Laboratory Works no. 2

Linux console based security audit

Nowadays there is a tendency to neglect the classic commands of Linux at the expense of more sophisticated complex tools that often also have a graphical interface. However, there are situations where auditing must be carried out from low-resource devices (such as microcontrollers) or a minimum operating system (e.g. a docker image) is required. In these circumstances, it is necessary for any expert in the field to master these basic and or advanced commands without any problems.

1. Simple commands for analyzing the behaviour and operation of the network:

a) **P**acket **I**nter**N**et **G**roper or *ping* as it is commonly known this command is a command that allows to verify the existence as active (so called "live") of a station on the local or international network. It is in fact the use of the radar principle only that this is done by means of an ICMP package sent and re-received. This can also measure the delay times (latency) of a communication with that node in the test, the life time of the network package - TTL and supports the following parameters:

- **/?** Lists command syntax options.
- **-t** Pings the specified host until stopped with Ctrl+C. ping -t is also known as the ping of death. It can be used as a denial-of-service (DoS) attack to cause a target machine to crash.
- **-a** resolve (if possible) the names of the addresses through which the network packet passes.
- **-n** counter - in the counter explicitly specify a number of test packages of your choice
- **-r** counter makes a record of the path (only for IPV4) and has a maximum value of 9 (for a larger number of nodes is used traceroute or specially developed program).
- **-s** count - time stamps for measuring transfer time from one node to another (only on IPV4).
- **-i** TTL with maximum value 255

Some simple examples:

ping www.yahoo.com

ping atsv2-fp-shed.wg1.b.yahoo.com (87.248.98.7) 56(84) bytes of data.

64 bytes from media-router-fp1.prod1.media.vip.ir2.yahoo.com (87.248.98.7): icmp_seq=1 ttl=48 time=59.1 ms

b) *Ifconfig* is already replaced for various reasons including efficiency and security with a subset of equivalent command parameters of the *ip* command, namely *ip* a (*ip* addr), *ip* link, *ip* -s (*ip* -stats). We will therefore discuss the *ip* command. It is used for the following purposes:

- finding communication interfaces configured in the current system
- finding out the status of the IP interface
- configuring the local loop-back loop for Ethernet and other IP interfaces
- starting or stopping the operation of a particular interface.
- display ARP or NDISC cache entries.
- associating, deleting, initializing IP networks, routes, subnets and other IP interface information.
- displaying and managing the status of all active networks
- providing information about multicast addresses
- deleting or initializing an entry in the input table
- displaying neighboring objects such as ARP cache, invalidating ARP cache, adding an ARP cache, and more.
- finding a path to a specific address - changing the status of the interface

As you can see, this command is an integrator for most commands specific to the configuration of the parameters of the communication interfaces of the host computer and beyond. The command supports three generic approaches outlined below:

ip OBJECT COMMAND
ip [options] OBJECT COMMAND
ip OBJECT help

The object referred to in the generic order examples may be chosen in a complete or abbreviated form from the possibilities set out in the table below.

Object	**Short form**	**Target**
link	l	Network device
address	a addr	Device protocol address (IP or IPv6)
addrlabel	addrl	Choosing label to select protocol address.
neighbour	n neigh	cache ARP or NDISC entry.
route	r	Routing table entry.
rule	ru	Routing policy database rule
maddress	m maddr	Multicast address
mroute	mr	Multicast routing cache entry
tunnel	t	IP tunnel.
xfrm	x	IPsec framework .

If you want information about an object, you can use help commands in the following manner:

ip OBJECT help
ip OBJECT h
ip a help
ip r help

Let's present some simple examples of use:

- the display of all information about all communication interfaces is done with:

ip a

- alternatively we can use:

ip addr

If you want only those that use only IPV4

ip -4 a

If I only want specific information about a specific network interface:

ip a show eth0

ip a list eth0

ip a show dev eth0

Or they can only display active interfaces with:

ip link ls up

To add an IP address for a specific interface:

Generic ip a add {ip_addr/mask} dev {interface}
For example, to assign address 192.168.1.200/255.255.255.0 to eth0:

ip a add 192.168.1.200/255.255.255.0 dev eth0
or
ip a add 192.168.1.200/24 dev eth0

Adding a broadcast address to an interface is not done by default by command. This must be explicitly specified using the following alternative syntax's.

