


Name of Student : Sunny Satish Halkatti		
Roll Number : 17		LAB Assignment Number: 5
Title of LAB Assignment :Program to simulate hybrid topology.		
DOP : 20-04-2023		DOS : 27-04-2023
CO Mapped : CO2	PO Mapped: PO1, PO2, PO3, PO5, PO7, PSO1	Signature : 

PRACTICAL 5

Aim: Write a program to simulate hybrid Topology.

Theory:

Understanding Hybrid Topology (Wireless Network Topology):

- A hybrid topology is a kind of network topology that is a combination of two or more network topologies, such as mesh topology, bus topology, and ring topology. Its usage and choice are dependent on its deployments and requirements like the performance of the desired network, and the number of computers, their location. The below figure is describing the structure of hybrid topology that contains more than one topology.
- However, a variety of technologies are needed for its physical implementation, and it offers a complex structure. Also, it includes an advantage as increasing flexibility; it can increase fault tolerance, and allows new basic topologies to be added or removed easily. The hybrid topology is more useful when you need to fulfill diversity in a Computer Network. In this topology, all network sections can include the configuration of different Network Topology. For instance, you can have a Hybrid network made by two different networks Star Backbone and the Ring Network. You can also use the Star Mesh Hybrid Topology in which if the main backbone fails, the entire network will destroy.

Why Do We Use the Hybrid Topology?

- Due to effective cost, there are different applications that use hybrid topology. As compared to other fundamental mechanisms, the mechanism of hybrid topology is efficient; it can also be deployed in different environments. Thus, it provides users the benefit to create, run and manage the organization.
- There are various sectors where the hybrid topology is widely used, such as many educational institutions, banking sector, automated industries, financial sector, research organizations, and multinational companies. For creating the structure of the new hybrid topology, there is a need to mix any two topologies like full mesh topology, extended star, partial star, point to point networks is used.
- The examples and applications of hybrid topology are increasing rapidly. It has a superpower set up and flexible option and declared as a smart option; hence, the people choose to deploy it in-home or office. A compact is provided for the small-scale industries by this topology, as well as to their subunits.
- Thus, it is good to use for multi-floor buildings and departments such as an office or home. This topology is placed to give its maximum efficiency on the basis of the requirements as it provides many benefits.

Advantages of Hybrid topology:

There are multiple advantages of Hybrid Topology; such are discussed below:

- **Reliable:** It is more reliable as it has better fault tolerance. If a node gets damaged between the network, it is possible in this network to single out the damaged node from

the rest of the network. Also, in this case, without impacting the processing of the network, the needed steps can be taken.

- **Effective:** This is the biggest advantage of hybrid topology. The weakness of the several topologies connected in this topology are ignored. And, there is a consideration only about the strengths of these different topologies. For case, the high tolerance capability is offered by star topology, and good data reliability is provided by ring topology. Therefore, in hybrid star-ring topology, these two-function work quite well.
- **Scalable:** Hybrid networks are the kind of network that is designed in a way, which led to making them capable of easy integration of additional concentration points or other new hardware components. Without disturbing existing architecture, it is very easy to extend the network size with the latest addition of new elements.

Disadvantages of Hybrid topology:

- **Complexity:** To manage the topology becomes challenging, as the different topologies are linked in the hybrid topology. It is a difficult job for designers and not easy to create this type of architecture. There is a need to be very efficient for the installation and configuration process.
- **Expensive:** Purchasing and maintaining the hybrid topology is much expensive while comparing with other topologies. The hubs are also required in this network topology that are used to connect two different networks, and they are also expensive. Furthermore, the hybrid topology may need advanced network devices, a lot of cables, and more as its architectures are usually larger in scale.
- One of the other disadvantages of hybrid topology; although it is able to detect faults easily, it needs a multistation access unit to bypass faulty devices.

