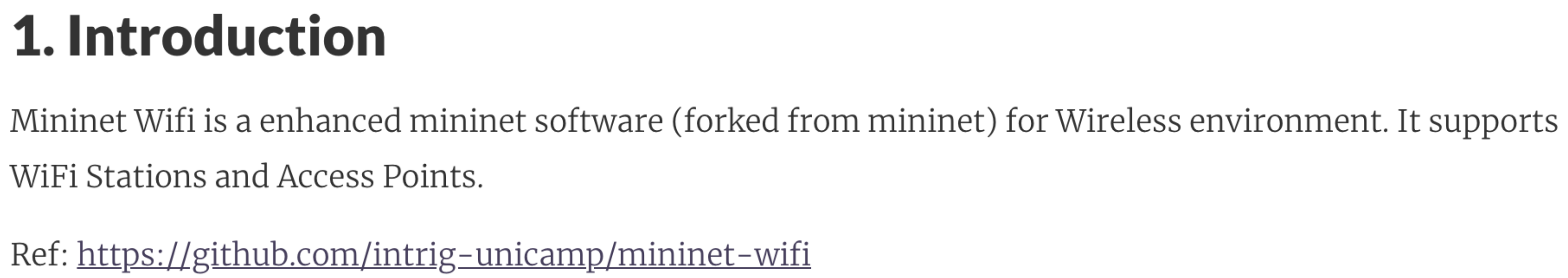
Software Defined Network
SDN

Ch 7

Mininet WiFi

Mininet WiFi Introduction

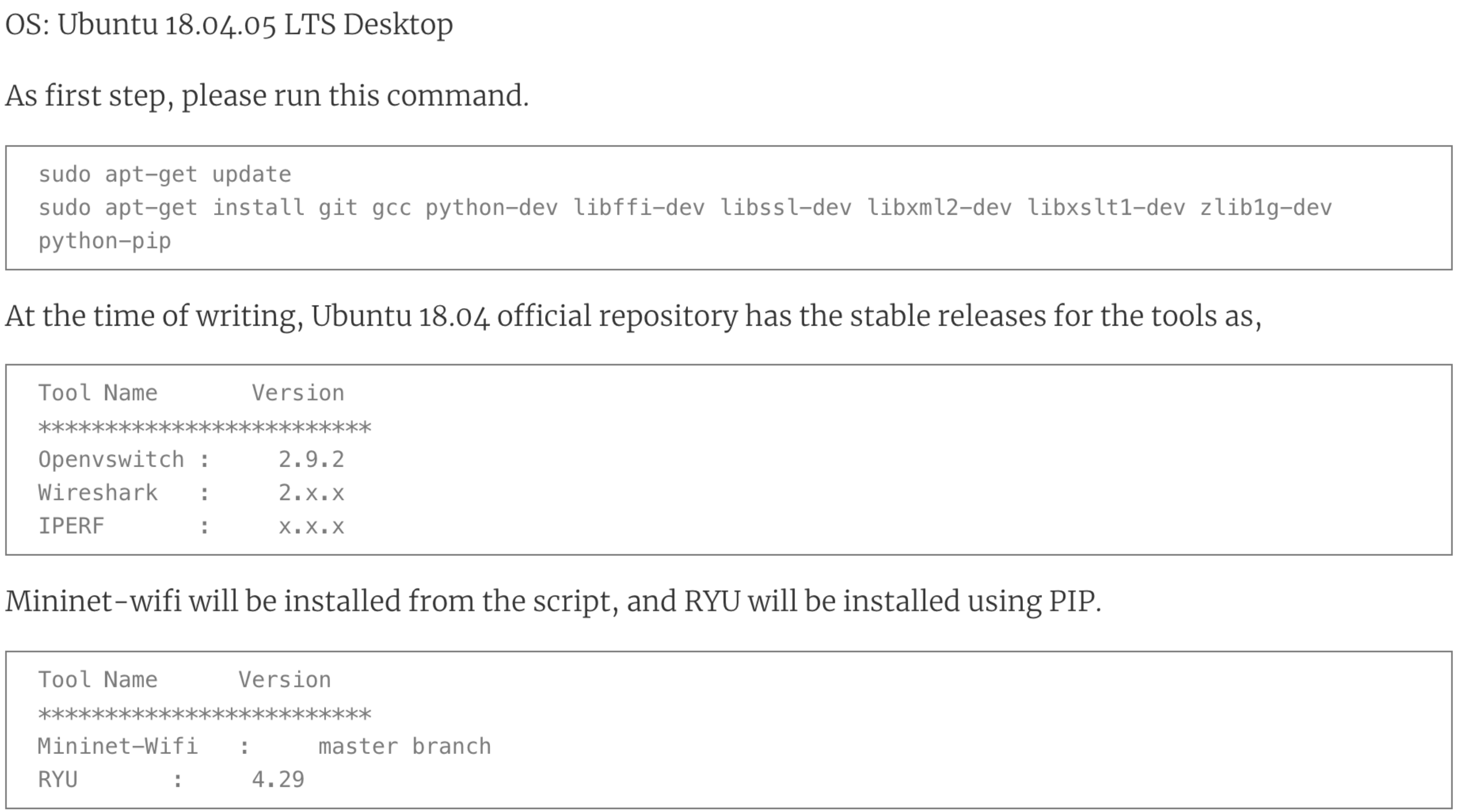


1. Introduction

Mininet Wifi is a enhanced mininet software (forked from mininet) for Wireless environment. It supports WiFi Stations and Access Points.

Ref: https://github.com/intrig-unicamp/mininet-wifi

Installation



OS: Ubuntu 18.04.05 LTS Desktop

As first step, please run this command.

```
sudo apt-get update
sudo apt-get install git gcc python-dev libffi-dev libssl-dev libxml2-dev libxslt1-dev zlib1g-dev
python-pip
```

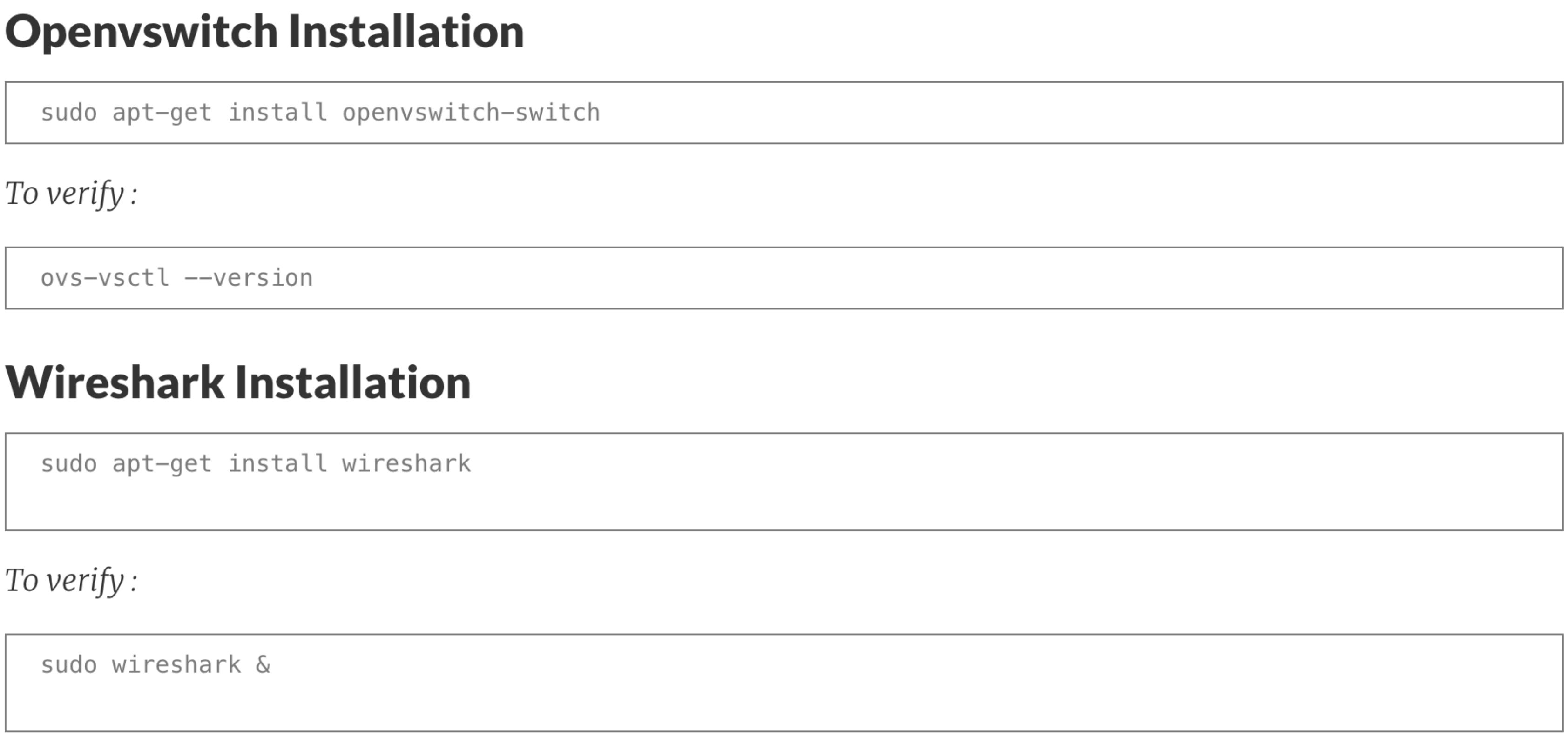
At the time of writing, Ubuntu 18.04 official repository has the stable releases for the tools as,

```
Tool Name       Version
*************************
Openvswitch :     2.9.2
Wireshark   :     2.x.x
IPERF       :     x.x.x
```

Mininet-wifi will be installed from the script, and RYU will be installed using PIP.

```
Tool Name      Version
*************************
Mininet-Wifi   :     master branch
RYU       :     4.29
```

Installation



Openvswitch Installation

```
sudo apt-get install openvswitch-switch
```

To verify :

```
ovs-vsctl --version
```

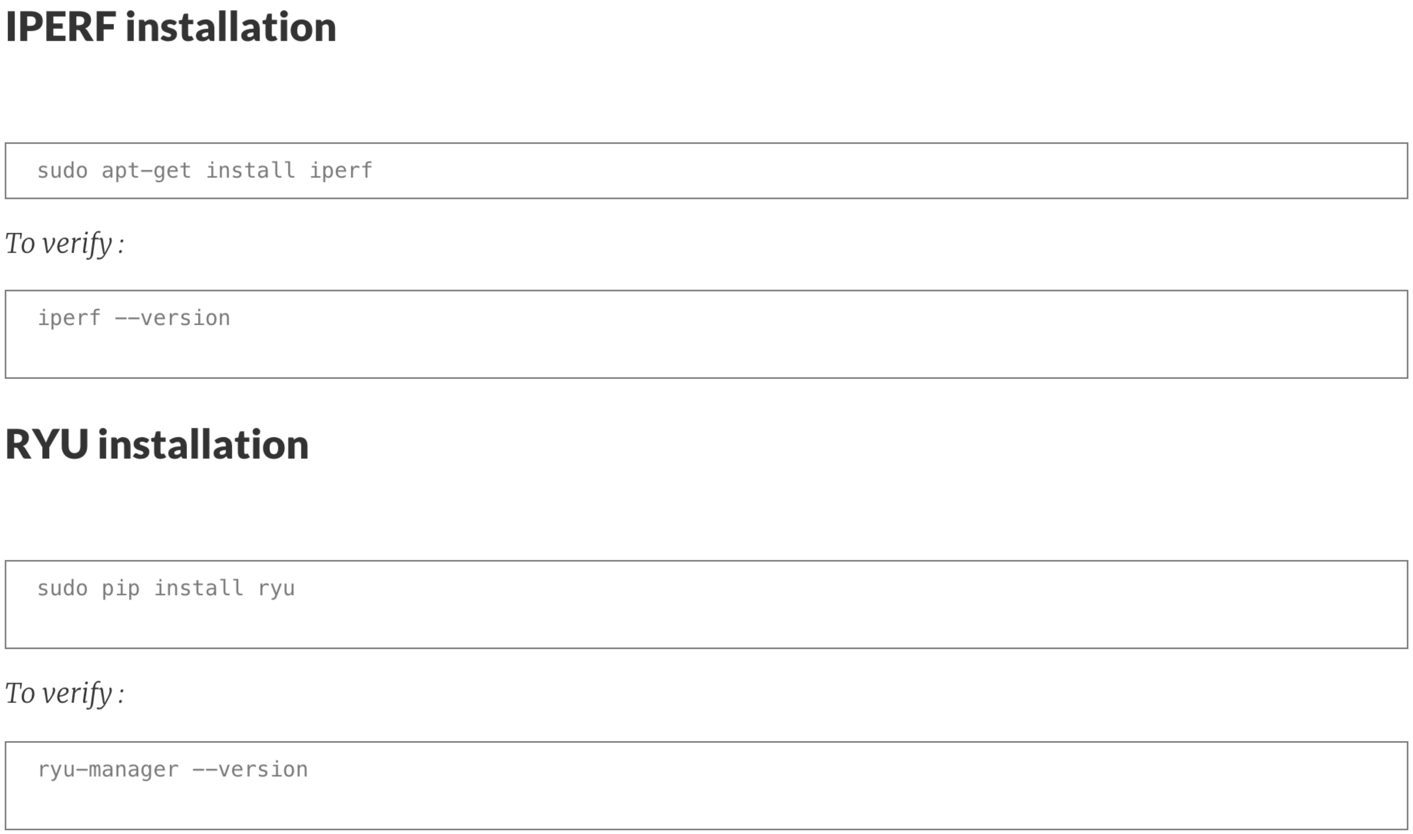
Wireshark Installation

```
sudo apt-get install wireshark
```

