GOKARAJU RANGARAJU INSTITUTE OF ENGINEERING AND TECHNOLOGY
(AUTONOMOUS)
MOBILE COMPUTING LAB

Course Code: GR22A4076 **L/T/P/C: 0/0/ 2/1**

IV Year I Semester

Course Outcome:

1. Recall the fundamentals of networking and protocols
2. Summarize the different categories of network technology
3. Experiment and evaluate the data flow using TCP/IP and UDP.
4. Experiment and evaluate the data flow in MANET and WSN.
5. Experiment and evaluate the WSN cluster using LEACH algorithm.

TASKS

PART - 1: STUDY EXPERIMENTS

1. Study of wired network, wireless network and protocols.
2. Study of MANET and protocols.
3. Study of Wireless Sensor Nodes and protocols.
4. Study of 3G, 4G and 5G network and comparison.
5. Study of Ns3 simulator and installation.
6. Study of OMNET++ simulator and installation.

PART-2: SIMULATION

Instructions:

- Students may use any simulator of their choice. (NS-3 and OMNET++ only)
- Simulation should contain appropriate network devices as per the experiment /environment specified.
- Analysis should be in graphical view.

1. Simulate a wired network and analyse the traffic flow with 10 nodes using TCP/IP.
2. Simulate two wired networks and analyse the traffic flow with other network using TCP/IP.
3. Simulate a wired network and a wireless network, record the traffic pattern between these networks.
4. Simulate a wireless network with 50 nodes and analysis the traffic using UDP.
5. Simulate two wireless networks with 50 nodes each and analysis the traffic with others using UDP.
6. Simulate a MANET and check the traffic effect. Record the packet loss when in motion.
7. Simulate a WSN network with 100 nodes. Analysis the Cluster using LEACH algorithm

Task: 1. Study of wired network, wireless network and protocols.

Aim: To Study the wired, wireless network and Protocols

Procedure:

Wired Networks:

- A wired network is one where the devices in the network are connected using cables.
- Most wired networks are Ethernet networks.
- In the networking world, "Wired" as the name suggests refers to any physical medium
- connected through wires and cables. The wires/cables can be copper wire, twisted pair or even fibre optic.
- Wired connectivity is responsible for providing high security with high Bandwidth provisioned for each user.
- Wired connectivity is considered highly reliable and incurs very low delay, unlike Wireless connectivity.

Wireless networks

- Wireless networks use radio waves to connect devices. There are a range of wireless technologies, but the most common are Wi-Fi and Bluetooth.
- "Wireless" as the term refers, uses air as a medium to send electromagnetic waves or infrared waves. Wireless devices have antennas for communication.
- Wireless connectivity provides a major benefit of user mobility and ease of deployment.
- Wireless becomes more useful in areas where Wires can't be reached.

EXAMPLES of wireless network:

1. Outdoor cellular technologies such as GSM, CDMA, WiMAX, LTE, Satellite etc.
2. Indoor wireless technologies such as Wireless LAN(or WiFi), Bluetooth, IrDA, Zigbee, Zwave etc.

Difference between wired and wireless network.

S.No	Wired Network	Wireless Network
1.	A wired network employs wires to link devices to the Internet or another network, such as laptops or desktop PCs.	"Wireless" means without wire, media that is made up of electromagnetic waves (EM Waves) or infrared waves. Antennas or sensors will be present on all wireless devices
2.	Faster transmission speed	Slow transmission speed
3.	Propagation delay is Low	Propagation delay is high
4.	More Secure & hence Reliable	Less Secure & hence less Reliable
5.	Devices must be hard-wired	Installation is Quick
6.	Less Expensive	More Expensive
7.	High installation & maintenance cost	Low installation & maintenance cost
8.	Hub, Switch, etc. devices are used	Wireless routers, access points, etc. are used.

S. No	PARAMETER	WIRED	WIRELESS
1.	**Communication Medium**	Copper, Fiber etc.	Radio-wave frequency
2.	**Standard**	IEEE 802.3	802.11 family
3.	**Mobility and Roaming**	Limited	Higher
4.	**Security**	High	Lower than Wired. Also easy to hack
5.	**Speed / Bandwidth**	High Speed up to 1 Gbps	Lower speed up to 50Mbps
6.	**Delay**	Low	High
7.	**Reliability**	High	Lower than Wired
8.	**Flexibility to change**	Less flexible to changes	More flexible configuration
9.	**Working principle**	CSMA/CD, operates by detecting the occurrence of a collision.	CSMA/CA, hence reduces possibility of collision be avoiding collision from happening
10.	**Interference and Fluctuations vulnerability**	Very Less	High
11.	**Installation activity**	Cumbersome and manpower intensive	Less labor intensive and easy
12.	**Installation Time**	Takes longer time to perform	Very less deployment time
13.	**Dedicated / Shared Connection**	Dedicated	Shared
14.	**Installation Cost**	High	Low
15.	**Maintenance (Upgrade) cost**	High	Low
16.	**Related equipment**	Router, Switch, Hub	Wireless Router, Access Point
17.	**Benefits**	Greater Speed Higher noise immunity Highly reliable Greater Security	No Hassles of Cable Best for mobile devices Greater mobility Easy installation and management

Protocol:

- In networking, a protocol is a set of rules for formatting and processing data.
- Network protocols are like a common language for computers.
- The computers within a network may use vastly different software and hardware; however, the use of protocols enables them to communicate with each other regardless.

Types of Network Protocols:

1. Transmission Control Protocol (TCP)
2. Internet Protocol (IP)
3. User Datagram Protocol (UDP)
4. Post office Protocol (POP)
5. Simple mail transport Protocol (SMTP)
6. File Transfer Protocol (FTP)
7. Hyper Text Transfer Protocol (HTTP)
8. Hyper Text Transfer Protocol Secure (HTTPS)

9. Telnet
10. Gopher

1. **Transmission Control Protocol (TCP):** TCP is a popular communication protocol which is used for communicating over a network. It divides any message into series of packets that are sent from source to destination and there it gets reassembled at the destination.

2. **Internet Protocol (IP):** IP is designed explicitly as addressing protocol. It is mostly used with TCP. The IP addresses in packets help in routing them through different nodes in a network until it reaches the destination system. TCP/IP is the most popular protocol connecting the networks.

3. **User Datagram Protocol (UDP):** UDP is a substitute communication protocol to Transmission Control Protocol implemented primarily for creating loss-tolerating and low-latency linking between different applications.

4. **Post office Protocol (POP):** POP3 is designed for receiving incoming E-mails.

5. **Simple mail transport Protocol (SMTP):** SMTP is designed to send and distribute outgoing E-Mail.

6. **File Transfer Protocol (FTP):** FTP allows users to transfer files from one machine to another. Types of files may include program files, multimedia files, text files, and documents, etc.

7. **Hyper Text Transfer Protocol (HTTP):** HTTP is designed for transferring a hypertext among two or more systems. HTML tags are used for creating links. These links may be in any form like text or images. HTTP is designed on Client-server principles which allow a client system for establishing a connection with the server machine for making a request. The server acknowledges the request initiated by the client and responds accordingly.

8. **Hyper Text Transfer Protocol Secure (HTTPS):** HTTPS is abbreviated as Hyper Text Transfer Protocol Secure is a standard protocol to secure the communication among two computers one using the browser and other fetching data from web server. HTTP is used for transferring data between the client browser (request) and the web server (response) in the hypertext format, same in case of HTTPS except that the transferring of data is done in an encrypted format. So it can be said that https thwart hackers from interpretation or modification of data throughout the transfer of packets.

9. **Telnet:** Telnet is a set of rules designed for connecting one system with another. The connecting process here is termed as remote login. The system which requests for connection is the local computer, and the system which accepts the connection is the remote computer.

10. **Gopher:** Gopher is a collection of rules implemented for searching, retrieving as well as displaying documents from isolated sites. Gopher also works on the client/server principle.

TASK-2 Study of MANET and protocols

MANET stands for Mobile Adhoc Network also called a wireless Adhoc network or Adhoc wireless network that usually has a routable networking environment on top of a Link Layer ad hoc network. They consist of a set of mobile nodes connected wirelessly in a self-configured, self-healing network without having a fixed infrastructure. MANET nodes are free to move randomly as the network topology changes frequently. Each node behaves as a router as they forward traffic to other specified nodes in the network.

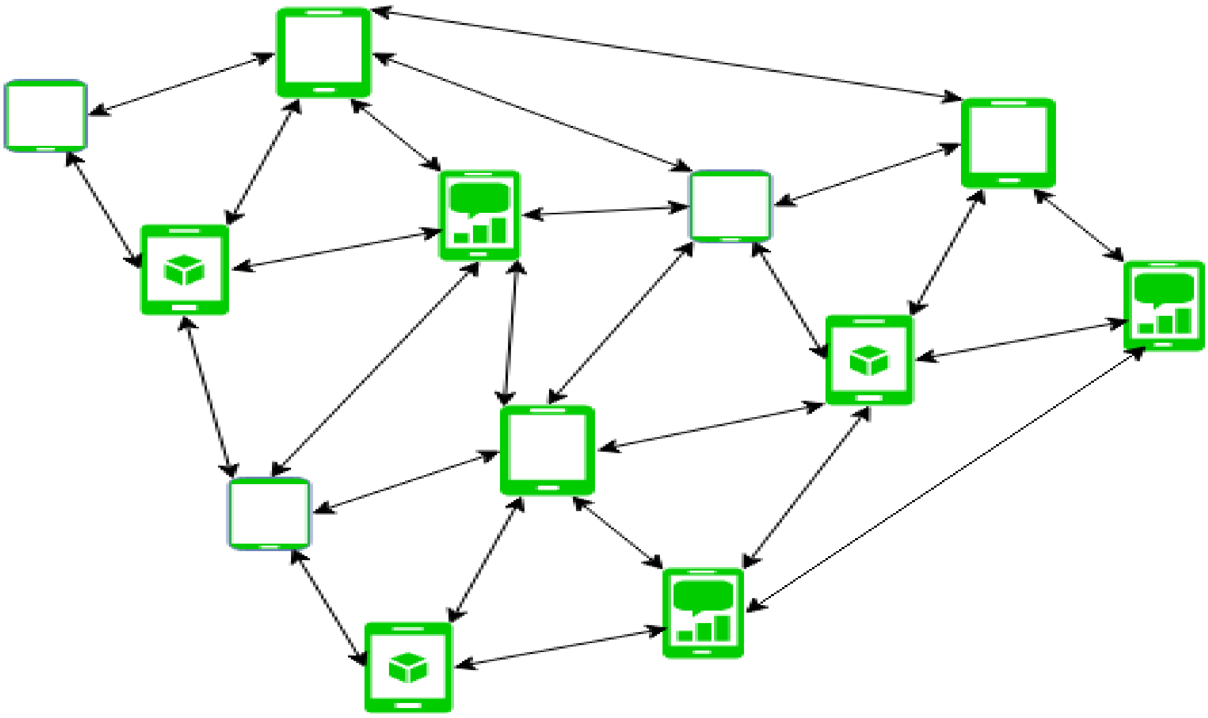


Figure - Mobile Ad Hoc Network

MANET may operate a standalone fashion or they can be part of larger internet. They form a highly dynamic autonomous topology with the presence of one or multiple different transceivers between nodes. The main challenge for the MANET is to equip each device to continuously maintain the information required to properly route traffic. MANETs consist of a peer-to-peer, self-forming, self-healing network MANET's circa 2000-2015 typically communicate at radio frequencies (30MHz-5GHz). This can be used in road safety, ranging from sensors for the environment, home, health, disaster rescue operations, air/land/navy defence, weapons, robots, etc.

Types of MANETS in Computer Network

1. Vehicular Ad hoc Network (VANETs) –

Enable effective communication with another vehicle or with the roadside equipment's. Intelligent vehicular ad hoc networks (InVANETs) deals with another vehicle or with roadside equipment's. VANETs use wireless communication technologies, such as WiFi or cellular, to enable vehicles to communicate with each other and with infrastructure devices, such as traffic lights or road-side units.

1. VANETs can be used to support a wide range of applications, such as:

- **Intelligent Transportation Systems (ITS):** VANETs can be used to improve traffic flow and reduce congestion by providing real-time traffic information and routing advice to drivers.
- **Road Safety:** VANETs can be used to improve road safety by providing information about the location of other vehicles, road conditions, and potential hazards.
- **Entertainment and infotainment:** providing in-vehicle entertainment and internet access to the passengers
- **Emergency Services:** VANETs can be used to support emergency services by providing Realtime information about accidents or other incidents on the road.
- **Commercial Services:** VANETs can be used for commercial services such as providing location-based advertisement and other location-based service to the driver or passengers.

VANETs are considered as one of the most critical applications of the Internet of Things (IoT) technology and the 5G technology

2. **Smart Phone Ad hoc Network (SPANC) –**

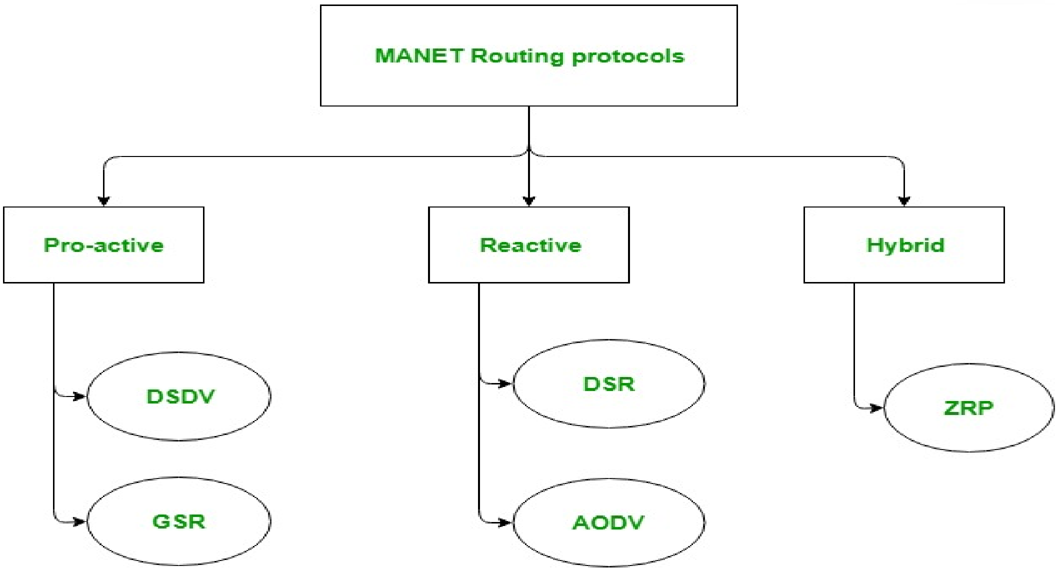
To create peer-to-peer networks without relying on cellular carrier networks, wireless access points, or traditional network infrastructure. Here peers can join or leave the network without destroying it. ad-hoc network that utilizes smartphones as the primary nodes for communication. In SPANC, smartphones can act as both routers and hosts, creating a decentralized network without the need for a central infrastructure. This allows for increased flexibility and scalability in wireless communication, especially in emergency or disaster scenarios where traditional communication infrastructure may be unavailable. Some examples of SPANC applications include disaster response, search and rescue, and urban crowd management.

Uses:

Smart Phone Ad hoc Network (SPANC) can be used for a variety of applications, including:

- **Emergency communication:** In the event of a natural disaster or other emergency, SPANCs can be used to establish a communication network quickly, allowing people to contact emergency services or stay in touch with loved ones.
- **Remote areas:** SPANCs can be useful in remote areas where traditional wireless networks are not available, such as rural communities or wilderness areas.
- **Event networking:** SPANCs can be used to create a temporary network for events or gatherings, allowing attendees to communicate and share information.
- **Military and emergency services:** SPANCs can be used by military and emergency services to establish a quick and reliable communication network in the field.
- **Content sharing:** SPANCs can be used to share various types of content such as pictures and videos, as well as other forms of multimedia.
- **Research and Development:** SPANCs can be used in various research and development projects such as security, routing, and energy consumption.
- **Crowdsourcing:** SPANCs can be used to gather data from a large group of people, such as in a survey or study.
- **Advertising and marketing:** SPANCs can be used to deliver targeted advertising and marketing messages to a specific group of people

MANET routing protocols: a reactive protocol (AODV), a proactive protocol (DSDV), and a location-based protocol (GPSR):



Pro-active routing protocols:

- These are also known as table-driven routing protocols.
- Each mobile node maintains a separate routing table which contains the information of the routes to all the possible destination mobile nodes.
- Since the topology in the mobile ad-hoc network is dynamic, these routing tables are updated periodically as and when the network topology changes.
- It has a limitation that it doesn't work well for the large networks as the entries in the routing table becomes too large since they need to maintain the route information to all possible nodes.

Destination Sequenced Distance Vector Routing Protocol (DSDV):

- It is a pro-active/table driven routing protocol.
- It actually extends the distance vector routing protocol of the wired networks as the name suggests. It is based on the Bellman-ford routing algorithm.
- Distance vector routing protocol was not suited for mobile ad-hoc networks due to count-to-infinity problem.
- Hence, as a solution Destination Sequenced Distance Vector Routing Protocol (DSDV) came into picture.
- Destination sequence number is added with every routing entry in the routing table maintained by each node.
- A node will include the new update in the table only if the entry consists of the new updated route to the destination with higher sequence number.

Global State Routing (GSR):

- It is a pro-active/table driven routing protocol.
- It actually extends the link state routing of the wired networks. It is based on the Dijkstra's routing algorithm.
- Link state routing protocol was not suited for mobile ad-hoc networks because in it, each node floods the link state routing information directly into the whole network i.e. Global flooding which may lead to the congestion of control packets in the network.
- Global state routing doesn't flood the link state routing packets globally into the network.
- In GSR, each of the mobile node maintains one list and three tables namely, adjacency list, topology table, next hop table and distance table.

Reactive routing protocols:

- These are also known as on-demand routing protocol. In this type of routing, the route is discovered only when it is required/needed.
- The process of route discovery occurs by flooding the route request packets throughout the mobile network.
- It consists of two major phases namely, route discovery and route maintenance.

Dynamic Source Routing protocol (DSR):

It is a reactive/on-demand routing protocol. In this type of routing, the route is discovered only when it is required/needed. The process of route discovery occurs by flooding the route request packets throughout the mobile network. In this protocol, Source node stores the complete path information and intermediate nodes do not need to maintain routing information.

It consists of two phases:

- **Route Discovery:** This phase determines the most optimal path for the transmission of data packets between the source and the destination mobile nodes.
- **Route Maintenance:** This phase performs the maintenance work of the route as the topology in the mobile ad-hoc network is dynamic in nature and hence, there are many cases of link breakage resulting in the network failure between the mobile nodes.

Ad-Hoc on Demand Vector Routing protocol (AODV):

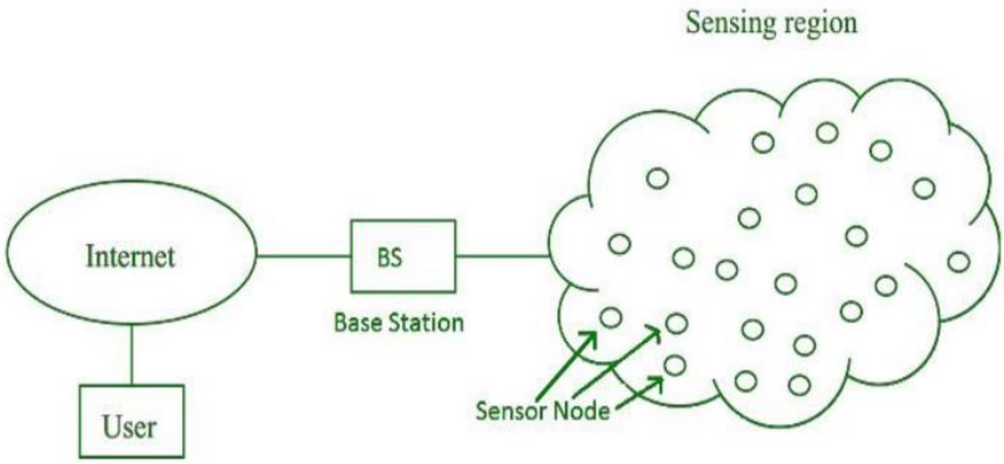
- It is a reactive/on-demand routing protocol.
- It is an extension of dynamic source routing protocol (DSR) and it helps to remove the disadvantage of dynamic source routing protocol.
- In DSR, after route discovery, when the source mobile node sends the data packet to the destination mobile node, it also contains the complete path in its header.
- Hence, as the network size increases, the length of the complete path also increases and the data packet's header size also increases which makes the whole network slow.
- Hence, Ad-Hoc On Demand Vector Routing protocol came as solution to it.
- The main difference lies in the way of storing the path, in AODV Source node does not stores complete path information, instead of that each not stores information of its previous and next node.
- It also operates in two phases: Route discovery and Route maintenance.

