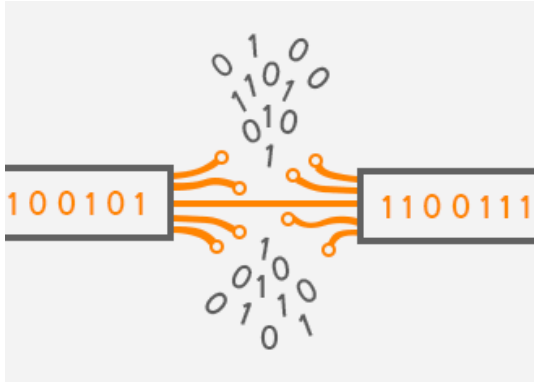
Network Security

Quality of services(QoS): Introduction

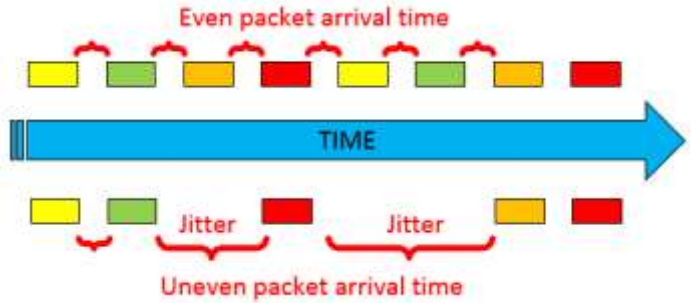
1. Overall performance of a service
2. Ability to achieve maximum bandwidth
3. Something a flow seeks to attain

Manages data traffic to reduce packet loss, latency and jitter on the network.

