第三届 eBPF开发者大会

www.ebpftravel.com

Cilium:基于eBPF的Kubernetes CNI

By 张子健

中国·西安

0. 个人介绍

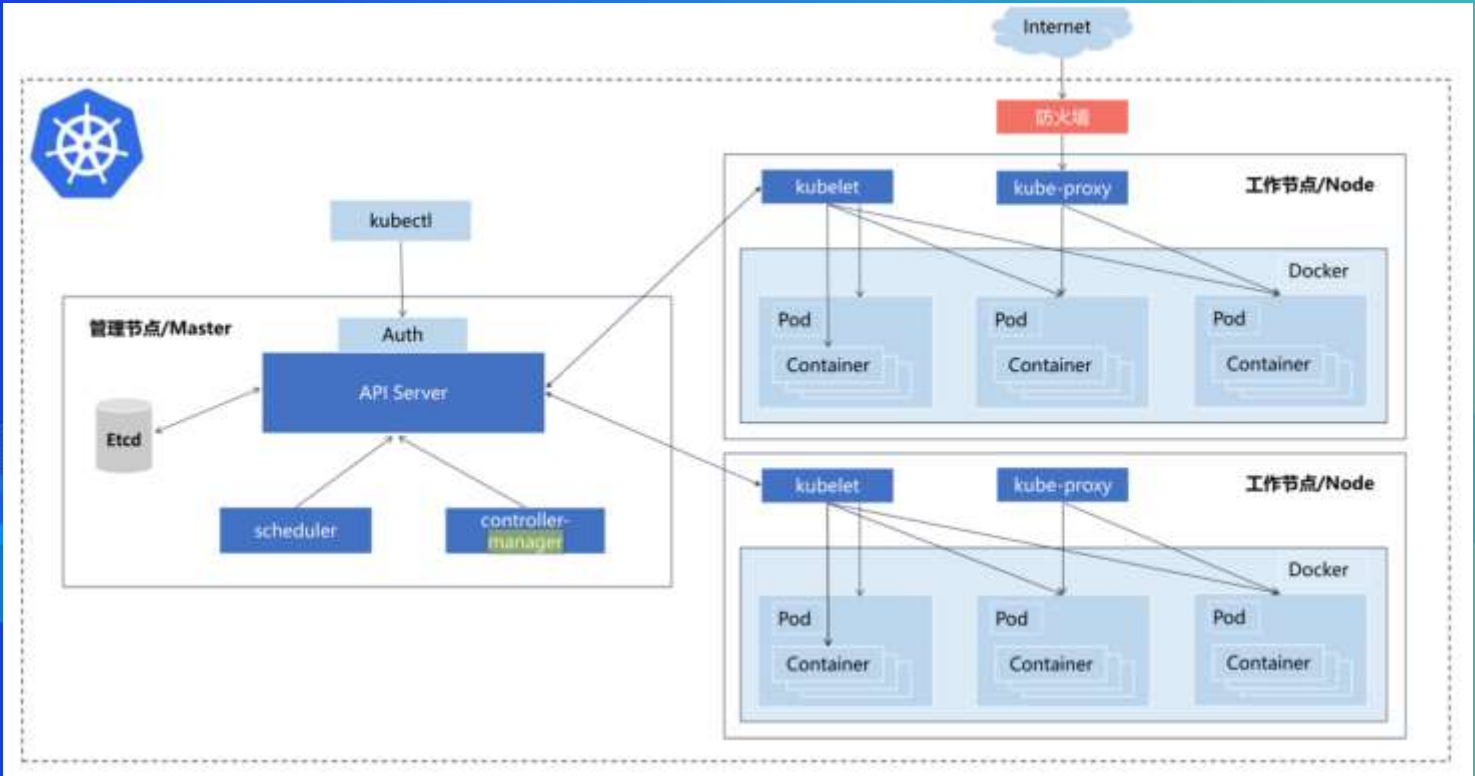
- 2020年获得武汉大学计算机科学学士学位
- 2022年获得哥伦比亚大学计算机科学硕士学位, 主要研究方向是Linux内核
- 现就职于字节跳动，主要工作内容是Linux内核网络相关以及eBPF
- Linux Netdev/eBPF模块以及Cilium开源贡献者

目录

1. K8S

Kubernetes（简称 K8s）是一个由 Google 开发、现由 CNCF 维护的开源容器编排平台，用于自动化容器化应用的部署、扩展和管理，帮助实现高效、可扩展、稳定的应用运行环境。

1. K8S 集群

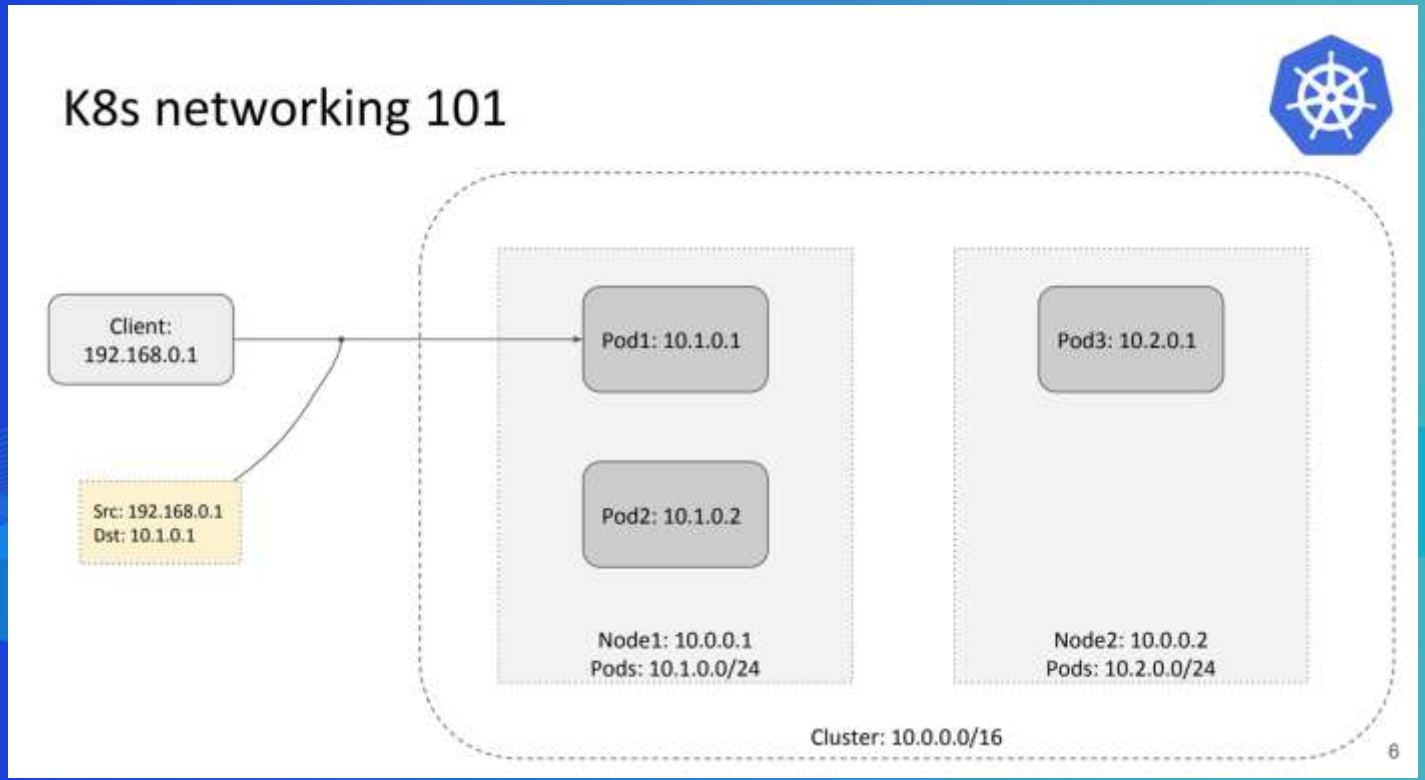

Internet
防火墙
kubectl
管理节点/Master
Auth
API Server
Etcd
scheduler
controller-manager
kubelet
kube-proxy
工作节点/Node
Docker
Pod
Container
Pod
Container
Pod
Container
kubelet
kube-proxy
工作节点/Node
Docker
Pod
Container
Pod
Container
Pod
Container

