بسمه تعالی

School of Computer Engineering

Advanced Computer Networks

Transport Layer and Congestion Control

Part 6

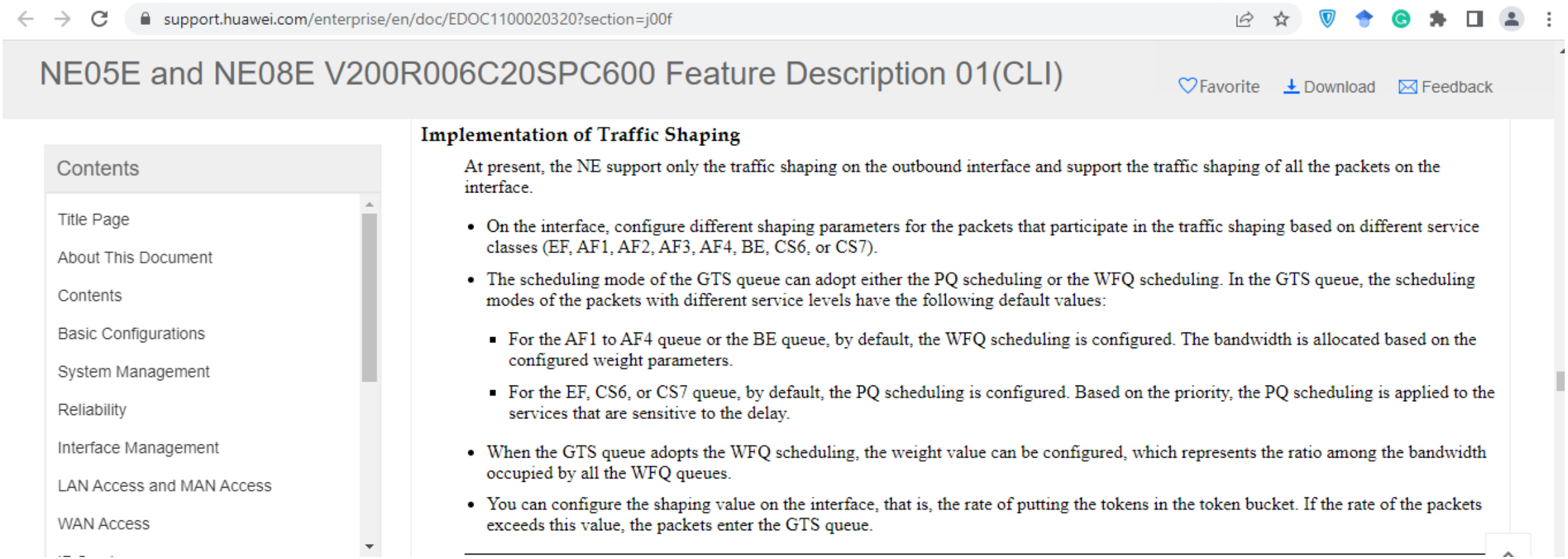
Seyed Hamed Rastegar

Fall 1401

Congestion Control: Queueing Disciplines

Fair Queueing: WFQ

- WFQ in practice (link)



Congestion Control: Traffic Shaping

- Before the network can make performance guarantees, it must know what traffic is being guaranteed.

- In the **telephone network**, this characterization is simple. For example, a voice call (in uncompressed format) needs 64 kbps and consists of one 8-bit sample every 125 µsec.

- However, **traffic in data networks is bursty**. It typically arrives at **non-uniform rates** as the traffic rate varies (e.g., videoconferencing with compression), users interact with applications (e.g., browsing a new Web page), and computers switch between tasks.

- **Bursts of traffic** are more difficult to handle than constant-rate traffic because they can **fill buffers** and cause **packets to be lost**.

