



QoS, HTB, Multicore

RouterOS v6.xx



Valens Riyadi (Citraweb)

info@mikrotik.co.id

About Me



Valens Riyadi, Citraweb (ID)

MikroTik Certified Engineer

(MTCNA, MTCWE, MTCRE, MTCTCE, MTCUME, MTCINE)

MikroTik Certified Trainer since 2004

MikroTik Certified Consultant

MikroTik Academy Coordinator

Citra.net.id WISP CEO

Manager for IDNIC (Indonesia National Internet Registry) 2009 – May 2015

IT Expert on Disaster Relief

Proud member of “Routed World” community

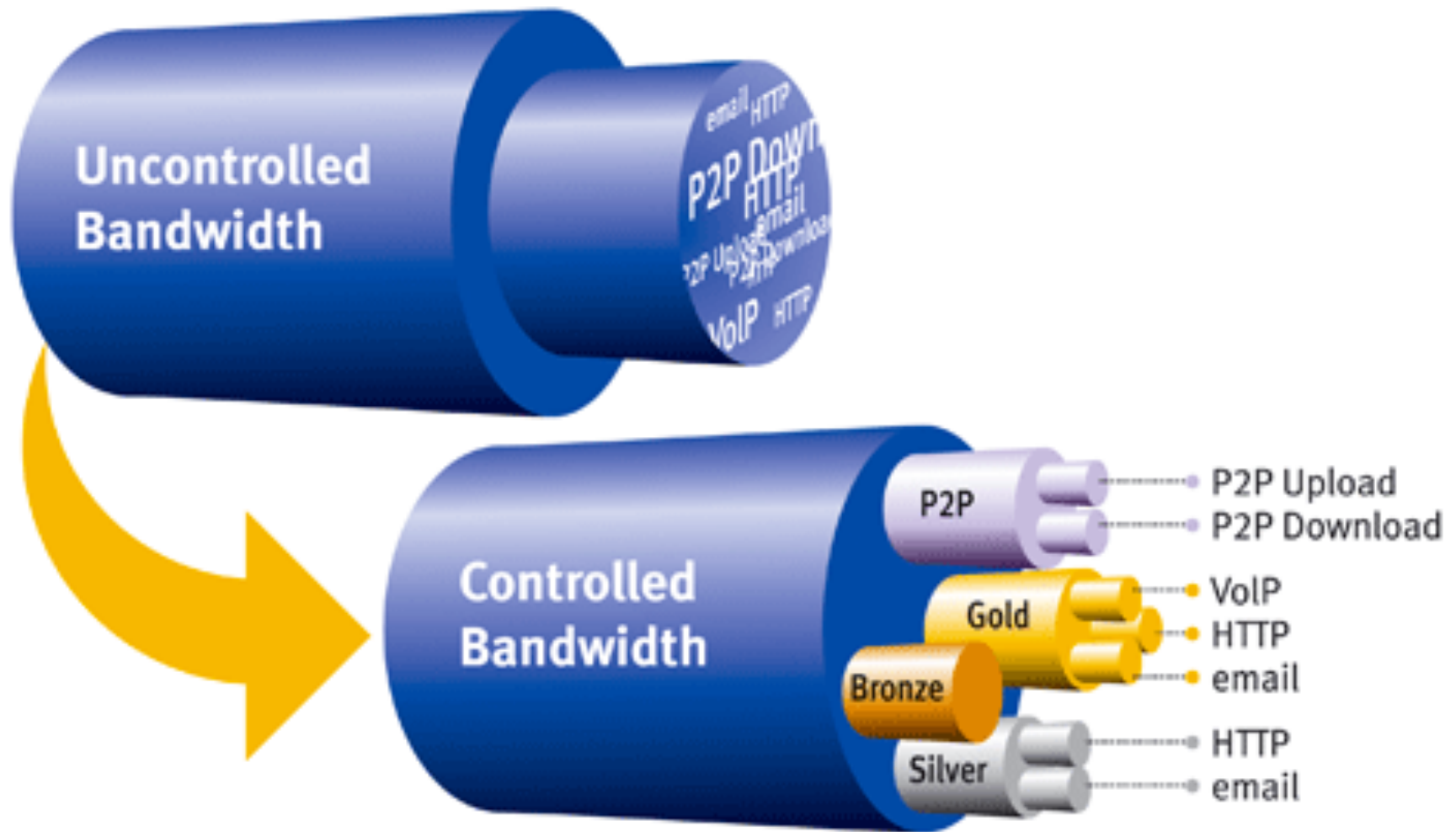
Recommended Resources

- CCR Status Update (Janis Megis – Video)
<http://tiktube.com/video/GJil3aqliCGJClqqpGnrGznrClGoJGJo=>
- Dynamic QoS on RouterOS v6 (Valens Riyadi)
<http://mum.mikrotik.com/presentations/IT14/valens.pdf>
- QoS on RouterOSv6 (Valens Riyadi)
<http://mum.mikrotik.com/presentations/HR13/valens.pdf>
- HTB vs PCQ (Valens Riyadi)
<http://mum.mikrotik.com/presentations/HU11/valens.pdf>
- QoS and Traffic Priorities (Janis Megis)
http://mum.mikrotik.com/presentations/CZ09/QoS_Megis.pdf
- HTB QoS (Valens Riyadi)
<http://mum.mikrotik.com/presentations/US09/Valens-MUM2009USA.pdf>

QoS concept

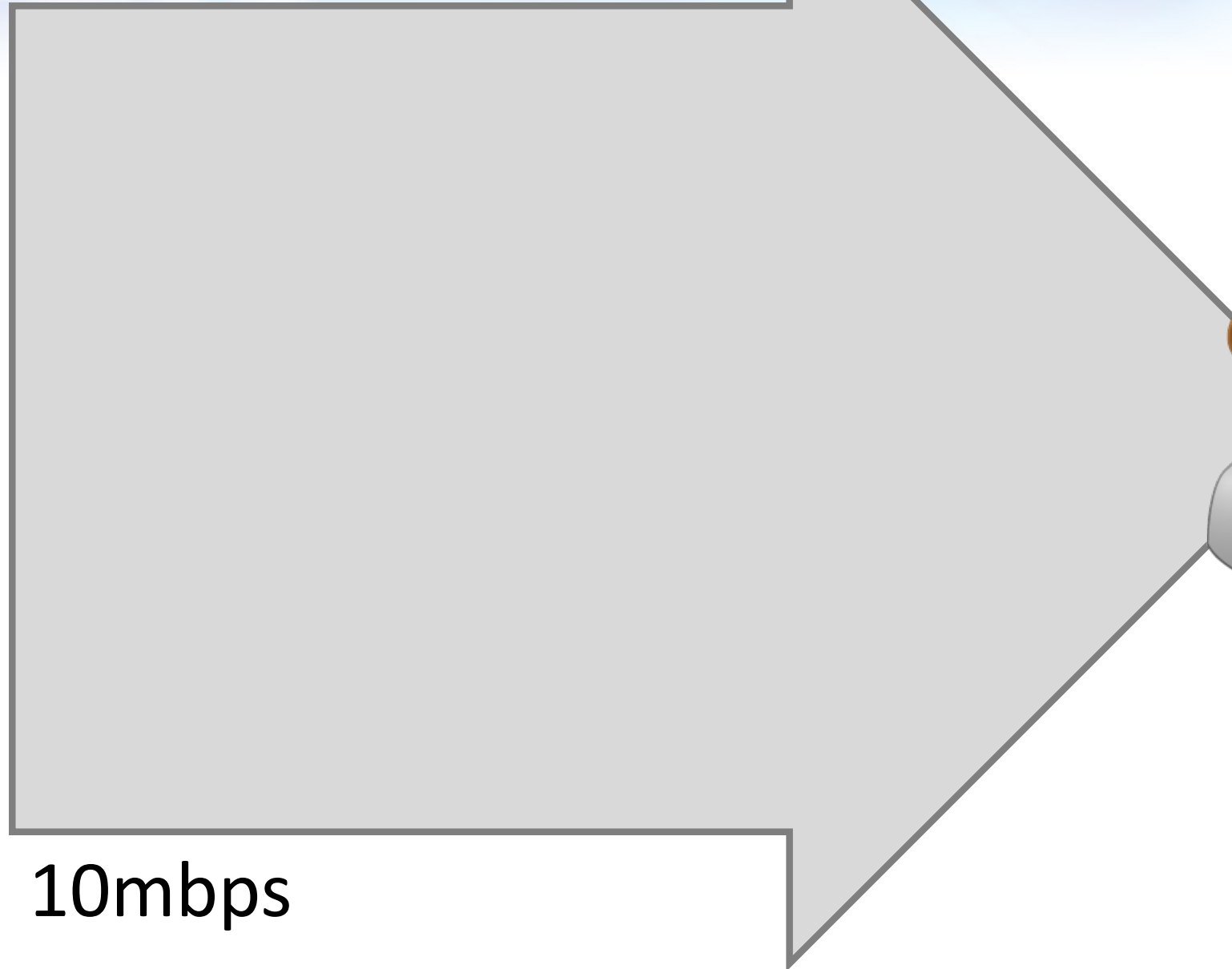
Fungsi untuk mengukur dan mengontrol komunikasi data (trafik, paket) pada jaringan, untuk menghindari overload, atau ada yang memonopoli jaringan, yang berakibat pada congestion dan performa jaringan yang buruk.

Untuk apa?

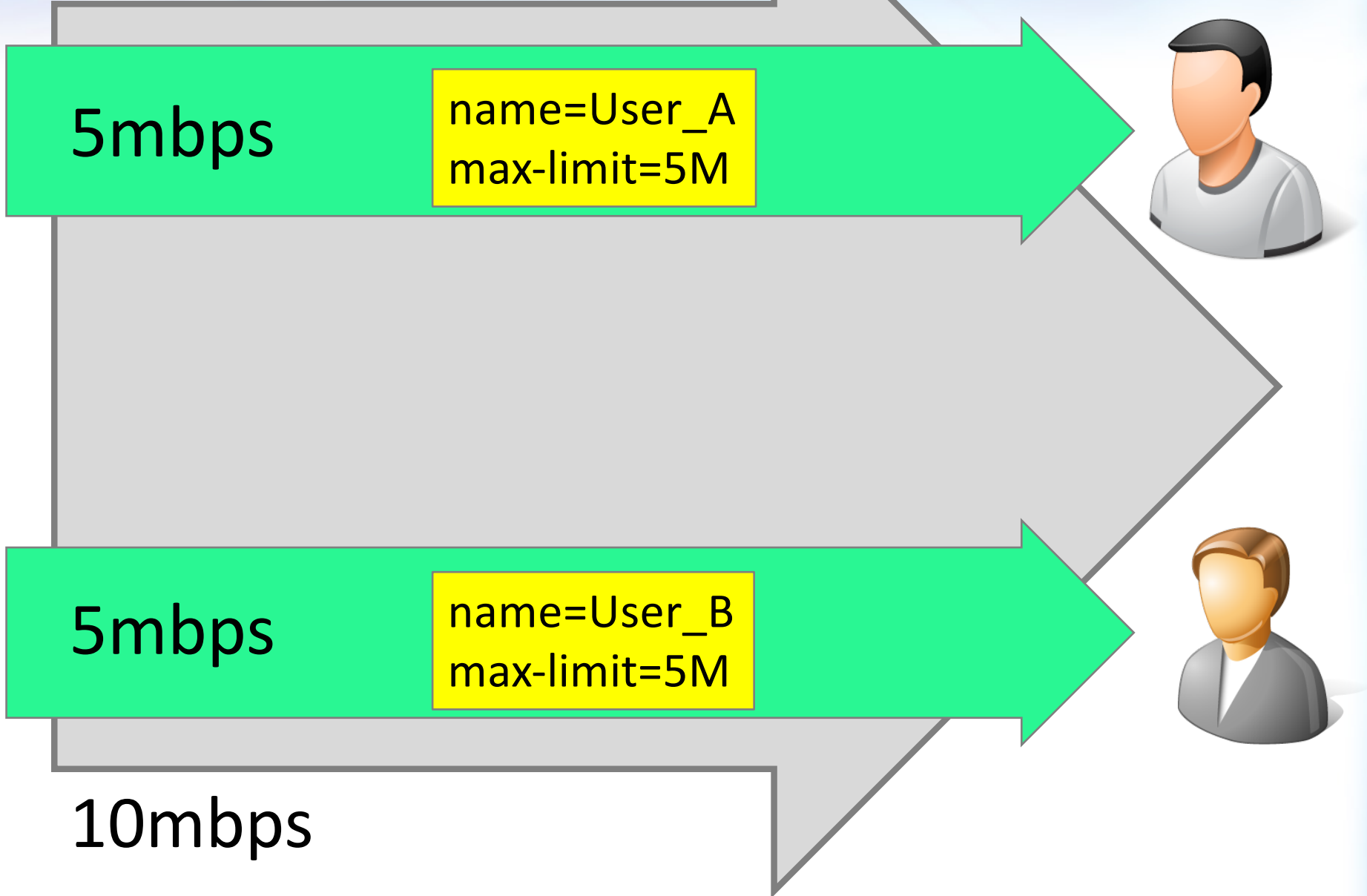


QoS on RouterOS

- MikroTik RouterOS adalah salah satu bandwidth management paling canggih, jika dibandingkan dengan merk lainnya.
- Mengapa?
 - Konfigurasi HTB yang mendalam
 - Double limitation + Burst
 - Ada banyak pilihan dan parameter → packet-mark



10mbps



5mbps

name=User_A
max-limit=5M



5mbps

name=User_B
max-limit=5M



10mbps

New Simple Queue

General

Advanced


Statistics

Traffic

Total

Total Statistics

Name:

Target Address: 

Target Upload

Target Download

Max Limit:



bits/s

—▲— Burst

Burst Limit:



bits/s

Burst Threshold:



bits/s

Burst Time:

s

—▼— Time

Max-limit

Besarnya kapasitas maksimum yang bisa dicapai oleh user tertentu



Bagaimana jika user A tidak menggunakan internet, user B dapat menggunakan semua sisa bandwidth yang tersedia?

