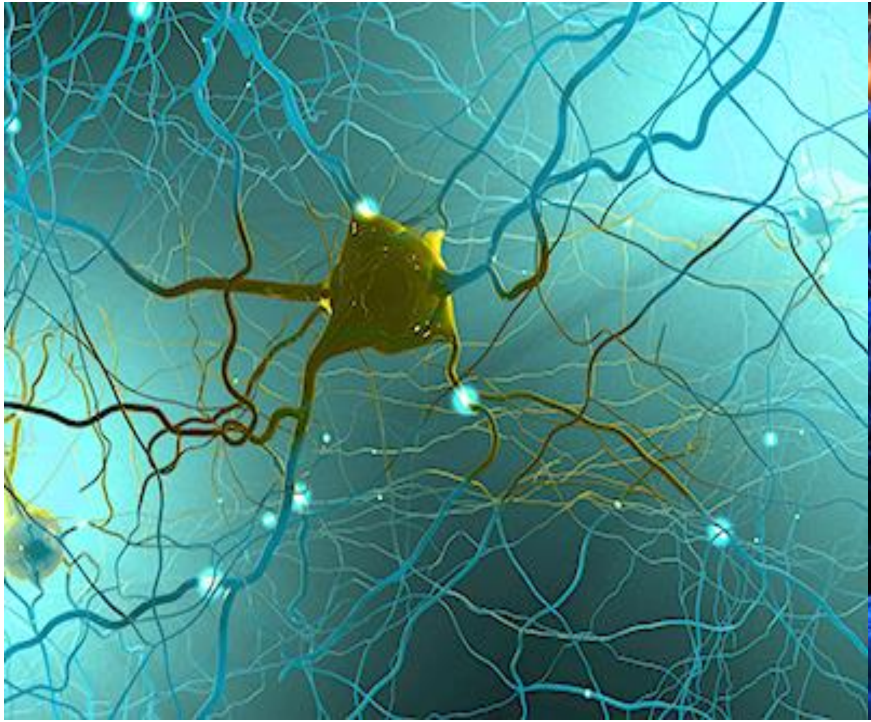
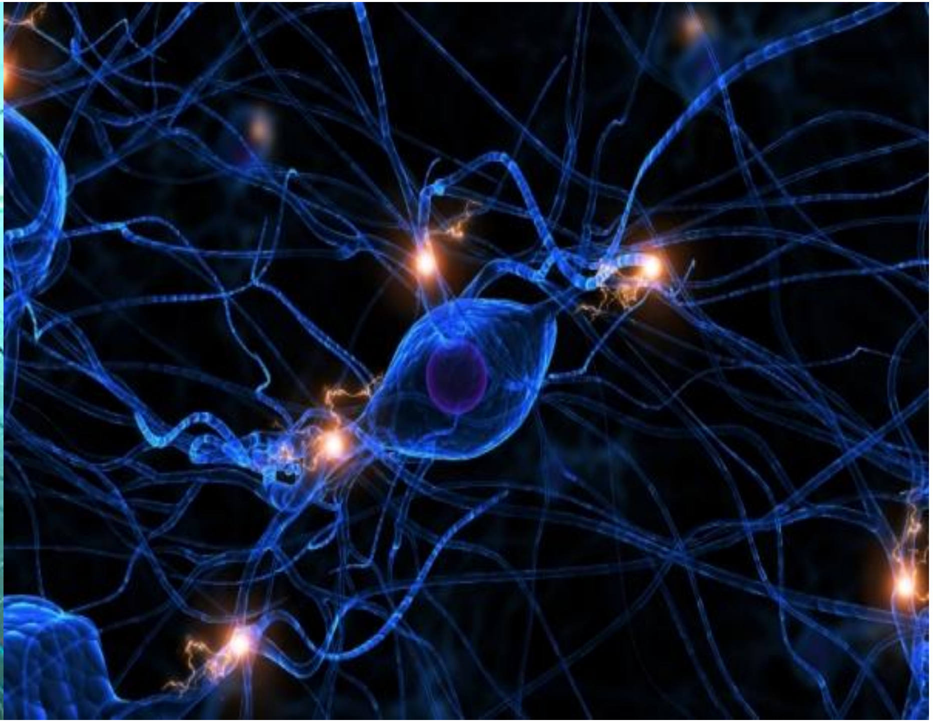
# Cilium as container networking solution

