



*HTB y Herramientas de Medida*

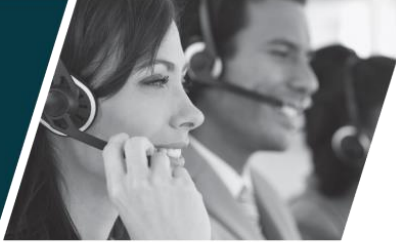
**MikroTik**



# Presentación Personal

- Juan Manuel Diaz Gómez
- Ingeniero de Telecomunicaciones Universidad Santo Tomás sede Medellín-Colombia
- Experiencia con Mikrotik desde el año 2008
- Certificaciones actuales: MTCNA-MTCTCE

# Presentación Personal





# Introducción

- Los que trabajamos con ISP's – WISP's en particular, siempre nos encontramos con que el cliente desea:
  - 1) Una red estable
  - 2) Que se garantice la velocidad
  - 3) Rapidez al abrir contenidos



# No queremos esto...

**SE CAYO LA RED!**

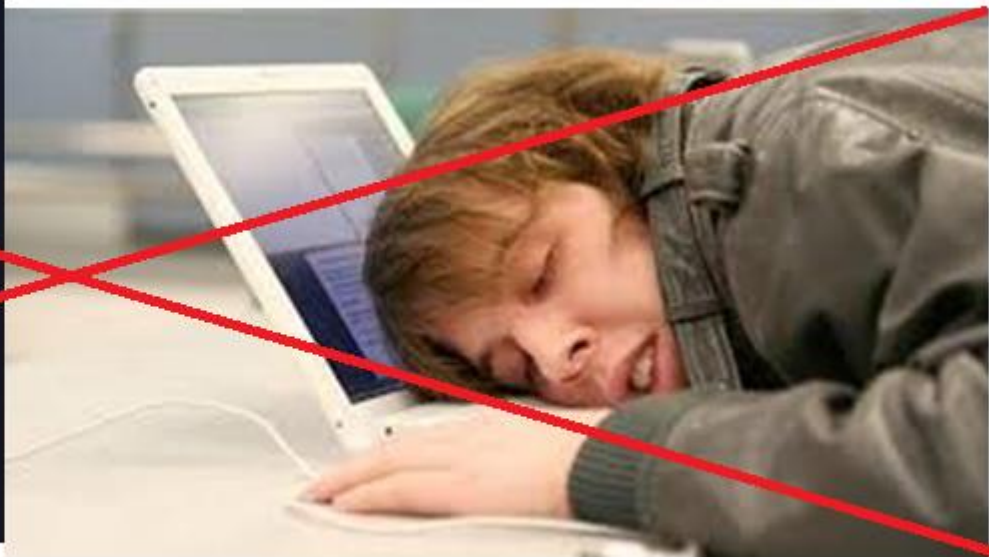


**VAMO A CALMARNO**

Imagen creada en [GeneradorMemes.com](https://www.generadorMemes.com)



# Ni tampoco esto!!!





# Queremos esto!!!



AulaFacil.com



# Objetivos

- Entender a fondo el funcionamiento y la estructura de HTB (Hierarchical Token Bucket)
- Implementar calidad de servicio en una red usando algoritmos de encolamiento de tráfico
- Usar las herramientas Traffic Generator y Bandwidth Test

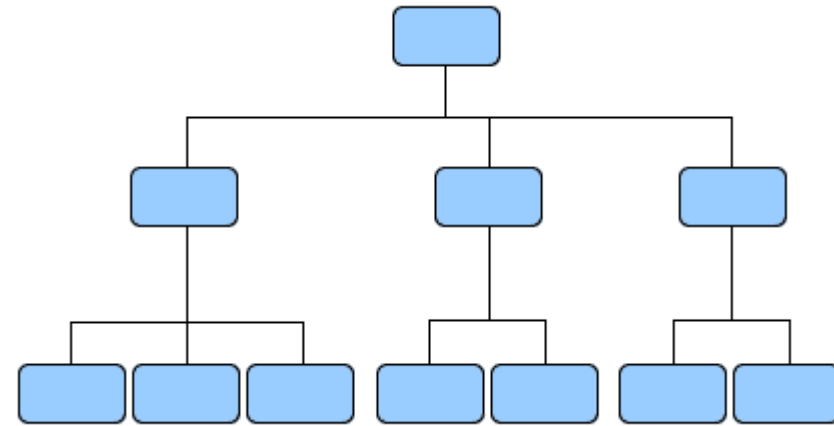


# HTB – HIERARCHICAL TOKEN BUCKET

 **MikroTik**

# Concepto de HTB

- Para realizar QoS, RouterOS se basa en el algoritmo Hierarchical Token Bucket
- Permite crear una estructura de colas jerárquicas, cada una con diferente prioridad
- La estructura jerárquica establece relaciones entre las colas
  - a) Padre – Hijo
  - b) hijo - hijo

