DATA COMMUNICATION AND NETWORKING – II

Project USING NS3

Under the guidance
of
Prof. Tao Han

ECGR-5187

MSEE Department

Jeevitha Mahankali (800966168)
Mail Id: jmahanka@uncc.edu

Raghava Adarsh Mandarapu (800937296)
Mail Id: rmandara@uncc.edu

Ruchita Jain (800966215)
Mail Id: rjain10@uncc.edu

INDEX

OBJECTIVES

• Network should have 3 LANs connecting to the Internet
• It should have at least 15 nodes in each LAN
• Network should have at least 3-application traffics among nodes in each LAN (intra-LAN connections)
• Network should have at least 3-application traffics among nodes in all LANs (inter-LAN connections)
• It should have both wired and wireless communications
• Having specified connections (between pairs of specified nodes) and general background connections
• It should have specified IP addresses for every nodes
• Not limiting to TCP networks
•Link failure should be applied

Network Architecture

Our network consists of three LANs. Three Routers were used to inter connect these LANs. We implemented the star topology in LAN1. It consists of a central hub with 15 spokes connected to it. In LAN2 CSMA Bus topology with 15 nodes was configured and it is connected to the router 1 with the help of a server. In LAN 3 Wi-Fi connection was established by taking an Access Point (AP) which distributes the connection to the 15 station nodes. We used 8 subnets for building this entire topology. The topology that we designed as in the picture depicted below.

