Rocky Enterprise Linux 9.2 Manual Pages on command 'ip-link.8'

$ man ip-link.8

IP-LINK(8)   Linux   IP-LINK(8)

NAME

     ip-link - network device configuration

SYNOPSIS

     ip link { COMMAND | help }

     ip link add [link DEVICE] [name] NAME

          [txqueuelen PACKETS]

          [address LLADDR] [broadcast LLADDR]

          [mtu MTU] [index IDX]

          [numtxqueues QUEUE_COUNT] [numrxqueues QUEUE_COUNT]

          [gso_max_size BYTES] [gso_max_segs SEGMENTS]

          type TYPE [ARGS]

     ip link delete { DEVICE | group GROUP } type TYPE [ARGS]

     ip link set { DEVICE | group GROUP }

          [{ up | down }]

          [type ETYPE TYPE_ARGS]

          [arp { on | off }]

          [dynamic { on | off }]

          [multicast { on | off }]

          [allmulticast { on | off }]

          [promisc { on | off }]

          [protodown { on | off }]

          [protodown_reason PREASON { on | off }]

[trailers { on | off }]

[txqueuelen PACKETS]

[name NEWNAME]

[address LLADDR]

[broadcast LLADDR]

[mtu MTU]

[netns { PID | NETNSNAME }]

[link-netnsid ID]

[alias NAME]

[vf NUM [mac LLADDR]

    [VFVLAN-LIST]

    [rate TXRATE]

    [max_tx_rate TXRATE]

    [min_tx_rate TXRATE]

    [spoofchk { on | off }]

    [query_rss { on | off }]

    [state { auto | enable | disable }]

    [trust { on | off }]

    [node_guid eui64]

    [port_guid eui64]]

[{ xdp | xdpgeneric | xdpdrv | xdpoffload } { off |

    object FILE [section NAME] [verbose] |

    pinned FILE }]

[master DEVICE]

[nomaster]

[vrf NAME]

[addrgenmode { eui64 | none | stable_secret | random }]

[macaddr [MACADDR]

    [{ flush | add | del } MACADDR]

    [set MACADDR]]

 ip link show [DEVICE | group GROUP] [up] [master DEVICE] [type ETYPE] [

    vrf NAME]

 ip link xstats type TYPE [ARGS]

ip link afstats [dev DEVICE]

ip link help [TYPE]

TYPE := [bridge | bond | can | dummy | hsr | ifb | ipoib | macvlan | macvtap |

    vcan | vxcan | veth | vlan | vxlan | ip6tnl | ipip | sit | gre | gretap |

    | erspan | ip6gre | ip6gretap | ip6erspan | vti | nlmon | ipvlan | ipv?

    tap | lowpan | geneve | bareudp | vrf | macsec | netdevsim | rmnet |

    xfrm]

ETYPE := [TYPE | bridge_slave | bond_slave]

VFVLAN-LIST := [VFVLAN-LIST] VFVLAN

VFVLAN := [vlan VLANID [qos VLAN-QOS] [proto VLAN-PROTO]]

ip link property add dev DEVICE [altname NAME ..]

ip link property del dev DEVICE [altname NAME ..]

DESCRIPTION

  ip link add - add virtual link

    link DEVICE

        specifies the physical device to act operate on.

        NAME specifies the name of the new virtual device.

        TYPE specifies the type of the new device.

        Link types:

            bridge - Ethernet Bridge device

            bond - Bonding device

            dummy - Dummy network interface

            hsr - High-availability Seamless Redundancy device

            ifb - Intermediate Functional Block device

            ipoib - IP over Infiniband device

            macvlan - Virtual interface base on link layer address (MAC)

            macvtap - Virtual interface based on link layer address (MAC) and TAP.

            vcan - Virtual Controller Area Network interface

            vxcan - Virtual Controller Area Network tunnel interface

            veth - Virtual ethernet interface

            vlan - 802.1q tagged virtual LAN interface

            vxlan - Virtual eXtended LAN

            ip6tnl - Virtual tunnel interface IPv4|IPv6 over IPv6

ipip - Virtual tunnel interface IPv4 over IPv4

sit - Virtual tunnel interface IPv6 over IPv4

gre - Virtual tunnel interface GRE over IPv4

gretap - Virtual L2 tunnel interface GRE over IPv4

erspan - Encapsulated Remote SPAN over GRE and IPv4

ip6gre - Virtual tunnel interface GRE over IPv6

ip6gretap - Virtual L2 tunnel interface GRE over IPv6

ip6erspan - Encapsulated Remote SPAN over GRE and IPv6

vti - Virtual tunnel interface

nlmon - Netlink monitoring device

ipvlan - Interface for L3 (IPv6/IPv4) based VLANs

ipvtap - Interface for L3 (IPv6/IPv4) based VLANs and TAP

lowpan - Interface for 6LoWPAN (IPv6) over IEEE 802.15.4 / Bluetooth

geneve - GEneric NEtwork Virtualization Encapsulation

bareudp - Bare UDP L3 encapsulation support

macsec - Interface for IEEE 802.1AE MAC Security (MACsec)

vrf - Interface for L3 VRF domains

netdevsim - Interface for netdev API tests

rmnet - Qualcomm rmnet device

xfrm - Virtual xfrm interface

numtxqueues QUEUE_COUNT

specifies the number of transmit queues for new device.

numrxqueues QUEUE_COUNT

specifies the number of receive queues for new device.

gso_max_size BYTES

specifies the recommended maximum size of a Generic Segment Offload packet the new

device should accept.

gso_max_segs SEGMENTS

specifies the recommended maximum number of a Generic Segment Offload segments the

new device should accept.

index IDX

specifies the desired index of the new virtual device. The link creation fails, if

the index is busy.

VLAN Type Support

For a link of type VLAN the following additional arguments are supported:

ip link add link DEVICE name NAME type vlan [protocol VLAN_PROTO] id VLANID [re?

order_hdr { on | off }] [gvrp { on | off }] [mvrp { on | off }] [loose_bind?

ing { on | off }] [bridge_binding { on | off }] [ingress-qos-map QOS-MAP] [

egress-qos-map QOS-MAP]

protocol VLAN_PROTO - either 802.1Q or 802.1ad.

id VLANID - specifies the VLAN Identifier to use. Note that numbers with a

leading " 0 " or " 0x " are interpreted as octal or hexadecimal, respec?

tively.

reorder_hdr { on | off } - specifies whether ethernet headers are reordered

or not (default is on).

If reorder_hdr is on then VLAN header will be not inserted immediately

but only before passing to the physical device (if this device does not

support VLAN offloading), the similar on the RX direction - by default

the packet will be untagged before being received by VLAN device. Re?

ordering allows to accelerate tagging on egress and to hide VLAN header

on ingress so the packet looks like regular Ethernet packet, at the

same time it might be confusing for packet capture as the VLAN header

does not exist within the packet.