To verify :

```
sudo wireshark &
```

Installation



IPERF installation

```
sudo apt-get install iperf
```

To verify :

```
iperf --version
```

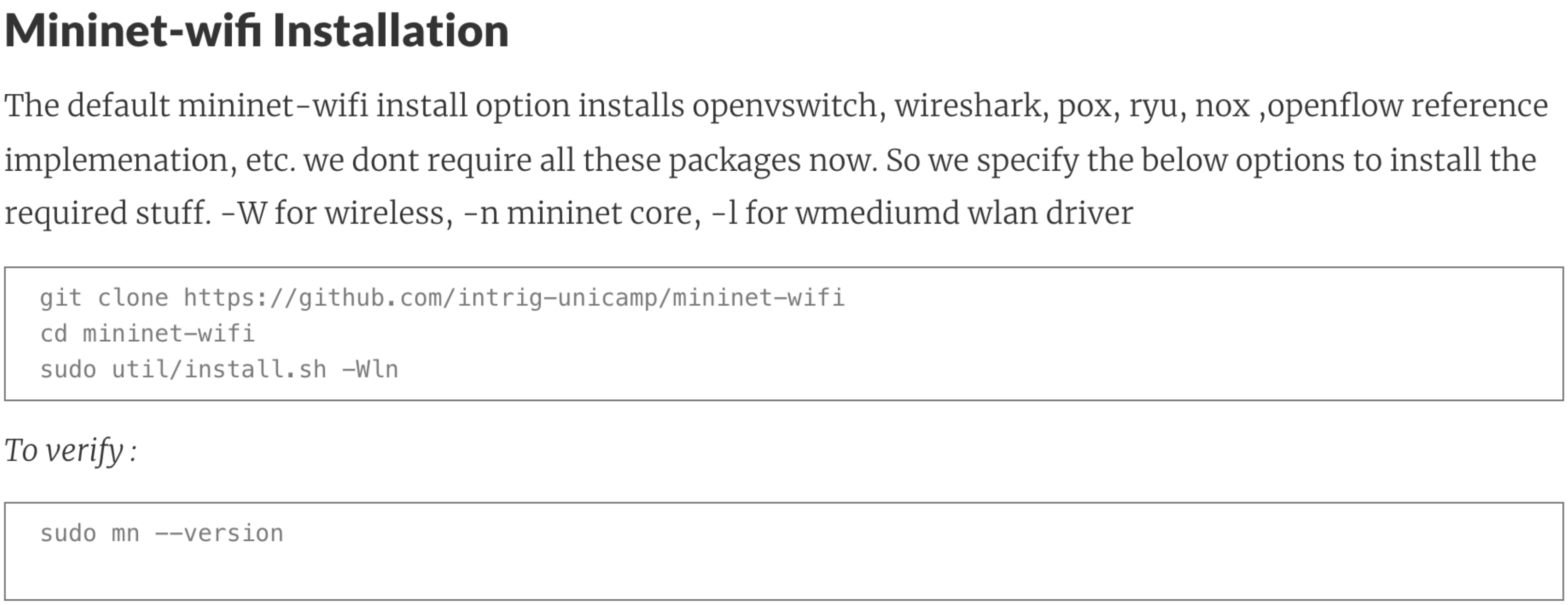
RYU installation

```
sudo pip install ryu
```

To verify :

```
ryu-manager --version
```

Installation



Mininet-wifi Installation

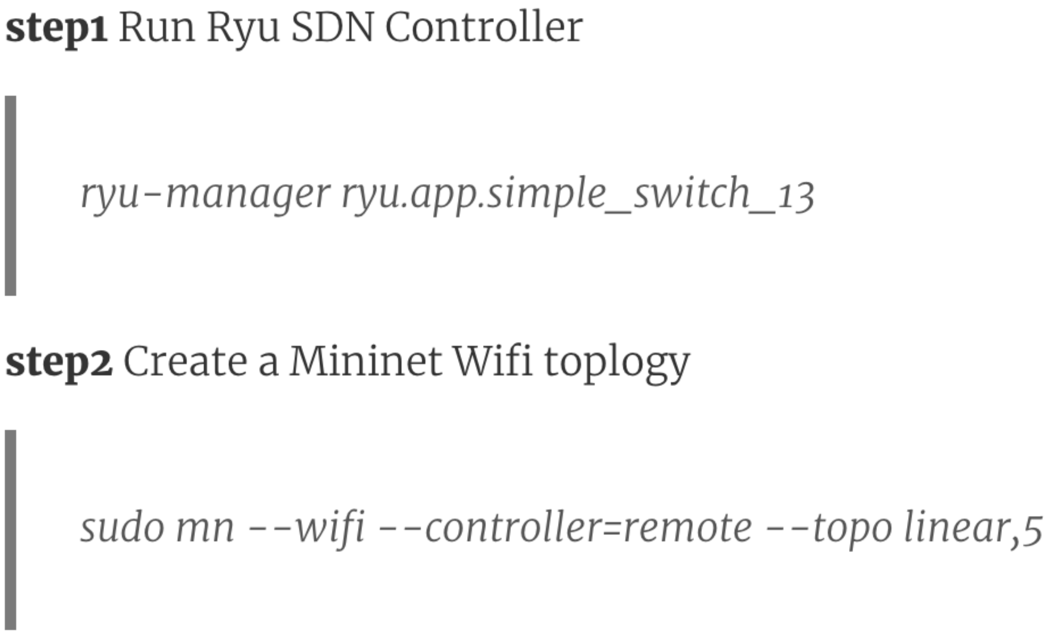
The default mininet-wifi install option installs openvswitch, wireshark, pox, ryu, nox ,openflow reference implemenation, etc. we dont require all these packages now. So we specify the below options to install the required stuff. -W for wireless, -n mininet core, -l for wmediumd wlan driver

```
git clone https://github.com/intrig-unicamp/mininet-wifi
cd mininet-wifi
sudo util/install.sh -Wln
```

To verify :

```
sudo mn --version
```

Quick Verification



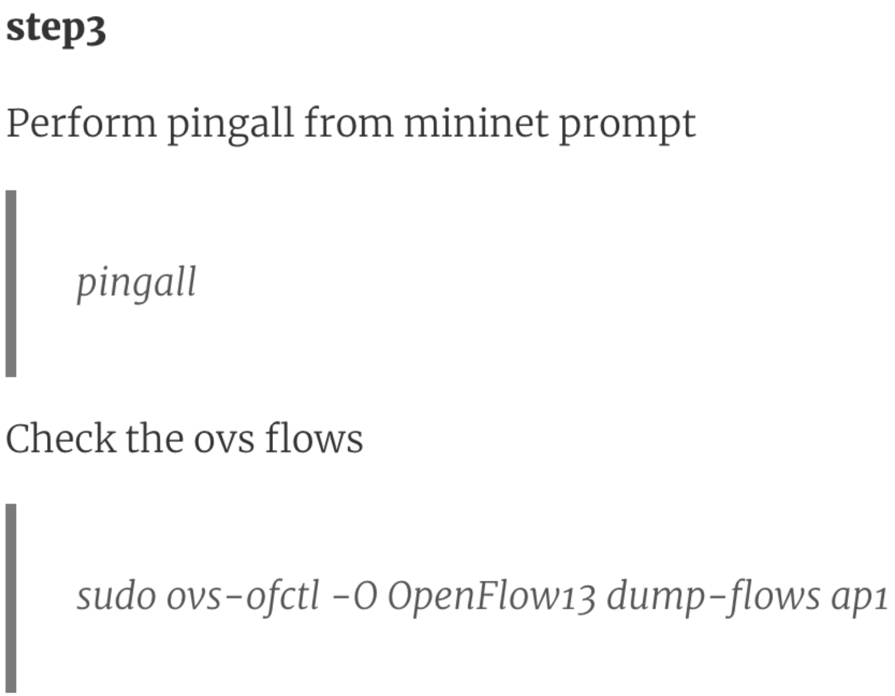
step1 Run Ryu SDN Controller

```
ryu-manager ryu.app.simple_switch_13
```

step2 Create a Mininet Wifi toplogy

```
sudo mn --wifi --controller=remote --topo linear,5
```

Quick Verification



step3

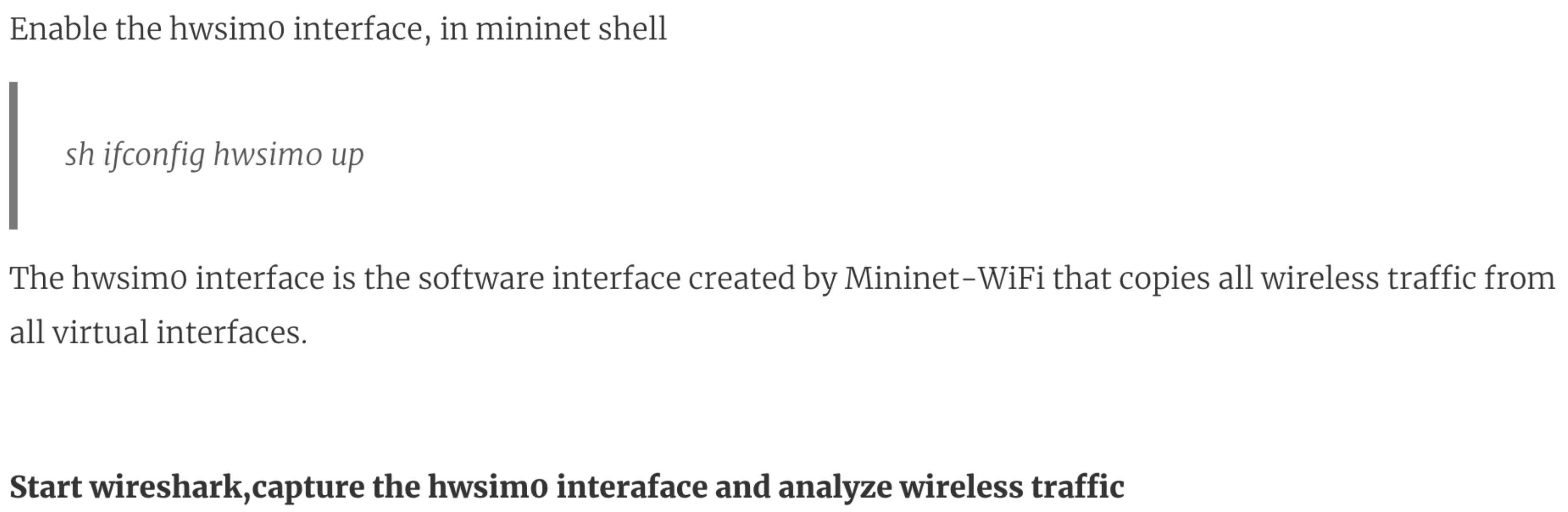
Perform pingall from mininet prompt

> *pingall*

Check the ovs flows

> *sudo ovs-ofctl -O OpenFlow13 dump-flows ap1*

Quick Verification



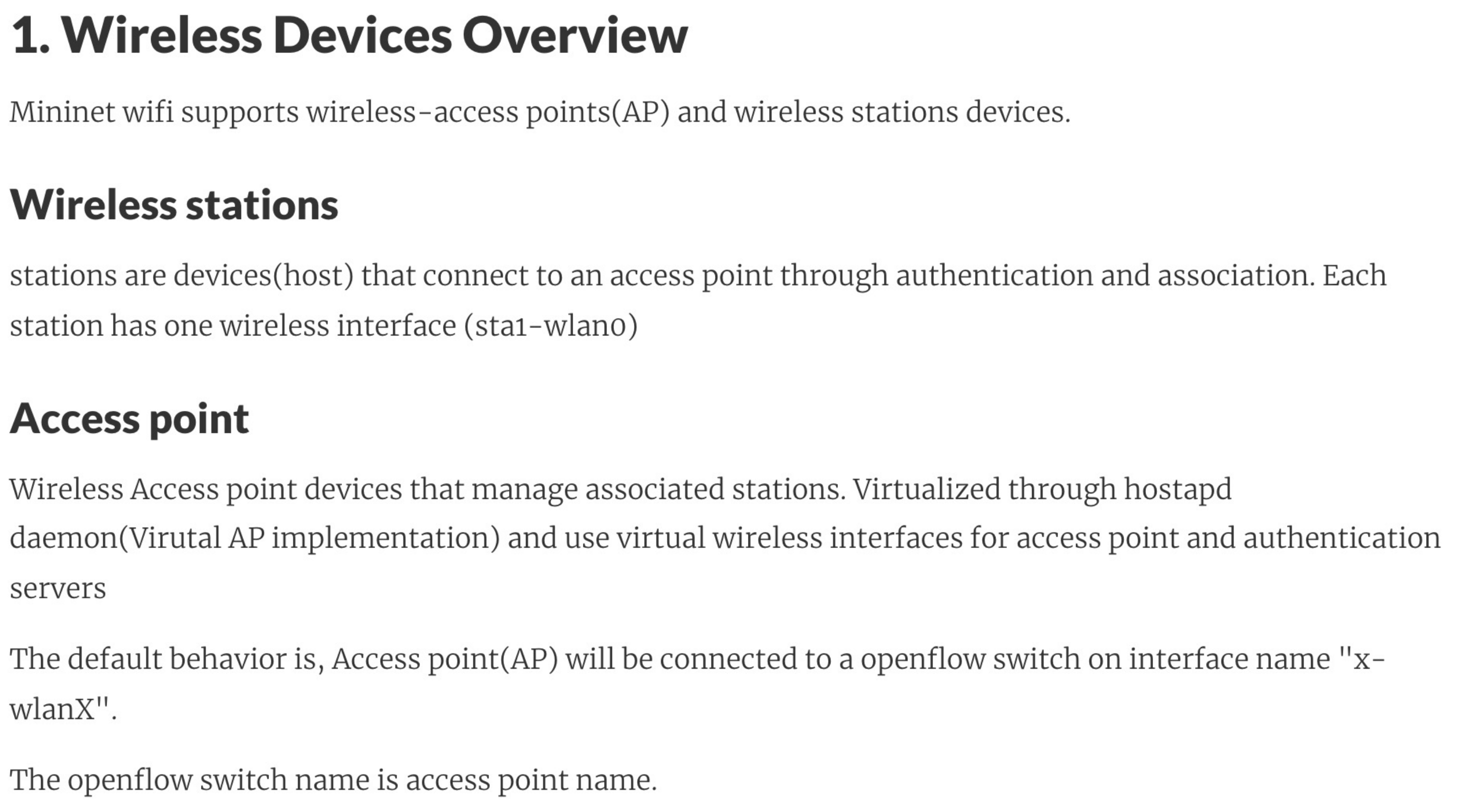
Enable the hwsim0 interface, in mininet shell

> *sh ifconfig hwsim0 up*

The hwsim0 interface is the software interface created by Mininet-WiFi that copies all wireless traffic from all virtual interfaces.

