

Policing and Shaping

Token Bucket Principles

(C) Herbert Haas 2009/05/02

Policing and Shaping



- **Policing**
 - ♦ Drops out-of-profile traffic (TCP retransmissions)
 - ♦ More resource efficient
 - ♦ Supports incoming and outgoing interfaces
 - ♦ Supports (re-)marking
- **Shaping**
 - ♦ Reduces bursts by queuing out-of-profile traffic
 - Minimizes TCP retransmits
 - ♦ **Only for outbound interfaces**
 - ♦ Does not support (re-)marking
 - ♦ **"Delay instead drop"**



(C) Herbert Haas 2009/05/02 www.perihel.at

2

Token Buckets

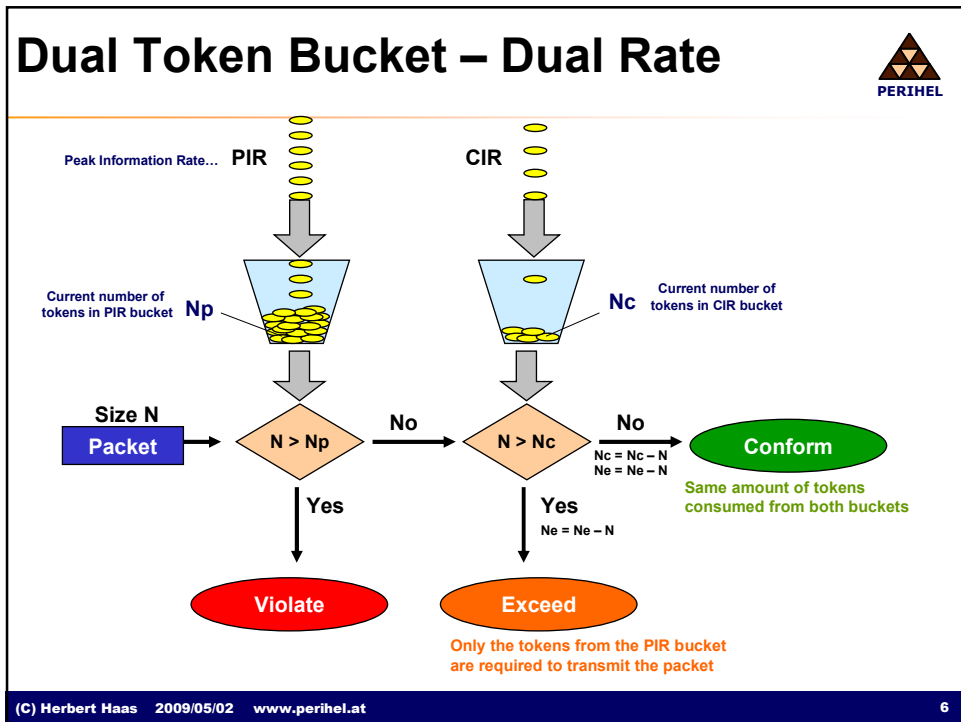
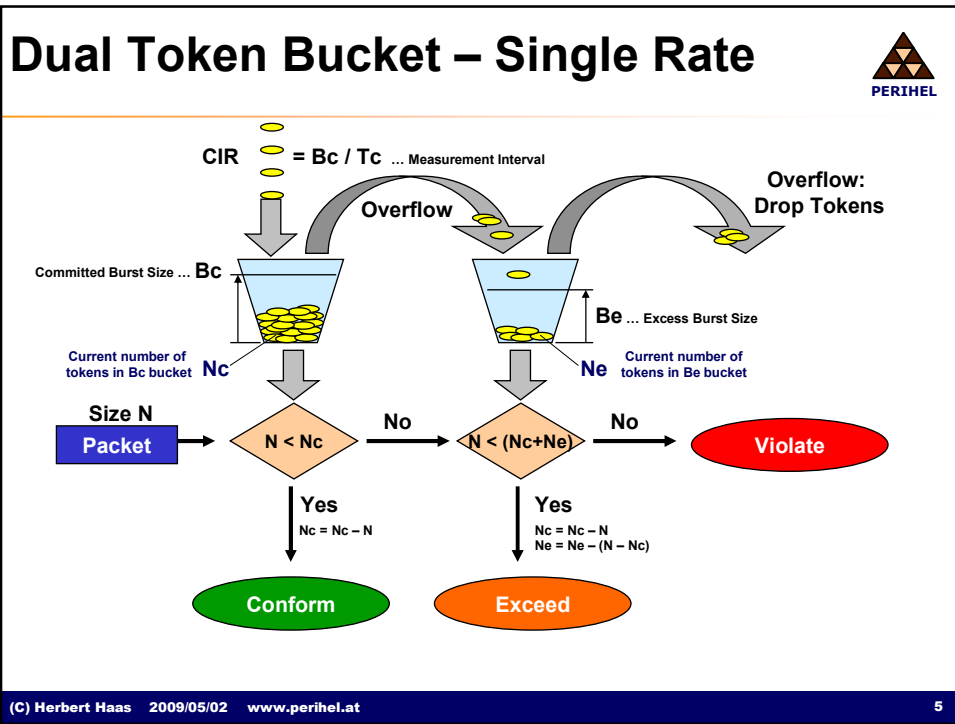


- **Single Token Bucket**
 - ◆ Token Rate is proportional to Committed Information Rate (CIR)
 - ◆ Token Bucket may hold Bc Tokens so all traffic cannot exceed CIR
- **Dual Token Bucket (single rate)**
 - ◆ Second bucket keeps extra tokens for bursts exceeding the CIR
 - These tokens are spillovers from the Bc buckets
 - ◆ Excess rate depends on utilization of committed rate
 - Bursts can only be transmitted if both buckets contain enough tokens