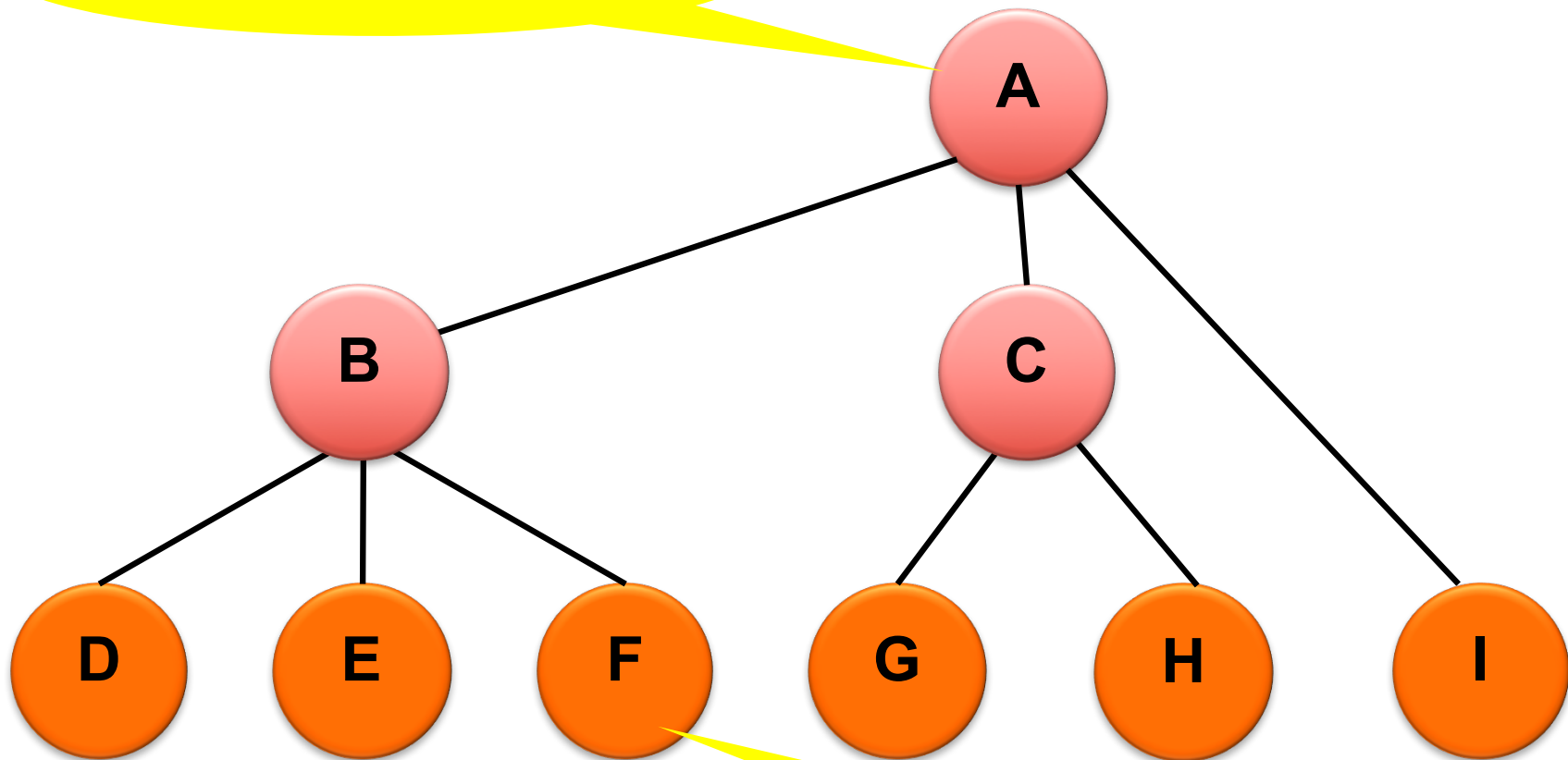


HTB

- Sebagian besar implementasi QoS pada RouterOS berdasarkan Hierarchical Token Bucket (HTB)
- HTB memungkinkan kita membuat struktur queue berjenjang dan menentukan relasi antara parent dengan child, ataupun antar sesama child.
- RouterOS v6 mengenal 1 virtual HTBs (global), dan satu di akhir setiap interface

HTB Sample

Parent/ inner queue



Child = leaf queue

HTB Features

- Hirarki
 - Di routerOS, kita bisa membuat hingga 8 level hirarki
- Grouping
 - Kita bisa mengelompokkan beberapa client di dalam satu parent
 - Client yang satu bisa meminjam bandwidth dari client lainnya, jika dibutuhkan dan tersedia
- Tiap leaf queue bisa memiliki setting yang berbeda. Semua leaf akan dianggap sama, di hirarki paling bawah

HTB Implementation Example

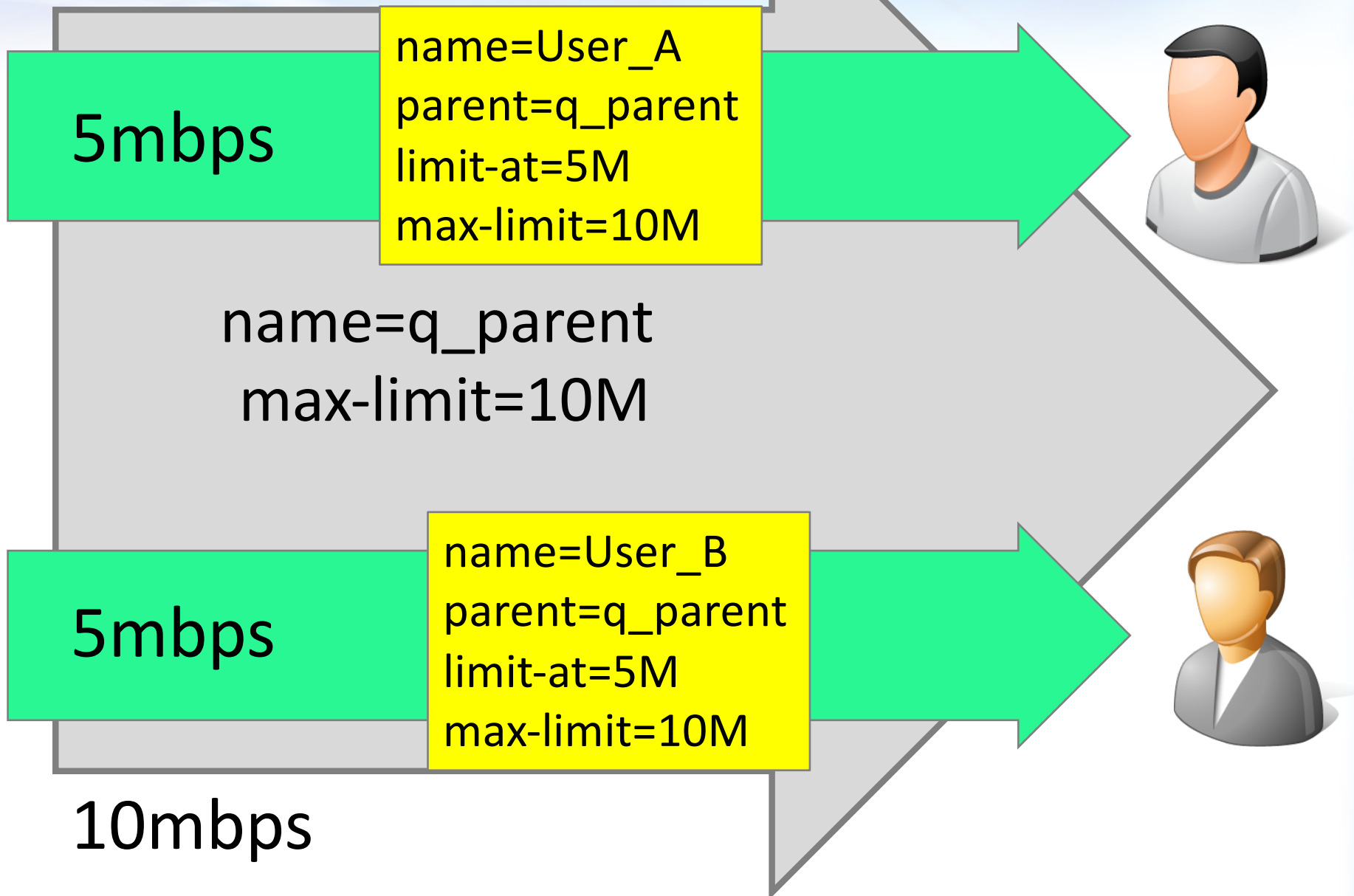
Queue List

Simple Queues Interface Queues Queue Tree Queue Types

+ - ✓ ✗ 📄 ⚙️

Reset Counters 00 Reset All Counters Find

Name	Parent	Packet Marks	Limit At (bits/s)	Max Limit (bits/s)	Avg. R...	Queued Bytes	Bytes	Packets
queue_0	ether2			10M	0 bps	0 B	0 B	0
queue_1	queue_0	packet_1551	468904	9800k	0 bps	0 B	0 B	0
queue_2	queue_1	packet_9220	268289	9600k	0 bps	0 B	0 B	0
queue_17	queue_2	packet_8507	613074	6600k	0 bps	0 B	0 B	0
queue_41	queue_17	packet_8440	371117	1800k	0 bps	0 B	0 B	0
queue_6	queue_2	packet_2299	687353	8800k	0 bps	0 B	0 B	0
queue_18	queue_6	packet_1165	366627	6400k	0 bps	0 B	0 B	0
queue_23	queue_18	packet_1093	538294	5400k	0 bps	0 B	0 B	0
queue_3	queue_0	packet_3333	166813	9400k	0 bps	0 B	0 B	0
queue_16	queue_3	packet_6309	529294	6800k	0 bps	0 B	0 B	0
queue_27	queue_16	packet_8970	562428	4600k	0 bps	0 B	0 B	0
queue_46	queue_27	packet_1154	420425	800k	0 bps	0 B	0 B	0
queue_31	queue_16	packet_8523	563538	3800k	0 bps	0 B	0 B	0
queue_37	queue_16	packet_8389	376173	2600k	0 bps	0 B	0 B	0
queue_40	queue_37	packet_8521	704484	2M	0 bps	0 B	0 B	0
queue_42	queue_40	packet_2889	430111	1600k	0 bps	0 B	0 B	0
queue_39	queue_16	packet_8281	227458	2200k	0 bps	0 B	0 B	0
queue_22	queue_3	packet_9689	370291	5600k	0 bps	0 B	0 B	0
queue_43	queue_22	packet_9101	607074	1400k	0 bps	0 B	0 B	0
queue_8	queue_3	packet_3057	644987	8400k	0 bps	0 B	0 B	0
queue_9	queue_3	packet_9444	433143	8200k	0 bps	0 B	0 B	0
queue_35	queue_9	packet_6885	149412	3M	0 bps	0 B	0 B	0
queue_44	queue_9	packet_6940	508058	1200k	0 bps	0 B	0 B	0
queue_4	queue_0	packet_1485	587640	9200k	0 bps	0 B	0 B	0
queue_5	queue_4	packet_8908	661059	9M	0 bps	0 B	0 B	0
queue_13	queue_5	packet_8132	746955	7400k	0 bps	0 B	0 B	0
queue_26	queue_13	packet_8397	692964	4800k	0 bps	0 B	0 B	0
queue_34	queue_13	packet_1227	483167	3200k	0 bps	0 B	0 B	0
queue_36	queue_13	packet_7635	412515	2800k	0 bps	0 B	0 B	0



Queue List

Simple Queues

Interface Queues

Queue Tree

Queue Types



Reset Counters

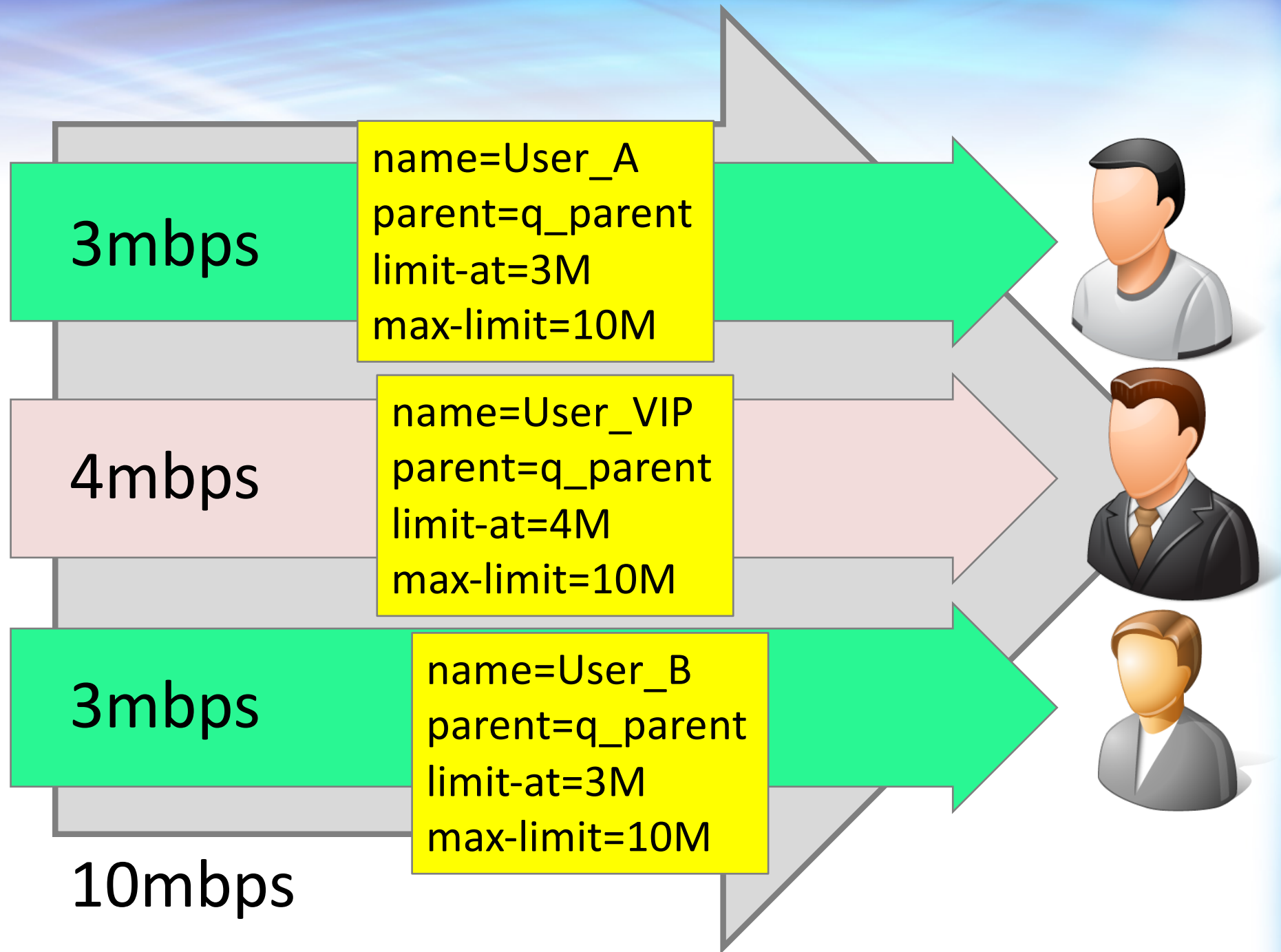
Reset All Counters

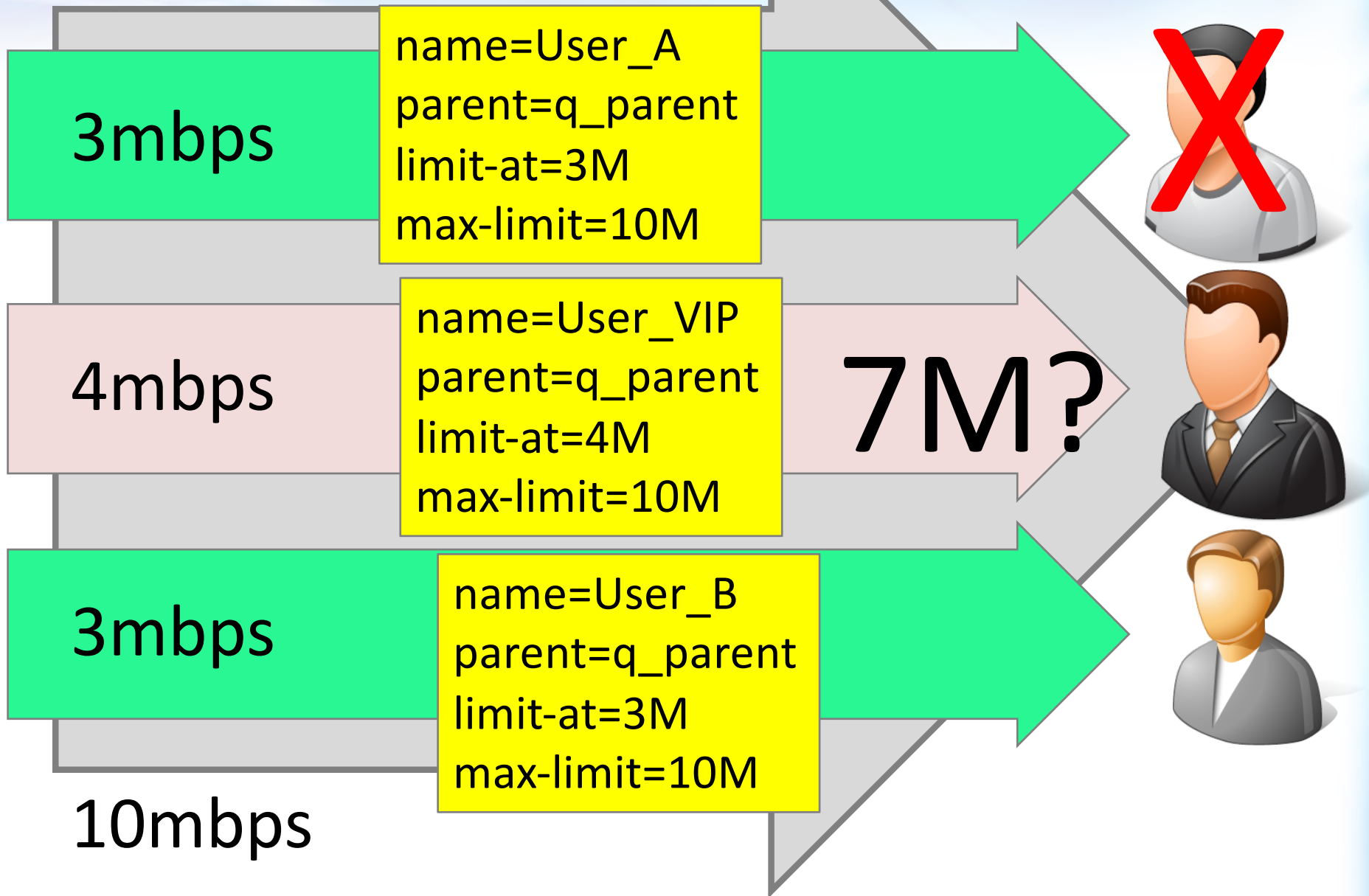
#	Name	Target Address	Rx Max Limit	Tx Max Limit	Rx Limit At	Tx Limit At	Parent
0	q_parent	192.168.1.0/24	10M	10M	unlimited	unlimited	none
1	queue_A	192.168.1.1	10M	10M	5M	5M	q_parent
2	queue_B	192.168.1.2	10M	10M	5M	5M	q_parent

