Mangle

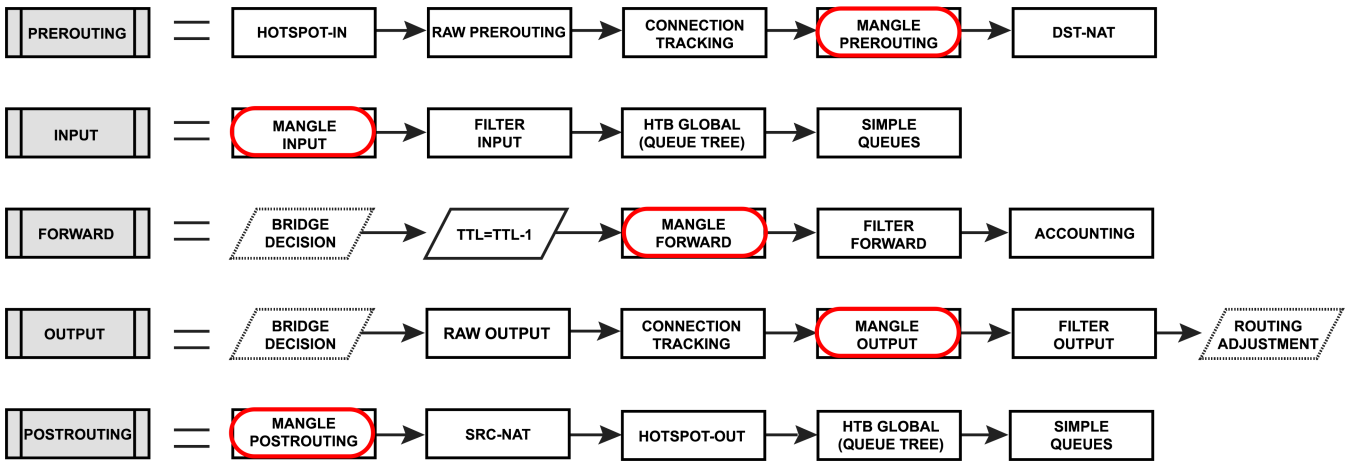
- Introduction
- Configuration example
  - Change MSS
  - Marking Connections
- Mangle Actions

Introduction

Mangle is a kind of 'marker' that marks packets for future processing with special marks. Many other facilities in RouterOS make use of these marks, e.g. queue trees, NAT, routing. They identify a packet based on its mark and process it accordingly. The mangle marks exist only within the router, they are not transmitted across the network.

Additionally, the mangle facility is used to modify some fields in the IP header, like TOS (DSCP) and TTL fields.

Firewall mangle rules consist of five predefined chains that cannot be deleted:



- The **PREROUTING** chain: Rules in this chain apply to packets as they just arrive on the network interface;
- The **INPUT** chain: Rules in this chain apply to packets just before they're given to a local process;
- The **OUTPUT** chain: The rules here apply to packets just after they've been produced by a process;
- The **FORWARD** chain: The rules here apply to any packets that are routed through the current host;
- The **POSTROUTING** chain: The rules in this chain apply to packets as they just leave the network interface;

Configuration example

Change MSS

It is a known fact that VPN links have a smaller packet size due to encapsulation overhead. A large packet with MSS that exceeds the MSS of the VPN link should be fragmented before sending it via that kind of connection. However, if the packet has a *Don't Fragment* flag set, it cannot be fragmented and should be discarded. On links that have broken path MTU discovery (PMTUD), it may lead to a number of problems, including problems with FTP and HTTP data transfer and e-mail services.

In the case of a link with broken PMTUD, a decrease of the MSS of the packets coming through the VPN link resolves the problem. The following example demonstrates how to decrease the MSS value via mangle:

```
/ip firewall mangle add out-interface=pppoe-out protocol=tcp tcp-flags=syn action=change-mss new-mss=1300
chain=forward tcp-mss=1301-65535
```

Marking Connections

Sometimes it is necessary to perform some actions on the packets belonging to specific connection (for example, to mark packets from/to specific host for queues), but inspecting each packets IP header is quite expensive task. We can use connection marks to optimize the setup a bit.

```
/ip firewall mangle
add chain=forward in-interface=local src-address=192.168.88.123 connection-state=new action=mark-connection new-
connection-mark=client_conn
add chain=forward connection-mark=client_conn action=mark-packet new-packet-mark=client_p
```

Warning: Packet marks are limited to a maximum of 4096 unique entries. Exceeding this limit will cause an error "bad new packet mark"

Mangle Actions

Table list mangle actions and associated properties. Other actions are listed here.

Property	Description
action (*action name*; Default: **accept**)	• `change-dscp` - change the Differentiated Services Code Point (DSCP) field value specified by the `new-dscp` parameter • `change-mss` - change the Maximum Segment Size field value of the packet to a value specified by `the new-mss` parameter • `change-ttl` - change the Time to Live field value of the packet to a value specified by the `new-ttl` parameter • `clear-df` - clear 'Do Not Fragment' Flag • `fasttrack-connection` - shows fasttrack counters, useful for statistics. After a packet is matched it is passed to the next rule in the list, similar as `passthrough` • `mark-connection` - place a mark specified by the new-connection-mark parameter on the entire connection that matches the rule • `mark-packet` - place a mark specified by the new-packet-mark parameter on a packet that matches the rule • `mark-routing` - place a mark specified by the new-routing-mark parameter on a packet. This kind of mark is used for policy routing purposes only. Do not apply any other routing marks besides "main" for the packets processed by FastTrack, since FastTrack can only work in the main routing table. • `route` - forces packets to a specific gateway IP by ignoring normal routing decisions (prerouting chain only) • `set-priority` - set priority specified by the new-priority parameter on the packets sent out through a link that is capable of transporting priority (VLAN or WMM-enabled wireless interface). Read more • `sniff-pc` - send a packet to a remote RouterOS CALEA server. After a packet is matched it is passed to the next rule in the list, similar as `passthrough` • `sniff-tzsp` - send a packet to a remote TZSP compatible system (such as Wireshark). Set remote target with `sniff-target` and `sniff-target-port` parameters (Wireshark recommends port 37008). After a packet is matched it is passed to the next rule in the list, similar as `passthrough` • `strip-ipv4-options` - strip IPv4 option fields from IP header, the action does not actually remove IPv4 options but rather replaces all option octets with NOP, further matcher with `ipv4-options=any` will still match the packet.
new-dscp (*integer: 0..63*; Default:)	Sets a new DSCP value for a packet
new-mss (*integer*; Default:)	Sets a new MSS for a packet. Clamp-to-pmtu feature sets (DF) bit in the IP header to dynamically discover the PMTU of a path. Host sends all datagrams on that path with the DF bit set until receives ICMP Destination Unreachable messages with a code meaning "fragmentation needed and DF set". Upon receipt of such a message, the source host reduces its assumed PMTU for the path.

new-packet-mark (*string*; Default:)	Sets a new `packet-mark` value
new-priority (*integer* | *from-dscp* | *from-dscp-high-3-bits* | *from-ingress*; Default:)	Sets a new priority for a packet. This can be the VLAN, WMM, DSCP or MPLS EXP priority Read more. This property can also be used to set an internal priority.
new-routing-mark (*string*; Default:)	Sets a new `routing-mark` value (in RouterOS v7 routing mark must be created before as a new Routing table)
new-ttl (*decrement* | *increment* | *set:integer*; Default:)	Sets a new Time to live value
route-dst (*IP, Default:*)	Matches packets with a specific gateway
passthrough (yes | no; *Default*: yes)	Whether passthrough is enabled for the rule