ip (iproute2) Linux networking

START HERE

`ip -br a`
Addresses at a glance

`ip neigh`
ARP / NDP cache

`ip -br link`
Interfaces and state

`ip link set <dev> up`
Bring link up

`ip route`
Routing table

`ip addr add <cidr> dev <dev>`
Add an address

`ip route get <addr>`
Which route is used?

`ip -c -j addr`
Colour / JSON output

CONCEPTS

ip is OBJECT COMMAND: objects are link, addr, route, neigh, netns. Any word can be abbreviated (ip a, ip r, ip l).

Network namespace own devices, routes, rules
- **Link (interface)** MAC, MTU, up/down
  - **Address** IPv4 / IPv6 on a link
    - **Route** which link and gateway for a prefix

What lives where

Changes made with ip are not persistent: they vanish on reboot. Use NetworkManager, systemd-networkd or netplan for permanent config.

Changing addresses or routes needs root. The show commands do not.

OPTIONS

`ip -br a`
Brief one-line-per-interface view

`ip -c <cmd>`
Colour output

`ip -4 <cmd>` / `ip -6 <cmd>`
Only IPv4 / only IPv6

`ip -j -p <cmd>`
JSON output, pretty-printed
Pipe -j into jq for scripting.

`ip -s link show <dev>`
Packet and error counters

`ip -d link show <dev>`
Show link type details

`ip -n <ns> <cmd>`
Run in a network namespace

`ip -o addr`
One record per line, for grep/awk

ADDRESSES

`ip addr`
Show all addresses

`ip addr show dev <dev>`
Addresses of one interface

ADDRESSES (cont.)

`ip addr add <cidr> dev <dev>`
Add an address
Example: 192.168.1.10/24. Several addresses per link are fine.

`ip addr del <cidr> dev <dev>`
Remove an address

`ip addr flush dev <dev>`
Remove all addresses of a link

`ip -4 addr show scope global`
Only global IPv4 addresses

`ip addr replace <cidr> dev <dev>`
Add or update an address

LINKS

`ip link`
List interfaces

`ip link set <dev> up`
Enable an interface

`ip link set <dev> down`
Disable an interface

`ip link set <dev> mtu <n>`
Set the MTU

`ip link set <dev> name <new>`
Rename (link must be down)

`ip link set <dev> address <mac>`
Change the MAC address

`ip link set <dev> promisc on`
Promiscuous mode

`ip link add <name> type <type>`
Create a virtual link
Types include bridge, dummy, veth, vlan, bond.

`ip link add <name> type veth peer name <peer>`
Create a veth pair

`ip link add link <dev> name <dev>.<id> type vlan id <id>`
Create a VLAN

`ip link set <dev> master <bridge>`
Attach to a bridge or bond

LINKS (cont.)

`ip link del <dev>`
Delete a virtual link

ROUTES

`ip route`
Show the main routing table

`ip -6 route`
Show IPv6 routes

`ip route get <addr>`
Show the route a destination would use

`ip route add default via <gw> dev <dev>`
Set the default gateway

`ip route add <cidr> via <gw>`
Add a static route

`ip route add <cidr> dev <dev>`
Add an on-link route

`ip route del <cidr>`
Delete a route

`ip route replace <cidr> via <gw>`
Add or overwrite a route

`ip route flush dev <dev>`
Remove all routes via a device

`ip route show table <id>`
Show a non-main table

`ip rule`
List policy routing rules

`ip rule add from <cidr> table <id>`
Route by source address

NEIGHBOURS

`ip neigh`
Show the ARP / NDP cache

`ip neigh show dev <dev>`
Neighbours on one interface

`ip neigh add <addr> lladdr <mac> dev <dev> nud permanent`
Add a static entry

`ip neigh del <addr> dev <dev>`
Remove an entry

`ip neigh flush dev <dev>`
Clear the cache of a link

NAMESPACES

`ip netns`
List network namespaces

`ip netns add <ns>`
Create a namespace

`ip netns exec <ns> <cmd>`
Run a command inside

`ip netns del <ns>`
Delete a namespace

`ip link set <dev> netns <ns>`
Move a link into a namespace

`ip netns identify <pid>`
Namespace of a process

`ip monitor`
Watch link, address and route changes

RECIPES

Static address and gateway

```
ip link set eth0 up
ip addr add 192.168.1.10/24 dev eth0
ip route add default via 192.168.1.1
ip route get 1.1.1.1
```

Two namespaces joined by a veth

```
ip netns add a; ip netns add b
ip link add va type veth peer name vb
ip link set va netns a; ip link set vb netns b
ip -n a addr add 10.0.0.1/24 dev va
ip -n b addr add 10.0.0.2/24 dev vb
ip -n a link set va up; ip -n b link set vb up
ip netns exec a ping -c1 10.0.0.2
```

Primary IPv4 address in a script

```
ip -j -4 route get 1.1.1.1 | jq -r '.[0].prefsrc'
```

Bridge with a port

```
ip link add br0 type bridge
ip link set eth1 master br0
ip link set br0 up
```