Advanced BPF Kernel Features for the Container Age

Daniel Borkmann, Cilium.io & co-maintainer BPF

FOSDEM 2021

BPF, a general purpose engine

Recap: minimal instruction set architecture with two major design goals:

BPF, a general purpose execution engine

1) Low overhead when mapping to native code, in particular: x86-64, arm64

2) Must be verifiable for **safety** by the kernel at program load time.

BPF, a general purpose execution engine

Safety to the extent that developers choose implementing programs in BPF rather than traditional kernel modules.

BPF, a general purpose execution engine

Kernel modules may cause kernel to panic. BPF programs are safety-checked and will **not** crash the kernel.

BPF, a general purpose execution engine

Kernel provides framework of building blocks and attachment points.

BPF, a general purpose execution engine

Is BPF a generic virtual machine? No.

BPF, a general purpose execution engine

Is BPF a fully generic instruction set? No.

BPF, a general purpose execution engine

It’s an instruction set with C calling convention in mind.

BPF, a general purpose execution engine

Why? Kernel is written in C and BPF needs to **efficiently** interact with the kernel.

Approx 150 BPF kernel helpers, 30 maps.

How far off is “BPF” C from generic C?

BPF, a general purpose execution engine

BPF has seen major advances such as
BPF-2-BPF function calls, bounded loops,
global variables, static linking, BTF,
up to 1 Mio instructions / program, ...

This already allows for solving a lot of interesting production issues.

Enter: **Cilium**

Cilium: cloud-native networking, security & observability built upon BPF

Cilium: use cases overview

Networking

- Highly efficient and flexible networking
- Routing, overlay, cloud-provider native
- IPv4, IPv6, NAT46
- Multi cluster routing

Load Balancing:

- Highly scalable L3-L4 (XDP) load balancing
- Kubernetes services (replaces kube-proxy)
- Multi-cluster
- Service affinity (prefer zones)

Network Security

- Identity-based network security
- API-aware security (HTTP, gRPC, ...), DNS-aware
- Transparent encryption

Observability

- Metrics (Network, DNS, Security, Latencies, HTTP, …)
- Flow logs (with datapath aggregation)

Servicemesh:

- Minimized overhead when injecting servicemesh sidecar proxies
- Istio integration

Deep dive 1: Kubernetes service load balancing with XDP, BPF & Maglev

https://cilium.io/blog/2020/11/10/cilium-19#maglev

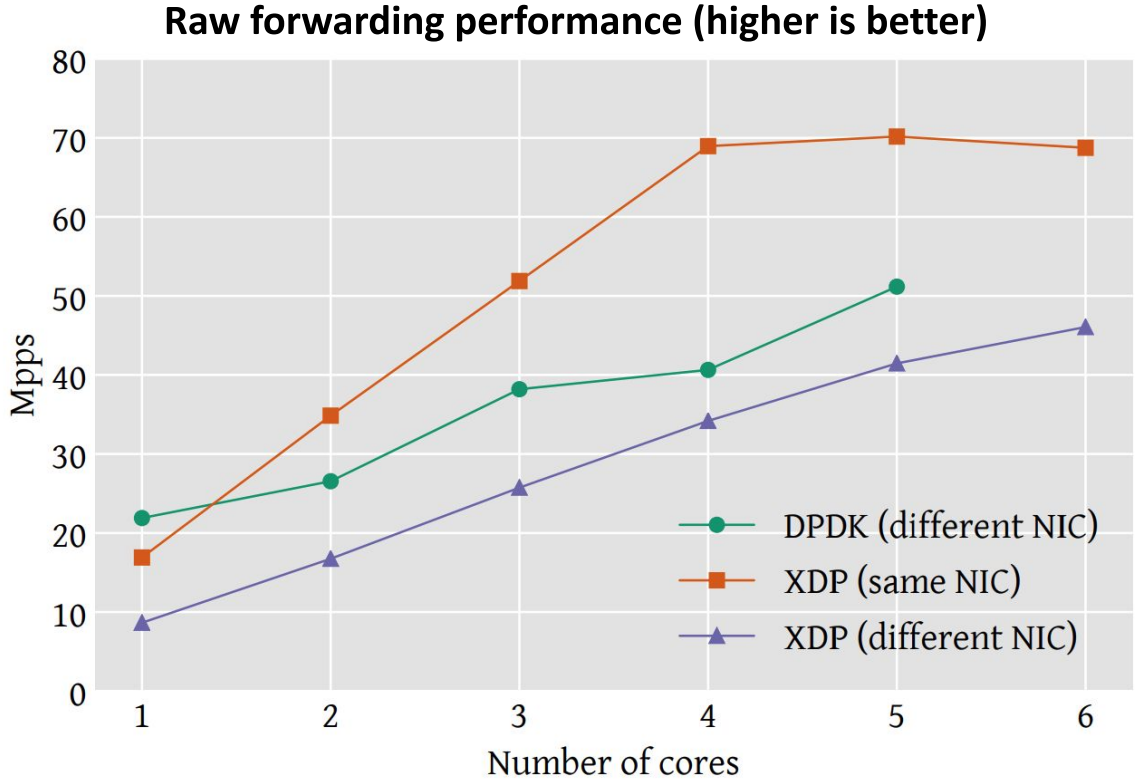
kube-proxy example: co-location of service load balancer with regular user workloads on every node.

Recent Cilium 1.9 and BPF kernel extensions

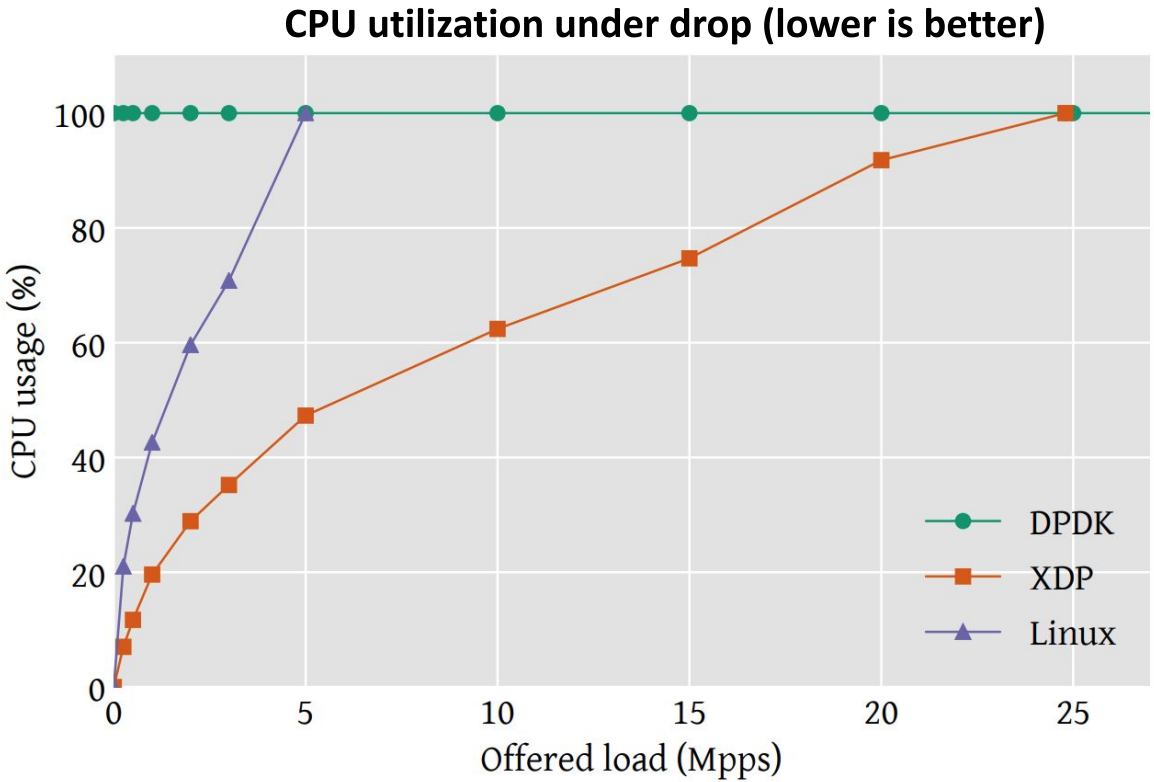
Migrating (N-S) load balancing from legacy subsystems to BPF at XDP layer reduces CPU cost significantly, and achieves DPDK speeds.

https://dl.acm.org/doi/pdf/10.1145/3281411.3281443

Recent Cilium 1.9 and BPF kernel extensions



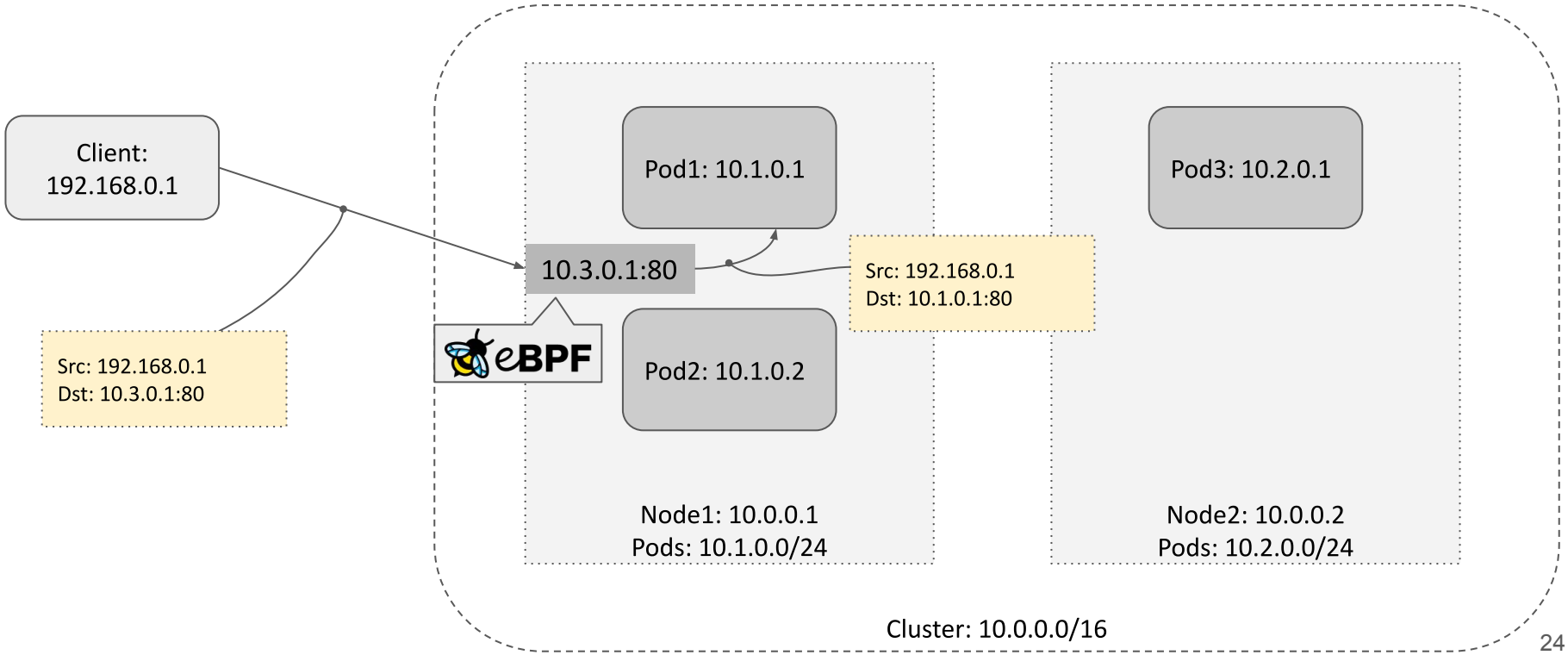
Recent Cilium 1.9 and BPF kernel extensions



