Performance Analysis of Cilium

An **eBPF** based **CNI**

1) **Anvaya Bheemanakone Narappa**
2) **Mohammed Syed Akbar Hashmi**

Under supervision of Federico Parola

Problem Statement

To perform the performance of analysis of Cilium (ebpf based CNI) and Calico (standard networking - kube-proxy) with respect to number of Layer 3/4 Layer 7 policies.

eBPF

eBPF (extended Berkeley Packet Filter) is a powerful and flexible technology that allows users to run custom programs within the Linux kernel. It was initially developed for network packet filtering and analysis, but has since been extended to other areas such as tracing and performance analysis.

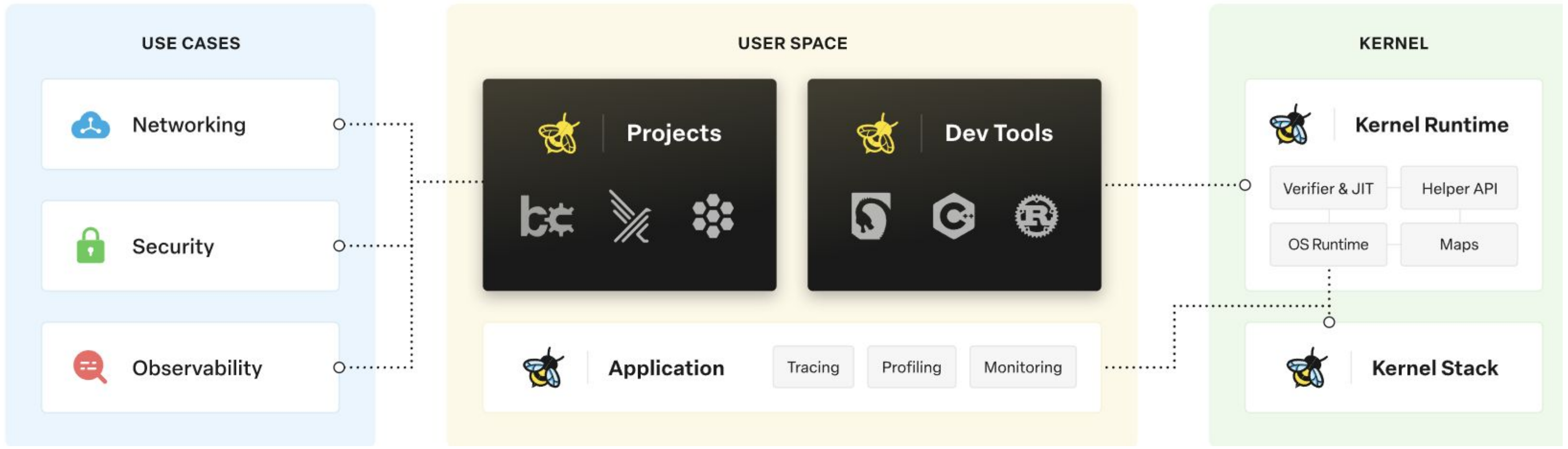
Why eBPF?

- ***Performance:*** eBPF drastically improves processing by being JIT compiled and running directly in the kernel.

- ***Security:*** eBPF programs are verified to not crash the kernel and can only be modified by privileged users.

- ***Flexibility***: Modify or add functionality and use cases to the kernel without having to restart or patch it.

- eBPF programs can be hooked anywhere on the kernel to modify functionality.
- Programs are verified to execute safely on the kernel
- Many applications such as bcc, Cilium, Falco, Katran use ebpf.
- The Use cases include Networking, Security and Observability.



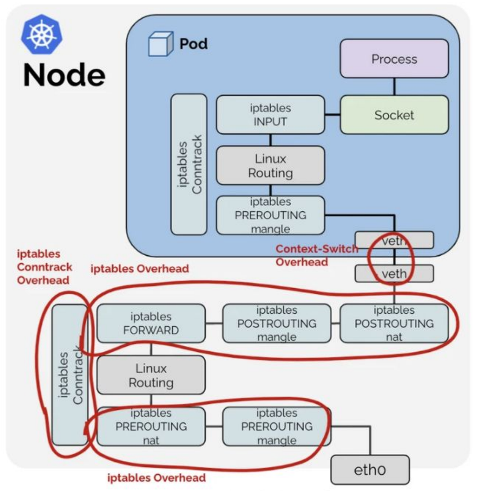
Cilium

- Open-source software-defined networking (SDN) solution
- Provides secure and efficient networking for containerized applications running in Kubernetes environments.
- Built on top of Linux kernel technologies such as eBPF (extended Berkeley Packet Filter) and XDP (eXpress Data Path)
- Cilium offers several features that help improve networking in Kubernetes environments such as High performance, Secure connectivity, Observability

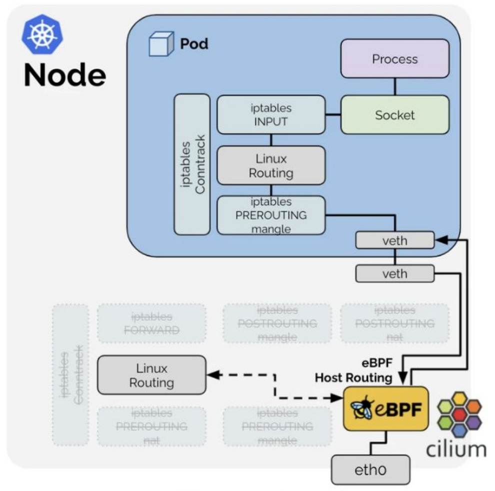
Container Networking (Standard vs Cilium ebpf)

- ebpf host routing in Cilium Allows to bypass all the iptables overhead and as well as some context switching overhead while traversing through the veth pair.

- Network packets are picked up as early as possible from the network device facing the network and delivered directly into the network namespace of the Kubernetes Pod



Standard Container Networking



Cilium eBPF Container Networking

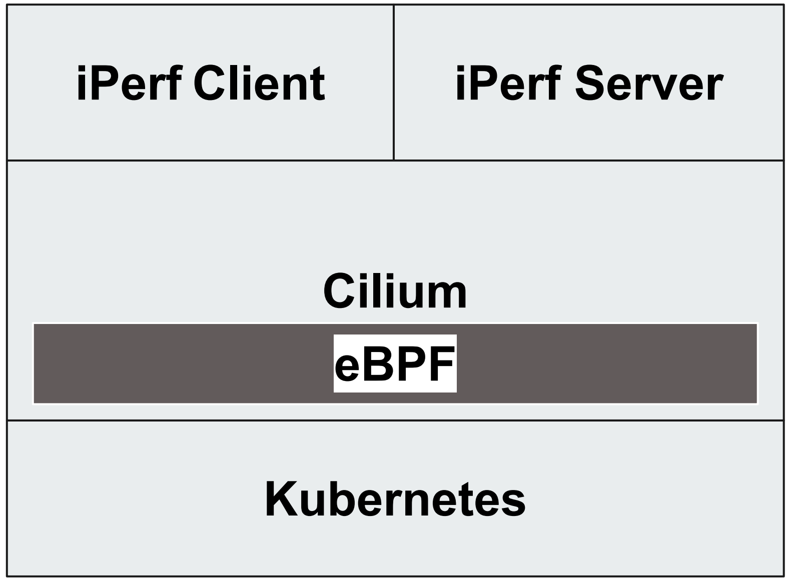
Tools/Frameworks

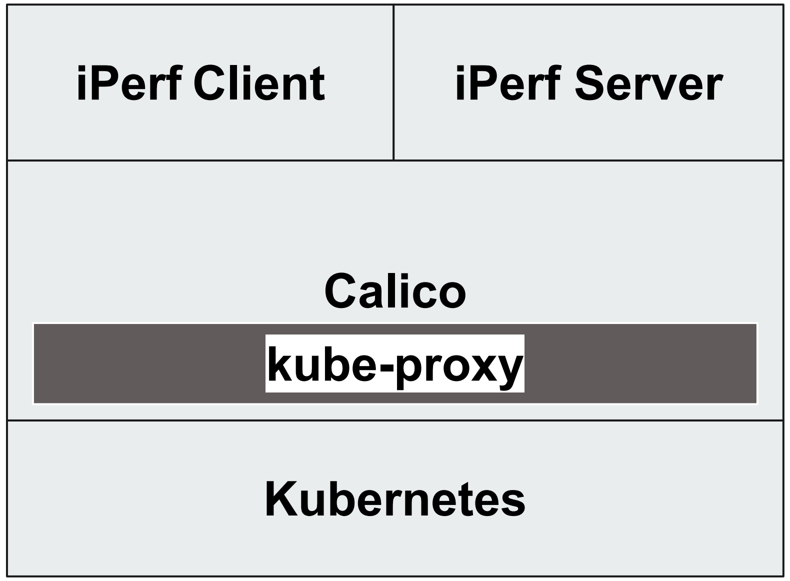
- Kubernetes
- GCP
- Cilium CNI
- Calico
- iperf/netperf
- Apache benchmark

Environment Setup

- Hosted on Google Cloud Platform
- Linux v5.4.0-1101-gcp -> Ubuntu 18.04.1
- 2 CPU Cores - Intel(R) Xeon(R) CPU @ 2.20GHz
- Cilium v1.13.0
- Calico v3.24.5
- NIC - 10Gbps
- Kubernetes v1.19 (1 Master +1 Worker)

Experiment Setup



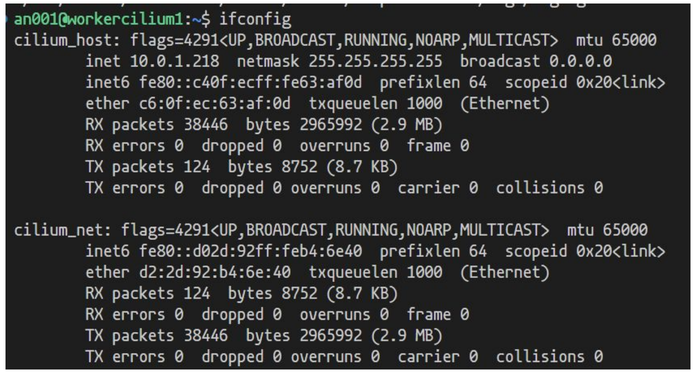


Parameter Tuning for Cilium

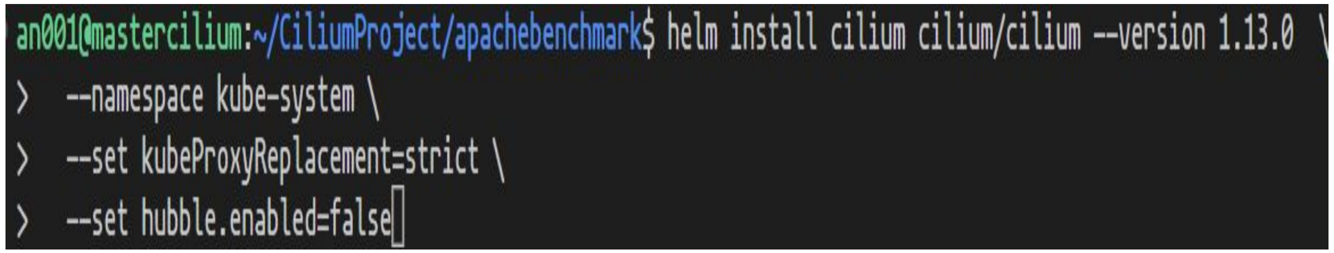
We tuned the environment to achieve maximum benchmark