1. K8S 网络模型

K8S 定义了一种简单、一致的网络模型，基于扁平网络结构的设计

1. 每个 Node 和 Pod 都有自己的 IP 地址，这个 IP 在集群范围内可达
2. 任意Node和Node，Pod和Pod，Node和Pod之间有连通性，而无需任何NAT
3. Kubernetes 的组件之间可以相互通信，也可以与 Pod 通信

1. K8S 网络模型


K8s networking 101
Client:
192.168.0.1
Src: 192.168.0.1
Dst: 10.1.0.1
Pod1: 10.1.0.1
Pod2: 10.1.0.2
Pod3: 10.2.0.1
Node1: 10.0.0.1
Pods: 10.1.0.0/24
Node2: 10.0.0.2
Pods: 10.2.0.0/24
Cluster: 10.0.0.0/16
6

2. CNI

CNI 是 Container Network Interface 的缩写，是一个用于配置 Linux 容器网络的标准接口规范。它是 CNCF（Cloud Native Computing Foundation）下的一个项目。K8S本身不实现网络模型而是由CNI实现。

当一个容器启动时，CNI 负责给这个容器分配网络、设置 IP、连接到网络桥、配置路由等事情，确保容器之间以及容器与外部世界之间能通信。

2. CNI组件组成

1. CNI 规范：定义了插件之间如何交互的规则。

1. CNI 配置文件：定义插件的使用方式和参数

1. CNI 插件：按照规范实现的网络插件，不同的插件有不同的特性，比如：Calico、Flannel、Cilium

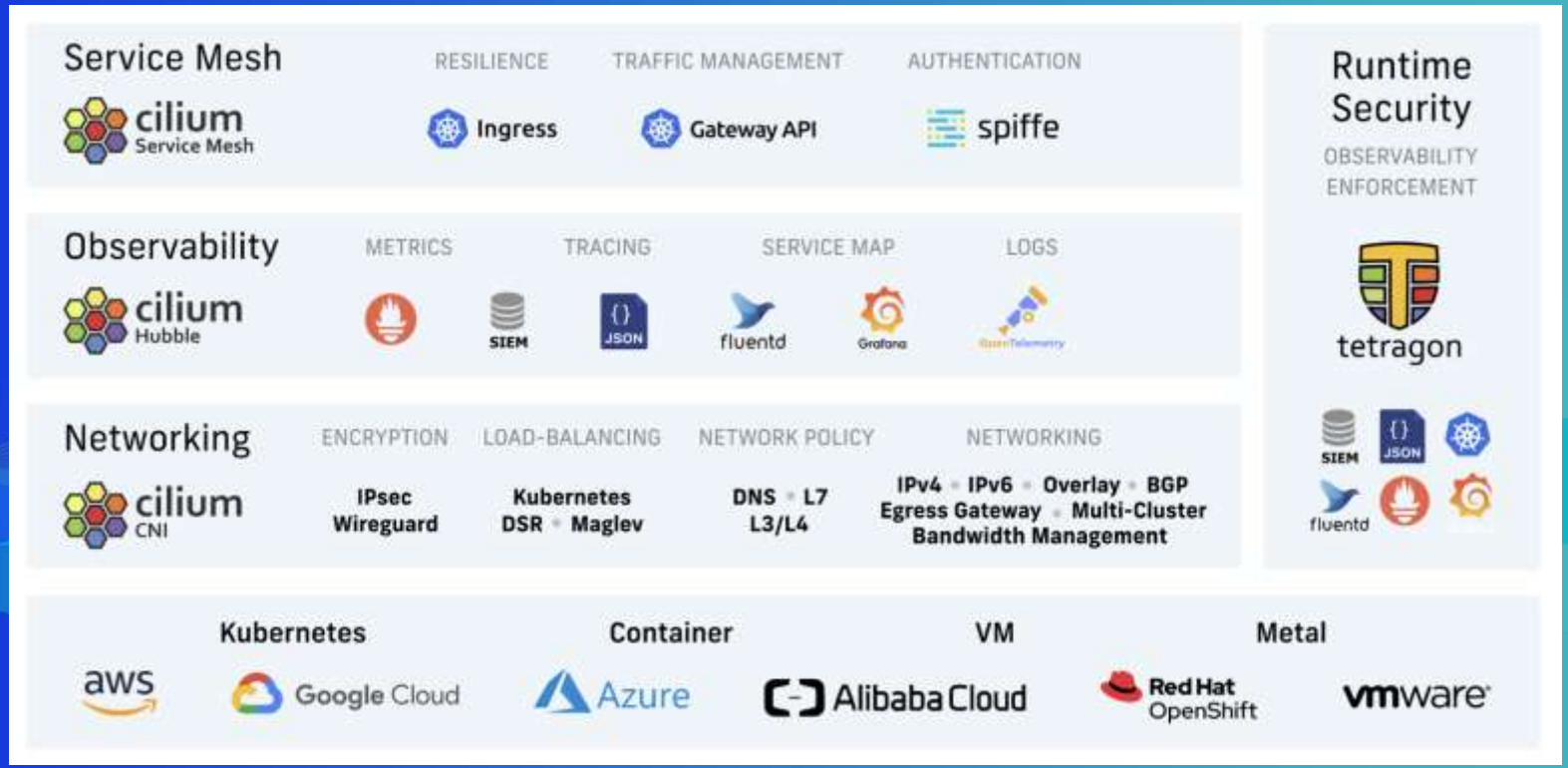
2. CNI插件举例 - Flannel

Flannel 是一个较为轻量级的 CNI 插件，专注于为 Kubernetes 提供基础的容器网络功能。它使用 overlay 网络来实现 Pod 之间的跨主机通信，用UDP包来封装原始数据包。Flannel 的架构简单，部署容易，适合对网络功能需求不高的小型或开发环境。但由于其主要定位是网络互通，缺乏复杂的网络策略控制、安全特性以及高级网络功能。

