Controlling Network Traffic using MikroTik RouterOS

MikroTik User Meeting

Venice 2014

By Ron Touw

LinITX, England

- My background?
- RF Wireless Engineering for UK Government for 25 years.
- Training certifications from companies such as Marconi, Hewlett-Packard, Rohde & Schwarz, Microsoft, Ruckus, Meru.
- User of MikroTik since 2006
- Certified Consultant and Trainer since 2009.

Who are LinITX?

- Largest MikroTik Distributor in UK
- Largest number of MikroTik Certified Consultants in one company in the UK
- Certified MikroTik Training Centre
- Provide Consultancy and Third Line Support

This sound familiar?

- Start up : few customers, everyone is happy.
- Initial growth : more new customers, but old customers no longer happy “It used to be much faster”. ☹
- Even more growth: Now *everyone* is unhappy. “If there was a better ISP around here, I would have moved to them by now”. ☹

Solution?

- Buy more bandwidth!
- Thank you for listening!

Reality?

- Buying more bandwidth is not always the available solution
- ‘Bandwidth’ can be very expensive.
- E.g. Guernsey C&W fibre pricing:
  - 2MB €6,800 per annum
  - 40MB €77,500 per annum

Real Solution?

- Simple - **Use less bandwidth!**
- But how?
- Managing users' usage by:
  - Identifying and allowing 'good traffic'
  - Identifying and restricting 'bad traffic'
- Identify at edge, mark with DSCP
- Apply priorities along the whole traffic route inside network, including wireless links

Stage1: Identifying

- "…**when you have eliminated the impossible, whatever remains, *however improbable*, must be the truth?**"
  ('Sherlock Holmes' / Arthur Conan Doyle)

- Identify what you immediately know to be good or bad traffic
- Analyse what is left – good/bad?
- Anything unknown / left over, assume is **bad**

Stage2: Marking

- Use Connection Tracking to your advantage!
  - Detect traffic by some uniqueness
  - Apply ‘Connection Mark’
  - Apply ‘Packet Mark’
  - Change DSCP value
  - Add Remote IP to ‘Address List’ to reduce detection requirement for similar traffic from other clients’ connections

Stage3: Managing

- Queue Tree using PCQ and SFQ queue types
- (Understanding the ‘Queue Size’)
- Utilise bursting for ‘Interactive’ traffic (HTTP)
- Higher priority for small ‘TCP ACK’ packets
- Use priorities on any wireless links
- Use NV2 with built-in QoS ☺

Stage4: Monitoring

- Applying Traffic Management is not a ‘one off’ solution
- Constant pro-active monitoring and maintenance required
- React quickly to customer ‘complaints’ of any slow down after applying traffic management
- Identify their traffic needs and identify any good traffic

Identifying

Port	Protocol	Comments
20/21	tcp	FTP
53	tcp/udp	DNS
22	tcp	SSH,SFTP
80	tcp	HTTP
123	udp	SNTP
443	tcp	HTTPS
500	udp	IPSec
1701	udp	L2TP
1723	tcp	PPTP

Problems of identifying **too** simply!

- E.g. HTTP on Port 80 and HTTPS on 443. Not **ONLY** used by HTTP traffic!
- Therefore it cannot be assumed traffic is good **just** because of the protocol and port! ☹
- Solution? Layer 7!
- However, Layer 7 is CPU intensive ☹
- But, **why** is Layer 7 processing difficult for CPU?

Layer 7

- Layer 7 is the payload part of packet, the user data.
- Layer 7 detection thus means we have to open every single packet and inspect the payload itself.
- Layer 7 mangle rules on RouterOS, takes the first 10 packets of any new connection, or the first 2KB.
- It then stores this in a buffer and begins to match our ‘regex string’ against this user data.
- But what if packets are small? It is possible that after say 9 packets, it is only the **10th** packet that is a match.
- What happened to the first 9 packets?
- In this example, only the 10th packet triggered an action, the previous 9 – nothing happened.
- Therefore it is not recommended to take direct action on packets, but instead use address lists or connection mark

Using Mangle rules effectively

- Jump to ‘user-created’ chains then process there

```
/ip firewall mangle
add action=jump chain=forward comment="Detect all
ICMP traffic" disabled=no jump-target=ICMP_Chain
protocol=icmp
```

- Use Firewall Connection Tracking to your advantage by using connection marks - yes, even with so-called ‘connection-less’ protocols ☺

Connection Mark | contains | UDP | Filter

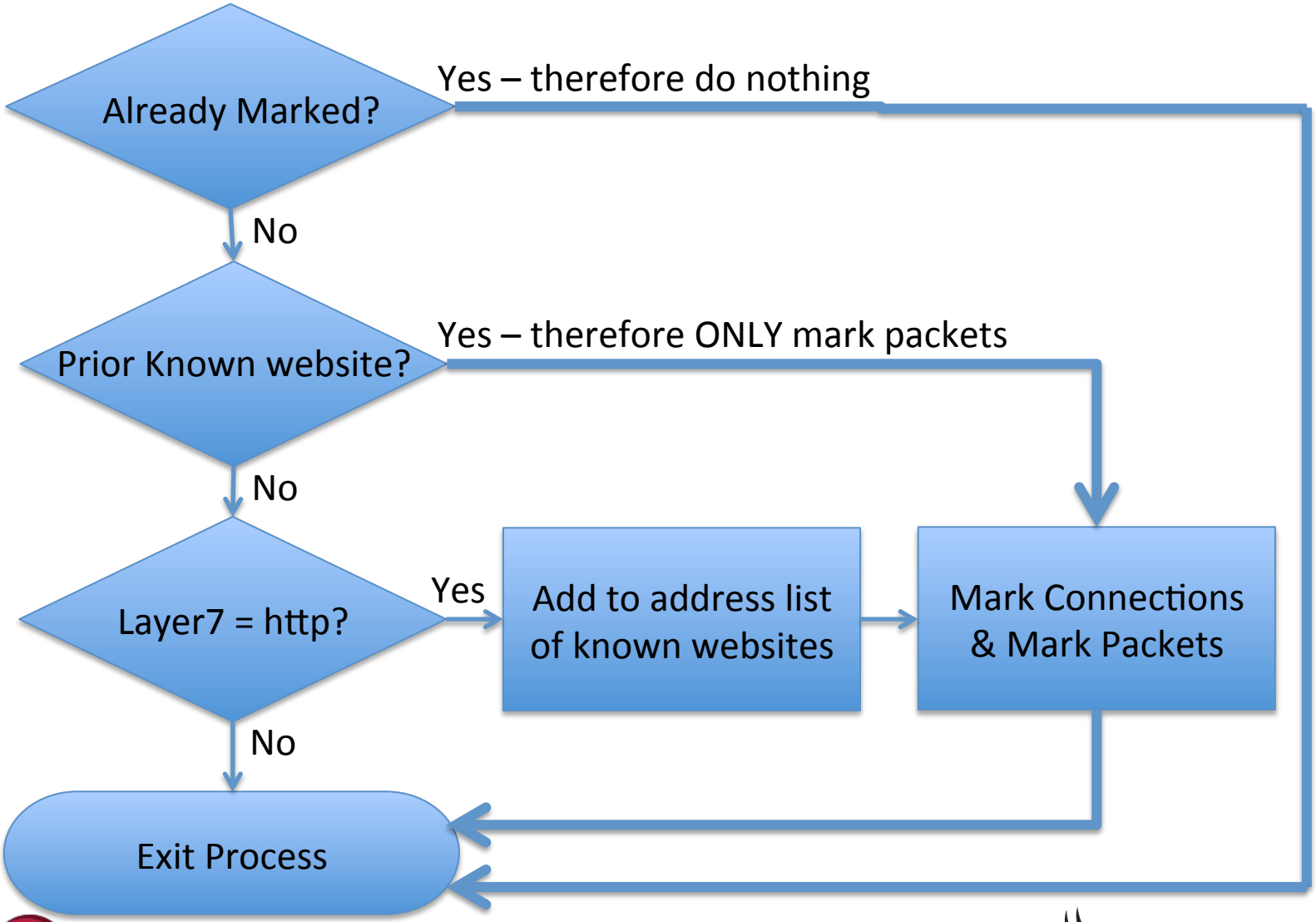
Src. Address	Dst. Address	Protocol	Connecti.	Conne...	2P	Timeout	TCP State
192.168.1.2:36934	90.207.238.97:53	17 (udp)		UDP		00:00:05	
192.168.2.198:58604	192.168.2.1:53	17 (udp)		UDP		00:00:05	
192.168.2.198:60680	192.168.2.1:53	17 (udp)		UDP		00:00:02	

Using Mangle rules effectively

- Ensure that high volume / high demand traffic is detected in higher up rules, move lower traffic demand rules further down – lowers CPU usage

- Reduce CPU by only testing a destination IP **once** by using address lists – therefore testing anyone's connection to that IP only the once ☺ - But how?

