

High-speed Networks, Cybersecurity, and Software-defined Networking Workshop

Jorge Crichigno, Elie Kfoury
University of South Carolina

Western Academy Support and Training Center (WASTC)
2020 Summer Conference
June 15 – June 19



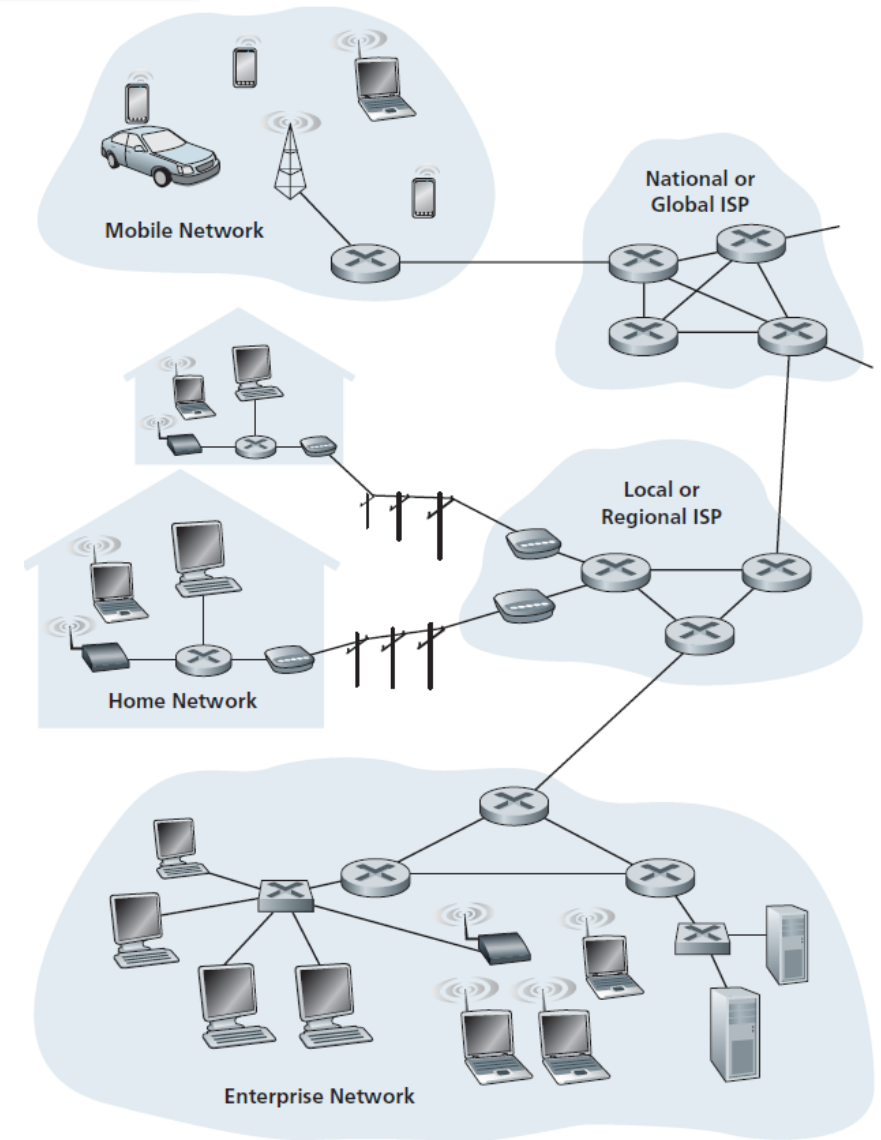
National Science Foundation (NSF), Office of Advanced Cyberinfrastructure (OAC) and
Advanced Technological Education (ATE)

Lab 5: Setting WAN Bandwidth with Token Bucket Filter (TBF)

