Manual:IP/Firewall/Filter

From MikroTik Wiki
< Manual:IP | Firewall

Applies to RouterOS: v3, v4+

Contents

Summary

Sub-menu: `/ip firewall filter`

The firewall implements packet filtering and thereby provides security functions that are used to manage data flow to, from and through the router. Along with the Network Address Translation it serves as a tool for preventing unauthorized access to directly attached networks and the router itself as well as a filter for outgoing traffic.

Network firewalls keep outside threats away from sensitive data available inside the network. Whenever different networks are joined together, there is always a threat that someone from outside of your network will break into your LAN. Such break-ins may result in private data being stolen and distributed, valuable data being altered or destroyed, or entire hard drives being erased. Firewalls are used as a means of preventing or minimizing the security risks inherent in connecting to other networks. Properly configured firewall plays a key role in efficient and secure network infrastrure deployment.

MikroTik RouterOS has very powerful firewall implementation with features including:

- stateful packet inspection
- Layer-7 protocol detection
- peer-to-peer protocols filtering
- traffic classification by:
- source MAC address
- IP addresses (network or list) and address types (broadcast, local,

multicast, unicast)
- port or port range
- IP protocols
- protocol options (ICMP type and code fields, TCP flags, IP options and MSS)
- interface the packet arrived from or left through
- internal flow and connection marks
- DSCP byte
- packet content
- rate at which packets arrive and sequence numbers
- packet size
- packet arrival time
- and much more!

Chains

The firewall operates by means of firewall rules. Each rule consists of two parts - the matcher which matches traffic flow against given conditions and the action which defines what to do with the matched packet.

Firewall filtering rules are grouped together in chains. It allows a packet to be matched against one common criterion in one chain, and then passed over for processing against some other common criteria to another chain. For example a packet should be matched against the IP address:port pair. Of course, it could be achieved by adding as many rules with IP address:port match as required to the forward chain, but a better way could be to add one rule that matches traffic from a particular IP address, e.g.: /ip firewall filter add src-address=1.1.1.2/32 jump-target="mychain" and in case of successfull match passes control over the IP packet to some other chain, id est mychain in this example. Then rules that perform matching against separate ports can be added to mychain chain without specifying the IP addresses.

There are three predefined chains, which cannot be deleted:

- **input** - used to process packets entering the router through one of the interfaces with the destination IP address which is one of the router's addresses. Packets passing through the router are not processed against the rules of the input chain
- **forward** - used to process packets passing through the router
- **output** - used to process packets originated from the router and leaving it through one of the interfaces. Packets passing through the router are not processed against the rules of the output chain

Packet flow diagrams illustrate how packets are processed in RouterOS.

When processing a chain, rules are taken from the chain in the order they are listed there from top to bottom. If a packet matches the criteria of the rule, then the specified action is performed on it, and no more rules are processed in that chain (the exception is the passthrough action). If a packet has not matched any rule within the built-in chain, then it is accepted.

Properties

Property	Description
`action` (*action name*; Default: **accept**)	Action to take if packet is matched by the rule: ▪ `accept` - accept the packet. Packet is not passed to next firewall rule. ▪ `add-dst-to-address-list` - add destination address to address list specified by `address-list` parameter ▪ `add-src-to-address-list` - add source address to address list specified by `address-list` parameter ▪ `drop` - silently drop the packet ▪ `jump` - jump to the user defined chain specified by the value of `jump-target` parameter ▪ `log` - add a message to the system log containing following data: in-interface, out-interface, src-mac, protocol, src-ip:port->dst-ip:port and length of the packet. After packet is matched it is passed to next rule in the list, similar as `passthrough` ▪ `passthrough` - if packet is matched by the rule, increase counter and go to next rule (useful for statistics). ▪ `reject` - drop the packet and send an ICMP reject message ▪ `return` - passes control back to the chain from where the jump took place ▪ `tarpit` - captures and holds TCP connections (replies with SYN/ACK to the inbound TCP SYN packet)
`address-list` (*string*; Default:)	Name of the address list to be used. Applicable if action is `add-dst-to-address-list` or `add-src-to-address-list`
`address-list-timeout` (*time*; Default: **00:00:00**)	Time interval after which the address will be removed from the address list specified by `address-`

