Project 1

TA： 陳怡伃
Email： adultfish17640336@gmail.com

Outline

- WHAT IS NS-3
- NS-3 CONCEPT
- SETUP NS-3 ENVIRONMENT
- PROJECT EXPLANATION

What is ns-3

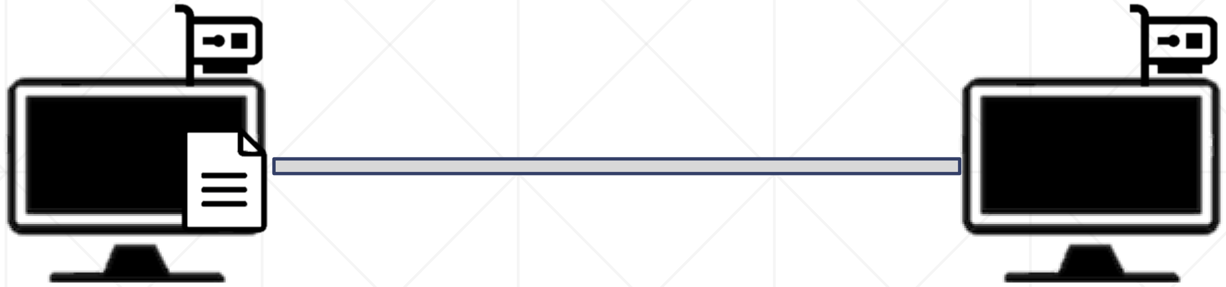
ns-3

- a series of **discrete event network simulators**
- primarily used in research and teaching
- built using **C++ and Python** with scripting capability

Real world V.S. ns-3

- Real world

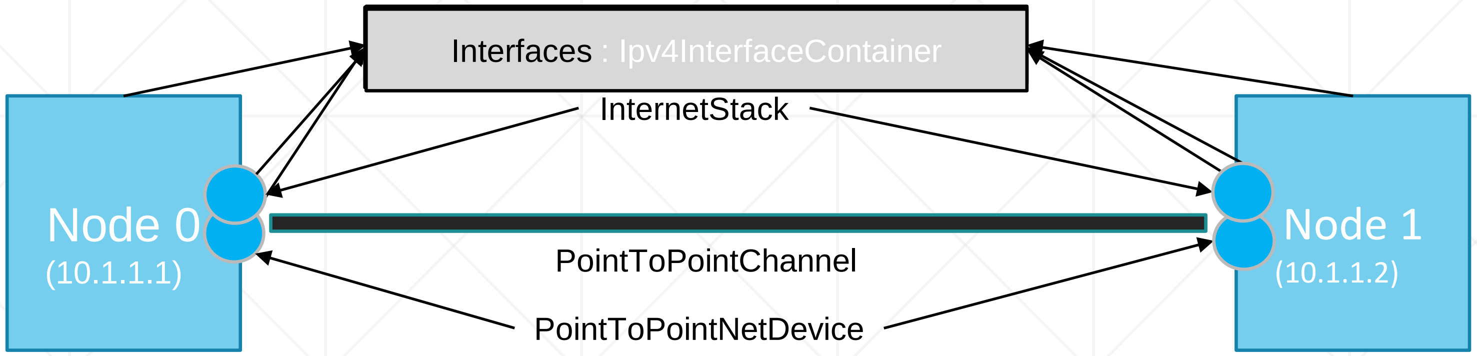
1. Build a channel to connect each other (ex:ethernet, wifi)
2. Need a Net Device to help us use channel
3. Use Application to transmit data



Real world V.S. ns-3

- ns-3 --- Topology

```
NodeContainer nodes;
nodes.Create (2);

PointToPointHelper pointToPoint;
NetDeviceContainer devices;
devices = pointToPoint.Install (nodes);

InternetStackHelper stack;
stack.Install (nodes);
Ipv4AddressHelper address;
address.SetBase ("10.1.1.0", "255.255.255.0");
Ipv4InterfaceContainer interfaces = address.Assign (devices);
```