Token Bucket Filter

Introduction to Token Bucket Algorithm

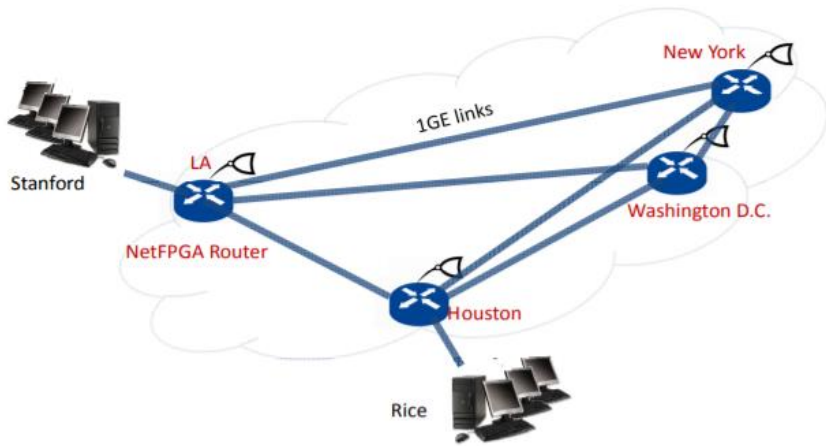
- When emulating networks, it is sometimes necessary to set the bandwidth of devices
- By manipulating the bandwidth / capacity of a link, the network behavior under different conditions can be observed



J. Kurose, K. Ross, Computer Networking: A Top-Down Approach, 7th Edition, Pearson, 2017

Introduction to Token Bucket Algorithm

- Example: what is the ideal buffer size in a router, so that the utilization of a bottleneck link is maximized?

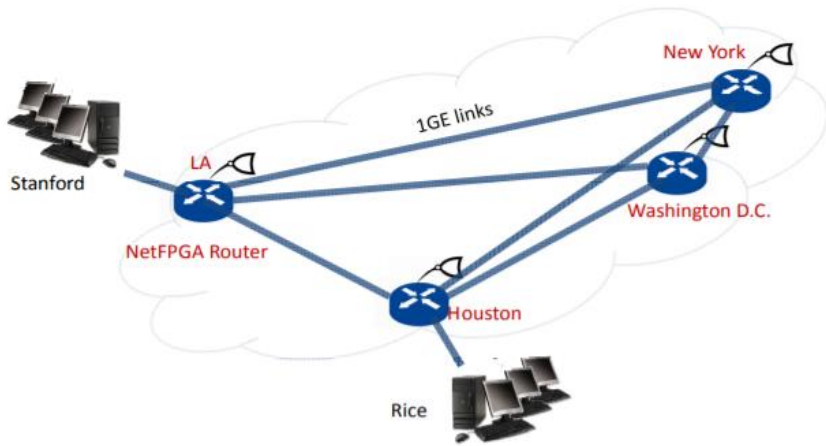


2009 Hardware experiment

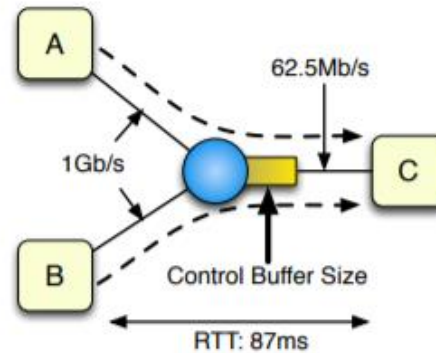
N. Beheshti, "Tiny Buffers for Electronic and Optical Routers," 2009. Online" <https://tinyurl.com/y8uug534>

Introduction to Token Bucket Algorithm

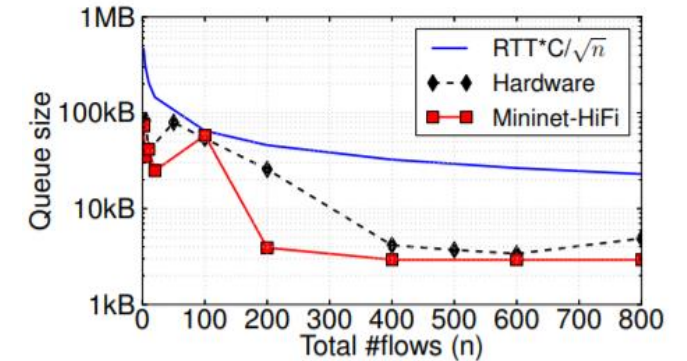
- Example: what is the ideal buffer size in a router, so that the utilization of a bottleneck link is maximized?



