COMAC Deep Dive

Pingping Lin, Hyunsun Moon, Badhrinath Padmanabhan, Doyoung Lee, Woojoong Kim
ONF

Contents

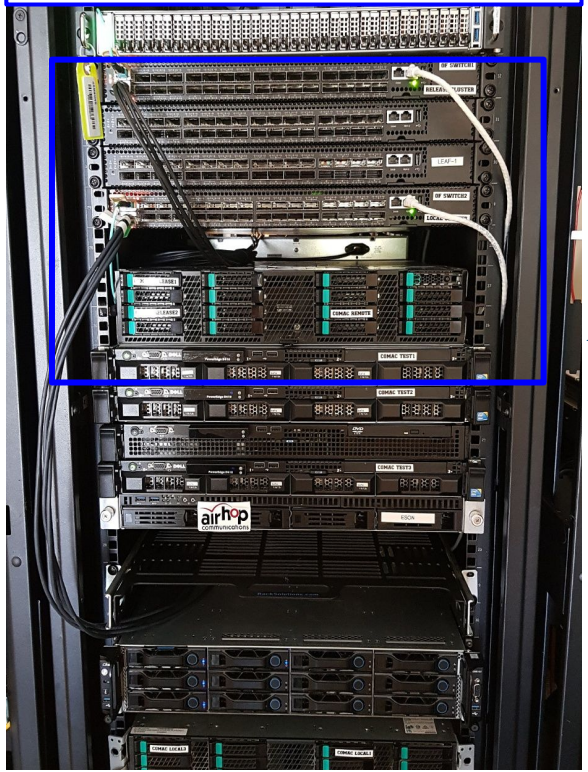
- Part 1: “Multi-cluster Physical Setup” Pingping
- Part 2: “K8S & Helm, OMEC” Hyunsun
- Part 3: “Subscriber Monitoring” Badhrinath
- Part 4: “Monitoring & Visualization” Doyoung
- Part 5: “CDN & XOS in COMAC” Woojoong

Contents

- **Part 1: “Multi-cluster Physical Setup” Pingping**
- Part 2: “K8S & Helm, OMEC” Hyunsun
- Part 3: “Subscriber Monitoring” Badhrinath
- Part 4: “Monitoring & Visualization” Doyoung
- Part 5: “CDN & XOS in COMAC” Woojoong

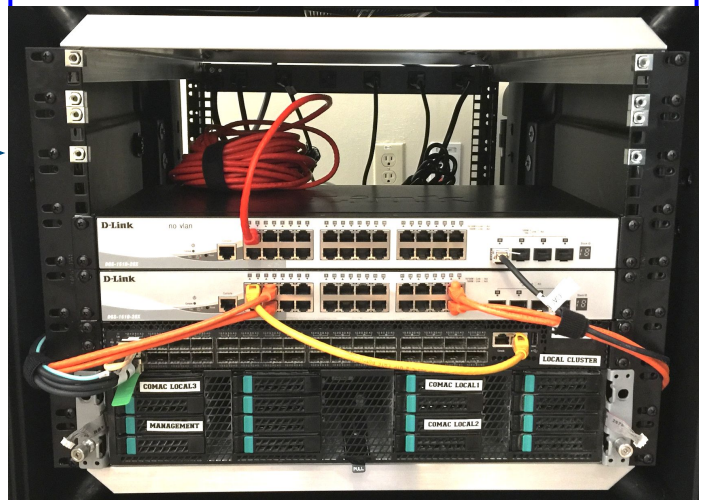
Multi-Cluster COMAC Demo Setup

Central Cluster in ONF Menlo Park



Internet

Edge Cluster in ONF-Connect Conference



Multi-Cluster BOM

	Quantity	Category	Brand	Model	Part
Edge Cluster **Standard setup**	3	64-bit x86 server	/	/	Haswell X86 microarchitecture or better 1G intf and 10G intf
	4	Openflow switch	EdgeCore	AS6712-32X	
	1	L2 Management Switch	D-Link	DGS-1510-28X	Support both 1G and 10G
	1	Cabling (data plane, fiber)	Robofiber	QSFP-40G-03C	
	6	Cabling (management, copper)	/	CAT6, 3M	
Central Cluster	3	64-bit x86 server	/		Haswell X86 microarchitecture or better
	1	L2 Management Switch	D-Link	DGS-1510-28X	
	3	Cabling (management, copper)	/	CAT6, 3M	

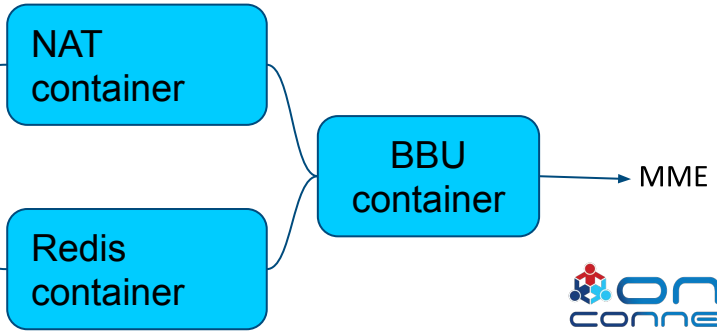
Integrated RAN and Phone So Far

- Phone: Samsung J5
  - Android version 7.1.1

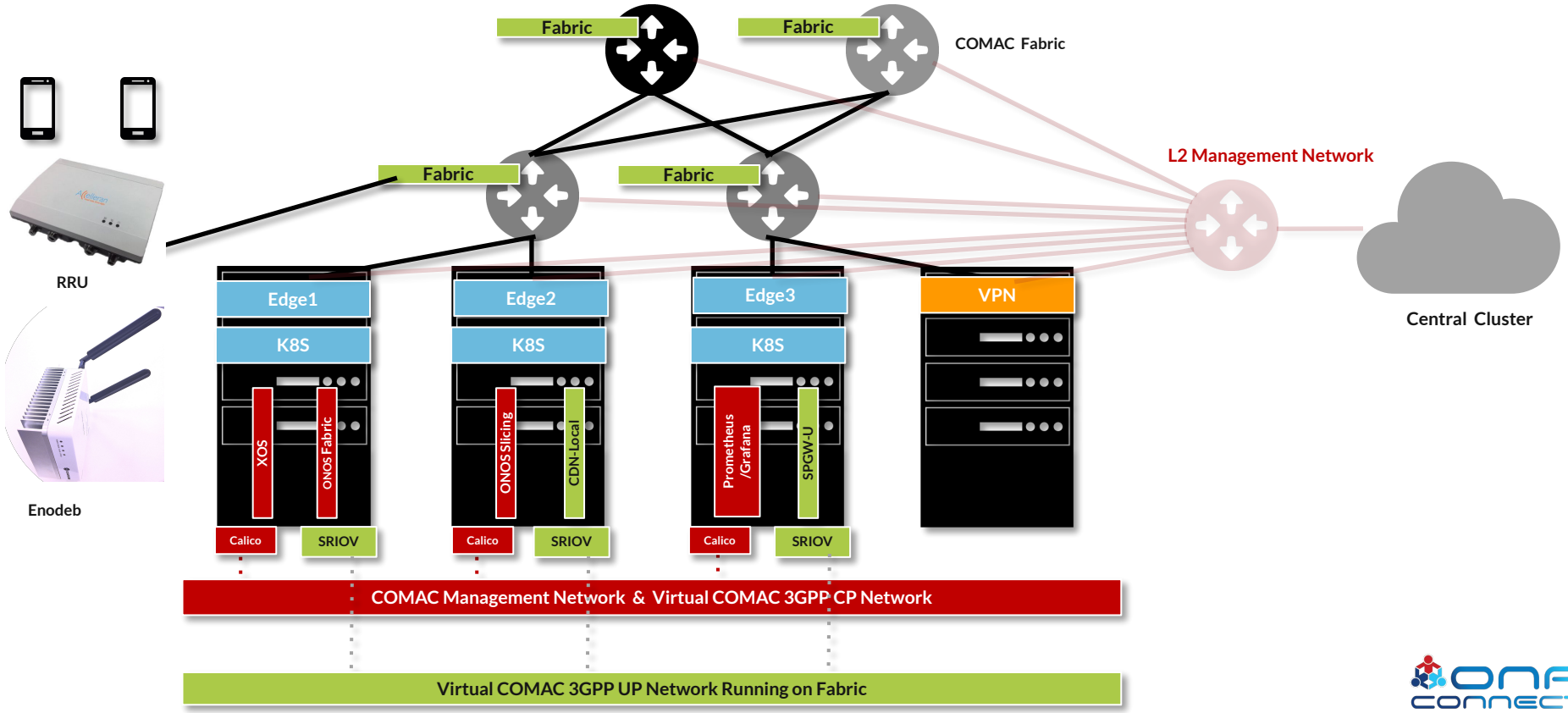
- Cavium standalone Enodeb
  - Model : CNF7100-RF2-RF17

- Accelleran splitted enodeb
  - Model : e1000

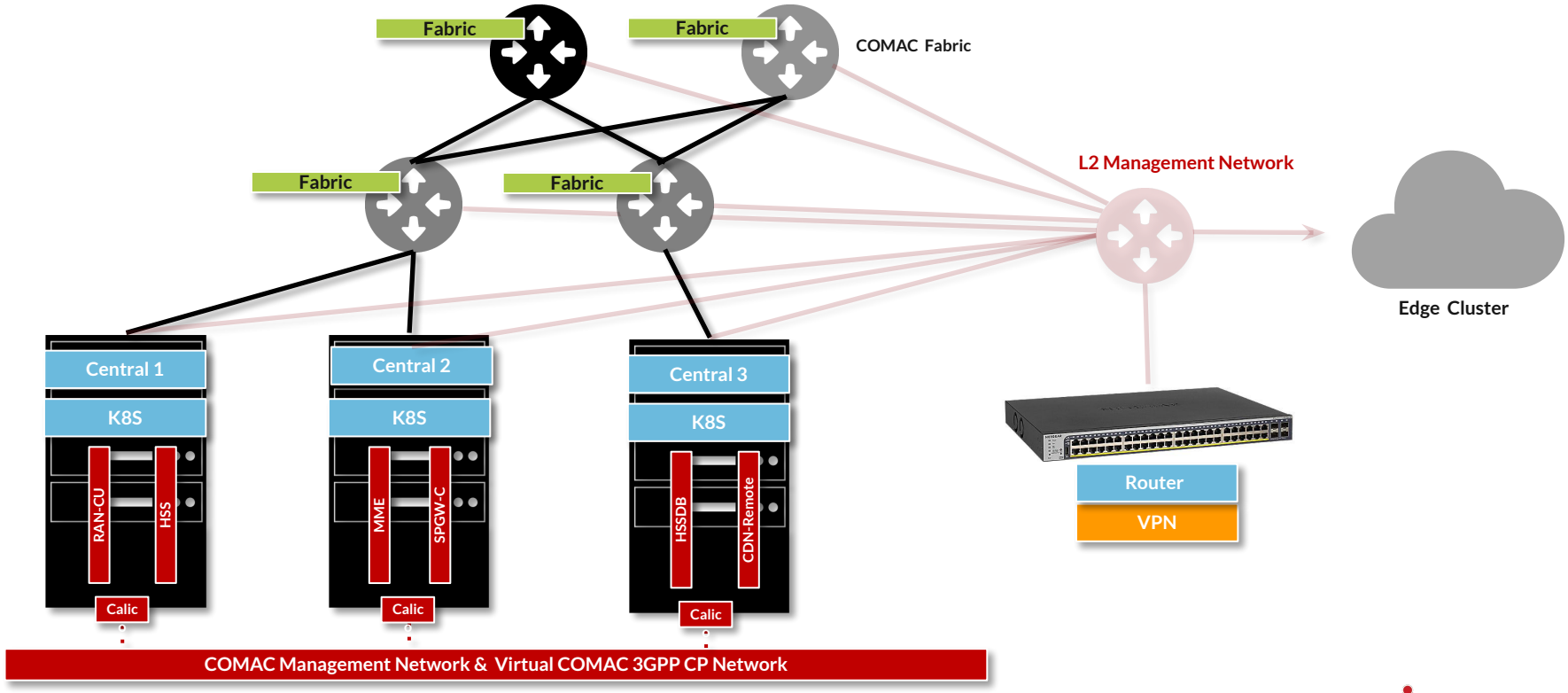
RRU



Edge Cluster Setup

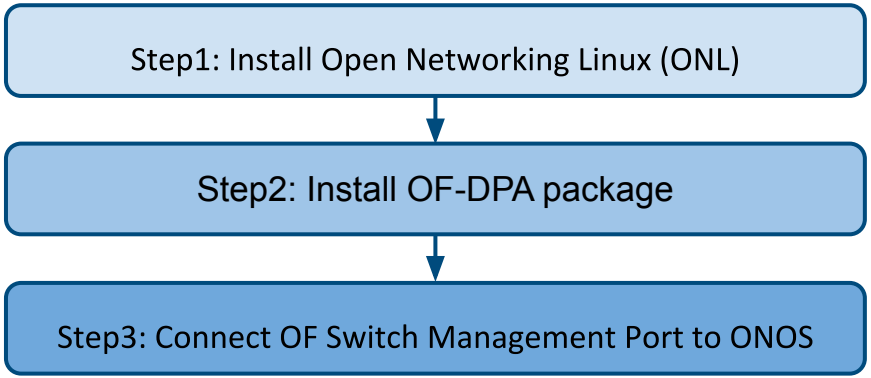


Central Cluster Setup



Trellis in COMAC

- Trellis wiki: https://wiki.opencord.org/display/CORD/Trellis+Underlay+Fabric
- One time manual work:



- Modify configuration on OF switch
  - All 32 ports are running in 1x40G mode.
  - Modify ***/etc/accton/ofdpa.conf*** to break out 1x40G into 4x10G.

