TRAFFIC MANAGEMENT

- The main objectives of traffic management are efficient use of network resources & deliver QoS.
- Traffic Management is classified into three levels that are Packet level, Flow level and Flow aggregated level.

Traffic Management at Packet Level

- Queueing & scheduling at switches, routers and multiplexers.

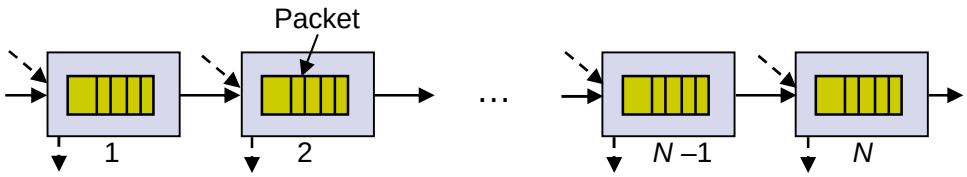

Figure: - End-to-End QoS of a packet along a path traversing N Queueing System

- The path traversed by packet through a network can be modeled as sequence of Queueing systems as shown in above figure.
- A packet traversing network encounters delay and possible loss at various multiplexing points.
- End-to-end performance is sum of the individual delays experienced at each system.
- Average end-to-end delay is the sum of the individual average delay.
- To meet the QoS requirements of multiple services, a queueing system must implement strategies for controlling the transmission bit rates.

The different strategies for Queue scheduling are:-

1. FIFO QUEUEING
2. PRIORITY QUEUEING
3. FAIR QUEUEING
4. WEIGHTED FAIR QUEUEING

1) FIFO QUEUEING

- Transmission Discipline: First-In, First-Out
- All packets are transmitted in order of their arrival.
- Buffering Discipline:- Discard arriving packets if buffer is full
- Cannot provide differential QoS to different packet flows
- Difficult to determine performance delivered
- Finite buffer determines a maximum possible delay
- Buffer size determines loss probability, but depends on arrival & packet length statistics.

FIFO Queueing with Discard Priority

FIFO queue management can be modified to provide different characteristics of packet-loss performance to different classes of traffic.

- The above Figure 7.42 (b) shows an example with two classes of traffic.
- When number of packets in a buffer reaches a certain threshold, arrivals of lower access priority (class 2) are not allowed into the system.
- Arrivals of higher access priority (class 1) are allowed as long as the buffer is not full.

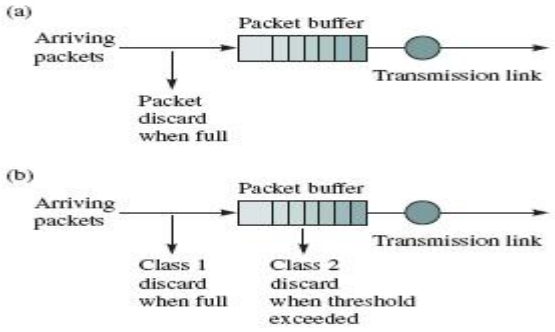

FIGURE 7.42 (a) FIFO queueing; (b) FIFO queueing with discard priority

2) Head of Line (HOL) Priority Queueing

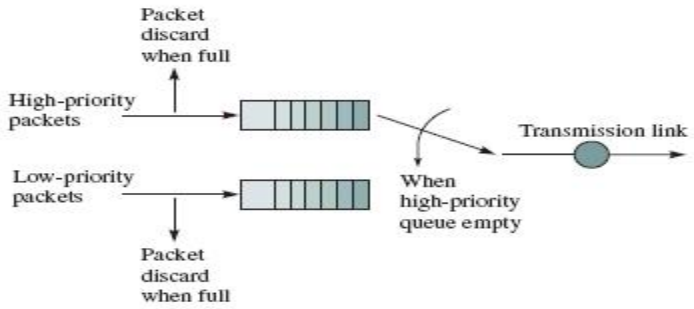

FIGURE 7.43 HOL priority queueing

- Second queue scheduling approach which defines number of priority classes.
- A separate buffer is maintained for each priority class.
- High priority queue serviced until empty and high priority queue has lower waiting time
- Buffers can be dimensioned for different loss probabilities
- Surge in high priority queue can cause low priority queue to starve for resources.
- It provides differential QoS.
- High-priority classes can hog all of the bandwidth & starve lower priority classes
- Need to provide some isolation between classes

Sorting packets according to priority tags/Earliest due Date Scheduling

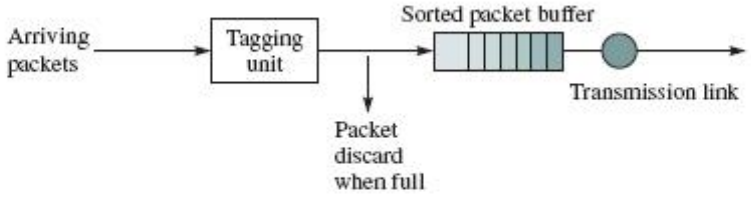

FIGURE 7.44 Sorting packets according to priority tag

- Third approach to queue scheduling
- Sorting packets according to priority tags which reflect the urgency of packet needs to be transmitted.
- Add Priority tag to packet, which consists of priority class followed by the arrival time of a packet.

