**Ex. No. 9** **Congestion Control** **Karthik D**

**29 September 2021** **UCS1511 - Networks Lab** **195001047**

## Aim

To analyze the difference between the congestion control mechanism offered by Tahoe and Reno TCP agents using the Network Simulator (NS2) tool and using a simple tcl-script generated network schedule comprising of FTP traffic using TCP packets

## Question

Write a .tcl script to simulate:

1. Create 3 nodes and the links between the nodes as
    a. 0→1 10Mb, 10 ms duplex link
    b. 1→2 2Mb, 10 ms duplex link
    c. Align the nodes properly
    d. Setup a TCP/Tahoe connection over 0 and 2 and its flow id, window size, packet
    e. Show the simulation in network animator and in trace file

Write tcl script to simulate

1. Create 3 nodes and the links between the nodes as
    a. 0 →1 10Mb, 10 ms duplex link
    b. 1→2 2Mb, 10 ms duplex link
    c. Align the nodes properly
    d. Setup a TCP/Reno connection over 0 and 2 and mention the same flow id, window size, packet used for TCP/Tahoe
    e. Show the simulation in network animator and in trace file

## Algorithm

### 1. Tahoe Agent

**Step 1:** Instantiate a new Simulator object. Define colors for different flows for visual distinction

**Step 2:** Set the simulator to record the simulations in the NAM format. Supply the output file name to be generated at the end

**Step 3:** Define three different nodes in the network simulator — n0, n1 and n2. Create duplex-links between ns2 and every other node. Define the link bandwidth, propagation delay and the type of packet-queuing procedure to follow at the packet receiver end.

**Step 4:** Create and attach a TCP Tahoe agent instance to node n0 and a TCP-sink agent instance to node n2 to receive these packets and send back an acknowledgement. Setup a logical connection between the n0 and n1 as well as n1 and n2 i.e set their destination IP and port addresses. Attach an FTP agent as the application layer protocol to the TCP agent.

**Step 5:** Set other connection parameters including window size, packet size and flow id

**Step 6:** Schedule the FTP connection to operate from between specific time instances.

**Step 7:** Terminate the simulation after 5ns. Flush the simulation data collected to the NAM file object created in step-2. Run the generated output trace file using the NAM simulator.

**2. Reno Agent**

**Step 1:** Instantiate a new Simulator object. Define colors for different flows for visual distinction

**Step 2:** Set the simulator to record the simulations in the NAM format. Supply the output file name to be generated at the end

**Step 3:** Define three different nodes in the network simulator — n0, n1 and n2. Create duplex-links between ns2 and every other node. Define the link bandwidth, propagation delay and the type of packet-queuing procedure to follow at the packet receiver end.

**Step 4:** Create and attach a TCP Reno agent instance to node n0 and a TCP-sink agent instance to node n2 to receive these packets and send back an acknowledgement. Setup a logical connection between the n0 and n1 as well as n1 and n2 i.e set their destination IP and port addresses. Attach an FTP agent as the application layer protocol to the TCP agent.

**Step 5:** Set other connection parameters including window size, packet size and flow id

**Step 6:** Schedule the FTP connection to operate from between specific time instances.

**Step 7:** Terminate the simulation after 5ns. Flush the simulation data collected to the NAM file object created in step-2. Run the generated output trace file using the NAM simulator.

**<u>TCL Program Code</u>**

1. <u>tcp-tahoe.tcl - Trace file generation script for TCP Tahoe agent</u>

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

#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red

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

#Define a 'finish' procedure
proc finish {} {
      global ns nf
      $ns flush-trace
      #Close the NAM trace file
      close $nf
      #Execute NAM on the trace file
      exec nam tahoe_out.nam &
      exit 0
}

#Create three nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]

#Create links between the nodes
$ns duplex-link $n0 $n1 10Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail

#Set Queue Size of link (n1-n2) to 5
$ns queue-limit $n1 $n2 5

#Give node position (for NAM)
```

```
$ns duplex-link-op $n0 $n1 orient right-up
$ns duplex-link-op $n1 $n2 orient right-down

#Monitor the queue for link (n2-n3). (for NAM)
$ns duplex-link-op $n1 $n2 queuePos 0.5

