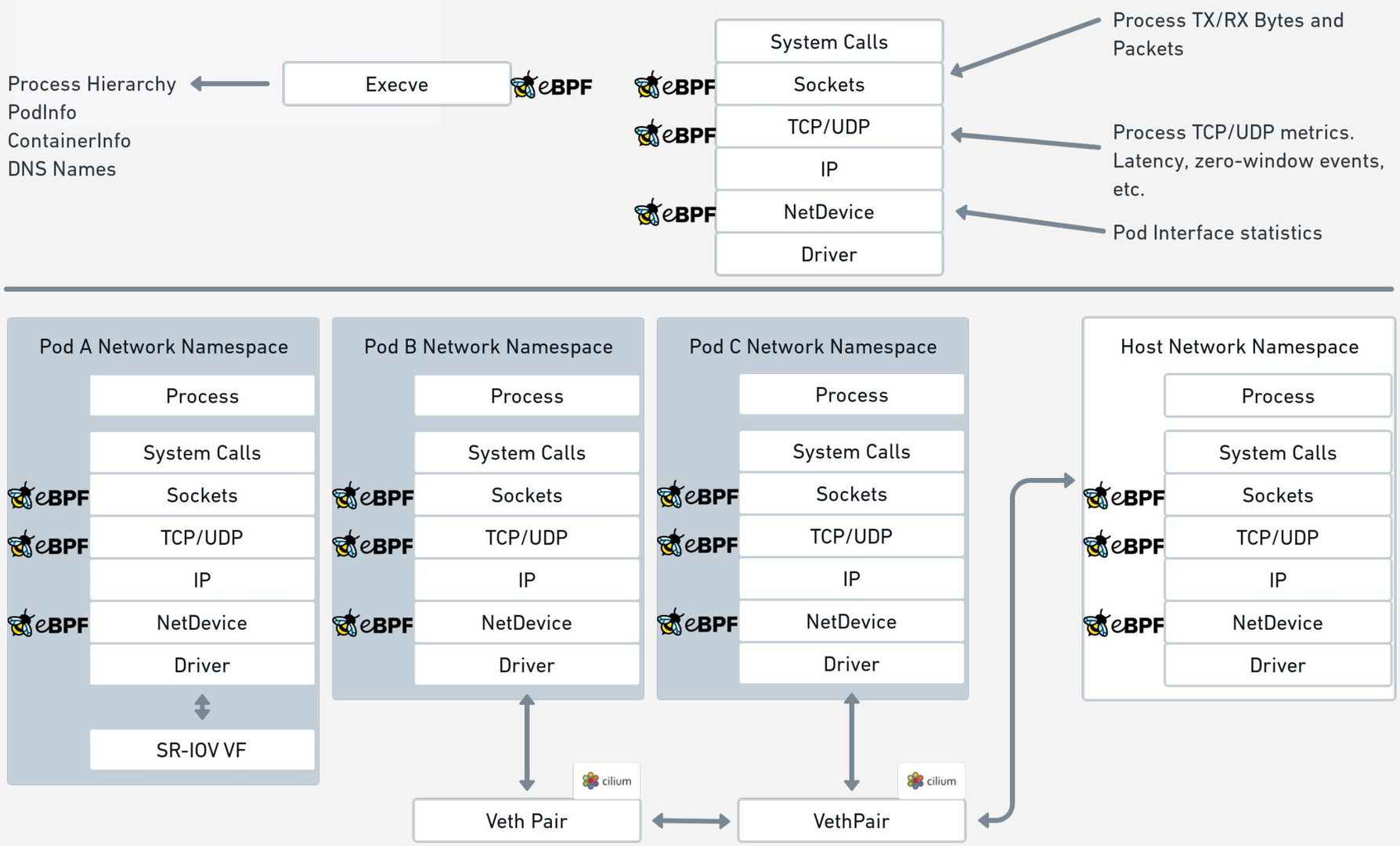
Problem: Observing Inside a Network Namespace

John Fastabend, Isovalent

Sorry only problems and
no code today.

