**University of Konstanz**

Dang Mai - MSc. Computer Science

Email: dang.mai@uni-konstanz.de

Software Define Network - Implementation Document

**January 17, 2020**

# Mininet - Simulate Network

**Installation**

* Install via VM ware (Virtual Box)
  + Download VM version from cloud
  + Download Virtual Box(if not)
  + Add file.vbox to Virtual Box using button Add
  + Setup network config for Virtual Box
    - Tools -> Network -> Create(check the name should be vboxnet0)
  + Run the Virtual Machine
  + Mininet Documentation: <http://mininet.org/download/>
* Install Ryu
  + Ryu documentation: <https://ryu.readthedocs.io/en/latest/getting_started.html>
  + Recommend:
    - Using virtual environment anaconda, so it does not effect to your system
    - Documentation: <https://docs.anaconda.com/anaconda/install/windows/>
    - Setup the virtual environment on python 2.7
  + Should install by source code so we easily config it for later experiment
    - sudo python setup.py install
  + Download source code for Ryu sdn controller:
    - <https://github.com/HaiDang9719/SDNImplementation.git>
* Install Floodlight
  + <https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/pages/1343544/Installation+Guide>
* Install directly to Ubuntu

**Manually setting(not recommend)**

1/ Set up wireshark

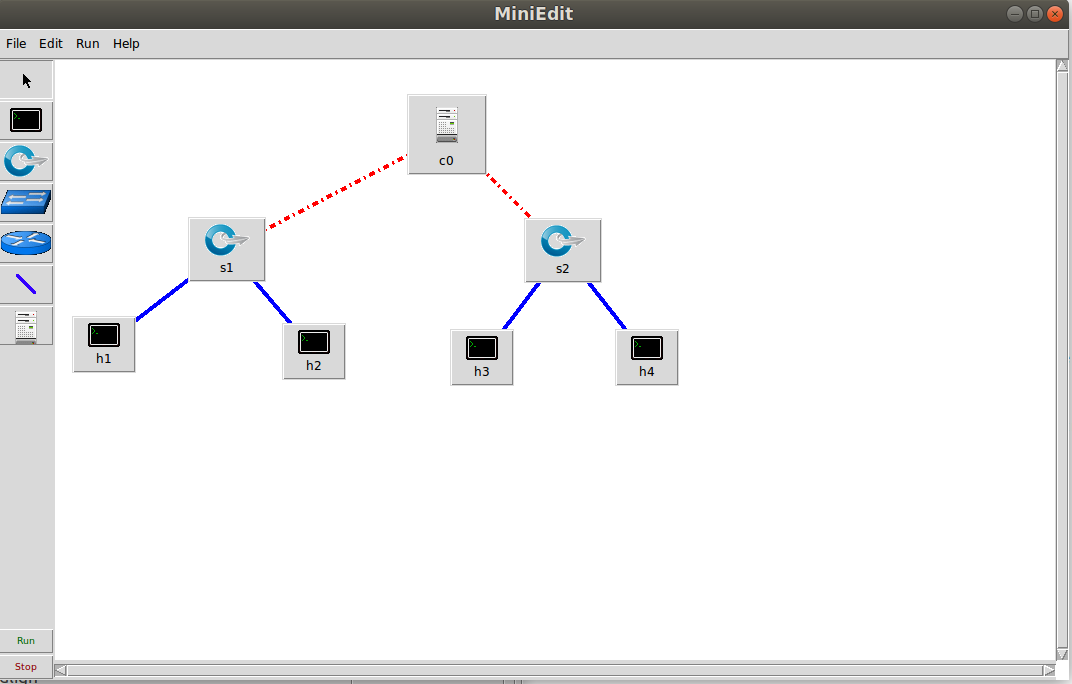
* Remove the old version on mininet
  + sudo apt-get remove --purge wireshark
* Install new version
  + sudo apt-get install wireshark
* Setup permission
  + sudo groupadd wireshark
  + sudo usermod -a -G wireshark YOUR\_USER\_NAME
  + sudo chgrp wireshark /usr/bin/dumpcap
  + sudo chmod 750 /usr/bin/dumpcap
  + sudo setcap cap\_net\_raw,cap\_net\_admin=eip /usr/bin/dumpcap
  + sudo getcap /usr/bin/dumpcap

2/ Set up ovs switch

* Remove old version
* Install new version from command line ubuntu

**Experiment**

1/ testing with topology network

* Run controller (Floodlight or Ryu): Detail at the experiment part below
* Remove any exist virtual network: sudo mn -c
* Create default network with default topology(check ip controller)
  + sudo mn --controller=remote,ip=134.34.231.214,port=6653 --switch ovsk,protocols=OpenFlow13
* Create network with custom topology
  + Create python script
    - sudo mn --custom topo.py --controller=remote,ip=134.34.231.214,port=6653 --switch ovsk,protocols=OpenFlow13 --topo mytopo
  + Create via GUI miniedit
    - sudo python ./mininet/examples/miniedit.py
    - Configuration - check appendix 1
    - Show flow
    - 
  + Should run one network at a time.

