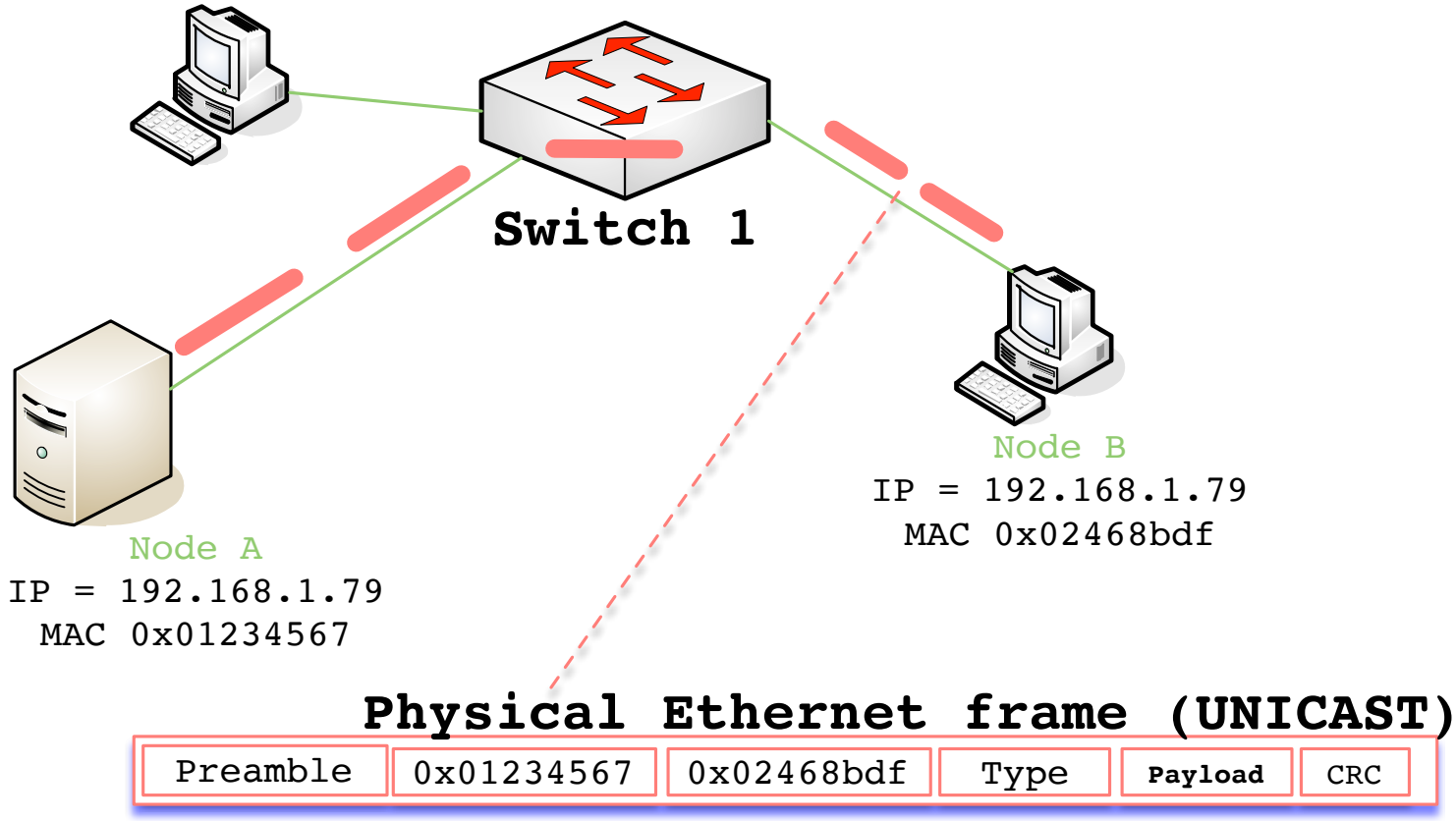
Libpcap intro practice

Brief

V 1.1

Based on textbook *Conceptual Computer Networks* by:

