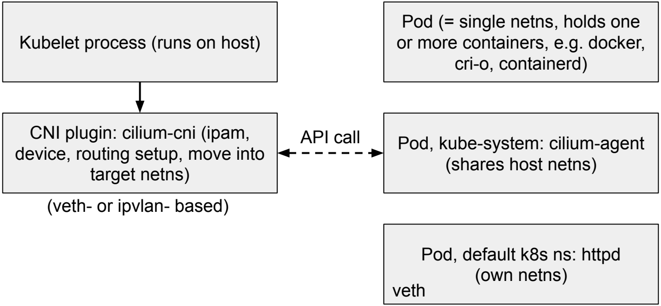
netconf: BPF mesh device, tc/BPF xfrm helpers.

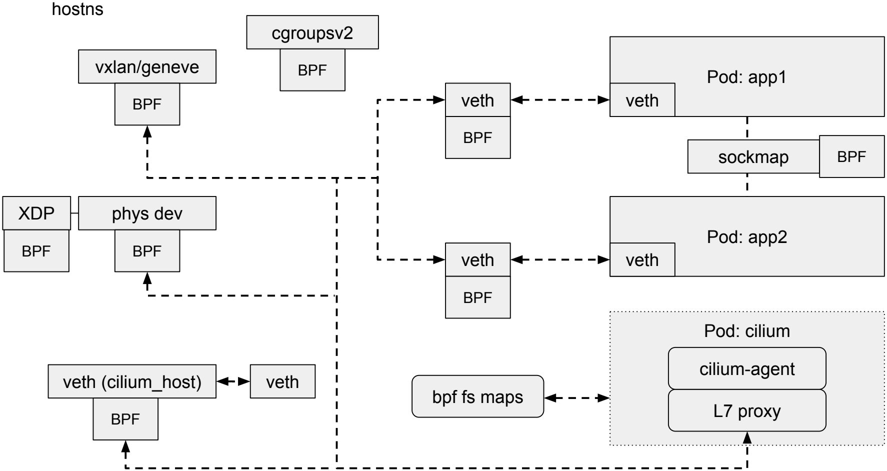
Daniel Borkmann
<daniel@cilium.io>
Cilium.io

netconf, June 22, 2019

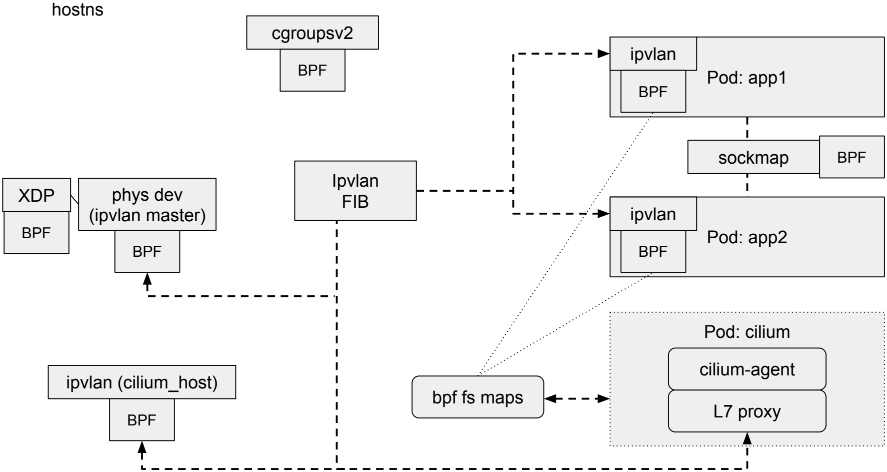
Cilium and Kubernetes: high level overview


Kubelet process (runs on host)
CNI plugin: cilium-cni (ipam, device, routing setup, move into target netns)
(veth- or ipvlan- based)
API call
Pod (= single netns, holds one or more containers, e.g. docker, cri-o, containerd)
Pod, kube-system: cilium-agent (shares host netns)
Pod, default k8s ns: httpd (own netns)
veth

