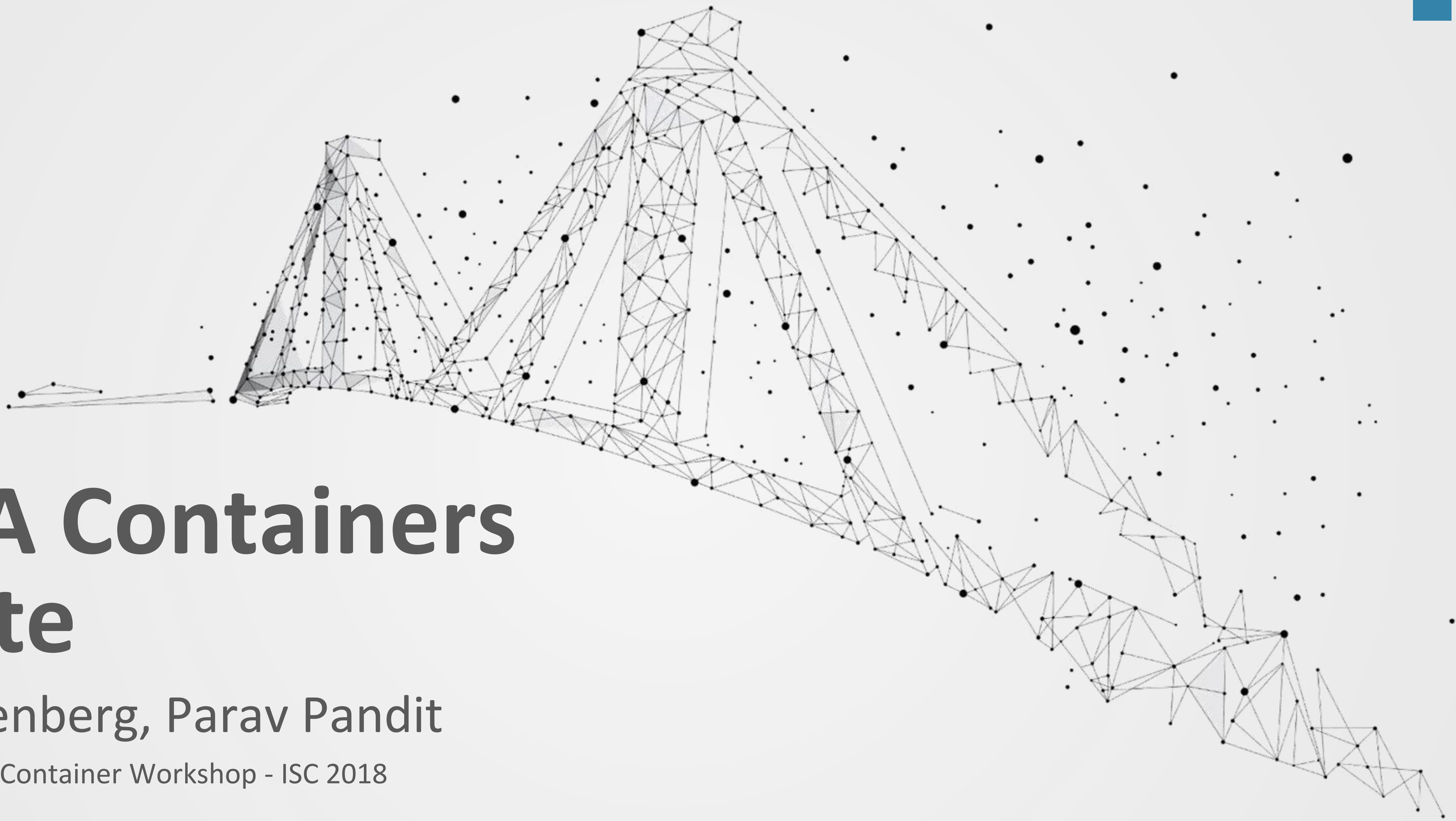


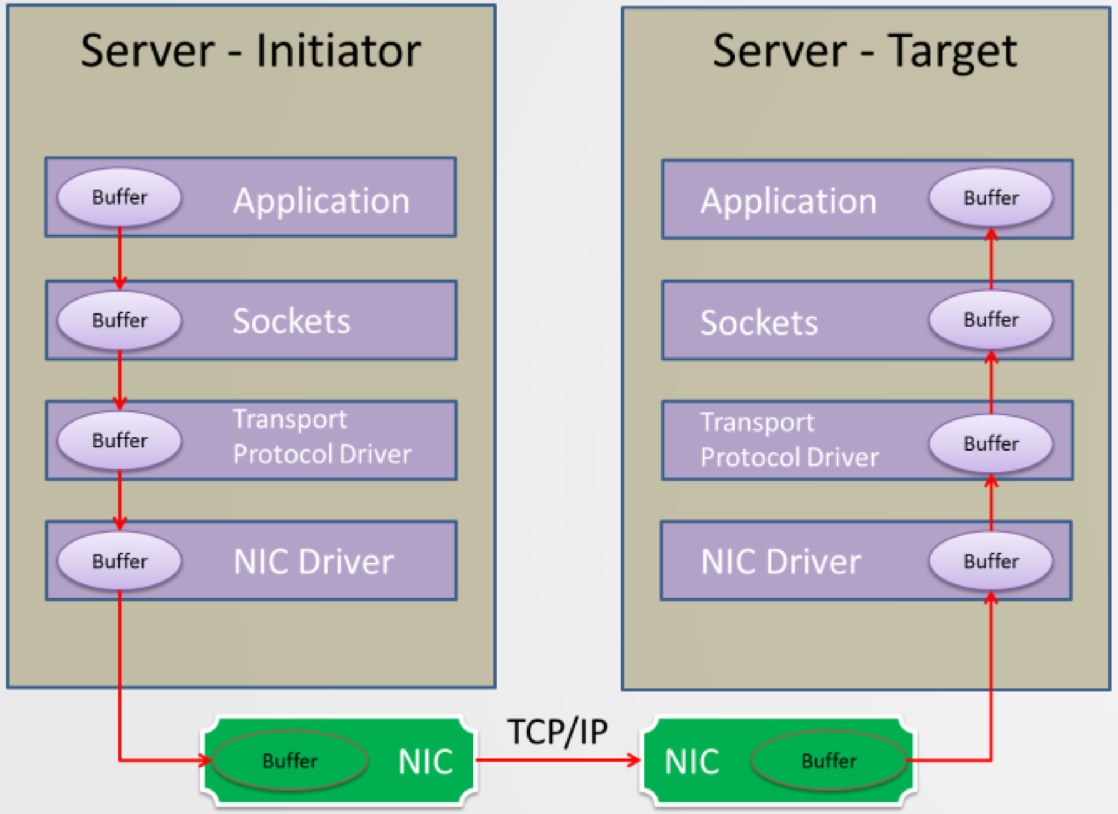
RDMA Containers Update

Dror Goldenberg, Parav Pandit

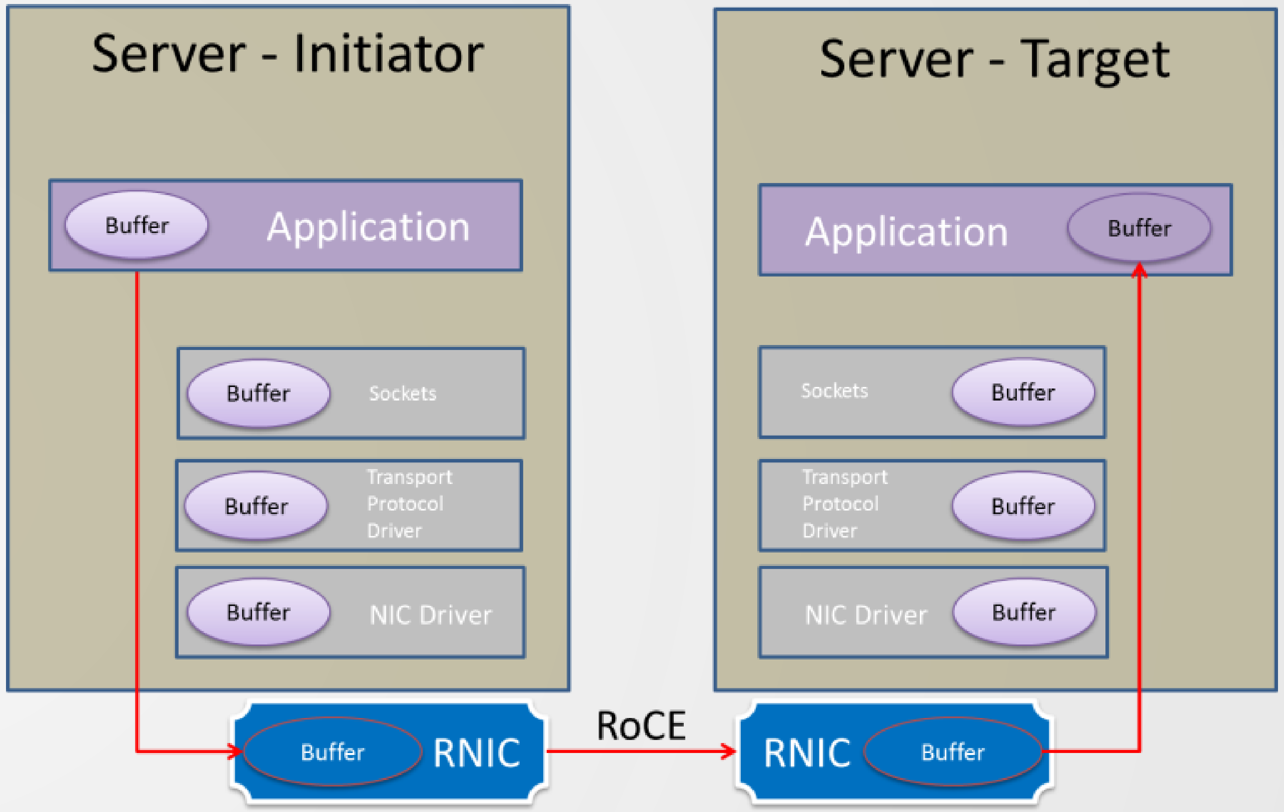
High Performance Container Workshop - ISC 2018

Why RDMA?