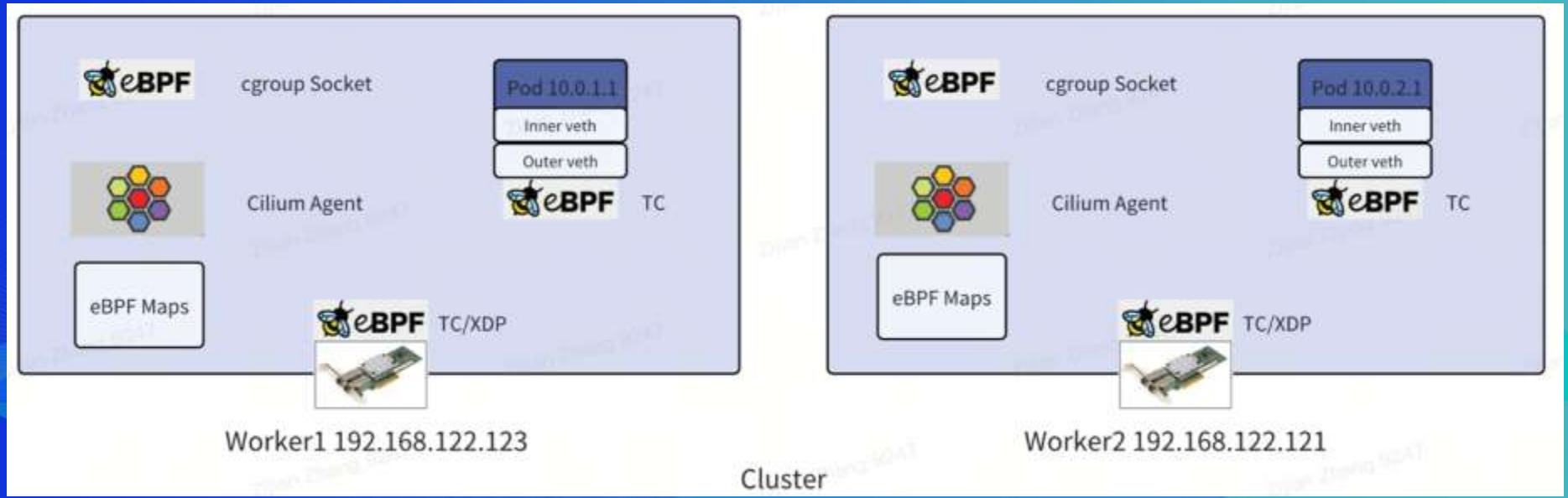
3. Cilium概述

Cilium 是一个基于 eBPF 的 CNI 插件。它不仅提供基础的容器网络功能，还支持 L7（应用层）可观测性与策略控制（比如对 HTTP、gRPC 等协议的可视化和控制），并与 Service Mesh、API 安全、透明加密等功能高度集成。得益于 eBPF，Cilium 在性能、扩展性和可调试性方面具有显著优势。

3. Cilium概述


Service Mesh
RESILIENCE
TRAFFIC MANAGEMENT
AUTHENTICATION
cilium Service Mesh
Ingress
Gateway API
spiffe
Observability
METRICS
TRACING
SERVICE MAP
LOGS
cilium Hubble
SIEM
JSON
fluentd
Grafana
Networking
ENCRYPTION
LOAD-BALANCING
NETWORK POLICY
NETWORKING
cilium CNI
IPsec
Wireguard
Kubernetes
DSR • Maglev
DNS • L7
L3/L4
IPv4 • IPv6 • Overlay • BGP
Egress Gateway • Multi-Cluster
Bandwidth Management
Runtime Security
OBSERVABILITY
ENFORCEMENT
tetragon
SIEM
JSON
fluentd
Kubernetes
Container
VM
Metal
aws
Google Cloud
Azure
Alibaba Cloud
Red Hat OpenShift
vmware

3. Cilium概述


eBPF
cgroup Socket
Pod 10.0.1.1
Inner veth
Outer veth
Cilium Agent
eBPF
TC
eBPF Maps
eBPF
TC/XDP
Worker1 192.168.122.123
eBPF
cgroup Socket
Pod 10.0.2.1
Inner veth
Outer veth
Cilium Agent
eBPF
TC
eBPF Maps
eBPF
TC/XDP
Worker2 192.168.122.121
Cluster

3. Cilium概述

```
root@master:~# bpftool net show
xdp:

tc:
ens2(2) clsact/ingress bpf_netdev_ens2.o:[from-netdev] id 25490
ens2(2) clsact/egress bpf_netdev_ens2.o:[to-netdev] id 25499
cilium_net(5) clsact/ingress bpf_host_cilium_net.o:[to-host] id 25481
cilium_host(6) clsact/ingress bpf_host.o:[to-host] id 25463
cilium_host(6) clsact/egress bpf_host.o:[from-host] id 25472
lxce26991a03789(13) clsact/ingress bpf_lxc.o:[from-container] id 25438
lxc501bb35ff726(15) clsact/ingress bpf_lxc.o:[from-container] id 25449
lxc_health(89) clsact/ingress bpf_lxc.o:[from-container] id 25450

flow_dissector:
```

3. Cilium概述

```
root@master:~# bpftool map show
115: hash  flags 0x1
        key 20B  value 48B  max_entries 65535  memlock 8916992B
116: lpm_trie  flags 0x1
        key 24B  value 12B  max_entries 512000  memlock 36868096B
118: percpu_hash  name cilium_metrics  flags 0x1
        key 8B  value 16B  max_entries 1024  memlock 344064B
119: hash  flags 0x1
        key 20B  value 20B  max_entries 65536  memlock 7344128B
120: lru_hash  flags 0x0
        key 16B  value 8B  max_entries 73603  memlock 7401472B
121: hash  flags 0x1
        key 12B  value 12B  max_entries 65536  memlock 6295552B
122: hash  flags 0x1
        key 4B  value 8B  max_entries 65536  memlock 5246976B
123: hash  flags 0x1
        key 2B  value 6B  max_entries 65536  memlock 5246976B
124: perf_event_array  flags 0x0
        key 4B  value 4B  max_entries 16  memlock 4096B
125: perf_event_array  flags 0x0
        key 4B  value 4B  max_entries 16  memlock 4096B
126: prog_array  flags 0x0
        key 4B  value 4B  max_entries 65535  memlock 528384B
        owner_prog_type sched_cls  owner jited
```