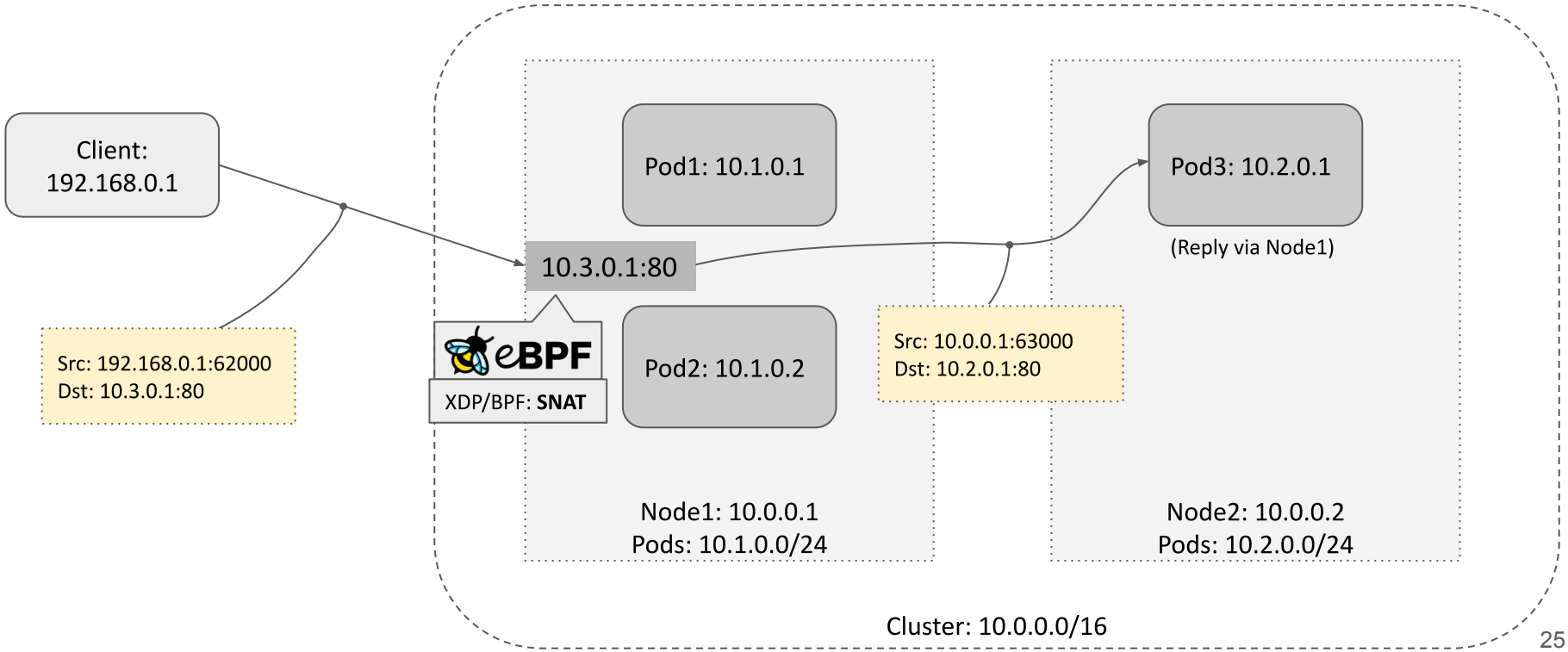
Recent Cilium 1.9 and BPF kernel extensions

... all with BPF on upstream kernel drivers. No busy-polling CPUs. No user-kernel boundary crossing.

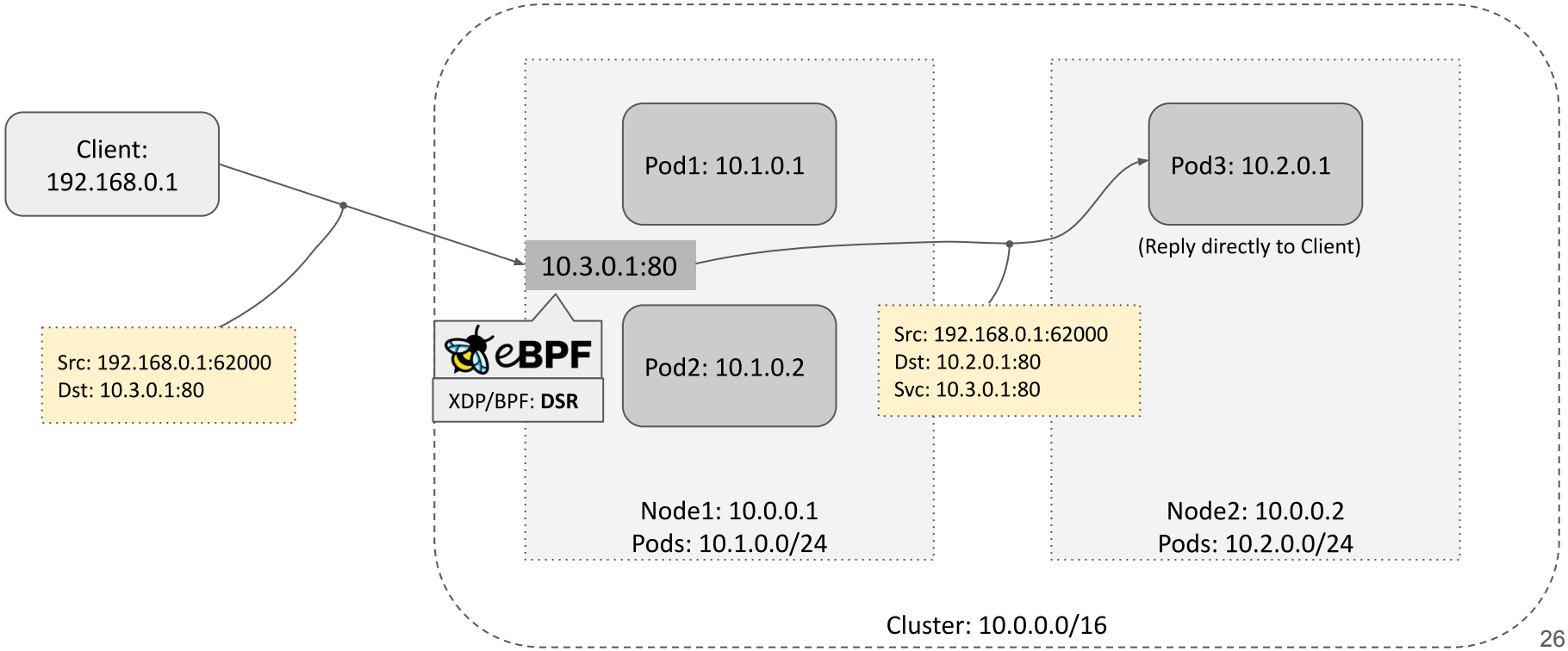
K8s LoadBalancer service on-prem example



K8s LoadBalancer service on-prem example



K8s LoadBalancer service on-prem example



K8s LoadBalancer service on-prem example

LoadBalancer implementation done by Cloud providers or MetalLB for on-prem.

MetalLB can announce via ARP/NDP or BGP.

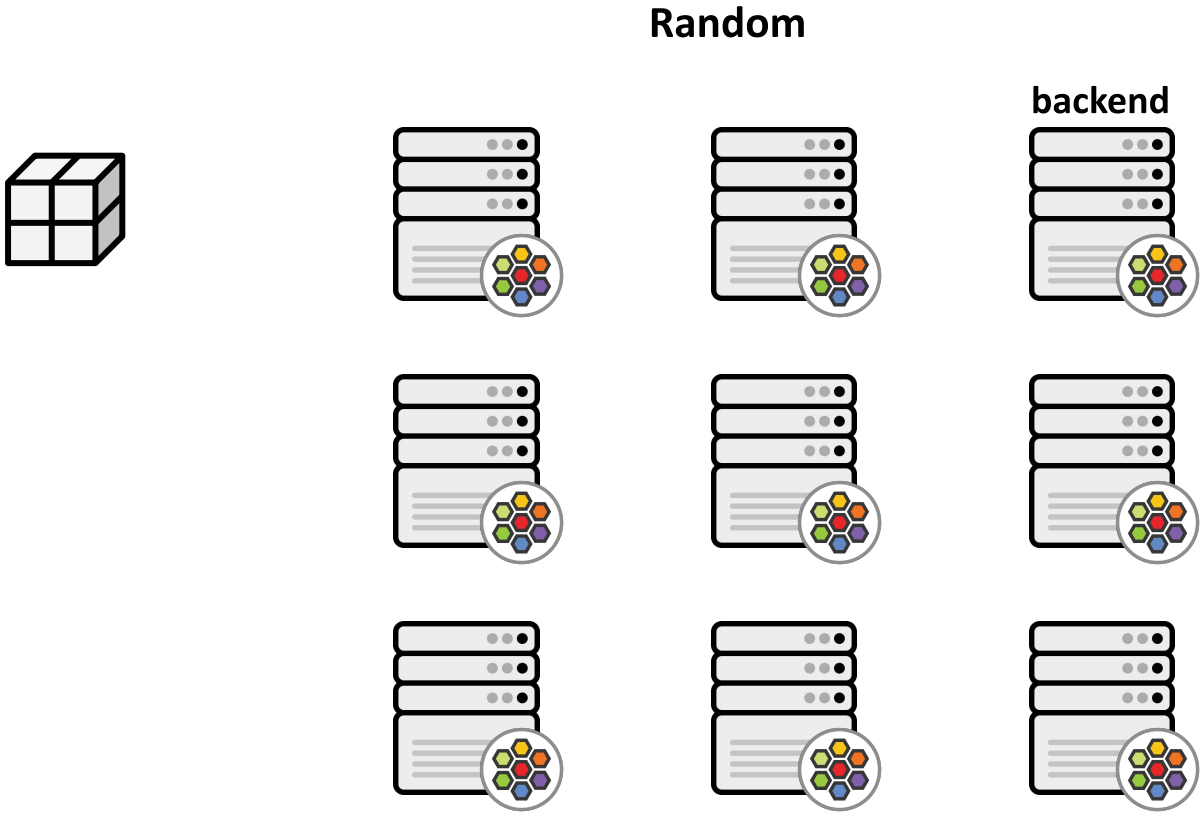
K8s LoadBalancer service on-prem example

MetalLB does IP address allocation and external announcement, but does not sit in critical fast path (hence works with XDP).

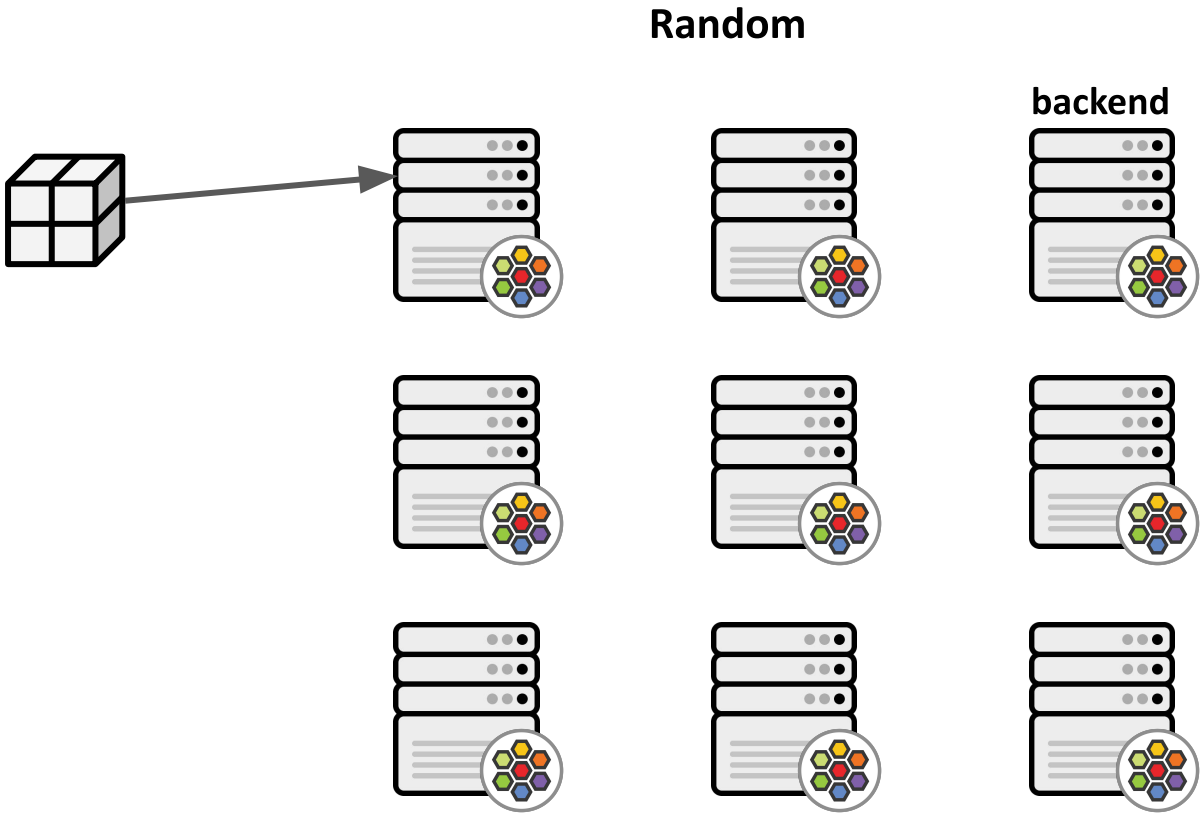
K8s LoadBalancer service on-prem example

Default backend selection random with node-local BPF CT for remaining sticky to backend. However ...