Using Layer 7 effectively



Using Layer 7 effectively

- 1. Inspect first packet of a connection but only if not already marked
- 2. If positively identified as a specific protocol – add destination IP to a long term dynamic address list
- 3. This destination server will not need inspecting again for a long time for **any other connections** from **any** other clients. ☺
- Therefore if it is a popular destination, it will only need testing **once**! ☺

Using L7 rules effectively

```
/ip firewall mangle
add action=jump chain=forward comment="Jump to HTTP
detection for all port 80 traffic if not examined
before" disabled=no jump-target=Test4WebSite
protocol=tcp connection-mark=no-mark src-port=80
```

- Test for TCP port 80 and jump to the L7 HTTP detection chain ‘Test4WebSite’, but only if NOT tested before (**connection-mark=no-mark**)
- ‘no-mark’ is a special RouterOS reserved name meaning that no mark has yet been applied

Using L7 rules effectively

```
add action=add-src-to-address-list address-
list=WebServer address-list-timeout=4w2d
chain=Test4WebSite comment="Detect if traffic is HTTP.
Add to address list WebServer if it is"
layer7-protocol=http-fast src-address-list=!WebServer
```

- Add IP to address list ‘WebServer’ for a month (4w2d) if :
  - a) it’s not already in the address list Webserver &
  - b) the L7 rule ‘http-fast’ successfully matched for a ‘http’ header
- Address List ‘WebServer’ will then contain a list of every web server IP visited by clients

Using L7 rules effectively

```
add action=mark-connection chain=Test4WebSite
new-connection-mark=WebServer src-address-
list=WebServer

add action=mark-packet chain=Test4WebSite
connection-mark=WebServer new-packet-mark=WebServer
passthrough=no
```

- In the user defined chain, mark connections and then mark packets if the source IP is in address list ‘WebServer’

L7 Packet Sniffing

- RouterOS examines the payload (the data inside a packet) for the first 10 packets or 2KB whichever is the smaller
- Cannot detect any strings inside SSL connections (obviously! – it's **encrypted**!)
- Look for initial certificate handshake instead ☺

```
/ip firewall layer7-protocol
add name=validcertssl regexp="^(.\?.\?\\x16\\x03.*\\x16\
\x03|.\?.\?\\x01\\x03\\x01\?.*\\x0b).*(thawte|equifax
secure|rsa data security, inc|verisign, inc|gte
cybertrust root|entrust\\.net limited)"
```

Layer 7 Filter Resources

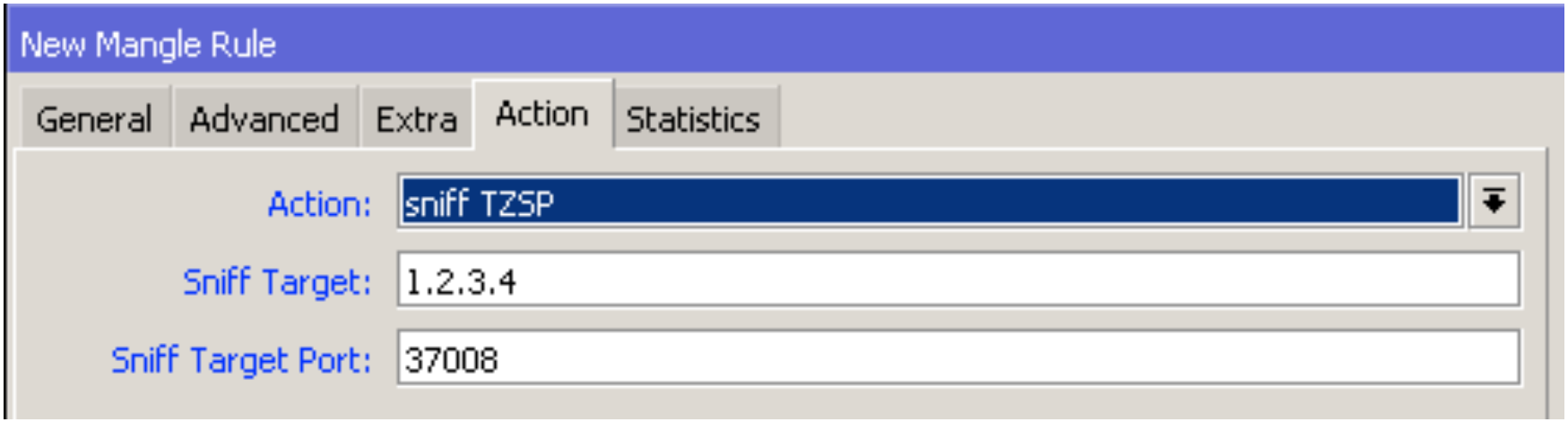
- MikroTik of course!
- http://wiki.mikrotik.com/wiki/Manual:IP/Firewall/L7
- http://www.mikrotik.com/download/l7-protos.rsc

- http://en.wikipedia.org/wiki/Regular_expression
- http://www.grymoire.com/Unix/Regular.html
- http://gskinner.com/RegExr/

- If all else fails – packet sniff and roll your own ☹

Packet Capture

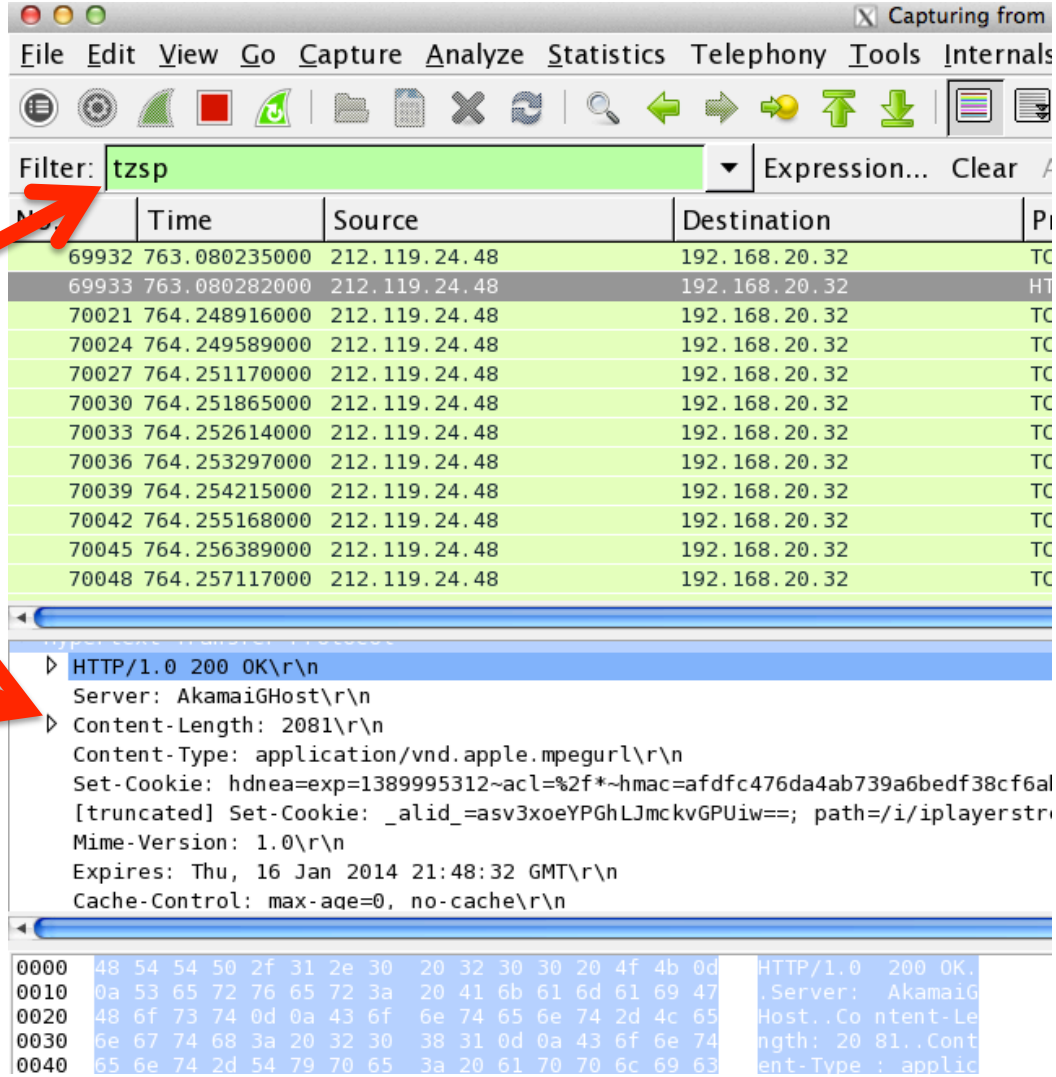
- Solutions:
  - RouterOS ‘Tools Packet Sniffer’
  - RouterOS Mangle rule action ‘sniff TZSP’
  - WireShark