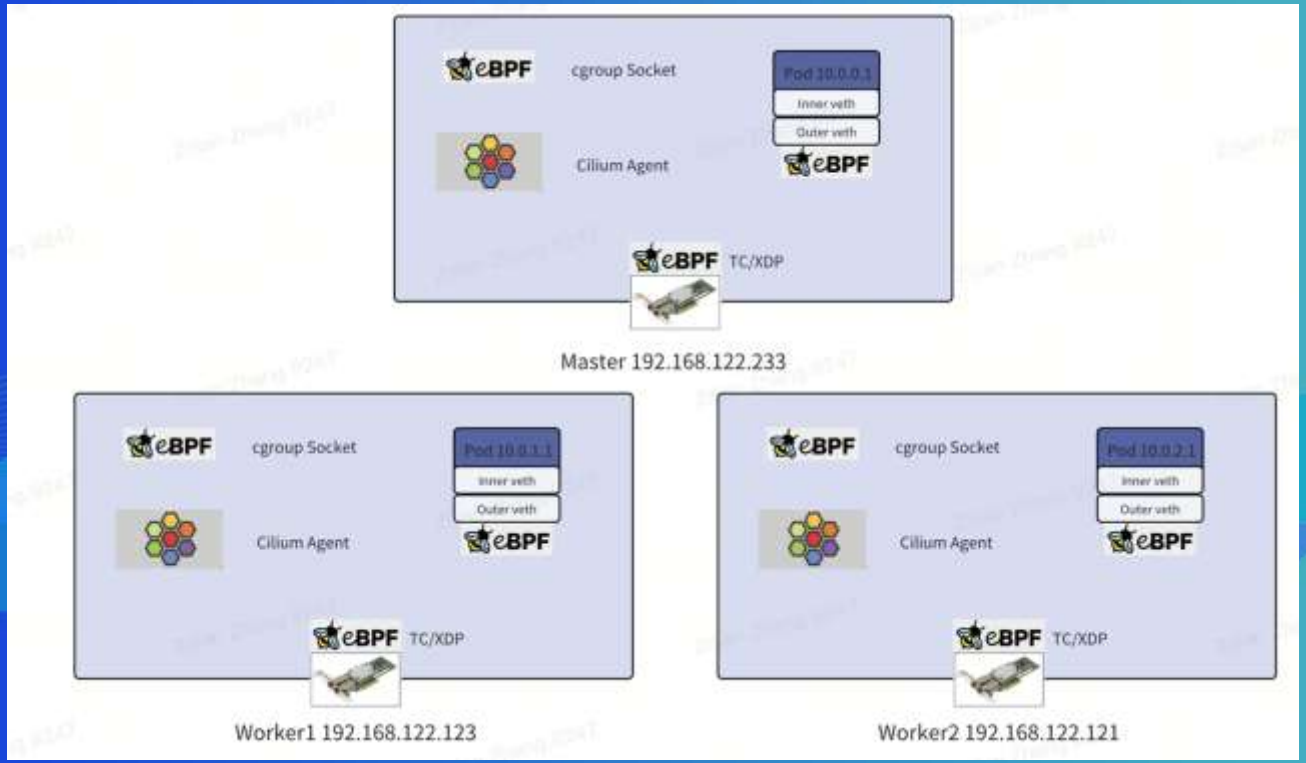
3. Cilium概述

```
root@master:~# bpftool cgroup tree
CgroupPath
ID       AttachType      AttachFlags     Name
/sys/fs/cgroup
25405    connect4                        sock4_connect
25385    connect6                        sock6_connect
25413    post_bind4                      sock4_post_bind
25393    post_bind6                      sock6_post_bind
25417    sendmsg4                        sock4_sendmsg
25397    sendmsg6                        sock6_sendmsg
25421    recvmsg4                        sock4_recvmsg
25401    recvmsg6                        sock6_recvmsg
25409    getpeername4                    sock4_getpeerna
25389    getpeername6                    sock6_getpeerna
```

3. Cilium概述 - 配置

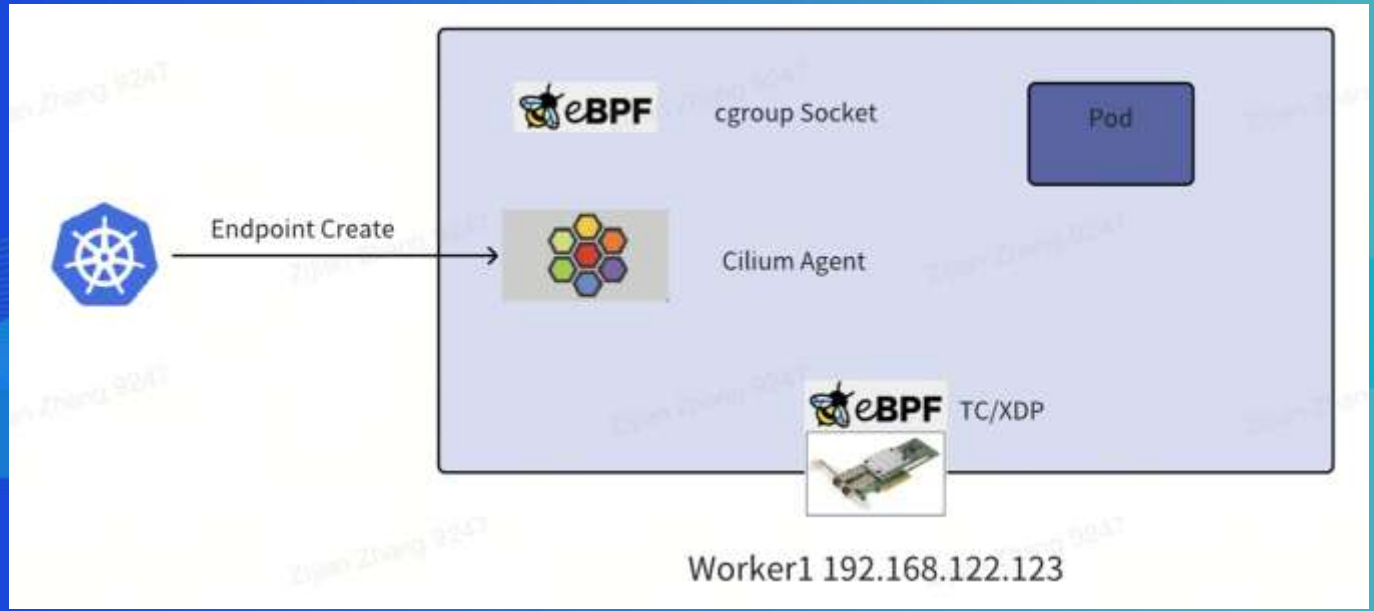
```
helm install cilium cilium/cilium --version 1.11.4 \
    --namespace kube-system \
    --set k8sServiceHost=${master_node_ip} \
    --set k8sServicePort=6443 \
    --set kubeProxyReplacement=strict \
    --set tunnel=disabled \
    --set autoDirectNodeRoutes=true \
    --set ipv4.enabled=true \
    --set ipv4NativeRoutingCIDR="10.0.0.0/16" \
    --set ipv6.enabled=false
```

