

PROJECT REPORT

*NETWORK TOPOLOGY USING NS3*

DATA COMMUNICATIOS AND NETWORKING 2 | 12/08/2017

Venkata Shiva Krishna Reddy Avula

800963515

vavula1@uncc.edu

# TABLE OF CONTENTS

1. Introduction ………………………………………………………… Page 3
2. Topology ………………………………………………………… Page 4
3. Applications Implemented ………………………………………………………… Page 5
4. Routing Tables ………………………………………………………… Page 6
5. Link Failure ………………………………………………………… Page 7
6. Wire Shark Analysis ……………………………………………………………Page 8
7. Conclusion ………………………………………………………… Page 9
8. References ………………………………………………………… Page 9

# INTRODUCTION

Ns-3 is a discrete-event computer network simulator that is built using C++ and Python with scripting capability. The ns-3 simulator features an integrated attribute-based system to manage default and per-instance values for simulation parameters. All of the configurable default values for parameters are managed by this system, integrated with command-line argument processing, Doxygen documentation, and an XML-based and optional GTK-based configuration subsystem.

# Container and Helper Classes

These classes are essential to simplifying the process of creating simple networks.

* *Container classes*: As the name suggests, the container classes actually contain something. The NodeContainer, for instance, contains in a vector, pointers to network nodes.
* NodeContainer
* NetDeviceContainer
* Ipv4InterfaceContainer
* *Helper classes*: Helper classes on the other hand help to simplify creation of network topologies and also set attributes for components. For instance, PointToPointHelper would help us to create and configure point-to-point networks. Similarly, CsmaHelper does the same, but for a bus topology. InternetStackHelper helps create TCP/IP stacks on network nodes and Ipv4AddressHelper helps assign IP addresses to network interfaces.
* PointToPointHelper
* CsmaHelper
* InternetStackHelper
* Ipv4AddressHelper

The Doxygen documentation was made use of for a deeper understanding of these classes.

# Building Point-to-Point Topologies

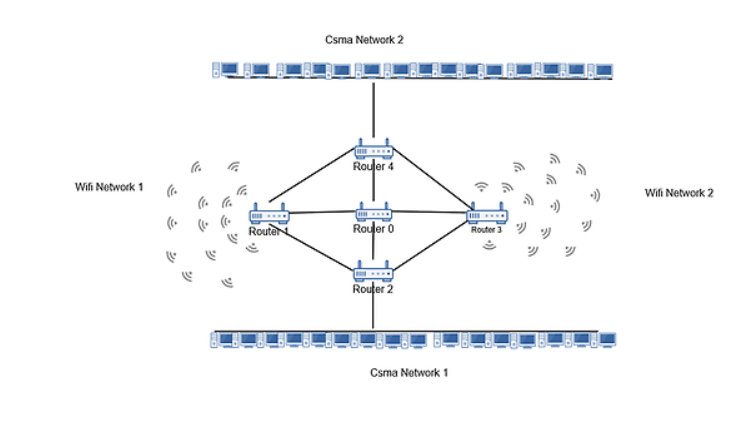
Any entity that is needed to be simulated using the NS-3 can be represented as a *node*. Basically, a node contains the following:

* *Applications*: A node would consist of several applications. Some built in applications are available in NS-3 such as OnOffApplication, UdpTraceClient, UdpEchoClient, UdpEchoServer and PacketSink.
* *Internet Stack*: A TCP/IP protocol suite is installed in every node in the simulation. Sockets and several protocols such as TCP, UDP, IPv4, IPv6, ICMP and ARP can be used.
* *Net Devices*: Nodes may have multiple NetDevices installed which is basically a Network Interface Card with a MAC address, an IP interface and an IP address. Several NetDevices are available on NS-3 such as PointToPointNetDevice, CsmaNetDevice, LoopbackNetDevice, LteEndNetDevice, etc.

