Jaypee Institute of Information Technology, Noida

Computer Networks Lab Project

Report

Comparison of network attributes based on network traffic and queuing policies.

Submitted To:

Mr. Gaurav Nigam

Submitted by:

Shriyam Verma (9916103188)

Vishal Chaudhary (9916103154)

Anant Patni (9916103180)

Objective

In this project, we are analysing throughput, packets dropped as well duration and total transmitted bits of different Queuing policies .

Description

In the course of time many network queuing disciplines have been developed. Each of these provides specific reordering or dropping of network packets inside various transmit or receive buffers. Queuing disciplines are commonly used as attempts to compensate for various networking conditions, like reducing the latency for certain classes of network packets, and are generally used as part of quality of services (QoS) measures. We have used the following queuing policies:

DropTail:

It is a simple queue mechanism that is used by the routers that when packets should to be drop. In this mechanism each packet is treated identically and when queue filled to its maximum capacity the newly incoming packets are dropped until queue have sufficient space to accept incoming traffic.

Problem:

When a queue is filled the router start to discard all extra packets thus dropping the tail of mechanism. The loss of packets (datagram's) causes the sender to enter slow start which decreases the throughput and thus increases its congestion window.

Random Early Detection (RED):

RED is a congestion avoidance queuing mechanism (as opposed to a congestion administration mechanism) that is potentially useful, particularly in high-speed transit networks. Sally Floyd and Van Jacobson projected it in various papers in the early 1990s.It is active queue management mechanism. It operates on the average queue size and drop packets on the basis of statistics information. If the buffer is empty all incoming packets are acknowledged. As the queue size increase the probability for discarding a packet also increase. When buffer is full probability becomes equal to 1 and all incoming packets are dropped.

Pros:

RED is capable to evade global synchronization of TCP flows, preserve high throughput as well as a low delay and attains fairness over multiple TCP connections, etc. It is the most common mechanism to stop congestive collapses.

Cons:

When the queue in the router starts to fill then a small percentage of packets are discarded. This is deliberate to start TCP sources to decrease their window sizes and hence suffocate back the data rate. This can cause low rates of packet loss in Voice over

IP streams. There have been reported incidences in which a series of routers apply RED at the same time, resulting in bursts of packet loss.

Fair Queuing (FQ):

It is a queuing mechanism that is used to allow multiple packets flow to comparatively share the link capacity. Routers have multiple queues for each output line for every user. When a line as available as idle routers scans the queues through round robin and takes first packet to next queue. FQ also ensure about the maximum throughput of the network. For more efficiency weighted queue mechanism is also used.

Stochastic Fair Queuing (SFQ):

This queuing mechanism is based on fair queuing algorithm and proposed by John Nagle in 1987. Because it is impractical to have one queue for each conversation SFQ uses a hashing algorithm which divides the traffic over a limited number of queues. It is not so efficient than other queues mechanisms but it also requires less calculation while being almost perfectly fair. It is called "Stochastic" due to the reason that it does not actually assign a queue for every session; it has an algorithm which divides traffic over a restricted number of queues using a hashing algorithm. SFQ assigns a pretty large number of FIFO queues.

Code

```
set ns [new Simulator]

$ns color 1 Blue
$ns color 2 Red
$ns color 3 Green

set nf [open task1.nam w]
$ns namtrace-all $nf
set tf [open task1.tr w]
$ns trace-all $tf

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]

$ns duplex-link $n0 $n2 1Mb 10ms DropTail #Replace DropTail with RED/SFQ and FQ
$ns duplex-link $n1 $n2 1Mb 10ms DropTail #Replace DropTail with RED/SFQ and FQ
```

```
#link L3
$ns duplex-link $n2 $n3 1Mb 10ms DropTail #Replace DropTail with RED/SFQ and FQ
$ns duplex-link $n3 $n5 1Mb 10ms DropTail #Replace DropTail with RED/SFQ and FQ
$ns duplex-link $n3 $n4 1Mb 10ms DropTail #Replace DropTail with RED/SFQ and FQ


#Setting up 2 TCP connections

set tcp_1 [new Agent/TCP]
$ns attach-agent $n0 $tcp_1
$tcp_1 set fid_ 1

set sink_1 [new Agent/TCPSink]
$ns attach-agent $n5 $sink_1
$ns connect $tcp_1 $sink_1

set tcp_2 [new Agent/TCP]
$ns attach-agent $n1 $tcp_2
$tcp_2 set fid_ 2

set sink_2 [new Agent/TCPSink]
$ns attach-agent $n5 $sink_2
$ns connect $tcp_2 $sink_2

# Setting up 2 ftp applications

set ftp_1 [new Application/FTP]
$ftp_1 attach-agent $tcp_1

set ftp_2 [new Application/FTP]
$ftp_2 attach-agent $tcp_2

#Connection 3 UDP

set udp [new Agent/UDP]
$ns attach-agent $n4 $udp
set null [new Agent/Null]
$ns attach-agent $n5 $null
$ns connect $udp $null
$udp set fid_ 3

set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set type_ CBR
```