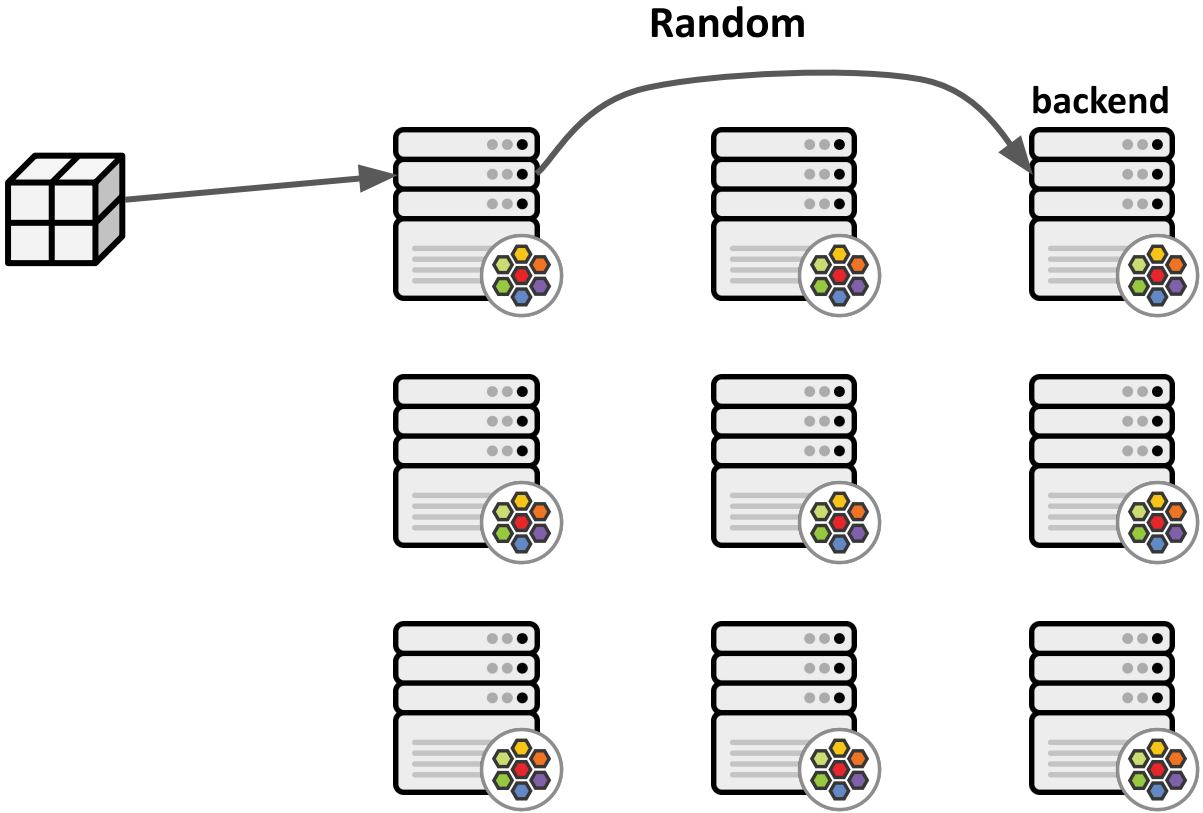
Recent Cilium 1.9 and BPF kernel extensions



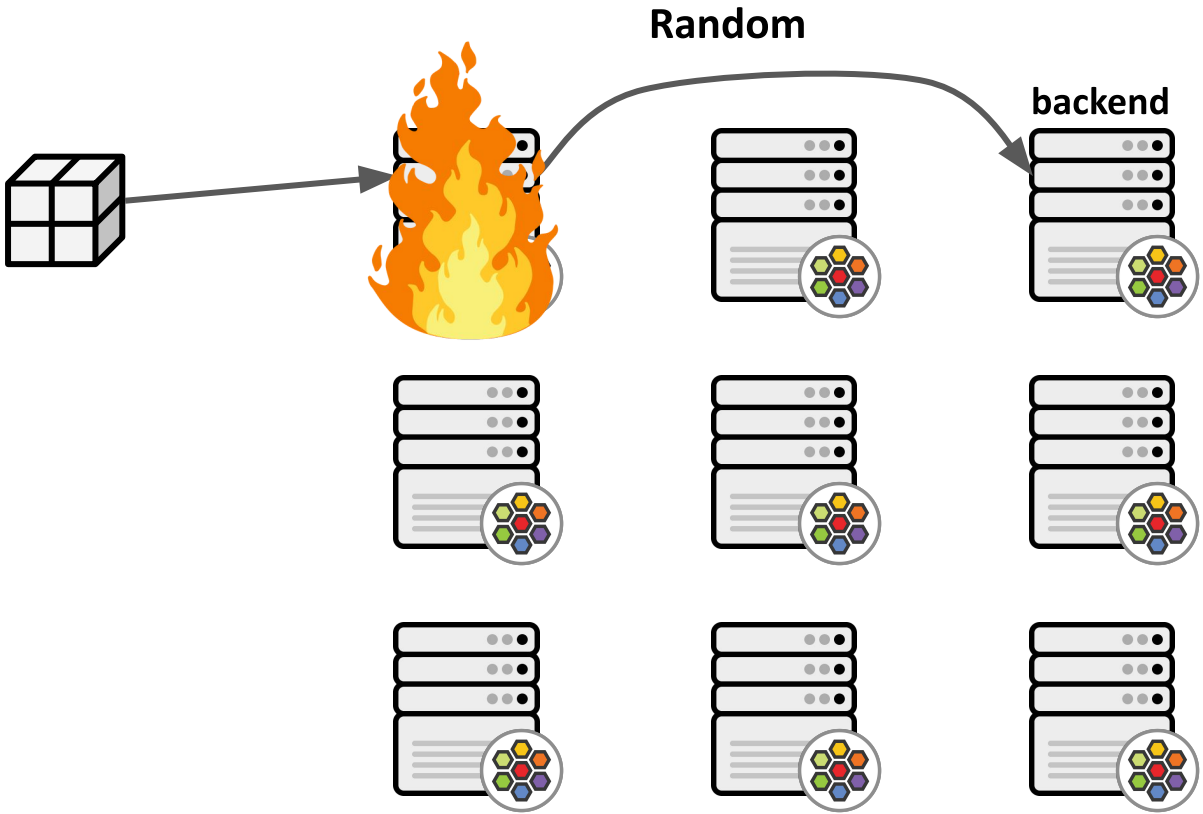
Recent Cilium 1.9 and BPF kernel extensions



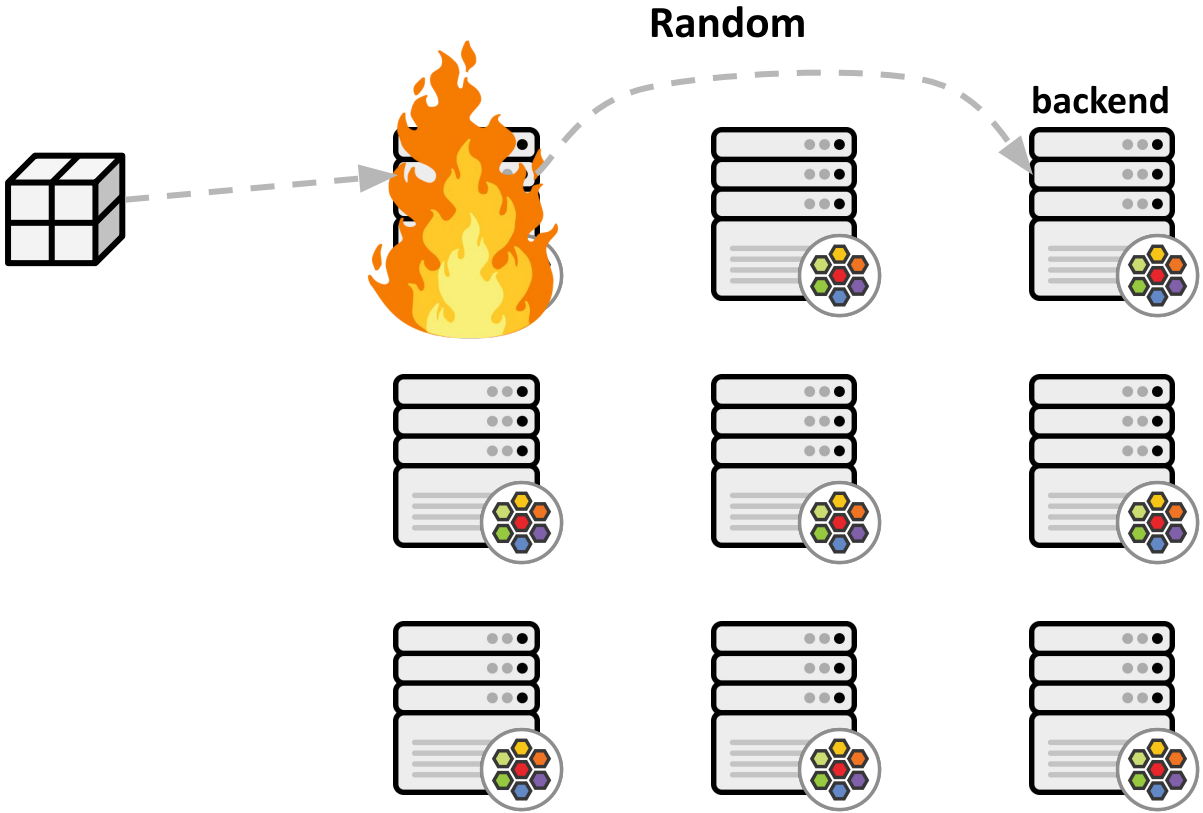
Recent Cilium 1.9 and BPF kernel extensions



