Open Mobile Evolved Core (OMEC)

Enabling 5G w/ Open Source

Christian Maciocco
Principal Engineer, Director of Telecom Systems Research – Intel Labs

ITU-T Net2030 Focus Group – Lisbon – Jan. 13, 2020

Intel Labs

Legal Disclaimer

- This presentation contains the general insights and opinions of Intel Corporation (“Intel”). The information in this presentation is provided for information only and is not to be relied upon for any other purpose than educational. Use at your own risk! Intel makes no representations or warranties regarding the accuracy or completeness of the information in this presentation. Intel accepts no duty to update this presentation based on more current information. Intel is not liable for any damages, direct or indirect, consequential or otherwise, that may arise, directly or indirectly, from the use or misuse of the information in this presentation.
- Intel technologies’ features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at intel.com, or from the OEM or retailer.
- No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.
- Intel, the Intel logo and Xeon are trademarks of Intel Corporation in the United States and other countries.
- *Other names and brands may be claimed as the property of others.
-

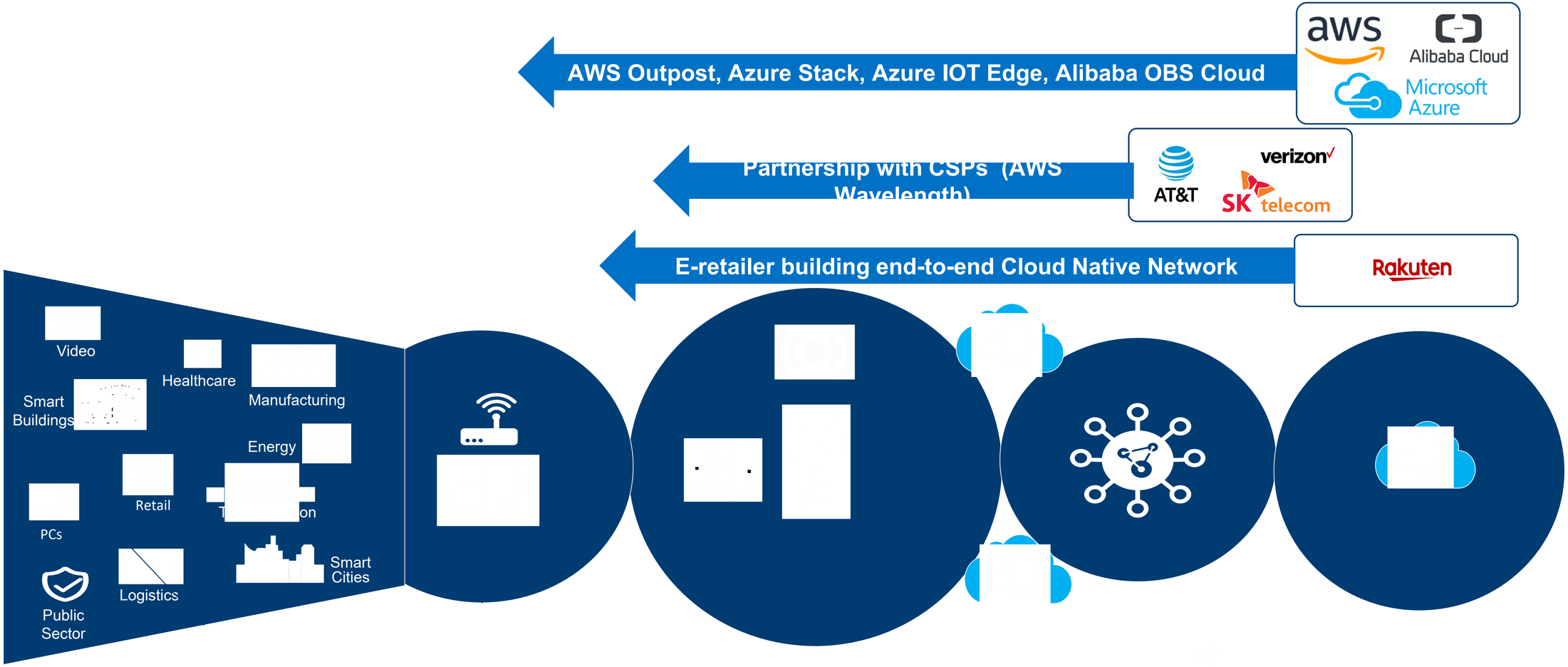
Agenda

- Toward the Network Edge(s)

- Open Mobile Evolved Core (OMEC) in the Open Networking Foundation
  - History, features, distributed edges with OMEC
  - Deployment options (VMs, containers, …) for performance
  - New Modular & Flexible Data Plane

- Summary / Next Steps

Toward the Network Edge(s) (1/2)



LATENCY REDUCTION

RESOURCE EFFICIENCY

LOCALIZED SERVICES

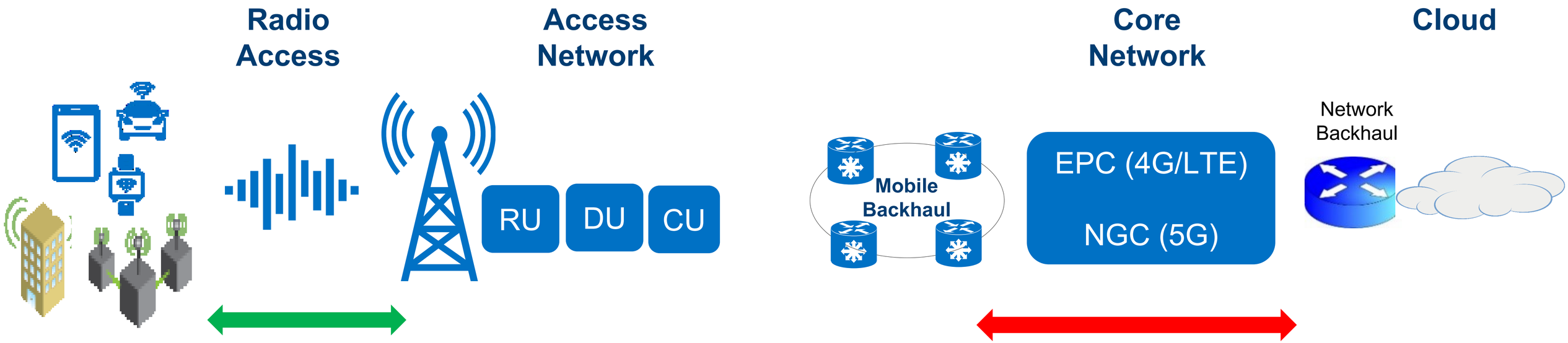
PRIVACY & SECURITY

MULTI-TENANCY

. . .

There is not “one” edge --- Different Edges from Client/Needs Point of View

Toward the Network Edge(s) (2/2)



3GPP 5G
Addresses Cellular Access Bandwidth & Low Latency

Need to Reduce 4G/LTE and 5G Core Latency

Collaborating with industry consortium Open Networking Foundation (ONF) to create open source solutions

RU: Radio Unit
DU: Distributed Unit
CU: Central Unit
EPC: Evolved Packet Core
NGC: Next Generation Core

Agenda

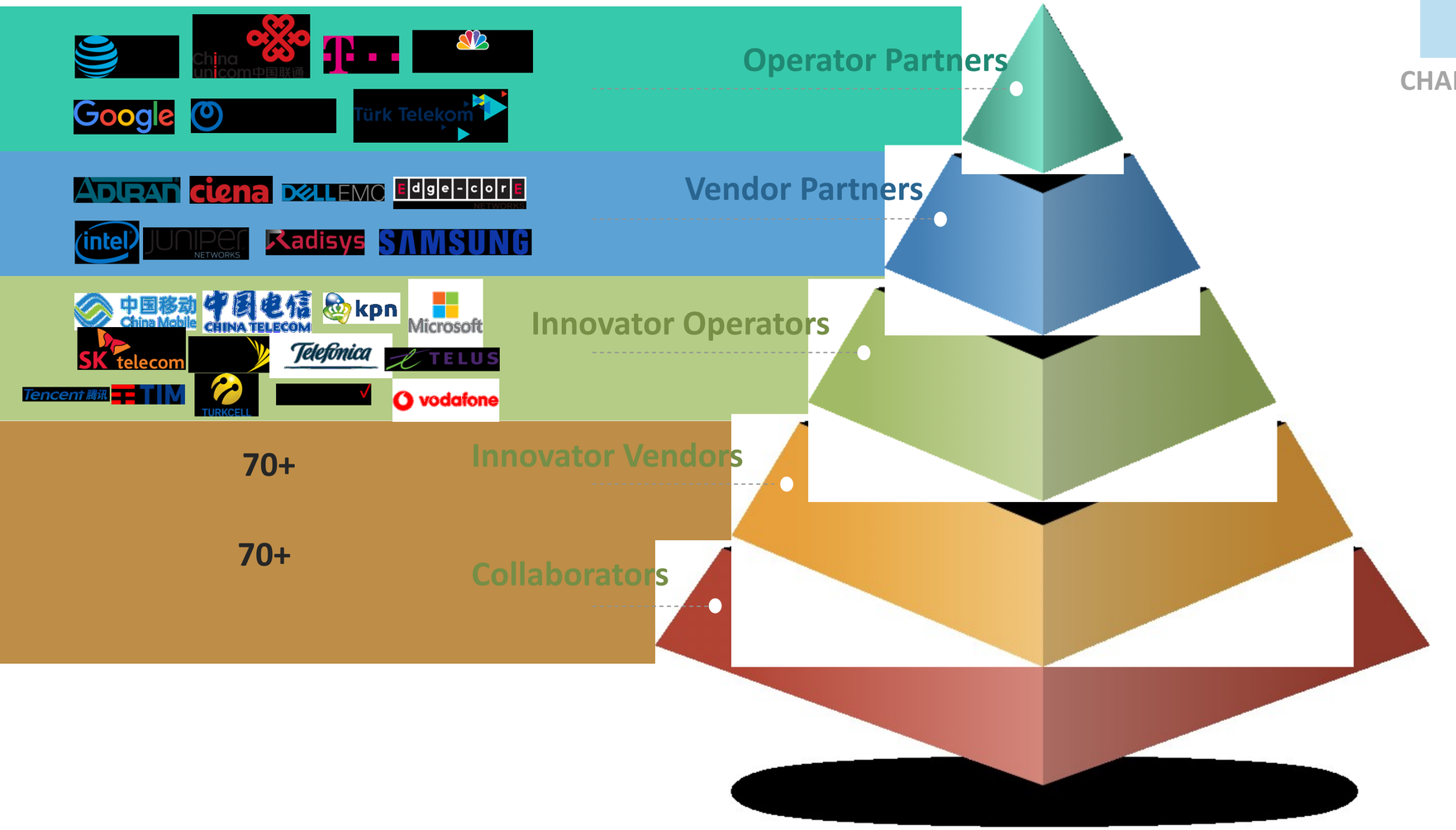
- Toward the Network Edge(s)