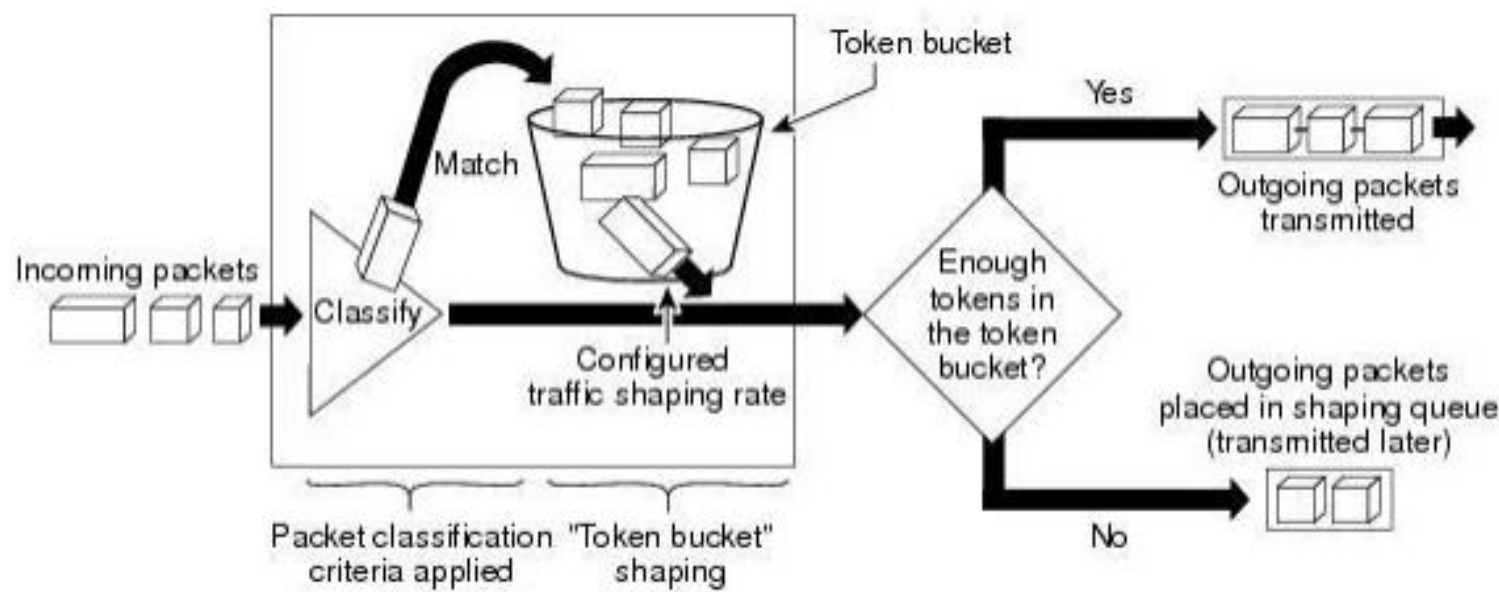


Captura de pantalla de la interfaz de configuración de colas de RouterOS, mostrando una lista de colas. El título es "Queue List". Hay pestañas para "Simple Queues", "Interface Queues", "Queue Tree" y "Queue Types". Hay botones para "+", "-", "✓", "✗", "📄", "🔍", "🔄" y "Reset Counters".

#	Name	Target	
0	all-limit	192.168.1.0/24	← Inner Queue
1	client-1	192.168.1.2	← } Leaf Queue
2	client-2	192.168.1.3	← }
3	client-3	192.168.1.4	← }
4	client-4	192.168.1.5	← }



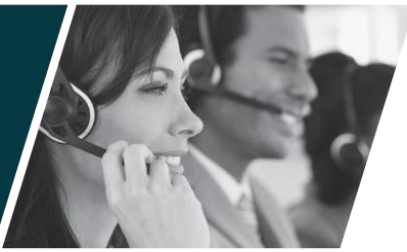
# Como trabaja HTB



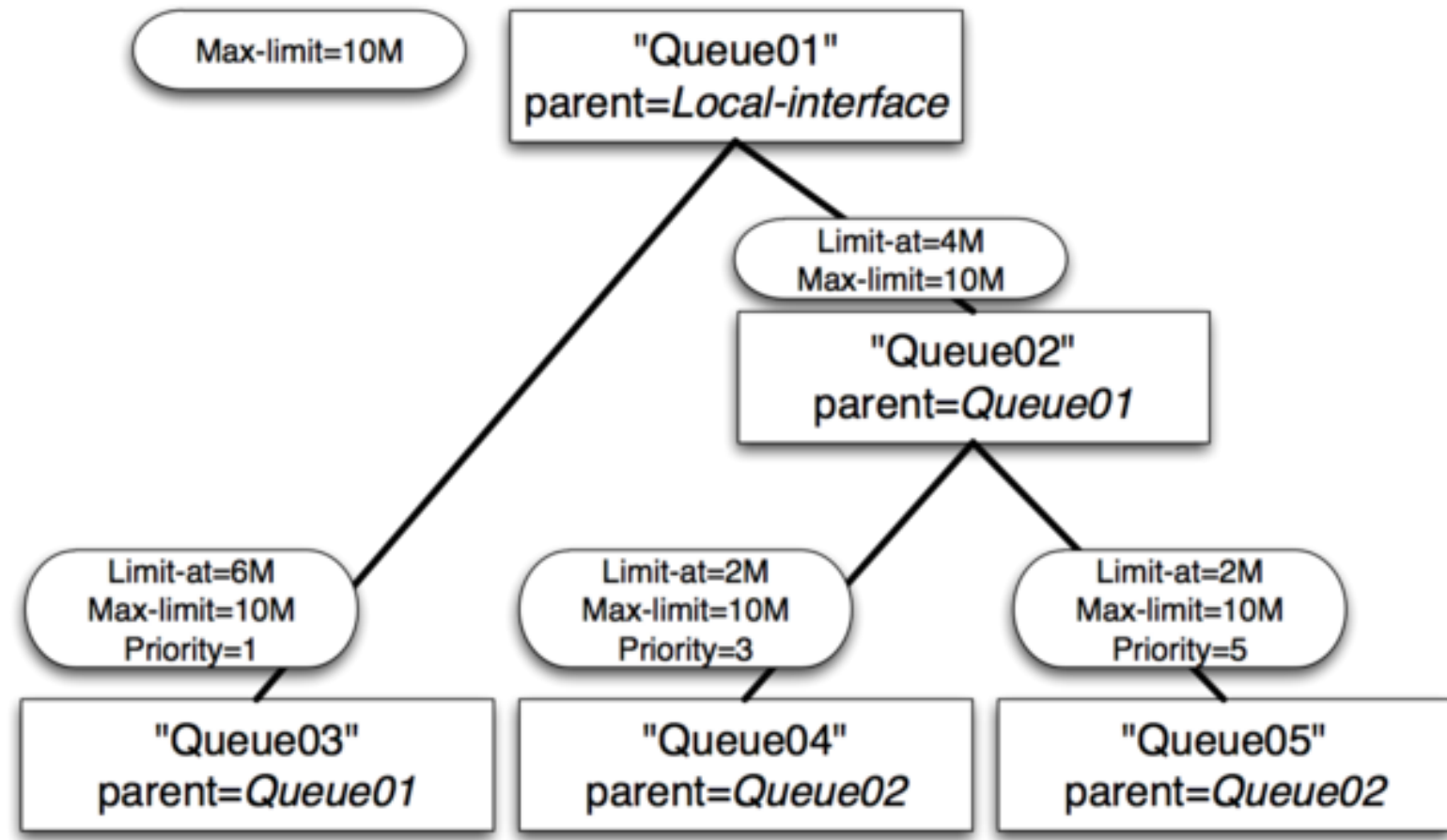


# Estructura HTB

- Una cola se convierte en cola Padre cuando tenga al menos una cola hija
- Las colas que van a consumir tráfico son las hijas, las padres lo reparten
- Las colas hijas lograrán satisfacer primero el limit-at y luego los padres reparten el tráfico restante