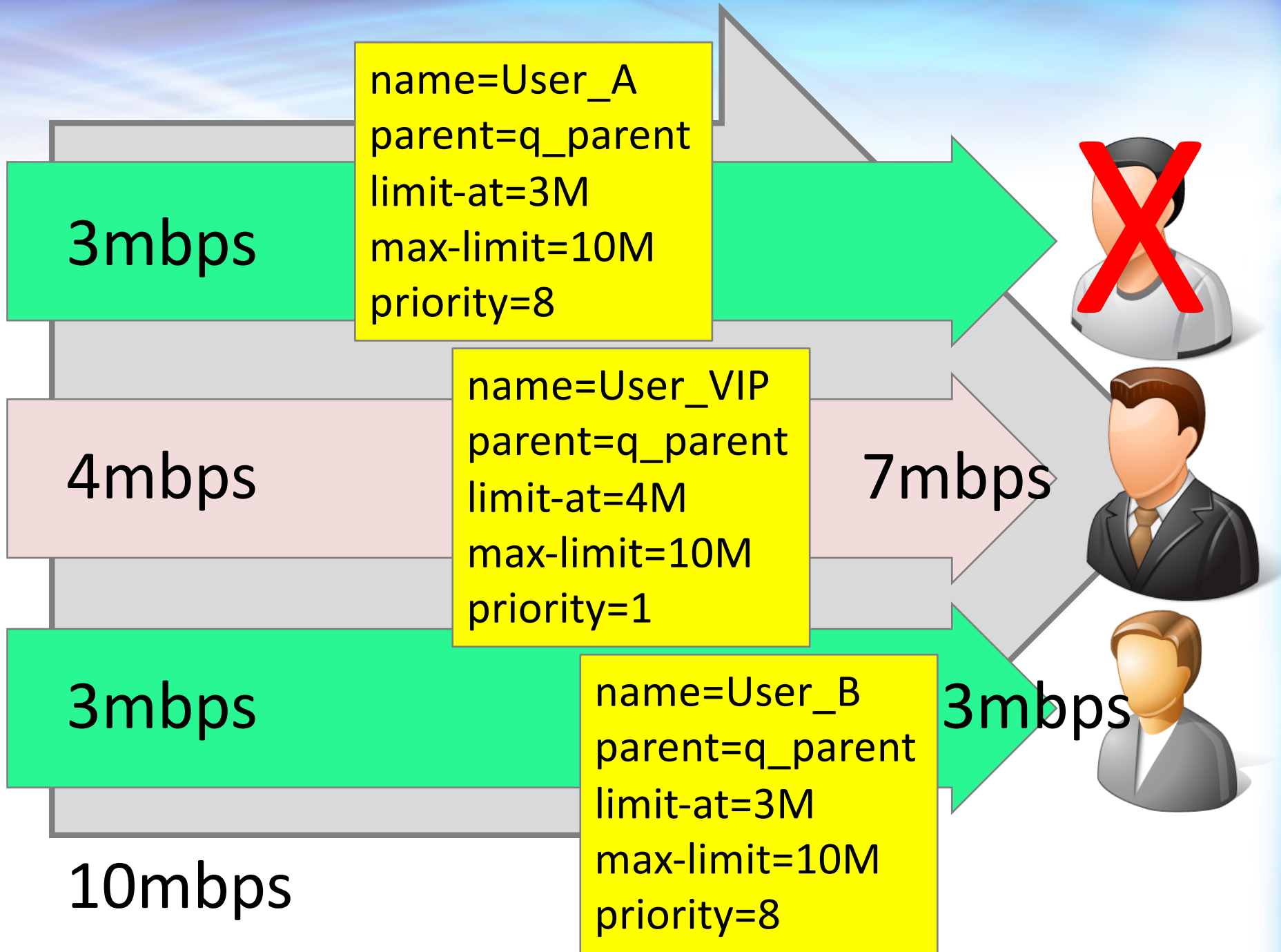
Limit-at

Besarnya kapasitas yang dijamin dapat diterima oleh user tertentu, selama bandwidth masih tersedia









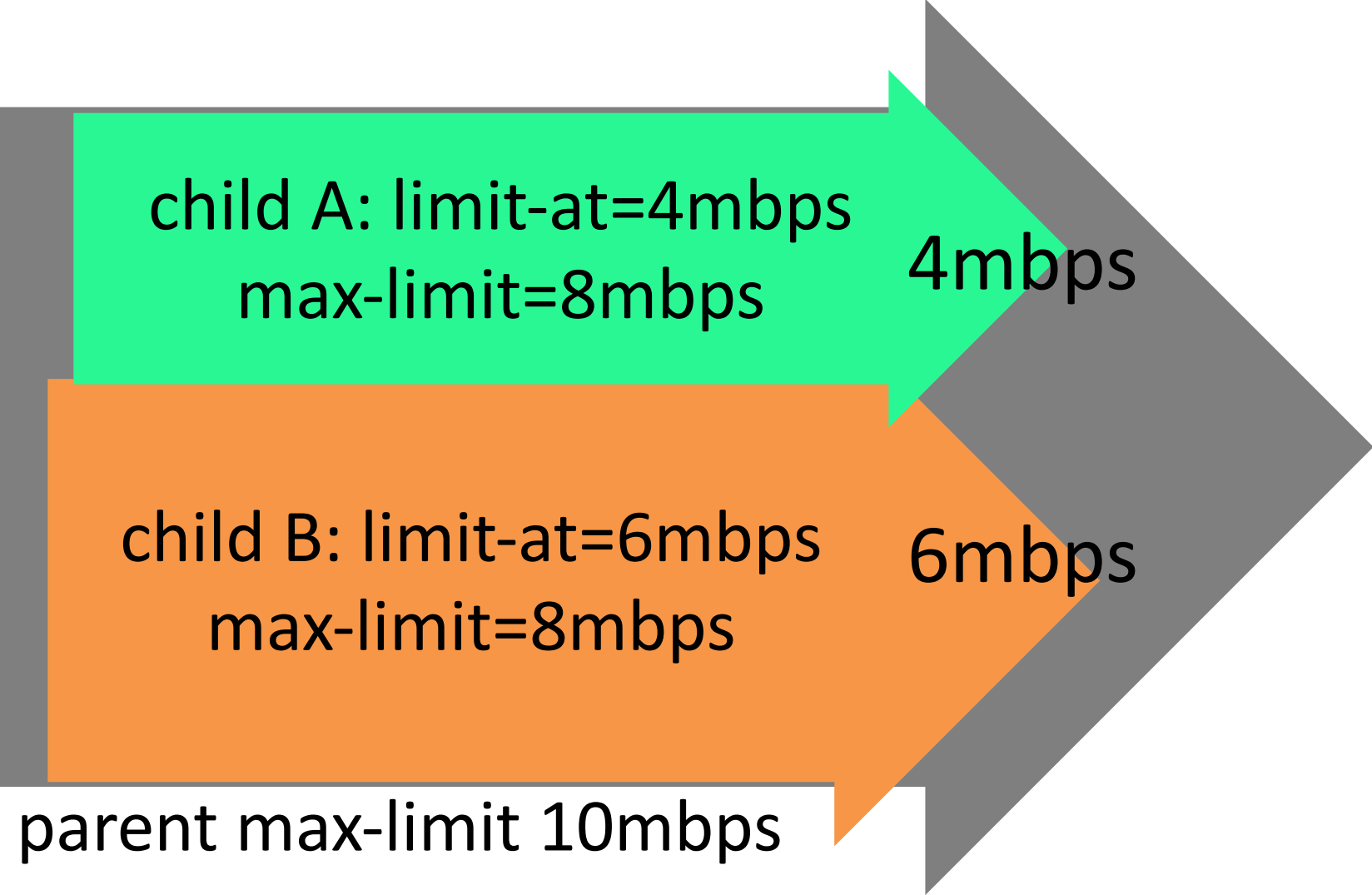
Priority

Memberikan prioritas untuk user tertentu dibanding user lainnya, jika semua limit-at sudah terpenuhi.



Queue Parameter

- limit-at (CIR)
- max-limit (MIR)
- burst (threshold, limit, time)
- queue type (FIFO, RED, SFQ, PCQ)
- parent