TCP/IP



Remote Direct Memory Access (RDMA)



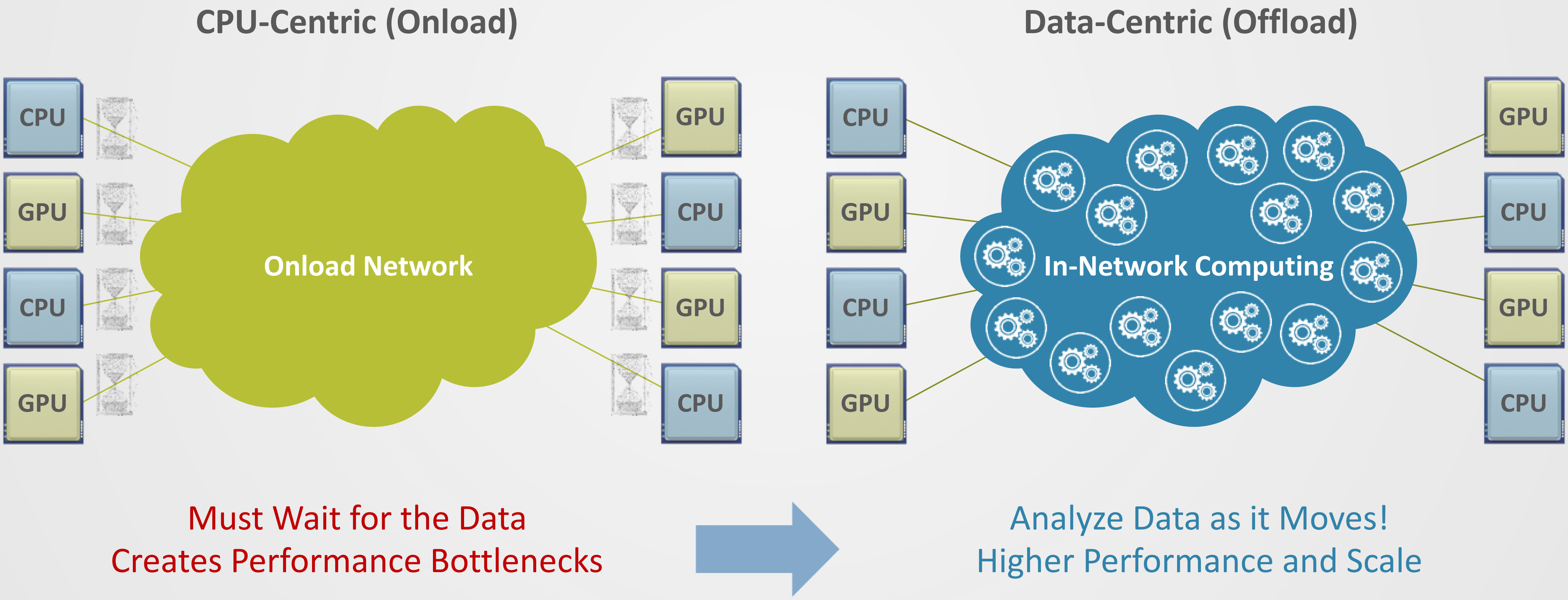
Powers 4 out of 5 top supercomputers
50% better AI performance
10s-100s% better HPC performance

RDMA

2X performance on S2D
10x faster live migration
10x faster messaging
50% faster big data SPARK

The Need for Intelligent and Faster Interconnect

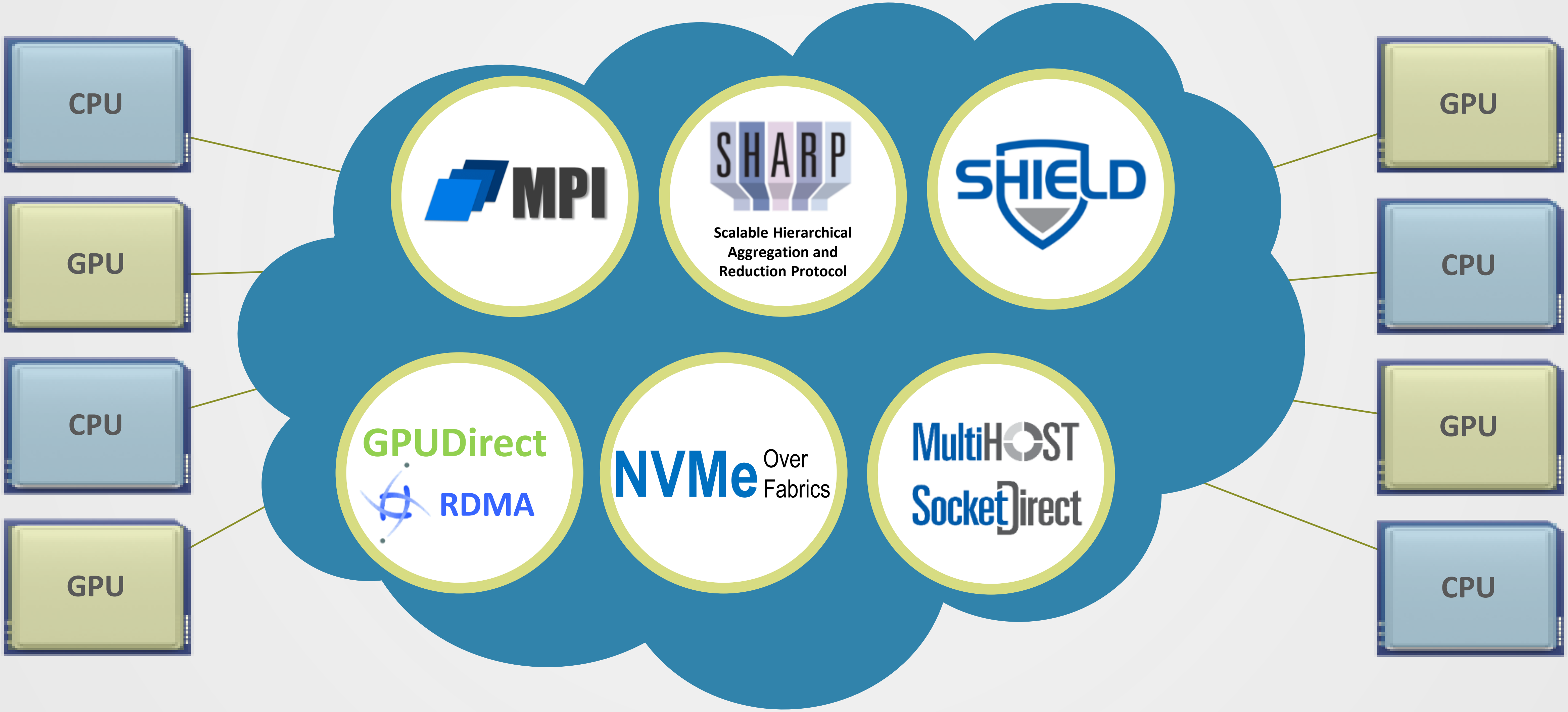
Faster Data Speeds and In-Network Computing
Enable Higher Performance and Scale



