

QoS on RouterOS with Token Bucket

RouterOS v6.35



Valens Riyadi (Citraweb)

 @valensriyadi

Valens Riyadi

- MikroTik Certified Engineer (MTCNA, MTCWE, MTCRE, MTCTCE, MTCUME, MTCINE)
- MikroTik Certified Trainer since 2004, Certified Consultant, and Academy Coordinator
- CEO for Citranet (WISP) and Citraweb (Mikrotik Distributor and System Developer)
- Manager for IDNIC (Indonesia National Internet Registry) 2009-2015
- Expert on Cyber Crime, Digital Forensic, IT for Disaster Relief, Live Streaming
- Proud member of “Routed World” community





mikrobits

MikroTik Training Center

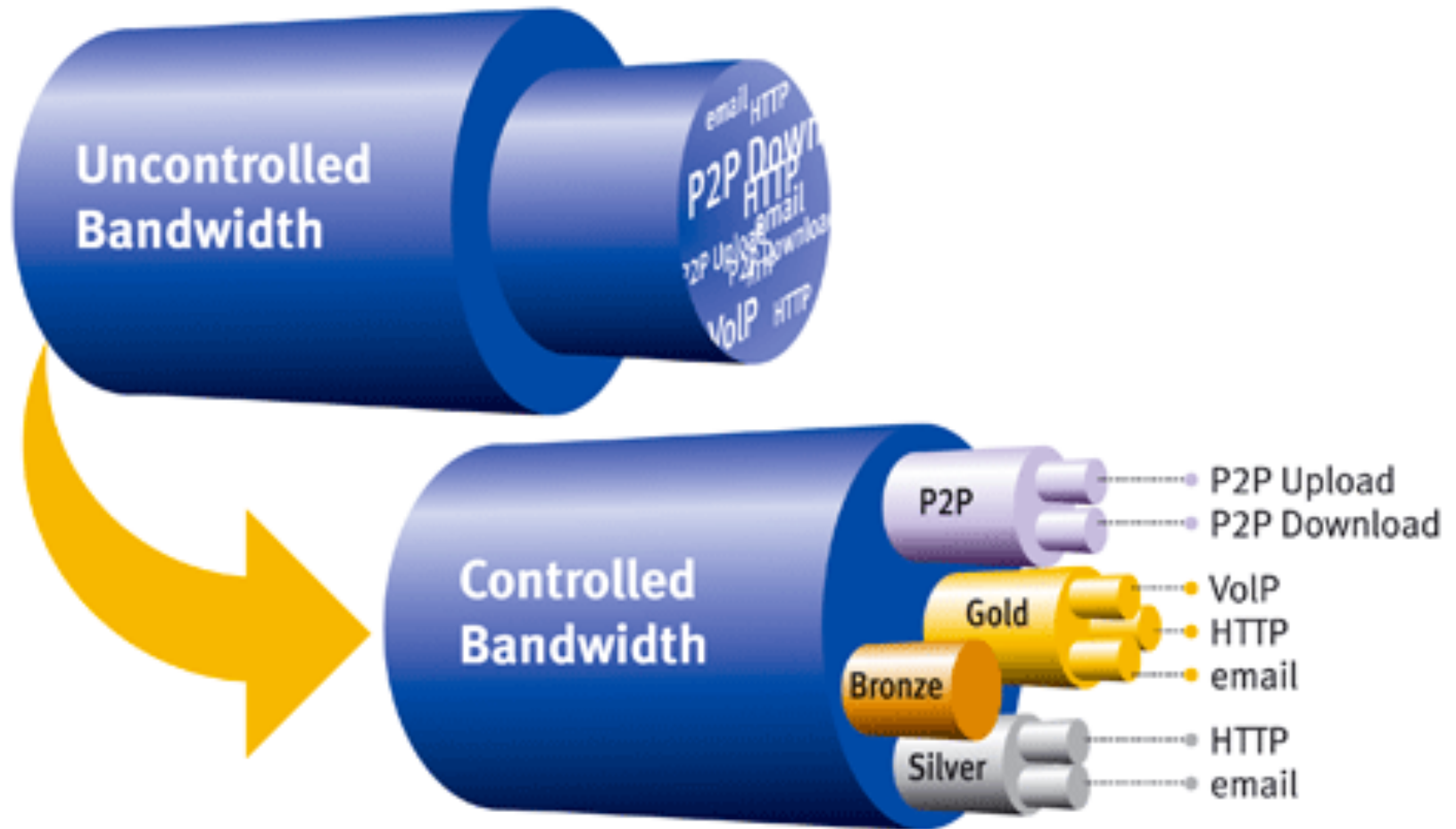
- The first MikroTik Training Center in Asia Pasific since 2004, more then 5500 participants (237 classes).
- Mikrotik Academy Coordinator.





Mikrotik Training for Indonesia Special Forces

Why do we need to manage bandwidth?



