SL STEPS

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

Download 3 tar files of Netkit from https://www.netkit.org/

Unzip them

```
tar -xjSf netkit-2.8.tar.bz2
tar -xjSf netkit-filesystem-i386-F5.2.tar.bz2
tar -xjSf netkit-kernel-i386-K2.8.tar.bz2
```

Type in terminal

```
gedit ~/.bashrc
```

add following lines at the end of bash file.

```
export NETKIT_HOME=/home/apsit/Downloads/netkit
 export PATH=$PATH:$NETKIT_HOME/bin
export MANPATH=:$NETKIT_HOME/man
export PATH=$NETKIT_HOME/bin:$PATH
```

Now, go to netkit folder and check configuration

```
cd netkit
./check_configuration.sh
```

Now, type

```
vstart pc1 --eth0=A
vstart pc2 --eth0=A
vstart pc3 --eth0=A
```

PC1: ifconfig eth0 192.168.1.11
PC2: ifconfig eth0 192.168.1.12
PC3: ifconfig eth0 192.168.1.13

In PC1: ping 192.168.1.13

Stop then write,

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

Now, in PC2: tcpdump -i any

In PC1: ping 192.168.1.13

Install arpwatch

```
sudo apt install arpwatch
```

check status

```
service arpwatch status
```

do ifconfig to check ip and mac addr

```
sudo ifconfig enp4s0 hw ether <mac addr>
```

Run

```
sudo tail -f /var/log/syslog
```

ping your ip

To study installation and configuration of Linux Kernel firewall iptables.

Making default policy of INPUT and OUTPUT Chain as DROP :

```
sudo iptables -A INPUT -j DROP
sudo iptables -L -n -v –line-numbers

sudo iptables -A INPUT -j DROP
sudo iptables -L -n -v –line-numbers
```

Flushing out all the rules:

```
sudo iptables -F
sudo iptables -L -n -v –line-numbers
```

Allowing the ports:

```
sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 443 -j ACCEPT
sudo iptables -L -n -v –line-numbers
```

Deleting the rule:

```
sudo iptables -D INPUT 1
sudo iptables -L -n -v –line-numbers
```

List current rules

sudo iptables -L

Allow Established and Related Incoming Connections

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

Allow Established Outgoing Connections

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

Allow All Incoming and Outgoing SSH

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

Allow outgoing SSH to Specific IP address or subnet

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

Allow Incoming Rsync from Specific IP Address or Subnet

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

Allow All Incoming HTTP and HTTPS

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

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

tcpdump -D : display all available interfaces

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

tcpdump -i any : capture traffic at any interface

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

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

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

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

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
```

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 "tcp[tcpflags] & (tcp-syn|tcp-ack) != 0"
```

To capture ssh packet:

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

To capture telnet packet:

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

WIRESHARk:

Install wireshark

Try telnet and ssh command

Then go to wireshark, start capture

Click on any field, then click on analyze – follow tcp stream

Enable Promiscous mode:

sudo ip link set wlo1 promisc off

To use nmap for network discovery and security auditing

Install NMAP

sudo apt-get install nmap

To scan a single system

nmap -sP 192.168.43.32

To scan the entire subnet

nmap -sP 192.168.43.32/24

To scan a multiple targets

nmap -sP 192.168.43.32 192.168.43.169

To see the list of all the hosts that are being scanned

nmap -sL 192.168.43.32 192.168.43.169

scan the target for port number 80,21 and 23.

nmap -p 80,21,23 192.168.43.32

To know the open ports on target system:

nmap -open 192.168.43.32

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

nmap --top-ports 5 192.168.43.32

OS Detection by using Nmap

Nmap -O 192.168.43.32

Idle Scan (-sI)

nmap -sI 192.168.43.169 192.168.43.32

Version Detection (-sV)

nmap -sV 192.168.43.169

FIN Scan (-sF)

nmap -sF 192.168.43.32

UDP Scan (-sU)

nmap -sU 192.168.43.32

TCP connect() scan (-sT)

nmap -sT 192.168.43.32

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

1 Create a file – touch a.txt
2 Check md5 hash of file - md5sum a.txt
3 edit file – cat > a.txt
4 Check md5 hash of file – md5

Repeat the above process for sha1sum, sha224, 256, 384 and 512

Check iso and other file also

To study Intrusion Detection system SNORT and its log analysis.