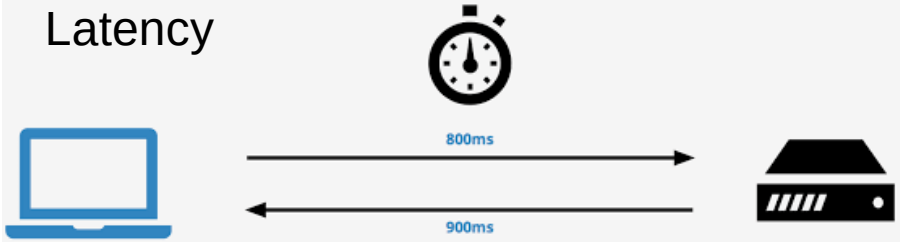
Packet Loss



Jitter



Latency



Flow of Characteristics

Reliability

Deliver the packets safe and sound to destination

Delay

Application can tolerate delay in different degree

Jitter

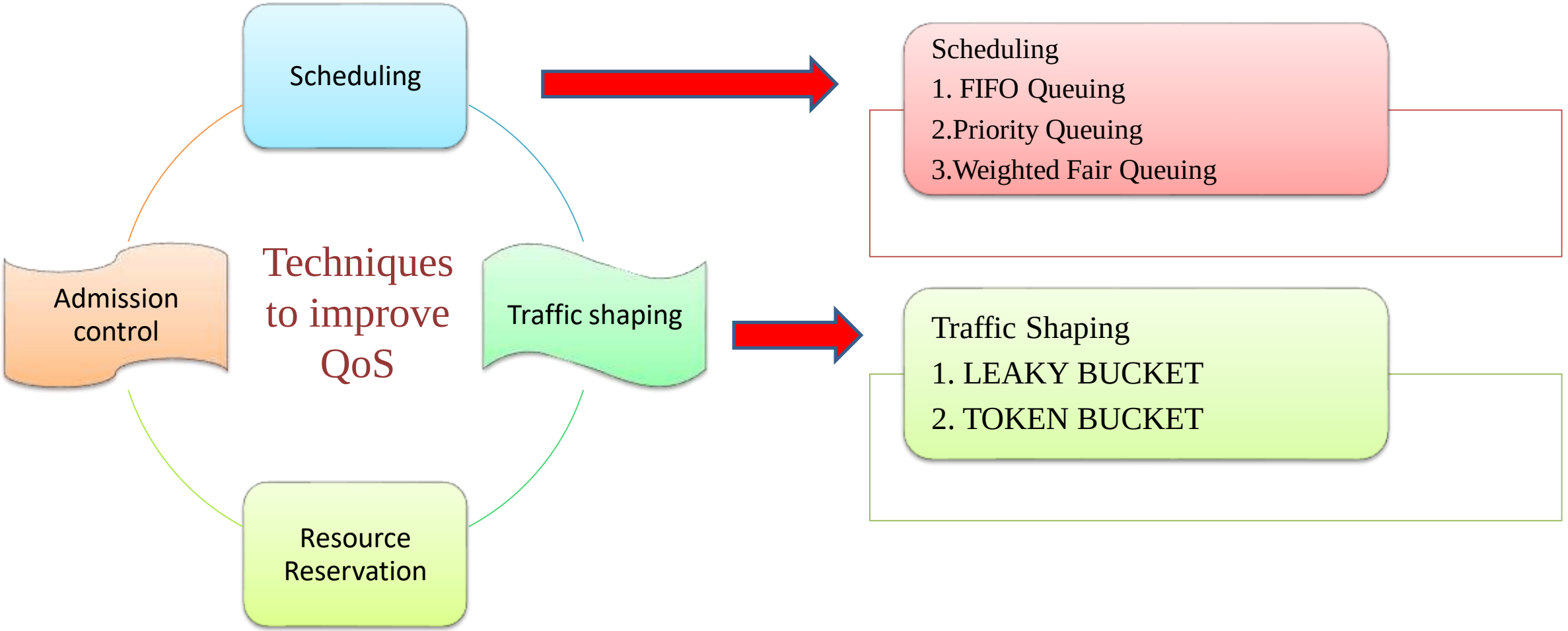
Variation in packets belonging to the same flow.

Bandwidth

Different applications need different bandwidths.