2/Setup experiment

* Run controller:
  + Run floodlight on eclipse or command line(if you would like to use Floodlight)
  + Run Ryu:
    - Redirect to the folder SDNImplementation -> SPT
    - Ryu-manager --observe-links simple\_switch\_stp\_13.py
    - Remember to kill previous ryu-controller if exists:
      * Pkill -9 ryu-manager
* Run Mininet VM from Virtual Box
* Connect through SSH:(check ip address of VM mininet)
  + ssh -X mininet@192.168.56.101
* Create a simple network:(check ip address of your machine and replace ip)
  + sudo mn --controller=remote,ip=134.34.231.214 --topo=single,3
* Create a custom network:
  + sudo mn --custom ~/SPT/topo.py --topo mytopo
* Run wireshark:
  + Open another terminal and connect to mininet through SSH
  + Run wireshark through command line:
    - Sudo wireshark &
  + Set interface:
    - Any
  + Set filter:
    - Openflow\_v1
* On the terminal that runs virtual network:
  + Test connection:
    - Command: pingall
  + Show flow entries from the switch:
    - Command: sh ovs-ofctl dump-flows s1
  + Add flow to the switch:
    - Command: sh ovs-ofctl add-flow s1 action=normal
  + Delete flow from the switch:
    - Command: sh ovs-ofctl del-flows s1
  + Test insert flow from SDN controller:
    - Command: h1 ping -c1 h3
    - Check the packet travel (Packet\_in, Packet\_out, Message…) using Wireshark
    - Check flow entries using sing command that show the flow entries from the switch
      * Example: cookie=0x20000007000000, duration=2.519s, table=0, n\_packets=0, n\_bytes=0, idle\_timeout=5, idle\_age=2, priority=1,ip,in\_port=1,dl\_src=4e:44:63:f8:46:ed,dl\_dst=ae:f1:b6:be:c7:7b,nw\_src=10.0.0.1,nw\_dst=10.0.0.3 actions=output:3
      * cookie=0x20000008000000, duration=2.518s, table=0, n\_packets=0, n\_bytes=0, idle\_timeout=5, idle\_age=2, priority=1,ip,in\_port=3,dl\_src=ae:f1:b6:be:c7:7b,dl\_dst=4e:44:63:f8:46:ed,nw\_src=10.0.0.3,nw\_dst=10.0.0.1 actions=output:1
    - Note: because the default setting from the SDN controller(Floodlight) that the survival time for the new entry is just 5 seconds, then it will be deleted.

**Error when setup**

1. Error internet connection:
   1. Check internet connection with ping google.com
   2. If connection is err,
      1. Change source:
         1. lsb\_release -r
         2. Go to <http://repogen.simplylinux.ch/> to generate a new sources.list
         3. Select your **country** & **release**
         4. Check the first 12 boxes:
         5. Generate and copy your new list
         6. Backup the old file to *sources.list.old*
            1. ***mv /etc/apt/sources.list /etc/apt/sources.list.old***
         7. You can now either open vi to save the new list by doing:
            1. Sudo nano /etc/apt/sources.list
      2. echo "nameserver 8.8.8.8" | sudo tee /etc/resolv.conf > /dev/null
      3. echo "nameserver 8.8.4.4" | sudo tee /etc/resolv.conf > /dev/null
      4. sudo /etc/init.d/networking stop -> start
      5. Restart virtual machine
2. Can’t open display xterm
   1. MAC OS:
      1. Install Xquartz
         1. brew cask install xquartz
      2. Add the XQuartz launcher to the system default (following the solution in this
         1. launchctl load -w /Library/LaunchAgents/org.macosforge.xquartz.startx.plist
      3. Restart system
      4. Test:
         1. Echo $DISPLAY

# Mininet - STP with Ryu

* Write application implement STP with Ryu controller and Mininet
* Change the default setting in ryu
  + BRIDGE\_GROUP\_ADDRESS = '01:80:c2:00:00:0e'
* Reinstall to apply the setting:
  + $ cd ryu
  + $ sudo python setup.py install
* Run network on mininet, redirect to SPT folder
  + sudo python spanning\_tree.py
* Run ryu application on host machine
  + ryu-manager --observe-links simple\_switch\_stp\_13.py
* Ping h1 to h2
* View loop by arp with the command on mininet:
  + tcpdump -i s1-eth2 arp
  + tcpdump -i s2-eth2 arp
  + tcpdump -i s3-eth2 arp
* Show how STP disable s3-eth2 to prevent loop
  + Command: net -> see how the components in the network connect together
* Test case with if connection fail:
  + Disable s2-eth2:
    - Ifconfig s2-eth2 down -> see STP recalculate the path in the network.
* Play around with some VSwitch command:
* Sudo ovs-ofctl dump-flows s1 -O OpenFlow (specific protocol for ovs)
* Reference: <https://osrg.github.io/ryu-book/en/html/spanning_tree.html>

# Mininet - SBP with Ryu

* Reference: <https://sdn-lab.com/2014/12/25/shortest-path-forwarding-with-openflow-on-ryu/>

# Mininet - Load Balancing with Ryu

Step 1: Create topology for the experiment: two host h1 and h2.

* Do via Python code
* Do via GUI miniedit
* There are two paths that connect h1 and h2. Path1: h1 -> s1 -> s2 -> h2, Path2: h1->s3->s4->s5->h2.
* Set up remote controller state and edit IP address. Config the protocol OpenFlow13
* View flows:
  + Sudo ovs-ofctl dump-flows s5 -O OpenFlow13
* View packet out of ports:
  + Sudo ovs-ofctl dump-ports s5 -O OpenFlow13

# Appendix

1/ Miniedit configuration

* Preferences:
  + Start CLI: Command line interface, display xterm window
* Save function:
  + Save: save mininet topology as \*.mn
  + Export Level 2 Script: save as python script