EXPERIMENT - XIX
NETWORK SIMULATOR NS-2

April 14, 2020

ADITHYA D RAJAGOPAL
ROLL NO : 9
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
COLLEGE OF ENGINEERING TRIVANDRUM

AIM

To install the network simulator NS-2 in the Linux operating system and simulate wired and wireless scenarios.

THEORY

NS

NS(network simulator) is a name for a series of discrete event network simulators, more specifically ns-1, ns-2, ns-3 and ns-4. All of these are discrete-event computer network simulators, primarily used in research and teaching. It is an open-source simulation tool that runs on Linux, 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.

Installation

STEP 1

Download the package for ns2 from here. Extraction of the zip file results in the extraction into a folder called "ns-allinone-2.35". A few packages and the GCC version 4.3 are prerequisites for ns2 for its proper working.

STEP 2

Necessary dependencies are to be installed and configured as follows:

- sudo apt-get install build-essential autoconf automake libxmu-dev
- One of the dependencies mentioned is the compiler GCC, which can be installed as follows
  sudo apt-get install gcc-[your-required-version]
- Now in the file named "ls.h", goto line 137 and change the word "erase" to "this->erase". To specify the version of gcc, change the Makefile.in file, by changing CC= @CC@ to CC=gcc-[your-version].

STEP 3

Now continue with the installation process by running the following command:
sudo su cd /ns-allinone-2.35/./install

STEP 4

The final step is to set the environment path using the ".bashrc" file. In that file, we need to add a few lines at the bottom corresponding to the environment paths. Make sure you replace them with your path. For example, if you have installed it in a folder "/home/username", then replace "/home/abc/ns-allinone-2.35/otcl-1.14" with "/home/username/ns-allinone-2.35/otcl-1.14".

STEP 5

After completing the above steps, you can try running ns in the terminal. A % denotes successful installation.

TCL

It is a high-level, general-purpose, interpreted, dynamic programming language. It casts everything into the mold of a command, including variable assignment and procedure definition. It supports multiple programming paradigms, including object- oriented, imperative and functional programming or procedural styles.

C++

NS2 uses C++ to run simulation. All C++ codes need to be compiled and linked to create an executable file. Since the body of NS2 is fairly large, the compilation time is not negligible.

SOURCE CODE

Wired Network

```
#Create a simulator object
set ns [new Simulator]
#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red
#Open the NAM trace file
set nf [open out.nam w]
$ns namtrace-all $nf
#Define a 'finish' procedure
proc finish {} {
global ns nf
$ns flush-trace
#Close the NAM trace file
close $nf
#Execute NAM on the trace file
exec nam out.nam &
exit 0
}
#Create four nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
#Create links between the nodes
$ns duplex-link $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns duplex-link $n2 $n3 1.7Mb 20ms DropTail
#Set Queue Size of link (n2-n3) to 10
$ns queue-limit $n2 $n3 10
#Give node position (for NAM)
$ns duplex-link-op $n0 $n2 orient right-down
```

```
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $n2 $n3 orient right
#Monitor the queue for link (n2-n3). (for NAM)
$ns duplex-link-op $n2 $n3 queuePos 0.5
#Setup a TCP connection
set tcp [new Agent/TCP]
$tcp set class_ 2
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink
$ns connect $tcp $sink
$tcp set fid_ 1
#Setup a FTP over TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ftp set type_ FTP
#Setup a UDP connection
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
set null [new Agent/Null]
$ns attach-agent $n3 $null
$ns connect $udp $null
$udp set fid_ 2
#Setup a CBR over UDP connection
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set type_ CBR
$cbr set packet_size_ 1000
$cbr set rate_ 1mb
$cbr set random_ false
#Schedule events for the CBR and FTP agents
$ns at 0.1 "$cbr start"
$ns at 1.0 "$ftp start"
$ns at 4.0 "$ftp stop"
$ns at 4.5 "$cbr stop"
```

```
#Detach tcp and sink agents (not really necessary)
$ns at 4.5 "$ns detach-agent $n0 $tcp ; $ns detach-agent $n3 $sink"
#Call the finish procedure after 5 seconds of simulation time
$ns at 5.0 "finish"
#Print CBR packet size and interval
puts "CBR packet size = [$cbr set packet_size_]"
puts "CBR interval = [$cbr set interval_]"
#Run the simulation
$ns run
```