The parameters that were changed were

- Disabled Hubble as this added an additional overhead of 1-15 %
- Increased the MTU size - the system was configured to use jumbo size frames
- Kube-Proxy replacement was set to strict



```
an001@workercilium1:~$ ifconfig
cilium_host: flags=4291<UP,BROADCAST,RUNNING,NOARP,MULTICAST>  mtu 65000
        inet 10.0.1.218  netmask 255.255.255.255  broadcast 0.0.0.0
        inet6 fe80::c40f:ecff:fe63:af0d  prefixlen 64  scopeid 0x20<link>
        ether c6:0f:ec:63:af:0d  txqueuelen 1000  (Ethernet)
        RX packets 38446  bytes 2965992 (2.9 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 124  bytes 8752 (8.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

cilium_net: flags=4291<UP,BROADCAST,RUNNING,NOARP,MULTICAST>  mtu 65000
        inet6 fe80::d02d:92ff:feb4:6e40  prefixlen 64  scopeid 0x20<link>
        ether d2:2d:92:b4:6e:40  txqueuelen 1000  (Ethernet)
        RX packets 124  bytes 8752 (8.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 38446  bytes 2965992 (2.9 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

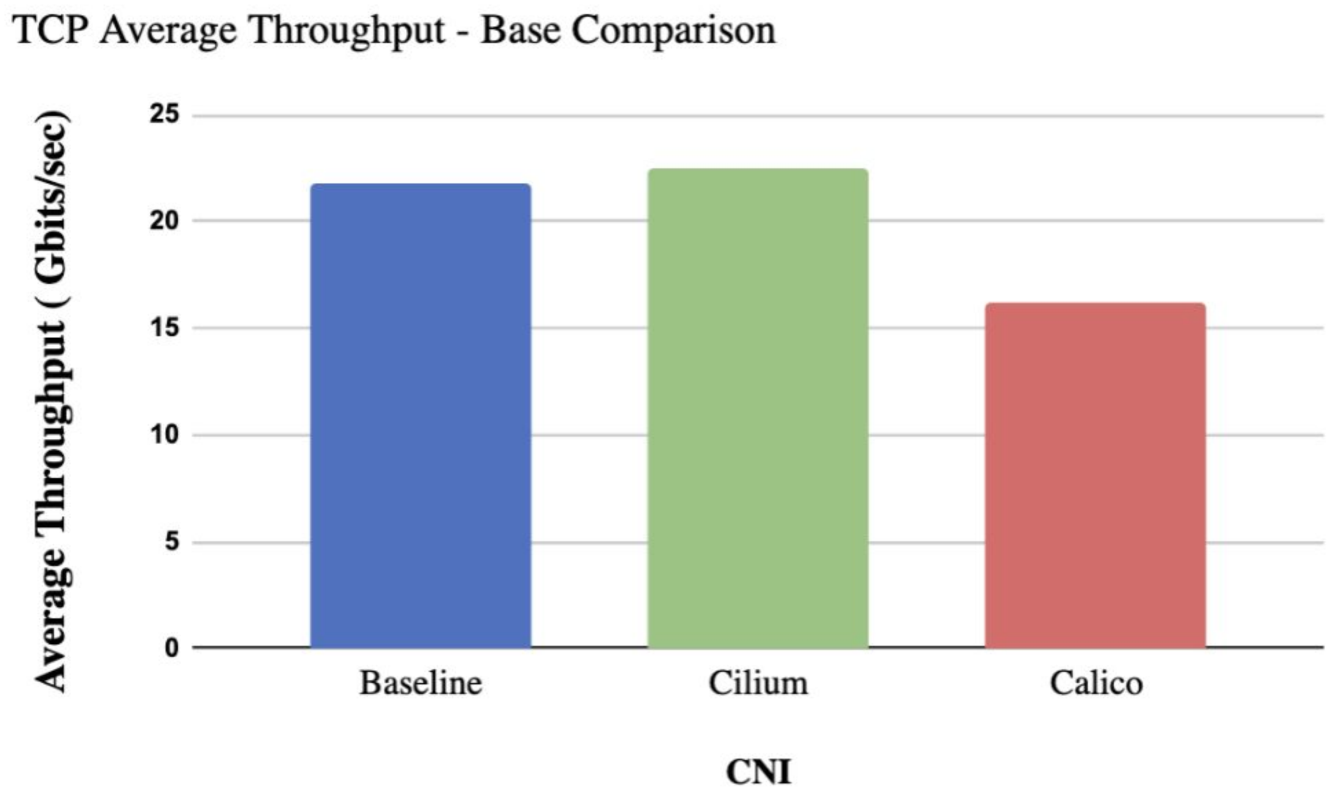


