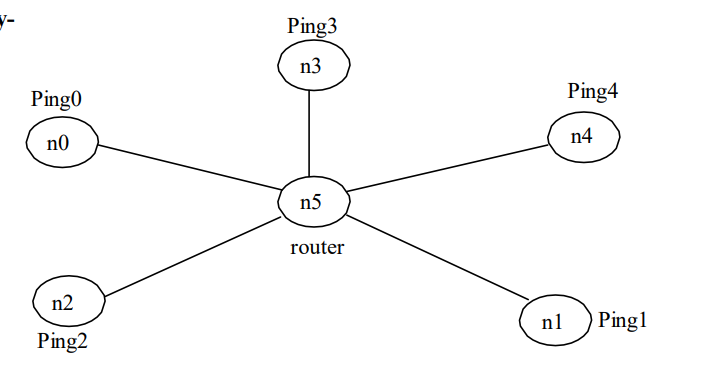
**Lab Experiment 4:**

Implement transmission of ping messages/traceroute over a network topology consisting of 6 nodes and find the number of packets dropped due to congestion.

**Topology-**

****

**Code:**

**#create Simulator object**

set ns [new Simulator]

**#open trace file**

set nt [open Lab4.tr w]

$ns trace-all $nt

**#open namtrace file**

set nf [open Lab4.nam w]

$ns namtrace-all $nf

**#create nodes**

set n0 [$ns node]

set n1 [$ns node]

set n2 [$ns node]

set n3 [$ns node]

set n4 [$ns node]

set n5 [$ns node]

**#label nodes**

$n0 label "ping0"

$n1 label "ping1"

$n2 label "ping2"

$n3 label "ping3"

$n4 label "ping4"

$n5 label "router"

**#create links, specify the type, nodes, bandwidth, delay and ARQ algorithm for it**

$ns duplex-link $n0 $n5 1Mb 10ms DropTail

$ns duplex-link $n1 $n5 1Mb 10ms DropTail

$ns duplex-link $n2 $n5 1Mb 10ms DropTail

$ns duplex-link $n3 $n5 1Mb 10ms DropTail

$ns duplex-link $n4 $n5 1Mb 10ms DropTail

**#set queue length**

$ns queue-limit $n0 $n5 5

$ns queue-limit $n1 $n5 5

$ns queue-limit $n2 $n5 2

$ns queue-limit $n3 $n5 5

$ns queue-limit $n4 $n5 2

$ns color 2 Red

$ns color 3 Blue

$ns color 4 Green

$ns color 5 Yellow

**#define ‘recv’ function for class Agent/Ping**

Agent/Ping instproc recv {from rtt} {

$self instvar node\_

puts "node [$node\_ id] received ping answer from $from with round-trip time $rtt ms"

}

**#create ping agent and attach them to node**

set p0 [new Agent/Ping]

$ns attach-agent $n0 $p0

$p0 set class\_ 1

set p1 [new Agent/Ping]

$ns attach-agent $n1 $p1

$p1 set class\_ 2

set p2 [new Agent/Ping]

$ns attach-agent $n2 $p2

$p2 set class\_ 3

set p3 [new Agent/Ping]

$ns attach-agent $n3 $p3

$p3 set class\_ 4

set p4 [new Agent/Ping]

$ns attach-agent $n4 $p4

$p4 set class\_ 5

**#connect 2 agents**

$ns connect $p2 $p4

$ns connect $p3 $p4

proc sendPingPacket { } {

global ns p2 p3

set intervalTime 0.001

set now [$ns now]

$ns at [expr $now + $intervalTime] "$p2 send"

$ns at [expr $now + $intervalTime] "$p3 send"

$ns at [expr $now + $intervalTime] "sendPingPacket"

}

proc finish { } {

global ns nt nf

$ns flush-trace

close $nt

close $nf

exec nam Lab4.nam &

exit 0

}

$ns at 0.1 "sendPingPacket"

$ns at 2.0 "finish"

$ns run

**Awk file**

BEGIN{

count=0;

}

{

if($1=="d")

count++

}

END{

printf ("Number of packets dropped is = %d\n",count);

}

Output:

node 3 received ping answer from 4 with round-trip time 66.4 ms

node 3 received ping answer from 4 with round-trip time 67.0 ms

node 3 received ping answer from 4 with round-trip time 66.5 ms

node 3 received ping answer from 4 with round-trip time 67.0 ms

node 3 received ping answer from 4 with round-trip time 66.5 ms

node 3 received ping answer from 4 with round-trip time 67.0 ms

**#awk -f lab4.awk lab4.tr**

Number of packets dropped is = 41

**Explanation:**

Agent/Ping instproc recv {from rtt} {

$self instvar node\_

puts "node [$node\_ id] received ping answer from $from with round-trip time $rtt ms"