Code:

```
#include "ns3/core-module.h"
#include "ns3/point-to-point-module.h"
#include "ns3/network-module.h"
#include "ns3/applications-module.h"
#include "ns3/mobility-module.h"
#include "ns3/csma-module.h"
#include "ns3/internet-module.h"
#include "ns3/yans-wifi-helper.h"
#include "ns3/ssid.h"
#include "ns3/netanim-module.h"
// Default Network Topology
// Wifi 10.1.3.0
// AP
// * * * *
// |||| 10.1.1.0
```

```
// n5 n6 n7 n0 ----- n1 n2 n3 n4
// point-to-point |||
// =====
// LAN 10.1.2.0
using namespace ns3;
NS_LOG_COMPONENT_DEFINE ("ThirdScriptExample");
int
main (int argc, char *argv[])
{
    bool verbose = true;
    uint32_t nCsmma = 3;
    uint32_t nWifi = 3;
    bool tracing = true;
    CommandLine cmd (__FILE__);
    cmd.AddValue ("nCsmma", "Number of \"extra\" CSMA nodes/devices", nCsmma);
    cmd.AddValue ("nWifi", "Number of wifi STA devices", nWifi);
    cmd.AddValue ("verbose", "Tell echo applications to log if true", verbose);
    cmd.AddValue ("tracing", "Enable pcap tracing", tracing);
    cmd.Parse (argc,argv);
    // The underlying restriction of 18 is due to the grid position
    // allocator's configuration; the grid layout will exceed the
    // bounding box if more than 18 nodes are provided.
    if (nWifi > 18)
    {
        std::cout << "nWifi should be 18 or less; otherwise grid layout exceeds the boundingbox" <<
        std::endl;
        return 1;
    }
    if (verbose)
    {
        LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
        LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);
    }
    NodeContainer p2pNodes;
    p2pNodes.Create (2);
    PointToPointHelper pointToPoint;
    pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
    pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));
    NetDeviceContainer p2pDevices;
    p2pDevices = pointToPoint.Install (p2pNodes);
```

```
NodeContainer csmaNodes;
csmaNodes.Add (p2pNodes.Get (1));
csmaNodes.Create (nCsma);
CsmaHelper csma;
csma.SetChannelAttribute ("DataRate", StringValue ("100Mbps"));
csma.SetChannelAttribute ("Delay", TimeValue (NanoSeconds (6560)));
NetDeviceContainer csmaDevices;

csmaDevices = csma.Install (csmaNodes);
NodeContainer wifiStaNodes;
wifiStaNodes.Create (nWifi);
NodeContainer wifiApNode = p2pNodes.Get (0);
YansWifiChannelHelper channel = YansWifiChannelHelper::Default ();
YansWifiPhyHelper phy = YansWifiPhyHelper::Default ();
phy.SetChannel (channel.Create ());