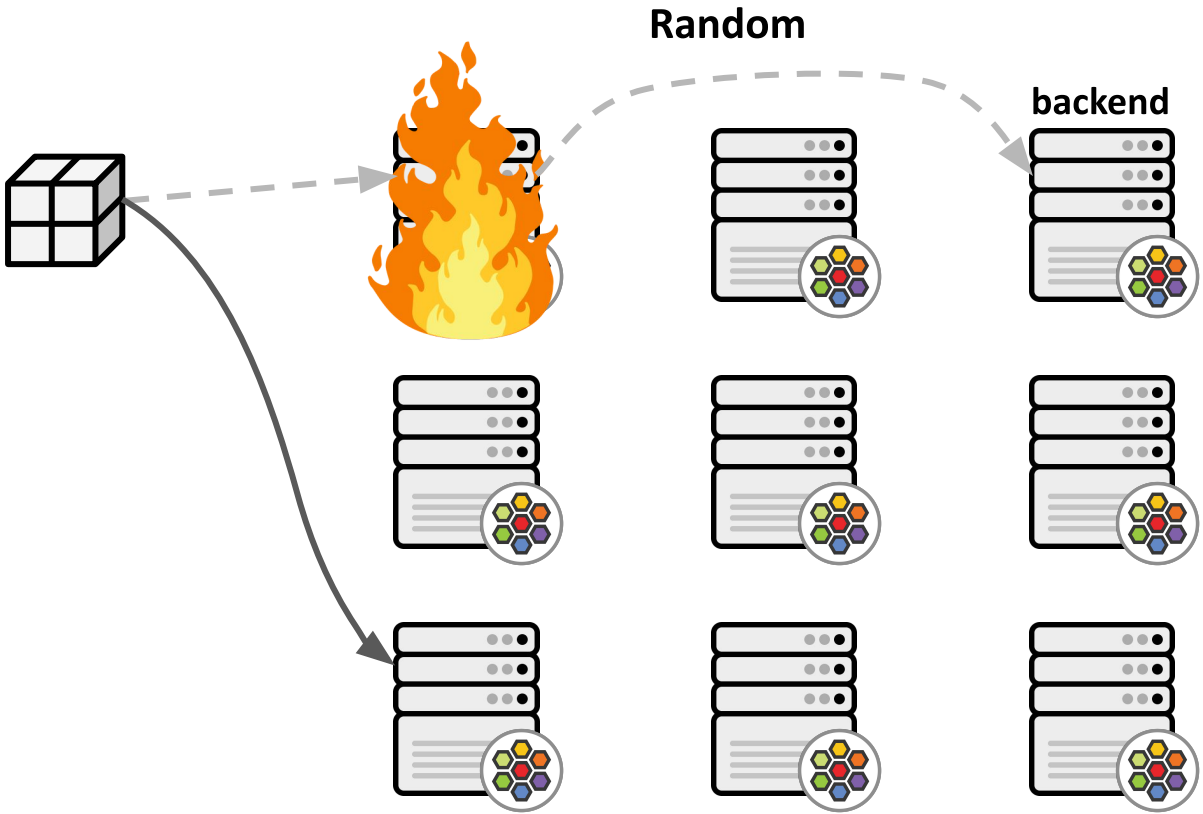
Recent Cilium 1.9 and BPF kernel extensions



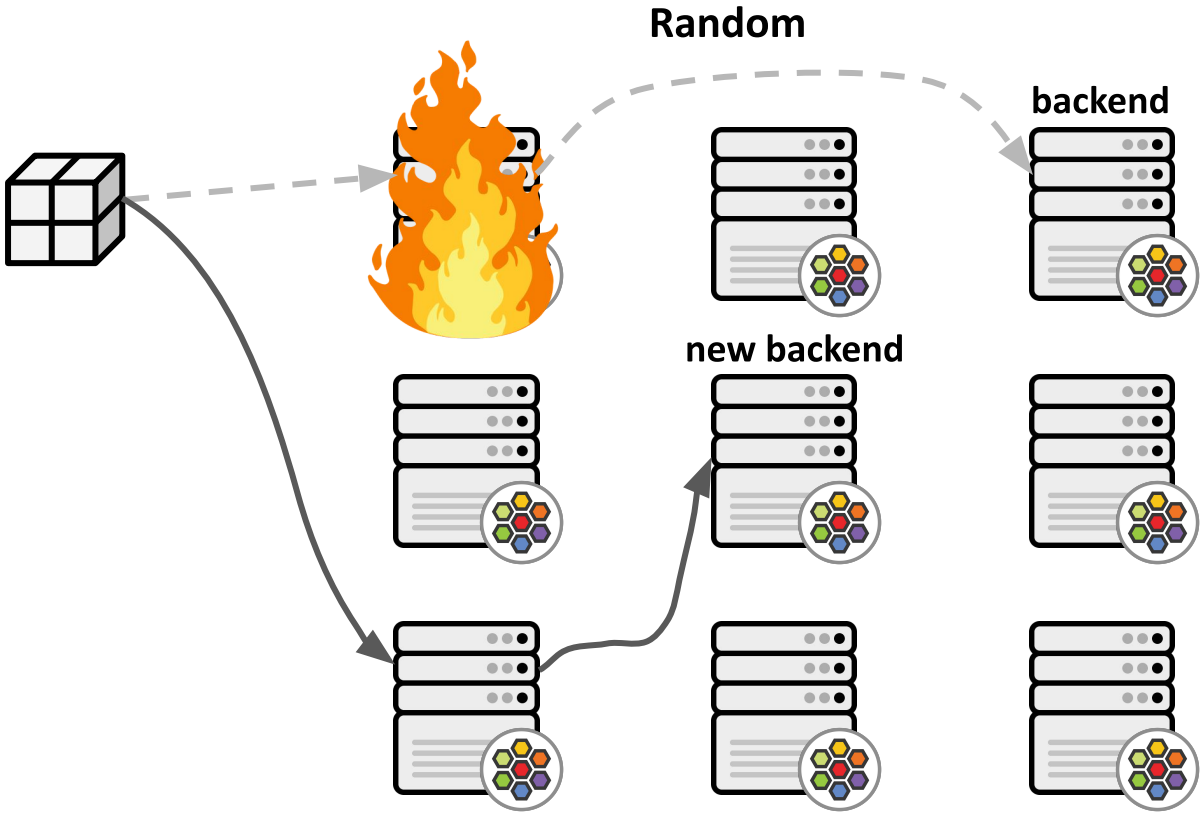
Recent Cilium 1.9 and BPF kernel extensions



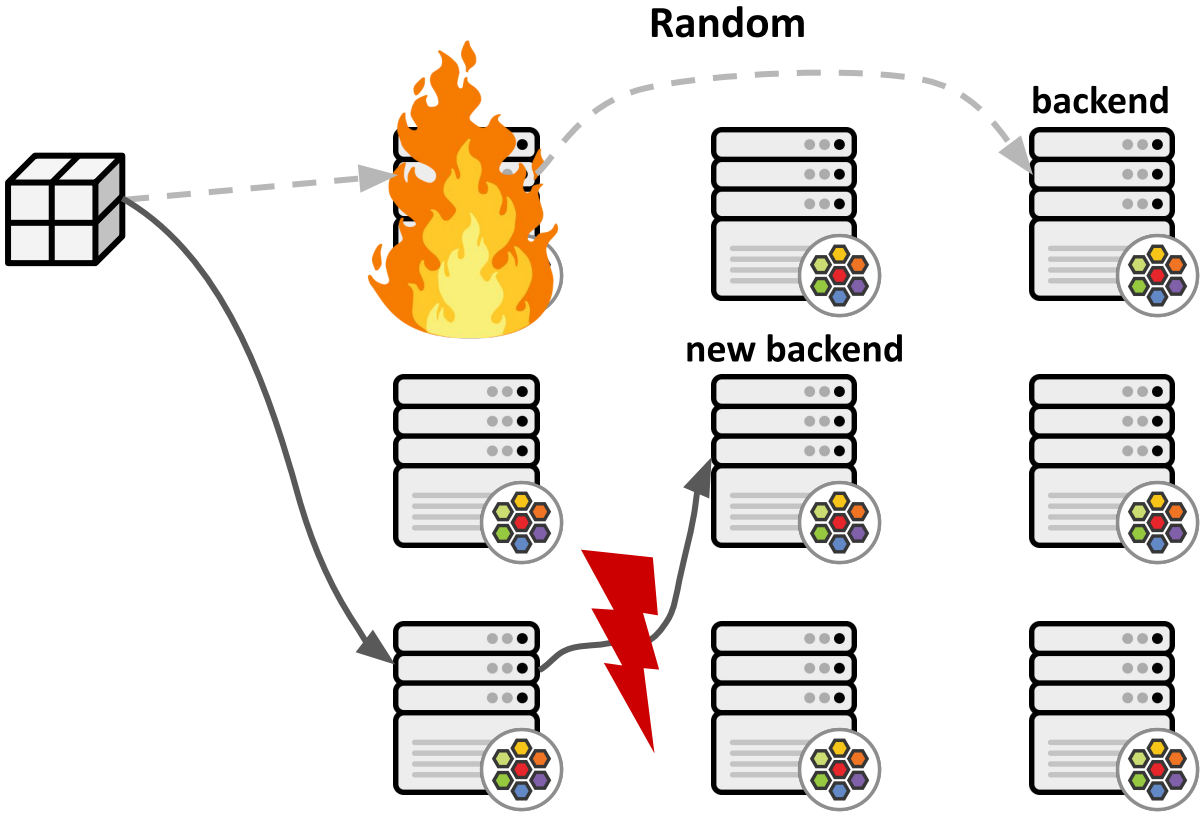
Recent Cilium 1.9 and BPF kernel extensions