3. Cilium概述 - 实验环境



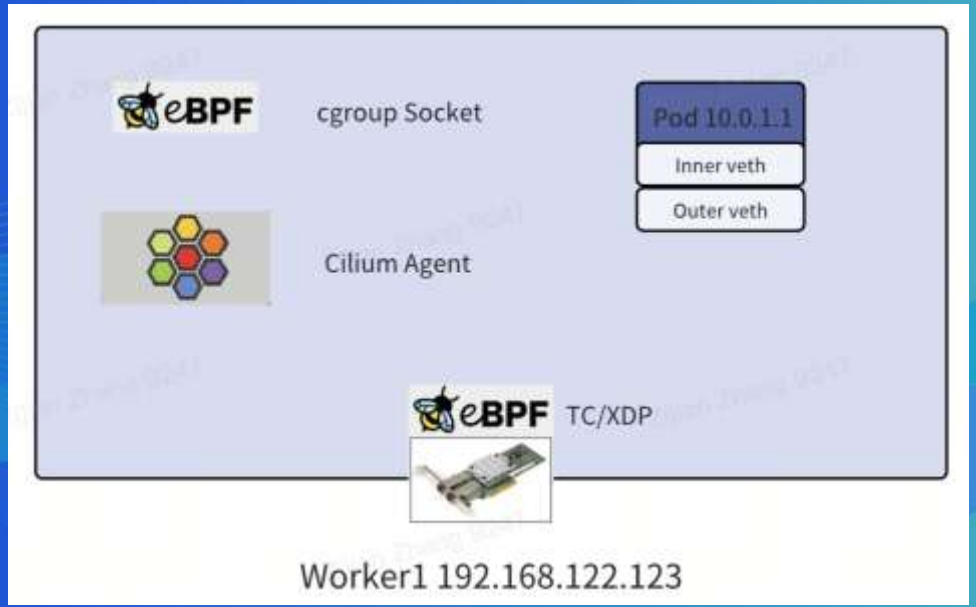
4. Cilium对K8S网络模型的实现 - Pod的创建

I. K8S给Cilium Agent发送一个“Endpoint Create”的请求



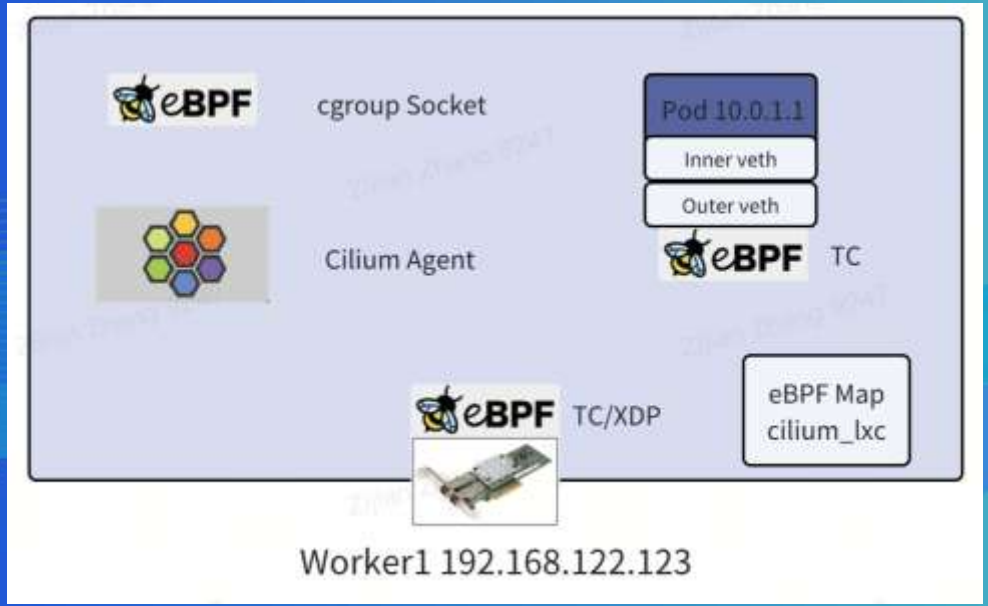
4. Cilium对K8S网络模型的实现 - Pod的创建

II. Cilium Agent为Pod创建veth对, 并且通过IPAM模块分配IP



4. Cilium对K8S网络模型的实现 - Pod的创建

III. Cilium Agent为Pod的外层veth设备附加TC eBPF程序,

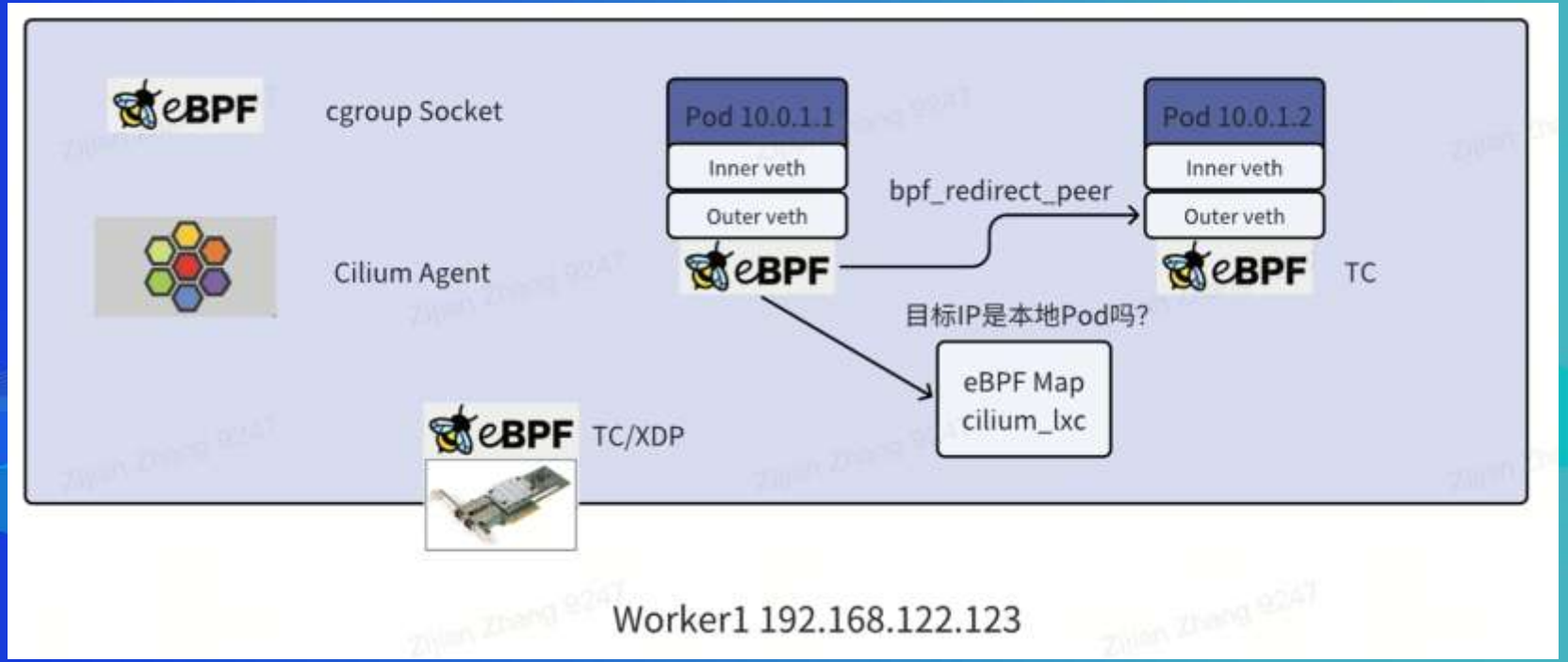


4. Cilium对K8S网络模型的实现 - Pod的创建

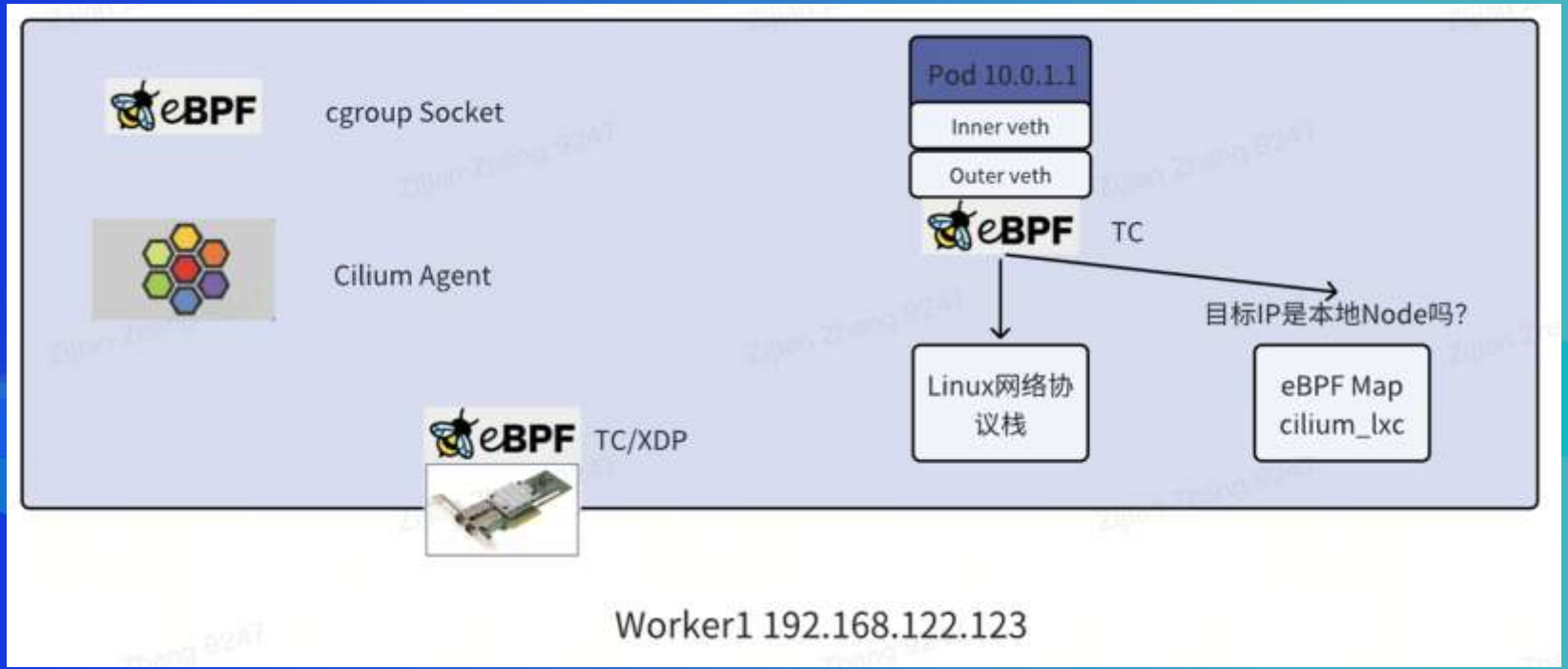
IIII. Cilium Agent更新相关eBPF Maps, 以记录与Pod相关的信息

```
root@master:/home/cilium# cilium map get cilium_lxc
Key             Value                                                                                  State   Error
10.0.0.63:0     id=811   flags=0x0000 ifindex=13  mac=8A:31:BC:72:83:96 nodemac=BE:7A:56:91:01:F7   sync
10.0.0.200:0    id=2405  flags=0x0000 ifindex=89  mac=CE:19:FC:21:7B:9D nodemac=C2:35:78:6C:D3:3D   sync
10.0.0.115:0    id=707   flags=0x0000 ifindex=15  mac=22:C1:1A:BF:14:60 nodemac=D6:D4:58:D4:60:97   sync