Wireless Network

```
# Define options
set val(chan) Channel/WirelessChannel ;# channel type
set val(prop) Propagation/TwoRayGround ;# radio-propagation model
set val(netif) Phy/WirelessPhy ;# network interface type
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) 50 ;# number of mobilenodes
set val(rp) DSDV ;# routing protocol
set val(x) 1000 ;# X dimension of topography
set val(y) 1000 ;# Y dimension of topography
set val(stop) 150 ;# time of simulation end
set ns [new Simulator]
set tracefd [open simple.tr w]
set namtrace [open simwrls.nam w]
$ns trace-all $tracefd
$ns namtrace-all-wireless $namtrace $val(x) $val(y)
# set up topography object
set topo [new Topography]
$topo load_flatgrid $val(x) $val(y)
create-god $val(nn)
# configure the nodes
```

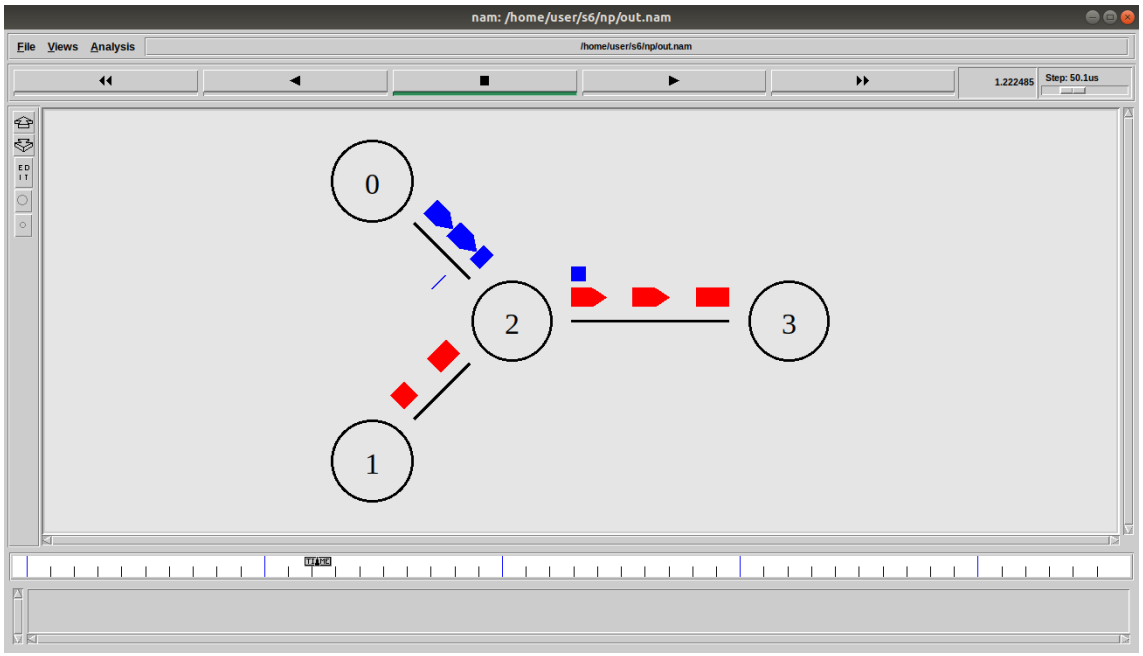
```
$ns node-config -adhocRouting $val(rp) \
-llType $val(ll) \
-macType $val(mac) \
-ifqType $val(ifq) \
-ifqLen $val(ifqlen) \
-antType $val(ant) \
-propType $val(prop) \
-phyType $val(netif) \
-channelType $val(chan) \
-topoInstance $topo \
-agentTrace ON \
-routerTrace ON \
-macTrace OFF \
-movementTrace ON
for {set i 0} {$i < $val(nn) } { incr i } {
set n($i) [$ns node]
}
# Provide initial location of mobilenodes
$n(0) set X_ 600.0
$n(0) set Y_ 547.0
$n(0) set Z_ 0.0
$n(1) set X_ 200.0
$n(1) set Y_ 332.0
$n(1) set Z_ 0.0
$n(2) set X_ -700.0
$n(2) set Y_ 345.0
$n(2) set Z_ 0.0
$n(3) set X_ 895.0
$n(3) set Y_ 200.0
$n(3) set Z_ 0.0
$n(4) set X_ -315.0
$n(4) set Y_ 421.0
$n(4) set Z_ 0.0
$n(5) set X_ 111.0
$n(5) set Y_ 111.0
```

```
$n(5) set Z_ 0.0
# Set a TCP connection between n(1) and n(31)
set tcp [new Agent/TCP/Newreno]
$tcp set class_ 2
set sink [new Agent/TCPSink]
$ns attach-agent $n(1) $tcp
$ns attach-agent $n(31) $sink
$ns connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns at 10.0 "$ftp start"
# Set a TCP connection between n(31) and n(43)
set tcp [new Agent/TCP/Newreno]
$tcp set class_ 2
set sink [new Agent/TCPSink]
$ns attach-agent $n(31) $tcp
$ns attach-agent $n(43) $sink
$ns connect $tcp $sink
#defining heads
$ns at 0.0 "$n(0) label CH"
$ns at 0.0 "$n(1) label Source"
#$ns at 0.0 "$n(2) label N2"
$ns at 10.0 "$n(5) setdest 785.0 228.0 5.0"
$ns at 13.0 "$n(26) setdest 700.0 20.0 5.0"
$ns at 15.0 "$n(14) setdest 115.0 85.0 5.0"
#Color change while moving from one group to another
$ns at 73.0 "$n(2) delete-mark N2"
$ns at 73.0 "$n(2) add-mark N2 pink circle"
$ns at 124.0 "$n(11) delete-mark N11"
$ns at 124.0 "$n(11) add-mark N11 purple circle"
$ns at 103.0 "$n(5) delete-mark N5"
$ns at 103.0 "$n(5) add-mark N5 white circle"
$ns at 87.0 "$n(26) delete-mark N26"
$ns at 87.0 "$n(26) add-mark N26 yellow circle"
$ns at 92.0 "$n(14) delete-mark N14"
```