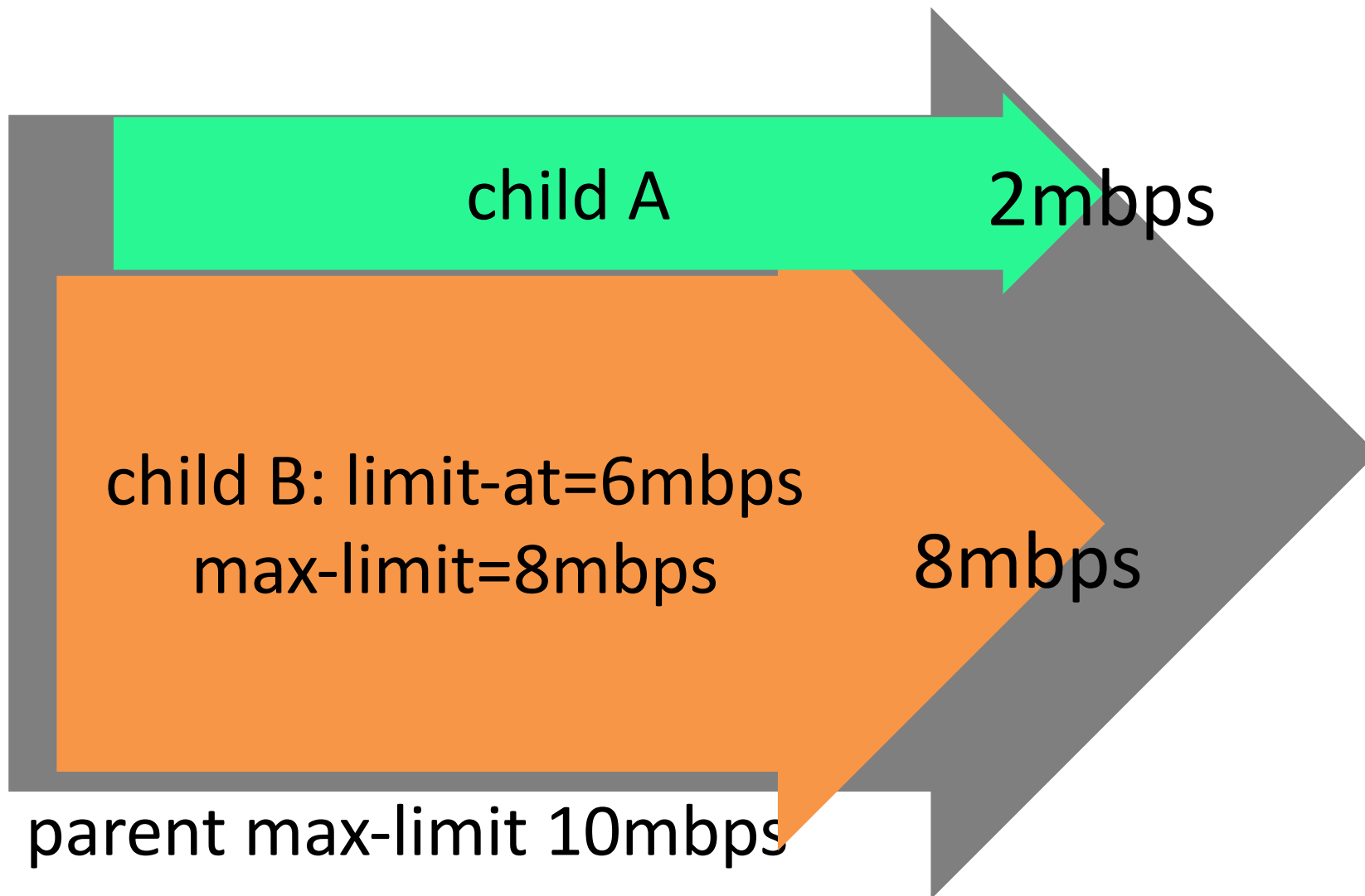
child A: limit-at=4mbps
max-limit=8mbps

4mbps

child B: limit-at=6mbps
max-limit=8mbps

6mbps

parent max-limit 10mbps



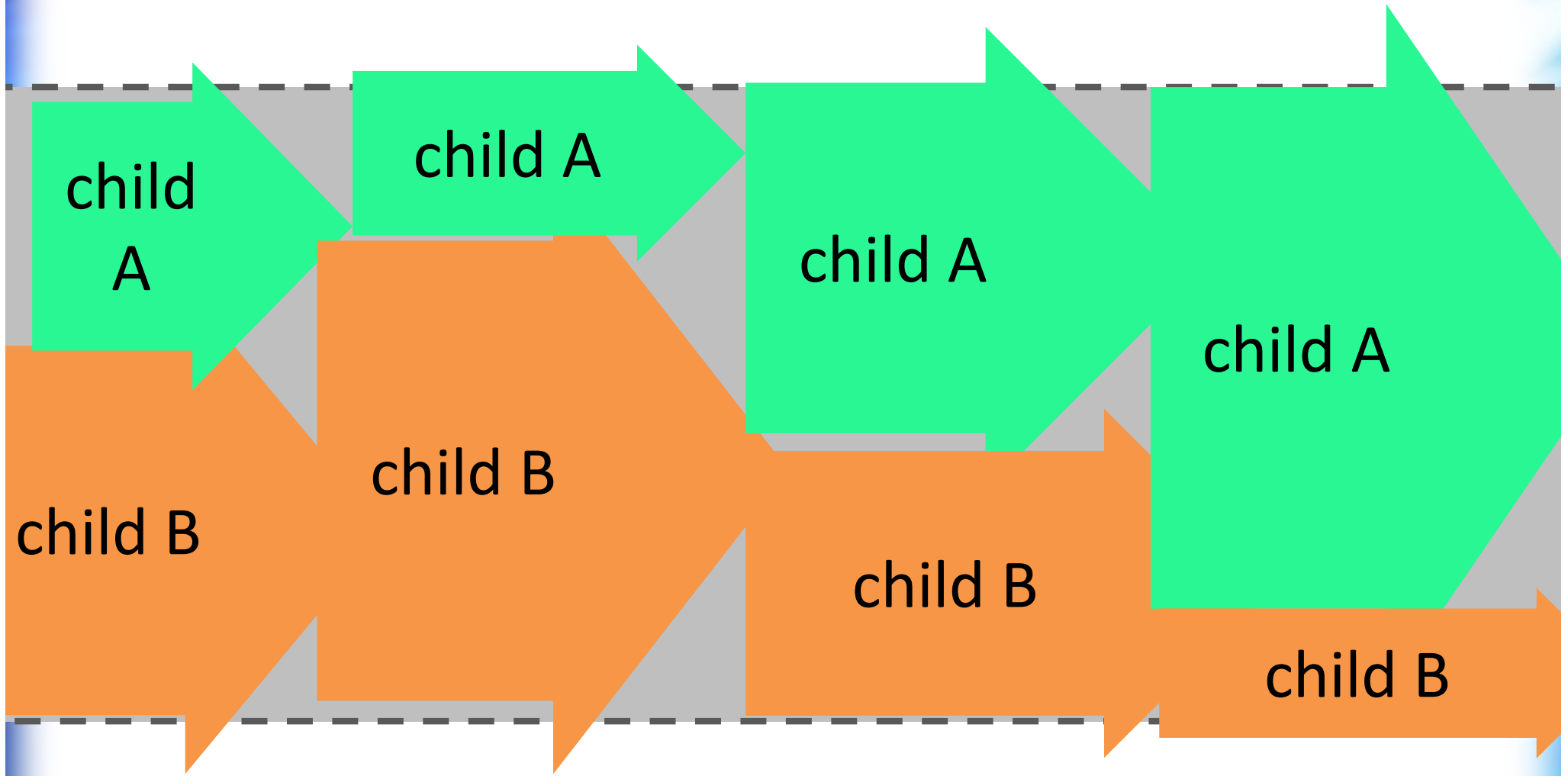
child A : 0mbps

child B: limit-at=6mbps
max-limit=8mbps

8mbps

parent max-limit 10mbps

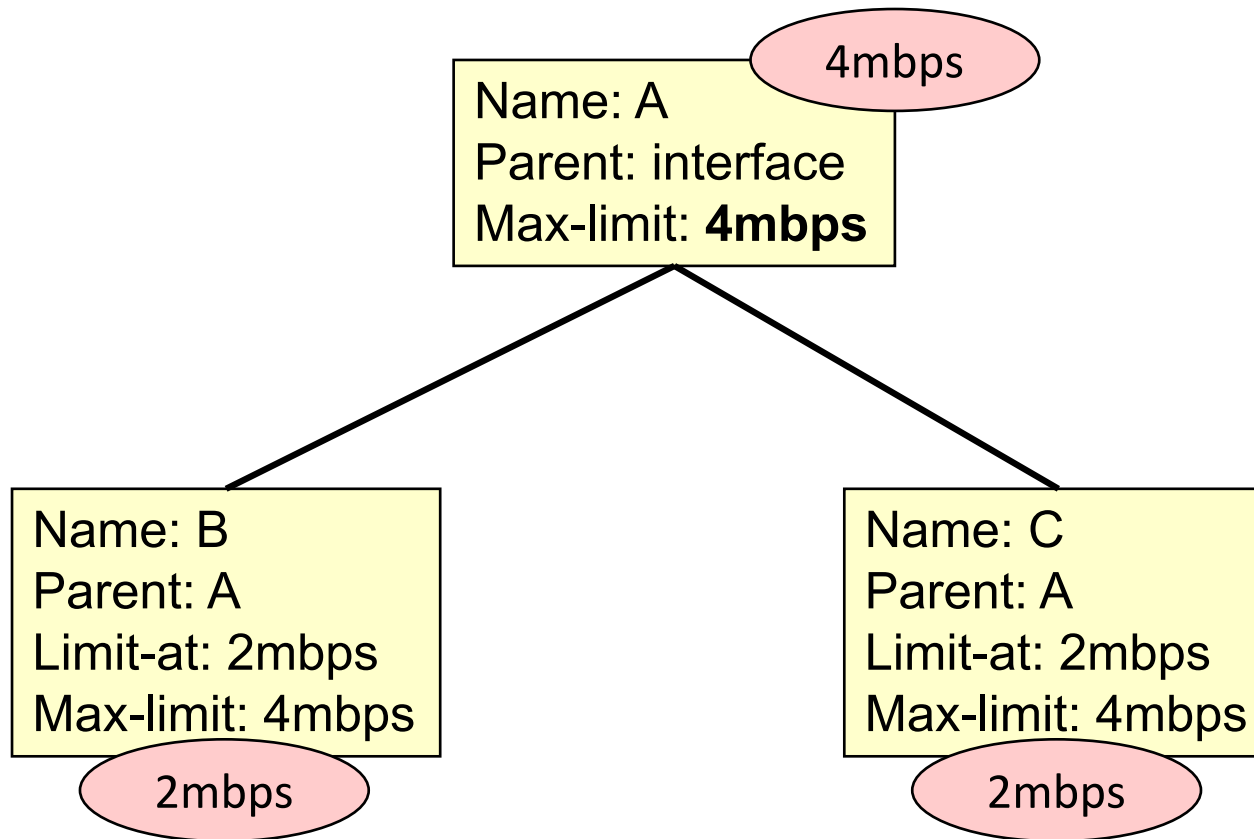
without parent, with 10mbps link





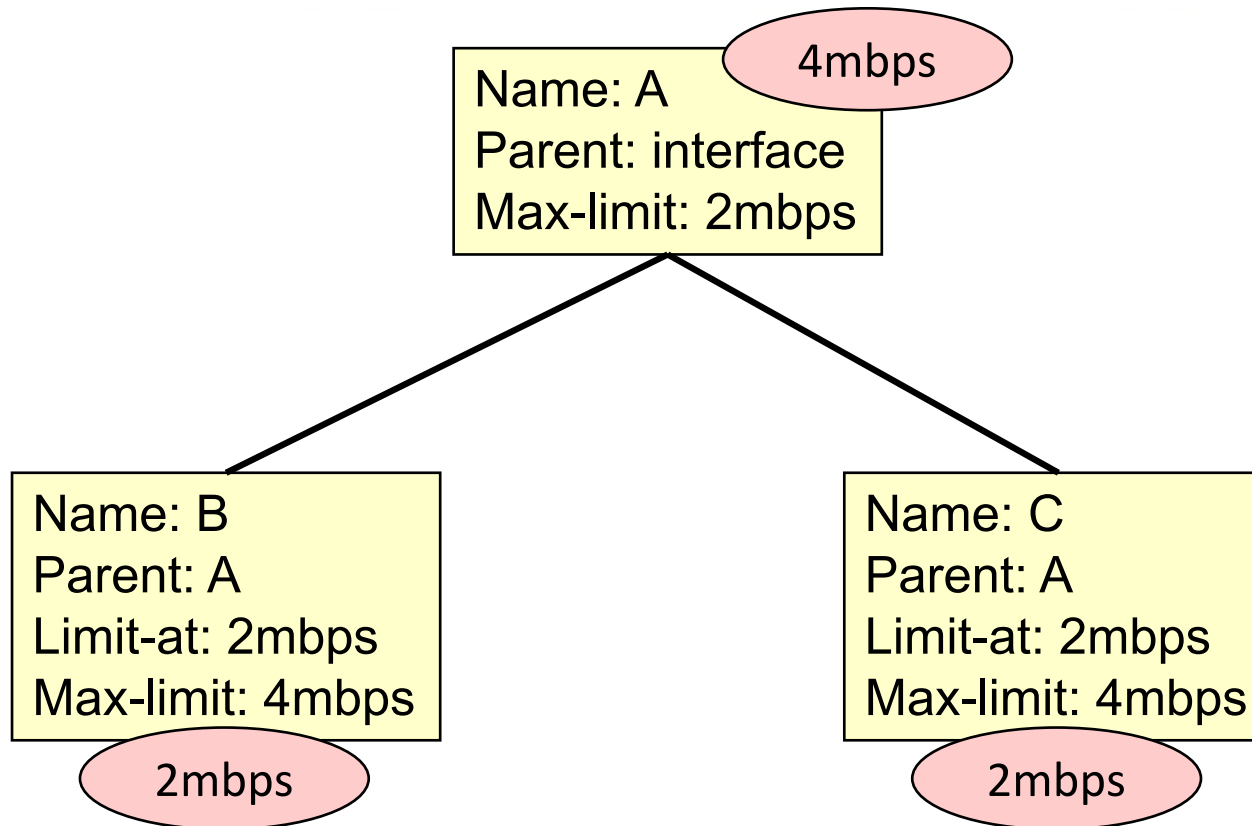
Tanpa parent,
parameter limit-at dan
priority akan diabaikan.

HTB Distribution (1)



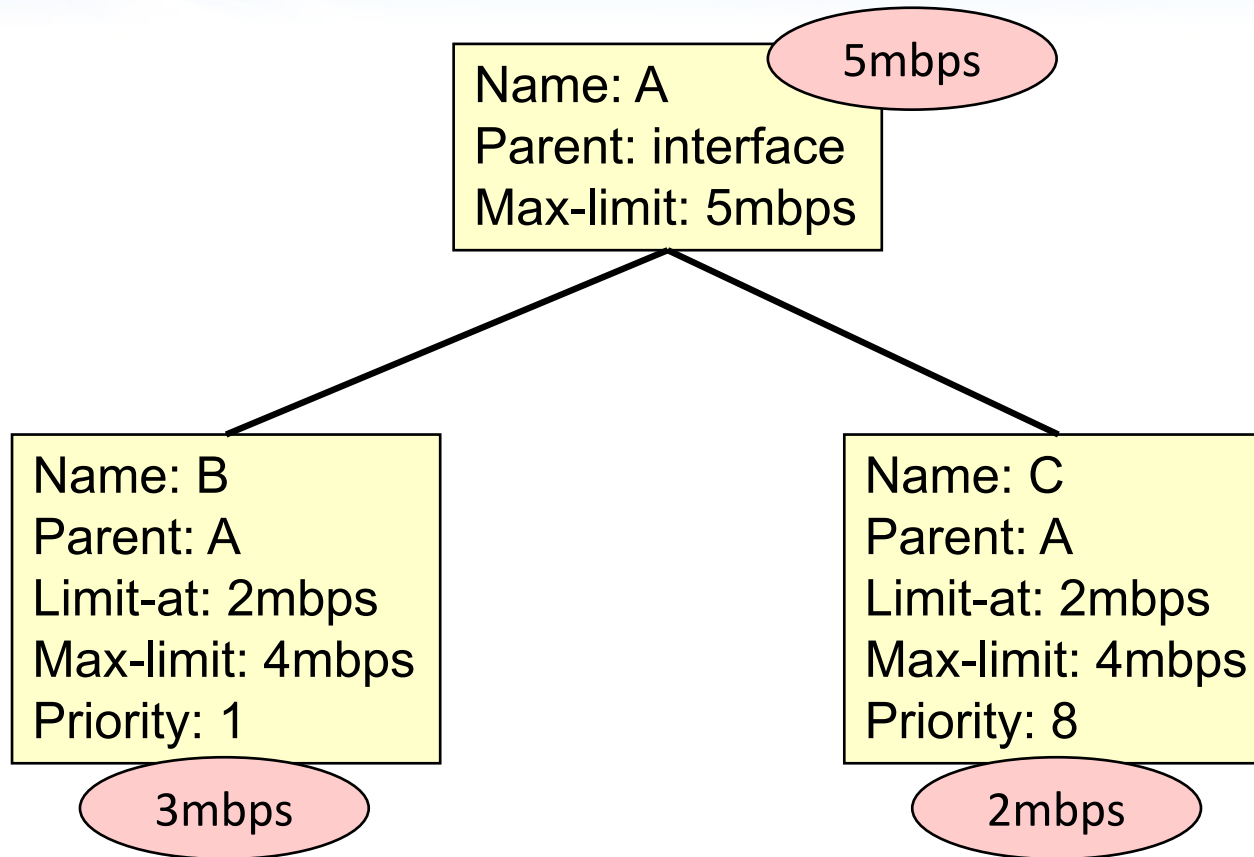
If Child B and C use the link, they will have 2mbps each, but if child C is not using it, child B will get 4 mbps.

HTB Distribution (2)



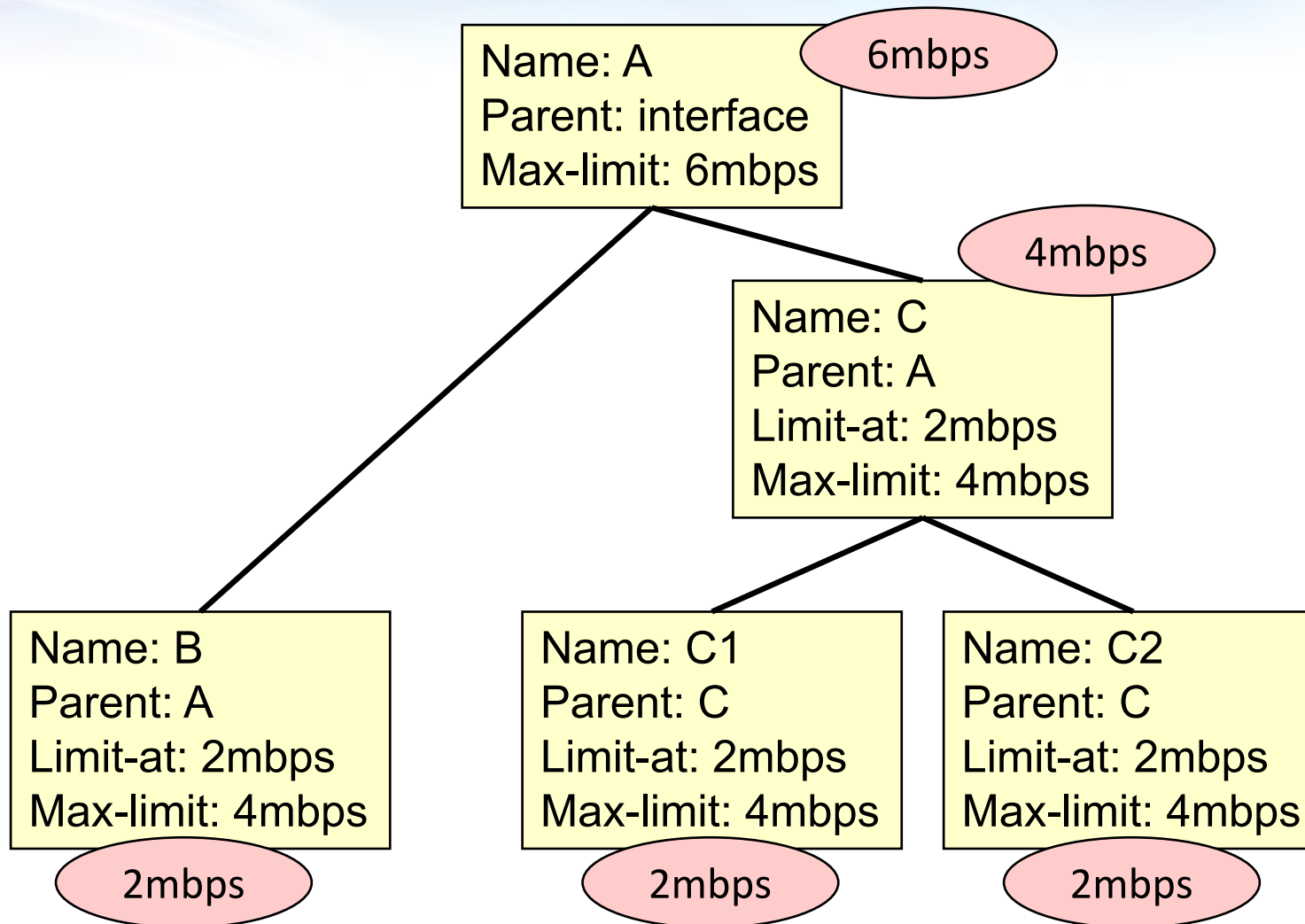
Even the max-limit of the parent is 2mbps, but child B and child C will still get 2 mbps each (total of 4 mbps). If C is not use the link, B will still get 2 mbps, can not extend to 4 mbps, because limited by max-limit of its max-limit parent. Max-limit of parent A should be at least 4 mbps.