Congestion Control: Traffic Shaping

- **Traffic shaping** is a technique for regulating the **average rate** and **burstiness** of a flow of data that enters the network.
- The goal is to allow applications to transmit a wide variety of traffic that suits their needs, including some bursts, yet have a simple and useful way to describe the possible traffic patterns to the network.
- When a flow is set up, the user and the network (i.e., the customer and the provider) agree on a certain traffic pattern (i.e., shape) for that flow.
- In effect, the customer says to the provider ‘‘My transmission pattern will look like this; can you handle it?

Congestion Control: Traffic Shaping

- Sometimes this agreement is called an **SLA (Service Level Agreement)**, especially when it is made over aggregate flows and long periods of time, such as all of he traffic for a given customer.

- As long as the customer fulfills his/her part of the bargain and only sends packets according to the agreed-on contract, the provider promises to deliver them all in a timely fashion.

- **Traffic shaping reduces congestion** and thus helps the network live up to its promise.

Congestion Control: Traffic Shaping

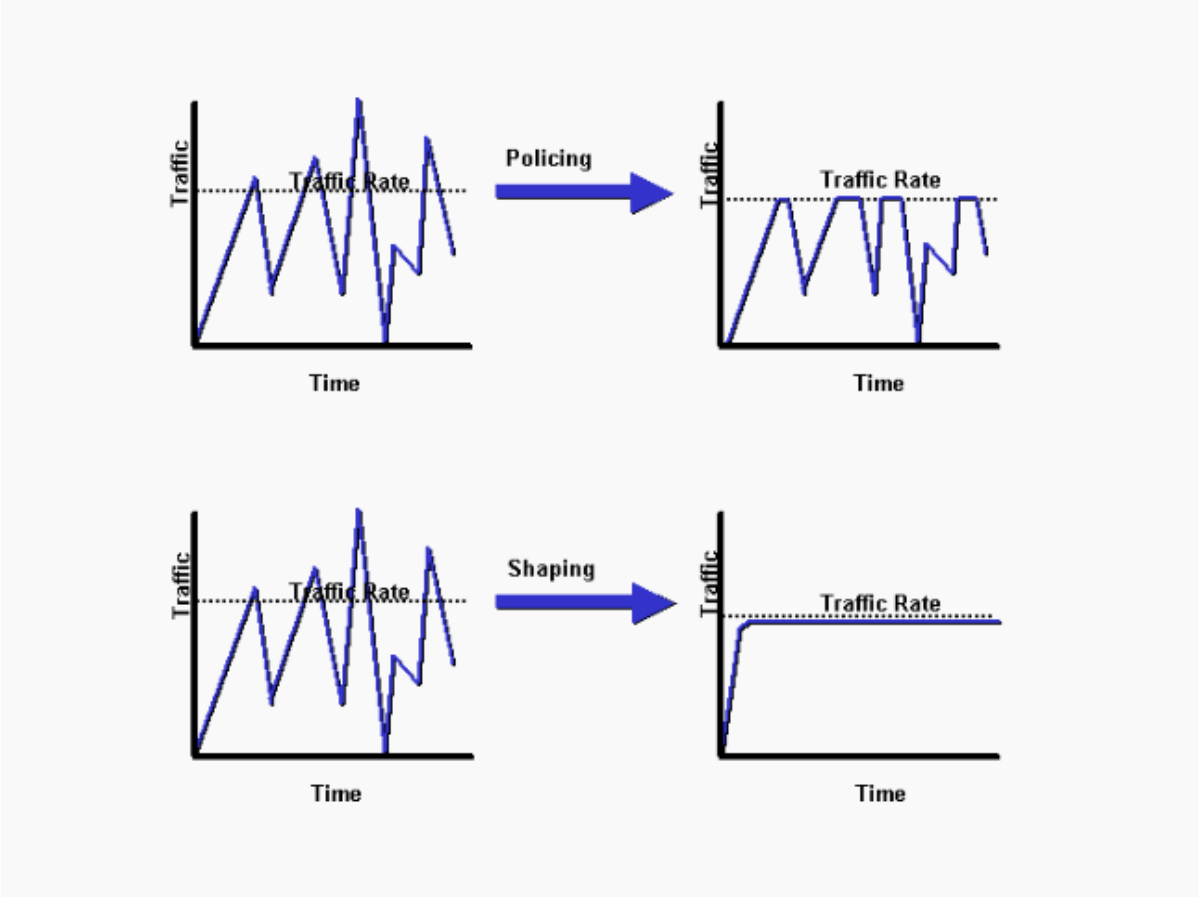
- However, to make it work, there is also the issue of how the provider can tell if the customer is following the agreement and what to do if the customer is not.