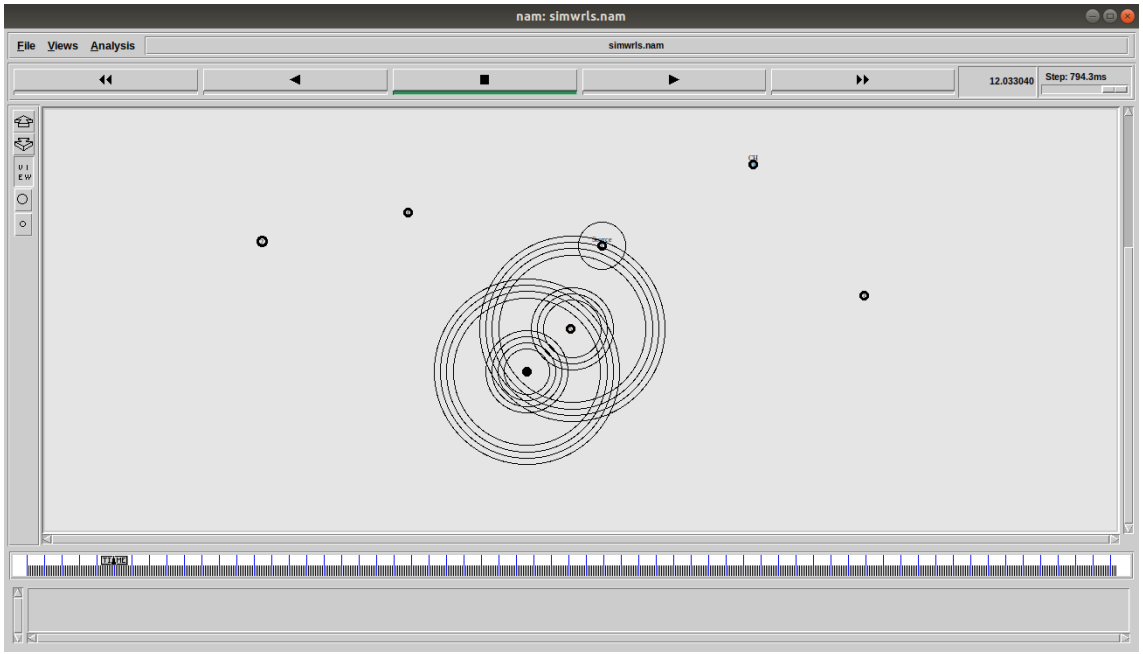
```
$ns at 92.0 "$n(14) add-mark N14 green circle"
# Define node initial position in nam
for {set i 0} {$i < $val(nn)} { incr i } {
# 20 defines the node size for nam
$ns initial_node_pos $n($i) 20
}
# Telling nodes when the simulation ends
for {set i 0} {$i < $val(nn) } { incr i } {
$ns at $val(stop) "$n($i) reset";
}
# ending nam and the simulation
$ns at $val(stop) "$ns nam-end-wireless $val(stop)"
$ns at $val(stop) "stop"
$ns at 150.01 "puts \"end simulation\" ; $ns halt"
proc stop {} {
global ns tracefd namtrace
$ns flush-trace
close $tracefd
close $namtrace
exec nam simwrls.nam &
}
$ns run
```

OUTPUT

Wired Network



Wireless Network



RESULT

The network simulator NS-2 has been installed and both wired and wireless networks have been simulated using NS-2 and the above output is obtained.