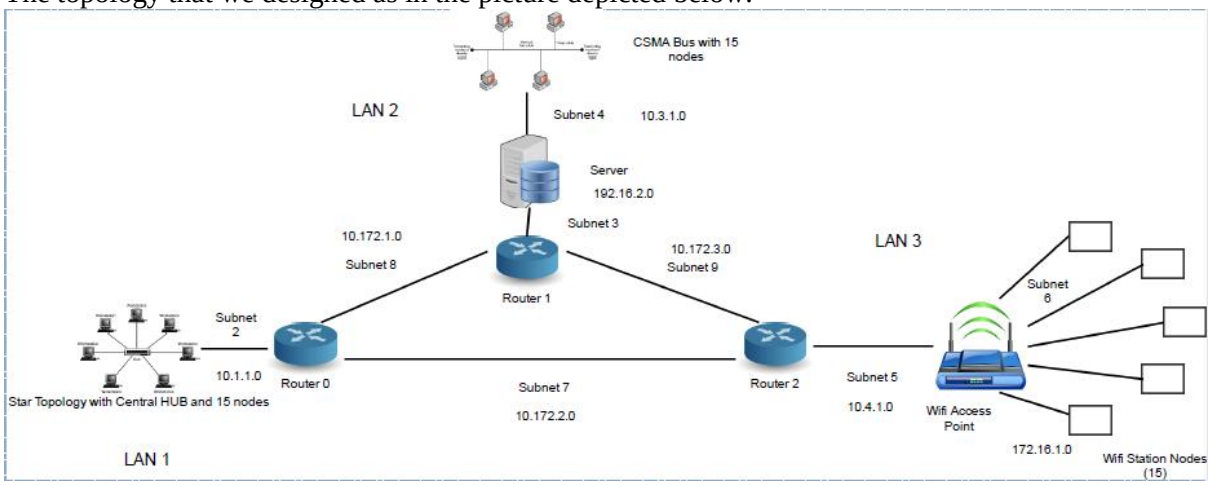


Figure 1

The Netanim simulation result as below

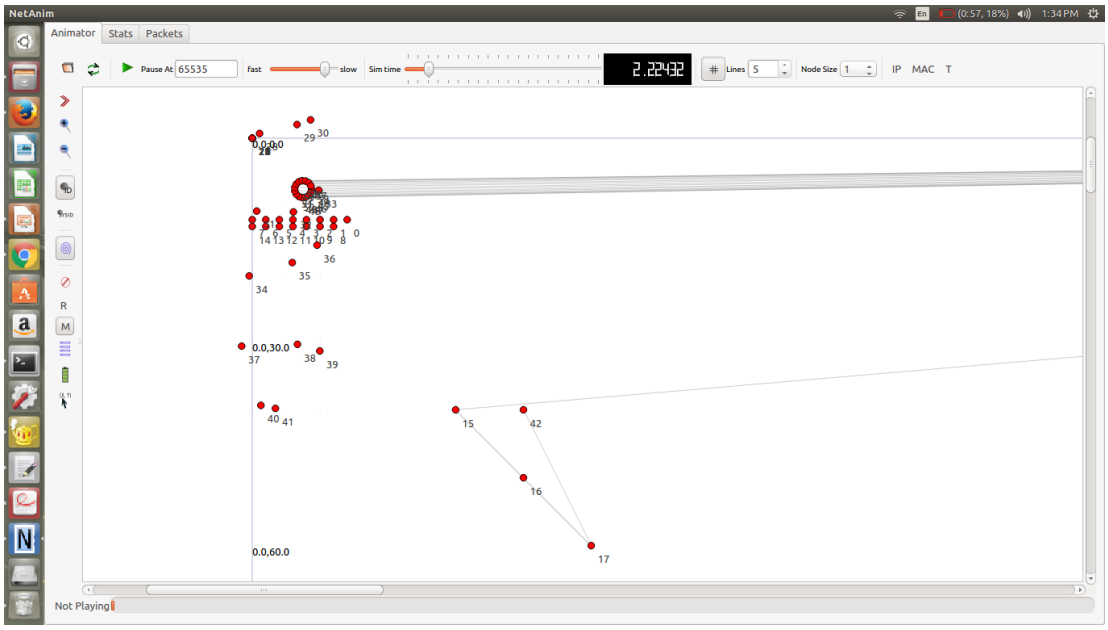


Figure 2

Code Snippets

Applications

1. UDP Echo Client-Server

Here we implemented two applications, one is; the spoke 1 of star topology (LAN1) acts as the server and node 6 of the CSMA topology (LAN2) as the client while the other is; the spoke 5 of the star topology acts as the server (LAN1) and Wi-Fi station node 8 acts the client(LAN3).

```
UdpEchoServerHelper echoServer (9);
ApplicationContainer serverApps = echoServer.Install (star.GetSpokeNode(1));
serverApps.Start (Seconds (5.0));
serverApps.Stop (Seconds (10.0));
UdpEchoClientHelper echoClient (star.GetSpokeIpv4Address(1), 9);
echoClient.SetAttribute ("MaxPackets", UintegerValue (1));
echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
echoClient.SetAttribute ("PacketSize", UintegerValue (1024));
ApplicationContainer clientApps = echoClient.Install (csmanodes1.Get(6));
clientApps.Start (Seconds (6.0));
clientApps.Stop (Seconds (10.0));
```

2. ON/OFF Application

In ON/OFF application the sender sends TCP packets to the destination. It sends TCP packets from spoke 7 to spoke 3 and also from spoke 10 to spoke 5 of the star network in LAN1.

```
uint16_t port = 50000;
Address LocalAddress (InetSocketAddress (Ipv4Address::GetAny (), port));
PacketSinkHelper packetSinkHelper1 ("ns3::TcpSocketFactory", LocalAddress);
ApplicationContainer App = packetSinkHelper1.Install (star.GetHub());
App.Start (Seconds (2.0));
App.Stop (Seconds (4.0));
// Create OnOff applications to send TCP to the node 3, from spoke node 7.
OnOffHelper onOffHelper ("ns3::TcpSocketFactory", Address ());
onOffHelper.SetAttribute("OnTime",StringValue("ns3::ConstantRandomVariable[Constant=1]"));
onOffHelper.SetAttribute("OffTime",StringValue("ns3::ConstantRandomVariable[Constant=0]");
ApplicationContainer spokeApps;
AddressValue remoteAddress1 (InetSocketAddress (star.GetHubIpv4Address (7), port));
onOffHelper.SetAttribute ("Remote", remoteAddress1);
spokeApps.Add (onOffHelper.Install (star.GetSpokeNode(7)));
spokeApps.Start (Seconds (2.0));
spokeApps.Stop (Seconds (4.0));
```

