
KubeCon
CloudNativeCon
OPEN SOURCE SUMMIT
China 2023


KubeCon | CloudNativeCon

OPEN SOURCE SUMMIT

China 2023

不受CNI限制的eBPF网络性能加速器

Yizhou Xu, Intel
Mengxin Liu, Alauda

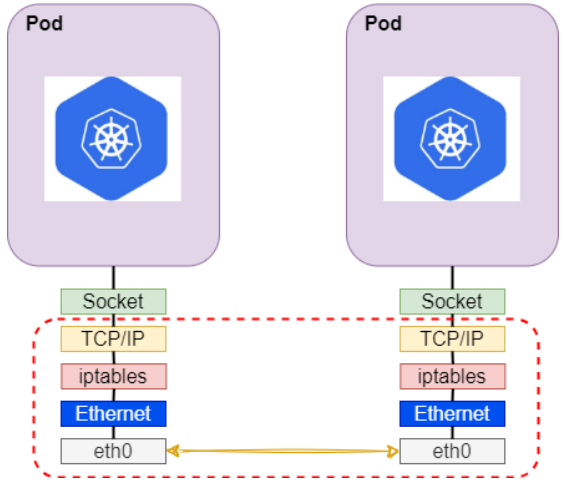
议程

- TCP/IP 网络协议栈的开销
- eBPF背景知识
- 如何使用eBPF绕过TCP/IP
- 性能分析
- KubeVirt的加速实践

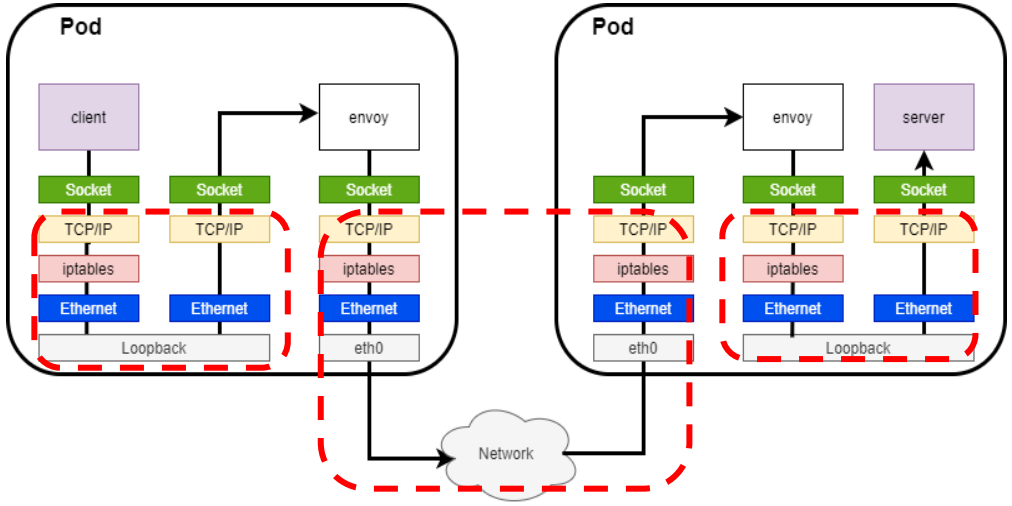
TCP/IP 网络协议栈的开销

在Kubernetes中,每个pod都有自己的网络协议栈, 数据包从一个pod到另一个pod时会经过整个协议栈多次

在服务网格场景中,此现象尤为严重



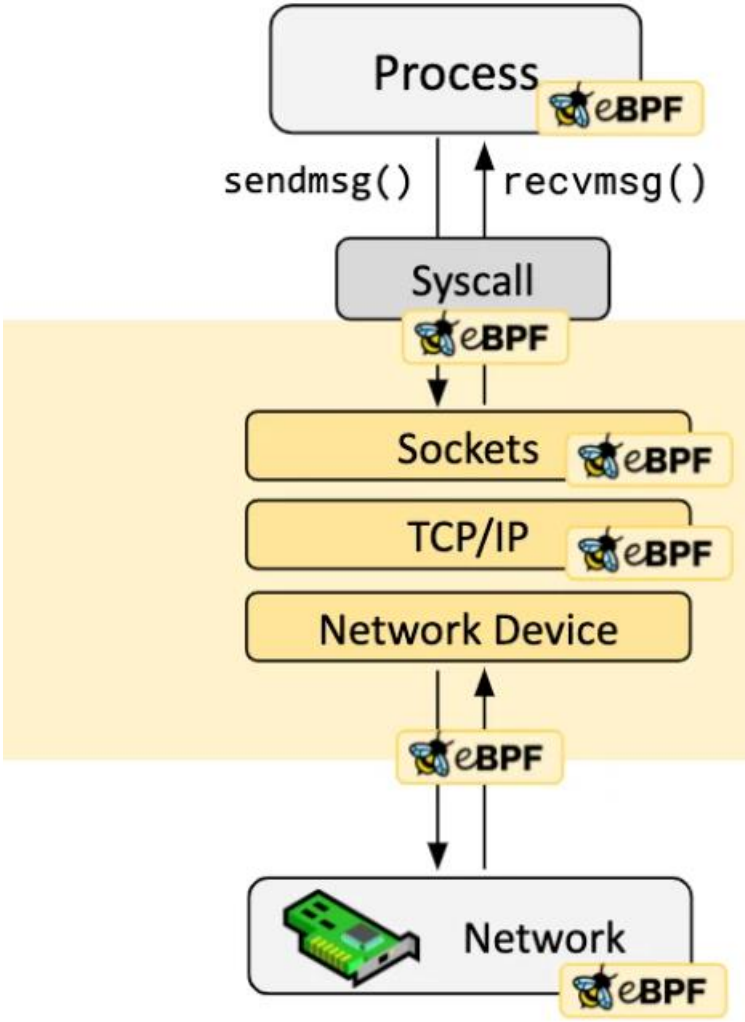
Tcp/ip overhead in Pod to Pod (same host)