Hybrid Routing protocol:

- It basically combines the advantages of both, reactive and pro-active routing protocols.
- These protocols are adaptive in nature and adapts according to the zone and position of the source and destination mobile nodes.
- One of the most popular hybrid routing protocols is Zone Routing Protocol (ZRP).
- The whole network is divided into different zones and then the position of source and destination mobile node is observed.
- If the source and destination mobile nodes are present in the same zone, then proactive routing is used for the transmission of the data packets between them.
- And if the source and destination mobile nodes are present in different zones, then reactive routing is used for the transmission of the data packets between them.

TASK: 3 Study of Wireless Sensor Nodes and protocols.

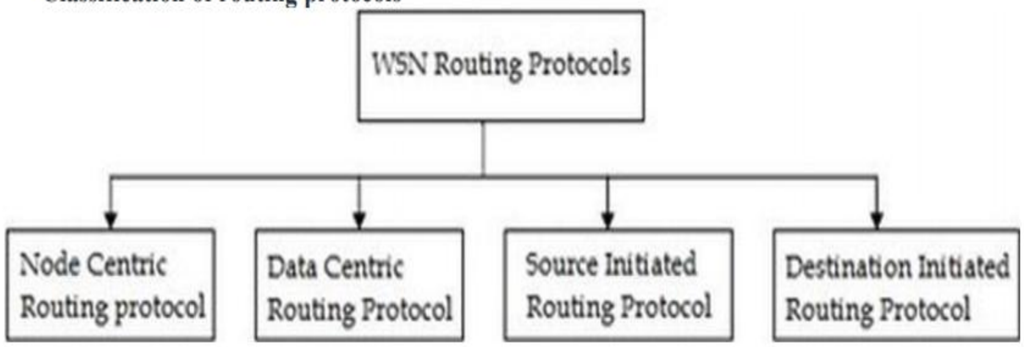
Wireless sensor nodes communicate via their radio modules. Two nodes are directly connected if they can transmit/receive data to/from each other. A sensor communication model (or a transmission model) is a mathematical model that quantifies the direct connectivity between sensor nodes.



Components of WSN:

1. **Sensors:**
   Sensors in WSN are used to capture the environmental variables and which is used for data acquisition. Sensor signals are converted into electrical signals.
2. **Radio Nodes:**
   It is used to receive the data produced by the Sensors and sends it to the WLAN access point. It consists of a microcontroller, transceiver, external memory, and power source.
3. **WLAN Access Point:**
   It receives the data which is sent by the Radio nodes wirelessly, generally through the internet.
4. **Evaluation Software:**
   The data received by the WLAN Access Point is processed by a software called as Evaluation Software for presenting the report to the users for further processing of the data which can be used for processing, analysis, storage, and mining of the data

Classification of routing protocols



TASK: 4 Study of 3G, 4G and 5G network and comparison

The3Gnetwork:

3G really remains for "the third generation", as it isthe third kind of access innovation that has been made generally economically accessible for associating cell phones. They were supplanted by the second era in the 1990s, which now utilized a more solid computerized flag, and empowered the utilization of content informing, or SMS (Short Message Service) benefit permits download rates of up to 7.2Mbps

The frequency band of 3G is 1.8-2.5GHz

- 802.11a uses a modulation technique includes BPSK, QPSK
- 3G uses Turbo codes for error correction
- The access used is wide band CDMA
- 3G uses circuit or packet switching

Advantages of 3G:

- Wireless broadband
- 3G is cheaper for providers
- Extremely faster than previous networks.

Disadvantages of 3G:

- Download speeds can sometimes be slower than expected with the signal strength very variable depending on your device.
- The radiation of magnetic waves

The 4Gnetwork:

4G remains for Fourth era remote system. 4G Innovation is fundamentally the expansion in the 3G innovation with more transfer speed and administrations offers in the 3G. The fourth generation 4G which has the ability to interface with wireline backbone network and that can transmit various multimedia and data across the world just started in 2002. 4G is a conceptual framework which needs high speed wireless network to transfer data.

The frequency band range of 4G is 2-8 GHz

- The bandwidth is as same as of 3G 5-20 MHz
- The data rate is more than 20 Mbps
- It uses the multi-carrier – CDMA or OFDM(TDMA) to 5 MbPS but potential estimated at a range of 10 to300 MbPS
- The switching technique used is packet switching, message switching
- Integration of wireless LAN and wide area.

Advantages of 4G:

- Quickly download files over a wireless network
- Extremely high voice quality
- Easily access Internet, IM, Social Networks, streaming media, video calling, etc.
- Higher bandwidth
- WiMAX, LTE, and HSPA+ are all versions of 4G, WiMAX is used by Sprint, LTE is used by Verizon and AT&T, HSPA+ is used by AT&T and TMobile
- 4G is 10 times faster than 3G

Disadvantages of 4G:

- New frequencies mean new components in cell towers.
- Higher data prices for consumers
- Consumer is forced to buy a new device to support the 4G
- It is impossible to make your current equipment compatible with the 4G network

The 5G:

The 5G wireless technology will be having various software which can be downloaded from the internet like the new error control schemes and the radio and modulation schemes. This development in the past years is the step towards the user terminals. The improvement is seen towards the client terminals as a focal point of the 5G versatile systems. 5G guarantees ultra-dependable, quick speeds and high data transmission portable availability, which gives crest paces of 20 times, contrasted and 4G. It is required to give 10 or more Gbps speeds, which enable access to high-transmission capacity mixed media and information administrations for different industrial applications. 5G is planned to help mission-basic applications, for example, money related exchanges and human services, and inactivity and rapid will be accomplished using Fiber optic links.

Advantages of 5G-

- Increased bandwidth for all users
- High resolution
- Bi – directional large bandwidth
- More efficient and easily manageable
- Uninterrupted uniform connectivity

Disadvantages of 5G –

- The research is still in progress
- It might be available in some places as there is nonrequired facilities.
- Old devices cannot support this
- Required infrastructure has to be developed

Task 5: NS3 Simulator Installation

For the installation of NS3, VMware workstation is required to be installed, along with an Ubuntu system.

1. Download VMWare workstation from the website: https://my.vmware.com/en/web/vmware/downloads/info/slug/desktop_end_user_computing/vmware_workstation_player/15_0

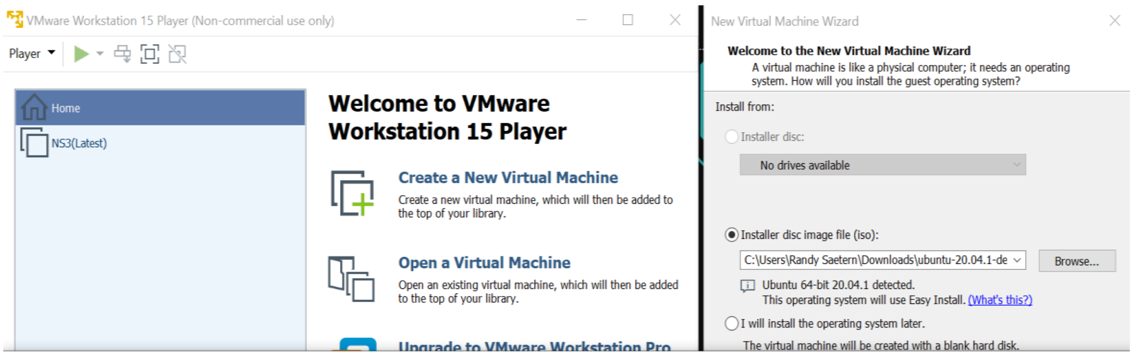
2. Download Ubuntu 20.04.01 Desktop AMD 64 from the website: https://ubuntu.com/download/desktop

3. Install VMWare workstation onto the computer system and open it

4. Set up the VMware workstation:
   a. Create a new virtual machine by selecting "Create New Virtual Machine."

Figure 1: Creation of a new VM.

   b. In the installer wizard, select installer disc image file(iso) and select the downloaded Ubuntu 20.04.01 AMD 64 iso file by browsing through the computer download files.



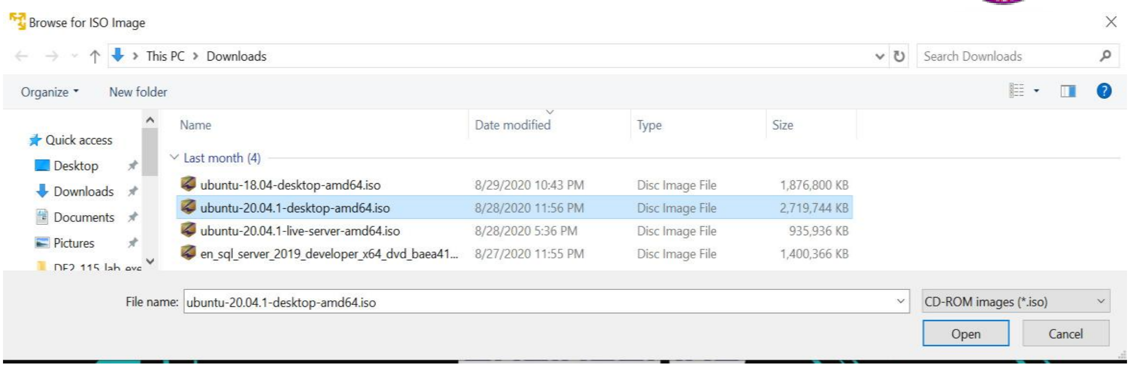


Figure 2: Installation with a disc image file(iso).

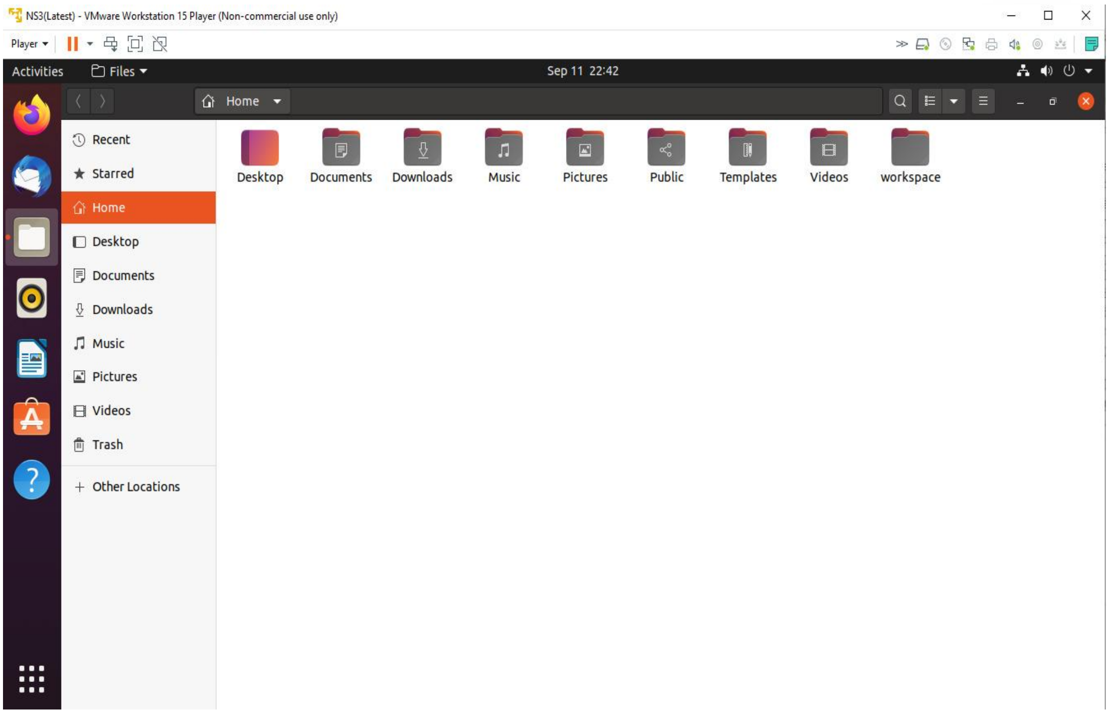
c. Name the machine and set the password.
d. Configure the Hardware:
    i. For memory: set the value to 4600 MB or above.
    ii. For faster VMware, set processors to 2.

5. Power on the virtual machine and let the machine update.

6. Within the Virtual machine, download NS3 on the VM by opening Mozilla firefox and downloading from the NS3 website.

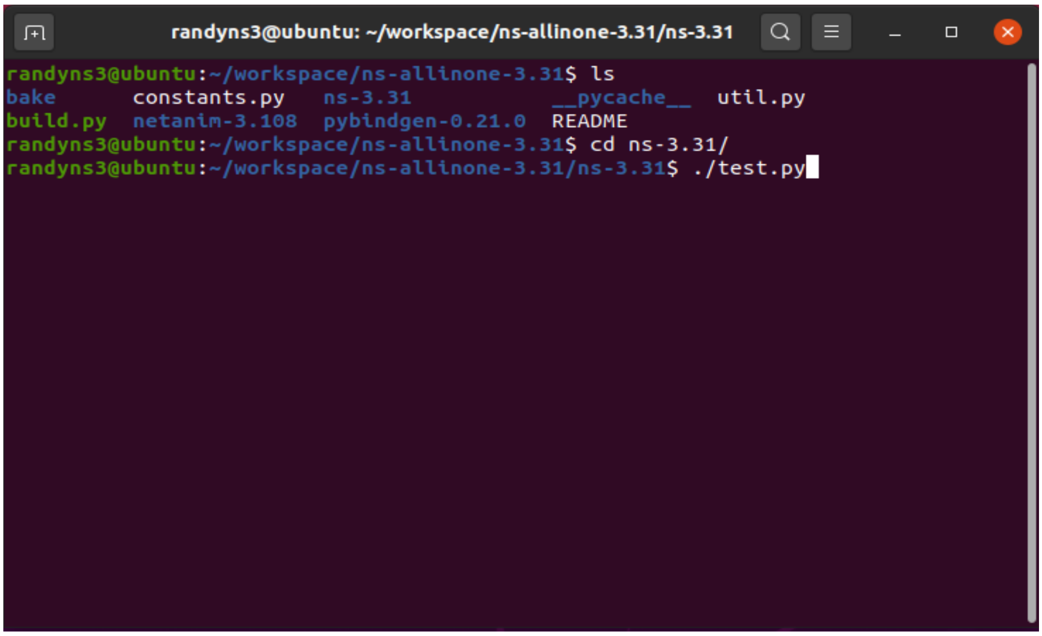
7. Install prereq packages on Ubuntu using terminal:
    a. Open the terminal by right clicking the desktop and select "open in terminal."
    b. Paste in this code and then press enter: sudo apt-get install g++ python3 python3-dev pkg-config sqlite3 python3- setuptools git qt5-default mercurial gir1.2-goocanvas-2.0 python-gi python-gi-cairo python3-gi python3-gi-cairo python3-pygraphviz gir1.2- gtk-3.0 ipython3 openmpi-bin openmpi-common openmpi-doc libopenmpi-dev autoconf cvs bzr unrar gdb valgrind uncrustify doxygen graphviz imagemagick texlive texlive-extra-utils texlive-latex-extra texlivefont- utils dvipng latexmk python3-sphinx dia gsl-bin libgsl-dev libgsl23 libgslcblas0 tcpdump sqlite sqlite3 libsqlite3-dev libxml2 libxml2-dev cmake libc6-dev libc6-dev-i386 libclang-6.0-dev llvm-6.0-dev automake python3-pip libgtk-3-dev synaptic vtun lxc uml-utilities
    c. After the packages have finished downloading, paste in this code and press enter: sudo pip3 install cxxfilt

8. After installing the required packages, create a folder named workspace in the home directory and then put the NS3 tar package into the workspace. See example figure below.



9. Go to terminal and input these commands consecutively after each command finishes executing:
   cd
   cd workspace
   tar xjf <name of NS3 downloaded file name>
   cd <name of extracted NS3>
   ./build.py --enable-examples --enable-tests

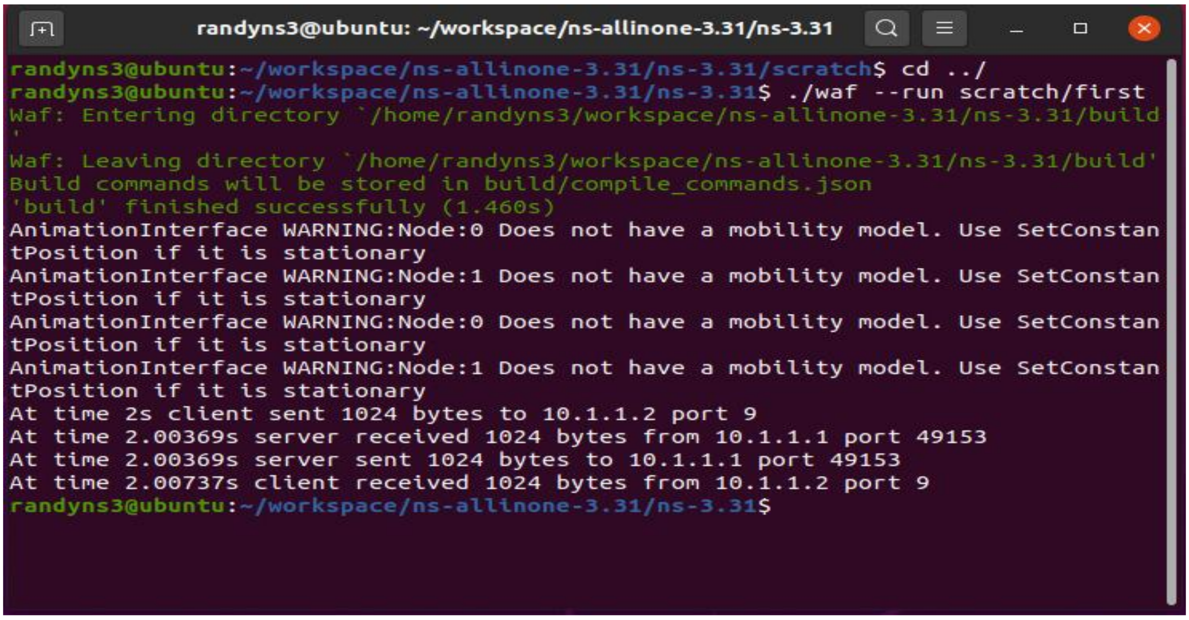
10. Test the NS3 build and installation success by running test.py in the ns directory using the following commands:
    cd ns-<version number>
    ./test.py



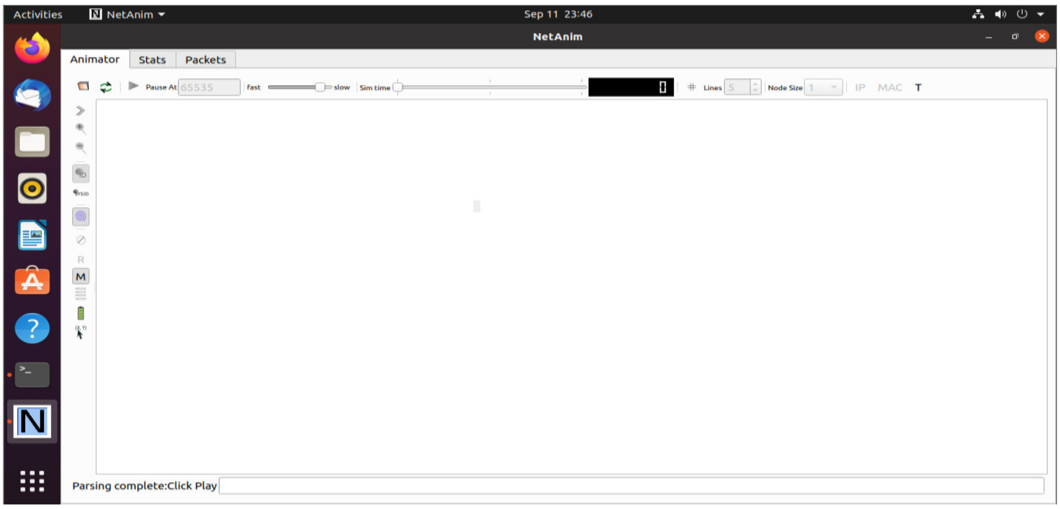
11. If all of the tests were passed, Congratulations! NS3 has now been installed successfully.

