Mininet 3+4 and Wireshark
Running GUI Apps through Mininet VM

Can you SSH into your VM?

- Yes, I can SSH into my Mininet VM
  - Install and run X11 Server
  - Mac users:
    - XQuartz: [https://www.xquartz.org/](https://www.xquartz.org/)
  - Windows users:
    - Use WSL, Putty, or other X11 compatible SSH client (not ssh from Windows 10 cmd.ex)
    - VcXsrv: [https://sourceforge.net/projects/vcxsrv/](https://sourceforge.net/projects/vcxsrv/)
  - Issues?
    - See here: [https://github.com/mininet/mininet/wiki/FAQ#x11-forwarding](https://github.com/mininet/mininet/wiki/FAQ#x11-forwarding)

- No, I am using the VirtualBox console or the above didn’t work
  - [https://github.com/mininet/mininet/wiki/FAQ#can-i-run-a-qui11-application-within-a-mininet-host](https://github.com/mininet/mininet/wiki/FAQ#can-i-run-a-qui11-application-within-a-mininet-host)
  - Follow those instructions to run Wireshark inside the VirtualBox window.
Software Defined Networking

SDN splits the Control and Data planes to allow programmatic control of networks

- **Data Plane**
  - Responsible for moving data from one part to another - the ‘flows’ in a switch
  - Needs to be very fast and low latency

- **Control Plane**
  - Responsible for deciding where data goes
  - It controls the Data plane
Why do we want or care about SDN?

Hopefully this picture sums it up
OpenFlow

SDN splits the Control and Data planes to allow programmatic control of networks.

OpenFlow is an standard SDN protocol.

The protocol can be and is implemented in real hardware.

- Used in datacenter switches
- Alternative to using Vendor-specific configuration tools such as Cisco IOS CLI

It can also be implemented in software-based virtual networks.

- Mininet
What is Mininet?

Software that creates a virtual network using process-based abstraction

Each process runs in its own virtual network namespace

- Has its own virtual network hardware as well as IP’s and MAC addresses

These virtual networks are used extensively in the cloud as well as for container networking (Docker!)
SDN in Mininet

Data Plane:

- OpenVSwitch ([https://www.openvswitch.org/](https://www.openvswitch.org/))
  - This is an OpenFlow compatible Virtual Switch
  - It is designed to enable massive network automation through programmatic extension, while still supporting standard management interfaces and protocols

Control Plane

- OpenFlow Controller which you implement in Mininet 2, 3, and 4.
- In our case we are using the Python-based Pox OpenFlow Library.
  - There are plenty of OpenFlow libraries in other languages
You created a virtual network topology containing multiple hosts and a switch using Mininet.
You implemented a basic Firewall using Pox-based Openflow controller

Your controller, on startup, installed data-flow rules (ofp_flow_mod) into the Switch to:

- Flood ARP packets
- Flood ICMP packets
- Drop all other (IPv4) packets
You implement a controller for a more complex virtual network topology

There are now multiple switches connected to a central router (cores21)

The switches - s1, s2, s3, and dcs31 - should be very simple and can just flood traffic

The router, core21, cannot flood all ports and should pick the destination port based off the destination subnet.

In mininet cli:

```plaintext
*** Starting CLI: mininet>net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
```

You will need to use the specific individual switch port numbers (not OFPP_FLOOD) in your rules

```python
of.ofp_action_output(port = PORTNUM)
```
Mininet 4

Similar topology to Mininet 3, but more intelligent, real routing.

Mininet 3 ‘brute-forces’ the network by forwarding packets between switches and routers.

In Mininet 4, you implement an actual router which the hosts talk to as a ‘gateway’:
Mininet 4

The Mininet 4 Gateway needs to do the following:

1. Proxy ARP messages for destinations that are outside the local subnet by responding with the gateway's MAC address (arbitrarily chosen by you in OpenFlow controller software).
   a. ARP Request messages should be sent to OpenFlow controller’s Handle_PacketIn method
   b. The OpenFlow controller should then generate an ARP reply telling the host to use the gateway’s address for packets destined to that subnet
   c. Now when the host tries to reach that subnet, it will direct its messages to the gateway’s MAC address.

2. Learn host IP’s from the received ARP messages/broadcasts
   a. Hosts broadcast ‘whohas’ ARP requests containing the host’s own IP address and MAC address
   b. The gateway learns the IP’s from requests and replies to them with its own MAC.
   c. It can install rules that match based on ‘destination address’ here
   d. Note that communication across subnets will not work until the switch has learned both hosts or subnets
   e. i.e. H10 <-> H20 will not work until both H10 and H20 both individually try to reach each other, teaching the gateway their IP address and port.
The Mininet 4 Gateway needs to do the following:

3. Route IP packets across subnets
   - Will need to change the Source and Destination MAC addresses
   - Incoming IP packets to gateway
     - Source MAC is the original host
     - Destination MAC is the gateway
   - Outgoing IP packets from gateway
     - Source MAC is the gateway
     - Destination MAC is the original destination (part of the ofp_flow_mod rule actions)
     - Hint: ofp_flow_mod actions: of.ofp_action_dl_addr
Wireshark General Demo

Links:
https://courses.cs.washington.edu/courses/cse461/20wi/section-slides/pcap-demo.md
https://courses.cs.washington.edu/courses/cse461/20wi/section-slides/461-demo.pcap

Commands:
wget https://courses.cs.washington.edu/courses/cse461/20wi/section-slides/461-demo.pcap
wireshark 461-demo.pcap