Computer Networks
@CS.NCTU

Lab. 2: Network Topology with Mininet

Location: EC-114

Instructor: 陸勇盛 (David Lu)

Agenda

- Objectives
- Overview
- Tasks
- Submission
- Grading Policy
- References

Objectives

In this lab, we are going to write a Python program which can generate a network topology for Mininet and use iPerf to measure the bandwidth of a path in this topology

1. Learn how to create a network topology for Mininet
2. Learn how to measure the bandwidth by using iPerf in Mininet

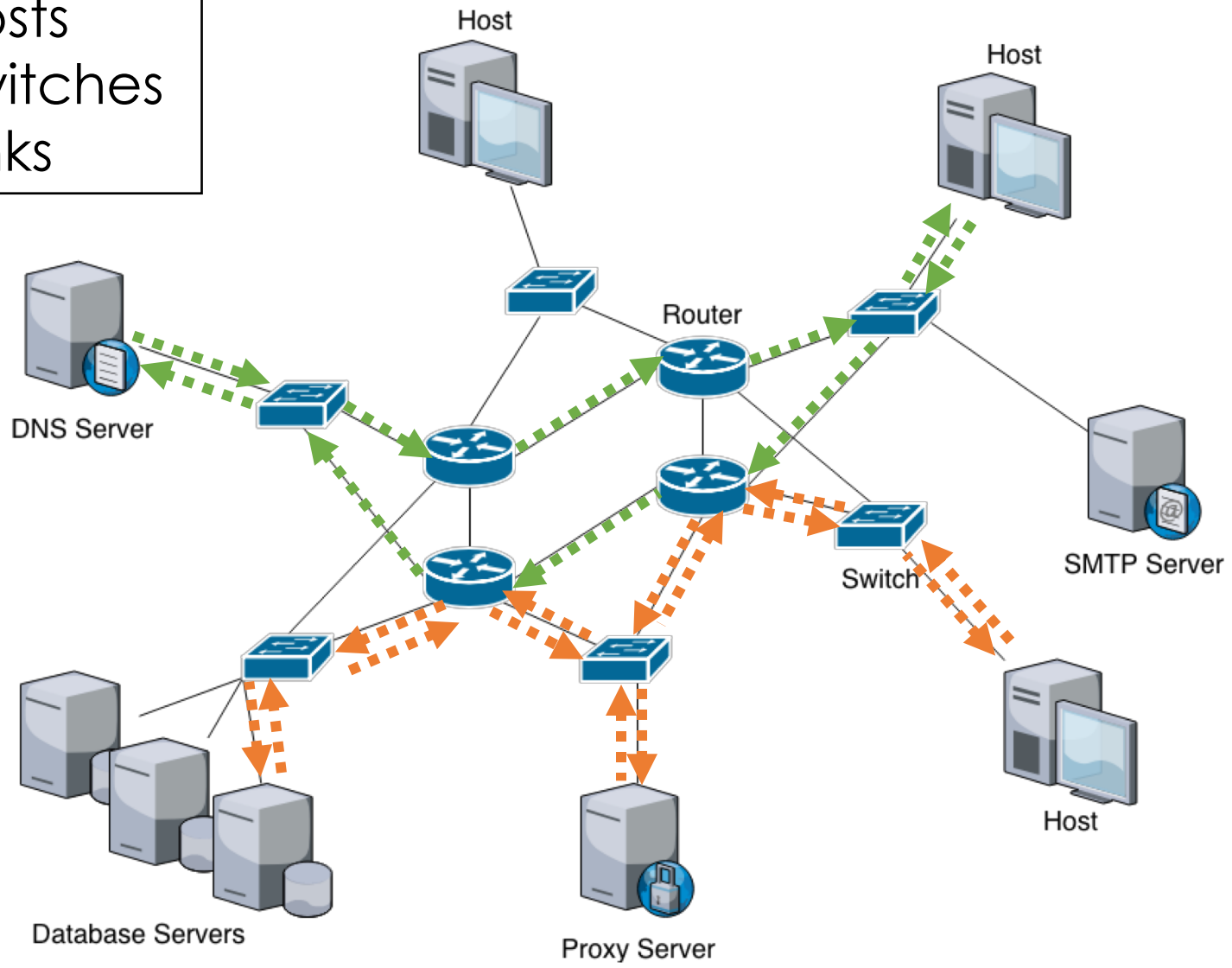
TODO

1. We will give you a Python code (example.py) that includes an example network topology of Mininet
2. We will get you a figure illustrating a new topology you should generate
3. Copy the necessary function code from example.py and write your Python code (topology.py) to generate this topology

