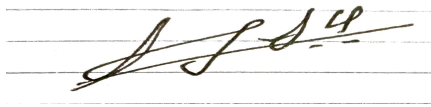


Name of Student : Sunny Satish Halkatti		
Roll Number : 17		LAB Assignment Number: 7
Title of LAB Assignment : Program to simulate DHCP server and n clients.		
DOP : 04-05-2023		DOS : 11-05-2023
CO Mapped : CO2	PO Mapped: PO1, PO2, PO3, PO5, PO7, PSO1	Signature : 

NWL Practical - 7

Aim: Program to simulate DHCP server and n clients.

Theory:

DHCP stands for Dynamic Host Configuration Protocol. It is the critical feature on which the users of an enterprise network communicate. DHCP helps enterprises to smoothly manage the allocation of IP addresses to the end-user clients' devices such as desktops, laptops, cellphones, etc. is an application layer protocol that is used to provide:

Subnet Mask (Option 1 - e.g., 255.255.255.0)
Router Address (Option 3 - e.g., 192.168.1.1)
DNS Address (Option 6 - e.g., 8.8.8.8)
Vendor Class Identifier (Option 43 - e.g.,
'unifi' = 192.168.1.9 ##where unifi = controller)

Components of DHCP

The main components of DHCP include:

DHCP Server: DHCP Server is basically a server that holds IP Addresses and other information related to configuration.

DHCP Client: It is basically a device that receives configuration information from the server. It can be a mobile, laptop, computer, or any other electronic device that requires a connection.

DHCP Relay: DHCP relays basically work as a communication channel between DHCP Client and Server.

IP Address Pool: It is the pool or container of IP Addresses possessed by the **DHCP Server**. It has a range of addresses that can be allocated to devices.

Subnets: Subnets are smaller portions of the IP network partitioned to keep networks under control.

Lease: It is simply the time that how long the information received from the server is valid, in case of expiration of the lease, the tenant must have to re-assign the lease.

DNS Servers: DHCP servers can also provide DNS (Domain Name System) server information to DHCP clients, allowing them to resolve domain names to IP addresses.

Default Gateway: DHCP servers can also provide information about the default gateway, which is the device that packets are sent to when the destination is outside the local network.

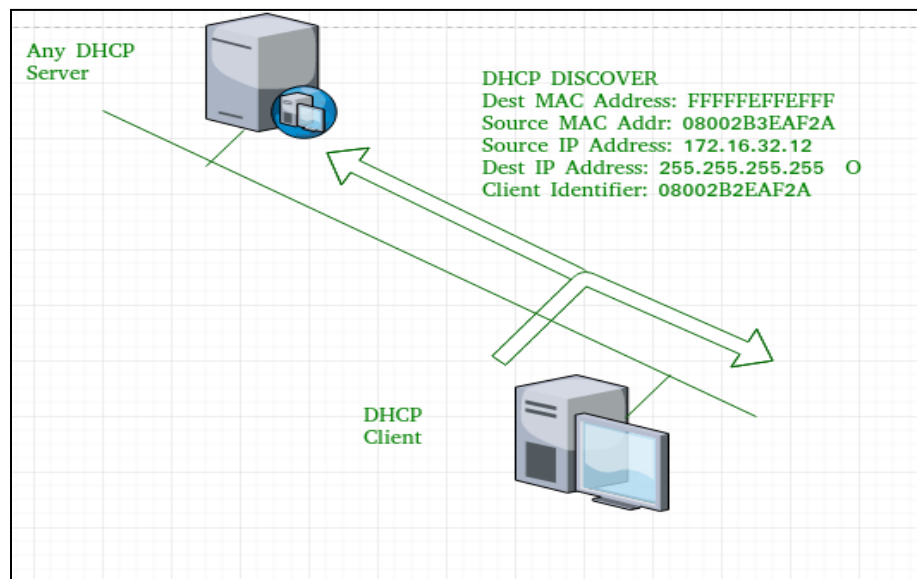
Options: DHCP servers can provide additional configuration options to clients, such as the subnet mask, domain name, and time server information.

Renewal: DHCP clients can request to renew their lease before it expires to ensure that they continue to have a valid IP address and configuration information.

Failover: DHCP servers can be configured for failover, where two servers work together to provide redundancy and ensure that clients can always obtain an IP address and configuration information, even if one server goes down.