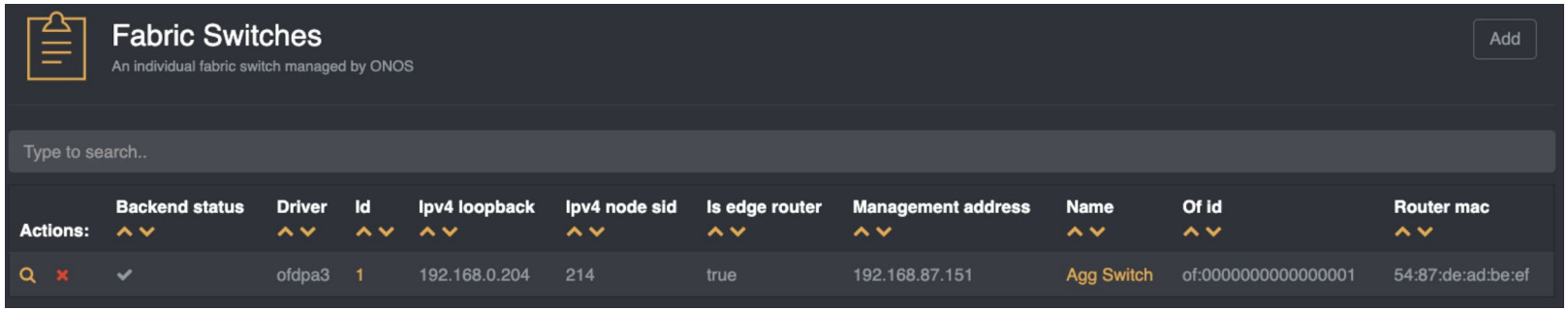
```
port_mode_1=4x10g    # front port 1
```

APPs Needed for Fabric

```
onos> apps -a -s
*  21 org.onosproject.hostprovider          1.13.9   Host Location Provider
*  27 org.onosproject.route-service         1.13.9   Route Service Server
*  43 org.onosproject.lldpprovider          1.13.9   LLDP Link Provider
*  44 org.onosproject.optical-model         1.13.9   Optical Network Model
*  45 org.onosproject.openflow-base         1.13.9   OpenFlow Base Provider
*  46 org.onosproject.openflow              1.13.9   OpenFlow Provider Suite
*  68 org.onosproject.drivers               1.13.9   Default Drivers
* 116 org.onosproject.netcfghostprovider    1.13.9   Network Config Host Provider
* 142 org.onosproject.mcast                 1.13.9   Multicast traffic control
* 146 org.onosproject.segmentrouting        1.13.9   Segment Routing
```

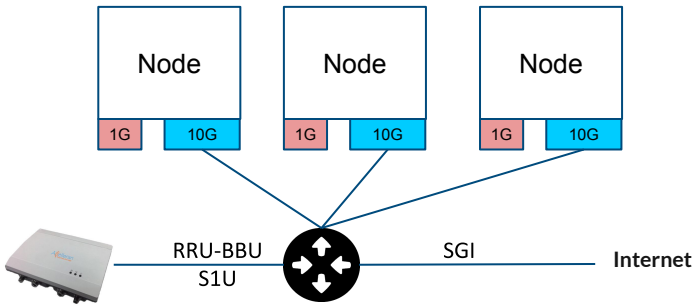
Fabric configuration

- Write your own fabric configuration according to the topo and load it
  - Example: https://gerrit.opencord.org/gitweb?p=pod-configs.git;a=blob;f=tosca-configs/mcord/mcord-local-cluster-fabric-accelleran.yaml;h=ae4c812ebbb6e08a7e9a75c2640282898ace3b46;hb=refs/heads/master
  - curl -H "xos-username: admin@opencord.org" -H "xos-password: letmein" -X POST --data-binary @mcord-local-cluster-fabric-accelleran.yaml http://192.168.87.151:30007/run



Fabric Configuration

- S1U subnet: 119.0.0./24
  RRU← →SPGWU container
  Node← →Node
- RRU-BBU & Node subnet: 116.0.0./24
  RRU← → BBU container
- SGI Subnet: 13.1.1./24
  SPGWU ← → CDN or Internet GW.




```
onos> netcfg
{
  "devices" : {
    "of:0000000000000001" : {
      "segmentrouting" : {
        "ipv4NodeSid" : 214,
        "name" : "Agg Switch",
        "ipv4Loopback" : "192.168.0.204",
        "adjacencySids" : [ ],
        "isEdgeRouter" : true,
        "routerMac" : "54:87:de:ad:be:ef"
      },
      "basic" : {
        "driver" : "ofdpa3",
        "name" : "Agg Switch"
      }
    }
  },
  "apps" : {
    "org.onosproject.provider.lldp" : {
      "suppression" : {
        "deviceTypes" : [ "ROADM", "OTN", "FIBER_SWITCH", "OPTICAL_AMPLIFIER" ],
        "annotation" : "{\"no-lldp\":null}"
      }
    }
  },
  "ports" : {
    "of:0000000000000001/34" : {
      "hostLearning" : {
        "enabled" : true
      },
      "interfaces" : [ {
        "ips" : [ "13.1.1.254/24", "119.0.0.254/24", "116.0.0.254/24" ],
        "vlan-untagged" : 20,
        "name" : "node_2",
        "mac" : "54:87:DE:AD:BE:EF"
      } ]
    },
    "of:0000000000000001/35" : {
      "hostLearning" : {
        "enabled" : true
      },
      "interfaces" : [ {
        "ips" : [ "13.1.1.254/24", "119.0.0.254/24", "116.0.0.254/24" ],
        "vlan-untagged" : 20,
        "name" : "node_3",
        "mac" : "54:87:DE:AD:BE:EF"
      } ]
    },
    "of:0000000000000001/33" : {
      "hostLearning" : {
        "enabled" : true
      },
      "interfaces" : [ {
        "ips" : [ "13.1.1.254/24", "119.0.0.254/24", "116.0.0.254/24" ],
        "vlan-untagged" : 20,
        "name" : "node_1",
        "mac" : "54:87:DE:AD:BE:EF"
      } ]
    },
    "of:0000000000000001/37" : {
      "hostLearning" : {
        "enabled" : true
      },
      "interfaces" : [ {
        "ips" : [ "13.1.1.254/24", "119.0.0.254/24", "116.0.0.254/24" ],
        "vlan-untagged" : 20,
        "name" : "rru",
        "mac" : "54:87:DE:AD:BE:EF"
      } ]
    }
  },
```

Contents

- Part 1: “Multi-cluster Physical Setup” Pingping
- **Part 2: “K8S & Helm, OMEC” Hyunsun**
- Part 3: “Subscriber Monitoring” Badhrinath
- Part 4: “Monitoring & Visualization” Doyoung
- Part 5: “CDN & XOS in COMAC” Woojoong

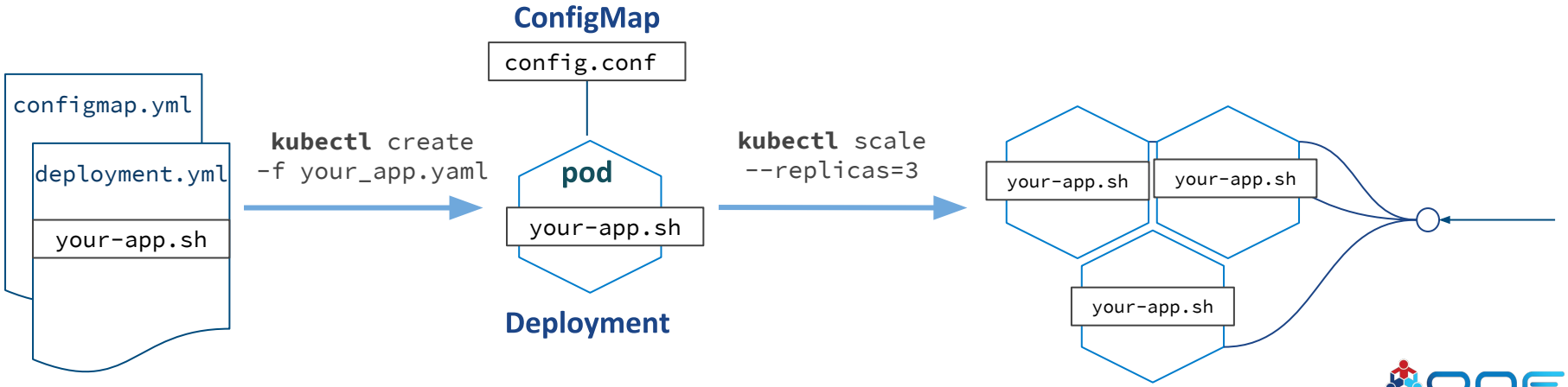
Kubernetes & Helm, OMEC

Hyunsun Moon, ONF

Kubernetes & Helm

Kubernetes

- Most popular open-source container-orchestration system
- Just define your application as K8S resources like Deployment, ConfigMap, Service, and so on
- Helps automating deployment, management, and scaling



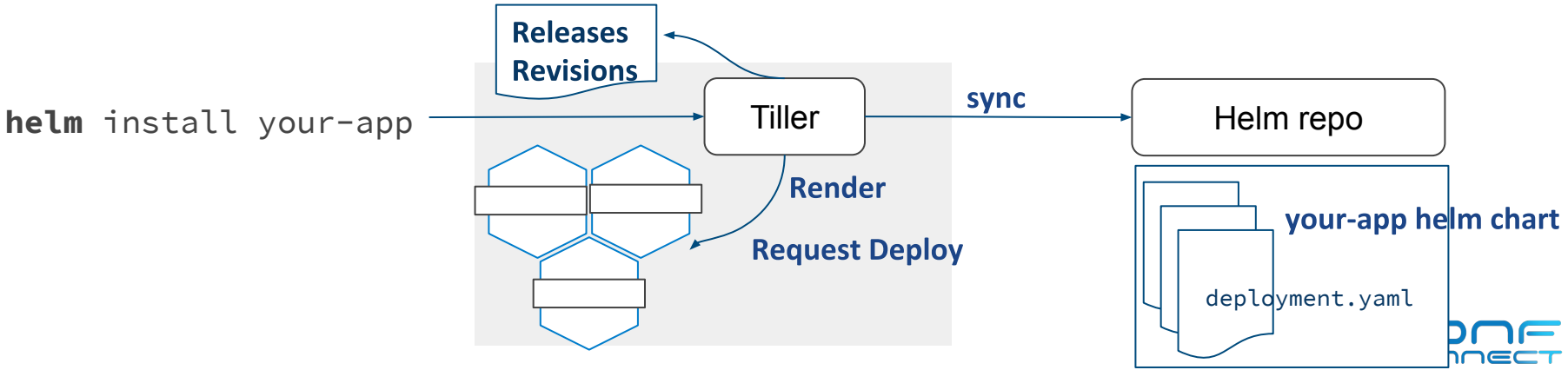
Helm and Helm Charts

Helm Charts

- Collection of files that describes related set of K8S resources
- Stored locally of fetched from remote chart repository

Helm

- Renders Helm Charts and requests deployment to K8S
- Composed of Tiller server and Helm client tool