VLAN offloading can be checked by ethtool(8):

ethtool -k <phy_dev> | grep tx-vlan-offload

where <phy_dev> is the physical device to which VLAN device is bound.

gvrp { on | off } - specifies whether this VLAN should be registered using

GARP VLAN

Registration Protocol.

mvrp { on | off } - specifies whether this VLAN should be registered using

Multiple VLAN

Registration Protocol.

loose_binding { on | off } - specifies whether the VLAN device state is

bound to the physical device state.

bridge_binding { on | off } - specifies whether the VLAN device link state

tracks the state of bridge ports that are members of the VLAN.

ingress-qos-map QOS-MAP - defines a mapping of VLAN header prio field to

the Linux internal packet priority on incoming frames. The format is

FROM:TO with multiple mappings separated by spaces.

egress-qos-map QOS-MAP - defines a mapping of Linux internal packet prior?

ity to VLAN header prio field but for outgoing frames. The format is the

same as for ingress-qos-map.

Linux packet priority can be set by iptables(8):

iptables -t mangle -A POSTROUTING [...] -j CLASSIFY --set-class 0:4

and this "4" priority can be used in the egress qos mapping to set VLAN

prio "5":

ip link set veth0.10 type vlan egress 4:5

VXLAN Type Support

For a link of type VXLAN the following additional arguments are supported:

ip link add DEVICE type vxlan id VNI [dev PHYS_DEV] [{ group | remote } IPADDR

] [local { IPADDR | any }] [ttl TTL] [tos TOS] [df DF] [flowlabel FLOWLA?

BEL] [dstport PORT] [srcport MIN MAX] [[no]learning] [[no]proxy] [[no]rsc

] [[no]l2miss] [[no]l3miss] [[no]udpcsum] [[no]udp6zerocsumtx] [

[no]udp6zerocsumrx] [ageing SECONDS] [maxaddress NUMBER] [[no]external] [

gbp] [gpe]

id VNI - specifies the VXLAN Network Identifier (or VXLAN Segment Identi?

fier) to use.

dev PHYS_DEV - specifies the physical device to use for tunnel endpoint

communication.

group IPADDR - specifies the multicast IP address to join. This parameter

cannot be specified with the remote parameter.

remote IPADDR - specifies the unicast destination IP address to use in out?

going packets when the destination link layer address is not known in the

VXLAN device forwarding database. This parameter cannot be specified with

the group parameter.

local IPADDR - specifies the source IP address to use in outgoing packets.

ttl TTL - specifies the TTL value to use in outgoing packets.

tos TOS - specifies the TOS value to use in outgoing packets.

df DF - specifies the usage of the Don't Fragment flag (DF) bit in outgoing

packets with IPv4 headers. The value inherit causes the bit to be copied

from the original IP header. The values unset and set cause the bit to be

always unset or always set, respectively. By default, the bit is not set.

flowlabel FLOWLABEL - specifies the flow label to use in outgoing packets.

dstport PORT - specifies the UDP destination port to communicate to the re?

mote

  VXLAN tunnel endpoint.

srcport MIN MAX - specifies the range of port numbers to use as UDP source

ports to communicate to the remote VXLAN tunnel endpoint.

[no]learning - specifies if unknown source link layer addresses and IP ad?

dresses are entered into the VXLAN device forwarding database.

[no]rsc - specifies if route short circuit is turned on.

[no]proxy - specifies ARP proxy is turned on.

[no]l2miss - specifies if netlink LLADDR miss notifications are generated.

[no]l3miss - specifies if netlink IP ADDR miss notifications are generated.

[no]udpcsum - specifies if UDP checksum is calculated for transmitted pack?

ets over IPv4.

[no]udp6zerocsumtx - skip UDP checksum calculation for transmitted packets

over IPv6.

[no]udp6zerocsumrx - allow incoming UDP packets over IPv6 with zero check?

sum field.

ageing SECONDS - specifies the lifetime in seconds of FDB entries learnt by

the kernel.

maxaddress NUMBER - specifies the maximum number of FDB entries.

[no]external - specifies whether an external control plane (e.g. ip route

encap) or the internal FDB should be used.

gbp - enables the Group Policy extension (VXLAN-GBP).

   Allows to transport group policy context across VXLAN network peers.

   If enabled, includes the mark of a packet in the VXLAN header for out?

   going packets and fills the packet mark based on the information found

   in the VXLAN header for incoming packets.

   Format of upper 16 bits of packet mark (flags);