#Setup a TCP-Tahoe connection
#Using Agent/TCP creates a Tahoe connection
set tcp_1 [new Agent/TCP]
$tcp_1 set class_ 2
#Set the source node
$ns attach-agent $n0 $tcp_1

#Create the sink for the connection
set sink_1 [new Agent/TCPSink]
$ns attach-agent $n2 $sink_1
#Establish the connection
$ns connect $tcp_1 $sink_1

#Set other properties
#default packet-size is 1000
$tcp_1 set fid_ 1
$tcp_1 set window_ 20
$tcp_1 set packetSize_ 1200

#Setup a FTP over TCP connection
set ftp_1 [new Application/FTP]
$ftp_1 attach-agent $tcp_1
$ftp_1 set type_ FTP

#Schedule events for the TCP agent
$ns at 0.5 "$ftp_1 start"
$ns at 5.0 "$ftp_1 stop"

#Call the finish procedure after 5.5 seconds of simulation time
$ns at 5.5 "finish"

#Run the simulation
$ns run
```

2. tcp-reno.tcl - Trace file generation script for TCP Reno agent

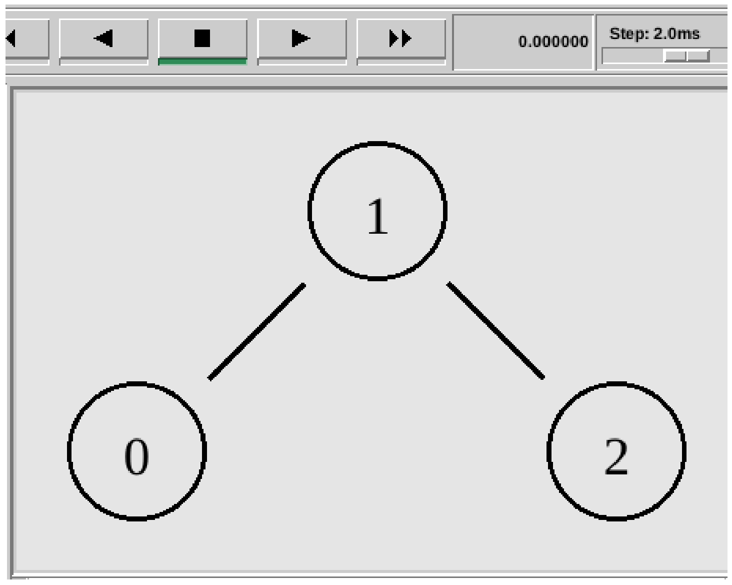
```
#Create a simulator object
set ns [new Simulator]

#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red

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

#Define a 'finish' procedure
proc finish {} {
      global ns nf
      $ns flush-trace
      #Close the NAM trace file
      close $nf
      #Execute NAM on the trace file
      exec nam tahoe_out.nam &
      exit 0
}

#Create three nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]

#Create links between the nodes
$ns duplex-link $n0 $n1 10Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail

#Set Queue Size of link (n1-n2) to 5
$ns queue-limit $n1 $n2 5

#Give node position (for NAM)
$ns duplex-link-op $n0 $n1 orient right-up
$ns duplex-link-op $n1 $n2 orient right-down
```

```
#Monitor the queue for link (n2-n3). (for NAM)
$ns duplex-link-op $n1 $n2 queuePos 0.5

#Setup a TCP-Tahoe connection
#Using Agent/TCP creates a Tahoe connection
set tcp_1 [new Agent/TCP/Reno]
$tcp_1 set class_ 2
#Set the source node
$ns attach-agent $n0 $tcp_1

#Create the sink for the connection
set sink_1 [new Agent/TCPSink]
$ns attach-agent $n2 $sink_1
#Establish the connection
$ns connect $tcp_1 $sink_1

#Set other properties
#default packet-size is 1000
$tcp_1 set fid_ 1
$tcp_1 set window_ 20
$tcp_1 set packetSize_ 1200

#Setup a FTP over TCP connection
set ftp_1 [new Application/FTP]
$ftp_1 attach-agent $tcp_1
$ftp_1 set type_ FTP

#Schedule events for the TCP agent
$ns at 0.5 "$ftp_1 start"
$ns at 5.0 "$ftp_1 stop"

#Call the finish procedure after 5.5 seconds of simulation time
$ns at 5.5 "finish"