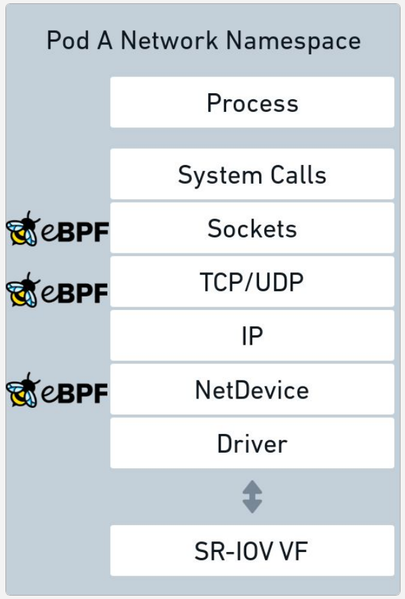


Traditional Networking Metrics:
- Host Network Namespace (miss container local traffic)
- 5-tuple (SIP, DIP, DPORT, SPORT, PROTO)
- Polling / Sampling
- May miss short lived sockets
- May miss short lived network artifacts (latency, bursts, etc)

Cilium Networking Metrics:
- Network Namespace Agnostic
- Process Identity (process, pod, container, 4-tuple, dstInfo, DNS)
- BPF Inline avoids polling/sampling missing
- BPF anomaly detection (RX burst, TX burst, etc.)
- Device agnostics SR-IOV VF/PF supported

Observing the Namespace

- Applications → SK_MSG, SK_SKB Programs
- Sockets → Sockops
- TCP/UDP → Kprobes/FEntry
- L3 → Ingress/Egress Hooks
- L2 → TC/XDP



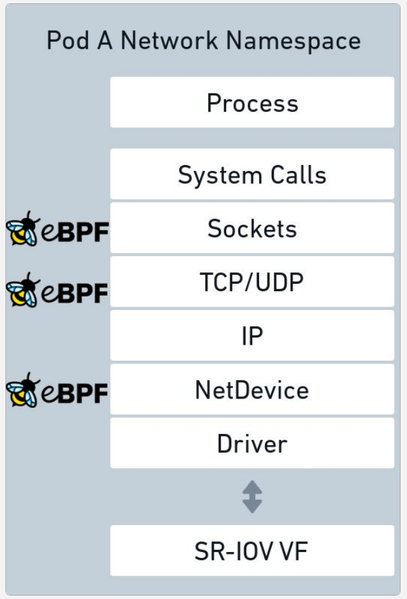
Observing the Namespace

L2 → TC/XDP

- TC is owned by Network Namespace
- XDP is owned by Network Namespace

- Kprobe/FEntry is slow per packet cost

- Polling the namespace/netdevice limited info



Observing the Namespace

L2 → TC/XDP

- TC is owned by Network Namespace
- XDP is owned by Network Namespace

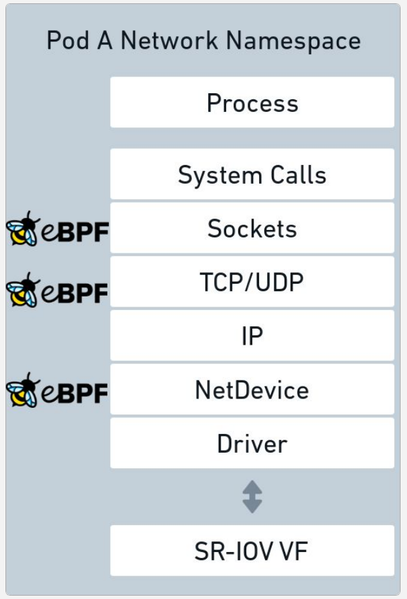
- Kprobe/FEntry is slow per packet cost

- Polling the namespace/netdevice limited info

Insecure and Efficient

Secure and Slow

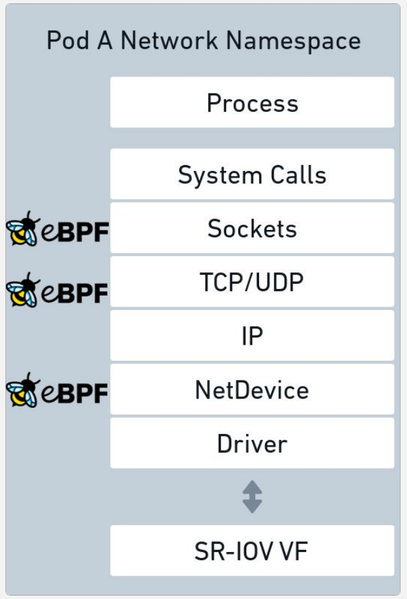
Efficient, Secure and polling timescales