   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

```
|-|-|-|-|-|-|-|-|-|-|D|-|-|A|-|-|-|

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

D := Don't Learn bit. When set, this bit indicates that the egress

VTEP MUST NOT learn the source address of the encapsulated frame.

A := Indicates that the group policy has already been applied to this

packet. Policies MUST NOT be applied by devices when the A bit is

set.

Format of lower 16 bits of packet mark (policy ID):

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Group Policy ID      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Example:

iptables -A OUTPUT [...] -j MARK --set-mark 0x800FF

gpe - enables the Generic Protocol extension (VXLAN-GPE). Currently, this

is only supported together with the external keyword.

VETH, VXCAN Type Support

For a link of types VETH/VXCAN the following additional arguments are supported:

ip link add DEVICE type { veth | vxcan } [peer name NAME]

peer name NAME - specifies the virtual pair device name of the VETH/VXCAN

tunnel.

IPIP, SIT Type Support

For a link of type IPIPorSIT the following additional arguments are supported:

ip link add DEVICE type { ipip | sit } remote ADDR local ADDR [encap { fou | gue

| none }] [encap-sport { PORT | auto }] [encap-dport PORT] [[no]encap-csum]

[[no]encap-remcsum] [mode { ip6ip | ipip | mplsip | any }] [external]

remote ADDR - specifies the remote address of the tunnel.

local ADDR - specifies the fixed local address for tunneled packets. It

must be an address on another interface on this host.

encap { fou | gue | none } - specifies type of secondary UDP encapsulation.

"fou" indicates Foo-Over-UDP, "gue" indicates Generic UDP Encapsulation.

encap-sport { PORT | auto } - specifies the source port in UDP encapsula?

tion. PORT indicates the port by number, "auto" indicates that the port

number should be chosen automatically (the kernel picks a flow based on the

flow hash of the encapsulated packet).

[no]encap-csum - specifies if UDP checksums are enabled in the secondary encapsulation.

[no]encap-remcsum - specifies if Remote Checksum Offload is enabled. This is only applicable for Generic UDP Encapsulation.

mode { ip6ip | ipip | mplsip | any } - specifies mode in which device should run. "ip6ip" indicates IPv6-Over-IPv4, "ipip" indicates "IPv4-Over-IPv4", "mplsip" indicates MPLS-Over-IPv4, "any" indicates IPv6, IPv4 or MPLS Over IPv4. Supported for SIT where the default is "ip6ip" and IPIP where the default is "ipip". IPv6-Over-IPv4 is not supported for IPIP.

external - make this tunnel externally controlled (e.g. ip route encap).

GRE Type Support

For a link of type GRE or GRETAP the following additional arguments are supported:

ip link add DEVICE type { gre | gretap } remote ADDR local ADDR [[no][i|o]seq] [[i|o]key KEY | no[i|o]key] [[no][i|o]csum] [ttl TTL] [tos TOS] [[no]pmtud? isc] [[no]ignore-df] [dev PHYS_DEV] [encap { fou | gue | none }] [encap-sport { PORT | auto }] [encap-dport PORT] [[no]encap-csum] [[no]encap-remcsum] [external]

remote ADDR - specifies the remote address of the tunnel.

local ADDR - specifies the fixed local address for tunneled packets. It must be an address on another interface on this host.

[no][i|o]seq - serialize packets. The oseq flag enables sequencing of out? going packets. The iseq flag requires that all input packets are serial? ized.

[i|o]key KEY | no[i|o]key - use keyed GRE with key KEY. KEY is either a number or an IPv4 address-like dotted quad. The key parameter specifies the same key to use in both directions. The ikey and okey parameters spec? ify different keys for input and output.

[no][i|o]csum - generate/require checksums for tunneled packets. The ocsum flag calculates checksums for outgoing packets. The icsum flag requires that all input packets have the correct checksum. The csum flag is equiva? lent to the combination icsum ocsum .

ttl TTL - specifies the TTL value to use in outgoing packets.

tos TOS - specifies the TOS value to use in outgoing packets.

[no]pmtudisc - enables/disables Path MTU Discovery on this tunnel. It is
enabled by default. Note that a fixed ttl is incompatible with this option:
tunneling with a fixed ttl always makes pmtu discovery.

[no]ignore-df - enables/disables IPv4 DF suppression on this tunnel. Nor?
mally datagrams that exceed the MTU will be fragmented; the presence of the
DF flag inhibits this, resulting instead in an ICMP Unreachable (Fragmenta?
tion Required) message. Enabling this attribute causes the DF flag to be
ignored.

dev PHYS_DEV - specifies the physical device to use for tunnel endpoint
communication.

encap { fou | gue | none } - specifies type of secondary UDP encapsulation.
"fou" indicates Foo-Over-UDP, "gue" indicates Generic UDP Encapsulation.

encap-sport { PORT | auto } - specifies the source port in UDP encapsula?
tion. PORT indicates the port by number, "auto" indicates that the port
number should be chosen automatically (the kernel picks a flow based on the
flow hash of the encapsulated packet).

[no]encap-csum - specifies if UDP checksums are enabled in the secondary
encapsulation.

[no]encap-remcsum - specifies if Remote Checksum Offload is enabled. This
is only applicable for Generic UDP Encapsulation.

external - make this tunnel externally controlled (e.g. ip route encap).

IP6GRE/IP6GRETAP Type Support

For a link of type IP6GRE/IP6GRETAP the following additional arguments are sup?
ported:

ip link add DEVICE type { ip6gre | ip6gretap } remote ADDR local ADDR [
[no][i|o]seq] [[i|o]key KEY | no[i|o]key] [[no][i|o]csum] [hoplimit TTL] [
encaplimit ELIM] [tclass TCLASS] [flowlabel FLOWLABEL] [dscp inherit] [
[no]allow-localremote] [dev PHYS_DEV] [external]

remote ADDR - specifies the remote IPv6 address of the tunnel.

local ADDR - specifies the fixed local IPv6 address for tunneled packets.
It must be an address on another interface on this host.

[no][i|o]seq - serialize packets. The oseq flag enables sequencing of out?

going packets. The iseq flag requires that all input packets are serial?

ized.

[i|o]key KEY | no[i|o]key - use keyed GRE with key KEY. KEY is either a

number or an IPv4 address-like dotted quad. The key parameter specifies

the same key to use in both directions. The ikey and okey parameters spec?

ify different keys for input and output.

[no][i|o]csum - generate/require checksums for tunneled packets. The ocsum

flag calculates checksums for outgoing packets. The icsum flag requires

that all input packets have the correct checksum. The csum flag is equiva?

lent to the combination icsum ocsum.

hoplimit TTL - specifies Hop Limit value to use in outgoing packets.

encaplimit ELIM - specifies a fixed encapsulation limit. Default is 4.

flowlabel FLOWLABEL - specifies a fixed flowlabel.

[no]allow-localremote - specifies whether to allow remote endpoint to have

an address configured on local host.

tclass TCLASS - specifies the traffic class field on tunneled packets,

which can be specified as either a two-digit hex value (e.g. c0) or a pre?

defined string (e.g. internet). The value inherit causes the field to be

copied from the original IP header. The values inherit/STRING or in?

herit/00..ff will set the field to STRING or 00..ff when tunneling non-IP