Remote Packet Capture

Filter on protocol ‘tzsp’

Examine data for potential L7 rule



Layer 7 Regex

- () - groups logical ‘matches’ together
- \ - is used to ‘escape’ special characters such as ?*+|^$
- [xyz], [0-9] [a-z] - match **any** of the enclosed characters once
- ^ - string required to match, occurs at start of packet
- $ - string required to match, occurs at end of packet
- . (fullstop) – match one, but **any** single character
- ? - Match zero or 1 positions of the preceding string

Layer 7 Regex

- * - Match zero or more positions of the preceding string
- + - Match 1 or more positions of the preceding string
- | - (pipe) denotes ‘or’, match either the left or the right part
- [\x09-\x0d -~] – match on all printable ASCII characters & space
- [\x09-\x0d] – match a TAB, LF, Vert Tab, FormFeed, CR or space
- [!-~] – match non-whitespace printable characters

Example Layer 7 Regex

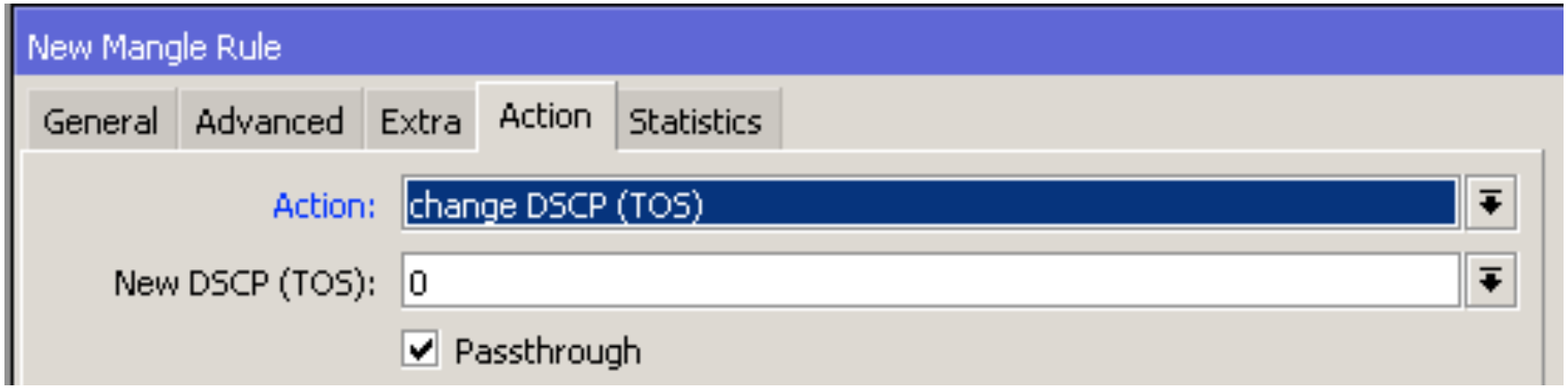
- Testing for HTTP:

```
http/(0\.9|1\.0|1\.1) ([1-5][0-9][0-9]|post) [\x09-\x0d -
~]* http/[01]\.[019]
```

- Look for a match of ‘http/0.9 ’ or ‘http/1.0 ’ or ‘http/1.1 ’ (note the trailing space)
  - *Also, RouterOS L7 is **not** case sensitive!*
- Then, match a 3 digit number between 100-599 or the word ‘post’
  - (the ‘|’ splits the regex in two ‘or’ halves)
- Followed by any amount of characters, then ‘ http/0.9’, ‘ http/1.0’ or ‘ http/1.1’ (note the space again)

Reduce Duplicating Effort

- Mark Traffic at the edge of network (‘change DSCP’)
- Apply QoS control **throughout** network (‘set priority’)
- Packet marks do not survive beyond router
- So - Use the DSCP/TOS header byte! ☺
- But – what value to use?

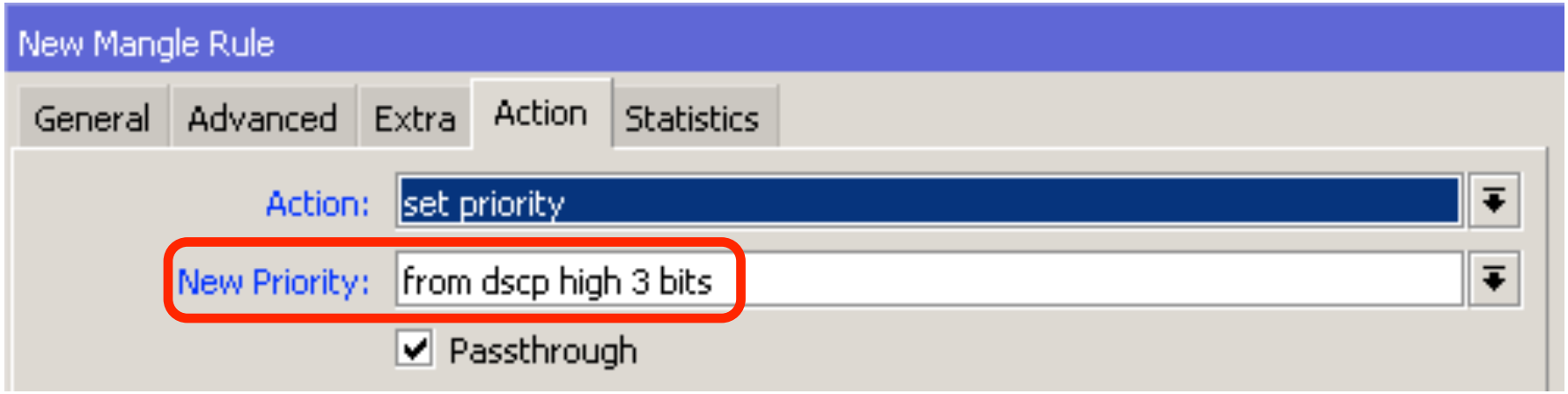


DSCP/TOS Header

- DSCP+TOS field = depending on the RFC, field is either 6, 7 or 8 bits. (See RFC791/RFC795/RFC1349), but…
- The 3 Most Significant bits → 'IP Precedence', ie **QoS**
  - 111 - Network Control
  - 110 - Internetwork Control
  - 101 - Critic/ECP
  - 100 - Flash Override
  - 011 - Flash
  - 010 - Immediate
  - 001 - Priority
  - 000 - Routine

Once again - Reduce Effort!