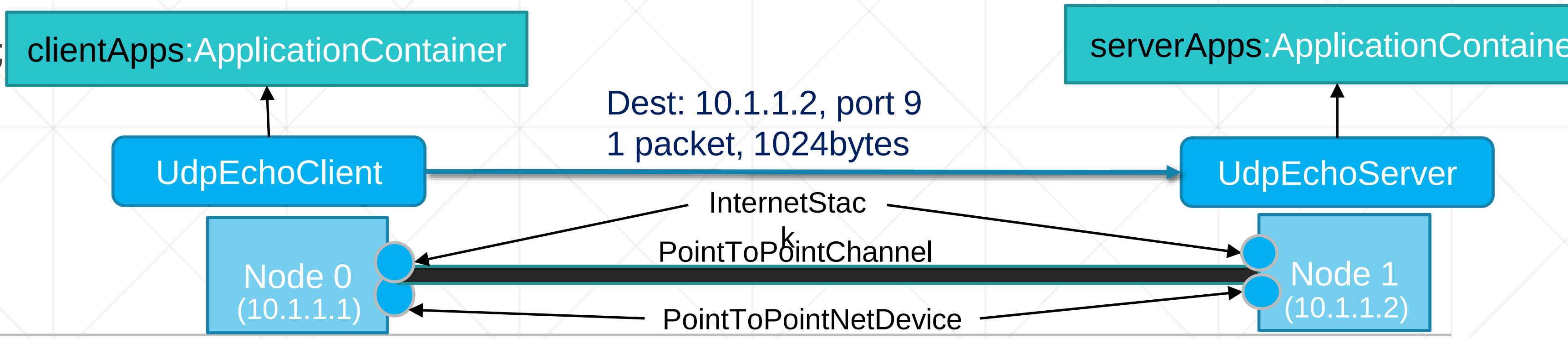
Real world V.S. ns-3

- ns-3 --- application

```
UdpEchoServerHelper echoServer (9);
ApplicationContainer serverApps = echoServer.Install (nodes.Get (1));
serverApps.Start (Seconds (1.0));
serverApps.Stop (Seconds (10.0));

UdpEchoClientHelper echoClient (interfaces.GetAddress (1), 9);
ApplicationContainer clientApps = echoClient.Install (nodes.Get (0));
clientApps.Start (Seconds (2.0));
clientApps.Stop (Seconds (10.0));
```

