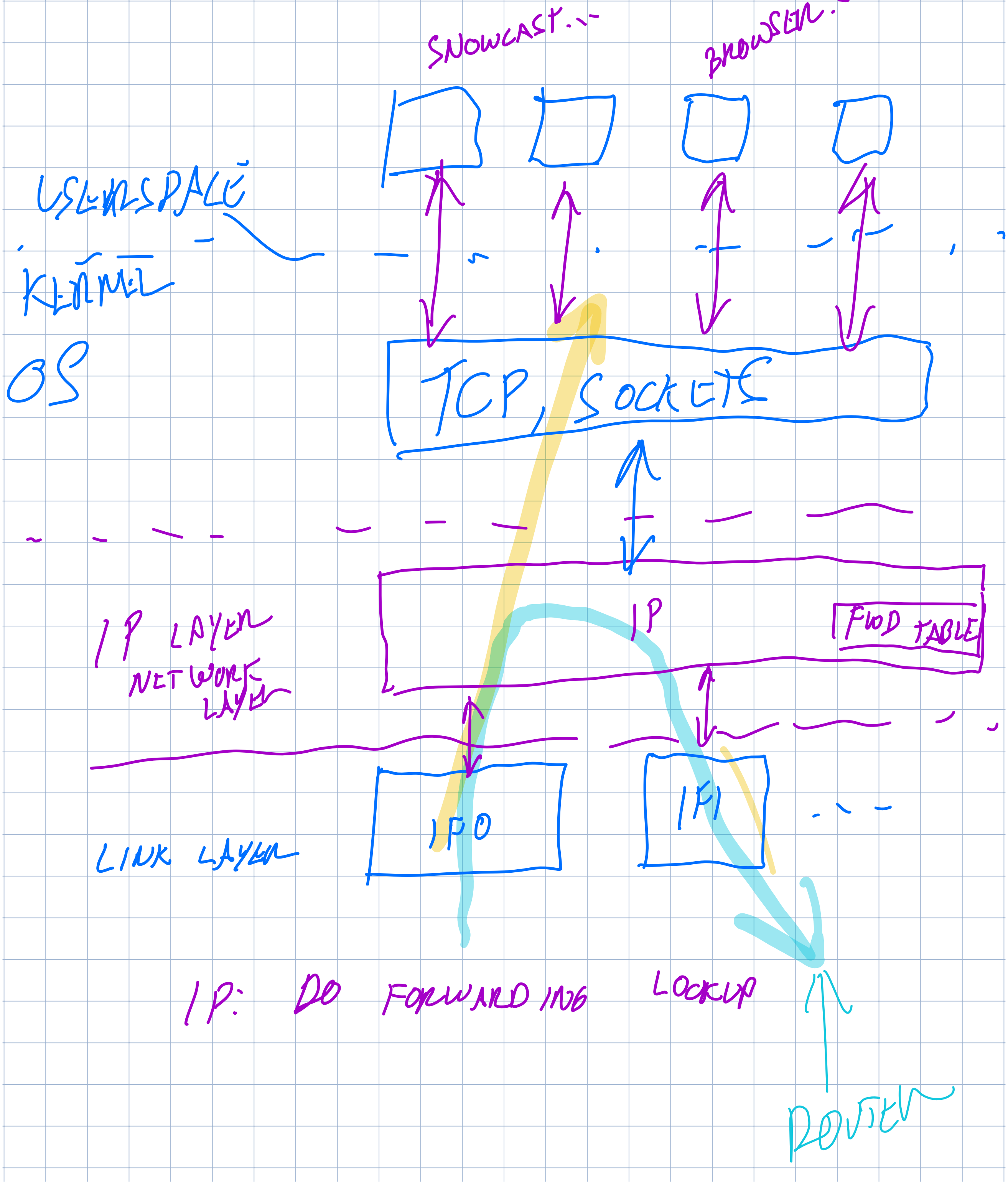
# IP gearup notes

This PDF contains my whiteboard notes from the gearup—for full details on how we used these, see the video.  I've also added some diagrams from lecture 7 which talk about the same concepts, which are relevant here.