COMAC Helm Charts

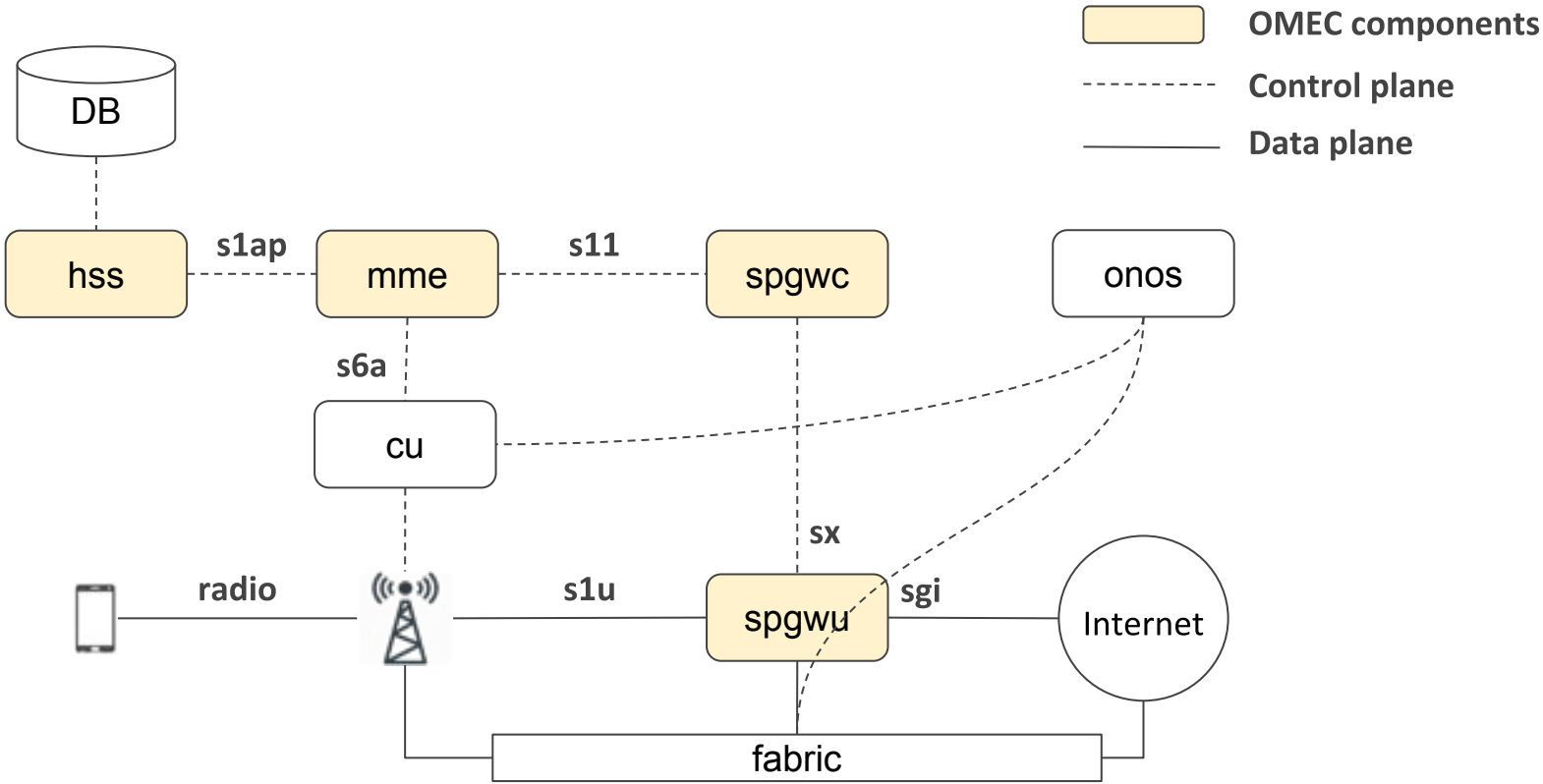
- cord-platform
  - kafka
  - onos
  - xos-core
  - logging
  - nem-monitoring
- comac-platform
  - base-kubernetes
  - mcord-profile

- omec-control-plane
- omec-data-plane
- cdn-local
- cdn-remote

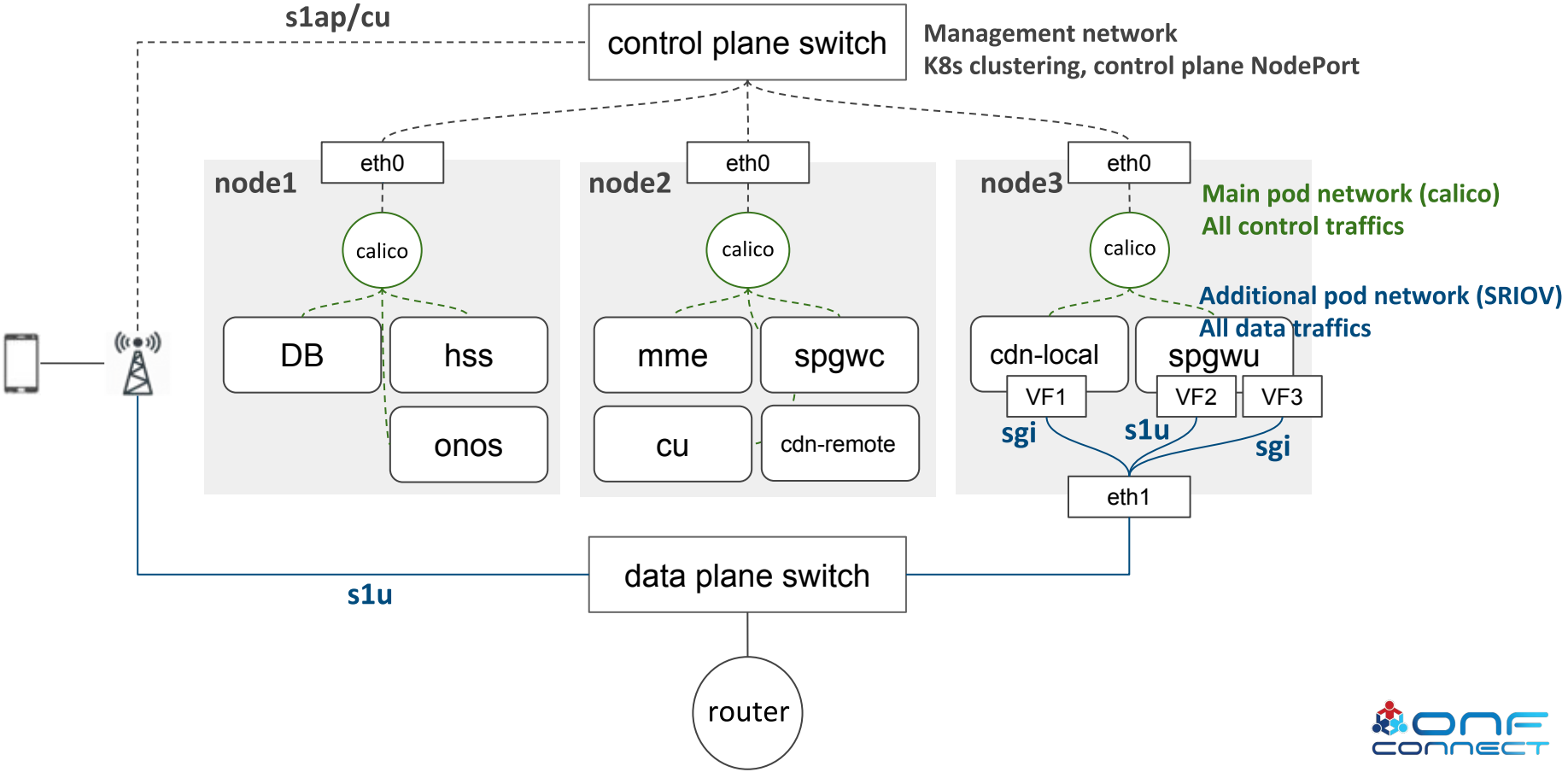
https://github.com/opencord/helm-charts

OMEC implementation on K8S

OMEC architecture



OMEC on K8S


s1ap/cu
control plane switch
Management network
K8s clustering, control plane NodePort
node1
eth0
calico
DB
hss
onos
node2
eth0
calico
mme
spgwc
cu
cdn-remote
node3
eth0
calico
Main pod network (calico)
All control traffics
Additional pod network (SRIOV)
All data traffics
cdn-local
spgwu
VF1
VF2
VF3
sgi
s1u
sgi
eth1
s1u
data plane switch
router

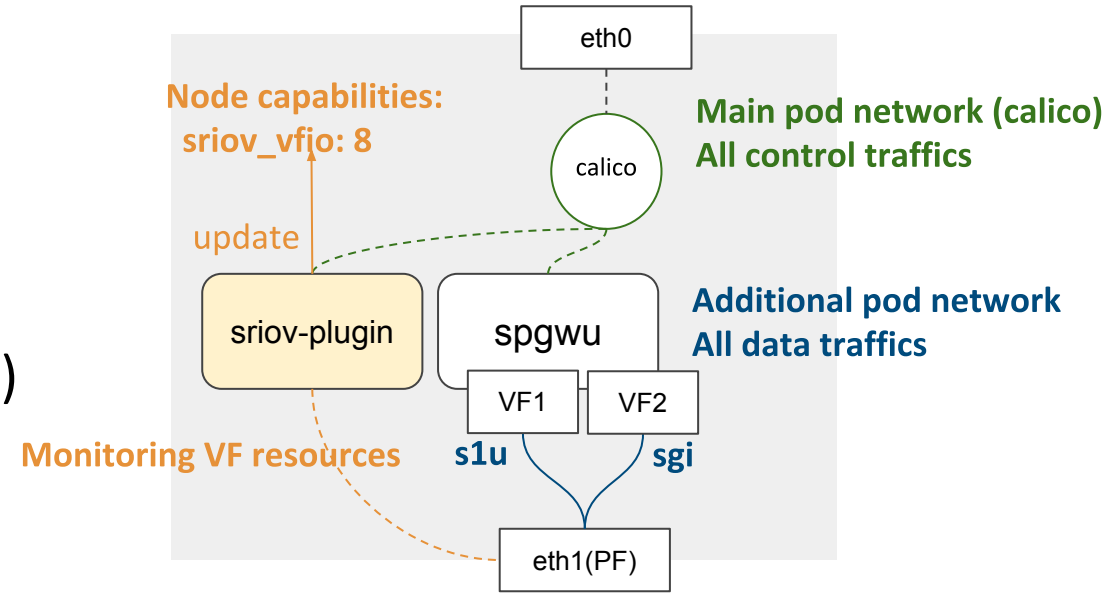
OMEC Data Plane Implementation

Network Attach Definitions

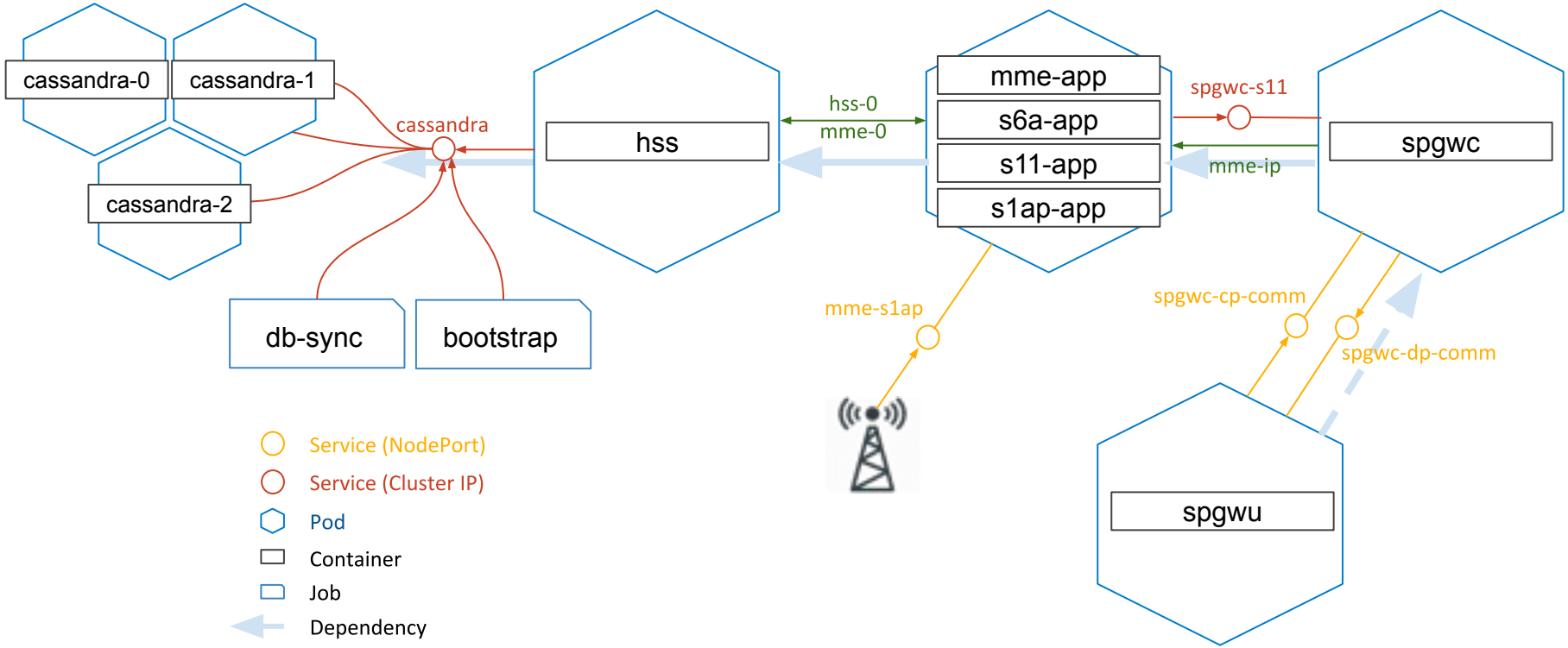
- (main pod network)
- sgi-net
- s1u-net

