NAME

IFE - encapsulate/decapsulate metadata

SYNOPSIS

tc ... **action ife** *DIRECTION* [*ACTION*] [**dst** *DMAC*] [**src** *SMAC*] [**type** *TYPE*] [*CONTROL*] [**index** *INDEX*]

DIRECTION := { **decode** | **encode** }

ACTION := { **allow** *ATTR* | **use** *ATTR value* }

ATTR := { **mark** | **prio** | **tcindex** }

CONTROL := { **reclassify** | **use** | **pipe** | **drop** | **continue** | **ok** | **goto chain CHAIN_INDEX** }

DESCRIPTION

The **ife** action allows for a sending side to encapsulate arbitrary metadata, which is then decapsulated by the receiving end. The sender runs in encoding mode and the receiver in decode mode. Both sender and receiver must specify the same ethertype. In the future, a registered ethertype may be available as a default.

OPTIONS

decode For the receiving side; decode the metadata if the packet matches.

encode For the sending side. Encode the specified metadata if the packet matches.

allow Encode direction only. Allows encoding specified metadata.

use Encode direction only. Enforce static encoding of specified metadata.

mark [*u32_value*] The value to set for the skb mark. The u32 value is required only when **use** is specified. If **mark** value is zero, it will not be encoded, instead "overlimits" statistics increment and **CONTROL** action is taken.

prio [*u32_value*] The value to set for priority in the skb structure. The u32 value is required only when **use** is specified.

tcindex [
u16_value] Value to set for the traffic control index in the skb structure. The u16 value is required only when **use** is specified.

dmac *DMAC*
smac *SMAC*
Optional six byte destination or source MAC address to encode.

type *TYPE*
Optional 16-bit ethertype to encode. If not specified value of 0xED3E will be used.

CONTROL
Action to take following an encode/decode.

index *INDEX*
Assign a unique ID to this action instead of letting the kernel choose one automatically. *INDEX* is a 32bit unsigned integer greater than zero.

EXAMPLES

On the receiving side, match packets with ethertype 0xdead and restart classification so that it will match ICMP on the next rule, at prio 3:

```
# tc qdisc add dev eth0 handle ffff: ingress
# tc filter add dev eth0 parent ffff: prio 2 protocol 0xdead \
      u32 match u32 0 0 flowid 1:1 \
      action ife decode reclassify
# tc filter add dev eth0 parent ffff: prio 3 protocol ip \
      u32 match ip protocol 0xff flowid 1:1 \
```

```
            action continue
```

Match with skb mark of 17:

```
      # tc filter add dev eth0 parent ffff: prio 4 protocol ip \
            handle 0x11 fw flowid 1:1 \
            action ok
```

Configure the sending side to encode for the filters above. Use a destination IP address of 192.168.122.237/24, then tag with skb mark of decimal 17. Encode the packaet with ethertype 0xdead, add skb->mark to whitelist of metadatum to send, and rewrite the destination MAC address to 02:15:15:15:15:15.

```
      # tc qdisc add dev eth0 root handle 1: prio
      # tc filter add dev eth0 parent 1: protocol ip prio 10 u32 \
            match ip dst 192.168.122.237/24 \
            match ip protocol 1 0xff \
            flowid 1:2 \
            action skbedit mark 17 \
            action ife encode \
            type 0xDEAD \
            allow mark \
            dst 02:15:15:15:15:15
```

SEE ALSO

tc(8), **tc-u32**(8)