- Let MikroTik to make it easy for us ☺
- Packets marked using the 3 bit ‘IP Precedence’ of the DSCP value allow all Internal Routers and wireless links to use this:



NV2 & DSCP/Priority Mapping

3 MSBs of DSCP Value	DSCP Value	802.1D Priority #	NV2 Priority Queues		
			8 Qs	4Qs	2Qs
111000-**111**FFF	56 - 63	7	7	3	1
110000-**110**FFF	48 - 55	6	6	3	1
101000-**101**FFF	40 - 47	5	5	2	1
100000-**100**FFF	32 - 39	4	4	2	1
011000-**011**FFF	24 - 31	3	3	1	0
010000-**010**FFF	16 - 23	2	2	1	0
001000-**001**FFF	8 - 15	1	1	0	0
000000-**000**FFF	0 - 7	0 (default)	0	0	0

High priority

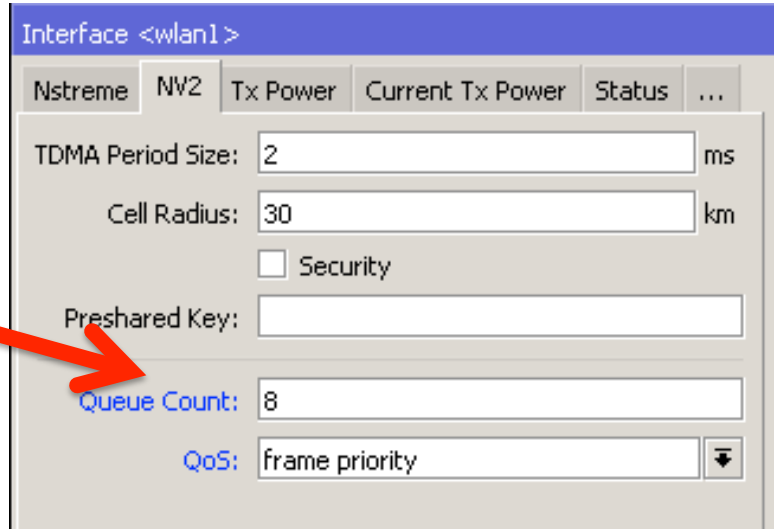
Low Priority

Priorities on NV2

- MikroTik NV2 wireless protocol has QoS already fully integrated ☺ (IEEE 802.1D-2004)
- See presentation from Lutz Kleeman
- http://mum.mikrotik.com/presentations/US13/lutz.pdf

- So, use it!

- (Set on AP)



MPLS QoS

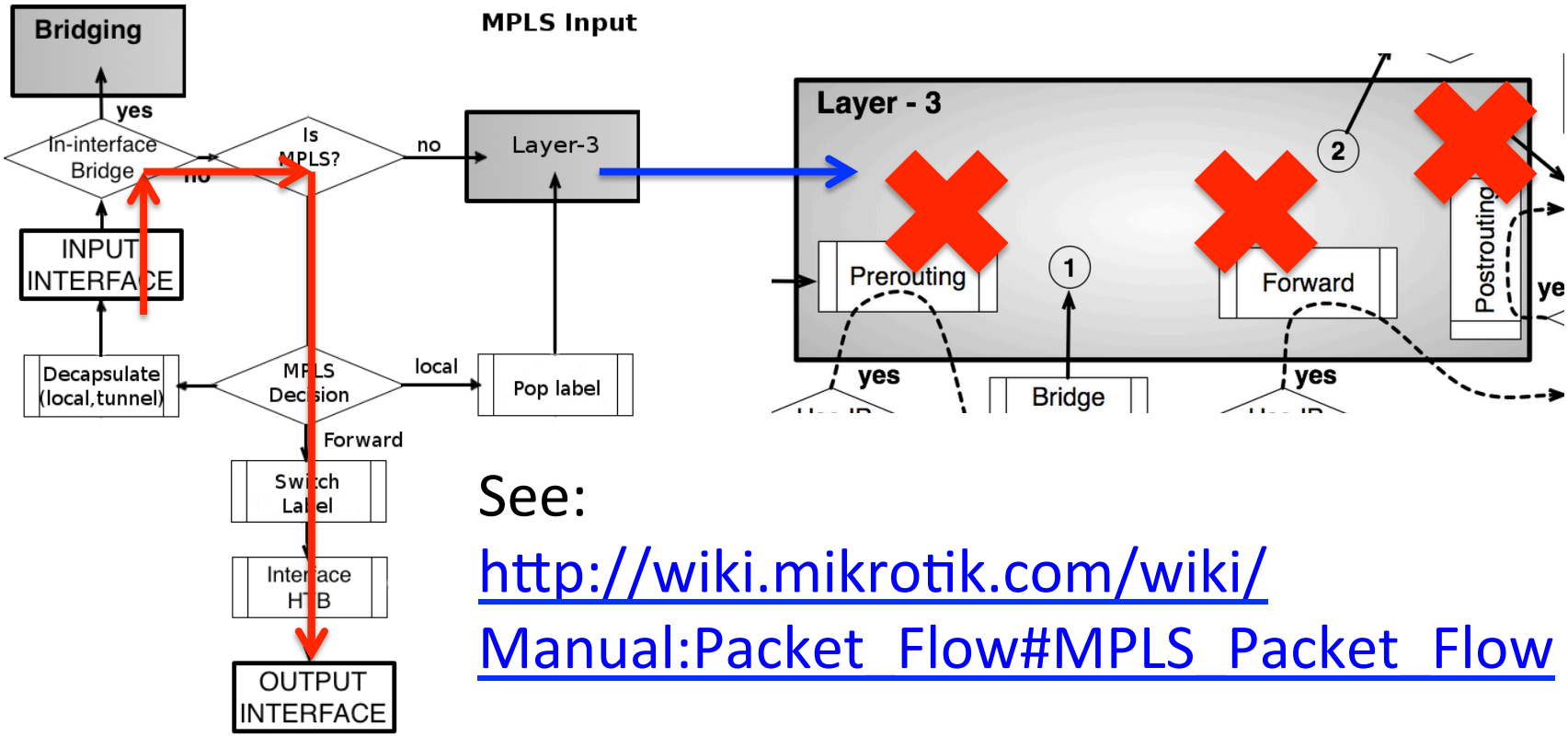
- EXP – “Experimental bits”
- 3 bits in length - commonly used for QoS
- Set desired priority on Edge (‘Ingress Router’)

MPLS Header			
Label Value 20 bits	**Exp** **3 bits**	Stacking Bit 1 bit	TTL 8 bits

- MPLS automatically sets priority of packet from the ‘EXP’ bit value on all MPLS internal routers, therefore NV2 wireless will know priority to set for each packet ☺
- http://wiki.mikrotik.com/wiki/Manual:MPLS/EXP_bit_behaviour

MPLS QoS

- Once packet is ‘inside’ MPLS switching system, it bypasses prerouting, forward and postrouting chains.
- Therefore ‘mangling’ is limited.

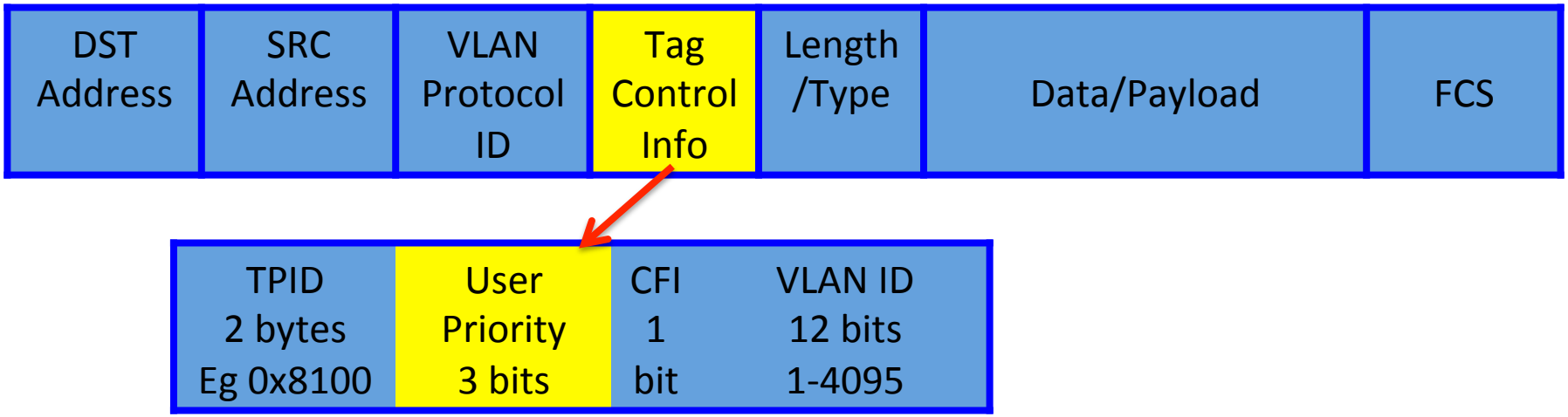


See:

http://wiki.mikrotik.com/wiki/Manual:Packet_Flow#MPLS_Packet_Flow

VLAN QoS

- VLANs also have a field to mark QoS.
- VLAN Tag added to packets contains a 3 bit ‘User Priority’ field or ‘vlan-priority’ field



http://wiki.mikrotik.com/wiki/Manual:Interface/Bridge

Setting Priorities

- E.g. set priority of packets from DSCP MSB 3 bits…

```
/ip firewall mangle
add action=set-priority chain=forward \
new-priority=from-dscp-high-3-bits
```

- MPLS ‘EXP field’ in label will then be automatically set to correct priority value (0-7) when MPLS label is ‘popped’ / attached in the ingress router ☺
- VLAN ‘vlan-priority’ (user priority) field will be automatically set to the calculated priority value
  - *(VLAN Priority on **bridge filter** rules are unable to use DSCP MSBs – only ‘from ingress’ or set ‘priority=0-7’ is available!)*.

Queue Types – In Detail

- RouterOS has 4 queue types:
  - FIFO – Simple First In First Out (Bytes or Packets)
  - RED – Random Early Detect (or Drop)
  - SFQ – Stochastic Fairness Queuing
  - PCQ – Per Connection Queuing (MikroTik Proprietary)

- Also, each queue type has 2 major characteristics:
  - **Shaper** (where packets are dropped to reduce traffic)
  - **Scheduler** (where packets are temporarily delayed)
    - *(More on shaper/scheduler characteristics later)*

FIFO – First In First Out

- Behaviour**:** First packet in is outputted, subsequent packets wait in buffer until previous packet has left buffer. Once buffer is full, **all** new incoming packets are dropped.

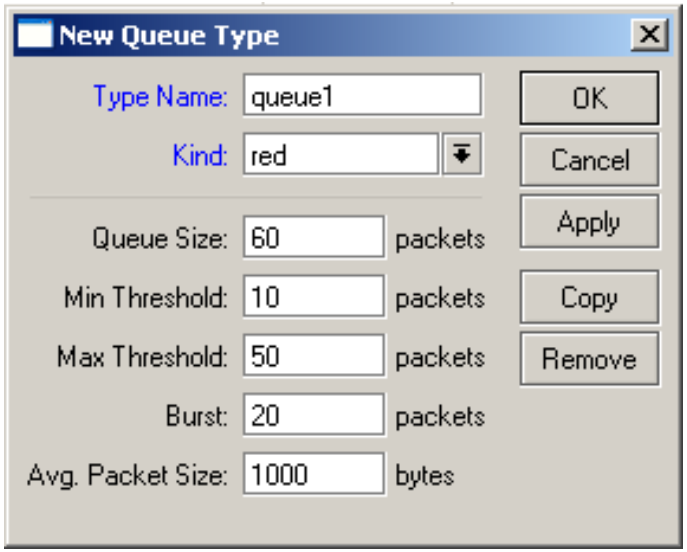
- Two types of FIFO
  - BFIFO – queue size is a physical buffer size (kb)
  - PFIFO – queue size is a physical number of packets (e.g. default, default-small, ethernet-default – used in PPP, DHCP, Hotspot etc)

- **NOT** recommended for **very** congested links as once queue is full, **ALL** traffic is dropped

Random Early Detect

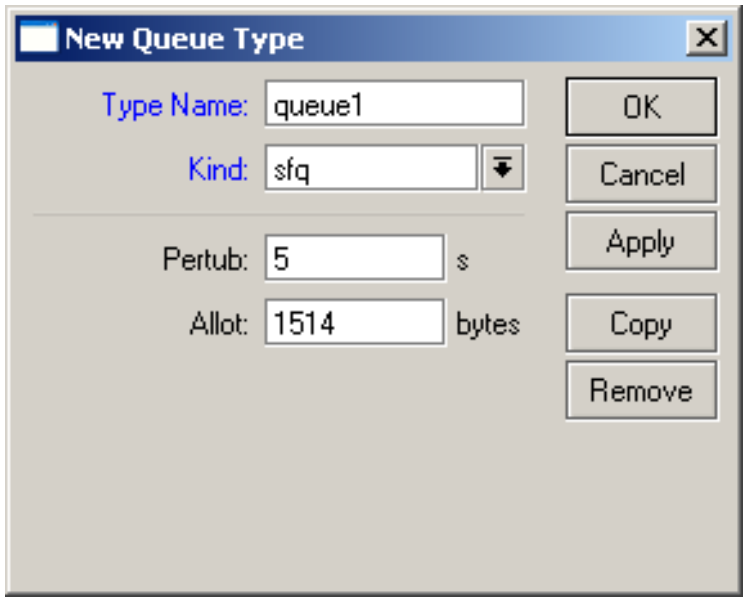
- Behaviour: Similar to FIFO, except randomly starts dropping packets **before** queue is full.
- Queue ‘dropping’ properties mainly controlled by max/min threshold values

- Recommended for congested links as once queue is **nearly** full, traffic is more fairly dropped.
- Works in a **softer** way than FIFO queues.



Stochastic Fairness Queuing

- Behaviour: Similar in some ways to PCQ except with very limited control over how sub-queues are created. Divides the traffic up into 1024 sub-queues based on SRC/DST IP and Port. Total size of SFQ Queue is only 128 packets.
- Round Robin algorithm then distributes 'Allot' amount of traffic into each of the multiple sub-queues
- Hash value recalculated every 'Pertub' time.
- Works in a **fair/equal** way on congested links.

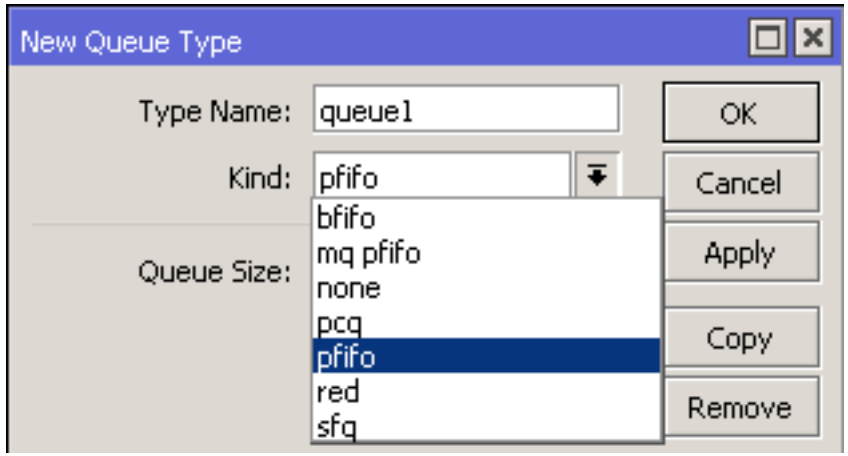


Per Connection Queuing

- PCQ is very well documented by Janis Megis and Velans Riyadi. Not going over same material! See:

http://mum.mikrotik.com/presentations/US09/megis_qos.pdf
http://mum.mikrotik.com/presentations/HR13/valens.pdf
http://mum.mikrotik.com/presentations/ID13/valens.pdf