Building NetAnim and Running a Simulation

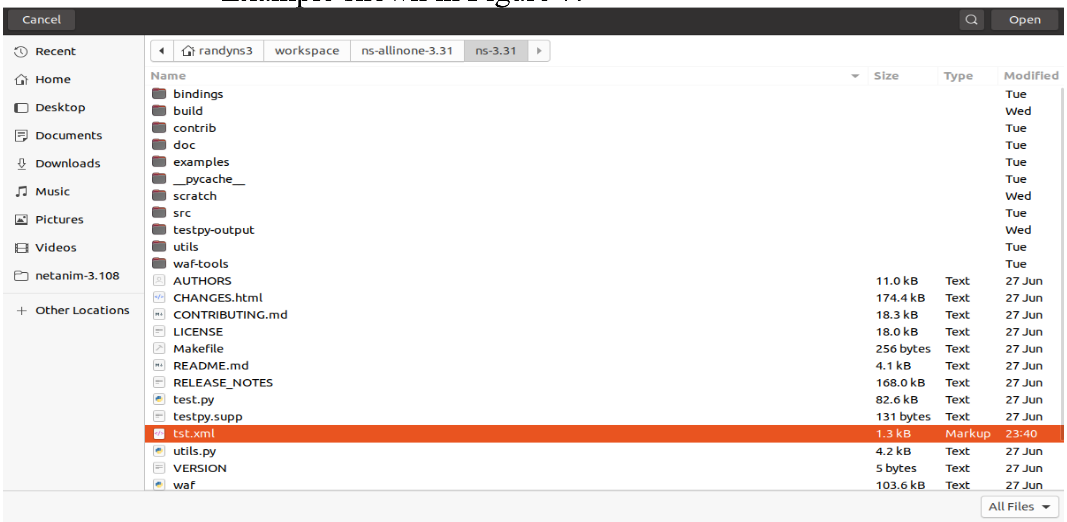
1. To build NetAnim, the qmake package will be utilized in the following process:
    a. Go to the NetAnim directory pasting these commands in the terminal:
    cd
    cd workspace
    cd <ns folder name>
    cd <netanim folder name>
    b. Clean make files using the command:
    make clean
    c. Make NetAnim using the commands:
    qmake NetAnim.pro
    make
    d. Test the NetAnim installation by pasting the following command in the
    e. terminal, while within the netanim directory:
    ./NetAnim
2. If NetAnim opens, congratulations! NetAnim is now installed.
3. Building and Running simulation procedures:
    a. Copy a .cc file to the scratch directory, located in the ns-<version#> directory.
    b. Edit the copied .cc file to include the netanim library files by pasting in the following code to the code:
    #include "ns3/netanim-module.h"
    c. Instantiate the animation interface into the code.
    d. Exit the scratch directory by using the following command in the terminal:
    cd ../
    e. Paste in the following commands to the terminal:
    ./waf –run scratch/<name of cc file>



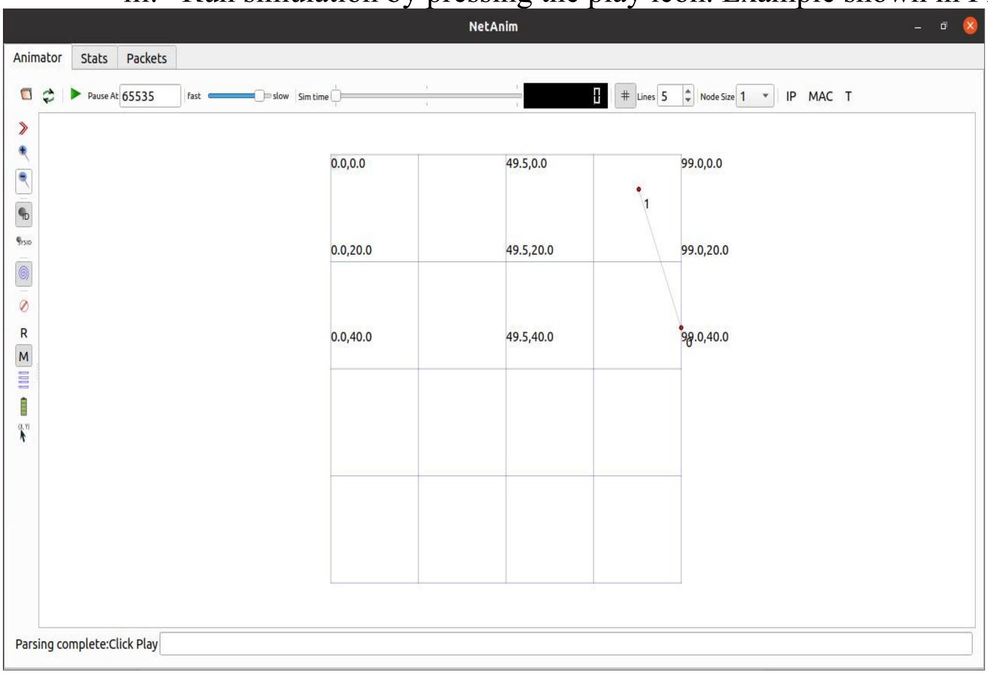
    f. Run NetAnim by entering the netanim directory in the terminal and use the code: ./NetAnim
    g. Select the created xml file and run resulting simulation by following these procedures:
        i. Click on the folder icon in the top left side, shown in Figure 6.



ii. Go into the ns-<version#> folder and select the xml file you built. Example shown in Figure 7.



iii. Run simulation by pressing the play icon. Example shown in Figure 8.



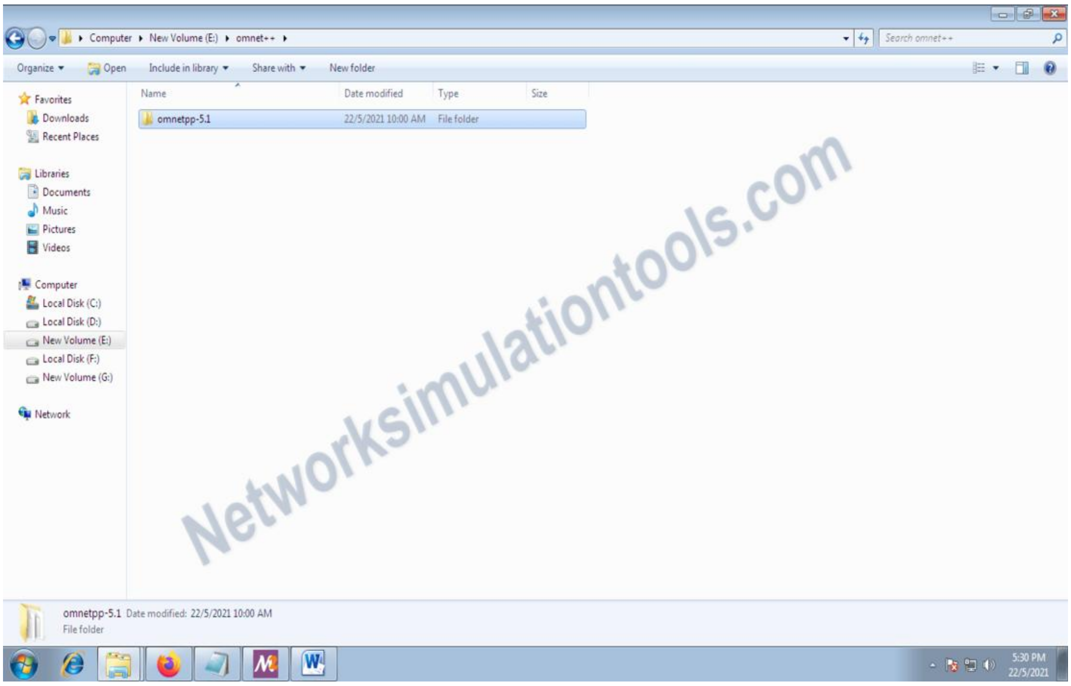
Task 6: Study of OMNET++ simulator and installation.

1. **System requirement**

   To perform the installation process, we need the following requirements,

   1) OS: Windows 10 pro

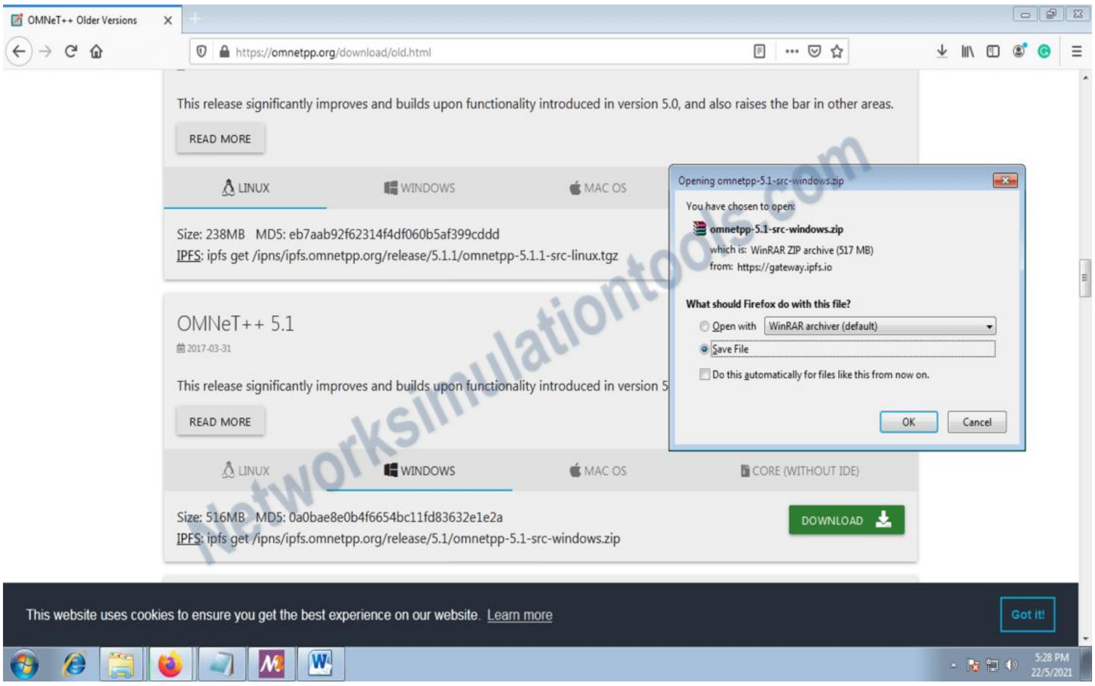
   2) Install Omnet++ 5.1



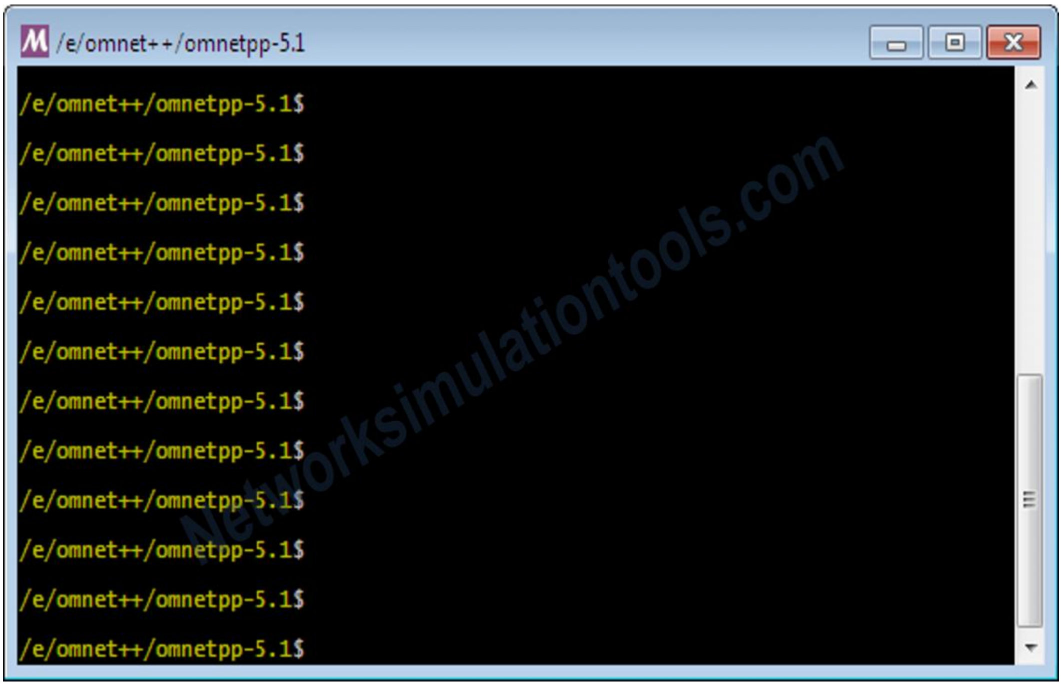
2. **Download the omnet++ package**

   Download the omnet++ package from the following url, in the site there multiple verion of the omnet++ is listed. Select your needed version of omnet++, here we select the eversion omnet++ 5.1 , after that click the download button , which is displayed in the below of selected version

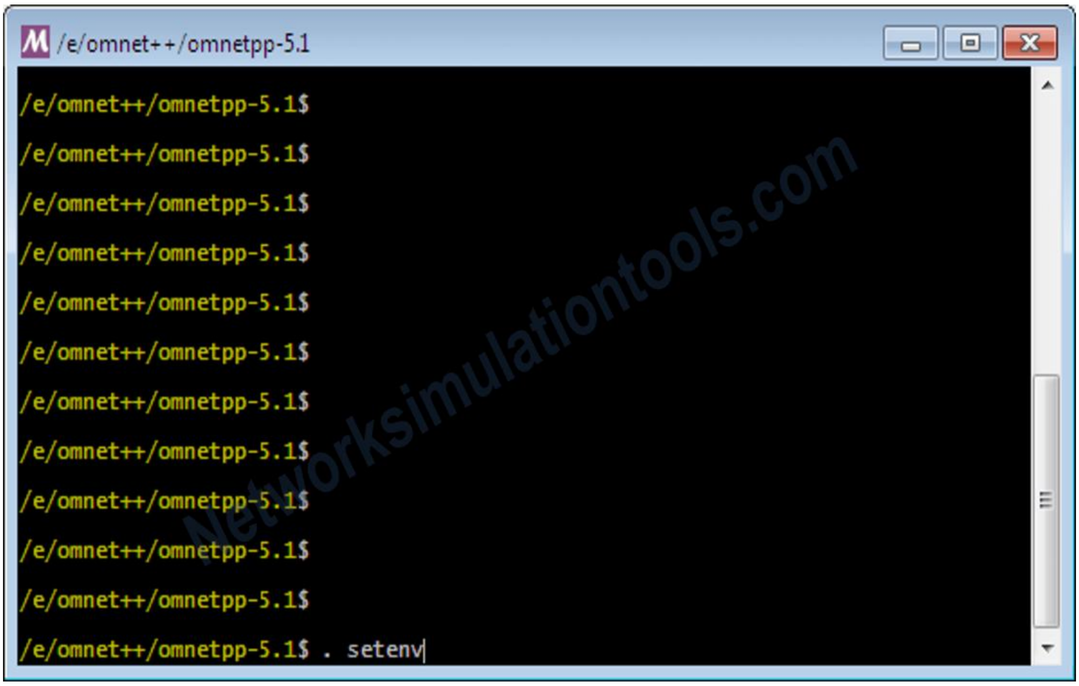
   https://omnetpp.org/download/old.html



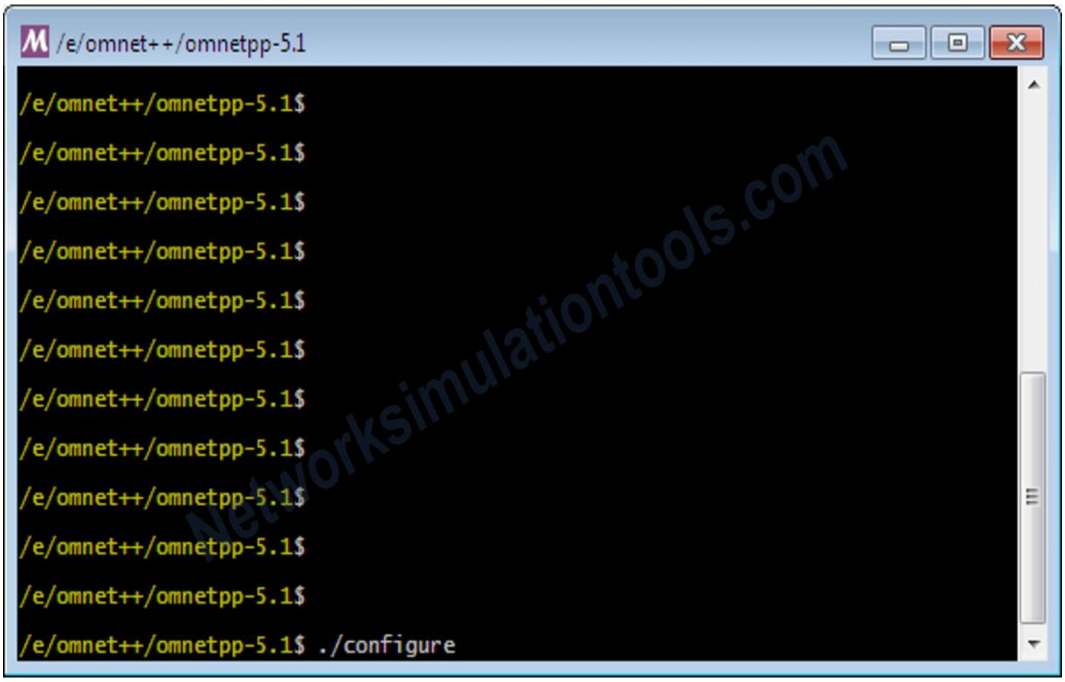
3. **Open the terminal/ mingwenv window**
   Open the mingwenv window. By double click mingwenv window from the omnet++ installed location.