Dynamic Updates: DHCP servers can also be configured to dynamically update DNS records with the IP address of DHCP clients, allowing for easier management of network resources.

Audit Logging: DHCP servers can keep audit logs of all DHCP transactions, providing administrators with visibility into which devices are using which IP addresses and when leases are being assigned or renewed.



Code:

```
#include "ns3/core-module.h"
#include "ns3/internet-apps-module.h"
#include "ns3/csma-module.h"
#include "ns3/internet-module.h"
#include "ns3/point-to-point-module.h"
```

```
#include "ns3/applications-module.h"
#include "ns3/netanim-module.h"
using namespace ns3;

NS_LOG_COMPONENT_DEFINE ("DhcpExample");
int main (int argc, char *argv[])
{
    CommandLine cmd (__FILE__);
    bool verbose = true;
    bool tracing = true;
    cmd.AddValue ("verbose", "turn on the logs", verbose);
    cmd.AddValue ("tracing", "turn on the tracing", tracing);

    cmd.Parse (argc, argv);
    if (verbose)
    {
        LogComponentEnable ("DhcpServer", LOG_LEVEL_ALL);
        LogComponentEnable ("DhcpClient", LOG_LEVEL_ALL);
        LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);
        LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
    }
    Time stopTime = Seconds (20);
    NS_LOG_INFO ("Create nodes.");
    NodeContainer nodes;
    NodeContainer router;
    nodes.Create (3);
    router.Create (2);
    NodeContainer net (nodes, router);
    NS_LOG_INFO ("Create channels.");
    CsmHelper csma;
    csma.SetChannelAttribute ("DataRate", StringValue ("5Mbps"));
    csma.SetChannelAttribute ("Delay", StringValue ("2ms"));
    csma.SetDeviceAttribute ("Mtu", UIntegerValue (1500));
    NetDeviceContainer devNet = csma.Install (net);
    NodeContainer p2pNodes;
    p2pNodes.Add (net.Get (4));
    p2pNodes.Create (1);
```

```
PointToPointHelper pointToPoint;
pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));
NetDeviceContainer p2pDevices;
p2pDevices = pointToPoint.Install (p2pNodes);
InternetStackHelper tcpip;
tcpip.Install (nodes);
tcpip.Install (router);
tcpip.Install (p2pNodes.Get (1));

Ipv4AddressHelper address;
address.SetBase ("172.30.1.0", "255.255.255.0");
Ipv4InterfaceContainer p2pInterfaces;
p2pInterfaces = address.Assign (p2pDevices);
// manually add a routing entry because we don't want to add a dynamic routing
Ipv4StaticRoutingHelper ipv4RoutingHelper;