Flow Characteristics

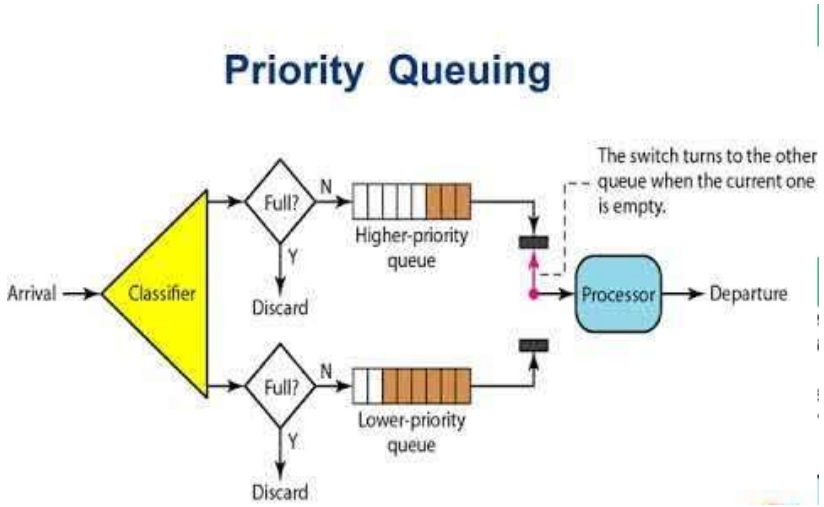
Techniques to improve the QoS

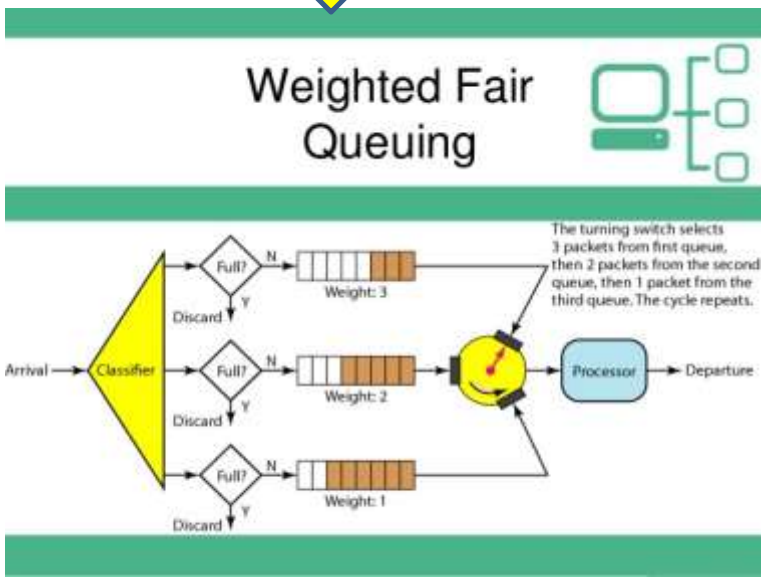


Scheduling

FIFO

Techniques to improve QoS





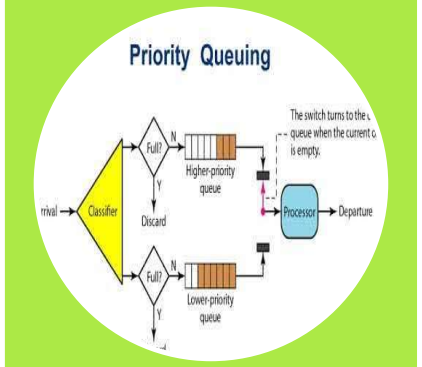
Technique I

Scheduling

Treat different flow in a fair and appropriate manner

FIFO

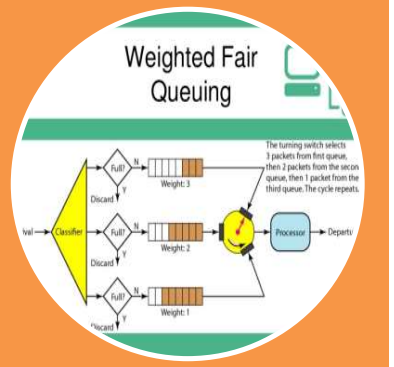
Packet wait in buffer until the node is ready to process



Priority

Packets are assigned to priority class, higher priority packets processed first

Problem: Starvation



Weighted

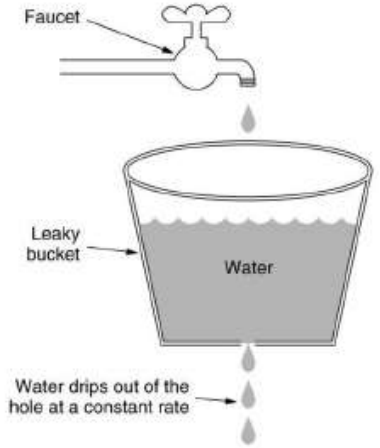
Process packets from high to low in Round Robin Fashion