# Outlines

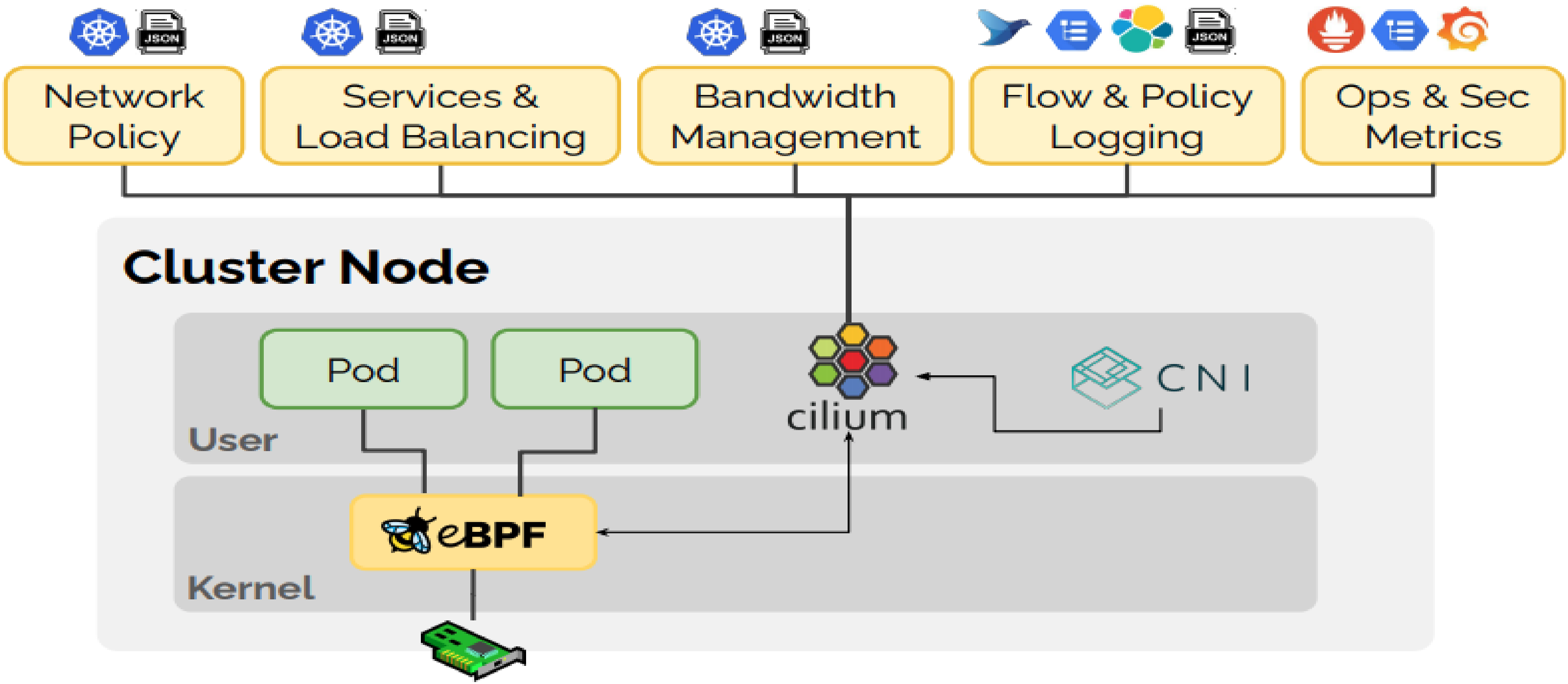
- What is cilium ?
- Why cilium ?
- Cilium components and datapath
- Demo

