SSN COLLEGE OF ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
UCS1511_NETWORKS LABORATORY

NAME: MOHAMED UVAIS MN
REG.NO: 205001064
EX.NO:11
DATE:28.11.2022

EXERCISE 11: PERFORMANCE EVALUATION OF TCP & UDP

AIM:

To Write tcl script to do evaluate the performance of TCP and UDP sharing a bottleneck link

ALGORITHM :

1. Create six nodes and the links between the nodes as
   a. 0 → 2 2Mb 10 ms duplex link
   b. 1→2 2Mb 10 ms duplex link
   c. 2→3 0.3Mb 100ms simplex link
   d. 3 →2 0.3Mb 100ms simplex link (link 2→3 is a bottleneck)
   e. 3→4 0.5Mb 40ms duplex link
   f. 3→ 5 0.5Mb 40ms duplex link

2. Align the nodes properly.

3. Set Queue Size of link (n2-n3) to 10 (or) 5.

4. Setup a TCP connection over 0 and 4 and its flow id, window size, packet size

5. Set Up a UDP connection over 1 and 5 with flow id, type, packet size, rate, random fields.

6. Set different colors for TCP and UDP.

7. Run the simulation for 5 seconds, and show the simulation in network animator and in trace file.

Analyze the performance of TCP and UDP from the simulation.

CODE:

TCL FILE:

```
set ns [new Simulator]
$ns color 1 Blue
$ns color 2 Red
set nf [open out.nam w]
$ns namtrace-all $nf
proc finish {} {
 global ns nf
$ns flush-trace
close $nf
 exec nam out.nam &
exit 0
}
set n(0) [$ns node]
set n(1) [$ns node]
set n(2) [$ns node]
set n(3) [$ns node]
set n(4) [$ns node]
set n(5) [$ns node]
$ns duplex-link $n(0) $n(2) 2Mb 10ms
 DropTail
```

```
$ns duplex-link $n(1) $n(2) 2Mb 10ms
DropTail
$ns simplex-link $n(2) $n(3) 0.3Mb 100ms
DropTail
$ns simplex-link $n(3) $n(2) 0.3Mb 100ms
DropTail
$ns duplex-link $n(3) $n(4) 0.5Mb 40ms
DropTail
$ns duplex-link $n(3) $n(5) 0.5Mb 40ms
DropTail
$ns queue-limit $n(2) $n(3) 10
$ns duplex-link-op $n(0) $n(2) orient right
$ns duplex-link-op $n(1) $n(2) orient down
$ns simplex-link-op $n(2) $n(3) orient right
$ns simplex-link-op $n(3) $n(2) orient left
$ns duplex-link-op $n(3) $n(4) orient down
$ns duplex-link-op $n(3) $n(5) orient right
set tcp [new Agent/TCP]
$tcp set packetSize_ 1000
$ns attach-agent $n(0) $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n(4) $sink
$ns connect $tcp $sink
$tcp set fid_ 1
set udp [new Agent/UDP]
$ns attach-agent $n(1) $udp
set null [new Agent/Null]
$ns attach-agent $n(5) $null
$ns connect $udp $null
$udp set fid_ 2
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp
```

```
$ftp1 set type_ FTP
$tcp set packet_size_ 1000
$ftp1 set rate_ 1mb
$ftp1 set random_ false
set cbr2 [new
Application/Traffic/CBR] $cbr2
attach-agent $udp
$cbr2 set type_ CBR
$cbr2 set packet_size_ 1000
$cbr2 set rate_ 1mb
$cbr2 set random_ false
$ns at 0.0 "$ftp1 start"
$ns at 0.0 "$cbr2 start"
$ns at 5.0 "$ftp1 stop"
$ns at 5.0 "$cbr2 stop"
$ns at 4.9 "$ns detach-agent $n(0) $tcp ;
$ns detach-agent $n(4) $sink ; $ns
detach-agent
$n(1) $udp ; $ns detach-agent $n(5) $null"
$ns at 5.0 "finish"
$ns run
```