packets. The default value is 00.

external - make this tunnel externally controlled (or not, which is the de?

fault). In the kernel, this is referred to as collect metadata mode. This

flag is mutually exclusive with the remote, local, seq, key, csum, ho?

plimit, encaplimit, flowlabel and tclass options.

IPoIB Type Support

For a link of type IPoIB the following additional arguments are supported:

ip link add DEVICE name NAME type ipoib [pkey PKEY] [mode MODE]

pkey PKEY - specifies the IB P-Key to use.

mode MODE - specifies the mode (datagram or connected) to use.

ERSPAN Type Support

For a link of type ERSPAN/IP6ERSPAN the following additional arguments are sup?

ported:

ip link add DEVICE type { erspan | ip6erspan } remote ADDR local ADDR seq key KEY

erspan_ver version [erspan IDX] [erspan_dir { ingress | egress }] [erspan_hwid

hwid] [[no]allow-localremote] [external]

remote ADDR - specifies the remote address of the tunnel.

local ADDR - specifies the fixed local address for tunneled packets. It

must be an address on another interface on this host.

erspan_ver version - specifies the ERSPAN version number. version indi?

cates the ERSPAN version to be created: 0 for version 0 type I, 1 for ver?

sion 1 (type II) or 2 for version 2 (type III).

erspan IDX - specifies the ERSPAN v1 index field. IDX indicates a 20 bit

index/port number associated with the ERSPAN traffic's source port and di?

rection.

erspan_dir { ingress | egress } - specifies the ERSPAN v2 mirrored traf?

fic's direction.

erspan_hwid hwid - an unique identifier of an ERSPAN v2 engine within a

system. hwid is a 6-bit value for users to configure.

[no]allow-localremote - specifies whether to allow remote endpoint to have

an address configured on local host.

external - make this tunnel externally controlled (or not, which is the de?

fault). In the kernel, this is referred to as collect metadata mode. This

flag is mutually exclusive with the remote, local, erspan_ver, erspan,

erspan_dir and erspan_hwid options.

GENEVE Type Support

For a link of type GENEVE the following additional arguments are supported:

ip link add DEVICE type geneve id VNI remote IPADDR [ttl TTL] [tos TOS] [df DF

] [flowlabel FLOWLABEL] [dstport PORT] [[no]external] [[no]udpcsum] [

[no]udp6zerocsumtx] [[no]udp6zerocsumrx]

id VNI - specifies the Virtual Network Identifier to use.

remote IPADDR - specifies the unicast destination IP address to use in out?

going packets.

ttl TTL - specifies the TTL value to use in outgoing packets. "0" or "auto"

means use whatever default value, "inherit" means inherit the inner proto?

col's ttl. Default option is "0".

tos TOS - specifies the TOS value to use in outgoing packets.

df DF - specifies the usage of the Don't Fragment flag (DF) bit in outgoing

packets with IPv4 headers. The value inherit causes the bit to be copied

from the original IP header. The values unset and set cause the bit to be

always unset or always set, respectively. By default, the bit is not set.

flowlabel FLOWLABEL - specifies the flow label to use in outgoing packets.

dstport PORT - select a destination port other than the default of 6081.

[no]external - make this tunnel externally controlled (or not, which is the

default). This flag is mutually exclusive with the id, remote, ttl, tos and

flowlabel options.

[no]udpcsum - specifies if UDP checksum is calculated for transmitted pack?

ets over IPv4.

[no]udp6zerocsumtx - skip UDP checksum calculation for transmitted packets

over IPv6.

[no]udp6zerocsumrx - allow incoming UDP packets over IPv6 with zero check?

sum field.

Bareudp Type Support

For a link of type Bareudp the following additional arguments are supported:

ip link add DEVICE type bareudp dstport PORT ethertype PROTO [srcportmin PORT] [

[no]multiproto]

dstport PORT - specifies the destination port for the UDP tunnel.

ethertype PROTO - specifies the ethertype of the L3 protocol being tun?

nelled. ethertype can be given as plain Ethernet protocol number or using

the protocol name ("ipv4", "ipv6", "mpls_uc", etc.).

srcportmin PORT - selects the lowest value of the UDP tunnel source port

range.

[no]multiproto - activates support for protocols similar to the one speci?

fied by ethertype. When ethertype is "mpls_uc" (that is, unicast MPLS),

this allows the tunnel to also handle multicast MPLS. When ethertype is

"ipv4", this allows the tunnel to also handle IPv6. This option is disabled

by default.

MACVLAN and MACVTAP Type Support

For a link of type MACVLAN or MACVTAP the following additional arguments are sup?

ported:

ip link add link DEVICE name NAME type { macvlan | macvtap } mode { private | vepa

| bridge | passthru [nopromisc] | source [nodst] } [bcqueuelen { LENGTH }]

type { macvlan | macvtap } - specifies the link type to use. macvlan cre?

ates just a virtual interface, while macvtap in addition creates a charac?

ter device /dev/tapX to be used just like a tuntap device.

mode private - Do not allow communication between macvlan instances on the

same physical interface, even if the external switch supports hairpin mode.

mode vepa - Virtual Ethernet Port Aggregator mode. Data from one macvlan

instance to the other on the same physical interface is transmitted over

the physical interface. Either the attached switch needs to support hairpin

mode, or there must be a TCP/IP router forwarding the packets in order to

allow communication. This is the default mode.

mode bridge - In bridge mode, all endpoints are directly connected to each

other, communication is not redirected through the physical interface's

peer.

mode passthru [nopromisc] - This mode gives more power to a single end?

point, usually in macvtap mode. It is not allowed for more than one end?

point on the same physical interface. All traffic will be forwarded to this

endpoint, allowing virtio guests to change MAC address or set promiscuous

mode in order to bridge the interface or create vlan interfaces on top of

it. By default, this mode forces the underlying interface into promiscuous

mode. Passing the nopromisc flag prevents this, so the promisc flag may be

controlled using standard tools.

mode source [nodst] - allows one to set a list of allowed mac address,

which is used to match against source mac address from received frames on

underlying interface. This allows creating mac based VLAN associations, in?

stead of standard port or tag based. The feature is useful to deploy 802.1x

mac based behavior, where drivers of underlying interfaces doesn't allows

that. By default, packets are also considered (duplicated) for destination-

based MACVLAN. Passing the nodst flag stops matching packets from also go?

ing through the destination-based flow.

bcqueuelen { LENGTH } - Set the length of the RX queue used to process

broadcast and multicast packets. LENGTH must be a positive integer in the
range [0-4294967295]. Setting a length of 0 will effectively drop all
broadcast/multicast traffic. If not specified the macvlan driver default
(1000) is used. Note that all macvlans that share the same underlying de?
vice are using the same queue. The parameter here is a request, the actual
queue length used will be the maximum length that any macvlan interface has