Overview

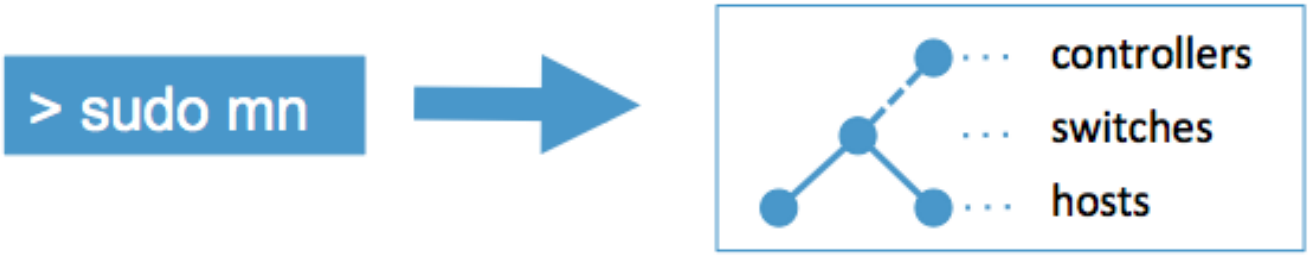
What is a Network Topology?

- Hosts
- Switches
- Links

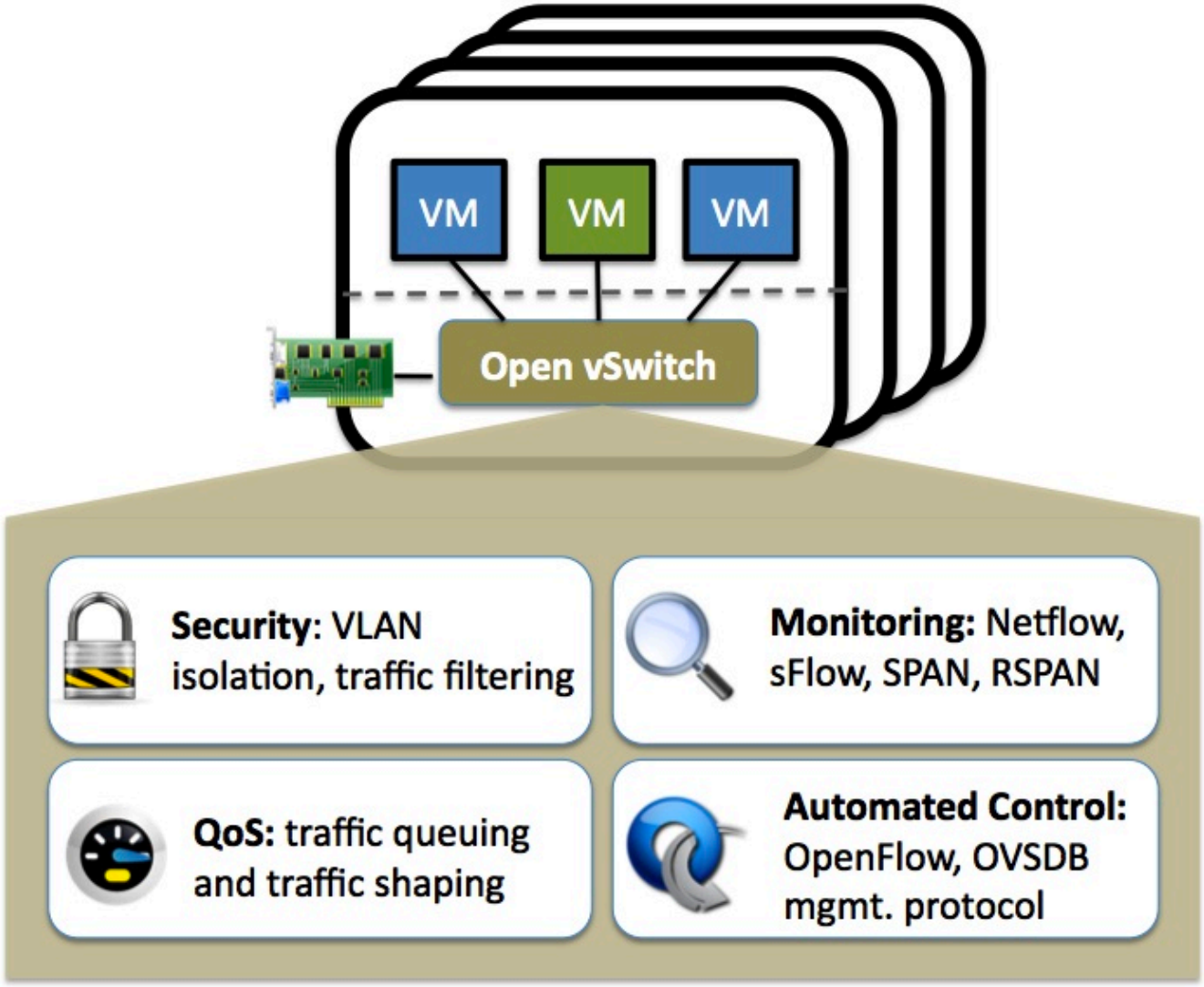


Mininet

- Mininet is a network emulator
  - Overview of Mininet - http://mininet.org/overview/
  - We have provided you a container that has installed Mininet
- Create a realistic virtual network, running real kernel, switch and application code, on a single machine (VM, cloud or native)
- Run a collection of end-hosts, switches, routers, and links on a single Linux kernel.



Open vSwitch (OvS)



Why Mininet?

- Fast and easily
- Create custom topologies
- Run real programs
- Customize packet forwarding
- Support OpenFlow and software-defined network (SDN)

Mininet CLI (Command-Line Interface)

- Start a minimal topology and enter the CLI

```
$ sudo mn
mininet> help
```

- Show the information of every nodes

```
mininet> nodes
```

- Show every links of all nodes

```
mininet> links
```

- Show the network topology

```
mininet> net
```

- Show all ports on every switches

```
mininet> ports
```

Mininet CLI (Command-Line Interface)

- Show all network interfaces

```
mininet> intfs
```

- Dump information about all nodes

```
mininet> dump
```

- Test the connectivity of all hosts

```
mininet> pingall
```

- Test TCP connection of two hosts with iPerf

```
mininet> iperf
```

