NEW YORK INSTITUTE OF TECHNOLOGY

INCS 775
Data Center Security
Lab1 – Summer 2025

Open vSwitch and Mininet

Dr. Zakaria Alomari
zalomari@nyit.edu

Open vSwitch

- An OpenFlow enabled virtual switch that runs on commodity Linux machines
  - *kernel module* forwards the packet (data plane)
  - *userspace module* talks to the controller
  - A *remote controller* can control an OVS instance (control plane)
  - For further details see:
    - Pfaff, Ben, *et al.* "The Design and Implementation of Open vSwitch." *NSDI* '15.
- **`ovs-vsctl`** → create/manage bridges
- **`ovs-ofctl`** → create/manage forwarding rules
- But we need a network first !!

Mininet

- *De facto* emulator for SDN
- Uses **Open vSwitch (ovs)** to create SDN *switches*
- Uses **network namespaces** to create hosts in their own network namespace
- Can emulate a whole network in one single machine (even on a Raspberry pi)

Mininet

- Switch
  - Switches are instances of ovs bridge
  - A bridge is an L2 forwarding device that can forward traffic based on MAC address
  - In OpenFlow the semantics of a switch is extended beyond just L2 forwarding. It can forward traffic based on the supported match fields
- Hosts
  - Hosts are processes isolated inside network namespaces
  - A network namespace gives a process it's own view of network interface and routing tables
  - Processes in different network namespaces do not share their network interface and routing table

Mininet Installation

- Install mininet

  -
    ```
    sudo apt-get install mininet
    ```

- Show mininet options

  -
    ```
    mn -h
    ```

Start Mininet

- Starting without any parameter creates a single switch topology with two hosts connected with it and opens mininet console
  - ○ `sudo mn`
- To view information about hosts and network use the following commands
  - ○ `nodes, net, dump`

Mininet Hosts

- Hosts are processes running in their own network namespace, *i.e.*, hosts are processes with their own network configuration
- Run a command inside some host
  - h# command
    - `h1 ifconfig`
    - `h1 ping -c 2 h2`

Mininet

- Open terminal to a host
  - xterm h#
    - e.g., `xterm h1`
- Test network connectivity
  - `pingall`
- Run an iperf between random pair of hosts
  - `iperf`
- Set link bandwidth and delays
  - `sudo mn --topo=single --link=tc,bw=10,delay=5ms`

More Mininet

- Python interpreter from Mininet terminal
  - `py …`
- Show the list of available methods in a host object
  - `py dir(h1)`
- Show the IP address of a host
  - `py h1.IP()`
- Set cpu usage limit for the hosts
  - `sudo mn --topo=linear,3 --host=cfs,cpu=0.1`