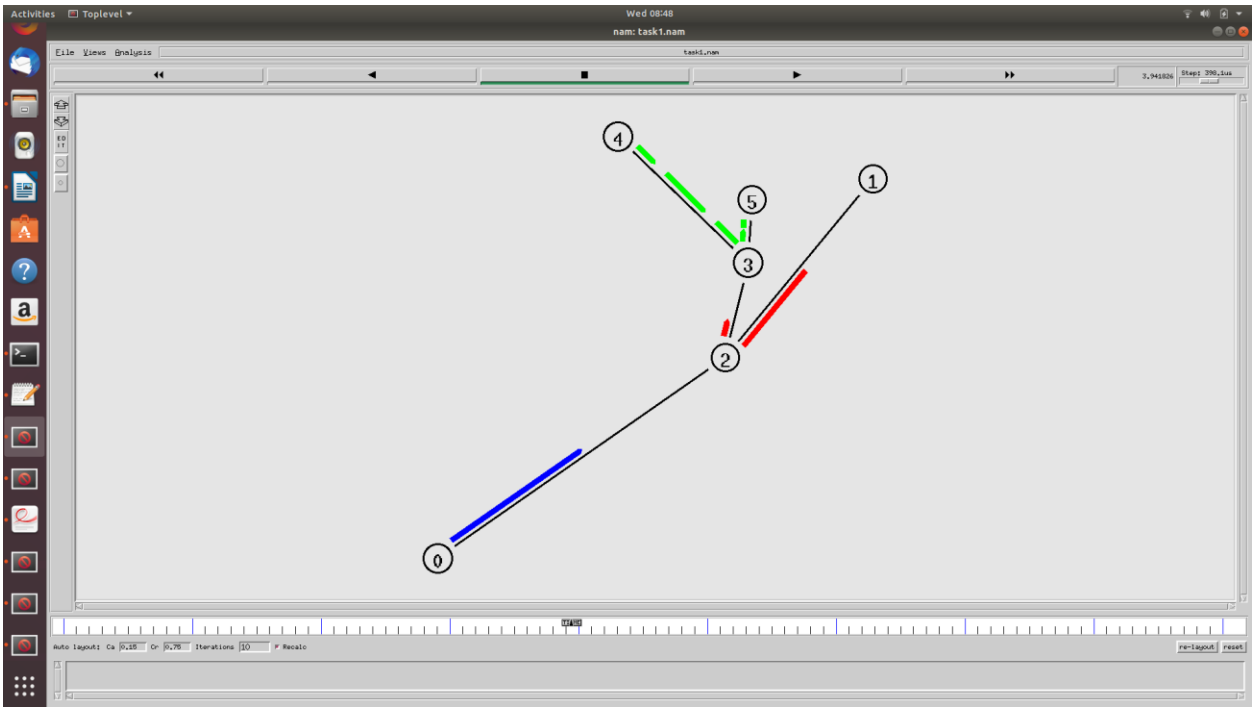
```
$cbr set packetSize_ 500
$cbr set interval_ 0.005

proc finish {} {
  global ns nf tf
  $ns flush-trace
  close $nf
  close $tf
  exec nam task1.nam &
  exit 0
}

$ns at 1 "$ftp_1 start"
$ns at 1 "$ftp_2 start"
$ns at 2 "$cbr start"
$ns at 8 "$ftp_1 stop"
$ns at 8 "$ftp_2 stop"
$ns at 9 "$cbr stop"
$ns at 10 "finish"
$ns run
```

AWK File –

```
BEGIN {

fromNode=1; toNode=2;
count = 0;
lineCount = 0;totalBits = 0;

}

/^r/&&$3==fromNode&&$4==toNode {

    totalBits += 8*$6;

if ( lineCount==0 ) {

timeBegin = $2; lineCount++;

} else {

timeEnd = $2;

};

};
```

```
{
if($1 == "d")
        count++;
}


END{

duration = timeEnd-timeBegin;

print "Number of records is " NR;

print "Output: ";

print " Transmission: N" fromNode "->N" toNode;

print "  - Total transmitted bits = " totalBits " bits";

print "  - duration = " duration " s";

print "  - Thoughput = "  totalBits/duration/1000 " kbps.";

printf("No.of dropped packets= %d \n", count);

};
```

SNAPSHOT



RESULT

Queuing Policy	Total Transmitted Bits	Duration (sec)	Throughput (kbps)	No. of packets dropped
DropTail	1331520	6.70208	198.673	87
RED	1406400	7.13896	197.003	155
FQ	2554560	6.99184	365.363	701
SFQ	2720960	6.778	401.44	841