Department of Computer Science and Engineering

S.G.Shivanirudh , 185001146, Semester V

14 October 2020

UCS1511 - Networks Laboratory

Exercise 8: Performance Evaluation of TCP and UDP

Objective:

Write ns2 program to do Performance Evaluation of **TCP and UDP sharing a bottleneck link.**

Simulation Code:

```
#1. Create Simulator object
set ns [new Simulator]

#2. Define different colors for data flows (for NAM)
$ns color 1 blue
$ns color 2 red
```

```
#6. 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]

#3. Open the Trace files
set f [open out.tr w]
$ns trace-all $f

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

#7. Create links between the nodes
$ns duplex-link $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns simplex-link $n2 $n3 0.3Mb 100ms DropTail
$ns simplex-link $n3 $n2 0.3Mb 100ms DropTail
$ns duplex-link $n3 $n4 0.5Mb 40ms DropTail
$ns duplex-link $n3 $n5 0.5Mb 40ms DropTail

#8. Align it properly
$ns duplex-link-op $n0 $n2 orient right-up
$ns duplex-link-op $n1 $n2 orient right-down
$ns simplex-link-op $n2 $n3 orient right
$ns simplex-link-op $n3 $n2 orient left
$ns duplex-link-op $n3 $n4 orient right-up
$ns duplex-link-op $n3 $n5 orient right-down

#9. Set Queue Size of link (n2-n3) to 10 (or) 5
$ns queue-limit $n2 $n3 10

#10. Setup a TCP connection over 0 and 4 and its flow id, window size,
set tcp [new Agent/TCP/Newreno]
$ns attach-agent $n0 $tcp
```

```
set sink [new Agent/TCPSink/DelAck]
$ns attach-agent $n4 $sink
$ns connect $tcp $sink
$tcp set fid_ 1
$tcp set window_ 8000
$tcp set packetSize_ 512
$tcp set class_ 2

#11. Setup a FTP over TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp

#12. Setup a UDP connection over 1 and 5. Set the flow id
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
$udp set class_ 1

#13. Setup a CBR over UDP connection with type, packet size, rate, rando
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set type_ CBR
$cbr set packet_size_ 1024
$cbr set rate_ 0.01mb
$cbr set random_ false

set null [new Agent/Null]
$ns attach-agent $n5 $null
$ns connect $udp $null

#14. Start and stop the cbr and ftp accordingly
$ns at 0.1 "$cbr start"
$ns at 0.5 "$ftp start"
$ns at 4.7 "$ftp start"
$ns at 5.0 "$cbr stop"

#15. Finish the simulation
$ns at 7.0 "finish"
```

```
puts "Packet size: [$cbr set packetSize_]"
puts "Interval: [$cbr set interval_]"

#5. Define a 'finish' procedure
proc finish {} {
        global ns f nf
        $ns flush-trace
        close $f
        close $nf

        puts "running nam..."
        exec nam out.nam &
        exit 0
}

$ns run
```

Inferences:

- The following specifications defined the connections in the TCL script.
  - $0 \longrightarrow 2$ has a duplex link with a bandwidth of 2Mb and 10ms transmission delay.
  - $1 \longrightarrow 2$ has a duplex link with a bandwidth of 2Mb and 10ms transmission delay.
  - $2 \longrightarrow 3$ has a simplex link with a bandwidth of 0.3Mb and 100ms transmission delay.
  - $3 \longrightarrow 2$ has a simplex link with a bandwidth of 0.3Mb and 100ms transmission delay.
  - $3 \longrightarrow 3$ has a duplex link with a bandwidth of 0.5Mb and 40ms transmission delay.
  - $3 \longrightarrow 5$ has a duplex link with a bandwidth of 0.5Mb and 40ms transmission delay.
- A FTP and CBR applications used the established TCP over 1 and 4, and UDP over 0 and 5 respectively.
- Starting from a lossless packet transfer, a build up of packets in the queue of the bottleneck link $2 \longrightarrow 3$ occured.
- Due to being overloaded, the **DropTail** protocol started taking effect around 3.36 seconds, when the first packet is dropped from 2, and continued till approximately 3.84 seconds. This process completely stopped once the load eased up from the termination of the FTP application at 4.7 seconds.
- Any bottleneck in a network affect the efficiency of the entire network, leading to loss of data.
- Solutions to this problem include:
  - Adaptive transmission rates.
  - Adjusting queue size.
  - Increase bandwidth and/or reduce transmission delay of bottleneck link.