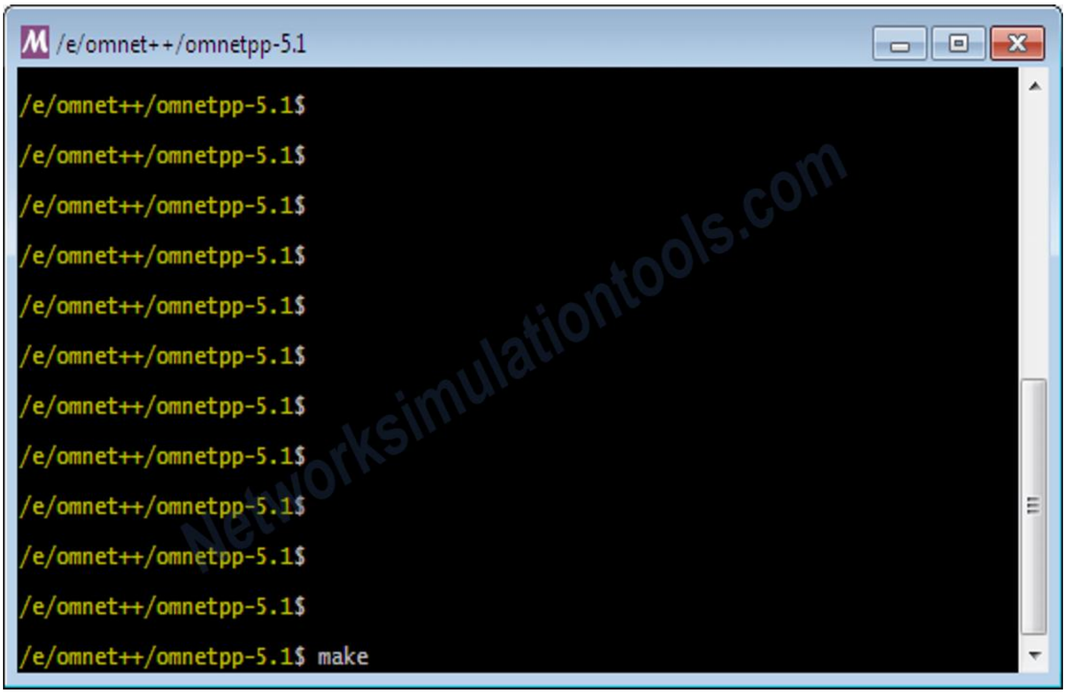
4. **Execute the command window**
   In the mingwenv command window ,
   Execute the command -> . setenv



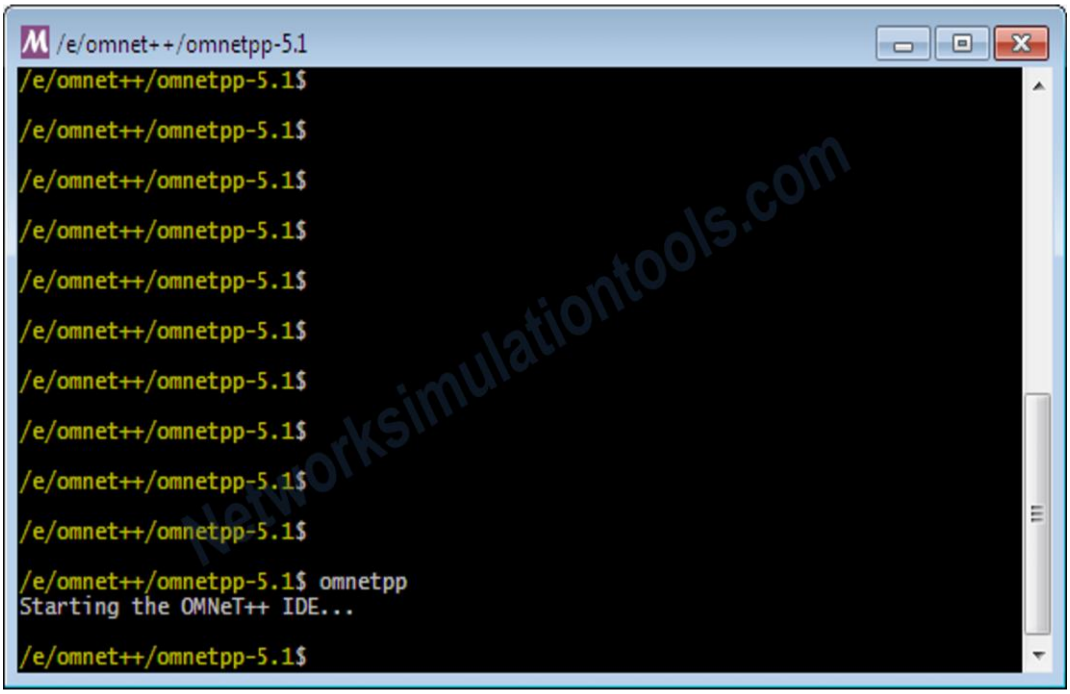
5. **Execute the configure command window**
   In the mingwenv command window ,
   Execute the command -> ./configure



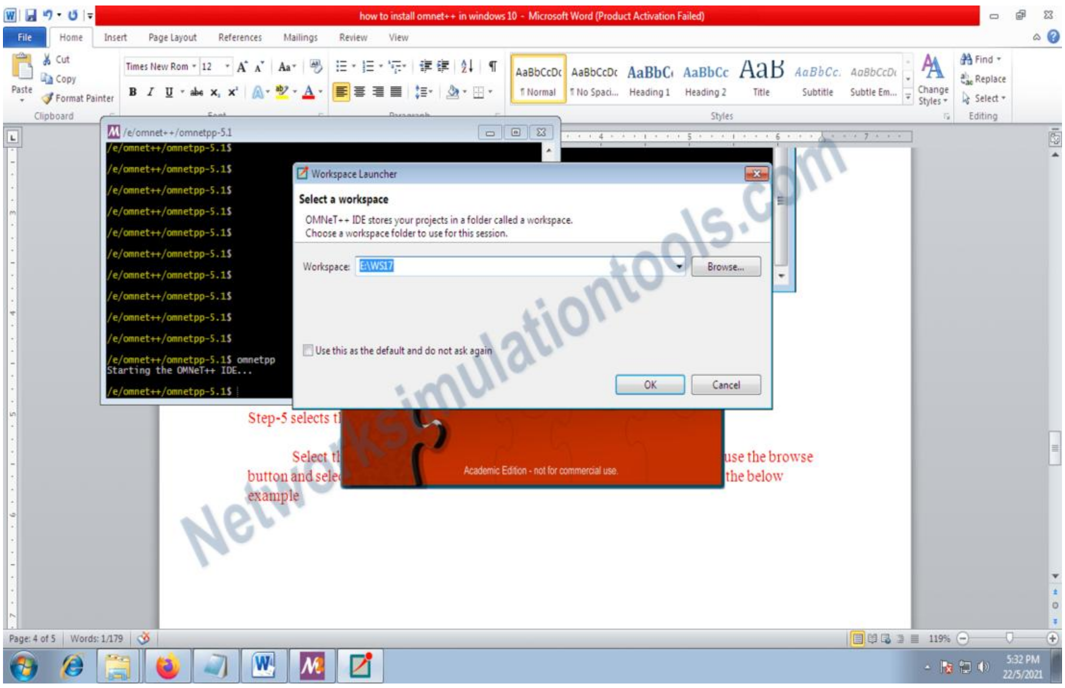
6. **Execute the Make command window**
   In the mingwenv command window ,
   Execute the command -> make



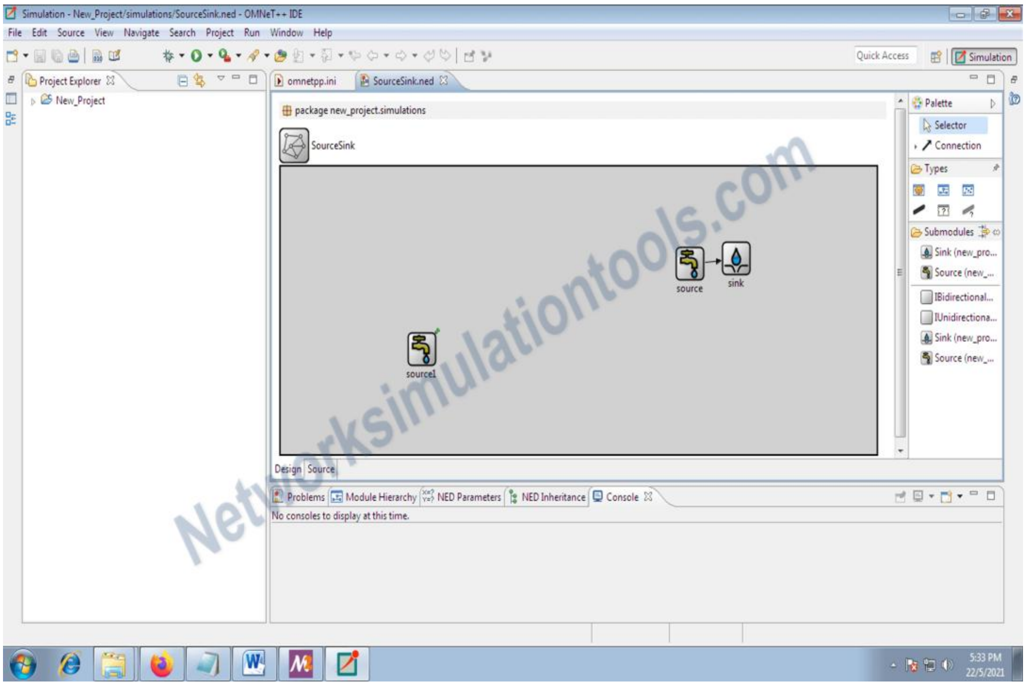
7. **Execute the omnet++ command window**
   In the mingwenv command window ,
   Execute the command -> omnetpp



8. **Select the workspace window**
   Select the stored workspace, with full location, to select the workspace , use the browse button and select the location and press ok button or type the full location . , like the below example



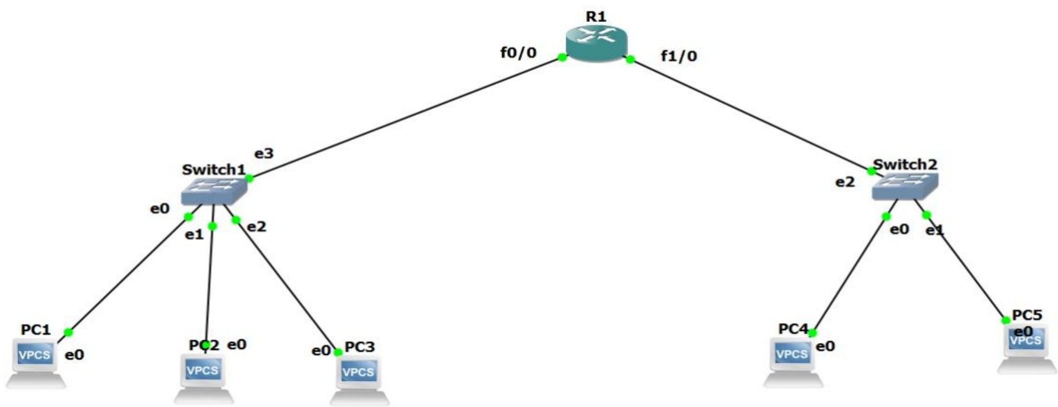
9. **Loaded workspace window**
   By use the browse button and select the location and press ok button or type the full location., like the below example
   View the loaded workspace window



Simulation - New_Project/simulations/SourceSink.ned - OMNeT++ IDE
File Edit Source View Navigate Search Project Run Window Help
Quick Access
Simulation
Project Explorer
New_Project
omnetpp.ini
SourceSink.ned
package new_project.simulations
SourceSink
source
sink
source1
Palette
Selector
Connection
Types
Submodules
Design Source
Problems Module Hierarchy NED Parameters NED Inheritance Console
No consoles to display at this time.

PART – II

Task I: Simulate a wired network and analyse the traffic flow with 10 nodes using TCP/IP.



Connect 8 Devices in two networks and find the data flow between end devices through Router

Configure PC1:

```
PC1> ip 2.2.2.2/12 2.2.2.1
Checking for duplicate address...
PC1 : 2.2.2.2 255.240.0.0 gateway 2.2.2.1

PC1> show ip

NAME        : PC1[1]
IP/MASK     : 2.2.2.2/12
GATEWAY     : 2.2.2.1
DNS         :
MAC         : 00:50:79:66:68:00
LPORT       : 10018
RHOST:PORT  : 127.0.0.1:10019
MTU:        : 1500

PC1> save
Saving startup configuration to startup.vpc
.  done
```

PC2:

```
PC2> ip 2.2.2.3/12 2.2.2.1
Checking for duplicate address...
PC1 : 2.2.2.3 255.240.0.0 gateway 2.2.2.1

PC2> show ip

NAME        : PC2[1]
IP/MASK     : 2.2.2.3/12
GATEWAY     : 2.2.2.1
DNS         :
MAC         : 00:50:79:66:68:01
LPORT       : 10022
RHOST:PORT  : 127.0.0.1:10023
MTU:        : 1500

PC2> save
Saving startup configuration to startup.vpc
.  done
```

PC3:

```
PC3> ip 2.2.2.4/12 2.2.2.1
Checking for duplicate address...
PC1 : 2.2.2.4 255.240.0.0 gateway 2.2.2.1

PC3> show ip

NAME        : PC3[1]
IP/MASK     : 2.2.2.4/12
GATEWAY     : 2.2.2.1
DNS         :
MAC         : 00:50:79:66:68:02
LPORT       : 10020
RHOST:PORT  : 127.0.0.1:10021
MTU:        : 1500

PC3> save
Saving startup configuration to startup.vpc
.  done
```

PC4:

```
PC4> ip 3.2.2.2/12 3.2.2.1
Checking for duplicate address...
PC1 : 3.2.2.2 255.240.0.0 gateway 3.2.2.1

PC4> show ip

NAME        : PC4[1]
IP/MASK     : 3.2.2.2/12
GATEWAY     : 3.2.2.1
DNS         :
MAC         : 00:50:79:66:68:03
LPORT       : 10024
RHOST:PORT  : 127.0.0.1:10025
MTU:        : 1500

PC4> save
Saving startup configuration to startup.vpc
.  done
```

PC5:

```
PC5> ip 3.2.2.3/12 3.2.2.1
Checking for duplicate address...
PC1 : 3.2.2.3 255.240.0.0 gateway 3.2.2.1

PC5> show ip

NAME        : PC5[1]
IP/MASK     : 3.2.2.3/12
GATEWAY     : 3.2.2.1
DNS         :
MAC         : 00:50:79:66:68:04
LPORT       : 10026
RHOST:PORT  : 127.0.0.1:10027
MTU:        : 1500

PC5> save
Saving startup configuration to startup.vpc
.  done
```

Configure Router:

```
R1#
R1#
R1#
R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#interface f0/0
R1(config-if)#ip address 2.2.2.1 255.240.0.0
R1(config-if)#no shut
R1(config-if)#exit
R1(config)#inte
*Oct 19 15:01:32.475: %LINK-3-UPDOWN: Interface FastEthernet0/0, changed state to up
*Oct 19 15:01:33.475: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up
R1(config)#interface f1/0
R1(config-if)#ip address 3.2.2.1 255.240.0.0
R1(config-if)#no shut
R1(config-if)#exit
R1(config)#
*Oct 19 15:01:57.339: %LINK-3-UPDOWN: Interface FastEthernet1/0, changed state to up
*Oct 19 15:01:58.339: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet1/0, changed state to up
R1(config)#exit
R1#
*Oct 19 15:02:00.019: %SYS-5-CONFIG_I: Configured from console by console
R1#
```

Connect PC1 and PC4:

```
PC1> ping 3.2.2.2
84 bytes from 3.2.2.2 icmp_seq=1 ttl=63 time=123.363 ms
84 bytes from 3.2.2.2 icmp_seq=2 ttl=63 time=31.550 ms
84 bytes from 3.2.2.2 icmp_seq=3 ttl=63 time=31.480 ms
84 bytes from 3.2.2.2 icmp_seq=4 ttl=63 time=32.305 ms
84 bytes from 3.2.2.2 icmp_seq=5 ttl=63 time=31.974 ms

PC1>
```

Similarly, connect other devices

TASK II - Simulate two wired networks and analyse the traffic flow with other network using TCP/IP.

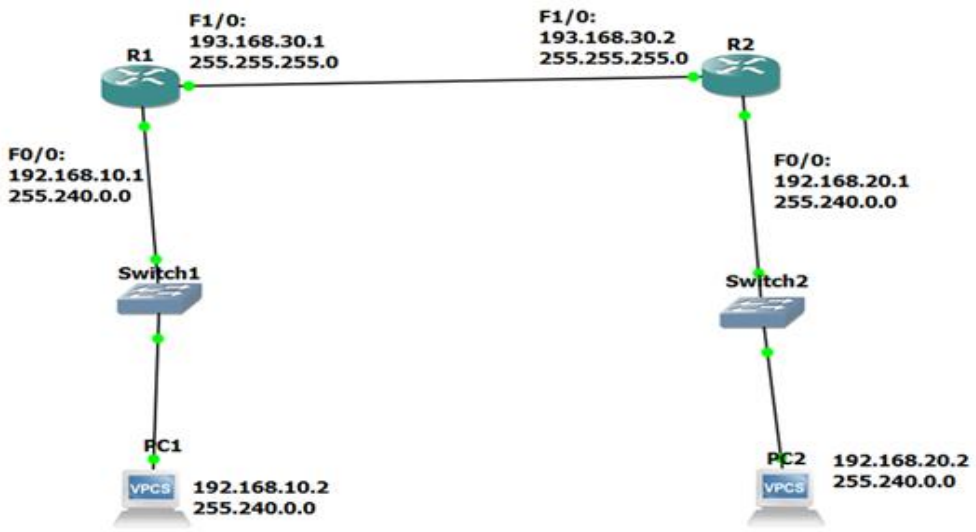
Router Configuration:
Change Router Name:

```
R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#hostname Bharath
Bharath(config)#exit
Bharath#
*Sep 14 09:33:01.815: %SYS-5-CONFIG_I: Configured from console by console
Bharath#
```

Assigning an IP Address to the Router's Interface:

```
R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#interface f0/0
R1(config-if)#no shutdown
R1(config-if)#
*Sep 14 11:12:20.823: %LINK-3-UPDOWN: Interface FastEthernet0/0, changed state to up
*Sep 14 11:12:21.823: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up
R1(config-if)#ip address 1.1.1.1 255.240.0.0
R1(config-if)#interface f1/0
R1(config-if)#no shutdown
R1(config-if)#
*Sep 14 11:14:02.855: %LINK-3-UPDOWN: Interface FastEthernet1/0, changed state to up
*Sep 14 11:14:03.855: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet1/0, changed state to up
R1(config-if)#ip address 2.2.2.1 255.240.0.0
R1(config-if)#end
R1#cop
*Sep 14 11:15:37.067: %SYS-5-CONFIG_I: Configured from console by console
R1#copy run start
```

Transmit data between two routers:



PC1 IP:

```
PC1> ip 192.168.10.2 255.240.0.0
Checking for duplicate address...
PC1 : 192.168.10.2 255.240.0.0

PC1> save
Saving startup configuration to startup.vpc
.  done

PC1> sh ip

NAME            : PC1[1]
IP/MASK         : 192.168.10.2/12
GATEWAY         : 255.240.0.0
DNS             :
MAC             : 00:50:79:66:68:00
LPORT           : 10018
RHOST:PORT      : 127.0.0.1:10019
MTU:            : 1500
```

PC2 IP:

```
PC2> ip 192.168.20.2 255.240.0.0
Checking for duplicate address...
PC1 : 192.168.20.2 255.240.0.0

PC2> save
Saving startup configuration to startup.vpc
.  done

PC2> show ip

NAME            : PC2[1]
IP/MASK         : 192.168.20.2/12
GATEWAY         : 255.240.0.0
DNS             :
MAC             : 00:50:79:66:68:01
LPORT           : 10020
RHOST:PORT      : 127.0.0.1:10021
MTU:            : 1500
```

R1 Interface Description:

```
R1#sh ip int brief
Interface                  IP-Address      OK? Method Status                Protocol
FastEthernet0/0            unassigned      YES unset  administratively down down
FastEthernet1/0            unassigned      YES unset  administratively down down
FastEthernet2/0            unassigned      YES unset  administratively down down
R1#
R1#
R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#interface f0/0
R1(config-if)#ip address 192.168.10.1 255.240.0.0
R1(config-if)#no shutdown
R1(config-if)#
*Sep 21 12:39:48.215: %LINK-3-UPDOWN: Interface FastEthernet0/0, changed state to up
*Sep 21 12:39:49.215: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up
R1(config-if)#exit
R1(config)#
R1(config)#
R1(config)#interface f1/0
R1(config-if)#ip address 193.168.30.1 255.255.255.0
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#
*Sep 21 12:40:23.459: %LINK-3-UPDOWN: Interface FastEthernet1/0, changed state to up
*Sep 21 12:40:24.459: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet1/0, changed state to up
R1(config)#exit
R1#
R1#
*Sep 21 12:40:31.503: %SYS-5-CONFIG_I: Configured from console by console
R1#sh ip int brief
Interface                  IP-Address      OK? Method Status                Protocol
FastEthernet0/0            192.168.10.1    YES manual up                    up
FastEthernet1/0            193.168.30.1    YES manual up                    up
FastEthernet2/0            unassigned      YES unset  administratively down down
```