	`list` parameter. Used in conjunction with `add-dst-to-address-list` or `add-src-to-address-list` actions Value of `00:00:00` will leave the address in the address list forever
`chain` (*name*; Default:)	Specifies to which chain rule will be added. If the input does not match the name of an already defined chain, a new chain will be created.
`comment` (*string*; Default:)	Descriptive comment for the rule.
`connection-bytes` (*integer-integer*; Default:)	Matches packets only if a given amount of bytes has been transfered through the particular connection. 0 - means infinity, for

	example `connection-bytes=2000000-0` means that the rule matches if more than 2MB has been transfered through the relevant connection
`connection-limit` (*integer,netmask*; Default:)	Matches connections per address or address block up to and including given value. Should be used together with connection-state=new and/or with tcp-flags=syn because matcher is very resource intensive.
`connection-mark` (*no-mark | string*; Default:)	Matches packets marked via mangle facility with particular connection mark. If **no-mark** is set, rule will match any unmarked connection.
`connection-nat-state` (*srcnat | dstnat*; Default:)	

	Can match connections that are srcnatted, dstnatted or both. Note that connection-state=related connections connection-nat-state is determined by direction of the first packet. and if connection tracking needs to use dst-nat to deliver this connection to same hosts as main connection it will be in connection-nat-state=dstnat even if there are no dst-nat rules at all.
`connection-rate` (*Integer 0..4294967295*; Default:)	Connection Rate is a firewall matcher that allow to capture traffic based on present speed of the connection. Read more >>
`connection-state` (*estabilished | invalid | new | related | untracked*; Default:)	Interprets the connection tracking analysis data for a particular packet: - `established` - a packet which belongs to an existing connection - `invalid` - a packet that does not have determined state in connection tracking (ussualy - sevear out-of-order packets, packets with wrong

	sequence/ack number, or in case of resource overusage on router), for this reason invalid packet will not participate in NAT (as only connection-state=new packets do), and will still contain original source IP address when routed. We strongly suggest to drop all connection-state=invalid packets in firewall filter forward and input chains ▪ `new` - the packet has started a new connection, or otherwise associated with a connection which has not seen packets in both directions. ▪ `related` - a packet which is related to, but not part of an existing connection, such as ICMP errors or a packet which begins FTP data connection ▪ `untracked` - packet which was set to bypass connection tracking in firewall RAW tables.
`connection-type` (*ftp | h323 | irc | pptp | quake3 | sip | tftp*; Default:)	Matches packets from related connections based on information from their connection tracking helpers. A relevant connection helper must be enabled under /ip firewall service-port
`content` (*string*; Default:)	Match packets that contain specified text
`dscp` (*integer: 0..63*; Default:)	Matches DSCP IP header field.
`dst-address` (*IP/netmask | IP range*; Default:)	Matches packets which destination is equal to specified IP or falls into specified IP range.

dst-address-list (*name*; Default:)	Matches destination address of a packet against user-defined address list
dst-address-type (*unicast | local | broadcast | multicast*; Default:)	Matches destination address type: - `unicast` - IP address used for point to point transmission - `local` - if dst-address is assigned to one of router's interfaces - `broadcast` - packet is sent to all devices in subnet - `multicast` - packet is forwarded to defined group of devices
dst-limit (*integer[/time],integer,dst-address | dst-port | src-address[/time]*; Default:)	Matches packets until a given rate is exceeded. Rate is defined as packets per time interval. As opposed to the `limit` matcher, every flow has it's own limit. Flow is defined by mode parameter. Parameters are written in following format: `count[/time],burst,mode[/expire]`. - **count** - packet count per time interval per flow to match - **time** - specifies the time interval in which the packet count per flow cannot be exceeded (optional, 1s will be used if not specified) - **burst** - initial number of packets per flow to match: this number gets recharged by one every `time/ count`, up to this number - **mode** - this parameter specifies what unique fields define flow (src-address, dst-address, src-and-dst-address, dst-address-and-port, addresses-and-dst-port) - **expire** - specifies interval after which flow with no packets will be allowed to be deleted (optional)
dst-port (*integer[-integer]: 0..65535*; Default:)	List of destination port numbers or port number ranges
fragment (*yes|no*; Default:)	Matches fragmented packets. First (starting) fragment does not count. If connection tracking is enabled there will be no

	fragments as system automatically assembles every packet
`hotspot` (*auth | from-client | http | local-dst | to-client*; Default:)	
`icmp-options` (*integer:integer*; Default:)	Matches ICMP type:code fileds
`in-bridge-port` (*name*; Default:)	Actual interface the packet has entered the router, if incoming interface is bridge. Works only if **use-ip-firewall** is enabled in bridge settings.
`in-bridge-port-list` (*name*; Default:)	Set of interfaces defined in interface list. Works the same as `in-bridge-port`
`in-interface` (*name*; Default:)	Interface the packet has entered the router
`in-interface-list` (*name*; Default:)	Set of interfaces defined in interface list. Works the same as `in-interface`
`ingress-priority` (*integer: 0..63*; Default:)	Matches ingress priority of