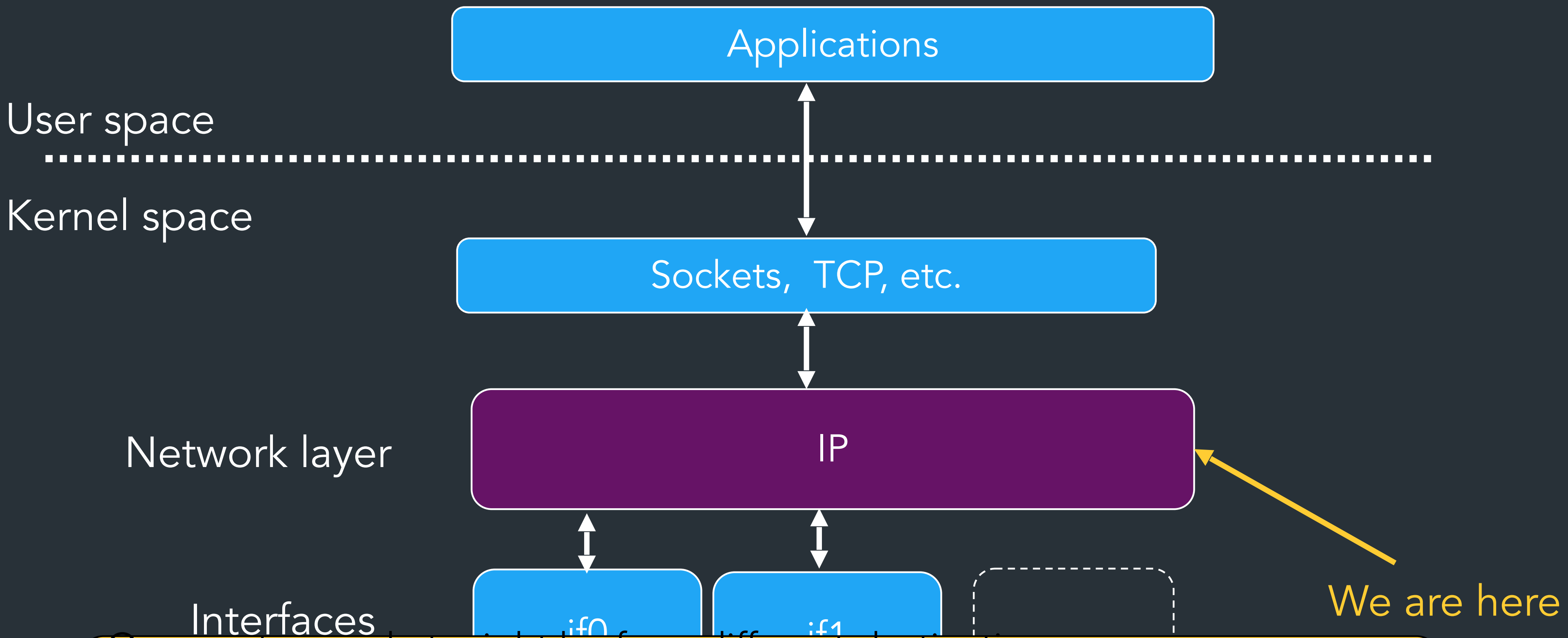
If you are reading this and have not watched lecture 7 already, I HIGHLY recommend doing so before starting this project, as it sets up the conceptual background you need to think about IP forwarding and networking stacks.

# THINKING ABOUT A NETWORKING STACK.

(FROM LECTURE – CONCEPTS APPLY HERE)