2009 Hardware experiment



2012 Mininet Experiment



Red: Mininet results (2012, after Mininet was created)
 Black: Internet2 backbone results (Los Angeles – Houston)
 Blue: Theoretical upper bound

N. Beheshti, "Tiny Buffers for Electronic and Optical Routers," 2009. Online" <https://tinyurl.com/y8uug534>

Nikhil Handigo et al., "Reproducible Network Experiments Using Container-Based Emulation," 2012. Online: <https://tinyurl.com/ybeebb4v>

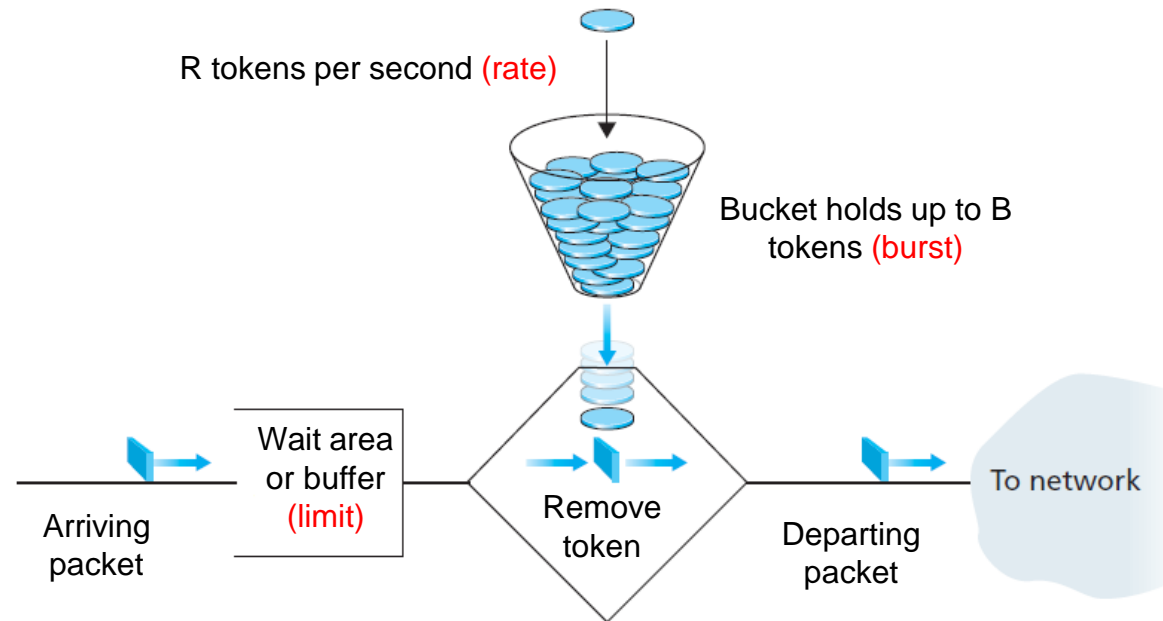
Reproducing Network Research Website. Online: <https://reproducingnetworkresearch.wordpress.com/>

Introduction to Token Bucket Algorithm

- The Token Bucket algorithm is used to limit the average rate (bps) and burst size of packets (or bits/bytes) on a link
 - Average rate gives the bps at which a traffic can be sent into the link
 - Burst size gives the maximum bps (the “burst”) that can be sent into the link over an extremely short interval of time

