MikroTik RouterOS Workshop
Load Balancing
Best Practice

Warsaw
MUM Europe 2012
About Me

Jānis Meģis, MikroTik

Jānis (Tehnical, Trainer, NOT Sales)

- Support & Training Engineer for almost 8 years
- Specialization: QoS, PPP, Firewall, Routing
- Teaching MikroTik RouterOS classes since 2005
Load Balancing

Load Balancing is a technique to distribute the workload across two or more network links in order to maximize throughput, minimise response time, and avoid overload.

Using multiple network links with load balancing, instead of single network links, may increase reliability through redundancy.
Types of Load Balancing

- Sub-Packet Load Balancing (MLPPP)
- Per Packet Load Balancing (Bonding)
- Per Connection Load Balancing (nth)
- Per address-pair Load Balancing (ECMP, PCC, Bonding)
- Custom Load Balancing (Policy Routing)
- Bandwidth based Load Balancing (MPLS RSVP-TE Tunnels)
Multi-Link PPP

- PPP Multi-link Protocol allows to divide packet equally and send each part into multiple channels

- MLPPP can be created:
  - over single physical link – where multiple channels run on the same link (anti-fragmentation)
  - over multiple physical links - where multiple channels run on the multiple link (load balancing)

- MLPPP must be supported by both ends

- (MLPPP is legacy stuff from modem era)
MLPPP configuration

- Server must have MLPPP support
- All lines must have same user name and password
- RouterOS has only the MLPPP client implementation
Bonding

Bonding is a technology that allows you to aggregate multiple Ethernet-like interfaces into a single virtual link, thus getting higher data rates and providing fail-over.

Bonding (load balancing) modes:
- 802.3ad
- Balance-rr
- Balance-xor
- Balance-tlb
- Balance-alb
802.3ad

- 802.3ad mode is an IEEE standard also called LACP (Link Aggregation Control Protocol).
Balance-rr and balance-xor

Balance-rr mode uses Round Robin algorithm - packets are transmitted in sequential order from the first available slave to the last.

When utilizing multiple sending and multiple receiving links, packets often are received out of order (problem for TCP)

Balance-xor balances outgoing traffic across the active ports based on a hash from specific protocol header fields and accepts incoming traffic from any active port
Balance-tlb

- The outgoing traffic is distributed according to the current load
- Incoming traffic is not balanced
- This mode is address-pair load balancing
- No additional configuration is required for the switch
Balance-alb

- In short alb = tlb + receive load balancing
- This mode requires a device driver capability to change the MAC address
ECMP Routes

- ECMP (Equal Cost Multi Path) routes have more than one gateway to the same remote network.

- Gateways will be used in Round Robin per SRC/DST address combination.

- Same gateway can be written several times!!
“Check-gateway” Option

- You can set the router to check gateway reachability using ICMP (ping) or ARP protocols.
- If the gateway is unreachable in a simple route – the route will become inactive.
- If one gateway is unreachable in an ECMP route, only the reachable gateways will be used in the Round Robin algorithm.
- If Check-gateway option is enabled on one route it will affect all routes with that gateway.
Interface ECMP Routing

In case you have more than one PPP connection from the same server, but MLPPP is impossible (different user names, server support missing) it is possible to use Interface routing.

Simple IP address routing is impossible for all PPP connections that have the same gateway IP address.

To enable interface routing just specify all PPP interfaces as route gateway-interfaces.

Works only on PPP interfaces.
ECMP and Masquerade

- As forwarding database is rebuilt every 10min in Linux Kernel, there is a chance that connection will jump to the other gateway
- In the case of masquerading this jump results in a change of source address and in eventual disconnect

More info at:
- http://marc.info/?m=105217616607144
- http://lkml.indiana.edu/hypermail/linux/net/0305.2/index.html#19
Configuration Setup

GW: 11.11.11.254

GW: 12.12.12.254

ISP1 /24

ISP2 /24

Internet

Masquerade

192.168.88.0/24
Basic Configuration

[admin@MikroTik] > /interface set 1 name=to_ISP1
[admin@MikroTik] > /interface set 2 name=to_ISP2
[admin@MikroTik] > /interface set 3 name=Local