Tcp/ip overhead in Service Mesh (same host)

选择eBPF的原因

- 工作在内核
- 非侵入式
- 不受CNI限制
- 安全高效



eBPF 背景知识– Map&Prog

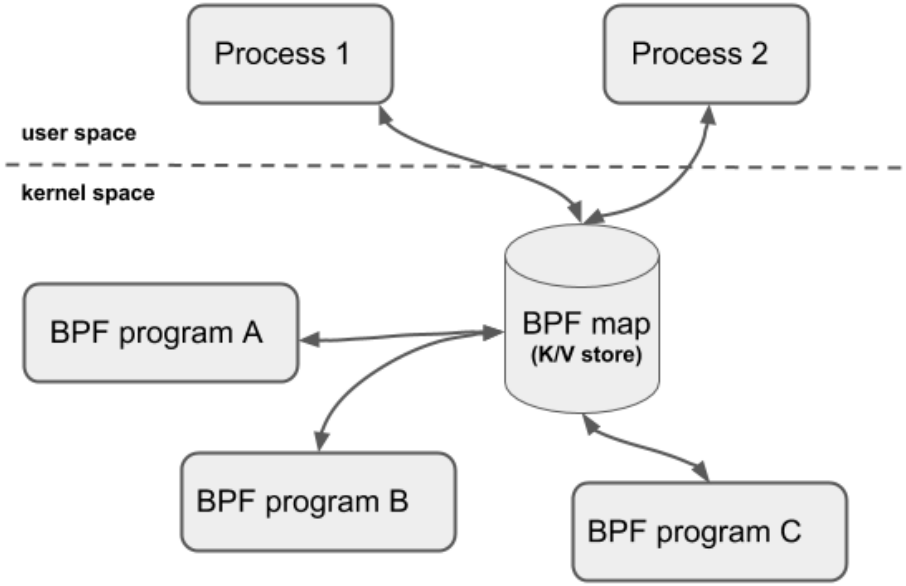
MAP

为用户态与内核态程序交互提供不同类型的通用数据结构,

- HASHMAP
- SOCKHASH

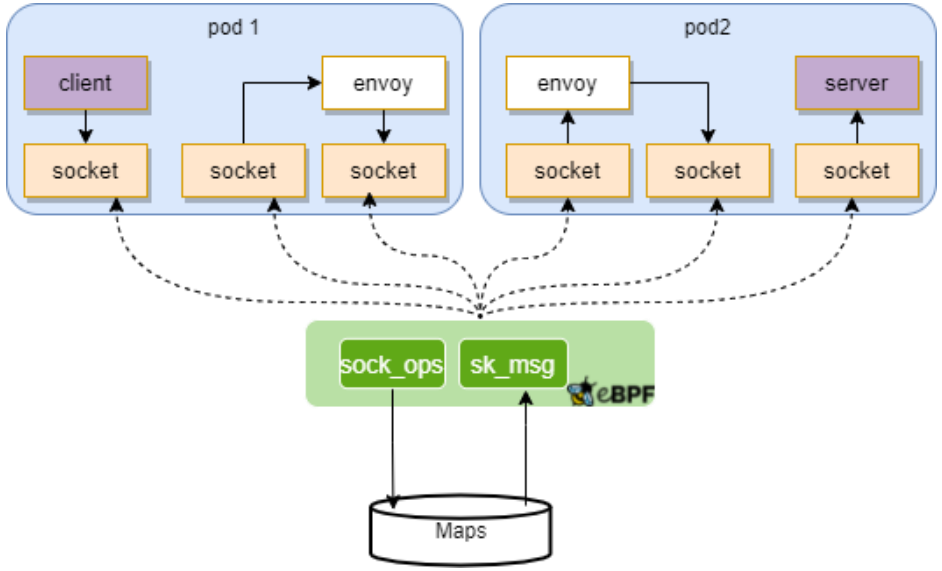