CNI plugins

- multus (meta-plugin)
- calico
- vfioveth



OMEC implementation in COMAC



Contents

- Part 1: “Multi-cluster Physical Setup” Pingping
- Part 2: “K8S & Helm, OMEC” Hyunsun
- **Part 3: “Subscriber Monitoring” Badhrinath**
- Part 4: “Monitoring & Visualization” Doyoung
- Part 5: “CDN & XOS in COMAC” Woojoong

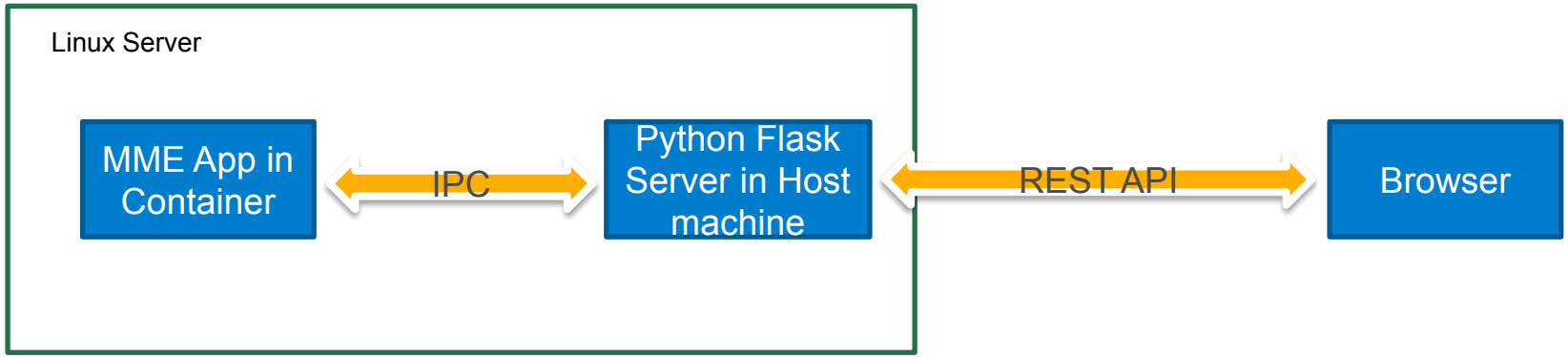
Subscriber Monitoring Tool

Badhrinath Padmanabhan
ONF

Tool Use Case

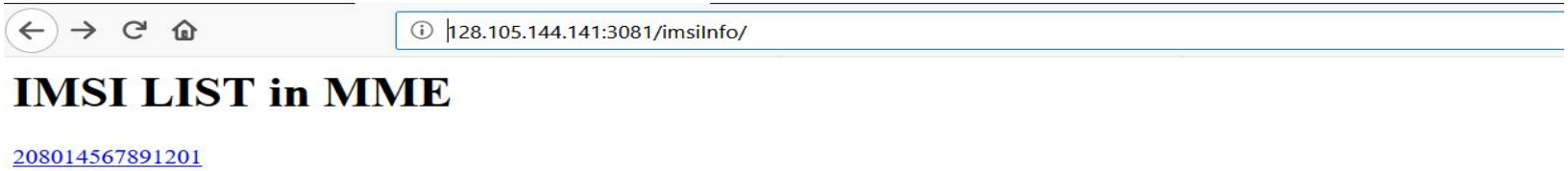
- This tool is required to query the EPC components for Subscriber Information based on various keys like IMSI/IMEI/TAC/ENB Id etc.
- This Information received would include info like QOS Info, TAI Info, Teid Info, data flow information per bearer and all user context information that is stored within the nodes.
- The query will be done using REST based APIs.
- This information would help operators to make better decisions on configuring the network.
- The Information will be categorized as per user login and authorization.

Design



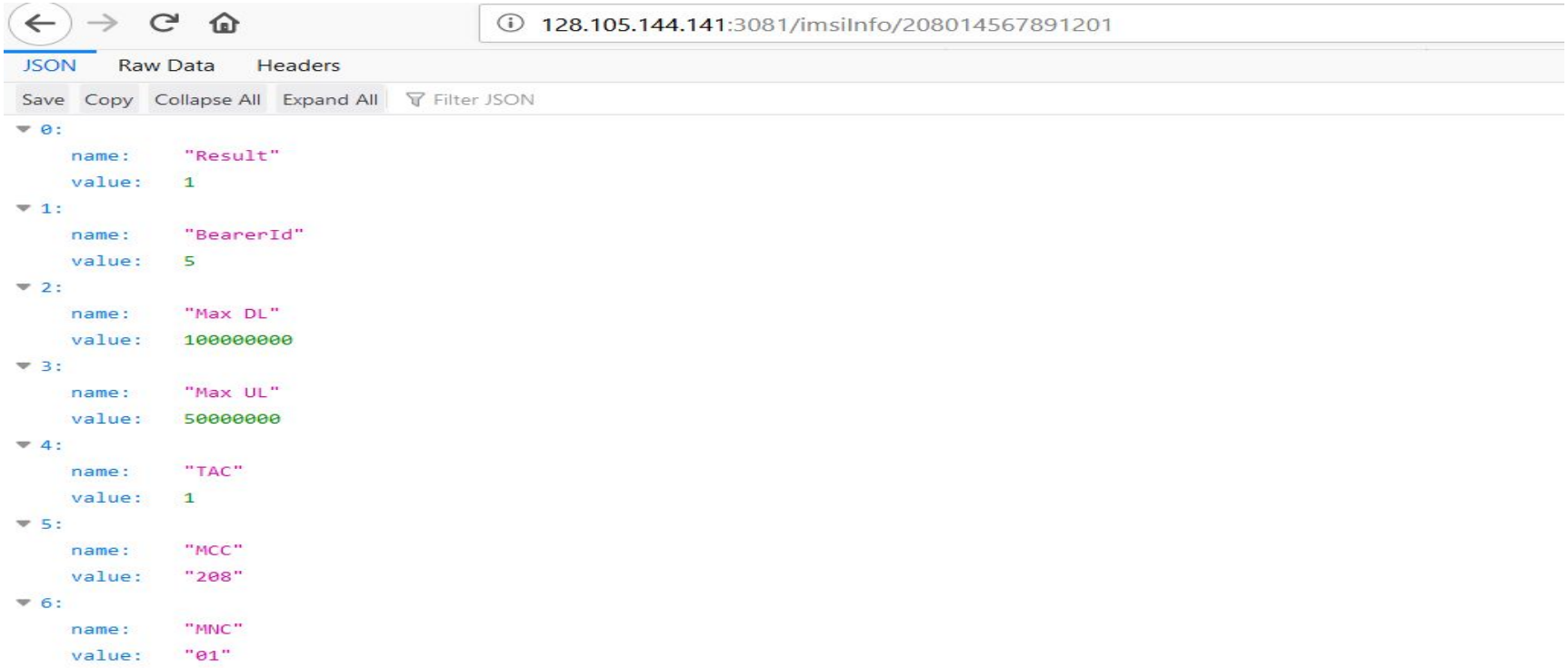
- This tool will query the MME to get the Subscriber list.
- The Tool is written in Python and uses Python Flask to create the HTTP server.
- The MME listens on a UNIX Domain socket. The MME runs on a K8 pod and the socket is created in a mounted drive from host machine.
- The Python tool also acts as a client and queries MME for various Subscriber Info.
- In the future this will be updated to fetch information from data store.

Tool Snapshot



The click on the IMSI gives the various parameters of the UE context.

Tool Snapshot



Contents

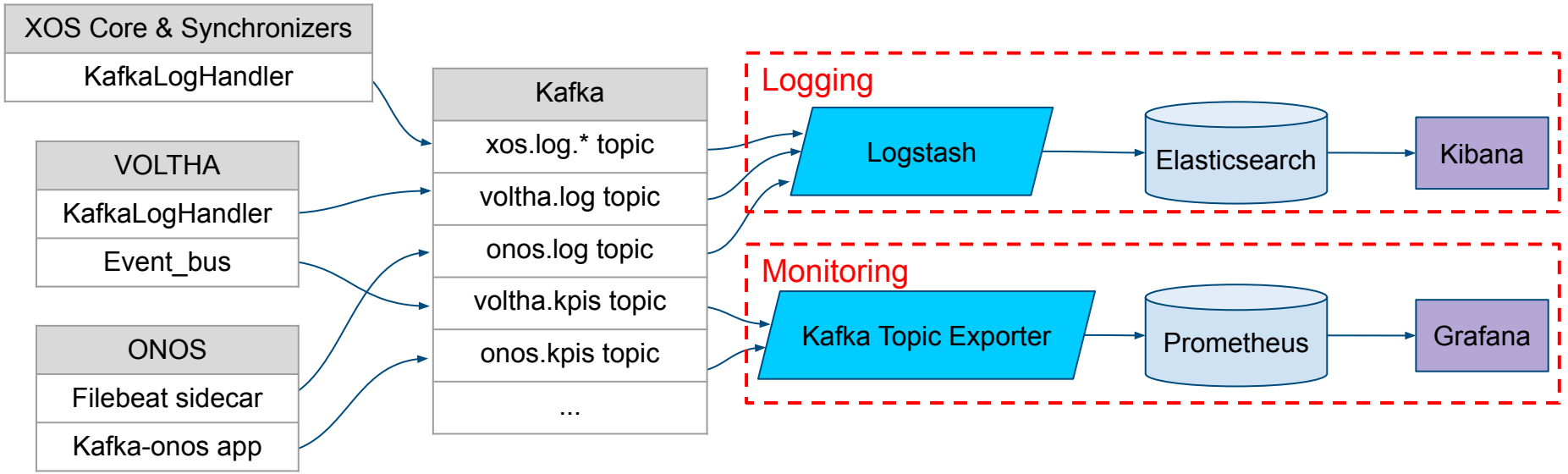
- Part 1: “Multi-cluster Physical Setup” Pingping
- Part 2: “K8S & Helm, OMEC” Hyunsun
- Part 3: “Subscriber Monitoring” Badhrinath
- **Part 4: “Monitoring & Visualization” Doyoung**
- Part 5: “CDN & XOS in COMAC” Woojoong

Monitoring and Visualization
(Grafana-via-Prometheus)

Doyoung Lee
ONF/POSTECH

Monitoring and Visualization (1/3)

- Consuming events/metrics/logs posted to Kafka
- Metrics: Collected by Prometheus and viewable using Grafana



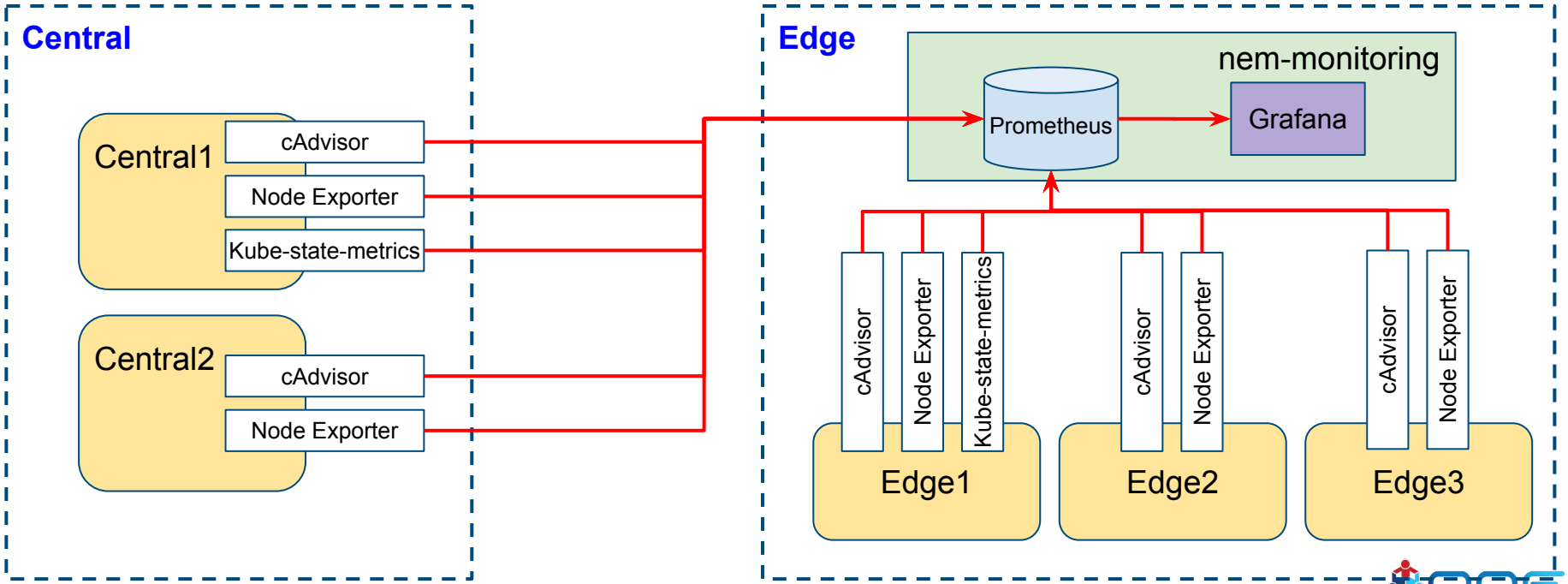
Monitoring and Visualization (2/3)