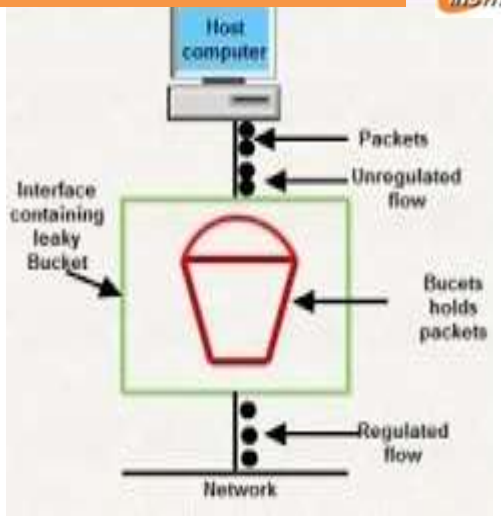
Types of Scheduling

Techniques to improve QoS

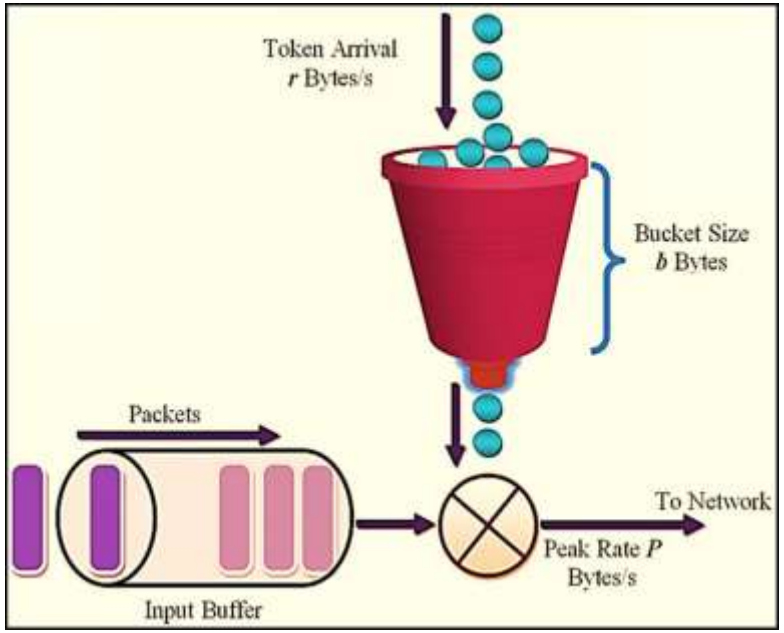
Traffic shaping

Control the amount/ rate of traffic





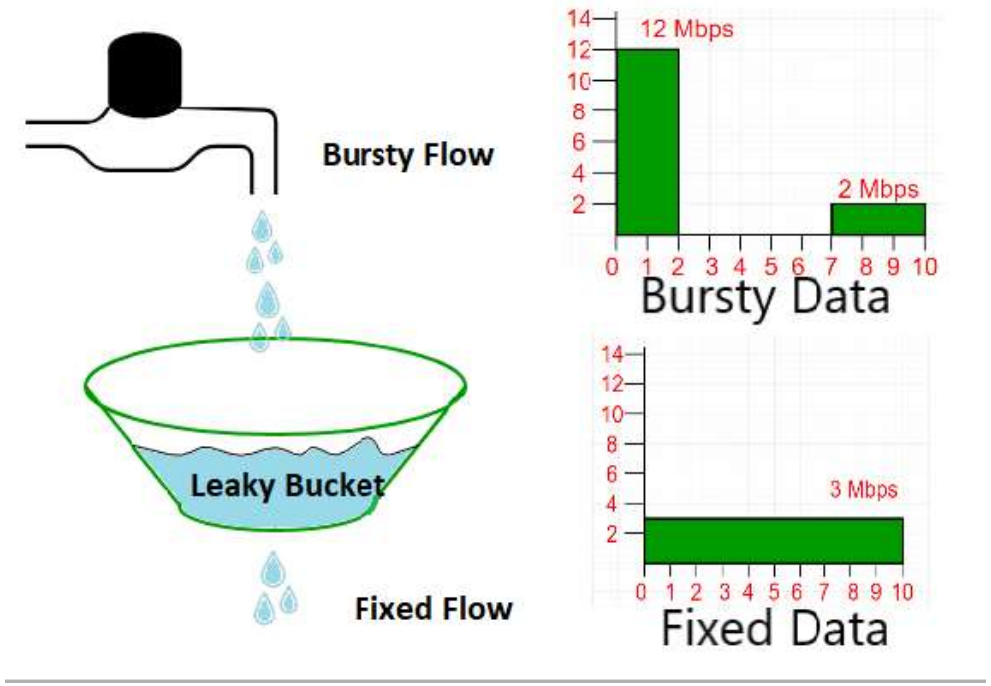
Leaky Bucket



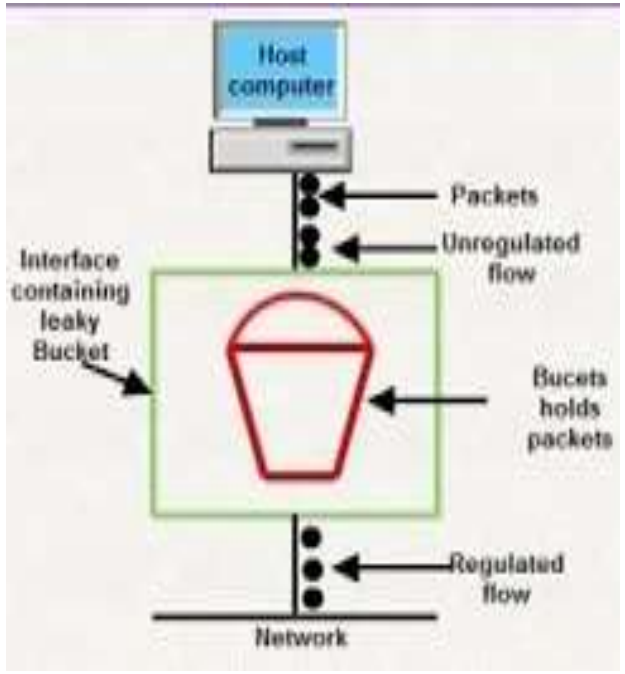
Token Bucket

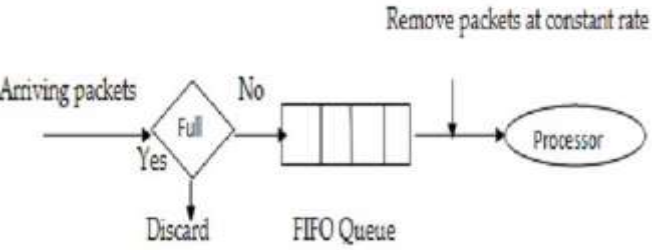
Technique II Traffic Shaping

1. Mechanism to **control** the amount and the rate of the **traffic** sent to the network
2. Helps to regulate rate of data transmission and reduces **congestion**.



Leaky Bucket





A Leaky bucket algorithm shapes bursty traffic into fixed rate traffic by averaging the data rate. It may drop the packets if the bucket is full.

The input rate can vary, but the output rate remains constant.

Algorithm

1. Counter = n at the tick of clock
2. If n > size of packet then send packet and – counter
3. Repeat step 2 until n< packet size
4. Reset the counter and go to step 1.

Let **n** = 1000

Packet =.

200	700	500	450	400	200