- Sort the packet in queue according to tag and serve according to HOL priority system
- Queue in order of “due date”.
- The packets which requires low delay get earlier due date and packets without delay get indefinite or very long due dates

3) Fair Queueing / Generalized Processor Sharing

- Fair queueing provides equal access to transmission bandwidth.
- Each user flow has its own logical queue which prevents hogging and allows differential loss probabilities
- C bits/sec is allocated equally among non-empty queues.
- The transmission rate = C / n bits/second, where n is the total number of flows in the system and C is the transmission bandwidth.
- Fairness: It protects behaving sources from misbehaving sources.
- Aggregation:
    - Per-flow buffers protect flows from misbehaving flows
    - Full aggregation provides no protection
    - Aggregation into classes provided intermediate protection
- Drop priorities:
    - Drop packets from buffer according to priorities
    - Maximizes network utilization & application QoS
    - Examples: layered video, policing at network edge.

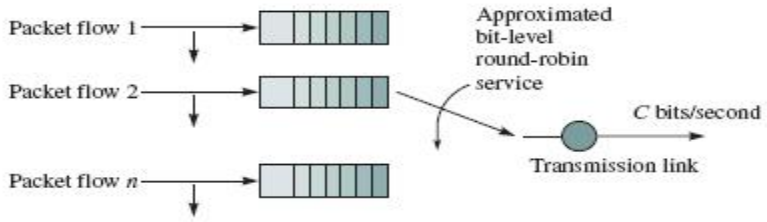

FIGURE 7.45 Fair queueing

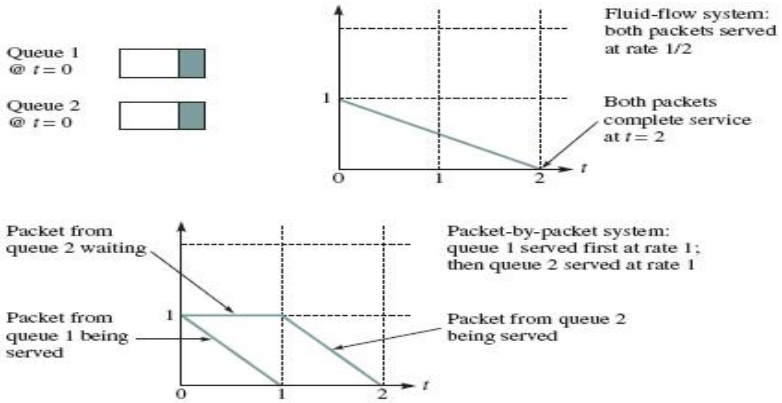

FIGURE 7.46 Fluid-flow and packet-by-packet fair queueing (two packets of equal length)

The above figure 7.46 illustrates the differences between ideal or fluid flow and packet-by-packet fair queueing for packets of equal length.

- Idealized system assumes fluid flow from queues, where the transmission bandwidth is divided equally among all non-empty buffers.
- The figure assumes buffer1 and buffer 2 has single L-bit packet to transmit at t=0 and no subsequent packet arrive.
- Assuming capacity of C=L bits/second=1 packet/second.
- Fluid-flow system transmits each packet at a rate of ½ and completes the transmission of both packets exactly at time=2 seconds.
- Packet-by-packet fair queueing system transmits the packet from buffer 1 first and then transmits from buffer 2, so the packet completion times are 1 and 2 seconds.

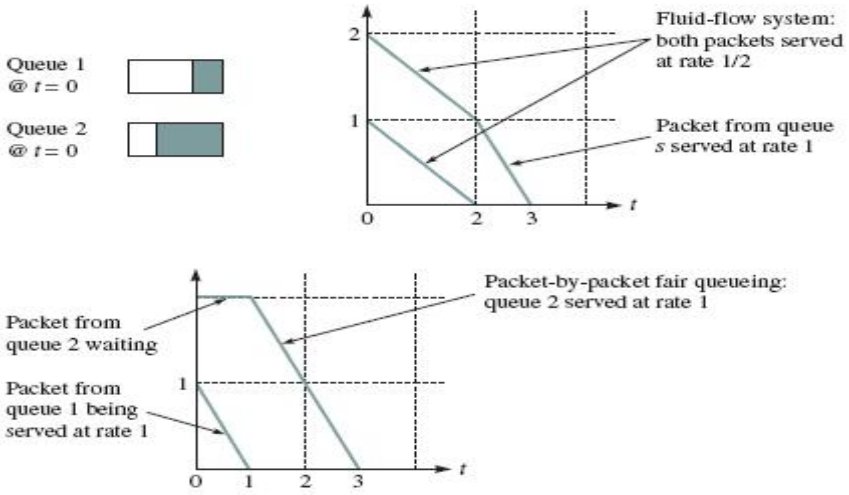

FIGURE 7.48 Fluid flow and packet-by-packet fair queueing (two packets of different lengths)

The above figure 7.48 illustrates the differences between ideal or fluid flow and packet-by-packet fair queueing for packets of variable length.