R2 Interface Description:

```
R2#
R2#sh ip int brief
Interface                  IP-Address      OK? Method Status                Protocol
FastEthernet0/0            unassigned      YES unset  administratively down down
FastEthernet1/0            unassigned      YES unset  administratively down down
FastEthernet2/0            unassigned      YES unset  administratively down down
R2#
R2#
R2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R2(config)#interface f0/0
R2(config-if)#ip address 192.168.20.1 255.240.0.0
R2(config-if)#no shutdown
R2(config-if)#exit
*Sep 21 12:40:47.995: %LINK-3-UPDOWN: Interface FastEthernet0/0, changed state to up
*Sep 21 12:40:48.995: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up
R2(config-if)#exit
R2(config)#interface f1/0
R2(config-if)#ip address 193.168.30.2 255.255.255.0
R2(config-if)#no shutdown
R2(config-if)#exit
*Sep 21 12:41:19.503: %LINK-3-UPDOWN: Interface FastEthernet1/0, changed state to up
*Sep 21 12:41:20.503: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet1/0, changed state to up
R2(config-if)#exit
R2(config)#exit
R2#
R2#
*Sep 21 12:41:30.895: %SYS-5-CONFIG_I: Configured from console by console
R2#sh ip int brief
Interface                  IP-Address      OK? Method Status                Protocol
FastEthernet0/0            192.168.20.1    YES manual up                    up
FastEthernet1/0            193.168.30.2    YES manual up                    up
FastEthernet2/0            unassigned      YES unset  administratively down down
```

Routing:

R1:

```
R1#
R1#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       + - replicated route, % - next hop override

Gateway of last resort is not set

C      192.160.0.0/12 is directly connected, FastEthernet0/0
       192.168.10.0/32 is subnetted, 1 subnets
L         192.168.10.1 is directly connected, FastEthernet0/0
       193.168.30.0/24 is variably subnetted, 2 subnets, 2 masks
C         193.168.30.0/24 is directly connected, FastEthernet1/0
L         193.168.30.1/32 is directly connected, FastEthernet1/0
R1#
R1#
R1#ping 193.168.30.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 193.168.30.2, timeout is 2 seconds:
.!!!!
Success rate is 80 percent (4/5), round-trip min/avg/max = 64/86/96 ms
R1#
R1#
R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#ip route 192.168.20.0 255.255.255.0 193.168.30.2
R1(config)#exit
R1#
*Sep 21 12:45:26.555: %SYS-5-CONFIG_I: Configured from console by console
R1#ping 192.168.20.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.20.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 92/114/128 ms
R1#
```

R2:

```
R2#
R2#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       + - replicated route, % - next hop override

Gateway of last resort is not set

C      192.160.0.0/12 is directly connected, FastEthernet0/0
       192.168.20.0/32 is subnetted, 1 subnets
L         192.168.20.1 is directly connected, FastEthernet0/0
       193.168.30.0/24 is variably subnetted, 2 subnets, 2 masks
C         193.168.30.0/24 is directly connected, FastEthernet1/0
L         193.168.30.2/32 is directly connected, FastEthernet1/0
R2#
R2#ping 193.168.30.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 193.168.30.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 88/88/92 ms
R2#
R2#
R2#ip route 192.168.10.0 255.255.255.0 193.168.30.1
      ^
% Invalid input detected at '^' marker.

R2#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R2(config)#ip route 192.168.10.0 255.255.255.0 193.168.30.1
R2(config)#exit
R2#
*Sep 21 12:44:12.523: %SYS-5-CONFIG_I: Configured from console by console
R2#ping 192.168.10.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.10.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 84/88/100 ms
R2#
```

Device to Device Transfer:

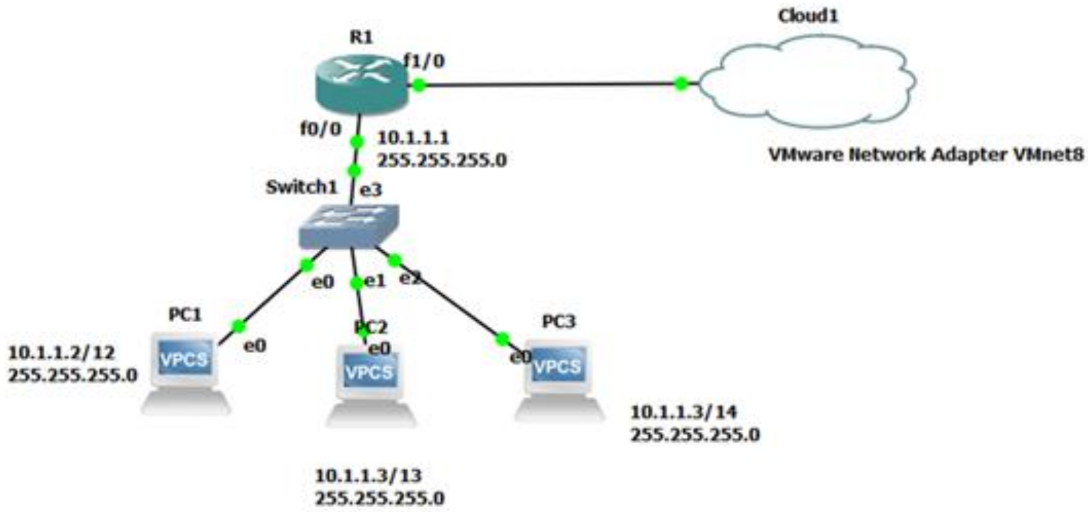
PC1:

```
PC1>
PC1> ping 192.168.20.2
84 bytes from 192.168.20.2 icmp_seq=1 ttl=62 time=107.368 ms
84 bytes from 192.168.20.2 icmp_seq=2 ttl=62 time=62.701 ms
84 bytes from 192.168.20.2 icmp_seq=3 ttl=62 time=63.576 ms
84 bytes from 192.168.20.2 icmp_seq=4 ttl=62 time=62.236 ms
84 bytes from 192.168.20.2 icmp_seq=5 ttl=62 time=61.512 ms

PC1>
```

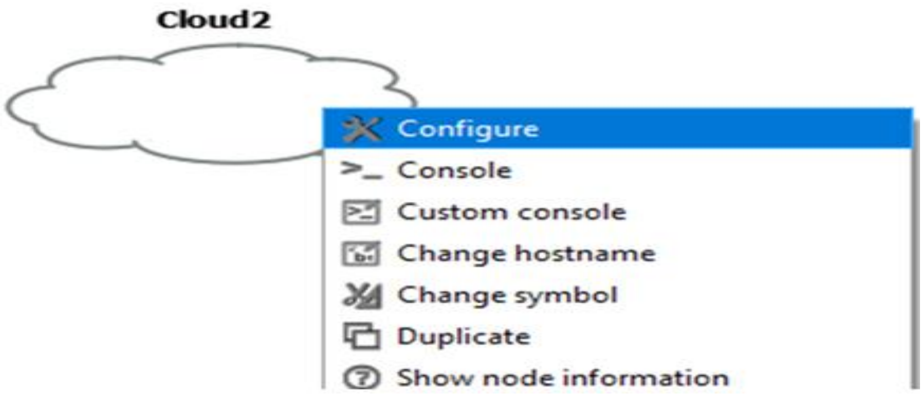
PC2:

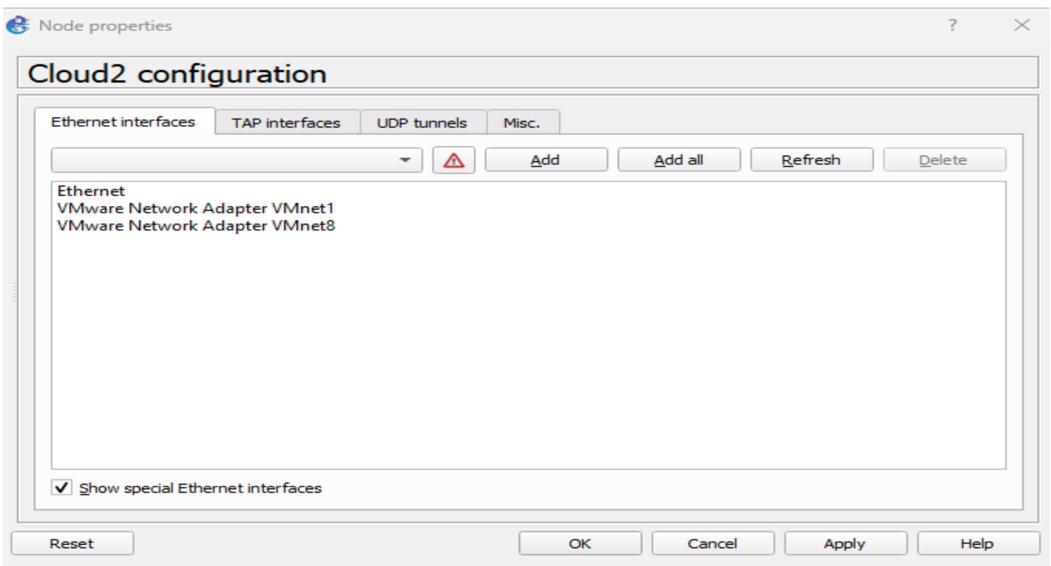
```
PC2> ping 192.168.10.2
84 bytes from 192.168.10.2 icmp_seq=1 ttl=62 time=63.797 ms
84 bytes from 192.168.10.2 icmp_seq=2 ttl=62 time=63.158 ms
84 bytes from 192.168.10.2 icmp_seq=3 ttl=62 time=61.654 ms
84 bytes from 192.168.10.2 icmp_seq=4 ttl=62 time=62.382 ms
84 bytes from 192.168.10.2 icmp_seq=5 ttl=62 time=63.321 ms

PC2>
```

Task III - Simulate a wired network and a wireless network, record the traffic pattern between these networks.



Configure Cloud:





Configure Router:

```
R1#
R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#int f1/0
R1(config-if)#ip add dhcp
R1(config-if)#no shut
R1(config-if)#exit
R1(config)#
*Nov  8 11:01:57.703: %LINK-3-UPDOWN: Interface FastEthernet1/0, changed state to up
*Nov  8 11:01:58.703: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet1/0, changed state to up
R1(config)#exit
R1#
*Nov  8 11:02:02.907: %SYS-5-CONFIG_I: Configured from console by console
R1#
*Nov  8 11:02:06.483: %DHCP-6-ADDRESS_ASSIGN: Interface FastEthernet1/0 assigned DHCP address 192.168.111.129, mask 255.255.
255.0, hostname R1

R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#int f0/0
R1(config-if)#ip add 10.1.1.1 255.255.255.0
R1(config-if)#no shut
R1(config-if)#exit
R1(config)#e
*Nov  8 11:02:36.343: %LINK-3-UPDOWN: Interface FastEthernet0/0, changed state to up
*Nov  8 11:02:37.343: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up
R1(config)#exit
R1#
*Nov  8 11:02:42.099: %SYS-5-CONFIG_I: Configured from console by console
R1#sh ip int brief
Interface                  IP-Address      OK? Method Status                Protocol
FastEthernet0/0            10.1.1.1        YES manual up                    up
FastEthernet1/0            192.168.111.129 YES DHCP   up                    up
FastEthernet2/0            unassigned      YES unset  administratively down down
FastEthernet3/0            unassigned      YES unset  administratively down down
FastEthernet3/1            unassigned      YES unset  administratively down down
```

Connect with Cloud:

```
R1#ping 8.8.8.8
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 8.8.8.8, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 20/31/40 ms
R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#ip domain-lookup
R1(config)#exit
R1#pin
*Nov  8 11:03:11.803: %SYS-5-CONFIG_I: Configured from console by console
R1#ping google.com
Translating "google.com"...domain server (192.168.111.2) [OK]

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 142.250.182.46, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 24/41/52 ms
R1#
*Nov  8 11:05:33.563: %LINK-3-UPDOWN: Interface FastEthernet0/0, changed state to down
*Nov  8 11:05:34.563: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to down
R1#
*Nov  8 11:06:10.563: %LINK-3-UPDOWN: Interface FastEthernet0/0, changed state to up
R1#
*Nov  8 11:06:11.563: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up
R1#
```

Configure PC's:

```
PC1> ip 10.1.1.2 10.1.1.1
Checking for duplicate address...
PC1 : 10.1.1.2 255.255.255.0 gateway 10.1.1.1

PC1> sh ip

NAME        : PC1[1]
IP/MASK     : 10.1.1.2/24
GATEWAY     : 10.1.1.1
DNS         :
MAC         : 00:50:79:66:68:00
LPORT       : 10008
RHOST:PORT  : 127.0.0.1:10009
MTU:        : 1500

PC1> save
Saving startup configuration to startup.vpc
.  done
```

Connect with Router:

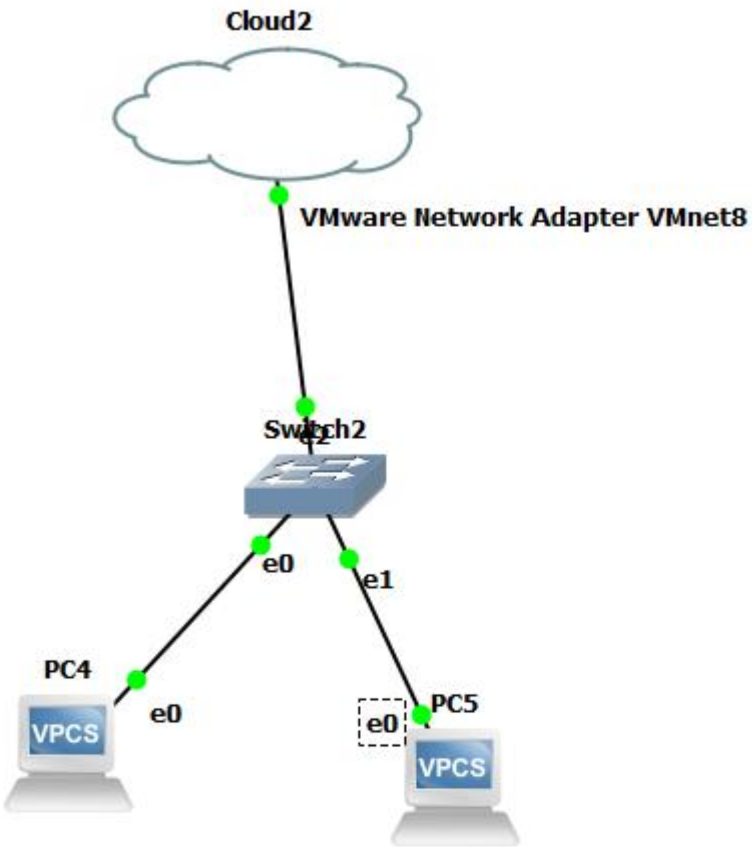
```
PC1> ping 10.1.1.1
84 bytes from 10.1.1.1 icmp_seq=1 ttl=255 time=15.352 ms
84 bytes from 10.1.1.1 icmp_seq=2 ttl=255 time=16.163 ms
84 bytes from 10.1.1.1 icmp_seq=3 ttl=255 time=15.325 ms
84 bytes from 10.1.1.1 icmp_seq=4 ttl=255 time=15.323 ms
84 bytes from 10.1.1.1 icmp_seq=5 ttl=255 time=15.772 ms
```

Connect with Cloud:

```
PC1> ping 8.8.8.8
84 bytes from 8.8.8.8 icmp_seq=1 ttl=127 time=76.816 ms
84 bytes from 8.8.8.8 icmp_seq=2 ttl=127 time=45.914 ms
84 bytes from 8.8.8.8 icmp_seq=3 ttl=127 time=92.769 ms
84 bytes from 8.8.8.8 icmp_seq=4 ttl=127 time=46.871 ms
84 bytes from 8.8.8.8 icmp_seq=5 ttl=127 time=46.712 ms

PC1> ping 1925.168.111.2
Cannot resolve 1925.168.111.2

PC1> ping 192.168.111.2
84 bytes from 192.168.111.2 icmp_seq=1 ttl=127 time=46.146 ms
84 bytes from 192.168.111.2 icmp_seq=2 ttl=127 time=31.468 ms
84 bytes from 192.168.111.2 icmp_seq=3 ttl=127 time=31.849 ms
84 bytes from 192.168.111.2 icmp_seq=4 ttl=127 time=31.655 ms
84 bytes from 192.168.111.2 icmp_seq=5 ttl=127 time=31.174 ms

PC1> ping 205.251.242.103
84 bytes from 205.251.242.103 icmp_seq=1 ttl=127 time=246.345 ms
84 bytes from 205.251.242.103 icmp_seq=2 ttl=127 time=246.253 ms
84 bytes from 205.251.242.103 icmp_seq=3 ttl=127 time=243.765 ms
84 bytes from 205.251.242.103 icmp_seq=4 ttl=127 time=245.532 ms
84 bytes from 205.251.242.103 icmp_seq=5 ttl=127 time=232.063 ms

PC1>
```

Similarly do it for remaining 7 PC's.

Task IV - Simulate a wireless network with 50 nodes and analysis the traffic using UDP.



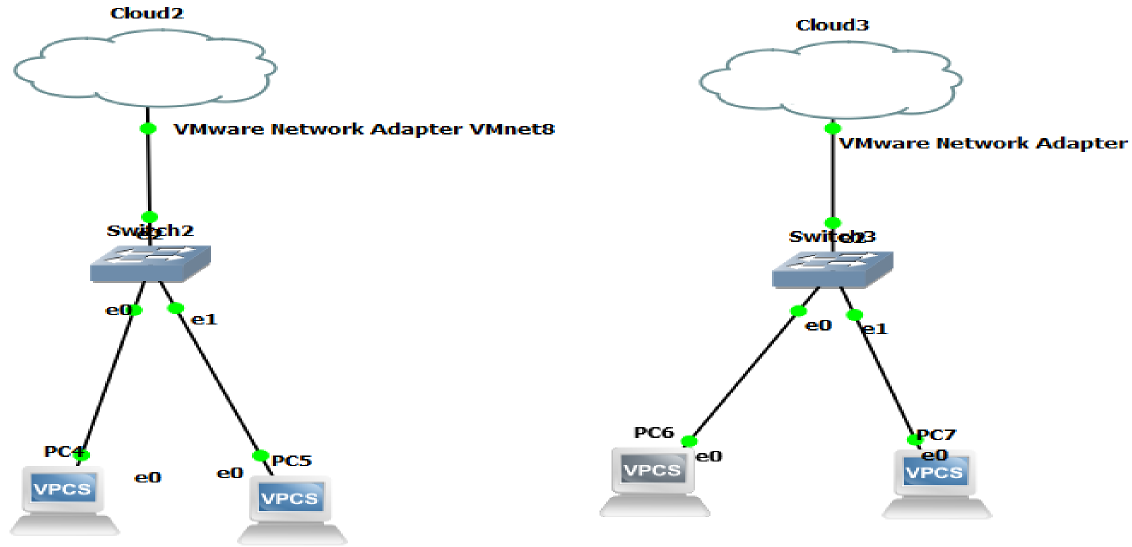
Configure PC1:

```
PC4> ip dhcp
DDORA IP 192.168.111.130/24 GW 192.168.111.2

PC4> sh ip

NAME        : PC4[1]
IP/MASK     : 192.168.111.130/24
GATEWAY     : 192.168.111.2
DNS         : 192.168.111.2
DHCP SERVER : 192.168.111.254
DHCP LEASE  : 1796, 1800/900/1575
DOMAIN NAME : localdomain
MAC         : 00:50:79:66:68:03
LPORT       : 10026
RHOST:PORT  : 127.0.0.1:10027
MTU:        : 1500

PC4> save
Saving startup configuration to startup.vpc
.  done
```

Connect with Cloud:

```
PC4> ping 8.8.8.8
84 bytes from 8.8.8.8 icmp_seq=1 ttl=128 time=16.690 ms
84 bytes from 8.8.8.8 icmp_seq=2 ttl=128 time=17.376 ms
84 bytes from 8.8.8.8 icmp_seq=3 ttl=128 time=16.997 ms
84 bytes from 8.8.8.8 icmp_seq=4 ttl=128 time=17.104 ms
84 bytes from 8.8.8.8 icmp_seq=5 ttl=128 time=16.954 ms

PC4> ping 205.251.242.103
84 bytes from 205.251.242.103 icmp_seq=1 ttl=128 time=216.577 ms
84 bytes from 205.251.242.103 icmp_seq=2 ttl=128 time=216.456 ms
84 bytes from 205.251.242.103 icmp_seq=3 ttl=128 time=215.748 ms
84 bytes from 205.251.242.103 icmp_seq=4 ttl=128 time=215.809 ms
84 bytes from 205.251.242.103 icmp_seq=5 ttl=128 time=216.002 ms
```

Task V - Simulate two wireless networks with 50 nodes each and analysis the traffic with others using UDP.