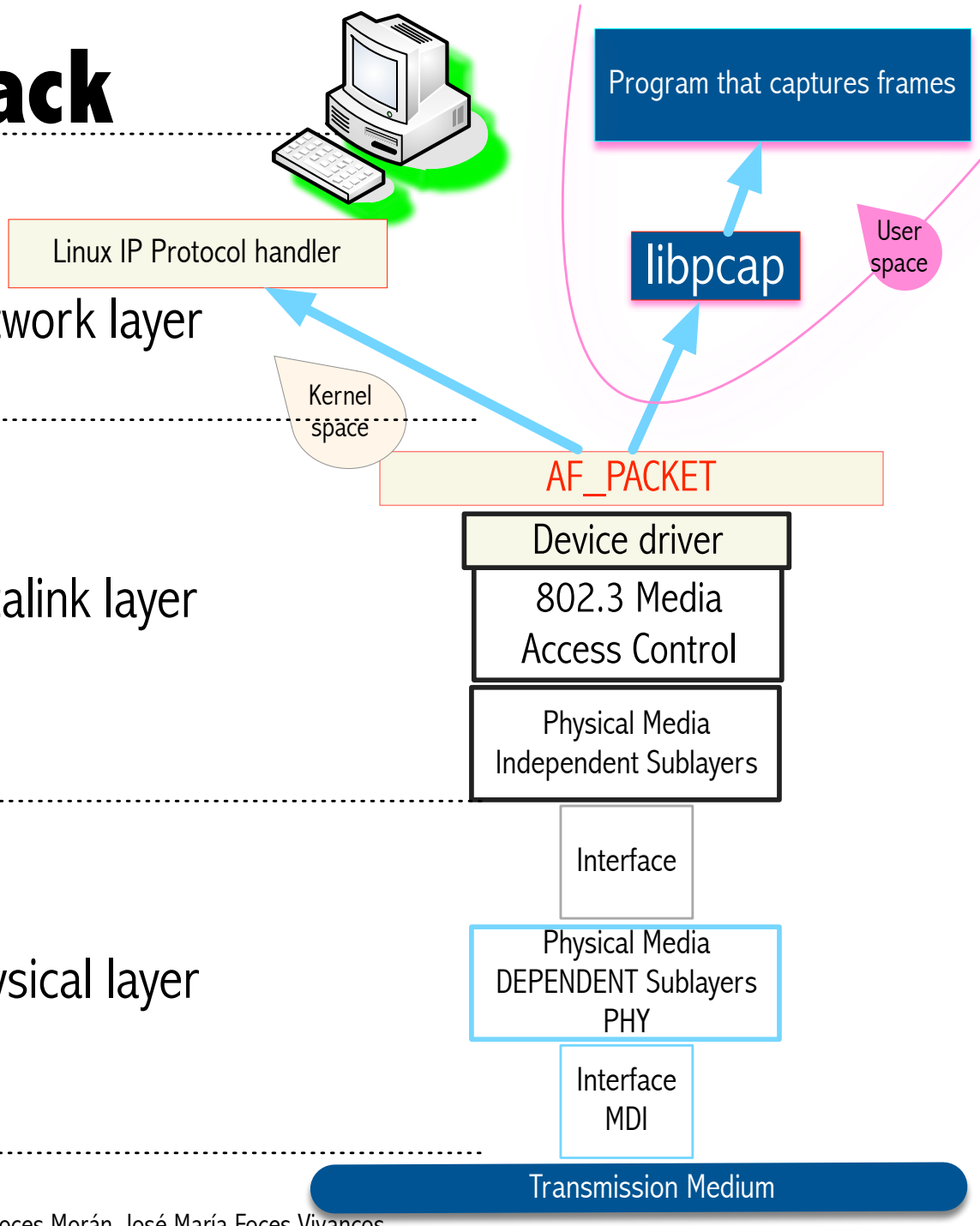
Ethernet frame



De-encapsulated Ethernet frame (UNICAST)

