Device Drivers Lab - 8
Short overview on
SNULL

SNULL (Simple Network Utility for Loading Localities) works by creating 2 interfaces to help the messages being transmitted from one interface loop and back to the other interface or in other words *what we send to one interface, it returns to the other interface*.

It is a network device driver that does not talk to the "actual" devices.

It is not exactly a loopback interface, but it simulates conversations with real remote hosts in order to better demonstrate the task of writing a network driver. It stimulates actual operations and communications with actual servers.

The Linux loopback driver is simple; it can be found in drivers/net/loopback.c
To be able to establish a communication through the SNULL interface 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 recognised as the local host.

Packages sent to the network A:

- connected to sn0,
- first interface,

Should appear as:

- packages of the network B,
- connected to sn1,

- Second interface

The same applies to the source address of received packets. SNULL toggles the least significant bit of the 3rd octet of the IP(a part of network id) of the source and destination. Changing the source is important otherwise when the packet comes back, the user would be able to observe the sender as himself / herself, but the most important part is changing the destination, otherwise we wouldn’t even receive the packet.

Example:

Let the IP[3] - represent the 32 bits.

IP - “IP[0] . IP[1] . IP[2] . IP[3]”
IP[0] - part 1, IP[1] - part 2, IP[2] - part 3, IP[3] - part 4.

```
Snull() {

// Some initializations

IP[2] ^= 1;

// Rebuild checksum
// Packet ready to be sent

}
```

After the packet is sent, the operation is done by swapping the last bit of the 3rd octet of IP using an XOR operator.

And thus for the user, a packet seems to magically come from some other network so that he can test other codes which takes care of incoming packets from other networks.

Aditya Gopakumar
CED17I036