HTB Distribution (3)



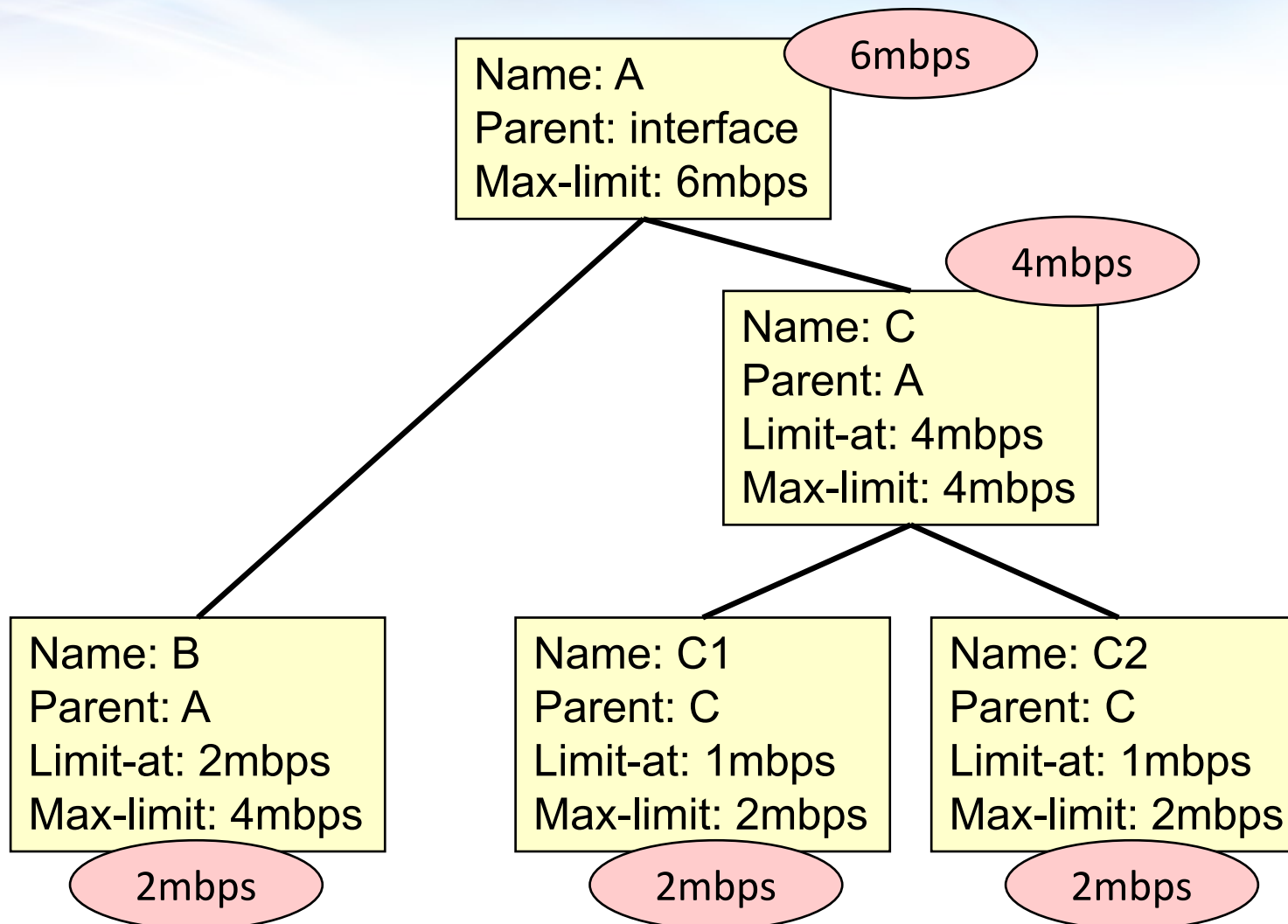
B can extend from 2 mbps to 3 mbps, because parent A still have remaining bandwidth, and B have higher priority than C.

HTB Distribution (4)



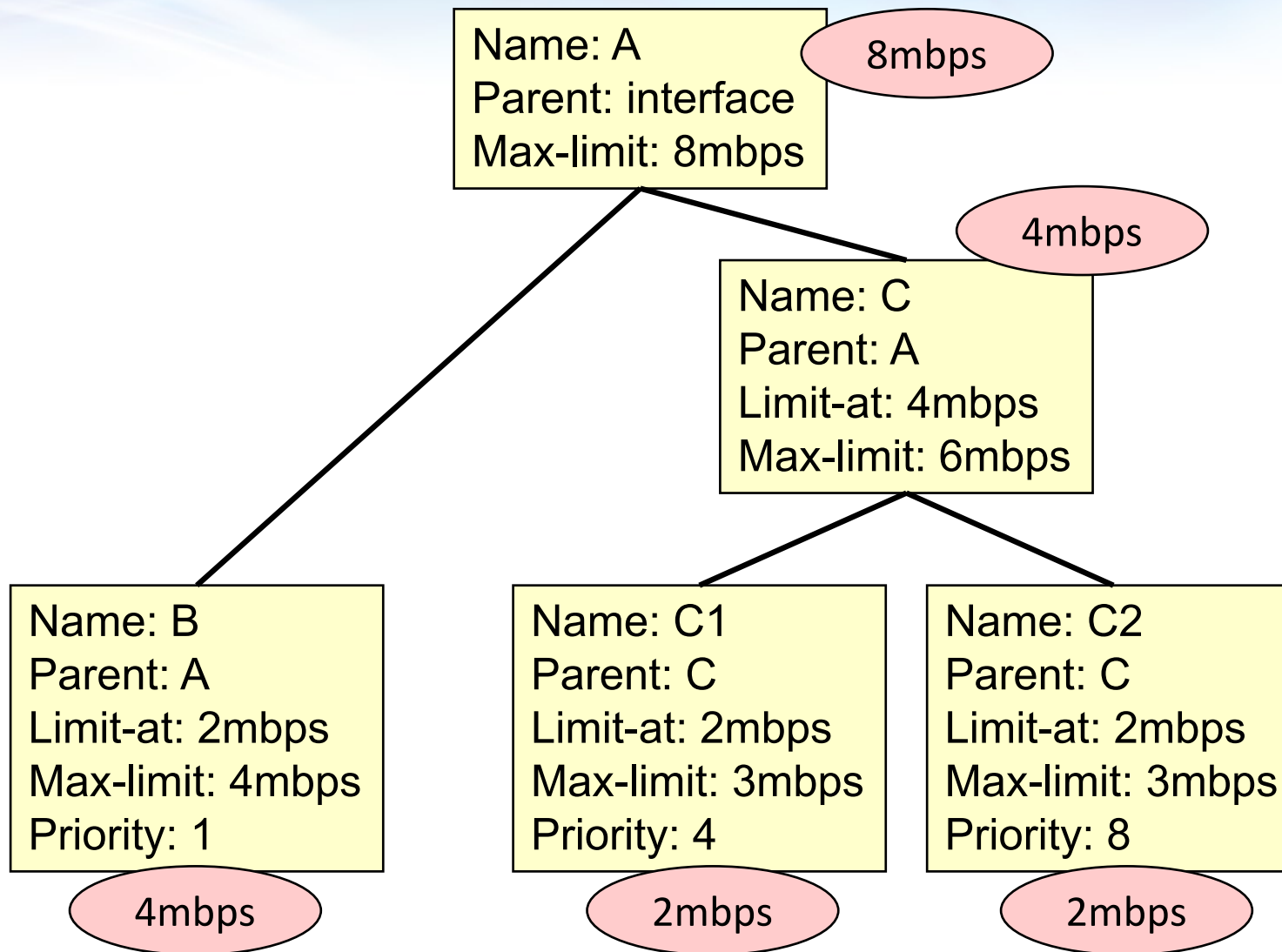
B, C1, and C2, will get 2 mbps each, as set at limit-at.

HTB Distribution (5)



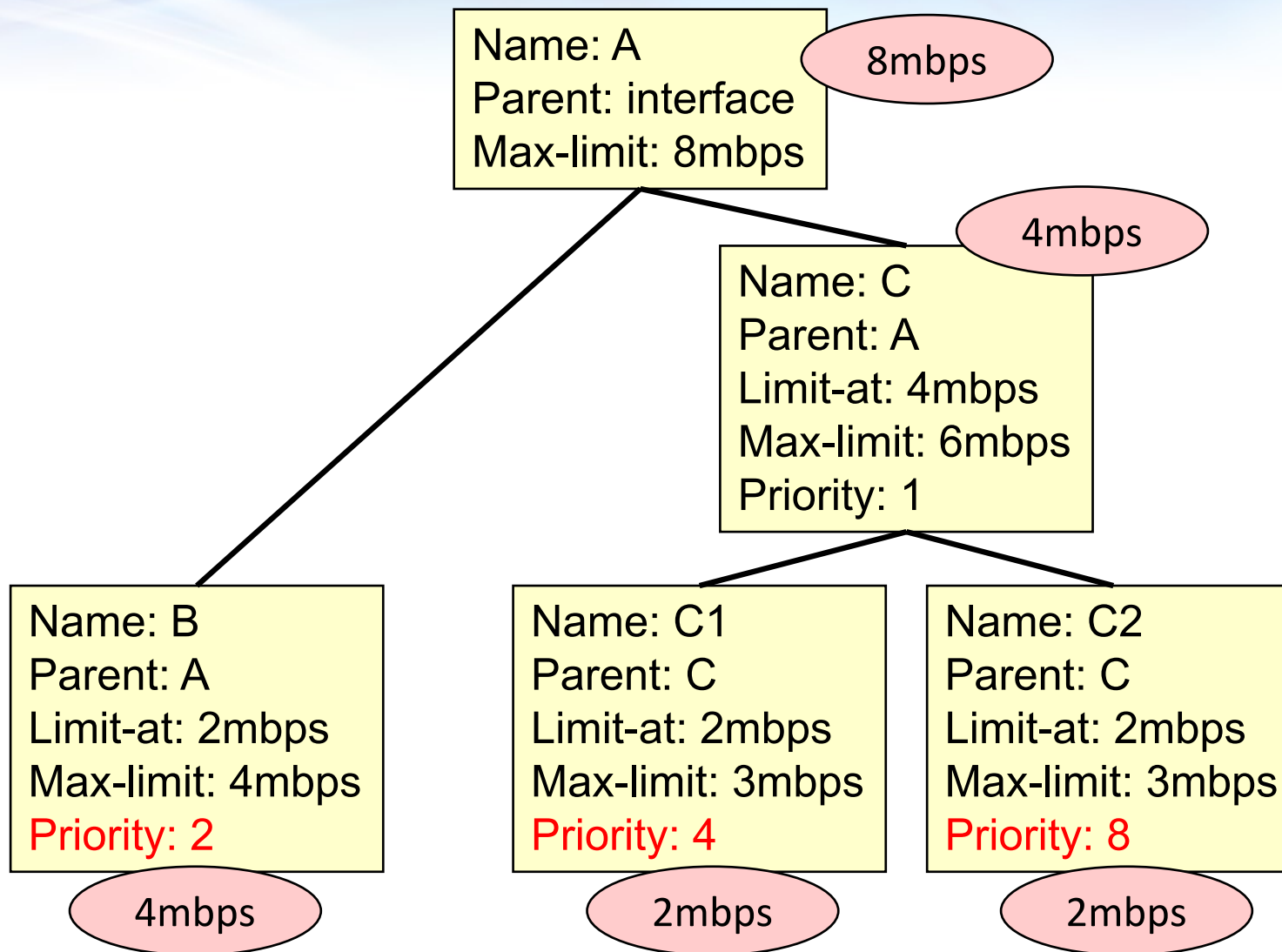
C1 and C2 can get 2 mbps (max-limit), because their parent (C) has limit-at 4 mbps.

HTB Distribution (6)



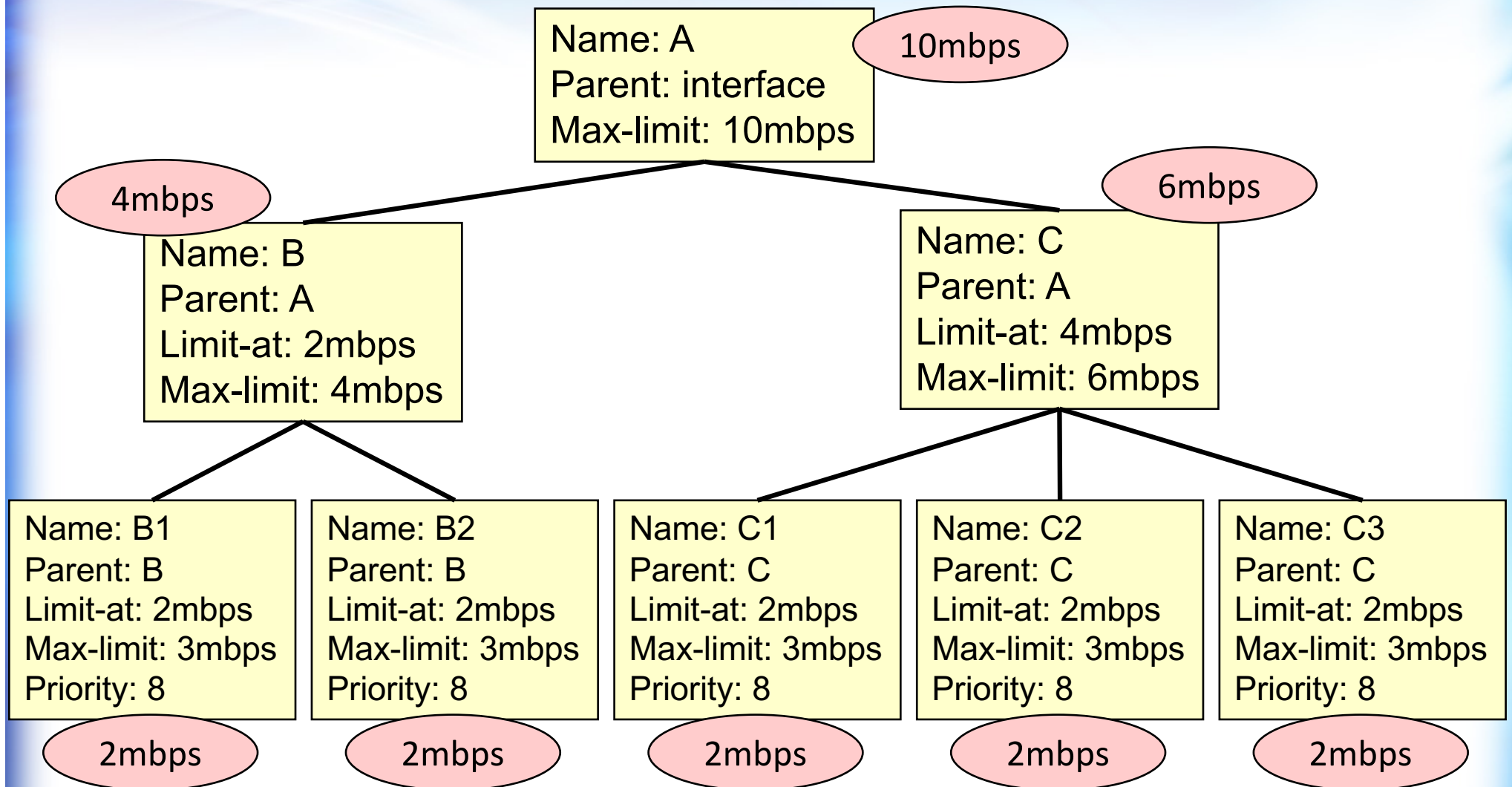
After all limit-at fulfill, remaining bandwidth will distributed base on priority

HTB Distribution (7)



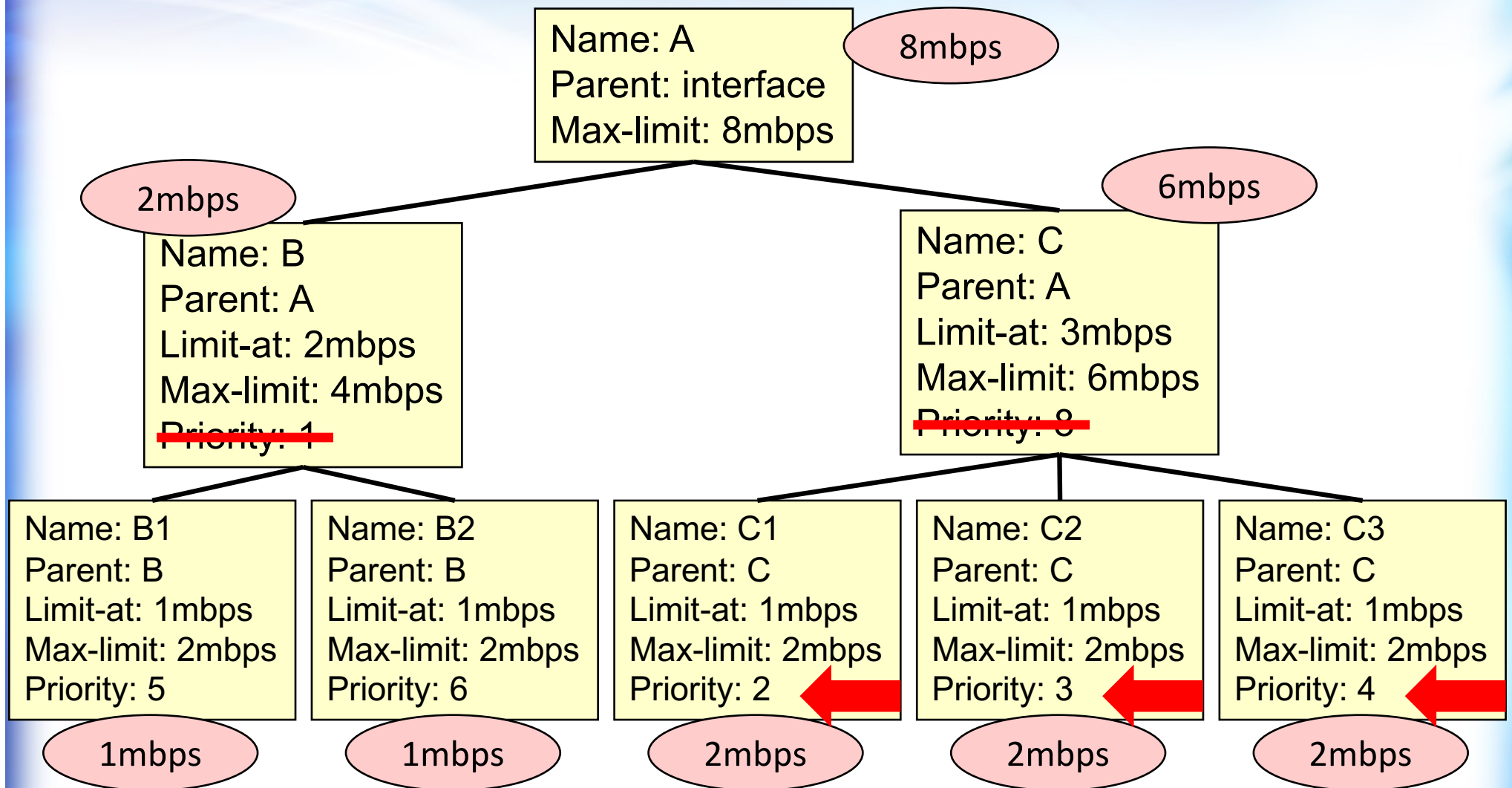
Only priority on client (leaf) will count.

