# Instalacja środowiska

**Mininet**

1) installing mininet -> https://www.youtube.com/watch?v=uR9ZEUEzB6E ; http://mininet.org/download/

download git -> sudo apt-get install git

get mininet from git -> git clone https://github.com/mininet/mininet.git

-> mininet/util/install.sh –a

this should get us open flow and wireshark

check:

-> sudo apt list --installed | grep wireshark

-> sudo apt list --installed | grep mininet

test:

-> sudo mn

-> sh wireshark &

2) **Instalacja Ryu**

<https://ernie55ernie.github.io/sdn/2019/03/25/install-mininet-and-ryu-controller.html>

git clone https://github.com/faucetsdn/ryu.git

cd ryu

sudo pip install -r tools/pip-requires

**Problem**:

sudo ./miniedit.py -> /usr/bin/env: ‘python’: No such file or directory

Solution:

sudo apt install python-is-python3

3) Instalacja samego OvS

apt-get install uml-utilities

apt install openvswitch-switch

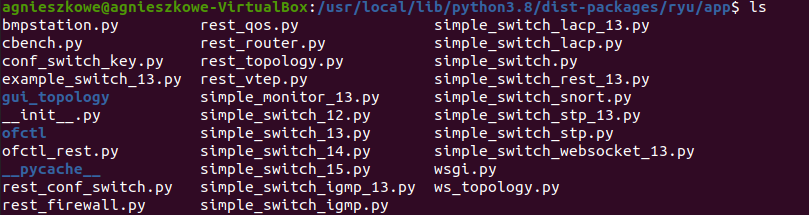
apt install qemu-kvm

## RYU

Odpalenie simple\_switch ryu: *ryu-manager ryu.app.simple\_switch*

Odpalenie dowolnego pliku ryu: *ryu-manager ryu.app.NAZWA\_PLIKU*

Ścieżka: /usr/local/lib/python3.8/dist-packages/ryu



[Ryu basics](https://www.youtube.com/watch?v=2VRsituJ6a8)

## **Porty**

PYTHON  
h1 python3.8 -m http.server 80 &

h2 wget -O - h1

Otworzenie portu na maszynie: <https://www.digitalocean.com/community/tutorials/opening-a-port-on-linux>

netstat -na | grep :4000

sudo ufw allow 4000

ls | nc -l -p 4000 &

telnet localhost 4000

nc -l -p 6666 &

! on sender (iptables)

iptables -A OUTPUT -p tcp --tcp-flags RST RST -j DROP

iptables -L

## 

## Podstawy Mininet

**Podstawowe komendy:**

pokaz dostepne nodes: *nodes*

pokaz siec: *net*

pokzanie flows: *dpctl dump-flows*

*sh wireshark &*

*h1 ifconfig*

*h1 xterm &*

*sh $command $who* - odpalenie komendy z shella

Odpalenie xterm dla hosta 2 z pokazywaniem nazwy dla terminala: h2 xterm -xrm 'XTerm.vt100.allowTitleOps: false' -T h2 &

**Przykłady oraz odpalenie topologii custom:**

mininet/custom -> przykład na 2 hostów i 2 switche

mininet/examples

Custom możemy zrobić w /custom lub /examples (więcej kontroli) odpalamy poprzez:

1. *sudo mn --custom ~/mininet/custom/$NAZWA.py --topo $mytopo --controller=remote*
2. ***sudo python $NAME.py***

W examples dobrym przykładem jest emptynet.py

**Miniedit (niezalecane):**

Odpalenie projektu:

Albo poprzez okno run

Albo komendą sudo python ./nazwaprojektu.py

# 

## OvS basics

Show flows*: sh ovs-ofctl dump-flows s1*

Show all flows: *dpctl dump-flows*

Show all tables and rules in them: *dpctl dump-tables*

Show how many flows are there in the table: *dpctl dump-aggregate*

Basic start of a command to use table flows: *ovs-ofctl*

Pokazanie jakie mamy bridge: s1 ovs-vsctl show

sh ovs-appctl dpif/dump-flows s1

Pokazanie tras: sh ovs-appctl ovs/route/show

Pokaż OvS version: mininet> s1 ovs-vsctl show

**dpctl dump-flows**

**dpctl dump-tables**

dpctl dump-aggregate

Mininet> **sh ovs-dpctl show** -> number of entries in EMC and Datapath

mininet> ovs-ofctl manual*->* <http://www.openvswitch.org/support/dist-docs/ovs-ofctl.8.txt>

Get dpid: mininet> py net.getNodeByName("s1").dpid

mininet> sh ovs-dpctl dump-flows system@ovs-system -- Prints to the console all flow entries in datapath dp's flow table

**Odrzucanie przepływów w razie przekroczenia ilości – ręczne (FMI):**

1. Make flow table 0 on bridge br0 refuse to accept more than 100 flows:
2. Make flow table 0 on bridge br0 evict flows, with fairness based on the matched ingress port, when there are more than 100:

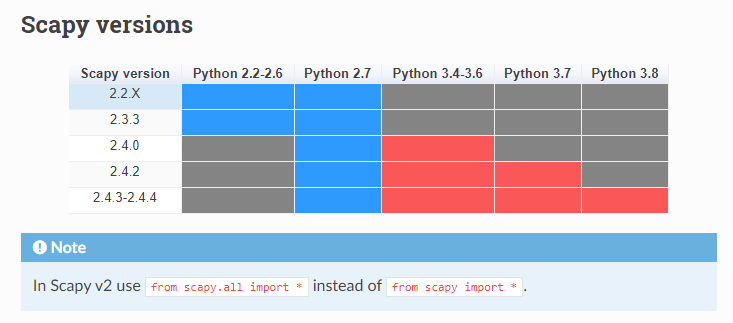
ovs-vsctl -- --id=@ft create Flow\_Table flow\_limit=1 overflow\_policy=refuse -- set Bridge s1 flow\_tables=0=@ft

ovs-vsctl -- --id=@ft create Flow\_Table flow\_limit=1 overflow\_policy=evict groups='"NXM\_OF\_IN\_PORT[]"' -- set Bridge s1 flow\_tables:0=@ft

ovs-vsctl -- --id=@ft2 create Flow\_Table flow\_limit=11 overflow\_policy=refuse -- set Bridge s1 flow\_tables=1=@ft2

ovs-vsctl -- --id=@ft2 create Flow\_Table flow\_limit=11 overflow\_policy=evict groups='"NXM\_OF\_IN\_PORT[]"' -- set Bridge s1 flow\_tables:1=@ft2

## SCAPY

Dokumentacja: https://scapy.readthedocs.io/en/latest/