- Monitoring helm-chart: *nem-monitoring*
  - Grafana
  - Prometheus including node-exporter, kube-state-metrics

- Deployed on edge cluster by default
  - Part of the *cord-platform* helm-charts
  - Monitor all nodes and pods deployed on edge cluster
  - Optionally, pulling metrics from central cluster
    - cAdvisor for resource usage of running containers
    - Kube-state-metrics for metrics of Kubernetes objects

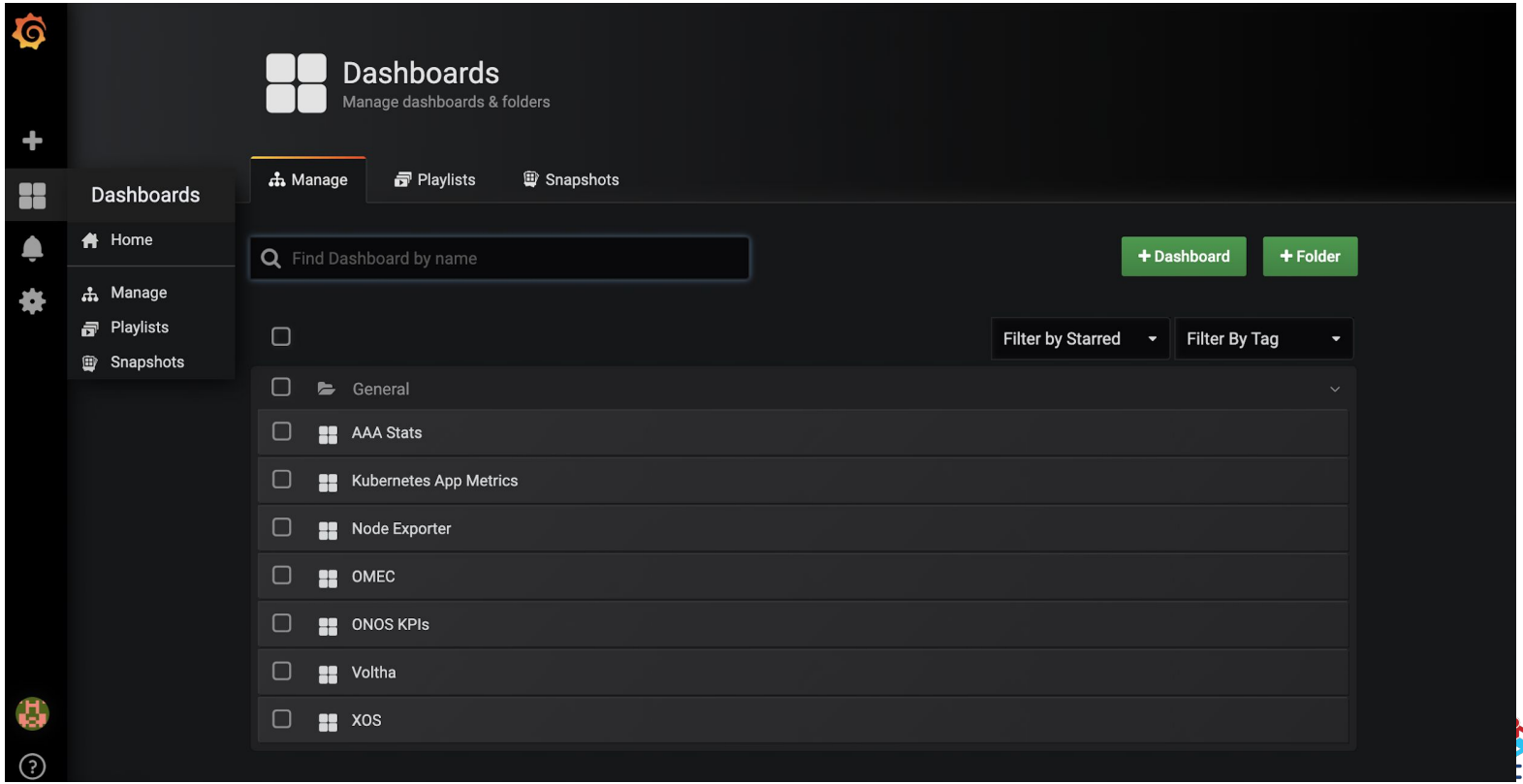
Monitoring and Visualization (3/3)

- Add central cluster into monitoring target



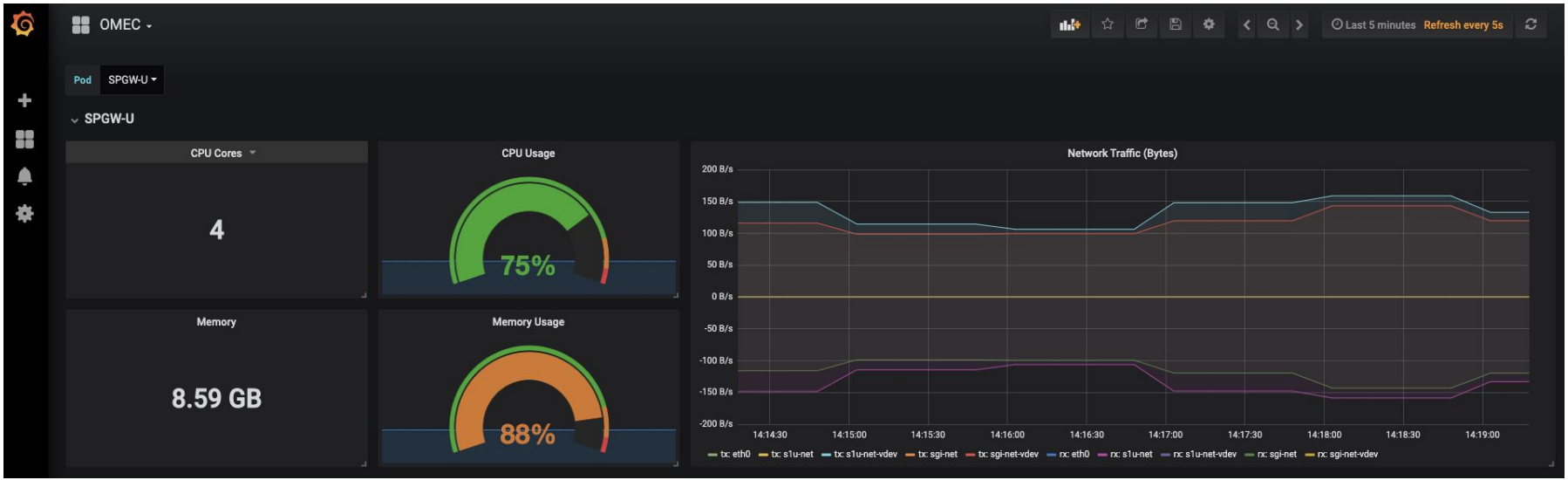
Grafana Dashboard Example (1/3)

- 7 default dashboards provided by *nem-monitoring*



Grafana Dashboard Example (2/3)

- OMEC Dashboard
  - Requested resources and current usage (CPU cores and memory)
  - Ingress/Egress traffics



Grafana Dashboard Example (3/3)

- Node Exporter Dashboard

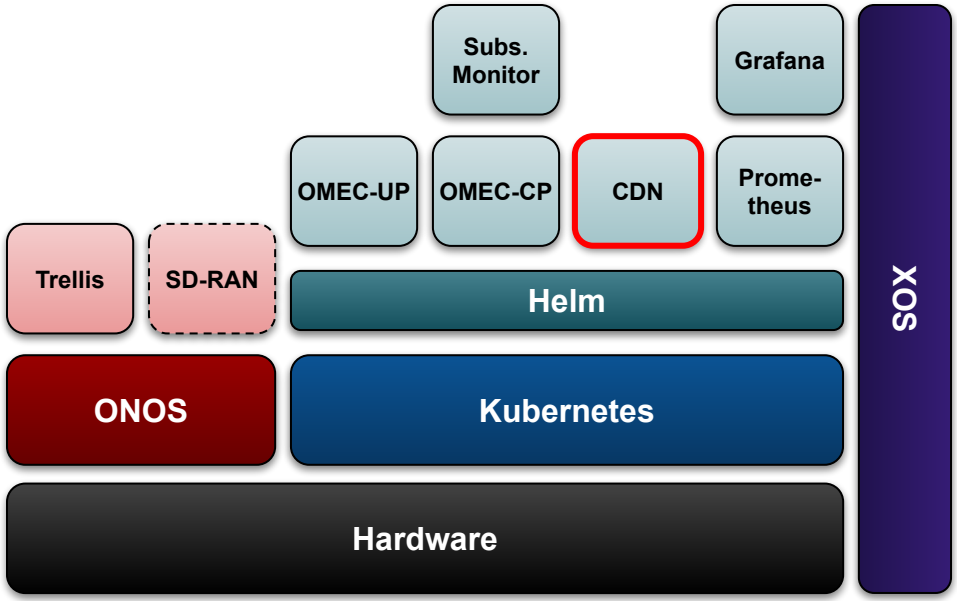
Contents

- Part 1: “Multi-cluster Physical Setup” Pingping
- Part 2: “K8S & Helm, OMEC” Hyunsun
- Part 3: “Subscriber Monitoring” Badhrinath
- Part 4: “Monitoring & Visualization” Doyoung
- **Part 5: “CDN & XOS in COMAC” Woojoong**

CDN

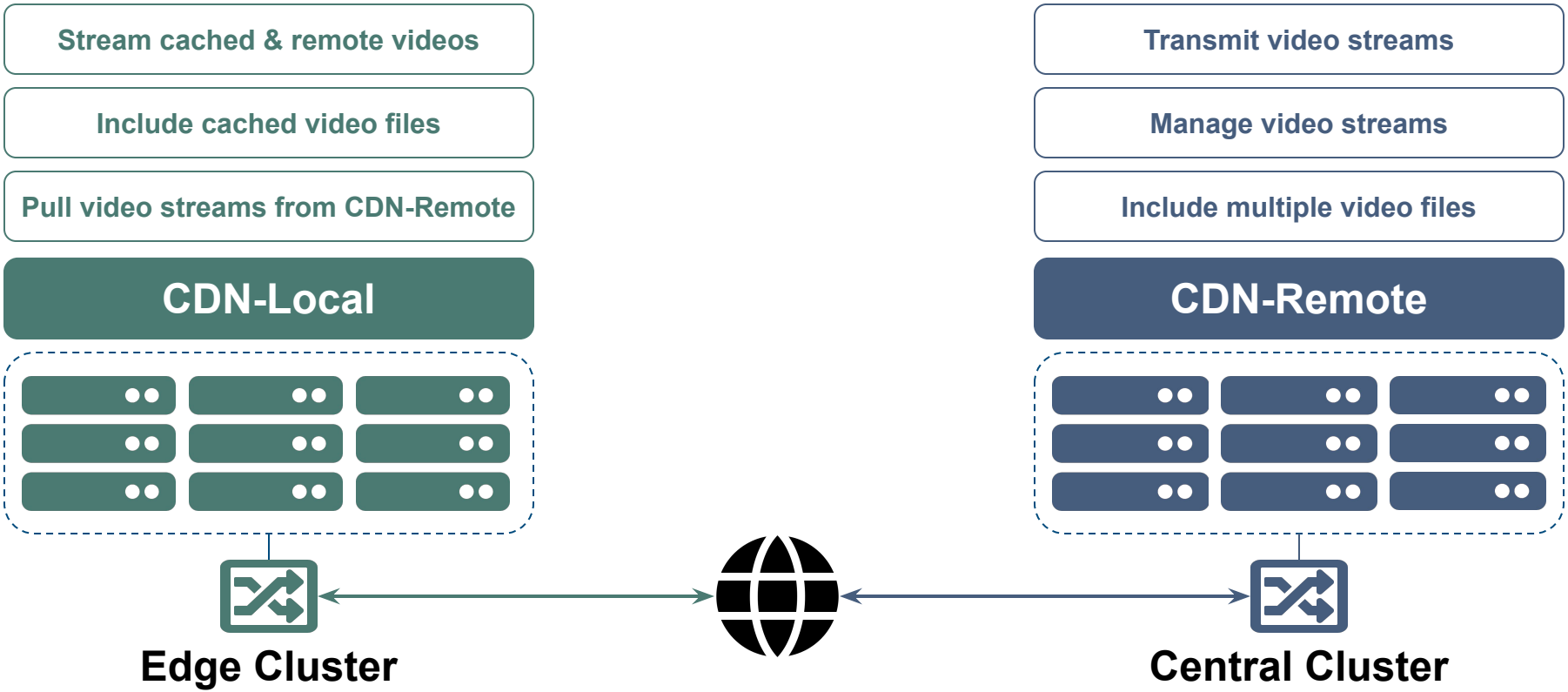
Woojoong Kim
Open Networking Foundation

CDN: an Overview



- Purpose
  - To support video streaming in multi-cluster environment
  - Open-source software
- Design
  - CDN-Remote
    - Ant media server
    - ffMPEG container
  - CDN-Local
    - NGINX