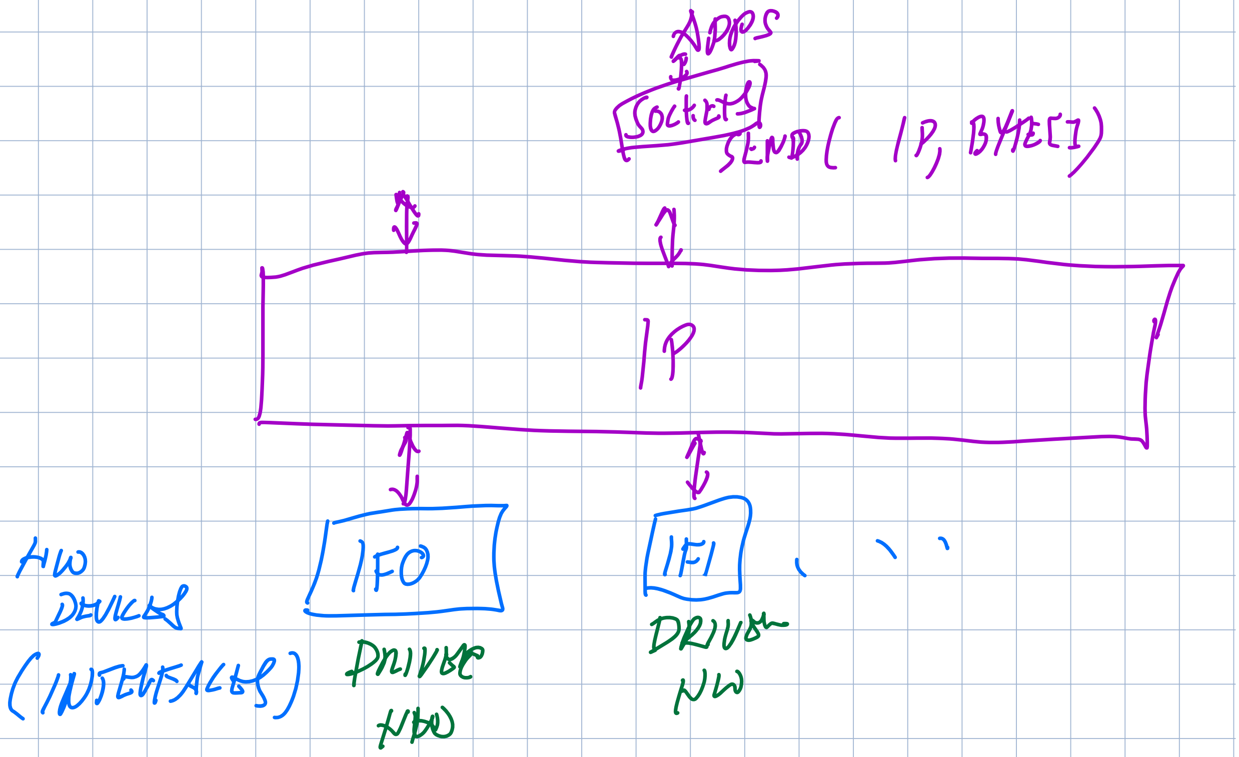
# A "networking stack"



On a router: packet might be for a different destination

=> send out another interface

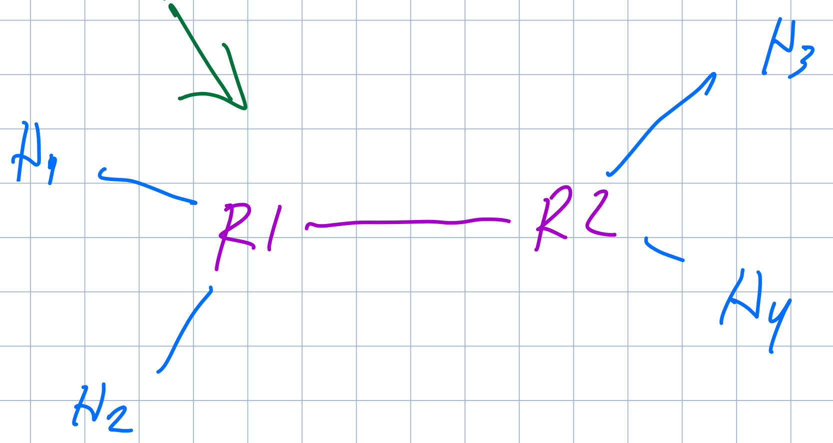
If it's for one for an IP assigned to this device, send to higher layer


APPS
SOCKETS
SEND( IP, BYTE[])
IP
IF0
IF1
HW DEVICES (INTERFACES)
DRIVER HW
DRIVER HW

NODE = "HOST OR ROUTER"

