

# Computer Networks

## @CS.NCTU

Lab. 3: Route Configuration

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 with Ryu SDN framework to build a simple software-defined network and compare the differences between two forwarding rules.

1. Learn how to build a simple software-defined networking with Ryu SDN framework
2. Learn how to add forwarding rules into each OpenFlow switch

# TODO

---

1. We will give you a Python code ([SimpleTopo.py](#)) that includes an example network topology and another Python code ([SimpleController.py](#)) that includes Ryu controller
2. We will get you a figure illustrating a topology you should generate
3. Copy the necessary function code from [SimpleTopo.py](#) and [SimpleController.py](#) to your Python code ([topo.py](#) and [controller.py](#)) to build your networks with forwarding rules

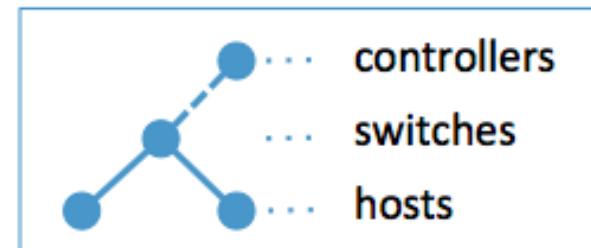
# Overview

# 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.

> sudo mn



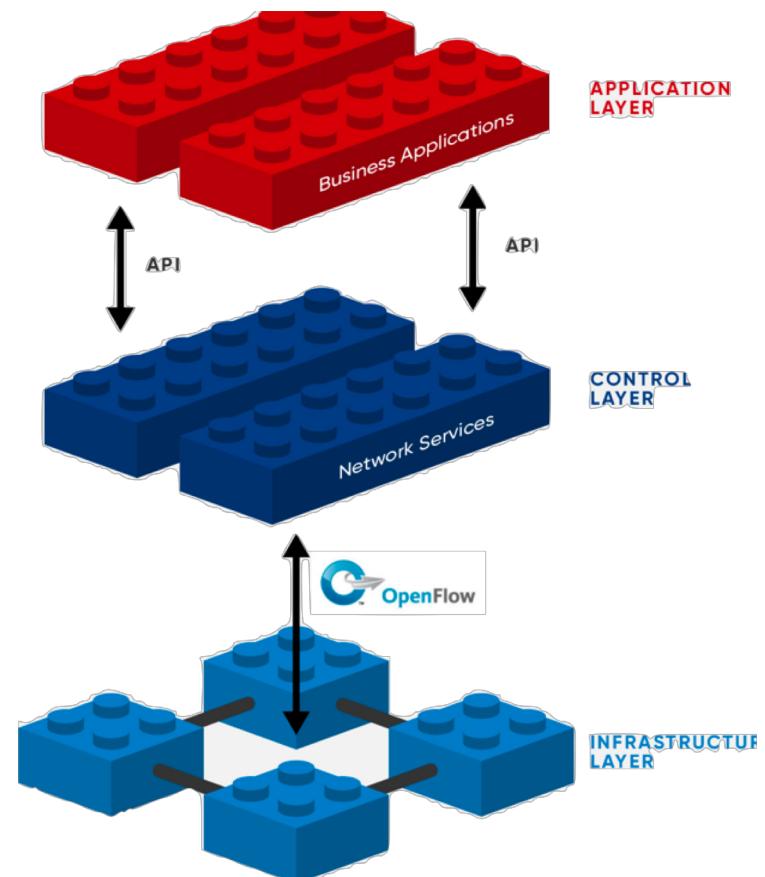
# Mininet References

---

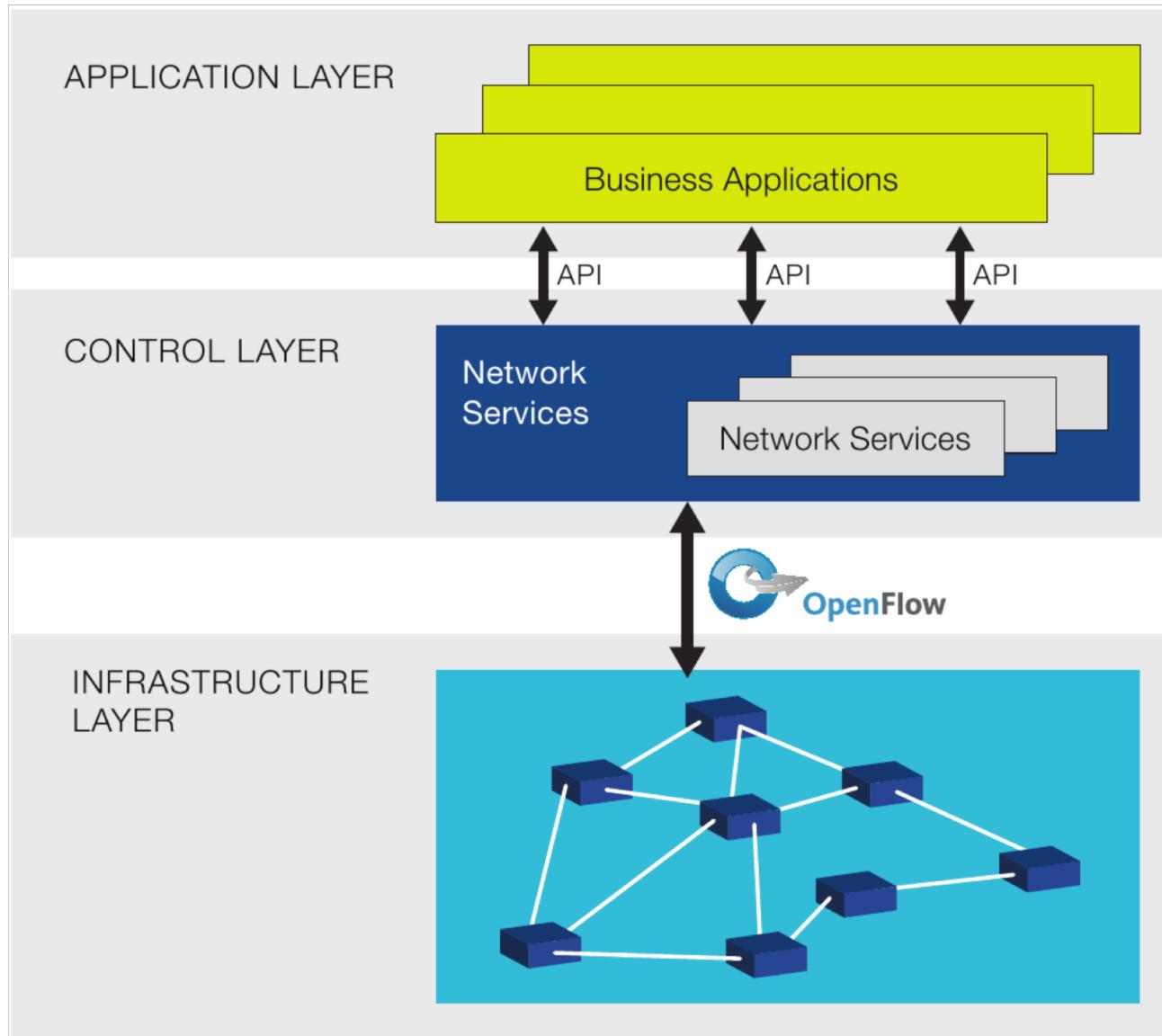
- English
  - [Mininet Walkthrough](#)
  - [Introduction to Mininet](#)
  - [Mininet Python API Reference Manual](#)
  - [A Beginner's Guide to Mininet](#)
- Chinese
  - [GitHub/OSE-Lab - 熟悉如何使用 Mininet](#)
  - [菸酒生的記事本 – Mininet 筆記](#)
  - [Hwchiu Learning Note – 手把手打造仿 mininet 網路](#)
  - [阿寬的實驗室 –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

# Software-Defined Networking (SDN)

- Software-defined  
= Programmable
  - Dynamic
  - Manageable
  - Cost-effective
  - Adaptable
- The OpenFlow protocol is a foundational element for building SDN



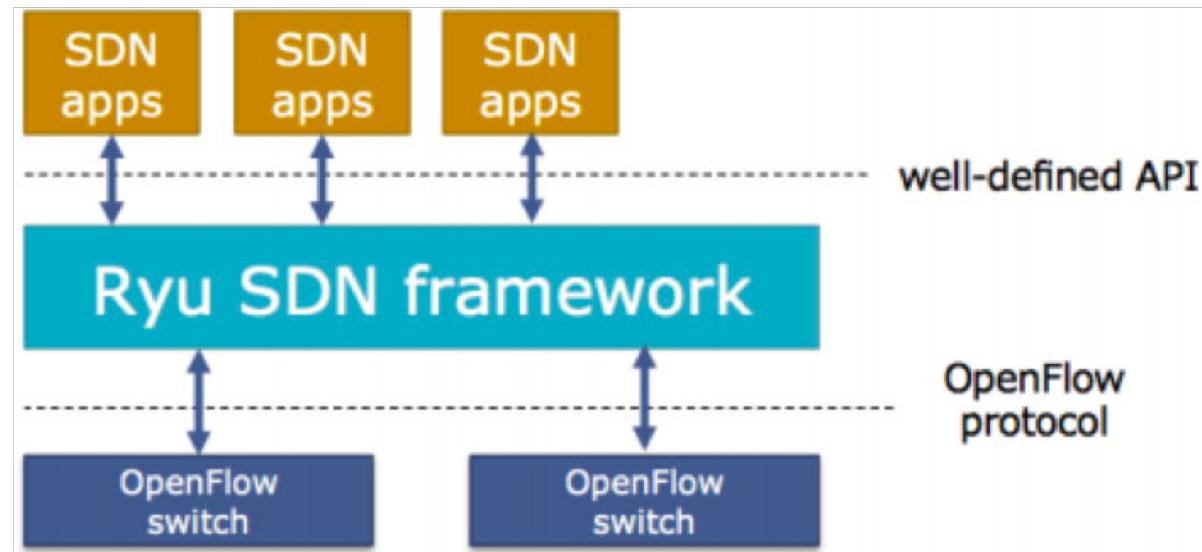
# OpenFlow





# Ryu SDN

- Ryu is a component-based software defined networking framework
  - Support various protocols for managing network devices, such as OpenFlow, etc
  - We have provided you a container that has installed Ryu