# Ejemplo





# Implementando HTB

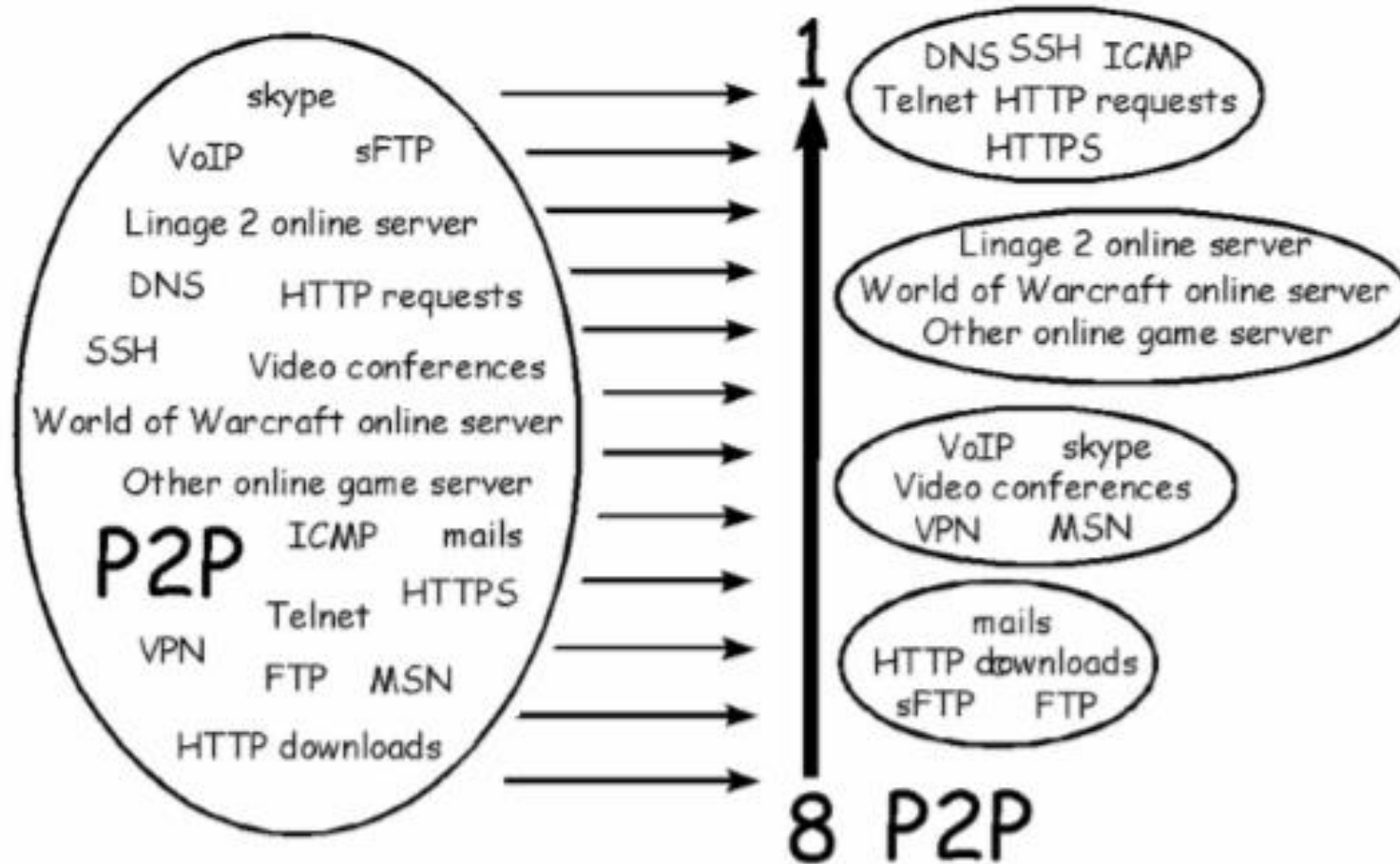
Para poder implementar HTB necesitamos seguir los siguientes pasos:

- Marcar conexiones
- Marcar paquetes
- Crear una cola padre de bajada
- Crear una cola padre de subida
- Crear las colas hijas de bajada
- Crear las colas hijas de subida

Nota: al tener el tráfico marcado podemos realizar la priorización de cada tipo de tráfico