- Packets in excess of the agreed pattern might be dropped by the network, or they might be marked as having lower priority.
  - ❑ Monitoring a traffic flow is called **traffic policing**.

- Shaping and policing are **not so important for peer-to-peer** and other transfers that will consume any and all available bandwidth, but they are of **great importance for real-time data**, such as audio and video connections, which have **stringent quality-of-service requirements.**

Congestion Control: Traffic Shaping

- Cisco view on Traffic Shaping vs Traffic Policing (link)



Congestion Control: Leaky and Token Bucket

We have already seen one way to limit the amount of data an application sends:

- Do you remind that?!
- **the sliding window**

- Sliding window uses one parameter to limit how much data is in transit at any given time, which indirectly limits the rate.

- Now we will look at a more general way to characterize traffic, with the **leaky bucket and token bucket** algorithms.

Congestion Control: Leaky and Token Bucket

- The regulation of the rate at which a class or flow is allowed to inject packets into the network, is an important QoS mechanism.

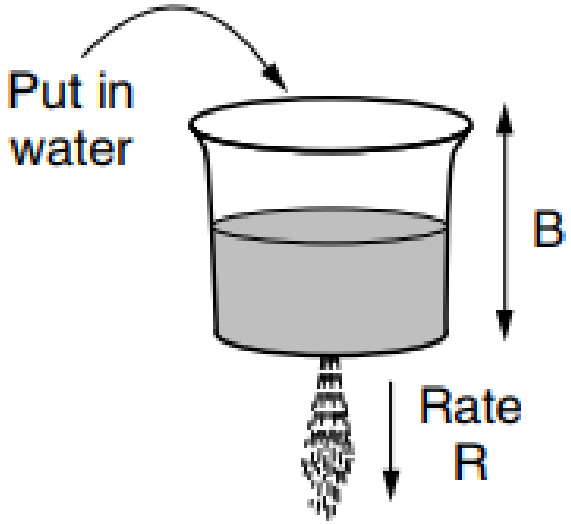
- But what aspects of a flow's packet rate should be policed?

- We can identify **three important policing criteria**, each differing from the other according to the time scale over which the packet flow is policed:
  - **Average Rate**
  - **Peak Rate**
  - **Burst Size**

Congestion Control: Leaky Bucket

Leaky Bucket

- Imagine a bucket with a **small hole in the bottom**, as illustrated in the Figure.



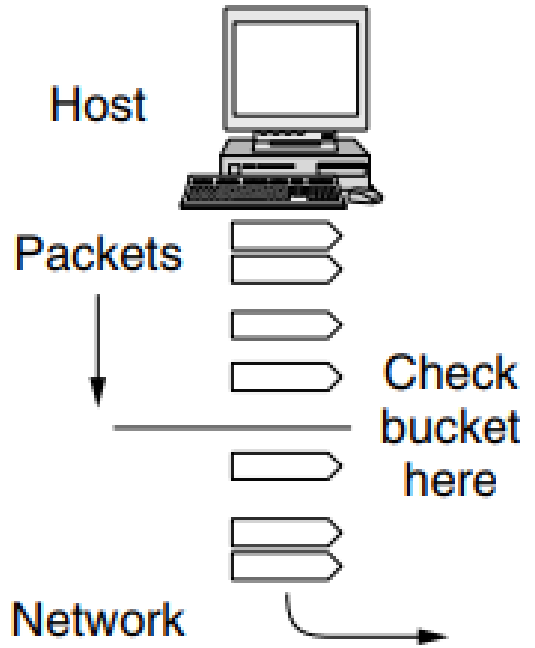
- No matter the rate at which water enters the bucket, the **outflow** is at a constant rate, *R*, when there is any water in the bucket and zero when the bucket is empty.