# Basic steps for creating networks using NS-3

1. Create the network nodes as the starting
2. Install an Internet Stack on each node
3. Attach one or more network devices to each node and attach the device itself to the channel
4. Create a network interface for each network device
5. Assign an IP address to each network interface

# TOPOLOGY



The aim of the project was to simulate the network as shown above by implementing different inter-LAN and intra-LAN applications traffics, a link failure. There are four LANS in the network- two CSMA LAN and two Wi-fi LAN. Each of the LAN has 15 devices with the router excluded. The design consists of 5 routers with one router each for the LAN and the last router connected to all the routers for avoiding congestion on the point to point links. The connections and the IP addresses allotted to each of the devices are as follows:

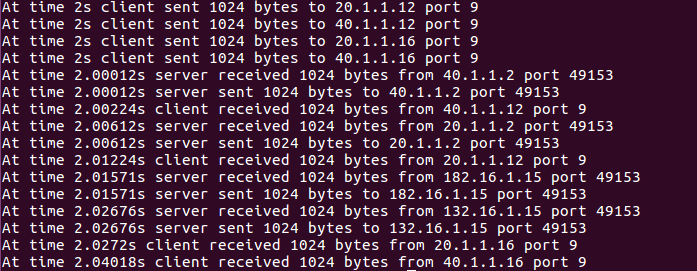
1. Router 1 acts as the access point for the 15 wifi nodes attached to it. The IP addresses allotted to the network are 182.16.1.0/24.
2. Router 3 acts as the access point for the 15 wifi nodes attached to it. The IP addresses allotted to the network are 132.16.1.0/24.
3. Router 2 connects to the CSMA network with 15 CSMA nodes with IP addresses in the range 20.1.1.0/24.
4. Router 4 connects to the CSMA network of 15 CSMA nodes with IP addresses in the range of 40.1.1.0/24.
5. Additionally, each of the routers are connected to each other by point-to-point links. All the routers 1,2,3 and 4 are connected to the central router 0 also via the point-to-point links. Each of the p2p links were given a speed of 5Mbps.

# Applications Implemented

The following applications were implemented in the network:

1. *UdpEcho*: This application consists of a client and server connection between the UdpEchoClient and UdpEchoServer. All the packets sent by the client are acknowledged by the server.

This application was used as both the inter-LAN and intra-LAN application. Intra-LAN was done between both the pairs of CSMA LAN and Wi-Fi LAN. The inter-LAN application was inside the CSMA LANS.

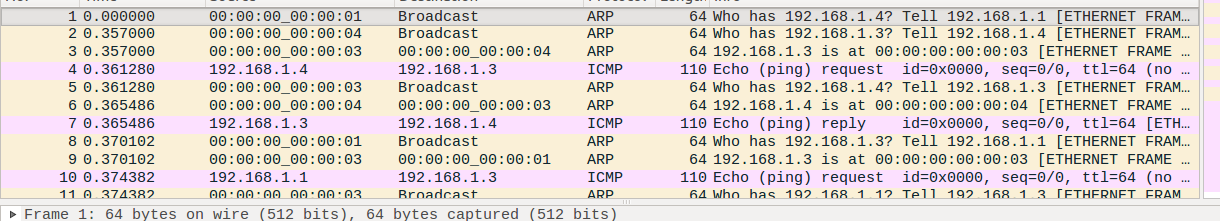


1. *OnOffApplication*: The OnOffApplication acts as a client application wherein packets are sent out during the “On” period and sending of packets is paused during the “Off” period.

The application was done between the CSMA LANS and the CSMA LAN1 and Wi-Fi LAN 1.

1. *Ping Application*: This creates a IPv4 ping application and associates it to a node.

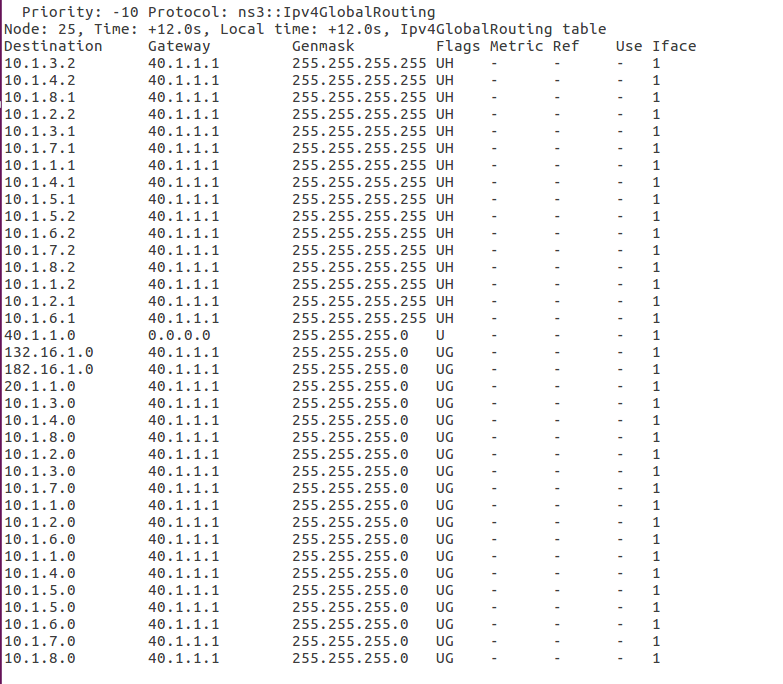
The sink was in the CSMA LAN 1 and it was pinged by nodes from all the LANS.