Introduction to Token Bucket Algorithm

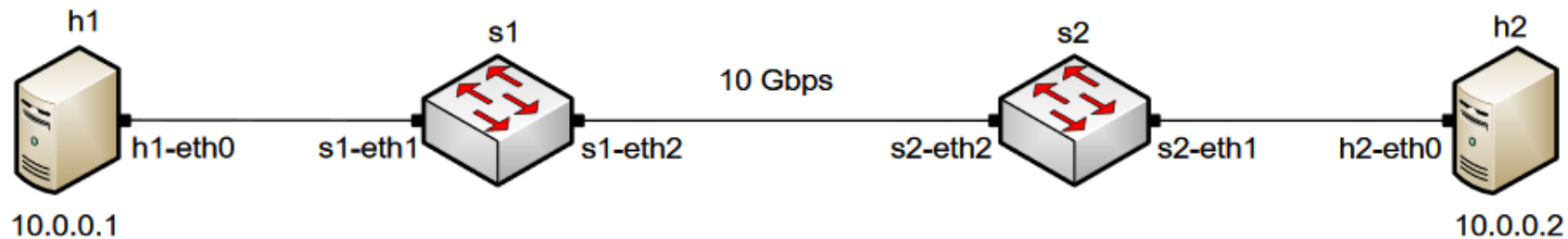
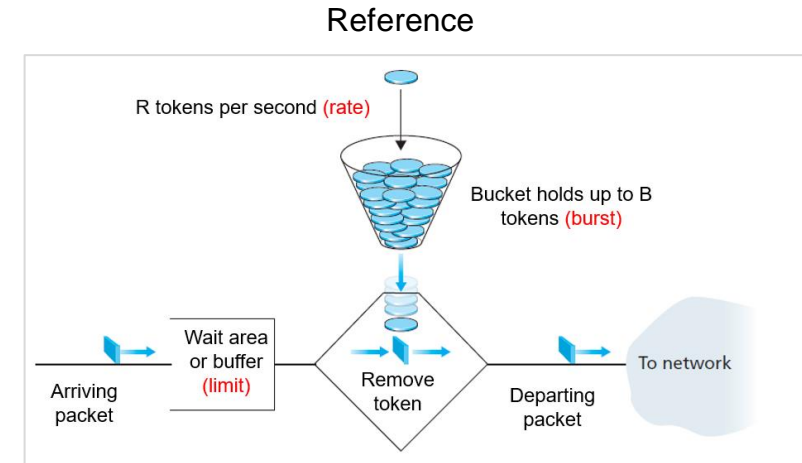
- The Token Bucket algorithm is used to limit the average rate (bps) and burst size of packets (or bits/bytes) on a link
 - R tokens are generated at a constant rate
 - A leaky bucket holds up to B tokens
 - A packet is sent to the network only if a token is available; one token is then removed from the bucket
 - If no token is available, the packet must wait in the buffer, until a token is available



Rate Limit on Switches

Rate Limiting on Switches

- The first figure shows the network topology
- The *tb*f parameters are the following:
 - rate: 10gbit (in bits per seconds)
 - burst: 5,000,000 (in bytes)
 - limit: 15,000,000 (in bytes)



```
admin@admin-pc: ~  
File Actions Edit View Help  
admin@admin-pc: ~  
admin@admin-pc:~$ sudo tc qdisc add dev s1-eth2 root tbf rate 10gbit burst 5000000 limit 15000000  
[sudo] password for admin:  
admin@admin-pc:~$
```

Identify Switch Interfaces

- The *ifconfig* command is used to display information related to network interfaces

```
admin@admin-pc: ~
File Actions Edit View Help
admin@admin-pc: ~
admin@admin-pc:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    ether 00:50:56:ae:1c:cd txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 198 bytes 32455 (32.4 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 19 base 0x2000

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 4484 bytes 297536 (297.5 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4484 bytes 297536 (297.5 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

s1-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::80d7:ff:fe14:b1a4 prefixlen 64 scopeid 0x20<link>
    ether 82:d7:00:14:b1:a4 txqueuelen 1000 (Ethernet)
    RX packets 216964 bytes 14138070762 (14.1 GB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 213105 bytes 14068378 (14.0 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

```
s1-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::4411:aff:fef7:ff3e prefixlen 64 scopeid 0x20<link>
    ether 46:11:0a:f7:ff:3e txqueuelen 1000 (Ethernet)
    RX packets 213075 bytes 14064944 (14.0 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 216993 bytes 14138074110 (14.1 GB)

s2-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::fc8d:6dff:fe08:76b0 prefixlen 64 scopeid 0x20<link>
    ether fe:8d:6d:08:76:b0 txqueuelen 1000 (Ethernet)
    RX packets 213045 bytes 14061510 (14.0 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 217023 bytes 14138077544 (14.1 GB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