Configure PC4:

```
PC4>
PC4> ip dhcp
DDORA IP 192.168.111.132/24 GW 192.168.111.2

PC4> sh ip

NAME        : PC4[1]
IP/MASK     : 192.168.111.132/24
GATEWAY     : 192.168.111.2
DNS         : 192.168.111.2
DHCP SERVER : 192.168.111.254
DHCP LEASE  : 1796, 1800/900/1575
DOMAIN NAME : localdomain
MAC         : 00:50:79:66:68:03
LPORT       : 10032
RHOST:PORT  : 127.0.0.1:10033
MTU:        : 1500

PC4> save
Saving startup configuration to startup.vpc
.  done
```

Connect with Internet:

```
PC4> ping 8.8.8.8
84 bytes from 8.8.8.8 icmp_seq=1 ttl=128 time=18.845 ms
84 bytes from 8.8.8.8 icmp_seq=2 ttl=128 time=18.714 ms
84 bytes from 8.8.8.8 icmp_seq=3 ttl=128 time=18.865 ms
84 bytes from 8.8.8.8 icmp_seq=4 ttl=128 time=19.340 ms
84 bytes from 8.8.8.8 icmp_seq=5 ttl=128 time=19.092 ms

PC4> ping 205.251.242.103
84 bytes from 205.251.242.103 icmp_seq=1 ttl=128 time=204.077 ms
84 bytes from 205.251.242.103 icmp_seq=2 ttl=128 time=204.118 ms
84 bytes from 205.251.242.103 icmp_seq=3 ttl=128 time=204.798 ms
84 bytes from 205.251.242.103 icmp_seq=4 ttl=128 time=204.483 ms
84 bytes from 205.251.242.103 icmp_seq=5 ttl=128 time=204.371 ms
```

Configure PC6:

```
PC6> ip dhcp
DDORA IP 192.168.111.131/24 GW 192.168.111.2

PC6> sh ip

NAME        : PC6[1]
IP/MASK     : 192.168.111.131/24
GATEWAY     : 192.168.111.2
DNS         : 192.168.111.2
DHCP SERVER : 192.168.111.254
DHCP LEASE  : 1797, 1800/900/1575
DOMAIN NAME : localdomain
MAC         : 00:50:79:66:68:05
LPORT       : 10036
RHOST:PORT  : 127.0.0.1:10037
MTU:        : 1500

PC6> save
Saving startup configuration to startup.vpc
.  done
```

Connect with Internet:

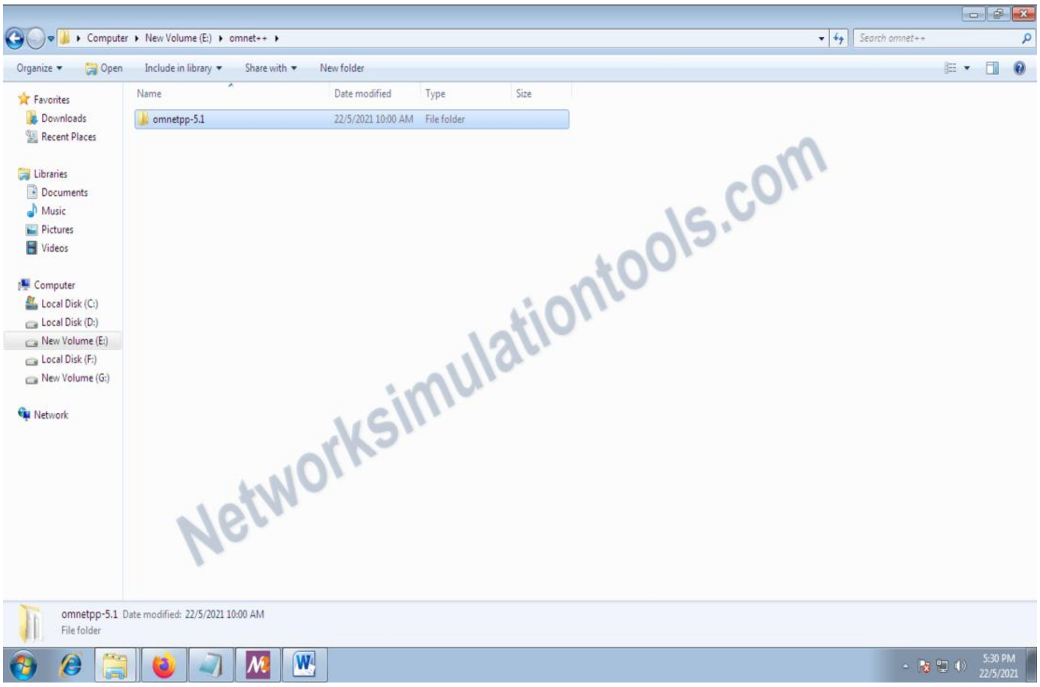
```
PC6> ping 8.8.8.8
84 bytes from 8.8.8.8 icmp_seq=1 ttl=128 time=19.637 ms
84 bytes from 8.8.8.8 icmp_seq=2 ttl=128 time=19.051 ms
84 bytes from 8.8.8.8 icmp_seq=3 ttl=128 time=19.455 ms
84 bytes from 8.8.8.8 icmp_seq=4 ttl=128 time=24.639 ms
84 bytes from 8.8.8.8 icmp_seq=5 ttl=128 time=18.832 ms
```

Connect with PC4:

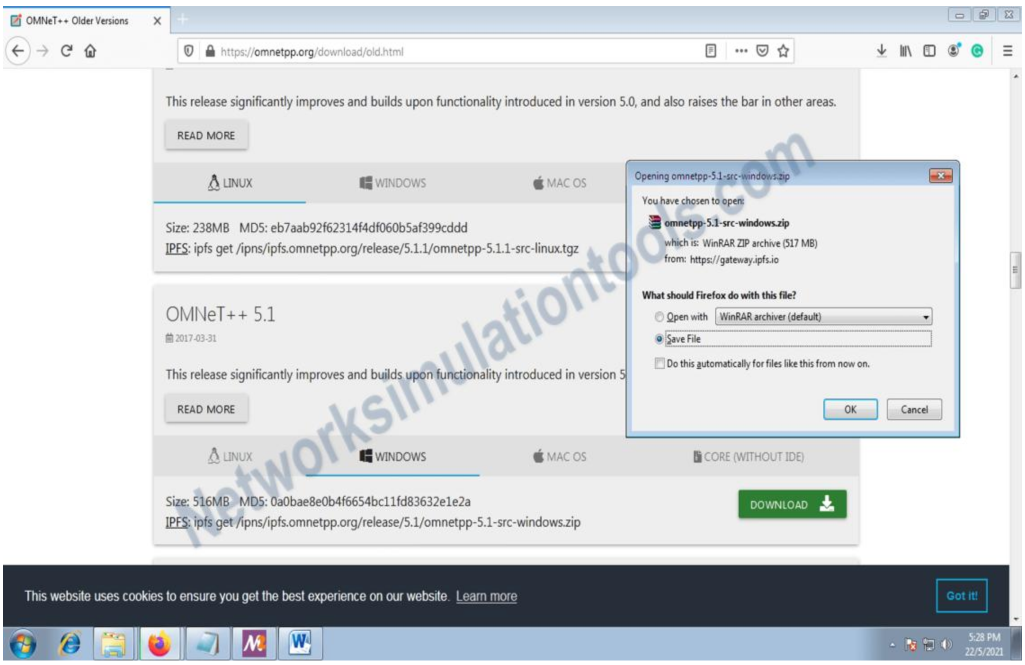
```
PC6> ping  192.168.111.132
84 bytes from 192.168.111.132 icmp_seq=1 ttl=64 time=1.445 ms
84 bytes from 192.168.111.132 icmp_seq=2 ttl=64 time=1.523 ms
84 bytes from 192.168.111.132 icmp_seq=3 ttl=64 time=1.283 ms
84 bytes from 192.168.111.132 icmp_seq=4 ttl=64 time=1.403 ms
84 bytes from 192.168.111.132 icmp_seq=5 ttl=64 time=1.363 ms
```

TASK VI - Simulate a MANET and check the traffic effect. Record the packet loss when inmotion.

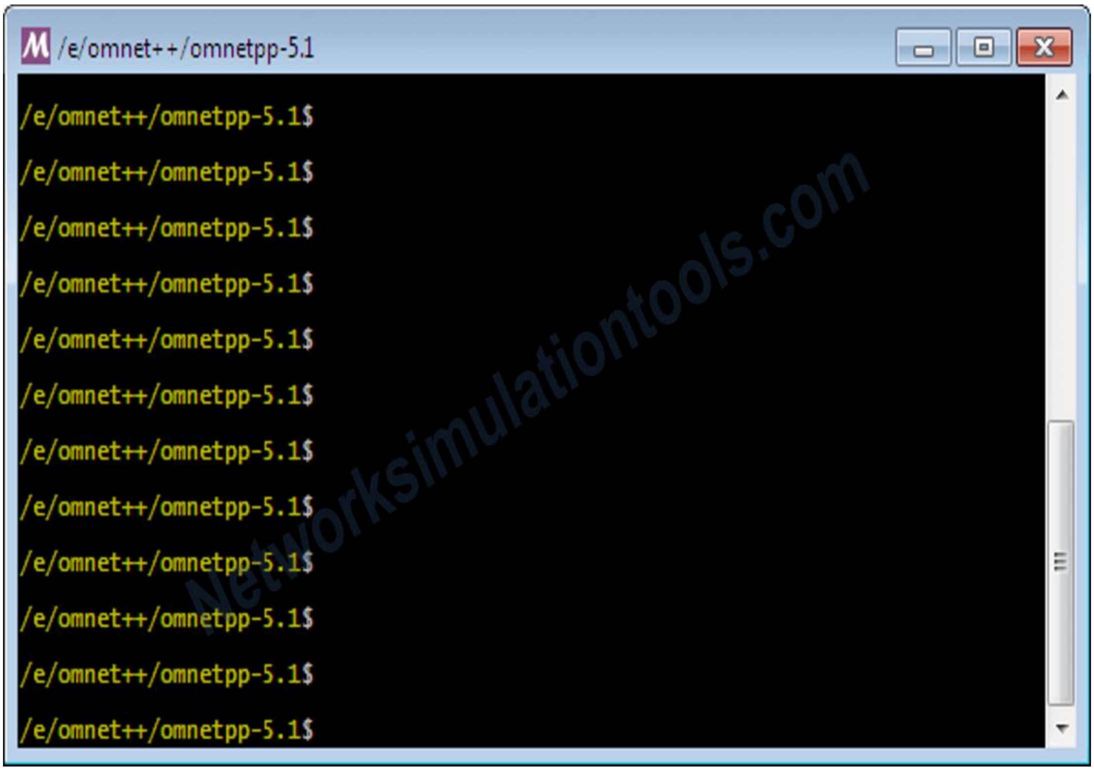
1. **System requirement**
   To perform the installation process, we need the following requirements,
   1) OS: Windows 10 pro
   2) Install Omnet++ 5.1



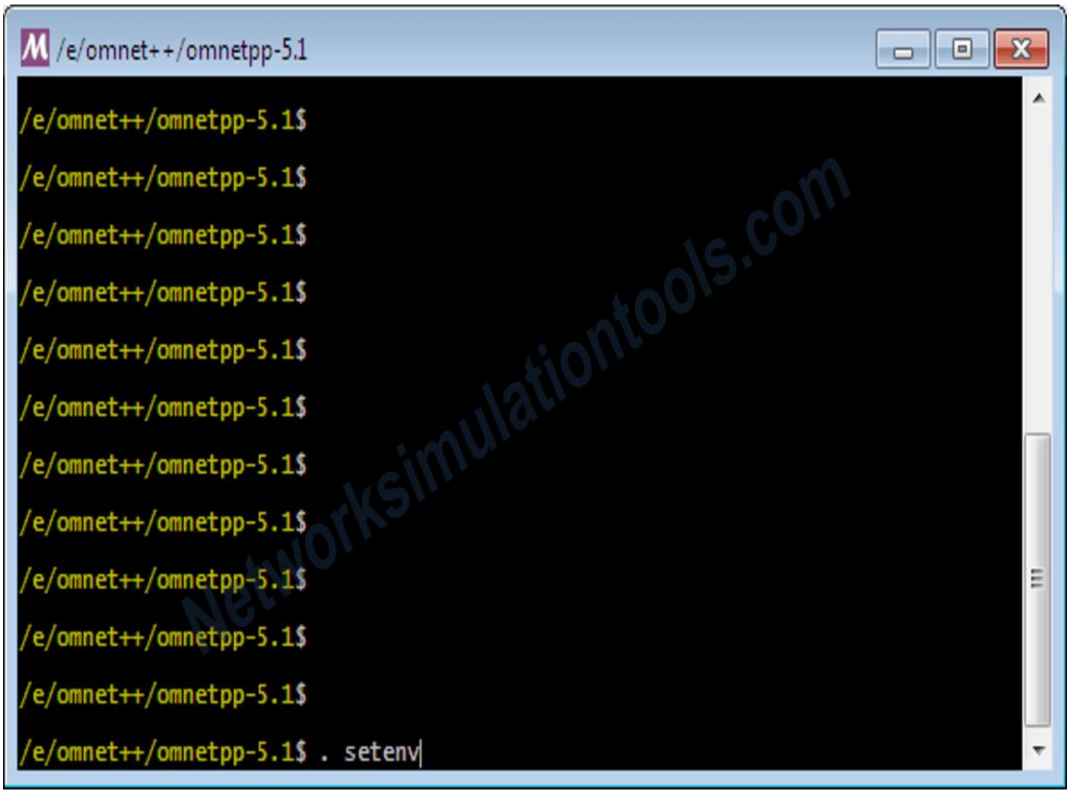
2. **Download the omnet++ package**
   Download the omnet++ package from the following url, in the site there multiple verion of the omnet++ is listed. Select your needed version of omnet++, here we select the eversion omnet++ 5.1 , after that click the download button , which is displayed in the below of selected version
   https://omnetpp.org/download/old.html



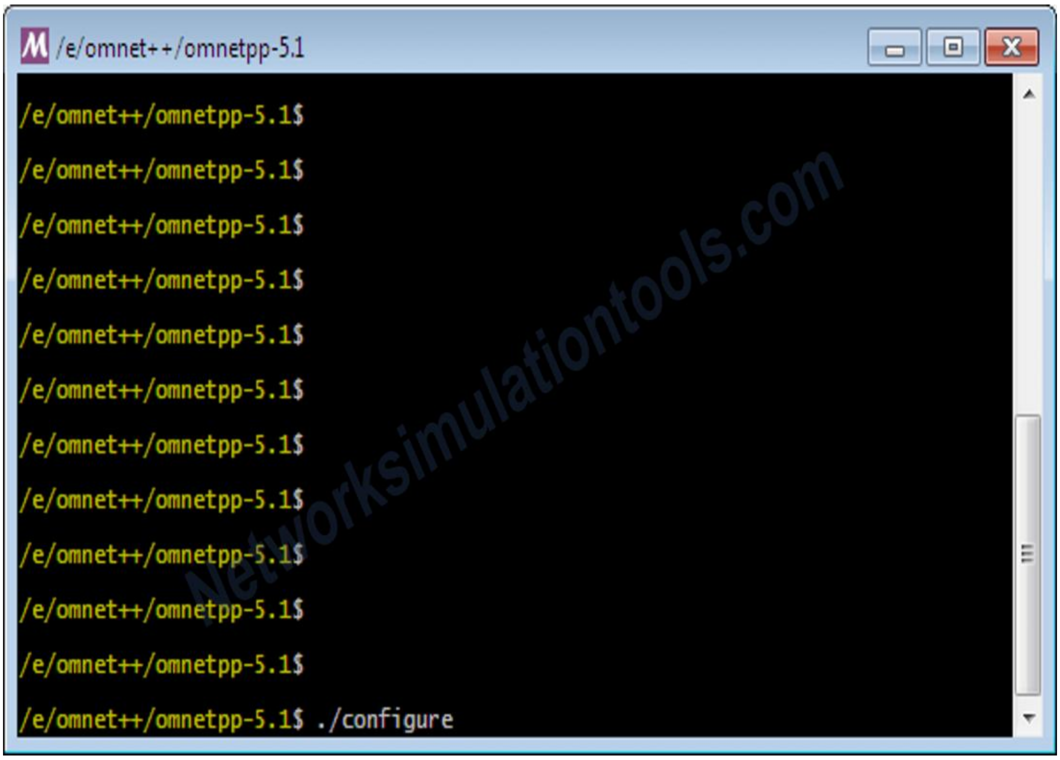
3. **Open the terminal/ mingwenv window**
   Open the mingwenv window. By double click mingwenv window from the omnet++ installed location.



