# Step #6

Read from misc device

For **Tuesday 19/04 23h59**

# Read

- In mode 1 and 2, you will add packets inside a per-device buffer
- Implement the read function of your misc device
- When read is called, you will give the complete content (skb->mac_header instead of skb->data, length = skb->len + skb->mac_len)* of the oldest (FIFO) packet received in the buffer provided by the user-space program. Only one packet will be copied per call. If the user buffer is too small, you will copy only that amount of the packet and trash the end of the packet data.
- The read function will return the amount of bytes copied from the packet.
- When passing from mode 1 to 2 and vice-versa, packets in the buffer are not lost (the buffer is not emptied).
- You can implement read in a blocking mode : if there is no packet, a context switch will happen until a packet is available. Having a blocking mode will be a bonus.
- Pay attention to packet cloning, skbuff usage counter, and locking ! Remember that in mode 2, the packet will be yours and only yours, but not in mode 1. Read about skb_clone and friends.

*The last point is really the difficult part of this step !*

# Session handling

- User cannot put two devices in fastnet mode using the same file descriptor. If he does, the ioctl will return the new error code 3* : “Multiple ioctl about different devices”, this will be checked just after the error code 2 (the check for a valid mode). It is only allowed if the last device was putted back in mode 0 before putting a new one in mode 1 or 2.
  - Eg : If the user wants to receive packet from two devices using fastnet, he will have to make two open() call to fastnet, and issue one ioctl per file descriptor.
- Note that each userlevel open() call on fastnet will create a new file descriptor (indexed by the return value of the open() function). In the kernel you receive directly the file pointer in the open/ioctl/read/… functions until the release call. In fact, the kernel use the file descriptor number you give him to retrieve the right file pointer. Therefore using the private pointer of the file structure can be done to remember things like the attached device with the ioctl for the whole session.
- Calling read directly after open (without an ioctl) should therefore make it return always 0, as the session is attached to no interface.

**As this is a late requirement, it won’t be marked*

# Free & Release

- When the file is closed, the device stays in its current mode, and the device's buffer continues to receive packets
  - When an ioctl is done for the same device, the session/file reattach itself to the device and recover the existing list of packets in the buffer for subsequent reads
- Do not forget to free all packets
  - When we go back in mode 0 for a specific device, even when we are in a different file session
  - When the module is unloaded

# Concurrency

- The rx_handler can interrupt your read, so you must protect the buffer
- Accessing different devices with different file descriptor is legal
- Two process/threads cannot access the same device, you do not have to worry about this case (I won't test/correct that, IRL it should be protected). This is the case where two process read() packets from the same device.

# Usage example

```
unsigned char buf[256];
int fd = open(“/dev/fastnet”);
read(fd,buf,256); //Incorrect, no ioctl has been done. From which devices should I read?
struct fastnet_register rgr = {.ifname=”eth0”,.mode=1};
ioctl(fd,1,&rgr);
read(fd,buf,256);
rgr.mode=2;
ioctl(fd,1,&rgr);
read(fd,buf,256);
rgr.ifname=”eth1”;
ioctl(fd,1,&rgr); //Error 3, only one device per open call
close(fd);
sleep(10);
fd = open(“/dev/fastnet”);
rgr.ifname=”eth0”;
ioctl(fd,1,&rgr);
read(fd,buf,256); //Should read packets received during the sleep(10);
Int fd2 = open(“/dev/fastnet”);
rgr.ifname=”eth1”;
ioctl(fd2,1,&rgr); //Legal, because this is another fd created by another open.
read(fd2,buf,256); //Read only packets from eth1
close(fd);
close(fd2);
```

# Report

- **Follow the same rules than for previous steps (comments, patch, report, submission, obligations, …)**
- Explain what you did, where, how why, how you finded it.
- The problems you had
- How you solved them