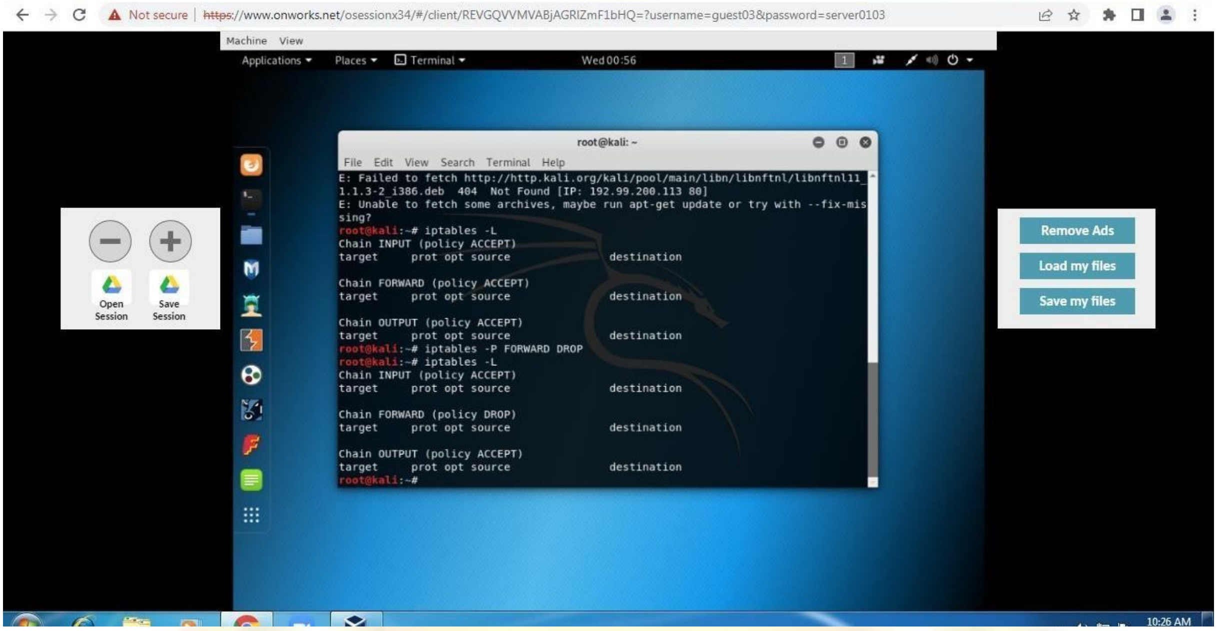
Saving rules

If you reboot the server now, all of the rules you defined will be erased. To maintain rules across system restarts, you must save them. The steps to do this depend on the Linux distribution you are running.

Results:

1. Displaying iptables:

Command - iptables - L

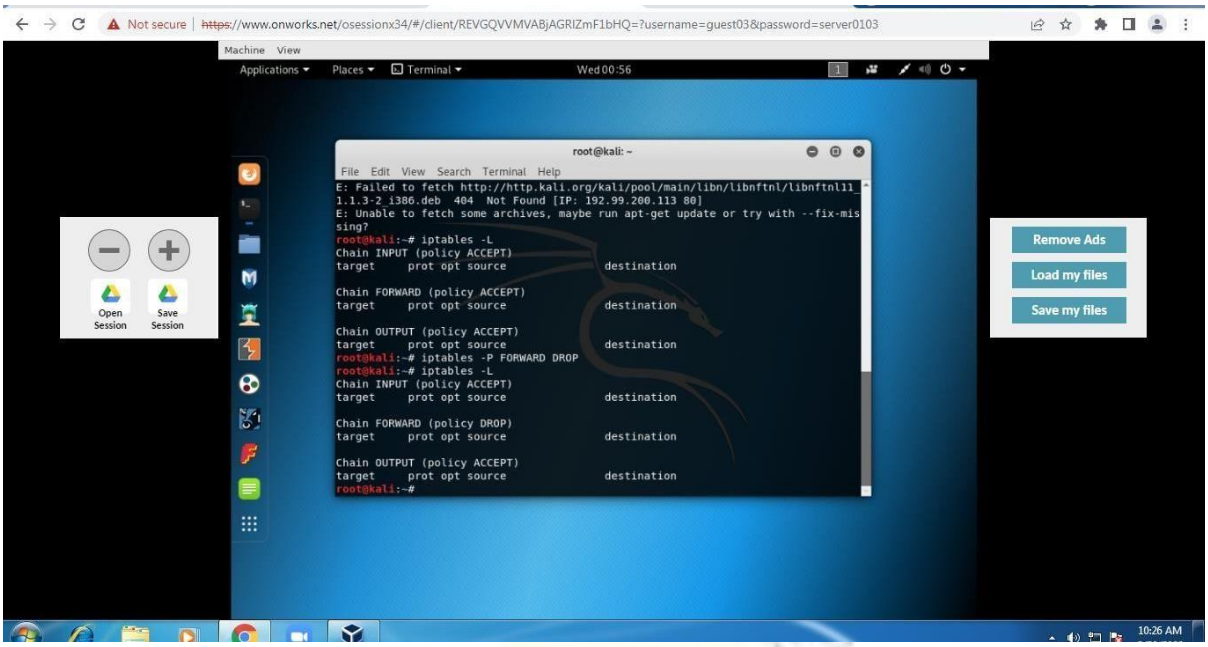


```
E: Failed to fetch http://http.kali.org/kali/pool/main/libn/libnftnl/libnftnl11_
1.1.3-2_i386.deb  404  Not Found [IP: 192.99.200.113 80]
E: Unable to fetch some archives, maybe run apt-get update or try with --fix-mis
sing?
root@kali:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@kali:~# iptables -P FORWARD DROP
root@kali:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@kali:~#
```

Not secure | https://www.onworks.net/osessionx34/#/client/REVGQVVMVABjAGRIZmF1bHQ=?username=guest03&password=server0103
Machine View
Applications ▾ Places ▾ Terminal ▾ Wed 00:56
root@kali: ~
File Edit View Search Terminal Help
Open Session
Save Session
Remove Ads
Load my files
Save my files
10:26 AM

2. To drop Forward

Command - *iptables -P FORWARD DROP*
iptables -L



Not secure | https://www.onworks.net/osessionx34/#/client/REVGQVVMVABjAGRIZmF1bHQ=?username=guest03&password=server0103
Machine View
Applications ▾
Places ▾
Terminal ▾
Wed 00:56
Open Session
Save Session
Remove Ads
Load my files
Save my files
root@kali: ~
File Edit View Search Terminal Help
10:26 AM

```
E: Failed to fetch http://http.kali.org/kali/pool/main/libn/libnftnl/libnftnl11_
1.1.3-2_i386.deb  404  Not Found [IP: 192.99.200.113 80]
E: Unable to fetch some archives, maybe run apt-get update or try with --fix-mis
sing?
root@kali:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@kali:~# iptables -P FORWARD DROP
root@kali:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@kali:~#
```

3. **To drop given IP address**

   Let’s assume we want to block the traffic coming from an IP address 192.168.0.23
   Command -
   Iptables -A INPUT -S 192.168.0.23 -j DROP iptables
   -L