Start wireshark,capture the hwsim0 interaface and analyze wireless traffic

Device Overview



1. Wireless Devices Overview

Mininet wifi supports wireless-access points(AP) and wireless stations devices.

Wireless stations

stations are devices(host) that connect to an access point through authentication and association. Each station has one wireless interface (sta1-wlan0)

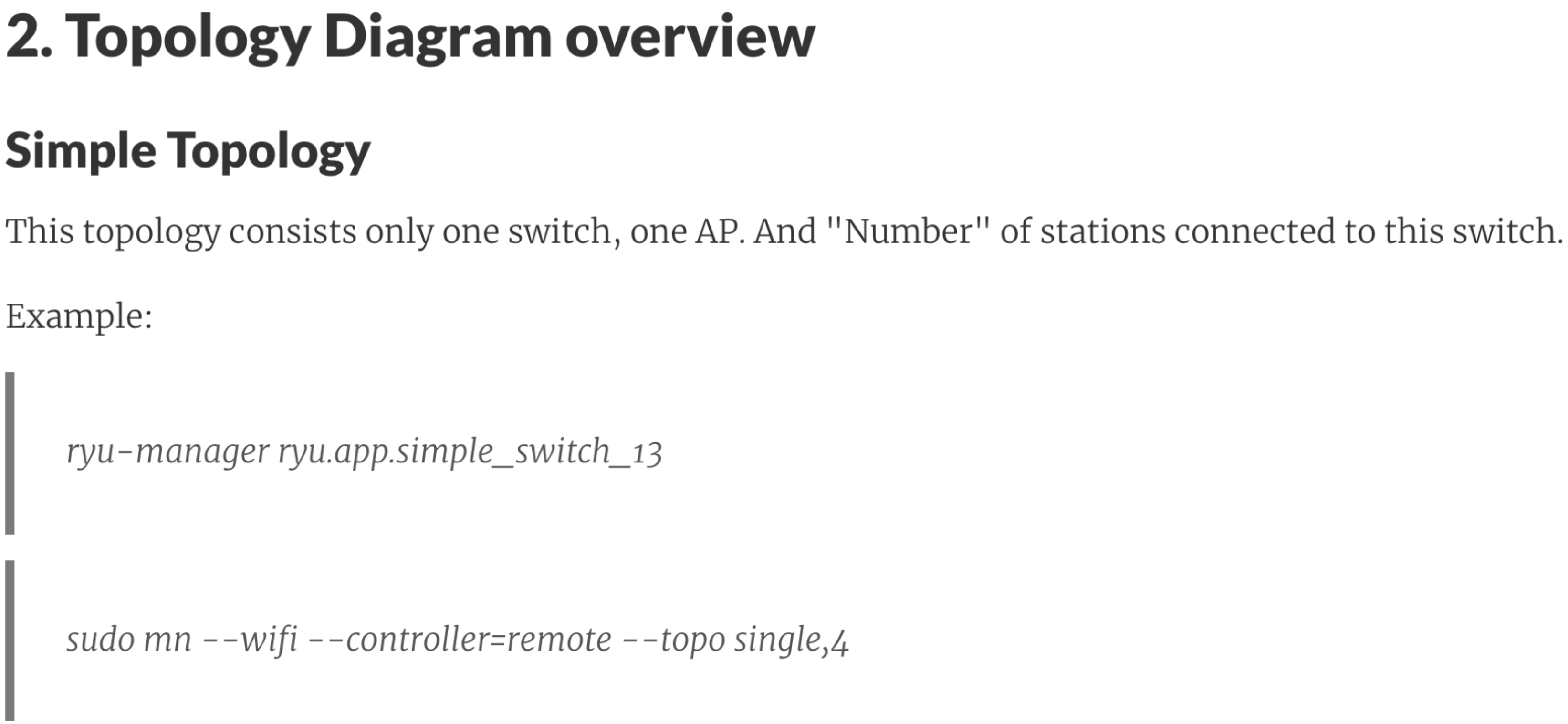
Access point

Wireless Access point devices that manage associated stations. Virtualized through hostapd daemon(Virutal AP implementation) and use virtual wireless interfaces for access point and authentication servers

The default behavior is, Access point(AP) will be connected to a openflow switch on interface name "x-wlanX".

The openflow switch name is access point name.

Device Overview



2. Topology Diagram overview

Simple Topology

This topology consists only one switch, one AP. And "Number" of stations connected to this switch.

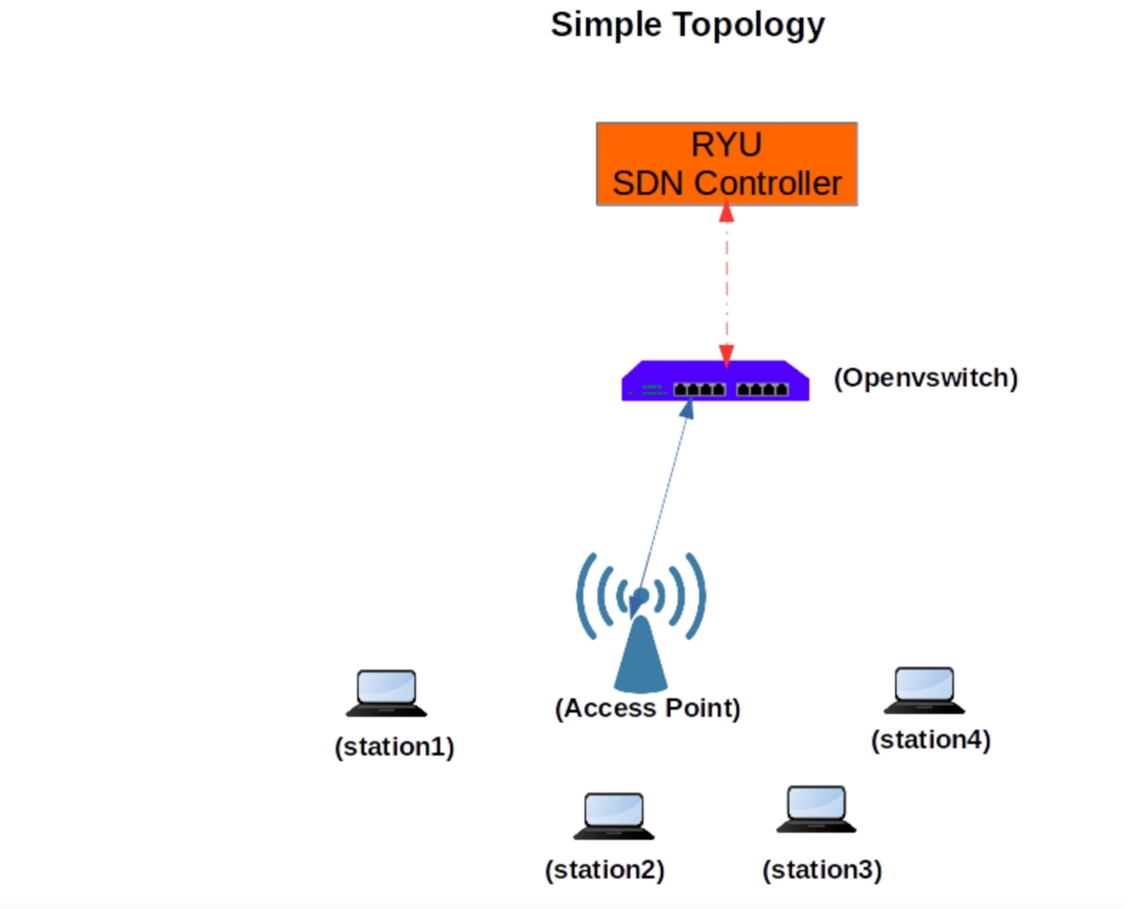
Example:

```
ryu-manager ryu.app.simple_switch_13
```

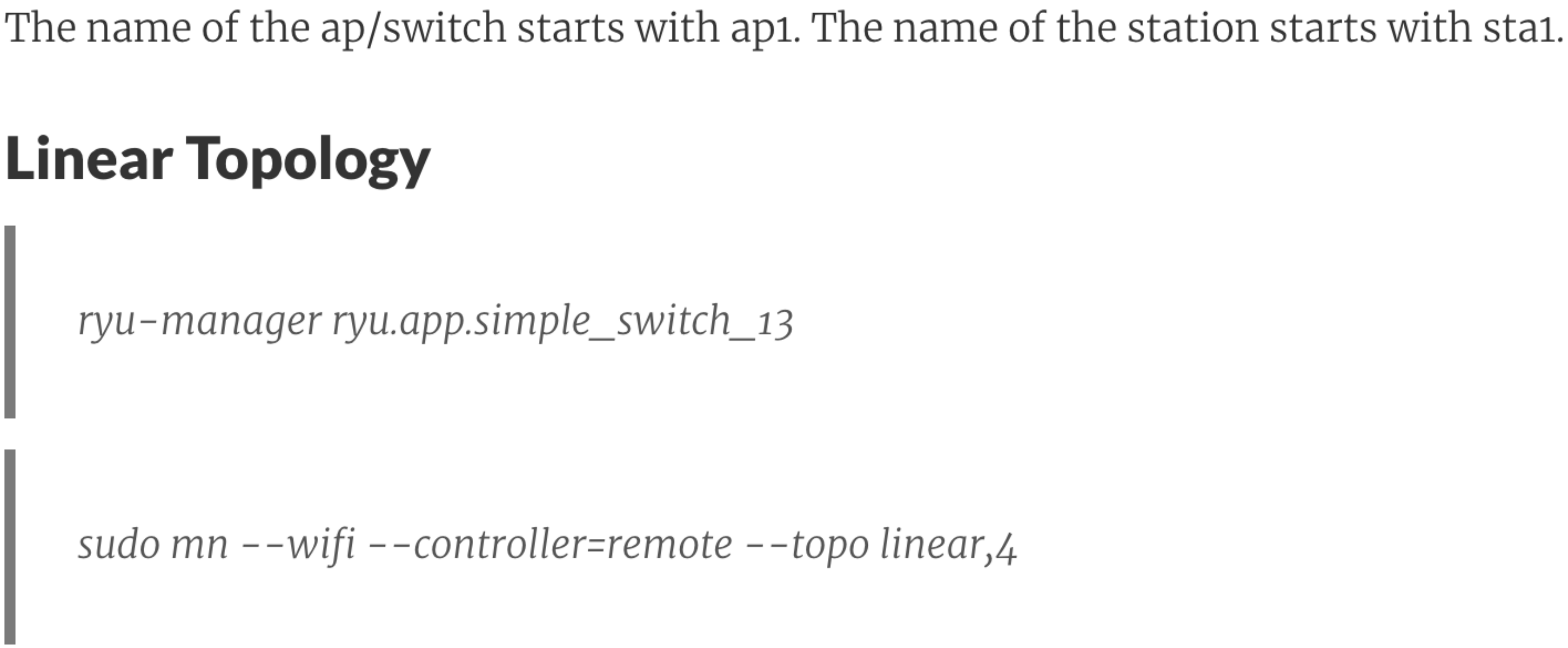
```
sudo mn --wifi --controller=remote --topo single,4
```

Device Overview

Here 4 Mobile stations connected to one AP.



Device Overview



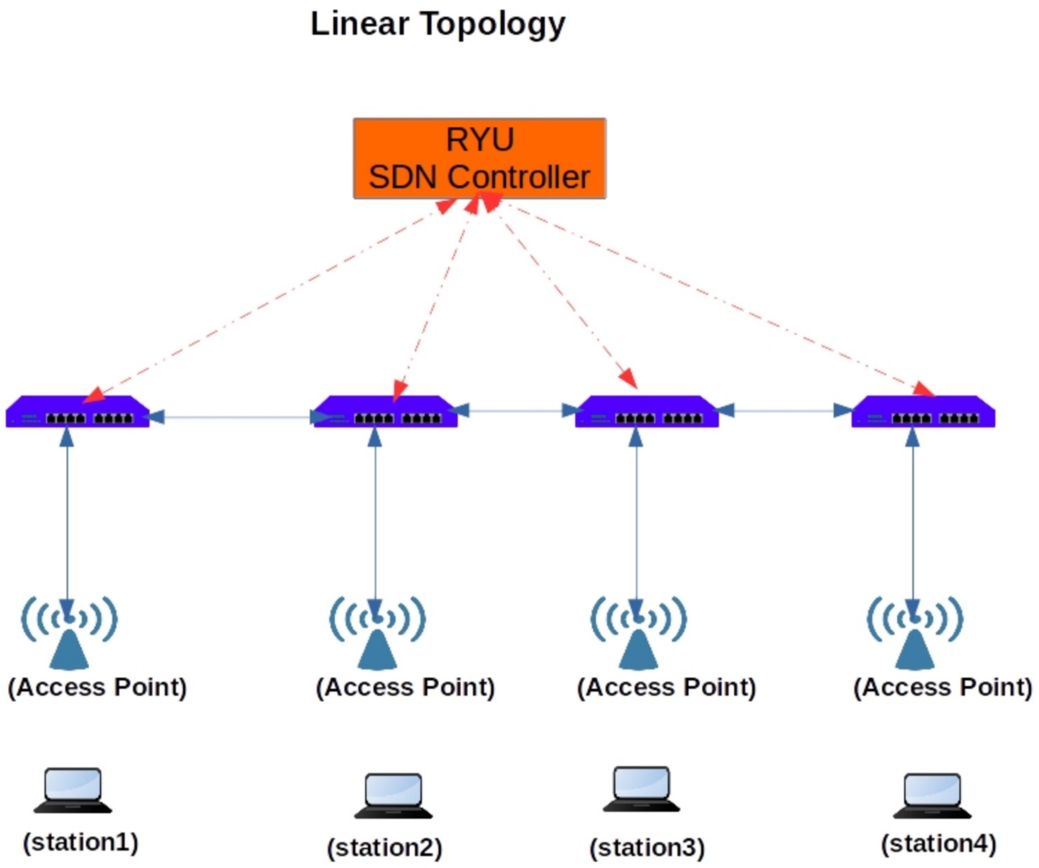
The name of the ap/switch starts with ap1. The name of the station starts with sta1.