- Also, **once the bucket is full** to capacity *B*, any additional water entering it spills over the sides and is lost.

Congestion Control: Leaky Bucket

Leaky Bucket

- This bucket can be used to **shape or police packets entering the network**, as shown in the **Figure**.
- Conceptually, **each host** is connected to the network by an **interface containing a leaky bucket**.
- **If a packet arrives when the bucket is full**, the packet must either be queued until enough water leaks out to hold it or be discarded.
  - **The former** might happen at a host shaping its traffic for the network as part of the operating system.
  - **The latter** might happen in hardware at a provider network interface that is policing traffic entering the network.



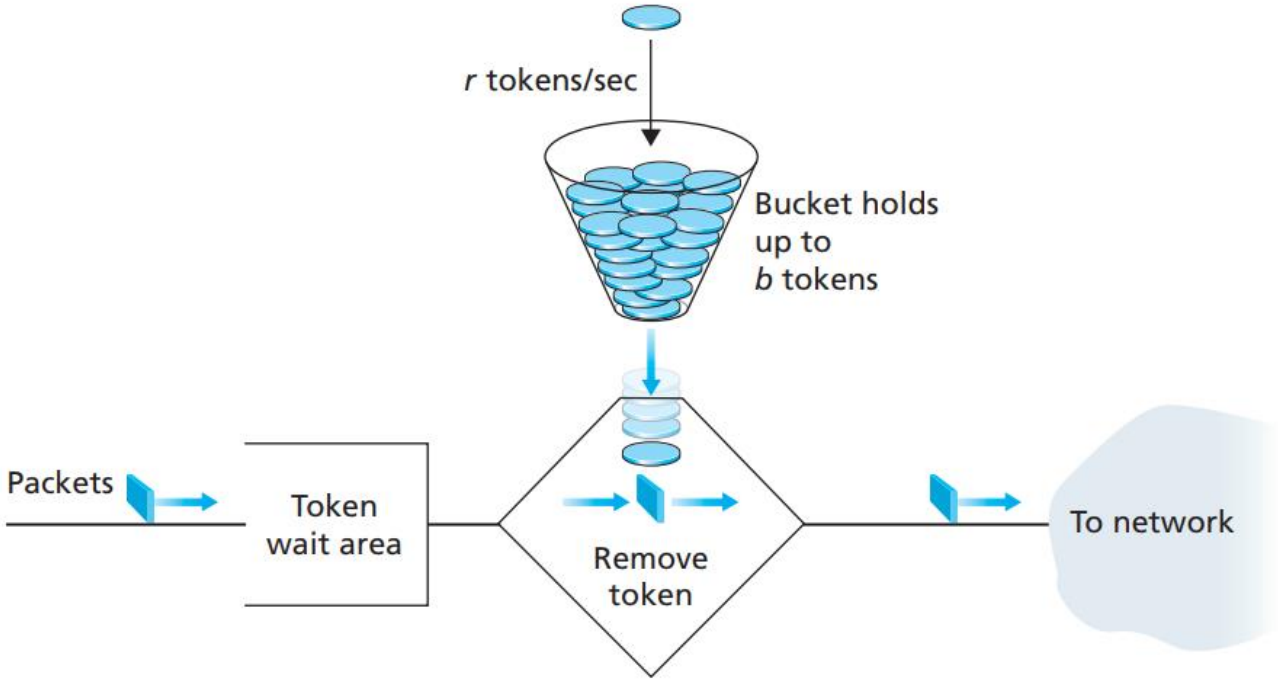
Turner, Jonathan. "New directions in communications (or which way to the information age?)." *IEEE communications Magazine* 24, no. 10 (1986): 8-15.

