Empowering Cloud Native Networking with Arm Ecosystem

Trevor Tao（Zijin Tao）, Hanyu Ding

Jianlin Lv, Song Zhu, Jingzhao Ni

Nov/2020

Agenda

- Introduction
- Reference Cloud Native Networking (CNN) Stack on Arm
- High Performance CNIs for CNN on Arm
- Multi-Interface Requirement with SRIOV support on Arm
- Performance Tests for various CNIs on Arm platform
- Cloud Native Networking Use Cases and Deployment on Arm
- Future Work(Provisional)

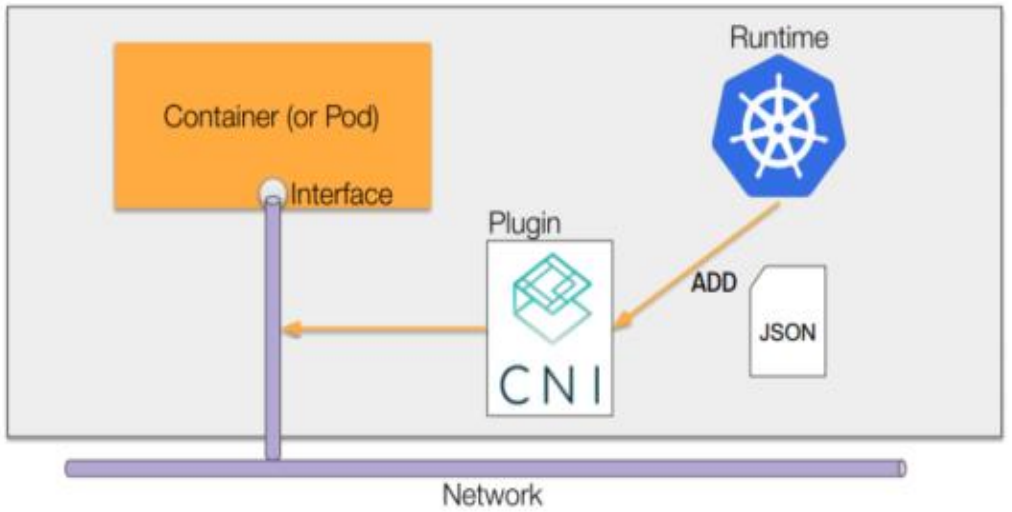
Introduction

arm

Introduction

What is Cloud Native Networking

- Cloud native computing means to use containerized open source software stack , where each part of the app is packaged in its own container, dynamically orchestrated so each part is actively scheduled and managed to optimize resource utilization
- Cloud native networking is an approach to provide the networking environment for building and running applications that exploit the advantages of the cloud computing delivery model.
- So high performance, flexible and function rich container networking technology is the key to success of cloud native computing
- Various CNIs (*Container Networking Interface*) are emerging to support cloud native networking
  - A Cloud Native Computing Foundation project, consists of a specification and libraries for writing plugins to configure network interfaces in Linux containers, along with a number of supported plugins



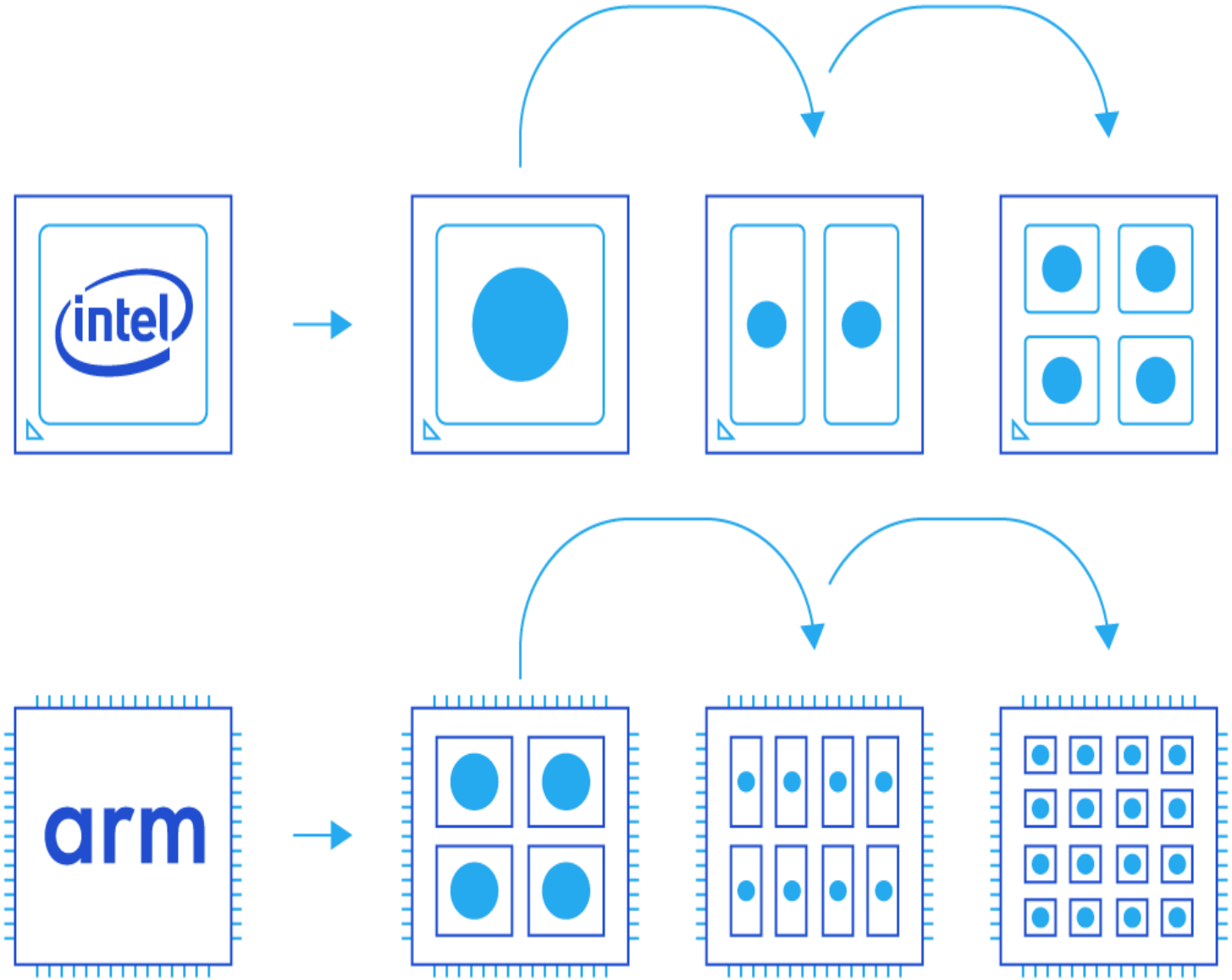
CNIs follows Kubernetes networking model:

- All Pods can communicate with all other Pods without using network address translation (NAT).
- All Nodes can communicate with all Pods without NAT.
- The IP that a Pod sees itself as is the same IP that others see it as.

- Networking objects
  - Container-to-Container networking
  - Pod-to-Pod networking
  - Pod-to-Service networking
  - Internet-to-Service networking

arm

Cloud Native Computing with Arm

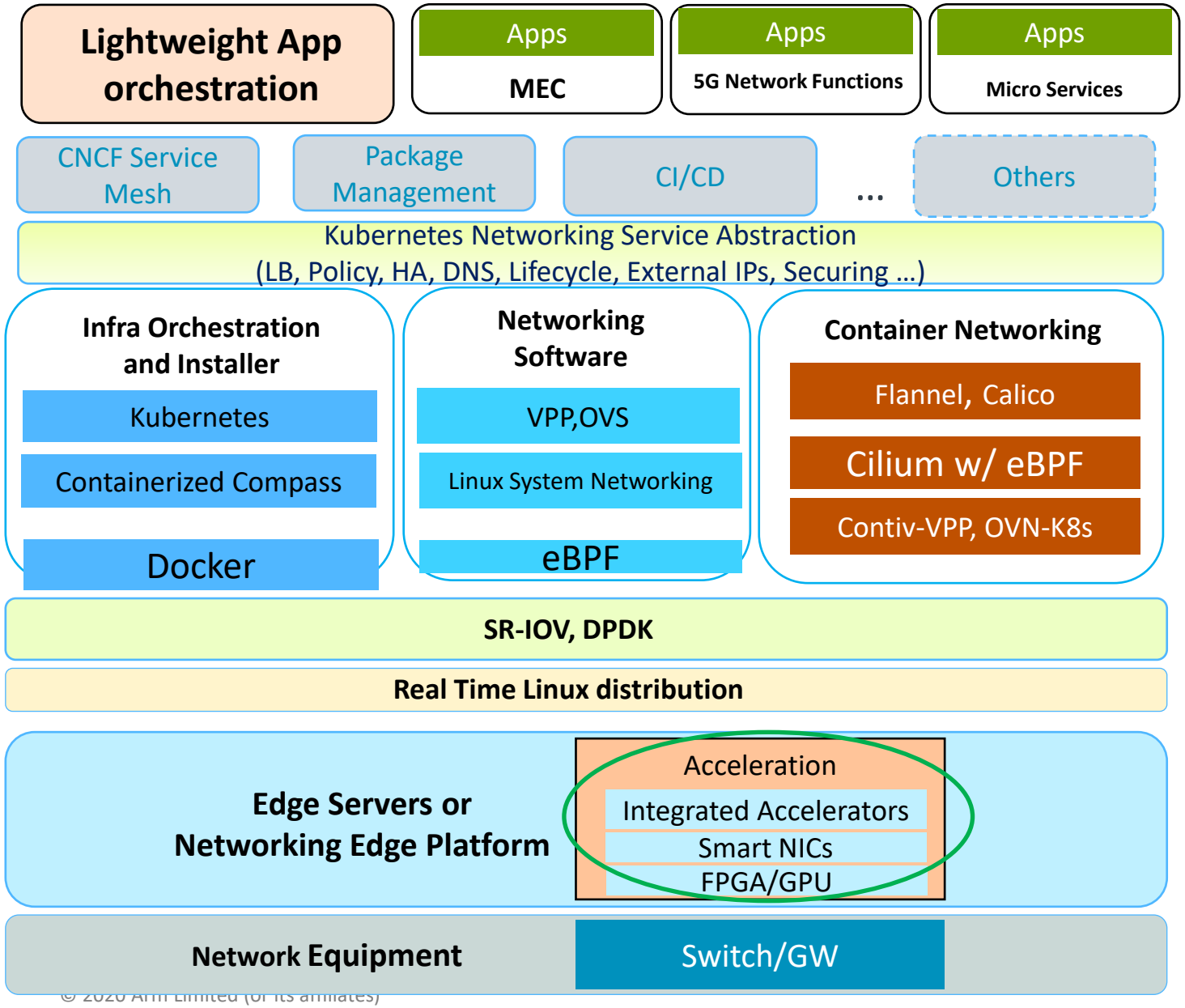


Arm CPU: Not just mobile CPU architecture for Data Centers (Ref)

Reference Cloud Native Networking Stack on arm

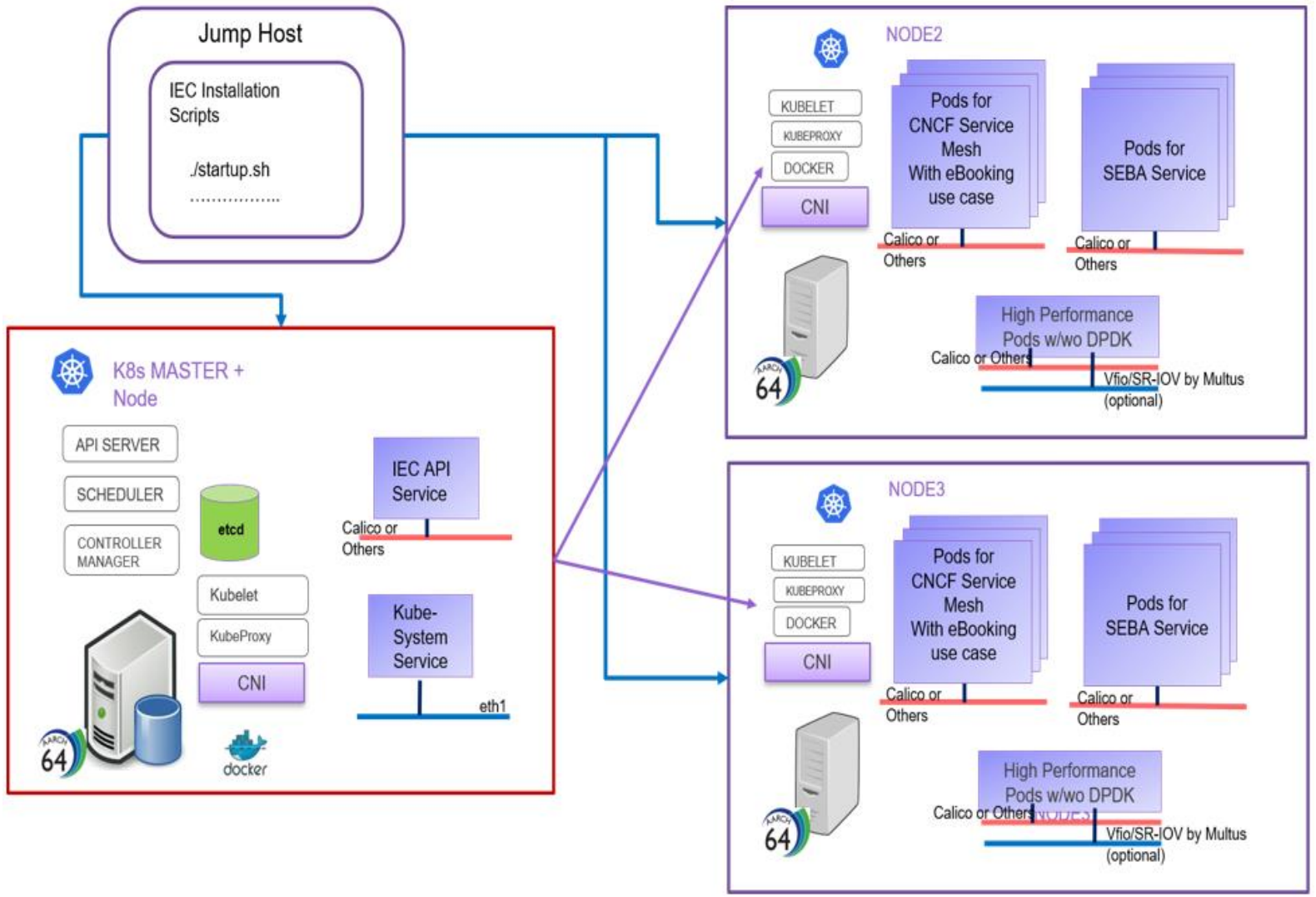
arm

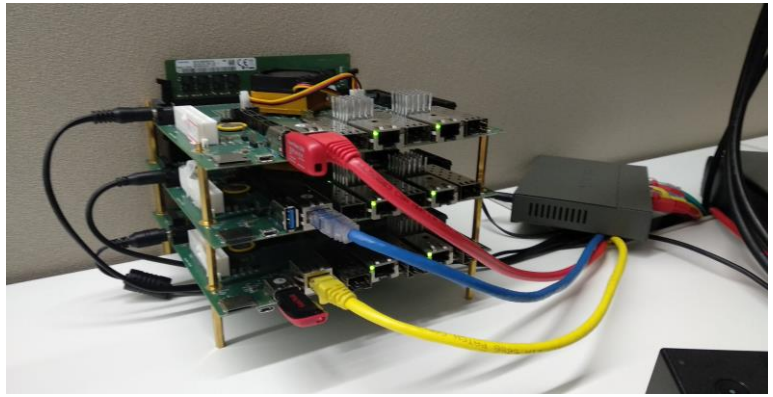
Reference Cloud Native Networking Stack



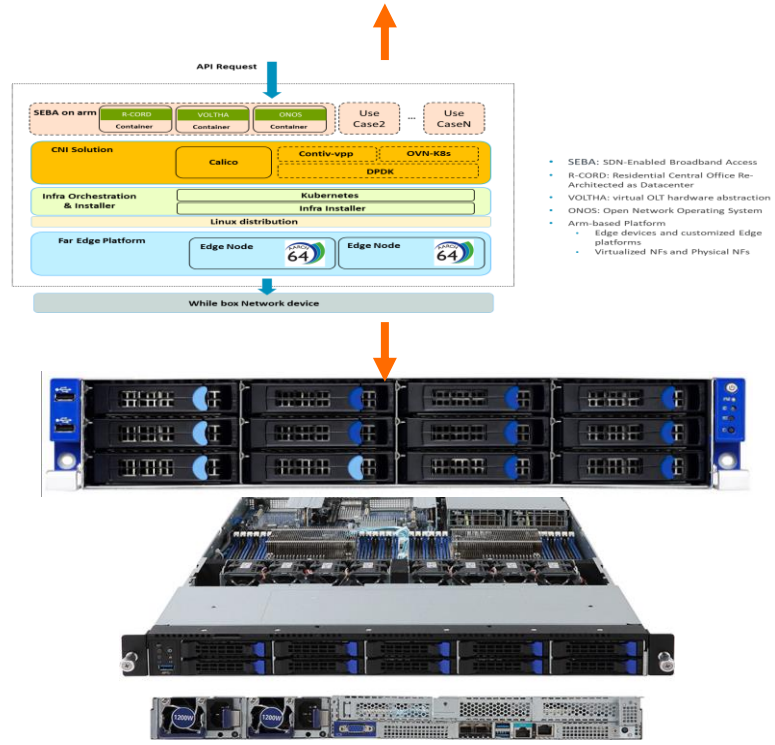
- Cloud native oriented architecture
  - Container based
  - Servers and customized Cloud platforms
  - Containerized/Virtualized NFs vs Physical NFs
  - Accelerator interface
- Resource constraints
  - Kubernetes
  - CNIs for Kubernetes
- Supporting network technologies
  - L2: Linux ethernet driver, bridge, DPDK, virtual switching(OVS, VPP)
  - L3: IPtables
  - Overlay tunnel: IPIP, VXLAN, NVGRE
  - L2/3: Kernel JIT: eBPF (programs, maps, registers, helper functions, ...)
  - L7: HTTP Proxy(ENVOY)
- HW Accelerations
  - Integrated accelerators
  - PCIe/CCIX attached accelerator (Smart NICs...)
  - OVS/eBPF offload with SmartNIC