Observing the Namespace

L2 → TC/XDP

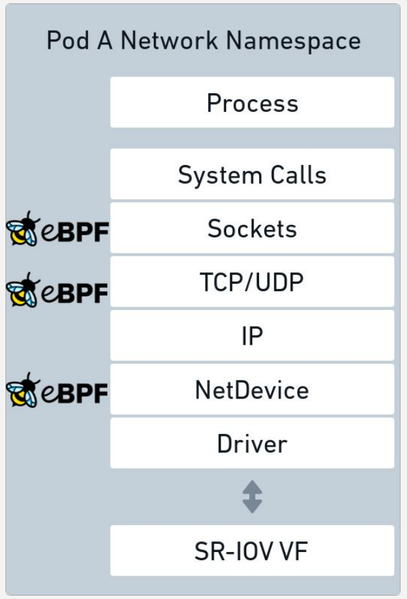
- What we want:
  - XDP Program applied to all net_devices in CGROUP
  - XDP Program “knows” net_device
  - Network namespace user does not own XDP
  - L3 ingress/egress hooks semantics at XDP layer



Observing the Namespace

L2 → TC/XDP

- What we want:
  - XDP Program applied to all net_devices
  - XDP Program “knows” net_device
  - Network namespace user does not own XDP
  - L3 ingress/egress hooks semantics at XDP layer

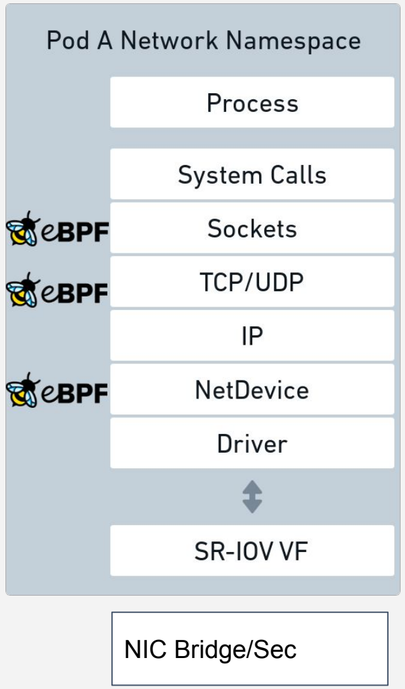


Efficient And Secure

Observing the Namespace

L2 → TC/XDP

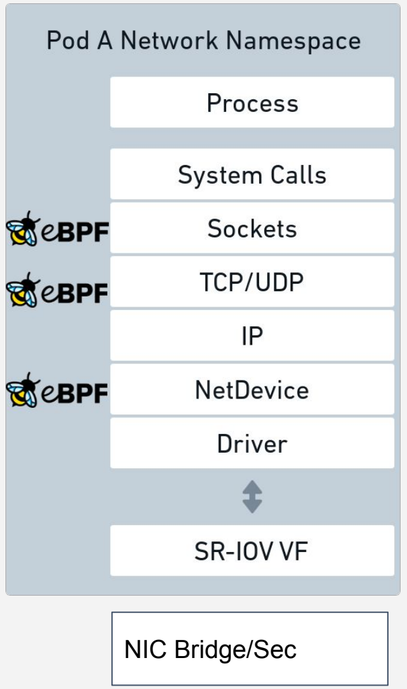
- NIC Bridge/Sec
  - Limited features (L2/L3/L4 filters)
  - Not GPU/programmable
  - Efficient but inflexible



Observing the Namespace

L2 → TC/XDP

- SRIOV XDP Requirements:
  - XDP Link so it can’t be removed
  - Kprobe BPF program can attach XDP program
  - Hook netdevice up
  - On init iterate network namespaces and netdevs with iterators and attach XDP programs.



Efficient And Secure

Observing the Namespace

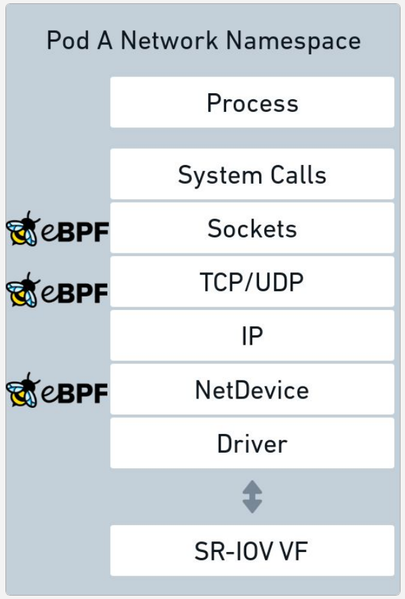
L2 → TC/XDP

- Kfunc: bpf_xdp_link_attach(struct net *,
- struct net_device *,
  bpf_prog *,
  u64 flags)

  flags := PINNED?