The network was not limited to TCP networks as the sockets were also initialized to UDP when the user wants to change to UDP networks.

# Routing Tables

The routing tables for the network simulation were written to a file. An example routing table look as follows:



For each destination from the node25 the routing table records the gateway through which it has pass first to reach the destination. Since the routing tables are formed dynamically, they always are formed in such a way that the destination is reached in minimum number of hops or the least cost path for effective communication between nodes. Suppose a node in CSMA LAN1 tries to end some packets to the node in Wi-Fi Lan 1, since a direct point to point link from router 2 to router 1 the packets are sent through this link rather than all the possible paths because this is the least cost path through which the packets can be sent. Also, the packets reach through this path with the minimum number hops out of all the paths.

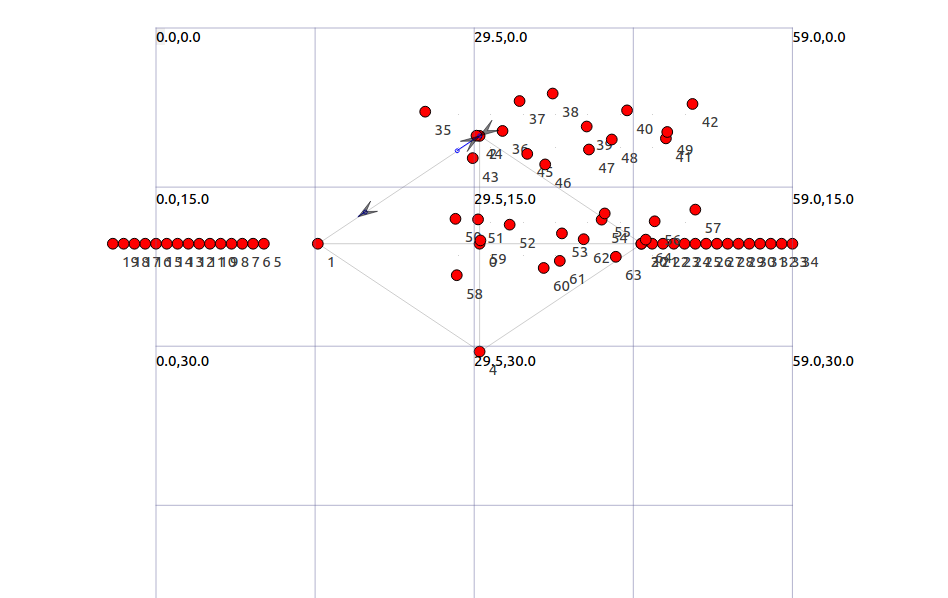
# Link Failure

# 

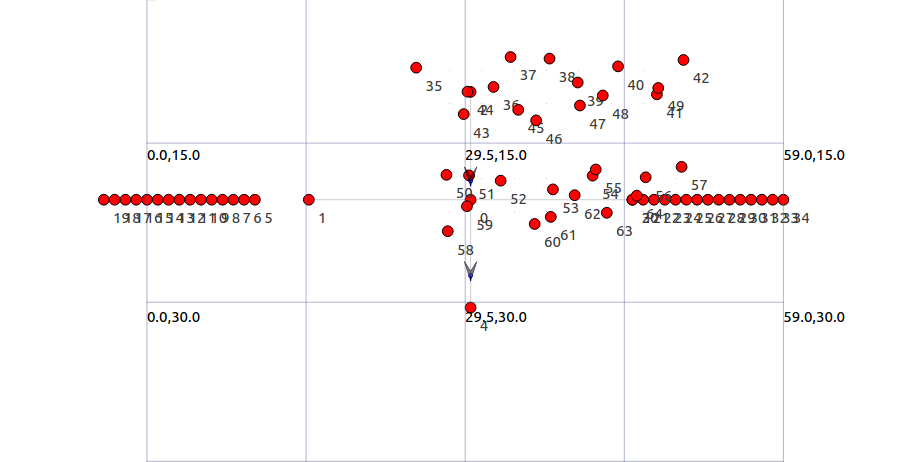
The link failure was implemented by removing the point to point connections between all the routers except that the individual connections between each of the routers 1,2,3,4 and the central router 0. Now when these links have failed the packets sent from any node in a particular LAN to any node in a different LAN, they go through the central router 0. This proves that the routing is dynamic and takes care of the link failure when there is any path between the source and destination using the existing links between the LANS. This was implemented with the help of existing applications in the project.

The link failure can be shown using the NetAnimation for the project.