- Open Mobile Evolved Core (OMEC) in the Open Networking Foundation
  - History, features, distributed edges with OMEC
  - Deployment options (VMs, containers, …) for performance
  - New Modular & Flexible Data Plane

- Summary / Next Steps

The Open Networking Foundation – 160+ Members Strong

Vibrant Operator Led Consortium Positioned for Success



ONF BOARD	E-SAB* Board
CHAIR:	

E-SAB*: Executive Supplier Advisory Board

ONF Goals

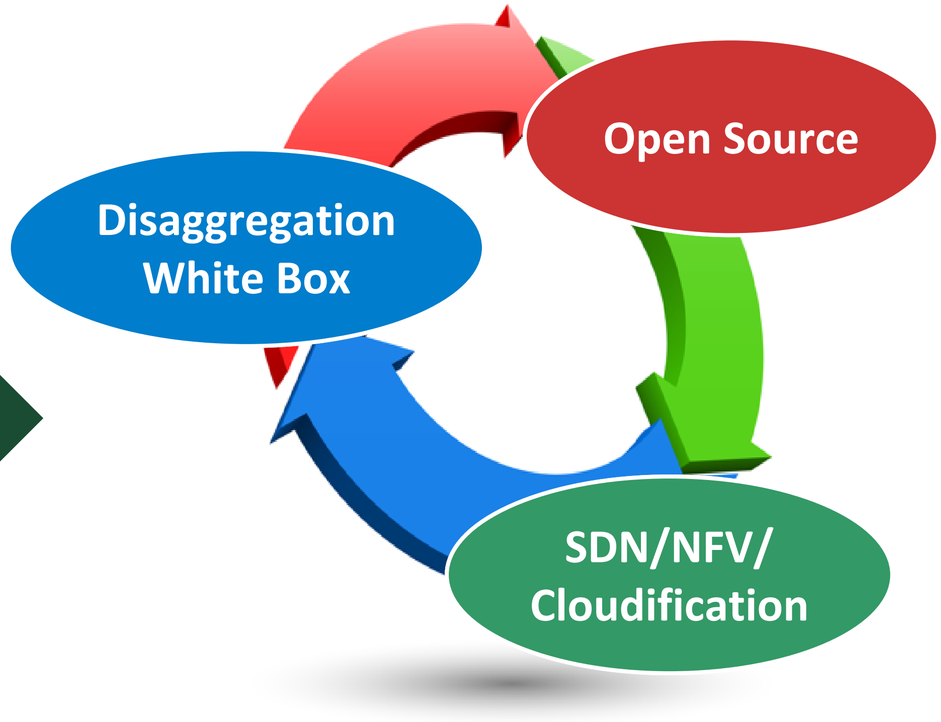
Transform Operator Networks

Why ?

… to bring capex & opex efficiencies & innovative services

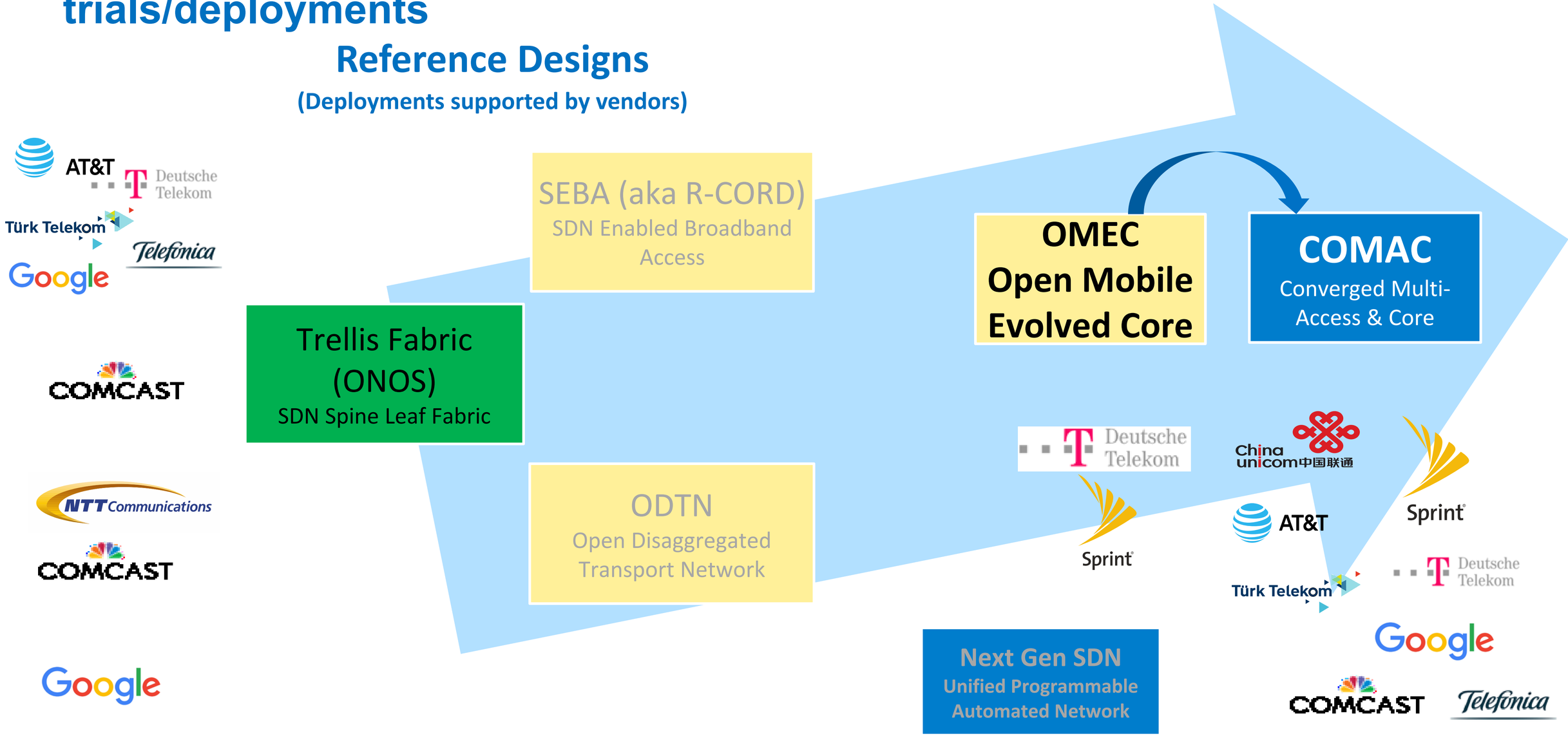
How ?

by leveraging



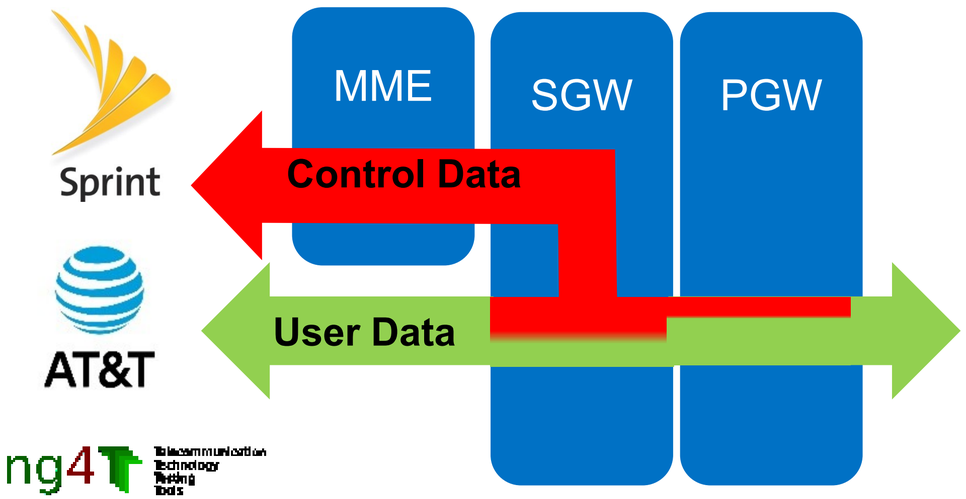
With the focus on Access and Edge

ONF Reference Designs ➔ Operators' Commitment to field trials/deployments



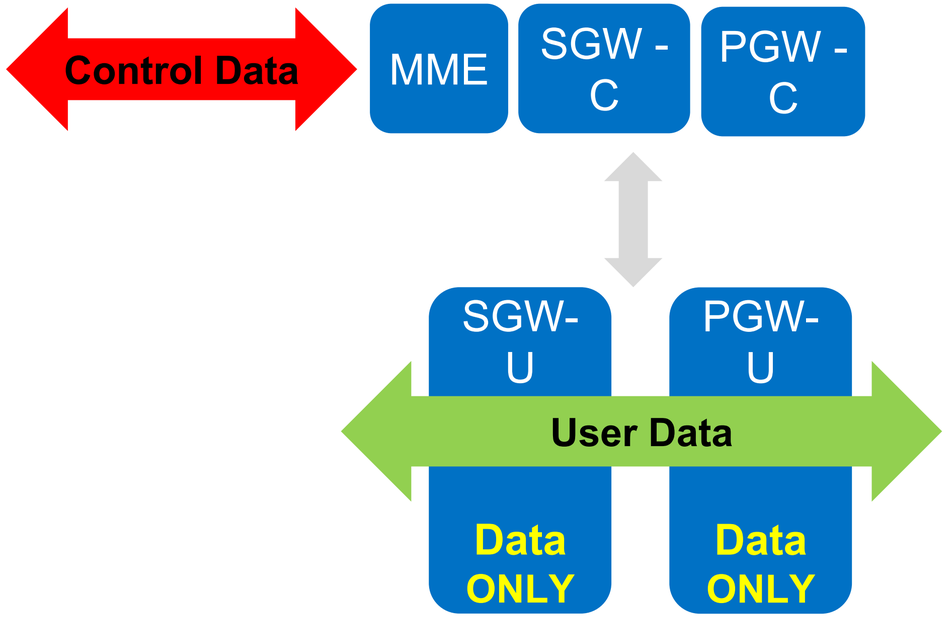
OMEC Evolution Towards the Network Edges

Traditional EPC Architecture



SW Defined Network (Separation of Control and Data)

Disaggregated Architecture



- Operators' real traffic
- Identified system's bottleneck
  - "Understanding Bottlenecks in Virtualizing Cellular Core Network functions", IEEE LANMAN '15
- No independent control or data scaling