Program type

- SOCK_OPS
- SK_MSG

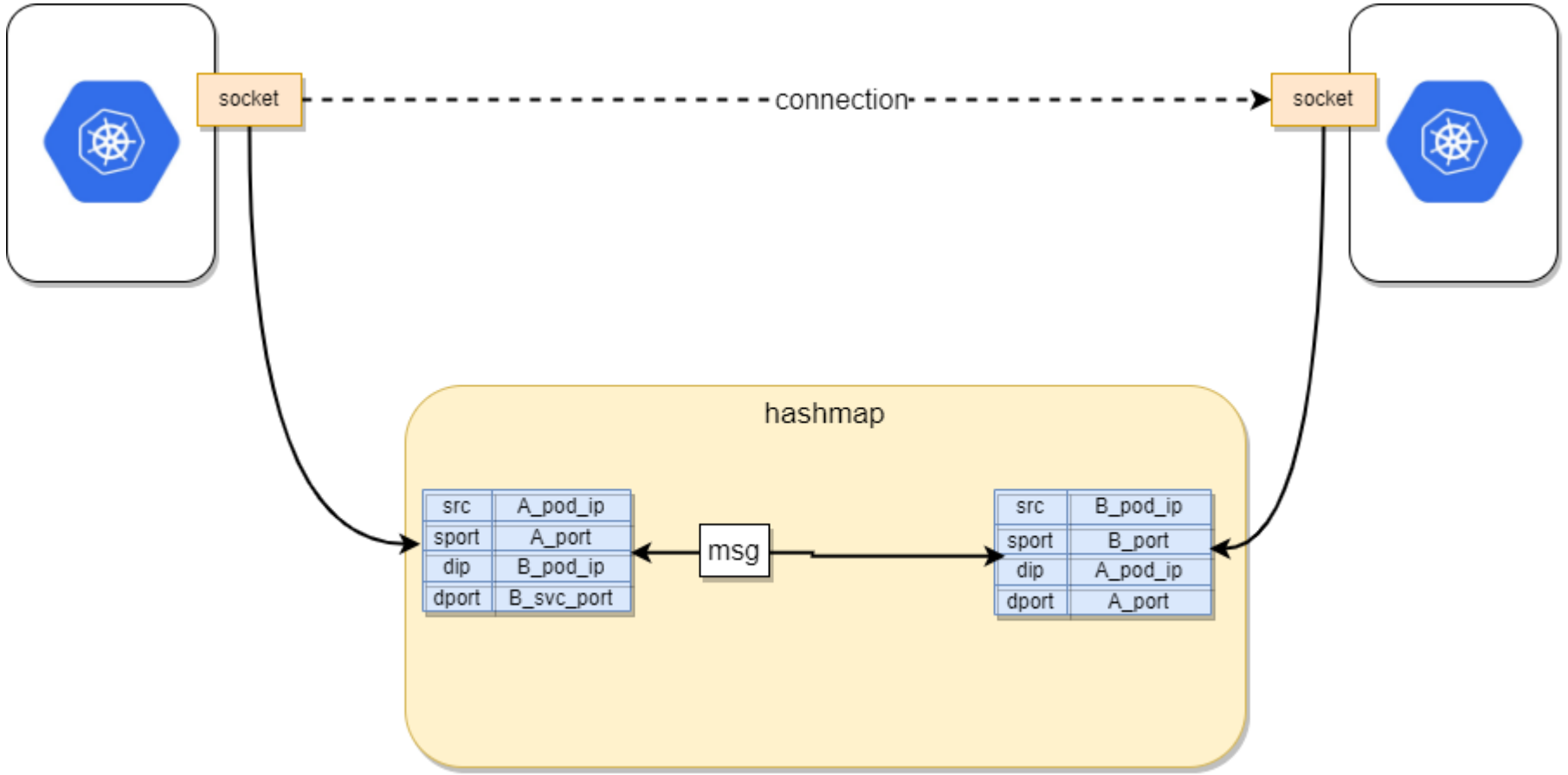


结合MAP与Program进行加速

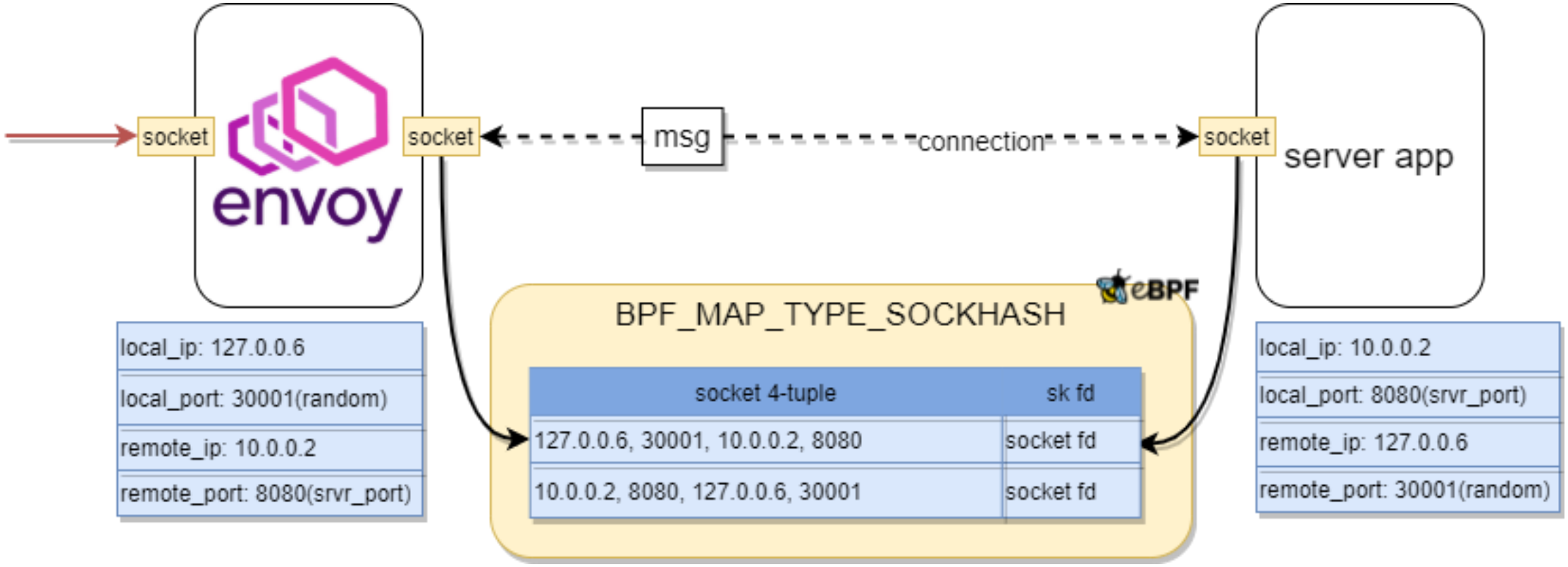
- sock_ops
  - 捕捉进入特殊状态的socket,并装入map中
- sk_msg
  - 当有socket有数据要发送,查找对端socket
  - 转发



