**Aim:** To write a program to create Mesh topology with n-nodes.

SOFTWARE REQUIRED:

**NS-2**

**Theory: Implementing Mesh Topology in NS2**

To implement the Mesh Topology that each node is connected to the several other nodes which allowing for numerous paths among any pair of the nodes. It delivers high redundancy and reliability, if one link fails then the data can get the alternative routes. In NS2 that is Network Simulator 2, we can simulate a mesh topology by making several duplex links among the nodes that forming a web of connections. We present stepwise approach that supports to execute this topology in the simulation NS2:

**Steps to Implement a Mesh Topology in NS2:**

1. **Set Up the Mesh Network Topology:**
   * Describe the nodes in the network.
   * Make a duplex links among the several nodes, make sure that every node is connected to at least two or more nodes that mimicking a mesh of connections.
2. **Simulate Traffic Flow:**
   * Traffic can flow among the any pair of nodes, and also the routing mechanism will ascertain the path the data takes.

**Algorithm:**

1. Create a simulator object.

2. Define different colours for different data flows.

2. Open a NAM trace file and define finish procedure.

3. Create client and server nodes that forms a network node.

4. Create duplex links between the nodes and add Orientation to the nodes.

5. Setup TCP Agents for communication between client and server pairs.

6. Generate CBR Traffic over the TCP connection.

7. Schedule events and run the program.

**Program:**

**# Create a new simulator**

set ns [new Simulator]

**# Open trace file for output**

set tracefile [open out.tr w]

$ns trace-all $tracefile

**# Define nodes (4 nodes for simplicity)**

set n0 [$ns node]; # Node 0

set n1 [$ns node]; # Node 1

set n2 [$ns node]; # Node 2

set n3 [$ns node]; # Node 3

**# Create duplex links to form a mesh topology where each node is connected to multiple nodes**

$ns duplex-link $n0 $n1 1Mb 10ms DropTail; # Link between node 0 and node 1

$ns duplex-link $n0 $n2 1Mb 10ms DropTail; # Link between node 0 and node 2

$ns duplex-link $n0 $n3 1Mb 10ms DropTail; # Link between node 0 and node 3

$ns duplex-link $n1 $n2 1Mb 10ms DropTail; # Link between node 1 and node 2

$ns duplex-link $n1 $n3 1Mb 10ms DropTail; # Link between node 1 and node 3

$ns duplex-link $n2 $n3 1Mb 10ms DropTail; # Link between node 2 and node 3

**# Define TCP agents and sinks for communication between nodes**

set tcp0 [new Agent/TCP]

set sink0 [new Agent/TCPSink]

$ns attach-agent $n0 $tcp0

$ns attach-agent $n1 $sink0

$ns connect $tcp0 $sink0

**# Simulate traffic from node 0 to node 1**

set ftp0 [new Application/FTP]

$ftp0 attach-agent $tcp0

$ns at 1.0 “$ftp0 start”

**# Set up another TCP connection from node 2 to node 3**

set tcp1 [new Agent/TCP]

set sink1 [new Agent/TCPSink]

$ns attach-agent $n2 $tcp1

$ns attach-agent $n3 $sink1

$ns connect $tcp1 $sink1

**# Simulate traffic from node 2 to node 3**

set ftp1 [new Application/FTP]

$ftp1 attach-agent $tcp1

$ns at 2.0 “$ftp1 start”

**# End the simulation after 10 seconds**

$ns at 10.0 “finish”

proc finish {} {

global ns tracefile

$ns flush-trace

close $tracefile

exit 0

}

**# Run the simulation**

$ns run

**Output:**