Reference Cloud Native Networking Cluster on Arm





SEBA Cluster on small Arm Devices



SEBA Cluster on Arm Servers

High Performance CNIs for CNN on Arm

arm

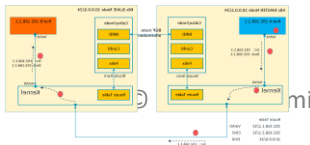
High Performance CNIs available for Arm Cloud Native Stack

Things now available in Akraino IEC Blueprint Arm edge stack as a ref:

IEC Arm Reference Stack

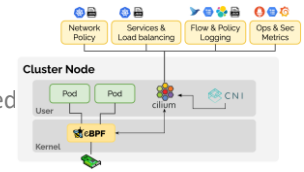
Calico

- pure IP networking fabric
- high-level network policy management by iptables
- Good scalability
- Support direct(non-overlay) and overlay(IPINIP, VxLAN) network connection
- Easy deployment
- Calico-VPP appears



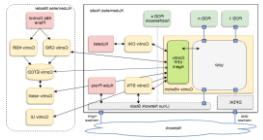
Cilium

- Linux-Native, API-Aware Networking and Security for Containers
- Linux eBPF based network policy, load balance and security which is believed to be with incredible performance
- L3 networking between hosts
- Good scalability too



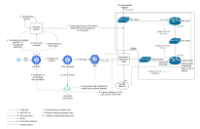
Contiv-VPP

- uses FD.io VPP to provide network connectivity between PODs
- Native DPDK interface support for phy NIC
- Native VPP ACL/NAT based network policy and access
- Good performance but with rather complex configuration
- DPDK based



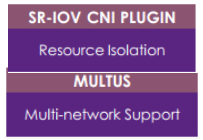
OVN-K8s

- OVS/OVN-controller based K8s networking solution
- Rather good performance with OVS inherited
- Use OVN logical switches/routers to connect Pods and for outside access
- No OVS-DPDK support now



Multus/ SRIOV

- Direct physical interfaces(**PF/VFs**) support for Pods
- High performance with direct Linux kernel eth driver or DPDK PMD driver
- Usually co-work with other CNIs, such as Flannel, Calico by Multus or other glue CNI
- Need resource description or annotation when do the configuration for CNI and Pod setup



Flannel

- Widely used and almost easiest deployment for a simple K8s networking
- Linux network bridge for pod connection and overlay based communication for inter-hosts access
- Easy to be integrated into other container networking solution, e.g., Cilium
- No good network policy support

flannel

Code Review / iec.git / tree

summary | shortlog | log | commit | commitdiff | review | tree
history | HEAD | snapshot

fix issue that deploy cilium on latest image

[iec.git] / src / foundation / scripts / cni /

drwxr-xr-x	..		
drwxr-xr-x	-	calico	tree | history
drwxr-xr-x	-	cilium	tree | history
drwxr-xr-x	-	contivpp	tree | history
drwxr-xr-x	-	danm	tree | history
drwxr-xr-x	-	flannel	tree | history
drwxr-xr-x	-	multus	tree | history
drwxr-xr-x	-	ovn-kubernetes	tree | history

Repo: https://gerrit.akraino.org/r/admin/repos/iec

Status of CNIs on Arm Platform and Our Contributions (only a small part of)

- Calico officially announced to be supported on arm platform
  - announced on the Calico blog back: https://www.projectcalico.org/announcing-calico-v3-2/
  - https://docs.projectcalico.org/archive/v3.2/releases#multiple-architectures
  - https://github.com/projectcalico/calico/issues/3783

Multiple Architectures

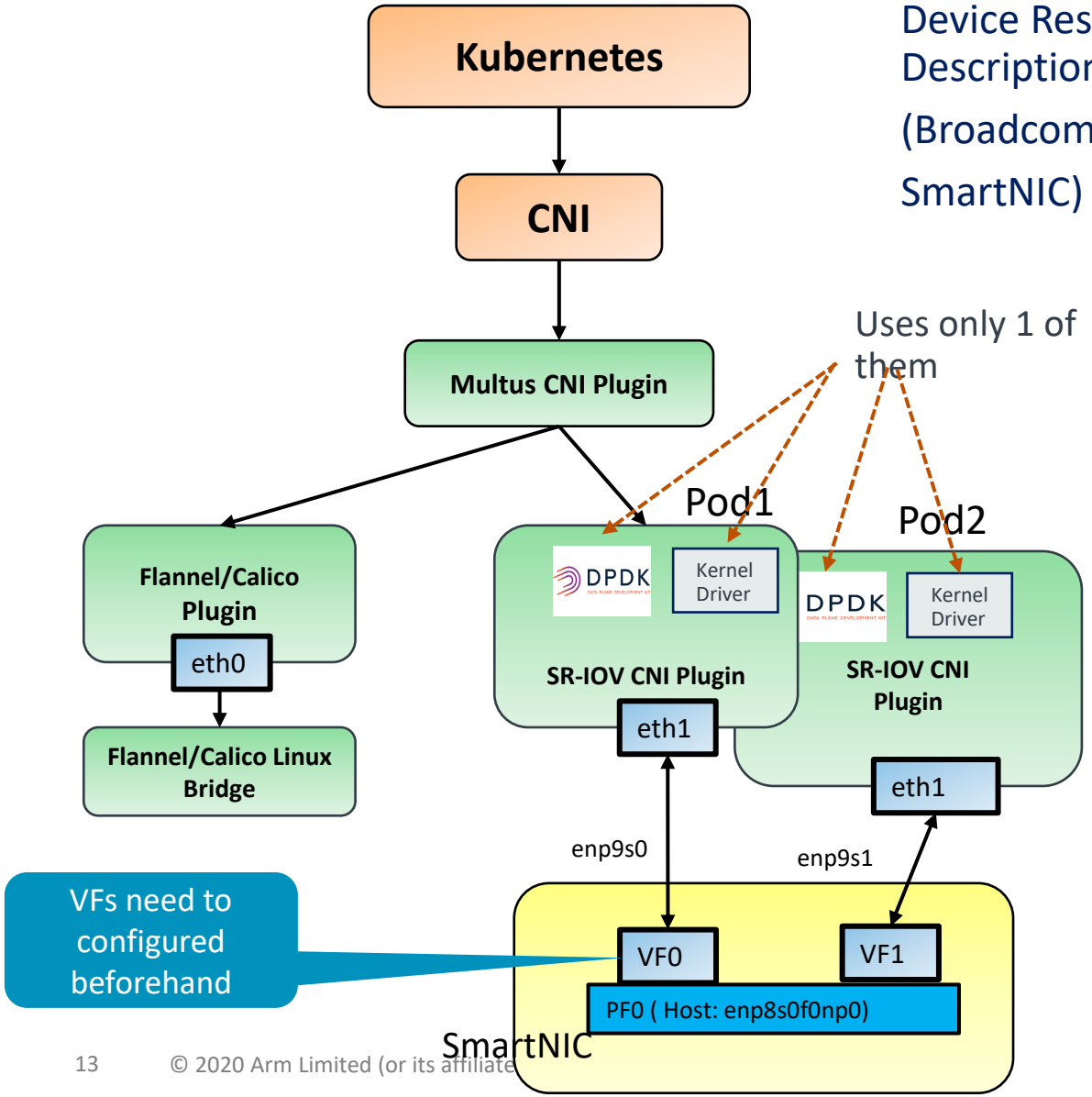
Thanks to concerted efforts from the community, images are now available for all components for the arm64 and ppc64le architectures.

- Cilium CNI (v1.8 and above) are generally available on arm platform
  - Ported Cilium master version on Arm server and deployed Nginx to verify and test k8s cluster
  - Fix compiling and running issue on Arm64;
  - Added arm64 support for building cilium-related images(cilium-agent, cilium-operator, cilium-runtime, cilium-builer, cilium-proxy,…);
  - Add multi-arch support for cilium-related images;
  - Add arm64 support for cilium CI/CD.
- Contiv-VPP CNI
  - Add arm64 support for Contiv/VPP docker images building to the upstream
- OVN-Kubernetes CNI
  - **Add arm64 support for daemonset image build**
  - **Add arm64 and multi-arch support for ubuntu image**
  - Add scripts for daemonset installation
  - **Upgrade ovs version in Dockerfile to 2.11**
  - **Remove the installation of containernetwork-cni**
  - **Change the value for selecting the master node**
- Multus CNI
  - Support for Multus CNI image building on Arm

Multi-Interface Requirement with SRIOV and SmartNIC support on Arm

arm

Multiple Interfaces Support by SRIOV CNI



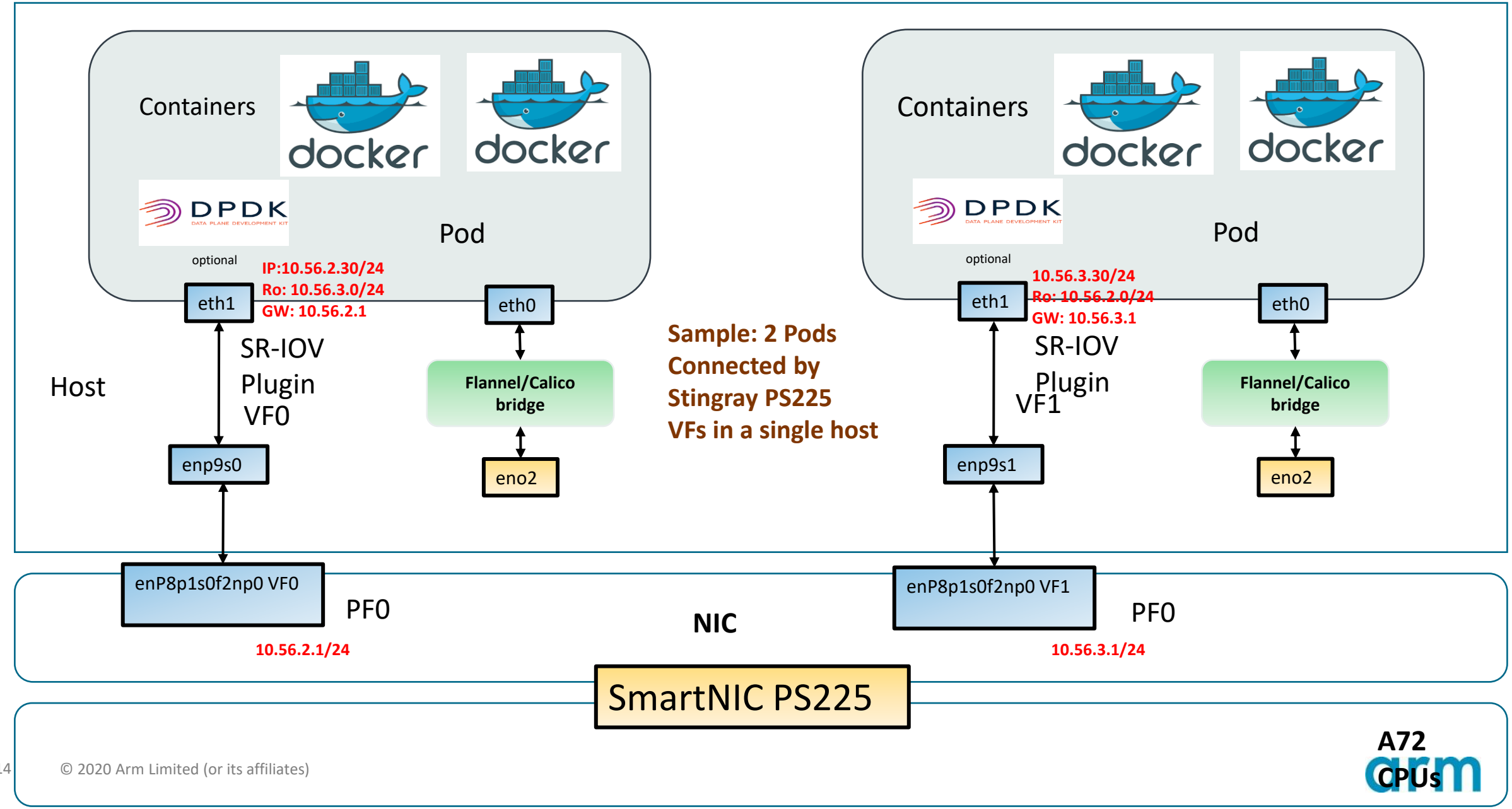
Device Resource Description (Broadcom PS225 SmartNIC)

apiVersion: v1
kind: ConfigMap

For DPDK device driver:

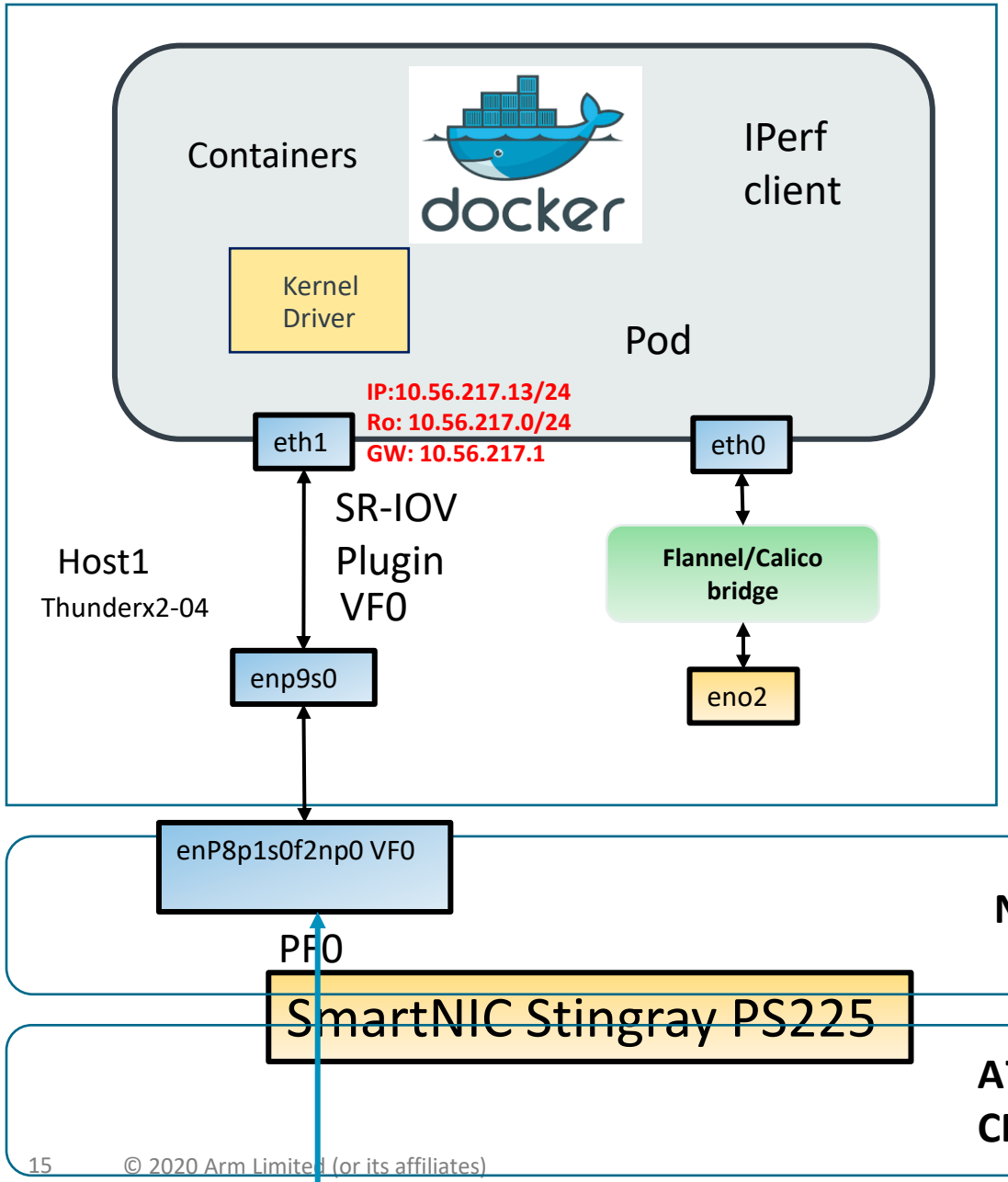
```
# An example of Pod spec using above SR-IOV CRD:
apiVersion: v1
kind: Pod
metadata:
  name: testpod1
  labels:
    env: test
  annotations:
    k8s.v1.cni.cncf.io/networks: sriov-net1
spec:
  containers:
  - name: appcntr1
    image: iecedge/centos-tools-arm64
    imagePullPolicy: IfNotPresent
    command: [ "/bin/bash", "-c", "--" ]
    args: [ "while true; do sleep 300000; done;" ]
    resources:
      requests:
       arm.com/ps225_sriov_netdevice: '1'
      limits:
       arm.com/ps225_sriov_netdevice: '1'
```

Networking Model for Containers in a Single Host by SRIOV CNI

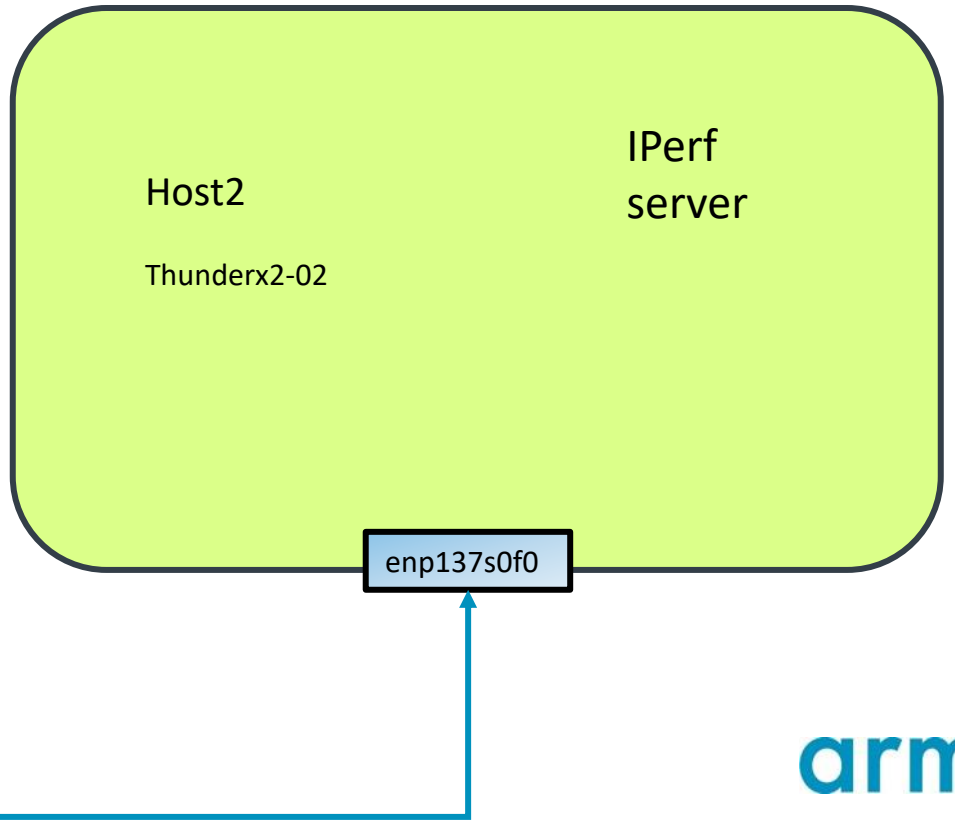


Performance Test with SRIOV CNI

Container Networking by SRIOV Interface for Inter Host Communication



Perf Test2: 1 Pod Connected with another host via SmartNIC VF

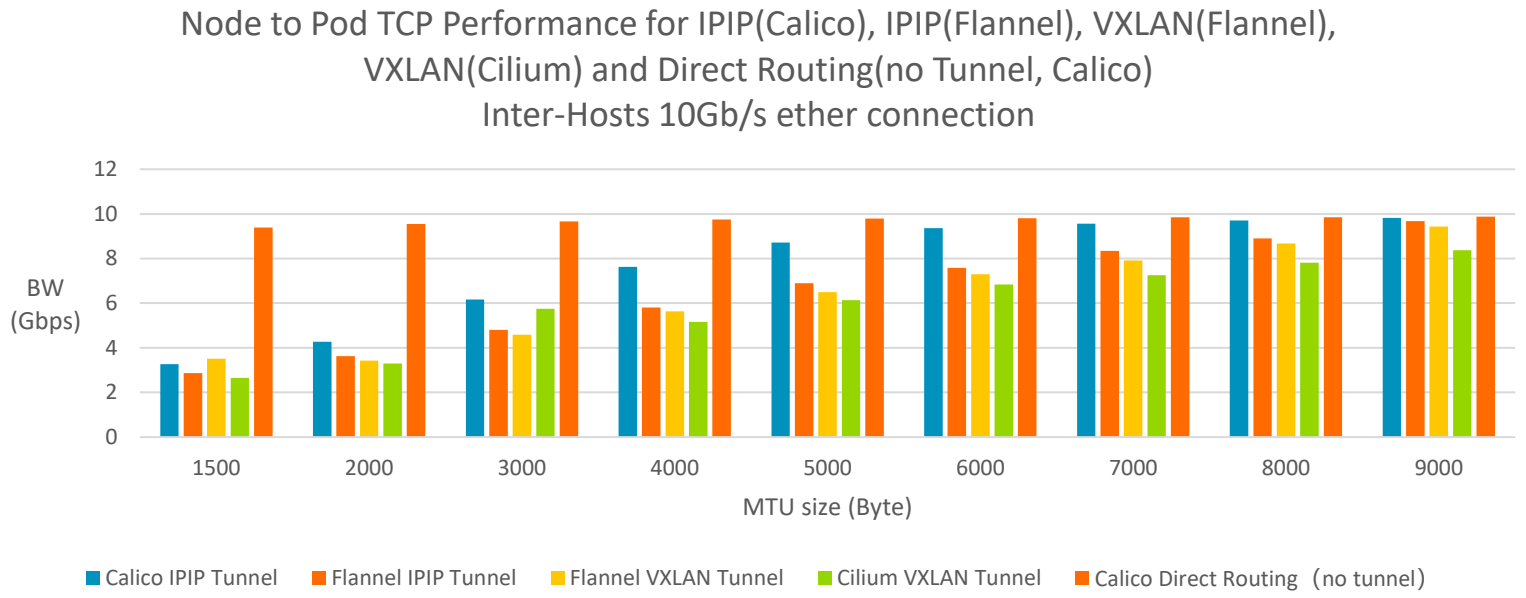


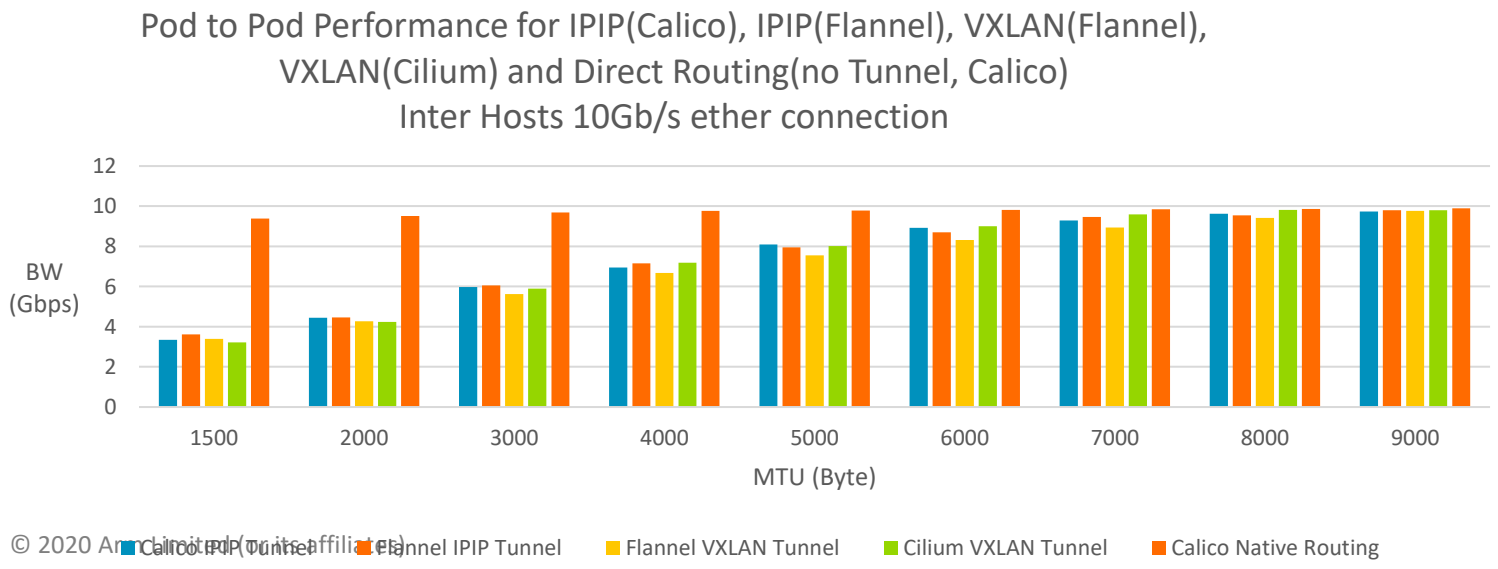
arm

Performance Tests for various CNIs on Arm platform

arm

Benchmarking Results of TCP Throughput for CNIs with Different Backends





Observation for TCP performance over CNIs

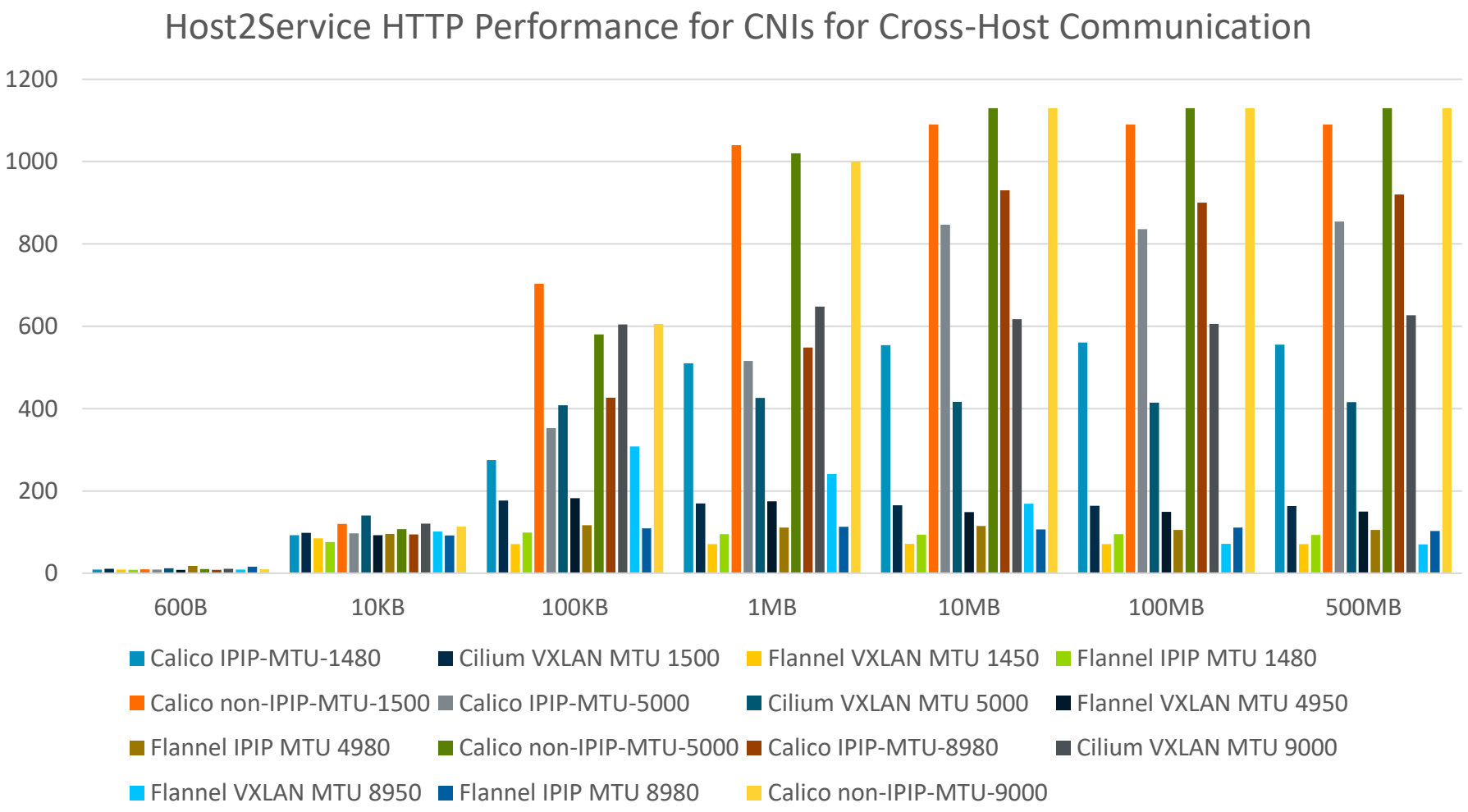
- The performance gap between CNIs are not so explicit when overlay tunnel is used;
- Calico and Flannel show a little bit better performance than Cilium for most MTUs here
- With IPIP/VXLAN overlay tunnel enabled, the larger MTU size, the throughput(BW) performance is better.
- When use direct routing(here by Calico, Cilium also support this mode), the throughput performance is not significantly affected by MTU size.
- The performance of direct routing here by Calico, Cilium also support this mode) is better than IPIP enabled.
- The IPIP tunnel is a little better than VXLAN tunnel
- In general, the node to pod TCP performance is better than that of pod 2 pod which flows one more step (of veth connection to the Linux kernel) .

