NS2 – LAB 1
Dr. Amjad Hawash
Network & Information Security Program.
An-Najah National University.

Network Topology
Ring Topology

Aim:
To create scenario and study the performance of token ring protocols through simulation.

Token ring is a LAN protocol operating in the MAC layer. Token ring is standardized as per IEEE 802.5. Token ring can operate at speeds of 4mbps and 16 mbps. The operation of token ring is as follows: When there is no traffic on the network a simple 3-byte token circulates the ring. If the token is free then the station may seize the token and start sending the data frame. As the frame travels around the ring ach station examines the destination address and is either forwarded (if the recipient is another node) or copied. After copying4 bits of the last byte is changed. This packet then continues around the ring till it reaches the originating station. After the frame makes a round trip the sender receives the frame and releases a new token onto the ring.

Steps:
1. Create a simulator object
2. Define different colors for different data flows
3. Open a nam trace file and define finish procedure then close the trace file, and execute nam on trace file.
4. Create five nodes that forms a network numbered from 0 to 4
5. Create duplex links between the nodes to form a Ring Topology.
6. Setup TCP Connection between n(1) and n(3)
7. Apply CBR Traffic over TCP 8. Schedule events and run the program.
8. After running the code, click the button “re-layout” to redraw the network.
9. click play button to start the simulation process.

After finishing executing the code, do the following:
1- Currently, the data transmission is between nodes n1 and n3. Make it also between n4 and n5.
2- Change the data transmission between nodes n1 and n3 to be UDP instead of TCP.
3- Redesign the network to have 10 nodes instead of 5 and make the communication between nodes: n1, n3 and n7 to be TCP.
4- For each of the previous steps, save the whole TCL code to be used in the report.

TCL Code:

```
#Create a simulator object
set ns [new Simulator]

#Open the nam trace file
set nf [open out.nam w]
$ns namtrace-all $nf

#Define a 'finish' procedure
proc finish {} {
# access the $ns and $nf variables from within the procedure
 global ns nf
#This command flushes the trace buffer and is typically called before the simulation run ends.
# it saves the data from memory to the opened file.
 $ns flush-trace
 #Close the trace file
 close $nf
 #Executenam on the trace file
 exec nam out.nam &
 exit0
}

#Create five 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]

#Create links between the nodes

$ns duplex-link $n0 $n1 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 1Mb 10ms DropTail
$ns duplex-link $n2 $n3 1Mb 10ms DropTail
$ns duplex-link $n3 $n4 1Mb 10ms DropTail
$ns duplex-link $n4 $n5 1Mb 10ms DropTail
$ns duplex-link $n5 $n0 1Mb 10ms DropTail

#Create a TCP agent and attach it to node n0
set tcp0 [new Agent/TCP]
$tcp0 set class_ 1
$ns attach-agent $n1 $tcp0

#Create a TCP Sink agent (a traffic sink) for TCP and attach it to node n3
set sink0 [new Agent/TCPSink]
$ns attach-agent $n3 $sink0

#Connect the traffic sources with the traffic sink
```

```
$ns connect $tcp0 $sink0

# Create a CBR traffic source and attach it to tcp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.01
$cbr0 attach-agent $tcp0

#Schedule events for the CBR agents
#We want the first CBR agent to start sending at 0.5 seconds and to stop at 4.5 seconds

$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"

#Print CBR packet size and interval
puts " cbr0 packet size = [$cbr0 set packet_size_]"
puts " cbr0 interval = [$cbr0 set interval_]"

#Call the finish procedure after 5 seconds of simulation time
$ns at 5.0 "finish"


#Run the simulation
$ns run
```