Congestion Control: Token Bucket

Token Bucket

- **Important Note**: Some references (including Kurose & Ross Book!), introduce token bucket as leaky bucket and do not discuss the leaky bucket concept independently.

- The token bucket mechanism is an abstraction that can be used to characterize the policing limits more accurately.



Congestion Control: Token Bucket

Token Bucket

- A Token bucket consists of a bucket that can hold up to ***b* tokens**.

- Tokens are added to this bucket as follows:

  - **New tokens**, which may potentially be added to the bucket, are always being generated at a **rate of *r* tokens per second**.

  - If the bucket is filled with **less than *b* tokens** when a token is generated, the newly generated token is added to the bucket; otherwise the newly generated token is ignored, and the token bucket **remains full with *b* tokens**.

Congestion Control: Token Bucket

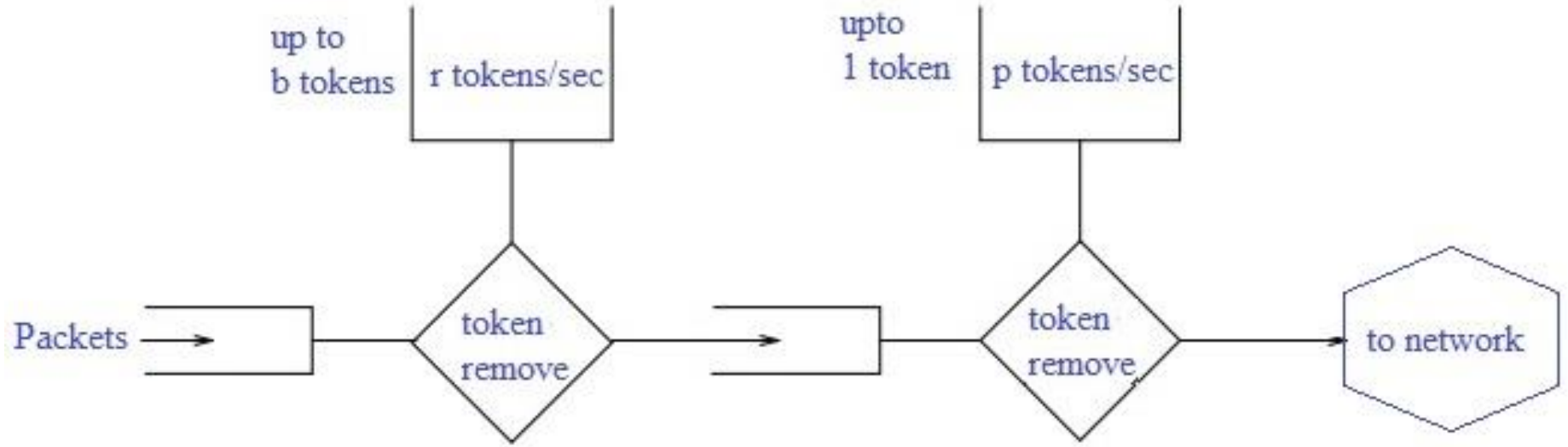
Token Bucket

- How the Token bucket can be used to police a packet flow:
  - **Assumption**: Suppose that before a packet is transmitted into the network, it must first remove a token from the token bucket. If the token bucket is empty, the packet must wait for a token. (An alternative is for the packet to be dropped)
  - **Policing Burst Size**: Because there can be at most ***b* tokens** in the bucket, the maximum burst size for a token-bucket policed flow is ***b* packets**.
  - **Policing Average Rate**: Furthermore, because the token generation rate is *r*, the maximum number of packets that can enter the network of any interval of time of length *t* is $rt + b$.
    - Thus, the token-generation rate, *r*, serves to limit the long-term average rate at which packets can enter the network.