# Ryu SDN References

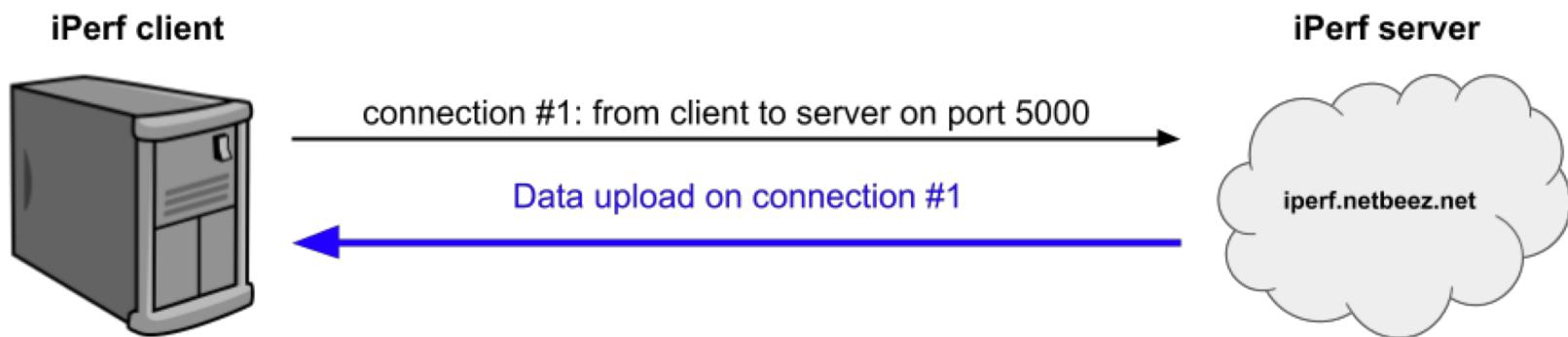
---

- English
  - [Ryubook Documentation](#)
  - [Ryubook \[PDF\]](#)
  - [Ryu 4.30 Documentation](#)
  - [Ryu Controller Tutorial](#)
  - [OpenFlow 1.3 Switch Specification](#)
- Chinese
  - [Ryubook 說明文件](#)
  - [GitHub - Ryu Controller 教學專案](#)
  - [Ryu SDN 指南 – Pengfei Ni](#)
  - [OpenFlow 通訊協定](#)

# 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)



`iperf3 -c iperf.netbeez.net -p 5000 -R`

`iperf3 -s -p 5000`

# File Structure

```
Route_Configuration/          # This is ./ in this repository
|--- src/                     # Folder of source code
|   |--- scripts/             # Folder of scripts
|   |   |--- run_mininet.sh  # Running script of Mininet
|   |   |--- run_ryu.sh      # Running script of Ryu manager
|   |--- topo/                # Folder of topology figure
|   |   |--- topo.png
|   |--- out/                 # Output files
|   |   |--- .gitkeep         # For keeping this folder
|   |--- SimpleTopo.py       # Example code of topology
|   |--- SimpleController.py # Example code of controller
|   |--- controller.py       # Your program should be here!
|   |--- topo.py              # Your program should be here!
|--- LICENSE
|--- README.md                # Your report of Lab3!
|--- .gitignore               # For ignoring useless files
```

# Tasks

# Tasks

---

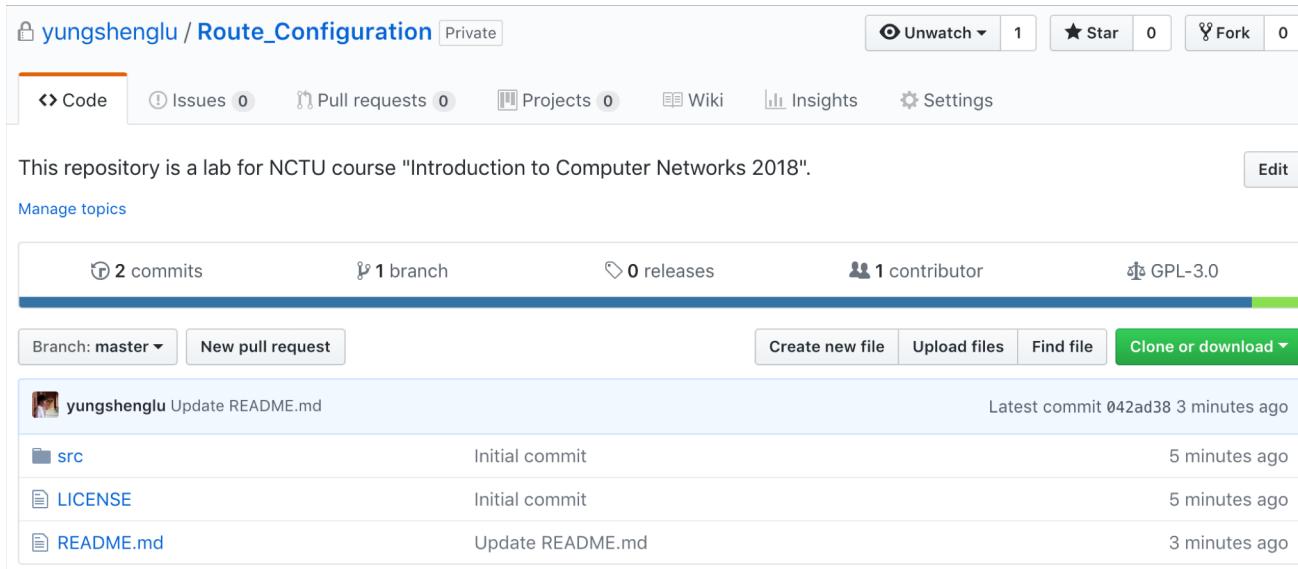
1. Environment Setup
2. Example of Ryu SDN
3. Mininet Topology
4. Ryu Controller
5. Measurement
6. Report

