Networking Lab Assignment 19

Network simulator NS-2

Albin Antony

10 February 2019

1 Network simulator NS-2

1.1 Aim

Install network simulator NS-2 in any of the Linux operating system and simulate wired and wireless scenarios.

1.2 Theory

1.2.1 NS-2

NS2 is an open-source simulation tool that runs on Linux. It is a discreet event simulator targeted at networking research and provides substantial support for simulation of routing, multicast protocols and IP protocols, such as UDP, TCP, RTP and SRM over wired and wireless (local and satellite) networks. It has many advantages that make it a useful tool, such as support for multiple protocols and the capability of graphically detailing network traffic. Additionally, NS2 supports several algorithms in routing and queuing. LAN routing and broadcasts are part of routing algorithms. Queuing algorithms include fair queuing, deficit round-robin and FIFO.

1.2.2 Installation

Step 1: Download ns2 from here The package downloaded will be named "ns-allinone-2.35.tar.gz". Copy it to the home folder. Then in a terminal use the following two commands to extract the contents of the package.:

```
cd ~/
tar -xvzf ns-allinone-2.35.tar.gz
```

Step 2:Building the dependencies

```
sudo apt-get install build-essential autoconf automake libxmu-dev
sudo apt-get install gcc-4.4
```

Now open the file named "ls.h" and scroll to the 137th line. In that change the word "erase" to "this->erase". The image below shows the line 137 (highlighted in the image below) after making the changes to the ls.h file.To open the file use the following command:

```
cd ~/ns-allinone-2.35/ns-2.35/linkstate
gedit ls.h
```

Now there is one more step that has to be done. We have to tell the ns which version of GCC will be used. To do so, go to your ns folder and type the following command:

```
sudo gedit ns-allinone-2.34/otcl-1.13/Makefile.in
```

In the file, change Change CC= @CC@ to CC=gcc-4.4, as shown in the image below.

Step 3:Installation

```
sudo su cd ~/ns-allinone-2.35/./install
```

Step 4:Setting up the environment path The final step is to tell the system, where the files for ns2 are installed or present. To do that, we have to set the environment path using the ".bashrc" file. In that file, we need to add a few lines at the bottom. The things to be added are given below. But for the path indicated below, many of those lines have "/home/albin/ns-allinone-2.35/...." , but that is where I have my extracted folder. Make sure you replace them with your path. For example, if you have installed it in a folder "/home/abc", then replace "/home/albin/ns-allinone-2.35/otcl-1.14" with "/home/abc/ns-allinone-2.35/otcl-1.14".

Step 5:Running NS-2

```
ns
```

1.2.3 TCL

Tcl (pronounced "tickle" or tee cee ell, /ti si l/) is a high-level, general-purpose, interpreted, dynamic programming language. It was designed with the goal of being very simple but powerful.Tcl casts everything into the mold of a command, even programming constructs like variable assignment and pro- cedure definition.

1.2.4 C++

NS2 uses OTcl to create and configure a network, and uses C++ to run simulation.

1.3 Program

1.3.1 Wired Network Simulation

```
# Create scheduler
set ns [new Simulator]


$ns color 1 darkmagenta
$ns color 2 yellow
$ns color 3 blue
$ns color 4 green
$ns color 5 black

#Tracing simulation results
set fin [open result.dat w]
$ns trace-all $fin

#Tracing for NAM(Network Animator)
set nfin [open out.nam w]
$ns namtrace-all $nfin
# Create a node
set n0 [$ns node]

# Create another node
set n1 [$ns  node]
# Create TCP agent and attach to first node
set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0

# Create TCP receiver and attach to second node
set tcp1 [new Agent/TCPSink]
$ns attach-agent $n1 $tcp1
# Connect both nodes with 1.5Mbps bandwidth, 5 milli seconds delay and DropTai
$ns duplex-link $n0 $n1 1Mb 5ms DropTail

# Connect tcp0 and tcp1 agents
$ns connect $tcp0 $tcp1

$tcp0 set fid_ 4
$tcp1 set fid_ 2
# Create a FTP object
set ftp [new Application/FTP]
# Attach ftp application with agent tcp0
 $ftp attach-agent $tcp0
 $ftp set fid_ 3
```

```
# Schedule events
$ns at 0.2 "$ftp start"
$ns at 2.0 "$ftp stop"
# Finish procedure to perform operation at the end of simulation

proc finish { } {
      # Mapping of global variable into local variable
      global ns fin nfin
        # Flush all buffers
        $ns flush-trace
        #close file objects
        close $fin
        close $nfin
        #Execute nam process in background
        exec nam out.nam &
        #Terminate current process
        exit 0
}
$ns at 2.2 "finish"
$ns run
```