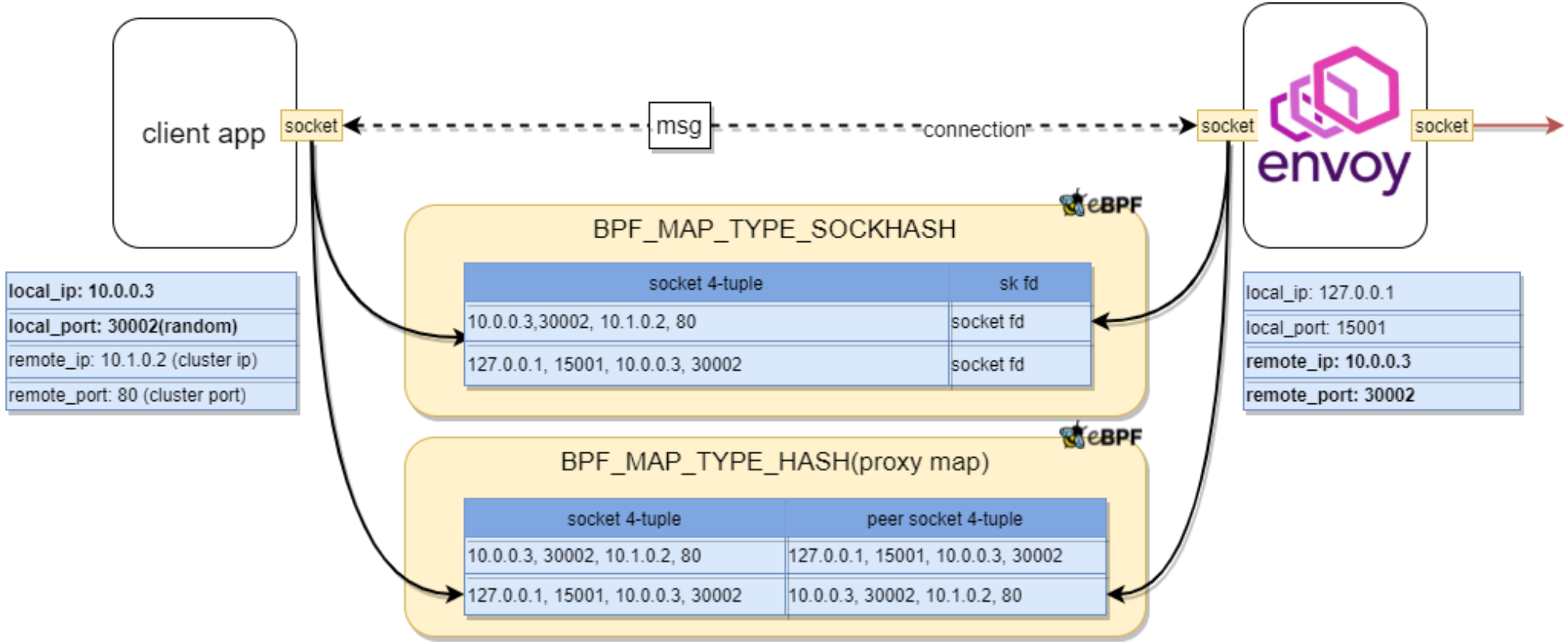
Pod 到 Pod 的加速(Kubernetes场景)