CDN-Local & CDN-Remote



CDN-Local & CDN-Remote

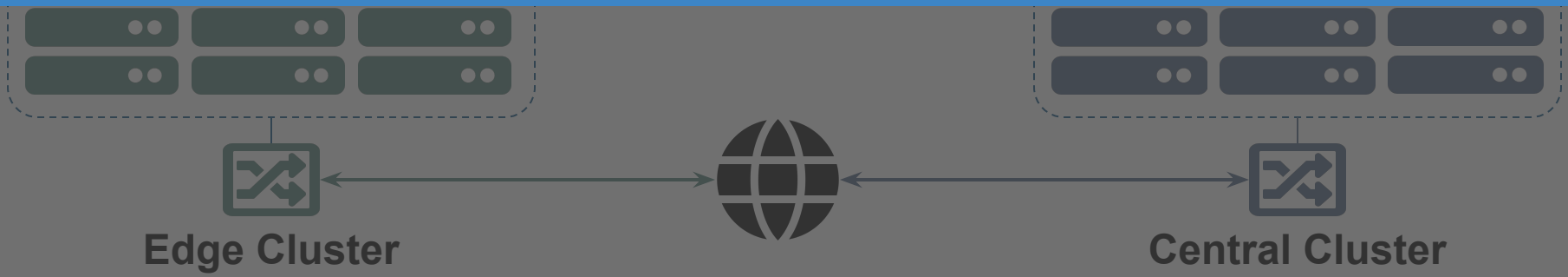
Stream cached & remote videos

Include cached video files

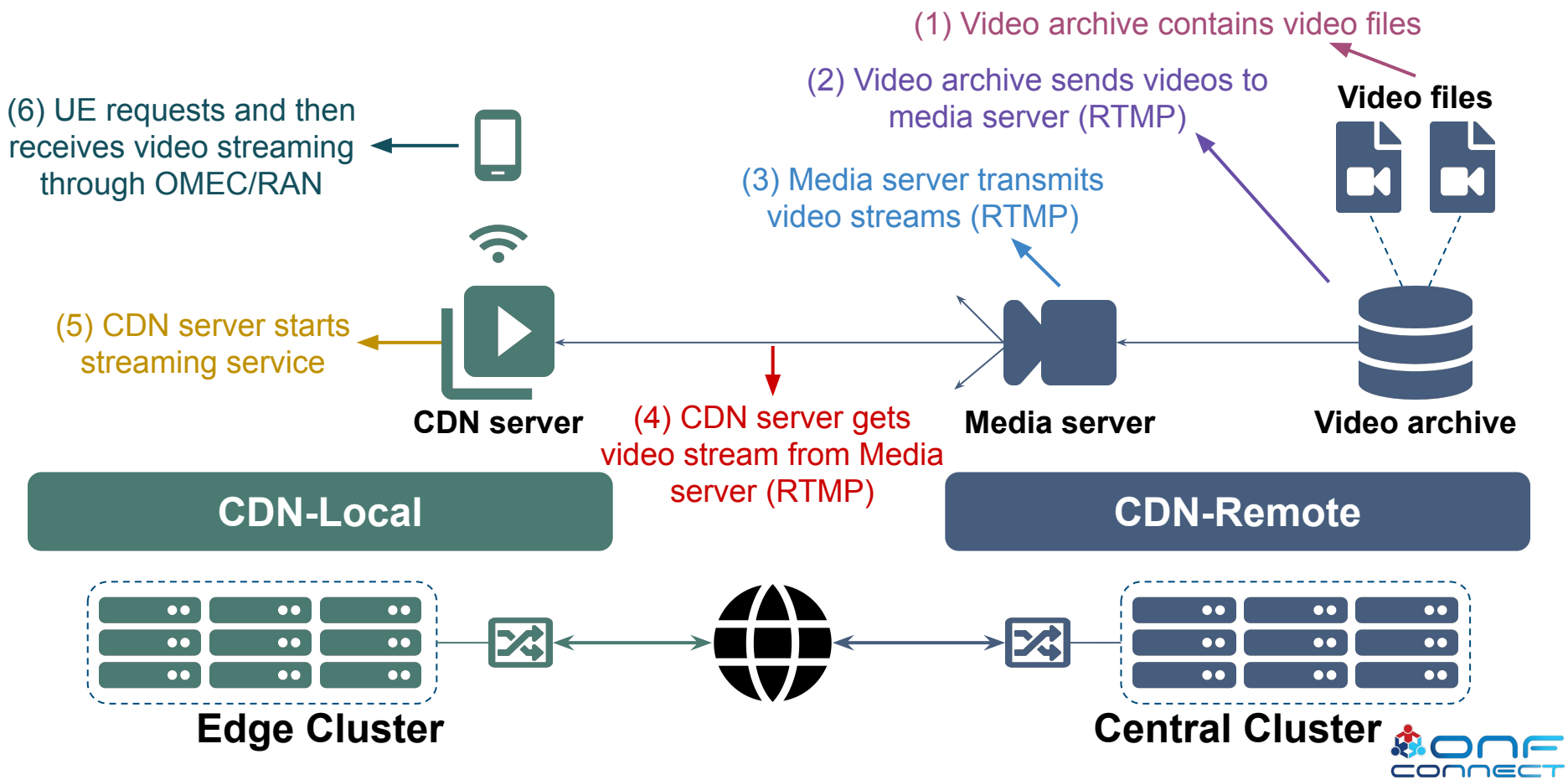
Transmit video streams

Manage video streams

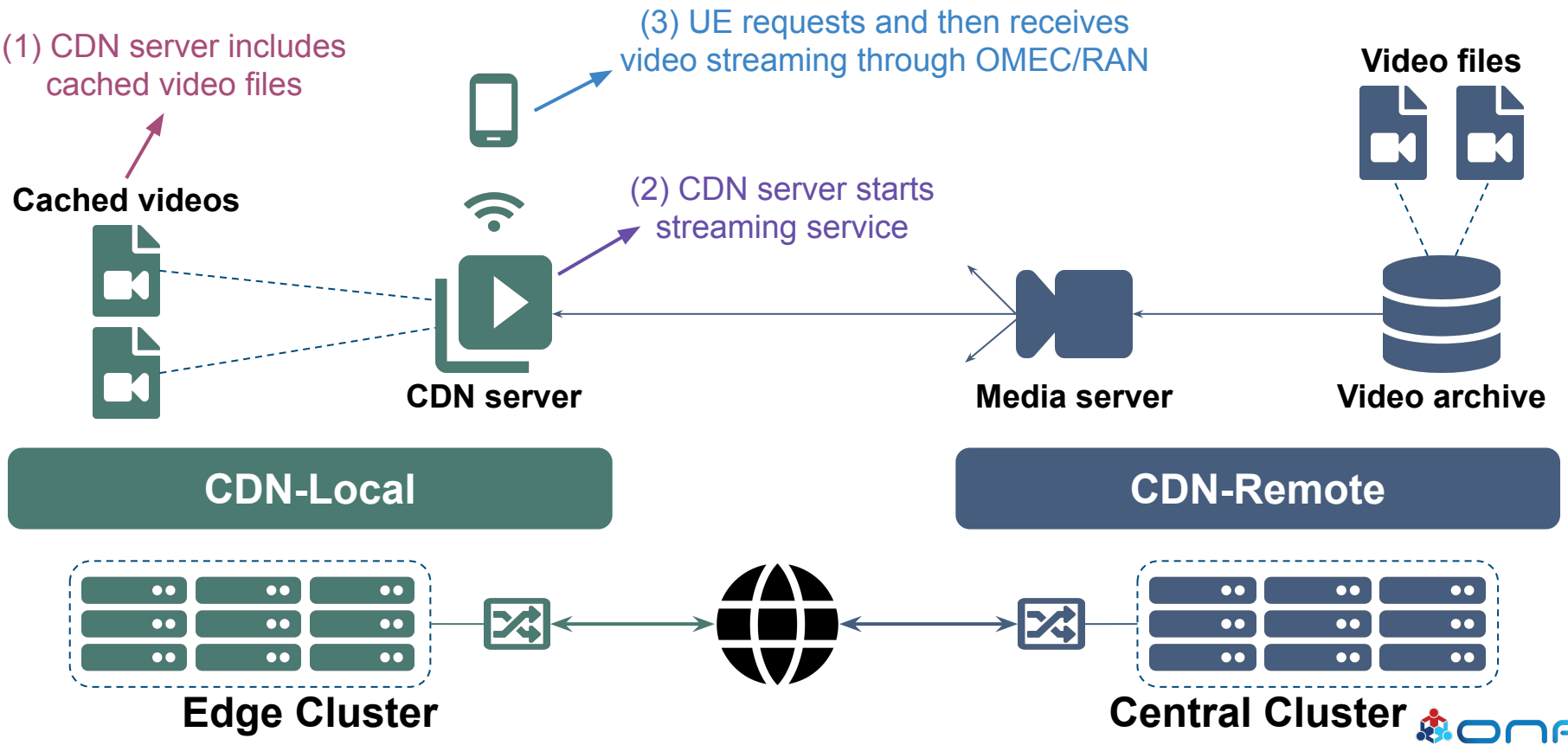
This is fully “Open-Source CDN”