Finally, compared with different scenarios, **it proves that IPIP/VXLAN overlay tunnel which are now implemented in the Linux kernel is the key factor which affects the performance of CNIs on arm**

Question:

Why the node to pod performance is no better than that of pod to pod case for Cilium?

HTTP Performance Benchmarking for CNIs with various backends

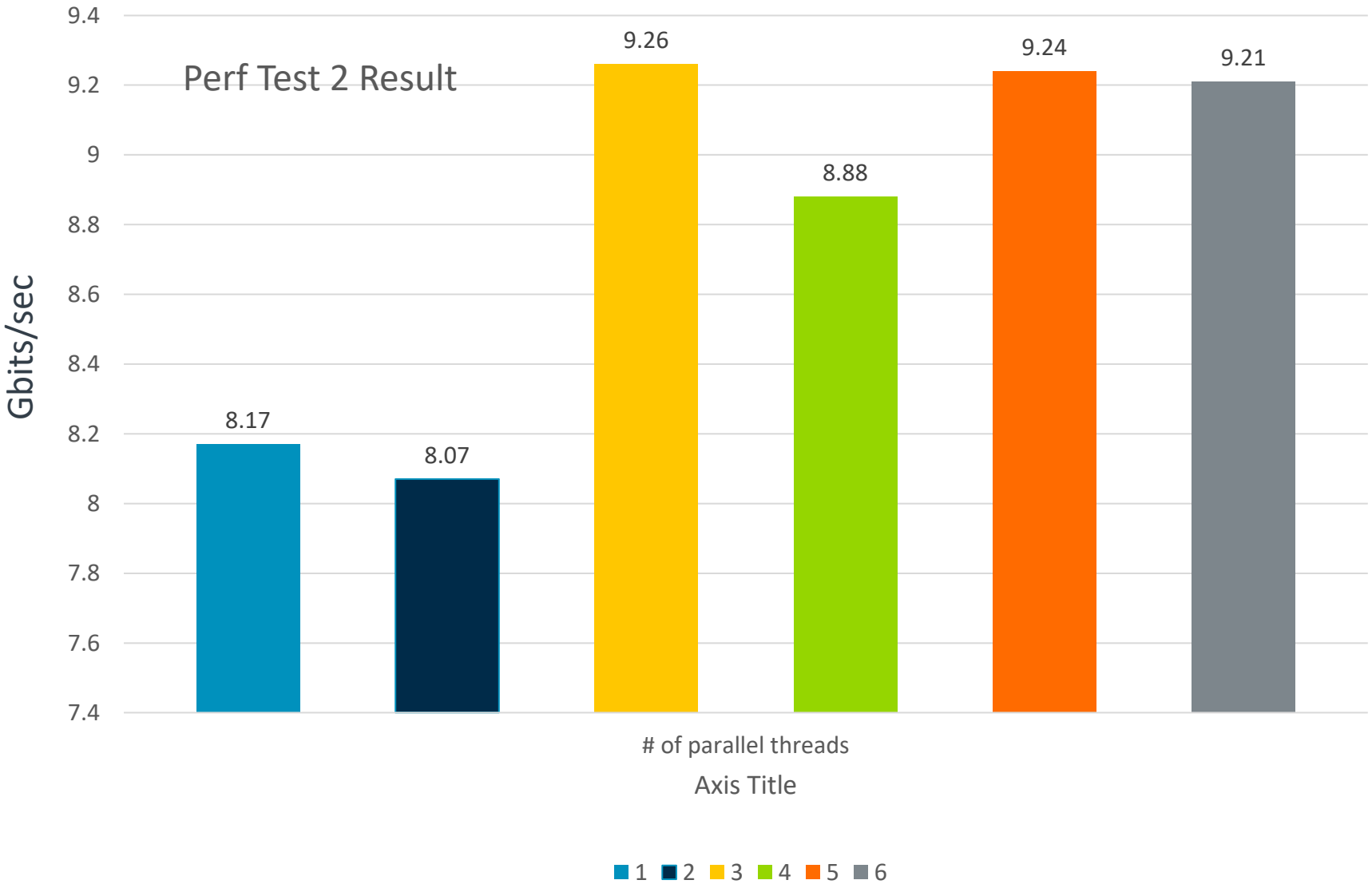


Observation:

- For 3 CNIs, the performance gap is minor when accessing small files
- As previous, the direct routing (no tunnel) mode shows the best performance with any other tunnel based approaches;
- For file size >= 100KB, the Calico shows explicitly the best performance over other 2 CNIs
- Flannel shows the worst host2service performance over other 2 CNIs, even with bigger MTUs, for either IPIP tunnel or VXLAN tunnel
- For large MTU and large file size, Cilium shows similar performance with the Calico CNI
- For non-IPIP, the performance gap between different MTU is not so explicit, so it's believed that the tunnel communication is actually the bottleneck, which is the same as previous

Performance Test Result

Pod Communication Performance by SRIOV CNI with another host via PS225 VF



arm

Cloud Native Networking Use Cases and Deployment on Arm

arm

Service Mesh -- Istio Introduction

Istio Architecture

- Control plane
  - Mixer

    Policy enforcement and telemetry checks
  - Pilot

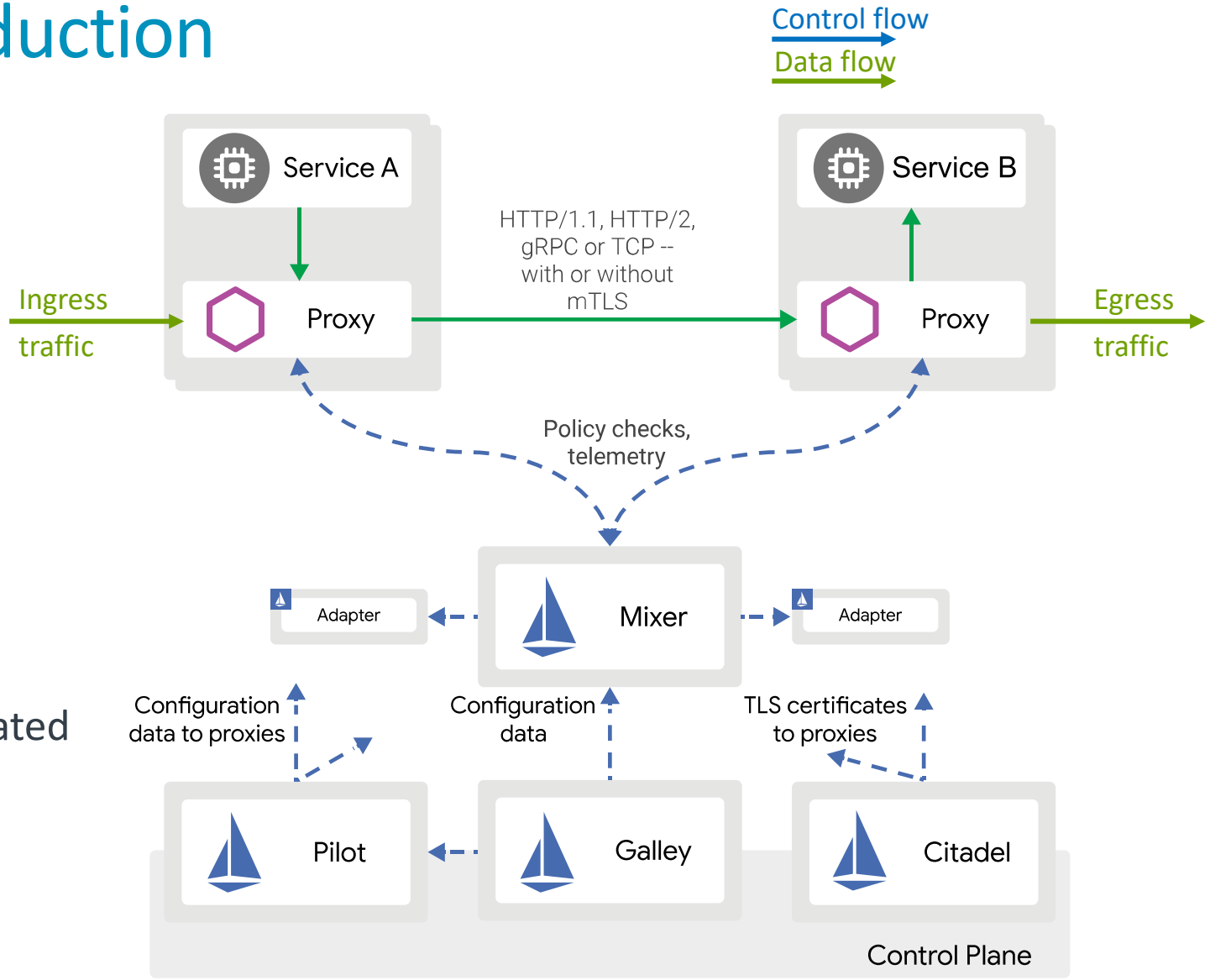
    Service communication policies configuration
  - Galley

    Control plane configuration
  - Citadel

    Security and credential management

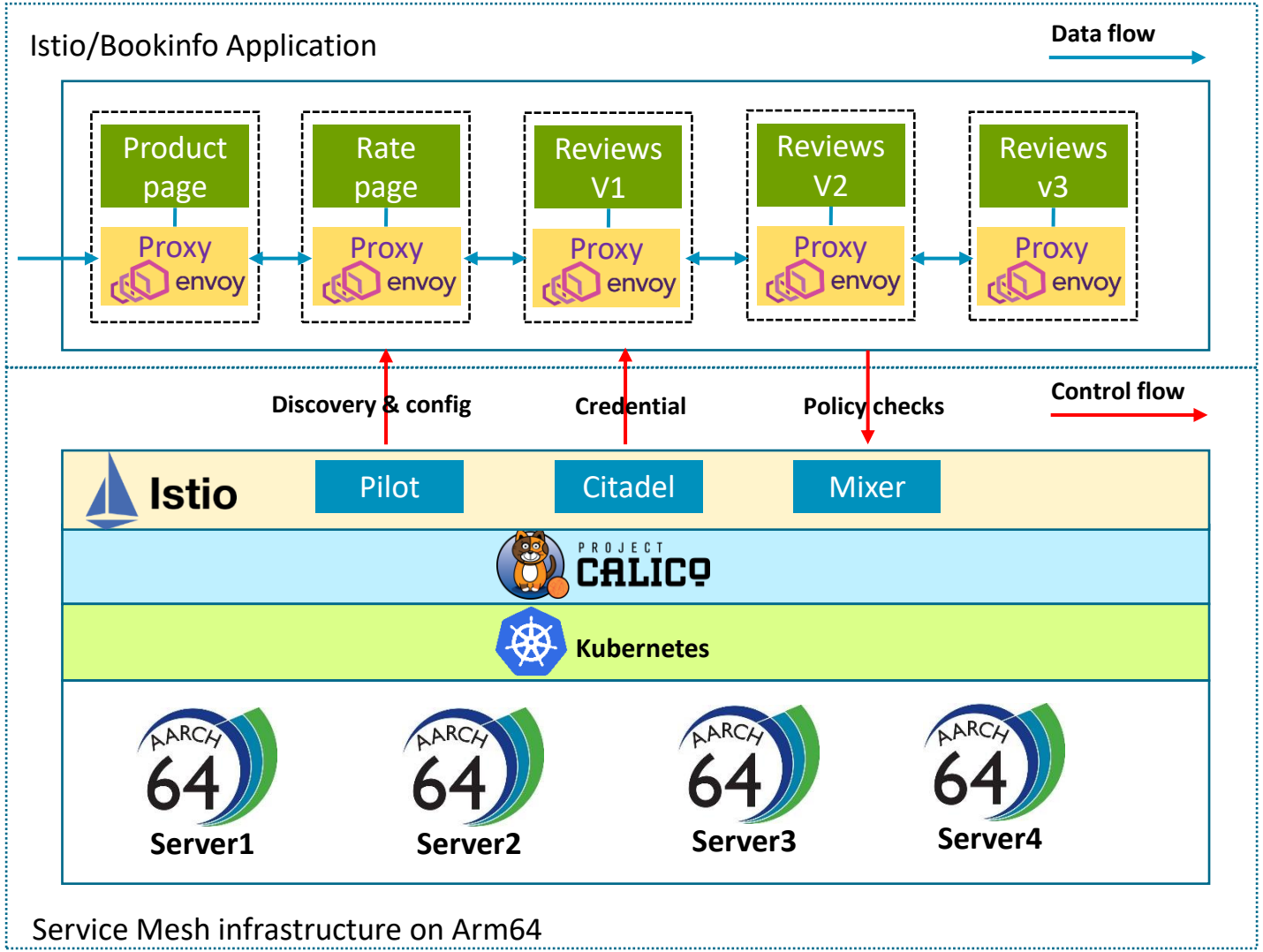
  In latest version, those modules are integrated into one component -- istiod
- Data Plane
  - Proxy (Envoy)

    High-performance proxy developed in C++



Service Mesh is the communication layer in a microservice setup. All requests, to and from each of the services go through the mesh.

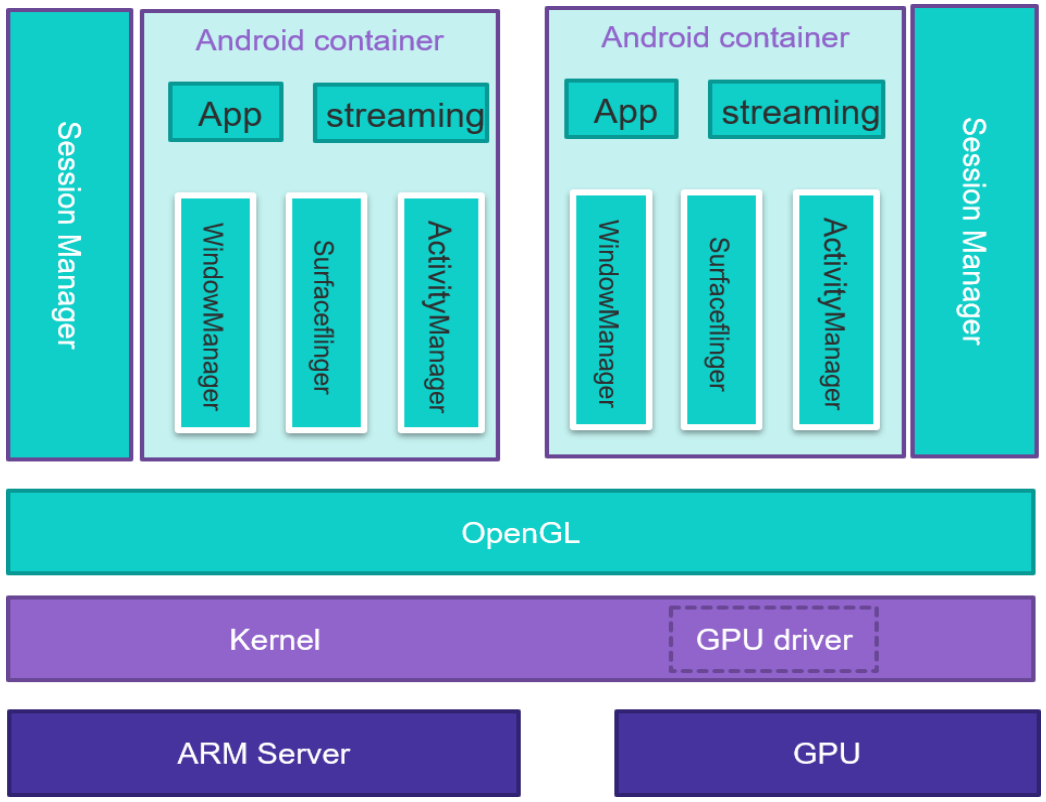
Istio – A Robust Service Mesh – on Arm64



- **Istio** An open platform to connect, manage, and secure microservices
- **Calico** Simple, scalable and secure Container Network Interface
- **Envoy** High-performance edge/middle/service proxy