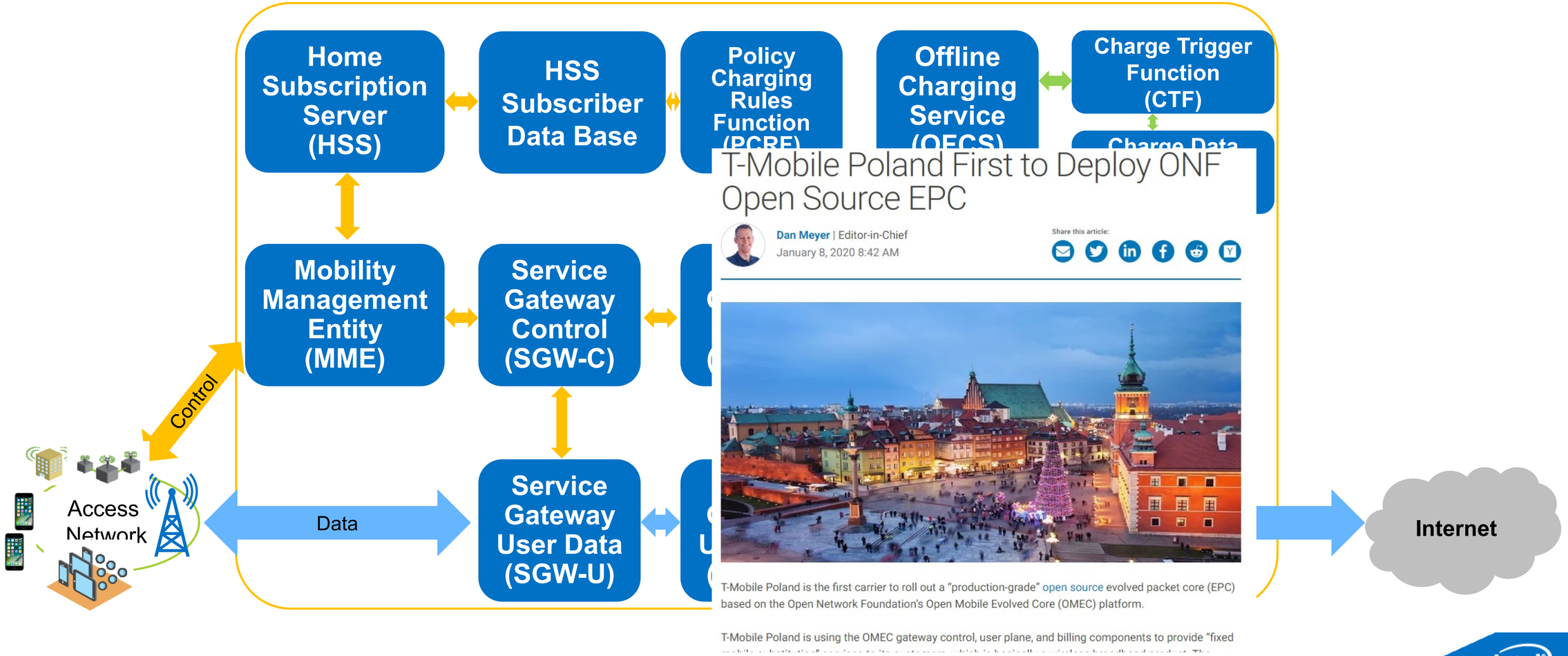
- SDN/NFV based architecture
- High Perf Match/Action semantic data plane
- Independent & scalable control & data
- Functional EPC per operator's requirements
  - E.g. Sprint, DT/T-Mobile Poland

OMEC

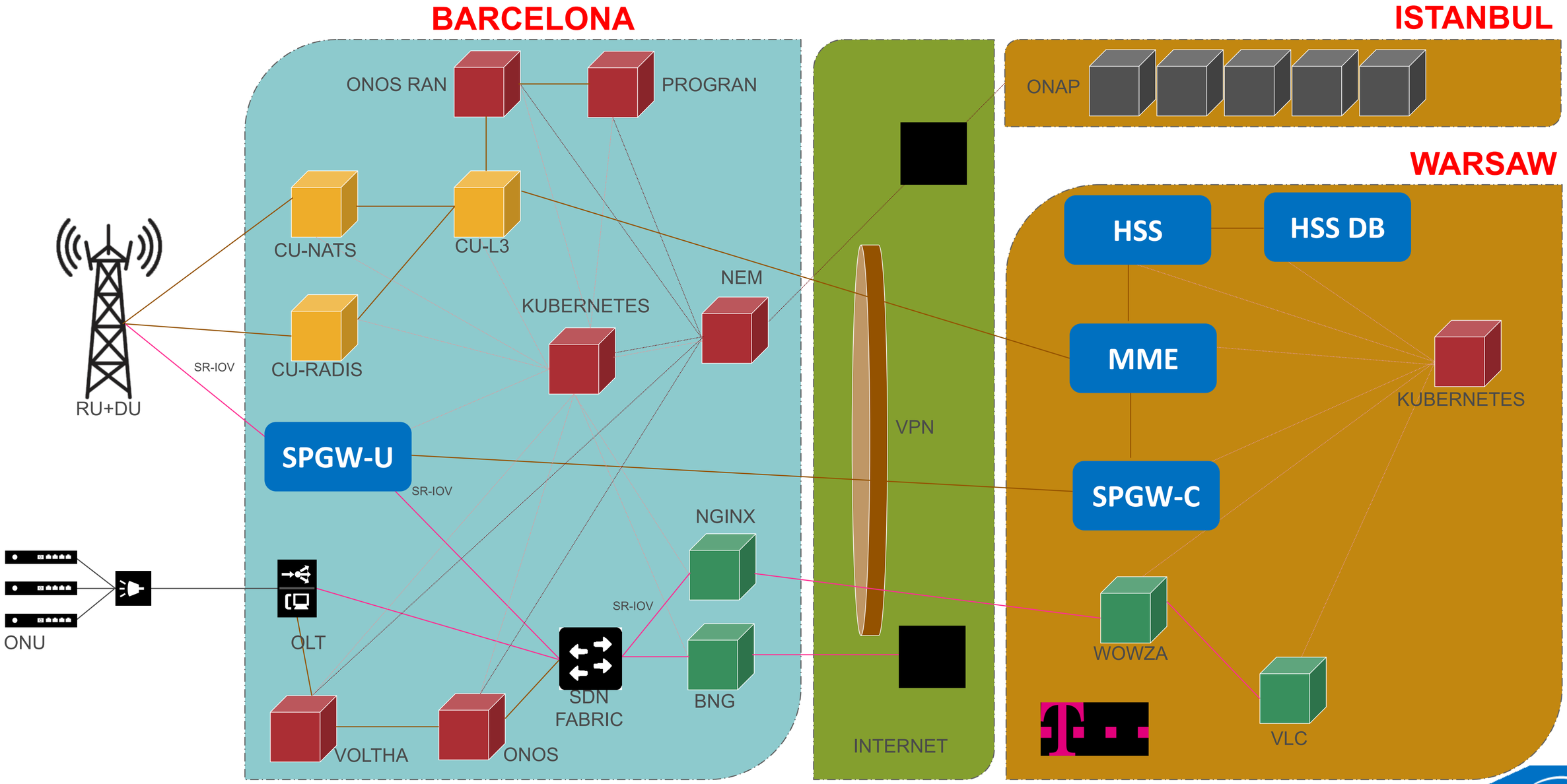
https://www.opennetworking.org/omec/

MME: Mobility Management Engine
HSS: Home Subscriber Services
PCRF: Policy and Charging Rules Function
SGW-C: Service Gateway Control
SGW-U: Serving Gateway User
PGW-C: Packet Gateway Control
PGW-U: Packet Gateway User
OFCS: Offline Charging Service
CTF: Charge Trigger Function
CDF: Charge Data Function

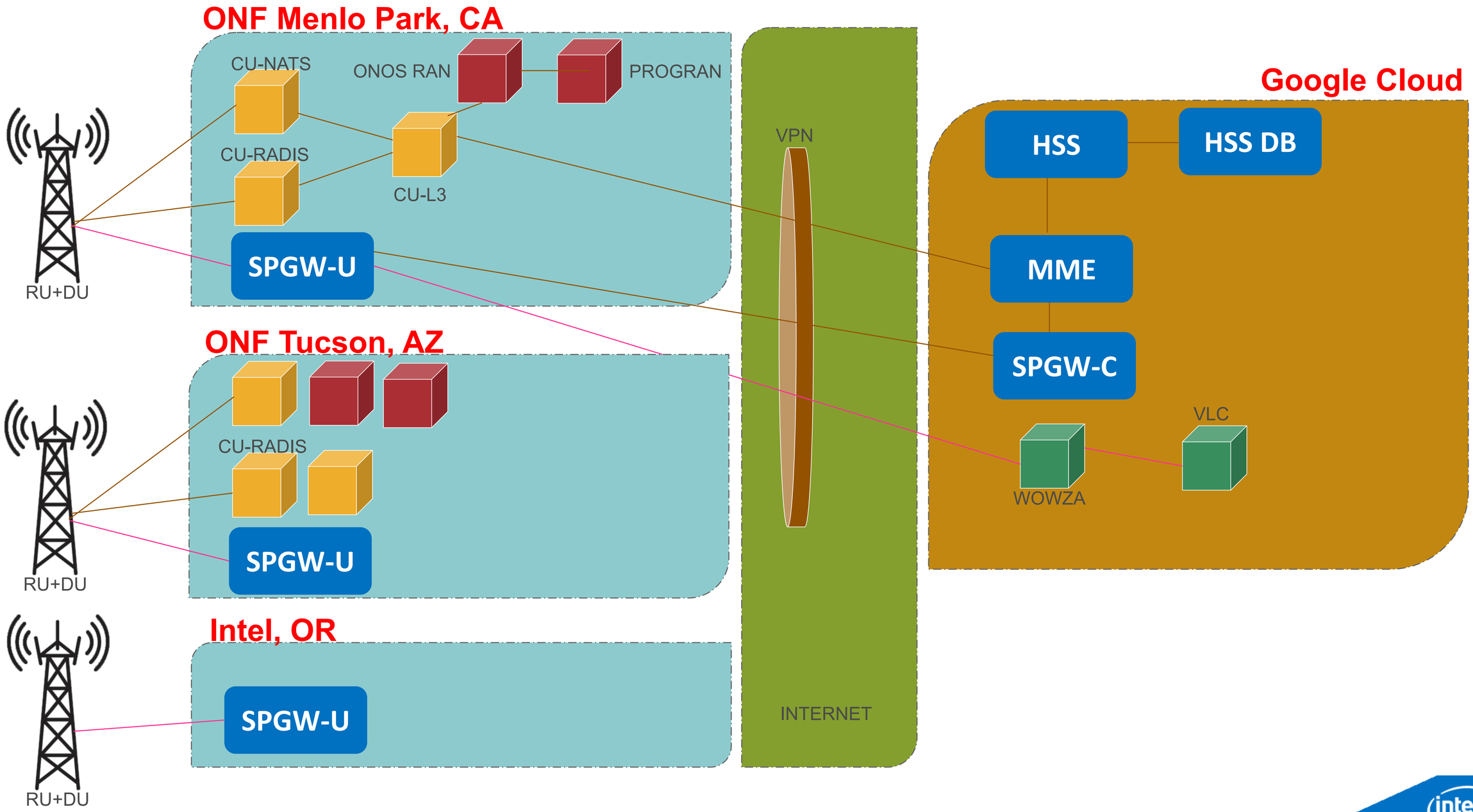
- Complete connectivity, billing and charging
  - Default bearers
  - Offline billing
  - Child protections (domain or 5-tuple)
  - Basic MME (initial attach/detach, etc)
- 3GPP Rel 13 compatibility
- DPDK based data plane, large number of subs
- Optimized for lightweight cost effective deployment
- ONF CI/CD test and verification infrastructure
  - 3GPP Compliance & Performance (w/ Spirent)
- Future
  - TBD: Based on users' requests and contributions