# That Cilium/ Cilia ?





# This cilium by isovalent



# What does it do ?

- Networking: Scalable CNI plugin, LB ( beta ) , bandwidth management ( Beta )
- Security: advance network policy ( L3-L7 ), transparent encryption
- Observability: Identity-aware observability
- CNI chaining
- Service-mesh ( beta )

# Why should we care ?

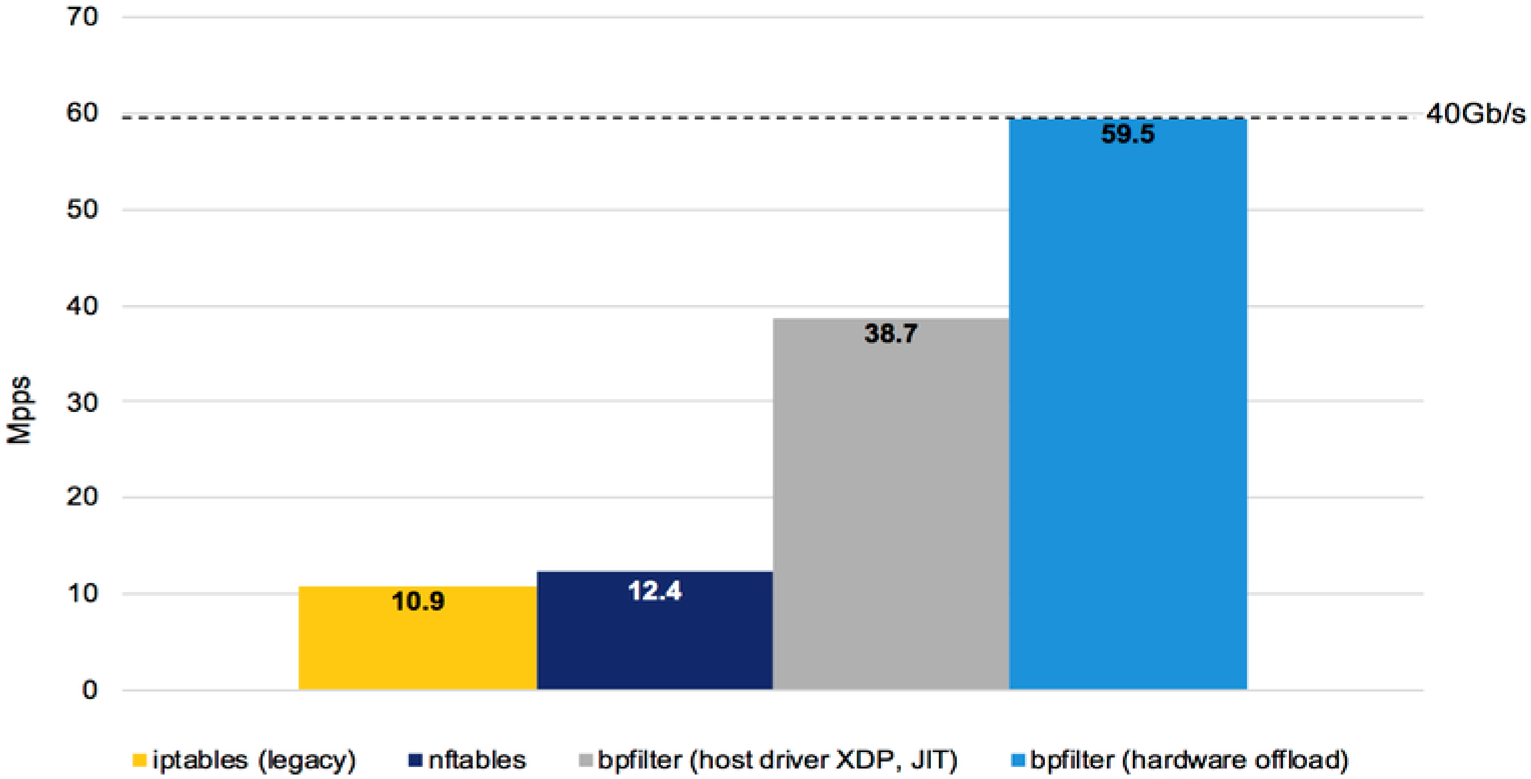
- High performance ( ebpf-based )
- Modern-feature rich for k8s workloads
- Iptables-based kube-proxy replacement