Ptr<Ipv4> ipv4Ptr = p2pNodes.Get (1)->GetObject<Ipv4> ();
Ptr<Ipv4StaticRouting> staticRoutingA = ipv4RoutingHelper.GetStaticRouting
(ipv4Ptr);
staticRoutingA->AddNetworkRouteTo (Ipv4Address ("172.30.0.0"), Ipv4Mask ("/24"),
Ipv4Address ("172.30.1.1"), 1);

NS_LOG_INFO ("Setup the IP addresses and create DHCP applications.");
DhcpHelper dhcpHelper;
// The router must have a fixed IP.
Ipv4InterfaceContainer fixedNodes = dhcpHelper.InstallFixedAddress (devNet.Get (4),
Ipv4Address ("172.30.0.17"), Ipv4Mask ("/24"));
// Not really necessary, IP forwarding is enabled by default in IPv4.
fixedNodes.Get (0).first->SetAttribute ("IpForward", BooleanValue (true));

// DHCP server
ApplicationContainer dhcpServerApp = dhcpHelper.InstallDhcpServer (devNet.Get
(3),
Ipv4Address ("172.30.0.12"), Ipv4Address ("172.30.0.0"), Ipv4Mask ("/24"),
Ipv4Address ("172.30.0.10"), Ipv4Address ("172.30.0.15"), Ipv4Address
("172.30.0.17"));
// This is just to show how it can be done.
```

```
DynamicCast<DhcpServer> (dhcpServerApp.Get (0))->AddStaticDhcpEntry
(devNet.Get (2)->GetAddress (), Ipv4Address ("172.30.0.14"));
dhcpServerApp.Start (Seconds (0.0));
dhcpServerApp.Stop (stopTime);
// DHCP clients
NetDeviceContainer dhcpClientNetDevs;
dhcpClientNetDevs.Add (devNet.Get (0));
dhcpClientNetDevs.Add (devNet.Get (1));
dhcpClientNetDevs.Add (devNet.Get (2));
ApplicationContainer dhcpClients = dhcpHelper.InstallDhcpClient (dhcpClientNetDevs);
dhcpClients.Start (Seconds (5.0));
dhcpClients.Stop (stopTime);
UdpEchoServerHelper echoServer (9);

ApplicationContainer serverApps = echoServer.Install (p2pNodes.Get (1));
serverApps.Start (Seconds (0.0));
serverApps.Stop (stopTime);

UdpEchoClientHelper echoClient (p2pInterfaces.GetAddress (1), 9);
echoClient.SetAttribute ("MaxPackets", UIntegerValue (100));
echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
echoClient.SetAttribute ("PacketSize", UIntegerValue (1024));

ApplicationContainer clientApps = echoClient.Install (nodes.Get (1));
clientApps.Start (Seconds (10.0));
clientApps.Stop (stopTime);
AnimationInterface anim("DHCP_1122.xml");

Simulator::Stop (stopTime + Seconds (10.0));

if (tracing)
{
csma.EnablePcapAll ("dhcp-csma");
pointToPoint.EnablePcapAll ("dhcp-p2p");
}

NS_LOG_INFO ("Run Simulation.");
```

```

Simulator::Run ();
Simulator::Destroy ();
NS_LOG_INFO ("Done.");
}

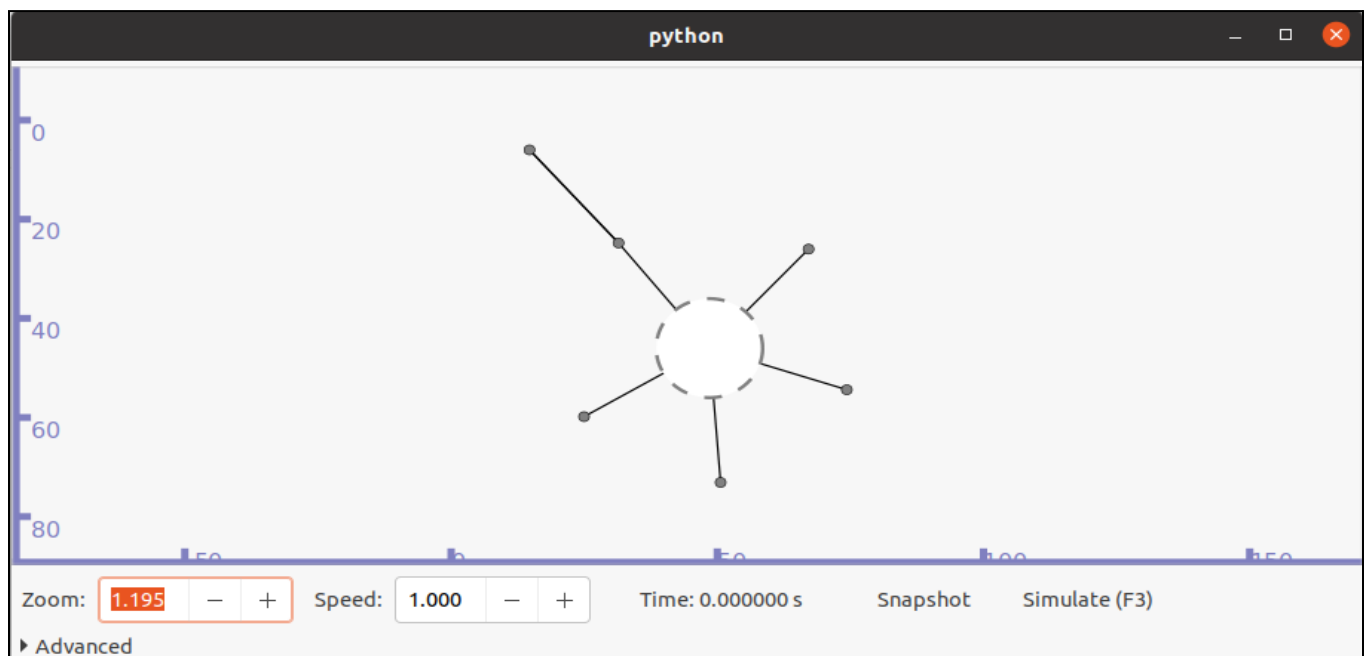
```