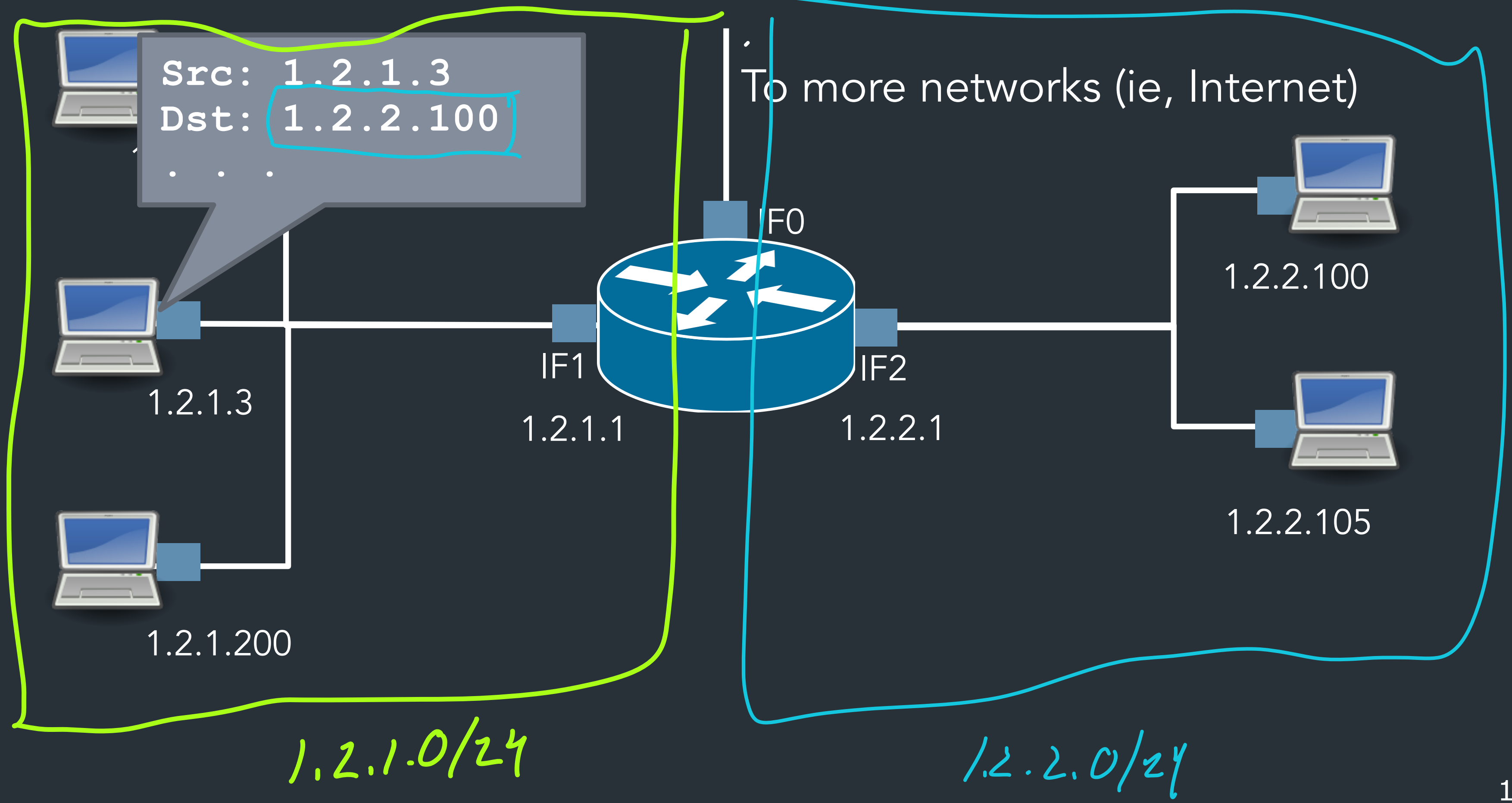
OUR VIRTUAL NETWORK

- HOSTS
- ROUTERS } TWO PROGRAMS THAT USE UDP



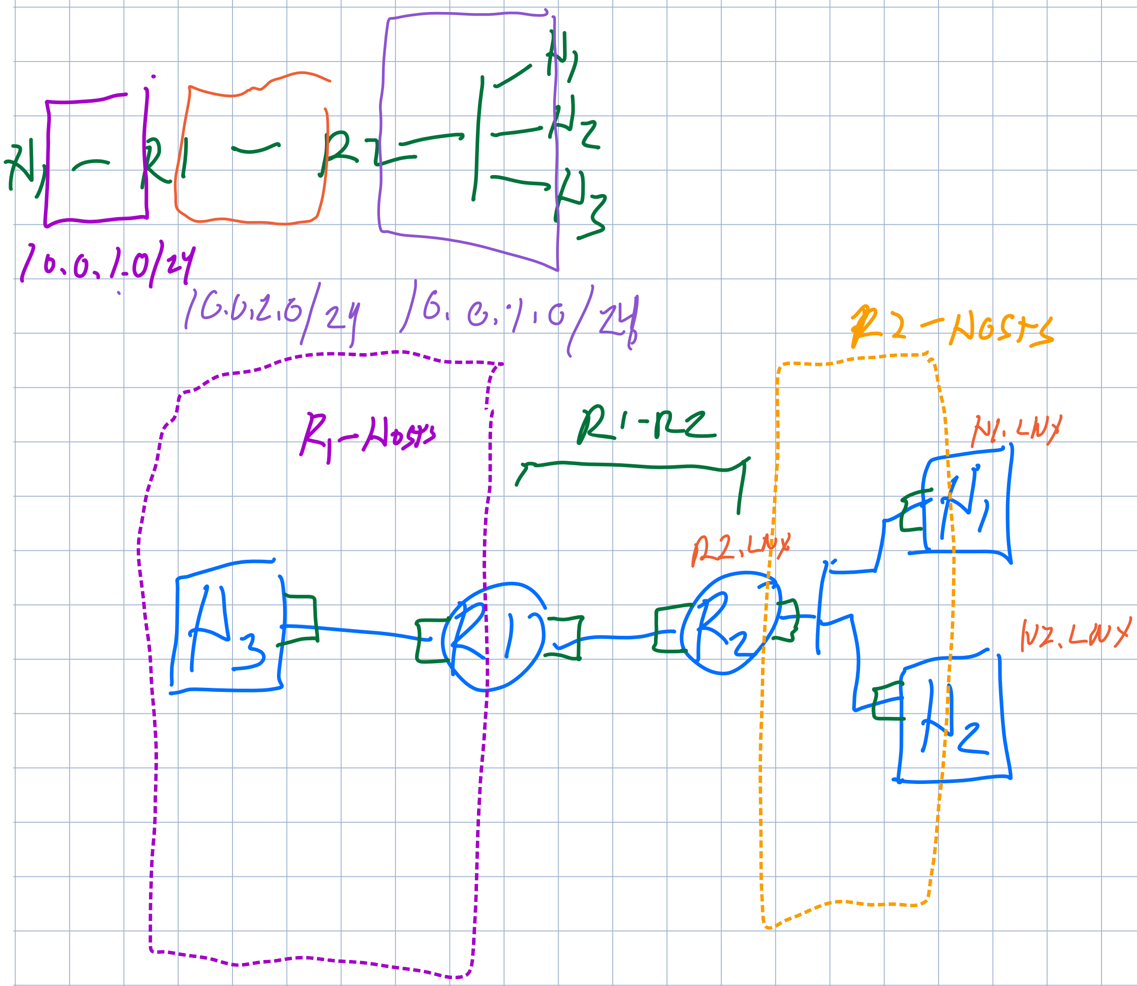
~~A~~ TOPOLOGY: HOW NODES ARE CONNECTED

# Forwarding IP packets



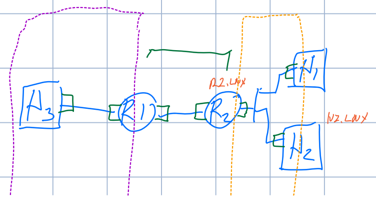
TWO CONFIG FILES DEFINE HOW NET IS SET UP:

- Network definition file:  (some-net.json)
  - How stuff is connected (adjacency list)