WifiHelper wifi;
wifi.SetRemoteStationManager ("ns3::AarfWifiManager");
WifiMacHelper mac;
Ssid ssid = Ssid ("ns-3-ssid");
mac.SetType ("ns3::StaWifiMac",
"Ssid", SsidValue (ssid),
"ActiveProbing", BooleanValue (false));
NetDeviceContainer staDevices;
staDevices = wifi.Install (phy, mac, wifiStaNodes);
mac.SetType ("ns3::ApWifiMac",
"Ssid", SsidValue (ssid));
NetDeviceContainer apDevices;
apDevices = wifi.Install (phy, mac, wifiApNode);
MobilityHelper mobility;
mobility.SetPositionAllocator ("ns3::GridPositionAllocator",
"MinX", DoubleValue (0.0),
"MinY", DoubleValue (0.0),
"DeltaX", DoubleValue (5.0),
"DeltaY", DoubleValue (10.0),
"GridWidth", UIntegerValue (3),
"LayoutType", StringValue ("RowFirst"));
mobility.SetMobilityModel ("ns3::RandomWalk2dMobilityModel", "Bounds", RectangleValue
(Rectangle (-50, 50, -50, 50)));
mobility.Install (wifiStaNodes);
mobility.SetMobilityModel ("ns3::ConstantPositionMobilityModel");
```

```
mobility.Install (wifiApNode);
InternetStackHelper stack;
stack.Install (csmaNodes);
stack.Install (wifiApNode);
stack.Install (wifiStaNodes);
Ipv4AddressHelper address;
address.SetBase ("10.1.1.0", "255.255.255.0");
Ipv4InterfaceContainer p2pInterfaces;
p2pInterfaces = address.Assign (p2pDevices);
address.SetBase ("10.1.2.0", "255.255.255.0");
Ipv4InterfaceContainer csmaInterfaces;
csmaInterfaces = address.Assign (csmaDevices);
address.SetBase ("10.1.3.0", "255.255.255.0");
address.Assign (staDevices);
address.Assign (apDevices);
UdpEchoServerHelper echoServer (9);
ApplicationContainer serverApps = echoServer.Install (csmaNodes.Get (nCsmas));
serverApps.Start (Seconds (1.0));
serverApps.Stop (Seconds (10.0));
UdpEchoClientHelper echoClient (csmaInterfaces.GetAddress (nCsmas), 9);
echoClient.SetAttribute ("MaxPackets", IntegerValue (1));
echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
echoClient.SetAttribute ("PacketSize", IntegerValue (1024));
ApplicationContainer clientApps =
echoClient.Install (wifiStaNodes.Get (nWifi - 1));
clientApps.Start (Seconds (2.0));

clientApps.Stop (Seconds (10.0));
Ipv4GlobalRoutingHelper::PopulateRoutingTables ();
AnimationInterface anim("hybrid_topology_urvi.xml");

Simulator::Stop (Seconds (10.0));
if (tracing == true)
{
pointToPoint.EnablePcapAll ("third");
phy.EnablePcap ("third", apDevices.Get (0));
csma.EnablePcap ("third", csmaDevices.Get (0), true);
}
Simulator::Run ();
Simulator::Destroy ();
```