ip addr add brd {ADDDRESS-HERE} dev {interface}
ip addr add broadcast {ADDDRESS-HERE} dev {interface}
ip addr add broadcast 172.20.10.255 dev dummy0

Special symbols such as +/- are also allowed to be used instead of the broadcast address by setting or resetting bits that describe the interface. The following example sets the address 192.168.1.50 with the network mask 255.255.255.0 (/24) which has standard broadcast and the label "eth0A" to the eth0 interface:

ip addr add 192.168.1.50/24 brd + dev eth0 label eth0Home

A circular reference can also be made to the system loop-back

ip addr add 127.0.0.1/8 dev lo brd + scope host

You can remove or delete an IP address associated with a particular interface, the command has the following syntax:

ip a del {ipv6_addr_OR_ipv4_addr} dev {interface}

If, for example, we want to delete 192.168.1.200/24 from eth0:

ip a del 192.168.1.200/24 dev eth0

Conditionally clear (flush) an IP address. Although it can be addressed with address there are situations where we want to process all the stations in a network segment. For example, all addresses on a 192.168.2.0/24 private network can be simultaneously deleted with:

ip -s -s a f to 192.168.2.0/24

If, for example, I want to disable all IP addresses for all interfaces I can give:

ip -4 addr flush label "ppp*"

Same for interfaces, etc.:This example will delete the path created in the previous example

ip -4 addr flush label "eth*"

The following syntax will be used to start or stop operating an interface

ip link set dev {DEVICE} {up|down}

For example to stop eth1

ip link set dev eth1 down
and to start it

ip link set dev eth1 up

you want to explicitly change the size of the txqueuelen transmission queue for a device, you can use:

ip link set txqueuelen {NUMBER} dev {DEVICE}

For example if I want to change the value of txqueuelen from 1000 to 10000 for eth0:

ip link set txqueuelen 10000 dev eth0
ip a list eth0

You can change including maximum transmission units or MTU

ip link set mtu {NUMBER} dev {DEVICE}
For example to change MTU for eth0 to 9000:

ip link set mtu 9000 dev eth0
and we verify that it has changed with:

ip a list eth0
to display cache neighbour/arp:
ip n show

ip neigh show

The last field of the result will show the status of the process of detection of the neighbor's availability for each entry and can be:

STALE - when the neighbor is functional but probably there is no path to it so the kernel will try to verify this on the first transmission.

DELAY – a package has been sent to a neighbor in STALE and the kernel is waiting to receive confirmation.

REACHABLE – the neighbour as well as communicating with him are functional.

To add a new ARP entry, use:

ip neigh add {IP-HERE} lladdr {MAC/LLADDRESS} dev {DEVICE} nud {STATE}

And an example where a permanent ARP will be added to neighbor 192.168.1.5 using the eth0 interface:

ip neigh add 192.168.1.5 lladdr 00:1a:30:38:a8:00 dev eth0 nud perm

In this are used a series of terms whose meaning is found in the table below.

Stare neighbour (nud)	Meaning
permanent	Neighbour entry is considered by default to be permanently valid and can only be removed by the administrator
noarp	The neighbour's entry is valid. No attempt will be made to verify it.
stale	The neighbour's entrance is valid, but there are suspicions. Using this option to ip neighbor will not change the status of the neighbor if it has been valid and the command does not contain any instructions to change the address.
reachable	The neighbour entry is valid until the preset time expires.

Delete an ARP entry

The following syntax can be used to delete or invalidate an ARP entry for neighbor 192.168.1.5 on the

eth1 device:ip neigh del {IPAddress} dev {DEVICE}

ip neigh del 192.168.1.5 dev eth1

To change the condition of neighbor 192.168.1.100 on the interface eth1

ip neigh chg 192.168.1.100 dev eth1 nud reachable

Clearing an ARP entry

Below is given the syntax for conditional lysing tables for neighbor/arp tables

ip -s -s n f {IPAddress}
one example

ip -s -s n f 192.168.1.5

or

ip -s -s n flush 192.168.1.5

This command can also be used to manage the routing table including handling the routing board at the kernel level. To display the board we will use:

ip r
ip r list
ip route list
ip r list [options]
ip route
list the route to 192.168.1.0/24:
ip r list 192.168.1.0/24
add new route syntax

