Rocky Enterprise Linux 9.2 Manual Pages on command 'tc-pedit.8'

C:\>man tc-pedit.8

Generic packet editor action in tc(8) Linux Generic packet editor action in tc(8)

NAME

pedit - generic packet editor action

SYNOPSIS

tc ... action pedit [ex] munge { RAW_OP | LAYERED_OP | EXTENDED_LAYERED_OP } [CON?

TROL]

RAW_OP := offset OFFSET { u8 | u16 | u32 } [AT_SPEC] CMD_SPEC

AT_SPEC := at AT offmask MASK shift SHIFT

LAYERED_OP := { ip IPHDR_FIELD | ip BEYOND_IPHDR_FIELD } CMD_SPEC

EXTENDED_LAYERED_OP := { eth ETHHDR_FIELD | ip IPHDR_FIELD | ip EX_IPHDR_FIELD |

ip6 IP6HDR_FIELD | tcp TCPHDR_FIELD | udp UDPHDR_FIELD } CMD_SPEC

ETHHDR_FIELD := { src | dst | type }

IPHDR_FIELD := { src | dst | tos | dsfield | ihl | protocol | precedence | nofrag |

firstfrag | ce | df }

BEYOND_IPHDR_FIELD := { dport | sport | icmp_type | icmp_code }

EX_IPHDR_FIELD := { ttl }

IP6HDR_FIELD := { src | dst | flow_lbl | payload_len | nexthdr | hoplimit }

TCPHDR_FIELD := { sport | dport | flags }

UDPHDR_FIELD := { sport | dport }

CMD_SPEC := { clear | invert | set VAL | add VAL | preserve } [retain RVAL]

CONTROL := { reclassify | pipe | drop | shot | continue | pass | goto chain

CHAIN_INDEX }

DESCRIPTION

The pedit action can be used to change arbitrary packet data. The location of data to change can either be specified by giving an offset and size as in RAW_OP, or for header values by naming the header and field to edit the size is then chosen automatically based on the header field size. Currently this is supported only for IPv4 headers.

OPTIONS

ex Use extended pedit. EXTENDED_LAYERED_OP and the add CMD_SPEC are allowed only in this mode.

offset OFFSET { u32 | u16 | u8 }

Specify the offset at which to change data. OFFSET is a signed integer, it's base is automatically chosen (e.g. hex if prefixed by 0x or octal if prefixed by 0). The second argument specifies the length of data to change, that is four bytes (u32), two bytes (u16) or a single byte (u8).

at AT offmask MASK shift SHIFT

This is an optional part of RAW_OP which allows to have a variable OFFSET depending on packet data at offset AT, which is binary ANDed with MASK and right-shifted by SHIFT before adding it to OFFSET.

eth ETHHDR_FIELD

Change an ETH header field. The supported keywords for ETHHDR_FIELD are:

src

dst Source or destination MAC address in the standard format:

XX:XX:XX:XX:XX:XX

type Ether-type in numeric value

ip IPHDR_FIELD

Change an IPv4 header field. The supported keywords for IPHDR_FIELD are:

src

dst Source or destination IP address, a four-byte value.

tos

dsfield

precedence

Type Of Service field, an eight-bit value.

ihl Change the IP Header Length field, a four-bit value.

protocol

Next-layer Protocol field, an eight-bit value.

nofrag

firstfrag

ce

df

mf Change IP header flags. Note that the value to pass to the set com?

mand is not just a bit value, but the full byte including the flags

field. Though only the relevant bits of that value are respected,

the rest ignored.

ip BEYOND_IPHDR_FIELD

Supported only for non-extended layered op. It is passed to the kernel as

offsets relative to the beginning of the IP header and assumes the IP header

is of minimum size (20 bytes). The supported keywords for BEYOND_IPHDR_FIELD

are:

dport

sport Destination or source port numbers, a 16-bit value. Indeed, IPv4

headers don't contain this information. Instead, this will set an

offset which suits at least TCP and UDP if the IP header is of mini?

mum size (20 bytes). If not, this will do unexpected things.

icmp_type

icmp_code

Again, this allows to change data past the actual IP header itself.

It assumes an ICMP header is present immediately following the (mini?

mal sized) IP header. If it is not or the latter is bigger than the

minimum of 20 bytes, this will do unexpected things. These fields are

eight-bit values.

ip EX_IPHDR_FIELD

Supported only when ex is used. The supported keywords for EX_IPHDR_FIELD

are:

ttl

ip6 IP6HDR_FIELD

The supported keywords for IP6HDR_FIELD are:

src

dst

flow_lbl

payload_len

nexthdr

hoplimit

tcp TCPHDR_FIELD

The supported keywords for TCPHDR_FIELD are:

sport

dport Source or destination TCP port number, a 16-bit value.

flags

udp UDPHDR_FIELD

The supported keywords for UDPHDR_FIELD are:

sport

dport Source or destination TCP port number, a 16-bit value.

clear Clear the addressed data (i.e., set it to zero).

invert Swap every bit in the addressed data.

set VAL

Set the addressed data to a specific value. The size of VAL is defined by

either one of the u32, u16 or u8 keywords in RAW_OP, or the size of the ad?

dressed header field in LAYERED_OP.

add VAL

Add the addressed data by a specific value. The size of VAL is defined by

the size of the addressed header field in EXTENDED_LAYERED_OP. This opera?

tion is supported only for extended layered op.

preserve

Keep the addressed data as is.

retain RVAL

This optional extra part of CMD_SPEC allows to exclude bits from being

changed. Supported only for 32 bits fields or smaller.

CONTROL

The following keywords allow to control how the tree of qdisc, classes, fil?

ters and actions is further traversed after this action.

reclassify

Restart with the first filter in the current list.

pipe Continue with the next action attached to the same filter.

drop

shot Drop the packet.

continue

Continue classification with the next filter in line.

pass Finish classification process and return to calling qdisc for further

packet processing. This is the default.

EXAMPLES

Being able to edit packet data, one could do all kinds of things, such as e.g. im?

plementing port redirection. Certainly not the most useful application, but as an

example it should do:

First, qdiscs need to be set up to attach filters to. For the receive path, a sim?

ple ingress qdisc will do, for transmit path a classful qdisc (HTB in this case) is

necessary:

```
tc qdisc replace dev eth0 root handle 1: htb
tc qdisc add dev eth0 ingress handle ffff:
```

Finally, a filter with pedit action can be added for each direction. In this case,

u32 is used matching on the port number to redirect from, while pedit then does the

actual rewriting:

```
tc filter add dev eth0 parent 1: u32 \
    match ip dport 23 0xffff \
    action pedit pedit munge ip dport set 22
tc filter add dev eth0 parent ffff: u32 \
    match ip sport 22 0xffff \
    action pedit pedit munge ip sport set 23
tc filter add dev eth0 parent ffff: u32 \
    match ip sport 22 0xffff \
    action pedit ex munge ip dst set 192.168.1.199
tc filter add dev eth0 parent ffff: u32 \
    match ip sport 22 0xffff \
    action pedit ex munge ip6 dst set fe80::dacb:8aff:fec7:320e
```

```
tc filter add dev eth0 parent ffff: u32 \
    match ip sport 22 0xffff \
    action pedit ex munge eth dst set 11:22:33:44:55:66
tc filter add dev eth0 parent ffff: u32 \
    match ip dport 23 0xffff \
    action pedit ex munge tcp dport set 22
```

SEE ALSO

tc(8), tc-htb(8), tc-u32(8)

iproute2 12 Jan 2015 Generic packet editor action in tc(8)