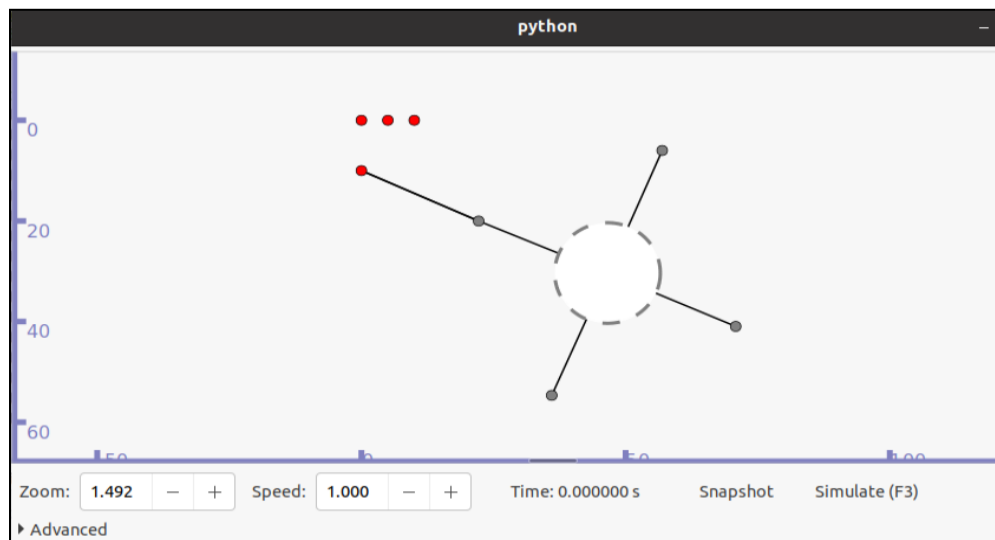
```
return 0;
}
```

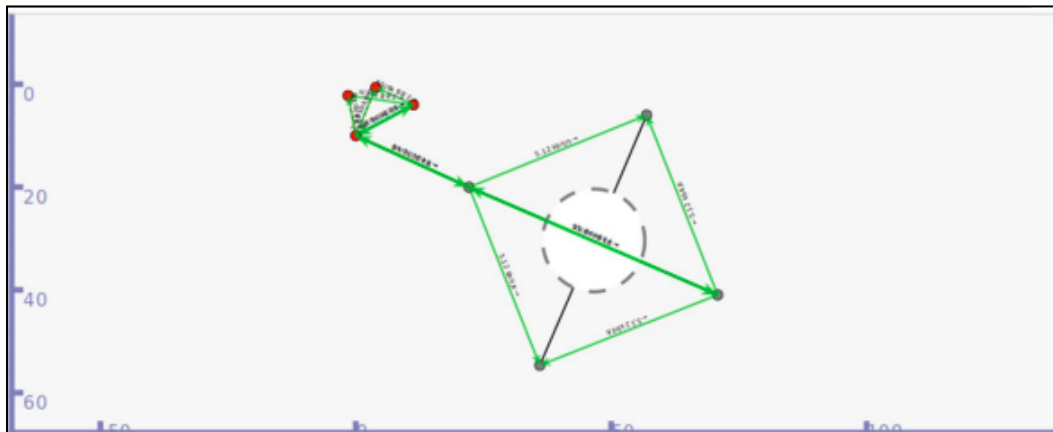
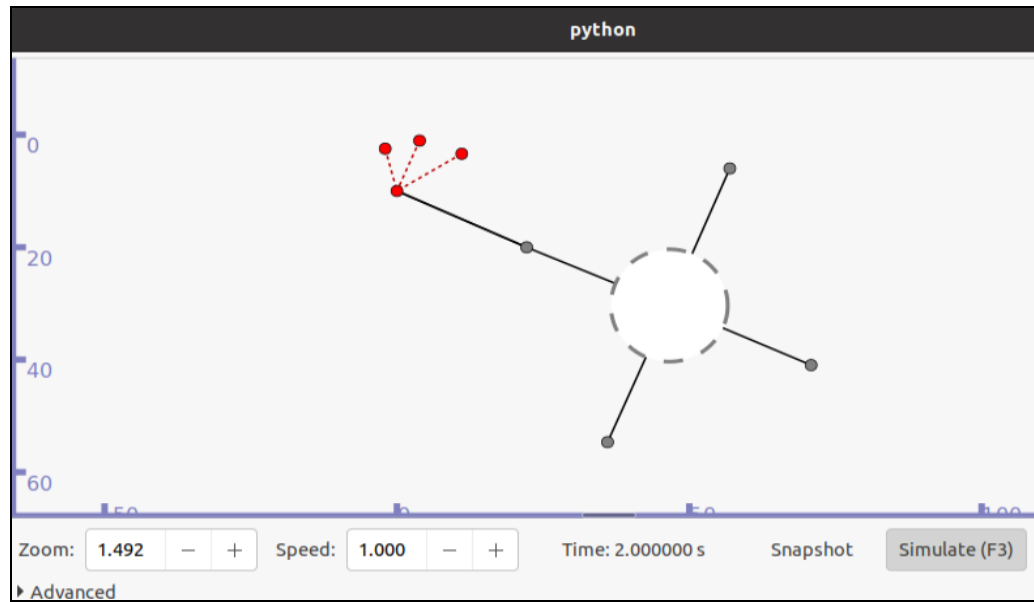
Output:**Compiling the file:**