Output:

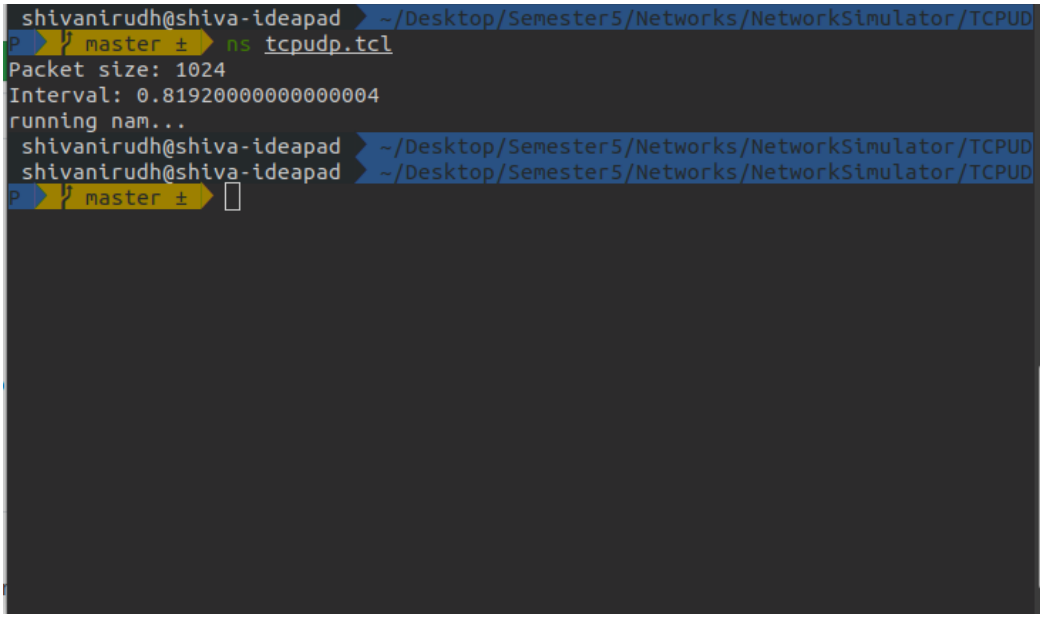

Figure 1: **CBR Packet Size:** 1024, **CBR Interval**: 0.8192s

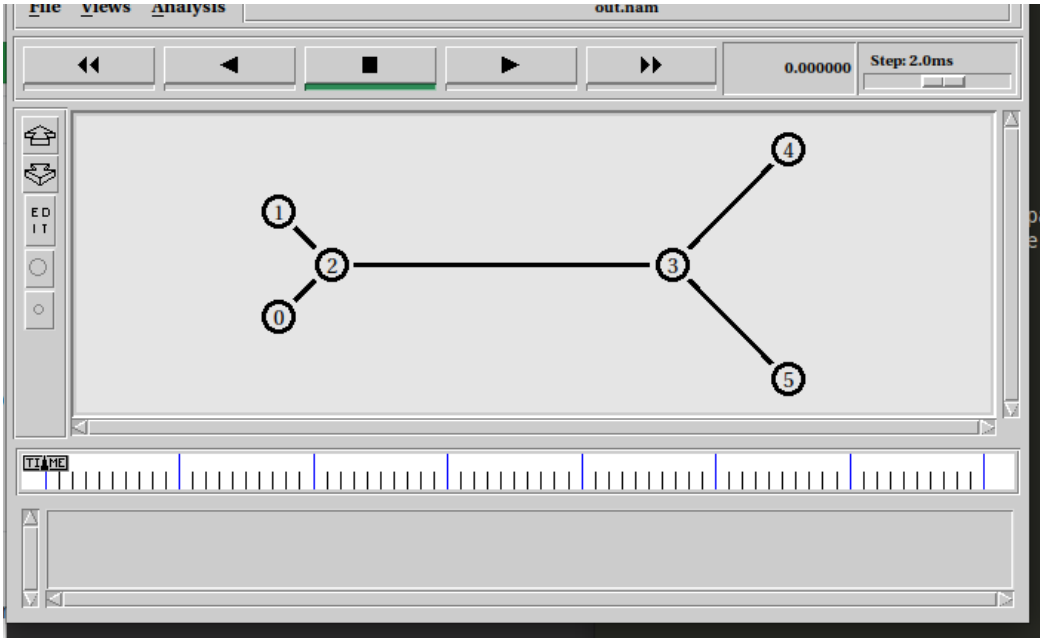