[admin@MikroTik] /ip address> add address=192.168.88.254/24 interface=Local
[admin@MikroTik] /ip address> add address=11.11.11.1/24 interface=to_ISP1
[admin@MikroTik] /ip address> add address=12.12.12.1/24 interface=to_ISP2

[admin@MikroTik] /ip route> add gateway=11.11.11.254 distance=2
[admin@MikroTik] /ip route> add gateway=12.12.12.254 distance=3

[admin@MikroTik] /ip firewall nat> add chain=srcnat out-interface=to_ISP1 action=masquerade
[admin@MikroTik] /ip firewall nat> add chain=srcnat out-interface=to_ISP2 action=masquerade
Policy Routing

Policy routing is a method that allows you to create separate routing policies for different traffic by creating custom routing tables.

In RouterOS these routing tables are created:
- For every table specified in /ip route rule
- For every routing-mark in mangle facility

Marked traffic is automatically assigned to the proper routing table (no need for lookup rules)
Routing-mark

- RouterOS attribute assigned to each packet
- Routing-mark can be changed in firewall mangle facility just before any routing decision:
  - chain Prerouting – for all incoming traffic
  - chain Output – for outgoing traffic from router
- Every new routing mark has its own routing table with the same name
- By default all packets have the “main” routing mark
Traffic to Connected Networks

As connected routes are available only in “main” routing table, it is necessary that traffic to connected networks stay in “main” routing table.

This will also allow proper communication between locally and remotely connected clients.

/ip firewall mangle> add chain=prerouting src-address=192.168.88.0/24 dst-address=11.11.11.0/24 action=accept
/ip firewall mangle> add chain=prerouting src-address=192.168.88.0/24 dst-address=12.12.12.0/24 action=accept
/ip firewall mangle> add chain=prerouting src-address=192.168.88.0/24 dst-address=192.168.88.0/24 action=accept
Remote Connections

In the case when a connection is initiated from a public interface it is necessary to ensure that these connections will be replied via the same interface (from the same public IP)

First we need to capture these connections (you can either use default connection mark “no-mark” or connection state “new” here)

```
/ip firewall mangle> add chain=prerouting connection-mark=no-mark in-interface=to ISP1
    action=mark-connection new-connection-mark=ISP1_conn

/ip firewall mangle> add chain=prerouting connection-mark=no-mark in-interface=to ISP2
    action=mark-connection new-connection-mark=ISP2_conn
```
Custom Policy Routing

Now we need to create a default route for every routing table (or else it will be resolved by main routing table)

/ip route> add gateway=11.11.11.254 routing-mark=ISP1_traffic
/ip route> add gateway=12.12.12.254 routing-mark=ISP2_traffic

Let's create a jump rule to your custom policy routing here

/ip firewall mangle> add chain=prerouting in-interface=Local connection-mark=no-mark action=jump jump-target=policy_routing
Mark Routing

- Mark routing rules in mangle chain “output” will ensure that router itself is reachable via both public IP addresses.
- Mark routing rules in mangle chain “prerouting” will ensure your desired load balancing.

```
/ip firewall mangle> add chain=prerouting connection-mark=ISP1_conn src-address=192.168.88.0/24 action=mark-routing new-routing-mark=ISP1_traffic
/ip firewall mangle> add chain=prerouting connection-mark=ISP2_conn src-address=192.168.88.0/24 action=mark-routing new-routing-mark=ISP2_traffic
/ip firewall mangle> add chain=output connection-mark=ISP1_conn action=mark-routing new-routing-mark=ISP1_traffic
/ip firewall mangle> add chain=output connection-mark=ISP2_conn action=mark-routing new-routing-mark=ISP2_traffic
```
Mangle configuration