OMEC @ MWC ’19: Multi-Cloud Deployment



Edge-as-a-Service with Distributed Data Plane for Lower Latency

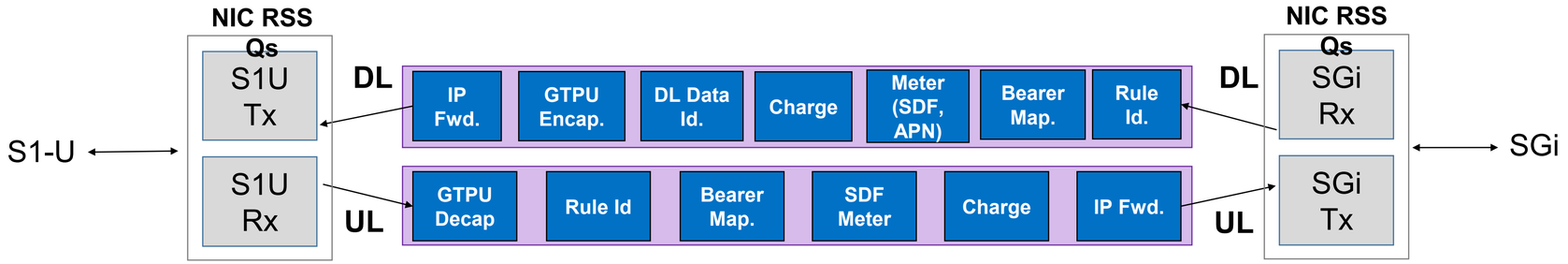


Agenda

- Toward the Network Edge(s)
- Open Mobile Evolved Core (OMEC) in the Open Networking Foundation
  - History, features, distributed edges with OMEC
  - Deployment options (VMs, containers, …) for performance
  - New Modular & Flexible Data Plane
- Summary / Next Steps

OMEC based Containers orchestrated by Kubernetes

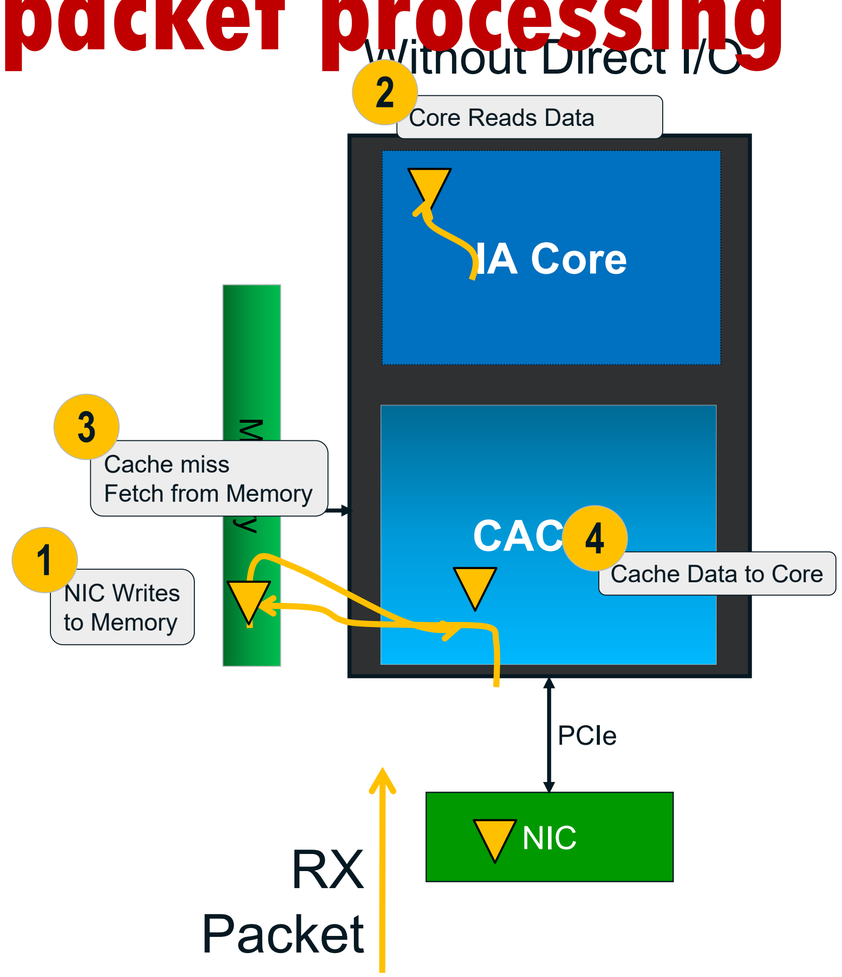
- Data Plane scaling up/out

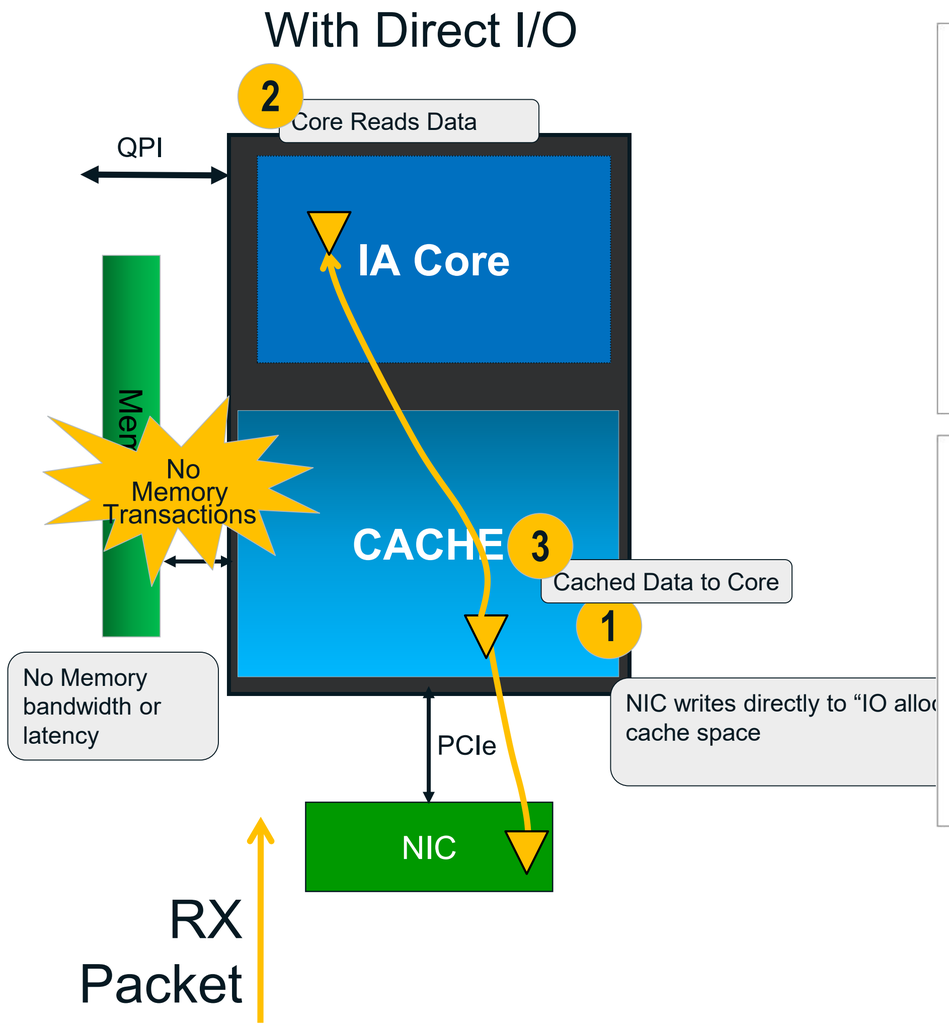


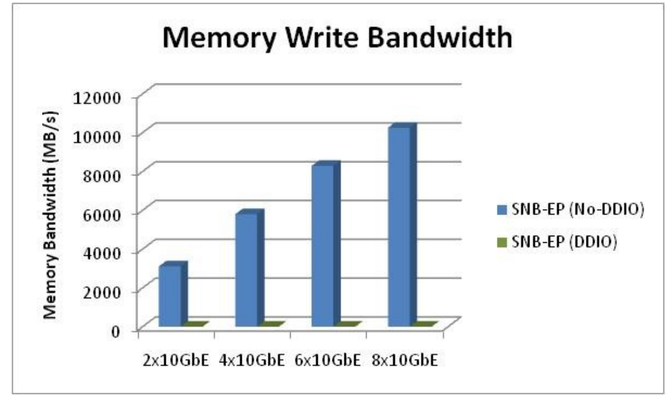
- Support multiple networks per POD with **high-throughput & low latency I/O**
- POD/Container performance optimizations
  - CPU core pinning and isolation
  - Huge Pages
- Ability to do service discovery on other networks

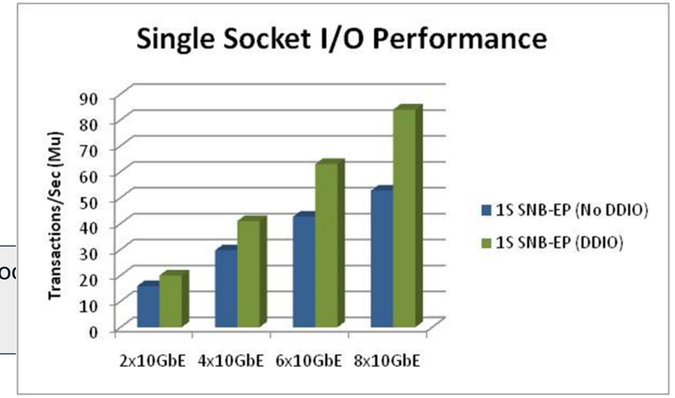
Intel® Direct Data IO (DDIO) technology for efficient packet processing