root@master:/home/cilium# cilium map get cilium_ipcache
Key                 Value                 State   Error
192.168.122.123/32  6 0 0.0.0.0           sync
192.168.122.233/32  1 0 0.0.0.0           sync
0.0.0.0/0           2 0 0.0.0.0           sync
10.0.1.132/32       4 0 192.168.122.123   sync
10.0.2.236/32       6 0 192.168.122.121   sync
10.0.0.141/32       1 0 0.0.0.0           sync
10.0.0.115/32       28723 0 0.0.0.0       sync
10.0.1.150/32       6 0 192.168.122.123   sync
192.168.122.121/32  6 0 0.0.0.0           sync
10.0.2.84/32        4 0 192.168.122.121   sync
10.0.0.63/32        28723 0 0.0.0.0       sync
10.0.0.200/32       4 0 0.0.0.0           sync
```

4. Cilium对K8S网络模型的实现 - 本地 Pod to Pod

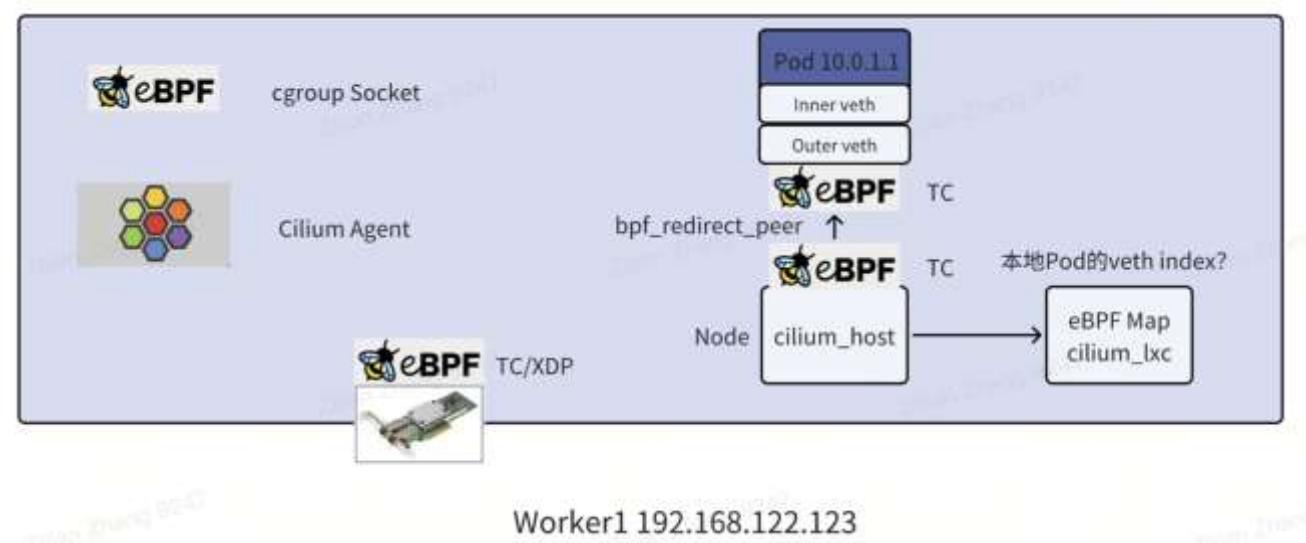


4. Cilium对K8S网络模型的实现 - 本地 Pod to Node



4. Cilium对K8S网络模型的实现 - 本地 Node to Pod

```
root@master:/home/cilium# ip r
default via 192.168.122.1 dev ens2
10.0.0.0/24 via 10.0.0.141 dev cilium_host src 10.0.0.141
10.0.0.141 dev cilium_host scope link
```