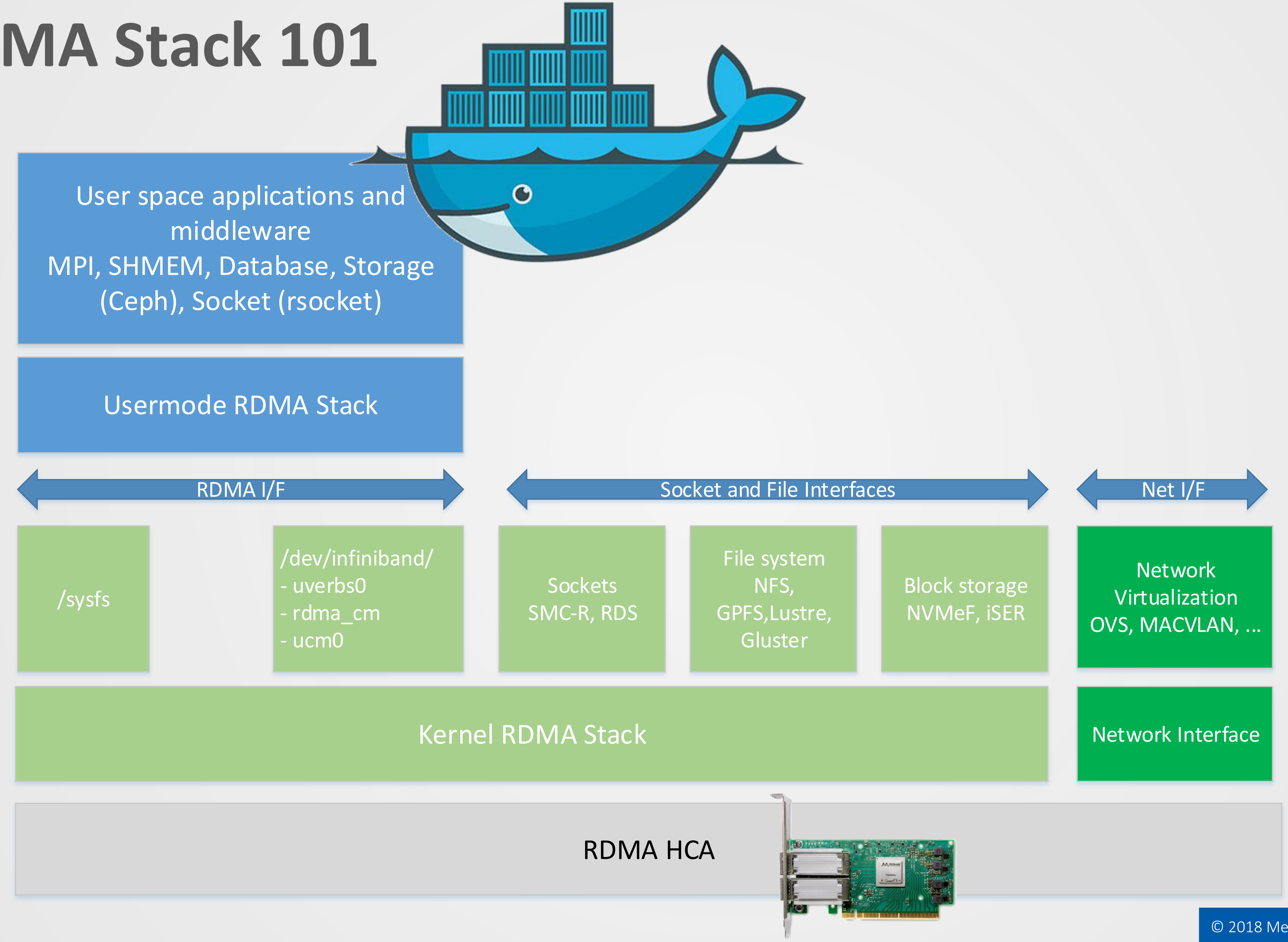
Must Wait for the Data
Creates Performance Bottlenecks

Analyze Data as it Moves!
Higher Performance and Scale

In-Network Computing to Enable Data-Centric Data Centers



RDMA Stack 101



User space applications and middleware
MPI, SHMEM, Database, Storage (Ceph), Socket (rsocket)

Usermode RDMA Stack

RDMA I/F

Socket and File Interfaces

Net I/F

/sysfs

/dev/infiniband/
- uverbs0
- rdma_cm
- ucm0

Sockets
SMC-R, RDS

File system
NFS,
GPFS,Lustre,
Gluster

Block storage
NVMeF, iSER

Network Virtualization
OVS, MACVLAN, …

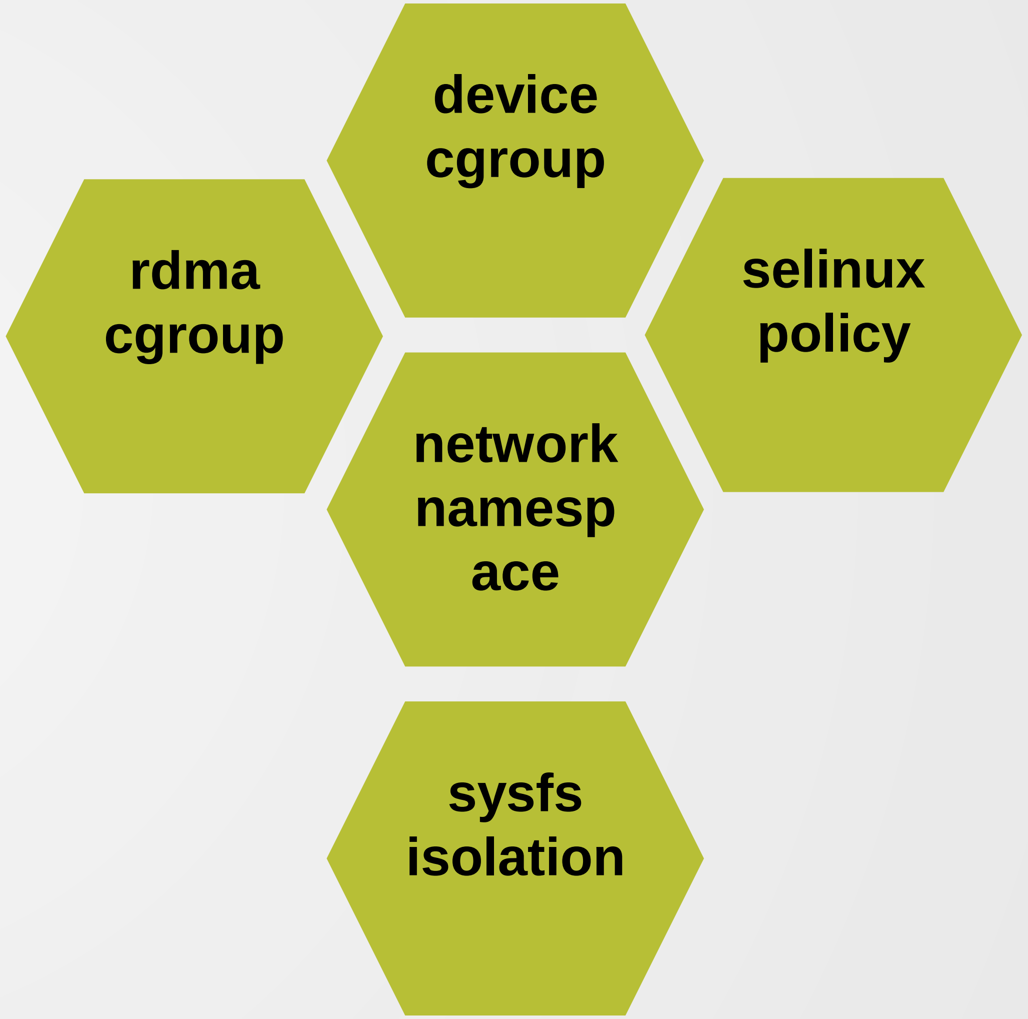
Kernel RDMA Stack

Network Interface

RDMA HCA

Kernel Components for Container Enablement

- Network namespace support
- RDMA cgroup configuration
- Device cgroup configuration
- /sysfs isolation
- SELinux policy enforcement



device cgroup

rdma cgroup

selinux policy

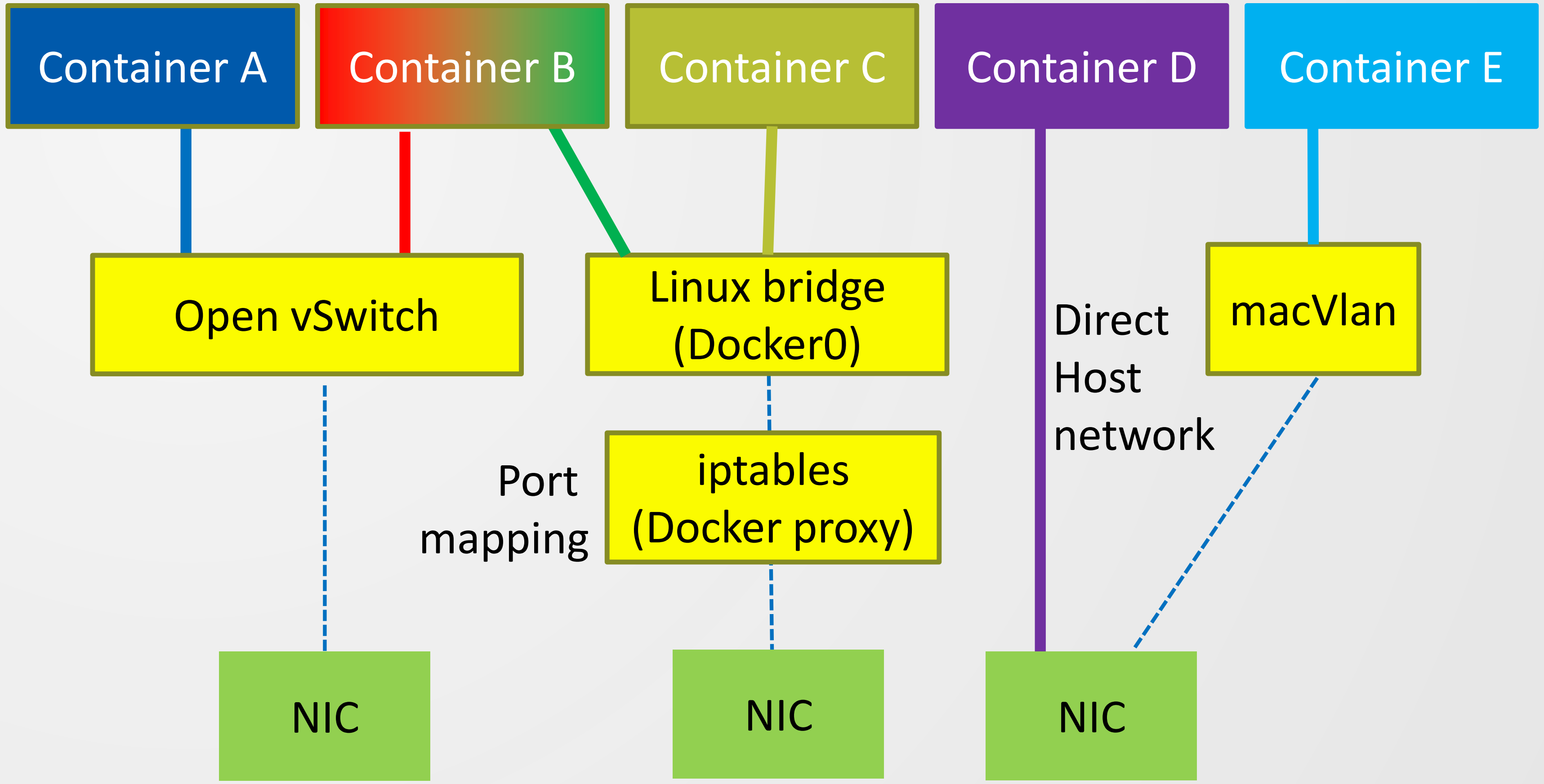
network namesp ace

sysfs isolation

RDMA Network Namespace

- RDMA connections and QPs operate only in caller namespaces
- RDMA underlying network inherit from namespace: IP, MAC, tunnel, InfiniBand L2…