```
an001@mastercilium:~/CiliumProject/apachebenchmark$ helm install cilium cilium/cilium --version 1.13.0  \
> --namespace kube-system \
> --set kubeProxyReplacement=strict \
> --set hubble.enabled=false
```

Experiments Performed

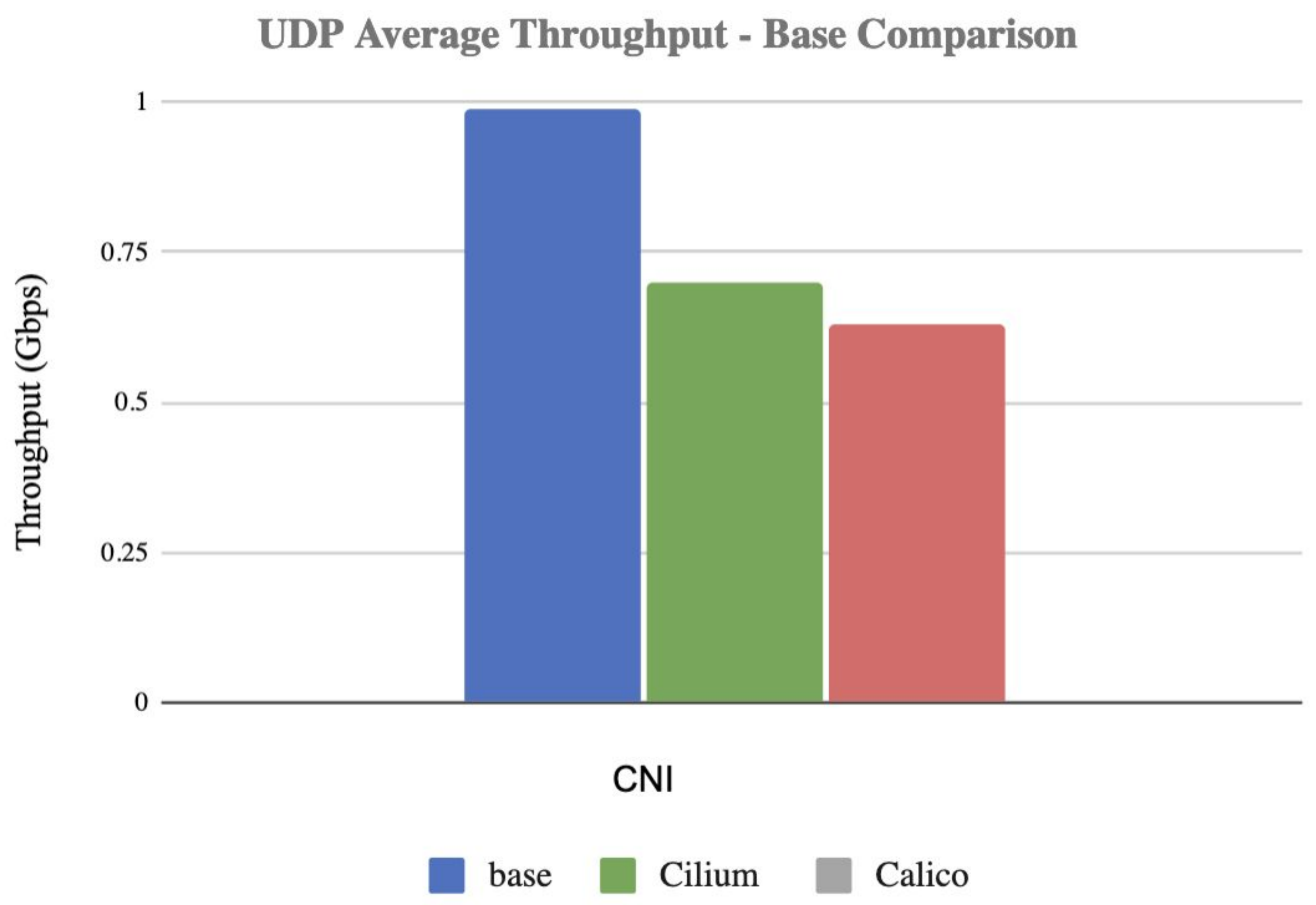
1. General benchmarks for TCP/UDP - Provides baseline
   a. Throughput - refers to the amount of data that can be transmitted over a network connection within a certain time period
2. Layer3/4 policy testing for TCP/UDP
   a. cpu utilisation - refers to the percentage of time that the CPU is busy processing instructions. It is used to evaluate the efficiency and capacity of a network connection or network device
3. L7 Policy Testing
   a. Time per request - measures the time taken to process a single request at the application layer

General Benchmarks - TCP



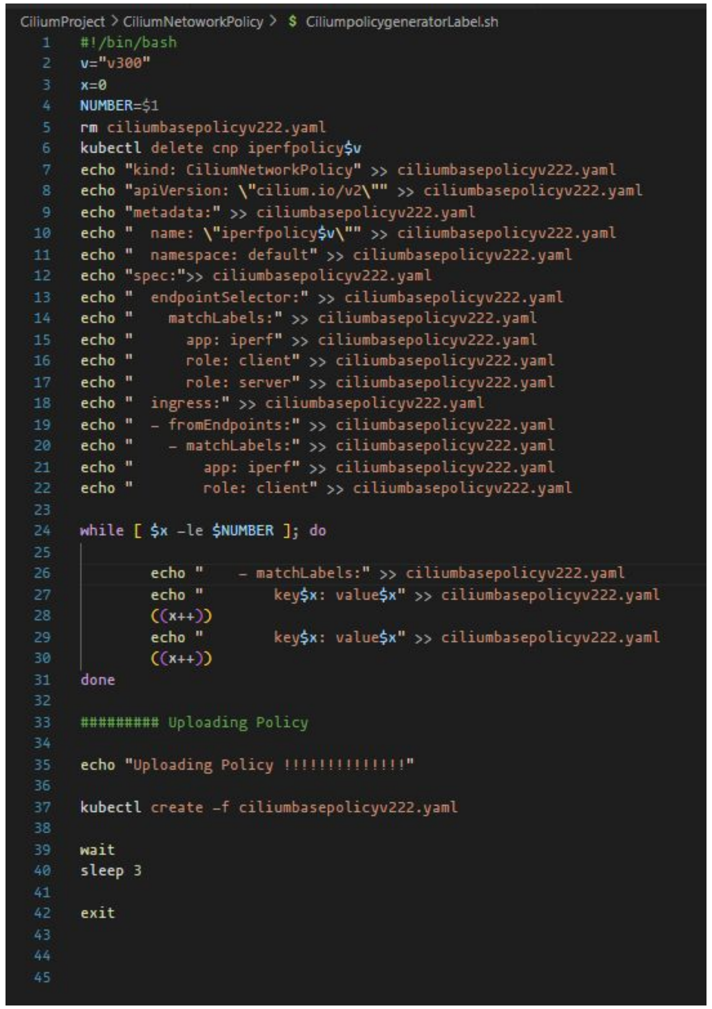
- Baseline model doesn’t have any CNI for routing
- No networking policies were applied

General Benchmarks - UDP



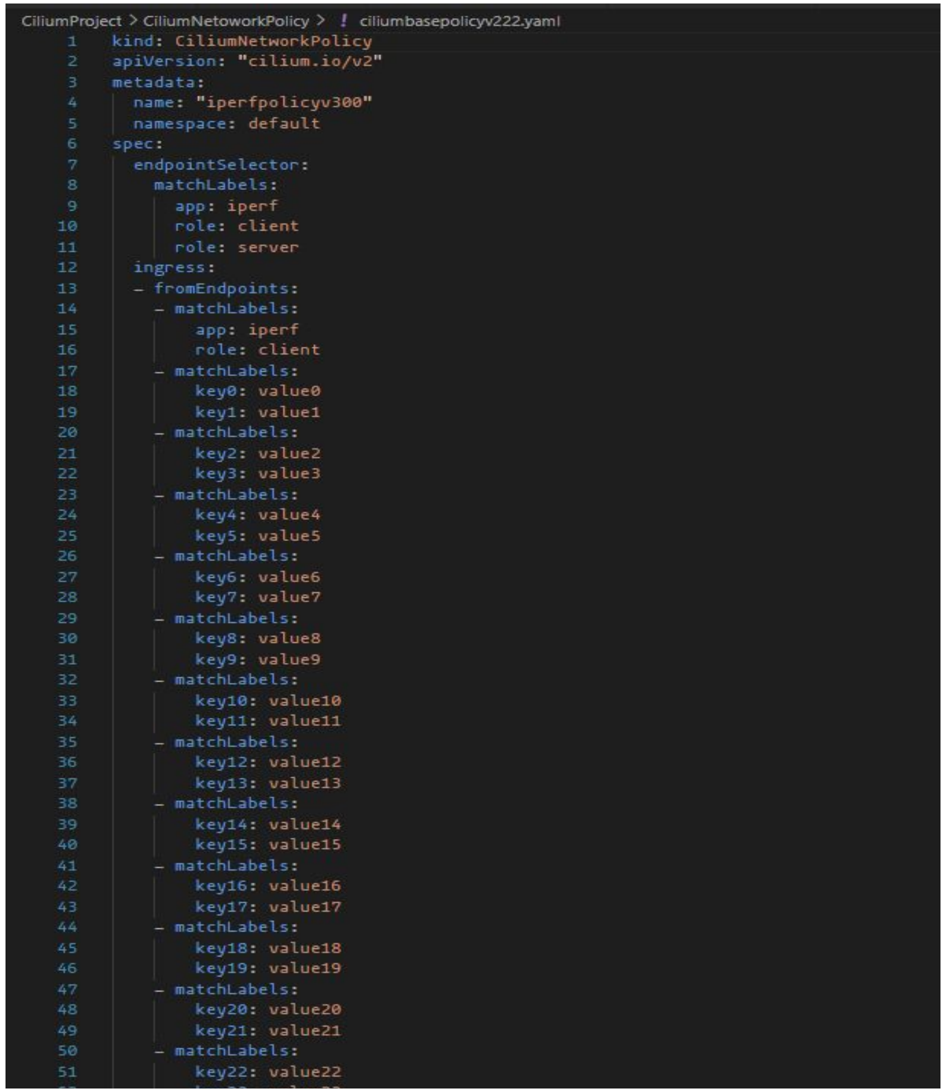
- tested with bandwidth tuned to 1 Gbps
- Cilium still outperforms Calico

Layer3/4 policy testing for TCP



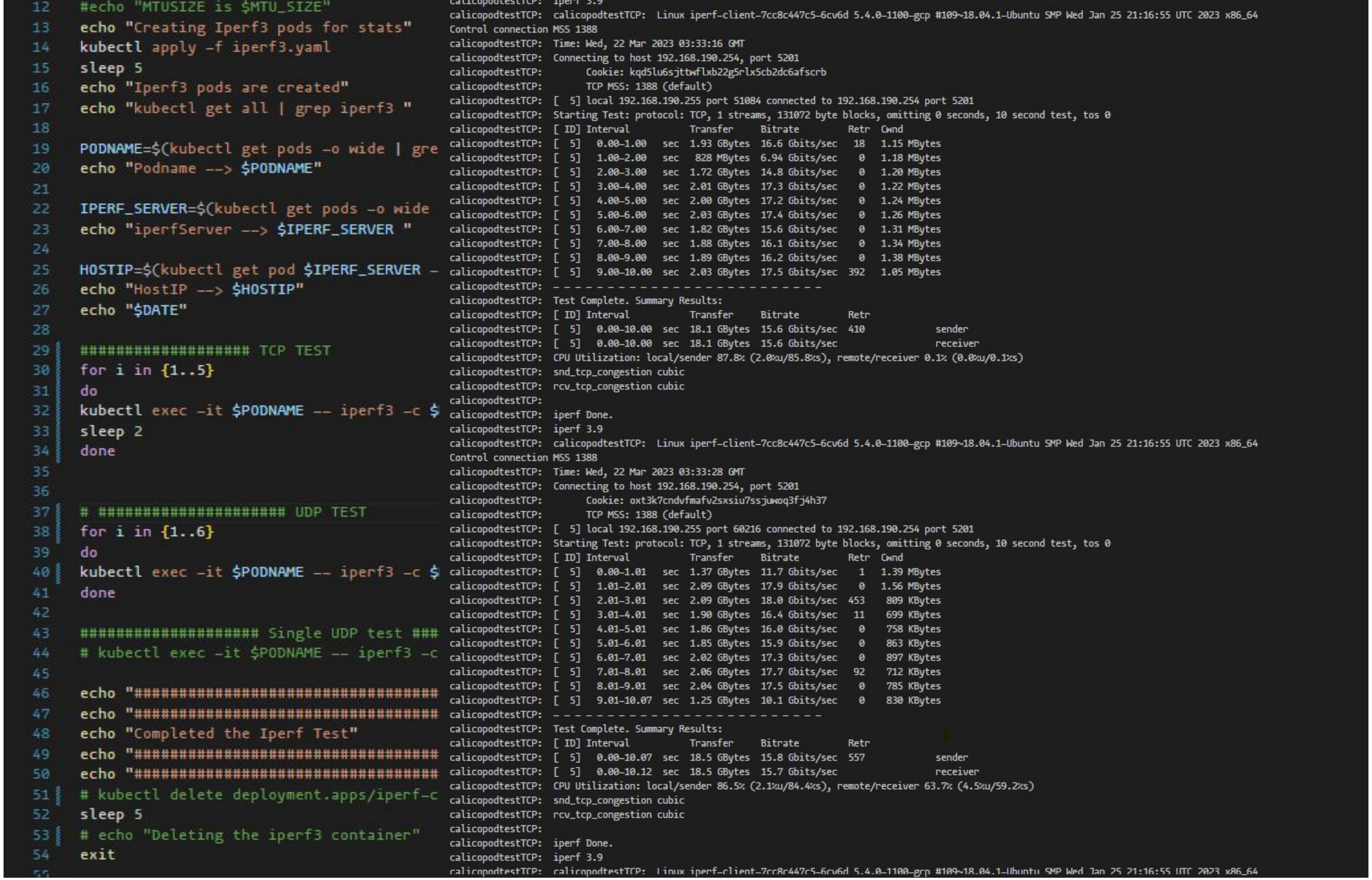
CiliumProject > CiliumNetoworkPolicy > $ CiliumpolicygeneratorLabel.sh

```bash
#!/bin/bash
v="v300"
x=0
NUMBER=$1
rm ciliumbasepolicyv222.yaml
kubectl delete cnp iperfpolicy$v
echo "kind: CiliumNetworkPolicy" >> ciliumbasepolicyv222.yaml
echo "apiVersion: \"cilium.io/v2\"" >> ciliumbasepolicyv222.yaml
echo "metadata:" >> ciliumbasepolicyv222.yaml
echo "  name: \"iperfpolicy$v\"" >> ciliumbasepolicyv222.yaml
echo "  namespace: default" >> ciliumbasepolicyv222.yaml
echo "spec:">> ciliumbasepolicyv222.yaml
echo "  endpointSelector:" >> ciliumbasepolicyv222.yaml
echo "    matchLabels:" >> ciliumbasepolicyv222.yaml
echo "      app: iperf" >> ciliumbasepolicyv222.yaml
echo "      role: client" >> ciliumbasepolicyv222.yaml
echo "      role: server" >> ciliumbasepolicyv222.yaml
echo "  ingress:" >> ciliumbasepolicyv222.yaml
echo "  - fromEndpoints:" >> ciliumbasepolicyv222.yaml
echo "    - matchLabels:" >> ciliumbasepolicyv222.yaml
echo "        app: iperf" >> ciliumbasepolicyv222.yaml
echo "        role: client" >> ciliumbasepolicyv222.yaml

while [ $x -le $NUMBER ]; do

        echo "    - matchLabels:" >> ciliumbasepolicyv222.yaml
        echo "        key$x: value$x" >> ciliumbasepolicyv222.yaml
        ((x++))
        echo "        key$x: value$x" >> ciliumbasepolicyv222.yaml
        ((x++))
done

######### Uploading Policy

echo "Uploading Policy !!!!!!!!!!!!!!"