# 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/RHNMq4Td>
- 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 [PuTTY](#) 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 -p <LAST_5_DIGITS_OF_STUDENT_ID> root@140.113.195.69  
Password: cn2018
```

# Task 1. Environment Setup (cont.)

- Step 3. Clone your GitHub repository

```
# Clone your GitHub repository to "Route_Configuration"
$ git clone https://github.com/nctucn/lab3-<GITHUB_ID>.git
Route_Configuration
Cloning into 'Route_Configuration'...
Username for 'https://github.com': <GITHUB_ID>
Password for 'https://<GITHUB_ID>@github.com':
<GITHUB_PASSWORD>
.....
```

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



After cloning the repository from your GitHub to on your container, you will see a folder "Route\_Configuration"

# 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".
```

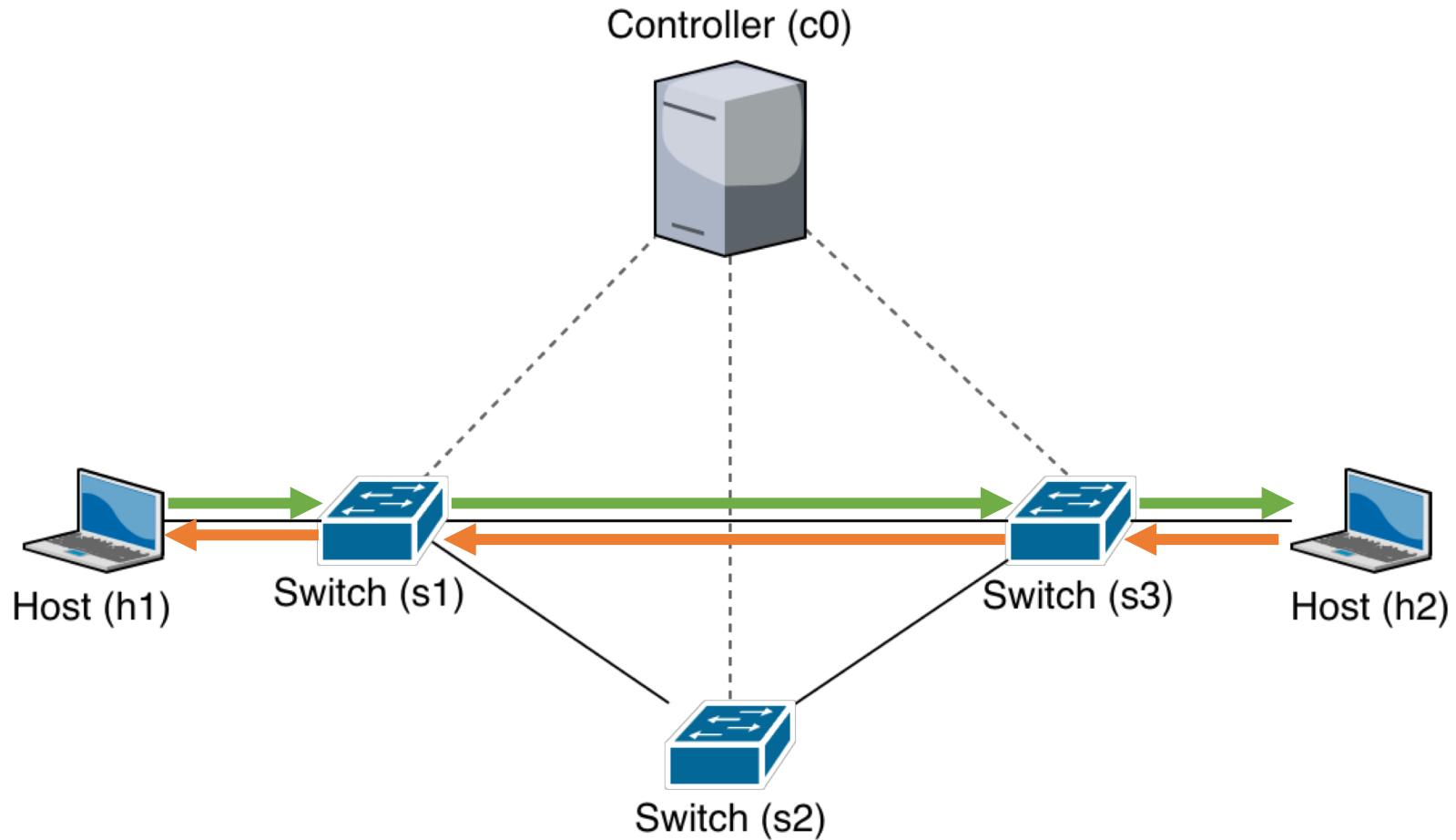
- **Solution**

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

## Task 2. Example of Ryu SDN

---

- Network topology of SimpleTopo.py



# Task 2. Example of Ryu SDN (cont.)

---

- **Step 1. Login to your container in two terminals**

- **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 -p <LAST_5_DIGITS_OF_STUDENT_ID> root@140.113.195.69  
Password: cn2018
```

# Task 2. Example of Ryu SDN (cont.)

---

- **Step 2. Run Mininet topology**

- Run `SimpleTopo.py` in one terminal **first**

```
# Change the directory into  
# /root/Route_Configuration/src/  
$ cd /root/Route_Configuration/src/  
# Run the SimpleTopo.py with Mininet  
$ [sudo] mn --custom SimpleTopo.py --topo topo  
--link tc --controller remote
```

# Task 2. Example of Ryu SDN (cont.)

---

- The result after running `SimpleTopo.py`

```
*** Creating network
*** Adding controller
Unable to contact the remote controller at 127.0.0.1:6653
Unable to contact the remote controller at 127.0.0.1:6633
Setting remote controller to 127.0.0.1:6653
