NAME

ip-rule − routing policy database management

SYNOPSIS

ip [*OPTIONS*] **rule** { *COMMAND* | **help** }

ip rule [**list** [*SELECTOR*]]

ip rule { **add** | **del** } *SELECTOR ACTION*

ip rule { **flush** | **save** | **restore** }

SELECTOR := [**not**] [**from** *PREFIX*] [**to** *PREFIX*] [**tos** *TOS*] [**fwmark** *FWMARK*[/*MASK*]] [**iif** *STRING*] [**oif** *STRING*] [**pref** *NUMBER*] [*l3mdev*] [**uidrange** *NUMBER*-*NUMBER*] [**ipproto** *PROTOCOL*] [**sport** [*NUMBER* | *NUMBER*-*NUMBER*]] [**dport** [*NUMBER* | *NUMBER*-*NUMBER*]] [**tun_id** *TUN_ID*]

ACTION := [**table** *TABLE_ID*] [**protocol** *PROTO*] [**nat** *ADDRESS*] [**realms** [*SRCREALM*/]*DSTREALM*] [**goto** *NUMBER*] *SUPPRESSOR*

SUPPRESSOR := [**suppress_prefixlength** *NUMBER*] [**suppress_ifgroup** *GROUP*]

TABLE_ID := [**local** | **main** | **default** | *NUMBER*]

DESCRIPTION

ip rule manipulates rules in the routing policy database control the route selection algorithm.

Classic routing algorithms used in the Internet make routing decisions based only on the destination address of packets (and in theory, but not in practice, on the TOS field).

In some circumstances we want to route packets differently depending not only on destination addresses, but also on other packet fields: source address, IP protocol, transport protocol ports or even packet payload. This task is called 'policy routing'.

To solve this task, the conventional destination based routing table, ordered according to the longest match rule, is replaced with a 'routing policy database' (or RPDB), which selects routes by executing some set of rules.

Each policy routing rule consists of a **selector** and an **action predicate.** The RPDB is scanned in order of decreasing priority (note that lower number means higher priority, see the description of *PREFERENCE* below). The selector of each rule is applied to {source address, destination address, incoming interface, tos, fwmark} and, if the selector matches the packet, the action is performed. The action predicate may return with success. In this case, it will either give a route or failure indication and the RPDB lookup is terminated. Otherwise, the RPDB program continues with the next rule.

Semantically, the natural action is to select the nexthop and the output device.

At startup time the kernel configures the default RPDB consisting of three rules:

1. Priority: 0, Selector: match anything, Action: lookup routing table **local** (ID 255). The **local** table is a special routing table containing high priority control routes for local and broadcast addresses.

2. Priority: 32766, Selector: match anything, Action: lookup routing table **main** (ID 254). The **main** table is the normal routing table containing all non-policy routes. This rule may be deleted and/or overridden with other ones by the administrator.

3. Priority: 32767, Selector: match anything, Action: lookup routing table **default** (ID 253). The **default** table is empty. It is reserved for some post-processing if no previous default rules selected the packet. This rule may also be deleted.

Each RPDB entry has additional attributes. F.e. each rule has a pointer to some routing table. NAT and masquerading rules have an attribute to select new IP address to translate/masquerade. Besides that, rules have some optional attributes, which routes have, namely **realms**. These values do not override those contained in the routing tables. They are only used if the route did not select any attributes.

The RPDB may contain rules of the following types:

unicast - the rule prescribes to return the route found in the routing table referenced by the rule.

blackhole - the rule prescribes to silently drop the packet.

unreachable - the rule prescribes to generate a 'Network is unreachable' error.

prohibit - the rule prescribes to generate 'Communication is administratively prohibited' error.

nat - the rule prescribes to translate the source address of the IP packet into some other value.

ip rule add - insert a new rule

ip rule delete - delete a rule

type *TYPE* **(default)**
the type of this rule. The list of valid types was given in the previous subsection.

from *PREFIX*
select the source prefix to match.

to *PREFIX*
select the destination prefix to match.

iif *NAME*
select the incoming device to match. If the interface is loopback, the rule only matches packets originating from this host. This means that you may create separate routing tables for forwarded and local packets and, hence, completely segregate them.

oif *NAME*
select the outgoing device to match. The outgoing interface is only available for packets originating from local sockets that are bound to a device.

tos *TOS*

dsfield *TOS*
select the TOS value to match.

fwmark *MARK*
select the **fwmark** value to match.

uidrange *NUMBER-NUMBER*
select the **uid** value to match.

ipproto *PROTOCOL*
select the ip protocol value to match.

sport *NUMBER* | *NUMBER-NUMBER*
select the source port value to match. supports port range.

dport *NUMBER* | *NUMBER-NUMBER*
select the destination port value to match. supports port range.

priority *PREFERENCE*
the priority of this rule. *PREFERENCE* is an unsigned integer value, higher number means lower priority, and rules get processed in order of increasing number. Each rule should have an explicitly set *unique* priority value. The options preference and order are synonyms with priority.

table *TABLEID*
the routing table identifier to lookup if the rule selector matches. It is also possible to use lookup instead of table.

protocol *PROTO*
the routing protocol who installed the rule in question. As an example when zebra installs a rule it would get RTPROT_ZEBRA as the installing protocol.

suppress_prefixlength *NUMBER*
reject routing decisions that have a prefix length of NUMBER or less.

suppress_ifgroup *GROUP*
reject routing decisions that use a device belonging to the interface group GROUP.

realms *FROM/TO*
Realms to select if the rule matched and the routing table lookup succeeded. Realm *TO* is only used if the route did not select any realm.

nat *ADDRESS*
The base of the IP address block to translate (for source addresses). The *ADDRESS* may be either the start of the block of NAT addresses (selected by NAT routes) or a local host address (or even zero). In the last case the router does not translate the packets, but masquerades them to this address. Using map-to instead of nat means the same thing.

Warning: Changes to the RPDB made with these commands do not become active immediately. It is assumed that after a script finishes a batch of updates, it flushes the routing cache with **ip route flush cache**.

ip rule flush - also dumps all the deleted rules.

protocol *PROTO*
Select the originating protocol.

ip rule show - list rules
This command has no arguments. The options list or lst are synonyms with show.

ip rule save

protocol *PROTO*
Select the originating protocol.

save rules table information to stdout
This command behaves like **ip rule show** except that the output is raw data suitable for passing to **ip rule restore**.

ip rule restore
restore rules table information from stdin
This command expects to read a data stream as returned from **ip rule save**. It will attempt to restore the rules table information exactly as it was at the time of the save. Any rules already in the table are left unchanged, and duplicates are not ignored.

SEE ALSO

ip(8)

AUTHOR

Original Manpage by Michail Litvak <mci@owl.openwall.com>