kubectl create -f ciliumbasepolicyv222.yaml

wait
sleep 3

exit
```



CiliumProject > CiliumNetoworkPolicy > ! ciliumbasepolicyv222.yaml

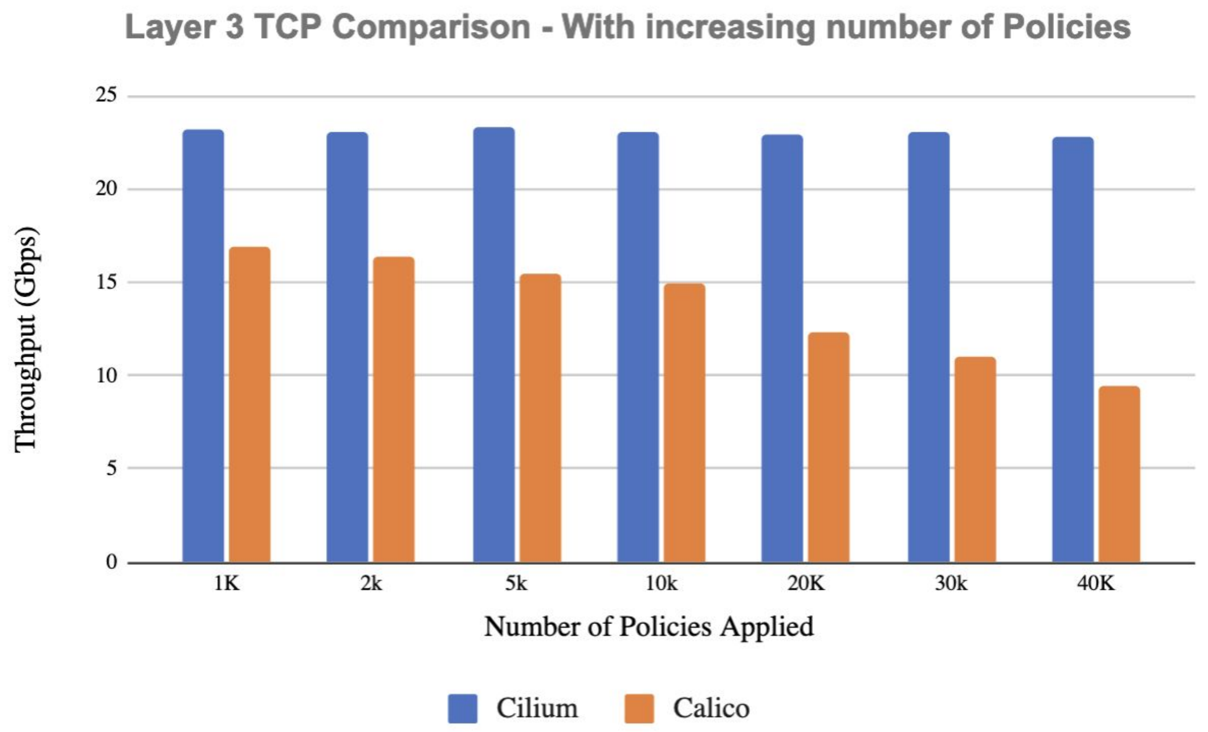
```yaml
kind: CiliumNetworkPolicy
apiVersion: "cilium.io/v2"
metadata:
  name: "iperfpolicyv300"
  namespace: default
spec:
  endpointSelector:
    matchLabels:
      app: iperf
      role: client
      role: server
  ingress:
  - fromEndpoints:
    - matchLabels:
        app: iperf
        role: client
    - matchLabels:
        key0: value0
        key1: value1
    - matchLabels:
        key2: value2
        key3: value3
    - matchLabels:
        key4: value4
        key5: value5
    - matchLabels:
        key6: value6
        key7: value7
    - matchLabels:
        key8: value8
        key9: value9
    - matchLabels:
        key10: value10
        key11: value11
    - matchLabels:
        key12: value12
        key13: value13
    - matchLabels:
        key14: value14
        key15: value15
    - matchLabels:
        key16: value16
        key17: value17
    - matchLabels:
        key18: value18
        key19: value19
    - matchLabels:
        key20: value20
        key21: value21
    - matchLabels:
        key22: value22
```

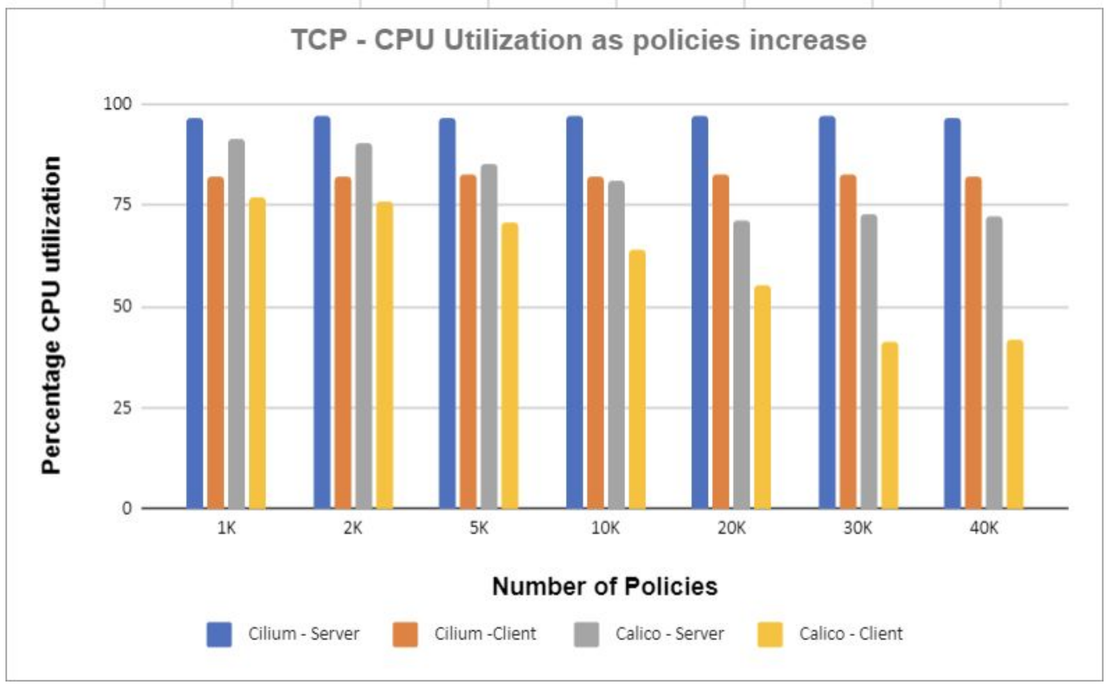


```
#echo "MTUSIZE is $MTU_SIZE"
echo "Creating Iperf3 pods for stats"
kubectl apply -f iperf3.yaml
sleep 5
echo "Iperf3 pods are created"
echo "kubectl get all | grep iperf3 "

PODNAME=$(kubectl get pods -o wide | gre
echo "Podname --> $PODNAME"

IPERF_SERVER=$(kubectl get pods -o wide
echo "iperfServer --> $IPERF_SERVER "

HOSTIP=$(kubectl get pod $IPERF_SERVER -
echo "HostIP --> $HOSTIP"
echo "$DATE"

#################### TCP TEST
for i in {1..5}
do
kubectl exec -it $PODNAME -- iperf3 -c $
sleep 2
done


# ###################### UDP TEST
for i in {1..6}
do
kubectl exec -it $PODNAME -- iperf3 -c $
done

##################### Single UDP test ###
# kubectl exec -it $PODNAME -- iperf3 -c

