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

Edge Cluster in ONF-Connect Conference

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

Central 1
- K8S
- Calic
- RAN-CU
- HSS

Central 2
- K8S
- Calic
- MME
- SPGW-C

Central 3
- K8S
- Calic
- HSS/DB
- CDN-Remote

COMAC Fabric

L2 Management Network

Edge Cluster

Router
- VPN

COMAC Management Network & Virtual COMAC 3GPP CP Network
Trellis in COMAC

- **Trellis wiki:** [https://wiki.opencord.org/display/CORD/Trellis+Underlay+Fabric](https://wiki.opencord.org/display/CORD/Trellis+Underlay+Fabric)

- **One time manual work:**
  1. Install Open Networking Linux (ONL)
  2. Install OF-DPA package
  3. Connect OF Switch Management Port to ONOS

- **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

<table>
<thead>
<tr>
<th>Name</th>
<th>Version</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>org.onosproject.hostprovider</td>
<td>1.13.9</td>
<td>Host Location Provider</td>
</tr>
<tr>
<td>org.onosproject.route-service</td>
<td>1.13.9</td>
<td>Route Service Server</td>
</tr>
<tr>
<td>org.onosproject.lldpprovider</td>
<td>1.13.9</td>
<td>LLDP Link Provider</td>
</tr>
<tr>
<td>org.onosproject.optical-model</td>
<td>1.13.9</td>
<td>Optical Network Model</td>
</tr>
<tr>
<td>org.onosproject.openflow-base</td>
<td>1.13.9</td>
<td>OpenFlow Base Provider</td>
</tr>
<tr>
<td>org.onosproject.openflow</td>
<td>1.13.9</td>
<td>OpenFlow Provider Suite</td>
</tr>
<tr>
<td>org.onosproject.drivers</td>
<td>1.13.9</td>
<td>Default Drivers</td>
</tr>
<tr>
<td>org.onosproject.netcfghostprovider</td>
<td>1.13.9</td>
<td>Network Config Host Provider</td>
</tr>
<tr>
<td>org.onosproject.mcast</td>
<td>1.13.9</td>
<td>Multicast traffic control</td>
</tr>
<tr>
<td>org.onosproject.segmentrouting</td>
<td>1.13.9</td>
<td>Segment Routing</td>
</tr>
</tbody>
</table>

* 21
* 27
* 43
* 44
* 45
* 46
* 68
* 116
* 142
* 146
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
Fabric Configuration

- **S1U subnet: 119.0.0./24**
  RRU $\rightarrow$ SPGWU container
  Node $\rightarrow$ Node

- **RRU-BBU & Node subnet: 116.0.0./24**
  RRU $\rightarrow$ BBU container

- **SGI Subnet: 13.1.1./24**
  SPGWU $\rightarrow$ CDN or Internet GW.
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 Charts

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

Helm

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

```
helm install your-app
```
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 components

Control plane

Data plane

hss
mme
spgwc
onos

DB

s1ap
s11
s6a
s1u
sx
sgi

radio

Internet

fabric

onos

cu

spgwu
OMECE Data Plane Implementation

Network Attach Definitions
• (main pod network)
• sgi-net
• s1u-net

CNI plugins
• multus (meta-plugin)
• calico
• vfioveth
OMECE implementation in COMAC

cassandra-0  
cassandra-1  
cassandra-2  
db-sync  
bootstrap  
hss  
mme-app  
s6a-app  
s11-app  
s1ap-app  
spgwc  
spgwu  

Service (NodePort)  
Service (Cluster IP)  
Pod  
Container  
Job  
Dependency  
Directly access pod
Contents

• Part 1: “Multi-cluster Physical Setup” Pingping
• Part 2: “K8S & Helm, OMEC” Hyunsun
• Part 3: “Subscriber Monitoring” Badhrinath
• Part 4: “Monitoring & Visualization” Doyoungh
• 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.
• 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.
The click on the IMSI gives the various parameters of the UE context.
```json
{
    0: {
        "name": "Result",
        "value": 1
    },
    1: {
        "name": "BearerId",
        "value": 5
    },
    2: {
        "name": "Max DL",
        "value": 100000000
    },
    3: {
        "name": "Max UL",
        "value": 50000000
    },
    4: {
        "name": "TAC",
        "value": 1
    },
    5: {
        "name": "MCC",
        "value": "208"
    },
    6: {
        "name": "MNC",
        "value": "01"
    }
}
```
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

---

XOS Core & Synchronizers
- KafkaLogHandler

VOLTHA
- KafkaLogHandler
- Event_bus

ONOS
- Filebeat sidecar
- Kafka-onos app

Kafka
- xos.log.* topic
- voltha.log topic
- onos.log topic
- voltha.kpis topic
- onos.kpis topic
- ...