HTB Distribution (8)



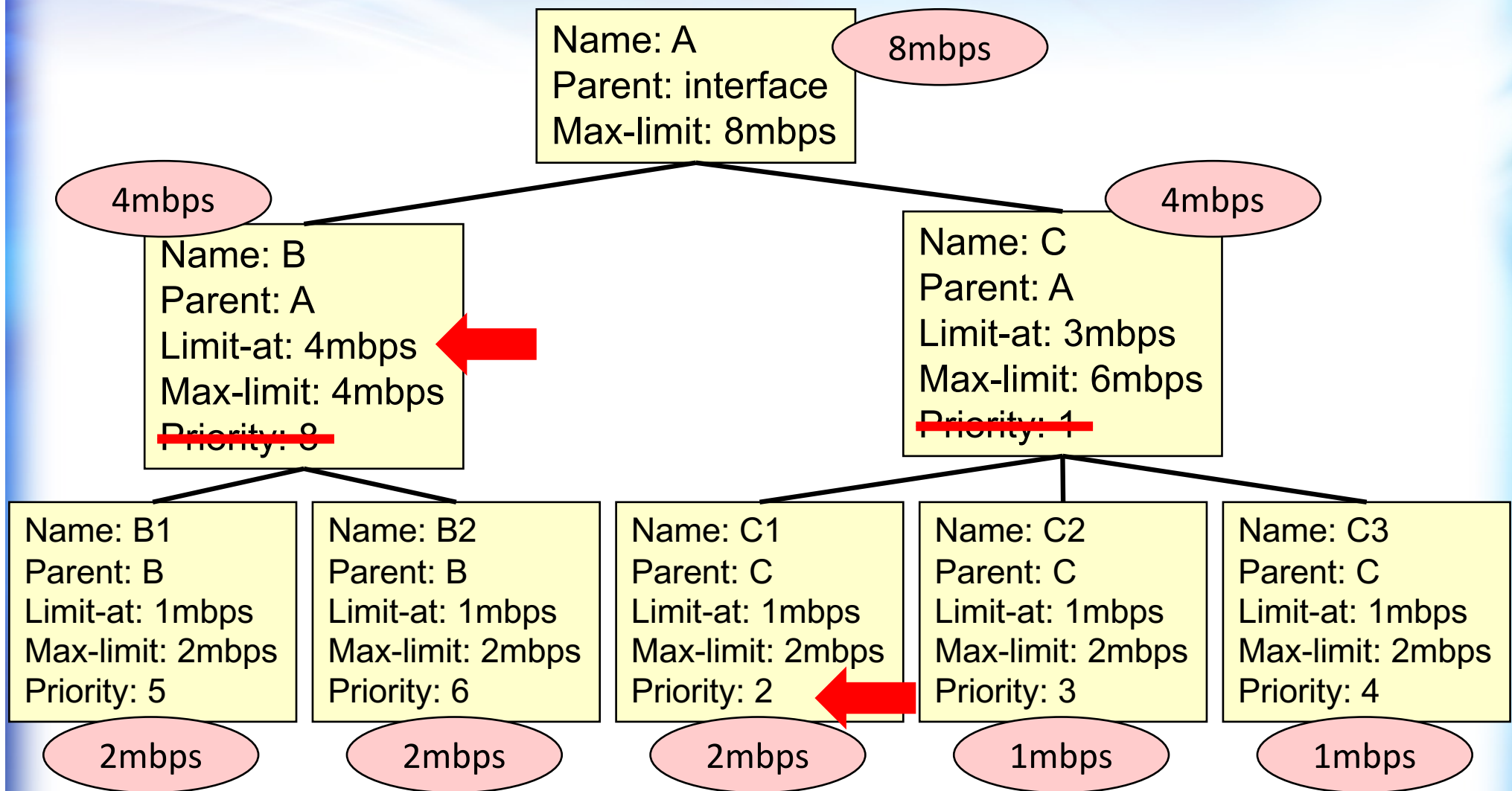
All leaf queue will get 2 mbps.

HTB Distribution (9)



C1, C2, C3 will get 2 mbps, because they have higher priority than B1 and B2

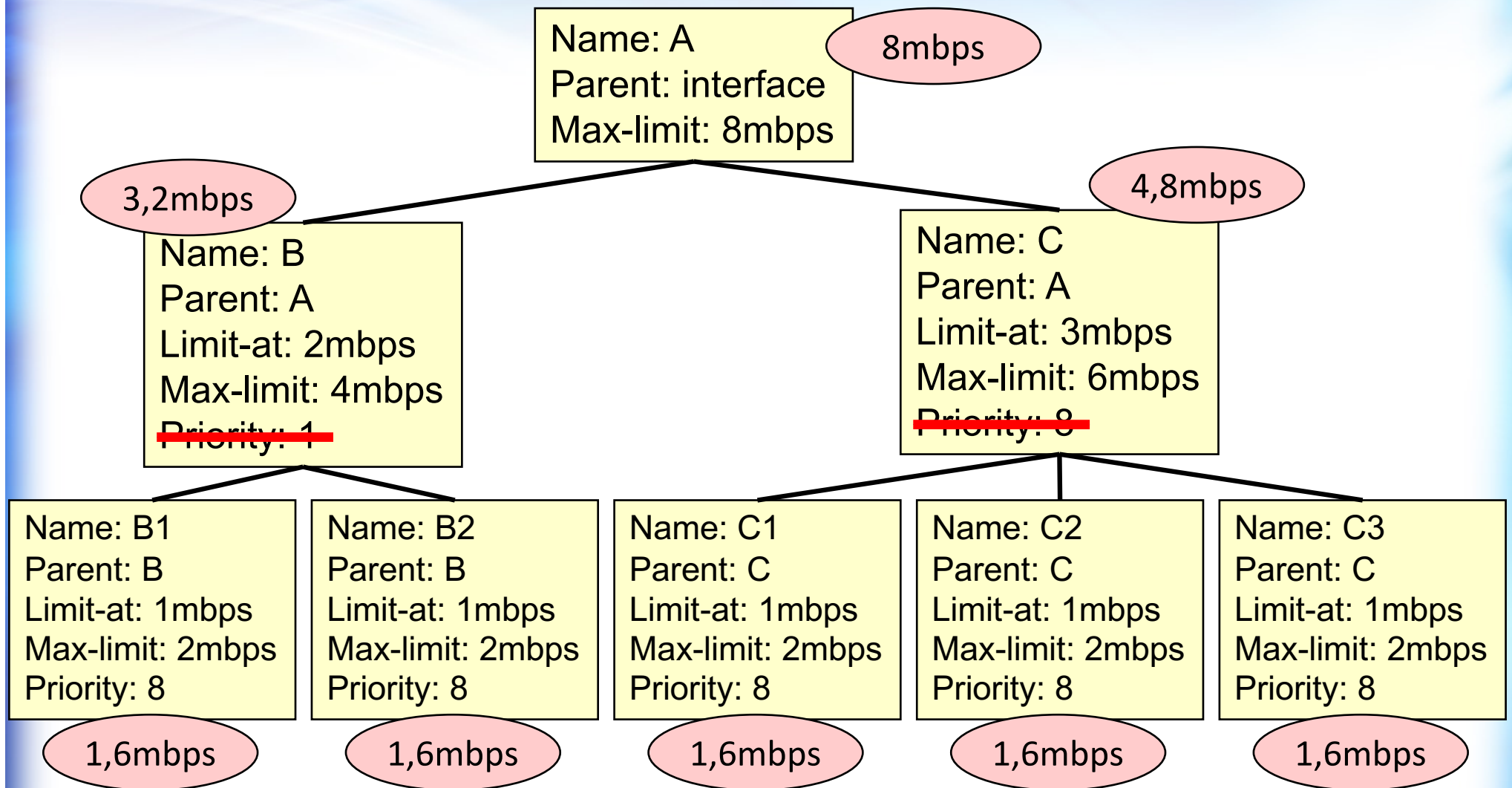
HTB Distribution (10)



B get 4 mbps because its limit-at, and then divided to B1 and B2 (2mbps each).

C1 > C2 and C3 because have higher priority.

HTB Distribution (11)



Bandwidth devided equally to B1, B2, C1, C2, C3 becace they have same priority.

Tipe Queue

- Simple Queue
- Queue Tree

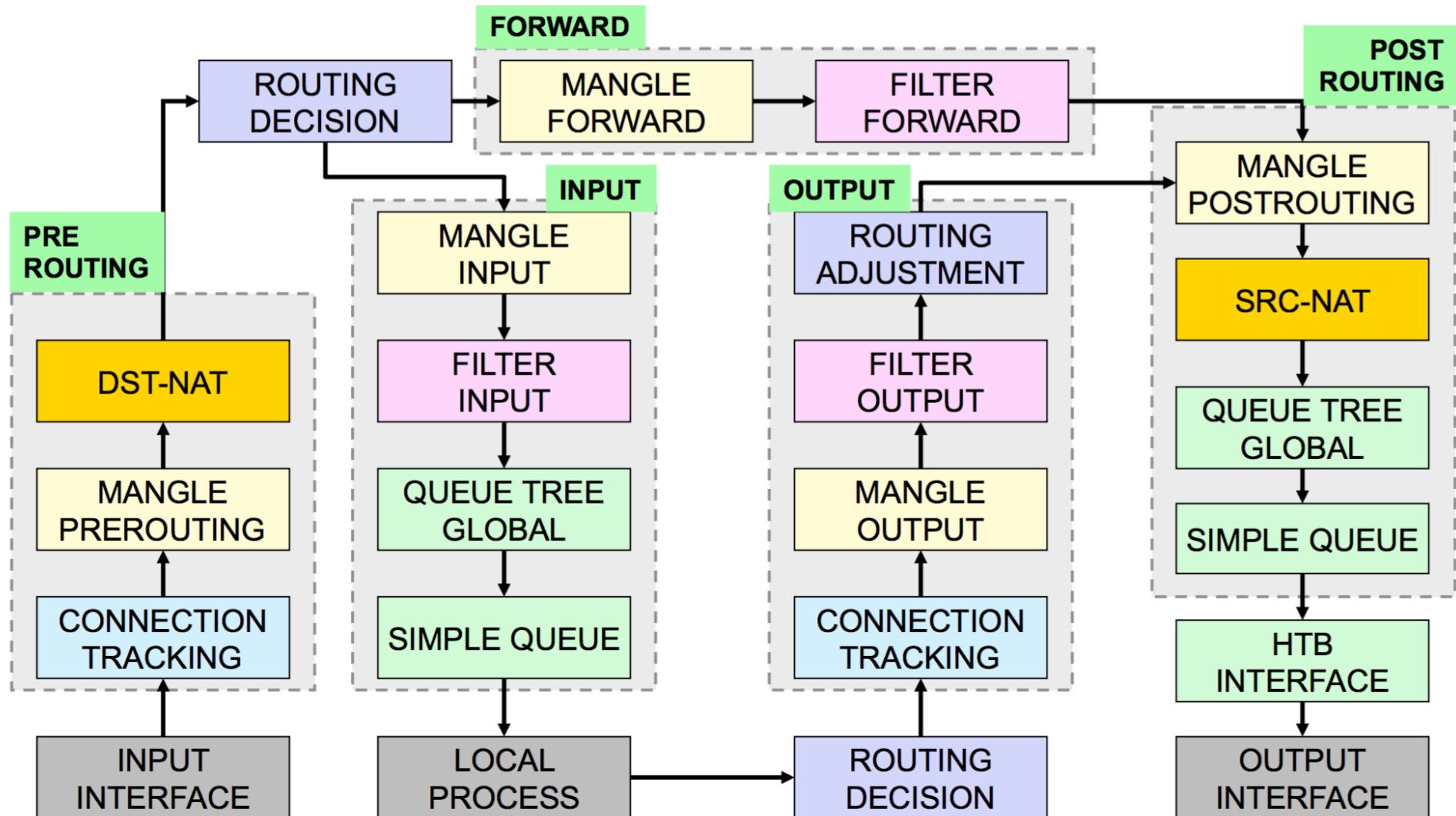
Simple Queue

- Mudah digunakan
- Untuk limitasi src-address dan dst-address, dapat digunakan tanpa fitur bantu lainnya (mangle)
- 1 rule dapat digunakan sekaligus untuk traffic uplink dan downlink
- Dapat digunakan untuk melimit total traffic (downlink + uplink)
- Jika dibutuhkan, dapat memanfaatkan packet-mark dan juga parent
- Proses hanya dapat dilakukan pada interface virtual (global)

Queue Tree

- Membutuhkan packet-mark untuk menandai traffic
- Dapat digunakan dengan priority dan parent
- 1 rule hanya untuk downlink atau uplink
- Proses dapat dilakukan di interface fisik, ataupun di interface virtual (global)

Packet Flow



MULTICORE?

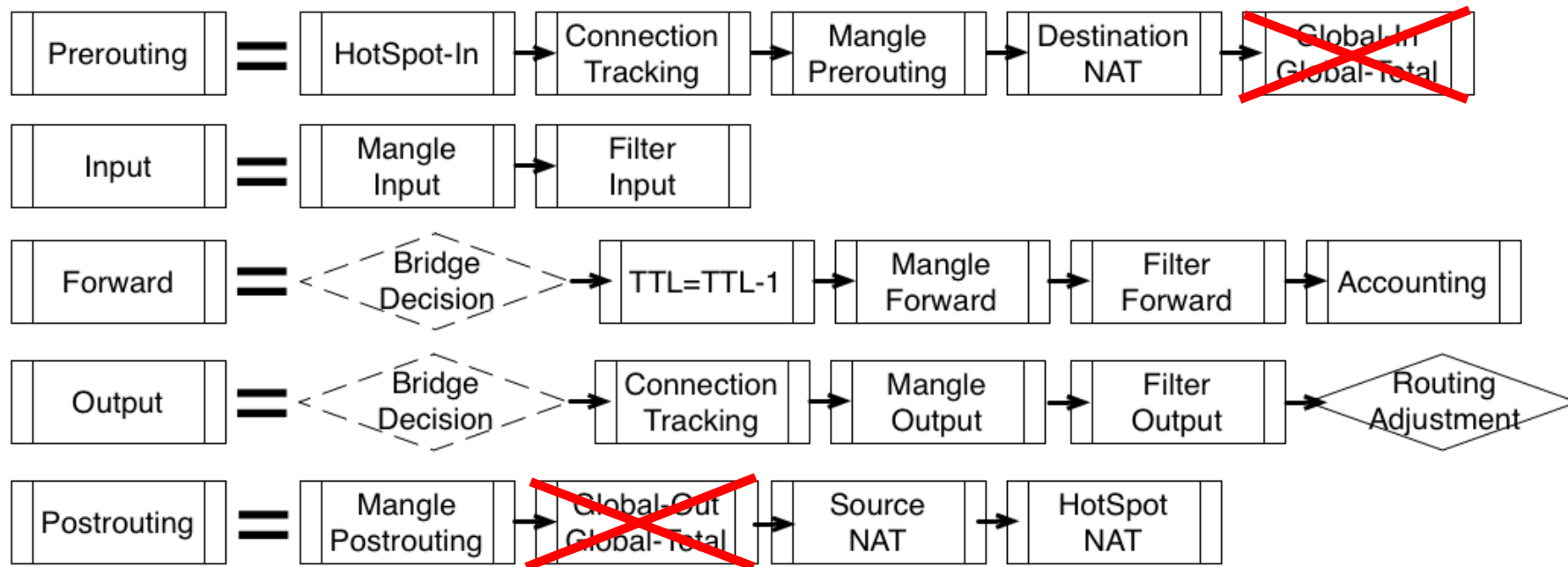
Queue & Multicore Processing

- Sebagian besar waktu yang dihabiskan sebuah paket adalah menunggu dalam queue.
- Supaya tidak memboroskan siklus CPU core saat menunggu, core tersebut akan meninggalkan paket di queue.
- Paket akan diambil secara random dari antrian untuk diproses pada core tertentu.
- Secara sederhana: queue akan membagikan paket untuk CPU Core tertentu.

