Netfilter HW offloads

Pablo Neira Ayuso <pablo@netfilter.org>

Linux Plumbers 2019, Lisbon, Portugal

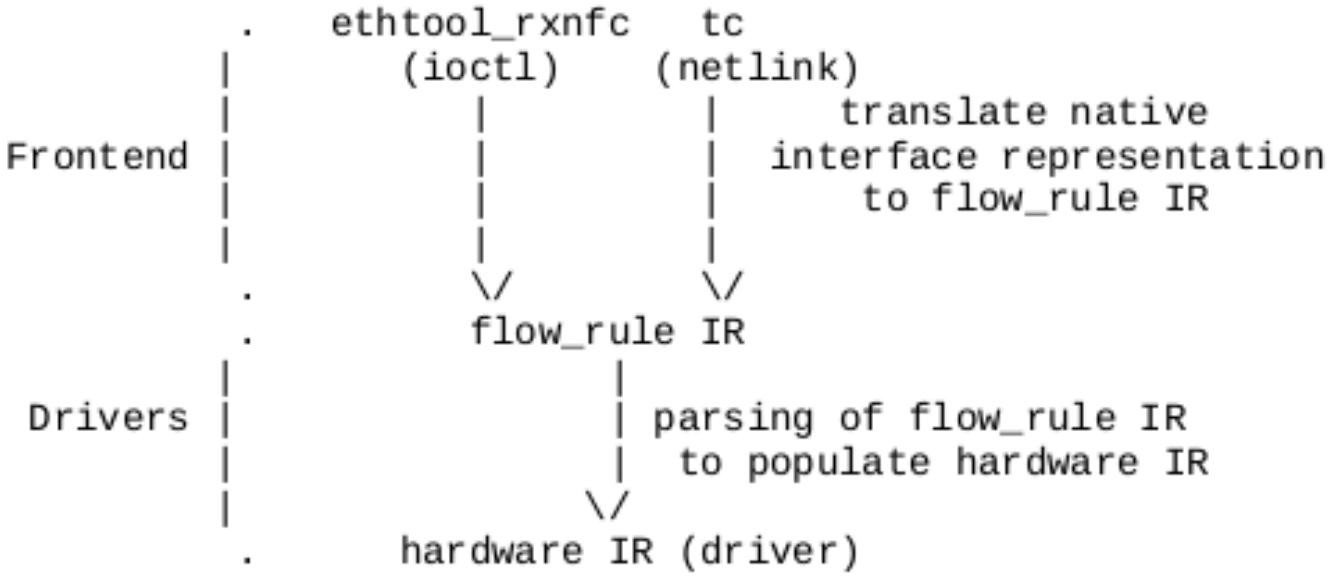
September 9th-11th

Intro

- Flow offload API
  - Policy HW offload
  - Ct flowtable bypass

Flow offload API

- Avoid duplicated driver code to configure offloads from ethtool and tc flower / clsall.



- Add netfilter ingress hook offload

Flow offload API

- Flow rule API based on:
  - cls_flower dissector
  - tc action API

```
struct flow_rule {
        struct flow_match       match;
        struct flow_action      action;
};
```

- Flow_block API

Flow rule API: match

- struct flow_match {
  ```
  struct flow_dissector   *dissector;
  void                    *mask;
  void                    *key;
  ```
  };
- enum flow_dissector_key_id definitions (in include/net/flow_dissector.h)
  - eg. FLOW_DISSECTOR_KEY_ETH_ADDRS

    ```
    struct flow_dissector_key_eth_addrs {
        unsigned char dst[ETH_ALEN];
        unsigned char src[ETH_ALEN];
    };
    ```
- struct flow_dissector {

  ```
  unsigned int used_keys;
  unsigned short int offset[FLOW_DISSECTOR_KEY_MAX];
  ```
  };

Flow rule API (2): actions

- struct flow_action {
  int num_entries;
  struct flow_action_entry *entries;
  };
- struct flow_action_entry {
  enum flow_action_entry_id id;
  union {
  …
  };
  };

Flow rule API (3): actions

- Accept / Drop: FLOW_ACTION_KEY_{ACCEPT,DROP}
- Redirect / mirror packet to netdev: FLOW_ACTION_KEY_{REDIRECT,MIRRED}
- VLAN encapsulation: FLOW_ACTION_KEY_VLAN_{PUSH,POP,MANGLE}
- payload mangling: FLOW_ACTION_KEY_{MANGLE,CSUM}
- Tunnel: FLOW_ACTION_KEY_TUNNEL_{ENCAP,DECAP}
- WOL: FLOW_ACTION_KEY_WAKE (ethtool)
- Packet steering: FLOW_ACTION_KEY_QUEUE (ethtool)
- ...