Linear Topology

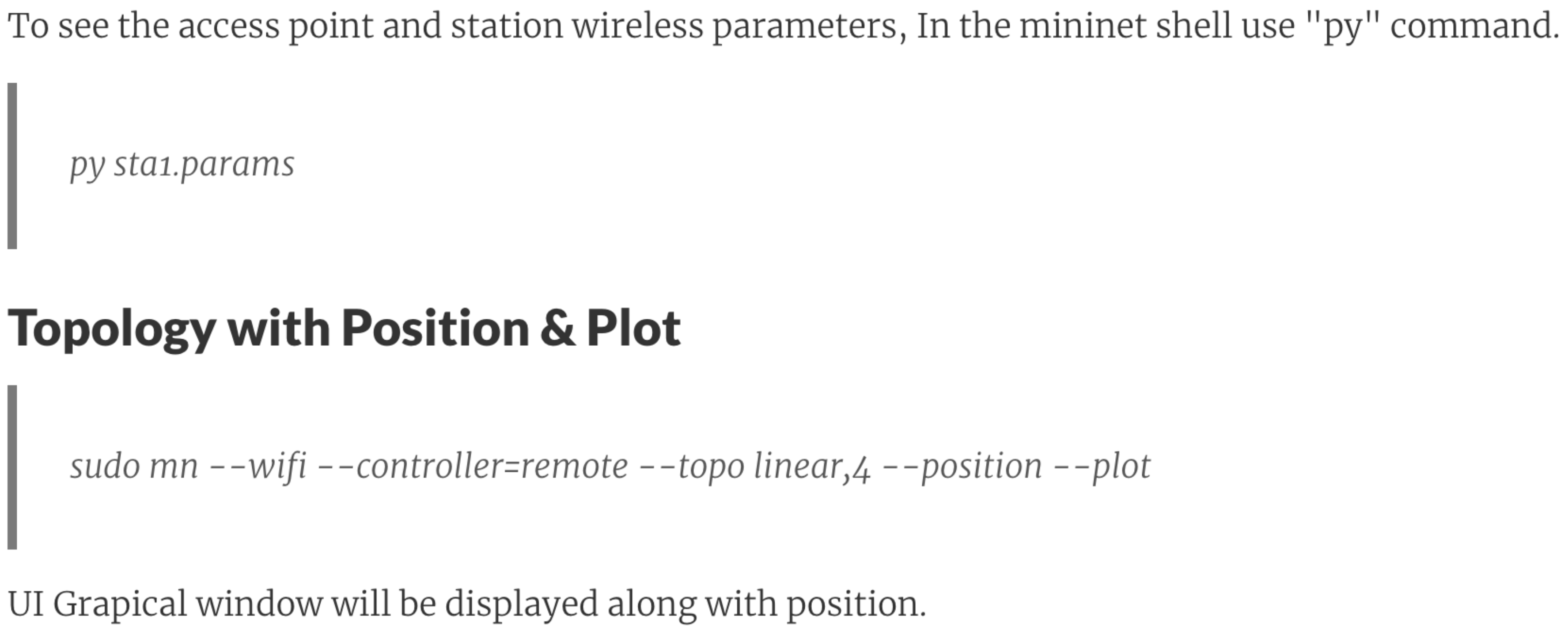
```
ryu-manager ryu.app.simple_switch_13
```

```
sudo mn --wifi --controller=remote --topo linear,4
```

Device Overview



Device Overview



To see the access point and station wireless parameters, In the mininet shell use "py" command.

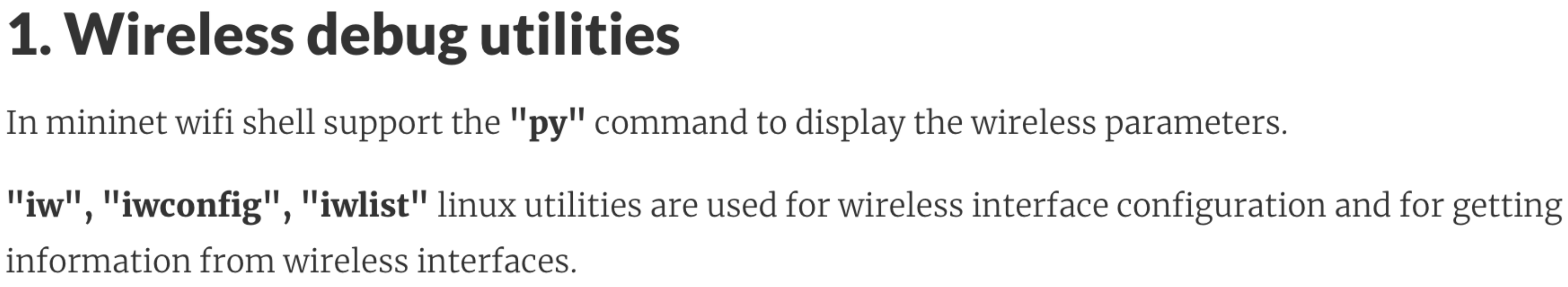
```
py sta1.params
```

Topology with Position & Plot

```
sudo mn --wifi --controller=remote --topo linear,4 --position --plot
```

UI Grapical window will be displayed along with position.

Wireless Debug Utilities

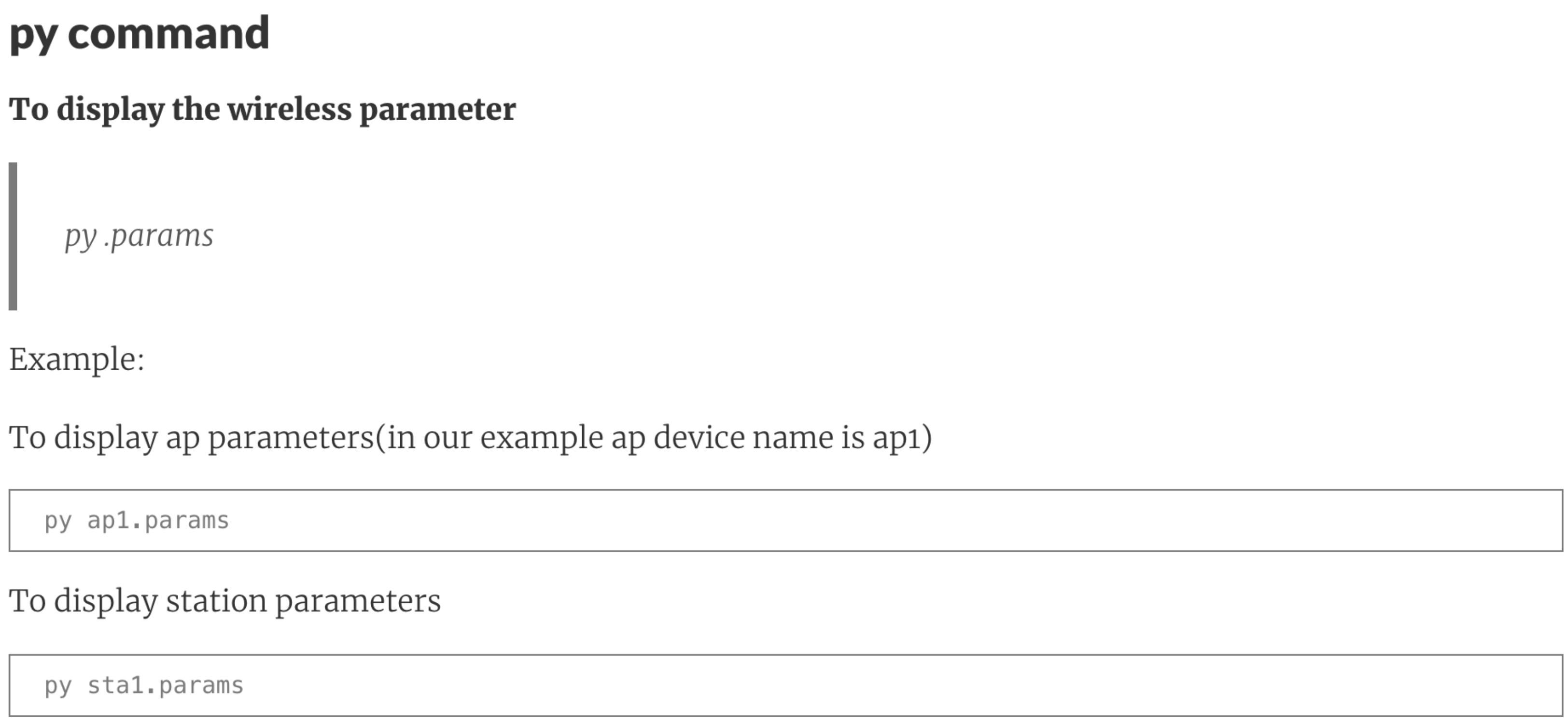


1. Wireless debug utilities

In mininet wifi shell support the **"py"** command to display the wireless parameters.

"iw", "iwconfig", "iwlist" linux utilities are used for wireless interface configuration and for getting information from wireless interfaces.

Wireless Debug Utilities



py command

To display the wireless parameter

> *py .params*

Example:

To display ap parameters(in our example ap device name is ap1)