Use Case in China Mobile

China Mobile and its partners jointly initiated and to be PTL as two Akraino IEC BP projects, which have successfully released the R3 version of the community;

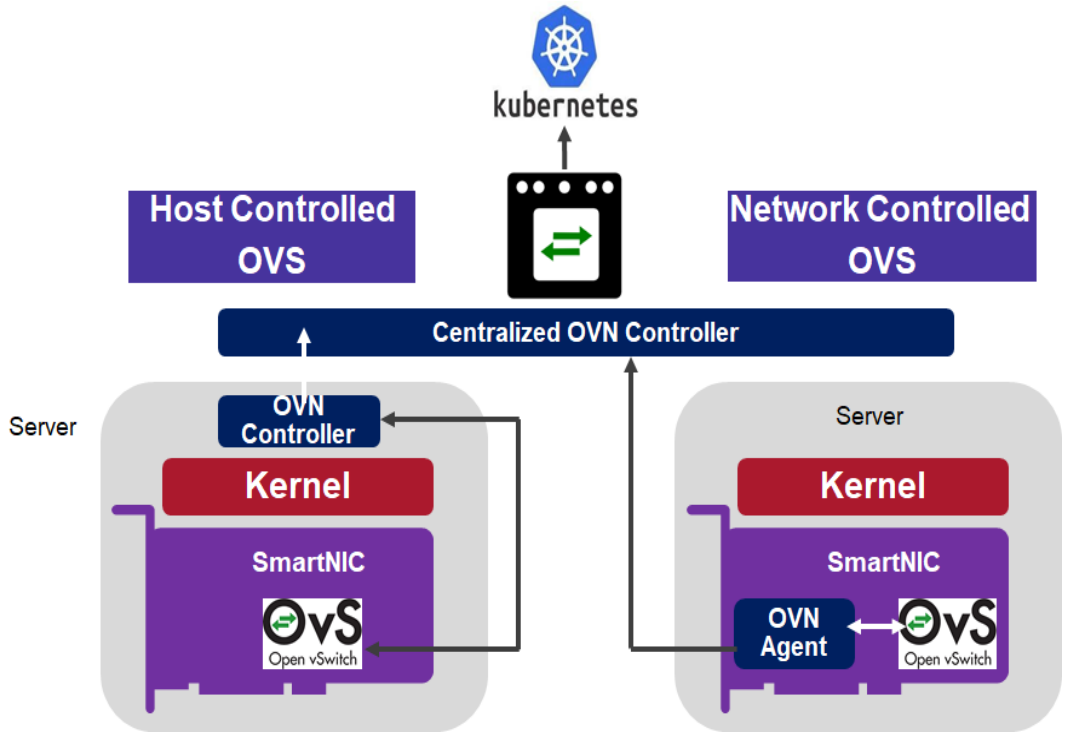


IEC Type 3：
Android cloud native applications on Arm servers in edge

- Mainly by deploying cloud game rendering, encoding and decoding, storage and distribution capabilities at edge nodes, the user interaction delay is reduced, so that users can enjoy the ultimate game experience without buying expensive game terminals；
- The project will provide Android cloud native running environment under arm architecture for mobile terminal applications at the edge, reduce the difficulty of deployment and development of applications on the edge cloud, and reduce the deployment cost of Android edge cloud;

arm

Use Case in China Mobile



IEC Type 5: SmartNIC

- The first release of the integrated edge cloud type 5 blueprint is based on the arm SoC architecture, and OVS-DPDK is unloaded to the smart network card, which can enhance the throughput performance of the edge network VPC, reduce the packet loss rate, and enhance the management of network card resources to save more computing resources;
- In the future, the performance of 5G UPF network elements deployed in the edge cloud data center of operators can also be enhanced by realizing the uninstallation function of network cards;

China Mobile is willing to further promote the maturity, business innovation of the cloud native computing through active exploration and practice in the field of Cloud Native & Edge Computing open source.

To create a win-win cloud native computing ecosystem !!

arm

Future Work(Provisional)

arm

Future Work(Provisional)

- CNIs support enhancements on arm platform(CI/CD, performance, feature comparisons,…)
- Performance optimization of overlay tunnel communication on arm platform
- Further uses cases and deployments about cloud native networking on arm platform
  - AR/VR,
  - 5G
  - MEC
- Service mesh integration with high performance CNIs, such as Cilium/Proxy
- Further DPDK incorporated container networking usage model and performance evaluation
- Further CI for high performance container networking solutions on arm with up-to-date processes

Some References

- Cisco: Cloud-Native Network Functions
- INTRODUCTION TO KUBERNETES* NETWORKING AND ACCELERATION WITH DPDK
- Flannel CNI
- Overview of Calico CNI for Kubernetes
- Kubernetes: Flannel networking
- Network Service Mesh
- Cilium
- Istio and Service Mesh
- Multus CNI
- SRIOV CNI
- SRIOV Network Device Plugin

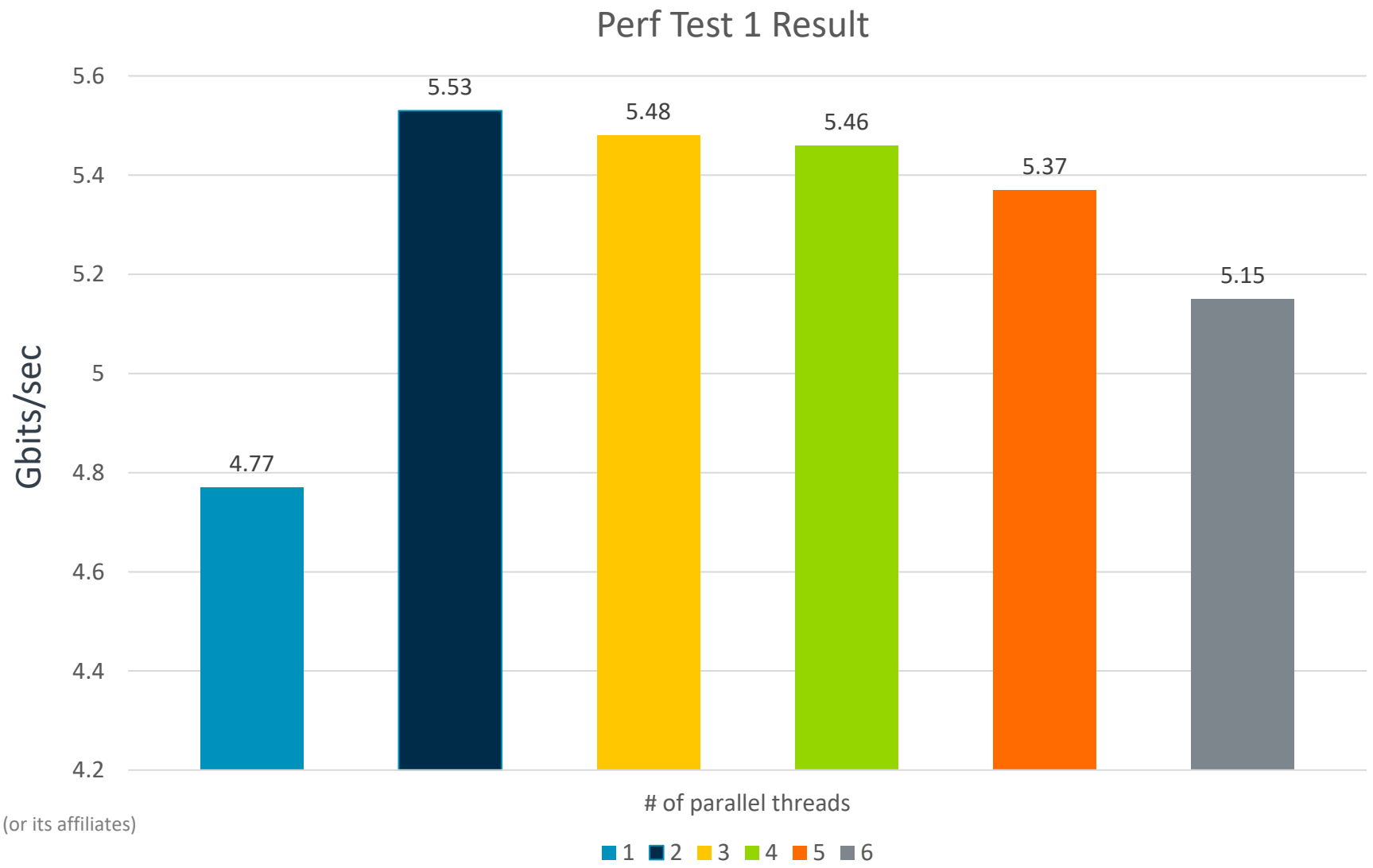
arm

Thank You
Danke
Merci
谢谢
ありがとう
Gracias
Kiitos
감사합니다
धन्यवाद
شكرًا
ধন্যবাদ
תודה

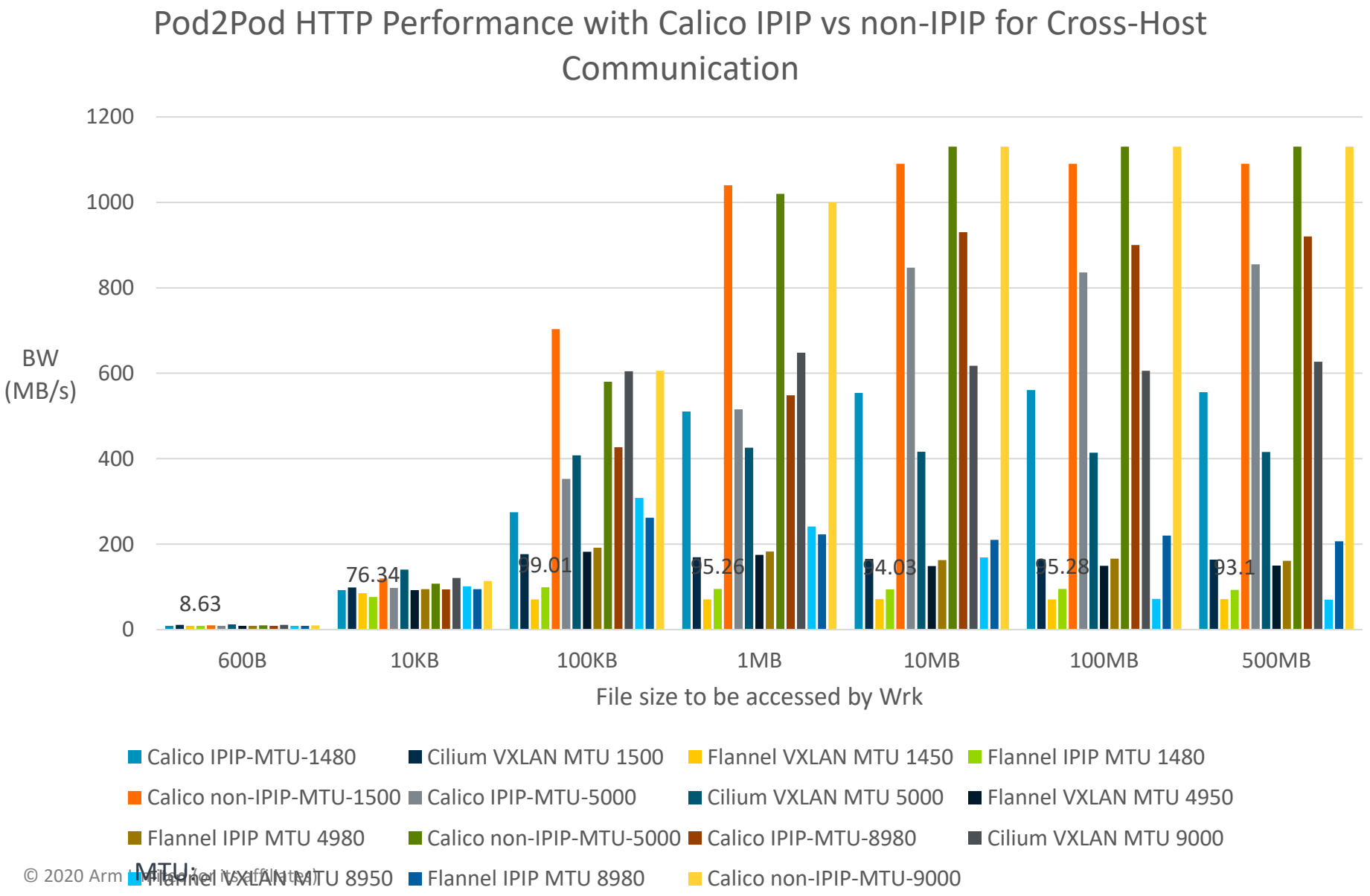
Performance Test Result

Note: Here we choose iperf instead of iperf3 due to its multi-threaded implementation