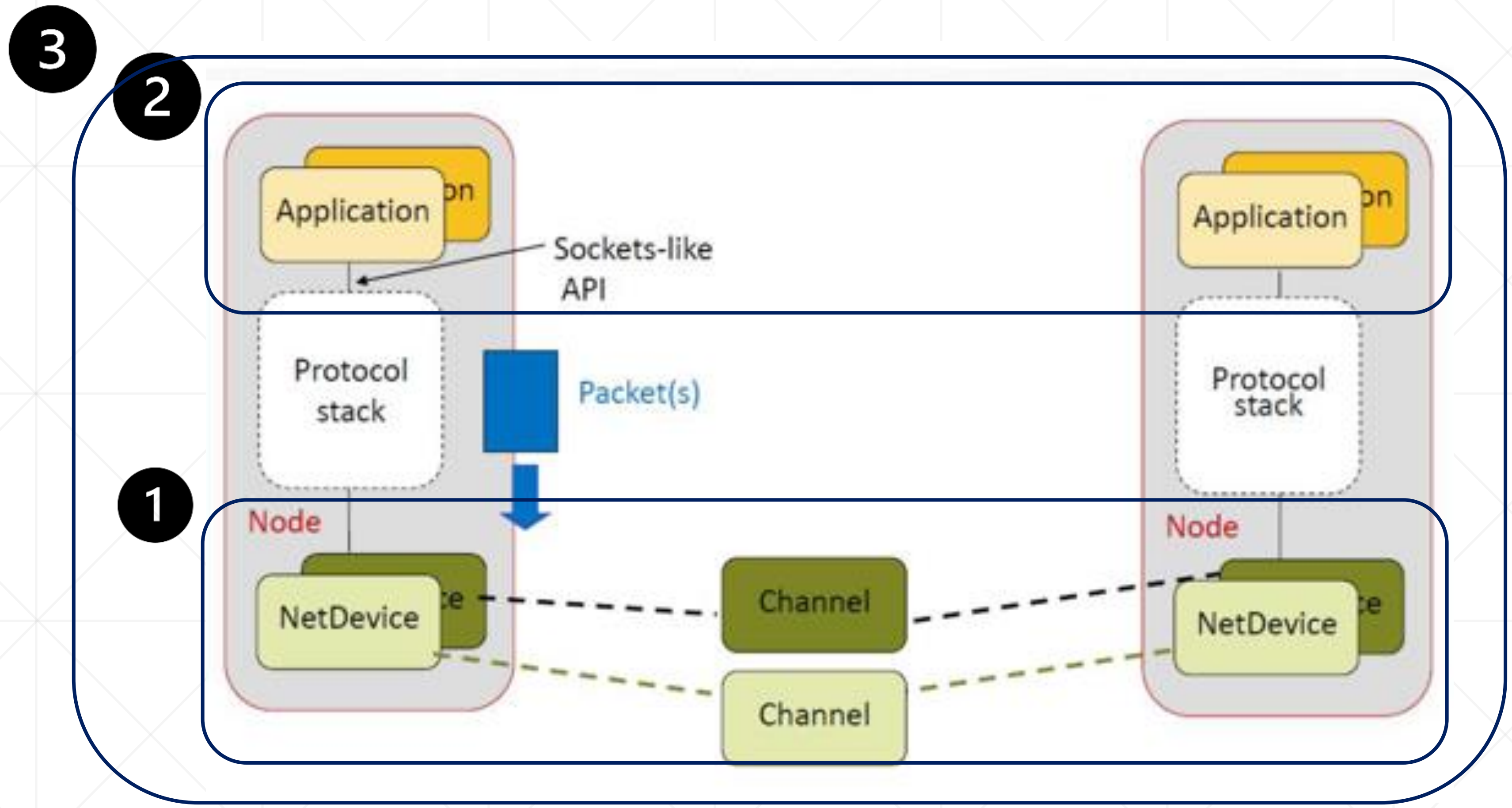
ApplicationContainer

We can use some application helper to help us build an application and install it on the nodes, then we can add to container.
An ApplicationContainer may includes different kind of application in a container.