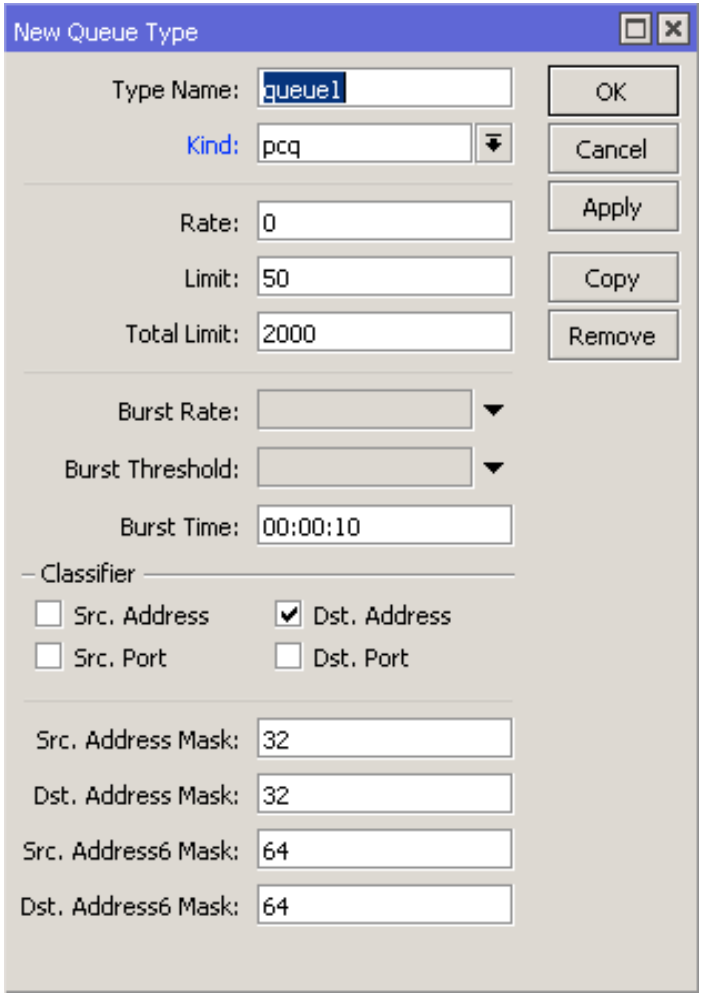
- But is PCQ best? (Answer: It depends on how it is configured!)



Per Connection Queuing

- Behaviour: Similar in some ways to SFQ except with much more detailed control over how sub-queues are created. Divides the traffic up into multiple sub-queues based on single or multiple SRC/DST IPs and/or Ports.

- Works in a **fair/equal** way on congested links.
- **Also permits sub-queue speed limits**.

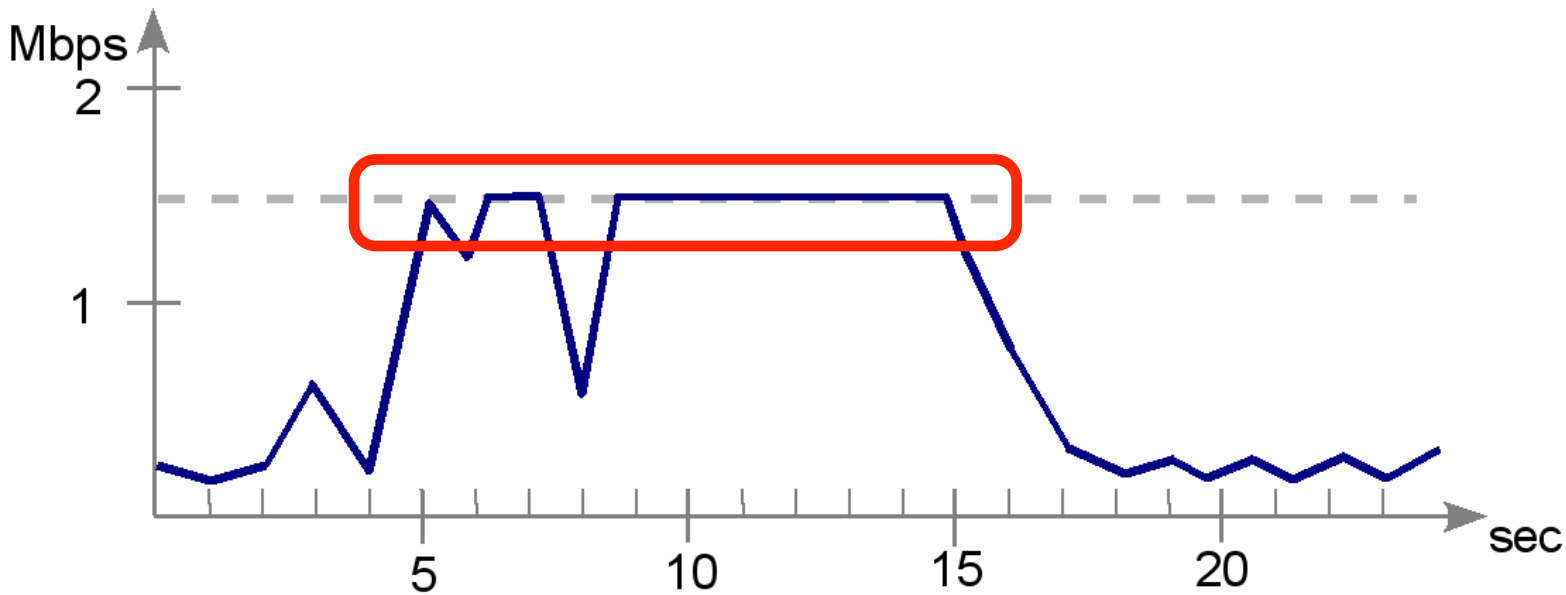


Choosing Queue size?

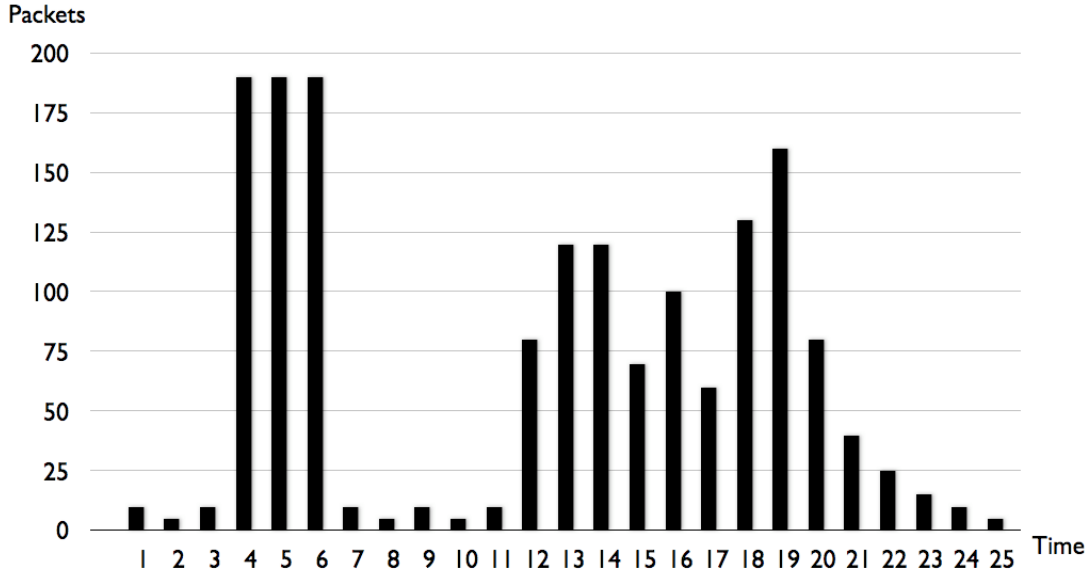
- All queues very nearly work the same **until there is congestion**.
- If too small size, packets are very quickly dropped and high packet drops – however, low latency
- If too large size, packets are delayed leaving router and latency will be poor however there is less packet loss (but not good for TCP as it reduces speed to compensate)
- **Choice of queue size is therefore important and usually a compromise**!
- ‘default-small’ is 10 (= low latency / higher packet loss)
- ‘default’ is 50 (higher latency but lower packet loss)