Recent Cilium 1.9 and BPF kernel extensions

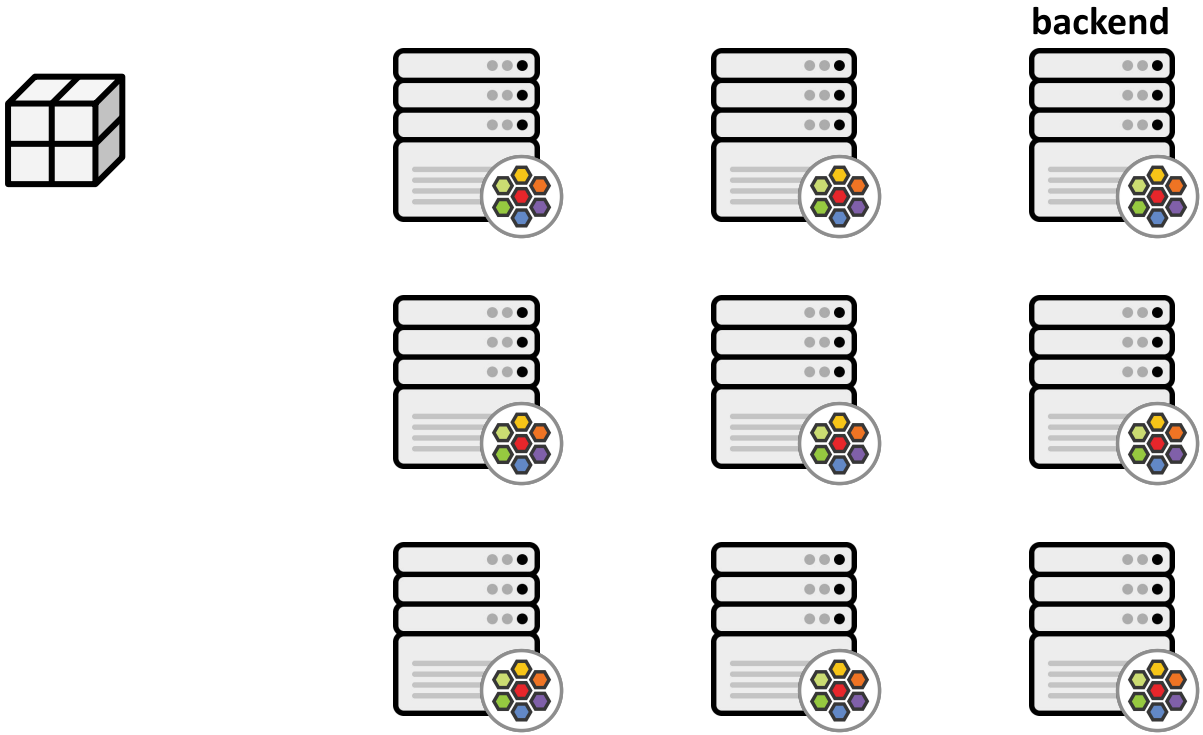


Recent Cilium 1.9 and BPF kernel extensions

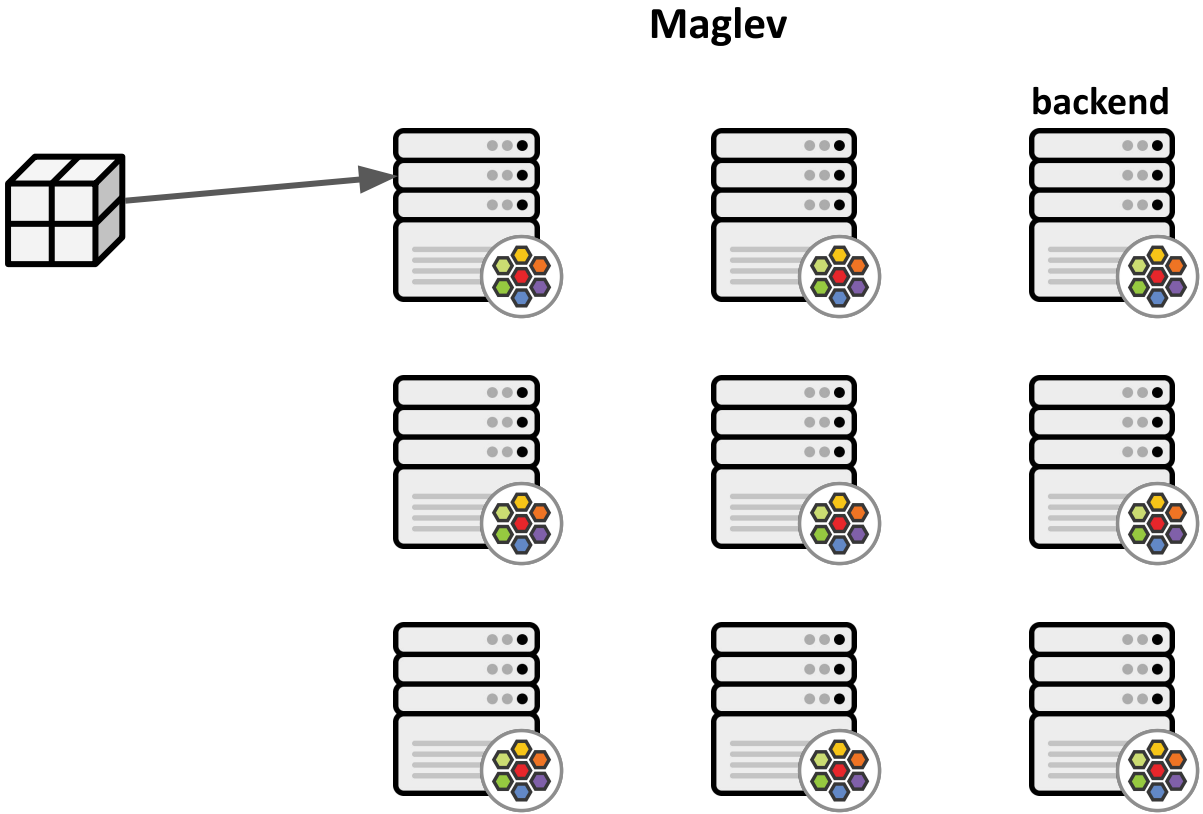


Needs consistent hashing for resiliency against failures. Cilium 1.9 supports Maglev-based selection from BPF/XDP side.

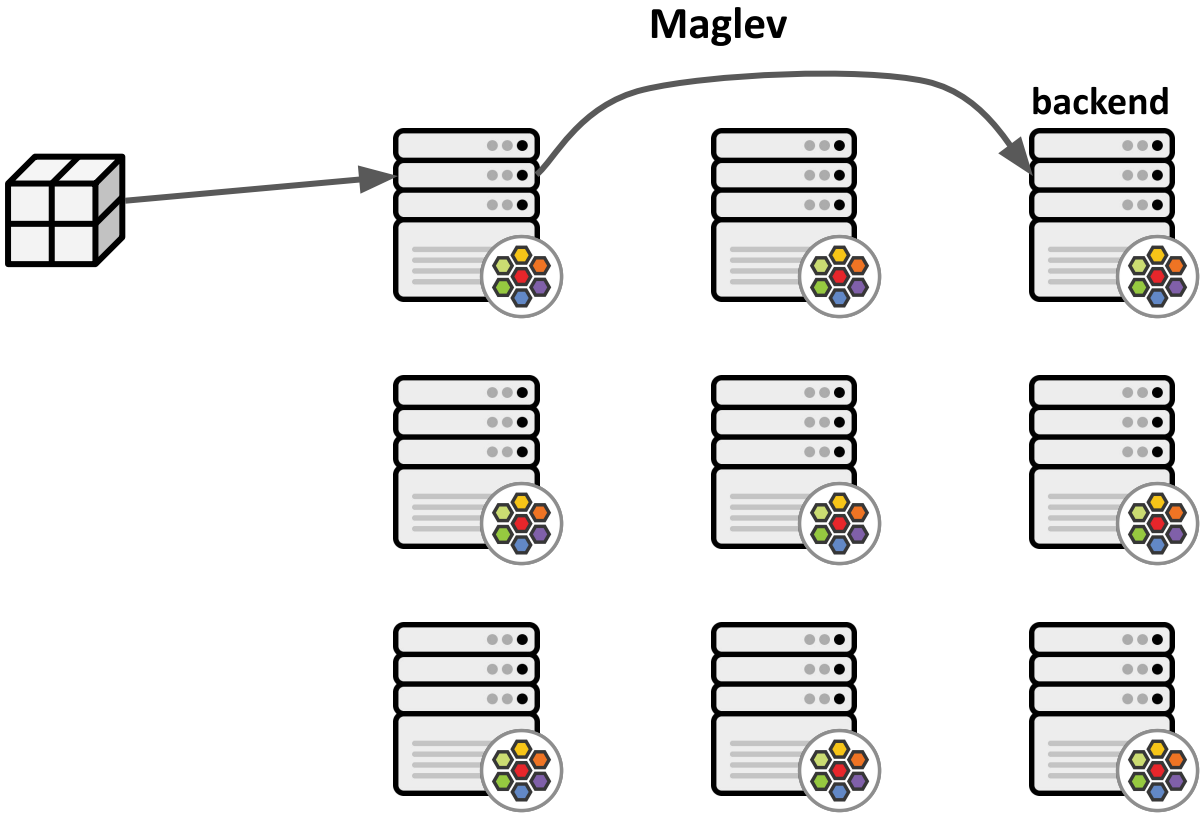
Recent Cilium 1.9 and BPF kernel extensions



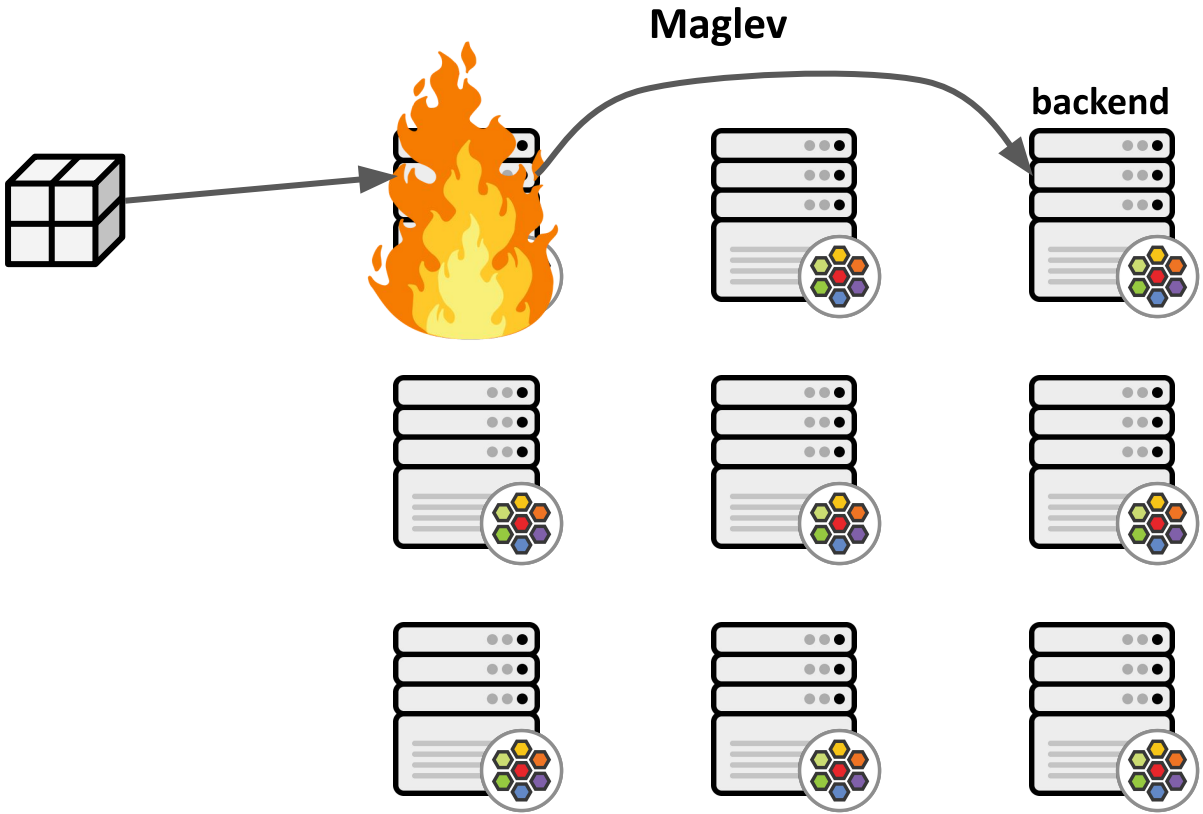
Recent Cilium 1.9 and BPF kernel extensions



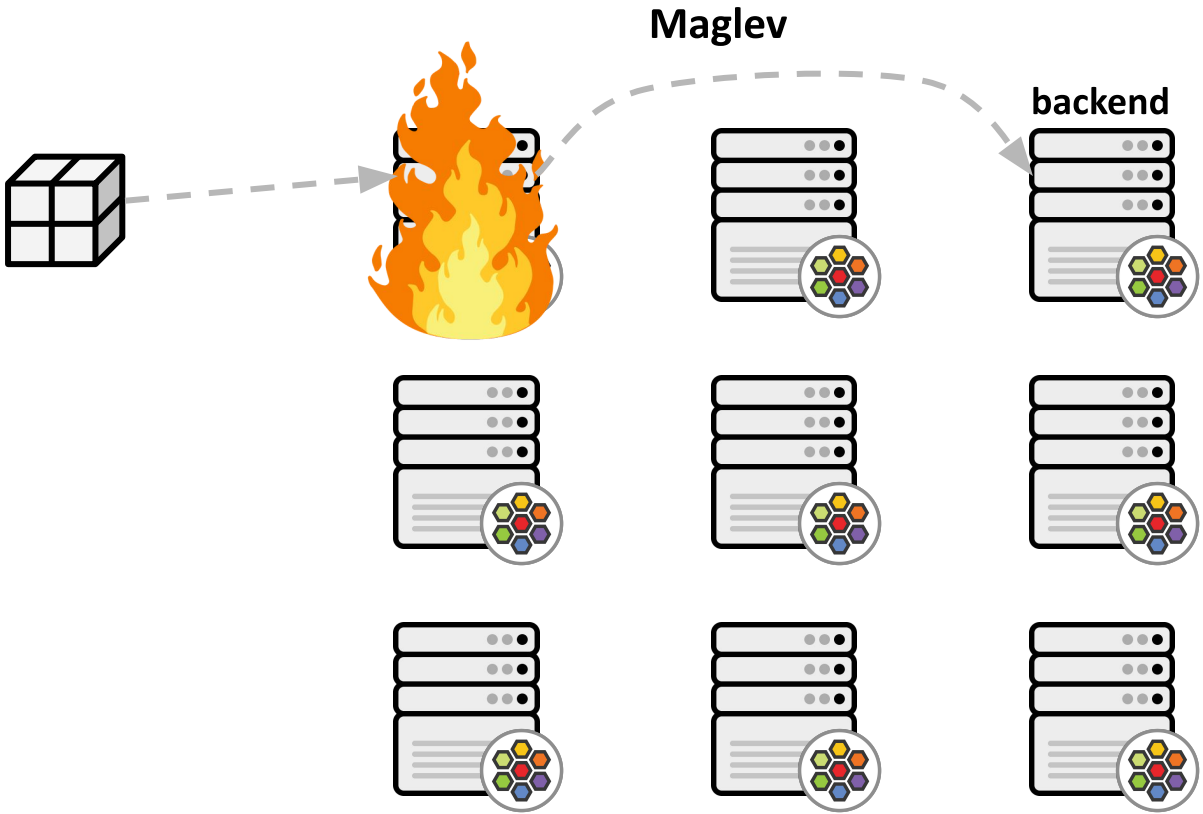
Recent Cilium 1.9 and BPF kernel extensions