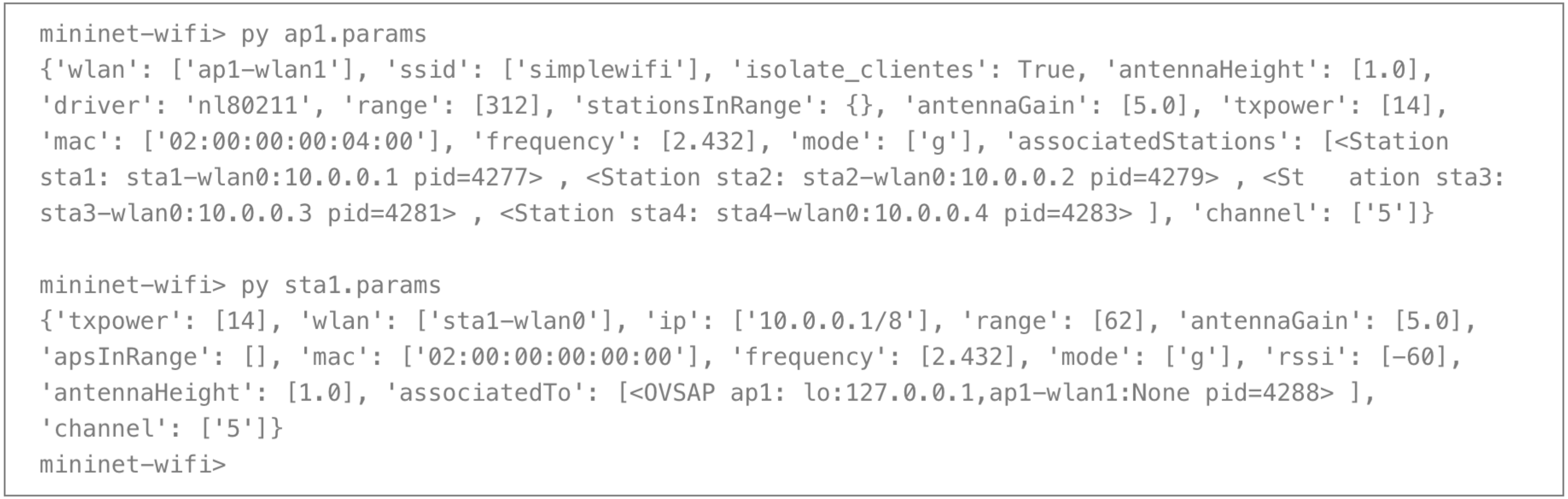
```
py ap1.params
```

To display station parameters

```
py sta1.params
```

Wireless Debug Utilities

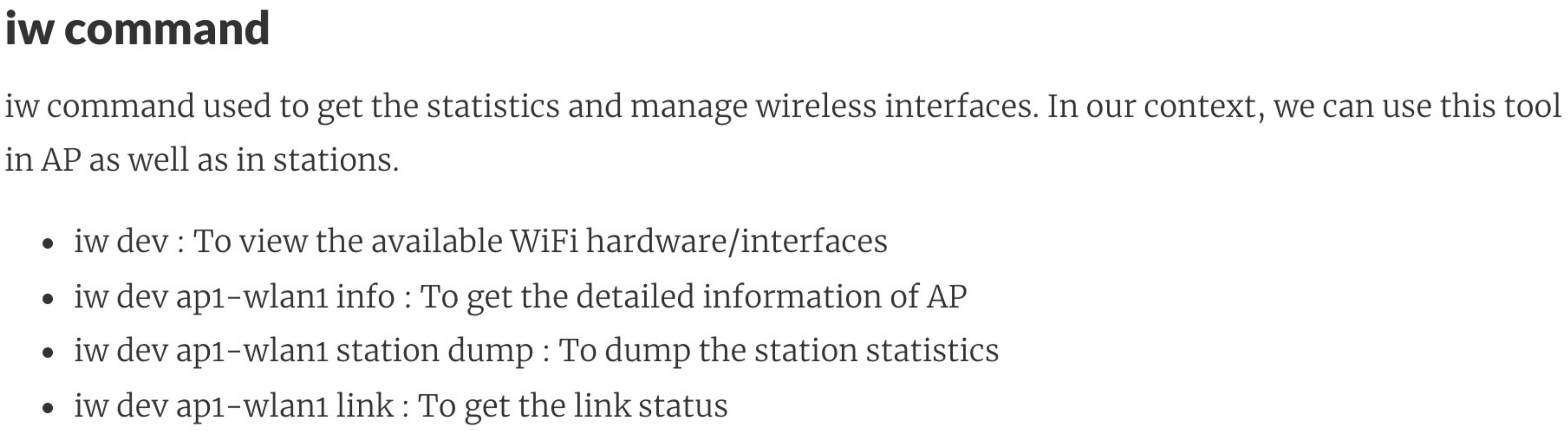
Examples:



```
mininet-wifi> py ap1.params
{'wlan': ['ap1-wlan1'], 'ssid': ['simplewifi'], 'isolate_clientes': True, 'antennaHeight': [1.0],
'driver': 'nl80211', 'range': [312], 'stationsInRange': {}, 'antennaGain': [5.0], 'txpower': [14],
'mac': ['02:00:00:00:04:00'], 'frequency': [2.432], 'mode': ['g'], 'associatedStations': [<Station
sta1: sta1-wlan0:10.0.0.1 pid=4277> , <Station sta2: sta2-wlan0:10.0.0.2 pid=4279> , <St   ation sta3:
sta3-wlan0:10.0.0.3 pid=4281> , <Station sta4: sta4-wlan0:10.0.0.4 pid=4283> ], 'channel': ['5']}

mininet-wifi> py sta1.params
{'txpower': [14], 'wlan': ['sta1-wlan0'], 'ip': ['10.0.0.1/8'], 'range': [62], 'antennaGain': [5.0],
'apsInRange': [], 'mac': ['02:00:00:00:00:00'], 'frequency': [2.432], 'mode': ['g'], 'rssi': [-60],
'antennaHeight': [1.0], 'associatedTo': [<OVSAP ap1: lo:127.0.0.1,ap1-wlan1:None pid=4288> ],
'channel': ['5']}
mininet-wifi>
```

Wireless Debug Utilities

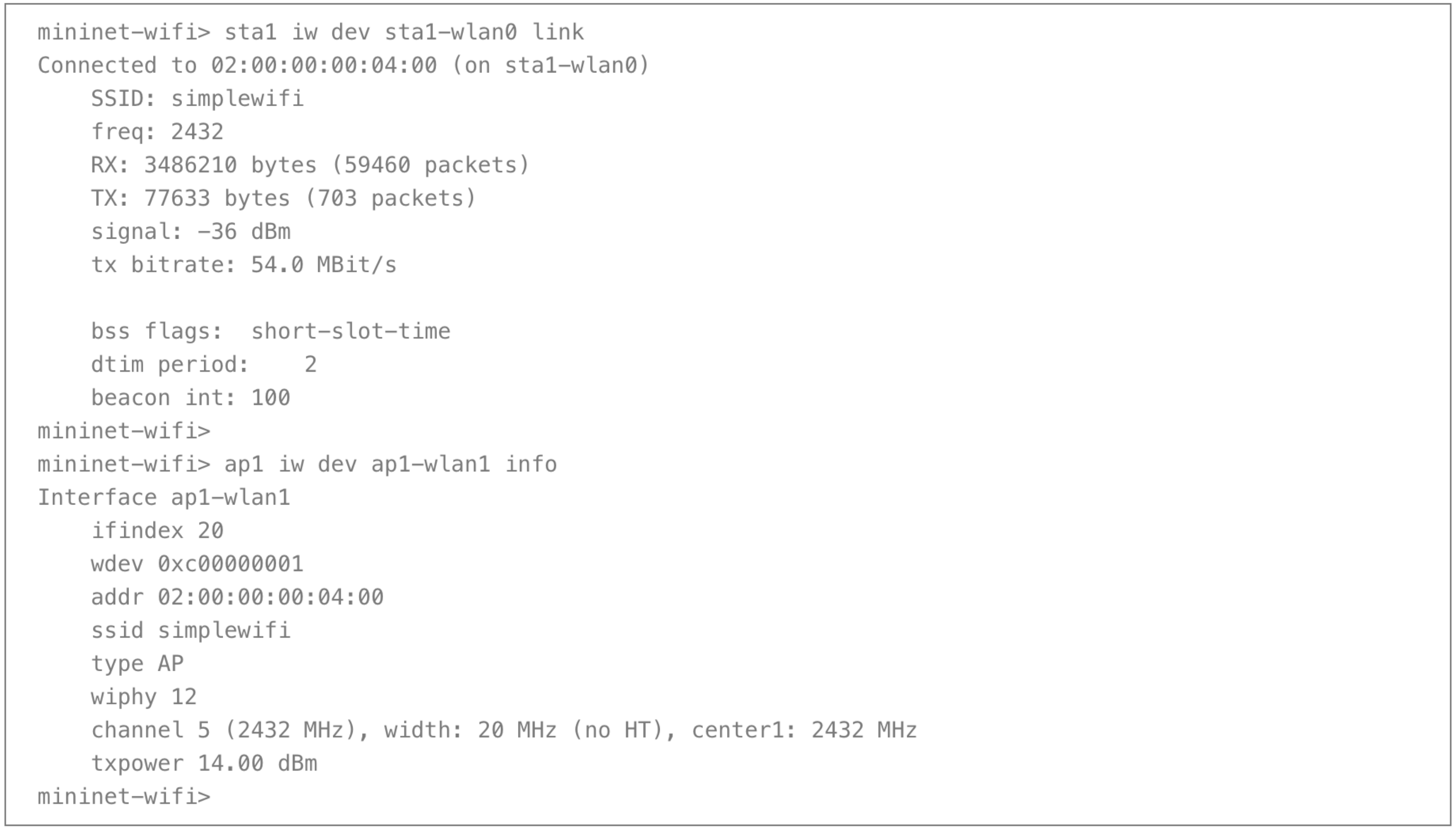


iw command

iw command used to get the statistics and manage wireless interfaces. In our context, we can use this tool in AP as well as in stations.

- iw dev : To view the available WiFi hardware/interfaces
- iw dev ap1-wlan1 info : To get the detailed information of AP
- iw dev ap1-wlan1 station dump : To dump the station statistics
- iw dev ap1-wlan1 link : To get the link status

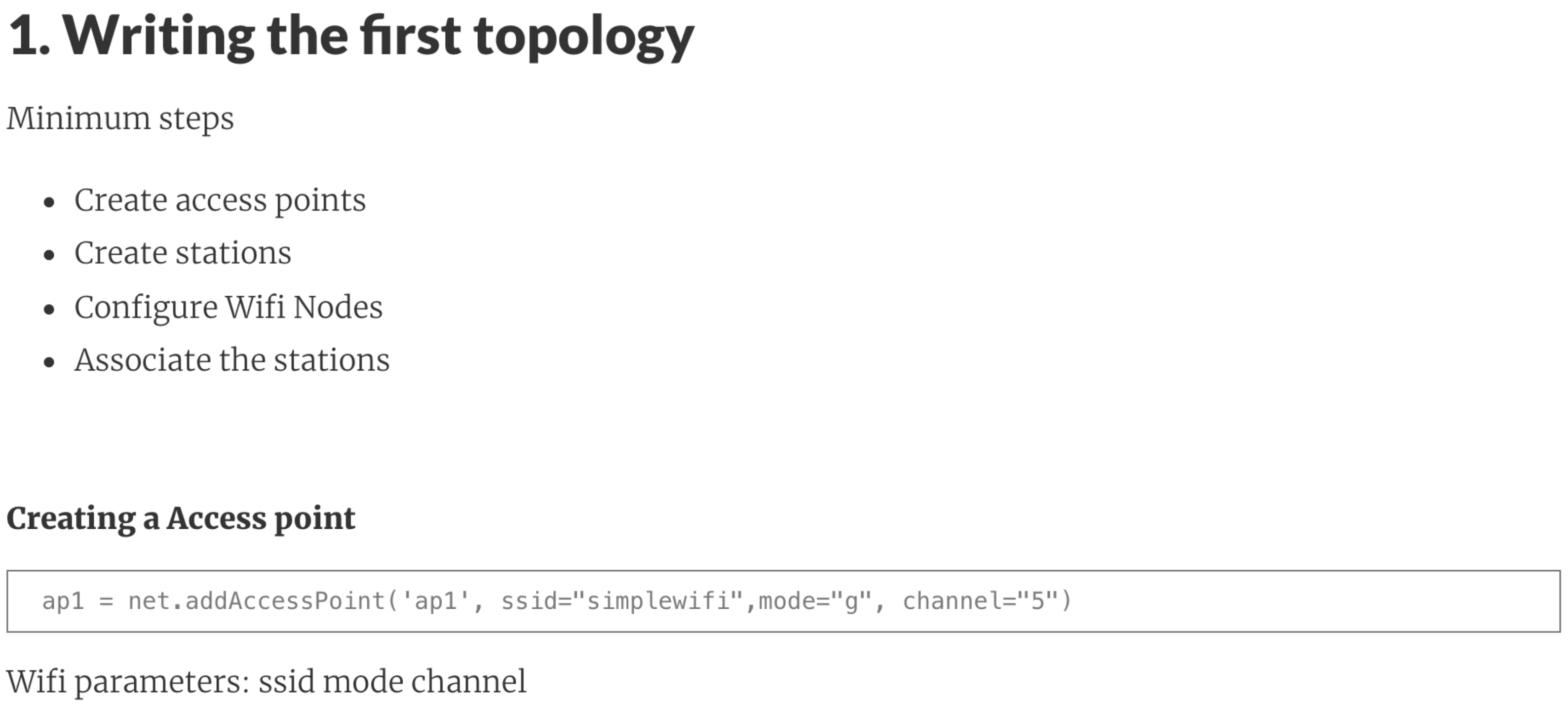
Wireless Debug Utilities



```
mininet-wifi> sta1 iw dev sta1-wlan0 link
Connected to 02:00:00:00:04:00 (on sta1-wlan0)
    SSID: simplewifi
    freq: 2432
    RX: 3486210 bytes (59460 packets)
    TX: 77633 bytes (703 packets)
    signal: -36 dBm
    tx bitrate: 54.0 MBit/s

    bss flags:  short-slot-time
    dtim period:    2
    beacon int: 100
mininet-wifi>
mininet-wifi> ap1 iw dev ap1-wlan1 info
Interface ap1-wlan1
    ifindex 20
    wdev 0xc00000001
    addr 02:00:00:00:04:00
    ssid simplewifi
    type AP
    wiphy 12
    channel 5 (2432 MHz), width: 20 MHz (no HT), center1: 2432 MHz
    txpower 14.00 dBm
mininet-wifi>
```

Custom Topology



1. Writing the first topology

Minimum steps

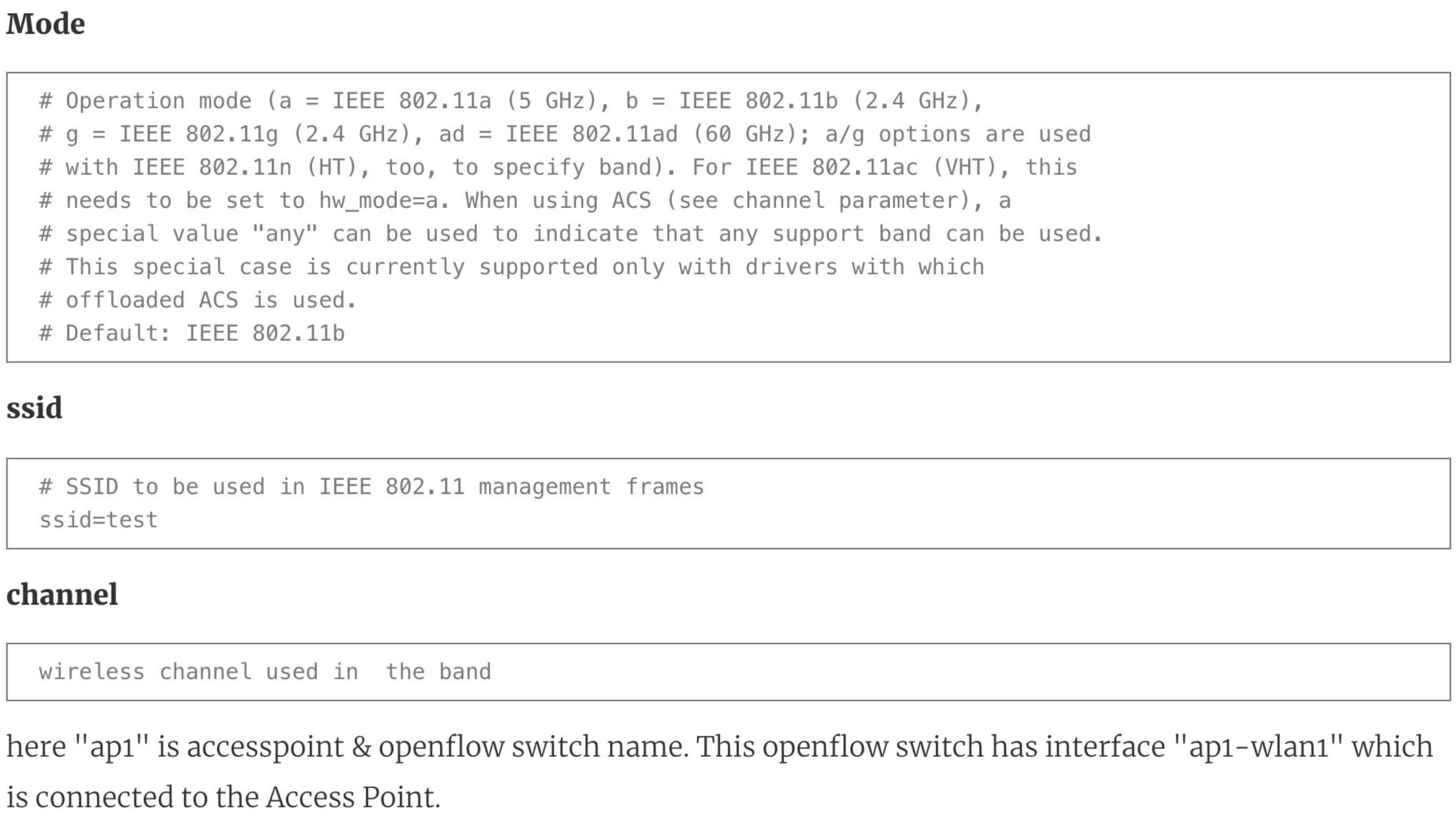
- Create access points
- Create stations
- Configure Wifi Nodes
- Associate the stations