- InfiniBand (kernel 4.9+)
- RoCE (kernel 4.19+)
- Upcoming MOFED 4.4 release



RDMA Control Group

- cgroup - Restricts RDMA resource usage per container
  - hca handles
  - hca objects (AH, CQ, QP, PD, MR, MW, Flows, SRQs…)

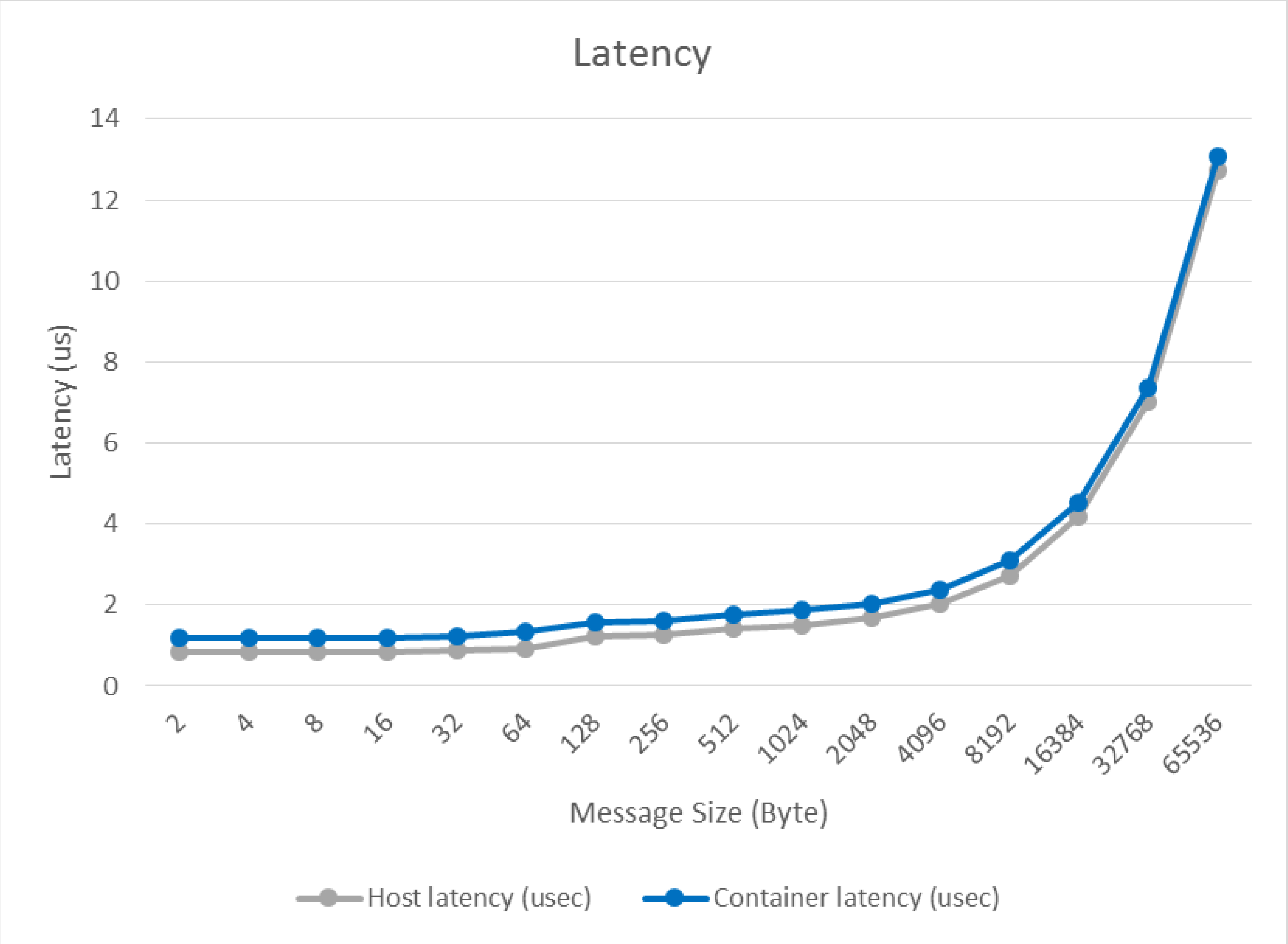
- Kernel
  - Part of kernel from Linux kernel 4.11
- User space
  - Runc spec: https://github.com/opencontainers/runtime-spec
  - containerd/cgroups library https://github.com/containerd/cgroups

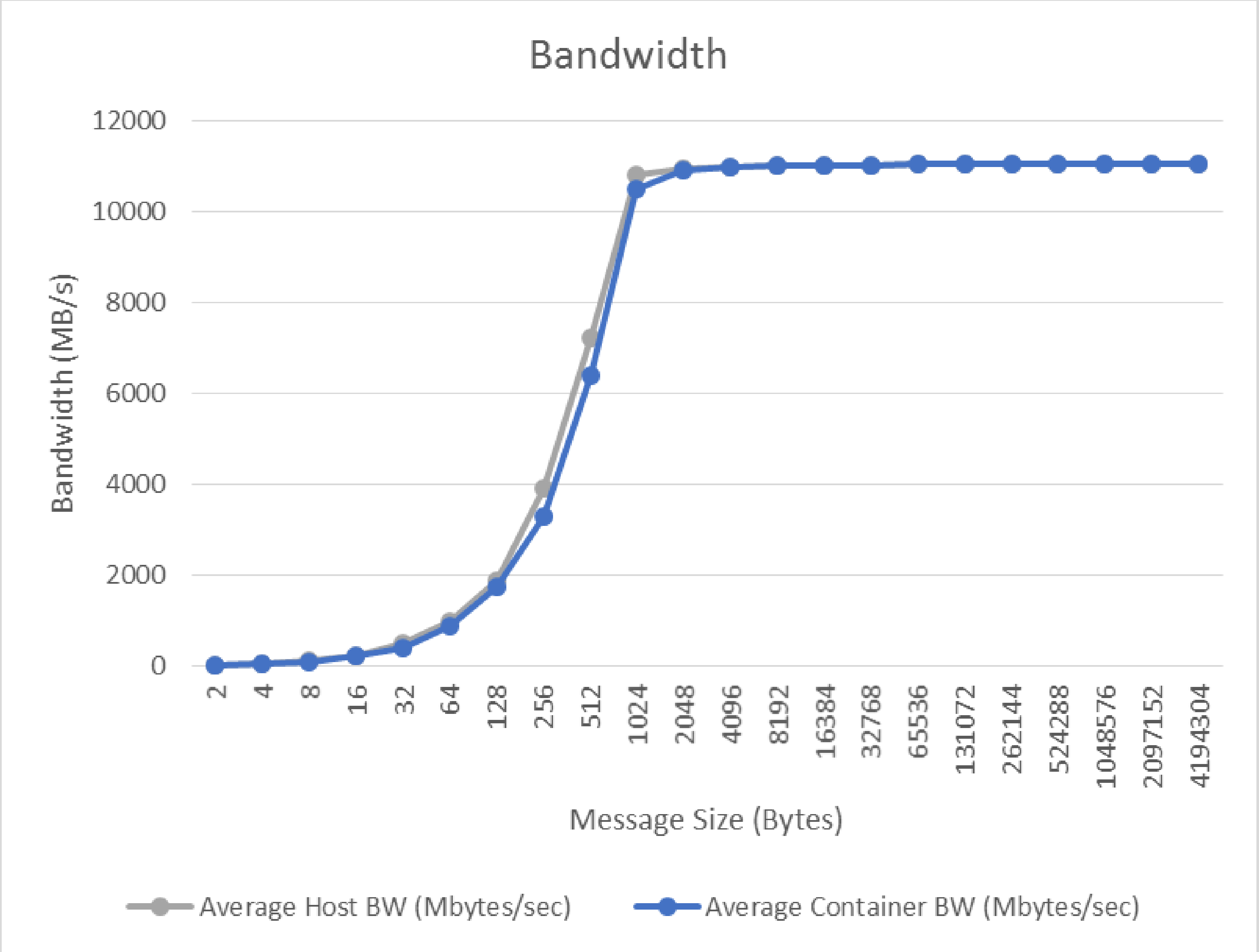
SELinux Policy for InfiniBand

- P_Key policy enforcements per subnet prefix, partition and user context
  - Label individual P_Keys
  - Admin defined policy to allow certain users or application access to labels