Pod Communication Performance with SRIOV CNI in the same host via PS225 VFs



HTTP Performance Benchmarking for CNIs with various backends

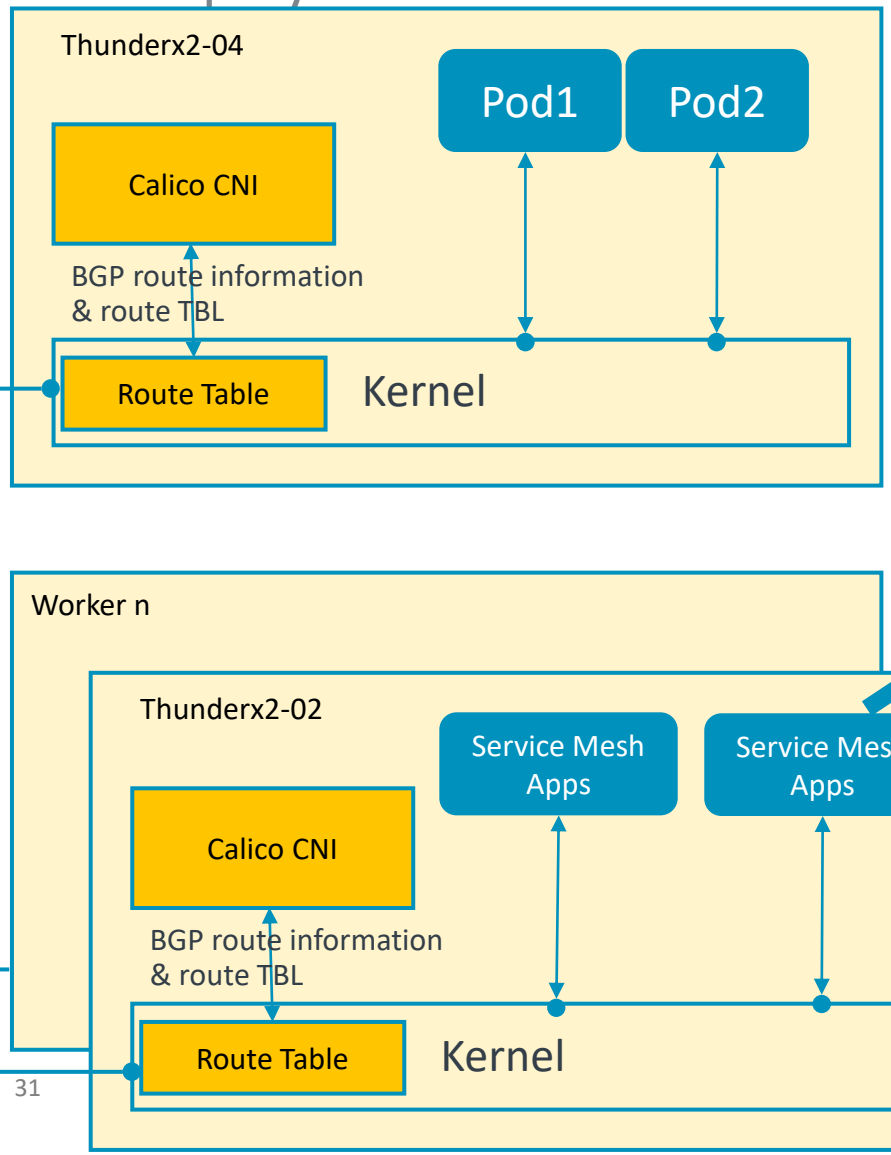


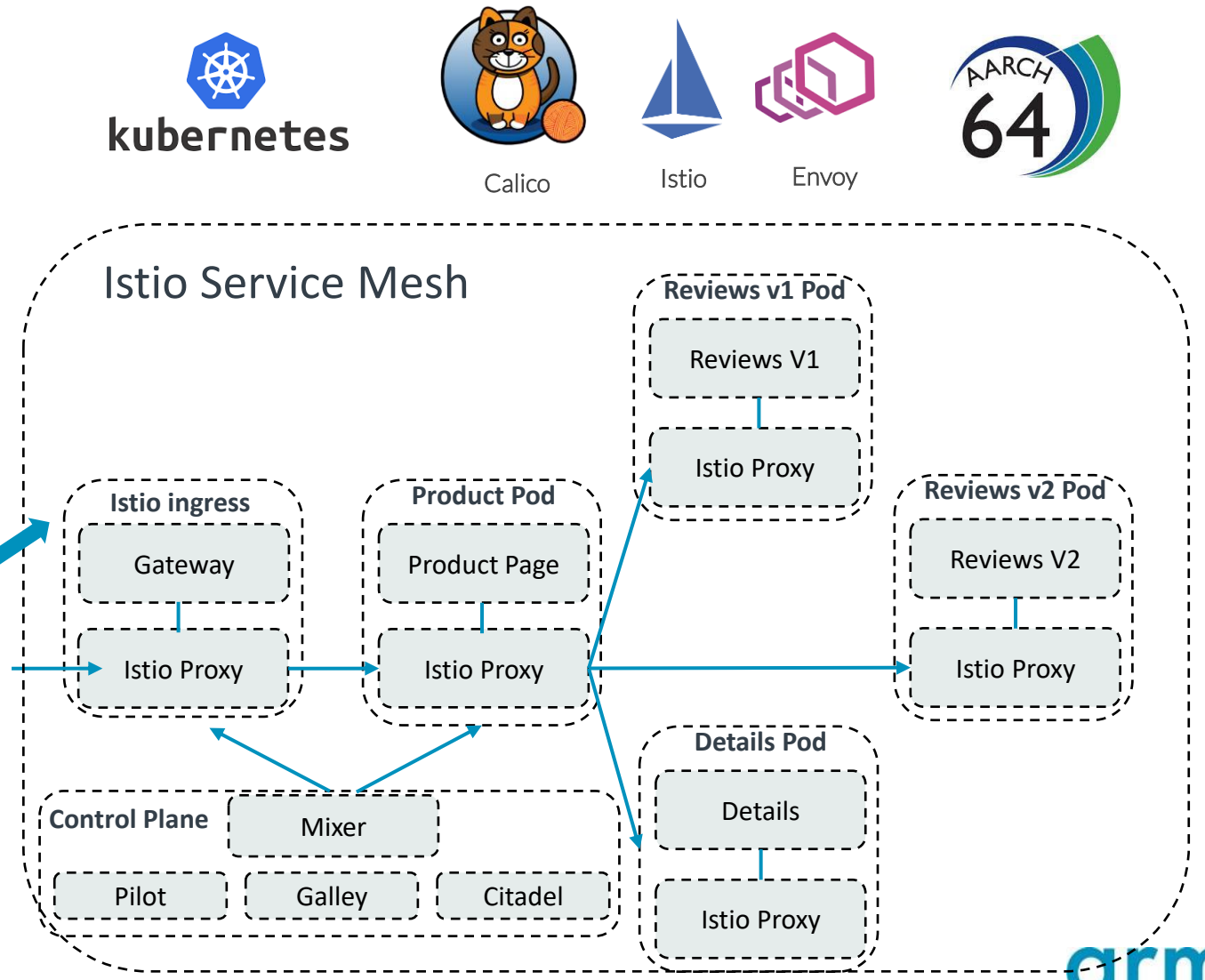
Initial observation:

- For file size > =10MB, the MTU has little effect to the final performance
- The performance is much higher than those of IPIP when file size >= 100KB
- When MTU is small, the performance gap between IPIP and non-IPIP is higher

Istio Simple Case on Arm64

Arm deployment case – Bookinfo system





A booking-information checking system

arm

Contiv/VPP on Arm

TrevorTaoARM commented on Jun 11

Add arm64 support for contiv/vpp docker images building
The naming rules of the built docker images is given below:
For a built image named as ${repo}:${tag}, or
${repo}/${component}:${tag},

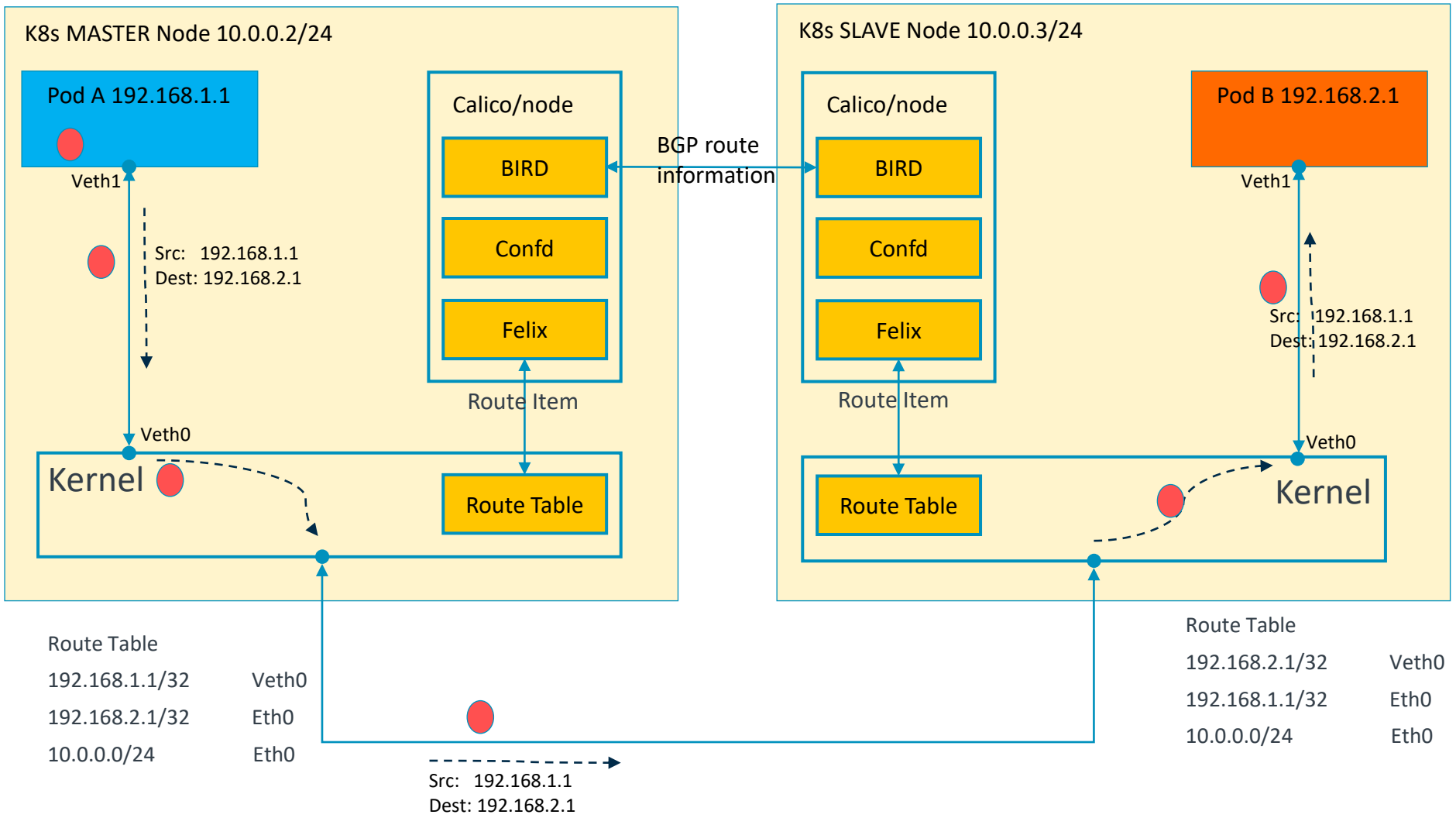
1. For x86_64 arch, it would be named as:
   ${repo}-amd64:${tag}, or ${repo}/${component}-amd64:${tag}
2. For arm64 arch, it would be named as:
   ${repo}-arm64:${tag}, or ${repo}/${component}-arm64:${tag}
3. The docker images built for amd64 are tagged as the default:
   ${repo}-amd64:${tag}--->${repo}:${tag}
   ${repo}/${component}-amd64:${tag}--->${repo}/${component}:${tag}

- What We Have Done:
- Add arm64 support for Contiv/VPP docker images building to the upstream
- Built the needed docker images for Contiv/VPP on arm64
- Create the corresponding Kubernetes setup yaml file for arm64 platform
- Contiv/VPP pods are booted up for Kubernetes on arm64 platform
- Basic container networking with Contiv/VPP on 1 single host(Arm server)

```
root@net-arm-thunderx-01:/home/trevor# kubectl get pod -n kube-system -o wide |grep contiv
contiv-etcd-0                           1/1     Running   0          25s     10.169.40.76   net-arm-thunderx-01
contiv-ksr-5f29x                        1/1     Running   0          23s     10.169.40.76   net-arm-thunderx-01
contiv-vswitch-5c4bw                    0/1     Running   0          24s     10.169.40.64   net-arm-d05-08
contiv-vswitch-qdgkm                    0/1     Running   0          24s     10.169.40.76   net-arm-thunderx-01
root@net-arm-thunderx-01:/home/trevor#
```

- TBD:
  - Tested: To be tested: container networking between 2 hosts(Arm servers)
  - Further networking function verification/enhancement
  - Possible improvement work: functional enhancement with more network interfaces supported by Contiv/VPP

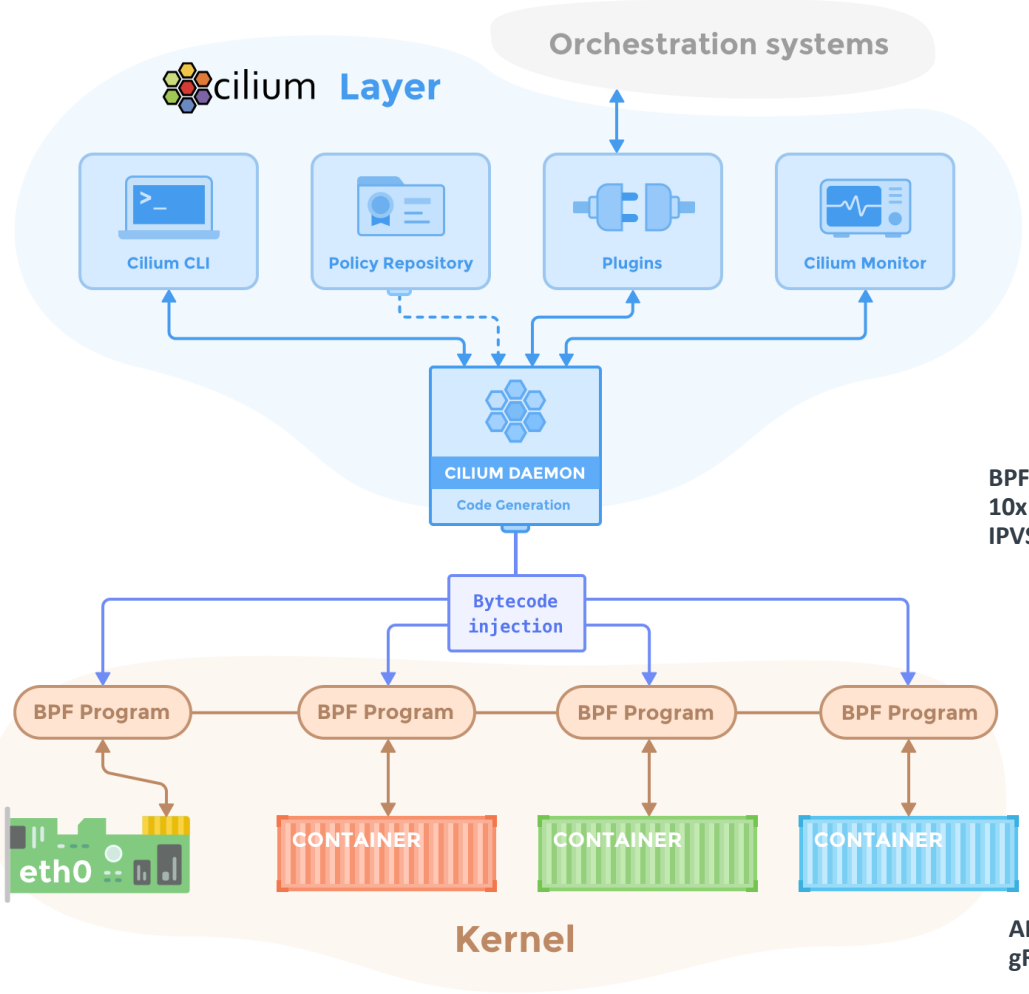
Calico on Kubernetes



1. Pure IP networking fabric
2. No encapsulation needed when simple L2 connection available for nodes
3. Easy deployment and debug
4. Supporting Kubernetes Network Policy by iptables
5. Good scalability with BGP based routing

arm

Cilium – API Aware Networking



[Cilium Architecture]

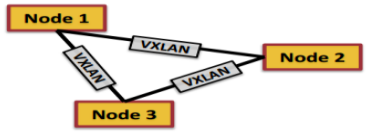
BPF/XDP load balancing
10x performance over IPVS

API Aware(HTTP, Kafka, gRPC)

quick-install.yaml

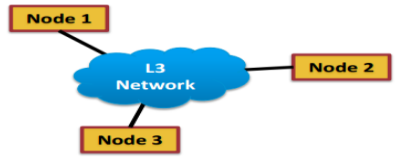
Networking Model: Encapsulation or Direct Routing

Mode I: Encapsulation



No further dependencies

Mode II: Direct Routing

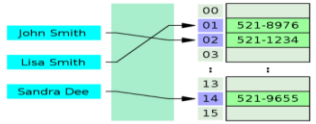


Integrations:
- Cloud routers
- kube-router, BIRD, ...