AWK FILE FOR UDP:

```
BEGIN {
 recvdSize = 0
 transSize = 0
 startTime = 400
 stopTime = 0
}
{
 event = $1
 time = $3
```

```
 send_id = $5
 rec_id = $7
 pkt_size = $11
 flow_id = $17
 type=$9
 # Store start time
 if (send_id == "1") {
 if (time < startTime) {
 startTime = time
 }
 if (event == "+") {
 # Store transmitted packet's size
 #transSize += pkt_size
 transSize+=1
 }
 }
 # Update total received packets' size and
store packets arrival time
 if (event == "r" && rec_id == "5") {
 if (time > stopTime) {
 stopTime = time
 }
 # Store received packet's size
 if (flow_id == "2") {
 #recvdSize += pkt_size
 recvdSize+=1
 }
 }
}
 END {
  printf("UDP throughput: %.2f
packets/sec\n",recvdSize/stopTime)
```

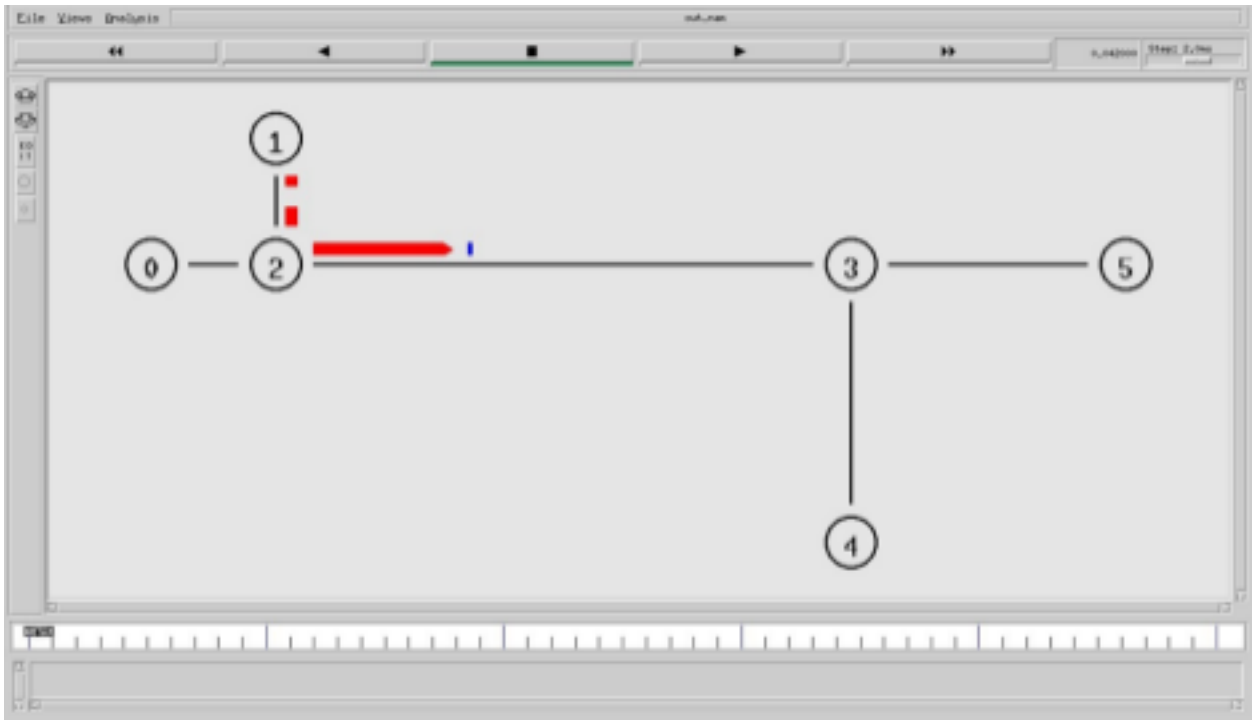
```
#printf("%i\t%i\t%.2f\t%.2f\t%.2f\n",
transSize, recvdSize, startTime,
stopTime,recvdSize/stopTime) }
```