ns-3 concept

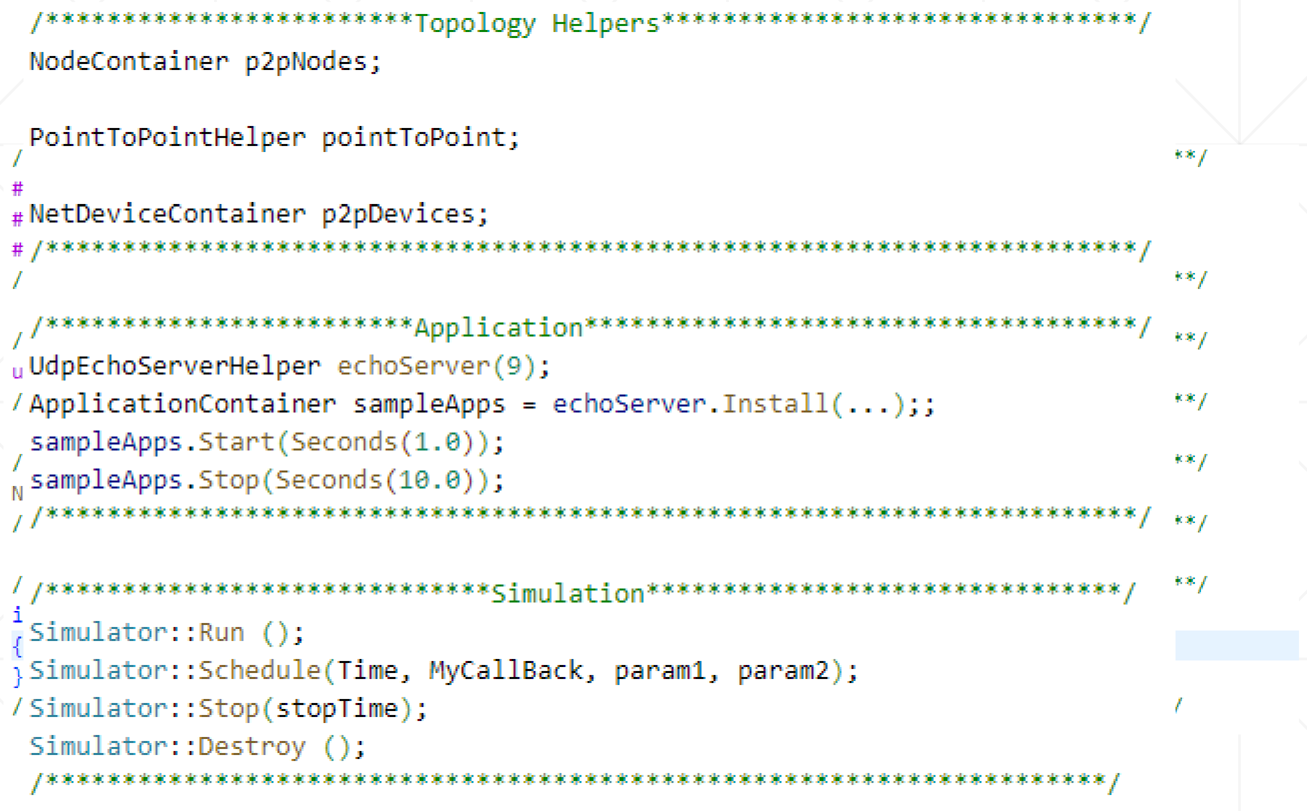
Architecture



Key Abstractions

- Node:
  - The Node class provides methods for managing the representations of computing devices in simulations.
- Channel:
  - Provides methods for managing communication subnetwork objects and connecting nodes to them.
- Application:
  - Run on ns-3 Nodes to drive simulations in the simulated world
- Net Device:
  - Provides methods for managing connections to Node and Channel objects.
- Topology Helpers:
  - Combine many distinct operations, such as connecting NetDevices to Nodes, NetDevices to Channels, assigning IP addresses, etc., into an easy to use model for your convenience

Programming Architecture



```
/************************Topology Helpers*******************************/
NodeContainer p2pNodes;

PointToPointHelper pointToPoint;

NetDeviceContainer p2pDevices;
/*********************************************************************/

/************************Application***********************************/
UdpEchoServerHelper echoServer(9);
ApplicationContainer sampleApps = echoServer.Install(...);;
sampleApps.Start(Seconds(1.0));
sampleApps.Stop(Seconds(10.0));
/*********************************************************************/

/*****************************Simulation******************************/
Simulator::Run ();
Simulator::Schedule(Time, MyCallBack, param1, param2);
Simulator::Stop(stopTime);
Simulator::Destroy ();
/*********************************************************************/
```

Setup ns-3 environment

ns-3 installation

- Environment Recommendation
  - Ubuntu20.04
    ref :
    https://www.c-sharpcorner.com/article/how-to-install-ubuntu-on-windows-10-using-virtualbox/
    https://cloudlinuxtech.com/how-to-install-ubuntu-20-04-vmware-workstation/
    - ns-3.33
- Download ns-3.33 on the web and extract the folder
- Packages requirements (Terminal):
  - apt-get install g++ python3 python3-dev pkg-config sqlite3 cmake
  - apt-get install mercurial qt5-default
  - apt-get install autoconf cvs bzr unrar gdb valgrind
  - apt-get install gsl-bin libgsl-dev libgslcblas0
  - apt-get install flex bison libfl-dev g++-3.4 gcc-3.4
  - apt-get install tcpdump sqlite sqlite3 libsqlite3-dev
  - apt-get install libxml2 libxml2-dev libgtk2.0-0 libgtk2.0-dev libgtk-3-dev
  - apt-get install vtun lxc uml-utilities uncrustify
  - apt-get install doxygen graphviz imagemagick
  - apt-get install texlive texlive-extra-utils texlive-latex-extra texlive-font-utils dvipng latexmk
  - apt-get install python3-sphinx dia
  - apt-get install gir1.2-goocanvas-2.0 python3-gi python3-gi-cairo python3-pygraphviz gir1.2-gtk-3.0 ipython3
  - apt-get install libxml2 libxml2-dev libboost-all-dev
  - apt-get install openmpi-bin openmpi-common openmpi-doc libopenmpi-dev