Creating a Access point

```
ap1 = net.addAccessPoint('ap1', ssid="simplewifi",mode="g", channel="5")
```

Wifi parameters: ssid mode channel

Custom Topology



Mode

```
# Operation mode (a = IEEE 802.11a (5 GHz), b = IEEE 802.11b (2.4 GHz),
# g = IEEE 802.11g (2.4 GHz), ad = IEEE 802.11ad (60 GHz); a/g options are used
# with IEEE 802.11n (HT), too, to specify band). For IEEE 802.11ac (VHT), this
# needs to be set to hw_mode=a. When using ACS (see channel parameter), a
# special value "any" can be used to indicate that any support band can be used.
# This special case is currently supported only with drivers with which
# offloaded ACS is used.
# Default: IEEE 802.11b
```

ssid

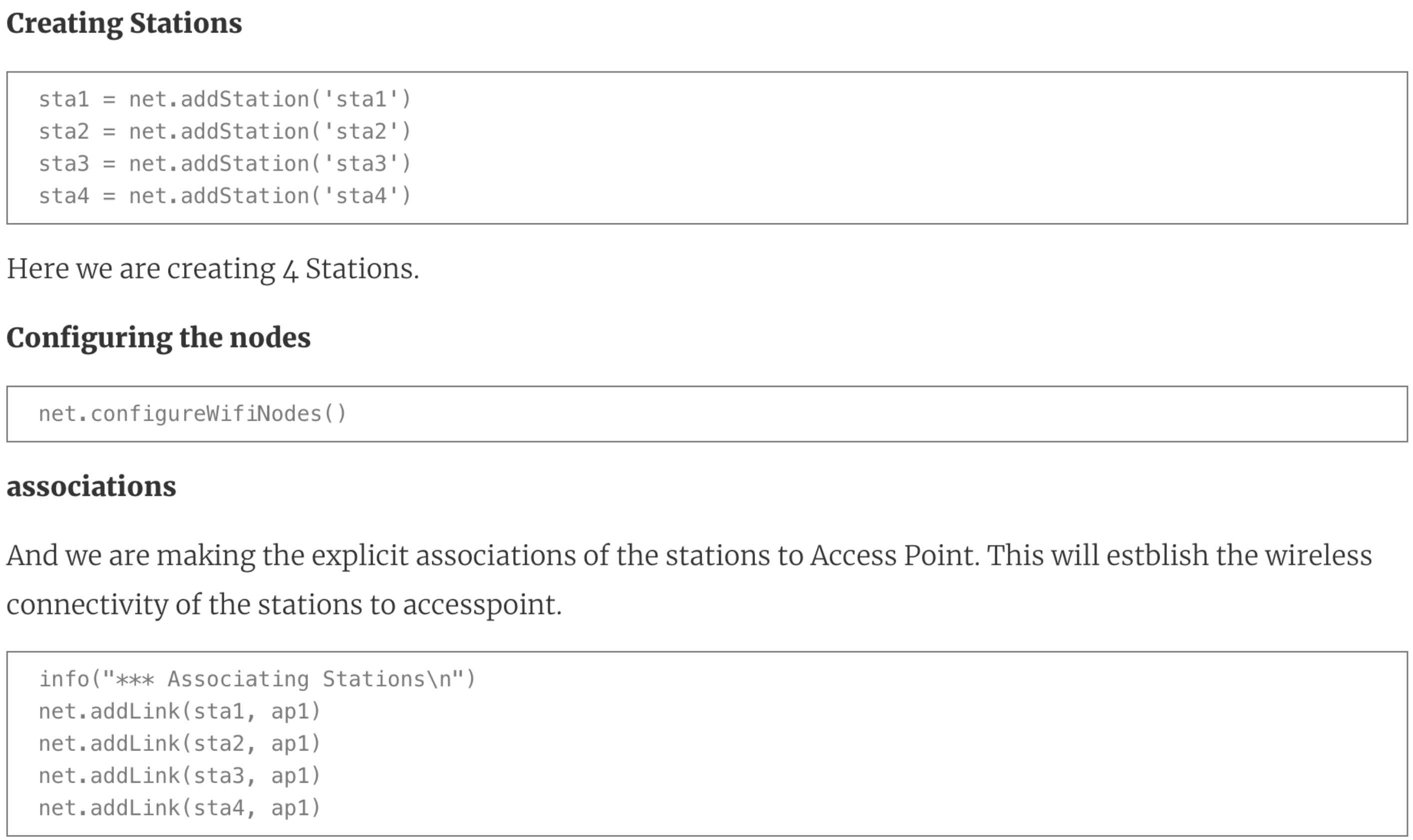
```
# SSID to be used in IEEE 802.11 management frames
ssid=test
```

channel

```
wireless channel used in  the band
```

here "ap1" is accesspoint & openflow switch name. This openflow switch has interface "ap1-wlan1" which is connected to the Access Point.

Custom Topology



Creating Stations

```
sta1 = net.addStation('sta1')
sta2 = net.addStation('sta2')
sta3 = net.addStation('sta3')
sta4 = net.addStation('sta4')
```

Here we are creating 4 Stations.

Configuring the nodes

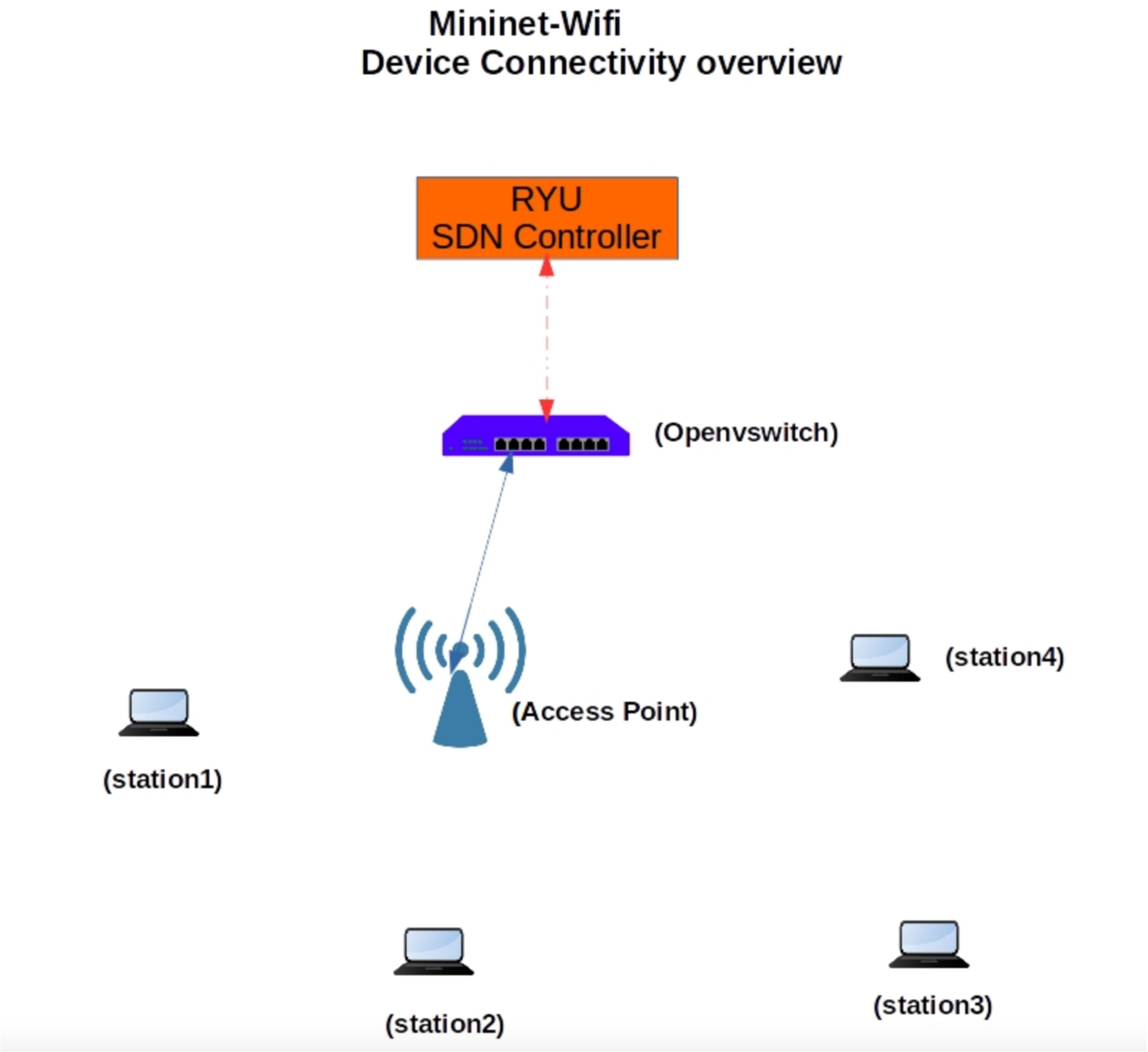
```
net.configureWifiNodes()
```

associations

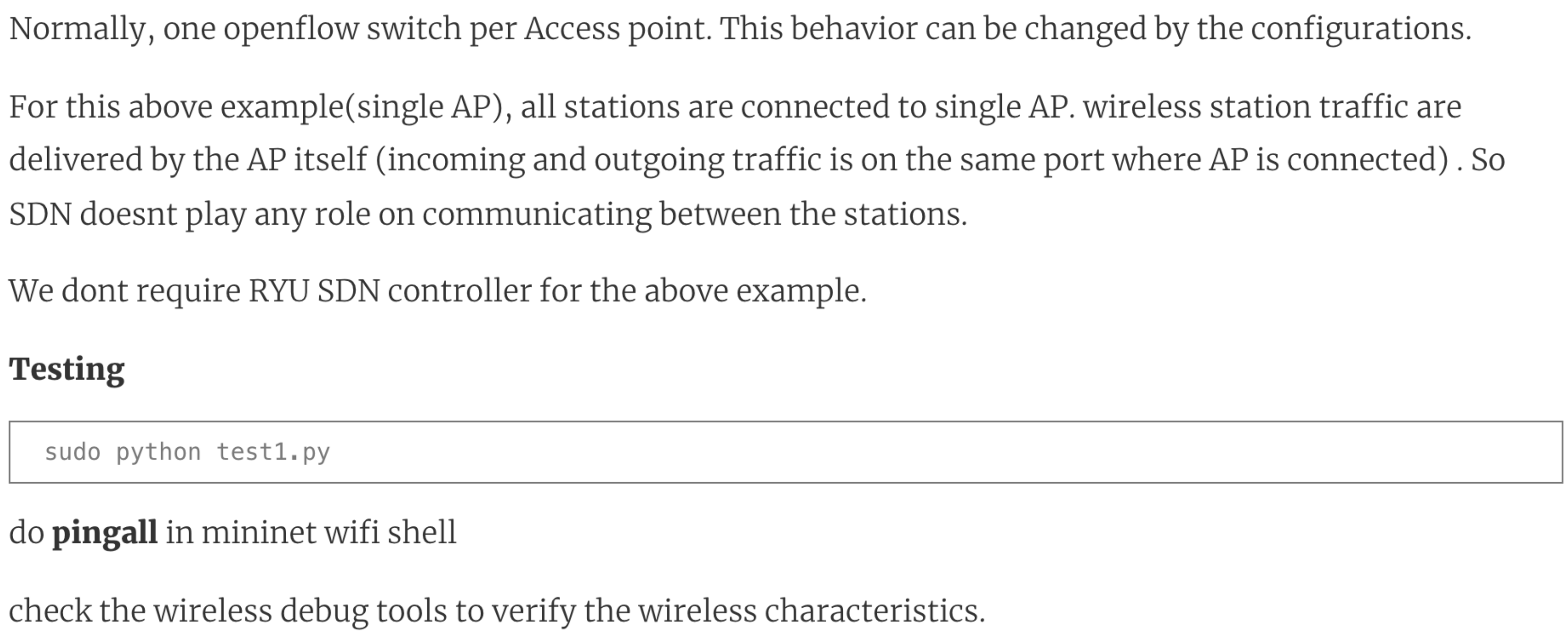
And we are making the explicit associations of the stations to Access Point. This will estblish the wireless connectivity of the stations to accesspoint.

```
info("*** Associating Stations\n")
net.addLink(sta1, ap1)
net.addLink(sta2, ap1)
net.addLink(sta3, ap1)
net.addLink(sta4, ap1)
```

Custom Topology



Custom Topology



Normally, one openflow switch per Access point. This behavior can be changed by the configurations.

For this above example(single AP), all stations are connected to single AP. wireless station traffic are delivered by the AP itself (incoming and outgoing traffic is on the same port where AP is connected) . So SDN doesnt play any role on communicating between the stations.

We dont require RYU SDN controller for the above example.

