

Pyroute2: a transactional approach to the network setup

<http://pyroute2.org/rtnl.pdf>

<http://docs.pyroute2.org/>

<https://github.com/svinota/pyroute2/>

network-related API towards the kernel:

- * ioctl — ifconfig
- * setsockopt, getsockopt — iptables
- * file operations — sysfs, procfs
- * AF_KEY — ipsec-tools
- * AF_NETLINK — iproute2, ...
- * ...

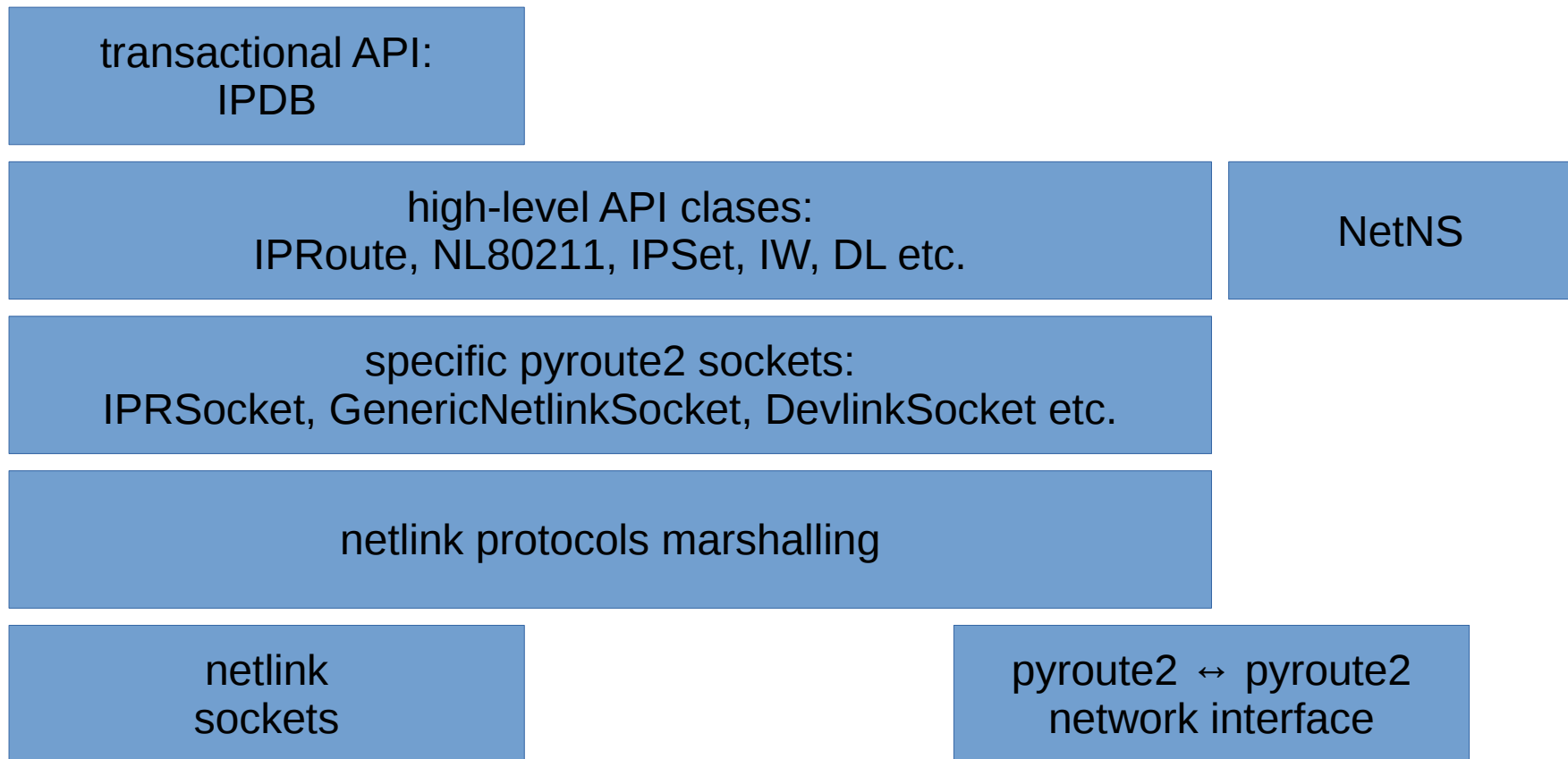
netlink families:

- * NETLINK_XFRM — ipsec
- * NETLINK_NFLOG — netfilter/iptables ULOG
- * NETLINK_ISCSI
- * NETLINK_GENERIC — nl80211, taskstats, events, ...
- * NETLINK_ROUTE — RTNL
- * ...