3. Broadcast

The broadcast to entire network is done from the Node 0 of CSMA bus. The code below makes the broadcast possible.

```
uint16_t port2 = 20;
NS_LOG_INFO ("Create Applications.");
OnOffHelperonoff3("ns3::UdpSocketFactory",Address(InetSocketAddress(Ipv4Address("255.255.2
55.255"), port2)));
onoff3.SetConstantRate (DataRate ("500kb/s"));
ApplicationContainer app3 = onoff3.Install (csmanodes1.Get(0));
app3.Start (Seconds (4.0));
app3.Stop (Seconds (6.0));
// Creating optional sink packet
PacketSinkHelper       sink       ("ns3::UdpSocketFactory",Address       (InetSocketAddress
(Ipv4Address::GetAny (), port2)));
 app3=sink.Install (csmanodes1.Get(0));
 for(int i=1; i<15; i++)
      {
          app3.Add(sink.Install (subnet4.Get(i)));
      }
 app3.Start (Seconds (4.0));
 app3.Stop (Seconds (6.0));
```

SIMULATION RESULTS

Applications

- UDP Echo-Client Server

```
shanthan@shanthan-ThinkPad-T430:~/Desktop/ns-allinone-3.26/ns-3.26$ ./waf --run scratch/jeevitha
Waf: Entering directory `/home/shanthan/Desktop/ns-allinone-3.26/ns-3.26/build'
[ 931/2663] Compiling scratch/jeevitha.cc
[2652/2663] Linking build/scratch/jeevitha
Waf: Leaving directory `/home/shanthan/Desktop/ns-allinone-3.26/ns-3.26/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (5.079s)
At time 3s client sent 1024 bytes to 192.168.6.2 port 22
At time 3.01991s server received 1024 bytes from 172.16.1.9 port 49153
At time 3.01991s server sent 1024 bytes to 172.16.1.9 port 49153
At time 3.03761s client received 1024 bytes from 192.168.6.2 port 22
At time 6s client sent 1024 bytes to 192.168.2.2 port 9
At time 6.02575s server received 1024 bytes from 10.3.1.6 port 49153
At time 6.02575s server sent 1024 bytes to 10.3.1.6 port 49153
At time 6.04151s client received 1024 bytes from 192.168.2.2 port 9
shanthan@shanthan-ThinkPad-T430:~/Desktop/ns-allinone-3.26/ns-3.26$
```

Figure 3

- ON and OFF Application

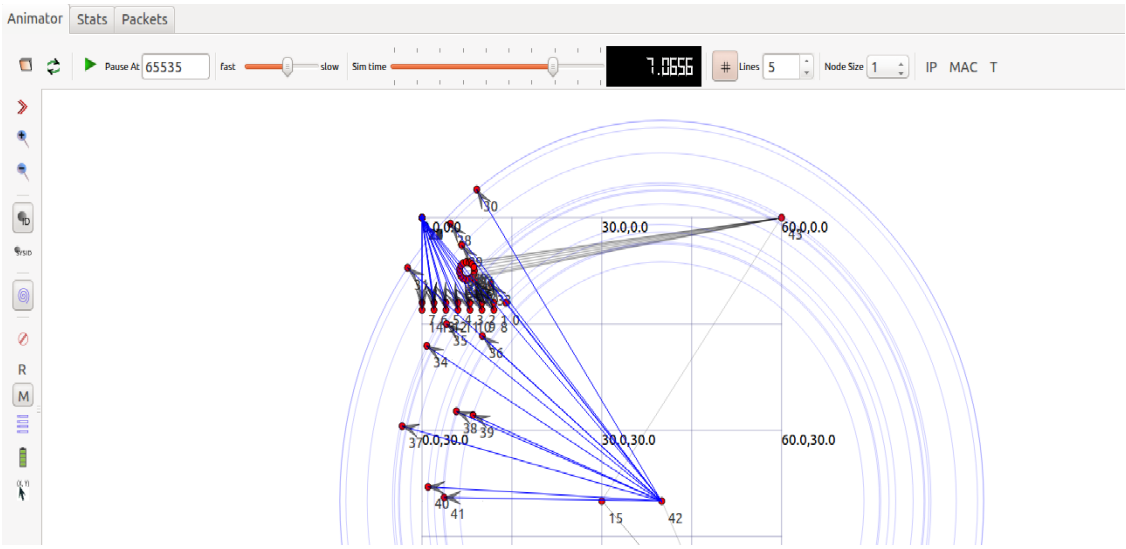


Figure 4

Network Analysis

As a part of network analysis we have tested the link failure and calculated the throughput.