#Run the simulation
$ns run
```

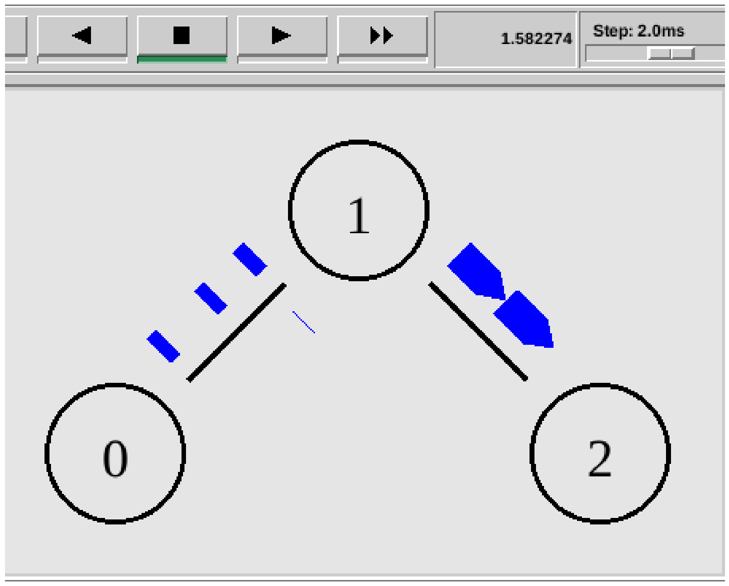
**Sample Output**

1. **TCP Tahoe Agent**

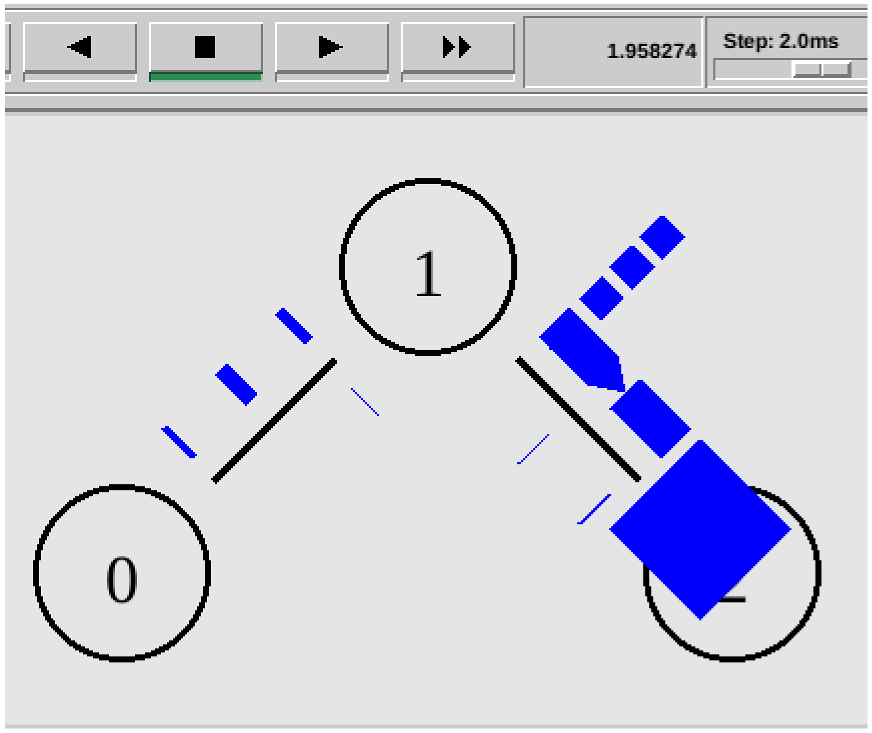
    a. **Initial State - No Traffic ( Time 0.000 )**



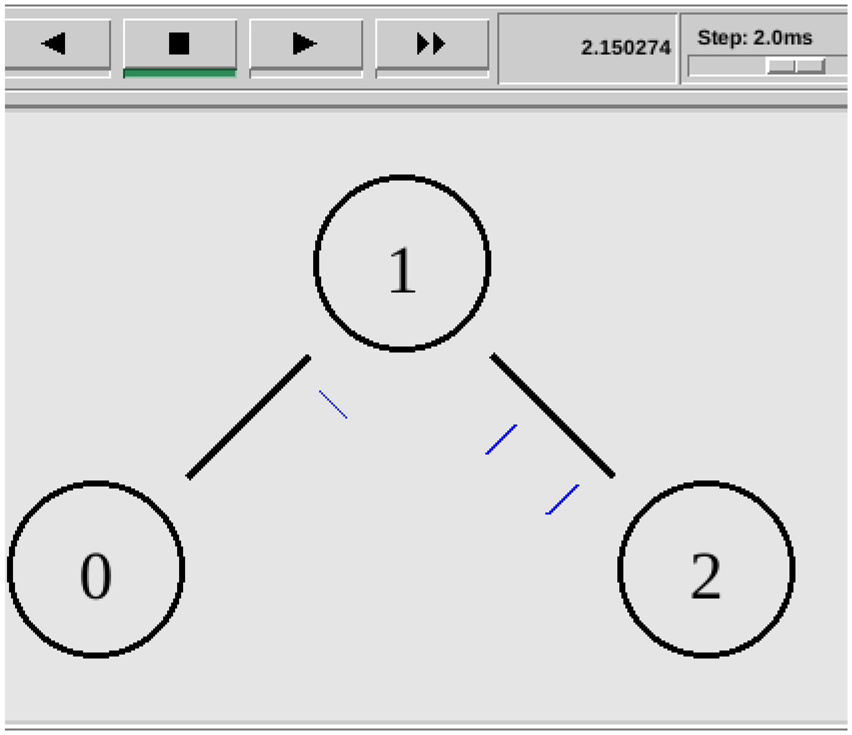
b. **Transmission Starts from CWD=1**



c. **Packets start queueing at the bottleneck (node-1) and packets are being dropped due to congestion in the pipeline. Duplicate acknowledgements start reaching the source (node-0)**

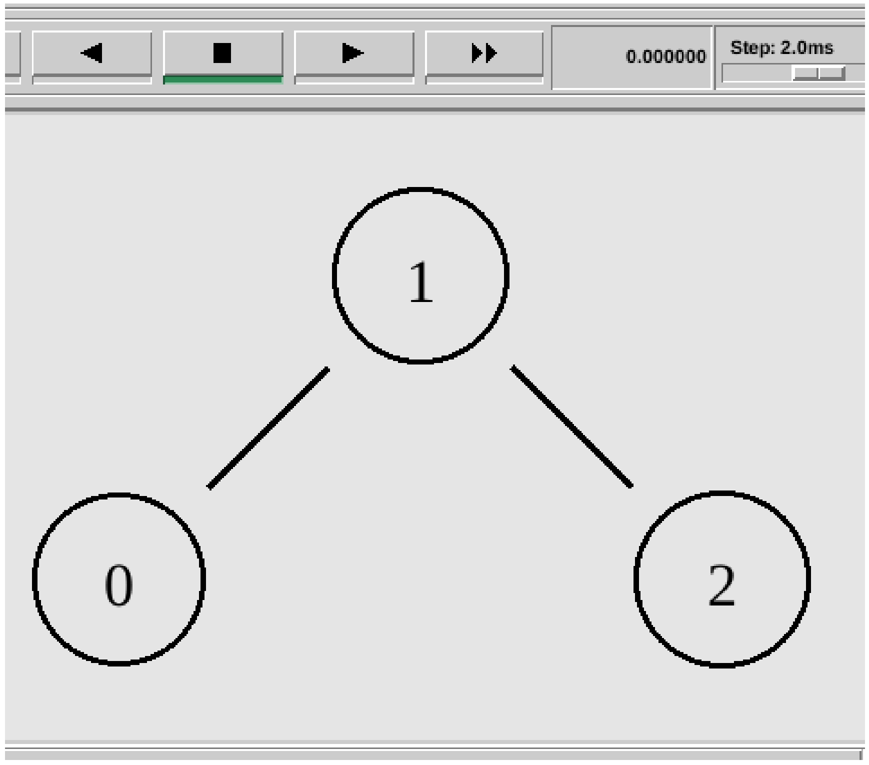