ip route add {NETWORK/MASK} via {GATEWAYIP}
ip route add {NETWORK/MASK} dev {DEVICE}
ip route add default {NETWORK/MASK} dev {DEVICE}
ip route add default {NETWORK/MASK} via {GATEWAYIP}
And an example of adding a path to network 192.168.1.0/24 using gateway 192.168.1.254

ip route add 192.168.1.0/24 via 192.168.1.254
To direct all traffic through the next gateway using the eth0 interface

ip route add 192.168.1.0/24 dev eth0

Delete a route

If I want to delete a gateway

ip route del default
This example will delete the path created in the previous example

ip route del 192.168.1.0/24 dev eth0
If you want to change the MAC address for the network card called NIC
NIC="eno1" ## <-- My Name ##
ip link show $NIC
ip link set dev $NIC down
establishing a new MAC address
ip link set dev $NIC address XX:YY:ZZ:AA:BB:CC
ip link set dev $NIC up

Because in many of the examples on the Internet the ifconfig command is often used below, there is a table of examples of use using the ip command.

Old command (obsolete)	It’s equivalent nowadays
ifconfig -a	ip a
ifconfig enp6s0 down	ip link set enp6s0 down
ifconfig enp6s0 up	ip link set enp6s0 up
ifconfig enp6s0 192.168.2.24	ip addr add 192.168.2.24/24 dev enp6s0
ifconfig enp6s0 netmask 255.255.255.0	ip addr add 192.168.1.1/24 dev enp6s0
ifconfig enp6s0 mtu 9000	ip link set enp6s0 mtu 9000
ifconfig enp6s0:0 192.168.2.25	ip addr add 192.168.2.25/24 dev enp6s0
netstat	ss
netstat -tulpn	ss -tulpn
netstat -neopa	ss -neopa
netstat -g	ip maddr
route	ip r
route add -net 192.168.2.0 netmask 255.255.255.0 dev enp6s0	ip route add 192.168.2.0/24 dev enp6s0
route add default gw 192.168.2.254	ip route add default via 192.168.2.254
arp -a	ip neigh
arp -v	ip -s neigh
arp -s 192.168.2.33 1:2:3:4:5:6	ip neigh add 192.168.3.33 lladdr 1:2:3:4:5:6 dev enp6s0
arp -i enp6s0 -d 192.168.2.254	ip neigh del 192.168.2.254 dev wlp7s0

c) NSlookup The main purpose of this command is to help solve specific DNS problems. It can be used in two ways: interactive and parameter. For the interactive entry, the keyboard command will be launched and it will present a prompter. In the other case we will have for example:

nslookup mail.yahoo.com

Which will extract additional information (using DNS reverse)

If you also want more specific information

nslookup qureytype=mx mail.yahoo.com

Obviously it needs to be tested on a less protected server because yahoo will not respond to such a command. If you do not want all the information about the node you can try using the following

HINFO parameters to find out the type of processor and the operating system installed

UNIFO gives us the name of the user

MB gives the domain name of the e-mail server

MG gives us the name of a group member of the e-mail server

MX gives the name of the e-mail server

Traceroute - allows us to display information about the nodes that pass from the point of origin to an address namely:

traceroute mail.yahoo.com

And if you want to avoid DNS resolution

traceroute -d mail.yahoo.com

d) Netstat - we are often interested to know detailed information about all communications initiated by the current station with the outside world. For this, it was entered unstated. Obviously you can use a grep in case we want more specific information Related to the information displayed 0.0.0.0 refers to a kind of generic address.

Test the command

netstat -sp 0.0.0.0

Or give an ip to see the local address of the station and give for example

Netstat -sp 192.168.0.4

If we are also to identify the process that communicates, we will use

Netstat -nao

That will show the pid

For those who already knows how to: play with nmap.

A tcpdump can provide interesting information/

Laboratory job

- Test each command with different parameter combinations (when possible).

Homework:

Do a full analysis of the subnet in the home (from the point of view of each of you) - extract the structure like a graph with the stations detected active and for each station make a card on those discovered from its analysis. Don't forget from your subnet that you're having problems with the node. Those who do not stay in the campus disconnect the connection with the isp and do analysis on computers on the local network (mom, dad, desktop etc.)