- **Link Failure**

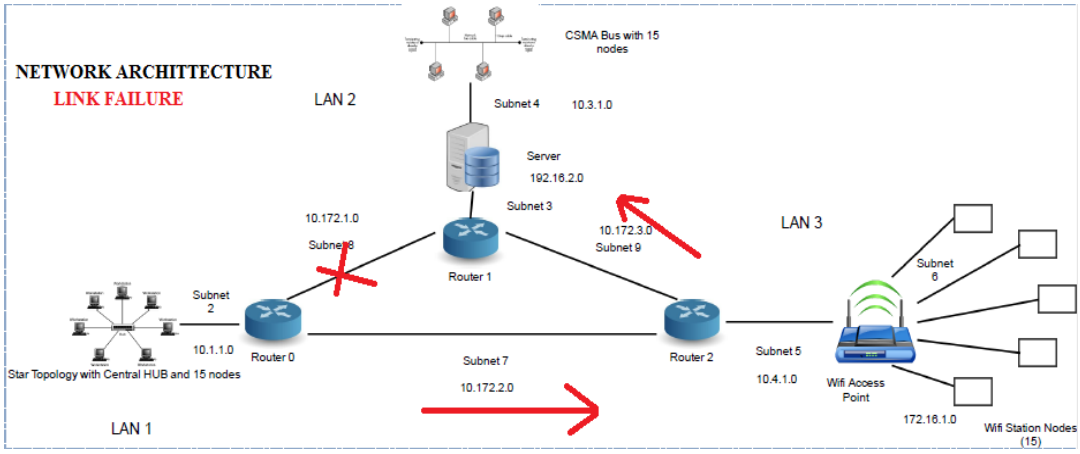


Figure 5

Here we disconnected the link between the router 0 and router 1 (i.e. subnet 8) and we observed that the network found an alternate path via router 2. As proof of this we can see the change in the gateway from the screenshot

- **Throughput**

Throughput is the amount of data moved successfully from one place to another. Here we are calculating the throughput of the data transferred between LAN1 and LAN 2 via subnet 8. We have also calculated the throughput from LAN1 to LAN2 via subnet 7 when the link is broken between the router 1 and router 0.

Throughput before Link Failure:

```
shanthan@shanthan-ThinkPad-T430:~/Desktop/ns-allinone-3.26/ns-3.26$ ./waf --run scratch/jeevitha
Waf: Entering directory `/home/shanthan/Desktop/ns-allinone-3.26/ns-3.26/build'
[2268/2663] Compiling scratch/jeevitha.cc
[2652/2663] Linking build/scratch/jeevitha
Waf: Leaving directory `/home/shanthan/Desktop/ns-allinone-3.26/ns-3.26/build'
Build commands will be stored in build/compile_commands.json
'build' finished successfully (5.969s)
At time 3s client sent 1024 bytes to 192.168.6.2 port 22
At time 3.01991s server received 1024 bytes from 172.16.1.9 port 49153
At time 3.01991s server sent 1024 bytes to 172.16.1.9 port 49153
At time 3.03761s client received 1024 bytes from 192.168.6.2 port 22
At time 6s client sent 1024 bytes to 192.168.2.2 port 9
At time 6.02207s server received 1024 bytes from 10.3.1.6 port 49153
At time 6.02207s server sent 1024 bytes to 10.3.1.6 port 49153
At time 6.03414s client received 1024 bytes from 192.168.2.2 port 9
 (172.16.1.9 -> 192.168.6.2)
  Tx Bytes:   1052
  Rx Bytes:   1052
  THROUGHPUT=: 0.403022 Mbps
 (10.3.1.6 -> 192.168.2.2)
  Tx Bytes:   1052
  Rx Bytes:   1052
  THROUGHPUT=: 0.363705 Mbps
The ip of Router : 10.172.1.1
shanthan@shanthan-ThinkPad-T430:~/Desktop/ns-allinone-3.26/ns-3.26$
```

Figure 6

Throughput after Link Failure:

Figure 7

From the above screenshots we can conclude that the throughput remains almost the same before and after the link failure. We can also see the change in the gateway when the link is broken.