<table>
<thead>
<tr>
<th>#</th>
<th>Action</th>
<th>Chain</th>
<th>Src. Address</th>
<th>Dst. Address</th>
<th>In. Interface</th>
<th>Connection Mark</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>accept</td>
<td>prerouting</td>
<td>192.168.88.0/24</td>
<td>11.11.11.0/24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>accept</td>
<td>prerouting</td>
<td>192.168.88.0/24</td>
<td>12.12.12.0/24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>accept</td>
<td>prerouting</td>
<td>192.168.88.0/24</td>
<td>192.168.88.0/24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>mark connection</td>
<td>prerouting</td>
<td></td>
<td>to_ISP1</td>
<td>no-mark</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>mark connection</td>
<td>prerouting</td>
<td></td>
<td>to_ISP2</td>
<td>no-mark</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>jump</td>
<td>prerouting</td>
<td></td>
<td>Local</td>
<td>no-mark</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>mark routing</td>
<td>prerouting</td>
<td>192.168.88.0/24</td>
<td>ISP1_conn</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>mark routing</td>
<td>prerouting</td>
<td>192.168.88.0/24</td>
<td>ISP2_conn</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>mark routing</td>
<td>output</td>
<td></td>
<td>ISP1_conn</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>mark routing</td>
<td>output</td>
<td></td>
<td>ISP2_conn</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Custom Policy Routing

There is no best way that we can suggest for load balancing, you can either:

- Balance based on client IP address (address list)
- Balance based on traffic type (p2p, layer-7, protocol, port)
- Use automatic balancing (PCC)

We do not suggest to use “nth” for policy routing of typical user traffic.
Per-address-pair Load Balancing

In many situations communication between two hosts consist of more than one simultaneous connection.

If those connections are taking different routing paths they might have different latency, drop rate, fragmentation or source address (NAT) – this way making multi-connection communications impossible.

That is why instead of per-connection load balancing we should think about per-address-pair load balancing.
Per Connection Classifier

PCC is a firewall matcher that allows you to divide traffic into equal streams with ability to keep packets with specific set of options in one particular stream.

You can specify set of options from src-address, src-port, dst-address, dst-port.

PCC Configuration

We just need to add 2 rules to our “policy_routing” chain to ensure automatic per-address-pair load balancing

```
ip firewall mangle> add chain=policy_routing dst-address-type=!local per-connection-classifier=both-addresses:2/0 action=mark-connection new-connection-mark=ISP1_conn

/ip firewall mangle> add chain=policy_routing dst-address-type=!local per-connection-classifier=both-addresses:2/1 action=mark-connection new-connection-mark=ISP2_conn
```
Usual Problems

- Be careful about using “no-mark” connection mark if you have other mangle configuration in a different chain.

- ISP specified DNS servers might block requests from non-ISP public IPs, so we suggest you use public (ISP independent) DNS servers.

- If you would like to ensure fail-over – enable “check-gateway” option in all default routes.
What about bandwidth based Load-Balancing?
Traffic Engineering

- TE is one of MPLS features that allow to establish unidirectional label switching paths.
- TE is based on RSVP (Resource ReSerVation Protocol) + RFC 3209 that adds support for explicit route and label exchange.
- TE tunnels are similar to LDP, but with additional features:
  - Usage of either full or partial explicit routes.
  - Constraint (such as bandwidth and link properties) based LSP (Label Switched Path) establishment.
How Does Constraints Work?

- Constraints are set by user and does not necessarily reflect actual bandwidth

- Constraints can be set for:
  - bandwidth of link participating in a RSVP TE network
  - bandwidth reserved for tunnel

- So, at any moment in time, the bandwidth available on TE link is bandwidth configured for link minus sum of all reservations made on the link (not physically available bandwidth)
TE Tunnel Establishment

TE tunnels can be established:

- along the current routing path (no additional configuration required)
- along a statically configured explicit path (it is necessary to manually input path)
- CSPF (Constrained Shortest Path First) - This option needs assistance from IGP routing protocol (such as OSPF) to distribute bandwidth information throughout the network.
Network Layout

Each router is connected to a neighbouring router using /30 network and each of them have unique Loopback address form 10.255.0.x network. Loopback addresses will be used as tunnel source and destination.

```
/system identity set name=Rx
/interface bridge add name=Loopback
/ip add add address=10.255.0.x/24 interface=Loopback
/ip add add address=192.168.33.x/30 interface=ether1
/ip add add address=192.168.33.y/30 interface=ether2
```
Network Layout

LAN1 192.168.10.0/24

R1: 192.168.33.1/30
10.255.0.1/32

R2: 192.168.33.2/30
10.255.0.2/32
192.168.33.5/30

R3: 192.168.33.6/30
10.255.0.3/32

R4: 192.168.33.10/30
10.255.0.4/32

LAN2 192.168.20.0/24
Loopback and CSPF

Loopback addresses need to be reachable from whole network – we will use OSPF to distribute that information