*** Adding hosts:
h1 h2
*** Adding switches:
s1 s2 s3
*** Adding links:
(s1, h1) (s1, s2) (s1, s3) (s3, h2) (s3, s2)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 3 switches
s1 s2 s3 ...
*** Starting CLI:
mininet>
```

# Task 2. Example of Ryu SDN (cont.)

---

- **Troubleshooting 1**

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

```
# Run the SimpleTopo.py with Mininet (SimpleTopo.py)
$ [sudo] mn --custom SimpleTopo.py --topo topo
--link tc --controller remote
*** 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 2. Example of Ryu SDN (cont.)

---

- **Troubleshooting 2**

- The following message which won't affect in this lab may show when you run `SimpleTopo.py` or your Mininet's program

```
# Run the example code (SimpleTopo.py)
$ [sudo] mn --custom SimpleTopo.py --topo topo
--link tc --controller remote
*** Error setting resource limits. Mininet's
performance may be affected.
*** Creating networkpo --link tc --
controller=remotecusto
.....
```

# Task 2. Example of Ryu SDN (cont.)

---

- **Step 3. Run Ryu manager with controller**

- Run `SimpleController.py` in **another** terminal

```
# Change the directory into
# /root/Route_Configuration/src/
$ cd /root/Route_Configuration/src/
# Run the SimpleController.py with Ryu manager
$ [sudo] ryu-manager SimpleController.py --observe-
links
loading app controller.py
loading app ryu.topology.switches
loading app ryu.controller.ofp_handler
instantiating app controller.py of SimpleController
instantiating app ryu.topology.switches of Switches
instantiating app ryu.controller.ofp_handler of
OFPHandler
```

# Task 2. Example of Ryu SDN (cont.)

---

- **Step 4. How to leave the Ryu controller?**

- Leave `SimpleTopo.py` in one terminal **first**

```
# Leave the Mininet CLI  
mininet> exit
```

- Then, leave `SimpleController.py` in another terminal

```
# Leave the Mininet CLI  
Ctrl-z  
# Make sure "RTNETLINK" is clean indeed  
$ mn -c
```