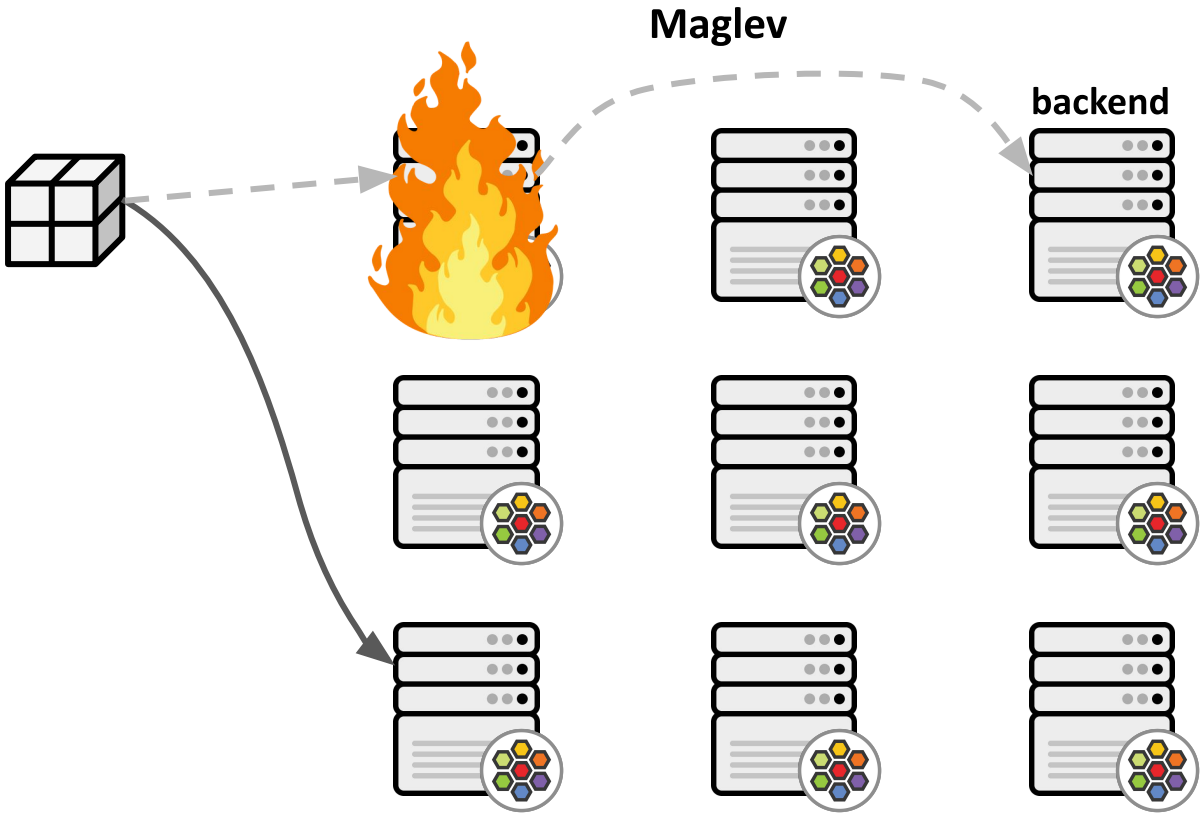
Recent Cilium 1.9 and BPF kernel extensions



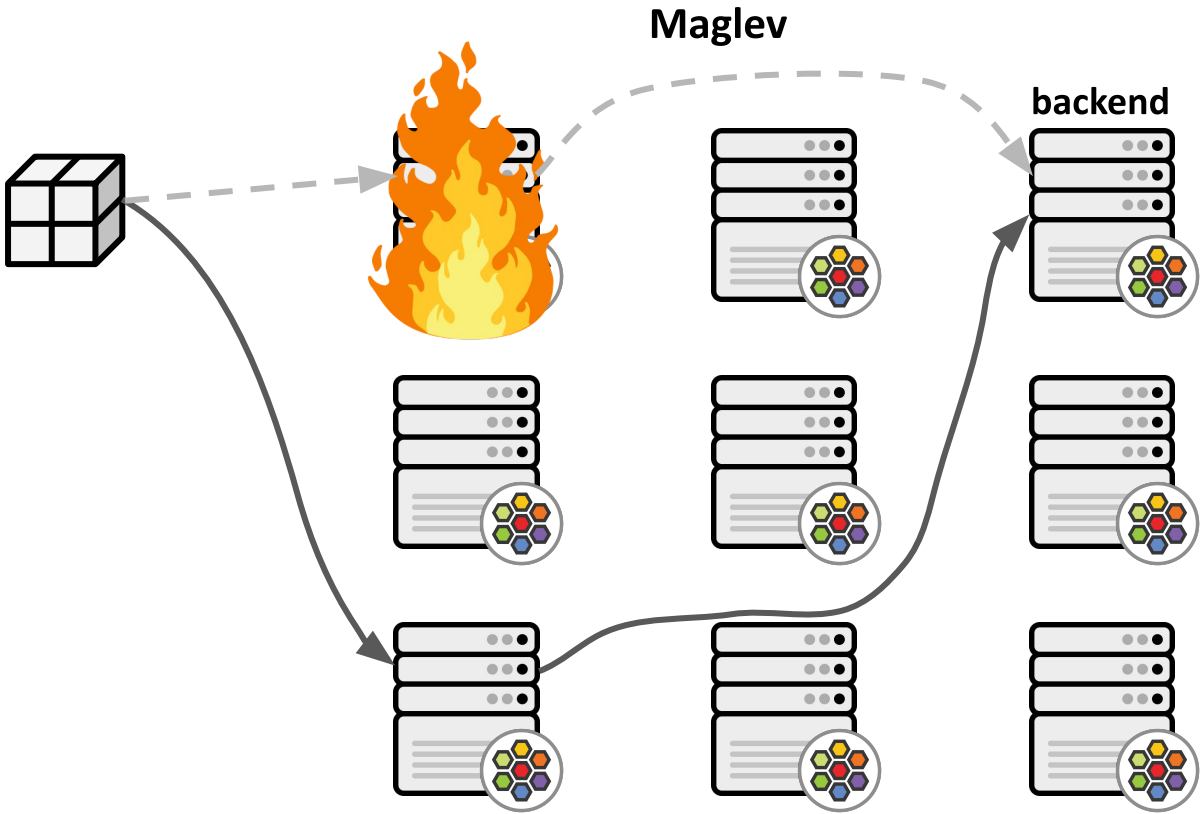
Recent Cilium 1.9 and BPF kernel extensions



Recent Cilium 1.9 and BPF kernel extensions



Recent Cilium 1.9 and BPF kernel extensions



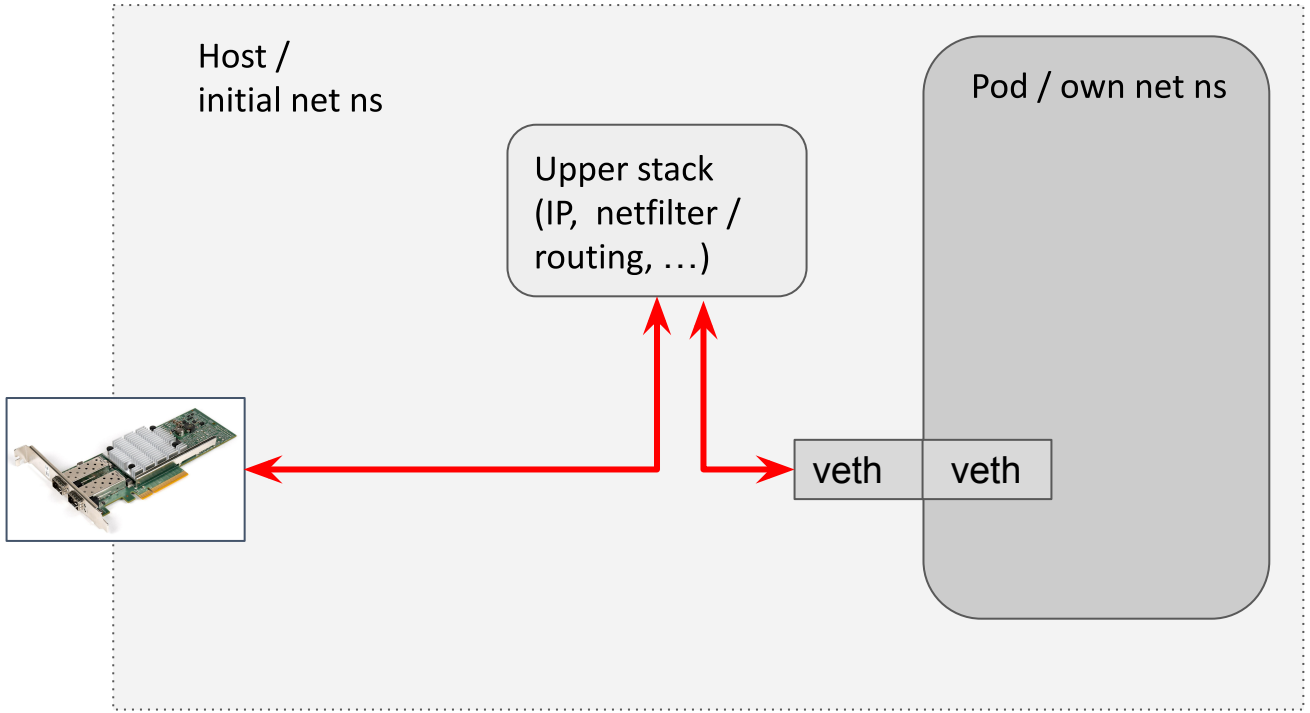
BPF/XDP datapath performs dynamic sized map-in-map lookup for optimized memory utilization. Backend selected via jhash of tuple over per-service Maglev table.

https://git.kernel.org/torvalds/c/4a8f87e60f6d

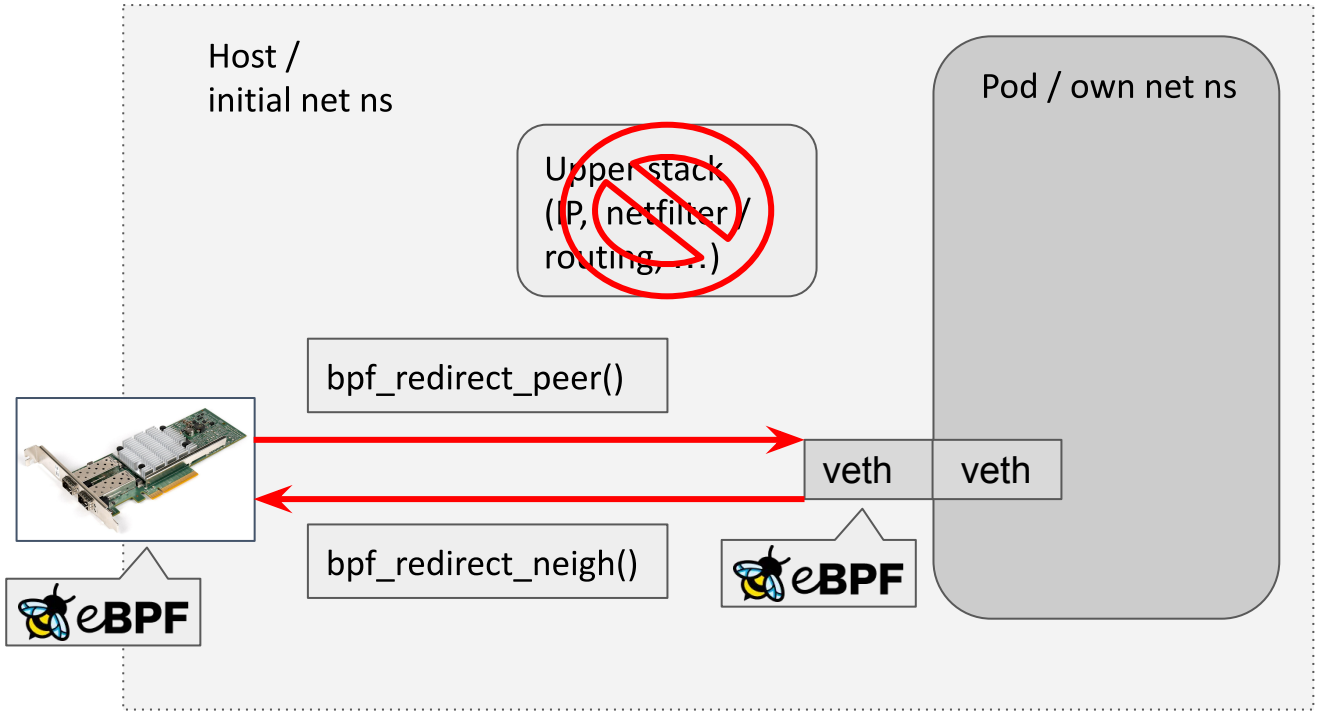
Deep dive 2: low-latency Pod fast-path with BPF