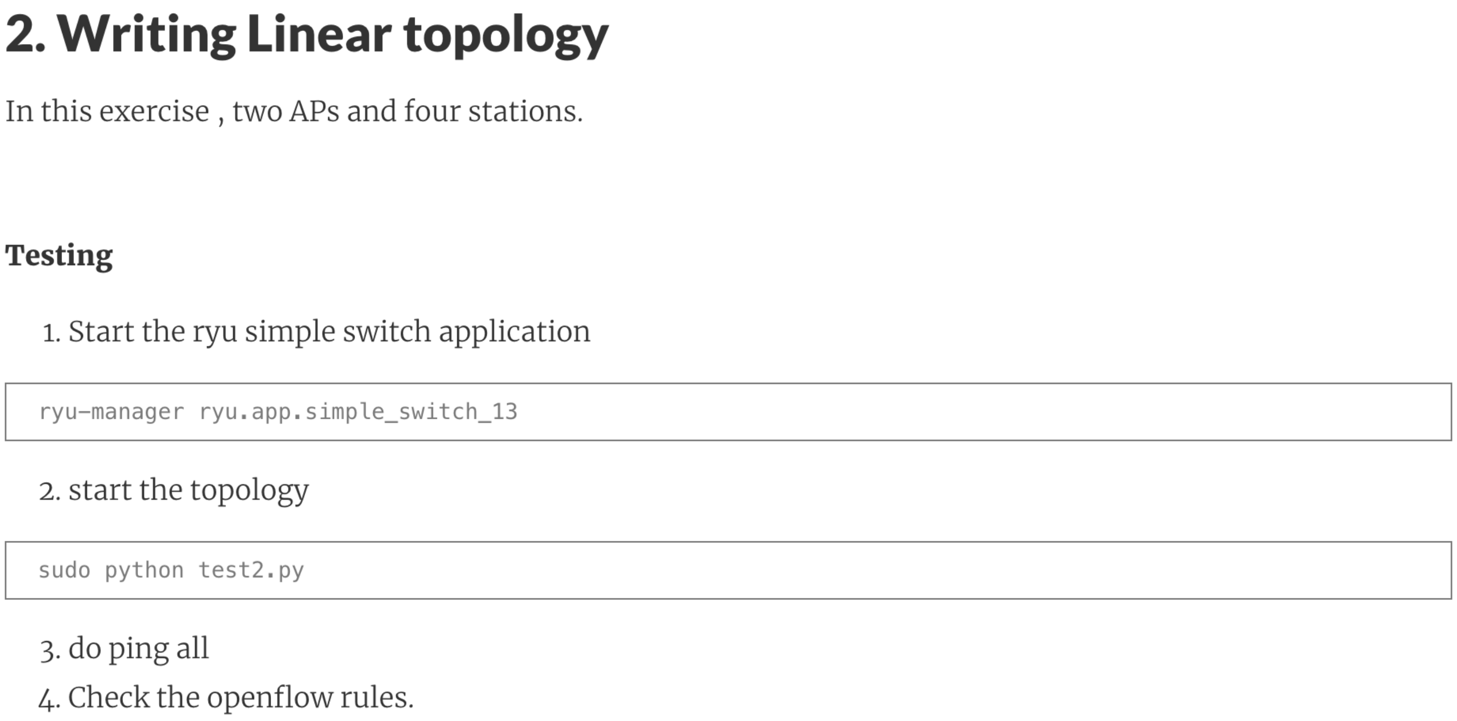
Testing

```
sudo python test1.py
```

do **pingall** in mininet wifi shell

check the wireless debug tools to verify the wireless characteristics.

Custom Topology



2. Writing Linear topology

In this exercise , two APs and four stations.

Testing

1. Start the ryu simple switch application

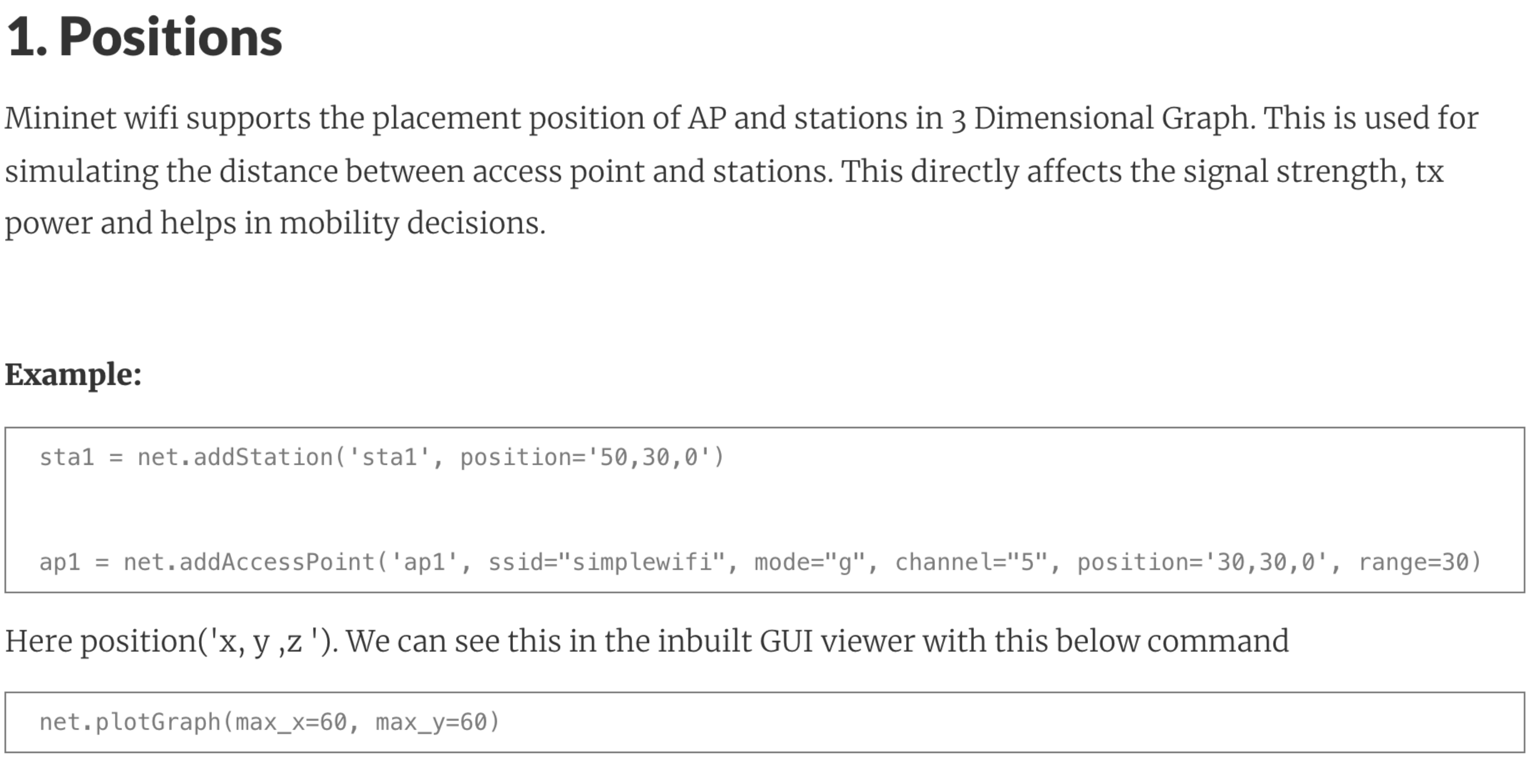
```
ryu-manager ryu.app.simple_switch_13
```

2. start the topology

```
sudo python test2.py
```

3. do ping all
4. Check the openflow rules.

Positions



1. Positions

Mininet wifi supports the placement position of AP and stations in 3 Dimensional Graph. This is used for simulating the distance between access point and stations. This directly affects the signal strength, tx power and helps in mobility decisions.

Example:

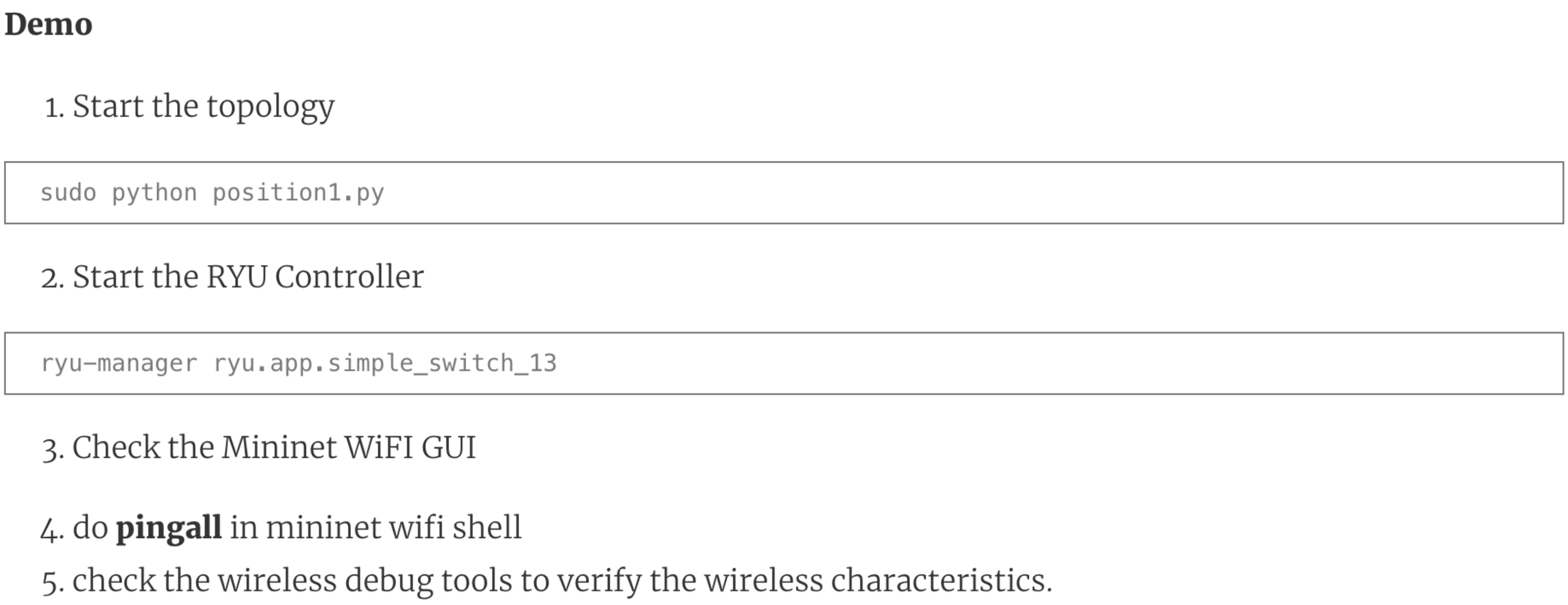
```
sta1 = net.addStation('sta1', position='50,30,0')


ap1 = net.addAccessPoint('ap1', ssid="simplewifi", mode="g", channel="5", position='30,30,0', range=30)
```

Here position('x, y ,z '). We can see this in the inbuilt GUI viewer with this below command

```
net.plotGraph(max_x=60, max_y=60)
```

Positions



Demo

1. Start the topology

```
sudo python position1.py
```

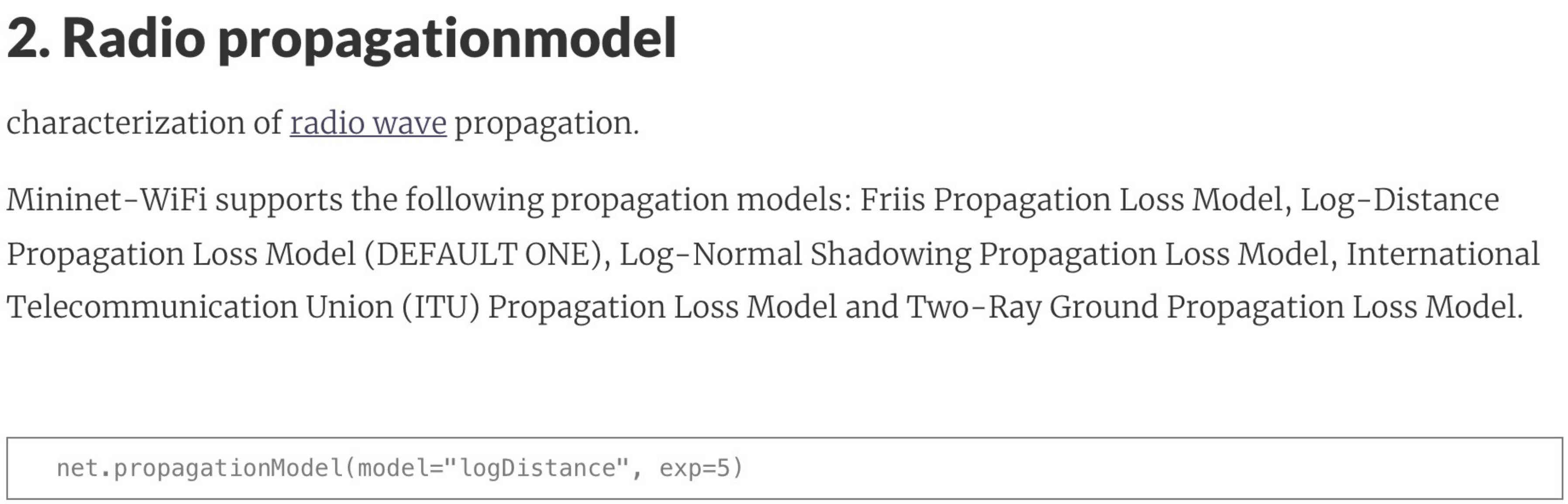
2. Start the RYU Controller

```
ryu-manager ryu.app.simple_switch_13
```

3. Check the Mininet WiFI GUI

4. do **pingall** in mininet wifi shell
5. check the wireless debug tools to verify the wireless characteristics.