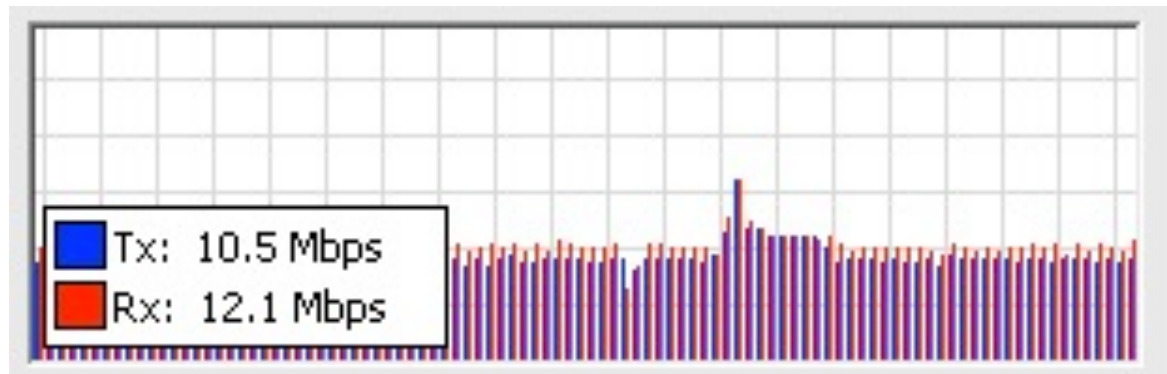
QoS on RouterOS

- MikroTik RouterOS is **one of the most advanced bandwidth management**, compared to any other brand.
- Why?
 - Advanced HTB configuration
 - Double limitation + Burst
 - A lot of option and parameter → packet-mark
 - Grouping, protocol, layer 7, connection size, traffic detection, etc

Burst

A mechanism to be able to provide additional bandwidth to a particular client if:

- Client not always on max-limit
- Additional bandwidth still available on related parent and interface



Happy customer



Burst will
significantly improve
customer experience
as they feel the
connection is fast

Token Bucket

- Since RouterOS v6.35, MikroTik introduce Token Bucket.
- It's been on the RouterOS engine for long time, but it's hidden and statically set to 0.1.
- Concept of Token Bucket also implemented on Linux.
- It works like burst feature, but much simpler.

Winbox Configuration

New Simple Queue

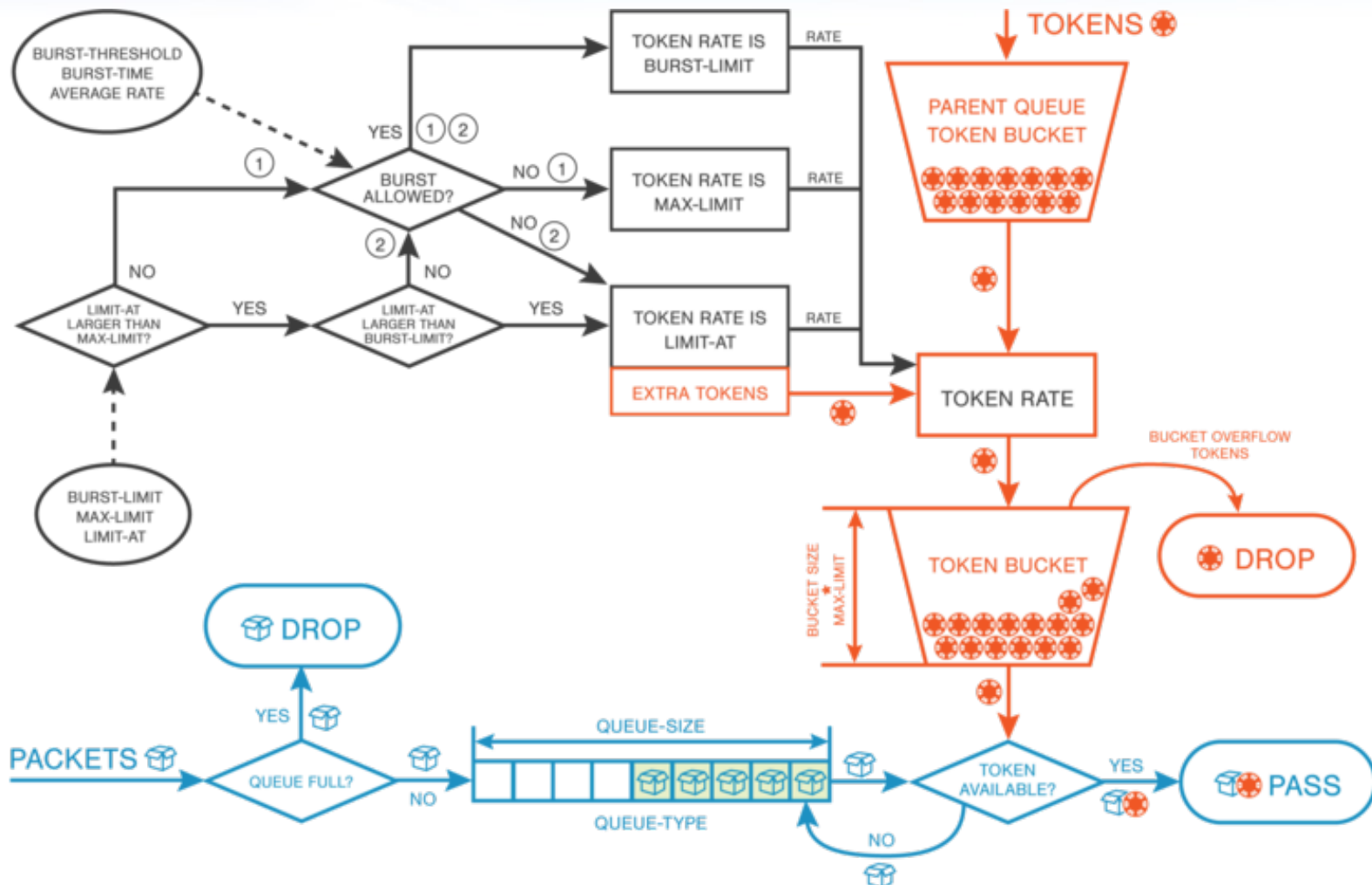
General Advanced Statistics Traffic Total Total Statistics