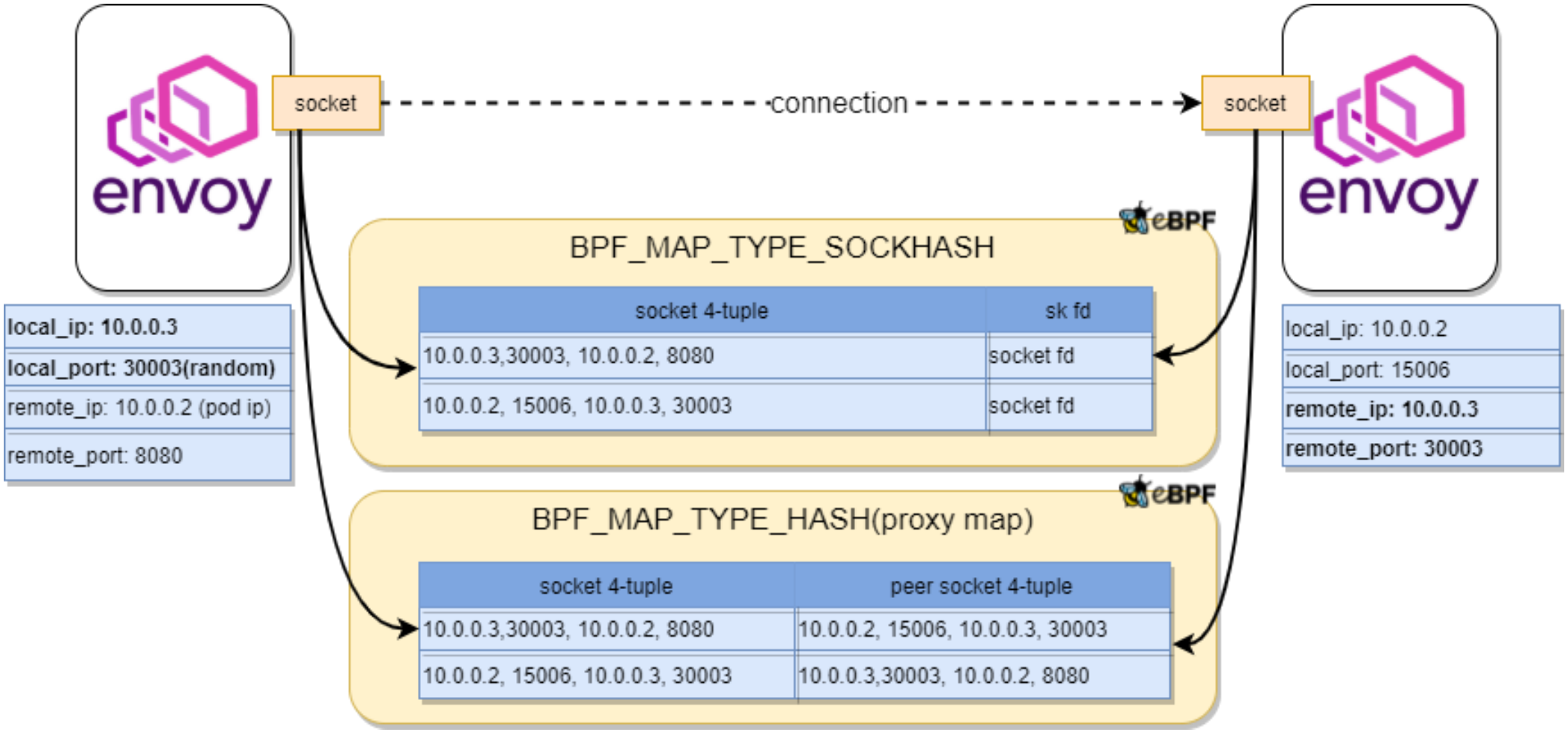
服务网格入方向加速


socket
envoy
socket
msg
connection
socket
server app
eBPF
BPF_MAP_TYPE_SOCKHASH
socket 4-tuple
sk fd
127.0.0.6, 30001, 10.0.0.2, 8080
socket fd
10.0.0.2, 8080, 127.0.0.6, 30001
socket fd
local_ip: 127.0.0.6
local_port: 30001(random)
remote_ip: 10.0.0.2
remote_port: 8080(srvr_port)
local_ip: 10.0.0.2
local_port: 8080(srvr_port)
remote_ip: 127.0.0.6
remote_port: 30001(random)

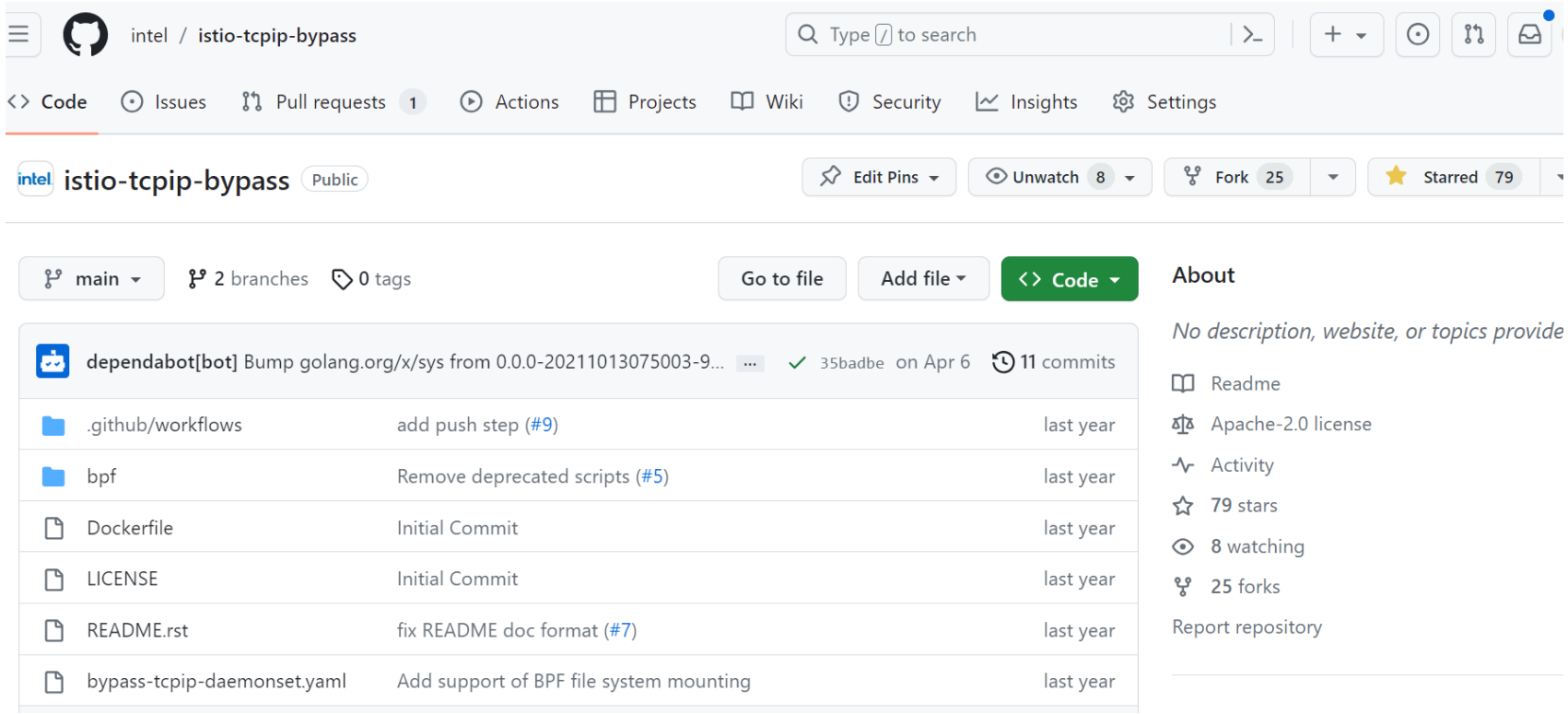
服务网格出方向加速


client app
socket
msg
connection
socket
envoy
socket
local_ip: 10.0.0.3
local_port: 30002(random)
remote_ip: 10.1.0.2 (cluster ip)
remote_port: 80 (cluster port)
eBPF
BPF_MAP_TYPE_SOCKHASH
socket 4-tuple
sk fd
10.0.0.3,30002, 10.1.0.2, 80
socket fd
127.0.0.1, 15001, 10.0.0.3, 30002
socket fd
local_ip: 127.0.0.1
local_port: 15001
remote_ip: 10.0.0.3
remote_port: 30002
eBPF
BPF_MAP_TYPE_HASH(proxy map)
socket 4-tuple
peer socket 4-tuple
10.0.0.3, 30002, 10.1.0.2, 80
127.0.0.1, 15001, 10.0.0.3, 30002
127.0.0.1, 15001, 10.0.0.3, 30002
10.0.0.3, 30002, 10.1.0.2, 80