Build







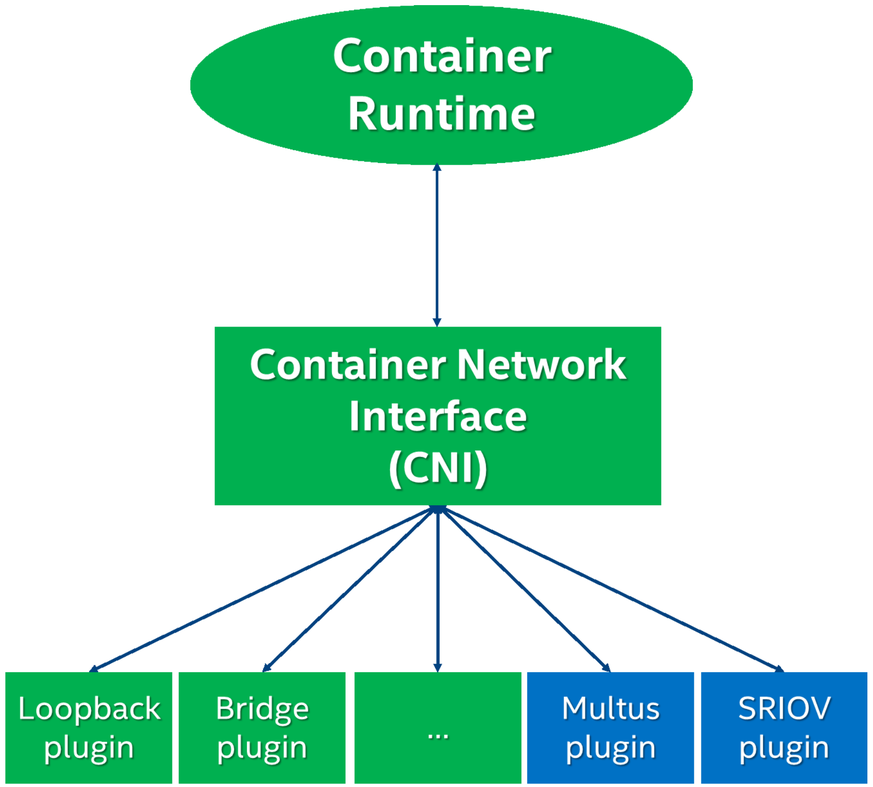


Intel® DDIO significantly improves data movement reducing latency and energy, improving throughput

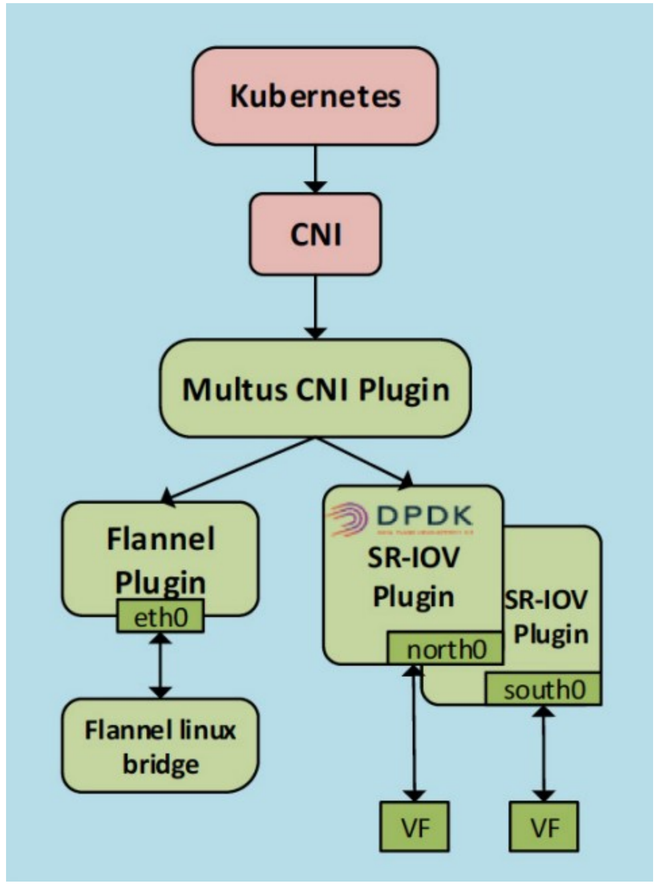
intel Labs

Overview: Multi-Network in Kubernetes (1/2)

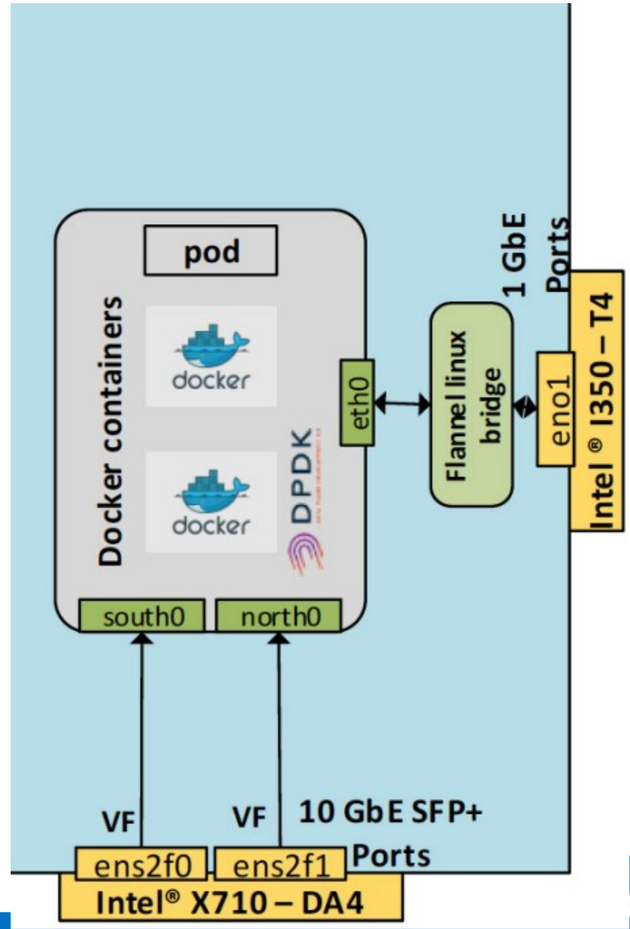
- K8s networking model limits one IP/interface per pod
- On Pod bring up kubelet calls out to the CNI registered on the node to setup networking
- For multi-interfaces we use Multus CNI which acts as a proxy, to set up extra networks



Logical



Physical Manifestation



K8s or 3rd party

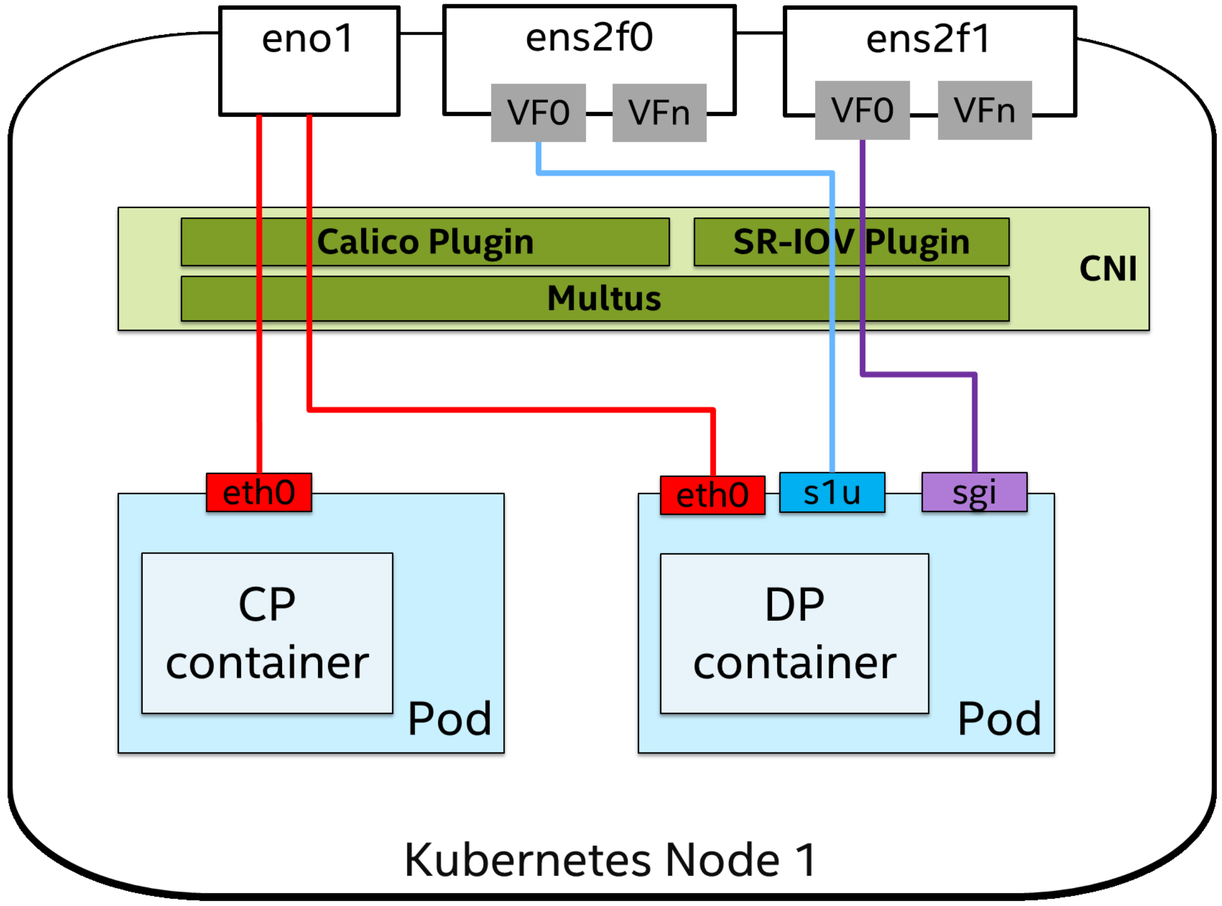
Intel

https://builders.intel.com/docs/networkbuilders/enabling_new_features_in_kubernetes_for_NFV.pdf

Overview: Multi-Network in Kubernetes (2/2)