}

* defines the **recv method** for the **Agent/Ping class.**
* Agent/Ping instproc recv {from rtt}: This line defines a method named recv for the Agent/Ping class. The method takes **two parameters**: from and rtt.
* $self instvar node\_: This line declares the variable **node\_** as an instance variable of the current object ($self). This variable is expected to **hold a reference to the node** associated with the Agent/Ping instance.
* puts "node [$node\_ id] received ping answer from $from with round-trip time $rtt ms": This line prints a message to the console using the puts command. The message includes information about the node that received the ping answer ([$node\_ id]), the source of the ping answer ($from), and the round-trip time ($rtt).
* In summary, this recv method is **responsible for handling received ping responses.** It prints a message indicating which node received a ping answer, the source of the answer, and the round-trip time.

set p0 [new Agent/Ping]

$ns attach-agent $n0 $p0

$p0 set class\_ 1

set p1 [new Agent/Ping]

$ns attach-agent $n1 $p1

$p1 set class\_ 2

set p2 [new Agent/Ping]

$ns attach-agent $n2 $p2

$p2 set class\_ 3

set p3 [new Agent/Ping]

$ns attach-agent $n3 $p3

$p3 set class\_ 4

set p4 [new Agent/Ping]

$ns attach-agent $n4 $p4

$p4 set class\_ 5

* This code is **responsible for creating instances** of the Agent/Ping class, **attaching them to specific nodes, and setting a class\_ attribute for each agent.**
* set p0 [new Agent/Ping]: Creates a new instance of the Agent/Ping class and assigns it to the variable p0.
* $ns attach-agent $n0 $p0: **Attaches** the Agent/Ping instance p0 to the node $n0.
* $p0 set class\_ 1: **Sets the class\_ attribute** of p0 to the value 1.
* This pattern is repeated for agents p1 through p4, each associated with a specific node ($n1 through $n4) and assigned a unique class\_ value ranging from 2 to 5.
* In the context of network simulation, these agents are likely used to generate ping traffic between nodes, and the class\_ attribute may be used to **differentiate or categorize different types of agents.** The class\_ attribute is a custom attribute that is being used in the simulation for a specific purpose.

proc sendPingPacket { } {

global ns p2 p3

set intervalTime 0.001

set now [$ns now]

$ns at [expr $now + $intervalTime] "$p2 send"

$ns at [expr $now + $intervalTime] "$p3 send"

$ns at [expr $now + $intervalTime] "sendPingPacket"

}

* The sendPingPacket procedure is **responsible for scheduling the sending of ping packets by agents p2 and p3.**
* global ns p2 p3: Declares that the variables ns, p2, and p3 are global, meaning **they can be accessed and modified within the procedure.**
* set intervalTime 0.001: **Sets the intervalTime variable to 0.001 seconds**. This variable represents the **time interval between successive ping packet transmissions.**
* set now [$ns now]: **Retrieves the current simulation time** using the **$ns now command** and assigns it to the **variable now.**
* $ns at [expr $now + $intervalTime] "$p2 send": **Schedules the send method of agent p2 to be executed after the current simulation time ($now) plus the specified intervalTime.**
* $ns at [expr $now + $intervalTime] "$p3 send": Similar to the previous line, schedules the send method of agent p3 to be executed after the specified time interval.
* $ns at [expr $now + $intervalTime] "sendPingPacket": Schedules the sendPingPacket procedure to be called again after the specified time interval, creating a loop for periodic ping packet transmissions.
* In summary, this procedure is used to **create a continuous loop that schedules the sending of ping packets by agents p2 and p3 at regular intervals** during the ns-2 simulation. The **specific time interval** is controlled by the intervalTime variable. This mechanism **simulates the periodic behavior of ping traffic** in the network.

**$ns at 0.1 "sendPingPacket"**

* The line $ns at 0.1 "sendPingPacket" in the ns-2 script schedules the execution of the sendPingPacket procedure after a delay of 0.1 seconds (100 milliseconds)
* $ns at 0.1 "sendPingPacket": This command schedules the execution of the sendPingPacket procedure at simulation time 0.1 seconds.
* The simulation time is an important aspect of discrete event simulation systems like ns-2, and it represents the elapsed time in the simulated environment..
* In this context, it means that **after 0.1 seconds into the simulation**, the sendPingPacket procedure will be **executed, initiating the periodic sending of ping packets** by agents p2 and p3 at the specified interval.

**$ns at 2.0 "finish"**

* The line $ns at 2.0 "finish" schedules the execution of the finish procedure **at simulation time 2.0 seconds.**
* $ns at 2.0 "finish": This command **schedules the execution of the finish procedure at simulation time 2.0 seconds**.
* The finish procedure, as described in the script, is responsible for flushing traces, closing trace files, opening the nam visualizer, and exiting the simulation. This ensures that the simulation results are properly handled and visualized after the specified simulation time.