0x01234567	0x02468bdf	Type	Payload
Byte 0-5	6-11	12-13	14…

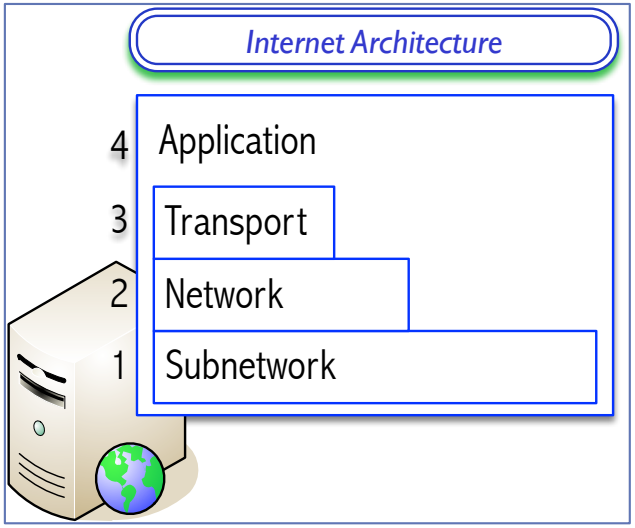
Protocol stack

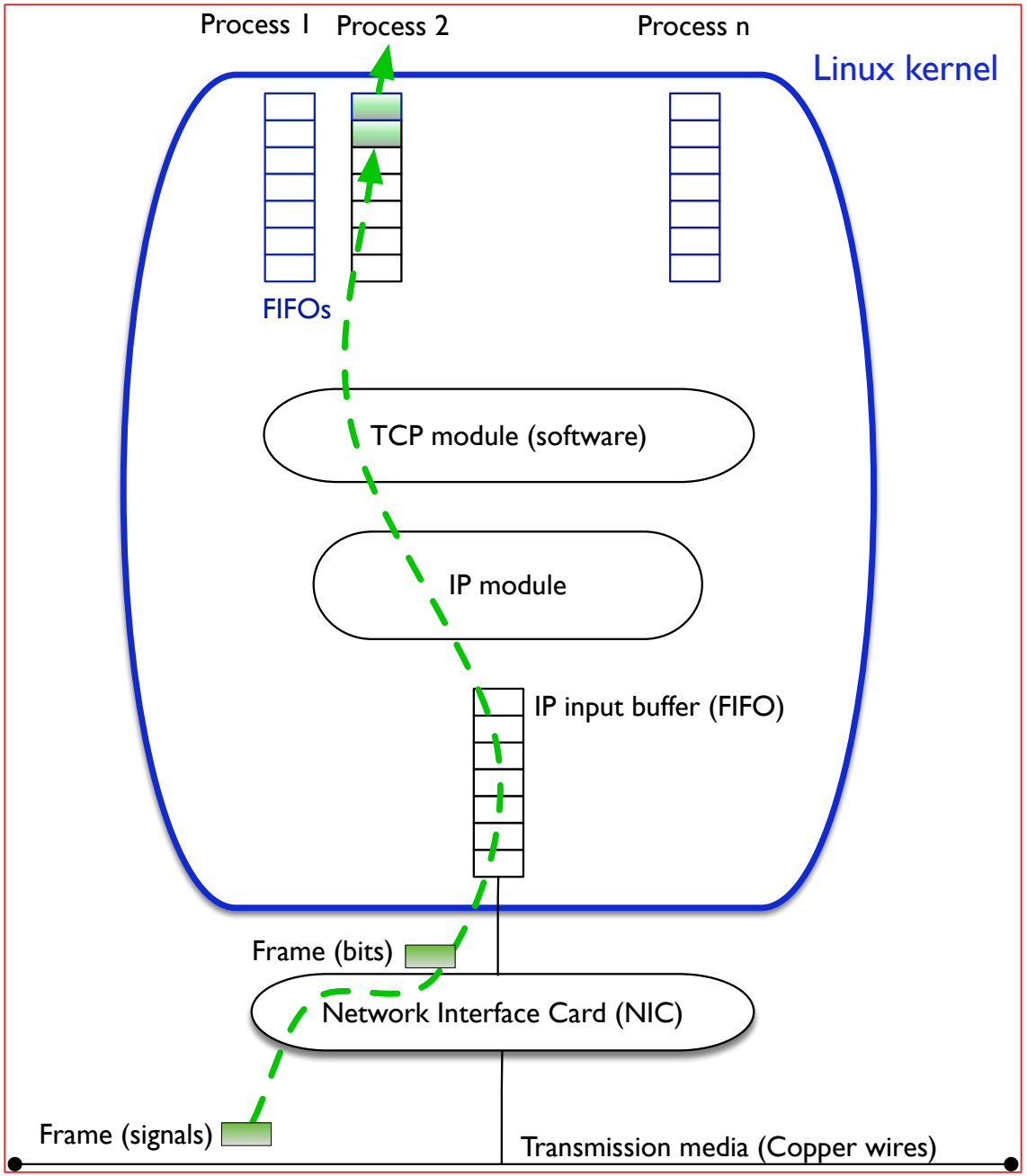


Flow of Libpcap calls

- 1. It seeks a capturable network device in your system: **pcap_lookupdev()**
- 2. It prints the network device's IP and Netmask by invoking the program's **printIPandMask()** function which invokes libpcap's:
  - **pcap_lookupnet()**
  - **inet_ntoa()**: Turn an inet address into an ASCII, printable string
  - `$ man ntohs  (Also htons)`
- 3. It opens the network device (Usually, it will be eth0): **pcap_open_live():** This call returns a pointer to an instance of a pcap_t type that will be used in the ensuing pcap call: **pcap_loop()**
- 4. The function call to **pcap_loop()** specifies the program's function getNewFrame() as the callback function that will be called by pcap whenever a new datalink frame passes through the interface (It is transmitted or received).

Implementation of protocols





Libpcap functions for Bridge

- **pcap_next_ex()**
  - read a packet
- **pcap_setdirection()**
  - PCAP_D_INOUT
  - PCAP_D_IN
  - PCAP_D_OUT
- **pcap_inject, pcap_sendpacket**
  - transmit a packet

Libpcap functions for Bridge

- **pcap_setnonblock()**

  pcap_getnonblock()

  - puts a capture handle into *non-blocking* mode

- **pcap_dispatch()**

  - will, if no packets are currently available to be **read**, return 0 immediately **rather than blocking** waiting for packets to arrive

- **pcap_loop()** and **pcap_next()** will not work in *non-blocking* mode

Based on textbook *Conceptual Computer Networks* by:

End of practice on libpcap