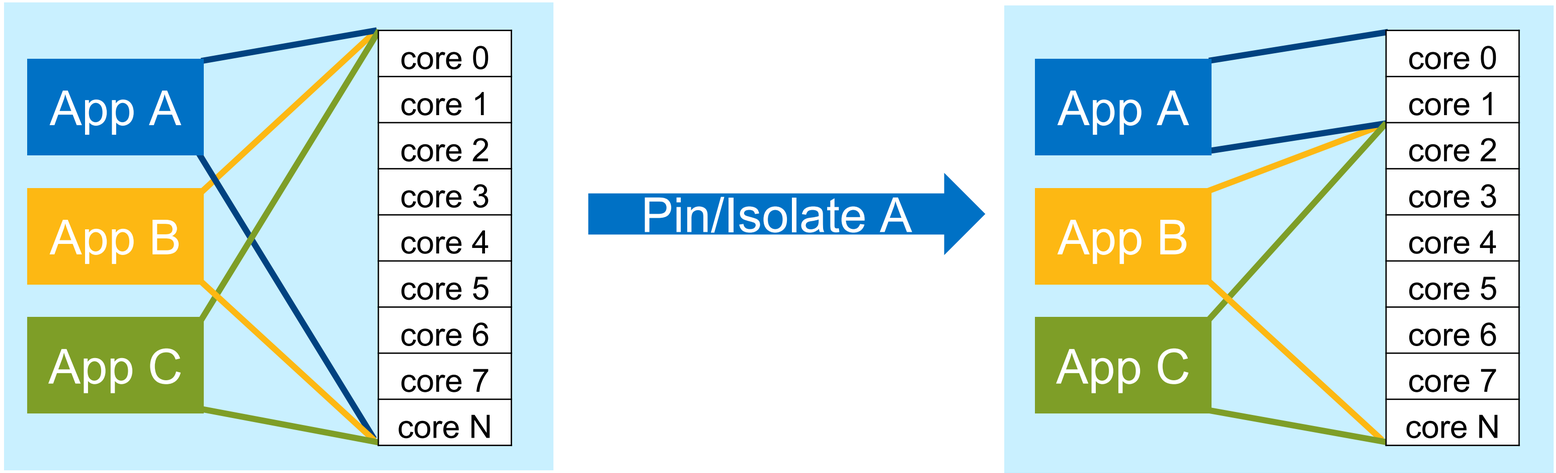
✓ Multiple networks with **high-throughput and lower latency I/O for DP**

- Multus CNI plugin and SR-IOV CNI plugin (enables VFs + DPDK user space drivers)



Overview: Performance w/ Kubernetes (1/3)

- Core pinning/affinity and isolation
  - CPU manager for K8s: Automated core mask gen for DPDK apps

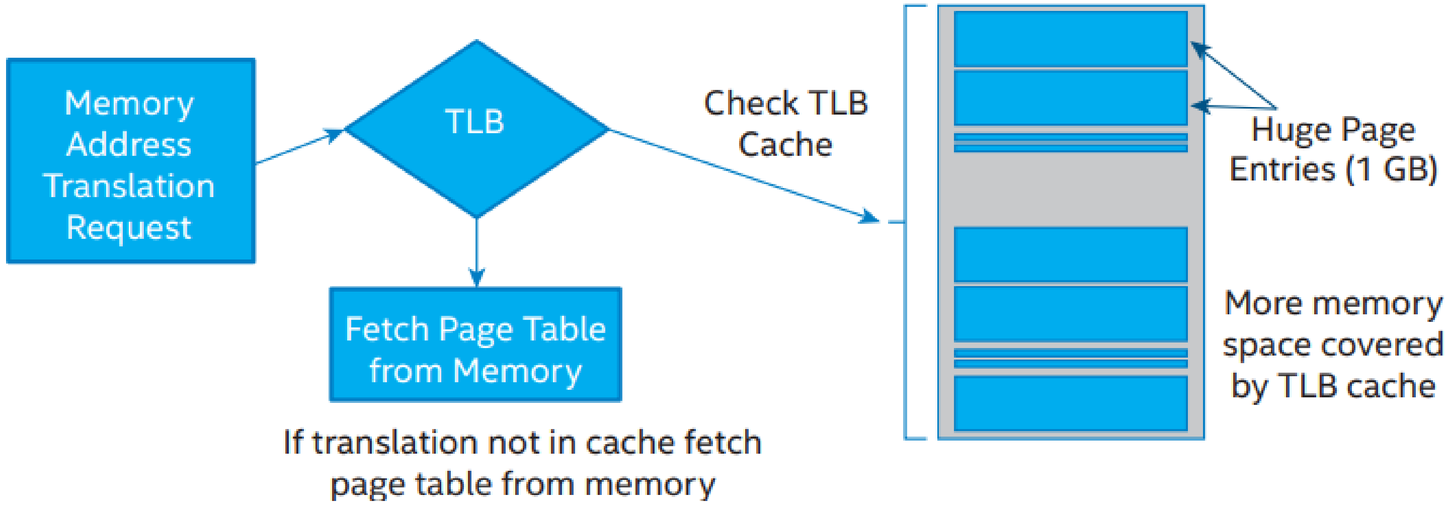



```
apiVersion: kubelet.config.k8s.io/v1beta1
kind: KubeletConfiguration
# Allowing for CPU pinning and isolation in case of guaranteed QoS class
cpuManagerPolicy: static
```

```
apiVersion: v1
kind: Pod
metadata:
  name: grtd-qos-example
spec:
  containers:
  - image: fedora
    command:
    - sleep
    - inf
    name: example
    resources:
      limits:
        memory: 200Mi
        cpu: 2
      requests:
        memory: 200Mi
        cpu: 2
```

Overview: Performance w/ Kubernetes (2/3)

- Huge Pages
  - Native resource in K8s




```
apiVersion: v1
kind: Pod
metadata:
  name: hugepages-example
spec:
  containers:
  - image: fedora
    command:
    - sleep
    - inf
    name: example
    volumeMounts:
    - mountPath: /hugepages
      name: hugepage
    resources:
      limits:
        hugepages-2Mi: 100Mi
        memory: 200Mi
      requests:
        memory: 100Mi
  volumes:
  - name: hugepage
    emptyDir:
      medium: HugePages
```

https://networkbuilders.intel.com/docs/kilo-a-path-to-line-rate-capable-nfv-deployments-with-intel-architecture-and-the-openstack-kilo-release.pdf

Overview: Performance w/ Kubernetes (3/3)

- Native – Running the binaries manually. No containers, no orchestration
- Automated: Kubernetes – Containers orchestrated with performance knobs ON

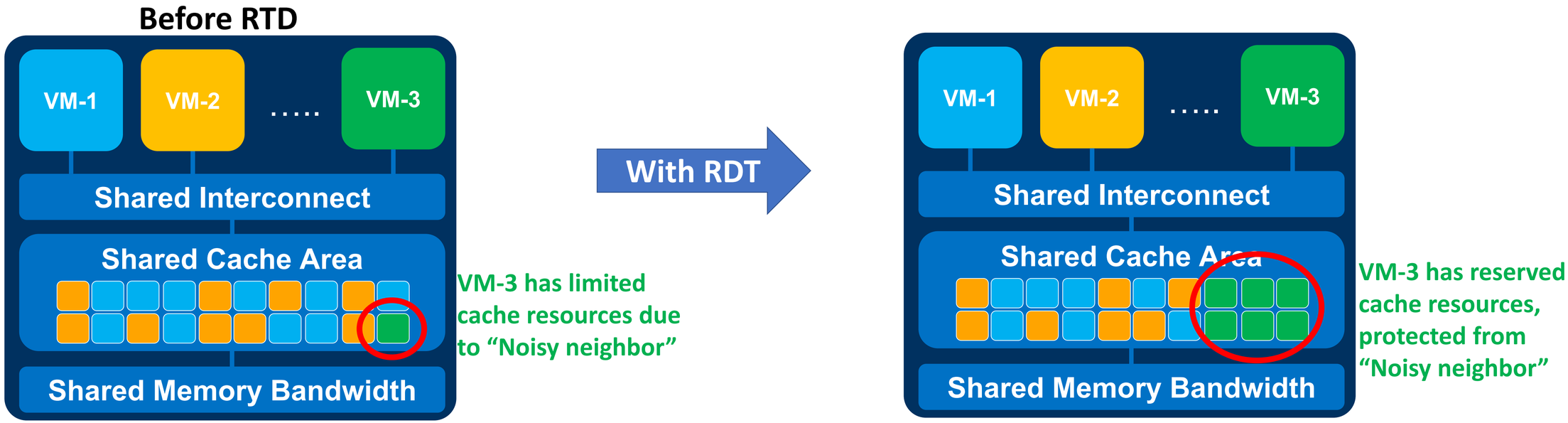
Test	Usr Sp Drv	Pinning	Huge	Pkts/sec*	*(w/noise)*
Native	yes	yes	yes	**1,550K**	*(1,100K)*
Kubernetes	yes	yes	yes	**1,450K**	*(1,150K)*
Kubernetes	**no**	yes	yes	**750K**	*(650K)*
Kubernetes	yes	**no**	yes	**1,450K**	***400K***
Kubernetes	yes	yes	**no**	**1,200K**	*(1,100K)*

* 50K Granularity (1 Worker Core)

Consider NUMA selecting which CPU core will run workload on which socket

Intel® Resource Director Technology for determinism and Qos

- Multi Core systems have shared resources : Cache area, memory bandwidth, IO bandwidth
- A Virtual Machine (VM) may over-utilize shared resources and be a “Noisy neighbor” to others
- Intel® RDT can monitor, detect and provide facilities to fix such issues – Shared resources can be allocated per VMs



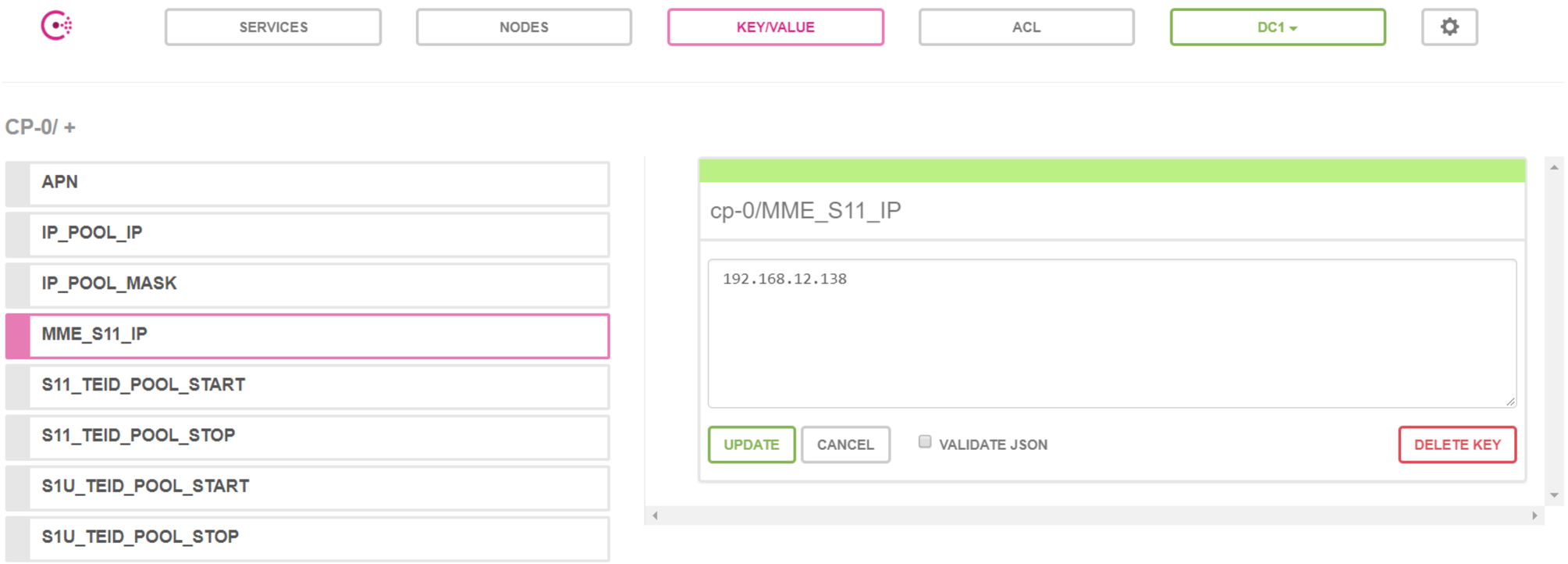
Intel® RDT provides guarantee on data movement inside the platform for improved determinism and QoS

intel Labs

Overview: Service Discovery

✓ Support multiple networks per POD and high-throughput I/O

- Ability to do service discovery on other networks
  - Any key/value service, e.g. used Consul* to store and distribute discovery/configuration data



*Other names and brands may be claimed as the property of others.