- Kernel
  - Initial version starting from Linux kernel 4.13
  - Stable version is 4.15 (IB core)

Container Performance on InfiniBand (SRIOV)





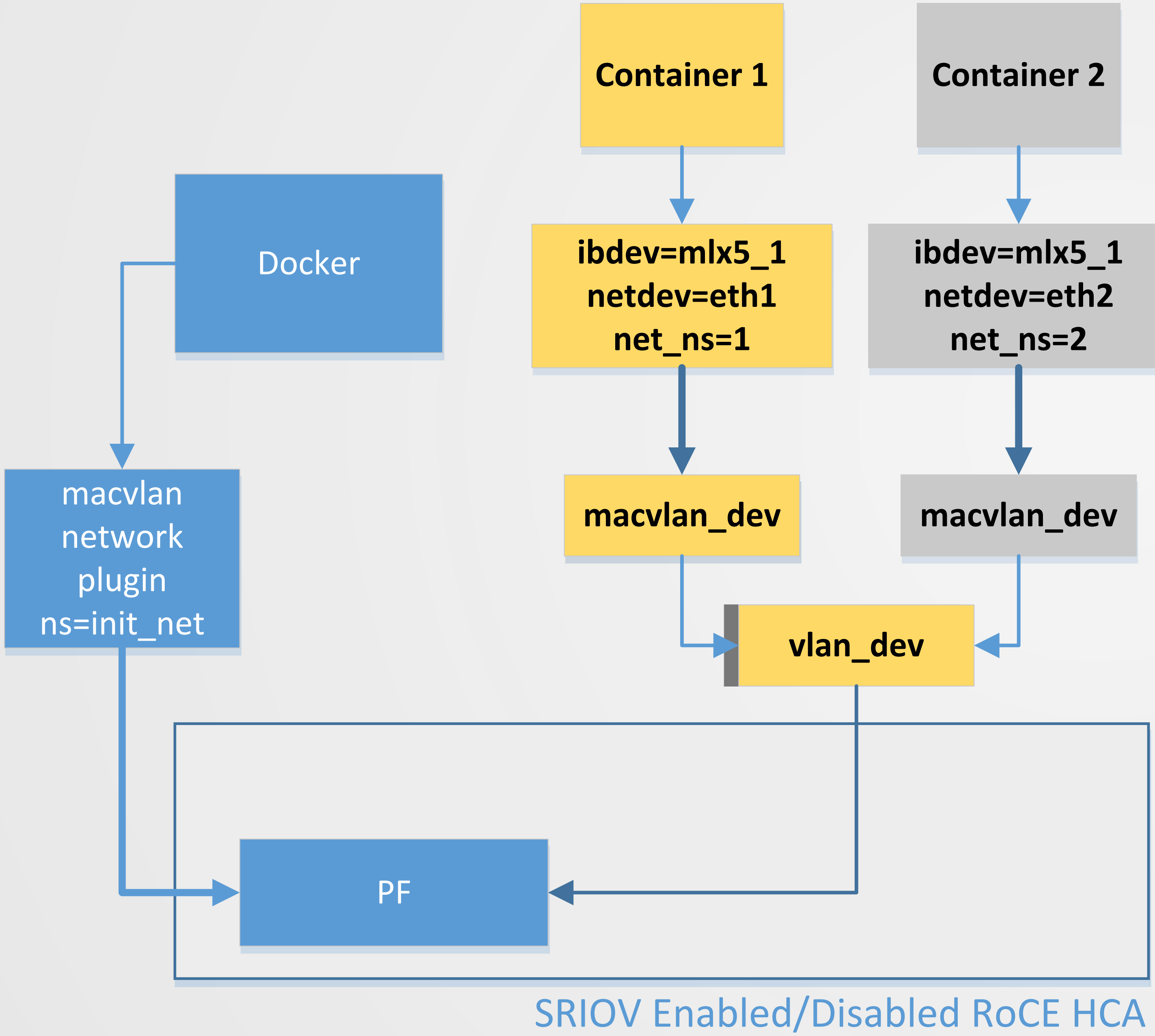
Container performance nearing native performance

Deployment Scenarios

	RoCE MAC VLAN	IB Overlay (Shared Dev)	IB/RoCE SRIOV Legacy	RoCE SRIOV OVS
Native Docker	Supported	Not supported	Supported	WIP
Docker Swarm	Not supported	Supported	Not supported	TBD
Kubernetes	Supported (not tested)	Supported	Supported	TBD

Native Docker: RoCE using macvlan

	RoCE MAC VLAN	IB Overlay (Shared Dev)	IB/RoCE SRIOV Legacy	RoCE SRIOV OVS
Native Docker	Supported	Not supported	Supported	WIP
Docker Swarm	Not supported	Supported	Not supported	TBD
Kubernetes	Supported (not tested)	Supported	Supported	TBD



Native Docker: RoCE using macvlan

- Native macvlan Docker driver
- One macvlan netdvice per container
- Single RDMA device shared among multiple containers
- RDMACM and non RDMACM applications are supported for connection management
- Easy to manage, orchestrate and use

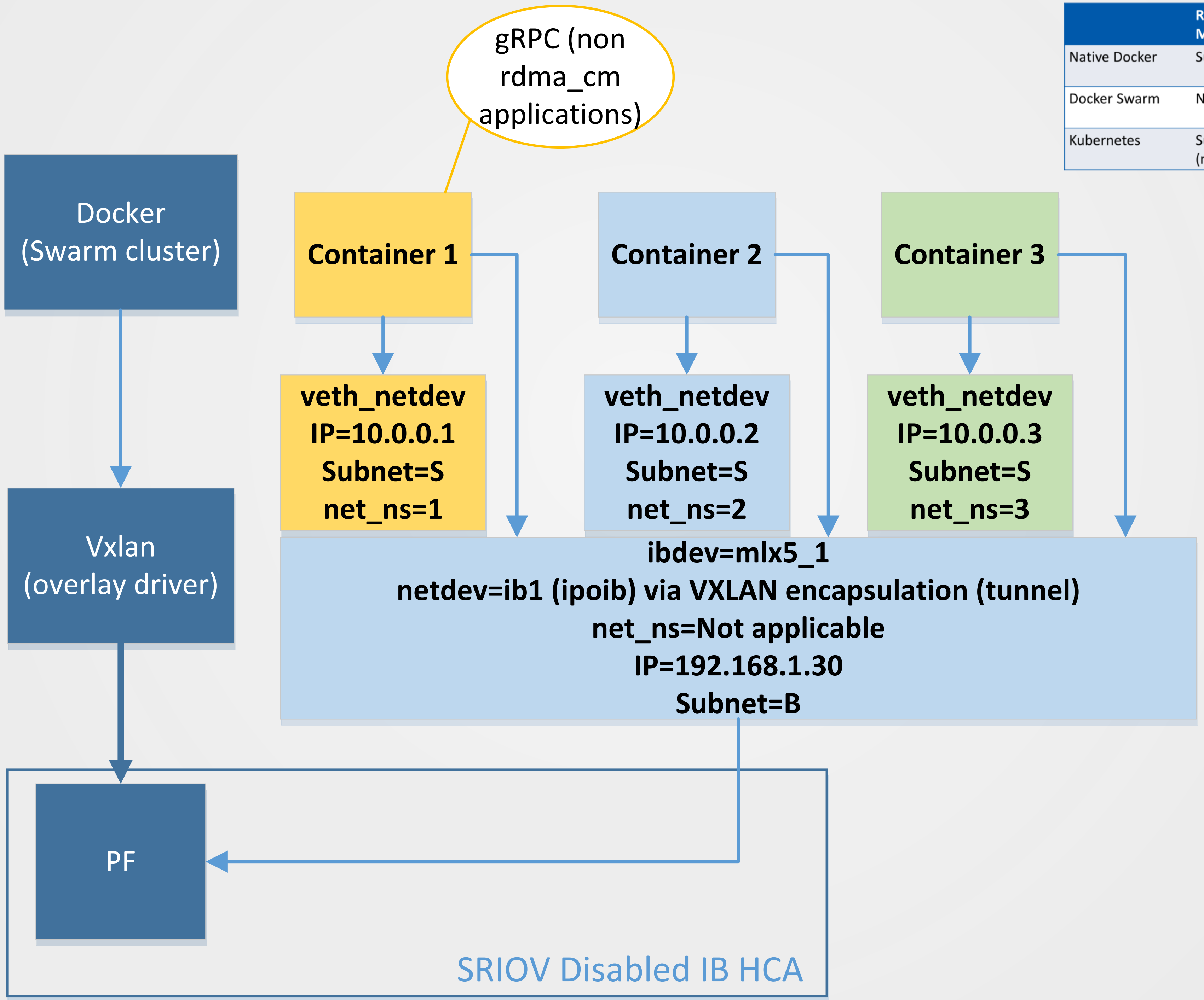
- Also useful with two level virtualization - running Docker containers inside a VM with VF mapped to VM
- With combination of VFs and macvlan, large number of containers can be deployed

- Example
  - Create network
    - *docker network create -d macvlan --subnet=192.168.1.0/24 -o parent=ens2f0 network1*