# Scalable solution to replace iptables-based kube-proxy

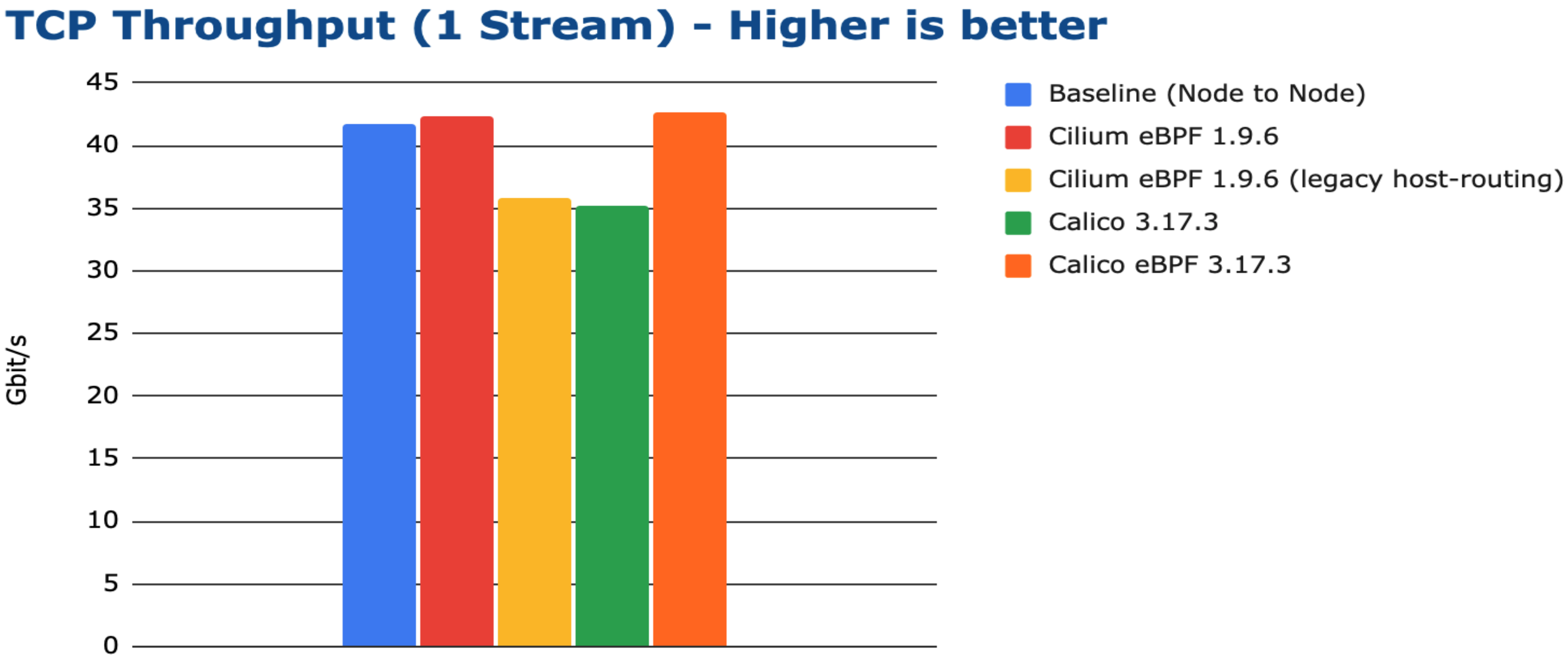
# Iptables as kube-proxy solution

- Not scalable:
    - Update requires recreating all rules in a single transaction [1]
    - More pod/svc, more rules, slower performance
- Not L7 aware