	the packet. Priority may be derived from VLAN, WMM or MPLS EXP bit. Read more>>
`ipsec-policy` (*in | out, ipsec | none*; Default:)	Matches the policy used by IpSec. Value is written in following format: **`direction, policy`**. Direction is Used to select whether to match the policy used for decapsulation or the policy that will be used for encapsulation. - `in` - valid in the PREROUTING, INPUT and FORWARD chains - `out` - valid in the POSTROUTING, OUTPUT and FORWARD chains - `ipsec` - matches if the packet is subject to IpSec processing; - `none` - matches packet that is not subject to IpSec processing (for example, IpSec transport packet). For example, if router receives Ipsec encapsulated Gre packet, then rule `ipsec-policy=in,ipsec` will match Gre packet, but rule `ipsec-policy=in,none` will match ESP packet.
`ipv4-options` (*any | loose-source-routing | no-record-route | no-router-alert | no-source-routing | no-timestamp | none | record-route | router-alert | strict-source-routing | timestamp*; Default:)	Matches IPv4 header options. - `any` - match packet with at least one of the ipv4 options - `loose-source-routing` - match packets with loose source routing option. This option is used to route the internet datagram based on information supplied by the source - `no-record-route` - match packets with no record route option. This option is used to route the internet datagram based on information supplied by the source - `no-router-alert` - match packets with no router alter option - `no-source-routing` - match packets with no source routing option - `no-timestamp` - match packets with no timestamp option - `record-route` - match packets with record route option - `router-alert` - match packets with router alter option - `strict-source-routing` - match packets with strict source routing option - `timestamp` - match packets with timestamp
`jump-target` (*name*; Default:)	Name of the target chain to jump to. Applicable only if `action=jump`
`layer7-protocol` (*name*; Default:)	Layer7 filter name defined in layer7 protocol menu.
`limit` (*integer,time,integer*; Default:)	

	Matches packets up to a limited rate (packet rate or bit rate). Rule using this matcher will match until this limit is reached. Parameters are written in following format: `count[/time],burst:mode`. ▪ **count** - packet or bit count per time interval to match ▪ **time** - specifies the time interval in which the packet or bit count cannot be exceeded (optional, 1s will be used if not specified) ▪ **burst** - initial number of packets or bits to match: this number gets recharged every 10ms so burst should be at least 1/100 of rate per second ▪ **mode** - packet or bit mode
`log-prefix` (*string*; Default:)	Adds specified text at the beginning of every log message. Applicable if `action=log`
`nth` (*integer,integer*; Default:)	Matches every nth packet. Read more >>
`out-bridge-port` (*name*; Default:)	Actual interface the packet is leaving the router, if outgoing interface is bridge. Works only if **use-ip-firewall** is enabled in bridge settings.
`out-bridge-port-list` (*name*; Default:)	Set of interfaces defined in interface list. Works the same as `out-bridge-port`
`out-interface` (; Default:)	Interface the packet is leaving the router

`out-interface-list` (*name*; Default:)	Set of interfaces defined in interface list. Works the same as `out-interface`
`p2p` (*all-p2p | bit-torrent | blubster | direct-connect | edonkey | fasttrack | gnutella | soulseek | warez | winmx*; Default:)	Matches packets from various peer-to-peer (P2P) protocols. Does not work on encrypted p2p packets.
`packet-mark` (*no-mark | string*; Default:)	Matches packets marked via mangle facility with particular packet mark. If **no-mark** is set, rule will match any unmarked packet.
`packet-size` (*integer[-integer]:0..65535*; Default:)	Matches packets of specified size or size range in bytes.
`per-connection-classifier` (*ValuesToHash:Denominator/Remainder*; Default:)	PCC matcher allows to divide traffic into equal streams with ability to keep packets with specific set of options in one