# Implementando HTB





# Implementando HTB

4 Algoritmos para encolamiento de tráfico:

✓ FIFO

✓ RED

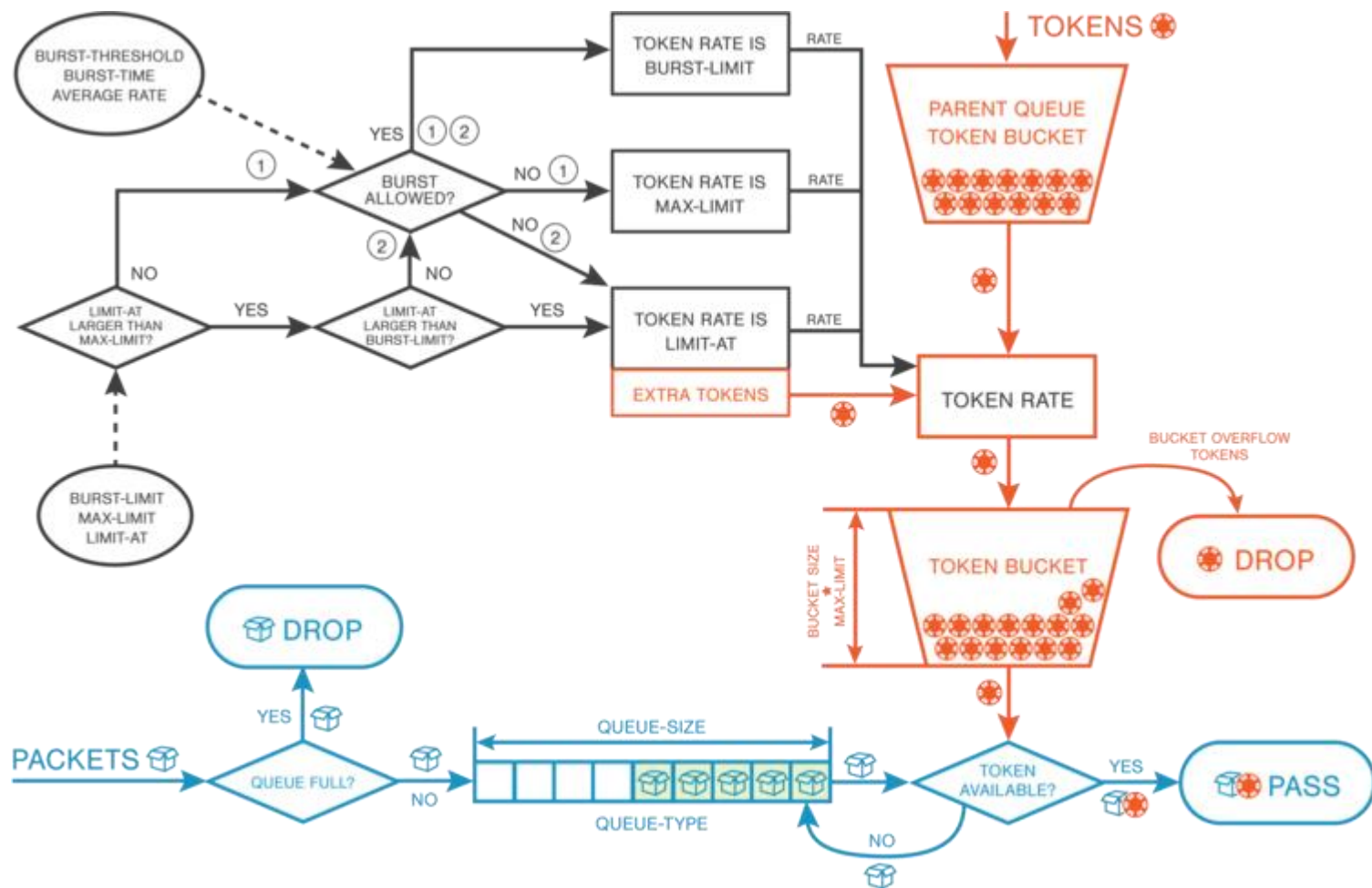
✓ SFQ

✓ PCQ





# Diagrama HTB





# Entendiendo el Diagrama HTB

El algoritmo HTB es una analogía a un balde, el cual tiene una capacidad, si se llena demasiado se derrama, en este caso se pierden o descartan los bytes (tokens)





# Prioridades HTB

- a) El limit-at CIR es el peor de los casos, lo mínimo que va a llegar al cliente
- b) El max-limit MIR se alcanza cuando el padre lo reparte a sus hijos, cuando las colas hijas tienen prioridad más alta lo alcanzan primero
- c) La prioridad más baja es 8, la más alta es 1



# Prioridades HTB

- a) El ancho de banda máximo de la cola padre debe ser igual o mayor que la sumas de las colas hijas
- b) El ancho de banda máximo de cualquier cola hija debe ser menor o igual que el ancho de banda máximo del padre

#	Name	Target	Download Max Limit	Download Limit At
0	all-limit	192.168.1.0/24	1M	unlimited
1	client-1	192.168.1.2	1M	256k
2	client-2	192.168.1.3	1M	256k
3	client-3	192.168.1.4	1M	256k
4	client-4	192.168.1.5	1M	256k



MIR

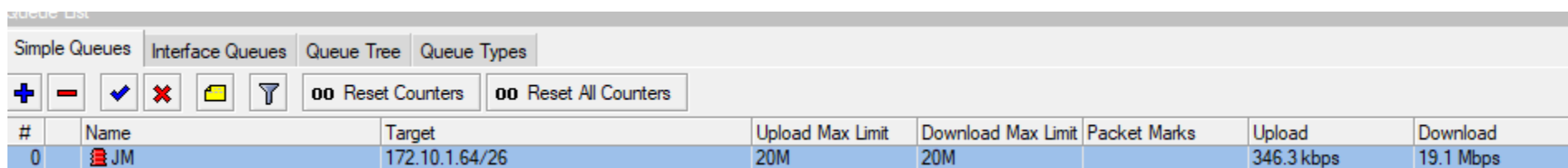


CIR

# Estados en queues

Dependiendo el consumo se tienen 3 estados ilustrados con colores, así:

- ▶ 0% - 50% available traffic used - green
- ▶ 51% - 75% available traffic used - yellow
- ▶ 76% - 100% available traffic used - red



Queue List

Simple Queues | Interface Queues | Queue Tree | Queue Types

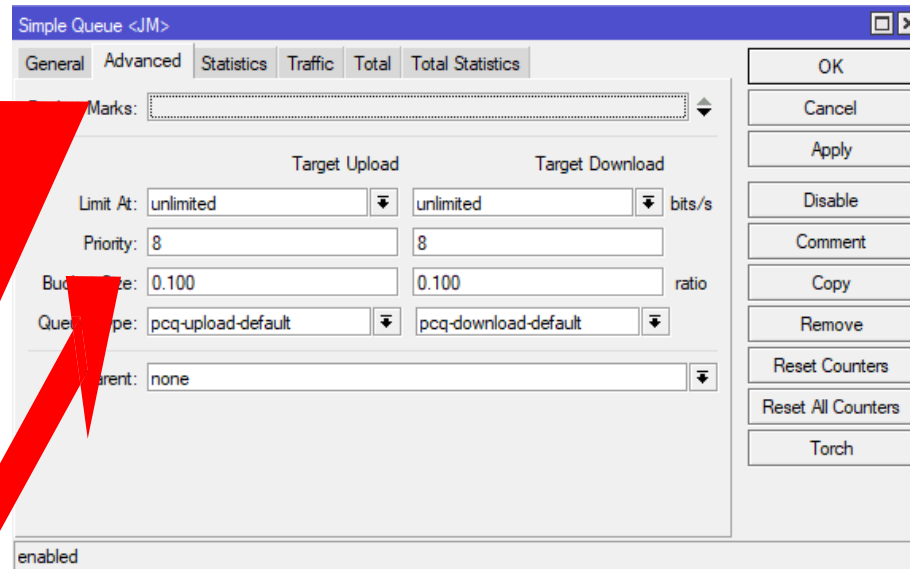
+ - ✓ ✗ 📁 📏 ⏪ ⏩ Reset Counters Reset All Counters

#	Name	Target	Upload Max Limit	Download Max Limit	Packet Marks	Upload	Download
0	JM	172.10.1.64/26	20M	20M		346.3 kbps	19.1 Mbps

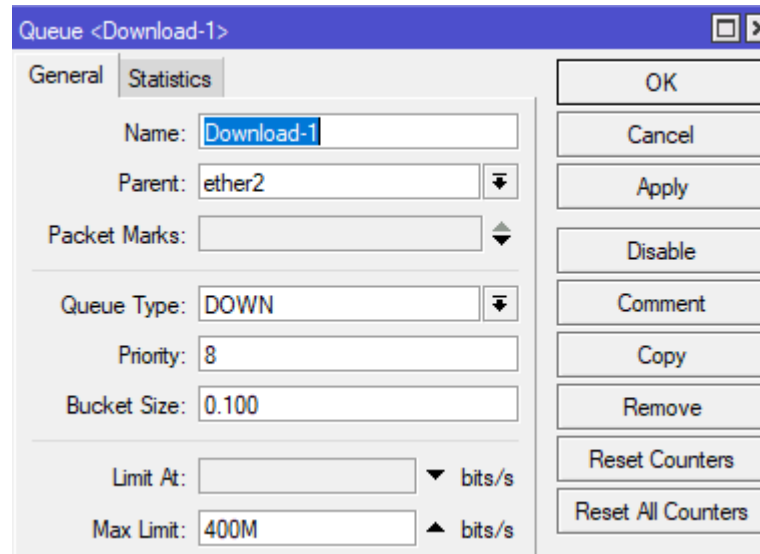
# Prioridad del tráfico

- Mikrotik maneja prioridad de 1 a 8 donde 1 es la más alta

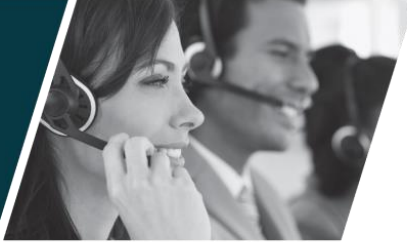
**Priority** está en la pestaña **Advanced**