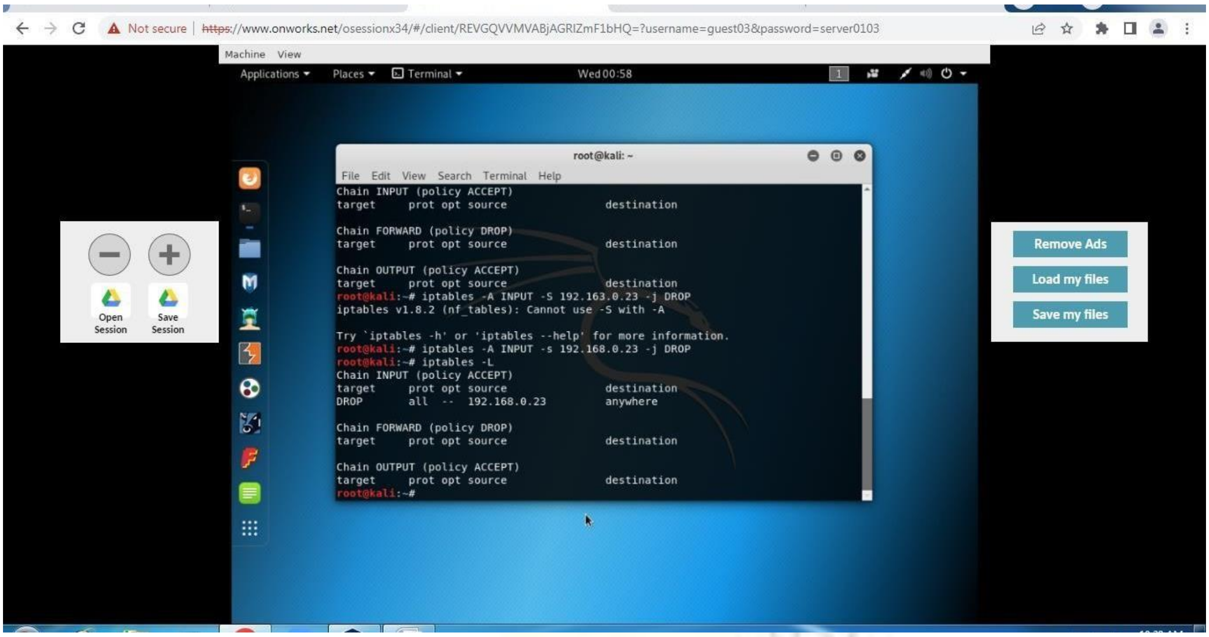
   A INPUT :- The flag -A is used to append a rule to the end of a chain. This part of the command tells the iptable that we want to add a rule to the end of the INPUT chain.

   -I INPUT:- In this flag the rules are added to the top of the chain.

   -s 192.168.0.23:- The flag -s is used to specify the source of the packet. This tells the iptable to look for the packets coming from the source 192.168.1.3

   -j DROP - This specifies what the iptable should do with the packet.

In short, the above command adds a rule to the INPUT chain which says, if any packet arrives whose source address is 192.168.0.23 then drop that packet, that means do not allow the packet reach the computer.