Kubernetes Services Implementation

BPF-based

- *Per-CPU Hash table*



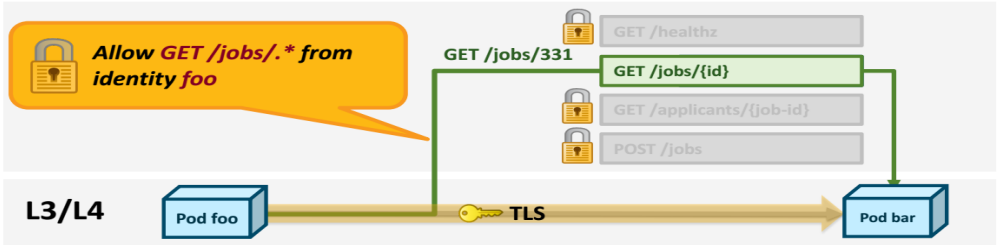
Algorithm	Average	Worst case
Space	O(n)[1]	O(n)
Search	O(1)	O(n)
Insert	O(1)	O(n)
Delete	O(1)	O(n)

iptables kube-proxy

- *Linear List*
- *All rules have to be replaced as a whole*

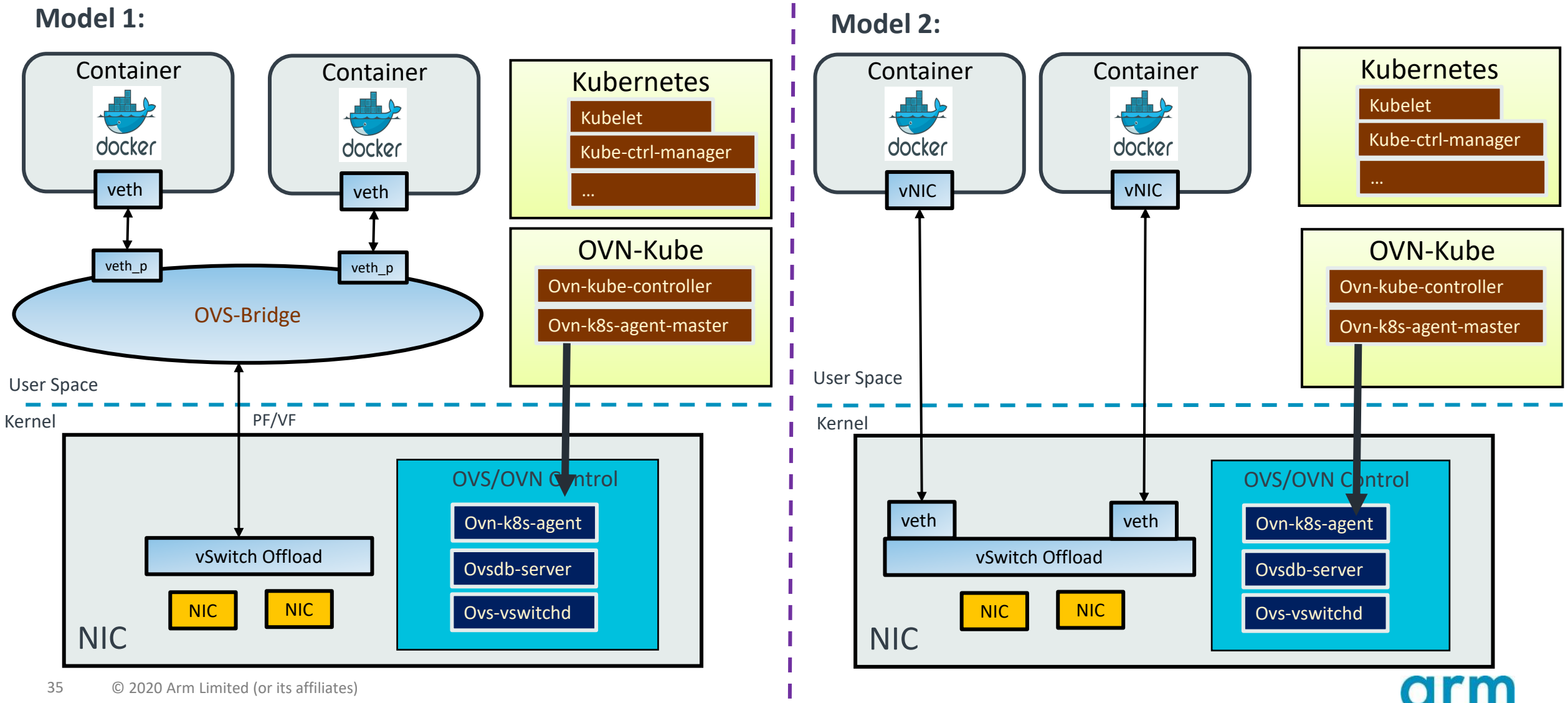
Algorithm	Average	Worst case
Space	O(n)	O(n)
Search	O(n)	O(n)
Insert	O(1)	O(1)
Delete	O(n)	O(n)

API Aware Security



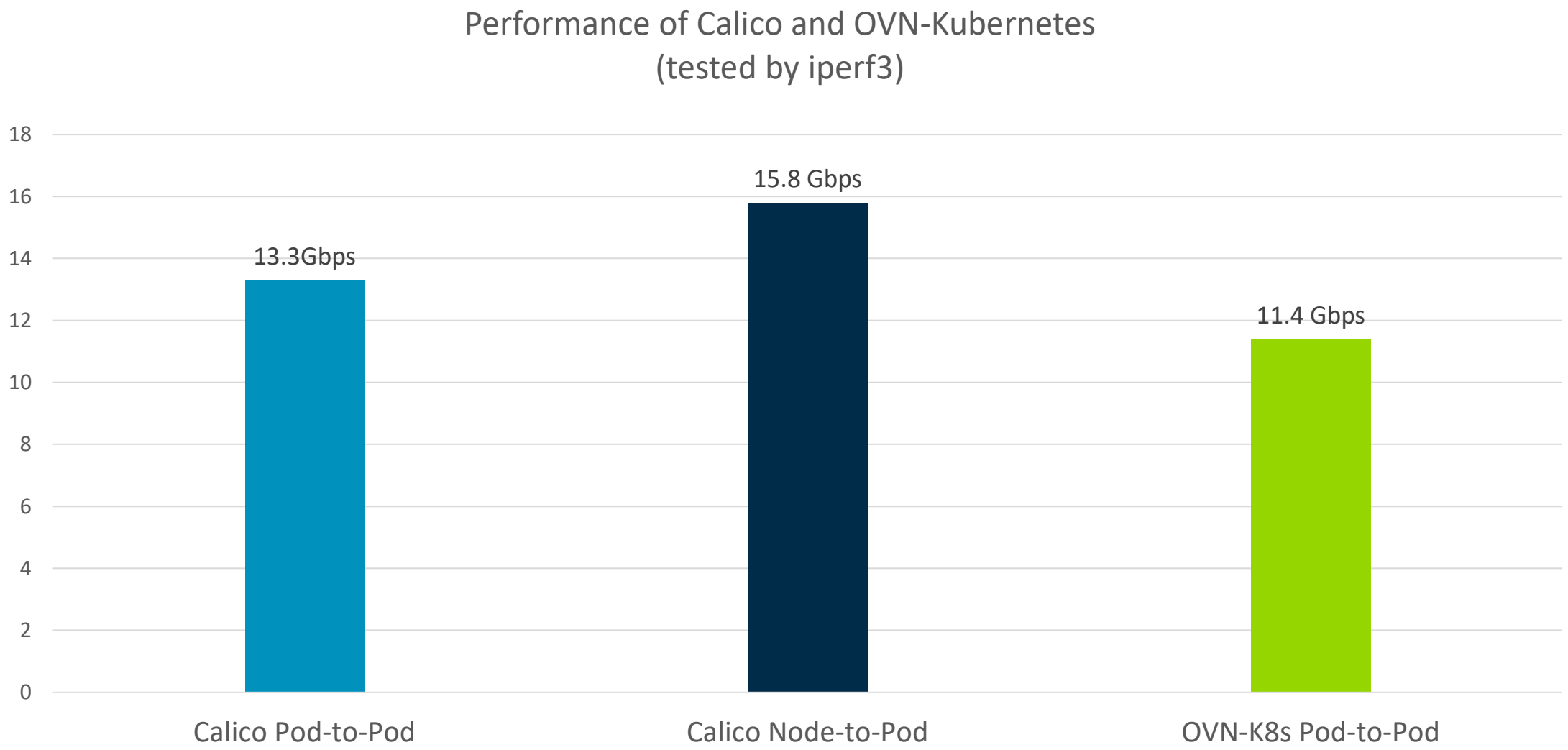
[Cilium: API Aware Netowrking & Network Security for Microservices using BPF & XDP]

A Very Raw Thinking on the Possible Model for Container Networking with SmartNIC OVS offload support



arm

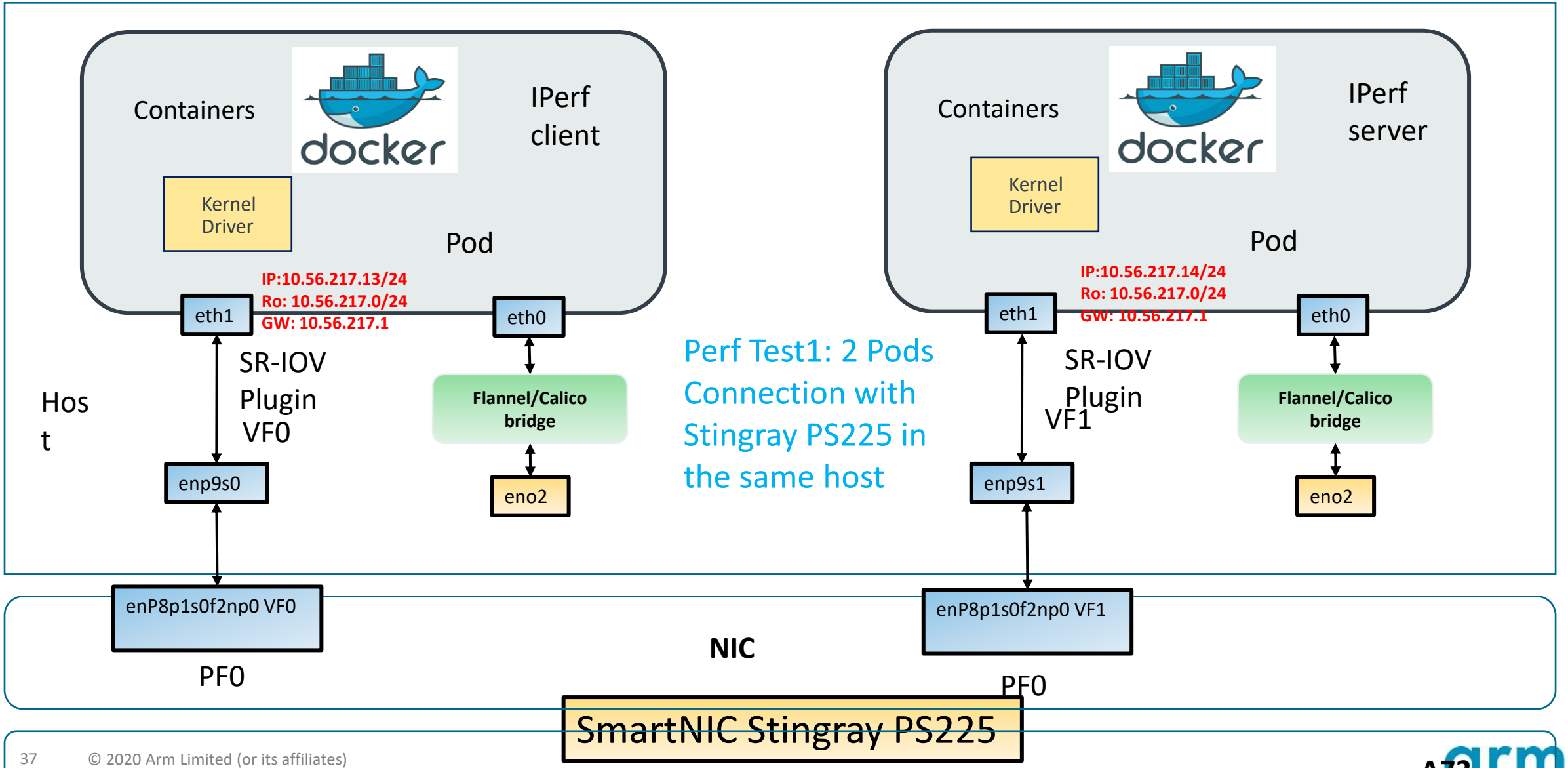
Performance of Calico and OVN-Kubernetes