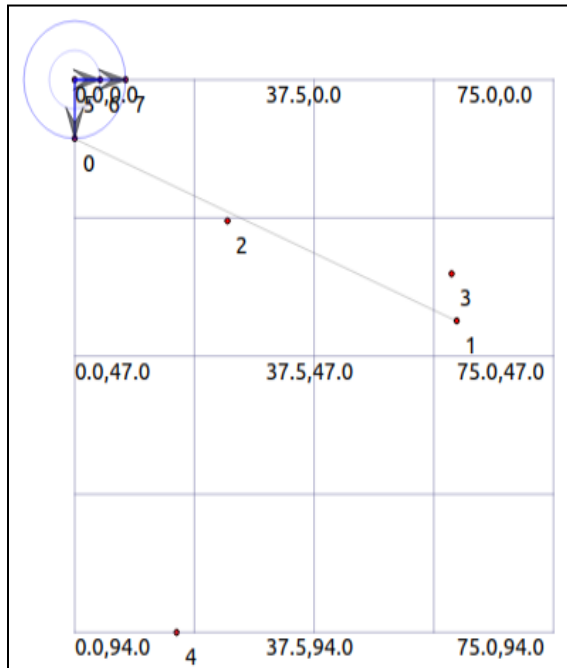
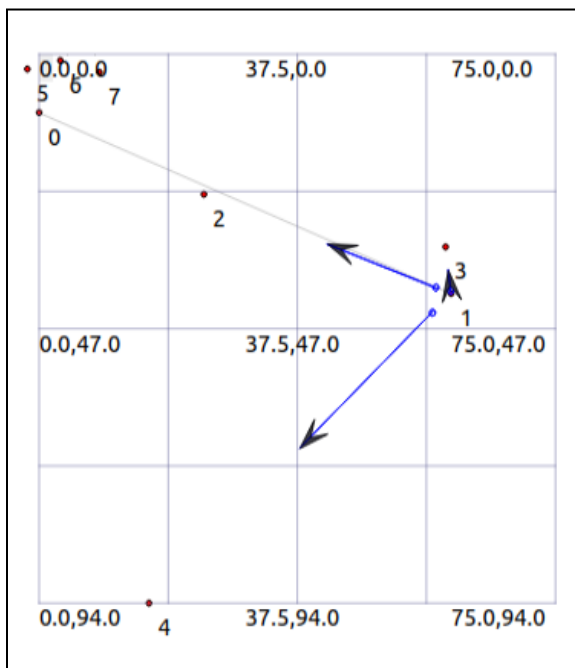
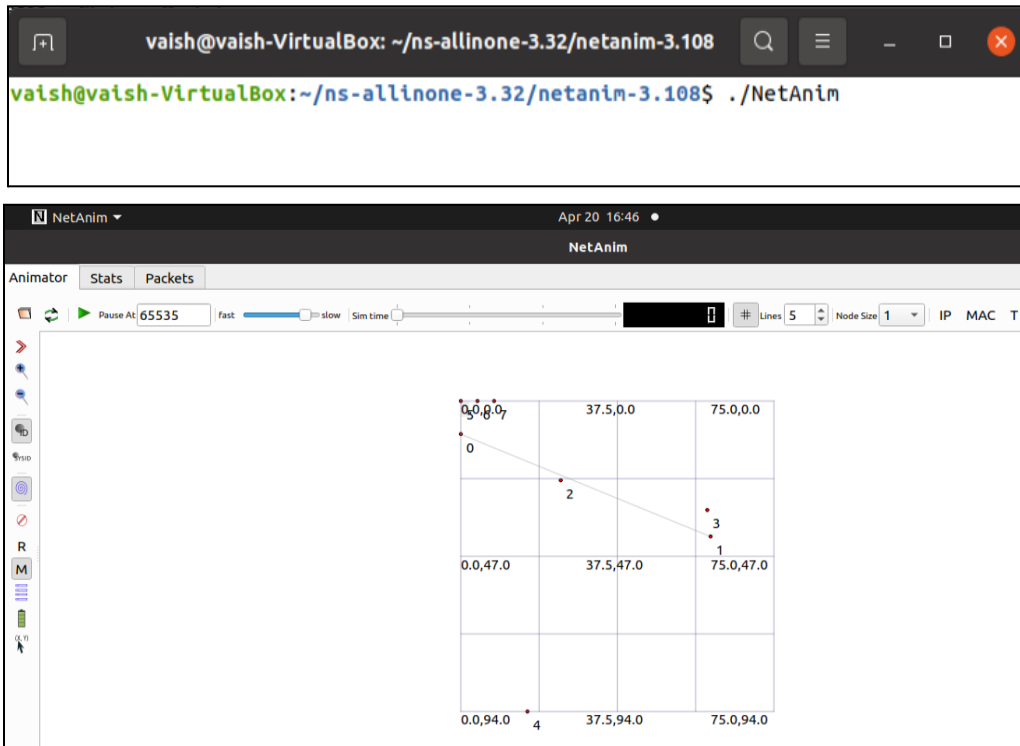
```
vaish@vaish-VirtualBox: ~/ns-allinone-3.32/ns-3.32
vaish@vaish-VirtualBox:~/ns-allinone-3.32/ns-3.32$ ./waf --run scratch/hybrid_urvi.cc
Waf: Entering directory `/home/vaish/ns-allinone-3.32/ns-3.32/build'
[1997/2068] Compiling scratch/hybrid_urvi.cc
[2028/2068] Linking build/scratch/hybrid_urvi
Waf: Leaving directory `/home/vaish/ns-allinone-3.32/ns-3.32/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (8.130s)
AnimationInterface WARNING:Node:1 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:2 Does not have a mobility model. Use SetConstantPosition if it is stationary
```