echo "#################################
echo "#################################
echo "Completed the Iperf Test"
echo "#################################
echo "#################################
# kubectl delete deployment.apps/iperf-c
sleep 5
# echo "Deleting the iperf3 container"
exit
```

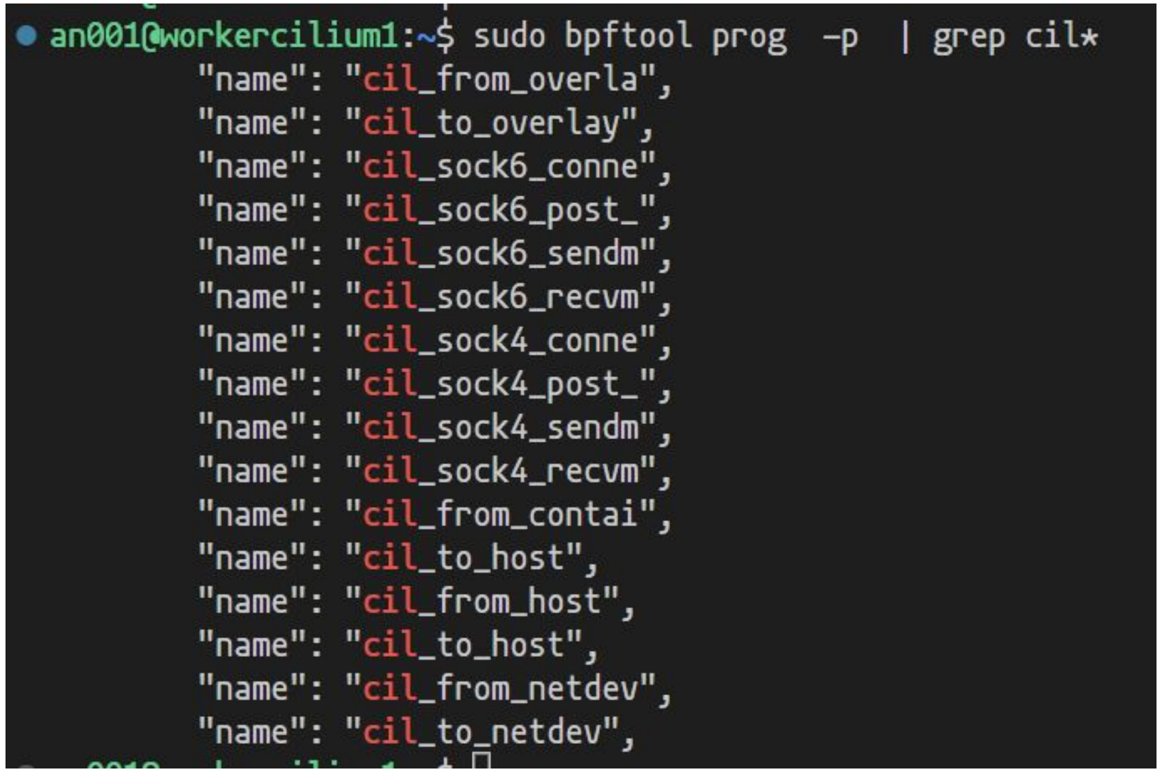
```
calicopodtestTCP:  iperf 3.9
calicopodtestTCP:  calicopodtestTCP:  Linux iperf-client-7cc8c447c5-6cv6d 5.4.0-1100-gcp #109~18.04.1-Ubuntu SMP Wed Jan 25 21:16:55 UTC 2023 x86_64
Control connection MSS 1388
calicopodtestTCP:  Time: Wed, 22 Mar 2023 03:33:16 GMT
calicopodtestTCP:  Connecting to host 192.168.190.254, port 5201
calicopodtestTCP:        Cookie: kqd5lu6sjttwflxb22g5rlx5cb2dc6afscrb
calicopodtestTCP:        TCP MSS: 1388 (default)
calicopodtestTCP:  [  5] local 192.168.190.255 port 51084 connected to 192.168.190.254 port 5201
calicopodtestTCP:  Starting Test: protocol: TCP, 1 streams, 131072 byte blocks, omitting 0 seconds, 10 second test, tos 0
calicopodtestTCP:  [ ID] Interval           Transfer     Bitrate         Retr  Cwnd
calicopodtestTCP:  [  5]   0.00-1.00   sec  1.93 GBytes  16.6 Gbits/sec   18   1.15 MBytes
calicopodtestTCP:  [  5]   1.00-2.00   sec   828 MBytes  6.94 Gbits/sec    0   1.18 MBytes
calicopodtestTCP:  [  5]   2.00-3.00   sec  1.72 GBytes  14.8 Gbits/sec    0   1.20 MBytes
calicopodtestTCP:  [  5]   3.00-4.00   sec  2.01 GBytes  17.3 Gbits/sec    0   1.22 MBytes
calicopodtestTCP:  [  5]   4.00-5.00   sec  2.00 GBytes  17.2 Gbits/sec    0   1.24 MBytes
calicopodtestTCP:  [  5]   5.00-6.00   sec  2.03 GBytes  17.4 Gbits/sec    0   1.26 MBytes
calicopodtestTCP:  [  5]   6.00-7.00   sec  1.82 GBytes  15.6 Gbits/sec    0   1.31 MBytes
calicopodtestTCP:  [  5]   7.00-8.00   sec  1.88 GBytes  16.1 Gbits/sec    0   1.34 MBytes
calicopodtestTCP:  [  5]   8.00-9.00   sec  1.89 GBytes  16.2 Gbits/sec    0   1.38 MBytes
calicopodtestTCP:  [  5]   9.00-10.00  sec  2.03 GBytes  17.5 Gbits/sec  392   1.05 MBytes
calicopodtestTCP:  - - - - - - - - - - - - - - - - - - - - - - - - -
calicopodtestTCP:  Test Complete. Summary Results:
calicopodtestTCP:  [ ID] Interval           Transfer     Bitrate         Retr
calicopodtestTCP:  [  5]   0.00-10.00  sec  18.1 GBytes  15.6 Gbits/sec  410             sender
calicopodtestTCP:  [  5]   0.00-10.00  sec  18.1 GBytes  15.6 Gbits/sec                  receiver
calicopodtestTCP:  CPU Utilization: local/sender 87.8% (2.0%u/85.8%s), remote/receiver 0.1% (0.0%u/0.1%s)
calicopodtestTCP:  snd_tcp_congestion cubic
calicopodtestTCP:  rcv_tcp_congestion cubic
calicopodtestTCP:
calicopodtestTCP:  iperf Done.
calicopodtestTCP:  iperf 3.9
calicopodtestTCP:  calicopodtestTCP:  Linux iperf-client-7cc8c447c5-6cv6d 5.4.0-1100-gcp #109~18.04.1-Ubuntu SMP Wed Jan 25 21:16:55 UTC 2023 x86_64
Control connection MSS 1388
calicopodtestTCP:  Time: Wed, 22 Mar 2023 03:33:28 GMT
calicopodtestTCP:  Connecting to host 192.168.190.254, port 5201
calicopodtestTCP:        Cookie: oxt3k7cndvfmafv2sxsiu7ssjuwoq3fj4h37
calicopodtestTCP:        TCP MSS: 1388 (default)
calicopodtestTCP:  [  5] local 192.168.190.255 port 60216 connected to 192.168.190.254 port 5201
calicopodtestTCP:  Starting Test: protocol: TCP, 1 streams, 131072 byte blocks, omitting 0 seconds, 10 second test, tos 0
calicopodtestTCP:  [ ID] Interval           Transfer     Bitrate         Retr  Cwnd
calicopodtestTCP:  [  5]   0.00-1.01   sec  1.37 GBytes  11.7 Gbits/sec    1   1.39 MBytes
calicopodtestTCP:  [  5]   1.01-2.01   sec  2.09 GBytes  17.9 Gbits/sec    0   1.56 MBytes
calicopodtestTCP:  [  5]   2.01-3.01   sec  2.09 GBytes  18.0 Gbits/sec  453    809 KBytes
calicopodtestTCP:  [  5]   3.01-4.01   sec  1.90 GBytes  16.4 Gbits/sec   11    699 KBytes
calicopodtestTCP:  [  5]   4.01-5.01   sec  1.86 GBytes  16.0 Gbits/sec    0    758 KBytes
calicopodtestTCP:  [  5]   5.01-6.01   sec  1.85 GBytes  15.9 Gbits/sec    0    863 KBytes
calicopodtestTCP:  [  5]   6.01-7.01   sec  2.02 GBytes  17.3 Gbits/sec    0    897 KBytes
calicopodtestTCP:  [  5]   7.01-8.01   sec  2.06 GBytes  17.7 Gbits/sec   92    712 KBytes
calicopodtestTCP:  [  5]   8.01-9.01   sec  2.04 GBytes  17.5 Gbits/sec    0    785 KBytes
calicopodtestTCP:  [  5]   9.01-10.07  sec  1.25 GBytes  10.1 Gbits/sec    0    830 KBytes
calicopodtestTCP:  - - - - - - - - - - - - - - - - - - - - - - - - -
calicopodtestTCP:  Test Complete. Summary Results:
calicopodtestTCP:  [ ID] Interval           Transfer     Bitrate         Retr
calicopodtestTCP:  [  5]   0.00-10.07  sec  18.5 GBytes  15.8 Gbits/sec  557             sender
calicopodtestTCP:  [  5]   0.00-10.12  sec  18.5 GBytes  15.7 Gbits/sec                  receiver
calicopodtestTCP:  CPU Utilization: local/sender 86.5% (2.1%u/84.4%s), remote/receiver 63.7% (4.5%u/59.2%s)
calicopodtestTCP:  snd_tcp_congestion cubic
calicopodtestTCP:  rcv_tcp_congestion cubic
calicopodtestTCP:
calicopodtestTCP:  iperf Done.
calicopodtestTCP:  iperf 3.9
calicopodtestTCP:  calicopodtestTCP:  Linux iperf-client-7cc8c447c5-6cv6d 5.4.0-1100-gcp #109~18.04.1-Ubuntu SMP Wed Jan 25 21:16:55 UTC 2023 x86_64
```

Layer3/4 policy testing for TCP

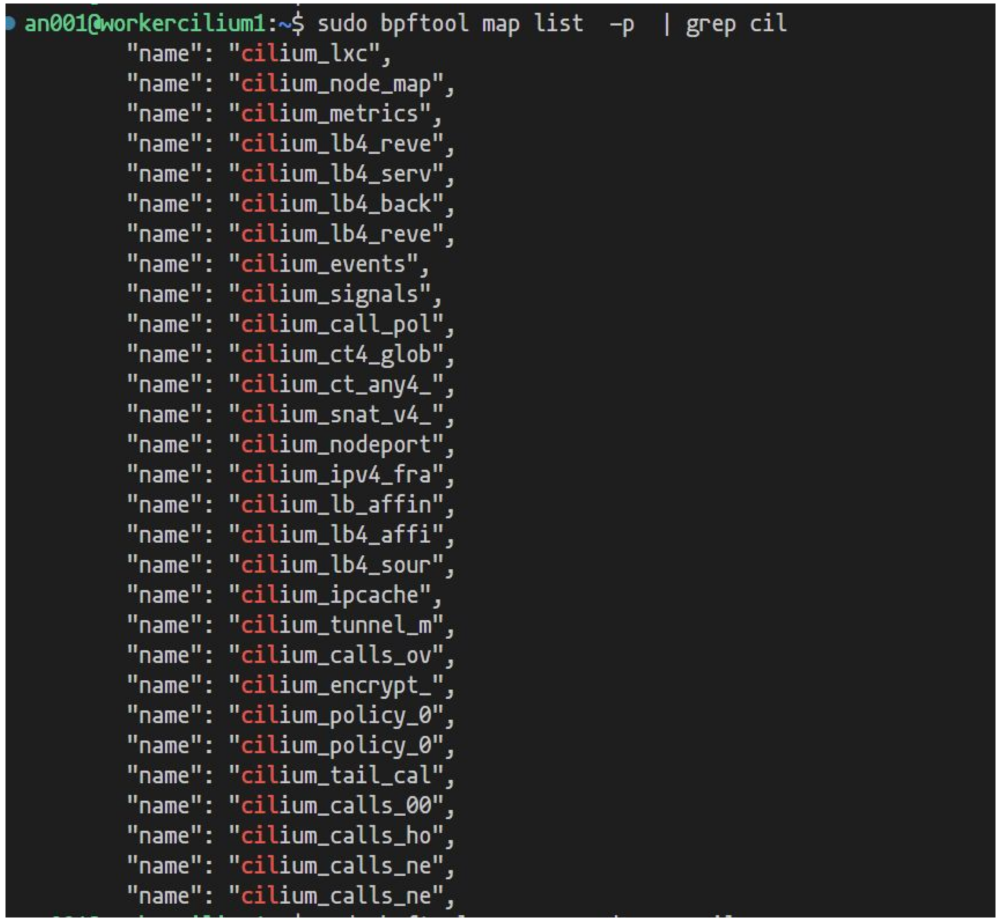




Cilium eBPF Maps and Programs

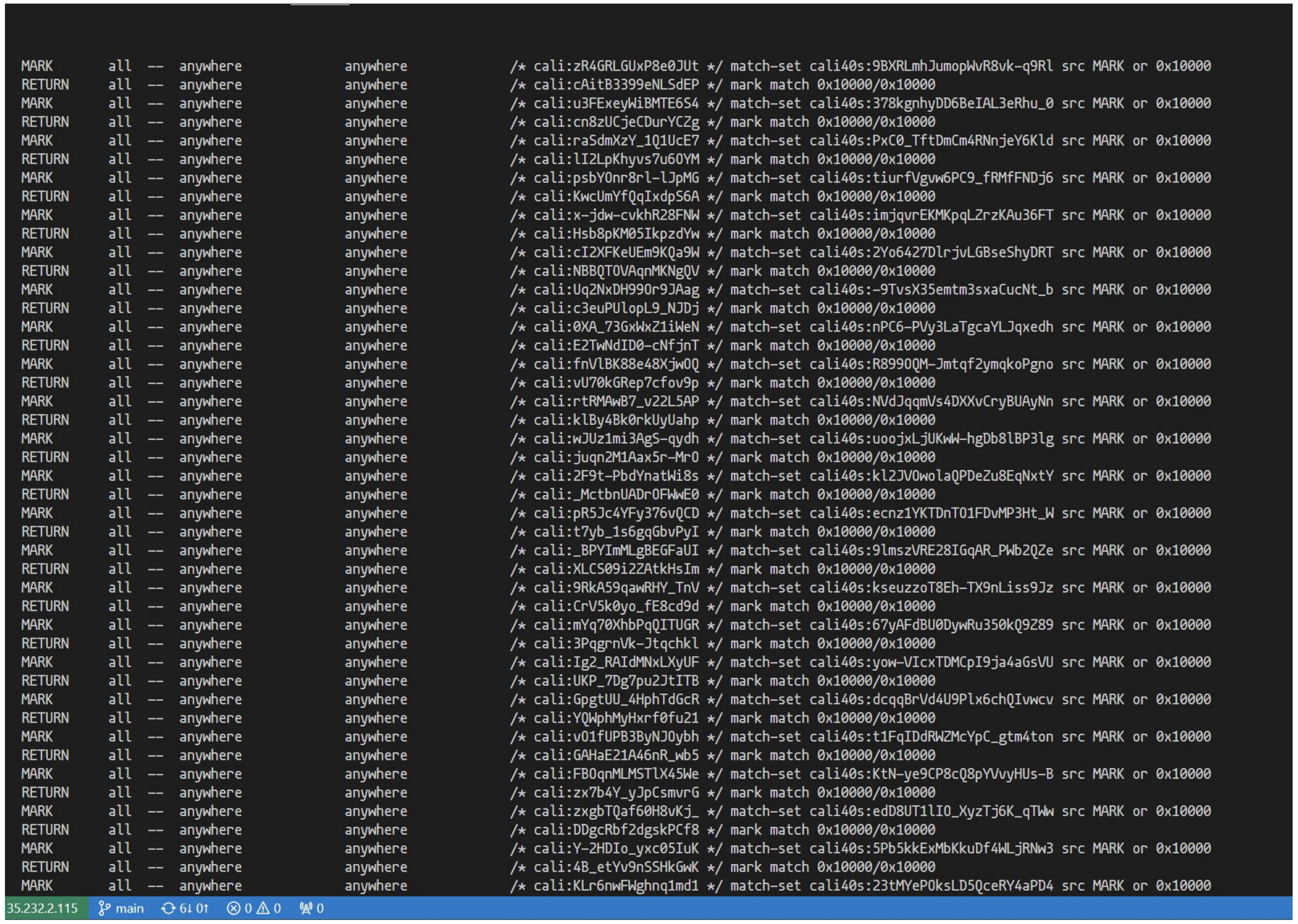