# Task 3. Mininet Topology (cont.)

---

- **Step 1. Build the topology via Mininet**

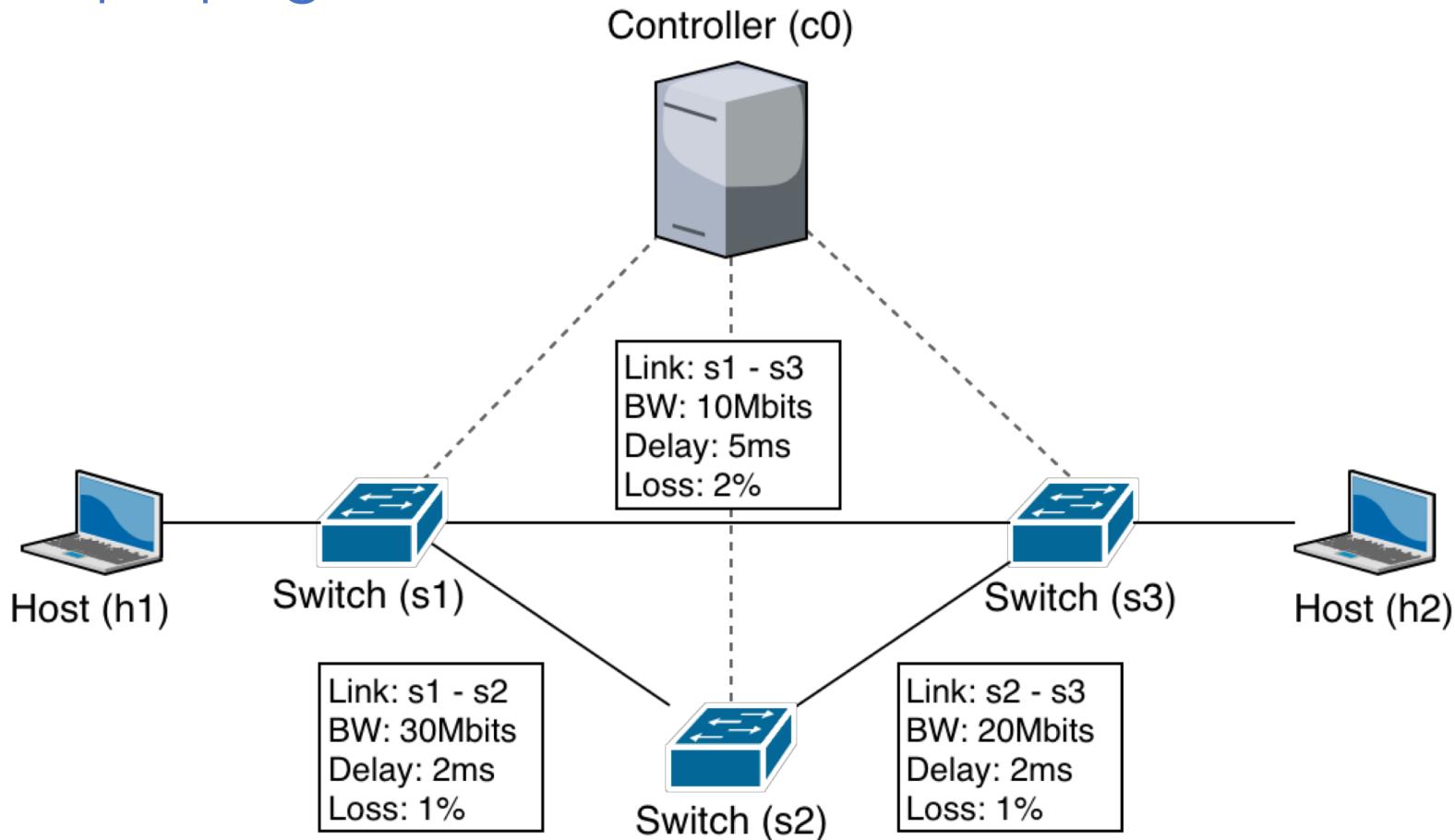
- Duplicate the example code `SimpleTopo.py` and name it `topo.py`

```
# Make sure the current directory is  
# /root/Route_Configuration/src/  
$ cp SimpleTopo.py topo.py
```

- Add the constraints (e.g., bandwidth, delay, and loss rate) by `/Route_Configuration/src/topo/topo.png`
- You don't need to set the bandwidth, delay, or loss rate if the figure don't specify.

# Task 3. Mininet Topology (cont.)

- Topology of [/Route\\_Configuration/src/topo/topo.png](#)



# Task 3. Mininet Topology (cont.)

---

- **Step 2. Run Mininet topology and controller**

- Run `topo.py` in one terminal **first**

```
# Run the topo.py with Mininet
$ [sudo] mn --custom topo.py --topo topo --link tc
--controller remote
.....
mininet>
```

- Then, run `SimpleController.py` in another terminal

```
# Run the SimpleController.py with Ryu manager
$ [sudo] ryu-manager SimpleController.py -observe-links
loading app controller.py
loading app ryu.topology.switches
loading app ryu.controller.ofp_handler
instantiating app controller.py of SimpleController
instantiating app ryu.topology.switches of Switches
instantiating app ryu.controller.ofp_handler of
OFPHandler
```

# Task 3. Mininet Topology (cont.)

---

## • Troubleshooting 3

- The following message means your controller's program has some error

```
$ ryu-manager SimpleController.py -observe-links
loading app SimpleController.py
Traceback (most recent call last):
  File "/usr/local/bin/ryu-manager", line 9, in
  .....
ImportError: No module named SimpleController.py
```

- The following message means your topology's program has some error

```
$ [sudo] mn --custom topo.py --topo topo --link tc
--controller remote
-----
Caught exception. Cleaning up...
SyntaxError: invalid syntax (SimpleTopo.py, line 19)
```