requested. When listing device parameters both the bcqueuelen parameter as
well as the actual used bcqueuelen are listed to better help the user un?
derstand the setting.

High-availability Seamless Redundancy (HSR) Support

For a link of type HSR the following additional arguments are supported:

ip link add link DEVICE name NAME type hsr slave1 SLAVE1-IF slave2 SLAVE2-IF [su?
pervision ADDR-BYTE] [version { 0 | 1 } [proto { 0 | 1 }]

type hsr - specifies the link type to use, here HSR.

slave1 SLAVE1-IF - Specifies the physical device used for the first of the
two ring ports.

slave2 SLAVE2-IF - Specifies the physical device used for the second of the
two ring ports.

supervision ADDR-BYTE - The last byte of the multicast address used for HSR
supervision frames. Default option is "0", possible values 0-255.

version { 0 | 1 } - Selects the protocol version of the interface. Default
option is "0", which corresponds to the 2010 version of the HSR standard.
Option "1" activates the 2012 version.

proto { 0 | 1 } - Selects the protocol at the interface. Default option is
"0", which corresponds to the HSR standard. Option "1" activates the Paral?
lel Redundancy Protocol (PRP).

BRIDGE Type Support

For a link of type BRIDGE the following additional arguments are supported:

ip link add DEVICE type bridge [ageing_time AGEING_TIME] [group_fwd_mask MASK]
[group_address ADDRESS] [forward_delay FORWARD_DELAY] [hello_time HELLO_TIME]
[max_age MAX_AGE] [stp_state STP_STATE] [priority PRIORITY] [vlan_filtering
VLAN_FILTERING] [vlan_protocol VLAN_PROTOCOL] [vlan_default_pvid VLAN_DE?
FAULT_PVID] [vlan_stats_enabled VLAN_STATS_ENABLED] [vlan_stats_per_port

VLAN_STATS_PER_PORT] [mcast_snooping MULTICAST_SNOOPING] [mcast_router MULTI?

CAST_ROUTER] [mcast_query_use_ifaddr MCAST_QUERY_USE_IFADDR] [mcast_querier

MULTICAST_QUERIER] [mcast_hash_elasticity HASH_ELASTICITY] [mcast_hash_max

HASH_MAX] [mcast_last_member_count LAST_MEMBER_COUNT] [

mcast_startup_query_count STARTUP_QUERY_COUNT] [mcast_last_member_interval

LAST_MEMBER_INTERVAL] [mcast_membership_interval MEMBERSHIP_INTERVAL] [

mcast_querier_interval QUERIER_INTERVAL] [mcast_query_interval QUERY_INTERVAL] [

mcast_query_response_interval QUERY_RESPONSE_INTERVAL] [mcast_startup_query_in?

terval STARTUP_QUERY_INTERVAL] [mcast_stats_enabled MCAST_STATS_ENABLED] [

mcast_igmp_version IGMP_VERSION] [mcast_mld_version MLD_VERSION] [nf_call_ipta?

bles NF_CALL_IPTABLES] [nf_call_ip6tables NF_CALL_IP6TABLES] [nf_call_arptables

NF_CALL_ARPTABLES]

ageing_time AGEING_TIME - configure the bridge's FDB entries ageing time,

ie the number of seconds a MAC address will be kept in the FDB after a

packet has been received from that address. after this time has passed, en?

tries are cleaned up.

group_fwd_mask MASK - set the group forward mask. This is the bitmask that

is applied to decide whether to forward incoming frames destined to link-

local addresses, ie addresses of the form 01:80:C2:00:00:0X (defaults to 0,

ie the bridge does not forward any link-local frames).

group_address ADDRESS - set the MAC address of the multicast group this

bridge uses for STP. The address must be a link-local address in standard

Ethernet MAC address format, ie an address of the form 01:80:C2:00:00:0X,

with X

 in [0, 4..f].

forward_delay FORWARD_DELAY - set the forwarding delay in seconds, ie the

time spent in LISTENING state (before moving to LEARNING) and in LEARNING

state (before moving to FORWARDING). Only relevant if STP is enabled. Valid

values are between 2 and 30.

hello_time HELLO_TIME - set the time in seconds between hello packets sent

by the bridge, when it is a root bridge or a designated bridges. Only rel?

evant if STP is enabled. Valid values are between 1 and 10.

max_age MAX_AGE - set the hello packet timeout, ie the time in seconds un?

til another bridge in the spanning tree is assumed to be dead, after recep?

tion of its last hello message. Only relevant if STP is enabled. Valid val?

ues are between 6 and 40.

stp_state STP_STATE - turn spanning tree protocol on (STP_STATE > 0) or off

(STP_STATE == 0). for this bridge.

priority PRIORITY - set this bridge's spanning tree priority, used during

STP root bridge election. PRIORITY is a 16bit unsigned integer.

vlan_filtering VLAN_FILTERING - turn VLAN filtering on (VLAN_FILTERING > 0)

or off (VLAN_FILTERING == 0). When disabled, the bridge will not consider

the VLAN tag when handling packets.

vlan_protocol { 802.1Q | 802.1ad } - set the protocol used for VLAN filter?

ing.

vlan_default_pvid VLAN_DEFAULT_PVID - set the default PVID (native/untagged

VLAN ID) for this bridge.

vlan_stats_enabled VLAN_STATS_ENABLED - enable (VLAN_STATS_ENABLED == 1) or

disable (VLAN_STATS_ENABLED == 0) per-VLAN stats accounting.

vlan_stats_per_port VLAN_STATS_PER_PORT - enable (VLAN_STATS_PER_PORT == 1)

or disable (VLAN_STATS_PER_PORT == 0) per-VLAN per-port stats accounting.

Can be changed only when there are no port VLANs configured.

mcast_snooping MULTICAST_SNOOPING - turn multicast snooping on (MULTI?

CAST_SNOOPING > 0) or off (MULTICAST_SNOOPING == 0).

mcast_router MULTICAST_ROUTER - set bridge's multicast router if IGMP

snooping is enabled. MULTICAST_ROUTER is an integer value having the fol?

lowing meaning:

    0 - disabled.

    1 - automatic (queried).

    2 - permanently enabled.

mcast_query_use_ifaddr MCAST_QUERY_USE_IFADDR - whether to use the bridge's

own IP address as source address for IGMP queries (MCAST_QUERY_USE_IFADDR >

0) or the default of 0.0.0.0 (MCAST_QUERY_USE_IFADDR == 0).

mcast_querier MULTICAST_QUERIER - enable (MULTICAST_QUERIER > 0) or disable

(MULTICAST_QUERIER == 0) IGMP querier, ie sending of multicast queries by

the bridge (default: disabled).

mcast_querier_interval QUERIER_INTERVAL - interval between queries sent by other routers. if no queries are seen after this delay has passed, the bridge will start to send its own queries (as if mcast_querier was en‐abled).

mcast_hash_elasticity HASH_ELASTICITY - set multicast database hash elas‐ticity, ie the maximum chain length in the multicast hash table (defaults to 4).

mcast_hash_max HASH_MAX - set maximum size of multicast hash table (de‐faults to 512, value must be a power of 2).

mcast_last_member_count LAST_MEMBER_COUNT - set multicast last member count, ie the number of queries the bridge will send before stopping for‐warding a multicast group after a "leave" message has been received (de‐faults to 2).

mcast_last_member_interval LAST_MEMBER_INTERVAL - interval between queries to find remaining members of a group, after a "leave" message is received.

mcast_startup_query_count STARTUP_QUERY_COUNT - set the number of IGMP queries to send during startup phase (defaults to 2).

mcast_startup_query_interval STARTUP_QUERY_INTERVAL - interval between queries in the startup phase.

mcast_query_interval QUERY_INTERVAL - interval between queries sent by the bridge after the end of the startup phase.

mcast_query_response_interval QUERY_RESPONSE_INTERVAL - set the Max Re‐sponse Time/Maximum Response Delay for IGMP/MLD queries sent by the bridge.

mcast_membership_interval MEMBERSHIP_INTERVAL - delay after which the bridge will leave a group, if no membership reports for this group are re‐ceived.