ns-3 installation

- Install NS-3 and NetAnim on ubuntu (Terminal):

```
# install ns-3
cd ns-allinone-3.33/ns-3.33/
./waf configure --enable-examples
./waf
# test whether the ns-3 is installed
cp -rf examples/tutorial/first.cc scratch
./waf --run scratch/first
# install NetAnim
cd ..
cd netanim-3.108
make clean
qmake NetAnim.pro
make
./NetAnim
```

- Reference

  ns-3 Install (link1、link2、link3)

  ns-3 Tutorial (link)

  ns-3 code Tutorial (link)

How to use NetAnim to animate our project?

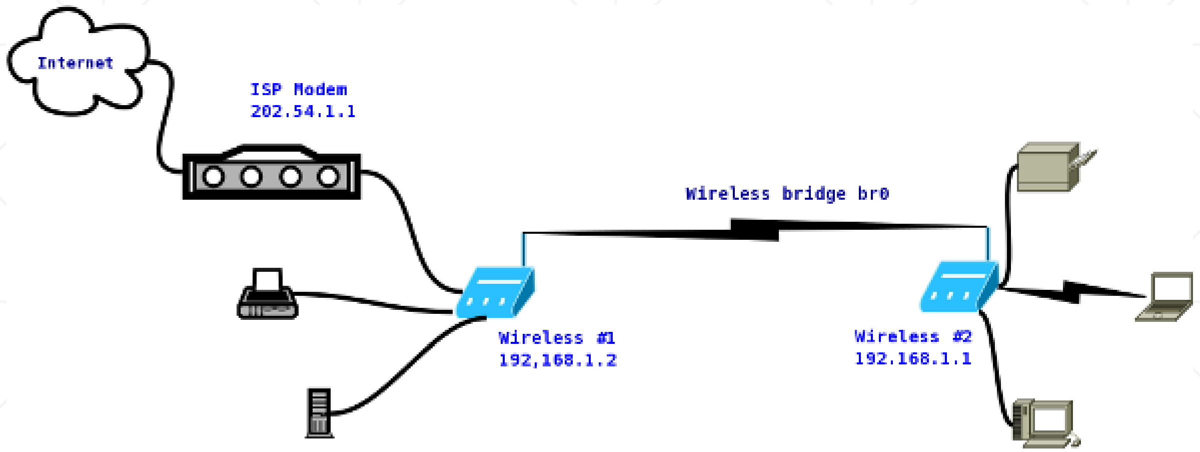
```
#include "ns3/netanim-module.h"

.
.
.

AnimationInterface anim ("ex.xml");
anim.SetStartTime (Seconds(0));
Anim.SetStopTime(Seconds(simTime));

Simulator::Run ();
```

Then, you can open NetAnim to import "ex.xml" file.

