Question: 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 (no reserved by a station of higher priority as explained later) 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.

ALGORITHM:

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.

PROGRAM:

```
#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 {} { global ns nf $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 $ns at 0.5 "$cbr0 start" $ns at 4.5 "$cbr0 stop"
#Call the finish procedure after 5 seconds of simulation time $ns at 5.0 "finish" #Run the simulation $ns run
```

AIM:

Question 2: To create scenario and study the performance of token ring protocols through simulation.

THEORY:

Star networks are one of the most common computer network topologies. In its simplest form, a star network consists of one central switch, hub or computer, which acts as a conduit to transmit messages. This consists of a central node, to which all other nodes are connected; this central node provides a common connection point for all nodes through a hub. In star topology, every node (computer workstation or any other peripheral) is connected to a central node called a hub or switch. The switch is the server and the peripherals are the clients. Thus, the hub and leaf nodes, and the transmission lines between them, form a graph with the topology of a star. If the central node is passive, the originating node must be able to tolerate the reception of an echo of its own transmission, delayed by the two-way transmission time (i.e. to and from the central node) plus any delay generated in the central node. An active star network has an active central node that usually has the means to prevent echo-related problems.

The star topology reduces the damage caused by line failure by connecting all of the systems to a central node. When applied to a bus-based network, this central hub rebroadcasts all transmissions received from any peripheral node to all peripheral nodes on the network, sometimes including the originating node. All peripheral nodes may thus communicate with all others by transmitting to, and

receiving from, the central node only. The failure of a transmission line linking any peripheral node to the central node will result in the isolation of that peripheral node from all others, but the rest of the systems will be unaffected.

ALGORITHM:

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 six nodes that forms a network numbered from 0 to 5
5. Create duplex links between the nodes to form a STAR Topology
6. Setup TCP Connection between n(1) and n(3)
7. Apply CBR Traffic over TCP
8. Schedule events and run the program.

PROGRAM:

#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 {} { global ns nf $ns flush-trace #Close the trace file close $nf #Executenam on the trace file exec nam out.nam & exit0
} #Create six nodes set n0 [$ns node] set n1 [$ns node] set n2 [$ns node] set n3 [$ns node] set n4 [$ns node] set n5 [$ns node] #Change the shape of center node in a star topology $n0 shape square #Create links between the nodes $ns duplex-link $n0 $n1 1Mb 10ms DropTail $ns duplex-link $n0 $n2 1Mb 10ms DropTail $ns duplex-link $n0 $n3 1Mb 10ms DropTail $ns duplex-link $n0 $n4 1Mb 10ms DropTail $ns duplex-link $n0 $n5 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 $ns at 0.5 "$cbr0 start" $ns at 4.5 "$cbr0 stop" #Call the finish procedure after 5 seconds of simulation time $ns at 5.0 "finish" #Run the simulation $ns run