Flow rule API (4): helpers

- cls_flower → flow_rule API
  - match is native
  - tc_setup_flow_action(...)
    - tc action → flow_rule API
- ethtool_rx_flow_spec → flow_rule API
  - ethtool_rx_flow_rule_create(...)

Flow Rule API (5): summary

- Upstream since 5.3
- File:
  - include/net/flow_offload.h
  - net/core/flow_offload.c
- Drivers using this infrastructure:
  - Mellanox: mlx5, mlxsw (flower)
  - Broadcom: bnxt (flower), bcm_sf2 (ethtool)
  - Chelsio: cxgdb4 (flower)
  - Intel: i40eia, iavf, igb (flower)
  - Qlogic: qede (ethtool + flower)
  - Mscc: ocelot (flower)
  - Netronome: nfp (flower)

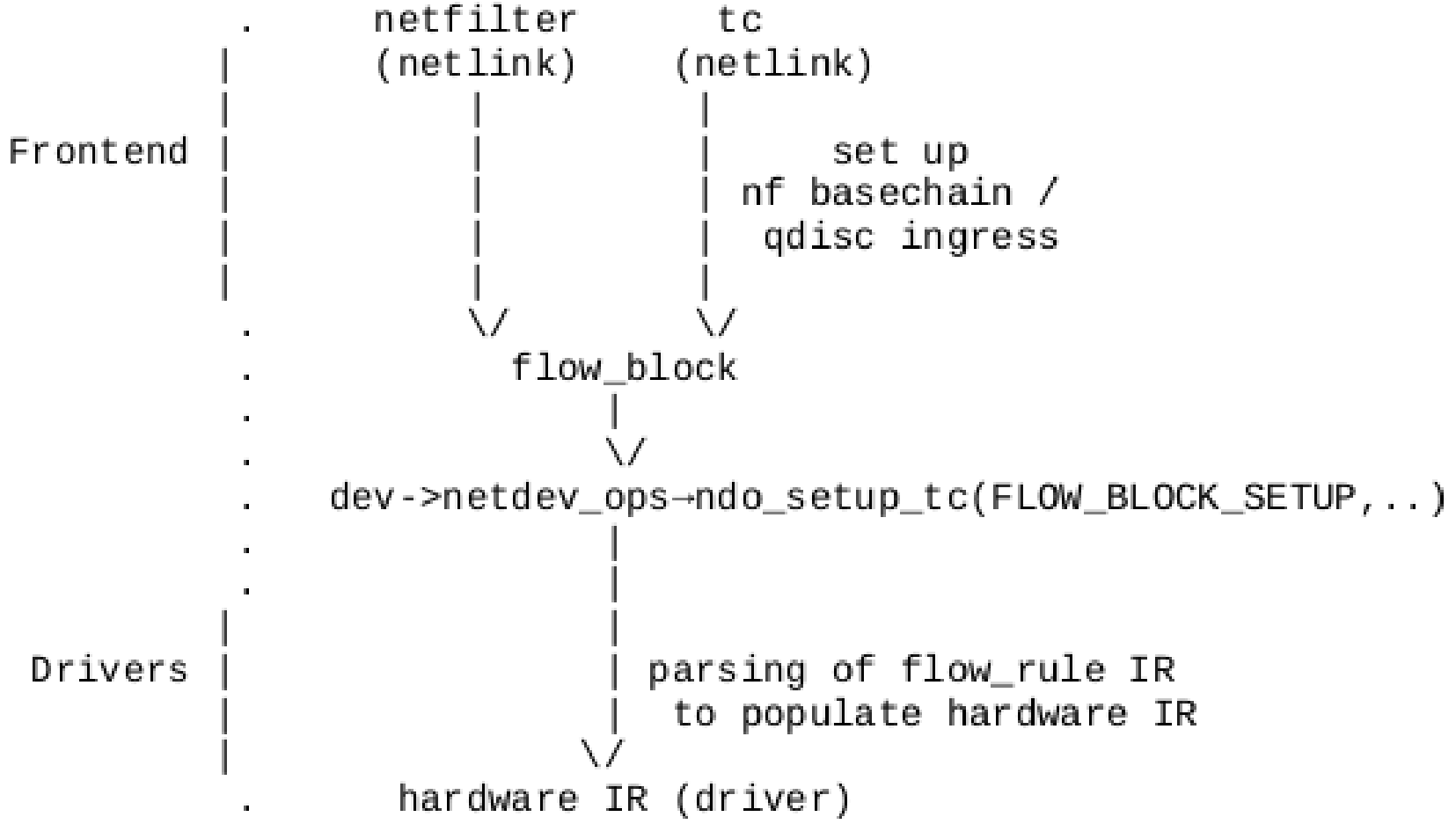
Flow block API

- share policy between several tc ingress “qdisc”
  - one tc block (with policy) ↔ multiple qdisc
- Block set up from front-end via ndo:
  - FLOW_BLOCK_BIND
  - FLOW_BLOCK_UNBIND
- On netfilter: one tc block ↔ one basechain
- Only one flow_block binding per subsystem at this stage (EBUSY)

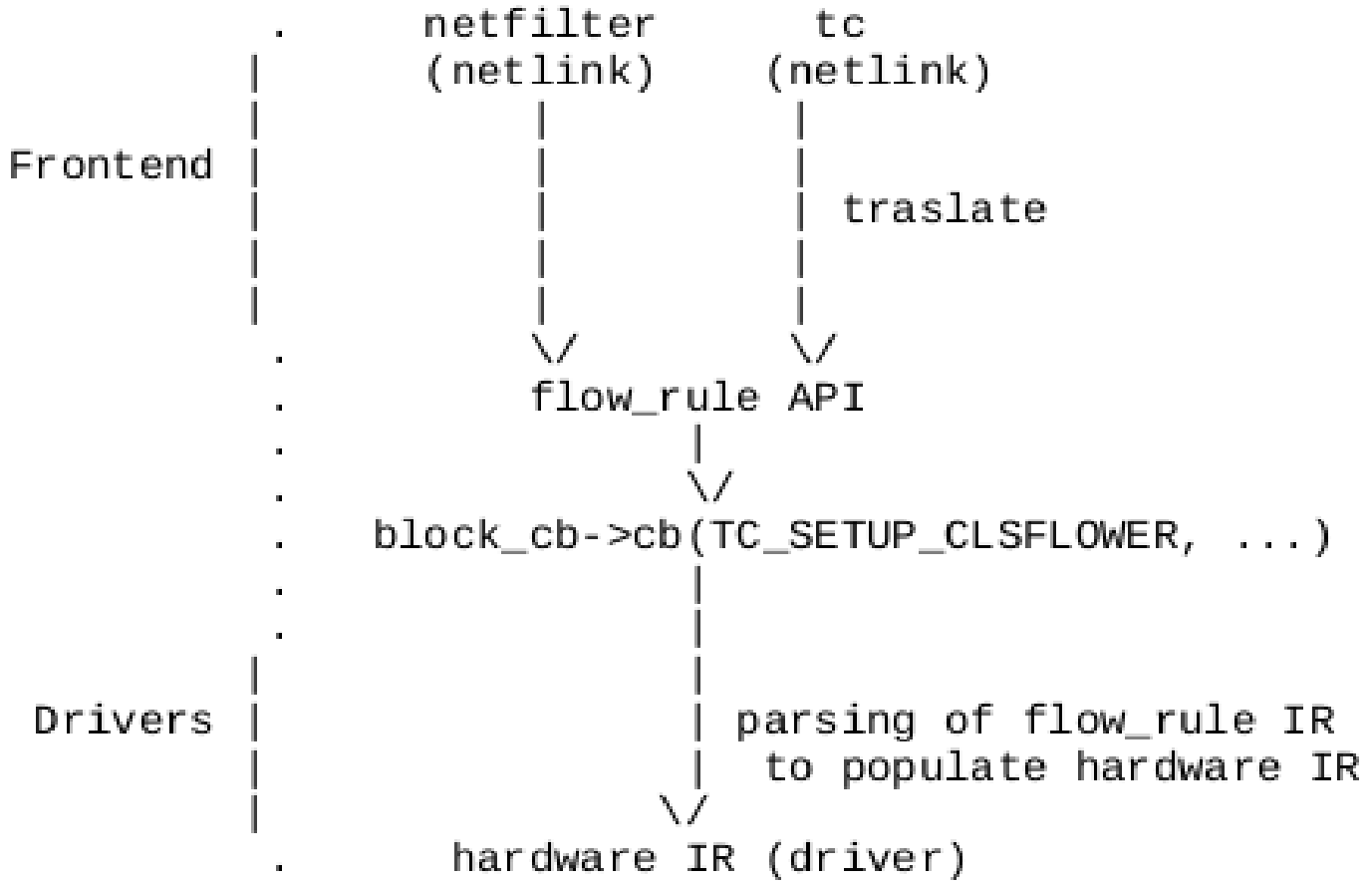
Flow block API (2)