# Ebpf vs iptables packet filter benchmark [1]

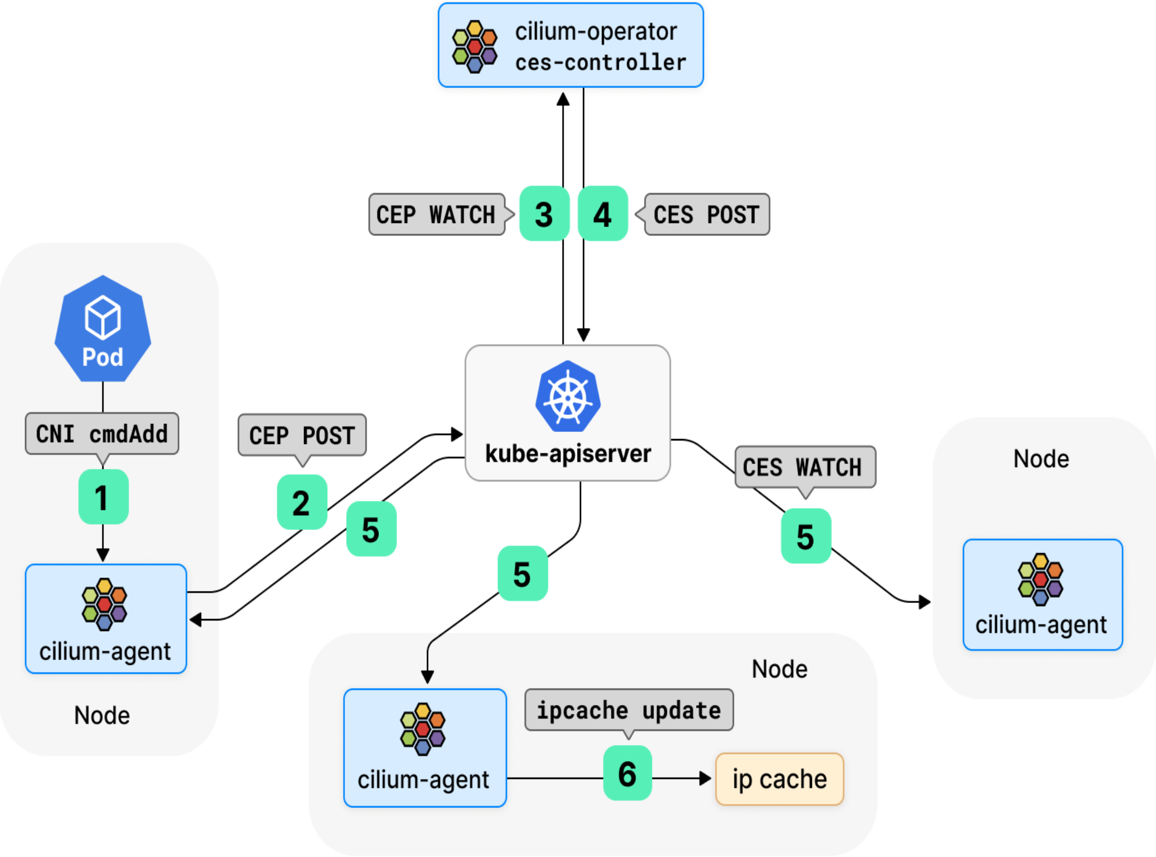


# CNI performance benchmark



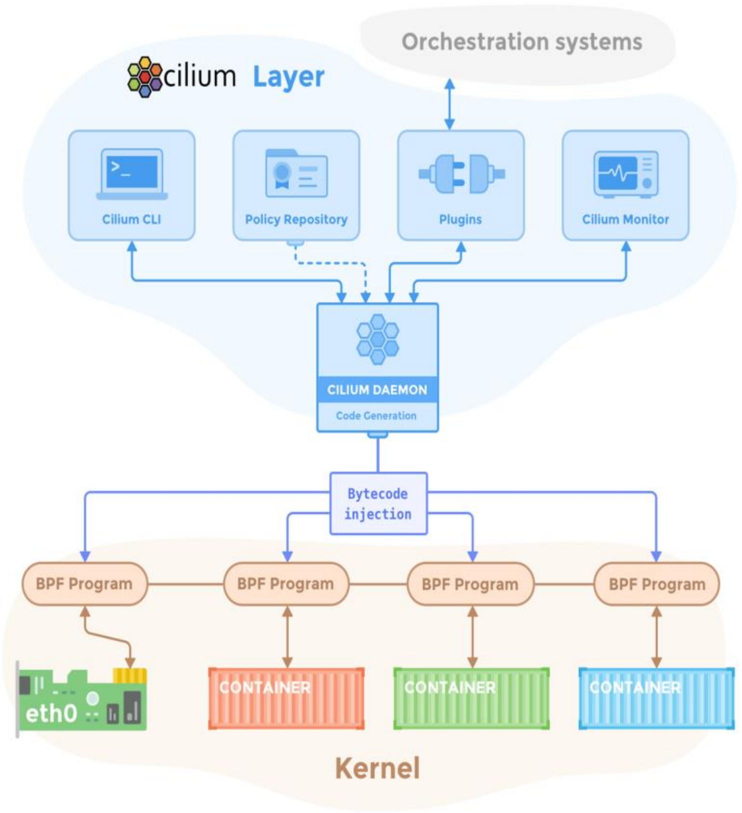
# Cilium Components

- Operator
- Agent
- ETCD storage
- CNI plugin
- Hubble as observe tool
- CLI



# Cilium agent

- Init BPF hooks , iptables rules
- Start built-in envoy
- Listen to CNI-plugin
- create k8s Cilium endpoints ( CEP )
- Update kvstore
- Watch cilium kv-store for other’s change
- Save/load/update BPF maps



# Cilium operator

- Watch k8s resources ( Services, nodes, Cilium endpoints / network policy )
- Synchronize them to cilium kvstore
- GC

# Cilium datapath

- Cilium agent doesn’t handle traffic itself.
- Real traffic:
  - Routing decision is based on iproute2 routing table for L2 interconnected hosts or BGP over BIRD or ( kube-router - beta )
  - BPF hooks to intercept traffic, use info in BPF map to decide ( DROP / ACCEPT / REDIRECT / LB )
  - Cilium envoy plugin handle L7 policy request with iptables packet marking and TPROXY support

# Cilium BPF programs

- Do all the heavy lifting with networking
- Was attached by cilium-agent to
  - XDP: Upon packet receiving/sending, device level ( lxc_*, cilium_* )
  - After sk_buffer is created, before L3 ( eg: for encryption )
  - Socket layer: upon establishment, send/receive packet
- Use BPF maps as input to process packet
- Use packet marking to interact with upper layer ( iptables TPROXY and envoy proxy )

# Simple Demo

- 2 cilium nodes with: Native routing, Host-reachable service
- Kubernetes IPAM, ETCD kv-store
- L7 network policy
- Available to the outside with static routing
- Hubble CLI & UI

# Thoughts

- Cilium vs our current calico pros and cons
- Should we switch to cilium now ?
- Its limits? [6]

# References

[1] https://www.netronome.com/blog/frnog-30-faster-networking-la-francaise/

[2] https://docs.cilium.io/en/stable/operations/performance/benchmark/

[3] https://www.slideshare.net/LCChina/scale-kubernetes-to-support-50000-services

[4] https://cilium.io/blog/2018/04/17/why-is-the-kernel-community-replacing-iptables

[5] https://docs.cilium.io/en/v1.11/concepts/#concepts

[6] https://docs.cilium.io/en/stable/operations/performance/scalability/report/