- Leave the Mininet's CLI mode

```
mininet> exit
```

Mininet References

- English
  - Mininet Walkthrough
  - Introduction to Mininet
  - Mininet Python API Reference Manual
  - A Beginner's Guide to Mininet
- Chinese
  - GitHub/OSE-Lab - 熟悉如何使用 Mininet
  - 菸酒生的記事本 – Mininet 筆記
  - Hwchiu Learning Note – 手把手打造仿 mininet 網路
  - 阿寬的實驗室 – Mininet 指令介紹
  - Mininet 學習指南

iPerf

- iPerf is a tool for active measurements of the maximum achievable bandwidth on IP networks
- Support tuning of various parameters related to timing, buffers and protocols (TCP, UDP, SCTP with IPv4 and IPv6)

File Structure

```
Network_Topology/                # This is ./ in this repository
|--- src/                        # Folder of source code
     |--- topo/                  # The figure of topology
          |--- topo0.png
          |--- topo1.png
          |--- topo2.png
     |--- expect/                # Expected result using iPerf
          |--- topo0
          |--- topo1
          |--- topo2
     |--- out/                   # Output files
          |--- .gitkeep          # For keeping this folder
     |--- example.py             # Example code of using Mininet
     |--- topology.py            # Your program should be here!
|--- LICENSE
|--- README.md                   # Your report of Lab2!
|--- .gitignore                  # For ignoring useless files
```