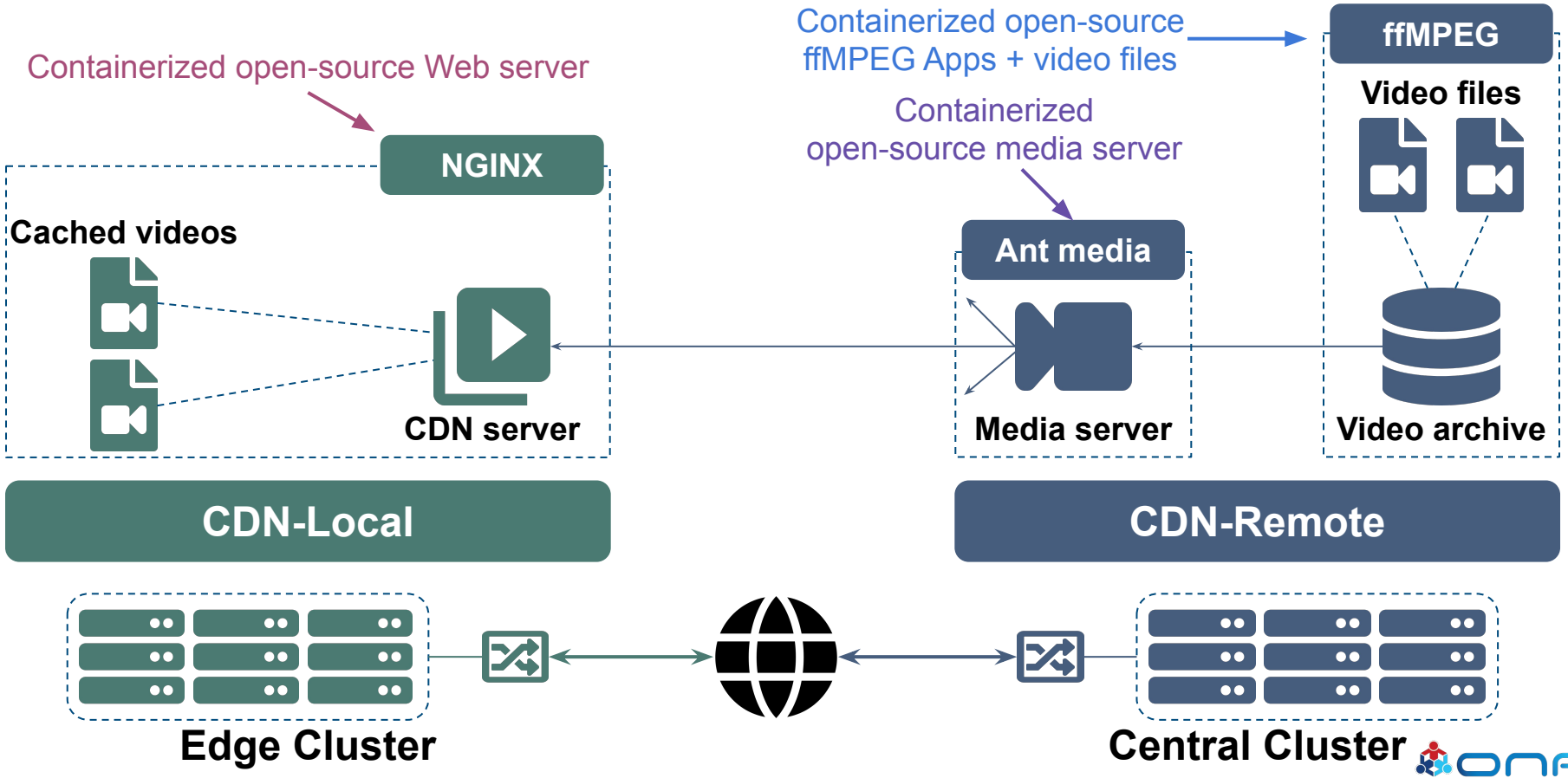
CDN-Remote: “Remote” Video Files



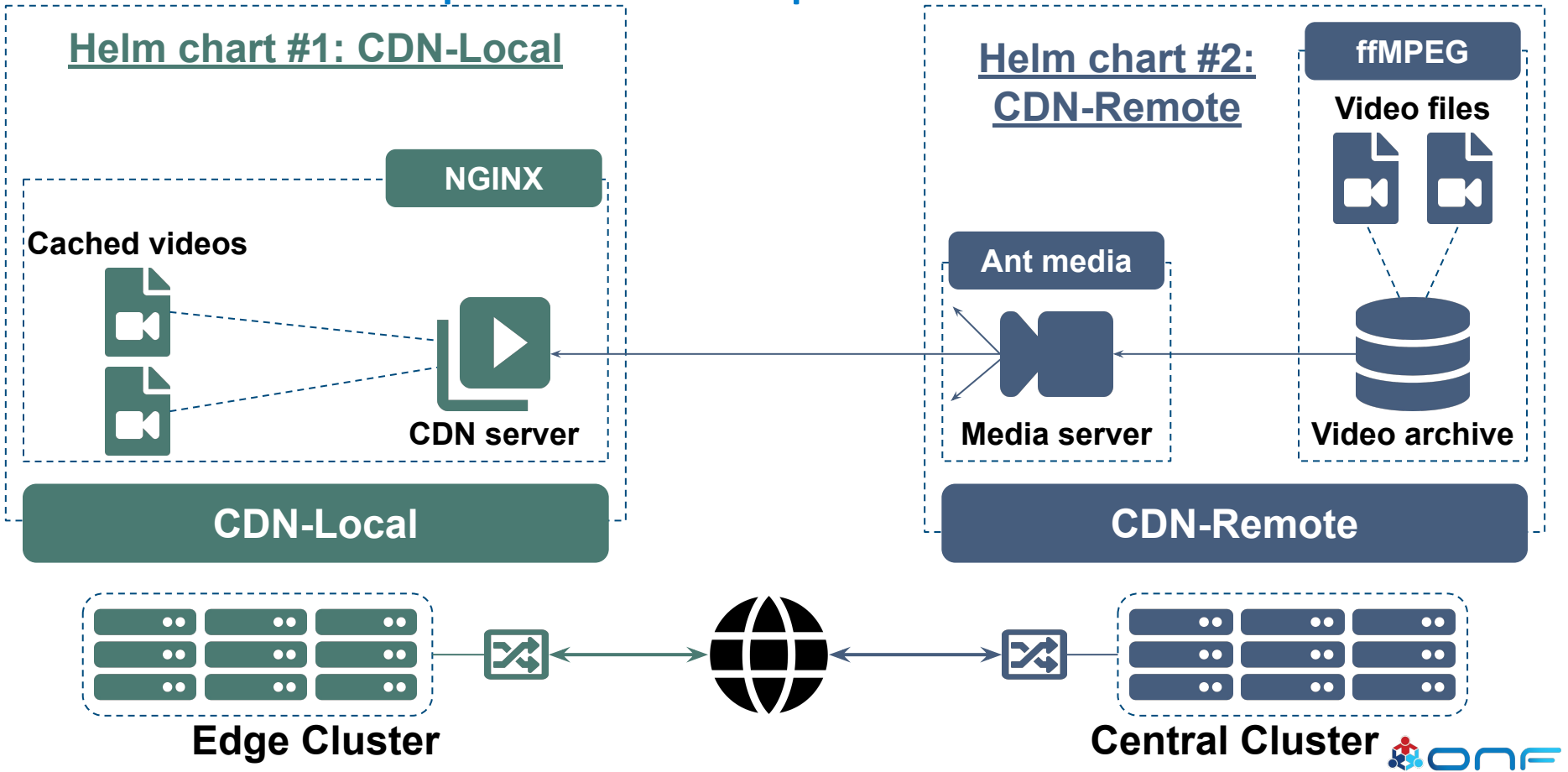
CDN-Local: “Cached” Video Files



Components in Open-Source CDN



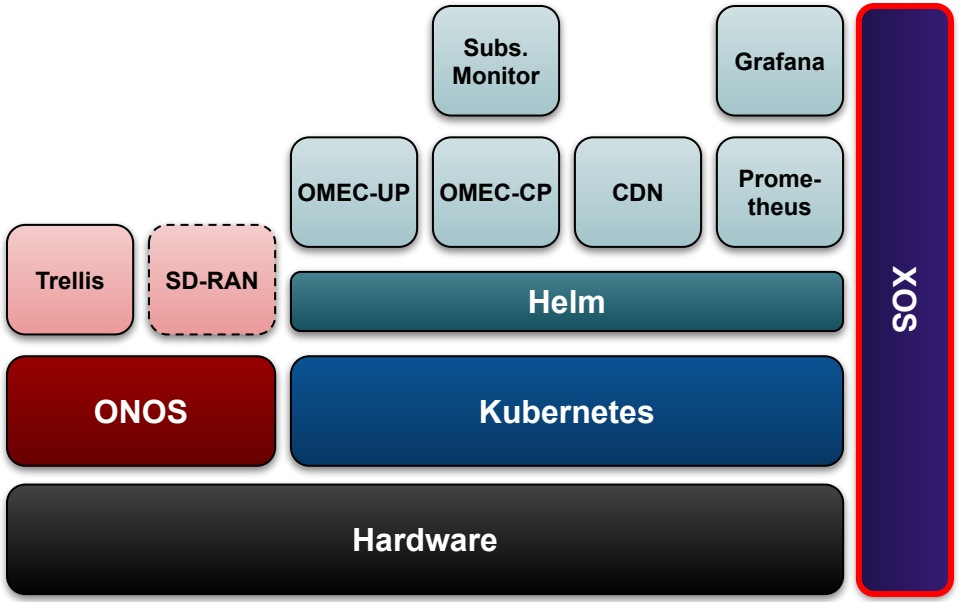
Components in Open-Source CDN



XOS

Woojoong Kim
Open Networking Foundation

XOS: an Overview



- Purpose - Orchestration
  - Define COMAC services
  - To monitor K8s PODs
  - To configure fabric networks
  - To configure RAN