Project 1

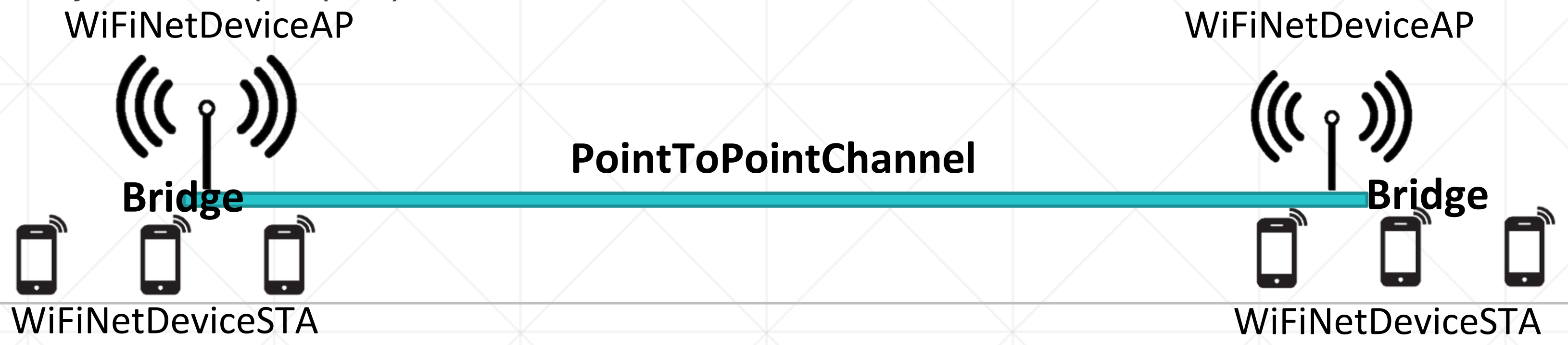
Project 1

- Goal :
  - Build a wifi wired bridging simulation to familiar with the ns-3 platform and architecture
    - How to build a topology
    - How to simulate a application
  - Familiar with NetAnim for simulation