服务网格Envoy间的加速 (位于同主机)



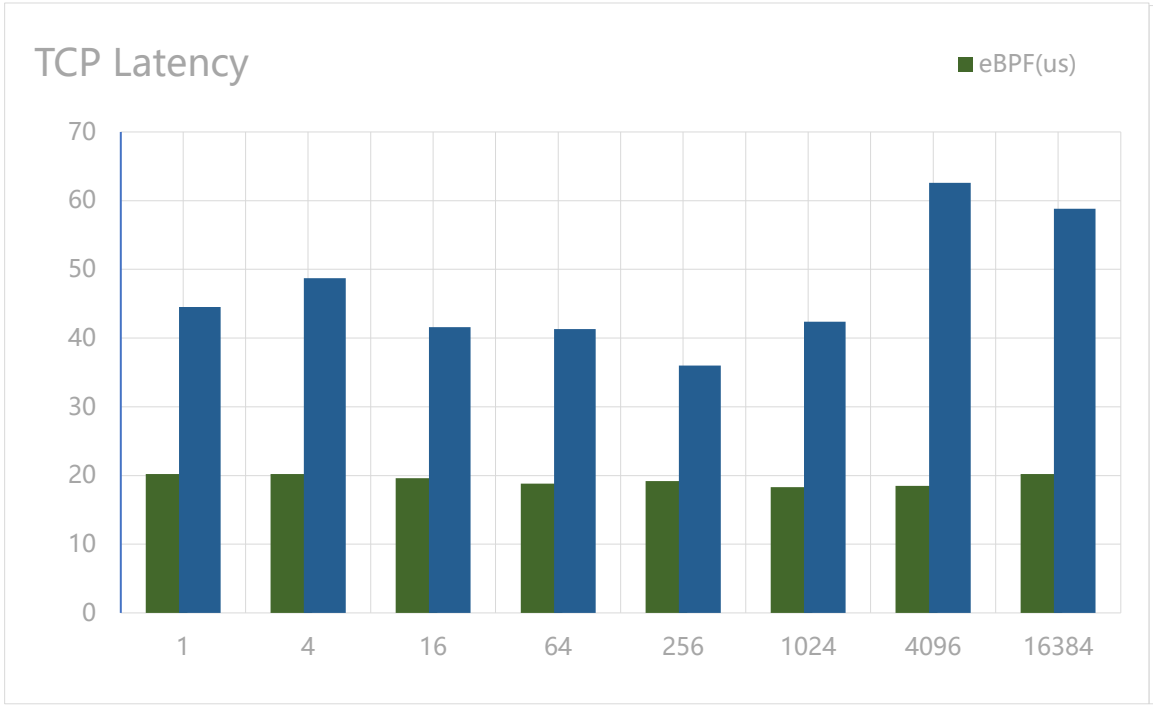
已开源,速来!

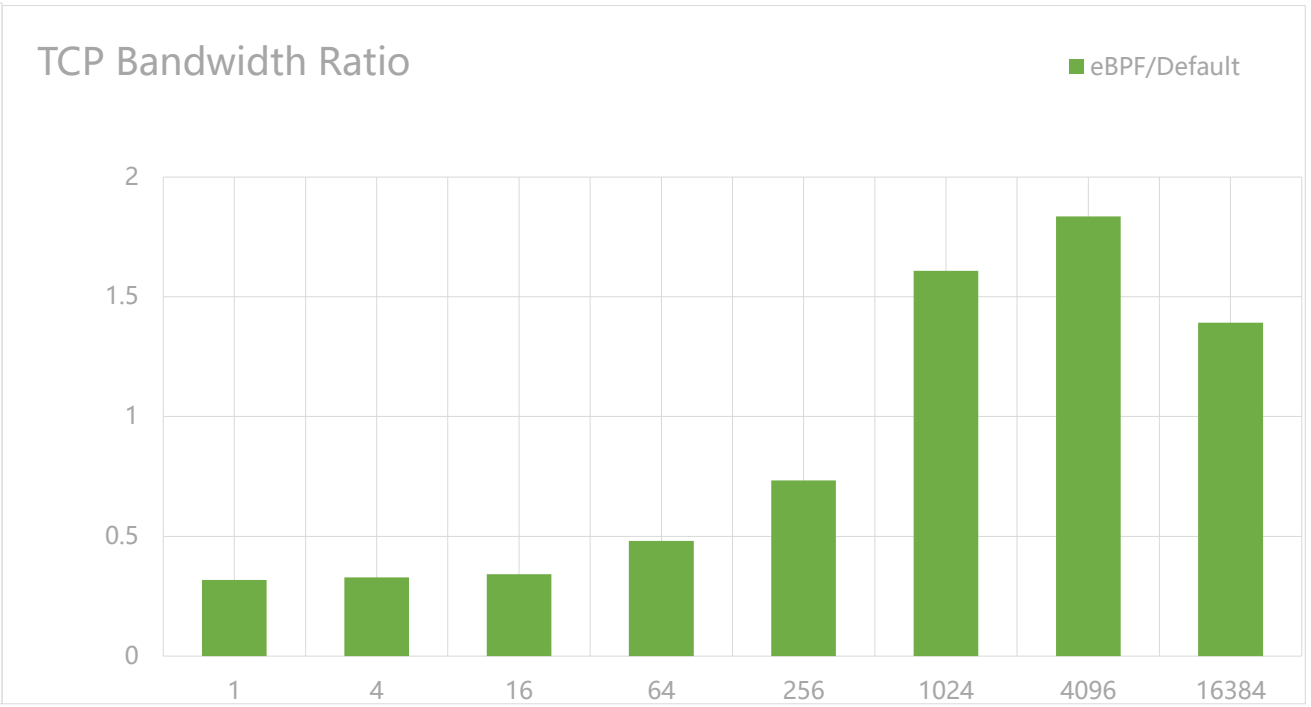


REPO: https://github.com/intel/istio-tcpip-bypass

性能测试

- 两个测试pod部署在同节点
- 使用 qperf 测试TCP 延迟/带宽, 包大小由 1byte 到 16KB
  - qperf -t 60 100.64.0.3 -ub -oo msg_size:1:16K:*4 -vu tcp_lat tcp_bw
- 对比未开启优化的数据