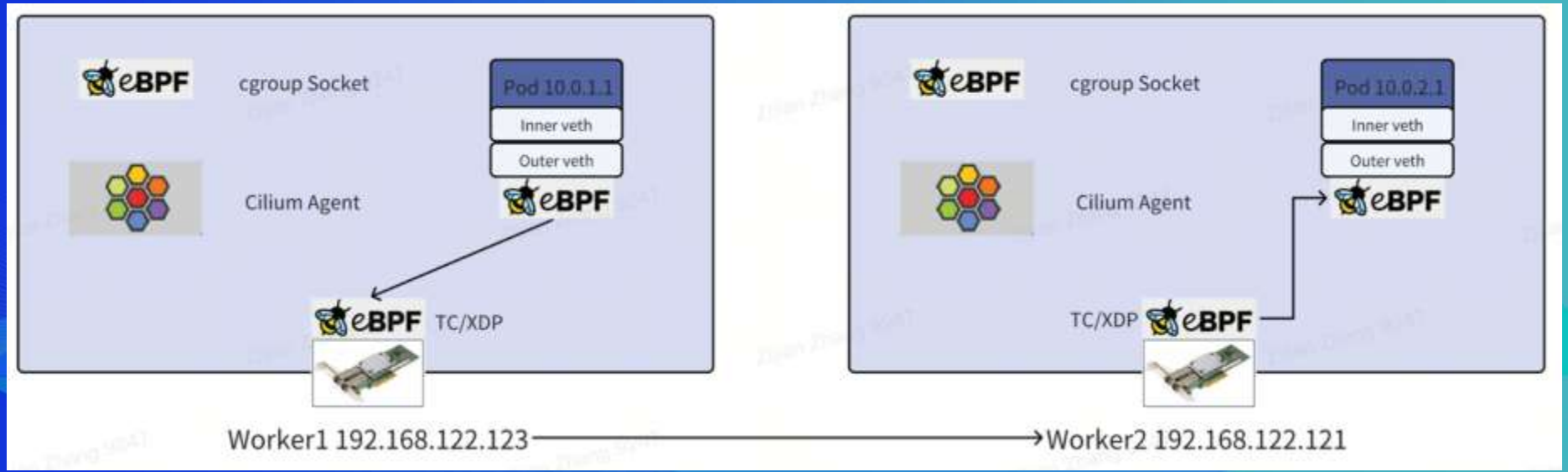


4. Cilium对K8S网络模型的实现 - 跨节点 Pod to Pod / Node

Cilium在Native Routing模式下，跨节点的连通性交给内核路由表来做，路由表的条目由Cilium Agent管理。

```
root@master:/home/cilium# ip r
default via 192.168.122.1 dev ens2
10.0.0.0/24 via 10.0.0.141 dev cilium_host src 10.0.0.141
10.0.0.141 dev cilium_host scope link
10.0.1.0/24 via 192.168.122.123 dev ens2
10.0.2.0/24 via 192.168.122.121 dev ens2
```

4. Cilium对K8S网络模型的实现 - 跨节点 Pod to Pod / Node



5. Cilium对K8S负载均衡的实现 - 实验配置

```
root@master:~# kubectl get all -o wide
NAME                                    READY   STATUS    RESTARTS   AGE   IP           NODE       NOMINATED NODE   READINESS GATES
pod/nginx-deployment-8d545c96d-fwwss    1/1     Running   0          21s   10.0.2.74    worker-2   <none>           <none>
pod/nginx-deployment-8d545c96d-x78md    1/1     Running   0          21s   10.0.1.205   worker-1   <none>           <none>

NAME                    TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE   SELECTOR
service/kubernetes      ClusterIP   10.96.0.1       <none>        443/TCP        23h   <none>
service/nginx-service   NodePort    10.103.111.79   <none>        80:31355/TCP   16s   app=nginx

NAME                               READY   UP-TO-DATE   AVAILABLE   AGE   CONTAINERS   IMAGES         SELECTOR
deployment.apps/nginx-deployment   2/2     2            2           21s   nginx        nginx:latest   app=nginx

NAME                                          DESIRED   CURRENT   READY   AGE   CONTAINERS   IMAGES         SELECTOR
replicaset.apps/nginx-deployment-8d545c96d    2         2         2       21s   nginx        nginx:latest   app=nginx,pod-template-hash=8d545c96d
```

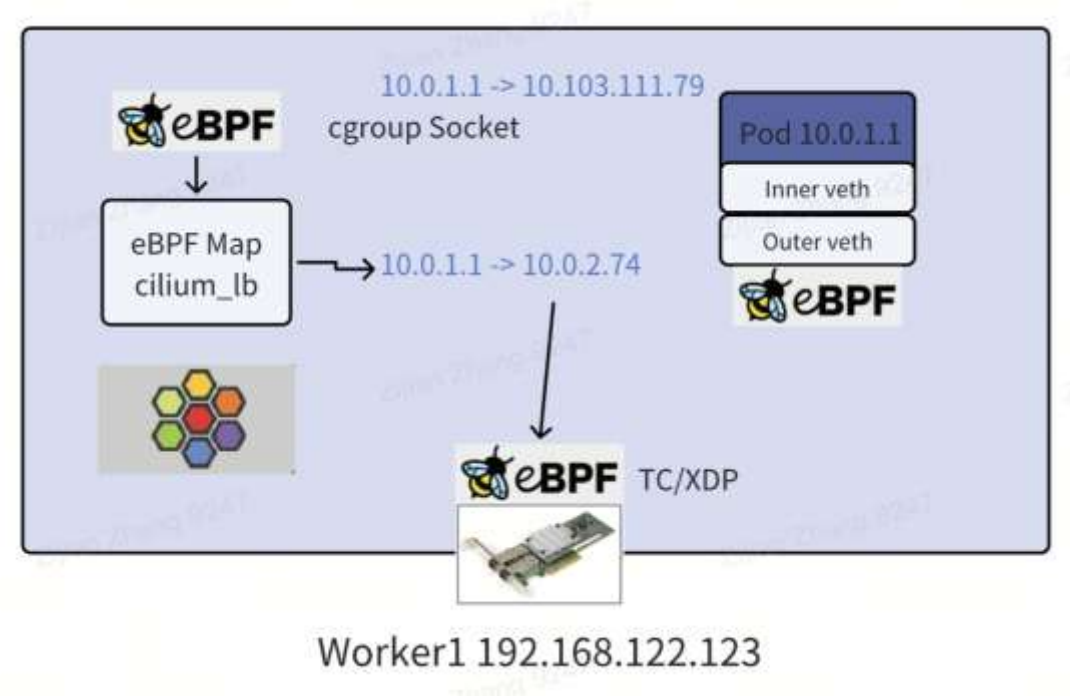
5. Cilium对K8S负载均衡的实现 - Service eBPF Map

```
root@master:/home/cilium# cilium bpf lb list
SERVICE ADDRESS          BACKEND ADDRESS
10.96.0.10:53            0.0.0.0:0 (2) [ClusterIP, non-routable]
                         10.0.0.115:53 (2)
                         10.0.0.63:53 (2)
10.96.0.10:9153          0.0.0.0:0 (3) [ClusterIP, non-routable]
                         10.0.0.115:9153 (3)
                         10.0.0.63:9153 (3)
0.0.0.0:31355            10.0.2.74:80 (6)
                         0.0.0.0:0 (6) [NodePort, non-routable]
                         10.0.1.205:80 (6)
192.168.122.233:31355    10.0.2.74:80 (5)
                         0.0.0.0:0 (5) [NodePort]
                         10.0.1.205:80 (5)
10.96.0.1:443            0.0.0.0:0 (1) [ClusterIP, non-routable]
                         192.168.122.233:6443 (1)
10.103.111.79:80         0.0.0.0:0 (4) [ClusterIP, non-routable]
                         10.0.2.74:80 (4)
                         10.0.1.205:80 (4)
```