Congestion Control: Token Bucket

Token Bucket

- How to set a *peak rate* with token bucket?

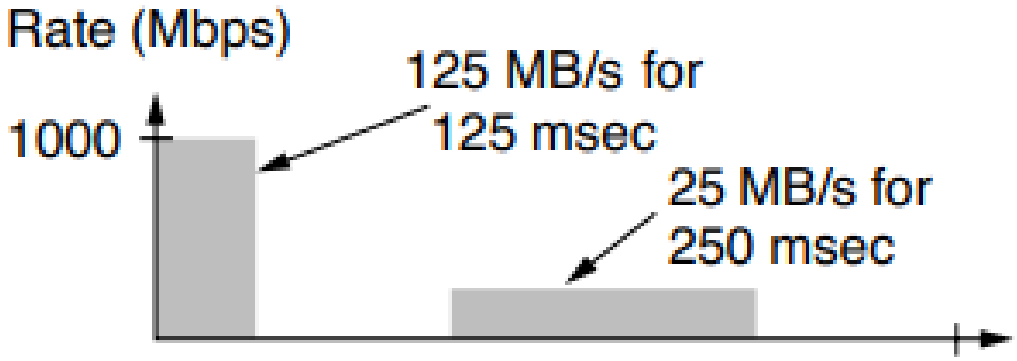


Congestion Control: Token Bucket

Token Bucket: Example

- As an example, imagine that a computer can produce data at up to 1000 Mbps (125 million bytes/sec) and that the first link of the network also runs at this speed.

- The pattern of traffic the host generates is shown in the Figure.



Congestion Control: Token Bucket

Token Bucket: Example

- This pattern is bursty.
  - ❑ The average rate over one second is 200 Mbps, even though the host sends a burst of 16,000 KB at the top speed of 1000 Mbps (for 1/8 of the second)

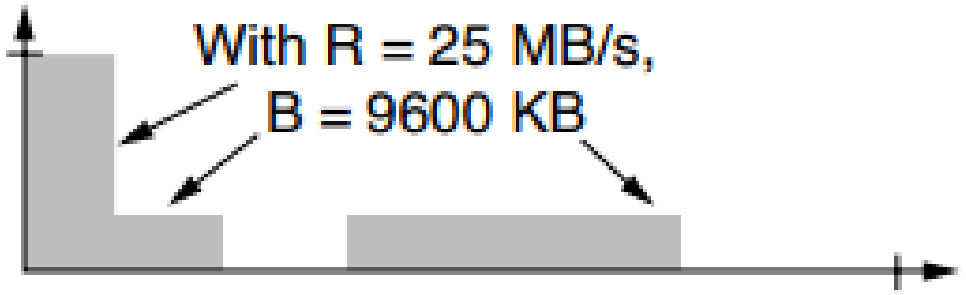
- Now suppose that the routers can accept data at the top speed only for short intervals, until their buffers fill up. The buffer size is 9600 KB, smaller than the traffic burst.

- For long intervals, the routers work best at rates not exceeding 200 Mbps (say, because this is all the bandwidth given to the customer).
  - ❑ The implication is that if traffic is sent in this pattern, some of it will be dropped in the network because it does not fit into the buffers at routers.