Container/Orchestration Current Status

- Dockerfiles: SPGW-C, SPGW-U, HSS, HSS-DB, OpenMME
- k8s YAMLs
  - SPGW-C, SPGW-U in tree omec-project/ngic-rtc/deploy/k8s
  - HSS and HSS-DB are in PR omec-project/c3po/pull/21
  - OpenMME - TBD
- COMAC-in-a-box

Agenda

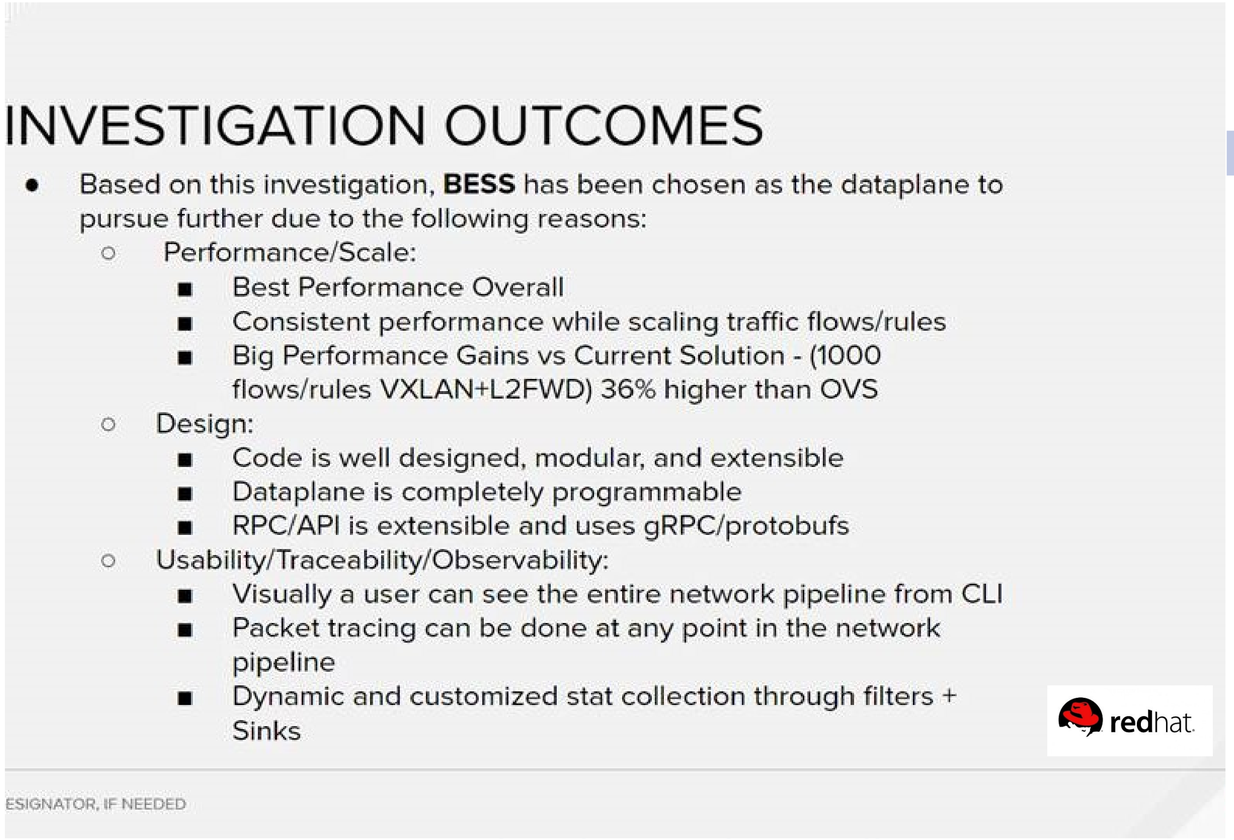
- Toward the Network Edge(s)
- Open Mobile Evolved Core (OMEC) in the Open Networking Foundation
  - History, features, distributed edges with OMEC
  - Deployment options (VMs, containers, …) for performance
  - New Modular & Flexible Data Plane
- Summary / Next Steps

New Modular and Flexible Data Plane

- Goals:
  - Modular architecture
  - Developers concentrate only on VNF development and not infrastructure development
  - Scalable & deployable in containerized environment (i.e. k8s automated deployment)
  - Ease of customization at run-time, e.g. CPU scheduling, adding/removing functionality
  - Monitoring at run-time
    - Debug facilities
    - Pipeline visualization tools w/ statistics, etc

Bess use across industry + academia

- **Redhat did an independent evaluation of data-planes Q4'19 - (VPP, VPP+XDP, XDP+eBPF, OVS, Open FastPath, Tungsten Fabric, BESS)**



INVESTIGATION OUTCOMES

- Based on this investigation, **BESS** has been chosen as the dataplane to pursue further due to the following reasons:
  - Performance/Scale:
    - Best Performance Overall
    - Consistent performance while scaling traffic flows/rules
    - Big Performance Gains vs Current Solution - (1000 flows/rules VXLAN+L2FWD) 36% higher than OVS
  - Design:
    - Code is well designed, modular, and extensible
    - Dataplane is completely programmable
    - RPC/API is extensible and uses gRPC/protobufs
  - Usability/Traceability/Observability:
    - Visually a user can see the entire network pipeline from CLI
    - Packet tracing can be done at any point in the network pipeline
    - Dynamic and customized stat collection through filters + Sinks

redhat

ESIGNATOR, IF NEEDED

1. CTO Control and Data Plane Full Investigation Doc

- ★ "**BEST PERFORMANCE OVERALL**"
- ★ "**Big Performance Gains vs Current Solution**"
- ★ "**.. modular, and extensible**"

- **ACM CoNEXT '19: "Comparing the Performance of State-of-the-Art Software Switches for NFV," Institut PolyTech de Paris, Nokia Bell Labs**

"BESS achieves both high throughput and low latency in physical-to-physical, physical-2-virtual, and 1-VNF loopback scenarios."

intel Labs

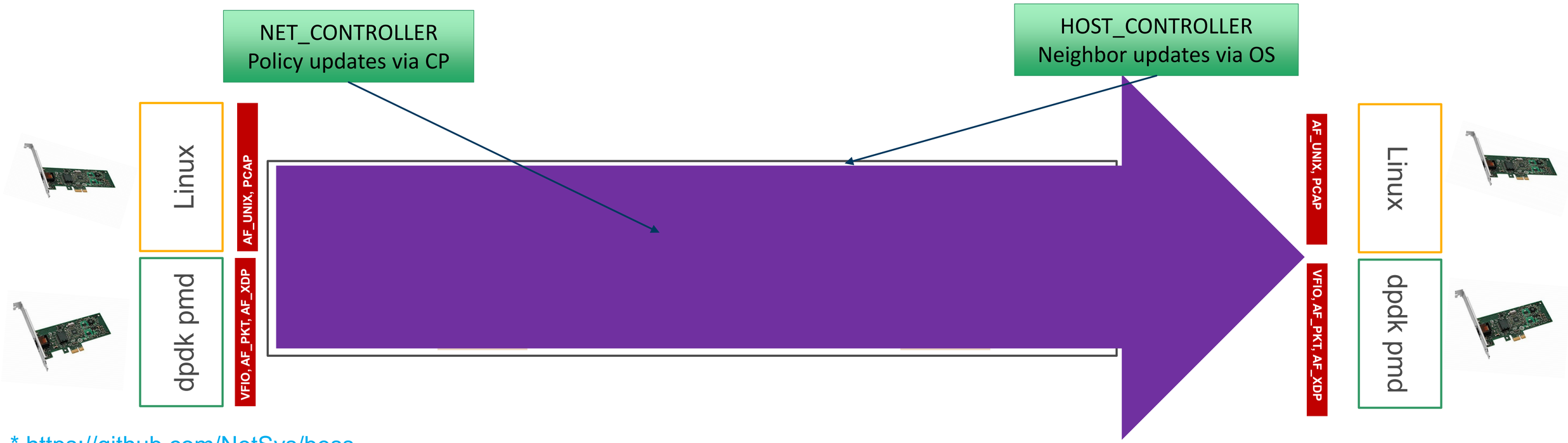
Berkeley Extensible Soft Switch (BESS)*

Clean-slate internal architecture with NFV in mind, highly flexible and customizable

BESS applications as modular pipeline represented as a directed acyclic graph (DAG)

- Each module can run arbitrary code, independently extensible and optimizable

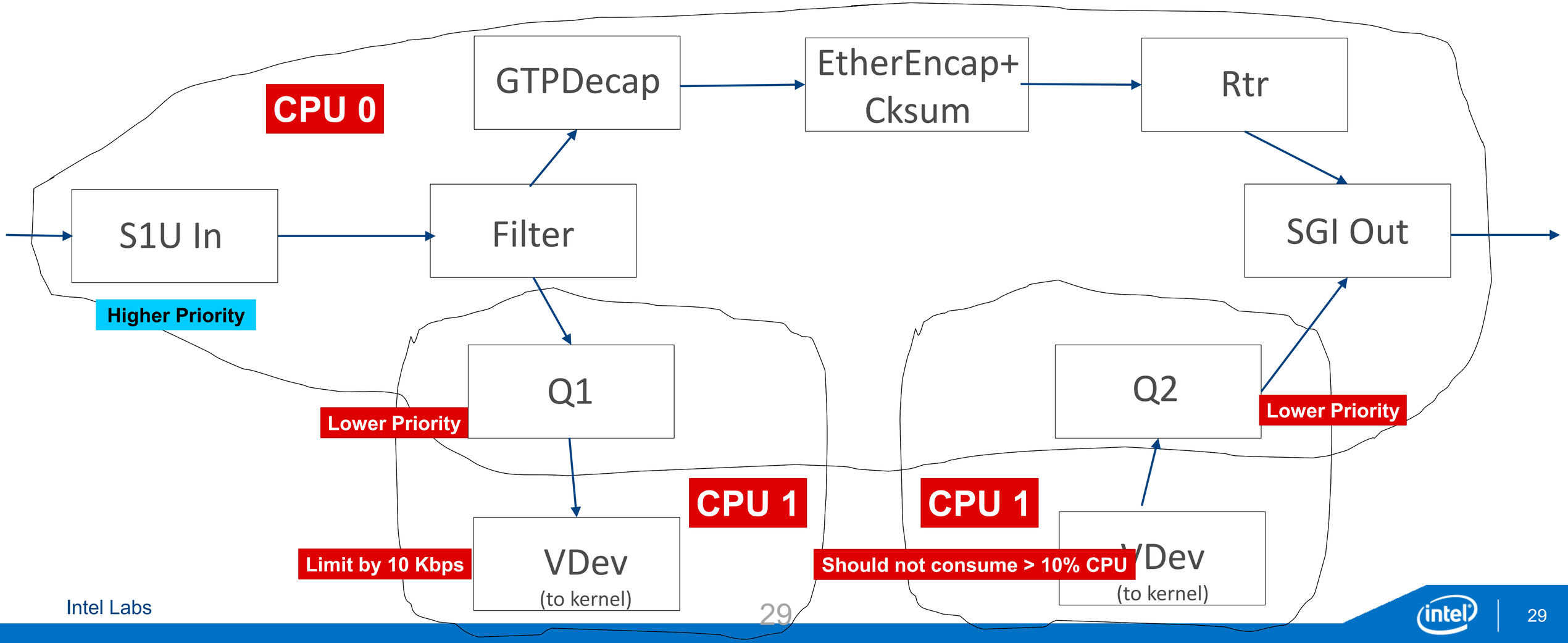
Configuration and control via NF controller