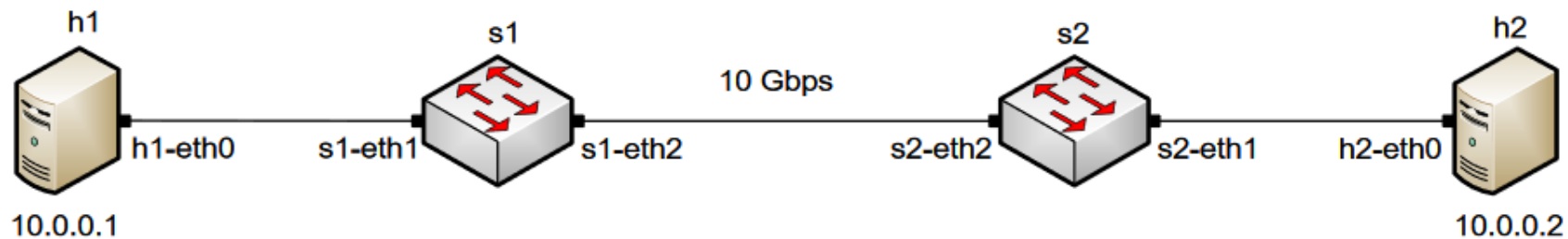
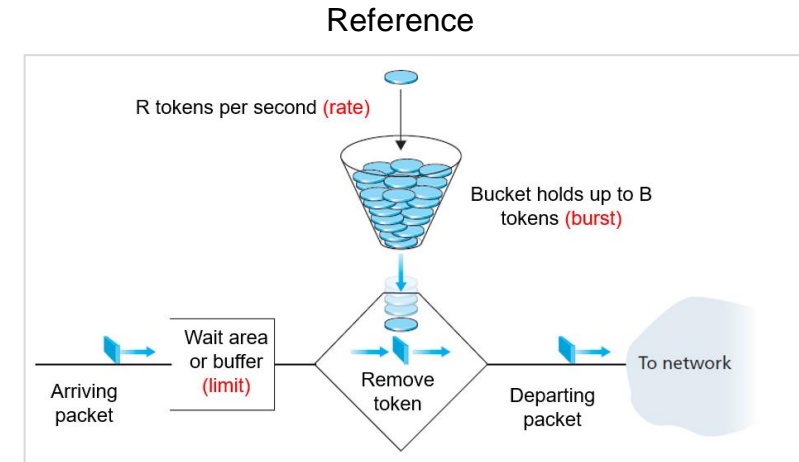
s2-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::ec53:8ff:fe09:8cb7 prefixlen 64 scopeid 0x20<link>
    ether ee:53:08:09:8c:b7 txqueuelen 1000 (Ethernet)
    RX packets 216993 bytes 14138074110 (14.1 GB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 213075 bytes 14064944 (14.0 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

admin@admin-pc:~$
```

Rate Limit on End Devices

Rate Limiting on End-hosts

- The first figure shows the network topology
- The *tb*f parameters are the following:
 - rate: 10gbit (in bits per seconds)
 - burst: 5,000,000 (in bytes)
 - limit: 15,000,000 (in bytes)



```
Host: h1
root@admin-pc:~# sudo tc qdisc add dev h1-eth0 root tbf rate 10gbit burst 5000000 limit 15000000
root@admin-pc:~#
```

Verification

- The user can now verify the previous configuration by using the iperf3 tool to measure throughput