Positions



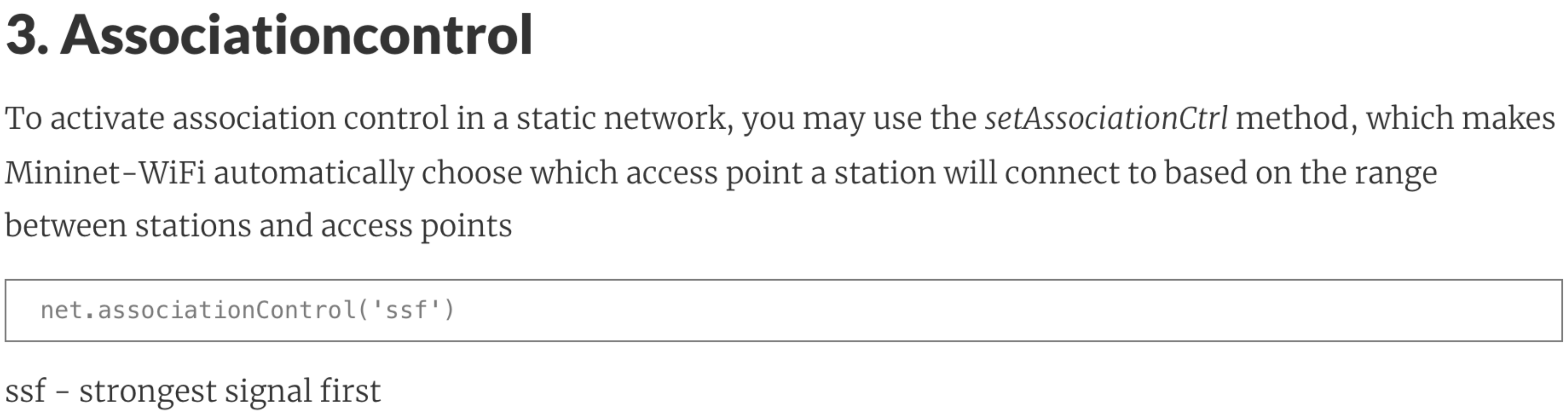
2. Radio propagationmodel

characterization of radio wave propagation.

Mininet-WiFi supports the following propagation models: Friis Propagation Loss Model, Log-Distance Propagation Loss Model (DEFAULT ONE), Log-Normal Shadowing Propagation Loss Model, International Telecommunication Union (ITU) Propagation Loss Model and Two-Ray Ground Propagation Loss Model.

```
net.propagationModel(model="logDistance", exp=5)
```

Positions



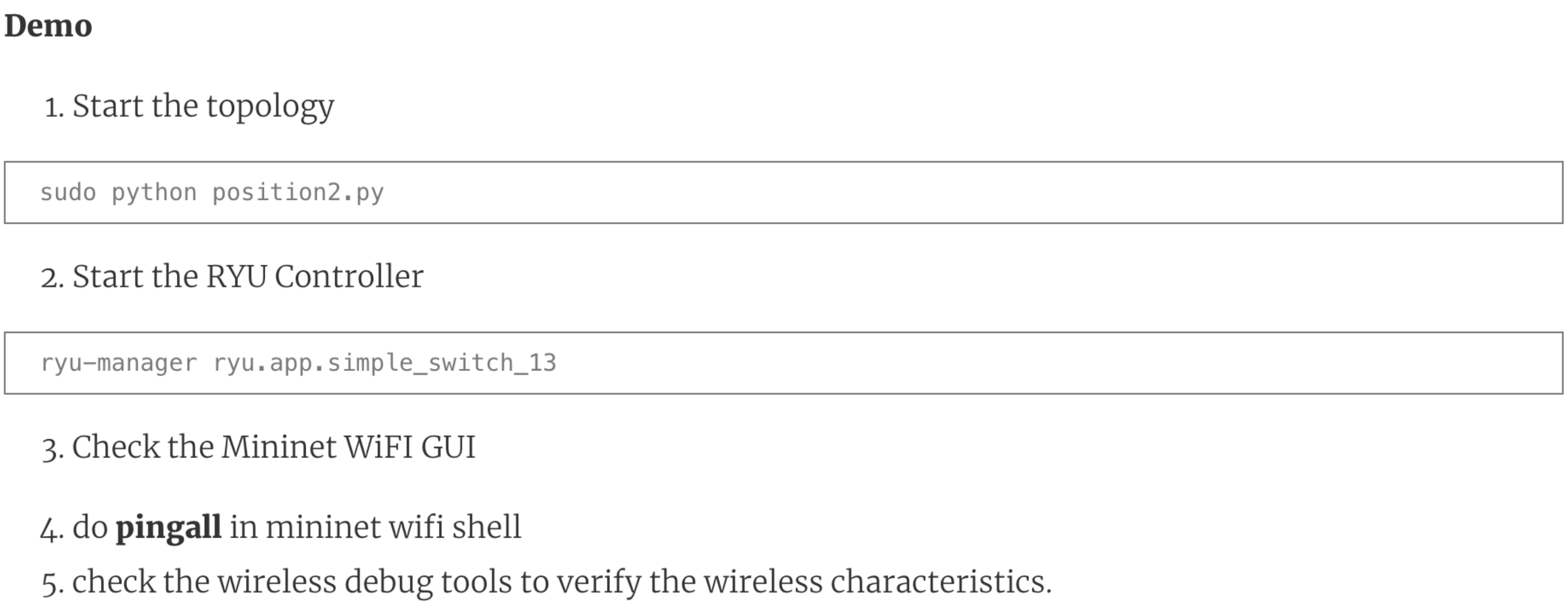
3. Associationcontrol

To activate association control in a static network, you may use the *setAssociationCtrl* method, which makes Mininet-WiFi automatically choose which access point a station will connect to based on the range between stations and access points

```
net.associationControl('ssf')
```

ssf - strongest signal first

Positions



Demo

1. Start the topology

```
sudo python position2.py
```

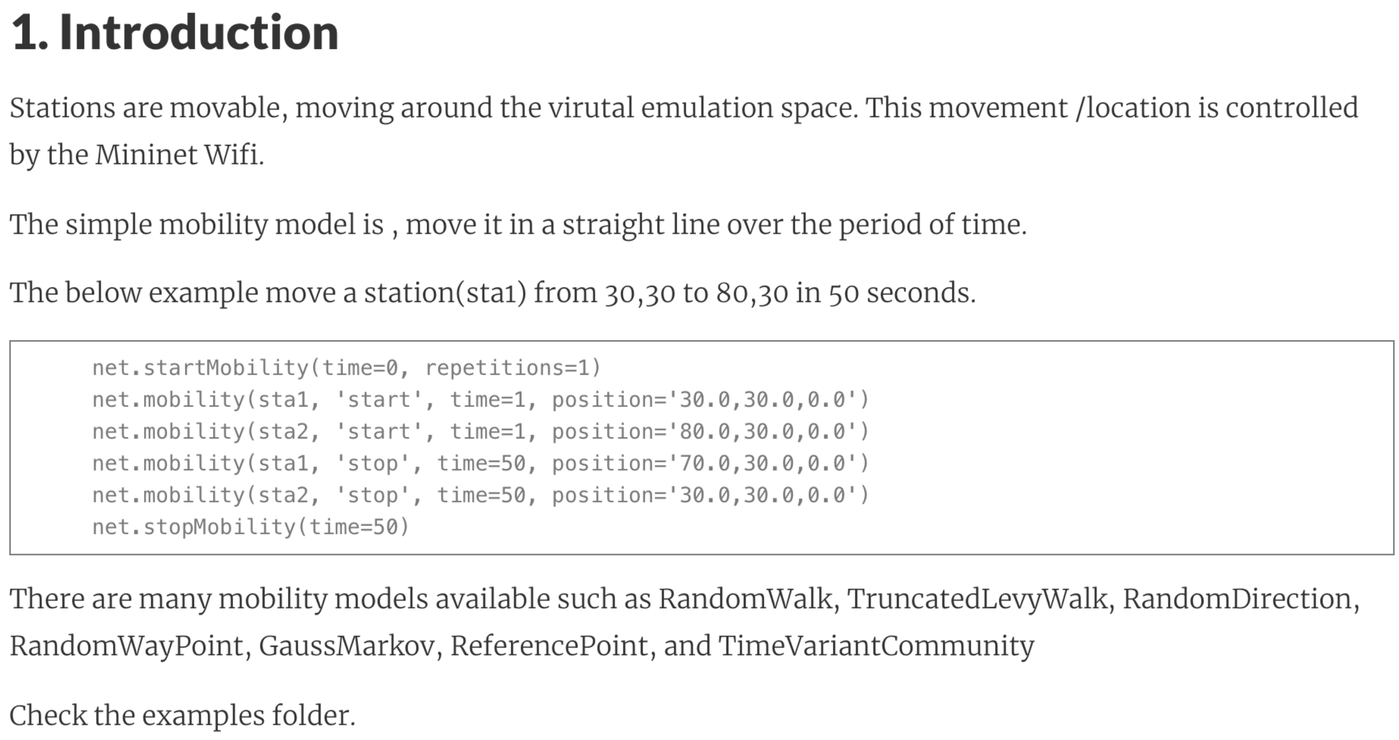
2. Start the RYU Controller

```
ryu-manager ryu.app.simple_switch_13
```

3. Check the Mininet WiFI GUI

4. do **pingall** in mininet wifi shell
5. check the wireless debug tools to verify the wireless characteristics.

Mobility



1. Introduction

Stations are movable, moving around the virutal emulation space. This movement /location is controlled by the Mininet Wifi.

The simple mobility model is , move it in a straight line over the period of time.

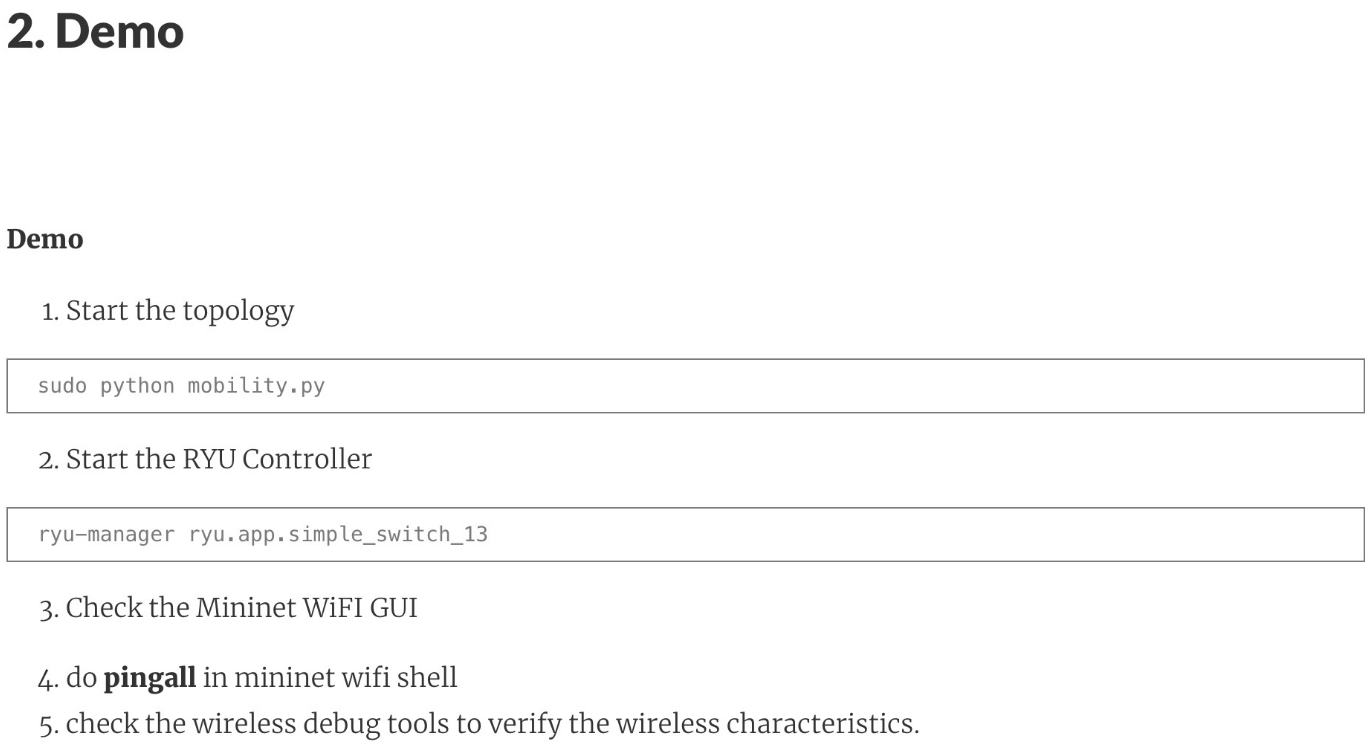
The below example move a station(sta1) from 30,30 to 80,30 in 50 seconds.

```
    net.startMobility(time=0, repetitions=1)
    net.mobility(sta1, 'start', time=1, position='30.0,30.0,0.0')
    net.mobility(sta2, 'start', time=1, position='80.0,30.0,0.0')
    net.mobility(sta1, 'stop', time=50, position='70.0,30.0,0.0')
    net.mobility(sta2, 'stop', time=50, position='30.0,30.0,0.0')
    net.stopMobility(time=50)
```

There are many mobility models available such as RandomWalk, TruncatedLevyWalk, RandomDirection, RandomWayPoint, GaussMarkov, ReferencePoint, and TimeVariantCommunity

Check the examples folder.

Mobility



2. Demo

Demo

1. Start the topology

```
sudo python mobility.py
```

2. Start the RYU Controller

```
ryu-manager ryu.app.simple_switch_13
```

3. Check the Mininet WiFI GUI

4. do **pingall** in mininet wifi shell
5. check the wireless debug tools to verify the wireless characteristics.