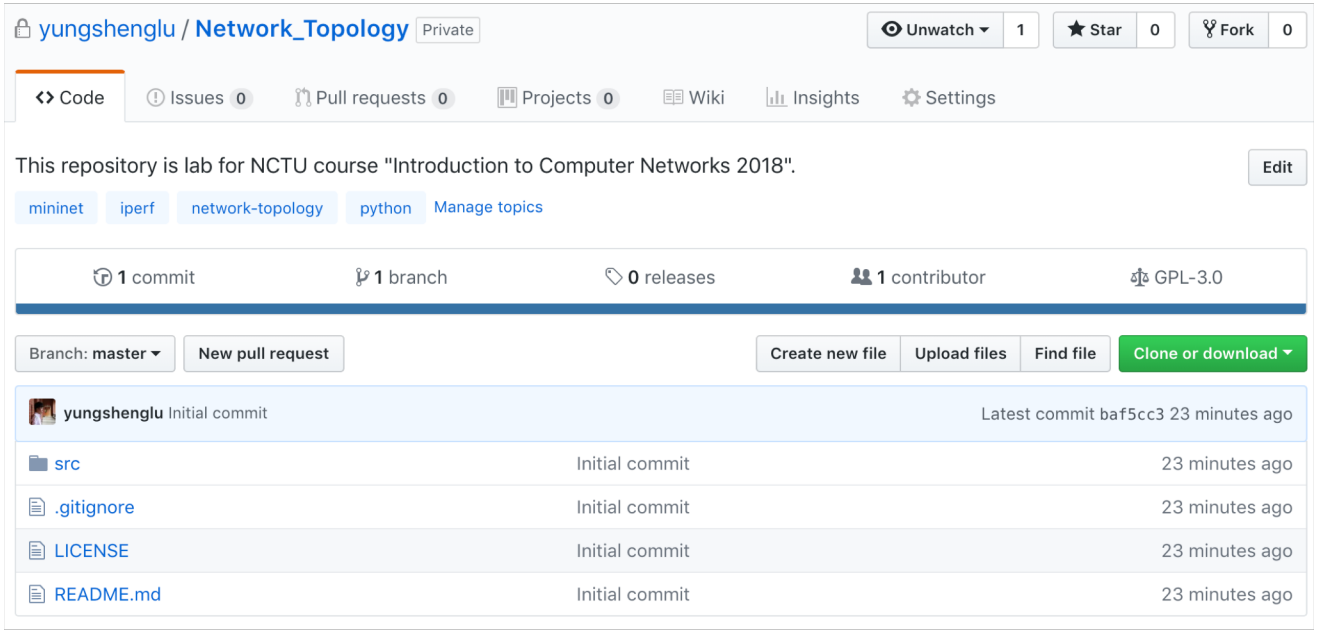
Tasks

Tasks

1. Environment Setup
2. Example of Mininet
3. Topology Generator
4. Measurement
5. Report

Task 1. Environment Setup

- **Step 1. Join this lab on GitHub Classroom**
  - Click the following link to join this lab
    - https://classroom.github.com/a/K8gaizQG
  - Go to our GitHub group to see your repository
    - https://github.com/nctucn
  - You will have an initial repository we prepared



Task 1. Environment Setup (cont.)

- **Step 2. Login to your container using SSH**
  - **For windows**
    - Open the PieTTY and connect to your container
      - IP address: `140.113.195.69`
      - Port: **Last 5 digits of student ID**
    - Login as root

```
Login: root
Password: cn2018
```

  - **For MacOS and Ubuntu Linux**

```
# Open the terminal to connect to your container
$ ssh root@140.113.195.69 -p <LAST_5_DIGITS_OF_STUDENT_ID>
Password: cn2018
```

Task 1. Environment Setup (cont.)

- **Step 3. Clone your GitHub repository**

```
# Clone your GitHub repository to “Network_Topology”
$ git clone https://github.com/nctucn/lab2-<GITHUB_ID>.git
Network_Topology
Cloning into 'Network_Topology'...
Username for 'https://github.com': <GITHUB_ID>
Password for 'https://<GITHUB_ID>@github.com':
<GITHUB_PASSWORD>
......

# Show all files in /root/
$ ls /root/
Network_Topology
```

After cloning the repository from your GitHub to on your container, you will see a folder ”Network_Topology”

Task 1. Environment Setup (cont.)