Figure 2: **Node Structure**

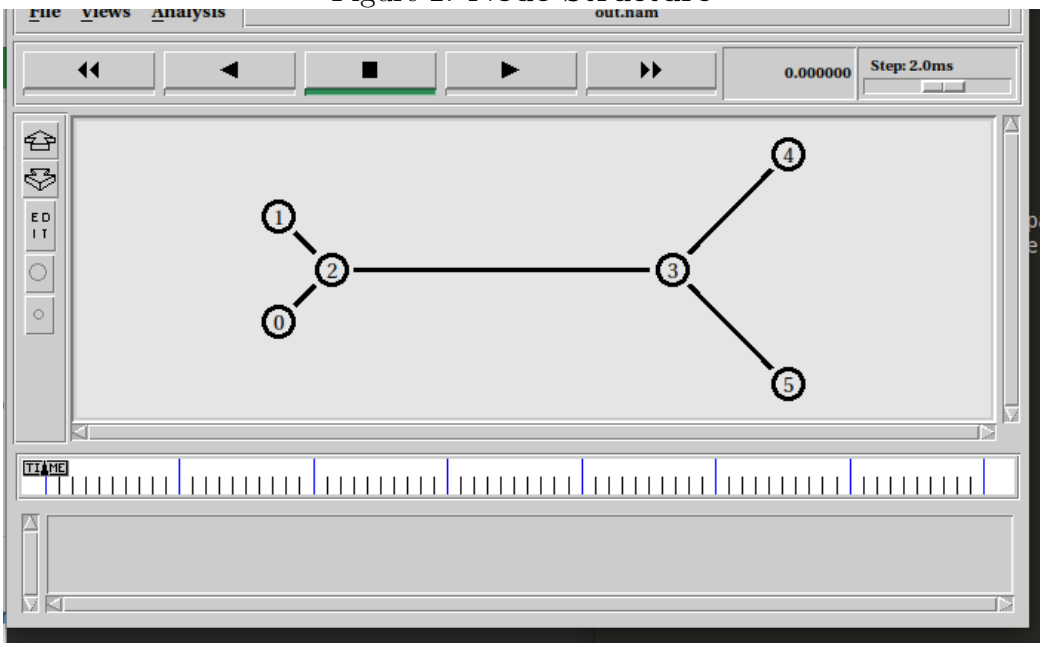

Figure 3: **N1 to N5 UDP connection CBR**

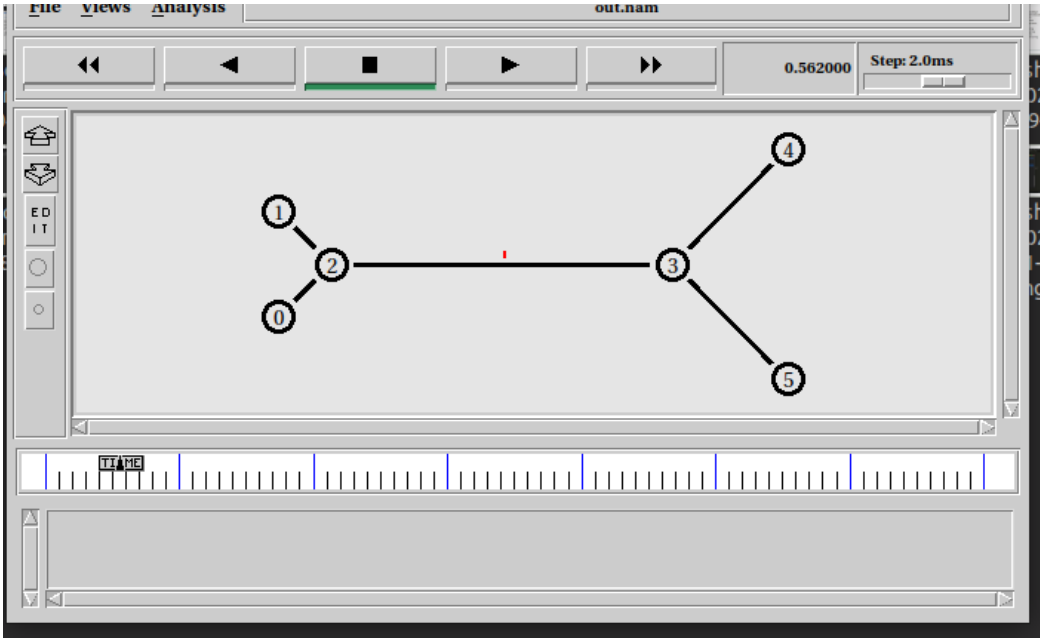