```
an001@workercilium1:~$ sudo bpftool prog  -p  | grep cil*
        "name": "cil_from_overla",
        "name": "cil_to_overlay",
        "name": "cil_sock6_conne",
        "name": "cil_sock6_post_",
        "name": "cil_sock6_sendm",
        "name": "cil_sock6_recvm",
        "name": "cil_sock4_conne",
        "name": "cil_sock4_post_",
        "name": "cil_sock4_sendm",
        "name": "cil_sock4_recvm",
        "name": "cil_from_contai",
        "name": "cil_to_host",
        "name": "cil_from_host",
        "name": "cil_to_host",
        "name": "cil_from_netdev",
        "name": "cil_to_netdev",
```



```
an001@workercilium1:~$ sudo bpftool map list  -p  | grep cil
        "name": "cilium_lxc",
        "name": "cilium_node_map",
        "name": "cilium_metrics",
        "name": "cilium_lb4_reve",
        "name": "cilium_lb4_serv",
        "name": "cilium_lb4_back",
        "name": "cilium_lb4_reve",
        "name": "cilium_events",
        "name": "cilium_signals",
        "name": "cilium_call_pol",
        "name": "cilium_ct4_glob",
        "name": "cilium_ct_any4_",
        "name": "cilium_snat_v4_",
        "name": "cilium_nodeport",
        "name": "cilium_ipv4_fra",
        "name": "cilium_lb_affin",
        "name": "cilium_lb4_affi",
        "name": "cilium_lb4_sour",
        "name": "cilium_ipcache",
        "name": "cilium_tunnel_m",
        "name": "cilium_calls_ov",
        "name": "cilium_encrypt_",
        "name": "cilium_policy_0",
        "name": "cilium_policy_0",
        "name": "cilium_tail_cal",
        "name": "cilium_calls_00",
        "name": "cilium_calls_ho",
        "name": "cilium_calls_ne",
        "name": "cilium_calls_ne",
```

Calico IPTABLES entries

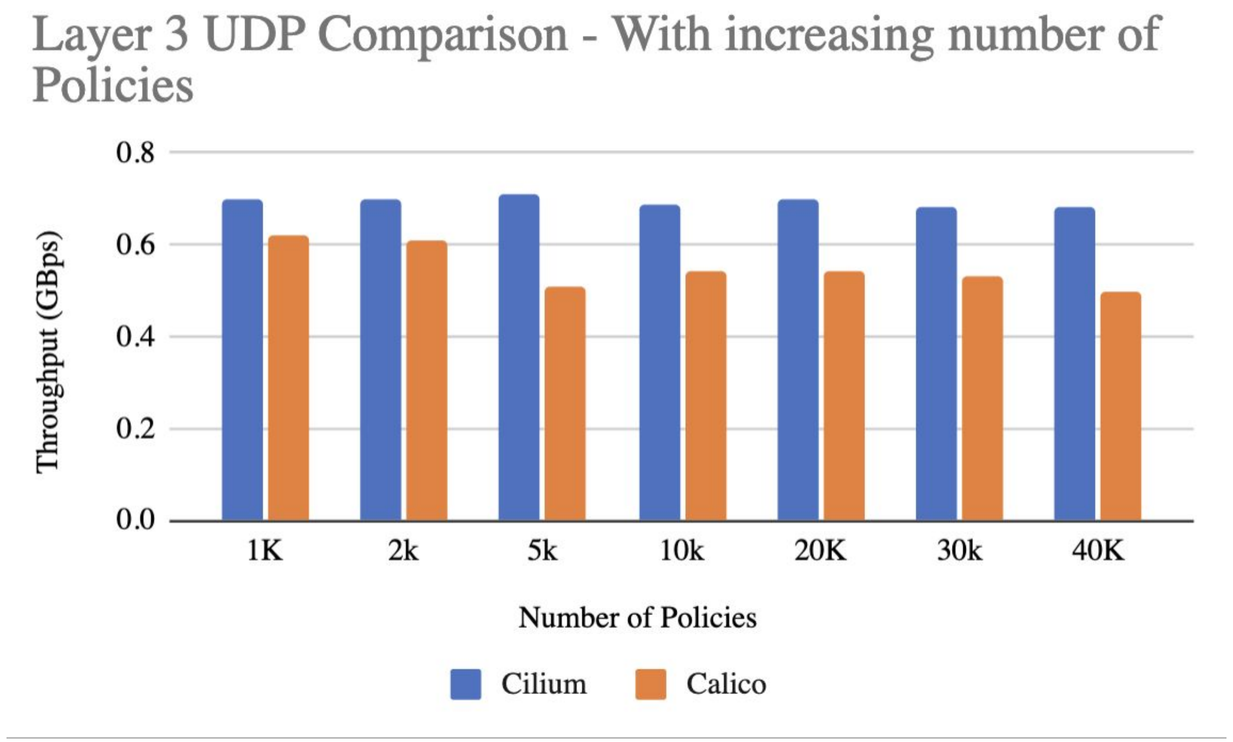


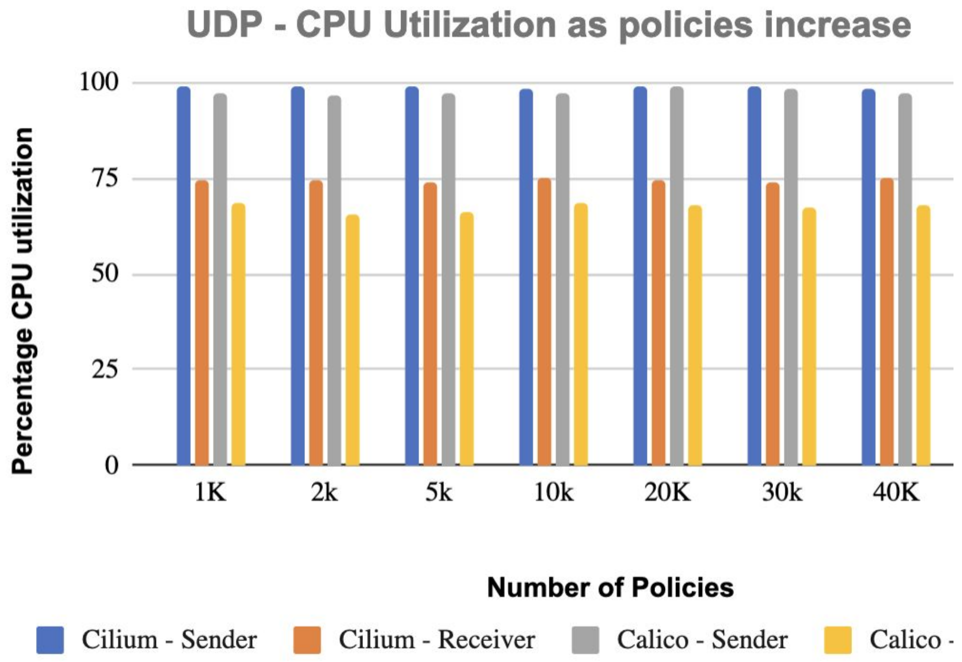
```
MARK       all  --  anywhere             anywhere             /* cali:zR4GRLGUxP8e0JUt */ match-set cali40s:9BXRLmhJumopWvR8vk-q9Rl src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:cAitB3399eNLSdEP */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:u3FExeyWiBMTE6S4 */ match-set cali40s:378kgnhyDD6BeIAL3eRhu_0 src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:cn8zUCjeCDurYCZg */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:raSdmXzY_1Q1UcE7 */ match-set cali40s:PxC0_TftDmCm4RNnjeY6Kld src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:lI2LpKhyvs7u60YM */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:psbYOnr8rl-lJpMG */ match-set cali40s:tiurfVgvw6PC9_fRMfFNDj6 src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:KwcUmYfQqIxdpS6A */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:x-jdw-cvkhR28FNW */ match-set cali40s:imjqvrEKMKpqLZrzKAu36FT src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:Hsb8pKM05IkpzdYw */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:cI2XFKeUEm9KQa9W */ match-set cali40s:2Yo6427DlrjvLGBseShyDRT src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:NBBQTOVAqnMKNgQV */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:Uq2NxDH99Or9JAag */ match-set cali40s:-9TvsX35emtm3sxaCucNt_b src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:c3euPUlopL9_NJDj */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:0XA_73GxWxZ1iWeN */ match-set cali40s:nPC6-PVy3LaTgcaYLJqxedh src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:E2TwNdID0-cNfjnT */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:fnVlBK88e48XjwOQ */ match-set cali40s:R899OQM-Jmtqf2ymqkoPgno