- **Step 4. Run Mininet for testing**
  - After logging to your container, you may meet the following error when running Mininet

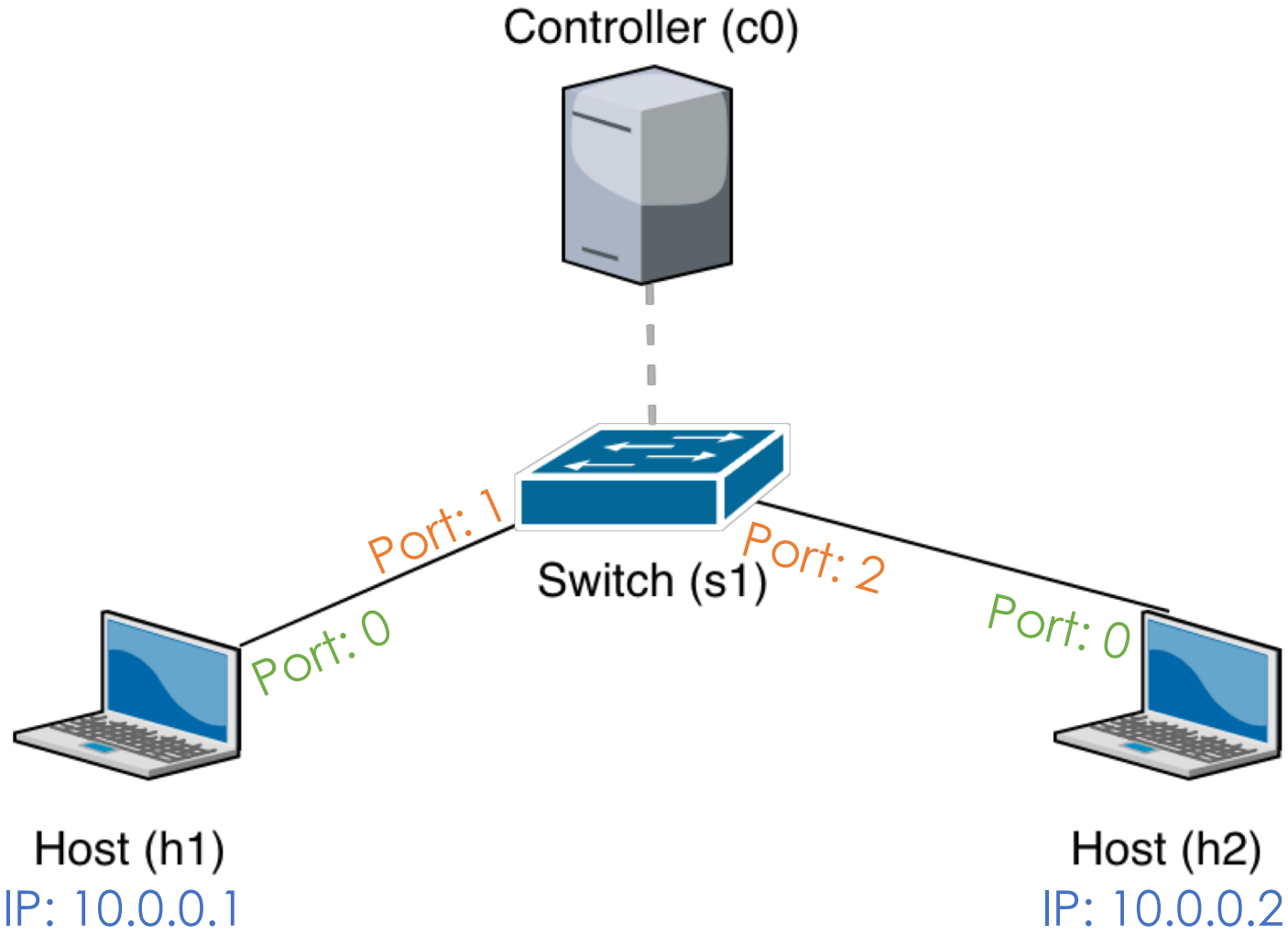
```
# Run Mininet for testing
$ [sudo] mn
......
*** Error connecting to ovs-db with ovs-vsctl
Make sure that Open vSwitch is installed, that ovsdb-
server is running, and that
"ovs-vsctl show" works correctly.
You may wish to try "service openvswitch-switch start".
```

  - **Solution**

```
# Start the service of Open vSwitch
$ [sudo] service openvswitch-switch start
# Run Mininet again!
$ [sudo] mn
```

Task 2. Example of Mininet

- Network topology of example.py



Task 2. Example of Mininet (cont.)