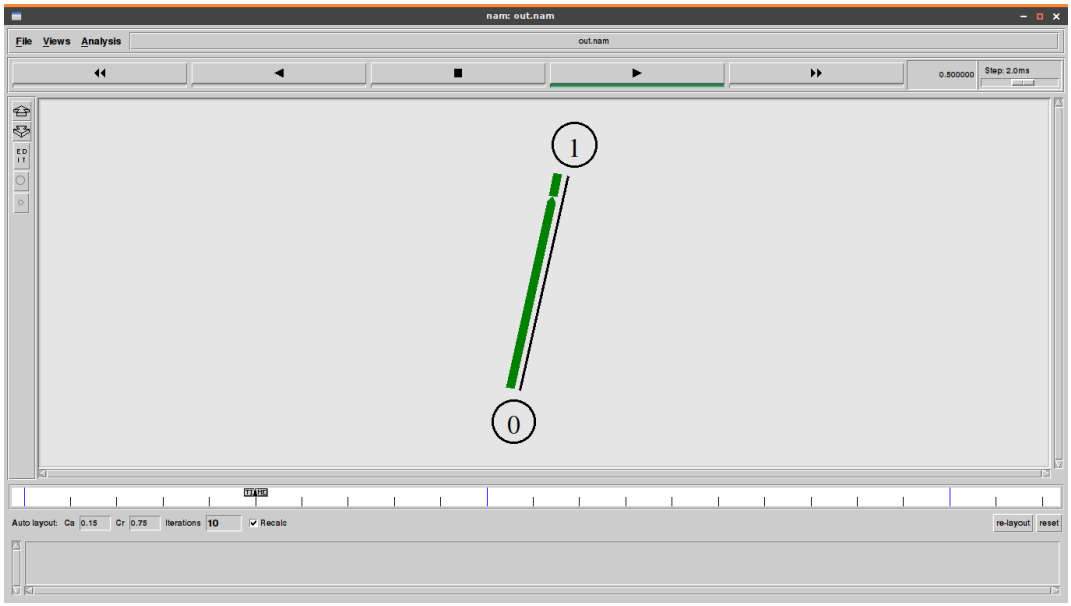

Figure 1: Wired Network

1.3.2 Wireless Network Simulation

```
#initialize the variables
```

```
set val(chan)           Channel/WirelessChannel    ;#Channel Type
set val(prop)           Propagation/TwoRayGround   ;# radio-propagation model
set val(netif)          Phy/WirelessPhy            ;# network interface type WAV
set val(mac)            Mac/802_11                 ;# MAC type
set val(ifq)            Queue/DropTail/PriQueue    ;# interface queue type
set val(ll)             LL                         ;# link layer type
set val(ant)            Antenna/OmniAntenna        ;# antenna model
set val(ifqlen)         50                         ;# max packet in ifq
set val(nn)             6                          ;# number of mobilenodes
set val(rp)             AODV                       ;# routing protocol
set val(x)   500   ;# in metres
set val(y)   500   ;# in metres
#Adhoc OnDemand Distance Vector

#creation of Simulator
set ns [new Simulator]

#creation of Trace and namfile
set tracefile [open wireless.tr w]
$ns trace-all $tracefile

#Creation of Network Animation file
set namfile [open wireless.nam w]
$ns namtrace-all-wireless $namfile $val(x) $val(y)

#create topography
set topo [new Topography]
$topo load_flatgrid $val(x) $val(y)

#GOD Creation - General Operations Director
create-god $val(nn)

set channel1 [new $val(chan)]
set channel2 [new $val(chan)]
set channel3 [new $val(chan)]

#configure the node
$ns node-config -adhocRouting $val(rp) \
  -llType $val(ll) \
  -macType $val(mac) \
  -ifqType $val(ifq) \
  -ifqLen $val(ifqlen) \
  -antType $val(ant) \
  -propType $val(prop) \
  -phyType $val(netif) \
  -topoInstance $topo \
```

```
  -agentTrace ON \
  -macTrace ON \
  -routerTrace ON \
  -movementTrace ON \
  -channel $channel1

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]

$n0 random-motion 0
$n1 random-motion 0
$n2 random-motion 0
$n3 random-motion 0
$n4 random-motion 0
$n5 random-motion 0

$ns initial_node_pos $n0 20
$ns initial_node_pos $n1 20
$ns initial_node_pos $n2 20
$ns initial_node_pos $n3 20
$ns initial_node_pos $n4 20
$ns initial_node_pos $n5 50

#initial coordinates of the nodes
$n0 set X_ 10.0
$n0 set Y_ 20.0
$n0 set Z_ 0.0

$n1 set X_ 210.0
$n1 set Y_ 230.0
$n1 set Z_ 0.0

$n2 set X_ 100.0
$n2 set Y_ 200.0
$n2 set Z_ 0.0

$n3 set X_ 150.0
$n3 set Y_ 330.0
$n3 set Z_ 0.0

$n4 set X_ 430.0
$n4 set Y_ 320.0
```

```
$n4 set Z_ 0.0

$n5 set X_ 270.0
$n5 set Y_ 120.0
$n5 set Z_ 0.0
#Dont mention any values above than 500 because in this example, we use X and Y

#mobility of the nodes
#At what Time? Which node? Where to? at What Speed?
$ns at 1.0 "$n0 setdest 490.0 340.0 35.0"
$ns at 1.0 "$n1 setdest 490.0 340.0 5.0"
$ns at 1.0 "$n2 setdest 330.0 100.0 10.0"
$ns at 1.0 "$n3 setdest 300.0 100.0 8.0"
$ns at 1.0 "$n4 setdest 300.0 130.0 5.0"
$ns at 1.0 "$n5 setdest 190.0 440.0 15.0"
#the nodes can move any number of times at any location during the simulation (r
$ns at 20.0 "$n5 setdest 100.0 200.0 30.0"

#creation of agents
set tcp [new Agent/TCP]
set sink [new Agent/TCPSink]
$ns attach-agent $n0 $tcp
$ns attach-agent $n5 $sink
$ns connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns at 1.0 "$ftp start"

set udp [new Agent/UDP]
set null [new Agent/Null]
$ns attach-agent $n2 $udp
$ns attach-agent $n3 $null
$ns connect $udp $null
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$ns at 1.0 "$cbr start"

$ns at 30.0 "finish"

proc finish {} {
 global ns tracefile namfile
 $ns flush-trace
 close $tracefile
 close $namfile
 exit 0
}
```

```
puts "Starting Simulation"
$ns run
```