  - Start containers
    - *docker run --cap-add=IPC_LOCK -it --net=network1 --device=/dev/infiniband --device=/dev/infiniband/rdma_cm mellanox/centos_7_4_mofed_4_2_1_2_0_0_60:latest bash*

Docker Swarm: IB using Overlay Networking

	RoCE MAC VLAN	IB Overlay (Shared Dev)	IB/RoCE SRIOV Legacy	RoCE SRIOV OVS
Native Docker	Supported	Not supported	Supported	WIP
Docker Swarm	Not supported	Supported	Not supported	TBD
Kubernetes	Supported (not tested)	Supported	Supported	TBD



gRPC (non rdma_cm applications)

Docker (Swarm cluster)

Vxlan (overlay driver)

Container 1

Container 2

Container 3

veth_netdev
IP=10.0.0.1
Subnet=S
net_ns=1

veth_netdev
IP=10.0.0.2
Subnet=S
net_ns=2

veth_netdev
IP=10.0.0.3
Subnet=S
net_ns=3

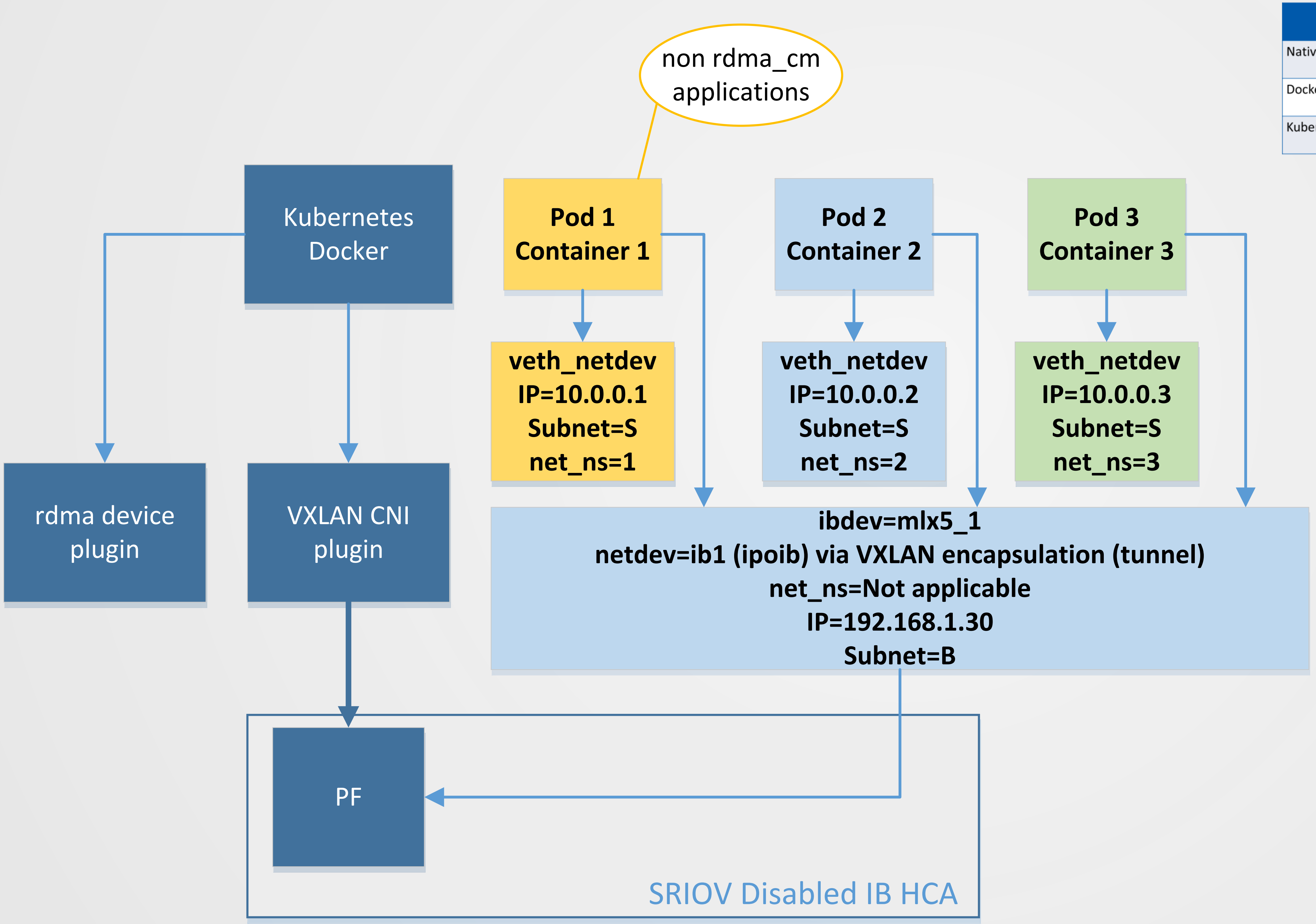
ibdev=mlx5_1
netdev=ib1 (ipoib) via VXLAN encapsulation (tunnel)
net_ns=Not applicable
IP=192.168.1.30
Subnet=B

PF

SRIOV Disabled IB HCA

Kubernetes Docker: IB using Overlay Networking

	RoCE MAC VLAN	IB Overlay (Shared Dev)	IB/RoCE SRIOV Legacy	RoCE SRIOV OVS
Native Docker	Supported	Not supported	Supported	WIP
Docker Swarm	Not supported	Supported	Not supported	TBD
Kubernetes	Supported (not tested)	Supported	Supported	TBD



non rdma_cm applications

Kubernetes Docker

Pod 1 Container 1

Pod 2 Container 2

Pod 3 Container 3

veth_netdev
IP=10.0.0.1
Subnet=S
net_ns=1

veth_netdev
IP=10.0.0.2
Subnet=S
net_ns=2

veth_netdev
IP=10.0.0.3
Subnet=S
net_ns=3

rdma device plugin

VXLAN CNI plugin

ibdev=mlx5_1
netdev=ib1 (ipoib) via VXLAN encapsulation (tunnel)
net_ns=Not applicable
IP=192.168.1.30
Subnet=B

PF

SRIOV Disabled IB HCA

Kubernetes Docker: IB using Overlay Networking

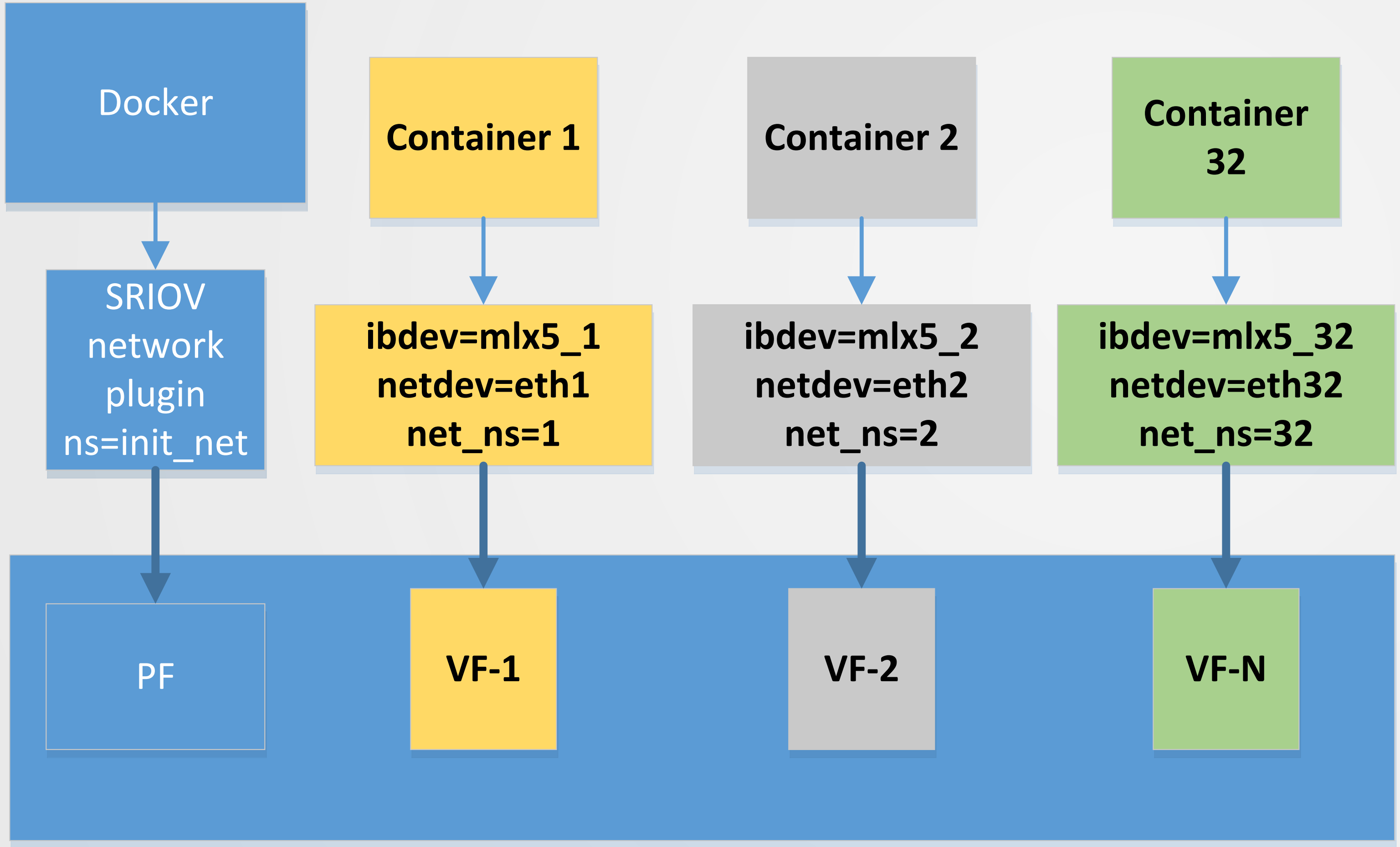
- Networking achieved via VXLAN overlay cni plugin
- IPoIB as parent netdevice
- VXLAN encapsulated packets using IPoIB IP packets