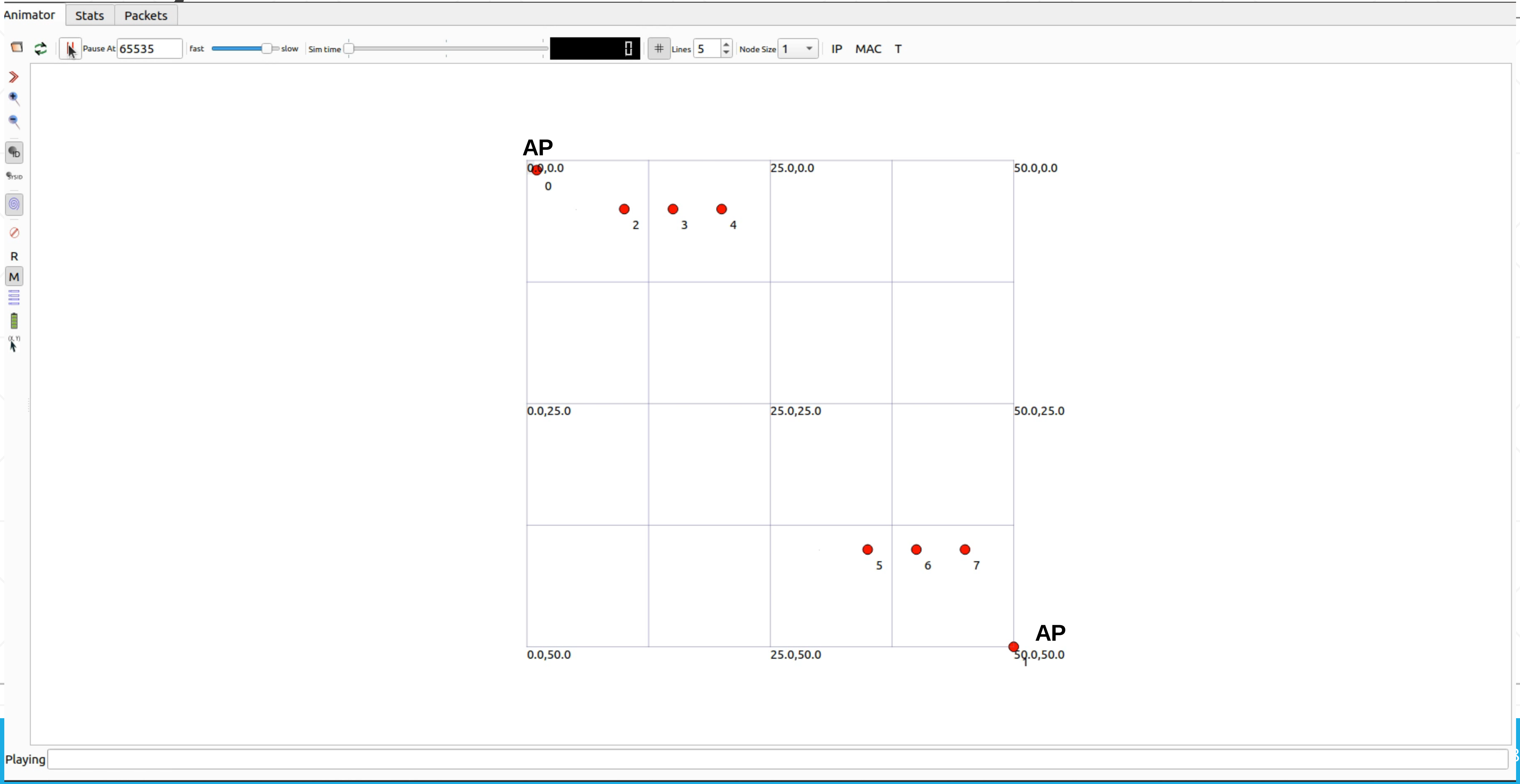


Project 1

- Requirement
  - Install ns-3 on ubuntu
  - Reference code “wifi-wired-bridging.cc” (ns-3.33/examples/wireless/) and “third.cc” (ns-3.33/examples/tutorial/)
    - Create **two WLAN with WiFi**
      - **1 AP** for each WLAN
      - **3 nodes** for each WLAN
    - Using **IPv6** as IP address base
    - Using **UdpEchoHelper** to build applications which **apply UDP protocol**
      - Each **AP talks to its associated STA** nodes
      - **All STA nodes use UDP to communicate**
    - Present simulation by NetAnim (ref. p.17)

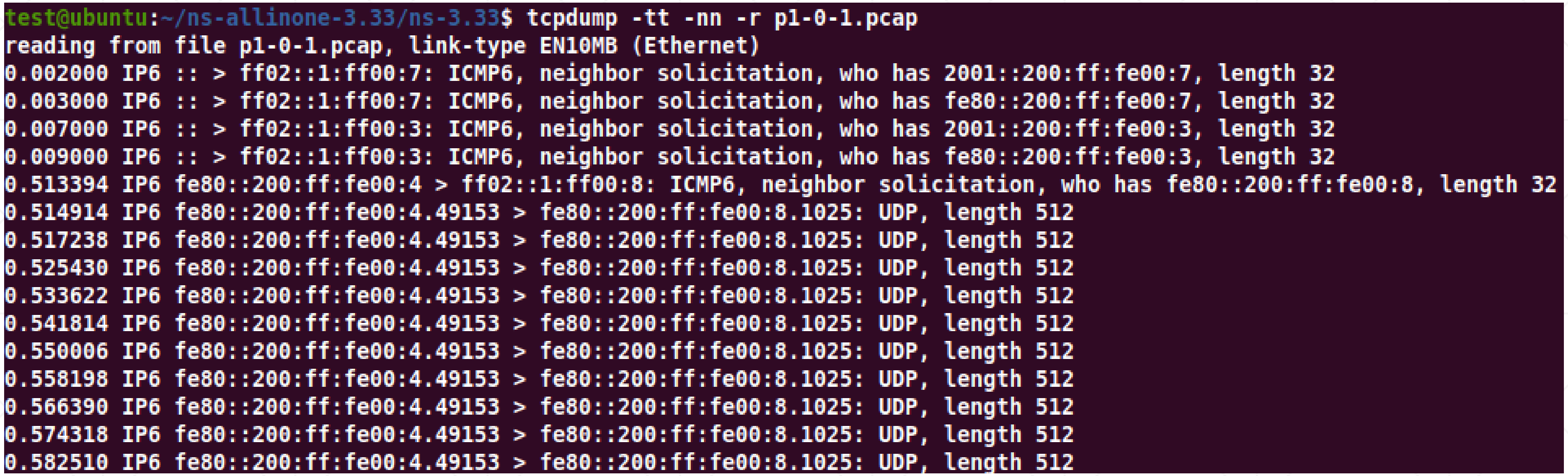


Project 1



Animator
Stats
Packets
Pause At 65535
fast
slow
Sim time
Lines 5
Node Size 1
IP
MAC
T
AP
0.0,0.0
25.0,0.0
50.0,0.0
0
2
3
4
0.0,25.0
25.0,25.0
50.0,25.0
5
6
7
AP
0.0,50.0
25.0,50.0
50.0,50.0
1
Playing