- The fluid flow fair queueing is not suitable, when packets have variable lengths.
- If the different user buffers are serviced one packet at a time in round-robin fashion, then we do not obtain fair allocation of transmission bandwidth.
- Finish tag is number used for the packet and the packet with smallest finish tag will be served first, and finish tag is computed as follows.
- Finish tag is used as priorities in packet-by-packet system.

Consider Bit-by-Bit Fair Queueing

- Assume n flows, n queues
- 1 round = 1 cycle serving all n queues
- If each queue gets 1 bit per cycle, then 1 round is the number of opportunities that each buffer has had to transmit a bit.
- Round number = number of cycles of service that have been completed

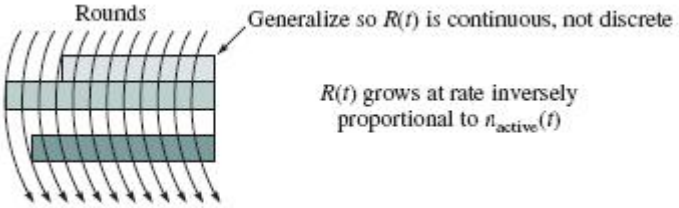

FIGURE 7.47 Computing the finishing time in packet-by-packet fair queueing and weighted fair queueing

- If packet arrives to idle queue:
  Finishing time = round number + packet size in bits
- If packet arrives to active queue:

Finishing time = finishing time of last packet in queue + packet size

Computing the Finishing Time

- F(i,k,t) = finish time of kth packet that arrives at time t to flow i
- P(i,k,t) = size of kth packet that arrives at time t to flow i
- R(t) = round number at time t

- Fair Queueing:

$$F(i,k,t) = \max\{F(i,k-1,t), R(t)\} + P(i,k,t)$$

4) Weighted Fair Queueing (WFQ)

- WFQ addresses the situation in which different users have different requirements.
- Each user flow has its own buffer and each user flow also has weight.
- Here weight determines its relative bandwidth share.
- If buffer 1 has weight 1 and buffer 2 has weight 3, then when both buffers are nonempty, buffer 1 will receive 1/(1+3)=1/4 of the bandwidth and buffer 2 will receive ¾ of the bandwidth.

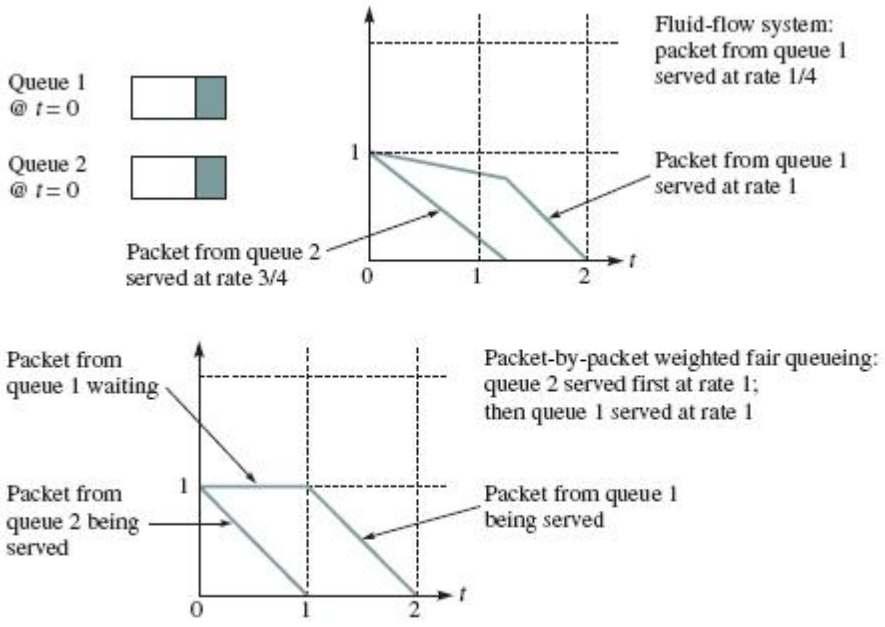

FIGURE 7.49 Fluid flow and packetized, weighted fair queueing

In the above figure,

- In Fluid-flow system, the transmission of each packet from buffer 2 is completed at time t=4/3, and the packet from buffer 1 is completed at t=2 seconds.
- In the above figure buffer1 would receive 1 bit/round and buffer 2 would receive 3 bits/second.
- Packet-by-packet weighted fair queueing calculates its finishing tag as follows

  $$F(i,k,t) = \max\{F(i,k-1,t), R(t)\} + P(i,k,t)/wi$$
- The above figure also shows the completion times for Packet-by-packet weighted fair queueing.
- The finish tag for buffer1 is $F(1,1)=R(0)+1/1 =1$ and finish tag for buffer 2 is $F(2,1) =R(0) + 1/3 =1/3$.
- Therefore the packet from buffer 2 is served first and followed by packet from buffer 1.

Source : http://elearningatria.files.wordpress.com/2013/10/unit2.pdf