- RDMA device plugin
  - Single RDMA device plugin to share among PODs (containers)

- Large number of PODs per host
- RDMA CM connection manager cannot operate
  - Can be used for applications that have TCP/ssh based connection management

Native Docker: IB/RoCE using SRIOV Networking

	RoCE MAC VLAN	IB Overlay (Shared Dev)	IB/RoCE SRIOV Legacy	RoCE SRIOV OVS
Native Docker	Supported	Not supported	Supported	WIP
Docker Swarm	Not supported	Supported	Not supported	TBD
Kubernetes	Supported (not tested)	Supported	Supported	TBD


Docker
Container 1
Container 2
Container 32
SRIOV network plugin ns=init_net
ibdev=mlx5_1 netdev=eth1 net_ns=1
ibdev=mlx5_2 netdev=eth2 net_ns=2
ibdev=mlx5_32 netdev=eth32 net_ns=32
PF
VF-1
VF-2
VF-N

IB/RoCE SRIOV Enabled HCA

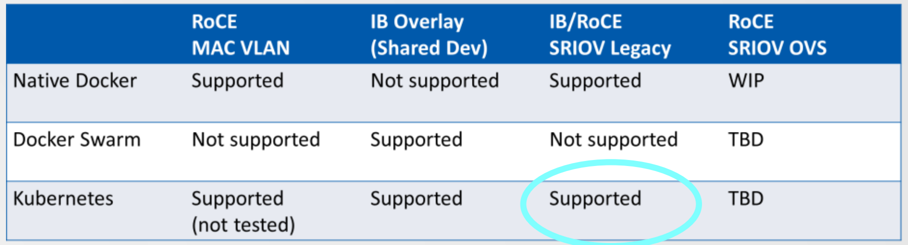
Native Docker: IB/RoCE using SRIOV Networking

- Per Container netdevice and RDMA device
- IB and RoCE link layers supported
  - For InfiniBand, IPoIB netdevice and IB device
  - For Ethernet, Ethernet netdevice and RoCE device
- Applicable for DPDK use-cases
- SR-IOV networking plugin to provision VF for a container
  - VLAN transparent offload
  - VF level rate limiting (BW)
- External docker_rdma_sriov tool for starting containers to provide isolation
  - WIP to eliminate the need for tool

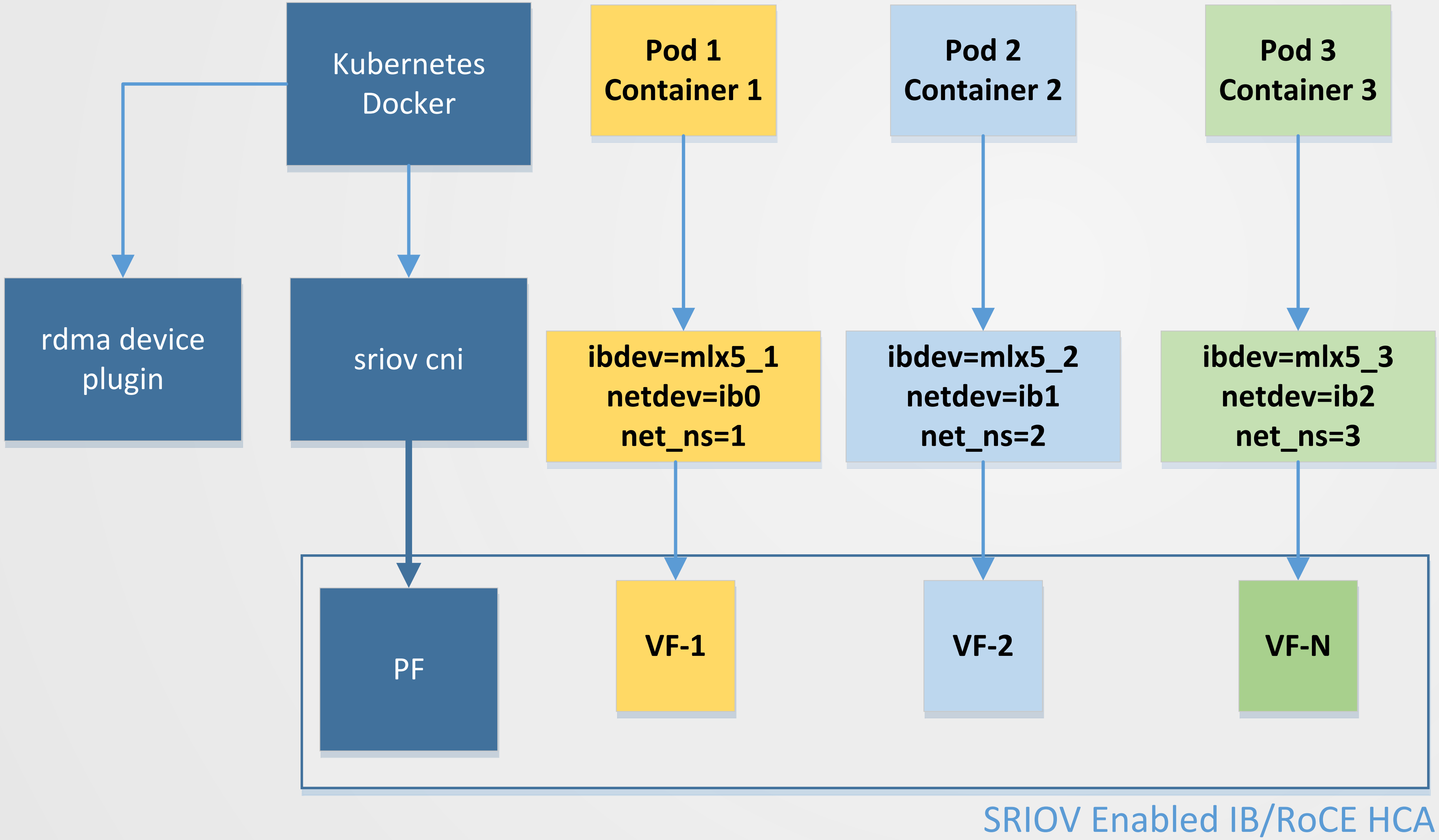
Docker SR-IOV Example

- Example
  - Run SRIOV plugin
    - #docker run -v /run/docker/plugins:/run/docker/plugins -v /etc/docker:/etc/docker --net=host --privileged mellanox/sriov-plugin
  - Get container start tool
    - #docker run --net=host -v /usr/bin:/tmp mellanox/container_tools_install
  - Create network
    - *#docker network create -d sriov --subnet=194.168.1.0/24 -o netdevice=ens2f0 tenant1*
  - Start container
    - *Container1:*
      - *#docker_rdma_sriov run --net=tenant1 -it --ip=192.168.1.9 mellanox/centos_7_4_mofed_4_2_1_2_0_0_60:latest bash*
    - *Container2:*
      - *#docker_rdma_sriov run --net=tenant1 -it --ip=192.168.1.10 mellanox/centos_7_4_mofed_4_2_1_2_0_0_60:latest bash*
- *How to run RDMA application using rdma_cm*
    - *Container-1*
      - *ib_send_bw -R*
    - *Container-2*
      - *ib_send_bw -R <container_1_ip_address>*

Kubernetes Docker: IB/RoCE using SR-IOV Networking



	RoCE MAC VLAN	IB Overlay (Shared Dev)	IB/RoCE SRIOV Legacy	RoCE SRIOV OVS
Native Docker	Supported	Not supported	Supported	WIP
Docker Swarm	Not supported	Supported	Not supported	TBD
Kubernetes	Supported (not tested)	Supported	Supported	TBD


Kubernetes Docker
rdma device plugin
sriov cni
Pod 1 Container 1
Pod 2 Container 2
Pod 3 Container 3
ibdev=mlx5_1 netdev=ib0 net_ns=1
ibdev=mlx5_2 netdev=ib1 net_ns=2
ibdev=mlx5_3 netdev=ib2 net_ns=3
PF
VF-1
VF-2
VF-N
SRIOV Enabled IB/RoCE HCA

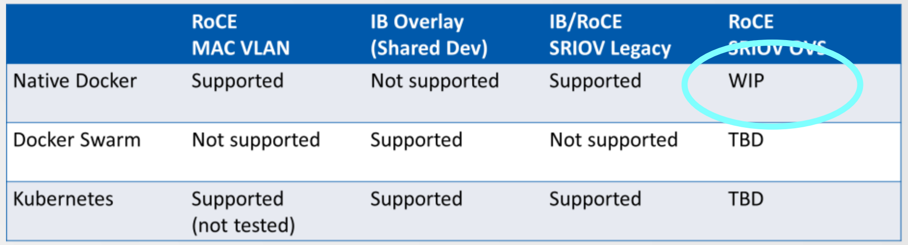
Kubernetes Docker: IB/RoCE using SR-IOV Networking