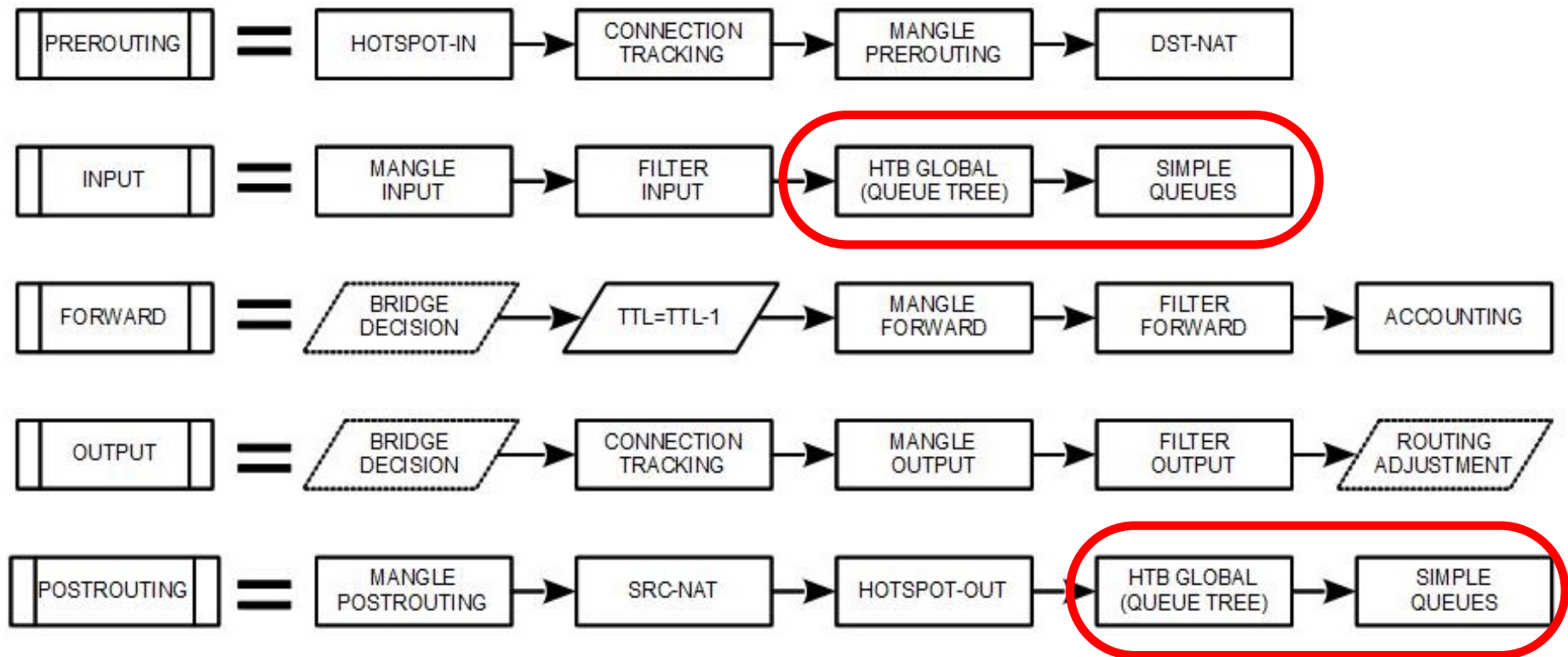
Perubahan Packet Flow

- Pada RouterOS v5.x, paket melalui proses queue beberapa kali, sehingga proses pemilihan core juga terjadi beberapa kali.
- Untuk RouterOS v6, proses QoS dirombak sehingga proses queue hanya terjadi di suatu kelompok proses, pada akhir flow.

HTB in RoS v5



HTB in RoS v6



Queue Tree on Multicore

- Jika ditinjau dari perspektif Kernel, keseluruhan HTB tree adalah satu queue, sehingga diproses hanya oleh satu core
- Optimasi seperti yang dilakukan pada simple queue akan juga dilakukan pada queue tree.
- Saran:
 - Gunakan HTB dengan interface, dan hindari menggunakan HTB global.
 - Gunakan simple queue.

Queue Change in 6.19

Di RouterOS v6.19, ada perubahan software untuk meningkatkan kinerja queue:

- Sebelumnya: core akan meninggalkan paket di queue, dan core lainnya akan dipilih secara random untuk mengatur paket tersebut
- Sekarang: core tersebut tidak hanya meninggalkan paket, tapi juga harus mengambil paket lainnya yang sudah ada di queue.
- Jika limit di queue belum tercapai, paket yang sama akan ditinggalkan dan langsung diambil lagi oleh core yang sama, membuat proses ini jauh lebih cepat.

Lab Test



2 CCR 1036 melakukan traffic generator, melalui 1 CCR 1036, routing mode.

Traffic Generator

Setiap mesin mengirimkan 10 streams (masing-masing 25mbps), dari 10 ip address yang berbeda, menuju 10 ip address yang berbeda.

Quick Start (Running)

Test ID: 1

Stream: [dropdown]

Port: [dropdown]

Interface: [dropdown]

Packet Size: 1480

PPS: [dropdown]

MBPS: 25

Tx Template: temp1, temp2, temp3, temp4, temp5, temp6, temp7, temp8, temp9, temp10

Seq	ID	Tx Rate	Rx Rate	Lost Packets	Lost Rate
121	2	24.9 Mbps	20.6 Mbps	1 063	4.3 Mbps
121	3	24.9 Mbps	20.6 Mbps	1 062	4.3 Mbps
121	4	24.9 Mbps	20.6 Mbps	1 063	4.3 Mbps
121	5	25.0 Mbps	20.6 Mbps	1 064	4.3 Mbps
121	6	24.9 Mbps	20.6 Mbps	1 062	4.3 Mbps
121	7	25.0 Mbps	20.6 Mbps	1 064	4.3 Mbps
121	8	24.9 Mbps	20.6 Mbps	1 062	4.3 Mbps
121	9	25.0 Mbps	20.6 Mbps	1 064	4.3 Mbps
121	TOT	249.9 Mbps	206.0 Mbps	9 972	43.9 Mbps
TOT	0	24.9 Mbps	19.9 Mbps	51 646	5.0 Mbps

Without any configuration

admin@192.168.130.2 (Router-DUT) - WinBox v6.27 on CCR1036-12G-4S (tile)

Memory: 3581.2 MIB Uptime: 18:56:45 CPU: 0%

Interface List

Interface Ethernet EoIP Tunnel IP Tunnel GRE Tunnel VLAN VRRP Bonding LTE

+ - ✓ ✗ 📁 🏠 Find

	Name	Type	L2 MTU	Tx	Rx	Tx Pac
R	ether6	Ethernet	1590	250.2 Mbps	250.2 Mbps	
R	ether1	Ethernet	1590	250.2 Mbps	250.2 Mbps	
RS	ether12	Ethernet	1590	2.2 Mbps	100.8 kbps	
R	bridge-remote	Bridge	1590	2.1 Mbps	79.4 kbps	
RS	ether10	Ethernet	1590	41.8 kbps	26.8 kbps	
RS	ether11	Ethernet	1590	41.8 kbps	26.8 kbps	
	ether2	Ethernet	1590	0 bps	0 bps	

Mangle

- Kita perlu membuat firewall mangle untuk packet marking jika akan menggunakan queue tree

```
/ip firewall mangle  
  
add action=mark-packet chain=prerouting \  
new-packet-mark=packet-src-0.255 \  
passthrough=no src-address=172.16.0.255  
  
add action=mark-packet chain=prerouting \  
new-packet-mark=packet-dst-0.255 \  
passthrough=no dst-address=172.16.0.255
```

- Korelasi antara packet-mark dan cpu-load?

Conn-Mark?

- Tidakkah seharusnya kita menggunakan connection-mark sebelum packet-mark di firewall mangle?
- Ya.
Tapi di lab-test ini kita ingin melihat seberapa banyak CCR dapat bertahan sehubungan dengan jumlah packet-mark

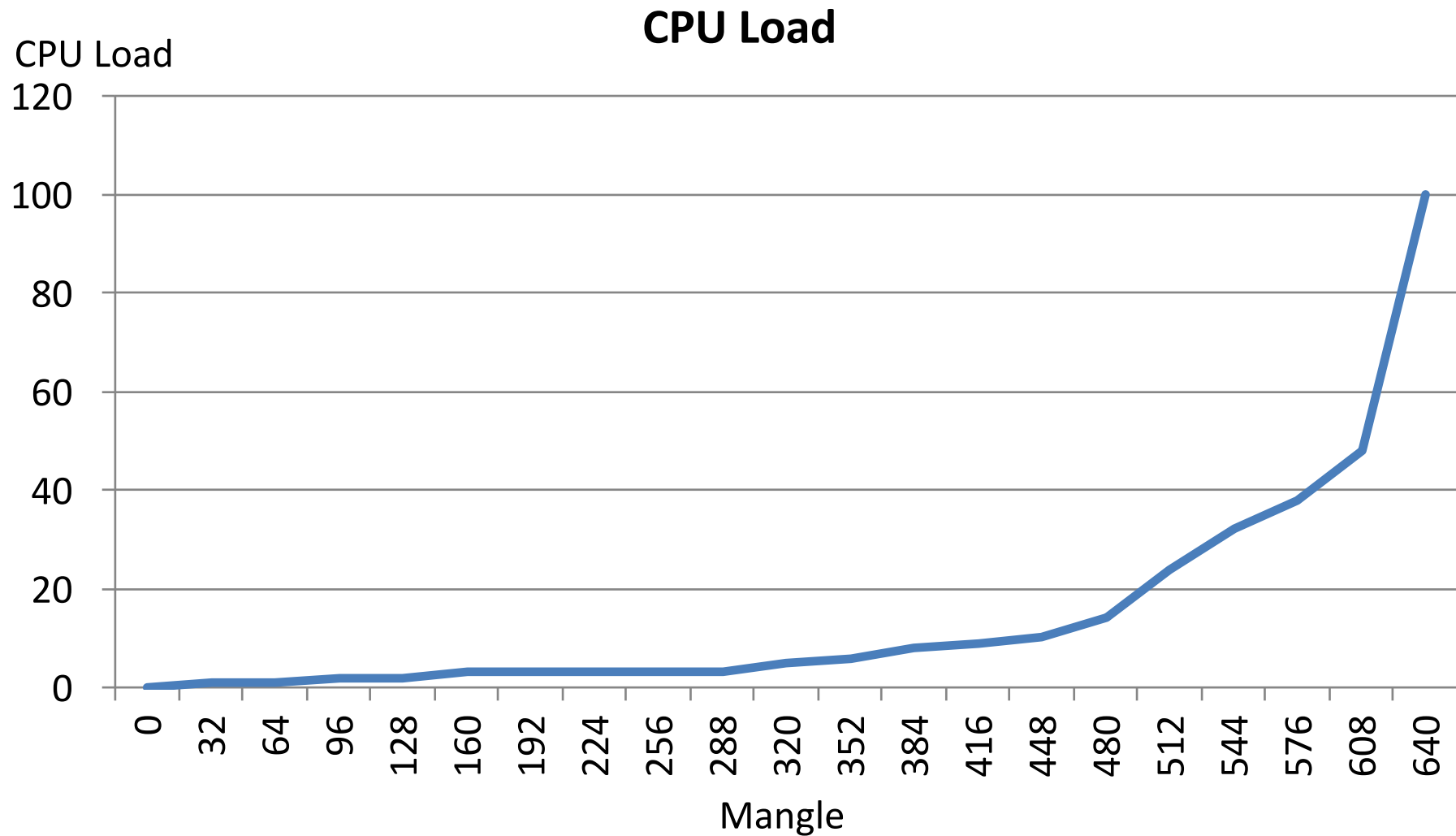
Later, I test with conn-mark, almost same result.

Firewall - Mangle

The screenshot shows the Mikrotik WinBox Firewall configuration window, specifically the Mangle tab. The window title is "Firewal". The tabs at the top are "Filter Rules", "NAT", "Mangle", "Service Ports", "Connections", "Address Lists", and "Layer7 Protocols". Below the tabs are several icons for adding, deleting, and enabling/disabling rules, along with buttons for "Reset Counters" and "Reset All Counters". A search bar contains the text "Find" and "all". The main area is a table with the following columns: #, Action, Chain, Src. Address, Dst. Address, Prot..., Src. Port, Dst. Port, In. Int..., and Out. The table contains 15 visible rows, numbered 773 to 787. Rows 773, 774, and 775 are disabled (indicated by an 'X' in the first column). Rows 776 through 787 are enabled (indicated by a red pencil icon in the first column). The "Chain" column for all rows is "prerouting". The "Src. Address" and "Dst. Address" columns show a sequence of IP addresses from 172.16.1.125 down to 172.16.1.118. At the bottom of the window, it says "1024 items (248 selected)".

#	Action	Chain	Src. Address	Dst. Address	Prot...	Src. Port	Dst. Port	In. Int...	Out
773	X ma...	prerouting		172.16.1.125					
774	X ma...	prerouting	172.16.1.124						
775	X ma...	prerouting		172.16.1.124					
776	ma...	prerouting	172.16.1.123						
777	ma...	prerouting		172.16.1.123					
778	ma...	prerouting	172.16.1.122						
779	ma...	prerouting		172.16.1.122					
780	ma...	prerouting	172.16.1.121						
781	ma...	prerouting		172.16.1.121					
782	ma...	prerouting	172.16.1.120						
783	ma...	prerouting		172.16.1.120					
784	ma...	prerouting	172.16.1.119						
785	ma...	prerouting		172.16.1.119					
786	ma...	prerouting	172.16.1.118						
787	ma...	prerouting		172.16.1.118					

Graphs



What next?

- Test sebelumnya hanya menggunakan packet –mark. Kita lanjutkan dengan queue tree (512 rules).
- Kita gunakan 512 mangles, 24% CPU load.

```
/queue tree
```

```
add max-limit=20M name=queue-src-1.1  
packet-mark=packet-src-1.1 parent=global  
queue=default
```

```
add max-limit=20M name=queue-dst-1.1  
packet-mark=packet-dst-1.1 parent=global  
queue=default
```

Queue List

Simple Queues | Interface Queues | Queue Tree | Queue Types

	Name	Parent	Packet...	Max Limit...	Avg. Rate	Bytes	Packets
	queue-src-1.3	global	packet...	20M	20.1 Mbps	166.6 ...	350 80
	queue-src-1.7	global	packet...	20M	20.1 Mbps	166.6 ...	350 85
	queue-src-1.8	global	packet...	20M	20.1 Mbps	166.6 ...	350 85
	queue-src-1.9	global	packet...	20M	20.1 Mbps	166.6 ...	350 80
	queue-src-1.10	global	packet...	20M	20.1 Mbps	166.6 ...	350 82
	queue-src-1.5	global	packet...	20M	20.1 Mbps	166.6 ...	350 80
	queue-src-1.4	global	packet...	20M	20.1 Mbps	166.6 ...	350 80
	queue-src-1.6	global	packet...	20M	20.1 Mbps	166.6 ...	350 80
	queue-src-1.2	global	packet...	20M	20.0 Mbps	166.5 ...	350 64
	queue-src-1.1	global	packet...	20M	20.0 Mbps	166.3 ...	118 90
	queue-src-1.100	global	packet...	20M	256 bps	260 B	
X	queue-dst-0.0	global	packet...	20M	0 bps	0 B	
X	queue-dst-0.1	global	packet...	20M	0 bps	0 B	
X	queue-dst-0.10	global	packet...	20M	0 bps	0 B	
X	queue-dst-0.100	global	packet...	20M	0 bps	0 B	
Y	queue-dst-0.101	global	packet...	20M	0 bps	0 B	

1024 items (1 select... | 286.1 KiB queued | 491 packets queued

CPU Load

- Dengan 512 mangles tanpa queue tree, 24% CPU load.
- Dengan 512 mangles + 512 queue tree, 43% CPU Load (hampir 2 kali lipat).