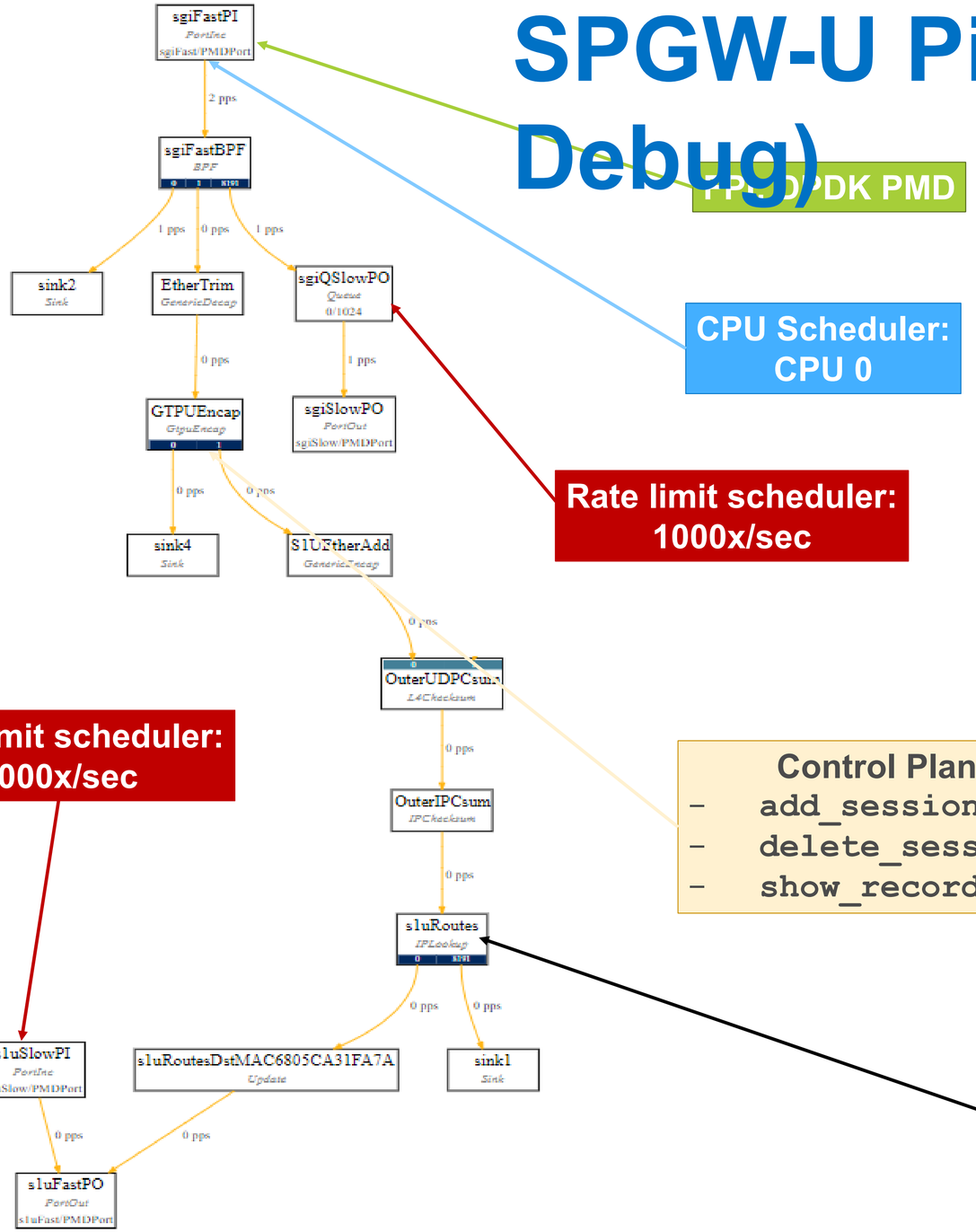
* https://github.com/NetSys/bess

BESS Flexible Resource Scheduling

E.g. CPU utilization & bandwidth



SPGW-U Pipeline Visualization (& Debug)



DPDK PMD

CPU Scheduler: CPU 0

Rate limit scheduler: 1000x/sec

Rate limit scheduler: 1000x/sec

Control Plane

- `add_session()`
- `delete_session()`
- `show_records()`

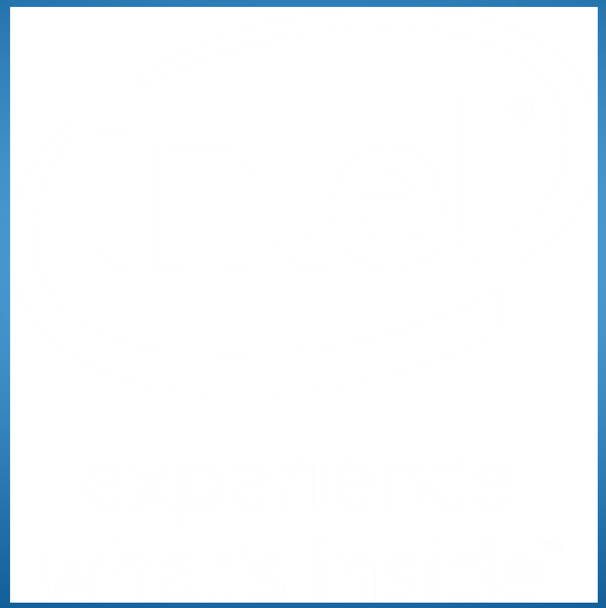
Route Control

- `insert_route()`
- `delete_route()`
- `add_neighbor()`
- `delete_neighbor()`

Summary

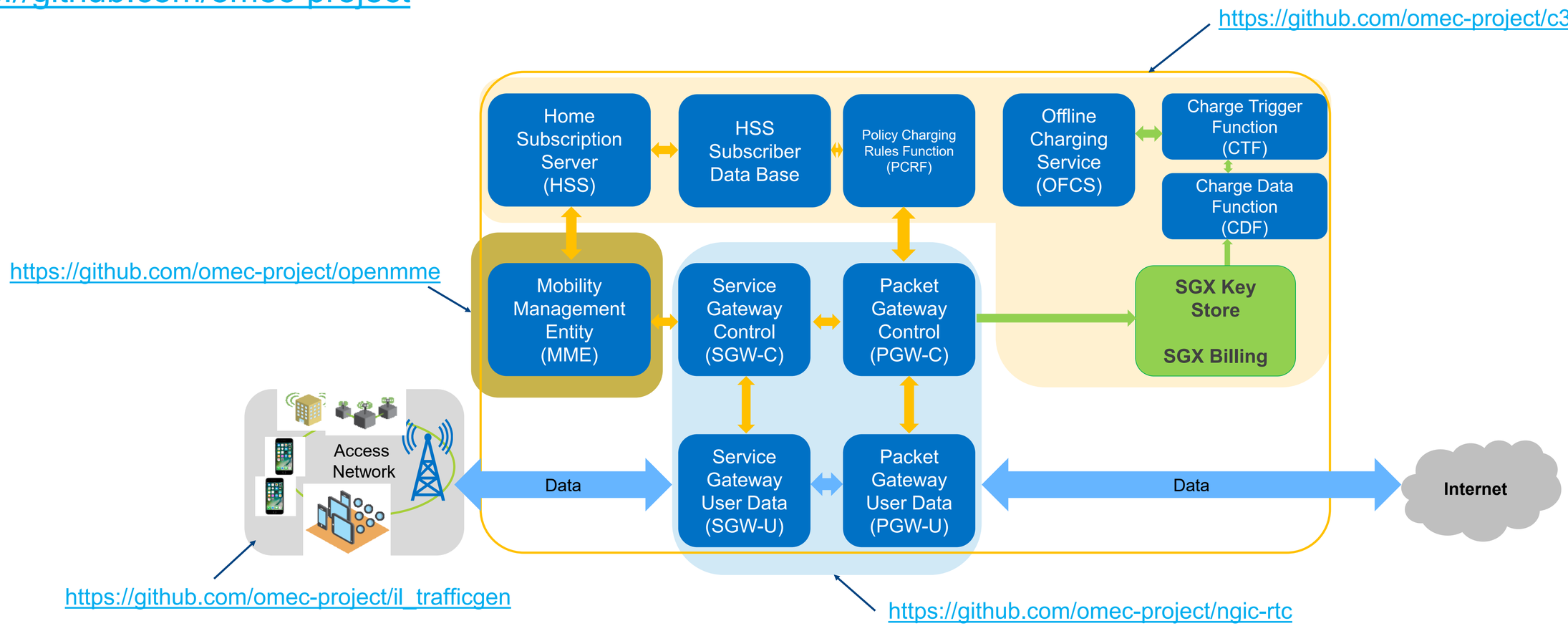
- OMEC is available
  - DTAG/T-Mobile Poland running field trials for Fixed Mobile Services in ‘19
  - System Integrators involved, e.g. GS.Lab, HCL, Infosys

- OMEC needs your contributions
  - Join OMEC GitHub @ https://github.com/omec-project
  - Contribute to any of the repos



OMEC GitHub Repositories

https://github.com/omec-project



Additional repos:

- CI/CD: https://github.com/omec-project/omec-project-ci

- Deployment: https://github.com/omec-project/deployment

- Free Diameter: https://github.com/omec-project/freediameter

- CLI, etc: https://github.com/omec-project/oss-util