The screenshot shows the 'Simple Queue <JM>' configuration window in Mikrotik WinBox. The 'Advanced' tab is selected. The 'Priority' field is set to 8. The 'Queue Type' is set to 'pcq-upload-default' for upload and 'pcq-download-default' for download. The 'Limit At' is set to 'unlimited' for both directions. The 'Bucket Size' is set to '0.100' ratio. The 'Parent' is set to 'none'. The 'enabled' checkbox is checked.



The screenshot shows the 'Queue <Download-1>' configuration window in Mikrotik WinBox. The 'General' tab is selected. The 'Name' is 'Download-1'. The 'Parent' is 'ether2'. The 'Queue Type' is 'DOWN'. The 'Priority' is 8. The 'Bucket Size' is '0.100'. The 'Limit At' is empty. The 'Max Limit' is '400M'. The 'enabled' checkbox is checked.



# Prioridad del tráfico

- La prioridad en colas simple rige siempre y cuando las colas dependan de un “Padre”, es decir que sean colas hijas

Queue List															
Simple Queues		Interface Queues		Queue Tree		Queue Types									
+		-		✓		✗		☰		☰		Reset Counters		Reset All Counters	
#	Name	▲	Target	Upload Max Limit	Download Max L										
0	COLA PADRE		0.0.0.0/0	5M	10M										
1	COLA HIJA1		0.0.0.0/0	unlimited	unlimited										
2	COLA HIJA2		0.0.0.0/0	unlimited	unlimited										
3	COLA HIJA3		0.0.0.0/0	unlimited	unlimited										



# Colas avanzadas – Queue Tree

- Reemplaza cientos de cola en sólo pocas
- Fija el mismo límite a cualquier usuario
- Ecuiliza el ancho de banda disponible entre los usuarios activos

Name	Parent	Packet ...	Limit At (b...	Max Limit ...
download	lan			
Corporativo_download	download		512k	1024k
corp_256_1_download	Corporativo_download	packet...	256k	512k
corp_256_2_download	Corporativo_download	packet...	256k	1024k
Residencial_download	download		512k	1024k
resi_128_1_download	Residencial_download	packet...	128k	512k
resi_128_2_download	Residencial_download	packet...	128k	512k
resi_256_1_download	Residencial_download		256k	512k
dns_resi_256_1_download	resi_256_1_download	packet...	10k	20k
http_resi_256_1_download	resi_256_1_download	packet...	100k	400k
other_resi_256_1_download	resi_256_1_download	packet...	100k	400k
sip_resi_256_1_download	resi_256_1_download	packet...	64k	128k
upload	wan			
Corporativo_upload	upload		512k	1024k
corp_256_1_upload	Corporativo_upload	packet...	256k	512k
corp_256_2_upload	Corporativo_upload	packet...	256k	1024k
Residencial_upload	upload		512k	1024k
resi_128_1_upload	Residencial_upload	packet...	128k	512k
resi_128_2_upload	Residencial_upload	packet...	128k	512k
resi_256_1_upload	Residencial_upload		256k	512k
dns_resi_256_1_upload	resi_256_1_upload	packet...	10k	20k
http_resi_256_1_upload	resi_256_1_upload	packet...	100k	400k
other_resi_256_1_upload	resi_256_1_upload	packet...	100k	400k
sip_resi_256_1_upload	resi_256_1_upload	packet...	64k	128k





# Colas avanzadas – Queue Tree

- Al usar PCQ en colas de árbol debemos tener presente:
  - 1) Realizar la suma de los hijos menores y restar el valor del **max-limit** del padre mayor
  - 2) Con el ancho de banda que nos queda disponible del padre mayor tenemos
    - a) Ver si los **limit-at** de los padres hijos están completos en caso contrario entregar ancho de banda al hijo con mayor **priority** de ese padre hasta satisfacer el **max-limit**



## Colas avanzadas – Queue Tree

Si está satisfecho su **limit-at** y el **limit-at** del padre hijo continuamos al siguiente grupo e iniciamos el paso 1.

b) Si después de completar los **limit-at** de los padres hijos queda ancho de banda libre:

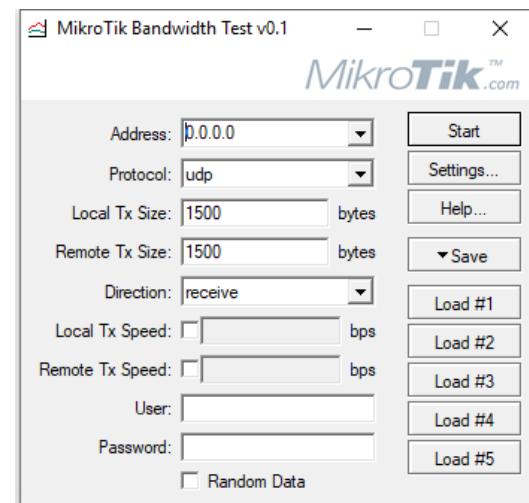
Asignar el ancho de banda restante al hijo con mayor **priority**

a) Si tenemos más de un hijo sin su **max-limit** completo debemos:

- Dividir por igual el ancho de banda restante y entregarlo hasta satisfacer sus **max-limit**
- Si solamente un hijo no tiene el **max-limit** satisfecho entregarle el ancho de banda disponible a él