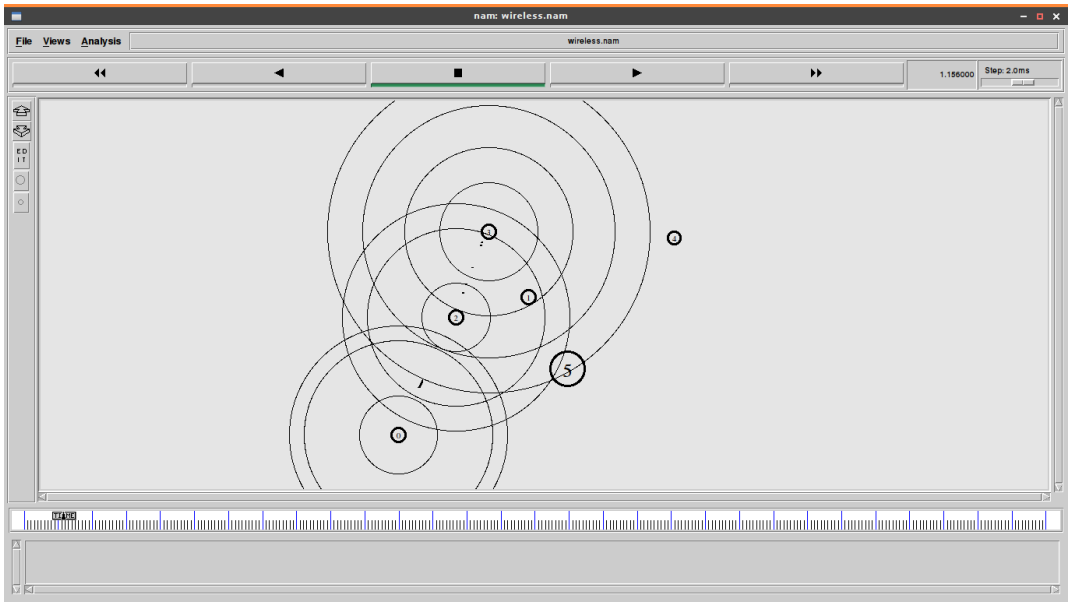

Figure 2: Wireless Network

1.4 Result

Installed NS-2 and implemented wired and wireless network simulation on NS-2 compiled on gcc 4.4 and executed on Debian 4.9 Kernel 4.9 and outputs were verified.