Routing Table:

Before Link Failure: The Red box shows the destination addresses before link failure.

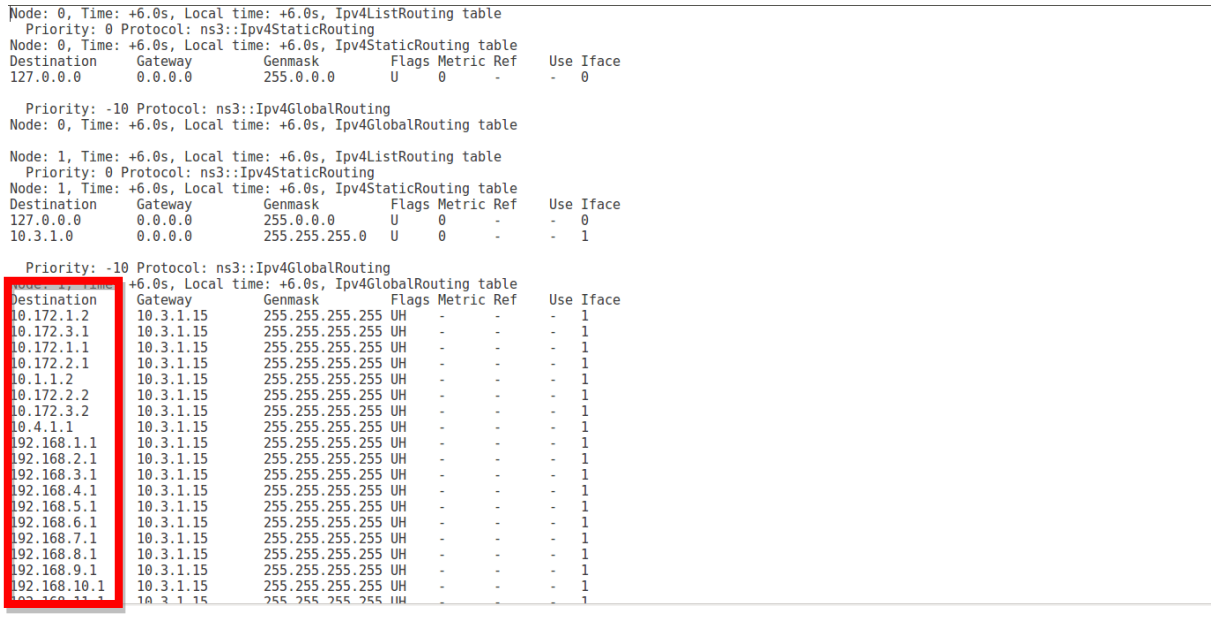
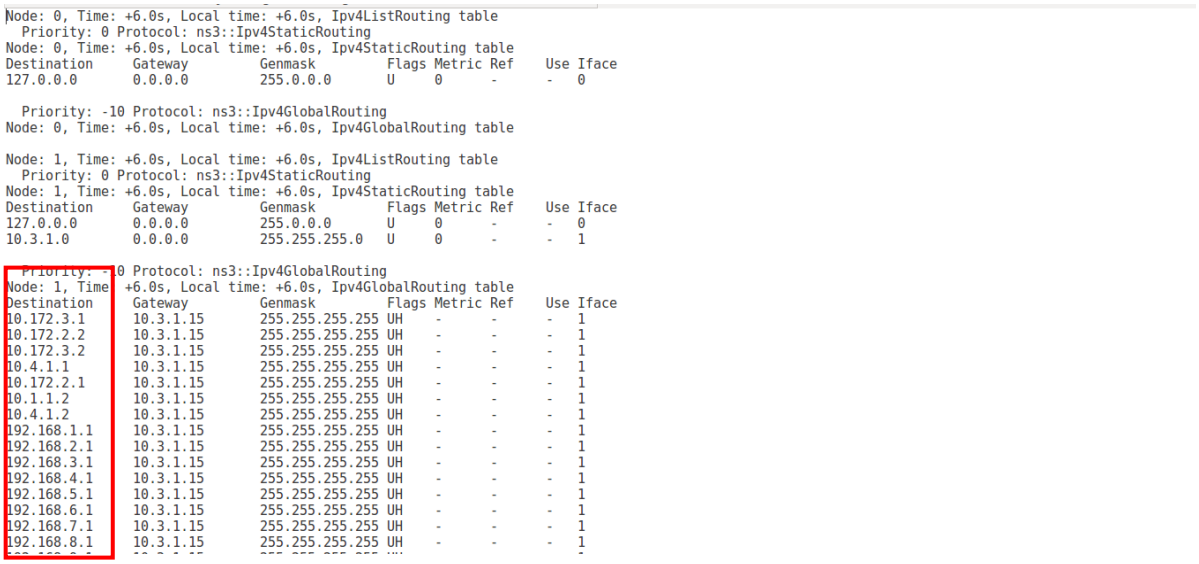


