Need to decide on the number of threads needed + shared data.

What are the different parts of your IP stack and what data structures do they use? How do these parts interact (API functions, channels, shared data, etc.)?

Vrouter:

- Multiple interfaces (UDP sockets)
- Forwarding table to route packets between subnets

Vhost:

- One interface (one UDP socket), send IP packets to hosts or routers.
- Two hosts can communicate directly with each other in a subnet
- Routers can communicate with each other between subnets. Within a subnet, two hosts know each other's MAC addresses.

What fields in the IP packet are read to determine how to forward a packet?

- Source IP
- Destination IP
- TTL (time to live)
- Checksum to verify validity of a packet

We'll need to be able to parse, and construct an IP packet.

What will you do with a packet destined for local delivery (ie, destination IP == your node's IP)?

- Not quite sure, maybe send it to the router, then the router sends it back??
    - Send to router, router sends back.
    - **Ask: ask mentor**

What structures will you use to store routing/forwarding information?

- Prefix trie for efficient subnet matching
- Maybe just a list or hashmap that maps IP prefix to the next hop destination interface. Sorted by prefix length?

What happens when a link is disabled? (ie, how is forwarding affected)?

- Router should update its own forwarding table, and re-route traffic that was using the disabled link.
- If there is no possible alternative route, the packet is dropped? Maybe TTL = 0?

    - **Ask how to drop a packet**

Threading structure:

- 1 thread per node?
- 1 thread per interface
    - Maybe threads need to share information about packets with each other.
- 2 threads per interface
    - Read/write thread
    - Read thread takes in UDP message, sends over channel to proper write thread, should be able to read and write packets at the same time?