Figure 4: **N1 to N4 TCP connection acknowledgement**

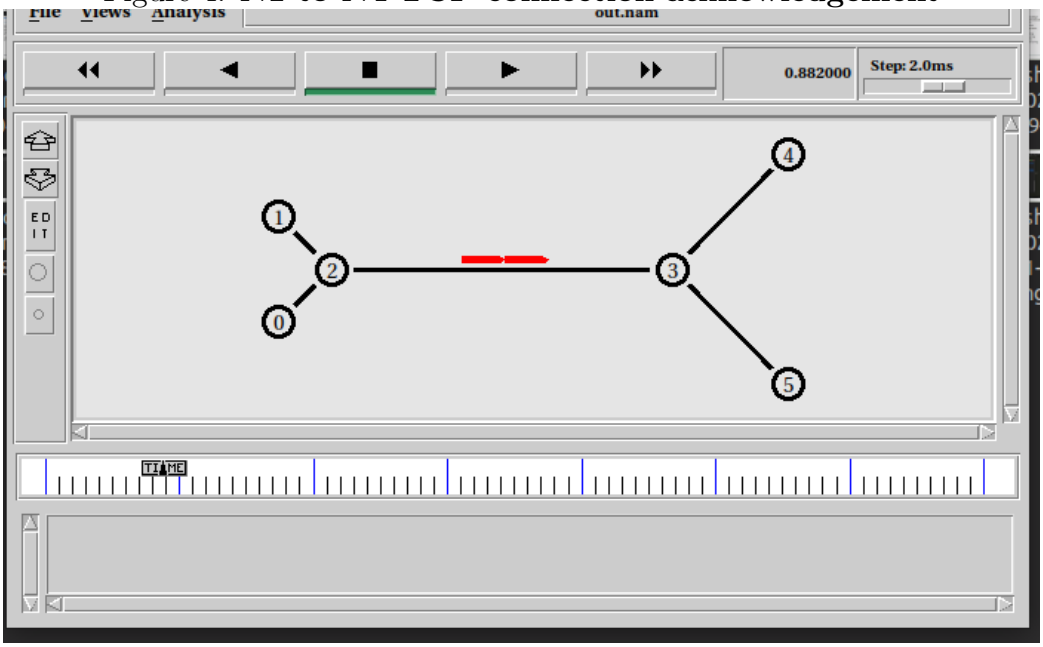

Figure 5: **N1 to N4 TCP connection packet transmission**