- Kfunc: bpf_devtx_link_attach(struct net_device *,
  bpf_prog *, u64 flags)

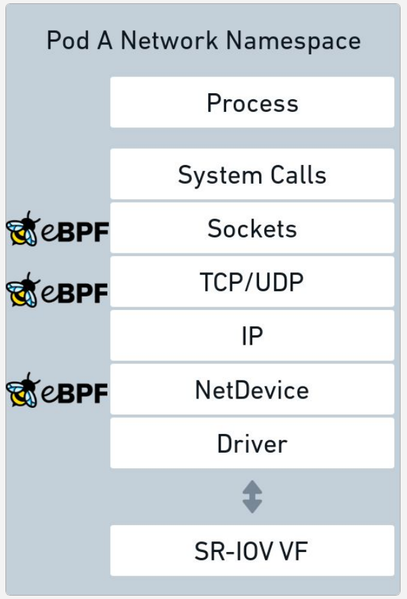
- netns/netdevice/progs globals and iterators



Observing the Namespace

L2 → TC/XDP

- What we want:
  - BPF Program applied to all net_devices in CGROUP
  - BPF Program above/after(?) Qdisc
  - Observability and Security hook not a Qdisc
  - BPF program can instantiate the program

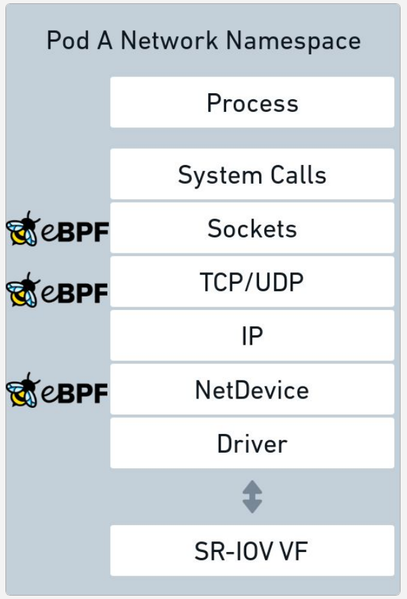


Efficient And Secure

Observing the Namespace

L2 → TC/XDP

- TC Proposal
  - Add a hook for BPF before Qdisc
  - BPF Link so it can’t be removed
  - Attach it to all netdevices



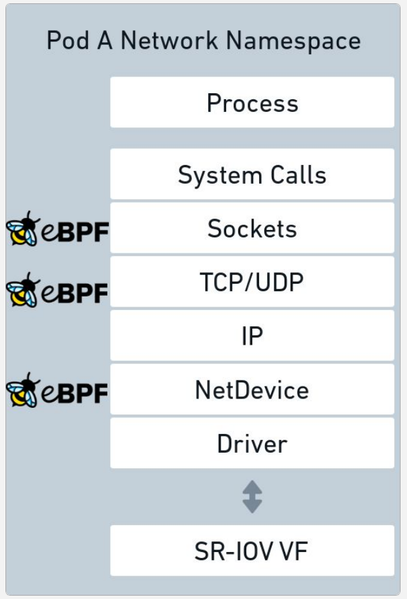
Efficient And Secure

Observing the Namespace

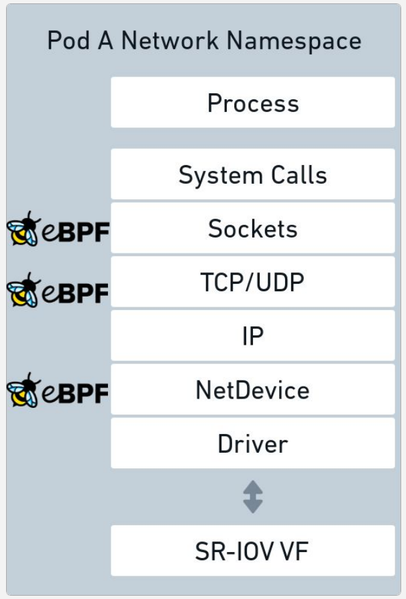
L2 → TC/XDP

- Kfunc: bpf_tcx_link_attach(struct net *,
  struct net_device *,
  bpf_prog *,
  u64 flags)

- netns/netdevice/progs globals and iterators



Thanks!



- BPF Programs to load XDP/TCX programs
- BPF globals to walk dev, netns, and progs from Fentry
- Pinned XDP/TCX to avoid removal