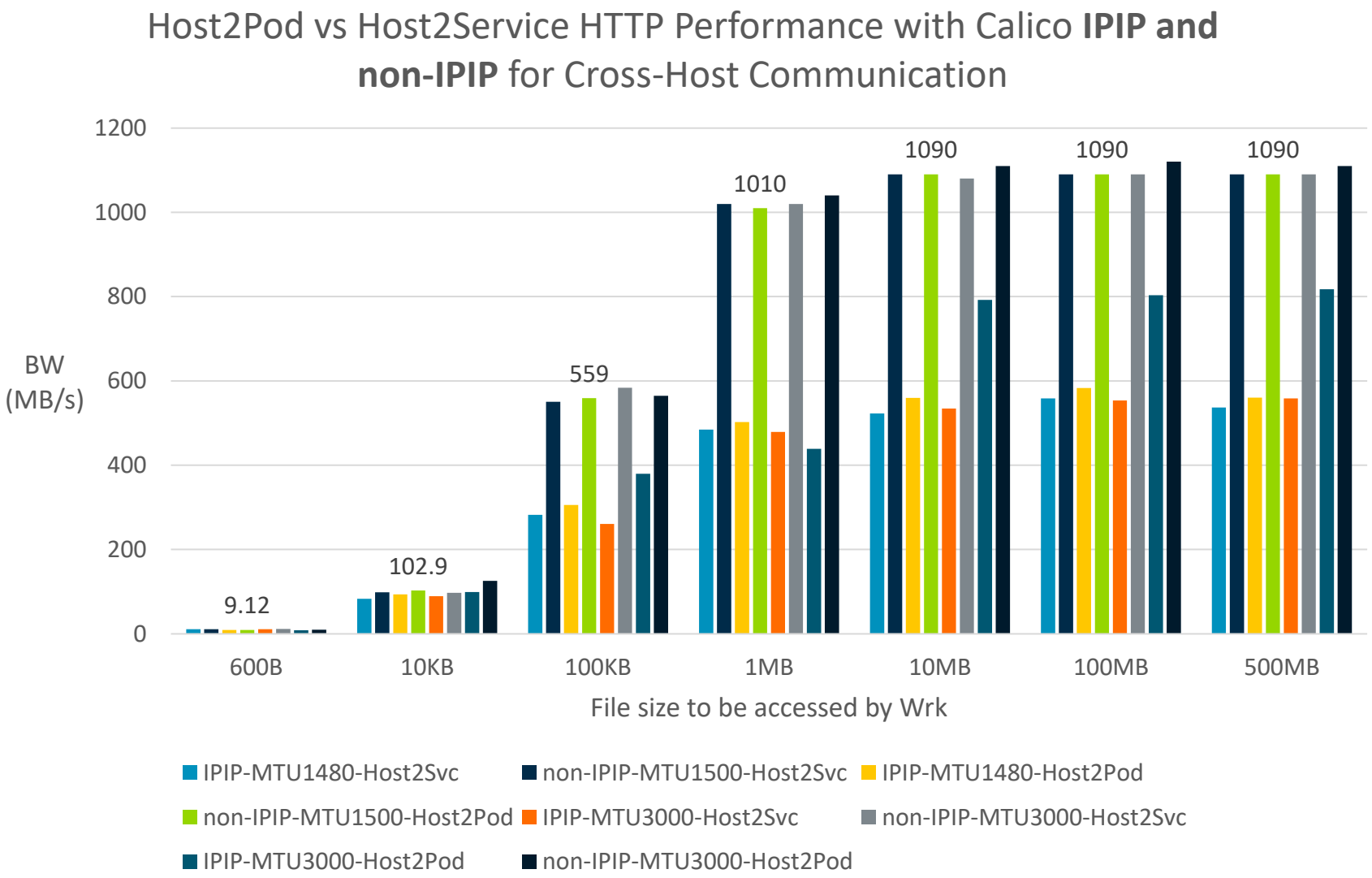
arm

Performance Test with SRIOV CNI

Container Networking by SRIOV Interface for Intra Host Communication



HTTP Performance Benchmarking for Calico CNI

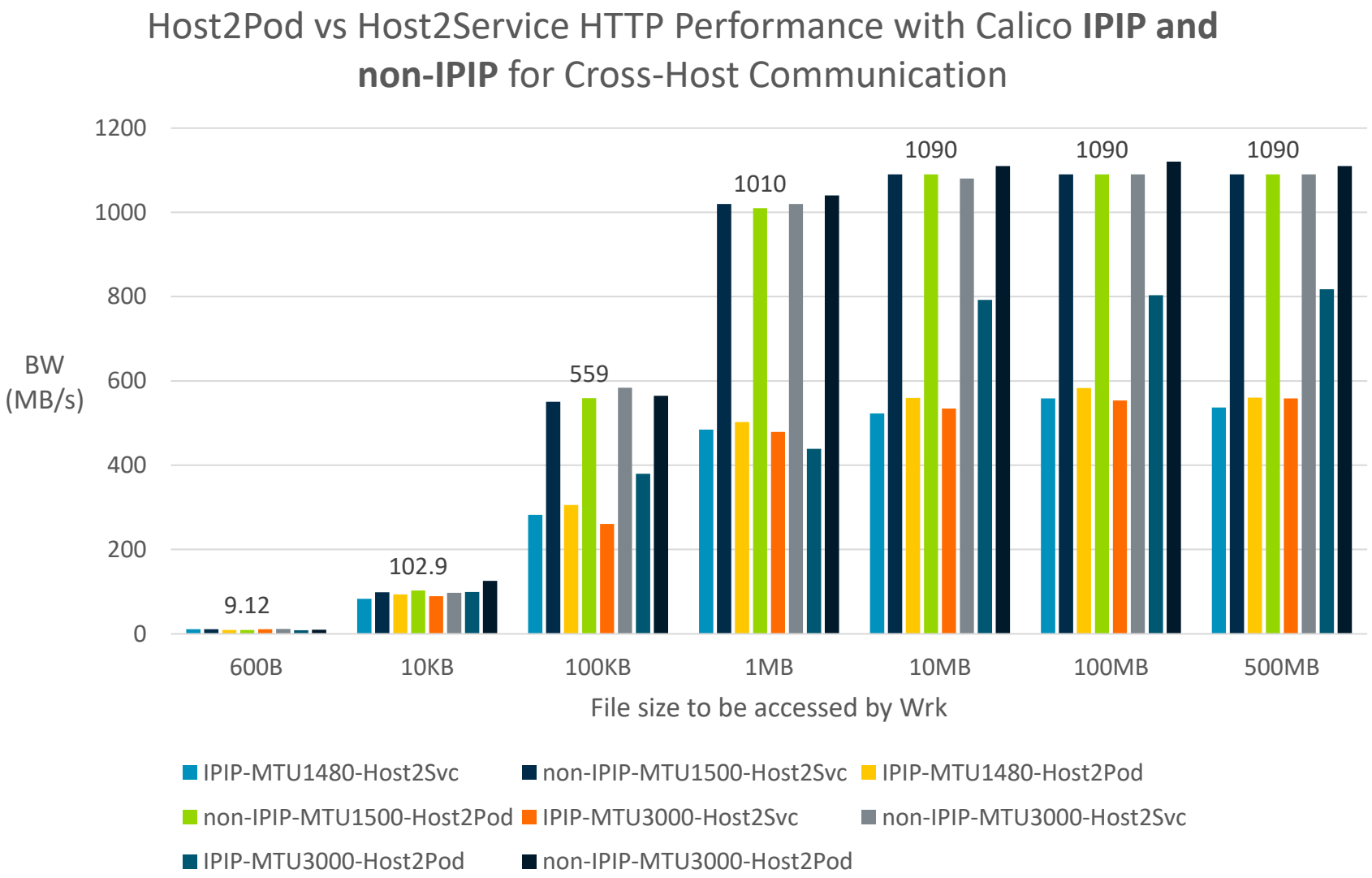


Wrk: thread 5, connections: 10

Observation:

- The performance gap is minor when accessing small files
- For small file size, the host2pod and host2service performance is almost the same, which means the service access(by iptables configured by kube-proxy) is not the bottleneck for HTTP service
- The performance of non-IPIP is much higher than those of IPIP when file size >= 100KB
- For large MTU and large file size, the host2pod performance is better than host2svc.
- For non-IPIP, the performance gap between different MTU is not so explicit, so it's believed the IPIP is actually the bottleneck, which is the same as previous

HTTP Performance Benchmarking for Calico CNI



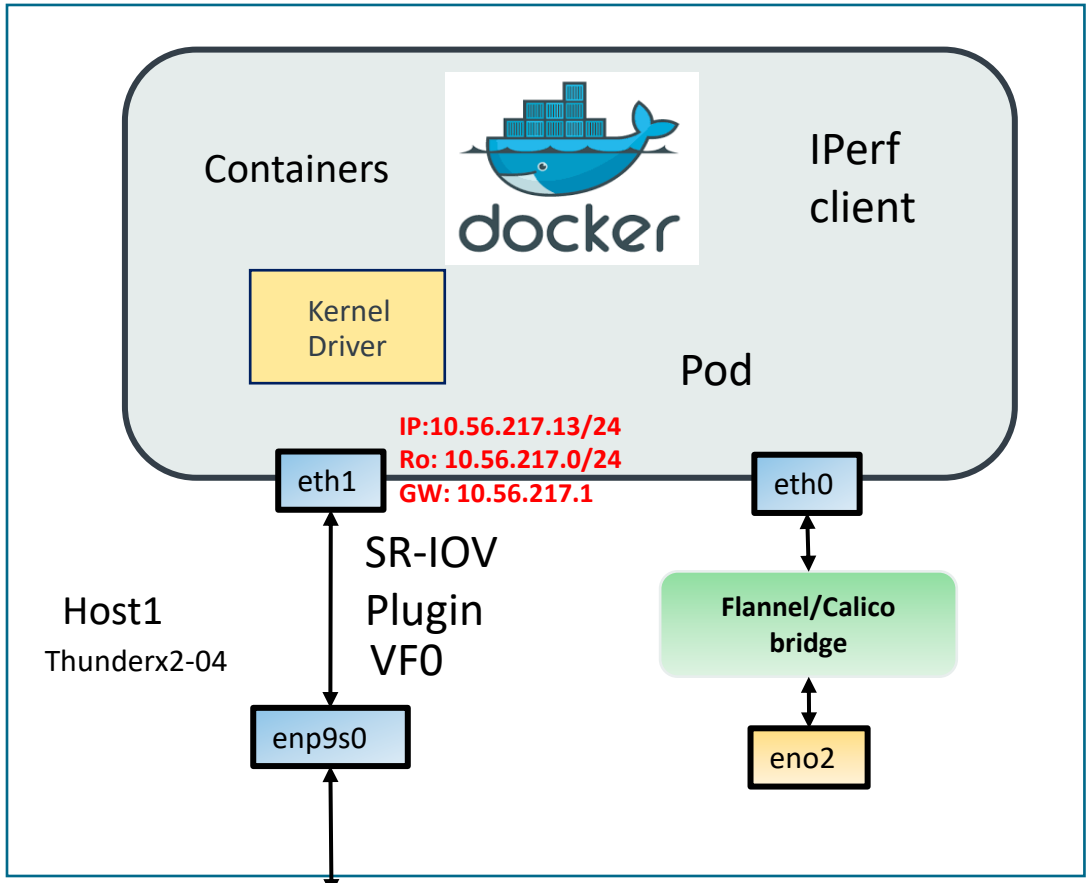
Wrk: thread 5, connections: 10

Observation:

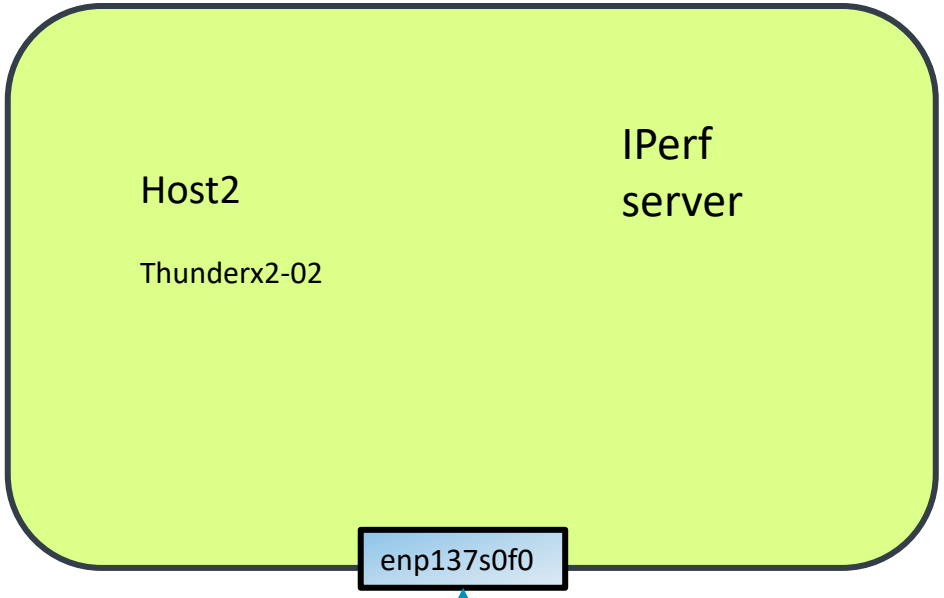
- The performance gap is minor when accessing small files
- For small file size, the host2pod and host2service performance is almost the same, which means the service access(by iptables configured by kube-proxy) is not the bottleneck for HTTP service
- The performance of non-IPIP is much higher than those of IPIP when file size >= 100KB
- For large MTU and large file size, the host2pod performance is better than host2svc.
- For non-IPIP, the performance gap between different MTU is not so explicit, so it's believed the IPIP is actually the bottleneck, which is the same as previous

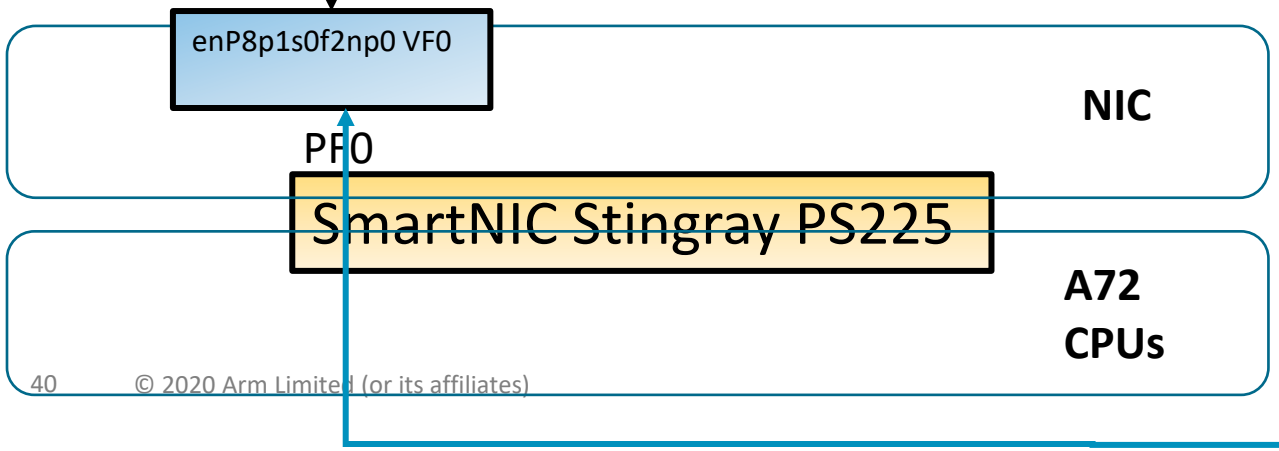
Performance Test with SRIOV CNI

Container Networking by SRIOV Interface for Inter Host Communication



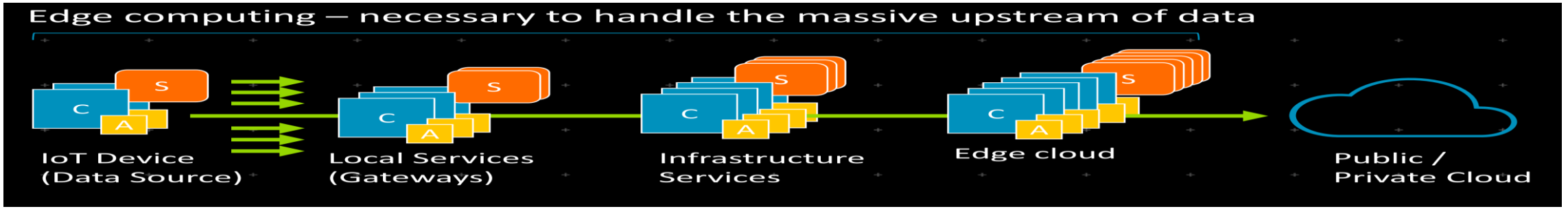
Perf Test2: 1 Pod Connected with another host via SmartNIC VF

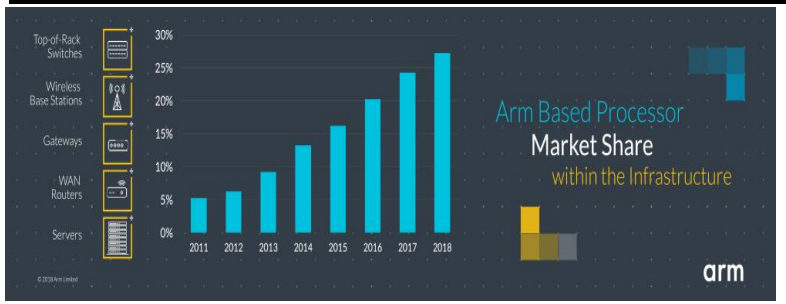


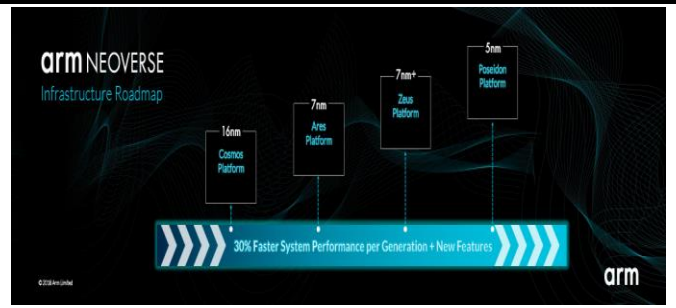


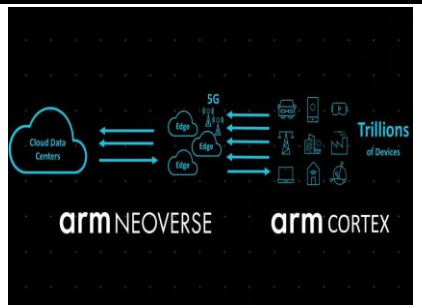
Introduction

- The purpose of edge computing and MEC is to bring real-time, high-bandwidth, low-latency access to latency-dependent applications, distributed at the edge of the network. Arm CPUs are cost effective, low power consumption and customize-allowable which are preferred to be used by edge cloud service vendors
- On the other side, a high performance, flexible and easy deployable container networking of edge software stack is the key to the success of using Arm platform on edge cloud.







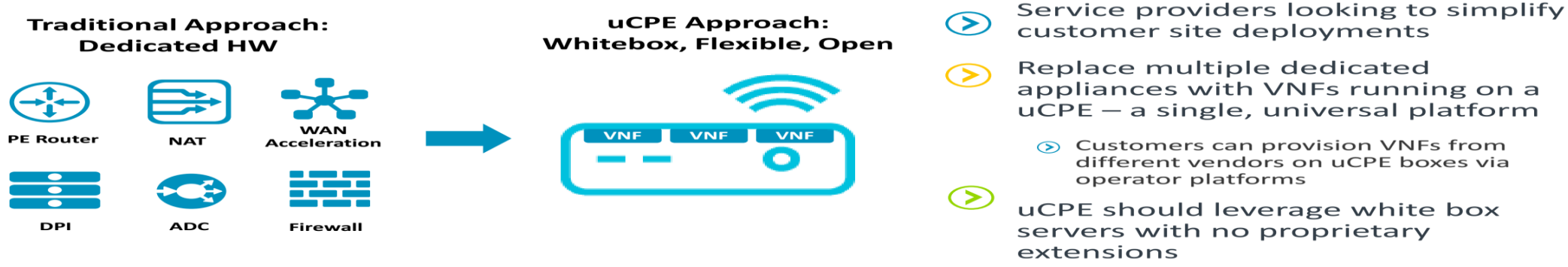


arm

Introduction

- The purpose of edge computing and MEC is to bring real-time, high-bandwidth, low-latency access to latency-dependent applications, distributed at the edge of the network. Arm CPUs are cost effective, low power consumption and customize-allowable which are preferred to be used by edge cloud service vendors
- On the other side, a high performance, flexible and easy deployable container networking of edge software stack is the key to the success of using Arm platform on edge cloud since the widely used VNFs are deployed by container-based orchestration engine, such as Kubernetes, OpenShift or others.

Universal CPE example of Edge compute



Arm ecosystem leverage price, power, acceleration advantages

https://community.arm.com/developer/ip-products/processors/b/processors-ip-blog/posts/arm-and-telco-systems-partner-to-redefine-universal-cpe