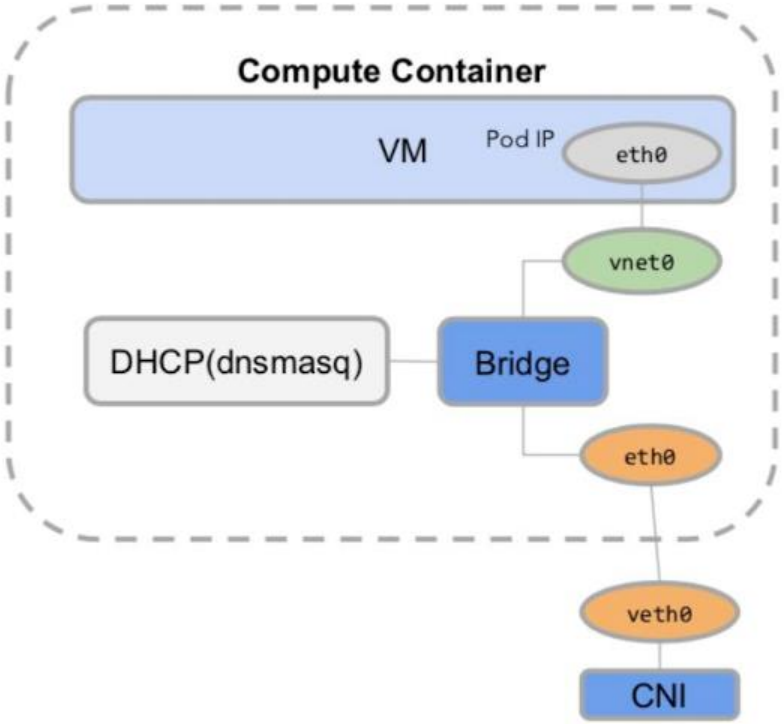


结果分析

- TCP 延迟降低 40% ~ 60%
- 在数据包大于1024 bytes情况下,吞吐增加 40% ~ 80%
- 由于无法 offload 需要逐包处理,在数据包小于512 bytes场景吞吐下降

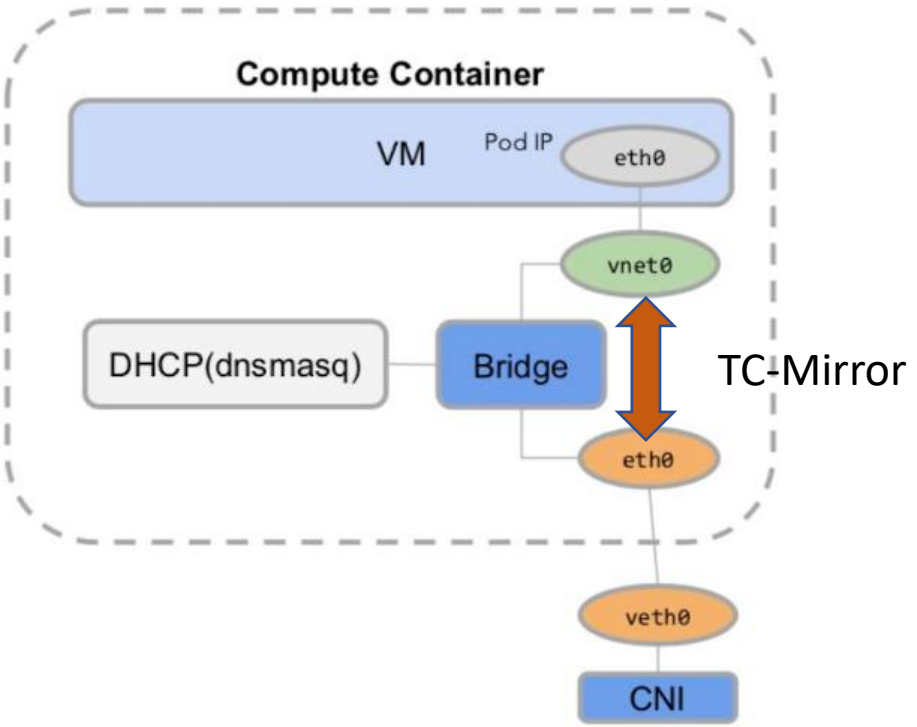
KubeVirt 网络性能

- KubeVirt 原生 Bridge 网络模式相比 Pod 网络性能有明显下降
  - 延迟增加 60%
  - PPS 下降 50%.
- 可能原因:
  - VM 内 Linux 网络栈的额外开销
  - KubeVirt Bridge 内额外的网络开销
- Kata 使用 tc-mirror 来跳过 Bridge 开销