Packet Marks:

	Target Upload	Target Download
Limit At:	<input type="text" value="unlimited"/>	<input type="text" value="unlimited"/>
Priority:	<input type="text" value="8"/>	<input type="text" value="8"/>
Bucket Size:	<input type="text" value="0.100"/>	<input type="text" value="0.100"/>
Queue Type:	<input type="text" value="default-small"/>	<input type="text" value="default-small"/>

bits/s ratio

Token Bucket Algorithm



http://wiki.mikrotik.com/wiki/Manual:HTB-Token_Bucket_Algorithm

Bucket

- **Bucket capacity is bucket size ratio compared to max-limit.**
- **Example:**
 - Max-limit = 5Mbps
 - Bucket size ration = 3 (max 10)
 - Bucket capacity = $5 * 3 = 15$ Mbit
- When the calculation start, bucket is always considered full of token.



Token rate

$\text{token-rate} = \text{max-limit}$

$\text{token-rate} = \text{limit-at}$

if limit-at on child > max-limit on parent

$\text{token-rate} = \text{burst-limit}$

if burst is active and allowed

Token Bucket Algoritma

If traffic = token-rate	All traffic will be delivered
If traffic < token-rate	All traffic will be delivered, and token will be added to the bucket, as much as the differences
If traffic > token-rate	Traffic will be delivered, and token will be taken from the bucket, as much as the differences. If token is not enough/bucket is empty, traffic will be delivered as token-rate.

Token Bucket Example

- Max-limit = 20M, Bucket-size=10
→ Bucket-capacity = 20 * 10 = 200Mbit
- Router will delivered additional (after token-rate) 200Mbit traffic without any limitation.
- 30Mbps will be delivered in seconds:

$$\text{Bucket Capacity} / (\text{Actual Speed} - \text{Token Rate})$$

$$200 / (30 - 20) = 200 / 10 = 20 \text{ seconds}$$

40Mbps will be delivered in seconds:

Second(s)	Traffic	Token Deducted/ Speed Exceed	Token left at bucket	Speed Delivered
0	0		200 Mbit	
1	40 Mbps	20 Mbit	180 Mbit	40 Mbps
2	40 Mbps	20 Mbit	160 Mbit	40 Mbps
3	40 Mbps	20 Mbit	140 Mbit	40 Mbps
4	40 Mbps	20 Mbit	120 Mbit	40 Mbps
5	40 Mbps	20 Mbit	100 Mbit	40 Mbps
6	40 Mbps	20 Mbit	80 Mbit	40 Mbps
7	40 Mbps	20 Mbit	60 Mbit	40 Mbps
8	40 Mbps	20 Mbit	40 Mbit	40 Mbps
9	40 Mbps	20 Mbit	20 Mbit	40 Mbps
10	40 Mbps	20 Mbit	0	40 Mbps
11	40 Mbps	0	0	20 Mbps

Token Bucket Example

- After the bucket empty, traffic will be limited to token-rate (20M)
- And if client utilize lower then token-rate, for example only 15Mbps, bucket will be full of tokens after seconds:

$$\text{Bucket Capacity} / (\text{Token Rate} - \text{Actual Speed})$$

$$200 / (20 - 15) = 200 / 5 = 40 \text{ seconds}$$

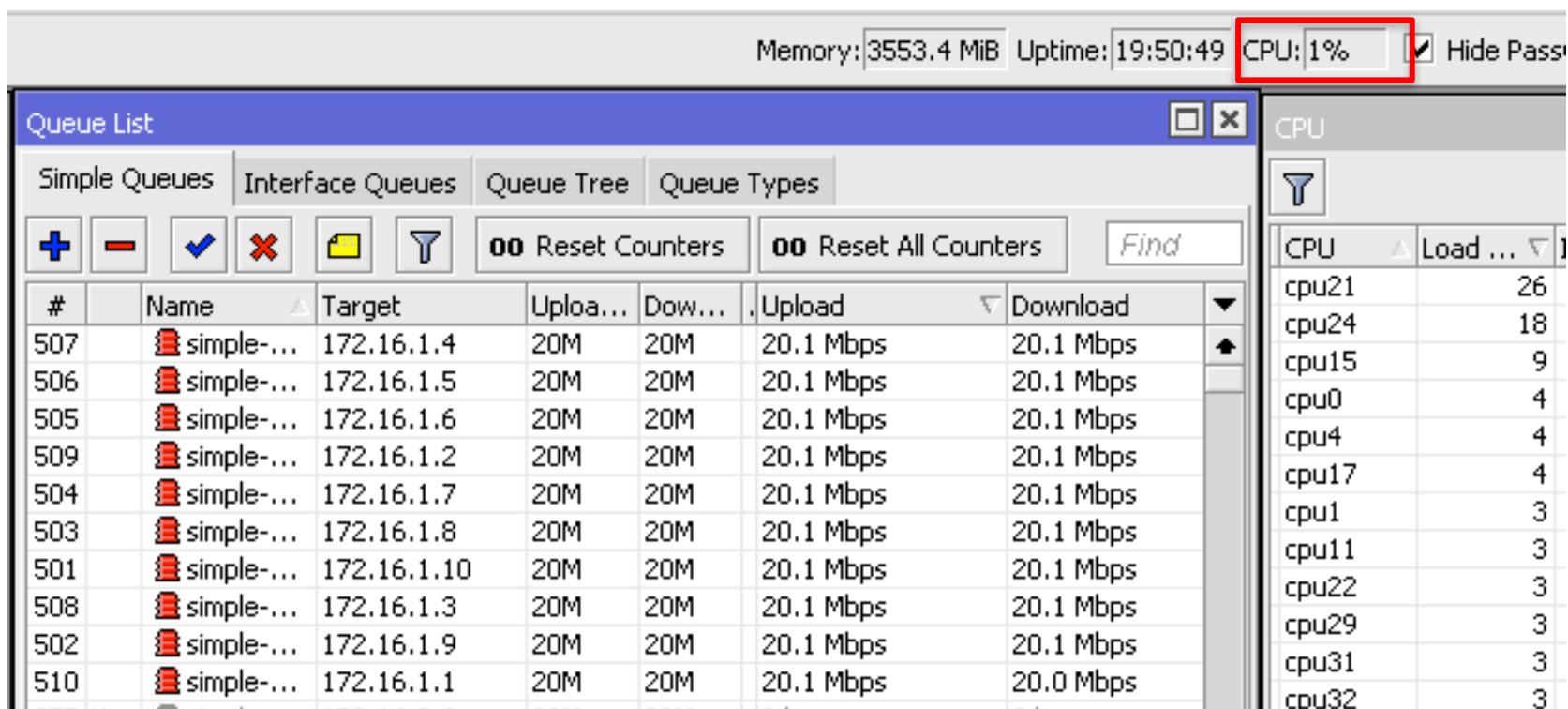
How about burst?

- In burst calculation, router have to remember speed per queue in the last (burst-time) seconds.
- In token-bucket, only token-rate and number of tokens in bucket will be considered.
- In high load application, token bucket might be more efficent then burst. Really?

Comparation

- Is token bucket really more efficient then burst system?
- It's really difficult to test:
 - CPU load is not only related with queue configuration, but the size of traffic delivered by router
 - Queue is very efficient. 500+ simple queue with 500mbps only make 1% CPU load.

With Simple Queue



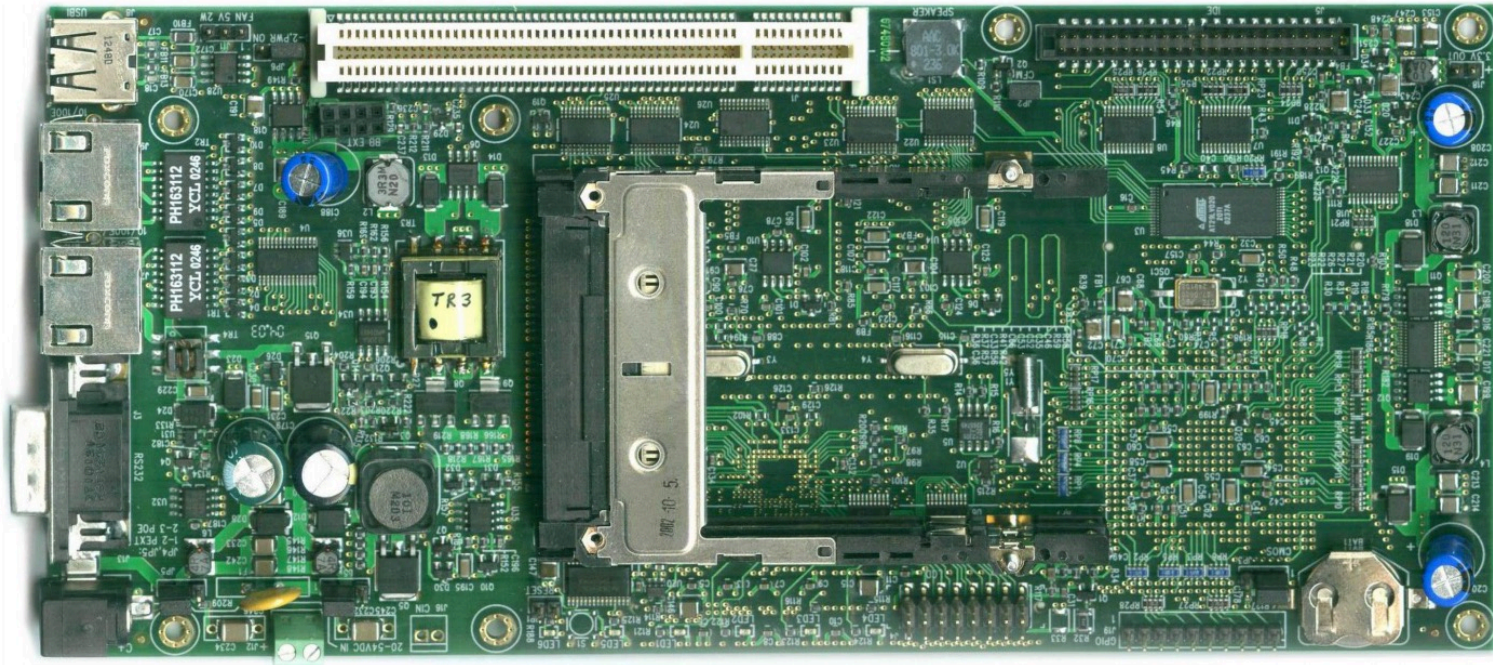
The screenshot shows the Mikrotik WinBox interface. At the top, the status bar displays: Memory: 3553.4 MiB, Uptime: 19:50:49, CPU: 1% (highlighted with a red box), and a checked 'Hide Pass' button. Below this is the 'Queue List' window, which has tabs for 'Simple Queues', 'Interface Queues', 'Queue Tree', and 'Queue Types'. The 'Simple Queues' tab is active, showing a table of queues. The table has columns for '#', 'Name', 'Target', 'Uploa...', 'Dow...', 'Upload', and 'Download'. The queues listed are simple-... with targets ranging from 172.16.1.1 to 172.16.1.10. The upload and download speeds are all 20.1 Mbps. To the right of the Queue List is a 'CPU' window showing a list of CPU loads for various processors, including cpu21 (26%), cpu24 (18%), and others.