Install Snort

sudo apt-get install snort

Editing snort configuration files

Type:

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

ipvar HOME_NET 192.168.43.130/24 {NOTE: put your ipaddr here. Remove number after last point and make it 0. Then put /24 at end}

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

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

alert icmp any any -> {your ip with subnet} any (msg:"ICMP test"; sid:1000001; rev:1;)

Run Snort test conf. and local rules

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

sudo snort -T -c /etc/snort/rules/local.rules

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

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

Now, ping from other computer and wait for some time. Snort is in listening mode. TaKe ss

Go to /var/log/snort to see alerts

cd /var/log/snort

ls

cat alert

Read log file by tcp dump

sudo tcpdump -r snort.log

To study access control list by configuring SQUID proxy server.

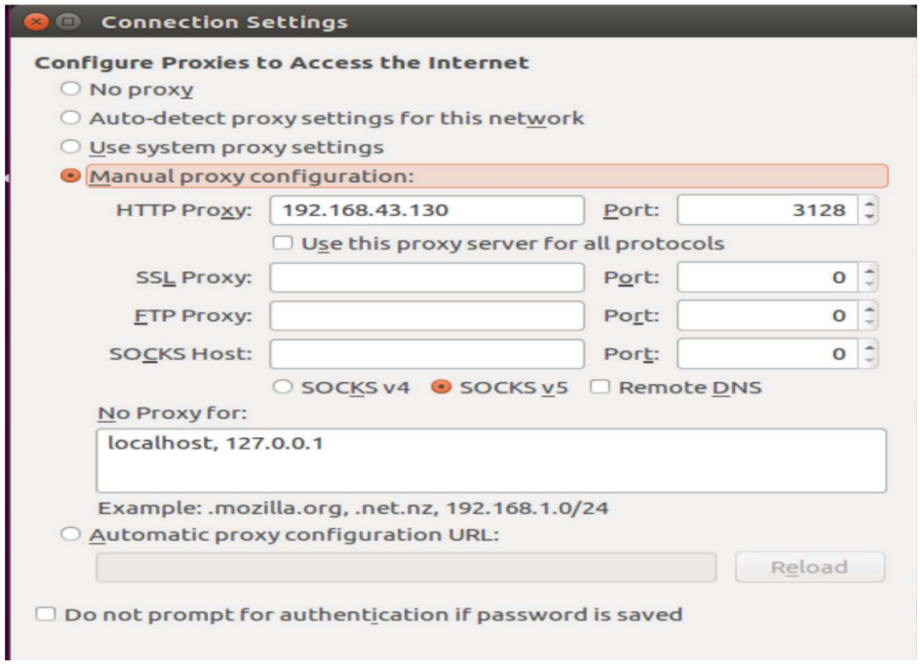
Installation of SQUID:

sudo apt-get install squid

Check status of SQUID :

cd /etc/squid
service squid status

Change Firefox proxy settings



Backing up the Squid configuration file:

cp squid.conf squid.conf.copy

Open Squid file and make changes:

sudo gedit squid.conf

Change in visible proxy name:

visible_hostname Name_Proxy_Server

Restricting Access By Client

```
acl resclient src 192.168.0.104
http_access allow localnet !resclient
```

To restrict access to Squid by domain

```
acl CONNECT method CONNECT
acl resclient src 10.211.55.4
acl forbidden dstdomain .facebook.com
http_access deny forbidden
http_access allow localnet !resclient
```

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 MWTFA 10:00-20:00
http_access allow forbidden workingHour
http_access deny forbidden
```

Restricting access by user authentication

```
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 APSIT
acl ncsa proxy_auth REQUIRED
http_access allow ncsa
```

Run below command to create passwd file

```
sudo chmod 777 /etc/squid/passwd
htpasswd -c /etc/squid/passwd apsit
```

NOTE: RESTART SQUID AFTER EACH FILE SAVE - service squid restart

To study symmetric and asymmetric encryption methods using Cryptool.

Heh

Aim: To study and implement IPSEC in Linux .

Installing IPSEC

```
Sudo apt-get install ipsec-tools strongswan-starter
```

Change directory and open ipsec configuration file

```
cd /etc
sudo gedit ipsec.conf
```

Add following lines at end

```
conn blue-to-red
        authby=secret
        auto=route
        keyexchange-ikev2
        ike=aes128-md5-modp1024
        left=192.168.87.13
        right=192.168.87.14
        type=transport
        esp=aes128-sha-modp1024!
```

Edit ipsec.secret file and write

```
        sudo gedit /etc/ipsec.secret

        192.168.87.13 192.168.87.14  :  PSK  "test123"
```

Now do same in and computer and write same .secret and .config file

Now in computer1, write

```
sudo ipsec restart
sudo ipsec up blue-to-red
```

If you write different password, it will show error