src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:vU70kGRep7cfov9p */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:rtRMAwB7_v22L5AP */ match-set cali40s:NVdJqqmVs4DXXvCryBUAyNn src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:klBy4Bk0rkUyUahp */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:wJUz1mi3AgS-qydh */ match-set cali40s:uoojxLjUKwW-hgDb8lBP3lg src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:juqn2M1Aax5r-MrO */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:2F9t-PbdYnatWi8s */ match-set cali40s:kl2JVOwolaQPDeZu8EqNxtY src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:_MctbnUADrOFWwE0 */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:pR5Jc4YFy376vQCD */ match-set cali40s:ecnz1YKTDnTO1FDvMP3Ht_W src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:t7yb_1s6gqGbvPyI */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:_BPYImMLgBEGFaUI */ match-set cali40s:9lmszVRE28IGqAR_PWb2QZe src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:XLCS09i2ZAtkHsIm */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:9RkA59qawRHY_TnV */ match-set cali40s:kseuzzoT8Eh-TX9nLiss9Jz src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:CrV5k0yo_fE8cd9d */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:mYq70XhbPqQITUGR */ match-set cali40s:67yAFdBU0DywRu350kQ9Z89 src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:3PqgrnVk-Jtqchkl */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:Ig2_RAIdMNxLXyUF */ match-set cali40s:yow-VIcxTDMCpI9ja4aGsVU src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:UKP_7Dg7pu2JtITB */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:GpgtUU_4HphTdGcR */ match-set cali40s:dcqqBrVd4U9Plx6chQIvwcv src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:YQWphMyHxrf0fu21 */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:v01fUPB3ByNJOybh */ match-set cali40s:t1FqIDdRWZMcYpC_gtm4ton src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:GAHaE21A46nR_wb5 */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:FBOqnMLMSTlX45We */ match-set cali40s:KtN-ye9CP8cQ8pYVvyHUs-B src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:zx7b4Y_yJpCsmvrG */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:zxgbTQaf60H8vKj_ */ match-set cali40s:edD8UT1lIO_XyzTj6K_qTWw src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:DDgcRbf2dgskPCf8 */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:Y-2HDIo_yxc05IuK */ match-set cali40s:5Pb5kkExMbKkuDf4WLjRNw3 src MARK or 0x10000
RETURN     all  --  anywhere             anywhere             /* cali:4B_etYv9nSSHkGwK */ mark match 0x10000/0x10000
MARK       all  --  anywhere             anywhere             /* cali:KLr6nwFWghnq1md1 */ match-set cali40s:23tMYePOksLD5QceRY4aPD4 src MARK or 0x10000
```

Layer3/4 policy testing for TCP

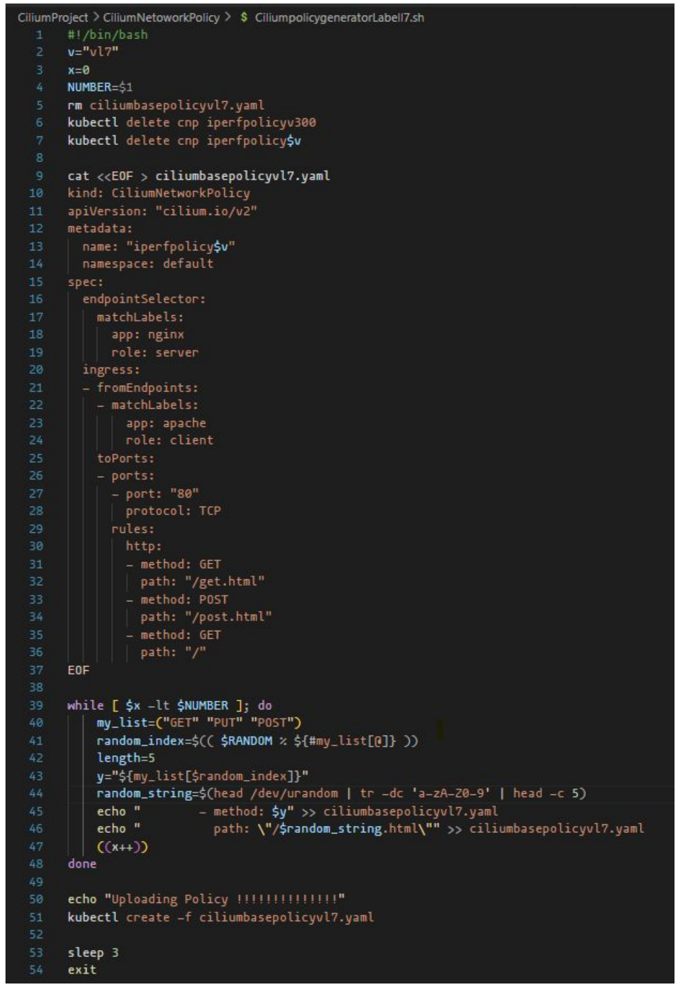
- Initial noticeable delay when starting the iperf experiment with the policies with Calico
- For Calico, the first few packets show least throughput but continue to improve slightly with time. This can be explained by the connection tracking Calico uses.
- Cilium throughput remains the same in both cases.

Layer3/4 based policy testing for UDP



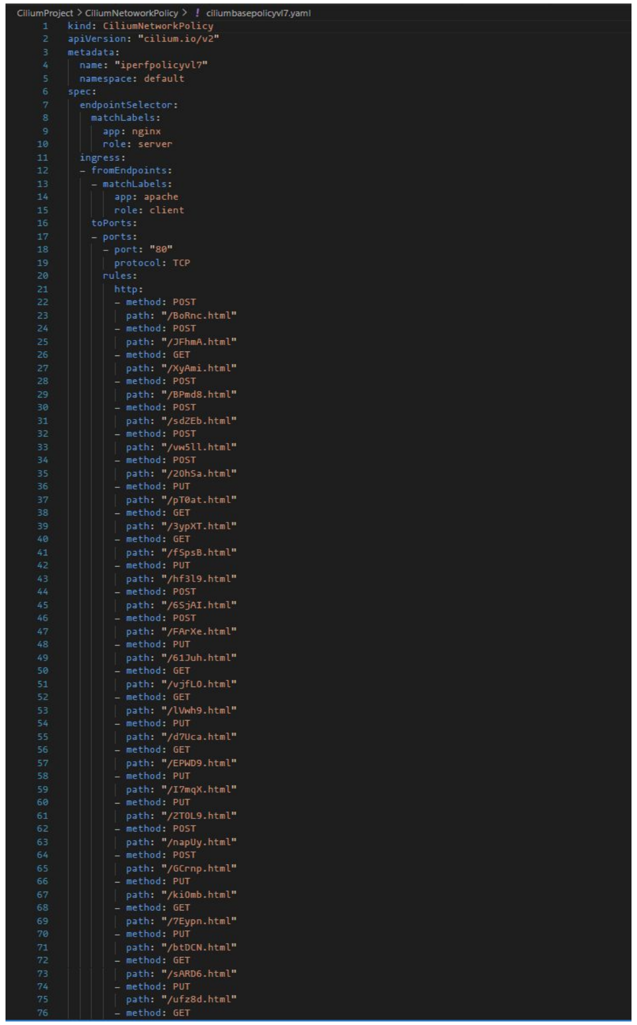


Scripts to Generate L7 Policies



CiliumProject > CiliumNetoworkPolicy > $ CiliumpolicygeneratorLabell7.sh

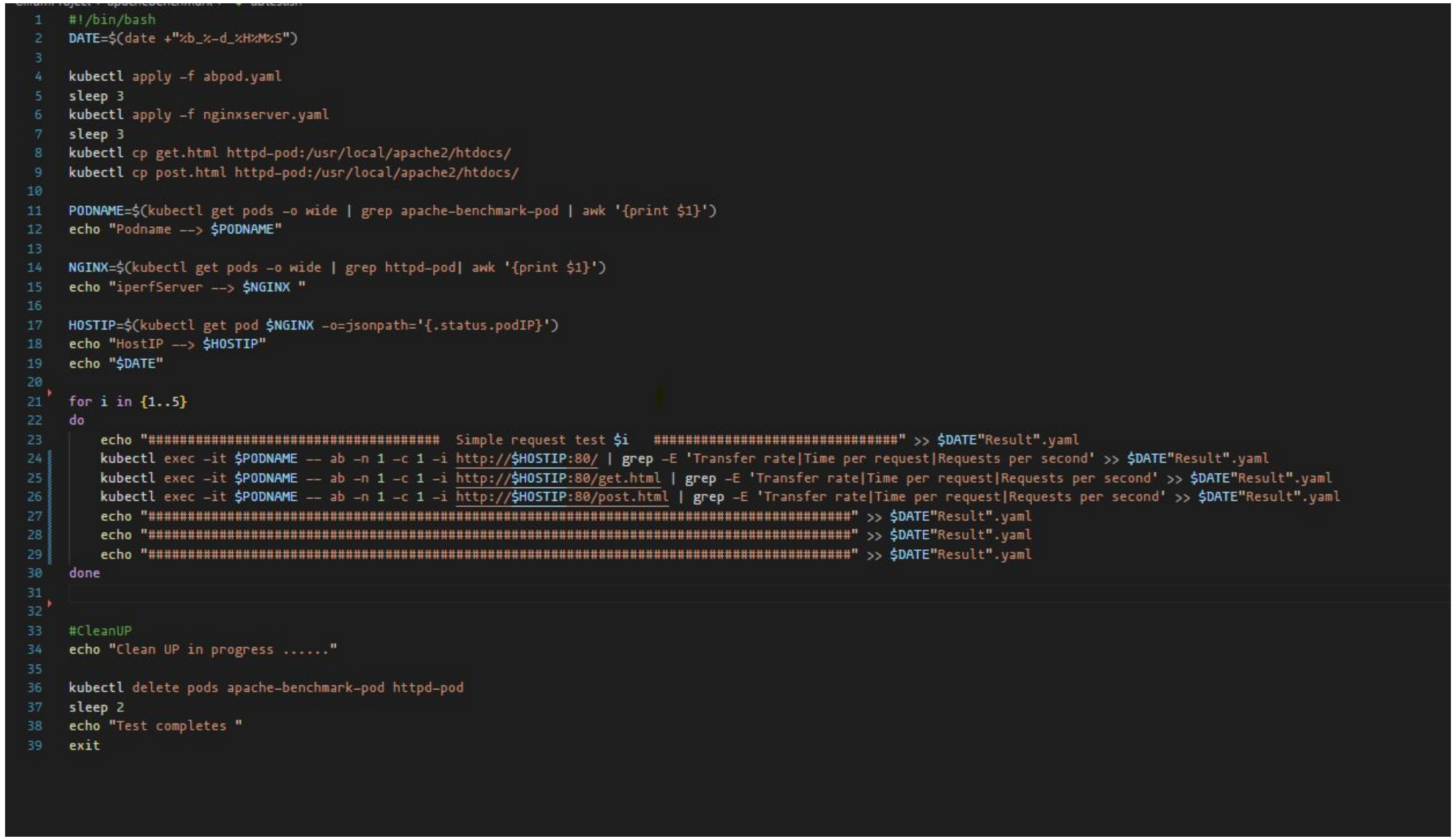
```bash
#!/bin/bash
v="vl7"
x=0
NUMBER=$1
rm ciliumbasepolicyvl7.yaml
kubectl delete cnp iperfpolicyv300
kubectl delete cnp iperfpolicy$v