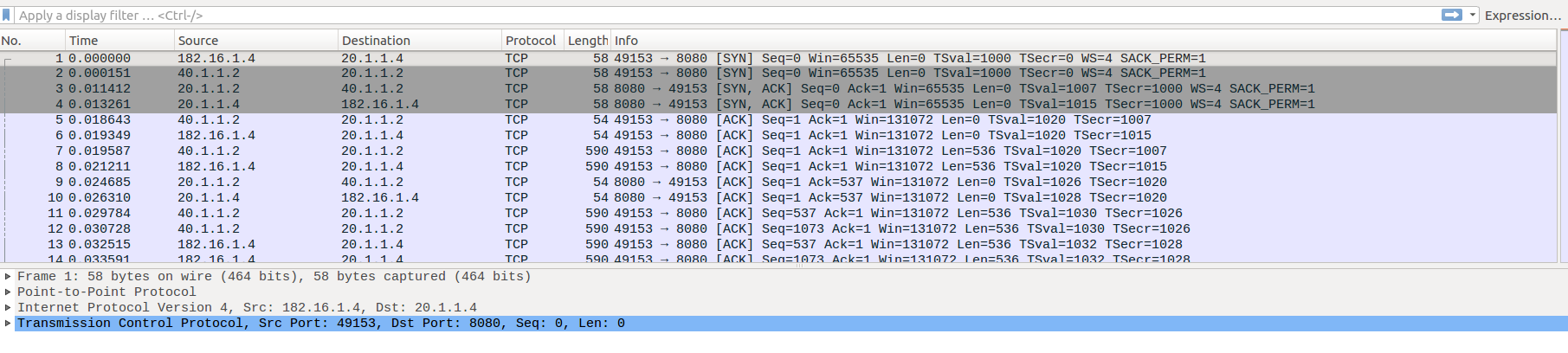
Before the link failure:



After Link Failure the transmission of packets happens through the central router 0. This can be shown as below:



# WireShark Pcap Files



In the above pcap file, it shows the data regarding the packets being sent during the simulation. For each communication between the nodes it displays the Protocol used to send the packet and the packet length.

CONCLUSION

Simulation of a network topology is achieved using ns3 tool. It is a discrete-event network simulator, targeted primarily for research and educational use. The ns-3 project is committed to building a solid simulation core that is well documented, easy to use and debug, and that caters to the needs of the entire simulation workflow, from simulation configuration to trace collection and analysis.

Furthermore, the ns-3 software infrastructure encourages the development of simulation models which are sufficiently realistic to allow ns-3 to be used as a real-time network emulator, interconnected with the real world and which allows many existing real-world protocol implementations to be reused within ns-3.

Using ns3 we could perform some analysis on the network created. We could assign the data rate of each node using which we could measure the throughput, good put, delay of the data being sent and the jitters through the network.

We could also perform congestion analysis between two nodes chosen for the traffic we assigned. We also performed the dynamic routing and the Link failure in a network which helped us to understand the real-world scenarios of the network behavior.

REFERENCES

[1] <https://en.wikipedia.org/wiki/Ns_(simulator)#ns-3>

[2] <https://www.nsnam.org/docs/release/3.25/doxygen/index.html>

[3] <https://www.youtube.com/watch?v=jx125o_kdDg>