	particular stream. Read more >>
`port` (*integer[-integer]: 0..65535*; Default:)	Matches if any (source or destination) port matches the specified list of ports or port ranges. Applicable only if `protocol` is TCP or UDP
`priority` (*integer: 0..63*; Default:)	
`protocol` (*name or protocol ID*; Default: **tcp**)	Matches particular IP protocol specified by protocol name or number
`psd` (*integer,time,integer,integer*; Default:)	Attempts to detect TCP and UDP scans. Parameters are in following format `WeightThreshold, DelayThreshold, LopPortWeight, HighPortWeight` - **WeightThreshold** - total weight of the latest TCP/UDP packets with different destination ports coming from the same host to be treated as port scan sequence - **DelayThreshold** - delay for the packets with different destination ports coming from the same host to be treated as possible port scan subsequence - **LowPortWeight** - weight of the packets with privileged (<=1024) destination port - **HighPortWeight** - weight of the packet with non-priviliged destination port
`random` (*integer: 1..99*; Default:)	Matches packets randomly with given probability.
`reject-with` (*icmp-admin-prohibited | icmp-net-prohibited | icmp-protocol-unreachable | icmp-host-prohibited | icmp-network-unreachable | tcp-reset | icmp-host-unreachable | icmp-port-unreachable*; Default: icmp-network-unreachable)	Specifies ICMP error to be sent back if packet is

	rejected. Applicable if `action=reject`
`routing-mark` (*string*; Default:)	Matches packets marked by mangle facility with particular routing mark
`src-address` (*Ip/Netmaks, Ip range*; Default:)	Matches packets which source is equal to specified IP or falls into specified IP range.
`src-address-list` (*name*; Default:)	Matches source address of a packet against user-defined address list
`src-address-type` (*unicast | local | broadcast | multicast*; Default:)	Matches source address type: - `unicast` - IP address used for point to point transmission - `local` - if address is assigned to one of router's interfaces - `broadcast` - packet is sent to all devices in subnet - `multicast` - packet is forwarded to defined group of devices
`src-port` (*integer[-integer]: 0..65535*; Default:)	List of source ports and ranges of source ports. Applicable only if protocol is TCP or UDP.
`src-mac-address` (*MAC address*; Default:)	Matches source MAC address of the packet

`tcp-flags` (*ack | cwr | ece | fin | psh | rst | syn | urg*; Default:)	Matches specified TCP flags - `ack` - acknowledging data - `cwr` - congestion window reduced - `ece` - ECN-echo flag (explicit congestion notification) - `fin` - close connection - `psh` - push function - `rst` - drop connection - `syn` - new connection - `urg` - urgent data
`tcp-mss` (*integer: 0..65535*; Default:)	Matches TCP MSS value of an IP packet
`time` (*time-time,sat | fri | thu | wed | tue | mon | sun*; Default:)	Allows to create filter based on the packets' arrival time and date or, for locally generated packets, departure time and date
`tls-host` (*string*; Default:)	Allows to match https traffic based on TLS SNI hostname. Accepts GLOB syntax (https://en.wikipedia.org/wiki/Glob_(programming)) for wildcard matching. Note that matcher will not be able to match hostname if TLS handshake frame is

	fragmented into multiple TCP segments (packets).
`ttl` (*integer: 0..255*; Default:)	Matches packets TTL value

Stats

`/ip firewall filter print stats` will show additional read-only properties

Property	Description
`bytes` (*integer*)	Total amount of bytes matched by the rule
`packets` (*integer*)	Total amount of packets matched by the rule

By default **print** is equivalent to **print static** and shows only static rules.

```
[admin@dzeltenais_burkaans] /ip firewall mangle> print stats
Flags: X - disabled, I - invalid, D - dynamic
 #   CHAIN              ACTION                 BYTES           PACKETS
 0   prerouting         mark-routing           17478158        127631
 1   prerouting         mark-routing           782505          4506
```

To print also dynamic rules use **print all**.

```
[admin@dzeltenais_burkaans] /ip firewall mangle> print all stats
Flags: X - disabled, I - invalid, D - dynamic
 #   CHAIN              ACTION                 BYTES           PACKETS
 0   prerouting         mark-routing           17478158        127631
 1   prerouting         mark-routing           782505          4506
 2 D forward            change-mss             0               0
 3 D forward            change-mss             0               0
 4 D forward            change-mss             0               0
 5 D forward            change-mss             129372          2031
```

Or to print only dynamic rules use **print dynamic**

```
[admin@dzeltenais_burkaans] /ip firewall mangle> print stats dynamic
Flags: X - disabled, I - invalid, D - dynamic
 #   CHAIN              ACTION                 BYTES           PACKETS
 0 D forward            change-mss             0               0
 1 D forward            change-mss             0               0
 2 D forward            change-mss             0               0
 3 D forward            change-mss             132444          2079
```

Menu specific commands

Property	Description
`reset-counters` (*id*)	Reset statistics counters for specified firewall rules.
`reset-counters-all` ()	Reset statistics counters for all firewall rules.