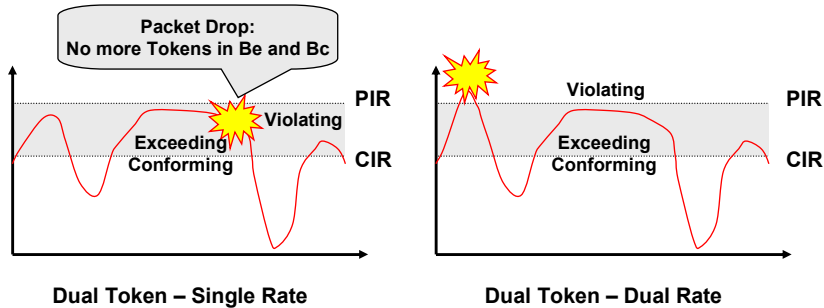
Token Buckets (cont.)



- **Dual-rate Dual Token Bucket**
 - ◆ Allows better bandwidth management
 - ◆ Supports sustained excess rates
 - ◆ Excess rate is independent of CIR – independent rate thresholds!
 - ◆ Typically applied on network edges
 - Allow conform packets
 - Remark excess packets (e. g. lower priority)



Practical Comparison



- Violation occurs if
 - ◆ Burst rate exceeds PIR
 - ◆ Or burst duration is too long

- Violation occurs only if burst rate exceeds PIR
 - ◆ Ideal for threshold-based traffic management

Other Bucket Principles



- Leaky Bucket
- Dual Leaky Bucket
 - ◆ Better performance than Dual Token Bucket
- Hierarchical Token Bucket
 - ◆ Used by Linux since kernel 2.6.20
- Many others...

Configuration



- Using Cisco's MQC within policy-map using the `police` command
- Different implementations depending on platform and IOS version
 - ◆ Single/dual Token Bucket with single/dual rates
 - ◆ Special Frame-Relay shaping options (per DLCI or subinterface, using FECN/BECN)

Configuration: One Rate DTB



```
(config-pmap-c)# police <avg-rate>
                    [<Bc> [<Be>]]
                    [conform-action <action>]
                    [exceed-action <action>]
                    [violate-action <action>]
```

- avg-rate: 8000..200,000,000 bps
- Bc: Normal burst size
 - ◆ Max{ 1500, CIR/32 } bytes (using Tc = 250 ms)
- Be: Excess burst size (default: =Bc)
- Actions:
 - ◆ drop | transmit (defaults for exceed/violation | conform)
 - ◆ set-dscp-transmit *value*
 - ◆ set-prec-transmit *value*
 - ◆ set-mpls-experimental-topmost-transmit *value*
 - ◆ ...
- **Single Token Bucket does not support violate-action**

Configuration: Two Rate DTB



```
(config-pmap-c)# police cir <cir> [bc <bc>]
                    pir <pir> [be <be>]]
                    [conform-action <action>]
                    [exceed-action <action>]
                    [violate-action <action>]
```

- avg-rate: 8000..200,000,000 bps
- Default Bc = Max{ 1500, CIR/32 } bytes (using Tc = 250 ms)
- Default Be = Max{ 1500, PIR/32 } bytes (using Tc = 250 ms)

Configuration: Two Rate DTB with %



```
(config-pmap-c)# police cir percent <percent> [bc <bc_in_ms>]
                    pir percent <percent> [be <be_in_ms>]]
                    [conform-action <action>]
                    [exceed-action <action>]
                    [violate-action <action>]
```

- Allows to use the same policy on different interfaces with different bandwidths
- bc and be can be specified in ms
 - ◆ As equivalent to a percentage

Example & Verification



```
policy-map MY_TRAFFIC_POLICY
  class MY_VOICE
    police cir 250000000 conform-action transmit
                          exceed-action transmit
                          violate-action drop
  class MY_DATA
    police cir 250000000 conform-action transmit
                          exceed-action set-prec-transmit 3
                          violate-action drop
interface fa0/0
  service-policy input MY_TRAFFIC_POLICY
```

```
# show policy-map interface fa0/0
```

Example: Multi-action



```
(config)# policy-map police
(config-pmap)# class class-default
(config-pmap-c)# police cir 1000000 pir 2000000
(config-pmap-c-police)# conform-action transmit
(config-pmap-c-police)# exceed-action set-prec-transmit 4
(config-pmap-c-police)# exceed-action set-frde
(config-pmap-c-police)# violate-action set-prec-transmit 2
(config-pmap-c-police)# violate-action set-frde-transmit
(config-pmap-c-police)# end
```

Shaping



- Either shape to
 - ♦ **Average rate** (CIR) – but allow up to $(Bc+Be)/Tc$
 - ♦ **Peak rate** – but during congestion drop down to CIR
- It is recommended to specify the bit-rate only and let the IOS calculate Bc and Be
 - ♦ By default Tc is 4 ms
 - ♦ If Bc given, then $Tc = Bc / CIR$

```
(config-pmap-c)# shape average|peak <bit-rate> [<Bc>] [<Be>]
```

```
(config-pmap-c)# shape average|peak percent <%> [<Bc_ms>] [<Be_ms>]
```

Optionally specify the length of the shaping queue (in packets aka "buffers")

```
(config-pmap-c)# shape max-buffers 1000
```

Shaping Example



```
shape {average | peak} <cir> [<bc>][<be>]
```

```
policy-map MY_FTP_POLICY
class af11-traffic
bandwidth remaining percent 15
random-detect dscp-based
random-detect dscp 10 26 40 10
shape average 16000
```