#	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

CPU	Load ...
cpu21	26
cpu24	18
cpu15	9
cpu0	4
cpu4	4
cpu17	4
cpu1	3
cpu11	3
cpu22	3
cpu29	3
cpu31	3
cpu32	3

It's only 1% of CPU Load with Simple Queue

Lets use (not) so fast
routerboard!



RB230

RB44GV

Routerboard

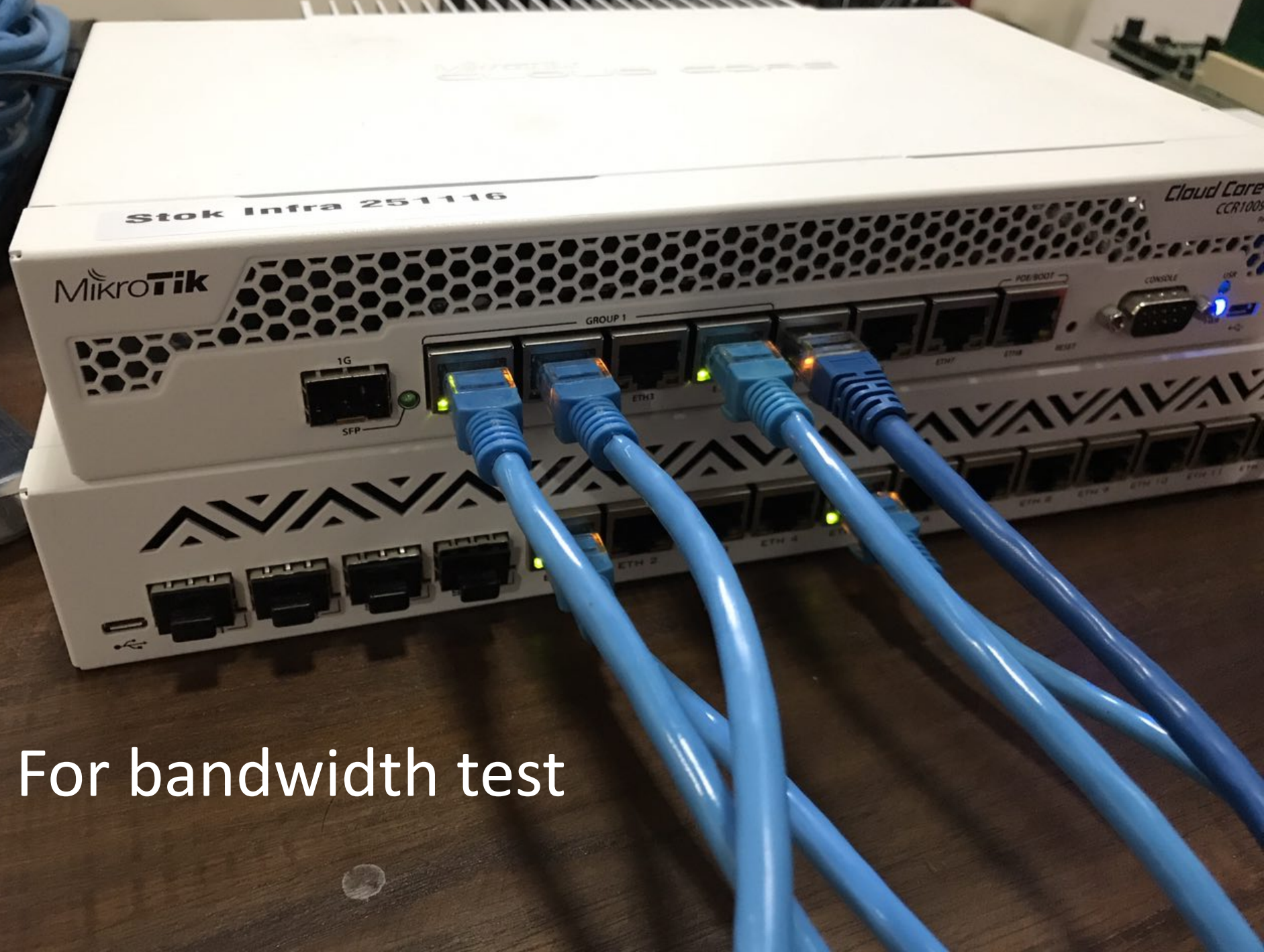
☒ Routerboard

Model: 230r3

Serial Number: 034B01DE3AD8

Current Firmware: 1.3.8 (apr/02/200...

Upgrade Firmware: 1.3.8 (apr/02/200...



For bandwidth test

Bridge							
Bridge		Ports	Filters	NAT	Hosts		
		Settings					
	Name	Type	L2 MTU	Tx	Rx	Tx Packet (p/s)	Rx Packet
R	bridge1	Bridge	1600	0 bps	0 bps		0

bridge

Bridge ports

Bridge					
Bridge		Ports	Filters	NAT	Hosts
	Interface	Bridge	Priority (...)	Path Cost	Role
	ether5	bridge1	80	10	designated port
	ether6	bridge1	80	10	designated port

Bridge Settings			
<input checked="" type="checkbox"/>	Use IP Firewall	<div>OK</div> <div>Cancel</div> <div>Apply</div>	
<input type="checkbox"/>	Use IP Firewall For VLAN		
<input type="checkbox"/>	Use IP Firewall For PPPoE		
<input checked="" type="checkbox"/>	Allow Fast Path		
<input checked="" type="checkbox"/>	Bridge Fast Path Active		
Bridge Fast Path Packets:	<input type="text" value="0"/>		
Bridge Fast Path Bytes:	<input type="text" value="0 B"/>		

Use IP firewall

Bandwidth Test (Running) □ ✕

Test To:

Protocol: ☐ udp ☒ tcp

Local UDP Tx Size:

Remote UDP Tx Size:

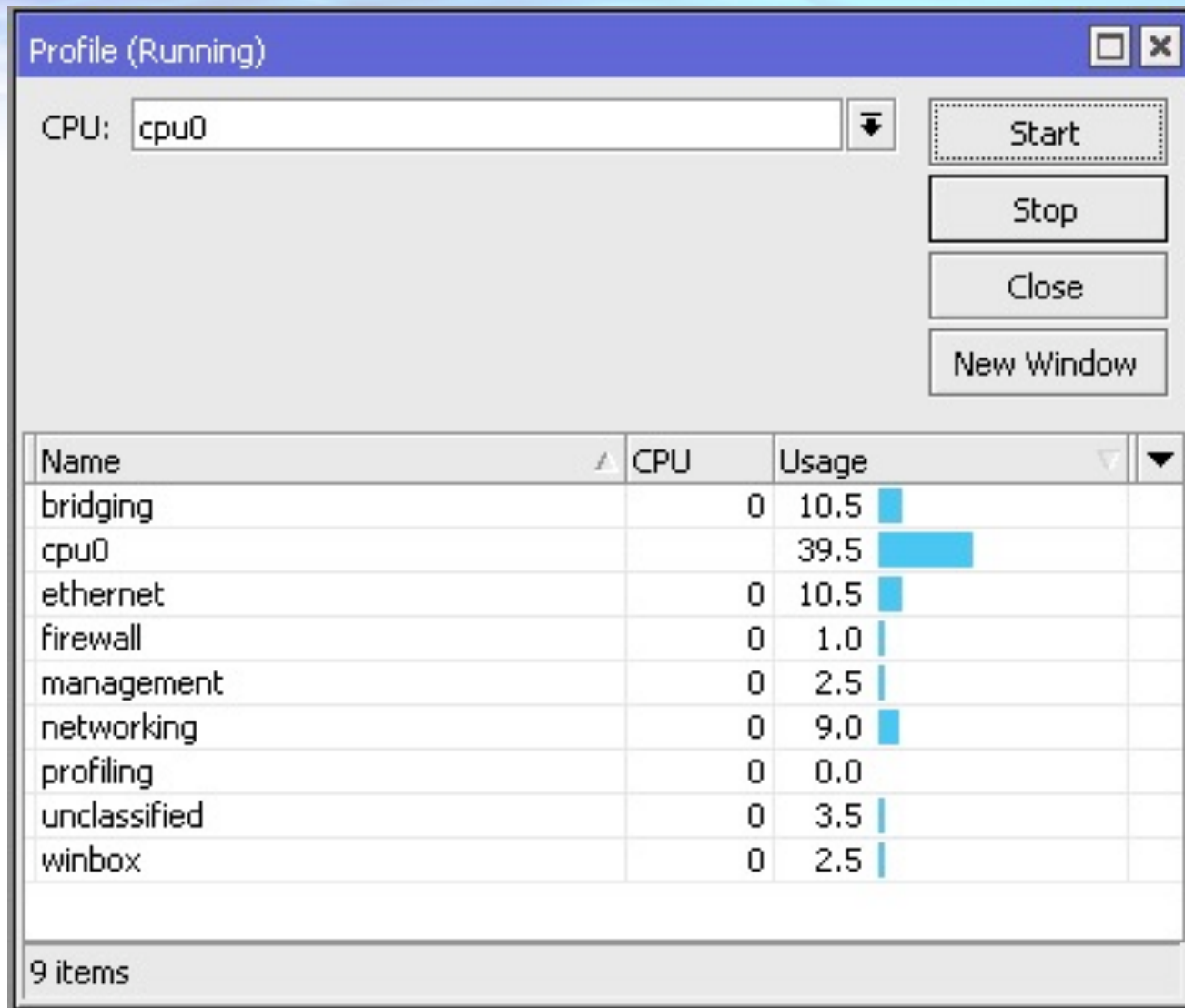
Direction: ▼

TCP Connection Count:

Local Tx Speed: ▲ bps

Remote Tx Speed: ▲ bps

Bandwidth test on CCR



Without queue

Simple Queue <queue1>

General

Advanced

Statistics

Traffic

Total

Total Statistics

Name:

queue1

Target:

10.10.10.2

⬇️⬆️

Dst.:

▼

Target Upload

Target Download

Max Limit:

10M

⬇️⬆️

10M

⬇️⬆️

bits/s

Queue List

Simple Queues

Interface Queues

Queue Tree

Queue Types

+

−

✓

✗

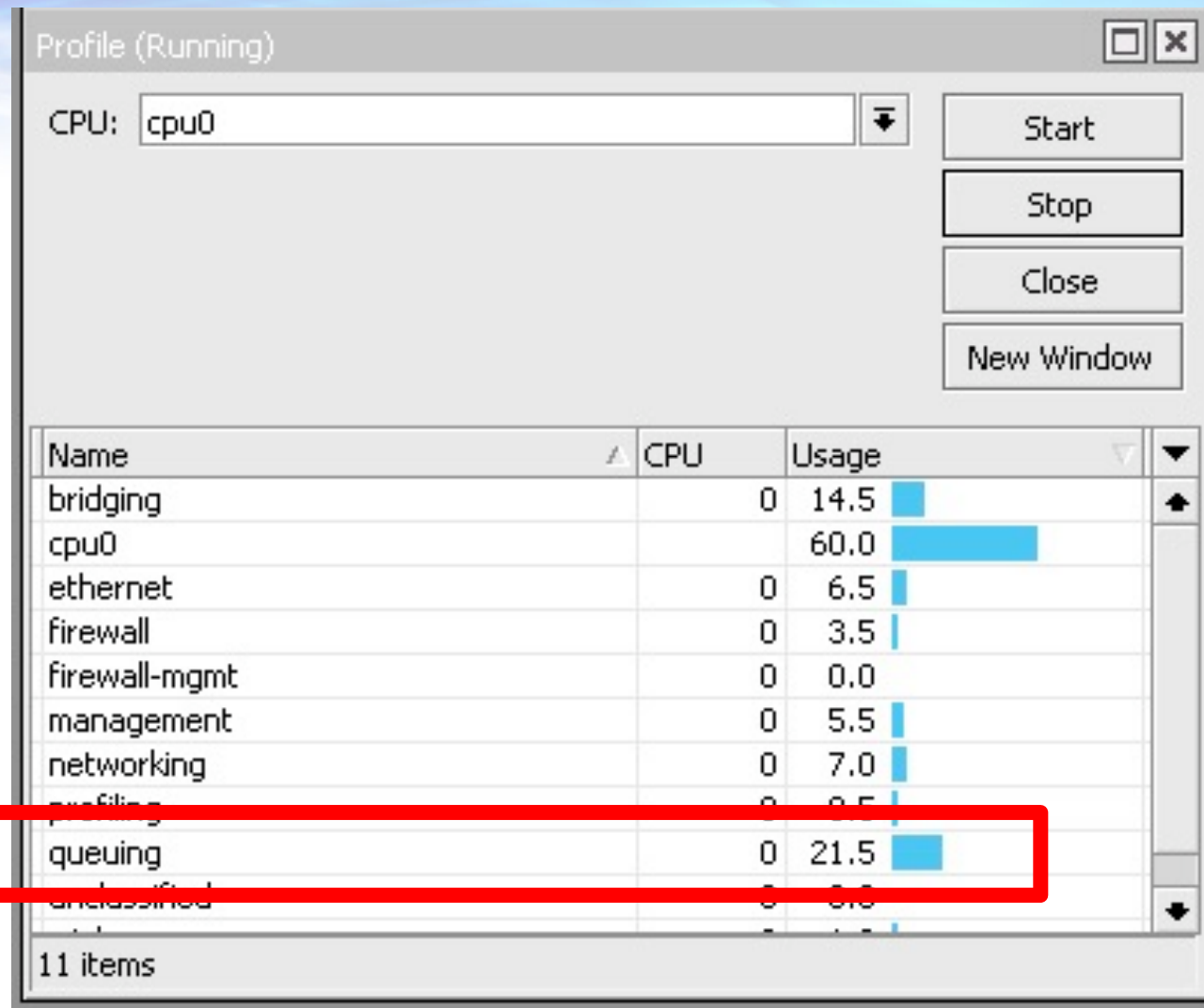
📄

🔍

⏮️ Reset Counters

⏮️ Reset All Counters

#	Name	Target	Uploa...	Downlo...	...	Upload Avg...	Download Av...	Tc
0	🔴 queue1	10.10.10.2	10M	10M		10.0 Mbps	10.0 Mbps	



With queue

Simple Queue <queue1>

General Advanced Statistics Traffic Total Total Statistics

Name: queue1

Target: 10.10.10.2

Dst.:

Target Upload Target Download

Max Limit: 10M 10M bits/s

Burst

Burst Limit: 14M 14M bits/s

Burst Threshold: 6M

Burst Time: 16

Time

Burst

Token
Bucket

Simple Queue <queue1>

General Advanced Statistics Traffic Total Total Statistics

Packet Marks:

Target Upload Target Download

