Example 1

**Remote Laboratory Interface (RLI)**

**Traffic**

- Options: 
- Parameter: 
- View: 

**Queue**

- Options: 
- Parameter: 
- View: 

### NPR.1

- 2 Mbps
- 0.683 Mbps
- 99.718 Mbps

### NPR.2

- 4 Mbps
- 0.8 Mbps

**UDP traffic**
Example 1 (Modified)

- **Traffic**
  - Max size pkts = 1500 bytes
  - n2p4 starts 10 sec after n2p1
  - n2p1: 2 Mbps = 167 pps
  - n2p4: 0.8 Mbps = 67 pps

- **Port Capacities**
  - port 2/2:
  - port 2/3:
  - port 1/1:

- **Queueing Rates**
  - port 2/2:
  - port 2/3:
  - port 1/1:

- **Q300**
  - Capacity: 100,000B
  - Overflows in sec

Example 1 (Modified)

- **Traffic**
  - Max size pkts = 1500 bytes
  - n2p4 starts 10 sec after n2p1
  - n2p1: 2 Mbps = 167 pps
  - n2p4: 0.8 Mbps = 67 pps

- **Port Capacities**
  - port 2/2: 1 Mbps = 84 pps
  - port 2/3: 10 Mbps = 840 pps
  - port 1/1: 1.5 Mbps = 126 pps

- **Queueing Rates**
  - port 2/2: 2-1 = 1 Mbps
  - port 2/3: 0 Mbps
  - port 1/1: (1+0.8)-1.5 = 0.3 Mbps
    - 0.3 Mbps = 37,500 Bps = 25 pps

- **Q300**
  - Capacity: 100,000B = 67 pkts
  - Overflows in 67/25 = 2.7 sec
Example 2 (Back-To-Back Traffic)

Different Traffic
- back-to-back UDP pkts from n2p1 and n2p4
- both flows start at the same time
- n2p1: 100 max-size pkts (1500 bytes each)
- n2p4: 100 max-size pkts (1500 bytes each)

What queues must be changed so that no pkt drops occur?

Example 2

Transmission Volumes
- bytes from each source
- pragmatics: ONL UDP pkts
  - 8-byte UDP hdr, 20-byte IP hdr
  - 1470-byte UDP payload
  - actual pkt size = 1498 bytes

Different Traffic
- back-to-back UDP pkts from n2p1 and n2p4
- both flows start at the same time
- n2p1: 100 max-size pkts (1500 bytes each)
- n2p4: 100 max-size pkts (1500 bytes each)

What queues must be changed so that no pkt drops occur?

File transfer time
Example 2

Transmission Volumes
- bytes from each source
  - 150,000 bytes = 100x1500
- pragmatics: ONL UDP pkts
  - 8-byte UDP hdr, 20-byte IP hdr
  - 1470-byte UDP payload
  - actual pkt size = 1498 bytes

Q300 (NPR.1)
- Capacity (for 2 flows)
  - 300,000 bytes = 200 pkts

Port 2/1, Datagram Queue
- Capacity (for 1 flow)
  - 150,000 bytes = 100 pkts

File transfer time
- 1.6 sec = (1.2+1.2)/1.5

Different Traffic
- back-to-back UDP pkts from n2p1 and n2p4
- both flows start at the same time
  - n2p1: 100 max-size pkts (1500 bytes each)
  - n2p4: 100 max-size pkts (1500 bytes each)

What queues must be changed so that no pkt drops occur?

Example 3 (Forward Delay)

Effect of 50 msec delay along forward path?
- arrival of first pkt to n1p1
  - finish time

Interpacket Times
- port 1/4:
- port 1/3:
  - n1p1:

Bandwidth-Delay Products
- n1p4-n1p1:
- n1p3-n1p1:
Example 3 (Forward Delay)

- Effect of 50 msec delay along forward path?
  - Arrival of first pkt to n1p1
  - Delayed by 50 msec
  - Finish time
  - 50 msec longer than without delay

- Interpacket Times
  - Port 1/4: $1/(84 \text{ pps}) = 11.9 \text{ msec}$
  - Port 1/3: $1/(8400 \text{ pps}) = 0.119 \text{ msec}$
  - N1p1: $1/(126 \text{ pps}) = 7.93 \text{ msec}$

- Bandwidth-Delay Products
  - N1p4-n1p1: $1 \text{ Mbps} \times 50 \text{ msec} = 50 \text{ Kb} = 33.3 \text{ pkts}$
  - N1p3-n1p1: $10 \text{ Mbps} \times 50 \text{ msec} = 500 \text{ Kb} = 333 \text{ pkts}$

Example 4 (ACK Delay)

- Suppose
  - Small ACK pkt returned along reverse path for each data pkt

  $T1 = L/(1 \text{ Gbps})$
  $T2 = L/1 \text{ Mbps}$
  $T3 = T2$
  $T4 = T2$
ONL Issues

- Port rate
  - Integer multiple of 0.683 Mbps
    - e.g., 12 Mbps ➝ 11.611
- Receiver socket buffer size
  - Default: 84 KB
  - Avoid ignoring incoming pkts for too long
  - Use setsockopt() to increase buffer size

```c
n = 500 * 1500; // 500 max-size pkts
rc = setsockopt(sockfd, SOL_SOCKET, SO_RCVBUF, &n, sizeof(n));
```

- C/C++ code examples (onlusr:~kenw/src/)
  - pkt-pair/
  - xstcp/

The Meaning of Port Rate
Queue Table ➔ Port Rate

- Controlled by a token bucket model
  - R: long-term average link rate (RLI parameter)
    - 54 Kbps granularity
  - b: maximum bucket depth (4000 bytes)
  - R': physical link rate (1 Gbps)
- Effect
  - avg output rate is R
  - peak rate is R'
- Operational definition
  - fill token bucket at rate R
  - transmit pkt when #tokens ≥ length of pkt at head of queue
- **Link regulator** model, NOT link emulator

The Packet-Pair Problem

- Keshav’s idea
  - Rcvr receives the two pkts at times t0 and t1
  - What will the **interpacket time** t1-t0 be?
  - average interbit time?
ONL Packet-Pair Experiment (1)

Ideal:

\[
T = \frac{L}{1 \text{ Gbps}}
\]

\[
T = \frac{L}{12 \text{ Mbps}}
\]

Note initial burst

ONL:

\[
T = \frac{L}{1.1611 \text{ Gbps}}
\]

ONL Packet-Pair Experiment (2)

Puts $n1p2$, etc. into Linux bash environment

rcvr: n2p1

# pkts: 7

20-byte IP hdr

8-byte UDP hdr

1470-byte payload