Abstract—At most site today condition of over-whelm may occur this may be due to exponential increase in population or providing the services to all the customers, this may lead to the failure of the site. Cost of maintenance comes out to be larger than prevention. That's why we adopt the method to prevent the site from failure.

Keywords—Trail drop Queuing, Bandwidth Management, Bottleneck Network, Modified tail drop queuing.

I. INTRODUCTION

Network congestion and poor performance of the network are the two major problem which we are facing nowadays. Both are directly proportional to one another. Due to Network congestion performance of network degrades automatically.

II. TRAIL DROP QUEUING

The following example illustrates the implementation of the Queue/Drop Tail object, which implements FIFO scheduling and drop-on-overflow buffer management typical of most present-day Internet routers.

/*
 * A bounded, drop-tail queue
 */
class DropTail : public Queue {
  Protected:
    void enqueue(Packet*);
    Packet* dequeue();
    PacketQueue q_;
};

The base class Queue, from which Drop Tail is derived, provides most of the needed functionality. The drop-tail queue maintains exactly one FIFO queue, implemented by including an object of the Packet Queue class. Drop-tail implements its own versions of enqueue and dequeue as follows:

/*
 * drop-tail
 */
void DropTail::enqueue(Packet* p)
{
    q_.enqueue(p);
    if (q_.length() >= qlim_) {
        q_.remove(p);
        drop(p);
    }
}
Packet* DropTail::dequeue()
{
    return (q_.dequeue());
}

Here, the enqueue function first stores the packet in the internal packet queue (which has no size restrictions), and then checks the size of the packet queue versus qlim_. Drop-on-overflow is implemented by dropping the packet most recently added to the packet queue if the limit is reached or exceeded. Note: in the implementation of enqueue above, setting qlim_ to n actually means a queue size of n-1. Simple FIFO scheduling is implemented in the dequeue function by always returning the first packet in the packet queue.

III. BANDWIDTH MANAGEMENT

Bandwidth management is the process of measuring and controlling the communications (traffic, packets) on a network link, to avoid filling the link to capacity or overfilling the link, which would result in network congestion and poor performance of the network.

Bandwidth management mechanisms and techniques are as follows:
A. Token bucket

The token bucket is an algorithm used in packet switched computer networks and telecommunications networks. It can be used to check that data transmissions, in the form of packets, conform to defined limits on bandwidth and burstiness (a measure of the unevenness or variations in the traffic flow) for example fig:-1. It can also be used as a scheduling algorithm to determine the timing of transmissions that will comply with the limits set for the bandwidth and burstiness.

B. Leaky bucket

The leaky bucket is an algorithm used in packet switched computer networks and telecommunications networks. It can be used to check that data transmissions, in the form of packets, conform to defined limits on bandwidth and burstiness (a measure of the unevenness or variations in the traffic flow) for example fig:-2. It can also be used as a scheduling algorithm to determine the timing of transmissions that will comply with the limits set for the bandwidth and burstiness. The leaky bucket algorithm is also used in leaky bucket counters, e.g. to detect when the average or peak rate of random or stochastic events or stochastic processes exceed defined limits.

IV. Bottleneck Network

A bottleneck is a phenomenon where the performance or capacity of an entire system is limited by a single or limited number of components or resources. The term bottleneck is taken from the 'assets are water' metaphor. As water is poured out of a bottle, the rate of outflow is limited by the width of the conduit of exit—that is, bottleneck for example fig:-3. By increasing the width of the bottleneck one can increase the rate at which the water flows out of the neck at different frequencies. Such limiting components of a system are sometimes referred to as bottleneck points.
V. MODIFIED DROP TAIL QUEUING

It is the combination of First-in-First-out algorithm and congestion window. Both are used to avoid the bottleneck and avoid large amount of packets drop. I used the congestion window technique together with FIFO scheduling so that min packets are dropped and communication is maintained between the nodes For example Fig: 4. By using this technique nodes will never waste their time waiting for packets to communicate as well as in all the communication less amount of resources will be used. To control the communications on a network link, to avoid network congestion and poor performance of the network we implement bottleneck technique within the network. This will control the flow between 2 nodes and allow only the amount of packets need to transfer, rest is drop down according to the trail drop algorithm.

VI. CONCLUSION

To control the communications on a network link, to avoid network congestion and poor performance of the network we need the process of Modified drop tail queuing. This is a new technique and very useful and effective if we use this method in a proper way. Now days there are various other algorithm use for avoiding network congestion but still it’s the best way to avoid it.

REFERENCES


Author’s Profile

Harshit Nigam obtained Btech (HONOURS) in (Computer Science and Engineering) from G.B.T.U in 2011 and pursuing Mtech (Computer Science) from United Institute of Technology Allahabad. He is working as a lecturer in S.P. Memorial Institute of Technology. His area of interest includes Information Security and Graph Theory. He has more than 3 years of experience.

Akhilesh Singh obtained Mtech degree from IIT Bombay and working as an assistant professor in United Institute of Technology. He obtained his Btech degree from United College of Engg. And Research From U.P.T.U. He is also the co-coordinator of Mtech and Head of Department of Computer Science department.