Mininet Built-in Topologies

- Linear topology with 3 switches

  -
    ```
    sudo mn --topo=linear,3 --switch ovsk
    ```
- Tree topology with depth 2

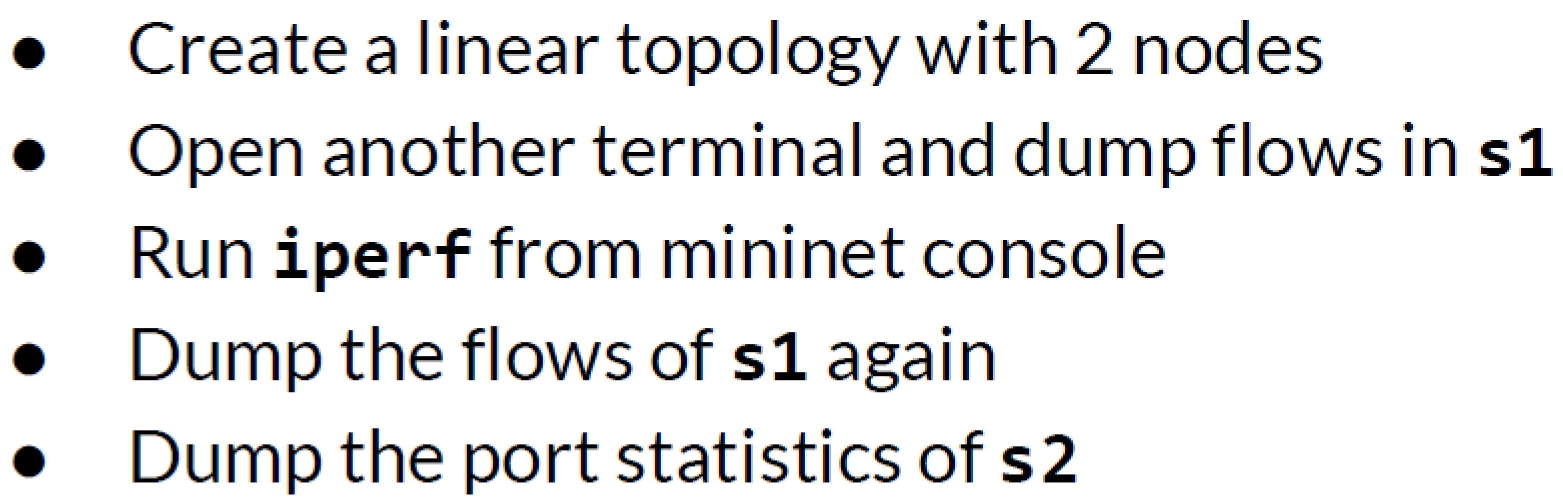
  -
    ```
    sudo mn --topo=tree,depth=2,fanout=2 --switch ovsk
    ```
- Topology with a single switch

  -
    ```
    sudo mn --topo=single --switch ovsk
    ```

Working with OVS

- Show details of switch s1
  - `sudo ovs-ofctl show s1`
- Show the flow rules in switch s1
  - `sudo ovs-ofctl dump-flows s1`
- Show port statistics in switch s1
  - `sudo ovs-ofctl dump-ports s1`
- Add a flow forwarding rule in switch s1
  - `sudo ovs-ofctl add-flow s1 <flow_spec>`
- More commands!
  - http://www.pica8.com/document/v2.3/html/ovs-commands-reference/

Quick Exercise



- Create a linear topology with 2 nodes
- Open another terminal and dump flows in **`s1`**
- Run **`iperf`** from mininet console
- Dump the flows of **`s1`** again
- Dump the port statistics of **`s2`**

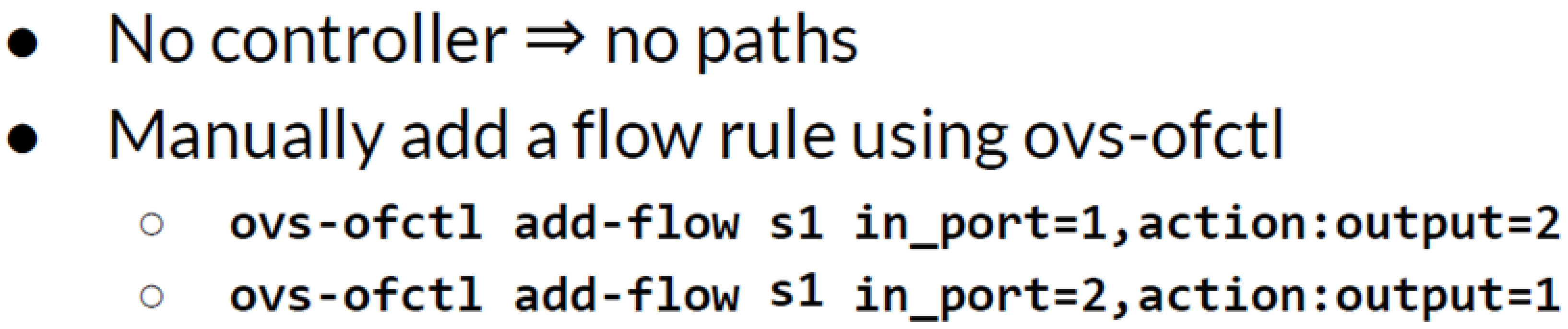
Mininet with Remote Controller

-
  ```
  sudo mn --topo=single --controller=remote,ip=127.0.0.1,port=6653
  ```
- Try to ping h2 from h1 (will not work if a remote controller is not running)

  -
    ```
    h1 ping h2
    ```
- Start floodlight from the floodlight directory

  -
    ```
    sudo java -jar target/floodlight.jar
    ```
- Floodlight’s web UI is accessible at http://localhost:8080/ui/index.html
- [Optional] Floodlight’s REST API manual: https://floodlight.atlassian.net/wiki/display/floodlightcontroller/Floodlight+REST+API

Manually Adding Flow Rules



- No controller ⇒ no paths
- Manually add a flow rule using ovs-ofctl
  - ○ `ovs-ofctl add-flow s1 in_port=1,action:output=2`
  - ○ `ovs-ofctl add-flow s1 in_port=2,action:output=1`