|  |  |
| --- | --- |
| Roll. No.: A016 | Name: Varun Khadayate |
| Sem/Year: VII/4 | Batch: 1 |
| Date of Experiment: 13/08/2022 | Date of Submission: 13/08/2022 |
| Grade -- |  |

# Aim

Create wireless network in ns2

# Theory

 Simple Wireless Program in NS2 is the best way to learn about how to code in NS2. NS-2 is one of the best simulation tools. It is used by majority of scholars today due to its highlighted features like support for OOPs concept, C++ programming fundamentals, real time emulation support etc. NS2 is used to simulate both wired and wireless networks; here we have focused on wireless network simulation in NS-2 due to its wide applicability. Regarding wired simulation in NS-2, refer our other articles available in this site.

Here, we have taken a simple wireless program in NS-2 to explain the students about how to work with wireless networks in NS-2. For further guidance and tutoring service on NS-2, approach us anytime, we are there for you at 24/7.

# Code

#Create a simulator object

set ns [new Simulator]

#Define different colors for data flows

$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 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 1Mb 10ms DropTail

$ns duplex-link $n1 $n2 1Mb 10ms DropTail

$ns duplex-link $n3 $n2 1Mb 10ms DropTail

$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 the link between node 2 and node 3

#set aa [$ns duplex-link-op $n2 $n3 queuePos 0.5]

#puts $aa

$ns duplex-link-op $n2 $n3 queuePos 0.5

$ns queue-limit $n2 $n3 10

#Create a UDP agent and attach it to node n0

set udp0 [new Agent/UDP]

$udp0 set class\_ 1

$ns attach-agent $n0 $udp0

# Create a CBR traffic source and attach it to udp0

set cbr0 [new Application/Traffic/CBR]

$cbr0 set packetSize\_ 500

$cbr0 set interval\_ 0.005

$cbr0 attach-agent $udp0

#Create a UDP agent and attach it to node n1

set udp1 [new Agent/UDP]

$udp1 set class\_ 2

$ns attach-agent $n1 $udp1

# Create a CBR traffic source and attach it to udp1

set cbr1 [new Application/Traffic/CBR]

$cbr1 set packetSize\_ 500

$cbr1 set interval\_ 0.005

$cbr1 attach-agent $udp1

#Create a Null agent (a traffic sink) and attach it to node n3

set null0 [new Agent/Null]

$ns attach-agent $n3 $null0

#Connect the traffic sources with the traffic sink

$ns connect $udp0 $null0

$ns connect $udp1 $null0

#Schedule events for the CBR agents

$ns at 0.5 "$cbr0 start"

$ns at 1.0 "$cbr1 start"

$ns at 4.0 "$cbr1 stop"

$ns at 4.5 "$cbr0 stop"

#Call the finish procedure after 5 seconds of simulation time

$ns at 5.0 "finish"

#Run the simulation

$ns run

# Output

A picture containing watch, clock

Description automatically generated

A picture containing diagram

Description automatically generated

A picture containing arrow

Description automatically generated

Arrow

Description automatically generated with medium confidence

# Conclusion

Hence, we were able to perform the experiment.