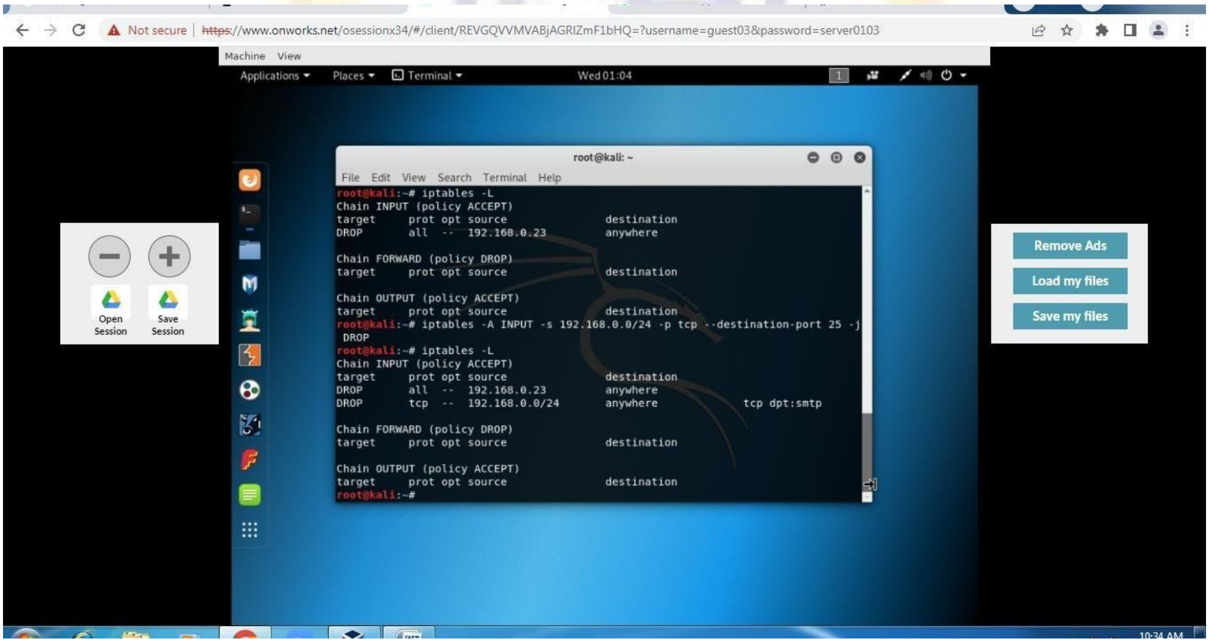


Not secure | https://www.onworks.net/osessionx34/#/client/REVGQVVMVABjAGRIZmF1bHQ=?username=guest03&password=server0103
Machine View
Applications ▾ Places ▾ Terminal ▾
Wed 00:58
root@kali: ~
File Edit View Search Terminal Help
Open Session
Save Session
Remove Ads
Load my files
Save my files

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@kali:~# iptables -A INPUT -S 192.163.0.23 -j DROP
iptables v1.8.2 (nf_tables): Cannot use -S with -A

Try `iptables -h' or 'iptables --help' for more information.
root@kali:~# iptables -A INPUT -s 192.168.0.23 -j DROP
root@kali:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  192.168.0.23         anywhere

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@kali:~#
```

4. **To drop emails coming from a specific email address as well as blocking all the traffic using tcp and smtp**

   Command -
   iptables -A INPUT -s 192.168.0.0/24 -p tcp - - destination-port 25 -j DROP
   iptables -L



Not secure | https://www.onworks.net/osessionx34/#/client/REVGQVVMVABjAGRIZmF1bHQ=?username=guest03&password=server0103
Machine View
Applications ▾ Places ▾ Terminal ▾
Wed 01:04
root@kali: ~
File Edit View Search Terminal Help
Open Session
Save Session
Remove Ads
Load my files
Save my files
10:34 AM

```
root@kali:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  192.168.0.23         anywhere

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@kali:~# iptables -A INPUT -s 192.168.0.0/24 -p tcp --destination-port 25 -j
 DROP
root@kali:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  192.168.0.23         anywhere
DROP       tcp  --  192.168.0.0/24       anywhere             tcp dpt:smtp

