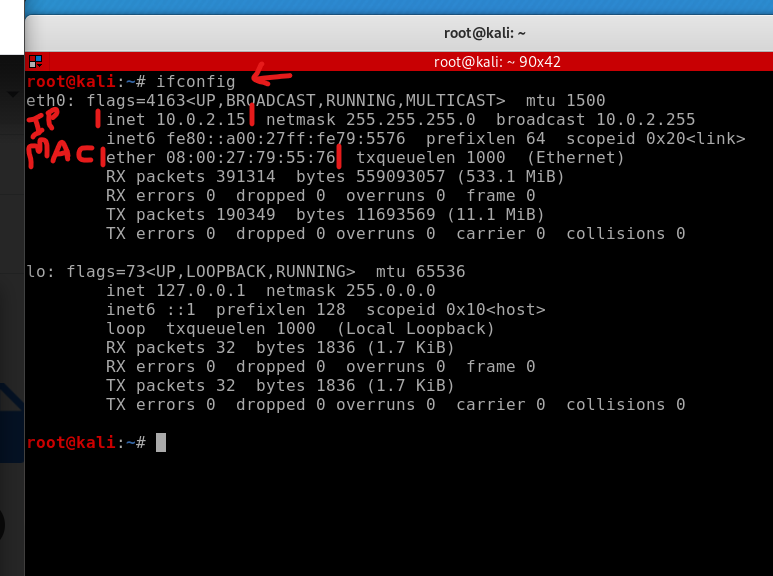
MAC address and how to change it…

Command “ifconfig”

This is used to display all the network interfaces connected to the computer either by virtual network or by the wireless adapter.



In this pic

inet <number> => this number represents the IP address of that network

ether <number> => this number represents the mac address of that system or computer.

Command:

ifconfig <connection\_name like above pics eth0> down

this command disables the mentioned network interface

**remember that the options of the network interface can be changed only if that particular interface is down or currently not working**.

Changing the mac address of the system:

ifconfig <connectionName> hw ether <newMACAddress>

conditions:

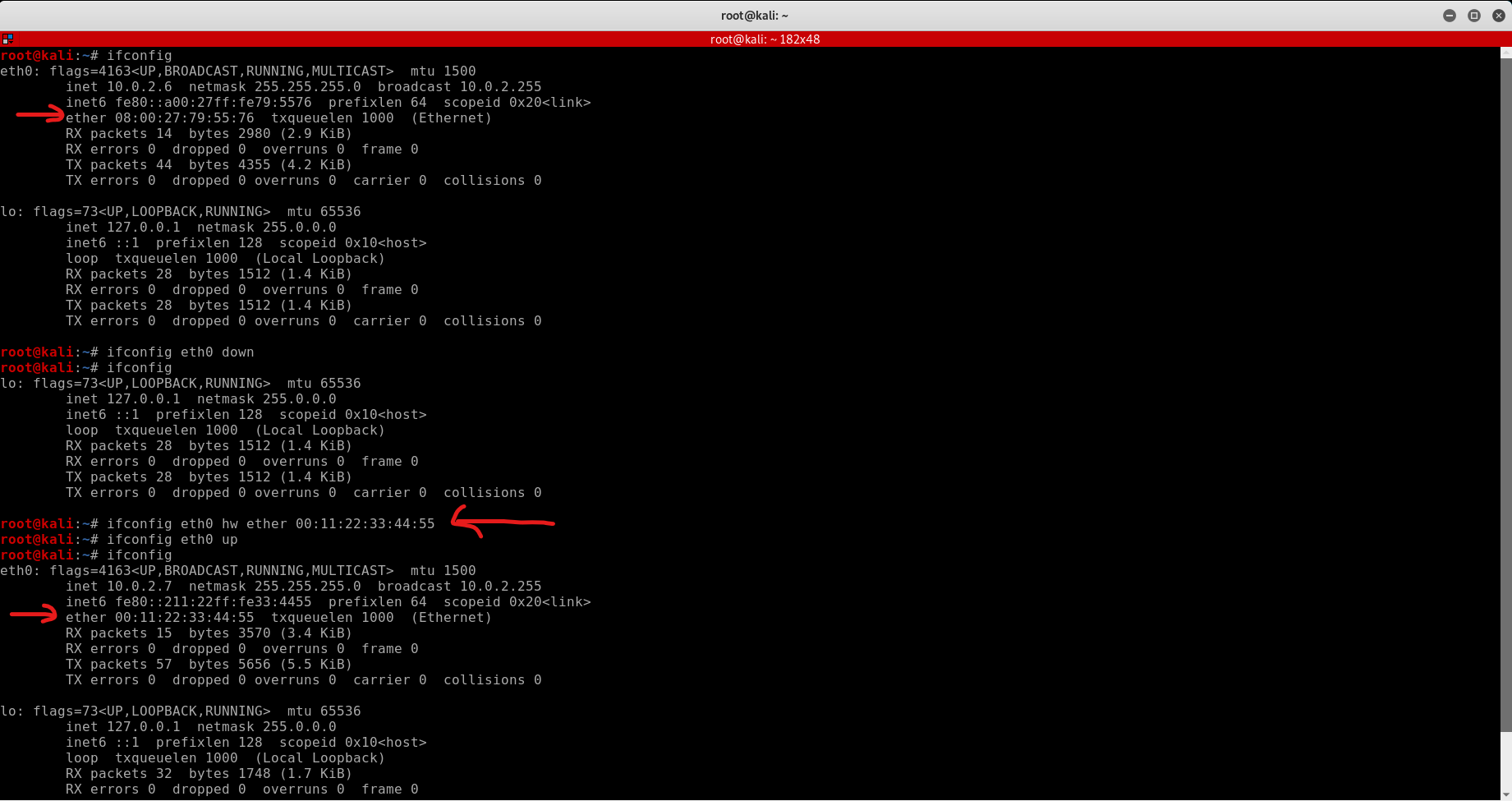
1. New mac address must contain 12 characters
2. Each two characters should be separated by semicolon(:)
3. Example MAC address: 1s:3d:fc:af:d6:78
4. **Also note that the characters in the MAC address can be either numbers from 0-9 and letters from a-f**
5. [**https://en.wikipedia.org/wiki/MAC\_address**](https://en.wikipedia.org/wiki/MAC_address)