- Hardware offload flag for netfilter basechain
  - NFT_CHAIN_HW_OFFLOAD
- Set up netfilter basechain
  - ndo_setup_tc(FLOW_BLOCK_SETUP, FLOW_BLOCK_BIND, ...)
- Add rules
  - block->cb(…, TC_SETUP_CLSFLOWER)
- Remove netfilter basechain
  - Delete rules
    - block->cb(…, TC_SETUP_CLSFLOWER)
  - ndo_setup_tc(FLOW_BLOCK_SETUP, FLOW_BLOCK_UNBIND, ...)

Flow block API (3)



Flow block API (4)



Netfilter through flow offload API

- Basechain:
  - ingress hook with flag offload set on
  - priorities from 1..65535
  - only accept default policy
- Payload matching (5.3)
- Accept / drop action (5.3)
- Netmask matching (5.4-rc)
- Fwd action (5.4-rc)
- Dup action (5.4-rc)

Netfilter offload: Payload mangling

- FLOW_ACTION_MANGLE uses tc pedit representation
- Offset alignment to 32-bits
- Drivers use mask to infer what part to mangle
  - eg. TCP sport (0xffff0000) or dport (0x0000ffff)
- Up to four actions to mangle an IPv6 address
- Patchset available:
  - Offset alignment to 8-bits
  - Adjust offset and length based on mask
- Problem? Allow to mangle only one byte of TCP port.

