NEW YORK INSTITUTE OF TECHNOLOGY

INCS 775 – Data Center Security

Summer 2025

Dr. Zakaria Alomari

Assignment - 1

Total points: 100

Due date: Wenesday, *11 June 2025 / 11:59 PM*

Important: On Wednesday, the last hour of class will be dedicated to a Lab session. Nevertheless, we will also cover the following content:

- Oracle VirtualBox or/and VMware Workstation Player installation
- Python
- User is not in the sudoers file.
- Mininet Installation
- Installing iPerf3 on Ubuntu
- Start Mininet
- Mininet Hosts
- Mininet
- Mininet Built-in Topologies
- Working with OVS

Important Submission Guidelines:

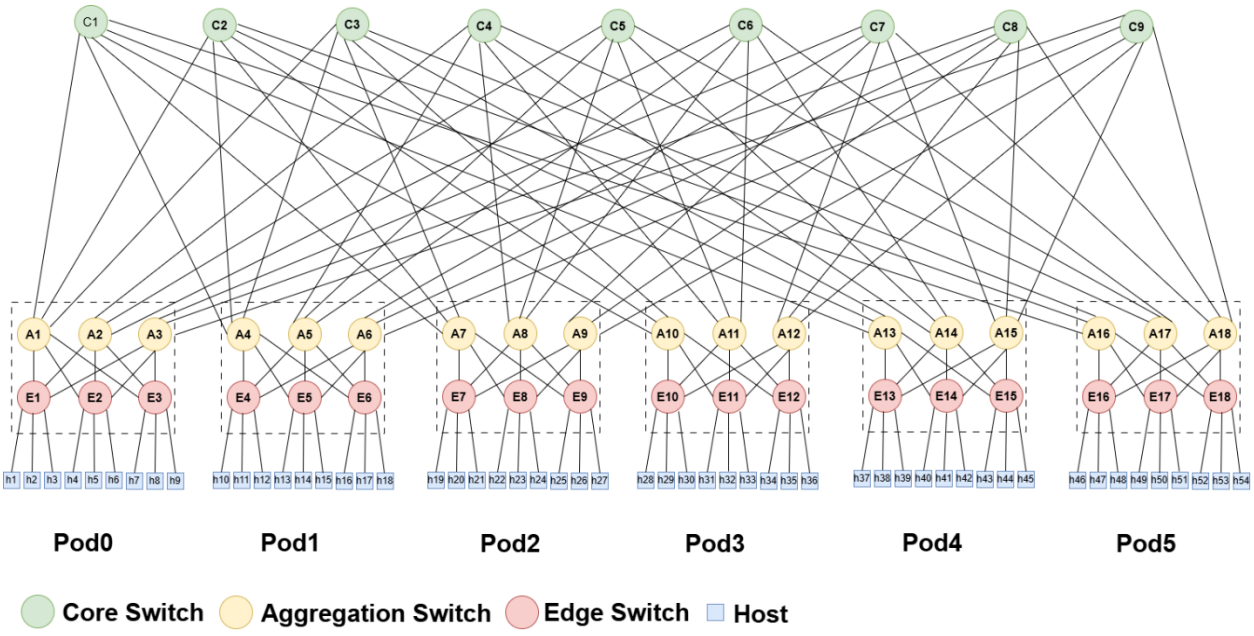
- A **10% penalty** will be applied **for each day** a submission is late.
- **Incorrect or inappropriate submissions** will **not be accepted under any circumstances**.
- **Resubmissions are not allowed**, regardless of the reason.
- Please **review the assignment instructions carefully** before submitting to ensure accuracy and completeness.
- If any part of the assignment is unclear, you are strongly advised to **contact the instructor immediately** for clarification.
- **Submitting the wrong file is solely the student's responsibility** and will not be excused or granted another submission opportunity.

Background:

The Fat-Tree topology, depicted in the Figure below, consists of k pods (k=6), each of which consisting of $k/2$ edge switches and $k/2$ aggregation switches.

Edge and aggregation switches connected as a clos topology and form a complete bipartite in each pod. Also each pod is connected to all core switches forming another bipartite graph.

Fat-Tree built with k-port identical switches in all layers of the topology and this topology supports $k^3/4$ hosts. With Fat-Tree topology issues with oversubscription, costly aggregation and core switches, fault tolerance, and scalability are resolved. Fat-Tree established a solid topology for researchers to work onto solve other important issues such as agility through virtualization.