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@kali:~#
```

5. To accept rule:

If you want to add rules to specific ports of your network,then the following commands can be used.

Syntax:- sudo iptables -A/-I chain_name -s source_ip -p protocol_name --dport port_number -j Action_to_take

-p protocol_name:-

This option is used to match the packets that follow the protocol protocol_name.

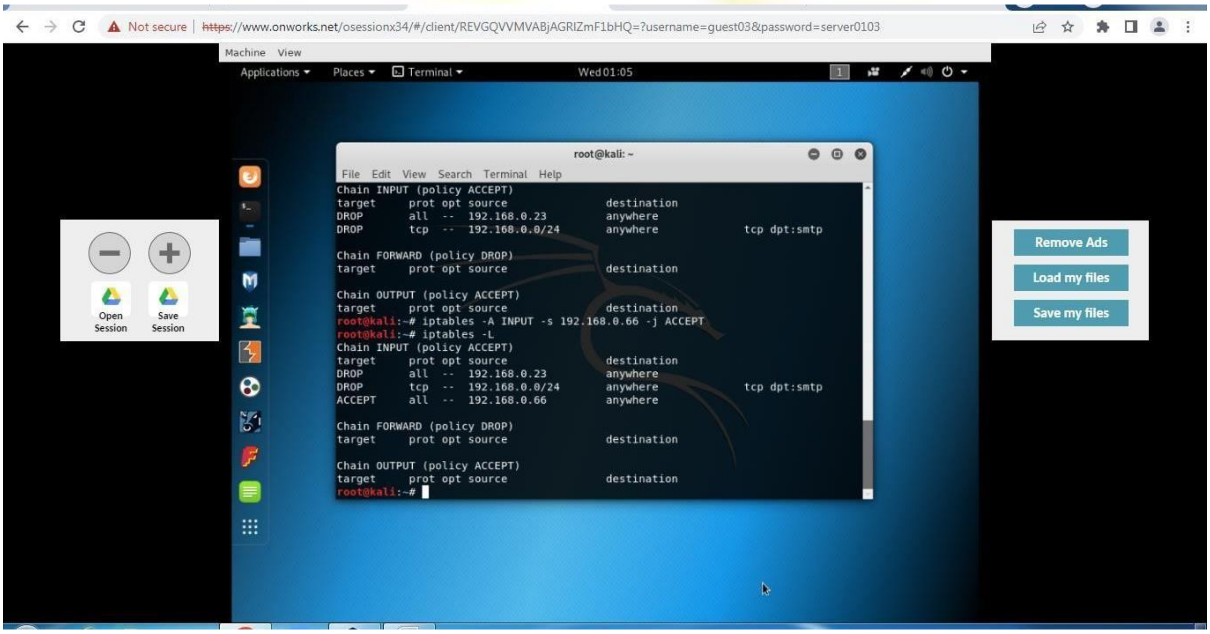
-dport port_number:

This is option is available only if you give the -p protocol_name option. It specifies to look for the packets that are going to the port "port_number".

Command -

iptables -A INPUT -s 192.168.0.66 -j ACCEPT

iptables -L



Not secure | https://www.onworks.net/osessionx34/#/client/REVGQVVMVABjAGRIZmF1bHQ=?username=guest03&password=server0103
Machine View
Applications ▾ Places ▾ Terminal ▾
Wed 01:05
root@kali: ~
File Edit View Search Terminal Help
Open Session
Save Session
Remove Ads
Load my files
Save my files

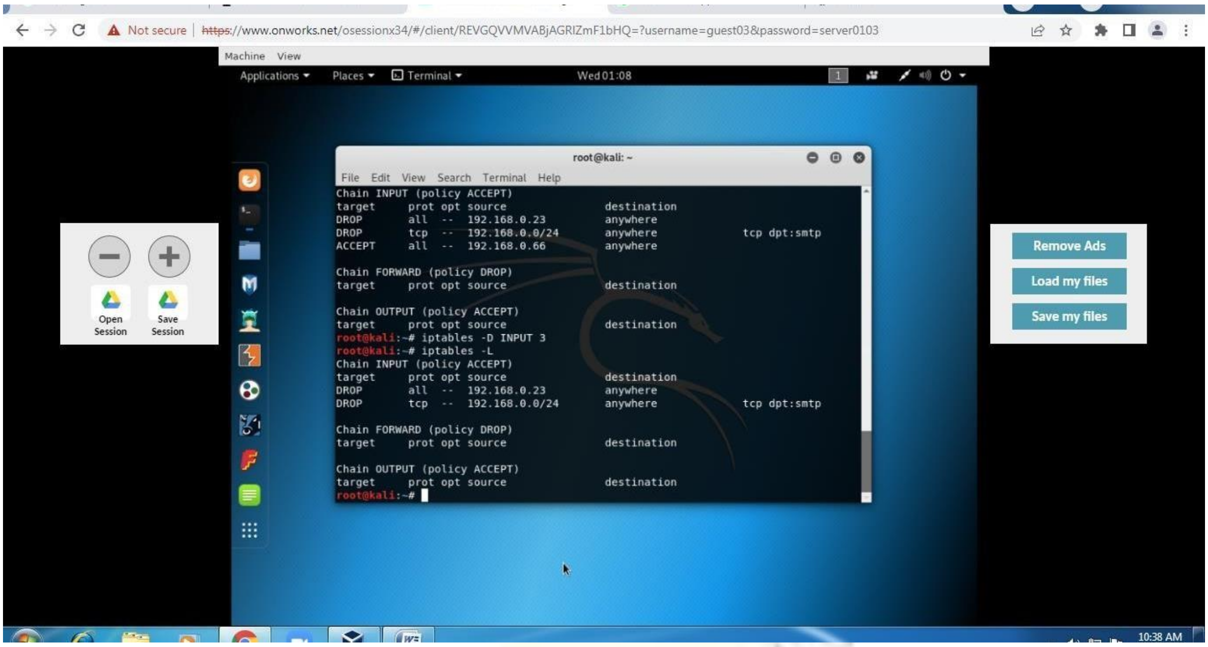
```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  192.168.0.23         anywhere
DROP       tcp  --  192.168.0.0/24       anywhere             tcp dpt:smtp

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@kali:~# iptables -A INPUT -s 192.168.0.66 -j ACCEPT
root@kali:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  192.168.0.23         anywhere
DROP       tcp  --  192.168.0.0/24       anywhere             tcp dpt:smtp
ACCEPT     all  --  192.168.0.66         anywhere