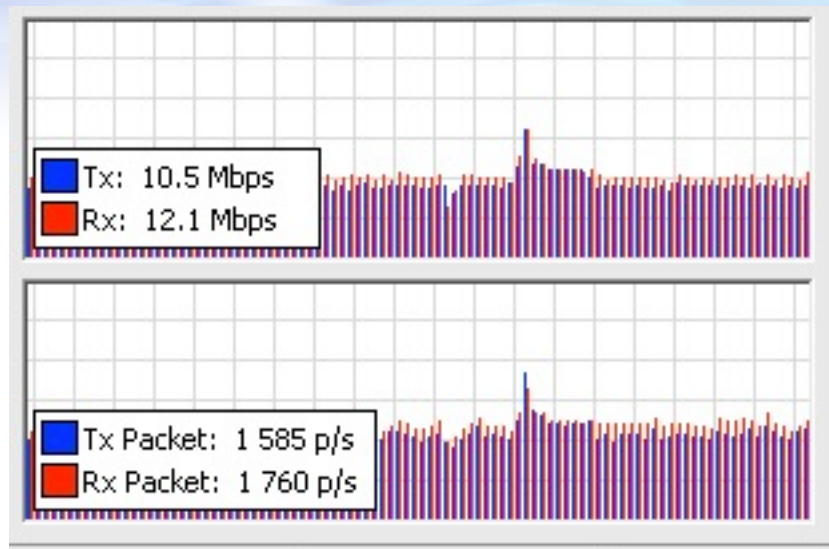
Limit At: unlimited unlimited bits/s

Priority: 8 8

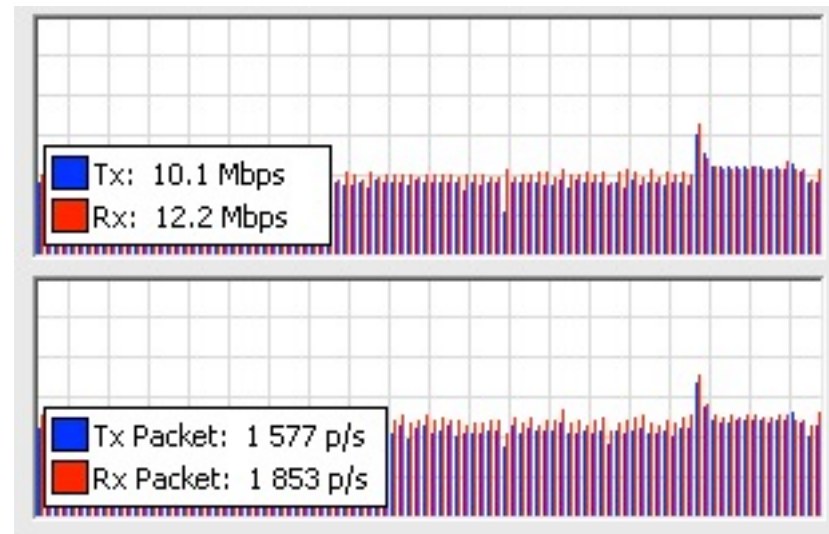
Bucket Size: 3,000 3,000 ratio

Queue Type: default-small default-small

Parent: none



With burst



with token bucket

Conclusions

- Token Bucket is alternatives to burst system
- Difference of efficiency between both system is still very small.

Thank you



Comments and suggestions:



www.mikrotik.id



info@mikrotik.co.id



[@mikrotik.id](https://www.facebook.com/mikrotik.id)



[@mikrotik.indonesia](https://www.instagram.com/mikrotik.indonesia)



[@mikrotik_id](https://www.twitter.com/mikrotik_id)



[Valens Riyadi](https://www.youtube.com/ValensRiyadi)

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.