# Task 3. Mininet Topology (cont.)

---

## • Troubleshooting 4

- 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 SimpleTopo.py with Mininet (SimpleTopo.py)
$ [sudo] mn --custom SimpleTopo.py --topo topo
--link tc --controller remote
.....
Exception: Error creating interface pair (s1-eth1,s2-
eth1): RTNETLINK answers: File exists
```

# Task 4. Ryu Controller

- Step 1. Trace the code of Ryu controller
  - Trace the example code SimpleController.py

```
class SimpleController(app_manager.RyuApp):  
    # Let the Ryu controller running in protocol OpenFlow 1.3  
    OFP_VERSIONS = [ofproto_v1_3.OFP_VERSION]  
    ....  
    # Class constructor (DO NOT MODIFY)  
    def __init__(self, *args, **kwargs):  
        ....  
        # Add a flow into flow table of each switch (DO NOT MODIFY)  
    def add_flow(self, datapath, priority, match, actions):  
        ....  
        # Handle the initial feature of each switch  
    def switch_features_handler(self, ev):  
        ....  
        # Handle the packet-in events (DO NOT MODIFY)  
    def packet_in_handler(self, ev):  
        ....  
        # Show the information of the topology (DO NOT MODIFY)  
    def get_topology_data(self, ev):  
        ....
```

Your should  
modify here!

# Task 4. Ryu Controller

---

- **Step 2. Write another Ryu controller**

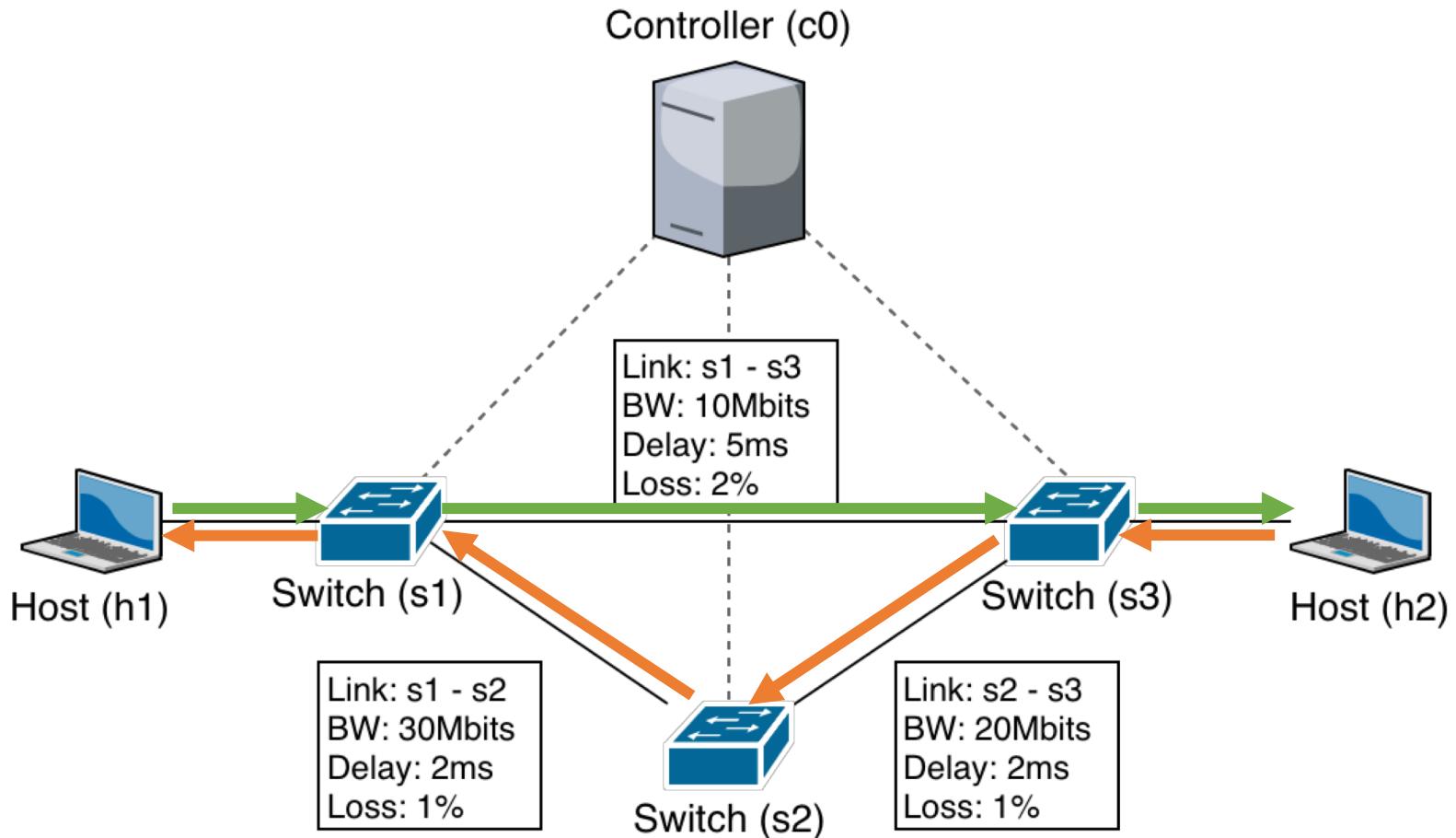
- Duplicate the example code `SimpleController.py` and name it `controller.py`