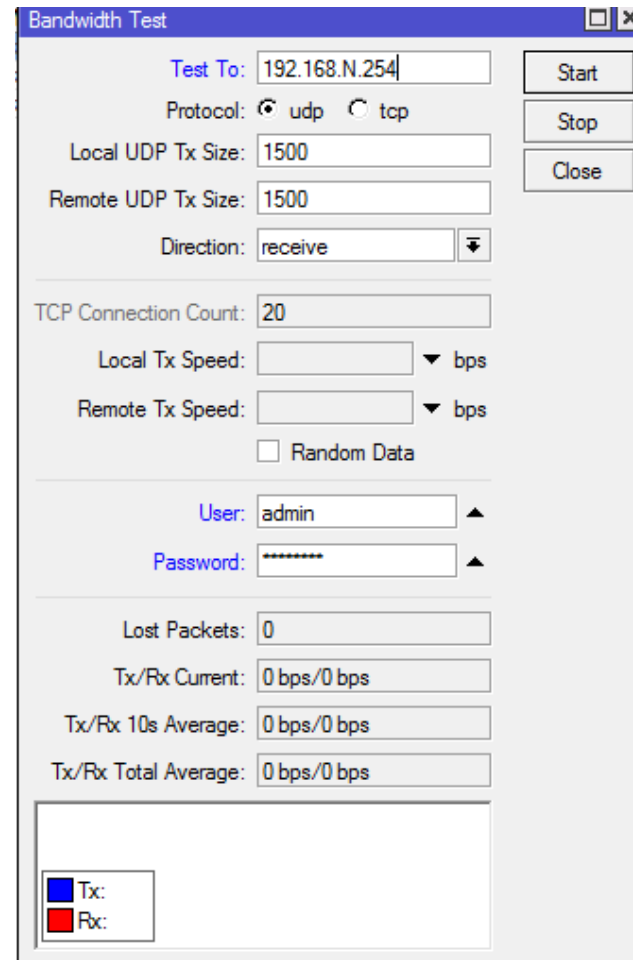
# Tools: Bandwidth Test

- La prueba del ancho de banda puede ser utilizada para monitorear el rendimiento hacia un dispositivo remoto.
- La prueba de ancho de banda funciona entre dos routers MikroTik (Anteriormente estaba disponible para usar en el pc)

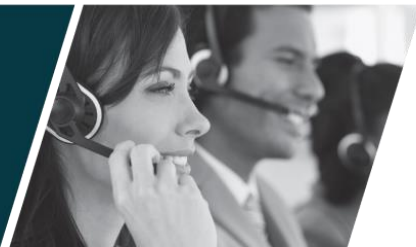


# Tools: Bandwidth Test

- Menú **tools** -> **Bandwidth Test**
- Fijar **Test To** como dirección para la prueba
- Seleccionar el protocolo
- TCP soporta multiples conexiones
- Hay que autenticarse con el router remoto por seguridad

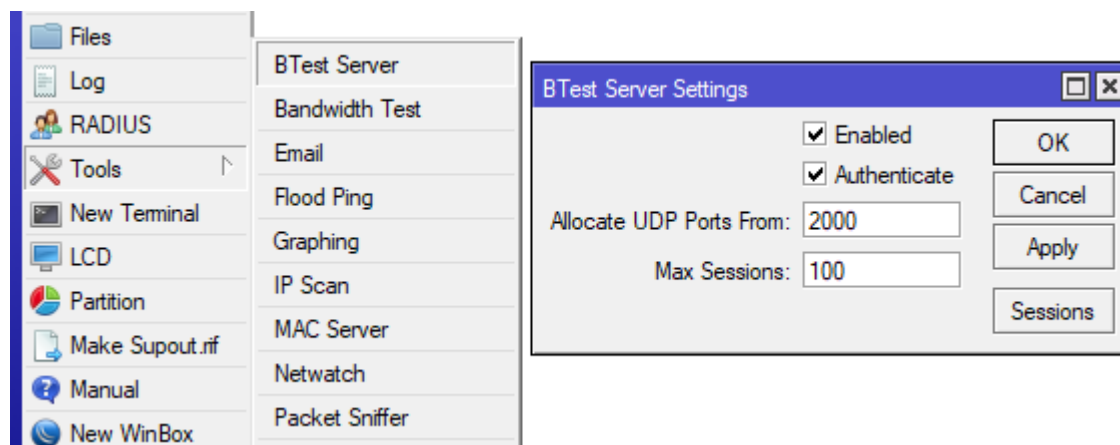


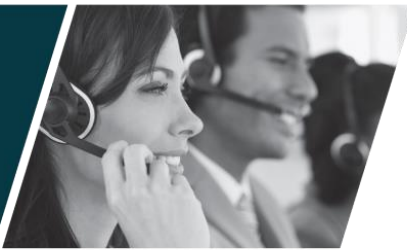
The screenshot shows the Mikrotik Bandwidth Test tool interface. The window title is "Bandwidth Test". The "Test To" field is set to "192.168.N.254". The "Protocol" is set to "udp". The "Local UDP Tx Size" and "Remote UDP Tx Size" are both set to "1500". The "Direction" is set to "receive". The "TCP Connection Count" is set to "20". The "Local Tx Speed" and "Remote Tx Speed" are both set to "0 bps". The "Random Data" checkbox is unchecked. The "User" field is set to "admin" and the "Password" field is masked with "\*\*\*\*\*". The "Lost Packets" field is set to "0". The "Tx/Rx Current", "Tx/Rx 10s Average", and "Tx/Rx Total Average" fields are all set to "0 bps/0 bps". A legend at the bottom left shows a blue square for "Tx:" and a red square for "Rx:".



# Tools: Bandwidth Test

- El servidor por defecto está habilitado
- Es recomendable dejar habilitada la autenticación





# Tools: Bandwidth Test

Bandwidth Test

Test To:  Start

Protocol:  udp  tcp Stop

Local UDP Tx Size:  ▼

Remote UDP Tx Size:  ▼

Direction:  ▼ Close

Connection Count:  ▼

Local Tx Speed:  ▼ bps

Remote Tx Speed:  ▼ bps

Random Data

User:  ▲

Password:  ▲

Lost Packets:

Tx/Rx Current:

Tx/Rx 10s Average:

Tx/Rx Total Average:

Tx:  Rx:

Bandwidth Test (Running)

Test To:  Start

Protocol:  udp  tcp Stop

Local UDP Tx Size:  ▼

Remote UDP Tx Size:  ▼

Direction:  ▼ Close

Connection Count:  ▼

Local Tx Speed:  ▲ bps

Remote Tx Speed:  ▼ bps

Random Data

User:  ▲

Password:  ▲

Lost Packets:

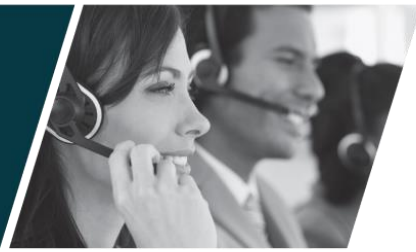
Tx/Rx Current:

Tx/Rx 10s Average:

Tx/Rx Total Average:

Tx: 100.0 Mbps  Rx: 1920 bps

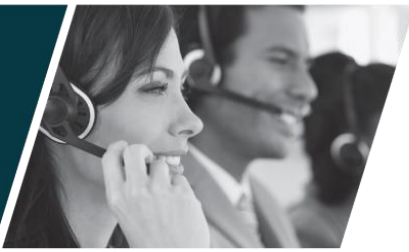
running...