Ct flowtable bypass

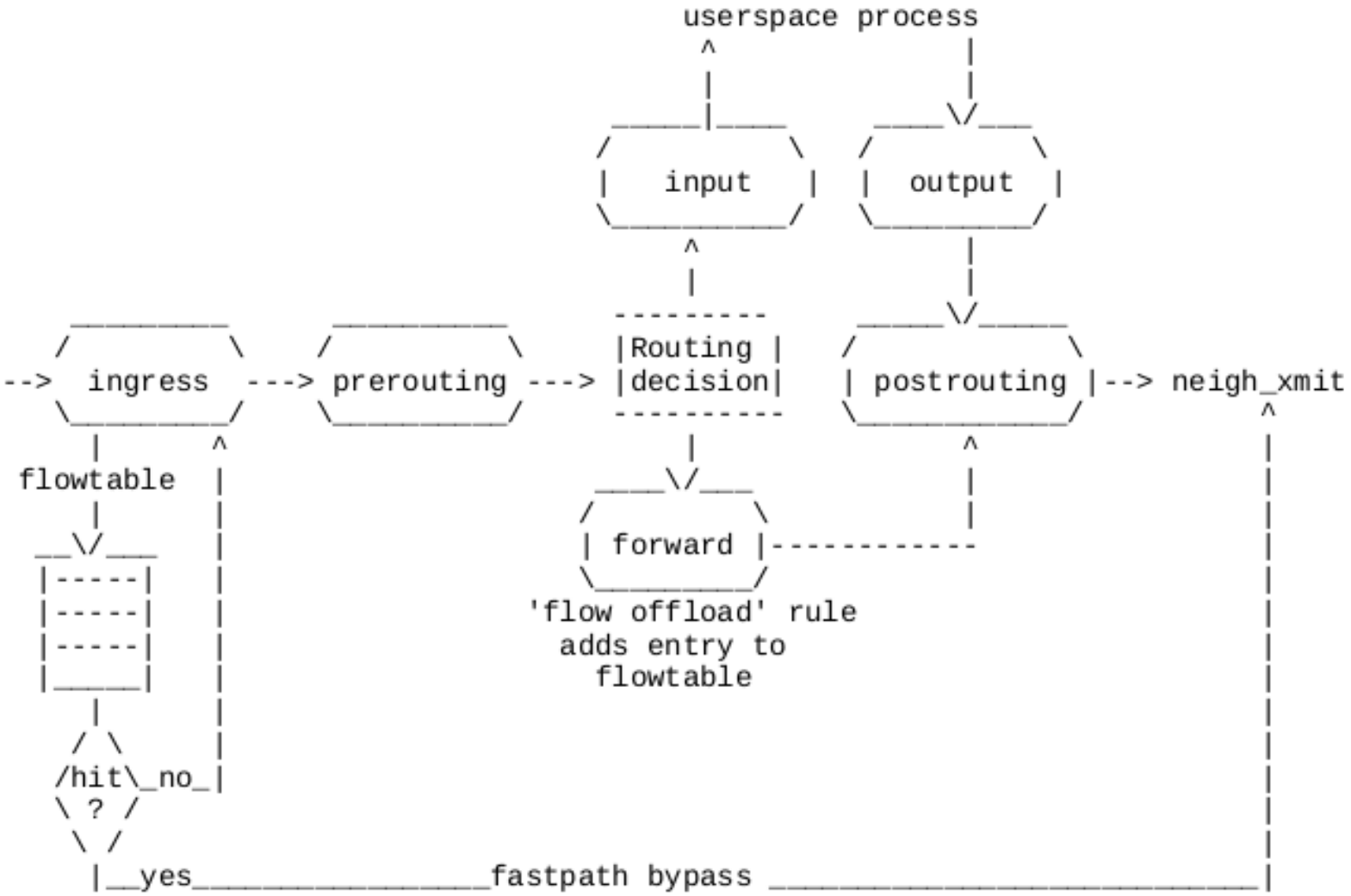


Fig.1 Netfilter hooks and flowtable interactions

Ct flowtable bypass (2)

- For each packet, extract tuple and perform look up at the flowtable.
  - Miss: packet follows the classic forwarding path.
  - Hit:
    - Attach route from flowtable entry (… flowtable acts as a cache).
    - NAT
    - Decrement TTL.
    - Send packet via neigh_xmit(...).
  - Exceptions (forces slow path): Packet over MTU / IP Options available.
- Tear down state
  - RST and FIN packets: send packet back to classic + pick up state
- Garbage collector expires that see no more packets after N seconds.
  - Back to conntrack, using pickup time in ESTABLISHED state

Ct flowtable bypass (3)

- Configure flow bypass through **one single rule**:

```
table ip x {
    flowtable f {
        hook ingress priority 0; devices = { eth0, eth1};
    }
    chain y {
        type filter hook forward priority 0;
        ip protocol tcp flow add @f
    }
}
```

- Conntrack entries are owned by the flowtable:
  # cat /proc/net/nf_conntrack
  ipv4 2 tcp 6 src=10.141.10.2 dst=147.75.205.195 sport=36392 dport=443 src=147.75.205.195 dst=192.168.2.195 sport=443 dport=36392 **[OFFLOAD]** mark=0 zone=0 use=2

Ct flowtable bypass (4): HW offload

- Add flow_block for ct flowtable
- Use flow_rule API
  - Represent 5-tuple matching via flow_match
  - Use flow_action redirect action
- Use workqueue to configure hw offload

Netfilter HW offloads

Pablo Neira Ayuso <pablo@netfilter.org>

Linux Plumbers 2019, Lisbon, Portugal