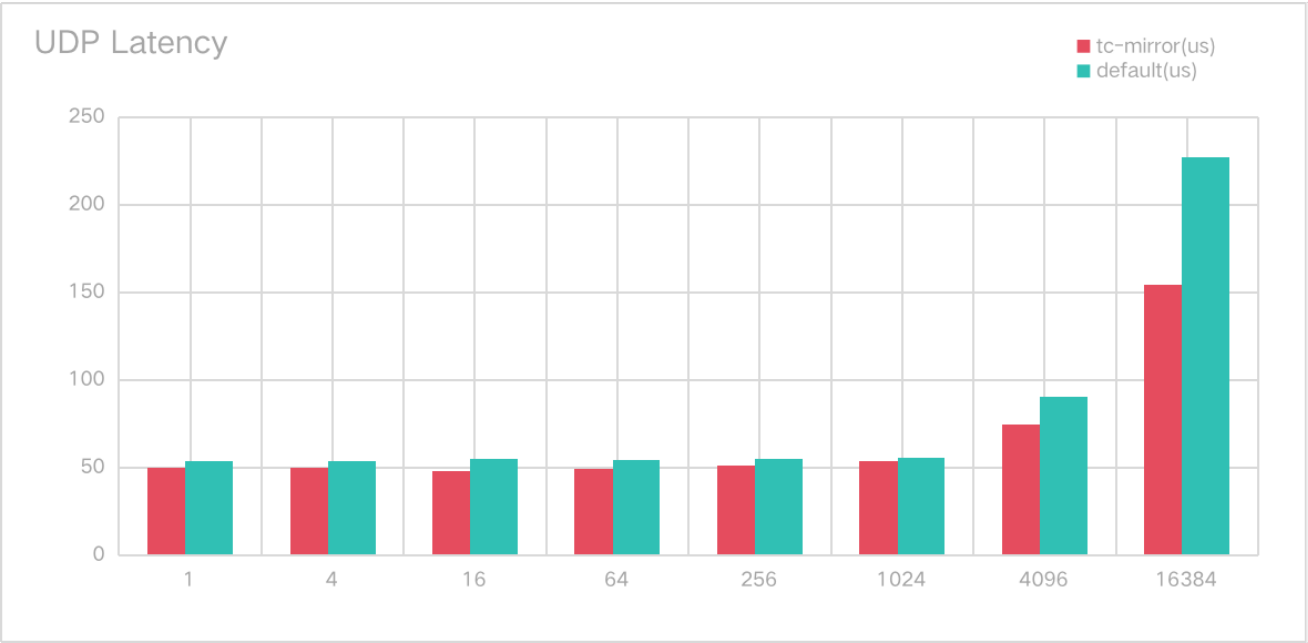
KuberVirt 网络加速

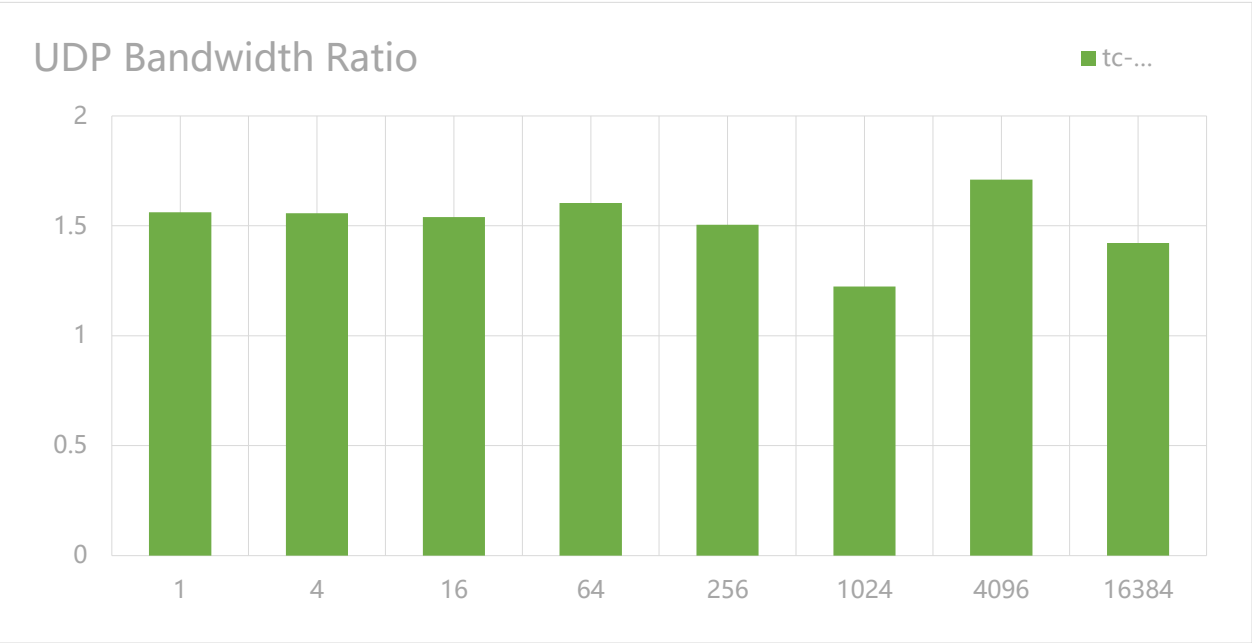
- 尝试使用 tc-mirror 和 bpf_redirect
  - tc-mirror 在绝大多数测试场景下性能略好于 bpf_redirect
  - tc-mirror 对内核版本要求较低
  - 目前没有找到从 VM eth0 直接短路到 veth0 的方法
  - 延迟下降 5% 但吞吐量也下降 20%
- 关闭 checksum
  - 对于机器内部的网络通信不需要进行 checksum 和 checksum 检查
  - 同时关闭 rx/tx 侧的checksum



性能测试

- 使用 qperf 进行 UDP 延迟和吞吐量测试， 包大小从 1byte 到 16KB
  - 延迟下降4%~30%
  - 吞吐量提升 20%~70%





未来展望

- 加速同节点 UDP 通信
- 细粒度控制选择 eBPF datapath 或者 kernel datapath
- 加速同节点内 Kubevirt VMs 通信

通知和免责声明

Intel technologies may require enabled hardware, software or service activation.
No product or component can be absolutely secure.
Your costs and results may vary.