**Computer Networks | Congestion Control**

What is **congestion**?

A state occurring in network layer when the message traffic is so heavy that it slows down network response time.

**Effects** of Congestion

* As delay increases, performance decreases.
* If delay increases, retransmission occurs, making situation worse.

**Congestion control algorithms**

* **Leaky Bucket Algorithm**

Let us consider an example to understand

Imagine a bucket with a small hole in the bottom.No matter at what rate water enters the bucket, the outflow is at constant rate.When the bucket is full with water additional water entering spills over the sides and is lost.

[](https://cdncontribute.geeksforgeeks.org/wp-content/uploads/leaky.jpg)

Similarly, each network interface contains a leaky bucket and the following **steps** are involved in leaky bucket algorithm:

1. When host wants to send packet, packet is thrown into the bucket.
2. The bucket leaks at a constant rate, meaning the network interface transmits packets at a constant rate.
3. Bursty traffic is converted to a uniform traffic by the leaky bucket.
4. In practice the bucket is a finite queue that outputs at a finite rate.

* **Token bucket Algorithm**

**Need** of token bucket Algorithm:-

The leaky bucket algorithm enforces output pattern at the average rate, no matter how bursty the traffic is. So in order to deal with the bursty traffic we need a flexible algorithm so that the data is not lost. One such algorithm is token bucket algorithm.

**Steps** of this algorithm can be described as follows:

1. In regular intervals tokens are thrown into the bucket. ƒ
2. The bucket has a maximum capacity. ƒ
3. If there is a ready packet, a token is removed from the bucket, and the packet is send.
4. If there is no token in the bucket, the packet cannot be send.

Let’s understand with an example,

In figure (A) we see a bucket holding three tokens, with five packets waiting to be transmitted.For a packet to be transmitted, it must capture and destroy one token. In figure (B) We see that three of the five packets have gotten through, but the other two are stuck waiting for more tokens to be generated.

Let’s understand with an example,

[](https://cdncontribute.geeksforgeeks.org/wp-content/uploads/leakybuk.jpg)

## Congestion control algorithms

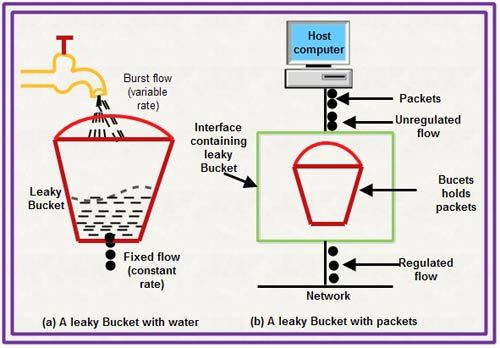
## Leaky Bucket Algorithm

• It is a traffic shaping mechanism that controls the amount and the rate of the traffic sent to the network.

• A leaky bucket algorithm shapes bursty traffic into fixed rate traffic by averaging the data rate.

• Imagine a bucket with a small hole at the bottom.

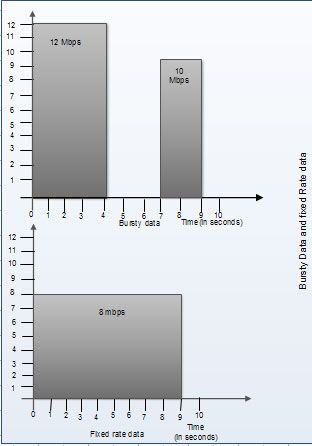
• The rate at which the water is poured into the bucket is not fixed and can vary but it leaks from the bucket at a constant rate. Thus (as long as water is present in bucket), the rate at which the water leaks does not depend on the rate at which the water is input to the bucket.

[](http://ecomputernotes.com/images/Leaky-Bucket.jpg)

• Also, when the bucket is full, any additional water that enters into the bucket spills over the sides and is lost.

• The same concept can be applied to packets in the network. Consider that data is coming from the source at variable speeds. Suppose that a source sends data at 12 Mbps for 4 seconds. Then there is no data for 3 seconds. The source again transmits data at a rate of 10 Mbps for 2 seconds. Thus, in a time span of 9 seconds, 68 Mb data has been transmitted.

If a leaky bucket algorithm is used, the data flow will be 8 Mbps for 9 seconds. Thus constant flow is maintained.

[](http://ecomputernotes.com/images/Bursty-data-and-fixed-rate-data.jpg)

## Token bucket Algorithm

• The leaky bucket algorithm allows only an average (constant) rate of data flow. Its major problem is that it cannot deal with bursty data.

• A leaky bucket algorithm does not consider the idle time of the host. For example, if the host was idle for 10 seconds and now it is willing to sent data at a very high speed for another 10 seconds, the total data transmission will be divided into 20 seconds and average data rate will be maintained. The host is having no advantage of sitting idle for 10 seconds.