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@kali:~#
```

6. Deleting rule from iptable

Command -

iptables -D INPUT

3 iptables -L



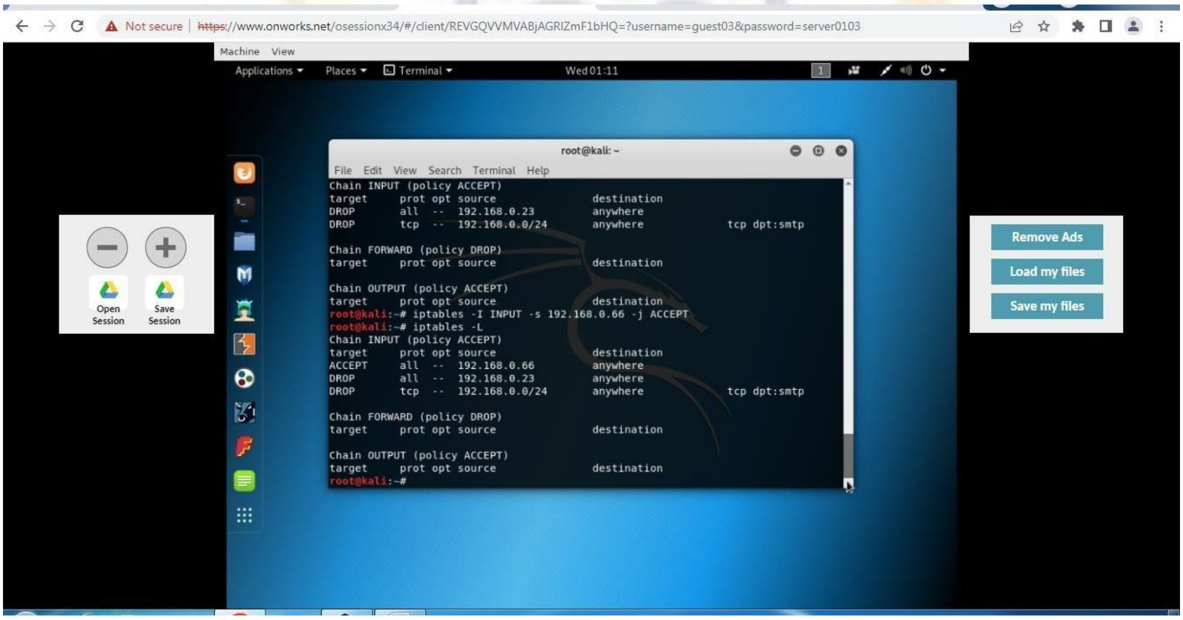
Not secure
https://www.onworks.net/osessionx34/#/client/REVGQVVMVABjAGRIZmF1bHQ=?username=guest03&password=server0103
Machine View
Applications ▾ Places ▾ Terminal ▾
Wed 01:08
root@kali: ~
File Edit View Search Terminal Help
Open Session
Save Session
Remove Ads
Load my files
Save my files
10:38 AM

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  192.168.0.23         anywhere
DROP       tcp  --  192.168.0.0/24       anywhere             tcp dpt:smtp
ACCEPT     all  --  192.168.0.66         anywhere

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@kali:~# iptables -D INPUT 3
root@kali:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  192.168.0.23         anywhere
DROP       tcp  --  192.168.0.0/24       anywhere             tcp dpt:smtp

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@kali:~#
```