API

```
s = socket(AF_NETLINK, SOCK_DGRAM, NETLINK_ROUTE, ...)  
s.bind(...) # optional, only to receive notifications  
  
s.send(...)  
  
s.recv(...)
```

the library architecture



netlink packet representation

```

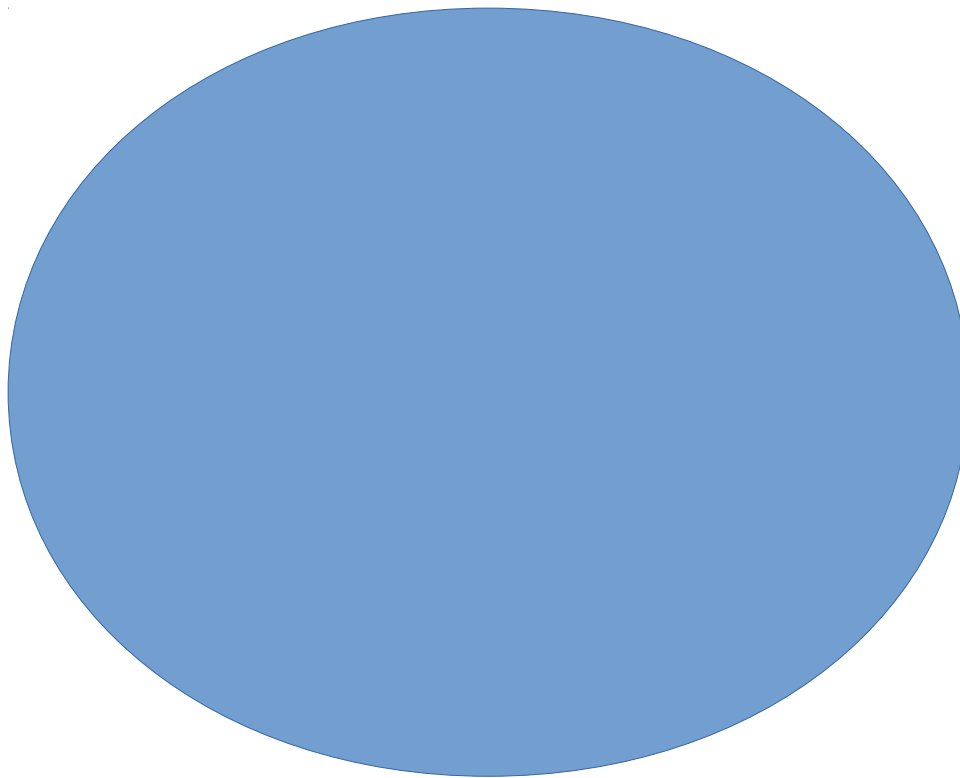
{'header': {'error': None,
            'flags': 2,
            'length': 1260,
            'pid': 9479,           # event initiator's pid
            'sequence_number': 256, # answer to packet id 256
            'type': 16},         # event type RTM_NEWLINK
  'change': 0,
  'family': 0,
# ...
  'index': 2,
  'attrs': [('IFLA_IFNAME', 'enp0s31f6'),
            ('IFLA_TXQLEN', 1000),
            ('IFLA_OPERSTATE', 'UP'),
            ('IFLA_MTU', 1500),
            ('IFLA_GROUP', 0),
# ...
            ('IFLA_ADDRESS', 'c8:5b:67:58:a1:b7'),
            ('IFLA_BROADCAST', 'ff:ff:ff:ff:ff:ff')]]}