Basic examples

Router protection

Lets say our private network is 192.168.0.0/24 and public (WAN) interface is ether1. We will set up firewall to allow connections to router itself only from our local network and drop the rest. Also we will allow ICMP protocol on any interface so that anyone can ping your router from internet.

```
/ip firewall filter
add chain=input connection-state=invalid action=drop \
        comment="Drop Invalid connections"
add chain=input connection-state=established action=accept \
        comment="Allow Established connections"
add chain=input protocol=icmp action=accept \
        comment="Allow ICMP"
add chain=input src-address=192.168.0.0/24 action=accept \
        in-interface=!ether1
add chain=input action=drop comment="Drop everything else"
```

Customer protection

To protect the customer's network, we should check all traffic which goes through router and block unwanted. For icmp, tcp, udp traffic we will create chains, where will be droped all unwanted packets:

```
/ip firewall filter
add chain=forward protocol=tcp connection-state=invalid \
        action=drop comment="drop invalid connections"
add chain=forward connection-state=established action=accept \
        comment="allow already established connections"
add chain=forward connection-state=related action=accept \
        comment="allow related connections"
```

Block "bogon" IP addresses

```
add chain=forward src-address=0.0.0.0/8 action=drop
add chain=forward dst-address=0.0.0.0/8 action=drop
add chain=forward src-address=127.0.0.0/8 action=drop
add chain=forward dst-address=127.0.0.0/8 action=drop
add chain=forward src-address=224.0.0.0/3 action=drop
add chain=forward dst-address=224.0.0.0/3 action=drop
```

Make jumps to new chains:

```
add chain=forward protocol=tcp action=jump jump-target=tcp
add chain=forward protocol=udp action=jump jump-target=udp
add chain=forward protocol=icmp action=jump jump-target=icmp
```

Create tcp chain and deny some tcp ports in it:

```
add chain=tcp protocol=tcp dst-port=69 action=drop \
        comment="deny TFTP"
add chain=tcp protocol=tcp dst-port=111 action=drop \
        comment="deny RPC portmapper"
add chain=tcp protocol=tcp dst-port=135 action=drop \
        comment="deny RPC portmapper"
add chain=tcp protocol=tcp dst-port=137-139 action=drop \
        comment="deny NBT"
add chain=tcp protocol=tcp dst-port=445 action=drop \
        comment="deny cifs"
add chain=tcp protocol=tcp dst-port=2049 action=drop comment="deny NFS"
add chain=tcp protocol=tcp dst-port=12345-12346 action=drop comment="deny NetBus"
add chain=tcp protocol=tcp dst-port=20034 action=drop comment="deny NetBus"
add chain=tcp protocol=tcp dst-port=3133 action=drop comment="deny BackOriffice"
add chain=tcp protocol=tcp dst-port=67-68 action=drop comment="deny DHCP"
```

Deny udp ports in udp chain:

```
add chain=udp protocol=udp dst-port=69 action=drop comment="deny TFTP"
add chain=udp protocol=udp dst-port=111 action=drop comment="deny PRC portmapper"
add chain=udp protocol=udp dst-port=135 action=drop comment="deny PRC portmapper"
add chain=udp protocol=udp dst-port=137-139 action=drop comment="deny NBT"
add chain=udp protocol=udp dst-port=2049 action=drop comment="deny NFS"
add chain=udp protocol=udp dst-port=3133 action=drop comment="deny BackOriffice"
```

Allow only needed icmp codes in icmp chain:

```
add chain=icmp protocol=icmp icmp-options=0:0 action=accept \
        comment="echo reply"
add chain=icmp protocol=icmp icmp-options=3:0 action=accept \
        comment="net unreachable"
add chain=icmp protocol=icmp icmp-options=3:1 action=accept \
        comment="host unreachable"
add chain=icmp protocol=icmp icmp-options=3:4 action=accept \
        comment="host unreachable fragmentation required"
add chain=icmp protocol=icmp icmp-options=4:0 action=accept \
        comment="allow source quench"
add chain=icmp protocol=icmp icmp-options=8:0 action=accept \
        comment="allow echo request"
add chain=icmp protocol=icmp icmp-options=11:0 action=accept \
        comment="allow time exceed"
add chain=icmp protocol=icmp icmp-options=12:0 action=accept \
        comment="allow parameter bad"
add chain=icmp action=drop comment="deny all other types"
```

other ICMP codes are found here (http://www.iana.org/assignments/icmp-parameters).

Brute force protection

Bruteforce_login_prevention_(FTP_&_SSH)

[Top | Back to Content]

Retrieved from "https://wiki.mikrotik.com/index.php?title=Manual:IP/Firewall/Filter&oldid=30405"