Also OSPF can help us to distribute TE reservations for CSPF

```
/routing ospf instance
set default mpls-te-area=backbone
mpls-te-router-id=Loopback router-id=10.255.0.x

/routing ospf network
add network=192.168.33.0/24 area=backbone
add network=10.255.0.x/32 area=backbone
```
Resource Reservation

- Lets set up TE resource for every interface on which we might want to run TE tunnel.
- Configuration on all the routers are the same:

```
/mpls traffic-eng interface
  add interface=ether1 bandwidth=10Mbps
  add interface=ether2 bandwidth=10Mbps
```

- Note that at this point this does not represent how much bandwidth will actually flow through the interface.
First Task
TE tunnel setup

We will use static path configuration as primary, and dynamic (CSPF) as secondary path if primary fails

```
/mpls traffic-eng tunnel-path
  add name=dyn use-cspf=yes
  add name=tun-first-link use-cspf=no
    hops=192.168.33.2:strict,192.168.33.5:strict,192.168.33.6:strict

/interface traffic-eng
  add bandwidth=5Mbps name=TE-to-R3 to-address=10.255.0.3 primary-path=tun-first-link
  secondary-paths=dyn record-route=yes from-address=10.255.0.1

/mpls traffic-eng tunnel-path
  add name=dyn use-cspf=yes
  add name=tun-second-link use-cspf=no

/interface traffic-eng
  add bandwidth=5Mbps name=TE-to-R1 to-address=10.255.0.1 primary-path=tun-second-link
  secondary-paths=dyn record-route=yes from-address=10.255.0.3
```
TE Tunnel Monitoring

```
[admin@R1] /mpls traffic-eng> path-state print
Flags: L - locally-originated, E - egress, F - forwarding, P - sending-path, R - sending-resv
#      SRC           DST          BANDWIDTH   OUT.. OUT-NEXT-HOP
0 LFP  10.255.0.1:1  10.255.0.3:15  5.0Mbps   eth.. 192.168.33.2
1 E R  10.255.0.3:1  10.255.0.1:8   5.0Mbps

[admin@R1] /mpls traffic-eng> resv-state print
Flags: E - egress, A - active, N - non-output, S - shared
#      SRC           DST          BANDWIDTH LABEL  INT...
0 AS  10.255.0.1:1   10.255.0.3:15  5.0Mbps   41   ether1

[admin@R1] /mpls traffic-eng>
[admin@R1] /mpls traffic-eng> interface print
Flags: X - disabled, I - invalid
#      INTERFACE     BANDWIDTH  TE-METRIC  REMAINING-BW
0 ether1          10Mbps
1 ether2          10Mbps     10.0Mbp
```
TE Tunnel Monitoring

If multiple tunnels are created and all the bandwidth on that particular interface is used, then the tunnel will try to look for different path.

```
[admin@R1] /interface traffic-eng> monitor 0
  tunnel-id: 14
  primary-path-state: established
  primary-path: tun-first-link
  secondary-path-state: not-necessary
    active-path: tun-first-link
  active-lspid: 1
  active-label: 39
  reserved-bandwidth: 5.0Mbps
```
Route traffic over TE

To route LAN traffic over a TE tunnel we will assign address 10.99.99.1/30 and 10.99.99.2/30 to each tunnel end.
Automatic Failover

By default the tunnel will try to switch back to the primary path every minute. This setting can be changed with `primary-retry-interval` parameter.
Additional Tunnels
Additional Tunnels

```
/mpls traffic-eng tunnel-path
add name=tun-second-link use-cspf=no \n  hops=192.168.33.13:strict,192.168.33.10:strict,192.168.33.9:strict

/interface traffic-eng
add name=TE-to-R3-VOIP to-address=10.255.0.3 bandwidth=5Mbps record-route=yes \n  primary-path=tun-second-link secondary-paths=dyn reoptimize-interval=5s

/ip address add address=10.100.100.1/30 interface=TE-to-R3-VOIP
/ip route add dst-address=192.168.20.250/32 gateway=10.100.100.2

/ip address add address=10.100.100.2/30 interface=TE-to-R1
```

© MikroTik 2012
Good luck!