```

IPDB object representation

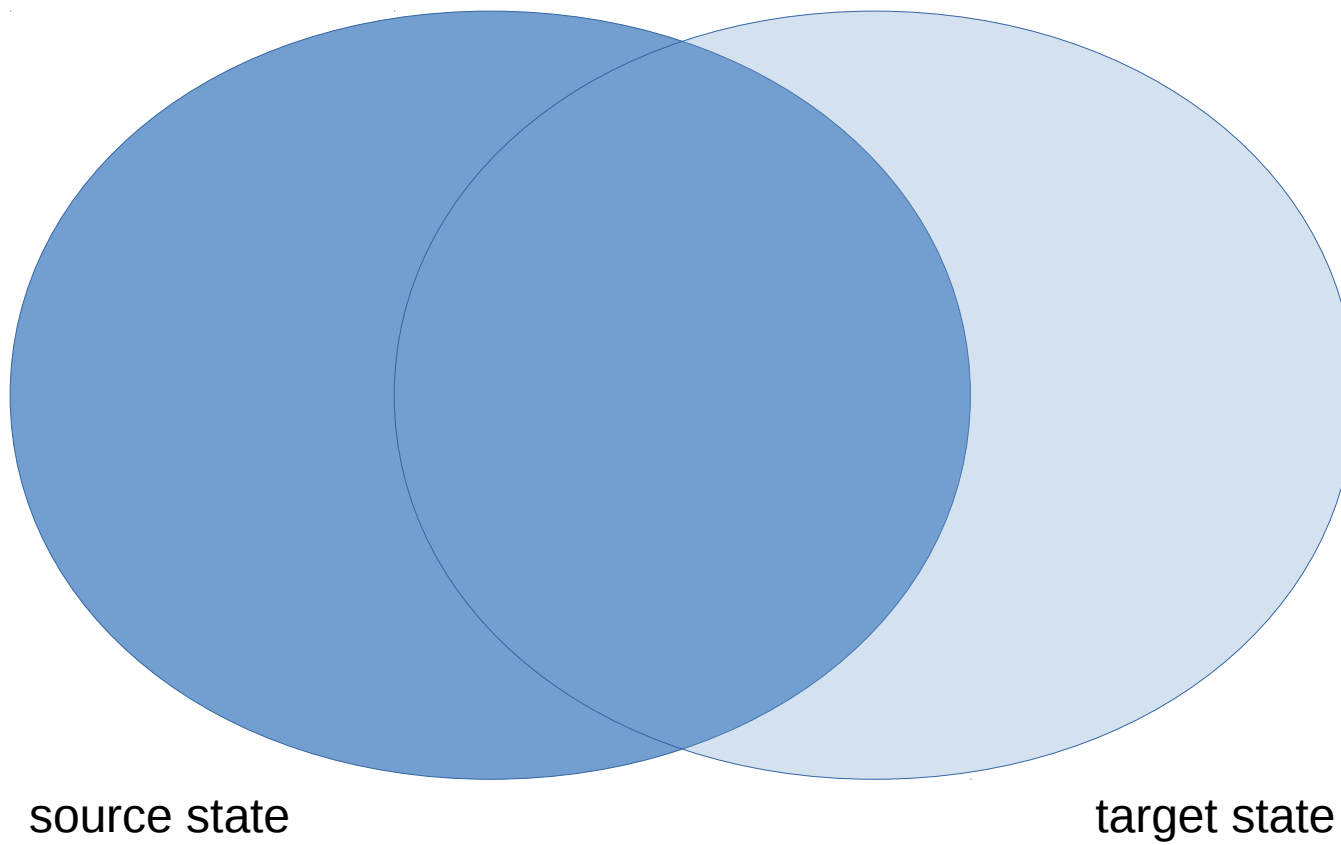
```
{'index': 4,  
  'ipaddr': (('172.16.254.105', 24),  
             ('10.0.1.12', 24),  
             ('10.0.1.13', 24),  
             ('fe80::7552:c31a:60c6:a427', 64)),  
  'ifname': 'eth0',  
  'broadcast': 'ff:ff:ff:ff:ff:ff',  
  'address': 'c8:5b:76:c8:ff:6e',  
  'mtu': 1500,  
  'flags': 4099}
```