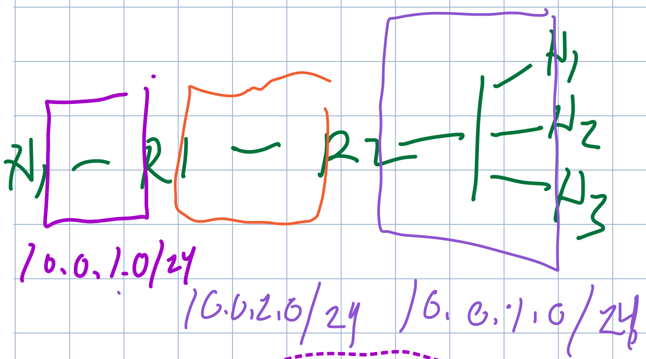
Every node in the network starts up with a configuration file that tells it how to set up its interfaces
 => lnx file
(We give you a parser)

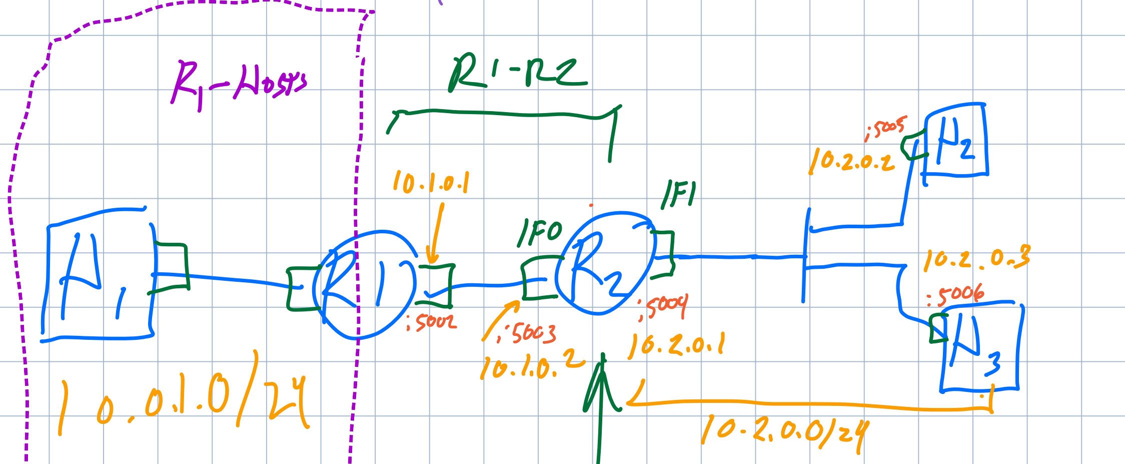
For every interface
- IP address
- Netmask (prefix size)
- Know how to reach neighbor nodes



TWO CONFIG FILES DEFINE HOW NET IS SET UP:

- Network definition file: (some-net.json)
    - How stuff is connected (adjacency list)





10.0.1.0/24 VIA 10.1.0.1

10.2.0.0/24 IF1

10.1.0.0/24 IF0

Every node in the network starts up with a configuration file that tells it how to set up its interfaces
=> lnx file
(We give you a parser)

For every interface
- IP address
- Netmask (prefix size)
- Know how to reach neighbor nodes

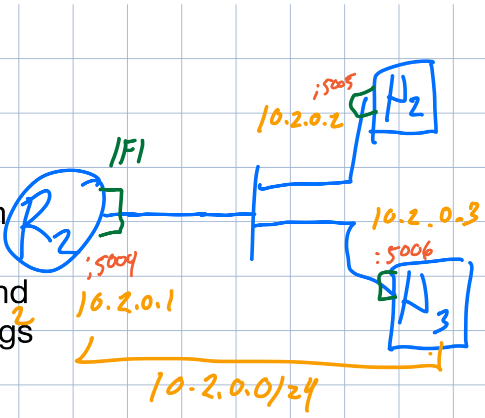
→ UDP PORT ✓

One lnx file defines what that node knows about at startup. A node always knows:
- Your own IP on each interface
- For each interface, which neighbors you can reach

How do the hosts networks work?