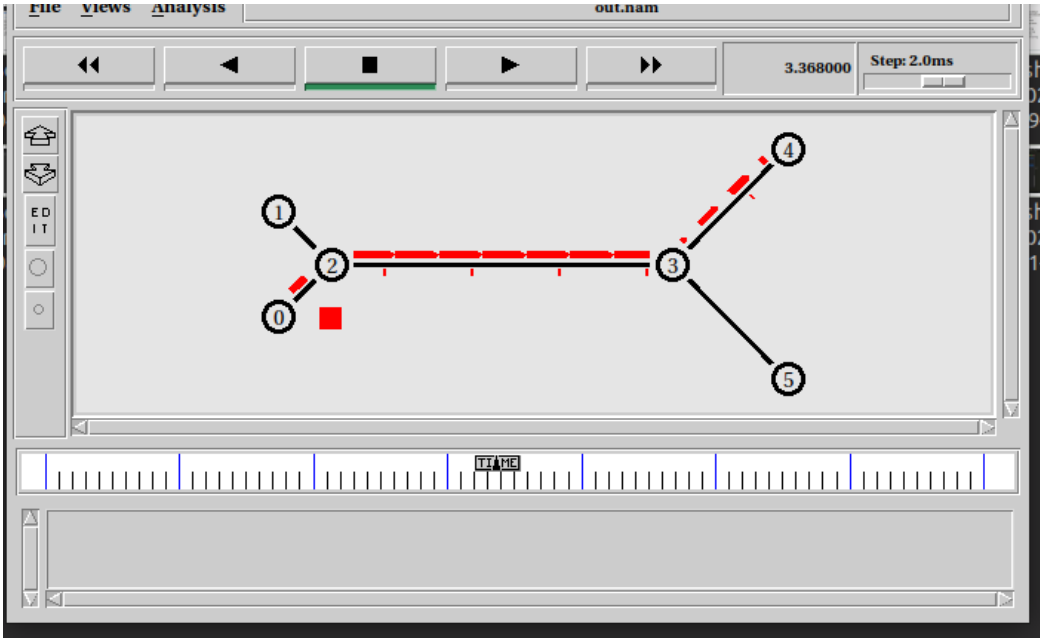

Figure 6: **First packet drop**

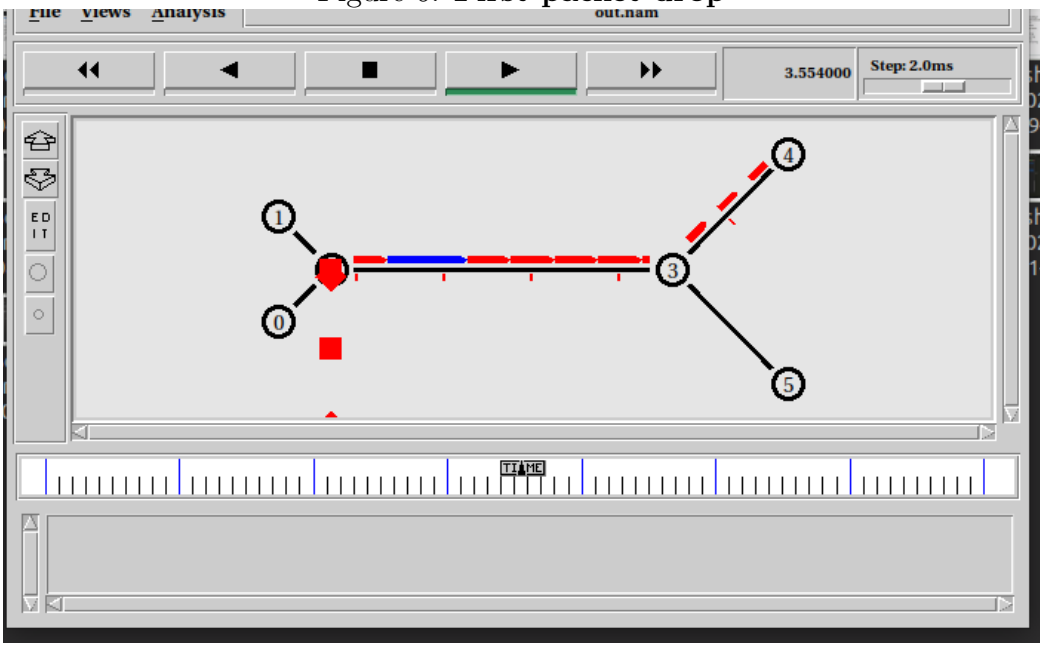

Figure 7: **Rest of the packets drop**