https://cilium.io/blog/2020/11/10/cilium-19#veth

Recent Cilium 1.9 and BPF kernel extensions



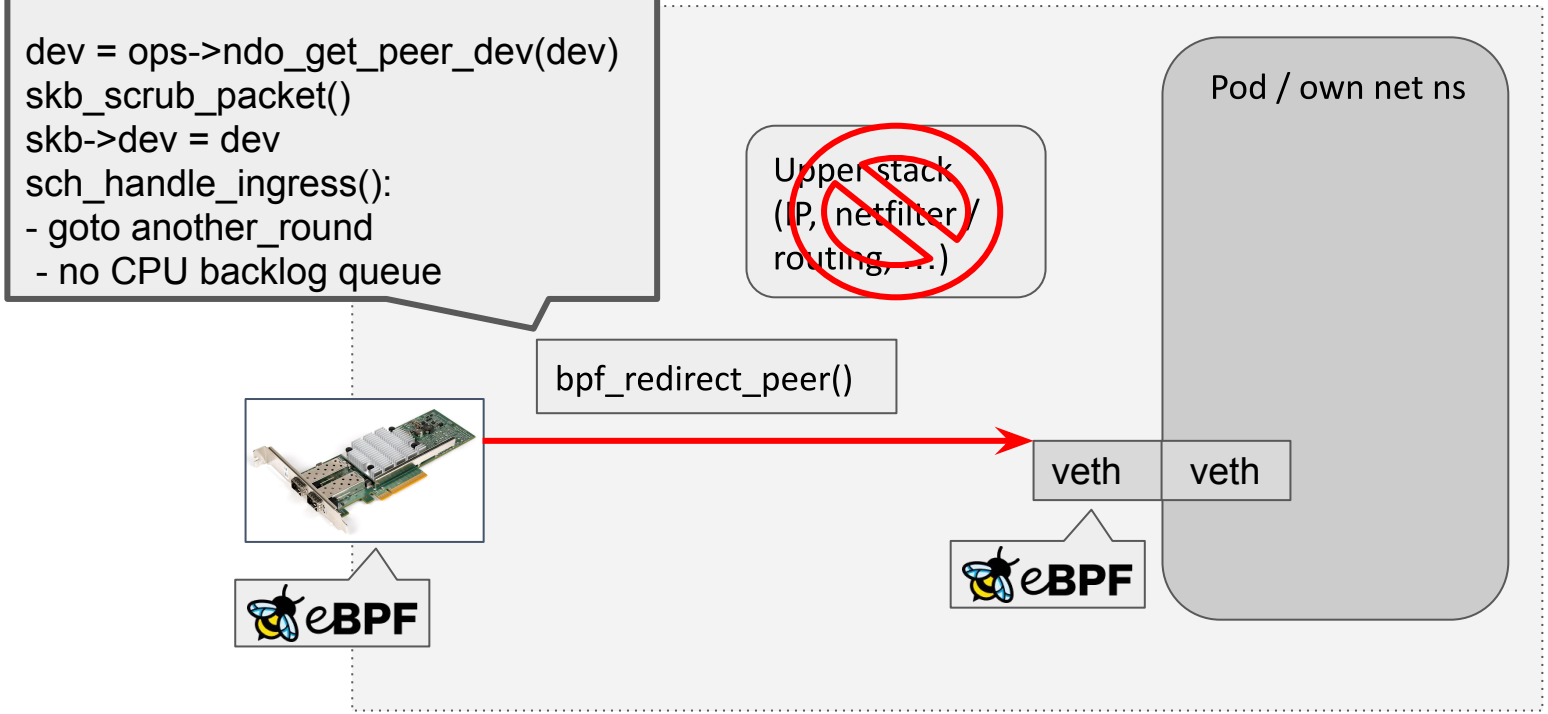
Recent Cilium 1.9 and BPF kernel extensions



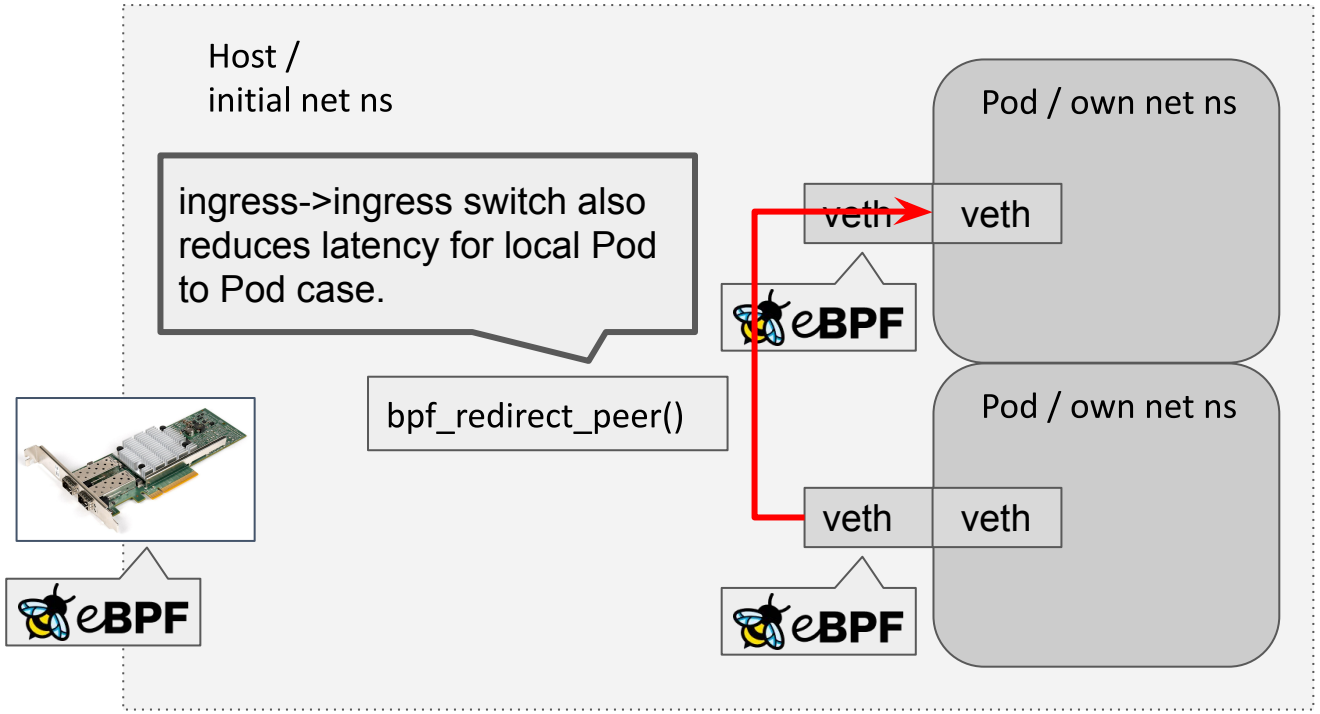
Recent Cilium 1.9 and BPF kernel extensions

High-level internals:

dev = ops->ndo_get_peer_dev(dev)
skb_scrub_packet()
skb->dev = dev
sch_handle_ingress():
- goto another_round
 - no CPU backlog queue



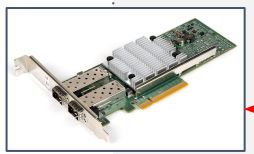
Recent Cilium 1.9 and BPF kernel extensions

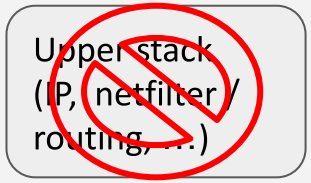


Recent Cilium 1.9 and BPF kernel extensions

High-level internals:

ip_route_output_flow()
skb_dst_set()
neigh_output()
- fills in neigh info
- retains skb->sk till Qdisc on phys





Pod / own net ns

veth | veth

bpf_redirect_neigh()

Recent Cilium 1.9 and BPF kernel extensions

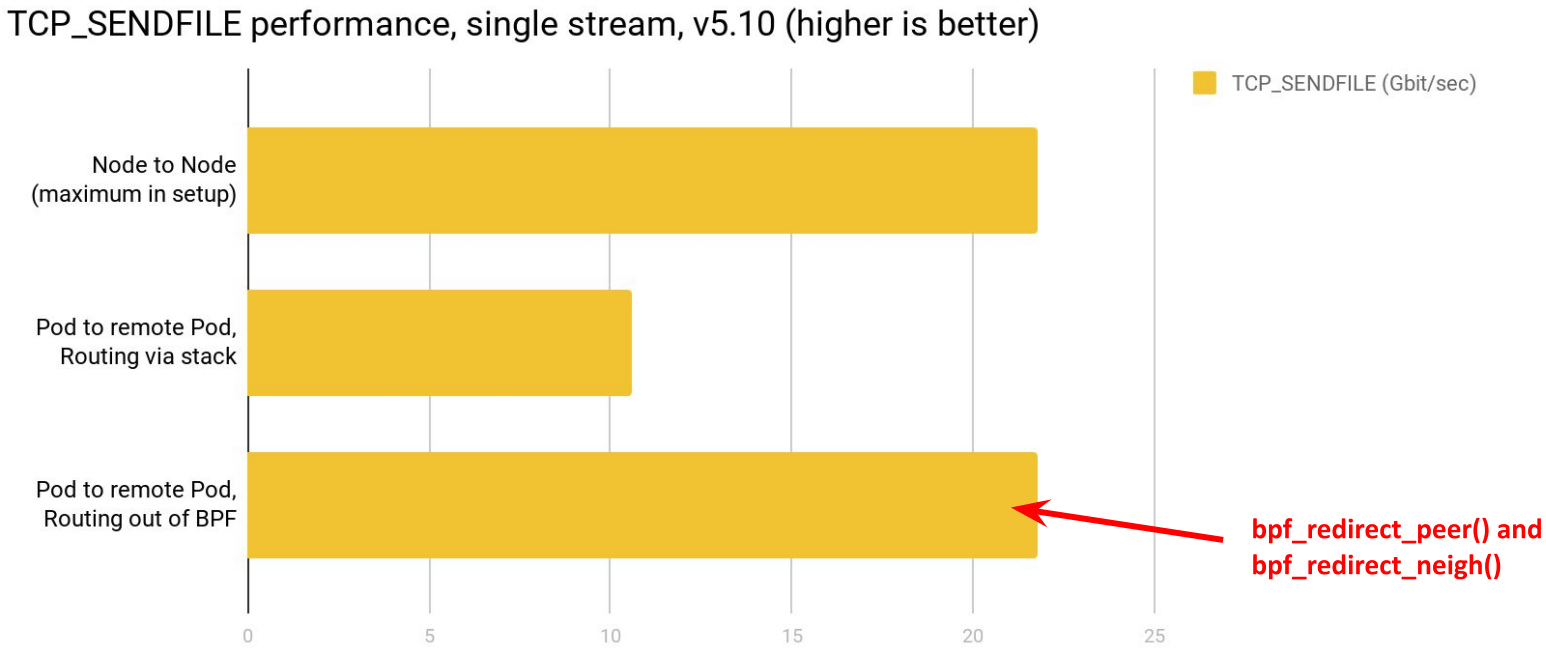
Low-latency net ns switch into Pod from BPF.