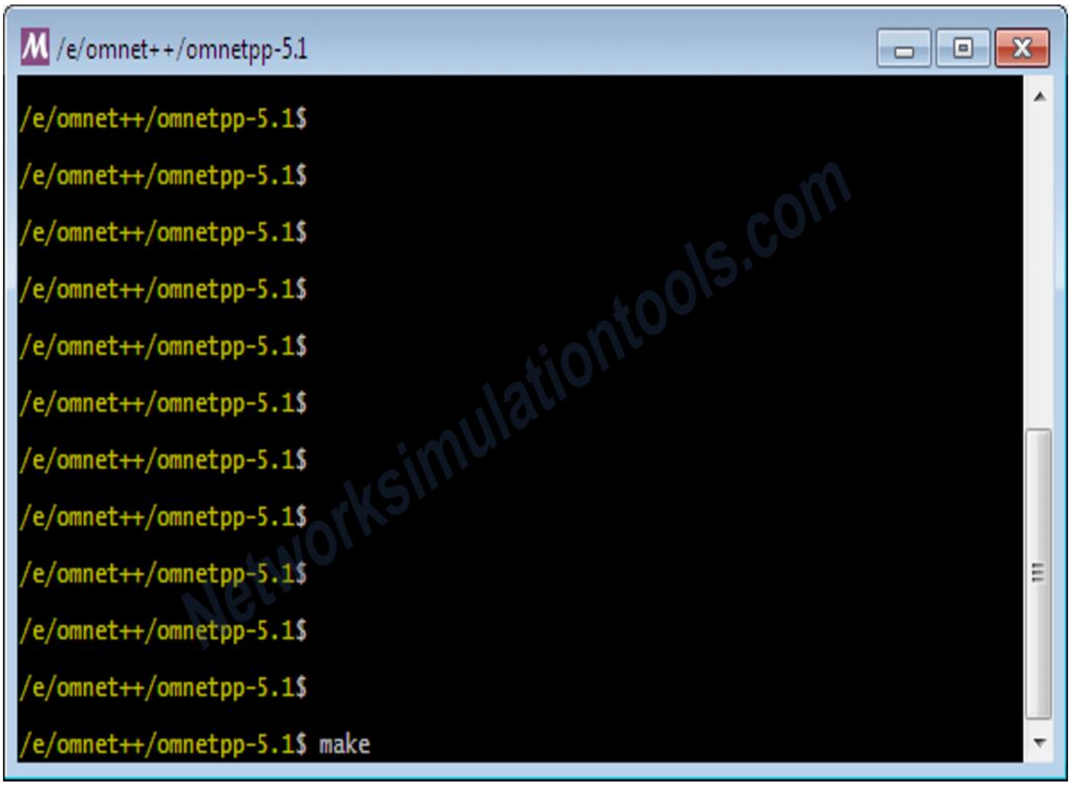
4. **Execute the command window**
   In the mingwenv command window ,
   Execute the command -> . setenv



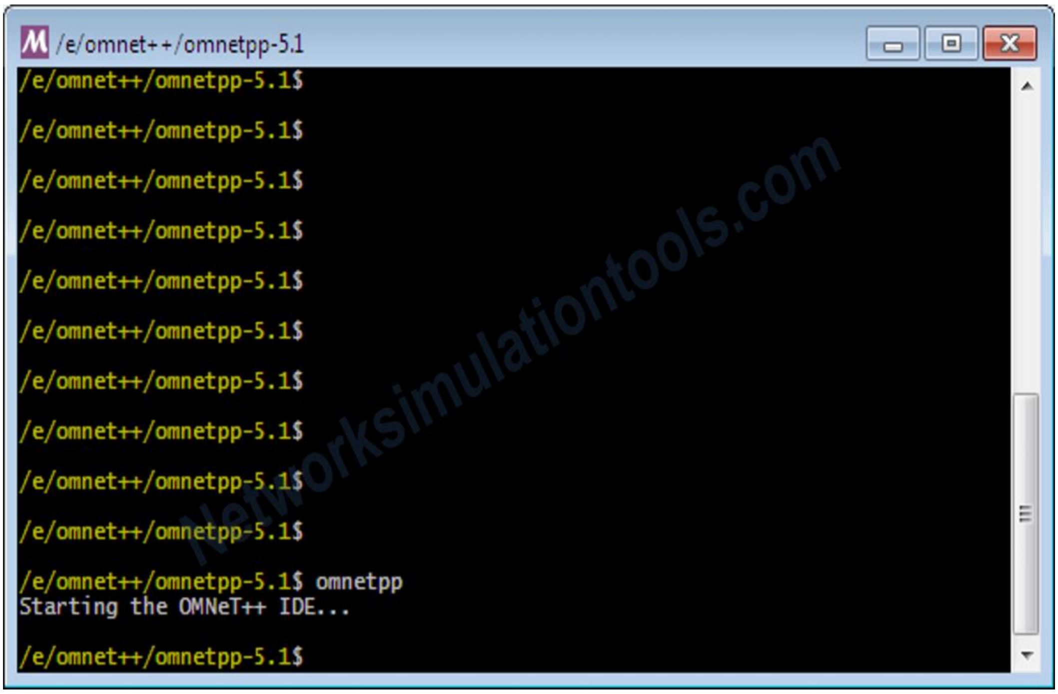
5. **Execute the configure command window**
   In the mingwenv command window ,
   Execute the command -> ./configure



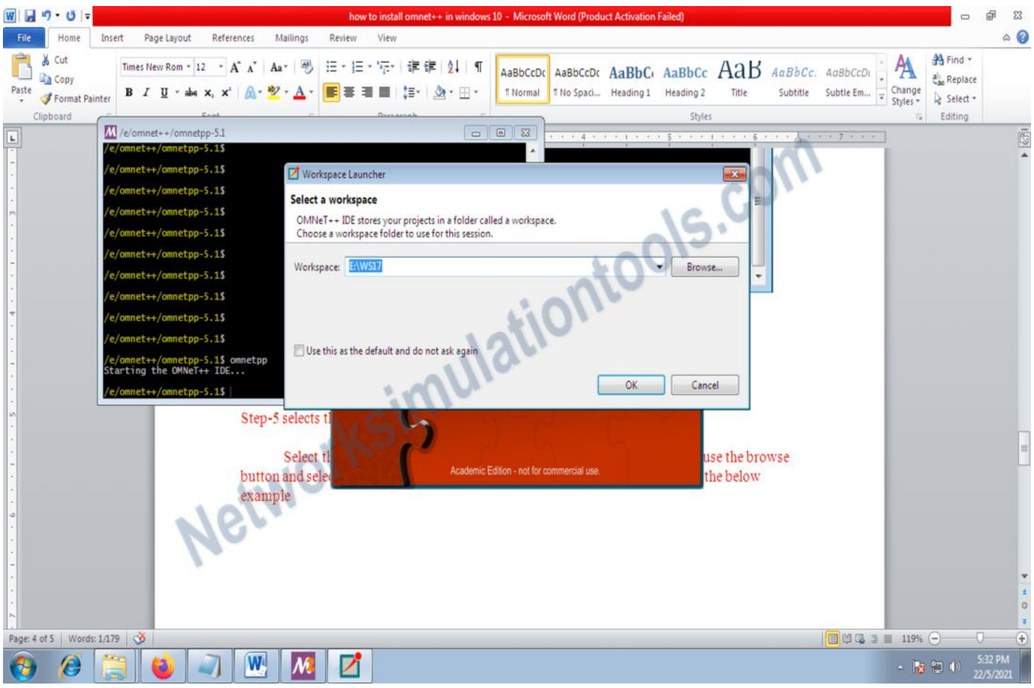
6. **Execute the Make command window**
   In the mingwenv command window ,
   Execute the command -> make



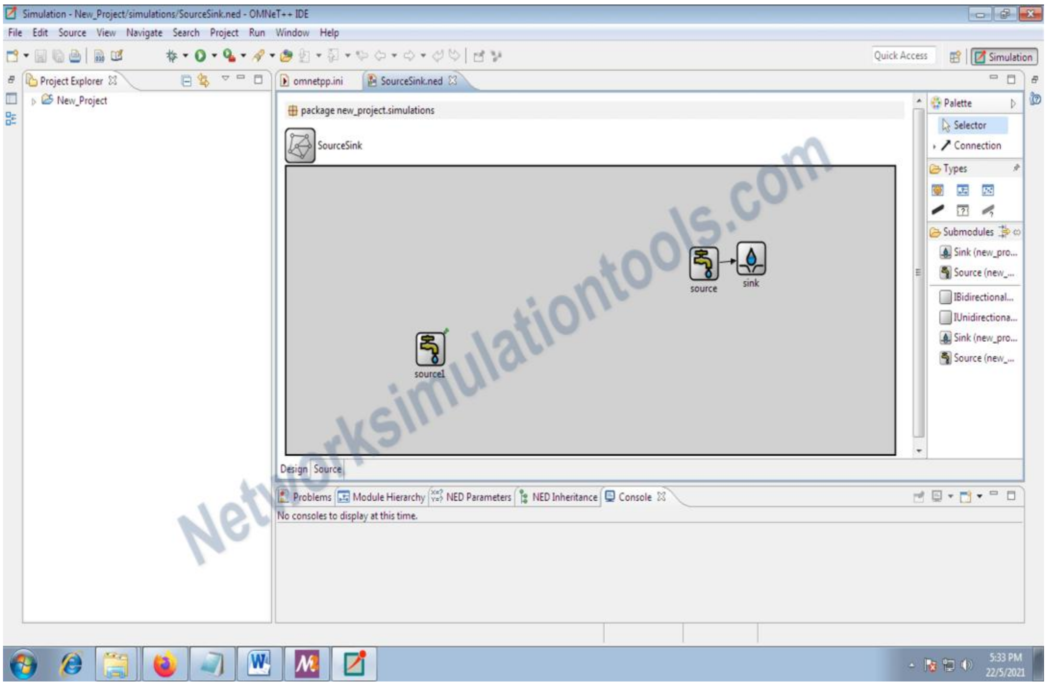
7. **Execute the omnet++ command window**
   In the mingwenv command window ,
   Execute the command -> omnetpp



8. **Select the workspace window**
   Select the stored workspace, with full location, to select the workspace , use the browse button and select the location and press ok button or type the full location . , like the below example



9. **Loaded workspace window**
   By use the browse button and select the location and press ok button or type the full location., like the below example
   View the loaded workspace window



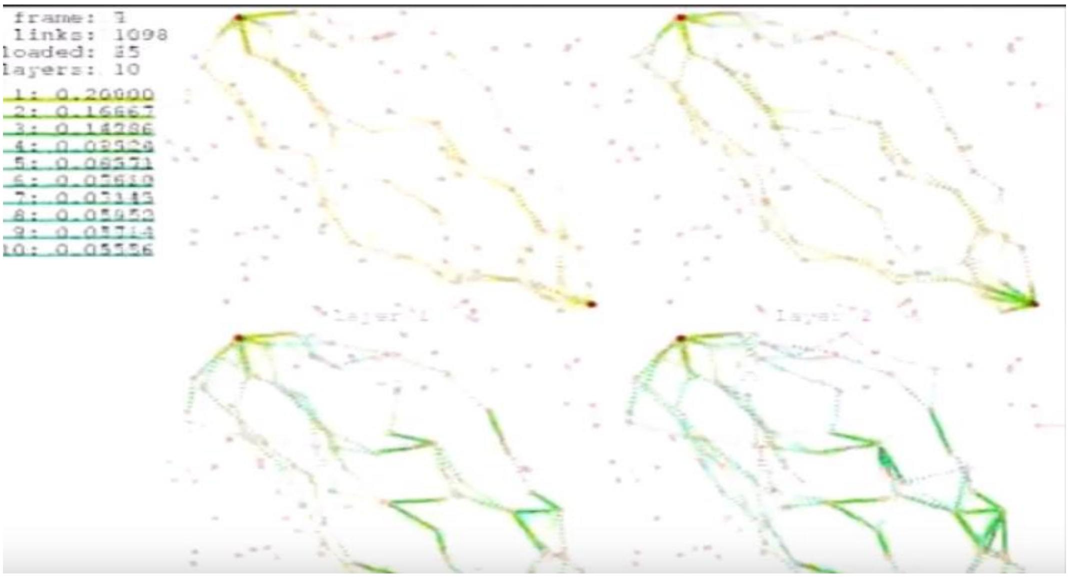
Simulation - New_Project/simulations/SourceSink.ned - OMNeT++ IDE
File Edit Source View Navigate Search Project Run Window Help
Project Explorer
New_Project
omnetpp.ini
SourceSink.ned
package new_project.simulations
SourceSink
source
sink
source1
Palette
Selector
Connection
Types
Submodules
Design Source
Problems
Module Hierarchy
NED Parameters
NED Inheritance
Console
No consoles to display at this time.
Networksimulationtools.com

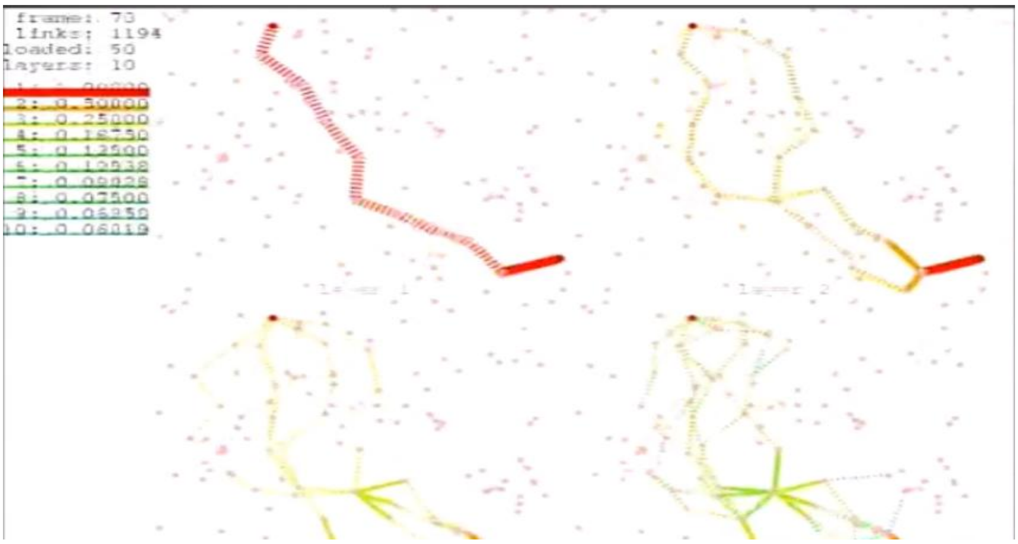
Task_5:

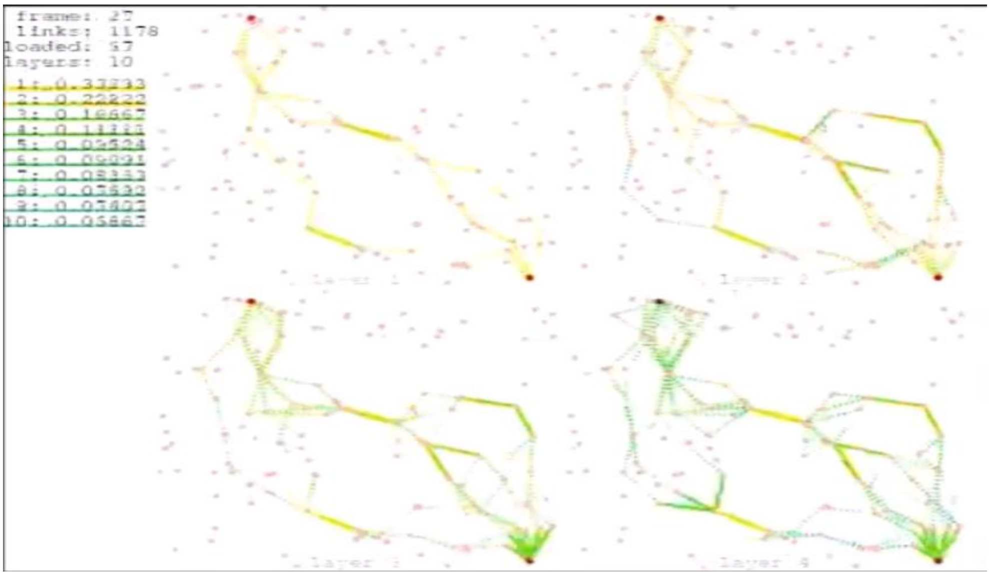
```
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/applications-module.h"
#include "ns3/mobility-module.h"
#include "ns3/netanim-module.h"
#include "ns3/wifi-module.h"

using namespace ns3;

int main (int argc, char *argv[]) {
   // Create nodes
   NodeContainer nodes;
   nodes.Create(5);

   // Install wireless devices and mobility models
   WifiHelper wifi;
   wifi.SetStandard(WIFI_PHY_STANDARD_80211a);
   YansWifiPhyHelper wifiPhy = YansWifiPhyHelper::Default();
   YansWifiChannelHelper wifiChannel = YansWifiChannelHelper::Default();
   wifiPhy.SetChannel(wifiChannel.Create());
   WifiMacHelper wifiMac;
   wifiMac.SetType("ns3::AdhocWifiMac");
   NetDeviceContainer devices = wifi.Install(wifiPhy, wifiMac, nodes);

   MobilityHelper mobility;
   mobility.SetPositionAllocator("ns3::GridPositionAllocator",
                        "MinX", DoubleValue(0.0),
                        "MinY", DoubleValue(0.0),
                        "DeltaX", DoubleValue(5.0),
                        "DeltaY", DoubleValue(5.0),
                        "GridWidth", UintegerValue(5),
                        "LayoutType", StringValue("RowFirst"));
   mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
   mobility.Install(nodes);

   // Install internet stack
   InternetStackHelper internet;
   internet.Install(nodes);

   // Set up routing (e.g., AODV, DSR, etc.)
   Ipv4StaticRoutingHelper staticRouting;
   Ipv4ListRoutingHelper listRouting;
   listRouting.Add(staticRouting, 0);
   internet.SetRoutingHelper(listRouting);

   // Create applications (e.g., OnOff applications)
   OnOffHelper onoff("ns3::UdpSocketFactory", Address());
   onoff.SetAttribute("OnTime", StringValue("ns3::ConstantRandomVariable[Constant=1]"));
   onoff.SetAttribute("OffTime",
StringValue("ns3::ConstantRandomVariable[Constant=1]"));
```

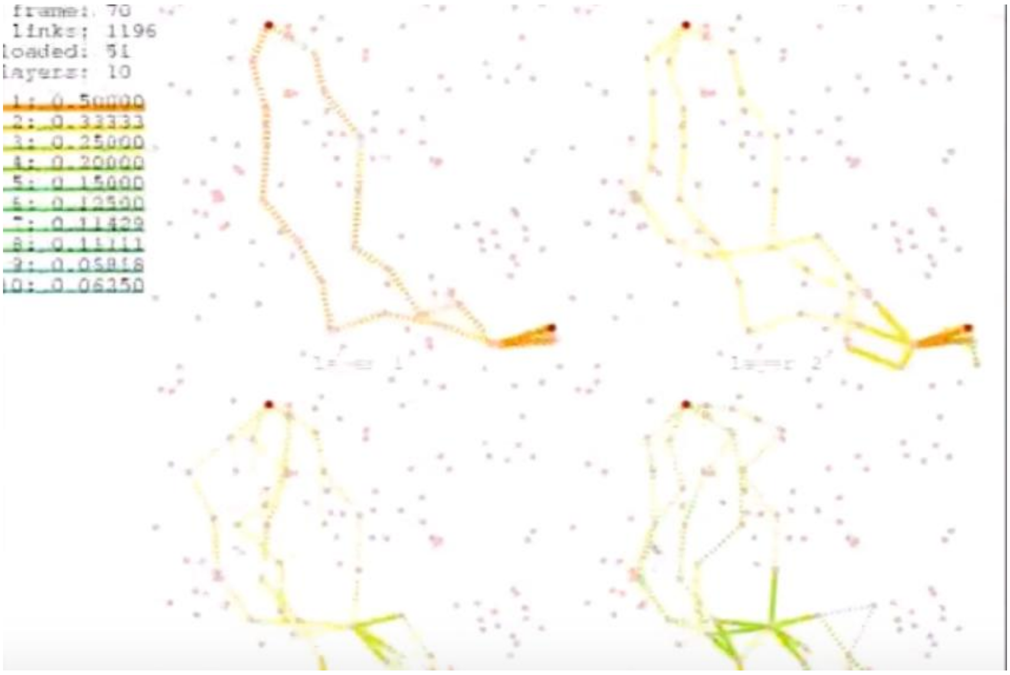
```
    ApplicationContainer apps = onoff.Install(nodes.Get(0));
    apps.Start(Seconds(1.0));
    apps.Stop(Seconds(10.0));

    // Create a NetAnim output file
    AnimationInterface anim("output.xml");

    // Run the simulation
    Simulator::Stop(Seconds(10.0));
    Simulator::Run();
    Simulator::Destroy();

    return 0;
}
```