cat <<EOF > ciliumbasepolicyvl7.yaml
kind: CiliumNetworkPolicy
apiVersion: "cilium.io/v2"
metadata:
  name: "iperfpolicy$v"
  namespace: default
spec:
  endpointSelector:
    matchLabels:
      app: nginx
      role: server
  ingress:
  - fromEndpoints:
    - matchLabels:
        app: apache
        role: client
    toPorts:
    - ports:
      - port: "80"
        protocol: TCP
      rules:
        http:
        - method: GET
          path: "/get.html"
        - method: POST
          path: "/post.html"
        - method: GET
          path: "/"
EOF

while [ $x -lt $NUMBER ]; do
    my_list=("GET" "PUT" "POST")
    random_index=$(( $RANDOM % ${#my_list[@]} ))
    length=5
    y="${my_list[$random_index]}"
    random_string=$(head /dev/urandom | tr -dc 'a-zA-Z0-9' | head -c 5)
    echo "        - method: $y" >> ciliumbasepolicyvl7.yaml
    echo "          path: \"/$random_string.html\"" >> ciliumbasepolicyvl7.yaml
    ((x++))
done

echo "Uploading Policy !!!!!!!!!!!!!!"
kubectl create -f ciliumbasepolicyvl7.yaml

sleep 3
exit
```



CiliumProject > CiliumNetoworkPolicy > ! ciliumbasepolicyvl7.yaml

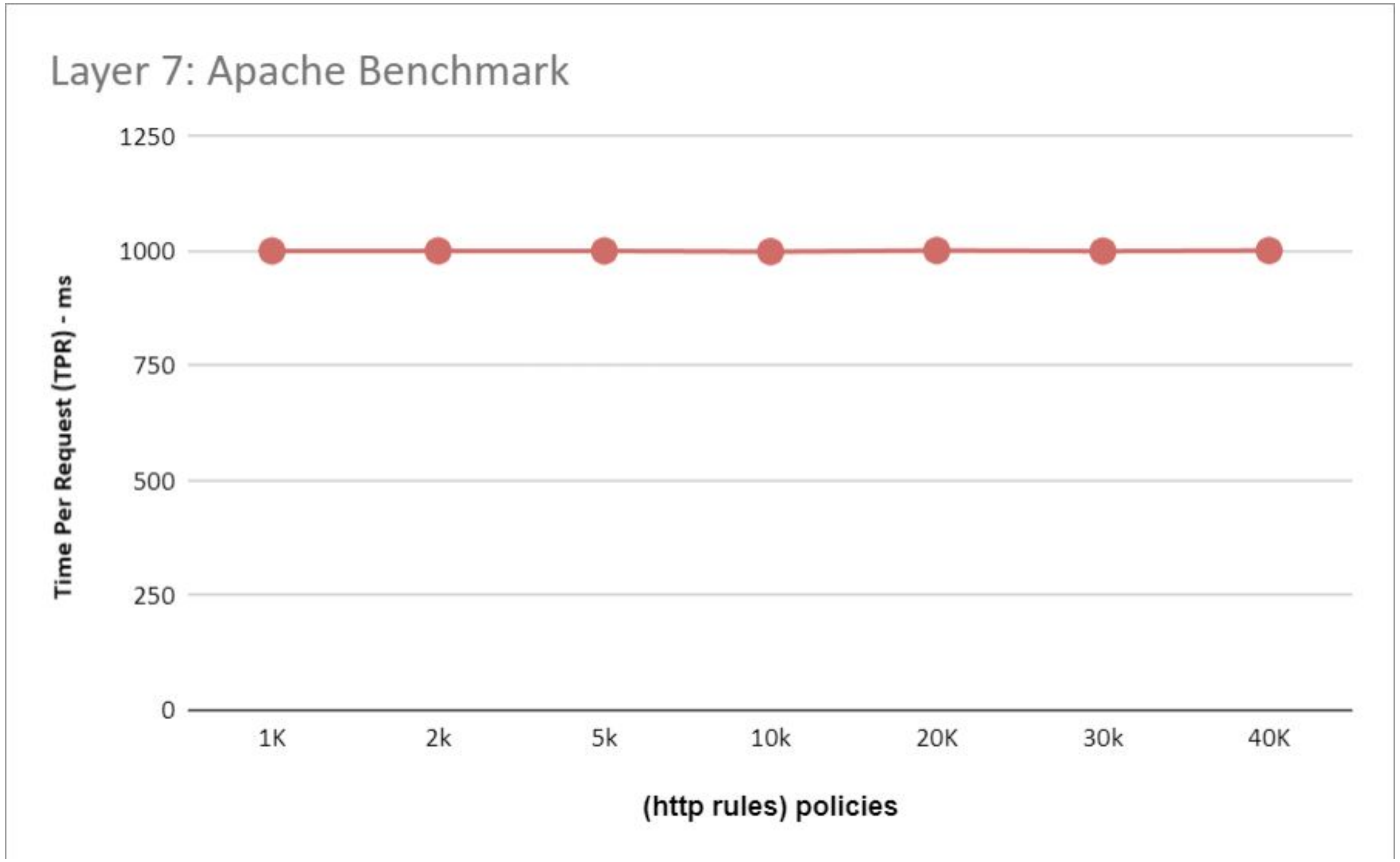
```yaml
kind: CiliumNetworkPolicy
apiVersion: "cilium.io/v2"
metadata:
  name: "iperfpolicyvl7"
  namespace: default
spec:
  endpointSelector:
    matchLabels:
      app: nginx
      role: server
  ingress:
  - fromEndpoints:
    - matchLabels:
        app: apache
        role: client
    toPorts:
    - ports:
      - port: "80"
        protocol: TCP
      rules:
        http:
        - method: POST
          path: "/BoRnc.html"
        - method: POST
          path: "/JFhmA.html"
        - method: GET
          path: "/XyAmi.html"
        - method: POST
          path: "/BPmd8.html"
        - method: POST
          path: "/sdZEb.html"
        - method: POST
          path: "/vw5ll.html"
        - method: POST
          path: "/2OhSa.html"
        - method: PUT
          path: "/pT0at.html"
        - method: GET
          path: "/3ypXT.html"
        - method: GET
          path: "/fSpsB.html"
        - method: PUT
          path: "/hf3l9.html"
        - method: POST
          path: "/6SjAI.html"
        - method: POST
          path: "/FArXe.html"
        - method: PUT
          path: "/61Juh.html"
        - method: GET
          path: "/vjfLO.html"
        - method: GET
          path: "/lVwh9.html"
        - method: PUT
          path: "/d7Uca.html"
        - method: GET
          path: "/EPWD9.html"
        - method: PUT
          path: "/I7mqX.html"
        - method: PUT
          path: "/ZTOL9.html"
        - method: POST
          path: "/napUy.html"
        - method: POST
          path: "/GCrnp.html"
        - method: PUT
          path: "/kiOmb.html"
        - method: GET
          path: "/7Eypn.html"
        - method: PUT
          path: "/btDCN.html"
        - method: GET
          path: "/sARD6.html"
        - method: PUT
          path: "/ufz8d.html"
        - method: GET
```

Apache-Benchmark Testing Script



```bash
#!/bin/bash
DATE=$(date +"%b_%-d_%H%M%S")

kubectl apply -f abpod.yaml
sleep 3
kubectl apply -f nginxserver.yaml
sleep 3
kubectl cp get.html httpd-pod:/usr/local/apache2/htdocs/
kubectl cp post.html httpd-pod:/usr/local/apache2/htdocs/

PODNAME=$(kubectl get pods -o wide | grep apache-benchmark-pod | awk '{print $1}')
echo "Podname --> $PODNAME"

NGINX=$(kubectl get pods -o wide | grep httpd-pod| awk '{print $1}')
echo "iperfServer --> $NGINX "

HOSTIP=$(kubectl get pod $NGINX -o=jsonpath='{.status.podIP}')
echo "HostIP --> $HOSTIP"
echo "$DATE"

for i in {1..5}
do
    echo "#####################################  Simple request test $i   ###############################" >> $DATE"Result".yaml
    kubectl exec -it $PODNAME -- ab -n 1 -c 1 -i http://$HOSTIP:80/ | grep -E 'Transfer rate|Time per request|Requests per second' >> $DATE"Result".yaml
    kubectl exec -it $PODNAME -- ab -n 1 -c 1 -i http://$HOSTIP:80/get.html | grep -E 'Transfer rate|Time per request|Requests per second' >> $DATE"Result".yaml
    kubectl exec -it $PODNAME -- ab -n 1 -c 1 -i http://$HOSTIP:80/post.html | grep -E 'Transfer rate|Time per request|Requests per second' >> $DATE"Result".yaml
    echo "#########################################################################################" >> $DATE"Result".yaml
    echo "#########################################################################################" >> $DATE"Result".yaml
    echo "#########################################################################################" >> $DATE"Result".yaml
done


#CleanUP
echo "Clean UP in progress ......"

kubectl delete pods apache-benchmark-pod httpd-pod
sleep 2
echo "Test completes "
exit
```

Layer 7 - Policy Testing



a. Cilium uses hashing based L7 ip mapping
b. consistent with our findings as TPR remains same for increasing number of requests

Summary of the Results

- Throughput: Cilium maintains a constant throughput as the number of policies increases, whereas Calico's throughput decreases due to iptables overhead.
- UDP Performance: Cilium demonstrates better performance in terms of packet loss and throughput consistency compared to Calico, which experiences higher packet loss.
- Time per request (without policies): Both Cilium and Calico have similar performance when there are no policies in place.
- L7 Policy Performance: Cilium is able to maintain constant throughput with an increasing number of L7 policies.

Conclusion

- Cilium's use of eBPF and XDP technologies provides fast and efficient packet filtering and forwarding, while its hash-based approach to L7 IP rule mapping allows for efficient rule matching.
- These results suggest that Cilium's eBPF-based approach provides significant performance advantages over traditional CNIs like Calico, particularly in terms of throughput consistency, UDP performance, and L7 policy handling.

Future work

- Perform similar testing with increased number of cores. This can help improve performance - cpu utilization, reduce packet loss, etc
- Perform testing directly on instances (bare metal) compared to VMs and compare results of both
- Use custom built linux kernels which allows to more finely tune network configurations

References

- eBPF
- CNI Benchmark: Understanding Cilium Network Performance
- Assessing Container Network Interface Plugins: Functionality, Performance, and Scalability -

  https://ieeexplore.ieee.org/document/9309003
- Performance Benchmarking and Tuning for Container Networking on Arm
- Cilium Documentation : Performance Evaluation — Cilium 1.9.15 documentation
- Similar experimentation: https://kinvolk.io/blog/2020/12/egress-filtering-benchmark-part-2-calico-and-cilium/
- GIthub : https://github.com/anvayabn/CiliumProject
-

Thank you