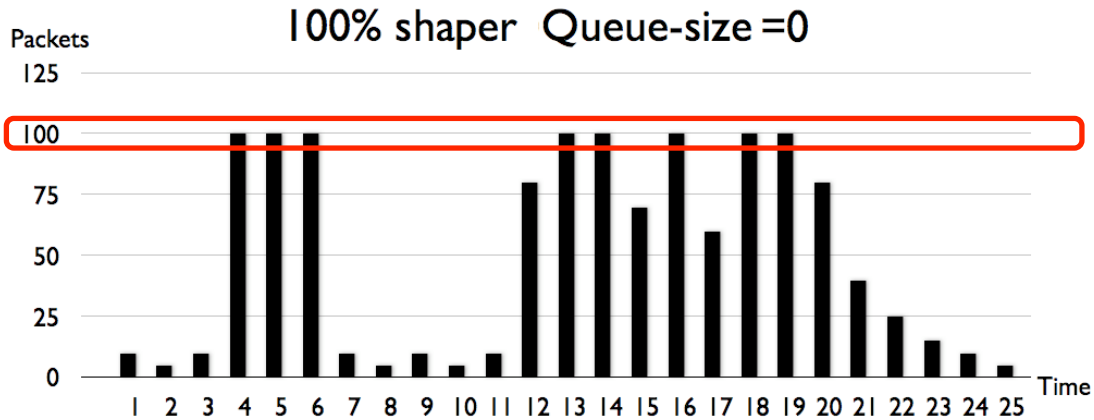
Queuing – 100% Shaper

- **100% Shaper**
  - all new packets are dropped once ‘max-limit’ is reached.
  - Size of queue is **zero.** It cannot hold **any** packets without dropping them, however **latency is low**.



Queuing – 100% Shaper

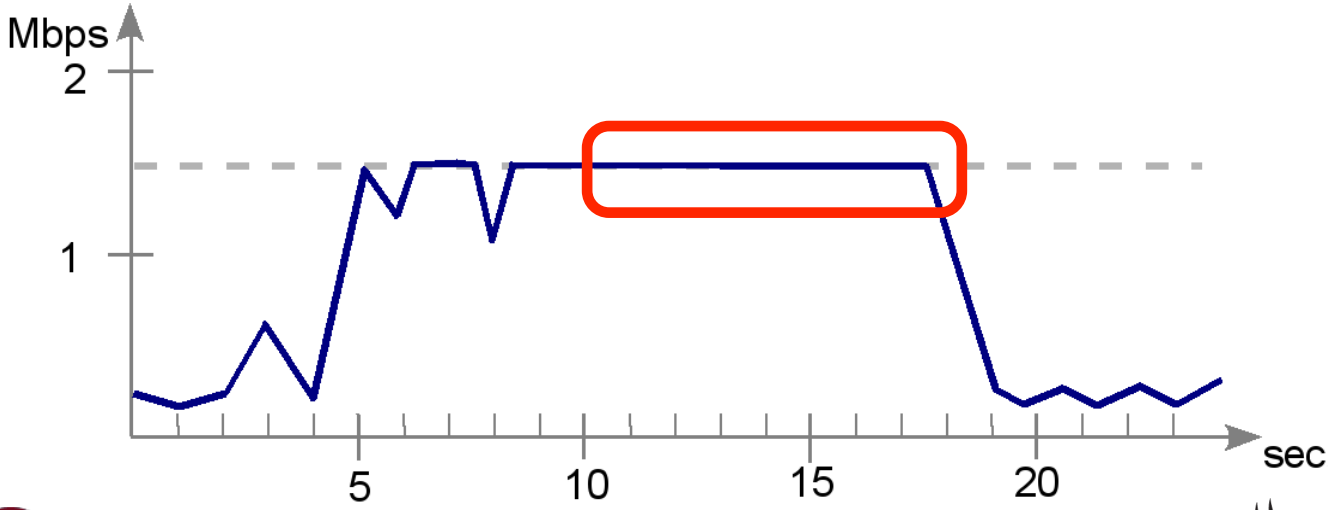




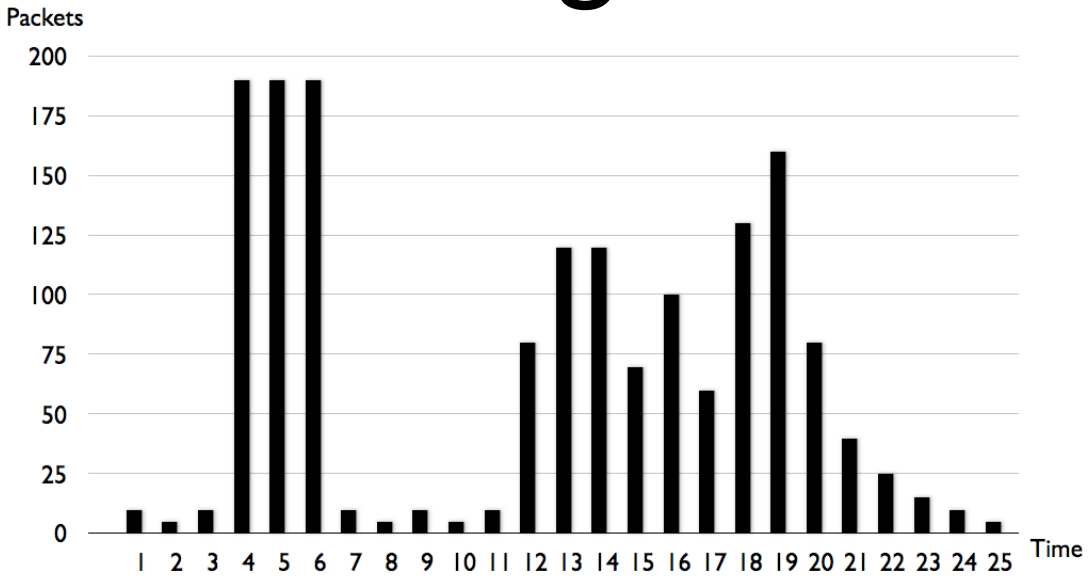
- Assume max-limit is '100'
- 100% shaper has no queue size
- Therefore packets are dropped when it reaches 100.
- In this example about 22% is dropped
- Latency is low ☺

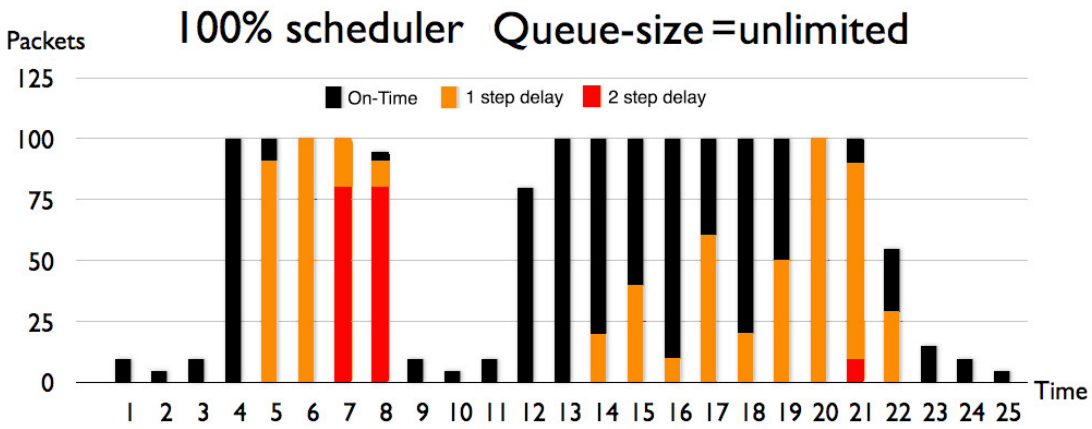
Queuing – 100% Scheduler

- **100% Scheduler**
  - Packets queued when ‘max-limit’ reached.
  - Chose size of queue to hold correct number of packets, to delay their departure from the interface long enough but latency is higher. ☹
  - When queue is full, packets are dropped.