Output:






frame: 27
links: 1178
loaded: 57
layers: 10
1: 0.33333
2: 0.22222
3: 0.16667

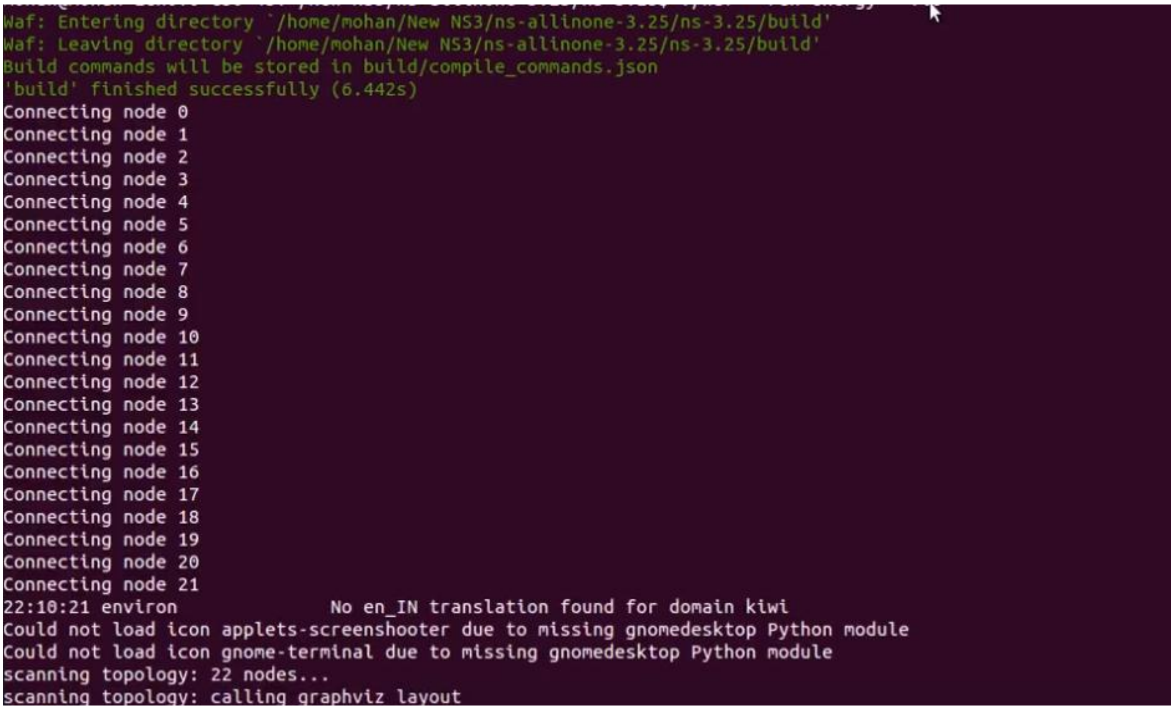

frame: 70
links: 1196
loaded: 51
layers: 10
1: 0.50000
2: 0.33333
3: 0.25000
4: 0.20000
5: 0.15000
6: 0.12500
7: 0.11429
8: 0.11111
9: 0.05818
10: 0.06350

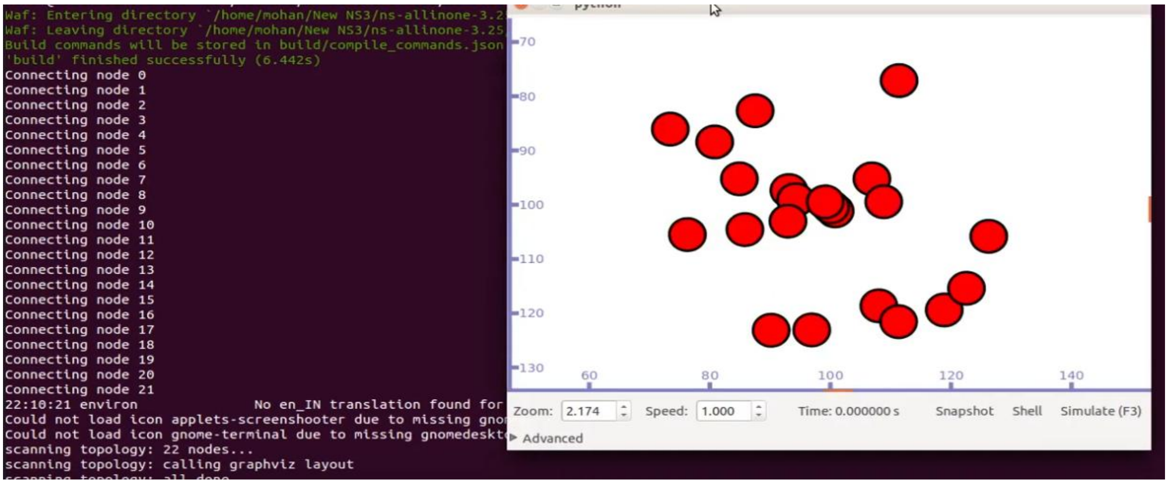
Task VII - Simulate a WSN network with 100 nodes. Analysis the Cluster using LEACH algorithm.

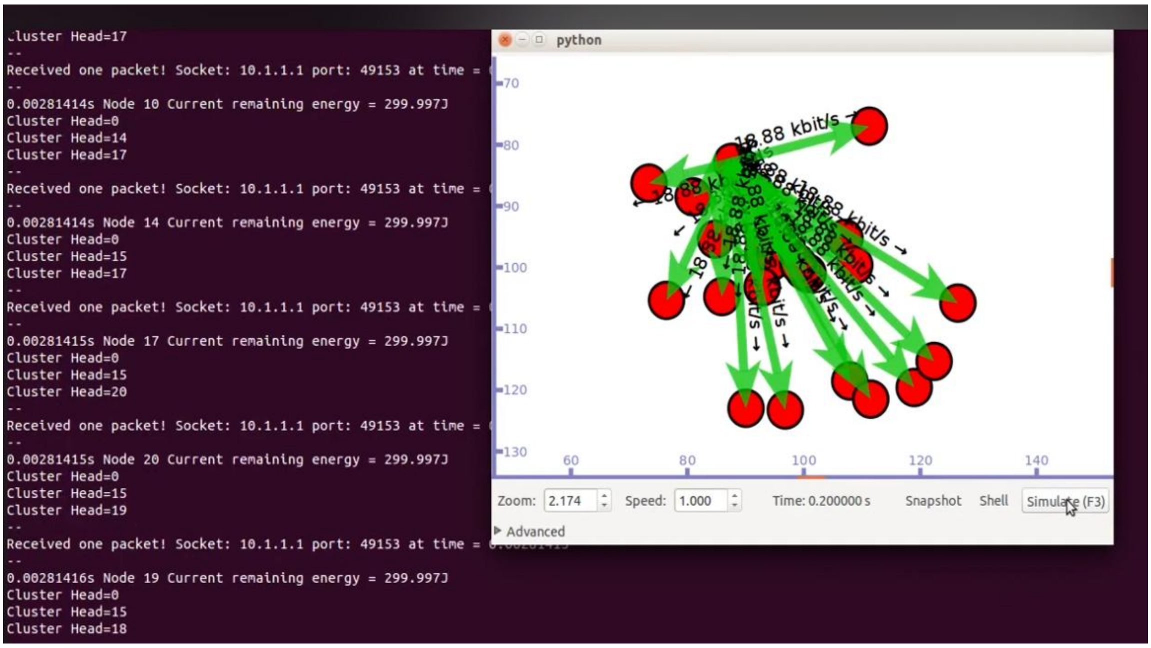
```
#include "ns3/core-module.h"
#include "ns3/network-module.h"
#include "ns3/internet-module.h"
#include "ns3/wifi-module.h"
#include "ns3/mobility-module.h"
#include "ns3/applications-module.h"
#include "ns3/leach-module.h"

using namespace ns3;

int main (int argc, char *argv[]) {
    // Initialize NS-3
    LogComponentEnable("UdpEchoClientApplication", LOG_LEVEL_INFO);
    LogComponentEnable("UdpEchoServerApplication", LOG_LEVEL_INFO);

    NodeContainer nodes;
    nodes.Create(100);  // Create 100 sensor nodes

    // Set up wireless channel and PHY
    YansWifiChannelHelper channel = YansWifiChannelHelper::Default();
    YansWifiPhyHelper phy = YansWifiPhyHelper::Default();
    phy.SetChannel(channel.Create());

    // Set up WiFi
    WifiHelper wifi = WifiHelper::Default();
    wifi.SetRemoteStationManager("ns3::AarfWifiManager");

    // Set up MAC
    NqosWifiMacHelper mac = NqosWifiMacHelper::Default();

    // Install wireless devices
    NetDeviceContainer devices = wifi.Install(phy, mac, nodes);

    // Set up mobility model
    MobilityHelper mobility;
    mobility.SetPositionAllocator("ns3::GridPositionAllocator",
                                  "MinX", DoubleValue(0.0),
                                  "MinY", DoubleValue(0.0),
                                  "DeltaX", DoubleValue(10.0),
                                  "DeltaY", DoubleValue(10.0),
                                  "GridWidth", UintegerValue(10),
                                  "LayoutType", StringValue("RowFirst"));
    mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
    mobility.Install(nodes);

    // Set up LEACH
    Ptr<LEACHClusterHead> leach = CreateObject<LEACHClusterHead>();
    leach->Set("PercentageHeads", DoubleValue(0.05));
    leach->Set("MaximumClusters", UintegerValue(20));
    leach->Set("MaximumDistance", DoubleValue(30.0));
```

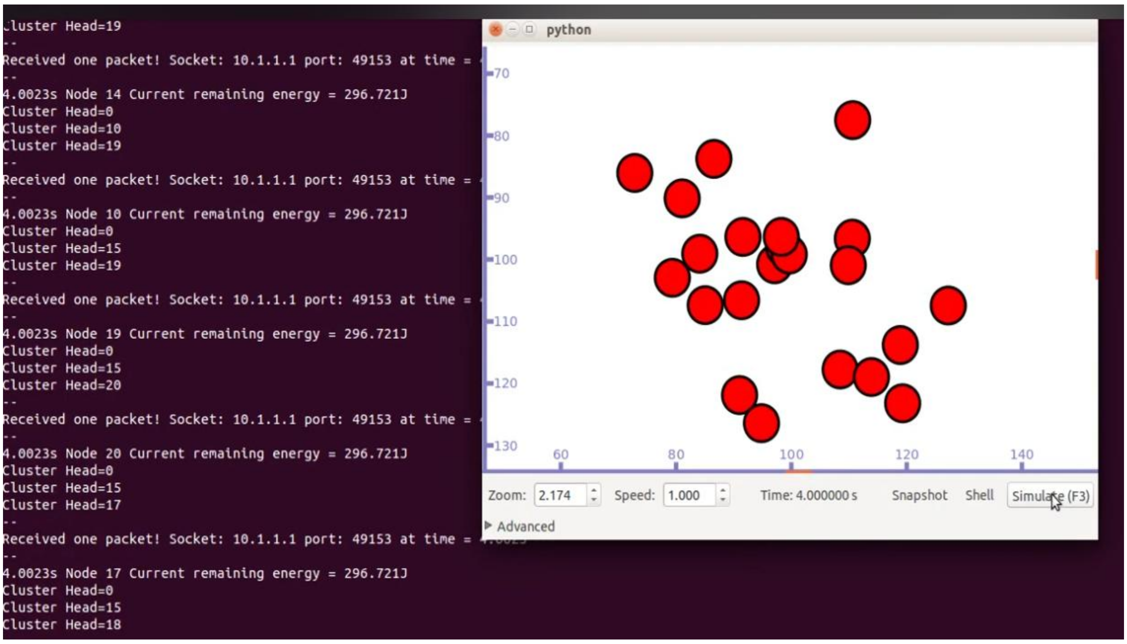
```
  leach->Set("MinimumDistance", DoubleValue(10.0));

  // Install LEACH on nodes
  leach->Install(nodes);

  // Set up the Internet stack
  InternetStackHelper internet;
  internet.Install(nodes);

  // Create a simple UDP client and server
  UdpEchoServerHelper echoServer(9);
  ApplicationContainer serverApps = echoServer.Install(nodes.Get(0)); // Install server on
the first node
  serverApps.Start(Seconds(1.0));
  serverApps.Stop(Seconds(100.0));

  UdpEchoClientHelper echoClient(nodes.Get(0)->GetObject<Ipv4>(),
Ipv4Address("1.1.1.1"), 9);
  echoClient.SetAttribute("MaxPackets", UintegerValue(1));
  echoClient.SetAttribute("Interval", TimeValue(Seconds(1.0)));
  echoClient.SetAttribute("PacketSize", UintegerValue(1024));
  ApplicationContainer clientApps = echoClient.Install(nodes.Get(1)); // Install client on the
second node
  clientApps.Start(Seconds(2.0));
  clientApps.Stop(Seconds(100.0));

  // Run the simulation
  Simulator::Stop(Seconds(100.0));
  Simulator::Run();
  Simulator::Destroy();

  return 0;
}
```

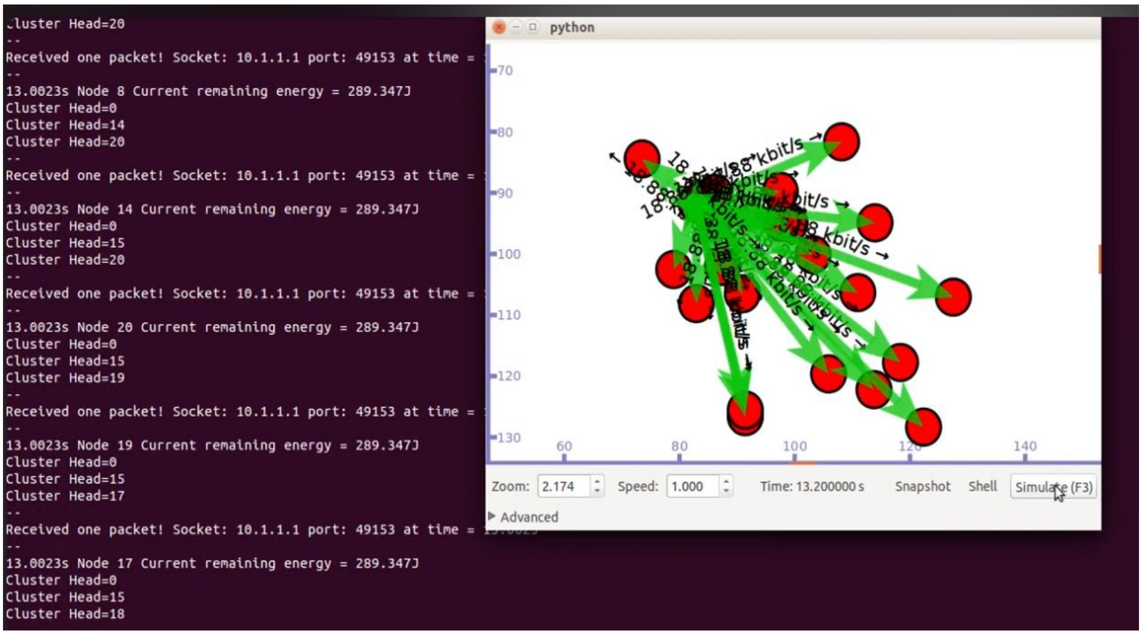
Output:

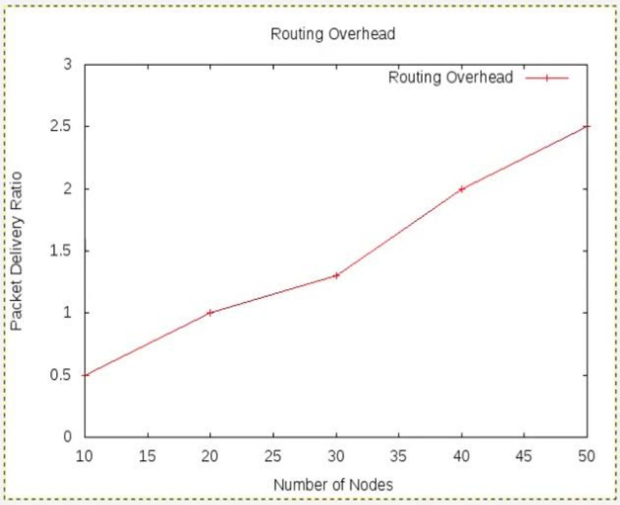



Waf: Entering directory `/home/mohan/New NS3/ns-allinone-3.2
Waf: Leaving directory `/home/mohan/New NS3/ns-allinone-3.25
Build commands will be stored in build/compile_commands.json
'build' finished successfully (6.442s)
Connecting node 0
Connecting node 1
Connecting node 2
Connecting node 3
Connecting node 4
Connecting node 5
Connecting node 6
Connecting node 7
Connecting node 8
Connecting node 9
Connecting node 10
Connecting node 11
Connecting node 12
Connecting node 13
Connecting node 14
Connecting node 15
Connecting node 16
Connecting node 17
Connecting node 18
Connecting node 19
Connecting node 20
Connecting node 21
22:10:21 environ No en_IN translation found for
Could not load icon applets-screenshooter due to missing gno
Could not load icon gnome-terminal due to missing gnomedeskt
scanning topology: 22 nodes...
scanning topology: calling graphviz layout
70
80
90
100
110
120
130
60
80
100
120
140
Zoom: 2.174
Speed: 1.000
Time: 0.000000 s
Snapshot
Shell
Simulate (F3)
Advanced


Cluster Head=17
--
Received one packet! Socket: 10.1.1.1 port: 49153 at time =
--
0.00281414s Node 10 Current remaining energy = 299.997J
Cluster Head=0
Cluster Head=14
Cluster Head=17
--
Received one packet! Socket: 10.1.1.1 port: 49153 at time =
--
0.00281414s Node 14 Current remaining energy = 299.997J
Cluster Head=0
Cluster Head=15
Cluster Head=17
--
Received one packet! Socket: 10.1.1.1 port: 49153 at time =
--
0.00281415s Node 17 Current remaining energy = 299.997J
Cluster Head=0
Cluster Head=15
Cluster Head=20
--
Received one packet! Socket: 10.1.1.1 port: 49153 at time =
--
0.00281415s Node 20 Current remaining energy = 299.997J
Cluster Head=0
Cluster Head=15
Cluster Head=19
--
Received one packet! Socket: 10.1.1.1 port: 49153 at time =
--
0.00281416s Node 19 Current remaining energy = 299.997J
Cluster Head=0
Cluster Head=15
Cluster Head=18
python
18.88 kbit/s
Zoom: 2.174
Speed: 1.000
Time: 0.200000 s
Snapshot
Shell
Simulate (F3)
Advanced


Cluster Head=19
--
Received one packet! Socket: 10.1.1.1 port: 49153 at time =
--
4.0023s Node 14 Current remaining energy = 296.721J
Cluster Head=0
Cluster Head=10
Cluster Head=19
--
Received one packet! Socket: 10.1.1.1 port: 49153 at time =
--
4.0023s Node 10 Current remaining energy = 296.721J
Cluster Head=0
Cluster Head=15
Cluster Head=19
--
Received one packet! Socket: 10.1.1.1 port: 49153 at time =
--
4.0023s Node 19 Current remaining energy = 296.721J
Cluster Head=0
Cluster Head=15
Cluster Head=20
--
Received one packet! Socket: 10.1.1.1 port: 49153 at time =
--
4.0023s Node 20 Current remaining energy = 296.721J
Cluster Head=0
Cluster Head=15
Cluster Head=17
--
Received one packet! Socket: 10.1.1.1 port: 49153 at time =
--
4.0023s Node 17 Current remaining energy = 296.721J
Cluster Head=0
Cluster Head=15
Cluster Head=18
python
Zoom: 2.174
Speed: 1.000
Time: 4.000000 s
Snapshot
Shell
Simulate (F3)
Advanced


Cluster Head=20
Received one packet! Socket: 10.1.1.1 port: 49153 at time =
13.0023s Node 8 Current remaining energy = 289.347J
Cluster Head=0
Cluster Head=14
Cluster Head=20
Received one packet! Socket: 10.1.1.1 port: 49153 at time =
13.0023s Node 14 Current remaining energy = 289.347J
Cluster Head=0
Cluster Head=15
Cluster Head=20
Received one packet! Socket: 10.1.1.1 port: 49153 at time =
13.0023s Node 20 Current remaining energy = 289.347J
Cluster Head=0
Cluster Head=15
Cluster Head=19
Received one packet! Socket: 10.1.1.1 port: 49153 at time =
13.0023s Node 19 Current remaining energy = 289.347J
Cluster Head=0
Cluster Head=15
Cluster Head=17
Received one packet! Socket: 10.1.1.1 port: 49153 at time =
13.0023s Node 17 Current remaining energy = 289.347J
Cluster Head=0
Cluster Head=15
Cluster Head=18
python
Zoom: 2.174
Speed: 1.000
Time: 13.200000 s
Snapshot
Shell
Simulate (F3)
Advanced


Routing Overhead
Routing Overhead
Packet Delivery Ratio
Number of Nodes
0
0.5
1
1.5
2
2.5
3
10
15
20
25
30
35
40
45
50