- Run the example code

```
# Change the directory into /Network_Topology/src/
$ cd /root/Network_Topology/src/
# Change to the executable mode of example.py
$ [sudo] chmod +x example.py
# Run example code (example.py)
$ [sudo] ./example.py
```

Task 2. Example of Mininet (cont.)

- The result after running example code

```
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(10.00Mbit 5ms delay 0.00000%
loss) (10.00Mbit 5ms delay
0.00000% loss) (h1, s1)
(10.00Mbit 5ms delay 0.00000%
loss) (10.00Mbit 5ms delay
0.00000% loss) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
C0
```

```
*** Starting 1 switches
s1 ... (10.00Mbit 5ms delay 0.00000%
loss) (10.00Mbit 5ms delay 0.00000%
loss)
Testing network connectivity
*** Ping: testing ping reachability
h1 -> h2
h2 -> h1
*** Results: 0% dropped (2/2
received)
*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
```

Task 2. Example of Mininet (cont.)

- **Troubleshooting 1**
  - The following error may occur when you run example.py or Mininet's program

```
# Change the directory into /root/Network_Topology/src/
# Run the example code (example.py)
$ [sudo] ./example.py
*** Creating network
......
Exception: Error creating interface pair (s1-eth1,s2-
eth1): RTNETLINK answers: File exists
```

  - **Solution:**

```
# If Mininet crashes for some reason, clean it up!
$ [sudo] mn -c
```

Task 3. Topology Generator

- **Step 1. View the topology you should generate**
  - Please divide your student ID by 3 to get the remainder and find the figure you should generate in folder /Network_Topology/src/topo/

Remainder	Topology figure
0	topo0.png
1	topo1.png
2	topo2.png

Task 3. Topology Generator (cont.)

- **Step 2. Generate the topology via Mininet**
  - Write a Python program named topology.py and put it at the same place with example.py
  - Your program need to generate a network topology for Mininet
    - Create hosts and switches
    - Construct links
    - Configure link bandwidth, delay, and loss rate
  - You can refer to the example.py and make sure you really understand each line of code

Task 3. Topology Generator (cont.)

- **Other requirements**
  - Dump every hosts' connections in your program

```
# Remember to import the following module first!
from mininet.util import dumpNodeConnections
# Dump every hosts' and switches' connections
dumpNodeConnections(net.hosts)
dumpNodeConnections(net.switches)
```

  - Enter in the Mininet's CLI mode in your program

```
# Remember to import the following module first!
from mininet.cli import CLI
# Add the following code and do NOT use net.stop()
CLI(net)
```

  - Please write your comments in your program; otherwise, you will be deemed as plagiarism

Task 3. Topology Generator (cont.)

- **Troubleshooting 2**
  - You can ping each link respectively by using the following command in the Mininet's CLI mode

```
# Example of testing the connectivity between h1 and h2
mininet> h1 ping h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=31.8 ms
......
```

  - Please refer to the **Troubleshooting 1** for solving the following error when running your program

```
# Run the example code (example.py)
$ sudo ./example.py
*** Creating network
......
Exception: Error creating interface pair (s1-eth1,s2-
eth1): RTNETLINK answers: File exists
```

Task 4. Measurement

- Use the following iPerf commands to measure the topology you built
  - For topo0.png

```
mininet> h2 iperf -s -u -i 1 > ./out/result &
mininet> h6 iperf -c 10.0.0.2 -u –i 1
```

  - For topo1.png

```
mininet> h4 iperf -s -u -i 1 > ./out/result &
mininet> h2 iperf -c 10.0.0.4 -u –i 1
```

  - For topo2.png

```
mininet> h6 iperf -s -u -i 1 > ./out/result &
mininet> h3 iperf -c 10.0.0.6 -u –i 1
```

  - The above commands will dump the result of iPerf's measurement into the file result
    - /Network_Topology/src/out/result

Task 4. Measurement (cont.)

- The expected result from the topo0.png

```
mininet> h2 iperf -s -u -i 1 > ./out/result &
mininet> h6 iperf -c 10.0.0.2 -u –i 1
------------------------------------------------------------
Client connecting to 10.0.0.2, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 10.0.0.6 port 41906 connected with 10.0.0.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec   129 KBytes  1.06 Mbits/sec
......
[  3] Server Report:
[  3]  0.0-10.3 sec   980 KBytes   802 Kbits/sec   15.210 ms
214/  893 (24%)
```

You will get the rate of packet loss which is an approximate value (21% ~ 26%)

Task 4. Measurement (cont.)