mcast_stats_enabled MCAST_STATS_ENABLED - enable (MCAST_STATS_ENABLED > 0) or disable (MCAST_STATS_ENABLED == 0) multicast (IGMP/MLD) stats account‐ing.

mcast_igmp_version IGMP_VERSION - set the IGMP version.

mcast_mld_version MLD_VERSION - set the MLD version.

nf_call_iptables NF_CALL_IPTABLES - enable (NF_CALL_IPTABLES > 0) or dis‐able (NF_CALL_IPTABLES == 0) iptables hooks on the bridge.

nf_call_ip6tables NF_CALL_IP6TABLES - enable (NF_CALL_IP6TABLES > 0) or

disable (NF_CALL_IP6TABLES == 0) ip6tables hooks on the bridge.

nf_call_arptables NF_CALL_ARPTABLES - enable (NF_CALL_ARPTABLES > 0) or

disable (NF_CALL_ARPTABLES == 0) arptables hooks on the bridge.

MACsec Type Support

For a link of type MACsec the following additional arguments are supported:

ip link add link DEVICE name NAME type macsec [[address <lladdr>] port PORT |

sci SCI] [cipher CIPHER_SUITE] [icvlen { 8..16 }] [encrypt { on | off }] [

send_sci { on | off }] [end_station { on | off }] [scb { on | off }] [protect

{ on | off }] [replay { on | off } window { 0..2^32-1 }] [validate { strict |

check | disabled }] [encodingsa { 0..3 }]

address <lladdr> - sets the system identifier component of secure channel

for this MACsec device.

port PORT - sets the port number component of secure channel for this MAC?

sec device, in a range from 1 to 65535 inclusive. Numbers with a leading "

0 " or " 0x " are interpreted as octal and hexadecimal, respectively.

sci SCI - sets the secure channel identifier for this MACsec device. SCI

is a 64bit wide number in hexadecimal format.

cipher CIPHER_SUITE - defines the cipher suite to use.

icvlen LENGTH - sets the length of the Integrity Check Value (ICV).

encrypt on or encrypt off - switches between authenticated encryption, or

authenticity mode only.

send_sci on or send_sci off - specifies whether the SCI is included in ev?

ery packet, or only when it is necessary.

end_station on or end_station off - sets the End Station bit.

scb on or scb off - sets the Single Copy Broadcast bit.

protect on or protect off - enables MACsec protection on the device.

replay on or replay off - enables replay protection on the device.

window SIZE - sets the size of the replay window.

validate strict or validate check or validate disabled - sets the valida?

tion mode on the device.

encodingsa AN - sets the active secure association for transmission.

VRF Type Support

For a link of type VRF the following additional arguments are supported:

ip link add DEVICE type vrf table TABLE

    table table id associated with VRF device

RMNET Type Support

For a link of type RMNET the following additional arguments are supported:

ip link add link DEVICE name NAME type rmnet mux_id MUXID

    mux_id MUXID - specifies the mux identifier for the rmnet device, possible

    values 1-254.

XFRM Type Support

For a link of type XFRM the following additional arguments are supported:

ip link add DEVICE type xfrm dev PHYS_DEV [if_id IF_ID]

    dev PHYS_DEV - specifies the underlying physical interface from which

    transform traffic is sent and received.

    if_id IF-ID - specifies the hexadecimal lookup key used to send traffic to

    and from specific xfrm policies. Policies must be configured with the same

    key. If not set, the key defaults to 0 and will match any policies which

    similarly do not have a lookup key configuration.

ip link delete - delete virtual link

dev DEVICE

    specifies the virtual device to act operate on.

group GROUP

    specifies the group of virtual links to delete. Group 0 is not allowed to be

    deleted since it is the default group.

type TYPE

    specifies the type of the device.

ip link set - change device attributes

Warning: If multiple parameter changes are requested, ip aborts immediately after any of

the changes have failed. This is the only case when ip can move the system to an unpre?

dictable state. The solution is to avoid changing several parameters with one ip link set

call. The modifier change is equivalent to set.

dev DEVICE

    DEVICE specifies network device to operate on. When configuring SR-IOV Virtual

    Function (VF) devices, this keyword should specify the associated Physical Function

(PF) device.

group GROUP

GROUP has a dual role: If both group and dev are present, then move the device to

the specified group. If only a group is specified, then the command operates on all

devices in that group.

up and down

change the state of the device to UP or DOWN.

arp on or arp off

change the NOARP flag on the device.

multicast on or multicast off

change the MULTICAST flag on the device.

allmulticast on or allmulticast off

change the ALLMULTI flag on the device. When enabled, instructs network driver to

retrieve all multicast packets from the network to the kernel for further process?

ing.

promisc on or promisc off

change the PROMISC flag on the device. When enabled, activates promiscuous opera?

tion of the network device.

trailers on or trailers off

change the NOTRAILERS flag on the device, NOT used by the Linux and exists for BSD

compatibility.

protodown on or protodown off

change the PROTODOWN state on the device. Indicates that a protocol error has been

detected on the port. Switch drivers can react to this error by doing a phys down

on the switch port.

protodown_reason PREASON on or off

set PROTODOWN reasons on the device. protodown reason bit names can be enumerated

under /etc/iproute2/protodown_reasons.d/. possible reasons bits 0-31

dynamic on or dynamic off

change the DYNAMIC flag on the device. Indicates that address can change when in?

terface goes down (currently NOT used by the Linux).

name NAME

change the name of the device. This operation is not recommended if the device is

running or has some addresses already configured.

txqueuelen NUMBER

txqlen NUMBER

    change the transmit queue length of the device.

mtu NUMBER

    change the MTU of the device.

address LLADDRESS

    change the station address of the interface.

broadcast LLADDRESS

brd LLADDRESS

peer LLADDRESS

    change the link layer broadcast address or the peer address when the interface is

    POINTOPOINT.

netns NETNSNAME | PID

    move the device to the network namespace associated with name NETNSNAME or process

    PID.

    Some devices are not allowed to change network namespace: loopback, bridge, wire?

    less. These are network namespace local devices. In such case ip tool will return

    "Invalid argument" error. It is possible to find out if device is local to a single

    network namespace by checking netns-local flag in the output of the ethtool:

        ethtool -k DEVICE

    To change network namespace for wireless devices the iw tool can be used. But it

    allows to change network namespace only for physical devices and by process PID.

alias NAME

    give the device a symbolic name for easy reference.

group GROUP

    specify the group the device belongs to. The available groups are listed in file

    /etc/iproute2/group.

vf NUM specify a Virtual Function device to be configured. The associated PF device must

    be specified using the dev parameter.