Now after changing the mac address we have to enable the network configuration to show up with the changed mac address

Command:

ifconfig <connectionName> up

this commands enables the network interface that we actually disabled during the starting of this process with the command “ifconfig <connectionName> down”



***NOTE:***

***Make sure that no other commands are running in the background while changing the MAC address.***

*Changing MAC address using Python*

NOTE: anywhere if i say to compile the python i mean to run the python script, python is a interpreter language which is different from cpp, c, c# which are need to be compiled before running the script

more about the difference in [here](https://www.geeksforgeeks.org/difference-between-compiler-and-interpreter/)

Python modules used in this program are

1.subprocess

2.optparse

in this python code our target is to execute the terminal or command shell commands via python

for this python has an inbuilt module named subprocess

<https://docs.python.org/3/library/subprocess.html>

in this module we use call (or) run method

How it works:

subprocess.call(“<command>”, shell=True)

Breaking the line of code:

subprocess is a python module, this module contains call method,

this call method takes two arguments:

1. command (that we execute in command prompt or terminal) as string
2. shell (boolean value either True (or) False)

This shell argument is to say to python that the command that i typed in string is the command that you need to execute in the terminal (if given the argument shell=True) and it automatically takes care of it.

Updates about call method:

this call method is in Python version 2 and the link that i attached above is the latest one (Python version 3)

the latest version of Python uses run method instead of call(we can use call also as our wish)

Example of the call method:

suppose that i need to list out all the wireless interfaces that are connected to my computer using python, then i write and compile the code

import subprocess

subprocess.call(“iwconfig”, shell=True)

thats it these two of code in python does the job for you,

Now all you have to do is to run this python file in the terminal or whatever you are using to run the python file.

now the code in python to change the mac address goes like this

import subprocess

interface = input(“interface: “)

newMACaddress = input(“new mac address: “)

subprocess.call("ifconfig {} down".format(interface), shell=True)

subprocess.call("ifconfig {} hw ether {}".format(interface,new\_mac\_address), shell=True)

subprocess.call("ifconfig {} up".format(interface), shell=True)

subprocess.call(“ifconfig”, shell=True)

run the python script which is just these colored text and give the input of interface and new mac address and 💥💥BOOM💥💥

our MAC address has been changed successfully..

Catch about this process:

works very well, we have changed the MAC address successfully, what's the catch about this??

well, here it is,

in linux or ubuntu, we can run multiple commands in a single line, with the commands being separated by semicolon(;)

Eg: ifconfig; ls; cd Desktop;

this command lists out the network interfaces, lists the folders or files in the current working directory, and finally changes you working directory to Desktop(inorder)

this means that the person who is trying to change the MAC address along with that he can also perform other tasks (security gone😭)

So to control this activity here comes the second type of call method..

How it works:

subprocess.call([“<command>”, “<options>”, “<arguments>”])

Breaking the lines of code:

we know how call methods works, so let's dive into this wired syntax,

the whole command that we need to run is enclosed in a square brackets and the command is divided in all the short strings whenever there is space in the command like

Eg: subprocess.call([“ls”, “-l”])

this commands lists all the folders and files in the current working directory along with the hidden files.

So, how exactly it works

in the string that appears inside square brackets, the first string is the actual command name (like ls, ifconfig, aircrack-ng, etc..)

and the rest of the strings are the options that are available for that particular command their values

So if anyone tries to insert other commands in this syntax  
 obviously a particular command cannot have another command as an option, so if we try to execute some other commands in between it gives us the error.

try this command for experience:

we have to list all the folders and files in the working directory along with the network interfaces,

import subprocess

subprocess.call([“ifconfig”, “;”, “ls”])

this gives you an error