- The expected result from the topo1.png

```
mininet> h4 iperf -s -u -i 1 > ./out/result &
mininet> h2 iperf -c 10.0.0.4 -u –i 1
------------------------------------------------------------
Client connecting to 10.0.0.4, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 10.0.0.2 port 41906 connected with 10.0.0.4 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec   129 KBytes  1.06 Mbits/sec
......
[  3] Server Report:
[  3]  0.0-10.5 sec   583 KBytes   455 Kbits/sec   31.512 ms
487/  893 (55%)
```

You will get the rate of packet loss which is an approximate value (51% ~ 58%)

Task 4. Measurement (cont.)

- The expected result from the topo2.png

```
mininet> h6 iperf -s -u -i 1 > ./out/result &
mininet> h3 iperf -c 10.0.0.6 -u -i 1
------------------------------------------------------------
Client connecting to 10.0.0.6, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 10.0.0.3 port 41906 connected with 10.0.0.6 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec   129 KBytes  1.06 Mbits/sec
......
[  3] Server Report:
[  3]  0.0-10.0 sec   1.06 MBytes   888 Kbits/sec   0.139 ms
138/  893 (15%)
```

You will get the rate of packet loss which is an approximate value (13% ~ 18%)

Task 5. Report

- **Write your report in Markdown format**
  - Cheat Sheet of Markdown Syntax
  - Finish the README.md as your report
- **Your README.md must include**
  - **Execution**
    - How to run your program?
    - Screenshot the result of using iPerf command in Mininet
  - **Description**
    - Describe how you finish this work in detail
    - How to use Mininet API in Python?
    - What does the iPerf command you used mean?
    - Which reference you refer to?

Task 5. Report (cont.)

- **Notice**
  - Please write your report in detail and do NOT just copy the content in this slide; otherwise, you won't get score
  - You can refer any other materials to help you finish your report but remember to attach the source
  - You can write your report in English or Chinese
  - Make sure your report is named README.md; otherwise, we won't mark your report

Submission

- **Submit your works to your GitHub repository**

```
# Add all files into staging area
$ git add .
# Commit your files
$ git commit -m "YOUR OWN COMMIT MESSAGE"
# Push your files to remote
$ git push origin master
```

- **Notice**
  - You should NOT commit your works only one time; otherwise, your lab may be deemed as plagiarism
  - You should write your commit message clearly
    - How to Write a Git Commit Message
  - Make sure that your final work is on `master` branch

Grading Policy

- **Deadline - Dec. 06, 2018. 23:00**
  - Upload all your works and report to your GitHub repository
  - Make sure the filename of each file is correct; otherwise; we won't mark it.
  - Do NOT attack our server; otherwise, you will get "F" this semester!
- **Homework, 100%**
  1. Python program, 80%
  2. Report, 20%

Grading Policy (cont.)

- **Late Policy** (follow syllabus)

$$(\textbf{Your score}) \times 0.8^{D},$$

where D is the number of days over due

- **Cheating Policy** (follow syllabus)
  - Academic integrity
  - Homework must be your own – cheaters share the score
  - Both the cheaters and the students who aided the cheater will be held responsible for the cheating

References

- **Mininet**
  - English
    - Mininet Walkthrough
    - Introduction to Mininet
    - Mininet Python API Reference Manual
    - A Beginner's Guide to Mininet
  - Chinese
    - GitHub/OSE-Lab - 熟悉如何使用 Mininet
    - 菸酒生的記事本 – Mininet 筆記
    - Hwchiu Learning Note – 手把手打造仿 mininet 網路
    - 阿寬的實驗室 – Mininet 指令介紹
    - Mininet 學習指南

References (cont.)

- **Python**
  - Python 2.7.15 Standard Library
  - Python Tutorial - Tutorialspoint
- **Others**
  - iPerf3 User Documentation
  - Cheat Sheet of Markdown Syntax
  - Vim Tutorial – Tutorialspoint
  - 鳥哥的 Linux 私房菜 – 第九章、vim 程式編輯器