Real-Time Guarantees

- Requirements on RT communication protocols:
  - delay (response times) small
  - jitter small
  - throughput high
  - error detection at receiver (and sender)
  - small error detection latency
  - no thrashing under peak load
  - limited/no packet loss

- Characteristics of traffic:
  - messages - packets (cells, frames, ...)
  - packet inter-arrival times
  - Quality-of-Service (QoS) parameters
  - integration of real-time and non-real-time traffic
  - buffer requirements
  - header overheads and per-packet processing overheads

Traffic Characteristics

- Traffic characteristics of real-time sources
  - constant bit rate: av traffic control
  - variable bit rate: voice traffic
  - periodic with variable packet size: video data

Flow Control

- Control of the information flow between sender and receiver, such that sender does not outpace receiver.
- Receiver should determine speed of communication.

- Implicit flow control:
  - a-priori established send rate (will not be exceeded)
  - communication can be uni-directional
  - error detection is responsibility of receiver
  - diffusion-based protocols rely on implicit flow control

- Explicit flow control:
  - sender sends message and waits for explicit acknowledgment
  - sender is “in the sphere of control of the receiver”, i.e., the receiver can slow down sender (“back pressure” flow control)
  - error detection is responsibility of sender
  - missing ACK: message (or ACK) has been lost or receiver is late or has failed
Switch Scheduling

- Work-conserving (greedy) vs. non-work-conserving (non-greedy) mechanisms.
- Rate-allocating disciplines: Allow packets to be served at higher rates than the guaranteed rate.
- Rate-controlled disciplines: Ensures each connection the guaranteed rate, but does not allow packets to be served above guaranteed rate.
- Priority-based scheduling:
  - fair queuing
  - earliest due date (Delay-EDD and Jitter-EDD)
- Weighted Round-Robin scheduling:
  - WRR

Weighted Fair Queueing

- Each flow i given a weight (importance) \( w_i \)
- WFQ guarantees a minimum service rate to flow i
  - \( r_i = R \cdot w_i / (w_1 + w_2 + \ldots + w_n) \)
  - Implies isolation among flows (one cannot mess up another)

Fluid Flow

- Water pipes
- Water buckets

\( w_1 \) \( w_2 \) \( w_3 \)
Bit-by-Bit Weighted Round Robin

- bit-by-bit round robin
- each connection is given a weight
- each queue served in FIFO order

Fluid Flow System: Example 1

<table>
<thead>
<tr>
<th></th>
<th>Packet In (bits)</th>
<th>Packet Inter-arrival time (ms)</th>
<th>Arrival Rate (Kbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flow 1</td>
<td>1000</td>
<td>10</td>
<td>100</td>
</tr>
<tr>
<td>Flow 2</td>
<td>500</td>
<td>10</td>
<td>50</td>
</tr>
</tbody>
</table>

Flow 2 (arrival traffic)

Flow 1 (arrival traffic)

Service in fluid flow system

Fluid Flow System: Example 2

- Red flow has packets backlogged between time 0 and 10
- Other flows have packets continuously backlogged
- All packets have the same size
“Real” System
• Packet transmission cannot be preempted.
• Solution: serve packets in the order in which they would have finished being transmitted in the fluid flow system

Example 2
• Select the first packet that finishes in the fluid flow system

Implementation Challenge
• Need to compute the finish time of a packet in the fluid flow system…
• … but the finish time may change as new packets arrive!
• Need to update the finish times of all packets that are in service in the fluid flow system when a new packet arrives
  – But this is very expensive: a high speed router may need to handle hundred of thousands of flows!
Example

- Four flows, each with weight 1

Virtual Time

- Key Observation: while the finish times of packets may change when a new packet arrives, the order in which packets finish doesn’t!
  - Only the order is important for scheduling
- Solution: instead of the packet finish time maintain the round # when a packet finishes (virtual finishing time)
  - Virtual finishing time doesn’t change when a packet arrives
- System virtual time V(t) – index of the round in the bit-by-bit round robin scheme

Virtual Time

- Suppose each packet is 1000 bits, so it takes 1000 rounds to finish
- So, packets of F1, F2, F3 finish at virtual time 1000
- When packet F4 arrives at virtual time 1 (after one round), the virtual finish time of packet F4 is 1001
- But the virtual finish time of packet F1,2,3 remains 1000
- Finishing order is preserved
**System Virtual Time (Round #): V(t)**

- $V(t)$ increases inversely proportionally to the sum of the weights of the backlogged flows.
- Since round # increases slower when there are more flows to visit each round.

![Diagram](image)  

**Scheduling with WFQ**

- When first packet arrives in empty queue, scheduler computes finish number and commences transmission.
- When scheduler busy, each new packet on idle connection receives a finish number which is inserted into SFN queue.
- When transmission of packet of connection $i$ completes, packet is removed from $i$ and entry containing finish number of this packet is removed from SFN queue. Next packet is chosen:
  - if $i$ is still backlogged, the scheduler computes the finish number of its new ready packet and inserts this number into SFN queue.
  - the head packet in the SFN queue is chosen.

**Fair Queueing Implementation**

- Define:
  - $V^k_i$: virtual finishing time of packet $k$ of flow $i$
  - $a^k_i$: arrival time of packet $k$ of flow $i$
  - $L^k_i$: length of packet $k$ of flow $i$
  - $w_i$: weight of flow $i$

- The finishing time of packet $k+1$ of flow $i$ is
  \[ F^{k+1} = \max(V^k_i + a^k_i, F^k_i) + L^k_i / w_i \]

- Smallest finishing time first scheduling policy