• To overcome this problem, a token bucket algorithm is used. A token bucket algorithm allows bursty data transfers.

• A token bucket algorithm is a modification of leaky bucket in which leaky bucket contains tokens.

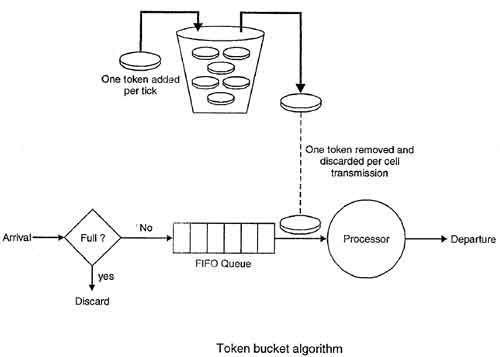
• In this algorithm, a token(s) are generated at every clock tick. For a packet to be transmitted, system must remove token(s) from the bucket.

• Thus, a token bucket algorithm allows idle hosts to accumulate credit for the future in form of tokens.

• For example, if a system generates 100 tokens in one clock tick and the host is idle for 100 ticks. The bucket will contain 10,000 tokens.

Now, if the host wants to send bursty data, it can consume all 10,000 tokens at once for sending 10,000 cells or bytes.

Thus a host can send bursty data as long as bucket is not empty.

[](http://ecomputernotes.com/images/Token-bucket-Algorithm.jpg)

**Quality-of-Service (QoS)**

**It** refers to traffic control mechanisms that seek to either differentiate performance based on application or network-operator requirements or provide predictable or guaranteed performance to applications, sessions or traffic aggregates. Basic phenomenon for QoS means in terms of packet delay and losses of various kinds.

**Need for QoS –**

* Video and audio conferencing require bounded delay and loss rate.
* Video and audio streaming requires bounded packet loss rate, it may not be so sensitive to delay.
* Time-critical applications (real-time control) in which bounded delay is considered to be an important factor.
* Valuable applications should be provided better services than less valuable applications.

**QoS Specification –**  
QoS requirements can be specified as:

1. Delay
2. Delay Variation(Jitter)
3. Throughput
4. Error Rate

There are two types of QoS Solutions:

1. **Stateless Solutions –**  
   Routers maintain no fine grained state about traffic, one positive factor of it is that it is scalable and robust. But it has weak services as there is no guarantee about kind of delay or performance in a particular application which we have to encounter.
2. **Stateful Solutions –**  
   Routers maintain per flow state as flow is very important in providing the Quality-of-Service i.e. providing powerful services such as guaranteed services and high resource utilization, provides protection and is much less scalable and robust.

**Integrated Services(IntServ) –**

1. An architecture for providing QoS guarantees in IP networks for individual application sessions.
2. Relies on resource reservation, and routers need to maintain state information of allocated resources and respond to new call setup requests.
3. Network decides whether to admit or deny a new call setup request.

**IntServ QoS Components –**

* Resource reservation: call setup signaling, traffic, QoS declaration, per-element admission control.
* QoS-sensitive scheduling e.g WFQ queue discipline.
* QoS-sensitive routing algorithm(QSPF)
* QoS-sensitive packet discard strategy.

**RSVP-Internet Signaling –**  
It creates and maintains distributed reservation state, initiated by the receiver and scales for multicast, needs to be refreshed otherwise reservation times out as it is in soft state. Latest paths discovered through “PATH” messages (forward direction) and used by RESV messages (reserve direction).

**Call Admission –**

* Session must first declare it’s QoS requirement and characterize the traffic it will send through the network.
* **R-specification:** defines the QoS being requested, i.e. what kind of bound we want on the delay, what kind of packet loss is acceptable, etc.
* **T-specification:** defines the traffic characteristics like bustiness in the traffic.
* A signaling protocol is needed to carry the R-spec and T-spec to the routers where reservation is required.
* Routers will admit calls based on their R-spec, T-spec and based on the current resource allocated at the routers to other calls.

**Diff-Serv –**  
Differentiated Service is a stateful solution in which each flow doesn’t means a different state. It provides reduced state services i.e. maintain state only for larger granular flows rather than end-to-end flows tries to achieve best of both worlds.

Intended to address the following difficulties with IntServ and RSVP:

1. **Flexible Service Models:**  
   IntServ has only two classes, want to provide more qualitative service classes: want to provide ‘relative’ service distinction.
2. **Simpler signaling:**  
   Many applications and users may only want to specify a more qualitative notion of service.

**Streaming Live Multimedia –**

* **Examples:** Internet radio talk show, Live sporting event.
* **Streaming:** playback buffer, playback buffer can lag tens of seconds after and still have timing constraint.
* **Interactivity:** fast forward is impossible, but rewind and pause is possible