State transition as set operations

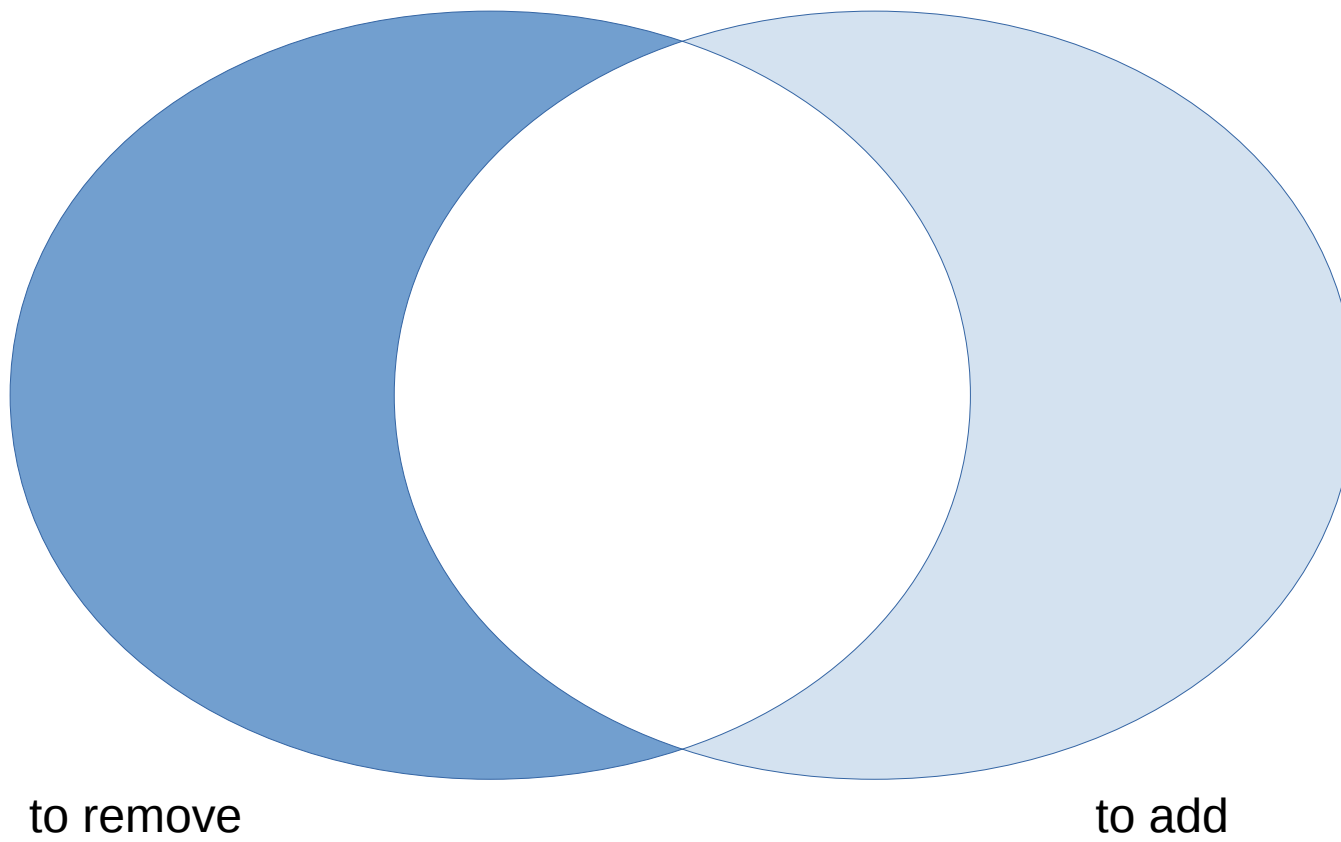


source state

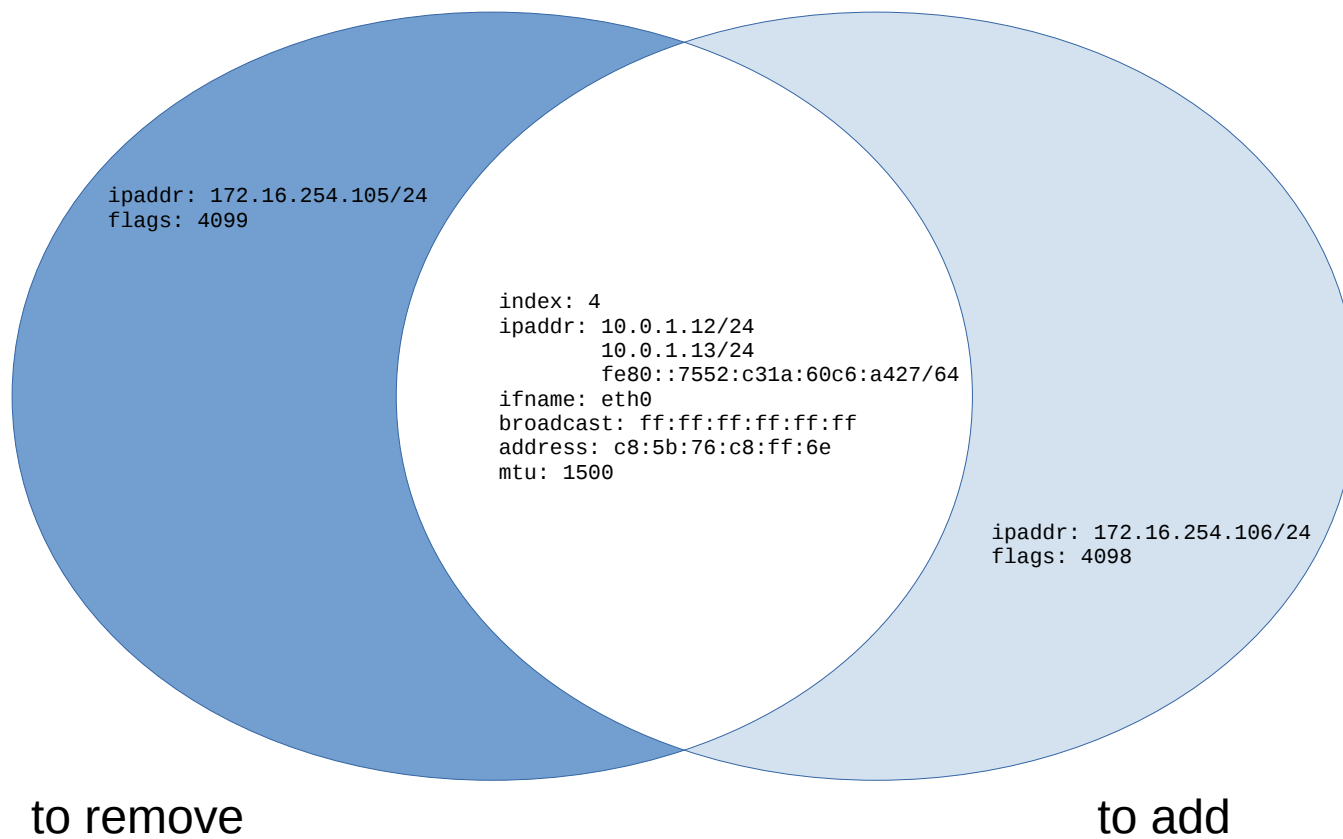
State transition as set operations



State transition as set operations



State transition as set operations



IPDB transaction

initial state

```
{'index': 4,  
 'ipaddr': (('172.16.254.105', 24),  
           ('10.0.1.12', 24),  
           ('10.0.1.13', 24),  
           ('fe80::7552:c31a:60c6:a427', 64)),  
 'ifname': 'eth0',  
 'broadcast': 'ff:ff:ff:ff:ff:ff',  
 'address': 'c8:5b:76:c8:ff:6e',  
 'mtu': 1500,  
 'flags': 4099}
```

added values

```
{'ipaddr': (('172.16.254.106', 24), ),  
 'flags': 4098}
```

target state

```
{'index': 4,  
 'ipaddr': (('172.16.254.106', 24),  
           ('10.0.1.12', 24),  
           ('10.0.1.13', 24),  
           ('fe80::7552:c31a:60c6:a427', 64)),  
 'ifname': 'eth0',  
 'broadcast': 'ff:ff:ff:ff:ff:ff',  
 'address': 'c8:5b:76:c8:ff:6e',  
 'mtu': 1500,  
 'flags': 4098}
```

removed values

```
{'ipaddr': (('172.16.254.105', 24), ),  
 'flags': 4099}
```

RTNL issues

- * **events order**: e.g., one can get RTM_NEWLINK after the bridge removal
- * **notifications**: dummy interfaces in the DOWN state send IPv6 addr notifications, while bridges don't
- * **no notifications**: there is no notifications for traffic controls, like queues, classes etc.
- * **structure declarations**: dynamic type resolution in the kernel via union
- * **compatibility**: there is a lot of the legacy code in the kernel netlink code, e.g. HTB / TBF speed map
- * ...

pyroute2 is fixing things

IPDB level

- * IPv4 vs. IPv6 notifications workarounds
- * fuzzy logic in the transaction target matching
- * retry some changes until completed
- * context-dependent message handling
- * object dependencies

socket level

- * automatically use sysfs to manage bridge and bonds on the old kernels
- * manage tuntap interfaces via ioctl

Pyroute2: a transactional approach to the network setup

<http://pyroute2.org/rtnl.pdf>

<http://docs.pyroute2.org/>

<https://github.com/svinota/pyroute2>