        mac LLADDRESS - change the station address for the specified VF. The vf pa?

        rameter must be specified.

        vlan VLANID - change the assigned VLAN for the specified VF. When speci?

fied, all traffic sent from the VF will be tagged with the specified VLAN

ID. Incoming traffic will be filtered for the specified VLAN ID, and will

have all VLAN tags stripped before being passed to the VF. Setting this pa?

rameter to 0 disables VLAN tagging and filtering. The vf parameter must be

specified.

qos VLAN-QOS - assign VLAN QOS (priority) bits for the VLAN tag. When spec?

ified, all VLAN tags transmitted by the VF will include the specified pri?

ority bits in the VLAN tag. If not specified, the value is assumed to be 0.

Both the vf and vlan parameters must be specified. Setting both vlan and

qos as 0 disables VLAN tagging and filtering for the VF.

proto VLAN-PROTO - assign VLAN PROTOCOL for the VLAN tag, either 802.1Q or

802.1ad. Setting to 802.1ad, all traffic sent from the VF will be tagged

with VLAN S-Tag. Incoming traffic will have VLAN S-Tags stripped before

being passed to the VF. Setting to 802.1ad also enables an option to con?

catenate another VLAN tag, so both S-TAG and C-TAG will be in?

serted/stripped for outgoing/incoming traffic, respectively. If not speci?

fied, the value is assumed to be 802.1Q. Both the vf and vlan parameters

must be specified.

rate TXRATE -- change the allowed transmit bandwidth, in Mbps, for the

specified VF. Setting this parameter to 0 disables rate limiting. vf pa?

rameter must be specified. Please use new API max_tx_rate option instead.

max_tx_rate TXRATE - change the allowed maximum transmit bandwidth, in

Mbps, for the specified VF. Setting this parameter to 0 disables rate lim?

iting. vf parameter must be specified.

min_tx_rate TXRATE - change the allowed minimum transmit bandwidth, in

Mbps, for the specified VF. Minimum TXRATE should be always <= Maximum

TXRATE. Setting this parameter to 0 disables rate limiting. vf parameter

must be specified.

spoofchk on|off - turn packet spoof checking on or off for the specified

VF.

query_rss on|off - toggle the ability of querying the RSS configuration of

a specific

VF. VF RSS information like RSS hash key may be considered sensitive

on some devices where this information is shared between VF and PF

and thus its querying may be prohibited by default.

state auto|enable|disable - set the virtual link state as seen by the spec?

ified VF. Setting to auto means a reflection of the PF link state, enable

lets the VF to communicate with other VFs on this host even if the PF link

state is down, disable causes the HW to drop any packets sent by the VF.

trust on|off - trust the specified VF user. This enables that VF user can

set a specific feature which may impact security and/or performance. (e.g.

VF multicast promiscuous mode)

node_guid eui64 - configure node GUID for Infiniband VFs.

port_guid eui64 - configure port GUID for Infiniband VFs.

xdp object | pinned | off

set (or unset) a XDP ("eXpress Data Path") BPF program to run on every packet at

driver level. ip link output will indicate a xdp flag for the networking device.

If the driver does not have native XDP support, the kernel will fall back to a

slower, driver-independent "generic" XDP variant. The ip link output will in that

case indicate xdpgeneric instead of xdp only. If the driver does have native XDP

support, but the program is loaded under xdpgeneric object | pinned then the kernel

will use the generic XDP variant instead of the native one. xdpdrv has the oppo?

site effect of requestsing that the automatic fallback to the generic XDP variant

be disabled and in case driver is not XDP-capable error should be returned. xdpdrv

also disables hardware offloads. xdpoffload in ip link output indicates that the

program has been offloaded to hardware and can also be used to request the "off?

load" mode, much like xdpgeneric it forces program to be installed specifically in

HW/FW of the apater.

off (or none) - Detaches any currently attached XDP/BPF program from the given de?

vice.

object FILE - Attaches a XDP/BPF program to the given device. The FILE points to a

BPF ELF file (f.e. generated by LLVM) that contains the BPF program code, map spec?

ifications, etc. If a XDP/BPF program is already attached to the given device, an

error will be thrown. If no XDP/BPF program is currently attached, the device sup?

ports XDP and the program from the BPF ELF file passes the kernel verifier, then it

will be attached to the device. If the option -force is passed to ip then any prior

attached XDP/BPF program will be atomically overridden and no error will be thrown

in this case. If no section option is passed, then the default section name

("prog") will be assumed, otherwise the provided section name will be used. If no

verbose option is passed, then a verifier log will only be dumped on load error.

See also EXAMPLES section for usage examples.

section NAME - Specifies a section name that contains the BPF program code. If no

section name is specified, the default one ("prog") will be used. This option is to

be passed with the object option.

verbose - Act in verbose mode. For example, even in case of success, this will

print the verifier log in case a program was loaded from a BPF ELF file.

pinned FILE - Attaches a XDP/BPF program to the given device. The FILE points to an

already pinned BPF program in the BPF file system. The option section doesn't apply

here, but otherwise semantics are the same as with the option object described al?

ready.

master DEVICE

set master device of the device (enslave device).

nomaster

unset master device of the device (release device).

addrgenmode eui64|none|stable_secret|random

set the IPv6 address generation mode

eui64 - use a Modified EUI-64 format interface identifier

none - disable automatic address generation

stable_secret - generate the interface identifier based on a preset

/proc/sys/net/ipv6/conf/{default,DEVICE}/stable_secret

random - like stable_secret, but auto-generate a new random secret if none is set

link-netnsid

set peer netnsid for a cross-netns interface

type ETYPE TYPE_ARGS

Change type-specific settings. For a list of supported types and arguments refer to

the description of ip link add above. In addition to that, it is possible to manip?

ulate settings to slave devices:

Bridge Slave Support

For a link with master bridge the following additional arguments are supported:

ip link set type bridge_slave [fdb_flush] [state STATE] [priority PRIO] [
cost COST] [guard { on | off }] [hairpin { on | off }] [fastleave { on | off
}] [root_block { on | off }] [learning { on | off }] [flood { on | off }] [
proxy_arp { on | off }] [proxy_arp_wifi { on | off }] [mcast_router MULTI?
CAST_ROUTER] [mcast_fast_leave { on | off}] [mcast_flood { on | off }] [
mcast_to_unicast { on | off }] [group_fwd_mask MASK] [neigh_suppress { on | off
}] [vlan_tunnel { on | off }] [isolated { on | off }] [backup_port DEVICE] [
nobackup_port]

fdb_flush - flush bridge slave's fdb dynamic entries.

state STATE - Set port state. STATE is a number representing the following

states: 0 (disabled), 1 (listening), 2 (learning), 3 (forwarding), 4