```
# Make sure the current directory is  
# /root/Route_Configuration/src/  
$ cp SimpleController.py controller.py
```

- Follow the the forwarding rules in the next slide and modify `controller.py`
- You **ONLY** need to modify the method `switch_features_handler(self, ev)`

# Task 3. Mininet Topology (cont.)

- Forwarding rules (for controller.py)



# Task 5. Measurement

---

- **Step 1. Run topology with SimpleController.py**

- Run `topo.py` in one terminal first

```
# Run the topo.py with Mininet
$ [sudo] mn --custom topo.py --topo topo --link tc
--controller remote
```

- Then, run `SimpleController.py` in another terminal

```
# Run the SimpleController.py with Ryu manager
$ [sudo] ryu-manager SimpleController.py -observe-links
loading app SimpleController.py
loading app ryu.topology.switches
loading app ryu.controller.ofp_handler
instantiating app SimpleController.py of
SimpleController
instantiating app ryu.topology.switches of Switches
instantiating app ryu.controller.ofp_handler of
OFPHandler
```

# Task 5. Measurement

---

- **Step 2. Measure the bandwidth**

- Use the following iPerf commands to measure the bandwidth in your network

```
# Run in the iPerf command in Mininet CLI  
mininet> h1 iperf -s -u -i 1 -p 5566 > ./out/result1 &  
mininet> h2 iperf -c 10.0.0.1 -u -i 1 -p 5566
```

- Leave topo.py in one terminal first

```
# Leave the Mininet CLI  
mininet> exit
```

- Then, leave SimpleController.py in another terminal

```
# Leave the Mininet CLI  
Ctrl-z  
# Make sure "RTNETLINK" is clean indeed  
$ mn -c
```

# Task 5. Measurement

---

- **Step 3. Run topology with controller.py**

- Run topo.py in one terminal **first**

```
# Run the topo.py with Mininet
$ [sudo] mn --custom topo.py --topo topo --link tc
--controller remote
```

- Then, run controller.py in another terminal

```
# Run the controller.py with Ryu manager
$ [sudo] ryu-manager controller.py -observe-links
loading app controller.py
loading app ryu.topology.switches
loading app ryu.controller.ofp_handler
instantiating app controller.py of SimpleController
instantiating app ryu.topology.switches of Switches
instantiating app ryu.controller.ofp_handler of
OFPHandler
```

# Task 5. Measurement

---

- **Step 4. Measure the bandwidth**
  - Use the following iPerf commands to measure the bandwidth in your network

```
# Run in the iPerf command in Mininet CLI  
mininet> h1 iperf -s -u -i 1 -p 5566 > ./out/result2 &  
mininet> h2 iperf -c 10.0.0.1 -u -i 1 -p 5566
```

- Leave **topo.py** in one terminal **first**

```
# Leave the Mininet CLI  
mininet> exit
```

- Then, leave **controller.py** in another terminal

```
# Leave the Mininet CLI  
Ctrl-z  
# Make sure “RTNETLINK” is clean indeed  
$ mn -c
```

# Task 6. 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?
    - What is the meaning of the executing command (both Mininet and Ryu controller)?
    - Screenshot the result of using iPerf command (both **SimpleController.py** and **controller.py**)
  - **Description**
    - Describe how you finish this work **in detail**
    - Which reference you refer to?

# Task 6. Report

---

- **Discussion** (also write in **README.md**)
  1. Describe the differences between packet-in and packet-out **in detail**.
  2. What is “table-miss” in SDN?
  3. Why is “`(app_manager.RyuApp)`” adding after the declaration of class in **controller.py**?
  4. Explain the following code in **controller.py**.

```
@set_ev_cls(ofp_event.EventOFPPacketIn, CONFIG_DISPATCHER)
```
  5. What is the meaning of “`datapath`” in **controller.py**?
  6. Why need to set “`ip_proto=17`” in the flow entry?
  7. Compare the differences between the iPerf results of **SimpleController.py** and **controller.py** **in detail**.
  8. Which forwarding rule is better? Why?

# Task 6. 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

---

- **Deadline1 - Dec. 30, 2018. 23:00 (5% bonus)**
- **Deadline2 - Jan. 14, 2019. 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!
  - No delay policy. You cannot submit after 2019/1/14
- **Homework, 100%**
  1. Python programs, 60%
    - topo.py, controller.py
  2. Report, 40%
    - README.md

# Grading Policy (cont.)

---

- **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

---

- **Ryu SDN**

- English

- [Ryubook Documentation](#)
    - [Ryubook \[PDF\]](#)
    - [Ryu 4.30 Documentation](#)
    - [Ryu Controller Tutorial](#)
    - [OpenFlow 1.3 Switch Specification](#)

- Chinese

- [Ryubook 說明文件](#)
    - [GitHub - Ryu Controller 教學專案](#)
    - [Ryu SDN 指南 – Pengfei Ni](#)
    - [OpenFlow 通訊協定](#)

# 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](#)