```
root@admin-pc:~# iperf3 -s
-----
Server listening on 5201
-----
```

Server (h2)

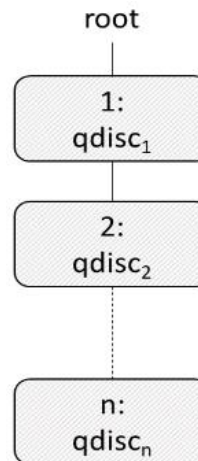
```
root@admin-pc:~# iperf3 -c 10.0.0.2
Connecting to host 10.0.0.2, port 5201
[ 15] local 10.0.0.1 port 34924 connected to 10.0.0.2
[ ID] Interval           Transfer     Bitrate
[ 15]  0.00-1.00    sec   1.12 GBytes   9.62 Gbits/sec
[ 15]  1.00-2.00    sec   1.11 GBytes   9.57 Gbits/sec
[ 15]  2.00-3.00    sec   1.11 GBytes   9.56 Gbits/sec
[ 15]  3.00-4.00    sec   1.11 GBytes   9.56 Gbits/sec
[ 15]  4.00-5.00    sec   1.11 GBytes   9.57 Gbits/sec
[ 15]  5.00-6.00    sec   1.11 GBytes   9.56 Gbits/sec
[ 15]  6.00-7.00    sec   1.11 GBytes   9.56 Gbits/sec
[ 15]  7.00-8.00    sec   1.11 GBytes   9.56 Gbits/sec
[ 15]  8.00-9.00    sec   1.11 GBytes   9.56 Gbits/sec
[ 15]  9.00-10.00   sec   1.11 GBytes   9.57 Gbits/sec
-----
[ ID] Interval           Transfer     Bitrate
[ 15]  0.00-10.00    sec   11.1 GBytes   9.57 Gbits/sec
[ 15]  0.00-10.04    sec   11.1 GBytes   9.53 Gbits/sec
-----
iperf Done.
root@admin-pc:~#
```

Client (h1)

Combining NETEM and TBF

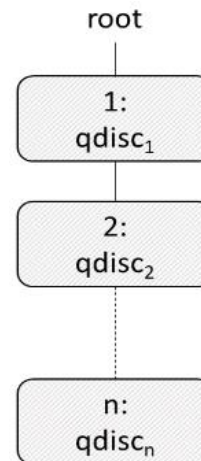
Combining Qdiscs

- The first qdisc (qdisc1) is attached to the root label
- Then, subsequent qdiscs can be attached to their parents by specifying the correct label



Combining NETEM and TBF

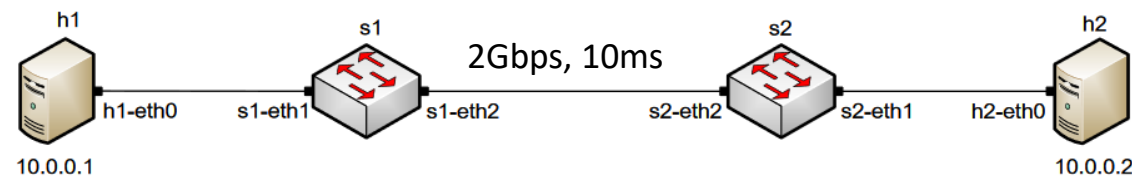
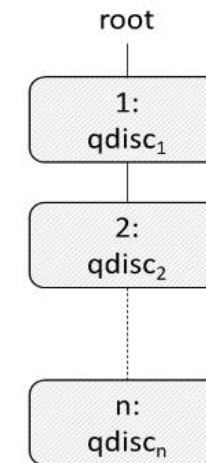
- NETEM is used to introduce delay, jitter, packet corruption, etc.
- TBF is used to limit the rate
- We can combine multiple queueing disciplines ('link features') and activate them at the same time



Combining NETEM and TBF

- The command below activates NETEM on interface s1-eth2 to emulate delay
- The new keyword in this command is *handle* and its value reflects the number shown above each qdisc in the figure on the right
- This means that our NETEM qdisc is attached to the root with the handle1

```
admin@admin-pc: ~  
File Actions Edit View Help  
admin@admin-pc: ~  
admin@admin-pc:~$ sudo tc qdisc add dev s1-eth2 root handle 1: netem delay 10ms  
admin@admin-pc:~$
```



Combining NETEM and TBF

- Now this command adds the second rule which applies rate limiting using tbf
- The new keyword in this command is *parent* and its value reflects the number of the parent qdisc
- The tbf qdisc is attached to the NETEM qdisc (configured in the previous slide)
- TBF has a handle 2: if we decide to attach further qdiscs

```
admin@admin-pc: ~  
File Actions Edit View Help  
admin@admin-pc: ~  
admin@admin-pc:~$ sudo tc qdisc add dev s1-eth2 parent 1: handle 2: tbf rate 2gbit  
burst 1000000 limit 2500000  
admin@admin-pc:~$
```