# Tools: Traffic Generator

- Otra herramienta que nos permite realizar pruebas de ancho de bando es “traffic generator”
- Nos permite realizarla en un solo puerto o en múltiples puertos





# Tools: Traffic Generator

Vamos a realizar la prueba con un solo puerto, para esto se requieren al menos dos equipos

1) Generador de tráfico

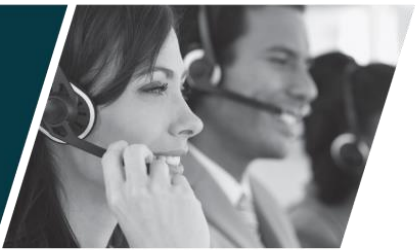
2) Equipo contra el que se realiza el test (DUT)



equipo generador de tráfico

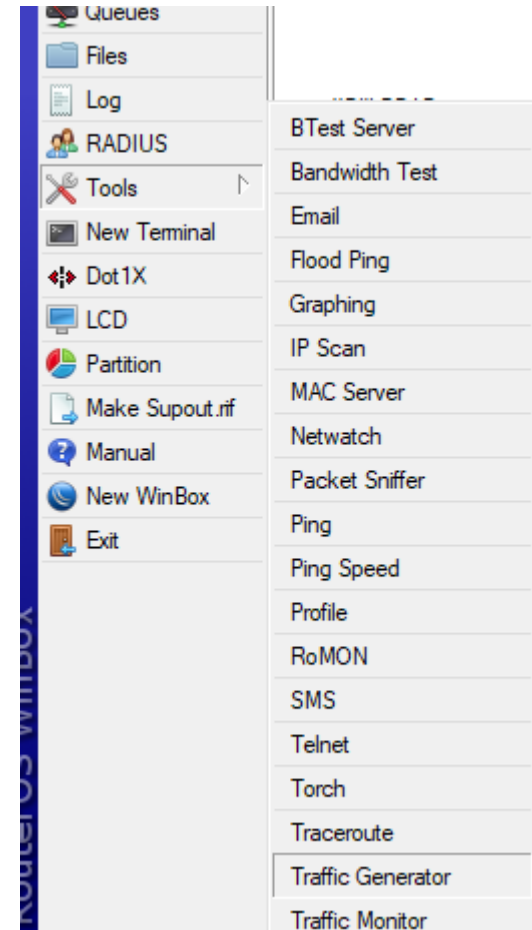
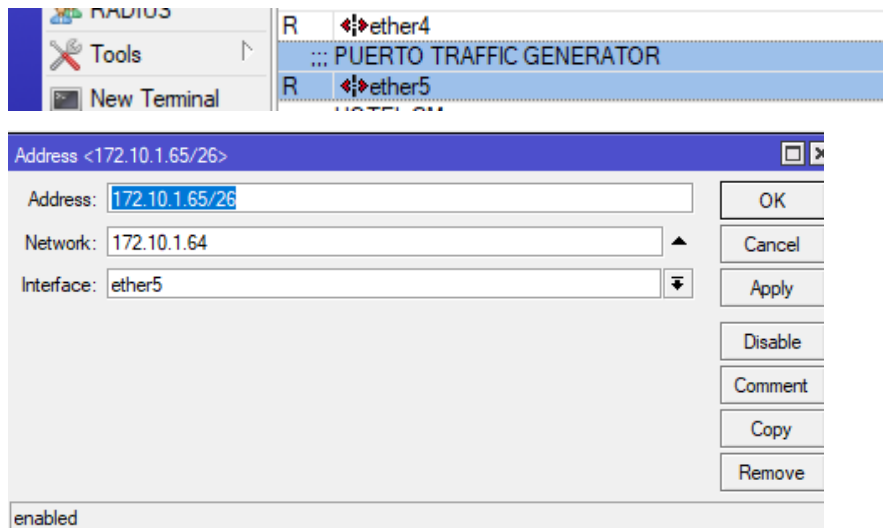


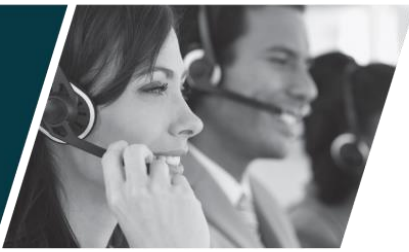




# Tools: Traffic Generator

1) Identificamos el puerto y la dirección IP, para este ejemplo ether5 con IP 172.10.1.65/26

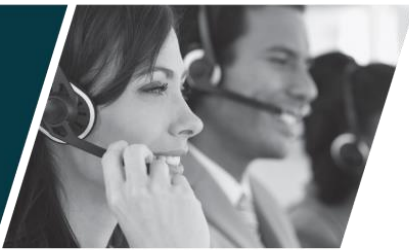




# Tools: Traffic Generator

2) Con los datos que tenemos procedemos a configurar

The screenshot displays two overlapping windows from the Mikrotik WinBox interface. The 'Traffic Generator Settings' window is in the foreground, showing fields for 'Test ID: 0', 'Latency Distribution Max.: 100 us', 'Stats Samples To Keep: 100', 'Latency Distribution Samples: 64', and 'Latency Distribution Measure Interval: 0-53.0ns'. A vertical column of buttons on the right includes 'OK', 'Cancel', 'Apply', 'Quick Start', 'Start', 'Stop', 'Inject Pcap', 'Stats', 'Ports', 'Packet Templates', 'Raw Packet Templates', and 'Streams'. The 'Traffic Generator Ports' window is partially visible behind it, showing a table with columns for 'Name', 'Interface', and 'First Header'. A 'New Traffic Generator Port' dialog box is open over it, with 'Name' set to 'port1', 'Interface' set to 'ether5', and 'First Header' empty. The dialog has buttons for 'OK', 'Cancel', 'Apply', 'Disable', 'Copy', and 'Remove'. The status 'enabled' is shown at the bottom of the dialog, and '0 items' is displayed at the bottom of the main window.



