Creating a friendlier UAPI for XDP_REDIRECT

Toke Høiland-Jørgensen (Red Hat)
Jesper Dangaard Brouer (Red Hat)

Netconf
Boston, June 2019

The XDP_REDIRECT UAPI is not very friendly

- Two different helpers (`bpf_redirect()` and `bpf_redirect_map()`).
  - Using the _map variant doubles performance (why?! asks the user)
- Can't lookup into redirect maps
  - So people use "shadow" maps of a different type
- The helper calls always succeed
  - But may just silently drop packets

The XDP_REDIRECT UAPI is not very friendly (cont.)

- No way to know if an interface supports REDIRECT
- Need to load a dummy XDP program on TX iface (some drivers)
- No QoS or rate management, only small queue
  - Ever tried REDIRECT'ing from 100Gbit -> 10 Gbit?
- No packet duplication (i.e., no multicast / L2 broadcast)

Why is this a problem?

- Difficult to get config right
  - Config failures lead to silent drops
- No way to react dynamically to redirect failures
  - E.g., fall back to XDP_PASS

What can we do about it? (short term)

In progress:

- Allow lookups into maps (3-line patch, thanks to `BPF_F_RDONLY_PROG`!)
- Return error codes from `bpf_redirect_map()` helper on lookup failure
- Use hidden maps to improve performance of `bpf_redirect()`
  - Allows users to use the API (helper) that makes sense for use case
  - E.g., `bpf_redirect(bpf_fib_lookup())`
  - But is this really a good idea? See next slides...

TX resource allocation

Can we ensure successful lookup == successful xmit?

Two approaches:

1. Allocate resources first, then insert those into map
   - Separate (ethtool?) UAPI to allocate TX resources
   - Change devmap to accept a handle instead of ifindex
   - Separate API => potentially more flexible(?)
2. Trigger resource allocation on map insert
   - Call allocation `ndo` when iface inserted into devmap
   - Fail insert if allocation fails
   - Probably needs more descriptive map values(?) BTF?

TX resource allocation: challenges

What are the barriers to achieve this?

- Don't know on map insert if we're using generic XDP
  - Maybe use different map types?
- Supporting configs other than "1 TXQ/CPU"? Automatic locking?
- What about `bpf_redirect()` and XDP offload?

Queueing

- Expand queueing capabilities of XDP_REDIRECT
  - Select queueing scheme along with TX resource?
  - Probably not full qdisc, but at least a few options
  - Programmable, or just configurable?

Multicast

- Multicast through multi-send
  - Allow calling redirect helper multiple times?
  - Or new redirect target that selects multiple output "ports"?
  - Or `bpf_redirect_map` FLAG that send to ALL ports in `devmap`
    - Multicast is done by creating a map with ports in multicast group
  - Should program be allowed to modify packets in-between?
  - To copy or not to copy?

Introducing an XDP TX hook

Can we add a separate hook just before TX?

- Would know HW TX-ring occupancy
  - Can provide programmable back pressure to stack/redirect map
  - Maybe even redirect packets again if TX ring is full?
- Enables programmable QoS/queueing
- Symmetry with XDP RX hook (first/last thing that happens to pkt)
  - In particular, after regular stack processing
  - Programmable packet verdict (AQMs)