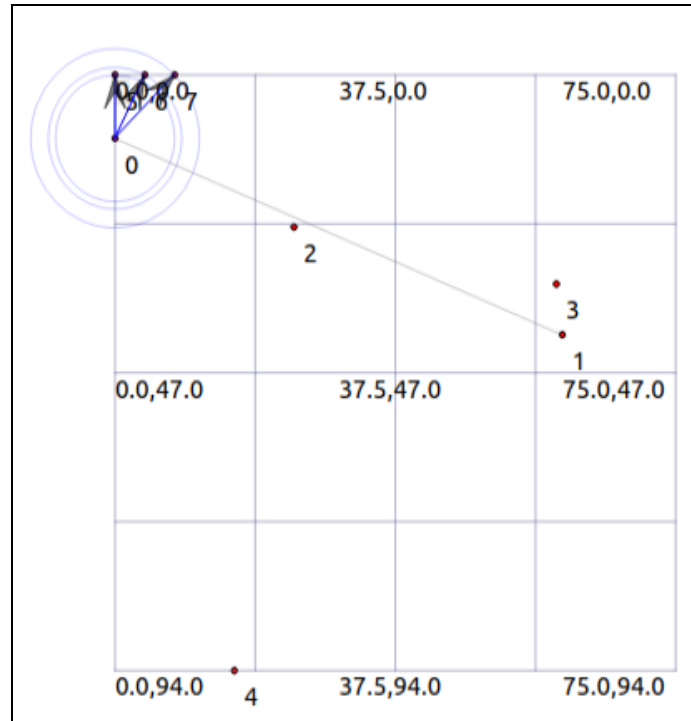
Running the file using Python Visualizer:

```
vaish@vaish-VirtualBox: ~/ns-allinone-3.32/ns-3.32
vaish@vaish-VirtualBox:~/ns-allinone-3.32/ns-3.32$ ./waf --run scratch/hybrid_urvi.cc --vis
Waf: Entering directory `/home/vaish/ns-allinone-3.32/ns-3.32/build'
Waf: Leaving directory `/home/vaish/ns-allinone-3.32/ns-3.32/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (0.801s)
AnimationInterface WARNING:Node:1 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:2 Does not have a mobility model. Use SetConstantPosition if it is stationary
```





Launching NetAnim:

**Conclusion:**

We have gained knowledge to simulate hybrid topology (Wireless Network Topology) using NetAnim and PyViz.