# Tools: Traffic Generator

3) Con los datos que tenemos procedemos a configurar

Traffic Generator Settings

Test ID: 0

Latency Distribution Max.: 100 us

Stats Samples To Keep: 100

Latency Distribution Samples: 64

Latency Distribution Measure Interval: 0-53.0ns

Running: no

Packet Template <trafficum>

General MAC IP UDP

Src: 172.10.1.65

Dst: 172.10.1.70

Protocol: [dropdown]

Gateway: [dropdown]

DSCP: [input]

IP ID: [input]

Frag. Offset: [input]

TTL: [input]

Assumed Src: [input]

Assumed Dst: [input]

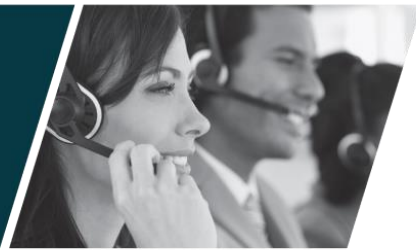
Assumed Protocol: 17 (udp)

Assumed DSCP: 0

Assumed IP ID: 0

Assumed Frag. Offset: 0

Assumed TTL: 64



# Tools: Traffic Generator

4) Con los datos que tenemos procedemos a configurar

Packet Stream <stream1>

Name:

Default Port:

Port:  ▼

ID:

Packet Size:

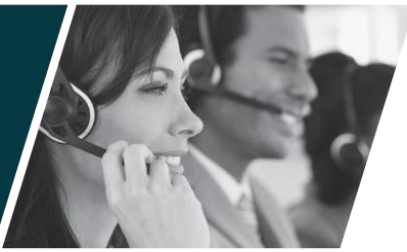
MBPS:  ▼

PPS:  ▼

Tx Template:  ▼

enabled

Buttons: OK, Cancel, Apply, Disable, Copy, Remove



# Tools: Traffic Generator

5) El último paso es correr la prueba

Quick Start (Running)

Test ID: 0

Stream:

Port:

Interface:

Packet Size:

PPS:

MBPS: 50

Tx Template: trafficum

Seq	ID	Tx Packets	Tx Rate	Rx Packets	Rx Rate	Lost Packets	Lost Rate	Lat. Min.	Lat. A...	Lat. M...	Jitter
1	0	4 121	49.9 Mbps	0	0 bps	4 121	49.9 Mbps				
2	0	4 127	49.9 Mbps	0	0 bps	4 127	49.9 Mbps				
3	0	4 129	50.0 Mbps	0	0 bps	4 129	50.0 Mbps				
4	0	4 127	49.9 Mbps	0	0 bps	4 127	49.9 Mbps				
5	0	4 128	49.9 Mbps	0	0 bps	4 128	49.9 Mbps				
6	0	4 129	50.0 Mbps	0	0 bps	4 129	50.0 Mbps				
7	0	4 128	49.9 Mbps	0	0 bps	4 128	49.9 Mbps				
8	0	4 128	49.9 Mbps	0	0 bps	4 128	49.9 Mbps				
9	0	4 128	49.9 Mbps	0	0 bps	4 128	49.9 Mbps				
10	0	4 128	49.9 Mbps	0	0 bps	4 128	49.9 Mbps				
11	0	4 128	49.9 Mbps	0	0 bps	4 128	49.9 Mbps				
12	0	4 128	49.9 Mbps	0	0 bps	4 128	49.9 Mbps				
TOT	0	49 529	49.9 Mbps	0	0 bps	49 529	49.9 Mbps				

13 items



# Tools: Traffic Generator

6) En la diapositiva anterior observamos que el tráfico es en un solo sentido tx, para que se ejecute en ambos tx y rx hacemos lo siguiente:

Packet Template <trafficum>

General MAC IP UDP

Src.:

Dst.: 172.10.1.65

Protocol:

Gateway: 172.10.1.70

DSCP:

IP ID:

Frag. Offset:

TTL:

Assumed Src.:

Assumed Dst.:

Assumed Protocol: 17 (udp)

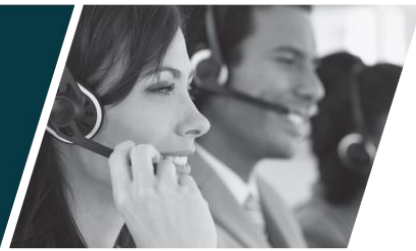
Assumed DSCP: 0

Assumed IP ID: 0

Assumed Frag. Offset: 0

Assumed TTL: 64

OK  
Cancel  
Apply  
Comment  
Copy  
Remove



# Tools: Traffic Generator

7) Repetimos la prueba:

Quick Start (Running)

Test ID: 0

Stream:

Port:

Interface:

Packet Size:

PPS:

MBPS: 50

Tx Template: trafficum

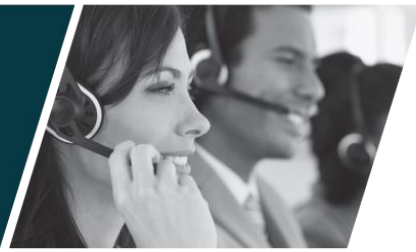
Start

Stop

Close

New Window

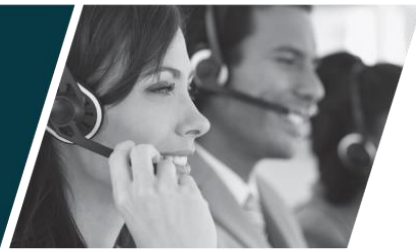
Seq	ID	Tx Packets	Tx Rate	Rx Packets	Rx Rate	Lost Packets	Lost Rate	Lat. Min.	Lat. A...	Lat. M...	Jitter
1	0	4 119	49.8 Mbps	4 119	49.8 Mbps	0	0 bps	305.us	526.us	3.10ms	2.80ms
2	0	4 128	49.9 Mbps	4 128	49.9 Mbps	0	0 bps	305.us	533.us	3.04ms	2.73ms
3	0	4 129	50.0 Mbps	4 129	50.0 Mbps	0	0 bps	305.us	528.us	3.01ms	2.71ms
4	0	4 128	49.9 Mbps	4 128	49.9 Mbps	0	0 bps	305.us	489.us	2.91ms	2.61ms
5	0	4 128	49.9 Mbps	4 128	49.9 Mbps	0	0 bps	305.us	332.us	2.05ms	1.75ms
6	0	4 128	49.9 Mbps	4 128	49.9 Mbps	0	0 bps	305.us	353.us	3.14ms	2.83ms
TOT	0	24 760	49.9 Mbps	24 760	49.9 Mbps	0	0 bps	305.us	460.us	3.14ms	2.83ms



# ¿PREGUNTAS?







# LINKS DE REFERENCIA

- [https://wiki.mikrotik.com/wiki/Manual:HTB-Token Bucket Algorithm](https://wiki.mikrotik.com/wiki/Manual:HTB-Token_Bucket_Algorithm)
- <https://wiki.mikrotik.com/wiki/Manual:Queue>
- [https://wiki.mikrotik.com/wiki/Manual:Queues - PCQ](https://wiki.mikrotik.com/wiki/Manual:Queues_-_PCQ)
- [https://wiki.mikrotik.com/wiki/Manual:Tools/Traffic Generator](https://wiki.mikrotik.com/wiki/Manual:Tools/Traffic_Generator)

*Gracias*