Since n > front of Queue i.e. n>200

Therefore, n= 1000-200 = 800

Packet size of 200 is sent to the network

200	700	500	450	400

Now Again n > front of queue i.e. n > 400

Therefore, n= 800-400 = 400

Packet size of 400 is sent to the network

200	700	500	450

Since n < front of queue .

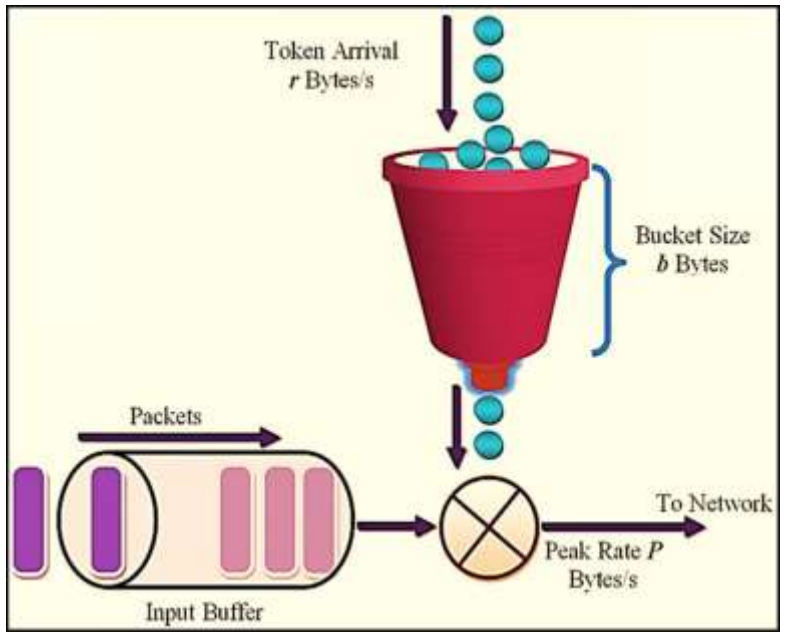
There fore, the procedure is stop.

And we initialize **n = 1000** on another tick of clock.

This procedure is repeated until all the packets is sent to the network.

Drawback: The time when the host was idle is not taken into account

Token Bucket



Idle host to accumulate credit for the future in the form of tokens. For each tick the system sends n tokens to the bucket.

Token bucket allows bursty traffic at a regulated maximum rate.

Algorithm

1. Token = 0
2. Each tick a token is added and counter is incremented by 1
3. Each time a unit of data is sent, the counter is decremented by 1.
4. If counter = 0 the host cannot send data.

Thank You