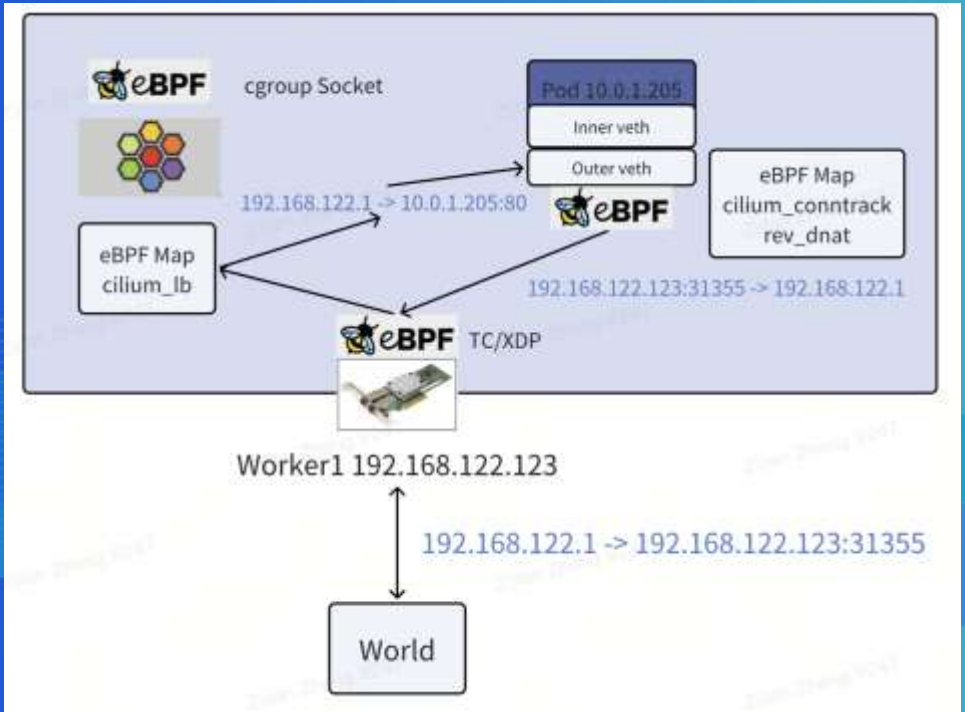
5. Cilium对K8S负载均衡的实现 - 集群内Service

```
root@master:~# bpftool cgroup tree
CgroupPath
ID       AttachType      AttachFlags     Name
/sys/fs/cgroup
25405    connect4                        sock4_connect
25385    connect6                        sock6_connect
25413    post_bind4                      sock4_post_bind
25393    post_bind6                      sock6_post_bind
25417    sendmsg4                        sock4_sendmsg
25397    sendmsg6                        sock6_sendmsg
25421    recvmsg4                        sock4_recvmsg
25401    recvmsg6                        sock6_recvmsg
25409    getpeername4                    sock4_getpeerna
25389    getpeername6                    sock6_getpeerna
```

5. Cilium对K8S负载均衡的实现 - 集群内Service



5. Cilium对K8S负载均衡的实现 - 集群外NodePort / ExternalIP



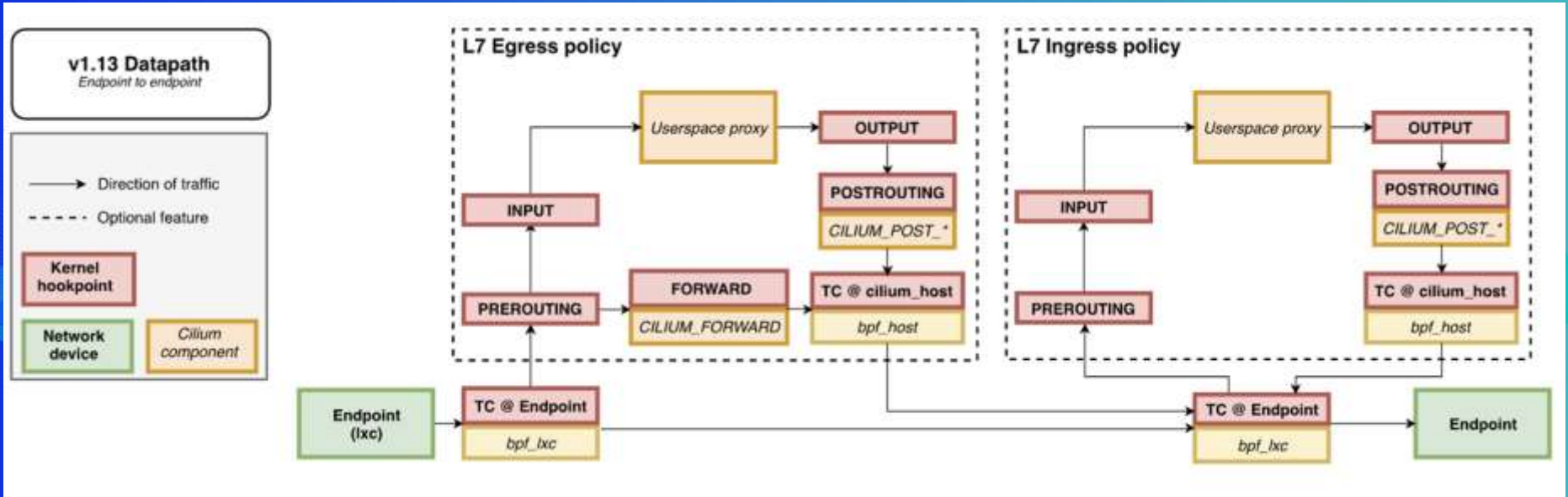
6. Cilium对网络策略的实现

```
apiVersion: "cilium.io/v2"
kind: CiliumNetworkPolicy
metadata:
  name: "l3-rule"
spec:
  endpointSelector:
    matchLabels:
      role: backend
  ingress:
  - fromEndpoints:
    - matchLabels:
        role: frontend
```

```
apiVersion: "cilium.io/v2"
kind: CiliumNetworkPolicy
metadata:
  name: "from-world-to-role-public"
spec:
  endpointSelector:
    matchLabels:
      role: public
  ingress:
    - fromEntities:
      - world
```

6. Cilium对网络策略的实现

L3策略, Cilium通过eBPF实现; L7策略Cilium通过Envoy实现



7. 总结

1. Cilium 利用 eBPF 技术深度集成 Linux 内核网络栈，实现了高性能、可编程的 Kubernetes 网络数据路径。
2. 通过 替代 iptables / kube-proxy 的方式，Cilium 更高效地实现了 Pod 到 Pod、Pod 到 Service 的通信路径。
3. 在 Service 负载均衡方面，Cilium 使用 eBPF 程序在内核空间执行 NAT 和负载分发，实现了更低延迟和更高可扩展性。
4. Cilium还有许多特性本篇没有提及，例如IP Masq，Ingress Controller, 可观测性hubble和网络安全等。

引用

1. https://www.cnblogs.com/linjiqin/p/15688218.html
2. https://lpc.events/event/7/contributions/674/attachments/568/1002/plumbers_2020_cilium_load_balancer.pdf
3. https://github.com/cilium/cilium
4. https://docs.cilium.io/en/stable/network/ebpf/intro/