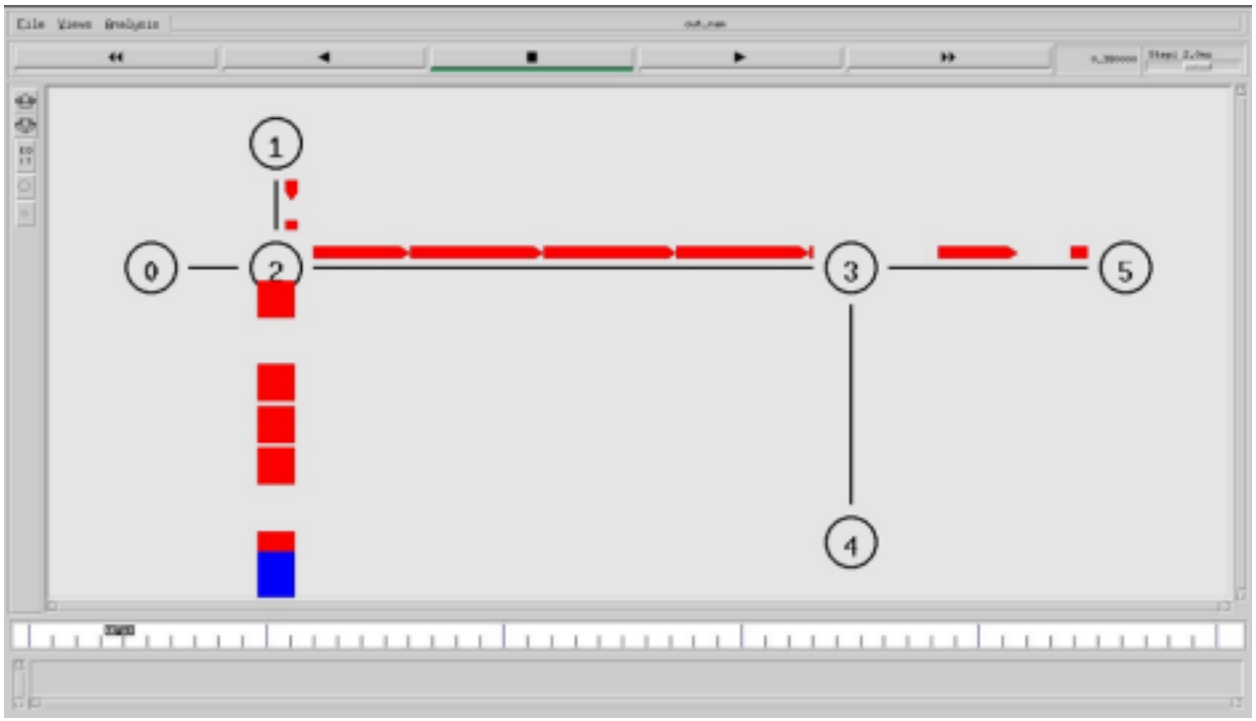
AWK FILE FOR TCP:

```
BEGIN {
 recvdSize = 0
 transSize = 0
 startTime = 400
 stopTime = 0
}
{
 event = $1
 time = $3
 send_id = $5
 rec_id = $7
 pkt_size = $11
 flow_id = $17
 type=$9
 # Store start time
 if (send_id == "0") {
 if (time < startTime) {
 startTime = time
 }
 if (event == "+") {
 # Store transmitted packet's size
 #transSize += pkt_size
 transSize+=1
 }
 }
 # Update total received packets' size and
store packets arrival time
```

```
 if (event == "r" && rec_id == "4") {
 if (time > stopTime) {
 stopTime = time
 }
 # Store received packet's size
 if (flow_id == "1") {
 #recvdSize += pkt_size
 recvdSize+=1
 }
 }
}
 END {
  printf("TCP throughput: %.2f
packets/sec\n",recvdSize/stopTime)
#printf("%i\t%i\t%.2f\t%.2f\t%.2f\n",
transSize, recvdSize, startTime,
stopTime,recvdSize/stopTime)
}
```

OUTPUT:






```
ssn@ssn-c16:~/Downloads$ ns A11.tcl
ssn@ssn-c16:~/Downloads$ awk -f A11TCP.awk out.nam
Percentage of packets lost: 80.00 percent
TCP throughput: 6.58 packets/sec
ssn@ssn-c16:~/Downloads$ awk -f A11UDP.awk out.nam
Percentage of packets lost: 70.47 percent
UDP throughput: 36.22 packets/sec
```

LEARNING OUTCOMES :

Learned to simulate and evaluate the performance of TCP and UDP sharing a bottleneck link has been written and executed.