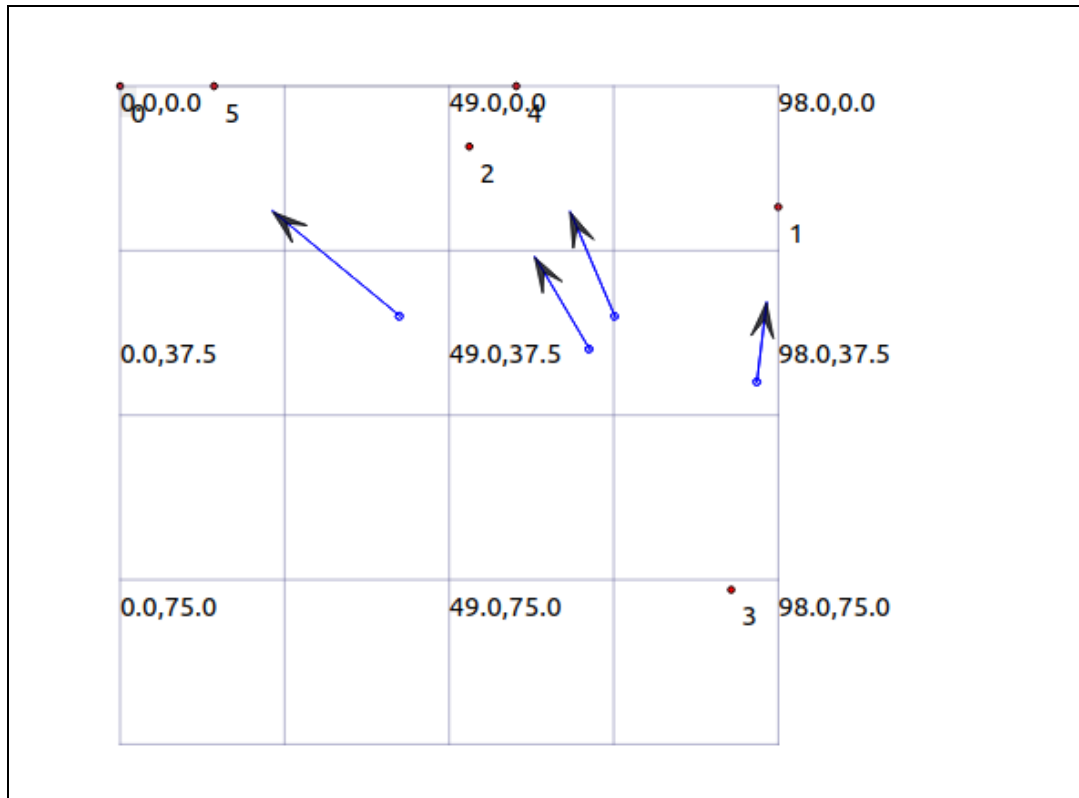
Output:

```

vaish@vaish-VirtualBox:~/Workspace/ns-allinone-3.32/ns-3.32$ ./waf --run prac7 --vis
Waf: Entering directory `~/home/vaish/Workspace/ns-allinone-3.32/ns-3.32/build'
Waf: Leaving directory `~/home/vaish/Workspace/ns-allinone-3.32/ns-3.32/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (0.728s)
AnimationInterface WARNING:Node:0 Does not have a mobility model. Use SetConstantPosition if it is stationary
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
AnimationInterface WARNING:Node:3 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:4 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:5 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:0 Does not have a mobility model. Use SetConstantPosition if it is stationary
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
AnimationInterface WARNING:Node:3 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:4 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:5 Does not have a mobility model. Use SetConstantPosition if it is stationary
Could not load plugin 'show_last_packets.py': No module named 'kiwi'
Could not load icon applets-screenshooter due to missing gnomedesktop Python module
scanning topology: 6 nodes...
scanning topology: calling graphviz layout
scanning topology: all done.
AnimationInterface WARNING:Node:0 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:0 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:0 Does not have a mobility model. Use SetConstantPosition if it is stationary

```



**Conclusions:**

We have successfully implemented a program that simulates DHCP Client Server communication.