Congestion Control: Token Bucket

Token Bucket: Example

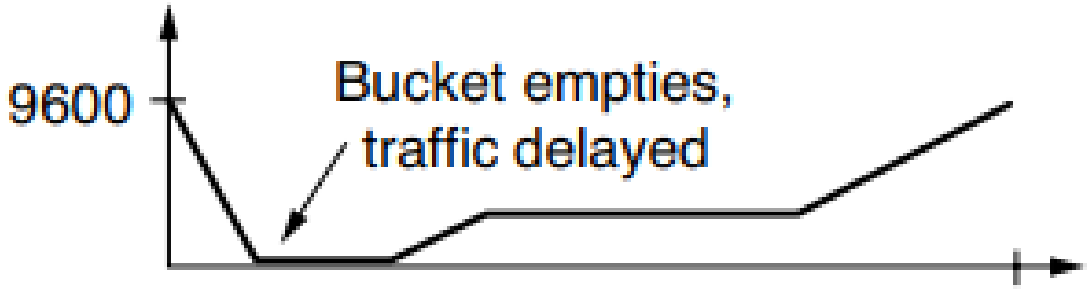
- To avoid this packet loss, we can shape the traffic at the host with a token bucket.
- If we use a rate, *R*, of 200 Mbps and a capacity, *B*, of 9600 KB, the traffic will fall within what the network can handle.
- The output of this token bucket is shown in the Figure.



Congestion Control: Token Bucket

Token Bucket: Example

- The host can send full throttle at 1000 Mbps for a short while until it has fully drained the bucket.
- Then it has to cut back to 200 Mbps until the burst has been sent.
- The effect is to spread out the burst over time because it was too large to handle all at once. The level of the token bucket is shown in the Figure.



Congestion Control: Token Bucket

Token Bucket: Example

- It starts off full and is depleted by the initial burst.

- When it reaches zero, new packets can be sent only at the rate at which the buffer is filling;
  - ❑ there can be no more bursts until the bucket has recovered.

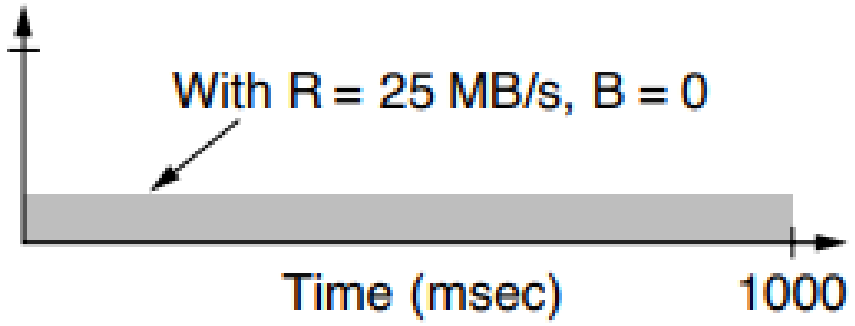
- The bucket fills when no traffic is being sent and stays flat when traffic is being sent at the fill rate.

Congestion Control: Token Bucket

Token Bucket: Example

- We can also shape the traffic to be less bursty.

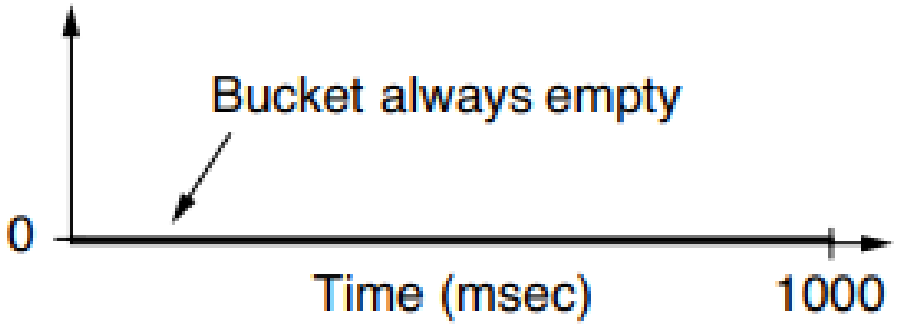
- The Figure shows the output of a token bucket with R = 200 Mbps and a capacity of 0.



Congestion Control: Token Bucket

Token Bucket: Example

- This is the **extreme case** in which the traffic has been **completely smoothed**.
- **No bursts are allowed**, and the traffic enters the network at a steady rate.
- The corresponding **bucket level**, shown in the Figure, **is always empty**.



- **Traffic is being queued on the host** for release into the network and there is always a packet waiting to be sent when it is allowed.