7. Accepting rule from iptables in order

Command -

iptables -I INPUT -s 192.168.0.66 -j ACCEPT
iptables -L

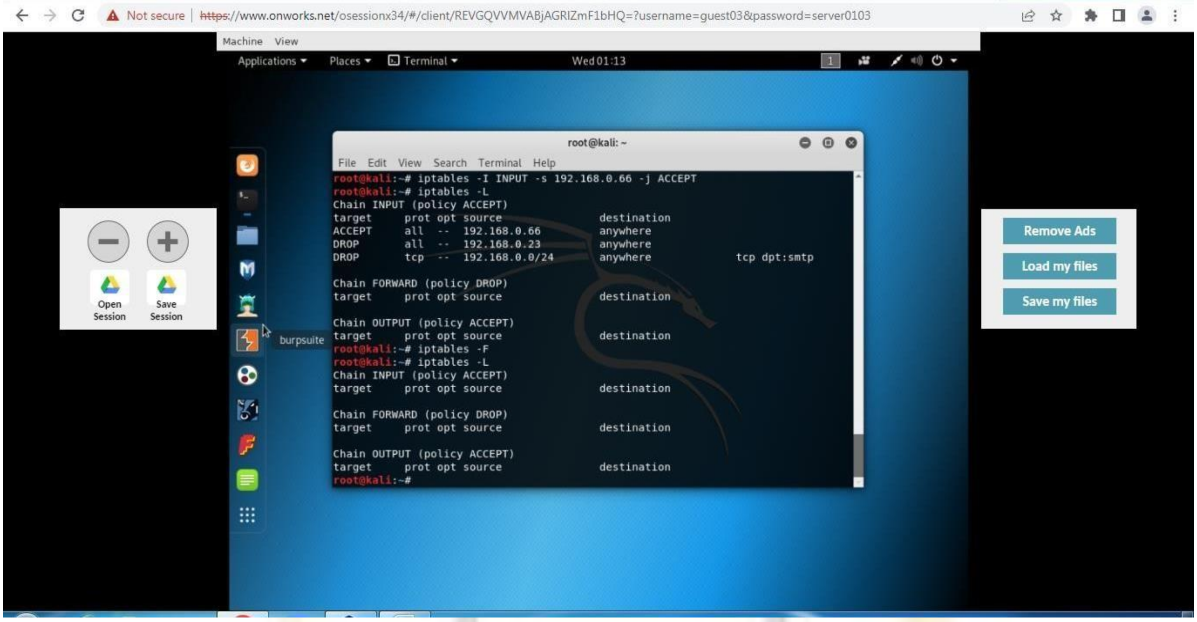


Not secure
https://www.onworks.net/osessionx34/#/client/REVGQVVMVABjAGRIZmF1bHQ=?username=guest03&password=server0103
Machine View
Applications ▾ Places ▾ Terminal ▾
Wed 01:11
root@kali: ~
File Edit View Search Terminal Help
Open Session
Save Session
Remove Ads
Load my files
Save my files

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  192.168.0.23         anywhere
DROP       tcp  --  192.168.0.0/24       anywhere             tcp dpt:smtp

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@kali:~# iptables -I INPUT -s 192.168.0.66 -j ACCEPT
root@kali:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  192.168.0.66         anywhere
DROP       all  --  192.168.0.23         anywhere
DROP       tcp  --  192.168.0.0/24       anywhere             tcp dpt:smtp

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@kali:~#
```

8. Flushing all rules from iptable

Command
*- iptables -
F iptables
-L*


Not secure | https://www.onworks.net/osessionx34/#/client/REVGQVVMVABjAGRlZmF1bHQ=?username=guest03&password=server0103
Machine View
Applications ▾ Places ▾ Terminal ▾ Wed 01:13
root@kali: ~
File Edit View Search Terminal Help
Open Session
Save Session
burpsuite
Remove Ads
Load my files
Save my files

```
root@kali:~# iptables -I INPUT -s 192.168.0.66 -j ACCEPT
root@kali:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  192.168.0.66         anywhere
DROP       all  --  192.168.0.23         anywhere
DROP       tcp  --  192.168.0.0/24       anywhere             tcp dpt:smtp

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@kali:~# iptables -F
root@kali:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@kali:~#
```

Analysis:

Using the iptables program, you can explicitly grant and deny access to selected services running on your server, as well as to selected IP addresses. Iptables is a powerful firewall program that you can use to secure your Linux server or VPS. What's great is that you can define various rules based on your preferences. In this iptables tutorial, you have learned how to install and use the tool. Now, we hope you can manage your sets of rules to filter incoming and outgoing packets.

Conclusion:

Successfully understood and implemented firewall using iptables and implemented various commands on kali Linux.

For Faculty use

Correction Parameters	**Formative Assessment [40%]**	**Timely completion of Practical [40%]**	**Attendance / Learning Attitude [20%]**	
Marks Obtained				