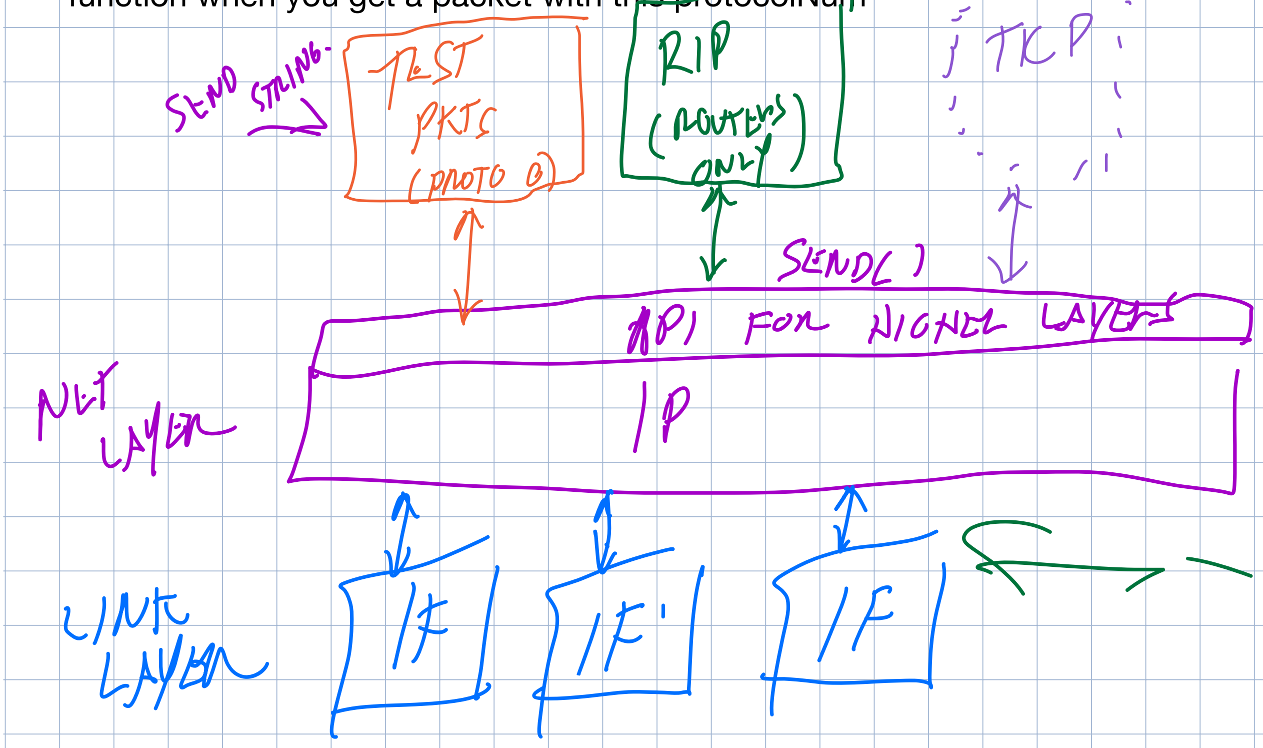
- All nodes can always communicate with each other (unless we take down links)
- For networks with hosts, you can pretend that there's a switch that connects all things in that subnet
- Hosts will only ever have one router in their subnet
- Hosts do not do RIP, they only have one (pre-configured route) that sends traffic to their one router

Routers will always be connected to each other in "point to point" networks (r1-r2, r2-r3, …)



Should be thinking about what kinds of functions you want to expose to higher layer stuff

- Initialize(config structure from lnx file)
- Send(dest ip, uint8 protocolNum, byte array)
- RegisterRecvHandler(uint8 protocolNum, callbackFunc) // Call this function when you get a packet with this protocolNum



Two implementation notes

1. Most languages have types of ip addresses, they have good methods and stuff you can use
   - For go, the type you want is netip.Addr (net.IP :( )
     - We have provided a library that uses this because Go hasn't caught up yet
   -

=> Totally okay to use libraries that have data structures for IPs and such

2. Talking about encapsulation
   = > You are sending UDP packets on UDP sockets

The thing you send IS AN IP PACKET