XOS Workflow in COMAC

TOSCA files

- Profile definition
- Fabric definition
- UE definition
- RAN slice definition

XOS

- XOS-TOSCA
- XOS services
- Service chain
- Configuration UI

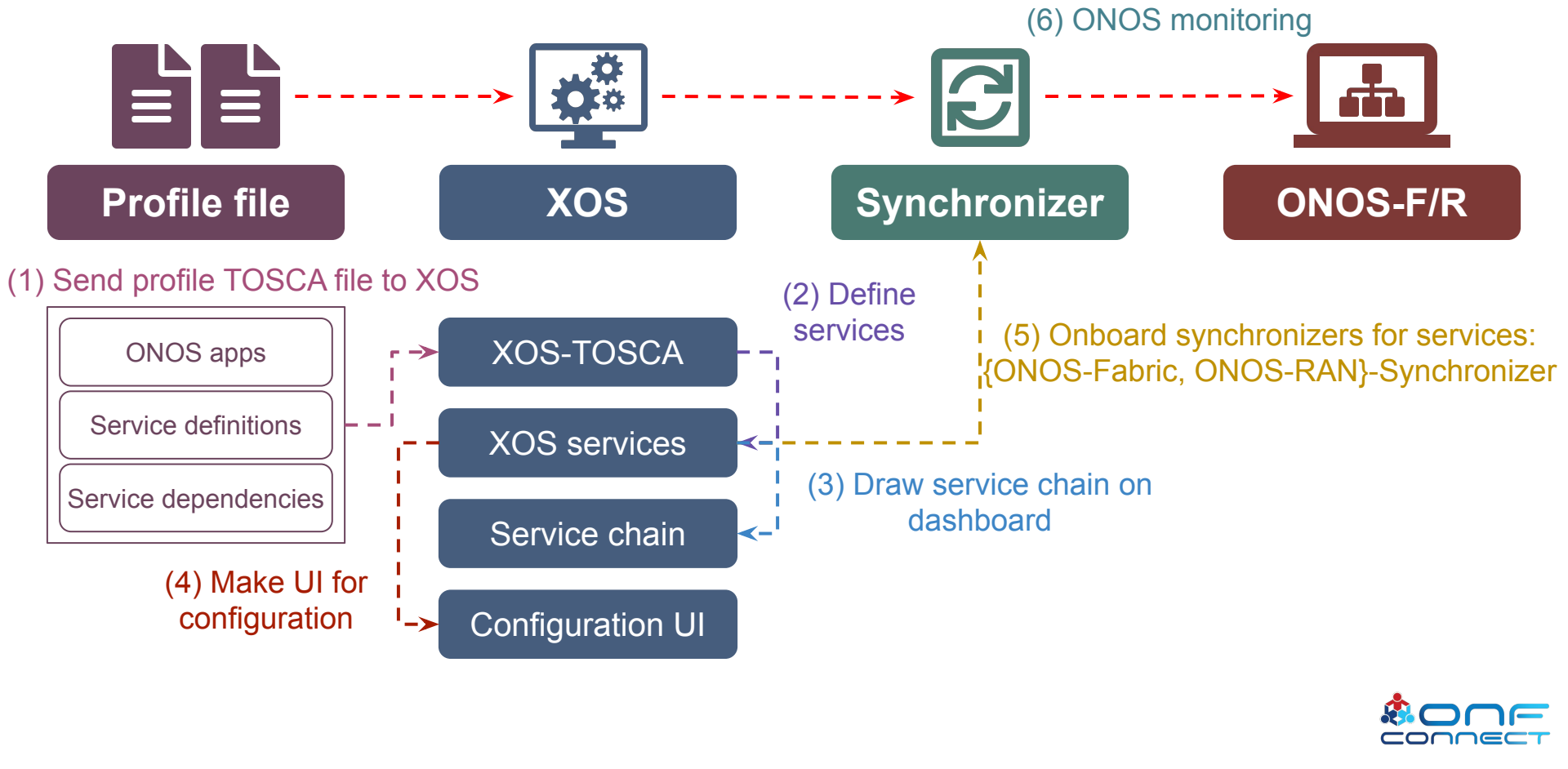
Synchronizer

- Monitoring
- Configuration

ONOS-F/R

- Fabric monitoring
- Fabric configuration
- RAN monitoring
- RAN configuration

XOS Workflow in COMAC: Profile

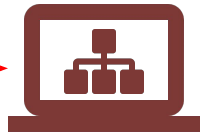


XOS Workflow in COMAC: Fabric

Fabric file

XOS

Synchronizer



ONOS-Fabric

(1) Send Fabric TOSCA file to XOS

- Node configuration
- Switch configuration
- Port configuration

(2) Save config values

XOS-TOSCA

XOS services

(3) Send config values

Configuration UI

(6) ONOS monitoring

Monitoring

Configuration

(5) Fabric monitoring

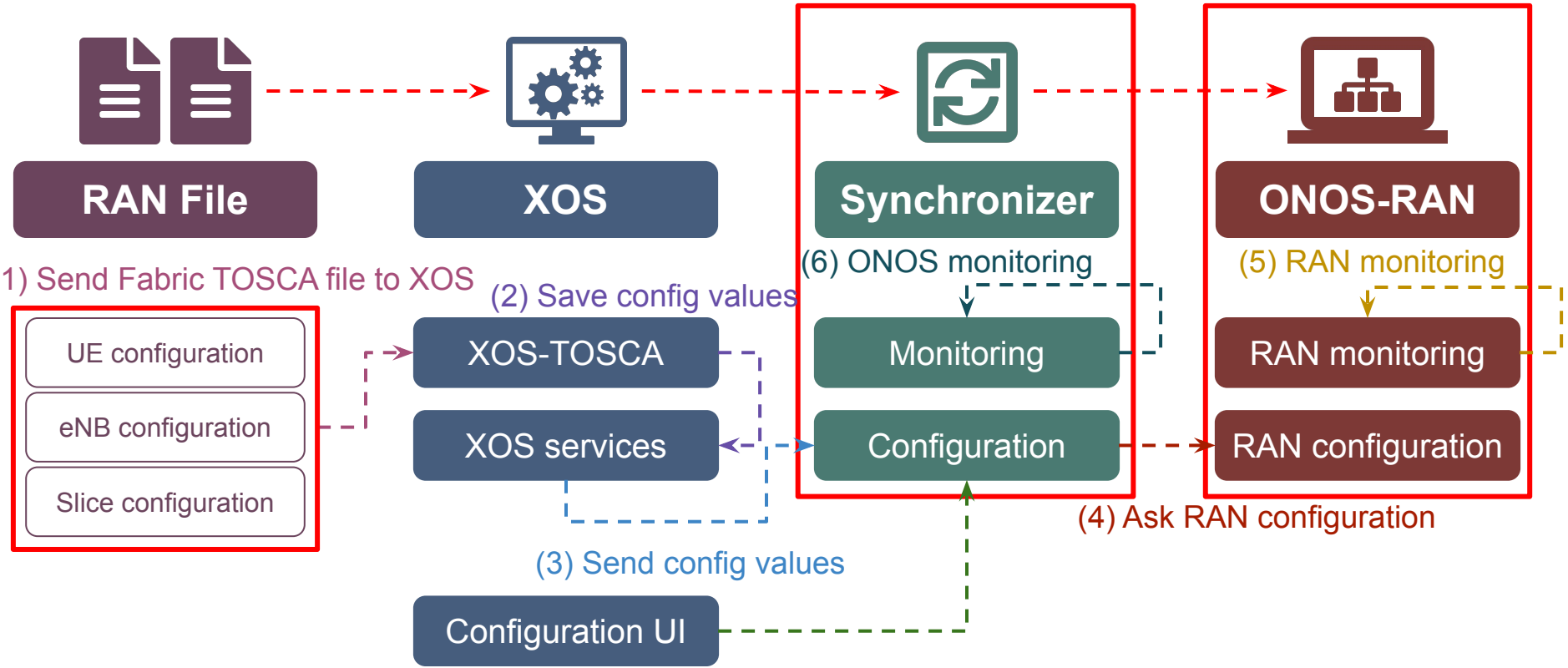
Fabric monitoring

Fabric configuration

(4) Ask fabric configuration

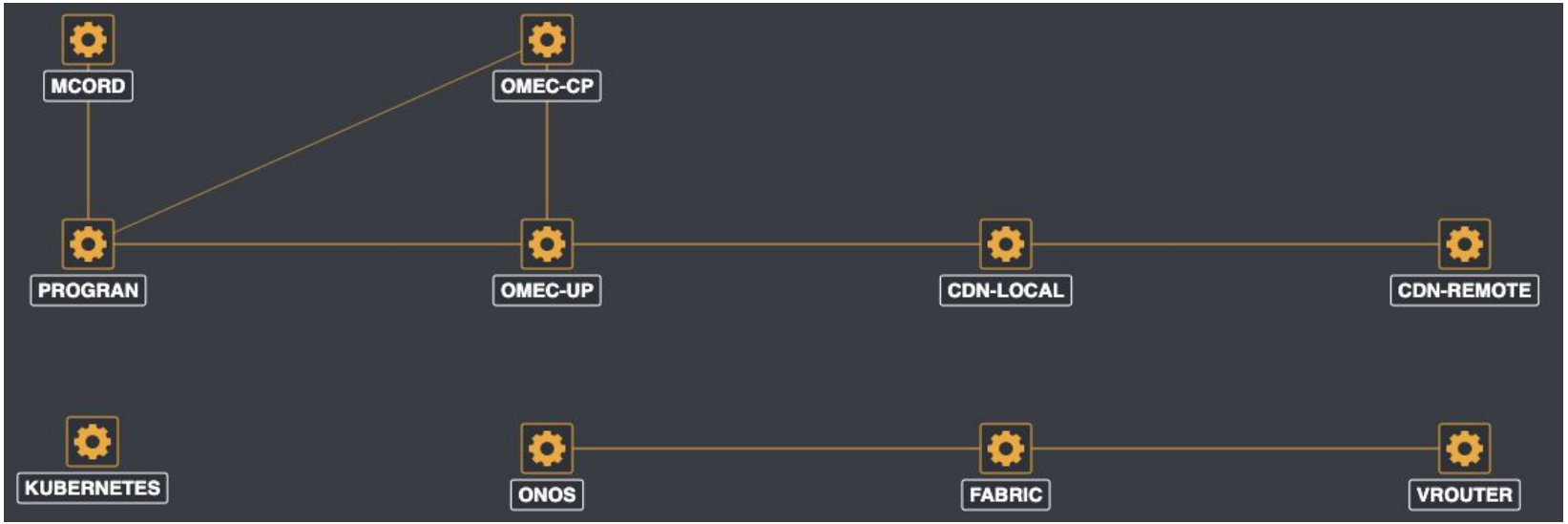
(7) W/O TOSCA, admin can configure manually through UI

XOS Workflow in COMAC: RAN Slicing



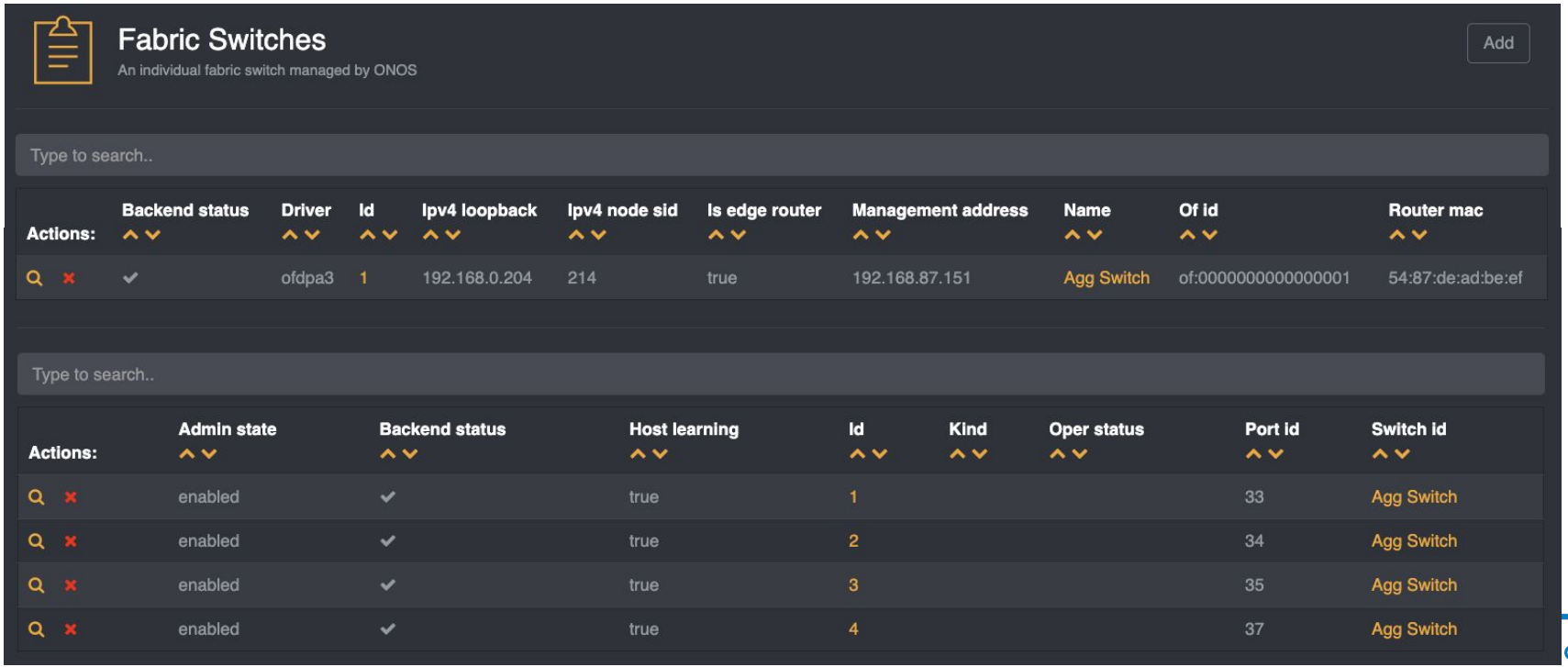
Example: Service Chain on Dashboard

An example of COMAC service graph



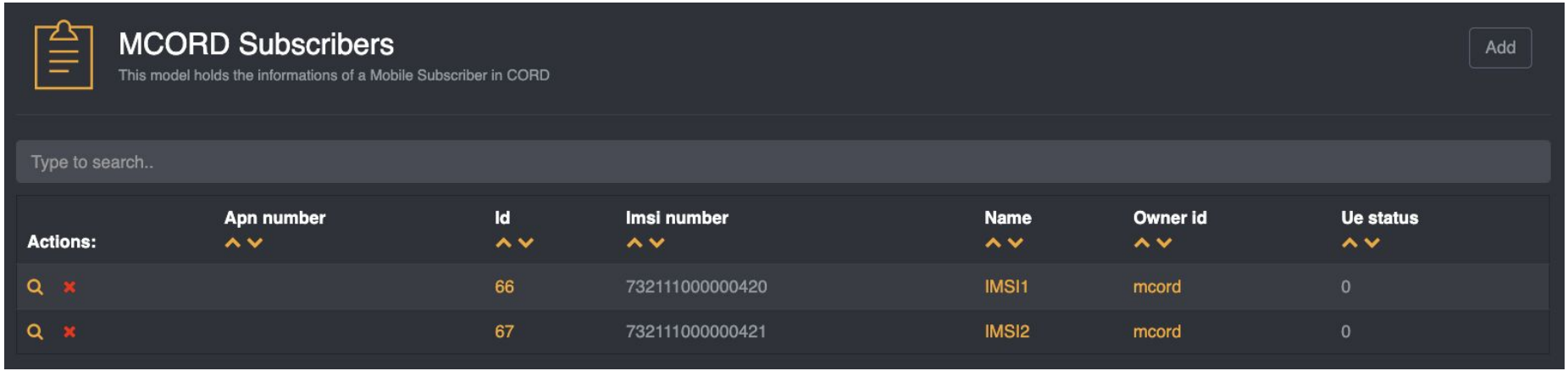
Example: Fabric Configuration

An example of fabric switch configuration in XOS



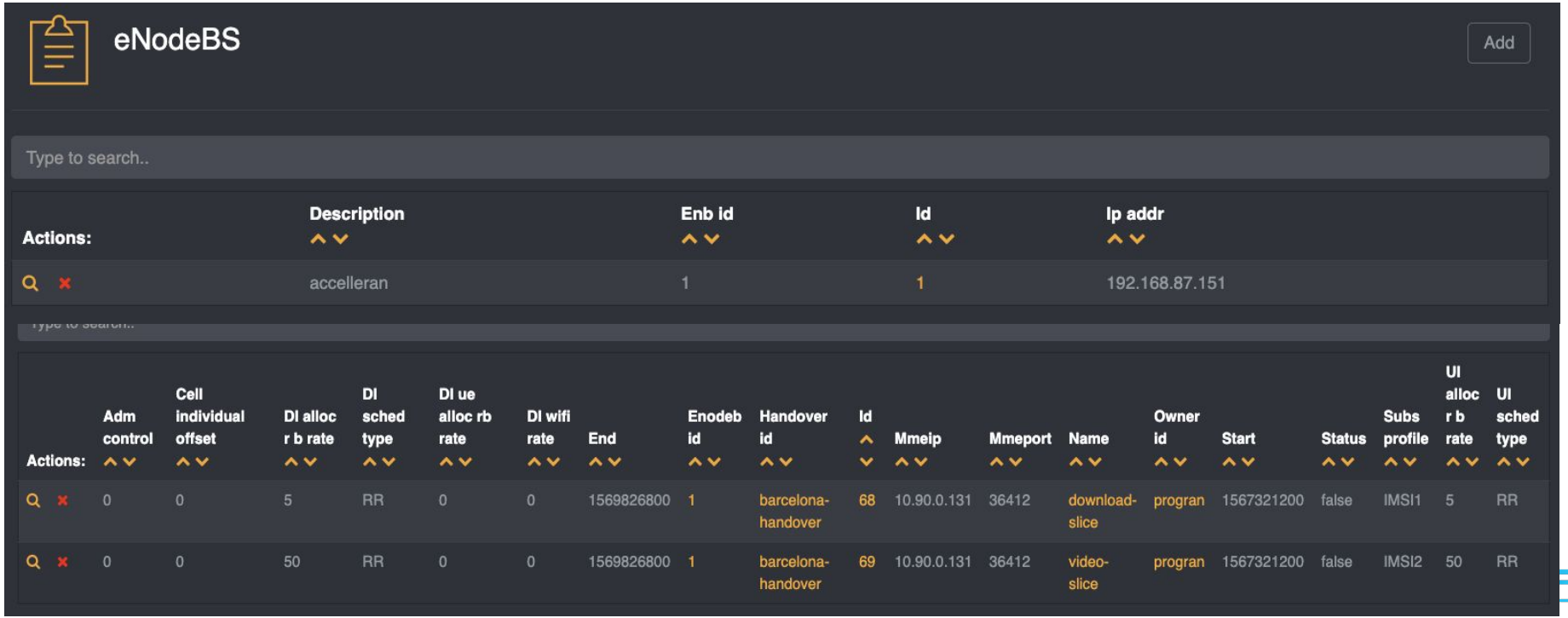
Example: UE Configuration in RAN

An example of UE/IMSI configuration in XOS



Example: RAN Slicing

An example of RAN configuration in XOS



Thank You

Follow Up Links:

https://guide.opencord.org