# **Device Drivers** **Exercise-8**

-CED17I034,

Duvvuru Jaswanth reddy.

# Introduction

SNULL can be described as how the network interfaces fit in with the rest of the Linux kernel and provide examples in the form of a memory-based modularized network interface, which is called (you guessed it) snull. To simplify the discussion, the interface uses the Ethernet hardware protocol and transmits IP packets. The knowledge you acquire from examining snull can be readily applied to protocols other than IP, and writing a non-Ethernet driver is different only in tiny details related to the actual network protocol.

SNULL is NOT a Loopback Interface, SNULL is similar to loopback interface but sends the same packet back to the sender as though it received from a different outside network.

# How SNULL is designed

* The snull module creates two interfaces.
* These interfaces are different from a simple loopback, in that whatever you transmit through one of the interfaces loops back to the other one, not to itself.
* It looks like you have two external links, but your computer is replying to itself.
* Unfortunately, this effect can’t be accomplished through IP number assignments alone, because the kernel wouldn’t send out a packet through interface A that was directed to its own interface B.
* Instead, it would use the loopback channel without passing through snull. To be able to establish communication through the snull interfaces, the source and destination addresses need to be modified during data transmission.
* In other words, packets sent through one of the interfaces should be received by the other, but the receiver of the outgoing packet shouldn’t be recognized as the localhost.
* The same applies to the source address of received packets.
* To achieve this kind of “hidden loopback,” the snull interface toggles the least significant bit of the third octet of both the source and destination addresses; that is, it changes both the network number and the host number of IP numbers.

i.e., Change TO address from 192.168.**0**.2 to 192.168.**1**.2

* The net effect is that packets sent to network A (connected to sn0, the first interface) appear on the sn1 interface as packets belonging to network B.
* This is done to test the interface, ability to change the header, test whether the packets are sent and received properly, and how much time it took to perform these actions.

# Steps involved in SNULL

* Add the necessary Header files.
* A Network device driver has to decide the size of the packet and the name of the device file say eth101. For each of the device files, a private structure is created in the program.
* Create another device file say eth101 and create a private structure in the Network device driver program.
* The devices have to be opened/closed using interrupt requests.
* Configure the device code i.e., change the IP address, change the network speed, etc.,
* Send the packet, create IP/TCP headers and change the least significant bit of the third octet of both the source and destination addresses.
* IO control is used for collecting statistics such as ping, how many packets of data is sent or received, delay of packets, etc.,
* Rebuild the L2 headers.
* Fix MTU (max transition unit – 1500 bytes)
* Finally, init and cleanup of the functions.