- Inherits native Docker features

- RDMA device plugin provisions VF
  - Enables SRIOV
  - Configures VF for RDMA and Ethernet
  - Per POD (per container) VF allocation
  - https://hub.docker.com/r/mellanox/k8s-rdma-sriov-dev-plugin/

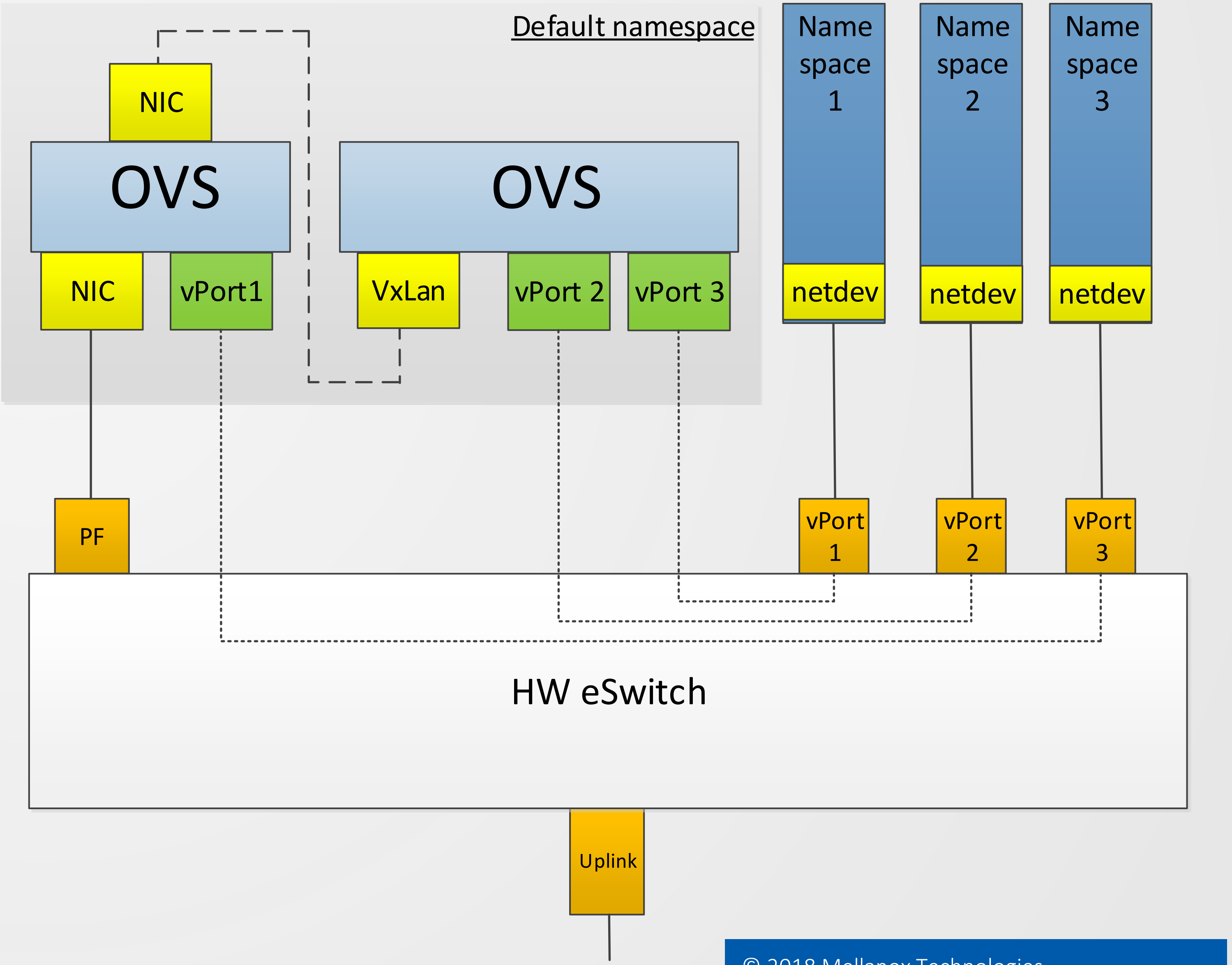
- RDMA sriov cni
  - Provisions netdevice (IPoIB or Ethernet) for POD
  - https://hub.docker.com/r/mellanox/k8s-sriov-cni-installer/

RoCE VXLAN for Containers

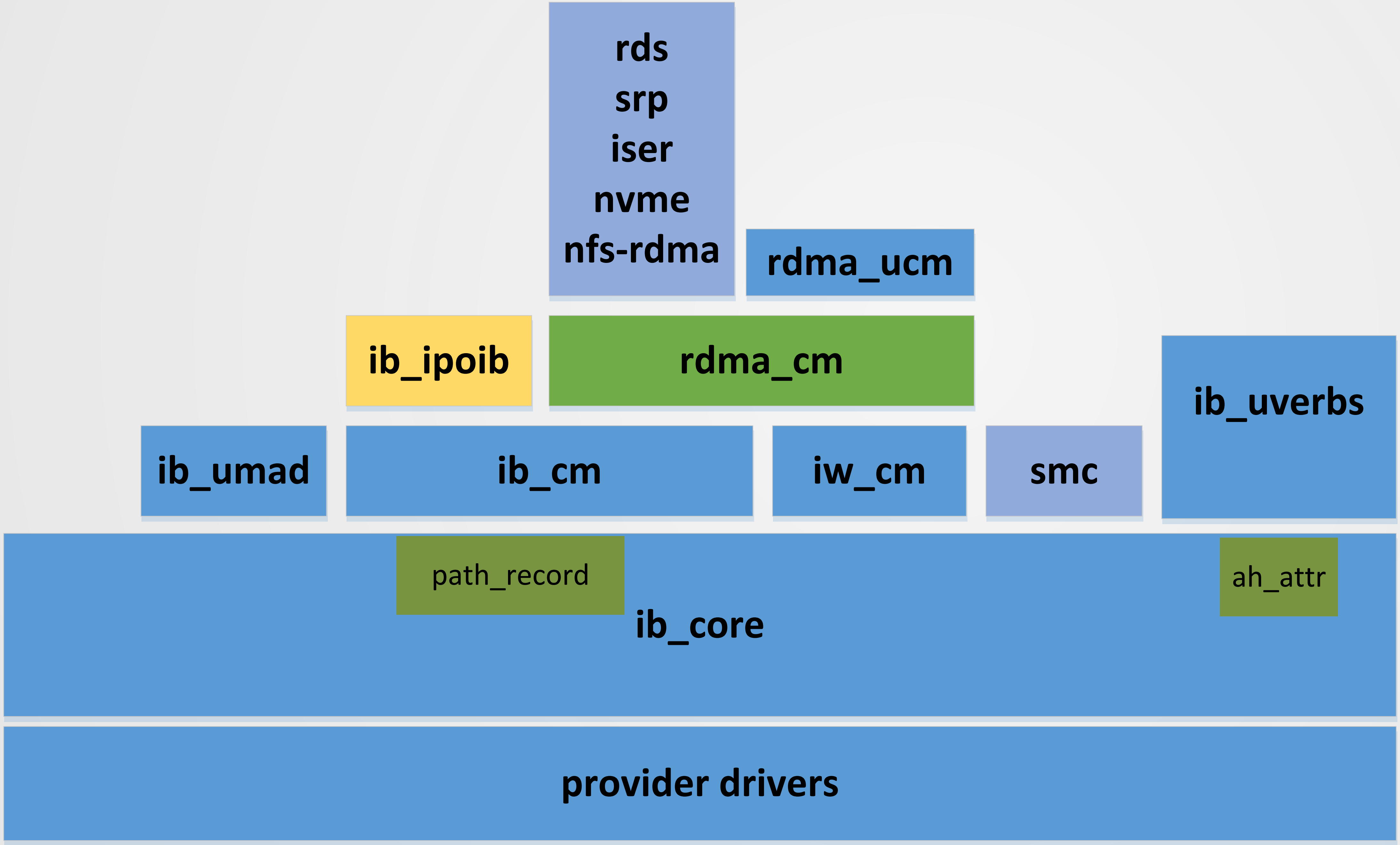


	RoCE MAC VLAN	IB Overlay (Shared Dev)	IB/RoCE SRIOV Legacy	RoCE SRIOV OVS
Native Docker	Supported	Not supported	Supported	WIP
Docker Swarm	Not supported	Supported	Not supported	TBD
Kubernetes	Supported (not tested)	Supported	Supported	TBD

- OVS is a managed switch that supports VXLAN
- OVS 2.8+ support HW offload
- RoCE & DPDK traffic will be offloaded by eSwitch
- Supported today by assign a VF per container



RDMA Stack View - Net Namespace Enforcement



Challenges and Future Plans

- Per containers rdma statistics/counters
- rdmatool
  - Extension to disable RoCEv1 - scales RoCE containers by 2x
  - Making netlink socket per net_ns for rdmatool
- Orchestration challenge
  - Isolation of character (network) device
  - Isolation of sysfs files, attributes
  - CNI extension?
  - Kubernetes
    - Device plugin and network plugin interaction?
      - Device cgroup configuration
      - RDMA cgroup configuration
      - /sysfs bind mounts
- Net namespace resident IB devices
  - IB device resides in single net namespace
  - Optional mode

Thank You