Project 1



```
test@ubuntu:~/ns-allinone-3.33/ns-3.33$ tcpdump -tt -nn -r p1-0-1.pcap
reading from file p1-0-1.pcap, link-type EN10MB (Ethernet)
0.002000 IP6 :: > ff02::1:ff00:7: ICMP6, neighbor solicitation, who has 2001::200:ff:fe00:7, length 32
0.003000 IP6 :: > ff02::1:ff00:7: ICMP6, neighbor solicitation, who has fe80::200:ff:fe00:7, length 32
0.007000 IP6 :: > ff02::1:ff00:3: ICMP6, neighbor solicitation, who has 2001::200:ff:fe00:3, length 32
0.009000 IP6 :: > ff02::1:ff00:3: ICMP6, neighbor solicitation, who has fe80::200:ff:fe00:3, length 32
0.513394 IP6 fe80::200:ff:fe00:4 > ff02::1:ff00:8: ICMP6, neighbor solicitation, who has fe80::200:ff:fe00:8, length 32
0.514914 IP6 fe80::200:ff:fe00:4.49153 > fe80::200:ff:fe00:8.1025: UDP, length 512
0.517238 IP6 fe80::200:ff:fe00:4.49153 > fe80::200:ff:fe00:8.1025: UDP, length 512
0.525430 IP6 fe80::200:ff:fe00:4.49153 > fe80::200:ff:fe00:8.1025: UDP, length 512
0.533622 IP6 fe80::200:ff:fe00:4.49153 > fe80::200:ff:fe00:8.1025: UDP, length 512
0.541814 IP6 fe80::200:ff:fe00:4.49153 > fe80::200:ff:fe00:8.1025: UDP, length 512
0.550006 IP6 fe80::200:ff:fe00:4.49153 > fe80::200:ff:fe00:8.1025: UDP, length 512
0.558198 IP6 fe80::200:ff:fe00:4.49153 > fe80::200:ff:fe00:8.1025: UDP, length 512
0.566390 IP6 fe80::200:ff:fe00:4.49153 > fe80::200:ff:fe00:8.1025: UDP, length 512
0.574318 IP6 fe80::200:ff:fe00:4.49153 > fe80::200:ff:fe00:8.1025: UDP, length 512
0.582510 IP6 fe80::200:ff:fe00:4.49153 > fe80::200:ff:fe00:8.1025: UDP, length 512
```

Project 1

- Grading Policy
  - Install ns-3 on Ubuntu successfully (20%)
  - Finish Project 1 (60%)
    - Topology - YansWiFiChannelHelper, YansWiFiPhyHelper, WifiHelper, WiFiMacHelper, MobilityHelper, InternetStackHelper, Ipv6AddressHelper
    - Application - UdpEchoHelper
    - Animation - NetAnim
    - Comment out the project code
  - Report (20%)
    - Topology
    - Observe the .pcap file and animation to write a report
- Required hand-in files
  - Commented codes (.cc)
  - Report (.pdf/.doc)
- **Deadline：10/11(二) 23：59**

Hints and Reminders

- Hints：
  - Ways of observing .pcap file
    - **Command-line**： tcpdump -nn –tt -r xxx.pcap
    - Using **Wireshark**
  - What information does .pcap file tell
    - What protocol was used to send packets
    - Packets send from/ to where
    - What information send from/ to where
- Reminders：
  - **Do the project as early as possible**
  - Refer to first.cc, second.cc, and third.cc to get more familiar with ns-3 (ns-3.33/examples/tutorial/)