(blocking).

priority PRIO - set port priority (allowed values are between 0 and 63, in?

clusively).

cost COST - set port cost (allowed values are between 1 and 65535, inclu?

sively).

guard { on | off } - block incoming BPDU packets on this port.

hairpin { on | off } - enable hairpin mode on this port. This will allow

incoming packets on this port to be reflected back.

fastleave { on | off } - enable multicast fast leave on this port.

root_block { on | off } - block this port from becoming the bridge's root

port.

learning { on | off } - allow MAC address learning on this port.

flood { on | off } - open the flood gates on this port, i.e. forward all

unicast frames to this port also. Requires proxy_arp and proxy_arp_wifi to

be turned off.

proxy_arp { on | off } - enable proxy ARP on this port.

proxy_arp_wifi { on | off } - enable proxy ARP on this port which meets ex?

tended requirements by IEEE 802.11 and Hotspot 2.0 specifications.

mcast_router MULTICAST_ROUTER - configure this port for having multicast

routers attached. A port with a multicast router will receive all multicast

traffic. MULTICAST_ROUTER may be either 0 to disable multicast routers on

this port, 1 to let the system detect the presence of routers (this is the

default), 2 to permanently enable multicast traffic forwarding on this port
or 3 to enable multicast routers temporarily on this port, not depending on
incoming queries.

mcast_fast_leave { on | off } - this is a synonym to the fastleave option
above.

mcast_flood { on | off } - controls whether a given port will flood multi?
cast traffic for which
  there is no MDB entry.

mcast_to_unicast { on | off } - controls whether a given port will repli?
cate packets using unicast
  instead of multicast. By default this flag is off.

group_fwd_mask MASK - set the group forward mask. This is the bitmask that
is applied to decide whether to forward incoming frames destined to link-
local addresses, ie addresses of the form 01:80:C2:00:00:0X (defaults to 0,
ie the bridge does not forward any link-local frames coming on this port).

neigh_suppress { on | off } - controls whether neigh discovery (arp and nd)
proxy and suppression is enabled on the port. By default this flag is off.

vlan_tunnel { on | off } - controls whether vlan to tunnel mapping is en?
abled on the port. By default this flag is off.

backup_port DEVICE - if the port loses carrier all traffic will be redi?
rected to the configured backup port

nobackup_port - removes the currently configured backup port

Bonding Slave Support

For a link with master bond the following additional arguments are supported:

ip link set type bond_slave [queue_id ID]

queue_id ID - set the slave's queue ID (a 16bit unsigned value).

MACVLAN and MACVTAP Support

Modify list of allowed macaddr for link in source mode.

ip link set type { macvlan | macvap } [macaddr COMMAND MACADDR ...]

Commands:

add - add MACADDR to allowed list

set - replace allowed list

del - remove MACADDR from allowed list

flush - flush whole allowed list

Update the broadcast/multicast queue length.

ip link set type { macvlan | macvap } [bcqueuelen LENGTH]

bcqueuelen LENGTH - Set the length of the RX queue used to process broad‐
cast and multicast packets. LENGTH must be a positive integer in the range
[0-4294967295]. Setting a length of 0 will effectively drop all broad‐
cast/multicast traffic. If not specified the macvlan driver default (1000)
is used. Note that all macvlans that share the same underlying device are
using the same queue. The parameter here is a request, the actual queue
length used will be the maximum length that any macvlan interface has re‐
quested. When listing device parameters both the bcqueuelen parameter as
well as the actual used bcqueuelen are listed to better help the user un‐
derstand the setting.

ip link show - display device attributes

dev NAME (default)

NAME specifies the network device to show.

group GROUP

GROUP specifies what group of devices to show.

up   only display running interfaces.

master DEVICE

DEVICE specifies the master device which enslaves devices to show.

vrf NAME

NAME specifies the VRF which enslaves devices to show.

type TYPE

TYPE specifies the type of devices to show.

Note that the type name is not checked against the list of supported types - in‐
stead it is sent as-is to the kernel. Later it is used to filter the returned in‐
terface list by comparing it with the relevant attribute in case the kernel didn't
filter already. Therefore any string is accepted, but may lead to empty output.

ip link xstats - display extended statistics

type TYPE

TYPE specifies the type of devices to display extended statistics for.

ip link afstats - display address-family specific statistics

dev DEVICE

   DEVICE specifies the device to display address-family statistics for.

 ip link help - display help

   TYPE specifies which help of link type to display.

 GROUP

   may be a number or a string from the file /etc/iproute2/group which can be manually

   filled.

EXAMPLES

   ip link show

      Shows the state of all network interfaces on the system.

   ip link show type bridge

      Shows the bridge devices.

   ip link show type vlan

      Shows the vlan devices.

   ip link show master br0

      Shows devices enslaved by br0

   ip link set dev ppp0 mtu 1400

      Change the MTU the ppp0 device.

   ip link add link eth0 name eth0.10 type vlan id 10

      Creates a new vlan device eth0.10 on device eth0.

   ip link delete dev eth0.10

      Removes vlan device.

   ip link help gre

      Display help for the gre link type.

   ip link add name tun1 type ipip remote 192.168.1.1 local 192.168.1.2 ttl 225 encap gue en?

   cap-sport auto encap-dport 5555 encap-csum encap-remcsum

      Creates an IPIP that is encapsulated with Generic UDP Encapsulation, and the outer UDP

      checksum and remote checksum offload are enabled.

   ip link set dev eth0 xdp obj prog.o

      Attaches a XDP/BPF program to device eth0, where the program is located in prog.o,

      section "prog" (default section). In case a XDP/BPF program is already attached, throw

      an error.

   ip -force link set dev eth0 xdp obj prog.o sec foo

Attaches a XDP/BPF program to device eth0, where the program is located in prog.o,

section "foo". In case a XDP/BPF program is already attached, it will be overridden by

the new one.

ip -force link set dev eth0 xdp pinned /sys/fs/bpf/foo

Attaches a XDP/BPF program to device eth0, where the program was previously pinned as

an object node into BPF file system under name foo.

ip link set dev eth0 xdp off

If a XDP/BPF program is attached on device eth0, detach it and effectively turn off

XDP for device eth0.

ip link add link wpan0 lowpan0 type lowpan

Creates a 6LoWPAN interface named lowpan0 on the underlying IEEE 802.15.4 device

wpan0.

ip link add dev ip6erspan11 type ip6erspan seq key 102 local fc00:100::2 remote

fc00:100::1 erspan_ver 2 erspan_dir ingress erspan_hwid 17

Creates a IP6ERSPAN version 2 interface named ip6erspan00.

SEE ALSO

ip(8), ip-netns(8), ethtool(8), iptables(8)

AUTHOR

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

iproute2   13 Dec 2012   IP-LINK(8)