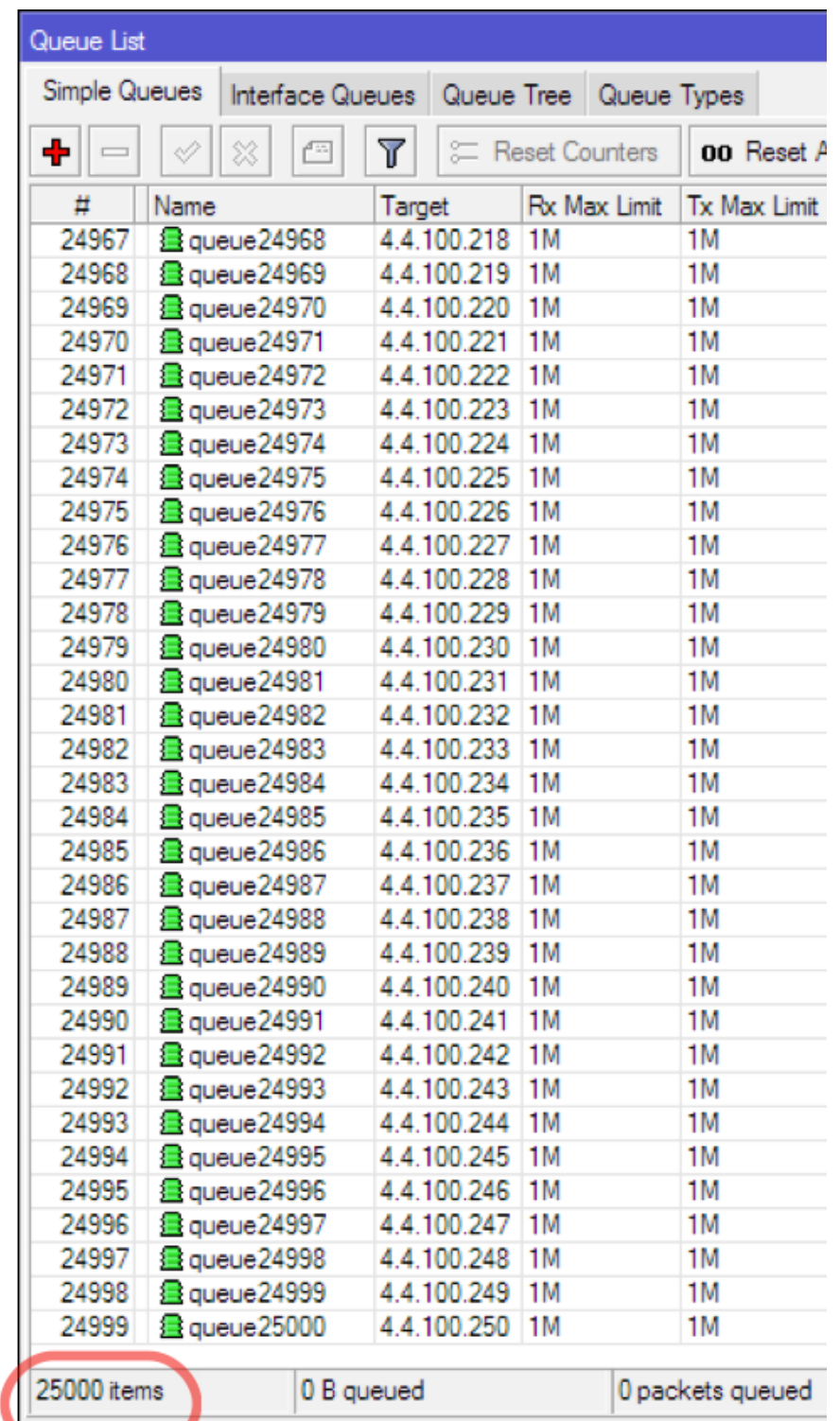
Memory: 3561.0 MIB Uptime: 19:44:47 CPU: 43%

- Tapi, on Tools – Profile, load untuk queue masih rendah.

Profile (Running)	
Name	Usage
bridging	0.0
dns	0.0
firewall	40.0
idle	51.2
management	0.4
networking	3.2
profiling	1.0
queuing	2.4
routing	0.0
unclassified	1.2
winbox	0.0

Simple Queue

- Algoritma pencocokan (matching) telah diubah:
 - berdasarkan hash
 - proses pencocokan lebih cepat
 - QoS akan optimal pada perangkat dengan multi core jika simple queue teratas (parent) berjumlah minimal 32, sehingga proses tersebut dapat disebar lebih merata.



The screenshot shows the Mikrotik WinBox interface for the Queue List. The interface has tabs for 'Simple Queues', 'Interface Queues', 'Queue Tree', and 'Queue Types'. Below the tabs are several control icons: a plus sign, a minus sign, a checkmark, a cross, a document icon, a funnel icon, and a 'Reset Counters' button. A 'Reset All' button is also visible. The main area contains a table with the following columns: '#', 'Name', 'Target', 'Rx Max Limit', and 'Tx Max Limit'. The table lists 32 queues, each with a unique ID (from 24967 to 24999) and a name (queue24968 to queue25000). The 'Target' column shows IP addresses ranging from 4.4.100.218 to 4.4.100.250. The 'Rx Max Limit' and 'Tx Max Limit' columns both show '1M'. At the bottom of the interface, there is a status bar with three sections: '25000 items' (circled in red), '0 B queued', and '0 packets queued'.

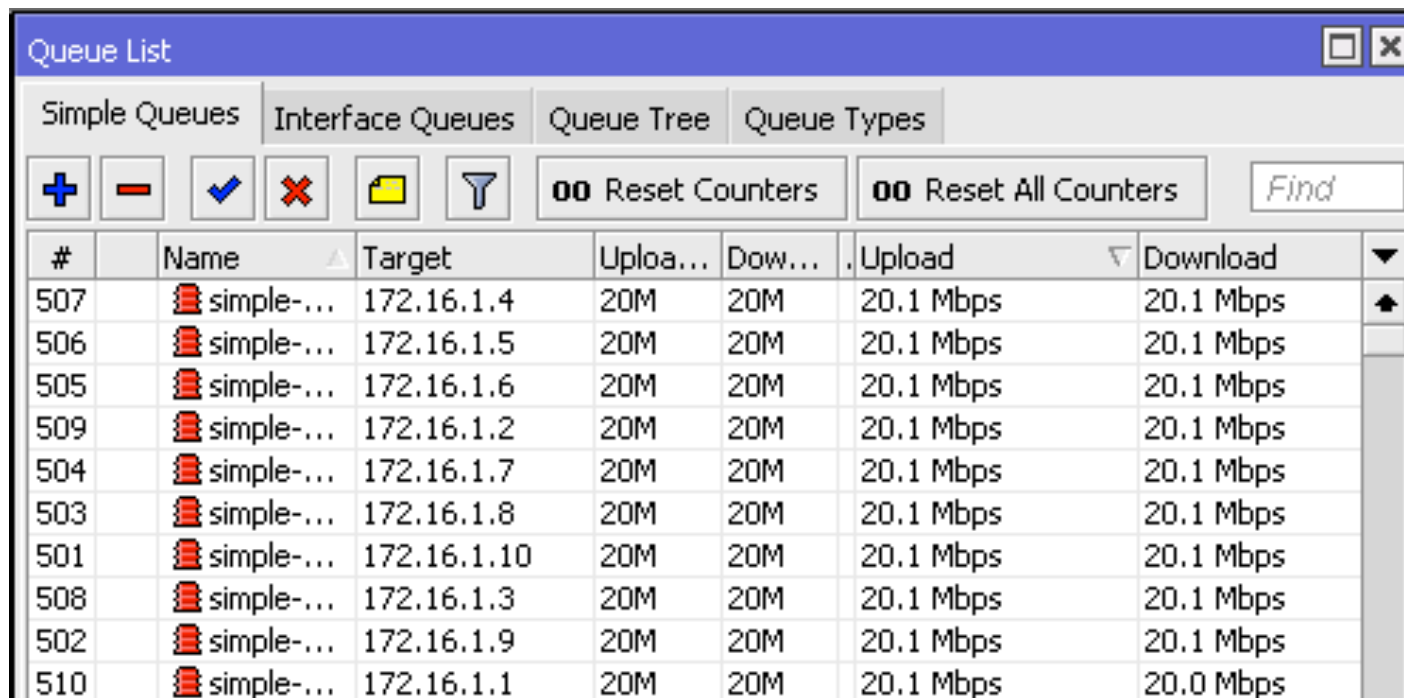
#	Name	Target	Rx Max Limit	Tx Max Limit
24967	queue24968	4.4.100.218	1M	1M
24968	queue24969	4.4.100.219	1M	1M
24969	queue24970	4.4.100.220	1M	1M
24970	queue24971	4.4.100.221	1M	1M
24971	queue24972	4.4.100.222	1M	1M
24972	queue24973	4.4.100.223	1M	1M
24973	queue24974	4.4.100.224	1M	1M
24974	queue24975	4.4.100.225	1M	1M
24975	queue24976	4.4.100.226	1M	1M
24976	queue24977	4.4.100.227	1M	1M
24977	queue24978	4.4.100.228	1M	1M
24978	queue24979	4.4.100.229	1M	1M
24979	queue24980	4.4.100.230	1M	1M
24980	queue24981	4.4.100.231	1M	1M
24981	queue24982	4.4.100.232	1M	1M
24982	queue24983	4.4.100.233	1M	1M
24983	queue24984	4.4.100.234	1M	1M
24984	queue24985	4.4.100.235	1M	1M
24985	queue24986	4.4.100.236	1M	1M
24986	queue24987	4.4.100.237	1M	1M
24987	queue24988	4.4.100.238	1M	1M
24988	queue24989	4.4.100.239	1M	1M
24989	queue24990	4.4.100.240	1M	1M
24990	queue24991	4.4.100.241	1M	1M
24991	queue24992	4.4.100.242	1M	1M
24992	queue24993	4.4.100.243	1M	1M
24993	queue24994	4.4.100.244	1M	1M
24994	queue24995	4.4.100.245	1M	1M
24995	queue24996	4.4.100.246	1M	1M
24996	queue24997	4.4.100.247	1M	1M
24997	queue24998	4.4.100.248	1M	1M
24998	queue24999	4.4.100.249	1M	1M
24999	queue25000	4.4.100.250	1M	1M

25000 items | 0 B queued | 0 packets queued

Let's try Simple Queue

- Kita membuat 512 simple queue :

```
/queue simple add max-limit=20M/20M  
name=simple-queue-1.1 target=172.16.1.1/32
```



The screenshot shows the 'Queue List' window in Mikrotik WinBox. It displays a table of simple queues with columns for ID, Name, Target, Upload limit, Download limit, Upload rate, and Download rate. The queues are numbered 501 to 510, with names like 'simple-queue-1.1' and targets like '172.16.1.1/32'. The upload and download limits are set to 20M, and the rates are 20.1 Mbps.

#	Name	Target	Uploa...	Dow...	Upload	Download
507	simple-...	172.16.1.4	20M	20M	20.1 Mbps	20.1 Mbps
506	simple-...	172.16.1.5	20M	20M	20.1 Mbps	20.1 Mbps
505	simple-...	172.16.1.6	20M	20M	20.1 Mbps	20.1 Mbps
509	simple-...	172.16.1.2	20M	20M	20.1 Mbps	20.1 Mbps
504	simple-...	172.16.1.7	20M	20M	20.1 Mbps	20.1 Mbps
503	simple-...	172.16.1.8	20M	20M	20.1 Mbps	20.1 Mbps
501	simple-...	172.16.1.10	20M	20M	20.1 Mbps	20.1 Mbps
508	simple-...	172.16.1.3	20M	20M	20.1 Mbps	20.1 Mbps
502	simple-...	172.16.1.9	20M	20M	20.1 Mbps	20.1 Mbps
510	simple-...	172.16.1.1	20M	20M	20.1 Mbps	20.0 Mbps

Dengan Simple Queue

The screenshot shows the Mikrotik WinBox interface. At the top, system statistics are displayed: Memory: 3553.4 MiB, Uptime: 19:50:49, and CPU: 1%. The CPU value is highlighted with a red box. Below this, the 'Queue List' window is open, showing a list of 'Simple Queues'. The table below contains the data from this list.

#	Name	Target	Uploa...	Dow...	Upload	Download
507	simple-...	172.16.1.4	20M	20M	20.1 Mbps	20.1 Mbps
506	simple-...	172.16.1.5	20M	20M	20.1 Mbps	20.1 Mbps
505	simple-...	172.16.1.6	20M	20M	20.1 Mbps	20.1 Mbps
509	simple-...	172.16.1.2	20M	20M	20.1 Mbps	20.1 Mbps
504	simple-...	172.16.1.7	20M	20M	20.1 Mbps	20.1 Mbps
503	simple-...	172.16.1.8	20M	20M	20.1 Mbps	20.1 Mbps
501	simple-...	172.16.1.10	20M	20M	20.1 Mbps	20.1 Mbps
508	simple-...	172.16.1.3	20M	20M	20.1 Mbps	20.1 Mbps
502	simple-...	172.16.1.9	20M	20M	20.1 Mbps	20.1 Mbps
510	simple-...	172.16.1.1	20M	20M	20.1 Mbps	20.0 Mbps

To the right of the Queue List window, a 'CPU' window is visible, showing a list of CPU cores and their load percentages. The list includes cpu21 (26%), cpu24 (18%), cpu15 (9%), cpu0 (4%), cpu4 (4%), cpu17 (4%), cpu1 (3%), cpu11 (3%), cpu22 (3%), cpu29 (3%), cpu31 (3%), and cpu32 (3%).

Hanya 1% of CPU Load dengan Simple Queue

Why Simple Queue?

- Jika kita gunakan simple queue, tidak harus menggunakan mangle (mangle membutuhkan CPU resources yang besar).
- Simple Queue di v6 memiliki proses hashing yang efisien.
- Untuk layanan non dedicated, bisa menggunakan fitur burst.
- Jika kita gunakan Queue tree, queue dalam satu interface parent akan diproses hanya oleh satu CPU core.

Kesimpulan

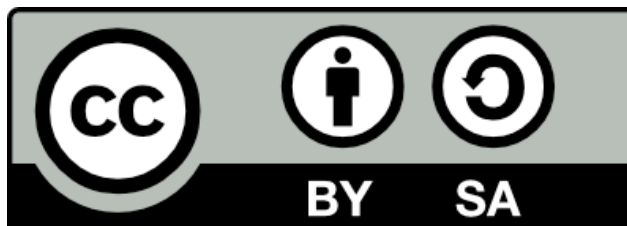
- Untuk jaringan dengan bandwidth yang tidak besar, overloaded network:
 - kombinasi packet-mark, dengan
 - queue tree, HTB, dan burst
- Untuk high throughput backbone:
 - gunakan multicore router:
 - CCR 36 – 72 core
 - Intel base quad core Xeon (8 thread)
 - simple queue (no parent)

Thank you

Comments and suggestions:

Valens Riyadi (info@mikrotik.co.id)

 @valensriyadi @mikrotik_id



This license lets others remix, tweak, and build upon your work even for commercial purposes, as long as they credit you and license their new creations under the identical terms. This license is often compared to “copyleft” free and open source software licenses. All new works based on yours will carry the same license, so any derivatives will also allow commercial use.