Queuing - 100% Scheduler



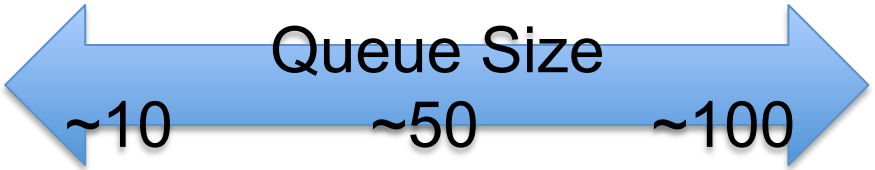


- Assume max-limit is '100'
- queue size is unlimited
- Therefore **no** packets are dropped when it reaches 100.
- In this example 39% are delayed once, 11% delayed twice
- Latency is high ☹

BFIFO/PFIFO Queue Size?

- Shaping? Or Scheduling? Which is best?
- Both Shaping AND Scheduling!
- Shaping/scheduling only happens when buffer is full
- The larger the queue size, the more delay that can occur
- My advice for size? Seriously – it's up to you! ☺ YMWV (Your Mileage **Will** Vary!) But with high throughput (100MBps+) 10-50 is too small! Try values ≥100

Low Latency,
Higher packet loss,
Good for VOIP



Shaping

Scheduling

Higher Latency,
Lower packet loss,
Good for video streaming

PCQ Queue Size?

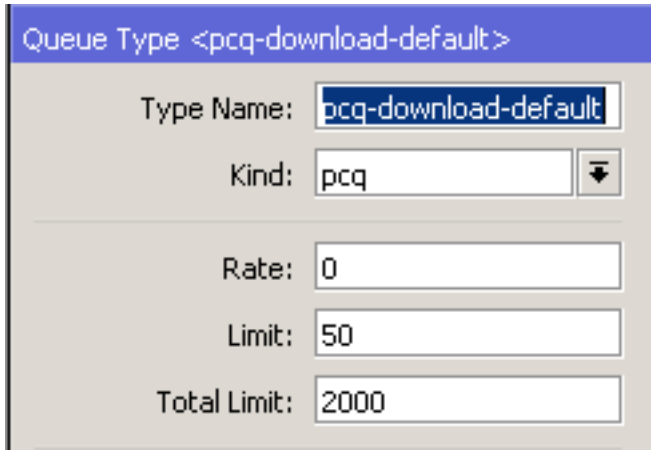
- MikroTik recommend PCQ sub-queue to be approx 10-20 packets per client connection (ie per sub-queue). This offers a good balance between shaping and scheduling of packets.

- E.g. A network has 350 clients, although it is observed that there is only a maximum of 175 client connections of the type captured by a particular PCQ Queue at any one time. Therefore:
- One could set ‘Total Limit’ to 7000 (350 clients x 20 packets per client) and ‘Limit’ to 40 (7000/40=175)

PCQ Queue size?

- So – how to know what packet sizes to allocate?
- Time for some Math's as it depends on number of client connections:

- Limit ("pcq-limit") = max number of packets in **one** sub-queue
- Total Limit ("pcq-total-limit") = max packets in **all** sub-queues
- Max quantity of sub-queues = (Total Limit ÷ Limit)
- ∴ Default is 2000 ÷ 50 = **only 40 active connections maximum!** ☹

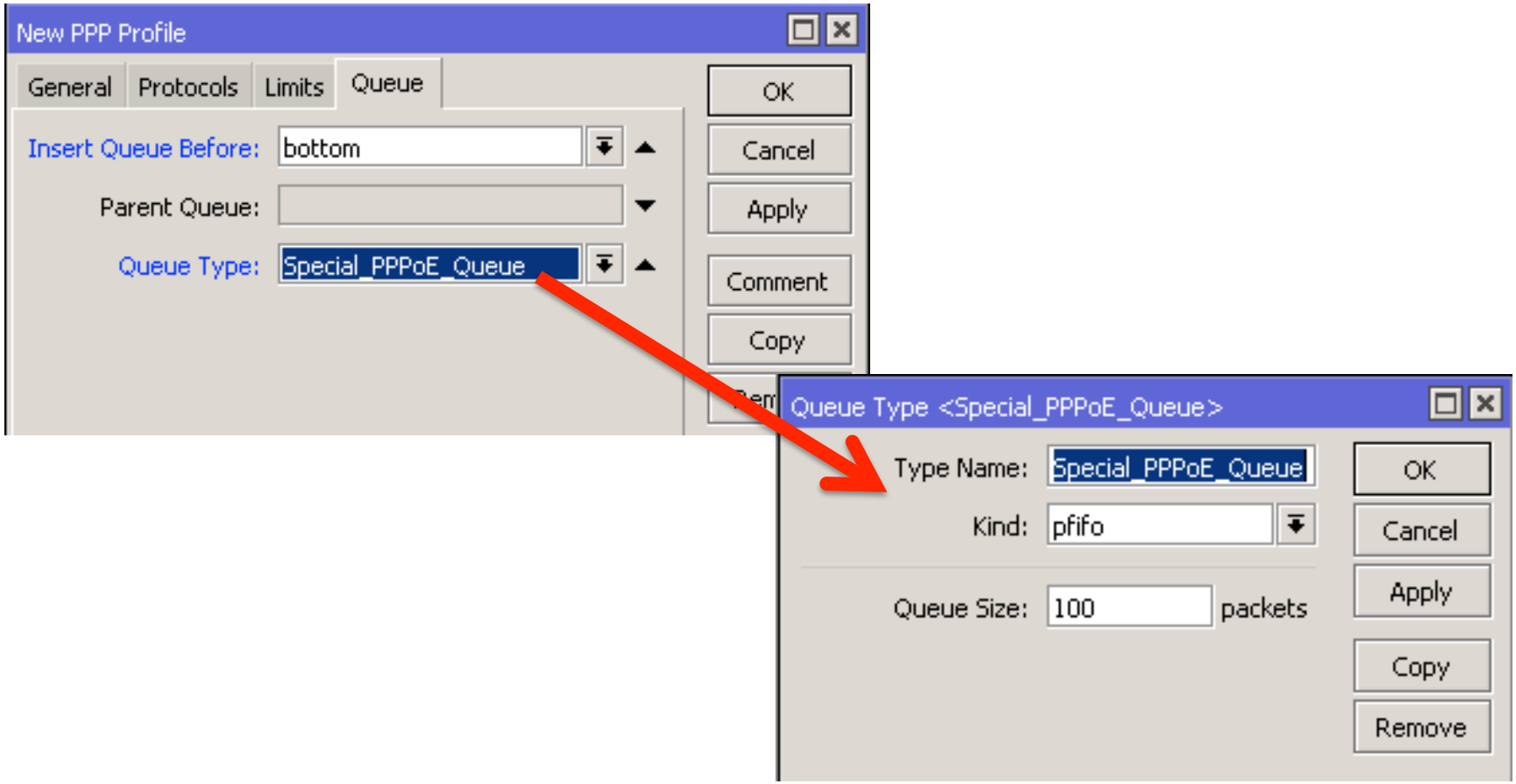


Default Queue Values

- Is MikroTik **always** correct with their default values?
- Do MikroTik really know **your** network? No.
- Do MikroTik fully understand **your** traffic? No.
- So how can the defaults **ALWAYS** be right for you?
- Well – obviously they **can't**… So…
- Change them! ☺
- Also, in v6 **you** can also control the default queues! ☺

Dynamic queue types

- Chose **your** default queue…



Notes / ToDo

- Packets that carry priority, mangle and “action=from ingress” – double sanity check into difference in priorities between ordering of CoS values 0,1,2 and actual priorities 0,1,2 in RouterOS!

Contact Details

Ron Touw – MikroTik Trainer & Consultant
LinITX
Tel: +44 1449 724250
VOIP: sip:contact@linitx.com
IRC: irc.z.je #routerboard
Email: shop@linitx.com
WWW: http://linitx.com

Materials Copyright

- Ron Touw of LinITX hereby acknowledges that some of the material contained within this presentation contains copyrighted images and descriptive text belonging to Mikrotīkls SIA.