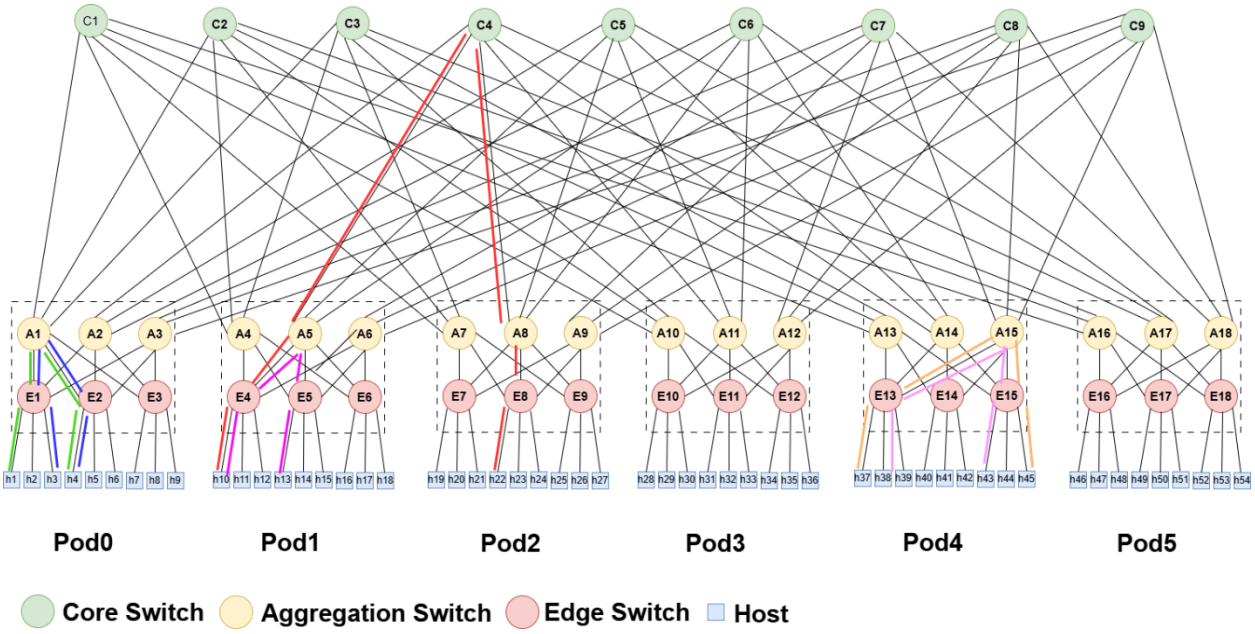


Link bandwidth = 12 *Mbps*

Link Delay = 2 *ms*

- Establish the Data Center topology demonstrated above and employ a Python script to construct a Fat Tree topology using Mininet. Do not add any controller for path setup.

- Use ovs-ofctl to create six bidirectional paths, i.e., the green path between **h1** and **h4**, the blue path between **h3** and **h4**, the red path between **h10** and **h22**, the purple path between **h10** and **h13**, the orange path between h37 and h45, and the pink path between h38 and h43. (No other pair of hosts should be able to communicate)



- After setting up the path run the following from **Mininet** console:
  - iperf h1 – h4
  - iperf h3 – h4
  - iperf h10 – h22
  - iperf h10 – h13
  - iperf h37 – h45
  - iperf h38 – h43

- Then run:
    - h1 ping h4
    - h3 ping h4
    - h10 ping h22
    - h10 ping h13
    - h37 ping h45
    - h38 ping h43

Tips/Resources

- Run mininet with **`--arp`** option to statically populate arp table in the hosts.
- A reference for example **`ovs-ofctl add-flow`**:
    - https://docs.pica8.com/pages/viewpage.action?pageId=3086345
- Set an **`idle_timeout`** of **0** so that flows do not expire.

- The username is not listed in the sudoers file. Please refer to the link provided below for further instructions:
  https://www.youtube.com/watch?v=ERh74y-3EW8

- Installing iPerf3 on Ubuntu. Please refer to the link provided below for further instructions:
  https://chrisjhart.com/Install-iperf3-on-Ubuntu-22.04/

- Learn how to manage Mininet and customize its topology. An example is provided to demonstrate how to create and execute a custom topology in Mininet. Please refer to the links provided below for further instructions:
  https://mininet.org/walkthrough/#custom-topologiesLinks to an external site.
  https://mininet.org/walkthrough/#custom-topologiesLinks to an external site.

What to submit?

- Put the following files inside a compressed folder named **<lastname_firstname.zip>** (Only one member of the group is required to submit the assignment)

- Create a text file called **Group_info** and fill it with the **names**, **student IDs, and email** of each group member.

- **Custom_FatTree_6Pods.py** - - script containing the code to construct the Fat Tree topology 6 pods using Mininet. (28)

- Files containing the **flow rules** using **ovs-ofctl** for the switches: **E1, A1, E2, E4, A5, E5, C4, A8, E8, E13, A15, E15. (Generate a separate file for each switch or router containing its respective flow rules; avoid consolidating all flow rules into a single file)** (30)

- Files created by the following commands (**after path setup**)
  - ovs-ofctl dump-flows E1 & > E1_dump (2)
  - ovs-ofctl dump-flows A1 & > A1_dump (2)
  - ovs-ofctl dump-flows E2 & > E2_dump (2)
  - ovs-ofctl dump-flows E4 & > E4_dump (2)
  - ovs-ofctl dump-flows A5 & > A5_dump (2)
  - ovs-ofctl dump-flows E5 & > E5_dump (2)
  - ovs-ofctl dump-flows C4 & > C4_dump (2)
  - ovs-ofctl dump-flows A8 & > A8_dump (2)
  - ovs-ofctl dump-flows E8 & > E8_dump (2)
  - ovs-ofctl dump-flows E13 & > E13_dump (2)
  - ovs-ofctl dump-flows A15 & > A15_dump (2)
  - ovs-ofctl dump-flows E15 & > E5_dump (2)

- Output of iperf commands (9)
    - Filename: iperf.out
    - One line for each iperf output in the following format
        - < host_id >-- < host_id >:<reported_bw>

      <h1> <h4>

      < h3> <h4>

      <h10> <h22>

      < h10> <13>

      < h37> < h45>

      < h38> <h43>

- Average of the first 20 reported round trip times from ping output (9)
    - Filename: latency.out
    - One line containing the average round trip time between each pair:

      <h1> <h4>

      < h3> <h4>

      <h10> <h22>

      < h10> <13>

      < h37> < h45>

      < h38> <h43>