d. **CWD is reset to 1 and the entire transmission pipe is emptied. Transmission re-starts as though this is the first packet**

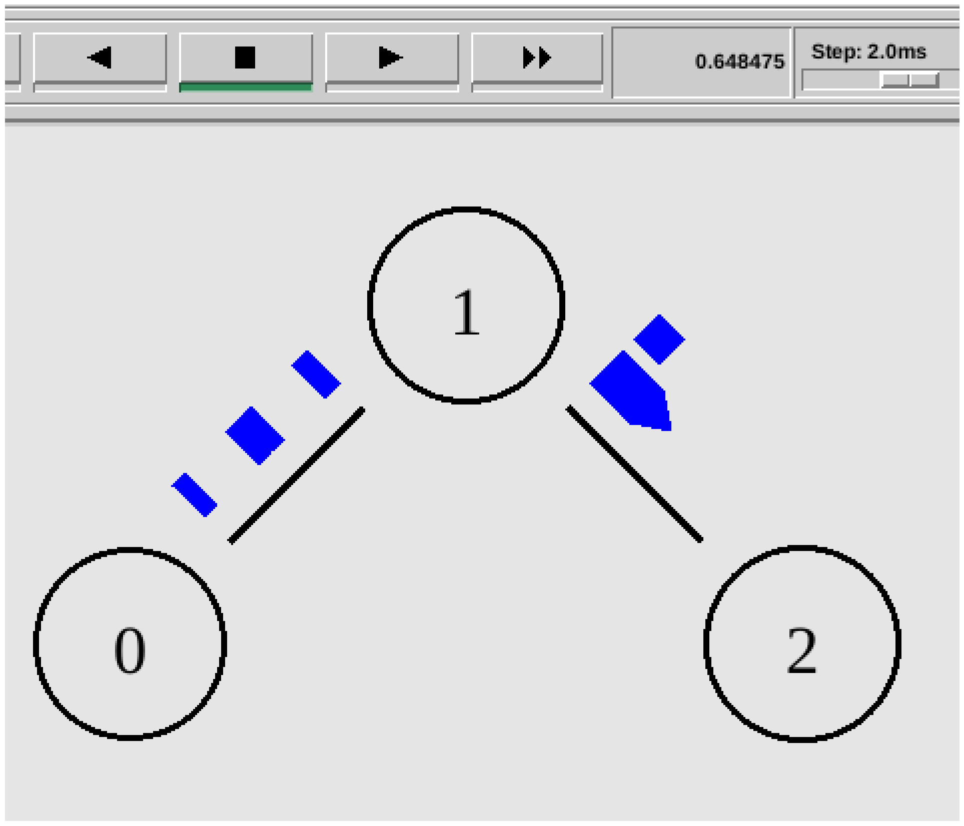


2. **TCP Reno Agent**

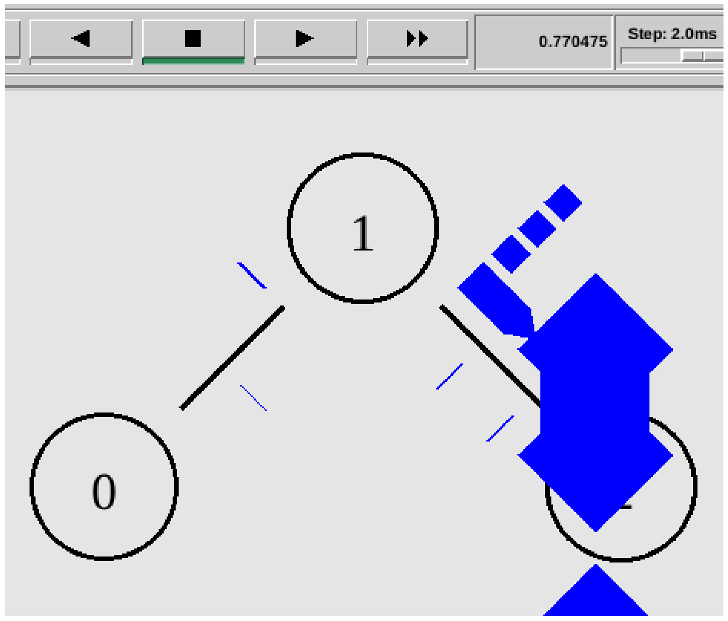
    a. **Initial State - No Traffic ( Time 0.000 )**



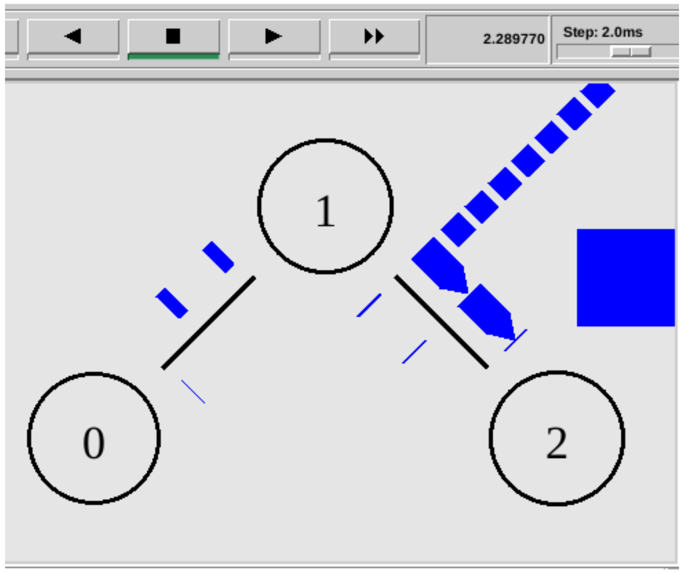
b. **Transmission Starts with CWD=1**



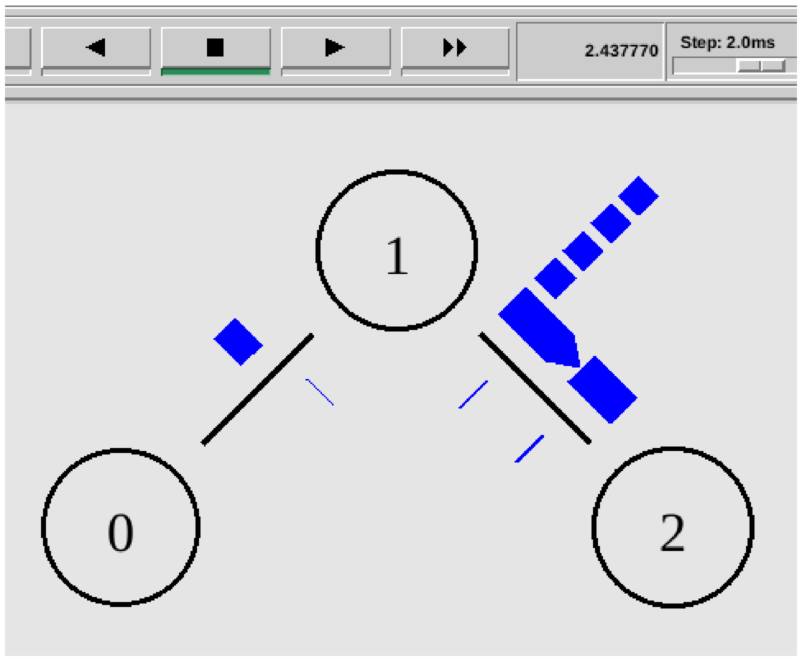
c. **Packets start queueing at the bottleneck (node-1) and packets are being dropped due to congestion in the pipeline. Duplicate acknowledgements start reaching the source (node-0)**



d. **Pipeline is not completely emptied. Reno agent follows a *Fast Recovery* mechanism. When duplicate packets are received, it does NOT reset CWND=1. Instead, it waits for enough duplicate ACKs and starts retransmission of lost packets, one for every duplicate-ACK received here on, until no more duplicate ACKs are received and the connection is back to normal. Hence, it quickly recovers from a congestion when compared to the Tahoe agent**



e. **Connection starts and proceeds at a much faster rate during retransmission, than Tahoe**



**Result**

Analyzed the difference between the congestion control mechanism offered by Tahoe and Reno TCP agents using the Network Simulator (NS2) tool and  using a simple tcl-script generated network schedule consisting of FTP traffic using TCP packets. Through this implementation, the following aspects were understood:

1. Basic scripting for NS and NAM using TCL language
2. Difference between the Tahoe and Reno mechanisms in TCP
3. Working with NS2 and NAM to observe network traffic conceptually