Figure 8

After Link Failure: The Red box shows the change in the IP address after link failure



```
Node: 0, Time: +6.0s, Local time: +6.0s, Ipv4ListRouting table
  Priority: 0 Protocol: ns3::Ipv4StaticRouting
Node: 0, Time: +6.0s, Local time: +6.0s, Ipv4StaticRouting table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
127.0.0.0       0.0.0.0         255.0.0.0       U     0      -      -   0

  Priority: -10 Protocol: ns3::Ipv4GlobalRouting
Node: 0, Time: +6.0s, Local time: +6.0s, Ipv4GlobalRouting table

Node: 1, Time: +6.0s, Local time: +6.0s, Ipv4ListRouting table
  Priority: 0 Protocol: ns3::Ipv4StaticRouting
Node: 1, Time: +6.0s, Local time: +6.0s, Ipv4StaticRouting table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
127.0.0.0       0.0.0.0         255.0.0.0       U     0      -      -   0
10.3.1.0        0.0.0.0         255.255.255.0   U     0      -      -   1

  Priority: -10 Protocol: ns3::Ipv4GlobalRouting
Node: 1, Time: +6.0s, Local time: +6.0s, Ipv4GlobalRouting table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.172.3.1      10.3.1.15       255.255.255.255 UH    -      -      -   1
10.172.2.2      10.3.1.15       255.255.255.255 UH    -      -      -   1
10.172.3.2      10.3.1.15       255.255.255.255 UH    -      -      -   1
10.4.1.1        10.3.1.15       255.255.255.255 UH    -      -      -   1
10.172.2.1      10.3.1.15       255.255.255.255 UH    -      -      -   1
10.1.1.2        10.3.1.15       255.255.255.255 UH    -      -      -   1
10.4.1.2        10.3.1.15       255.255.255.255 UH    -      -      -   1
192.168.1.1     10.3.1.15       255.255.255.255 UH    -      -      -   1
192.168.2.1     10.3.1.15       255.255.255.255 UH    -      -      -   1
192.168.3.1     10.3.1.15       255.255.255.255 UH    -      -      -   1
192.168.4.1     10.3.1.15       255.255.255.255 UH    -      -      -   1
192.168.5.1     10.3.1.15       255.255.255.255 UH    -      -      -   1
192.168.6.1     10.3.1.15       255.255.255.255 UH    -      -      -   1
192.168.7.1     10.3.1.15       255.255.255.255 UH    -      -      -   1
192.168.8.1     10.3.1.15       255.255.255.255 UH    -      -      -   1
```

Figure 9

CONCLUSION

We learnt the use of NS3 by building a personal network. Using three LAN's with different topology and running applications between the LAN. We used one star topology, one csma and one wifi topologies. Through NetAnim, we could see the on-off application, udp echo client server and broadcast application. Analysis of the network was implemented to calculate the throughput of the network and we also observe the link failure conditions before and after removing one of the links between the routers.