Automatic L2 resolution for traffic from Pod.

Both directions now avoid host stack.

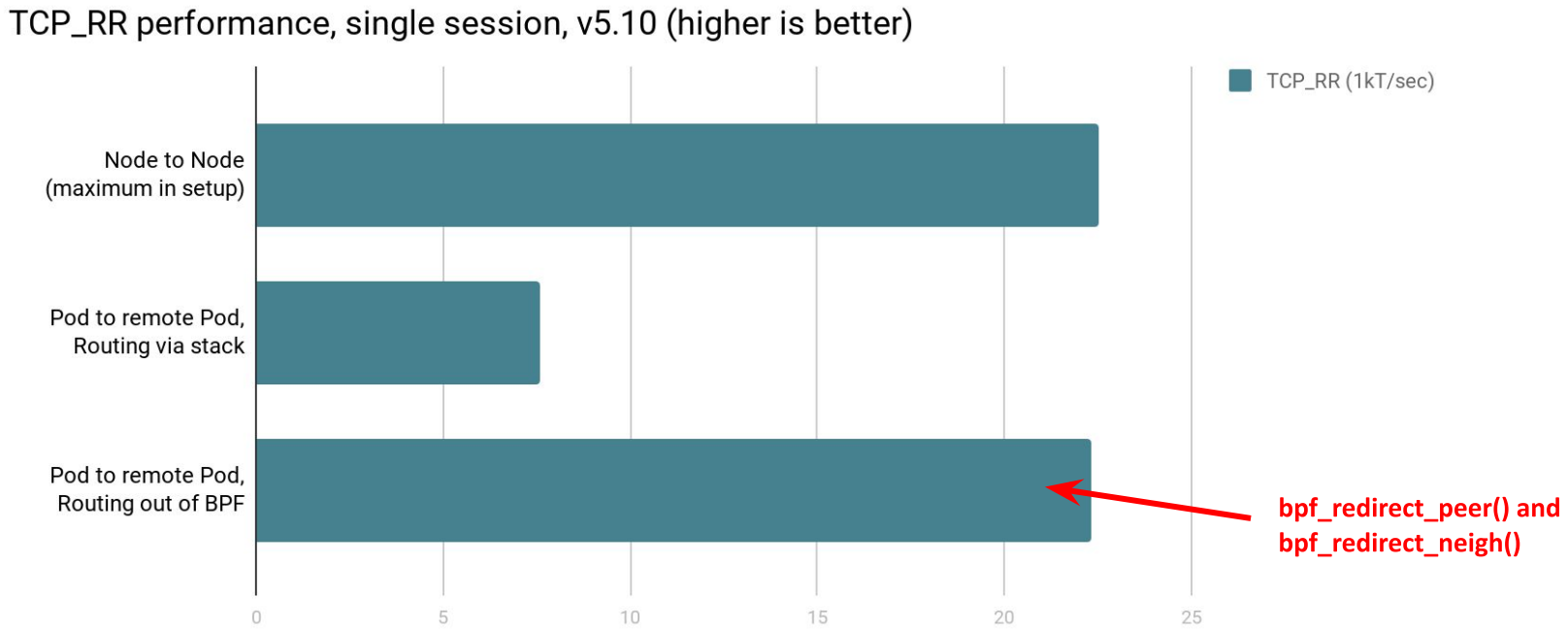
https://git.kernel.org/torvalds/c/9aa1206e8f48
https://git.kernel.org/torvalds/c/b4ab31414970

Recent Cilium 1.9 and BPF kernel extensions



Xeon E3-1240, 3.4 GHz each (4 cores, HT off), back to back via nfp, IRQs pinned to CPUs, tuned with profile network-throughput
From Pod ns: netperf -H <remote-pod-ip> -t TCP_SENDFILE -T0,0 -P0 -s2 -l 60 -D 2 -f g

Recent Cilium 1.9 and BPF kernel extensions



Xeon E3-1240, 3.4 GHz each (4 cores, HT off), back to back via nfp, IRQs pinned to CPUs, tuned with profile network-latency
From Pod ns: percpu_netperf <remote-pod-ip> (https://github.com/borkmann/netperf_scripts/blob/master/percpu_netperf)

Deep dive 3: EDT rate-limiting for Pods via BPF

https://cilium.io/blog/2020/11/10/cilium-19#bwmanager

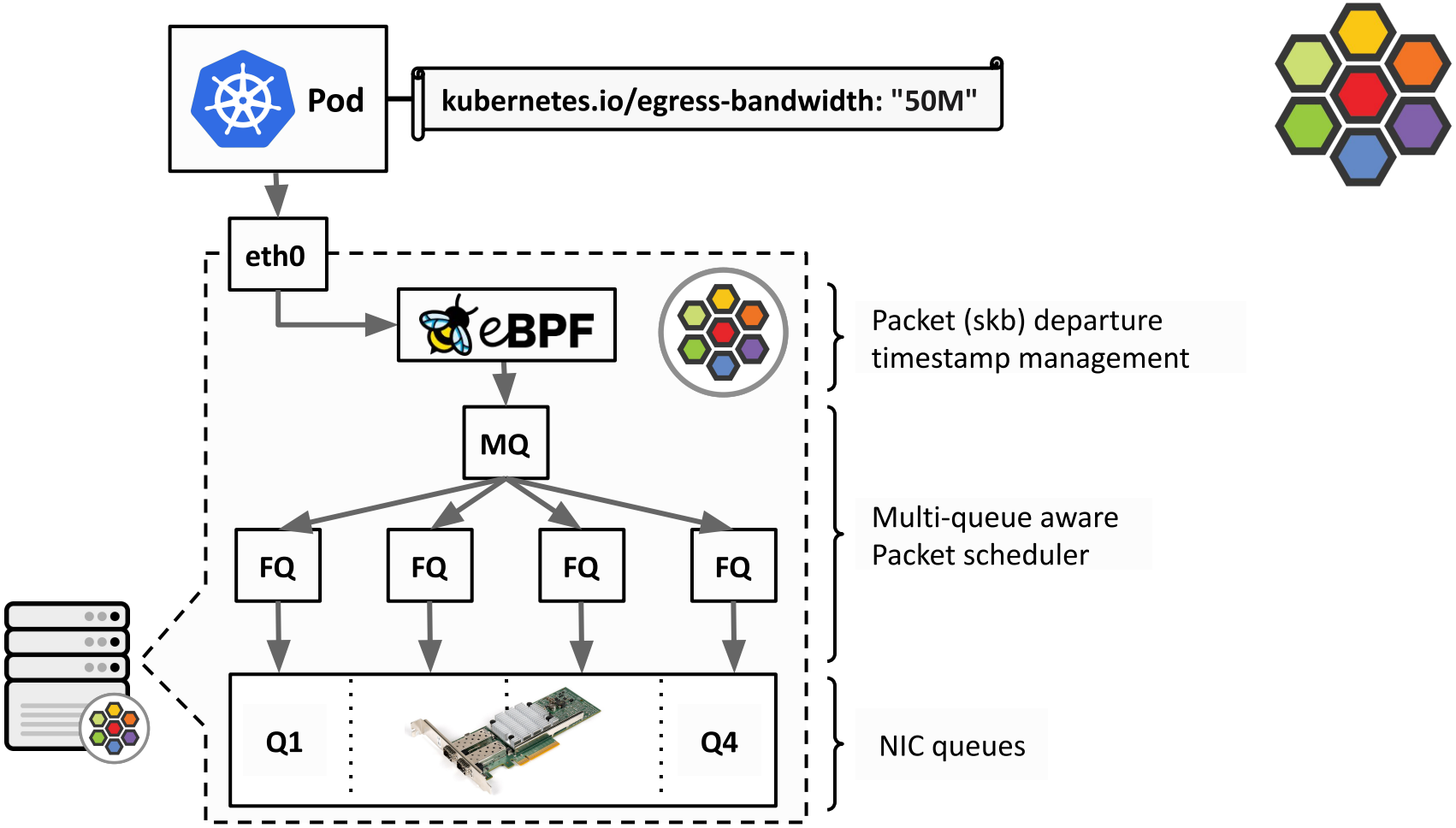
Old way: CNI chaining with bandwidth plugin that sets up TBF qdisc. Even sets up ifb for ingress shaping. Not scalable.

https://www.cni.dev/plugins/meta/bandwidth/

Cilium native: lock-less Pod rate-limiting on multi-queue via BPF and Earliest Departure Time (EDT).

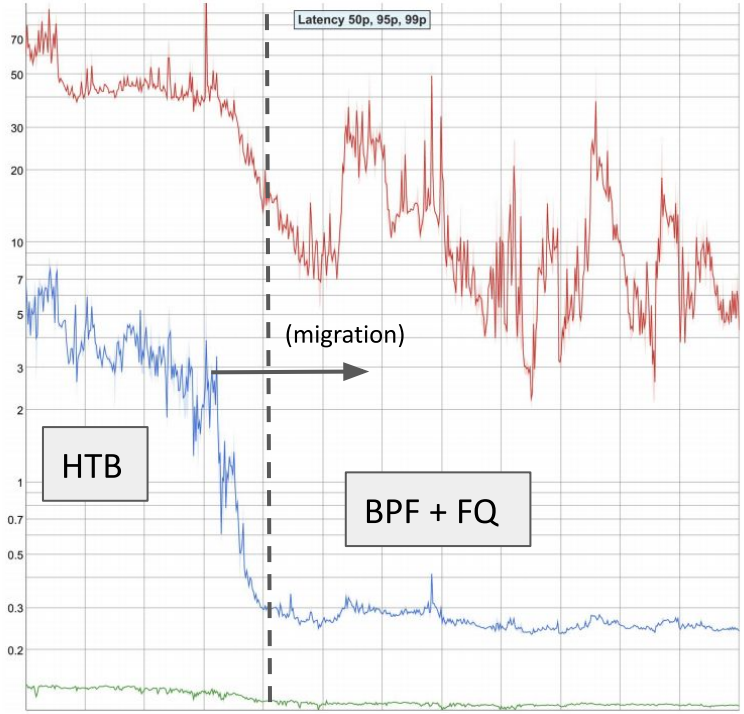
BPF classifies network traffic to Pod and then sets packet departure time (skb->tstamp) based on user defined bandwidth rate.

FQ qdisc schedules packet under this time constraint.


Pod
kubernetes.io/egress-bandwidth: "50M"
eth0
eBPF
MQ
FQ
FQ
FQ
FQ
Q1
Q4
Packet (skb) departure timestamp management
Multi-queue aware Packet scheduler
NIC queues

Recent Cilium 1.9 and BPF kernel extensions

Transmission latency (lower is better)



99p → **10x latency reduction**

95p → **20x latency reduction**

50p

https://netdevconf.info/0x14/pub/papers/55/0x14-paper55-talk-paper.pdf

Recap

Unlike other kernel subsystems BPF is in a unique position to innovatively tackle and optimize such complex production issues.

Recap

Cilium with BPF at its core delivers this technological revolution to mainstream Kubernetes.



Urs Hölzle @uhoelzle · Aug 29

ICYMI: GKE is, once again, the first managed kubernetes service to provide a key new feature. This time it's better networking with **eBPF** support.

Tim Hockin
@thockin

Cilium and the team around it have impressed me from the beginning. My mind is spinning with the possibilities of eBPF.

Kelsey Hightower ✓
@kelseyhightower

The latest @ciliumproject release is taking this Kubernetes networking thing to another level: multi-cluster service routing, IPVLAN support, transparent encryption, and DNS authorization.
cilium.io/blog/2019/02/1...

Thanks! Questions?

github.com/cilium/cilium

cilium.io
ebpf.io