Cilium veth-based data path



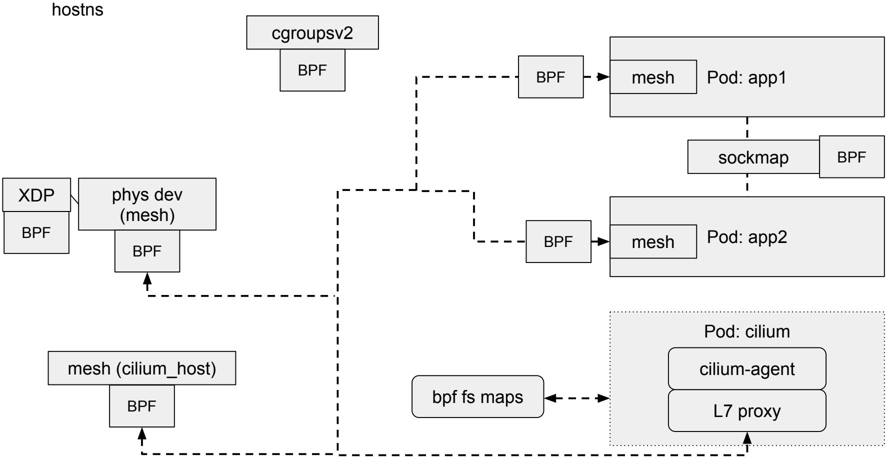
Cilium ipvlan-based data path



Issues with ipvlan mode:

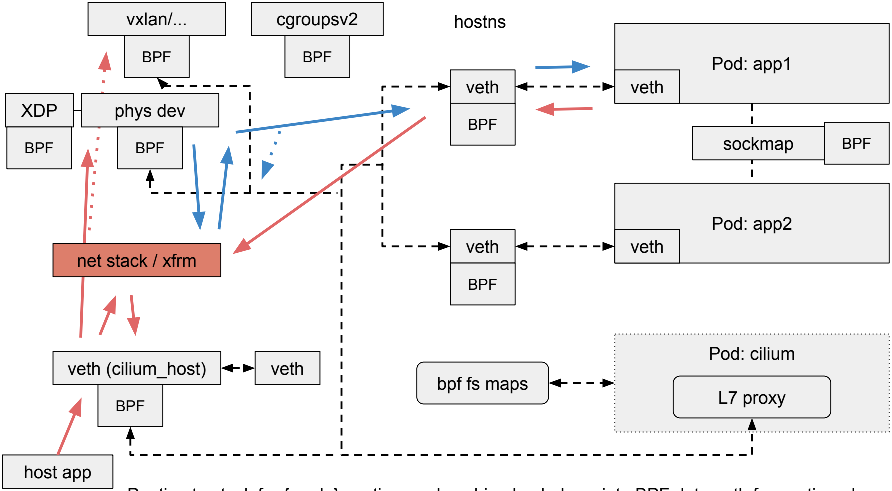
- BPF programs can be bypassed (raw sockets, or unloaded via netns' CAP_NET_ADMIN)
- Inflexible due to ipvlan internal FIB
- L3S mode (we've implemented BPF based SNAT which works in L3)
- BPF based NAT64 not working, etc, etc

Cilium BPF mesh data path (aka BPF veth/ipvlan hybrid)



- BPF orchestrated only from host, “inside the virtual dev”, not at tc egress of pod
- Fully programmable via BPF, forwarding to any other device part of mesh group
- Compat with existing tc BPF programs as much as possible
- ingress->ingress and egress->egress dev/netns switch, and also usual egress->ingress
- Option to also integrate tc’s routing lookup helpers

tc/BPF xfrm integration



- Punting to stack for {en,de}cryption, and pushing back down into BPF data path for continued processing, BPF may still rewrite src/dst IPs etc, plus doing forwarding
- xfrm processing based on the (pkt tuple + key id from skb->mark)
- Needed: tc(/XDP) BPF based helper for xfrm to avoid stack detour and have native integration E.g. helper to populate meta data for xfrm + return redirect_xfrm(ifindex, {0,BPF_F_INGRESS})