Logging
- Logstash
- Elasticsearch
- Kibana

Monitoring
- Kafka Topic Exporter
- Prometheus
- 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
- 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

- Stream cached & remote videos
- Include cached video files
- Pull video streams from CDN-Remote

CDN-Local

- Transmit video streams
- Manage video streams
- Include multiple video files

CDN-Remote

Edge Cluster

Central Cluster
CDN-Local & CDN-Remote

- Stream cached & remote videos
- Include cached video files
- Pull content from CDN-Remote

Transmit video streams
Manage video streams
Include multiple video files

This is fully “Open-Source CDN”
CDN-Remote: “Remote” Video Files

(1) Video archive contains video files
(2) Video archive sends videos to media server (RTMP)
(3) Media server transmits video streams (RTMP)
(4) CDN server gets video stream from Media server (RTMP)
(5) CDN server starts streaming service
(6) UE requests and then receives video streaming through OMEC/RAN

Video files

Video archive

Media server

CDN server

CDN-Remote

CDN-Local

Edge Cluster

Central Cluster
CDN-Local: “Cached” Video Files

(1) CDN server includes cached video files

(2) CDN server starts streaming service

(3) UE requests and then receives video streaming through OMEC/RAN

Cached videos

CDN server

Video files

Media server

Video archive

CDN-Local

Edge Cluster

CDN-Remote

Central Cluster
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

1. Send profile TOSCA file to XOS
2. Define services
3. Draw service chain on dashboard
4. Make UI for configuration
5. Onboard synchronizers for services: {ONOS-Fabric, ONOS-RAN}-Synchronizer
6. ONOS monitoring
XOS Workflow in COMAC: Fabric

1. Send Fabric TOSCA file to XOS
2. Save config values
3. Send config values
4. Ask fabric configuration
5. Fabric monitoring
6. ONOS monitoring
7. W/O TOSCA, admin can configure manually through UI

- Fabric file
- XOS
- Synchronizer
- ONOS-Fabric

Node configuration
Switch configuration
Port configuration

XOS-TOSCA
XOS services
Configuration UI

Fabric configuration
Monitoring
Configuration
XOS Workflow in COMAC: RAN Slicing

1. Send Fabric TOSCA file to XOS
2. Save config values
3. Send config values
4. Ask RAN configuration
5. RAN monitoring
6. ONOS monitoring
7. W/O TOSCA, admin can configure manually through UI
Example: Service Chain on Dashboard

An example of COMAC service graph
An example of fabric switch configuration in XOS

## Fabric Switches

<table>
<thead>
<tr>
<th>Actions:</th>
<th>Backend status</th>
<th>Driver</th>
<th>Id</th>
<th>Lpv4 loopback</th>
<th>Lpv4 node sid</th>
<th>Is edge router</th>
<th>Management address</th>
<th>Name</th>
<th>Of id</th>
<th>Router mac</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>ofdpa3</td>
<td>1</td>
<td>192.168.0.204</td>
<td>214</td>
<td>true</td>
<td>192.168.87.151</td>
<td>Agg Switch</td>
<td>of:0000000000000001</td>
<td>54:87:de:ad:be:ef</td>
</tr>
</tbody>
</table>

## Type to search..

<table>
<thead>
<tr>
<th>Actions:</th>
<th>Admin state</th>
<th>Backend status</th>
<th>Host learning</th>
<th>Id</th>
<th>Kind</th>
<th>Oper status</th>
<th>Port id</th>
<th>Switch Id</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>enabled</td>
<td>✓</td>
<td>true</td>
<td>1</td>
<td></td>
<td></td>
<td>33</td>
<td>Agg Switch</td>
</tr>
<tr>
<td></td>
<td>enabled</td>
<td>✓</td>
<td>true</td>
<td>2</td>
<td></td>
<td></td>
<td>34</td>
<td>Agg Switch</td>
</tr>
<tr>
<td></td>
<td>enabled</td>
<td>✓</td>
<td>true</td>
<td>3</td>
<td></td>
<td></td>
<td>35</td>
<td>Agg Switch</td>
</tr>
<tr>
<td></td>
<td>enabled</td>
<td>✓</td>
<td>true</td>
<td>4</td>
<td></td>
<td></td>
<td>37</td>
<td>Agg Switch</td>
</tr>
</tbody>
</table>
An example of UE/IMSI configuration in XOS

<table>
<thead>
<tr>
<th>Actions</th>
<th>Apn number</th>
<th>Id</th>
<th>Imsi number</th>
<th>Name</th>
<th>Owner Id</th>
<th>Ue status</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>66</td>
<td>732111000000420</td>
<td>IMSI1</td>
<td>mcord</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>67</td>
<td>732111000000421</td>
<td>IMSI2</td>
<td>mcord</td>
<td>0</td>
</tr>
</tbody>
</table>
**Example: RAN Slicing**

An example of RAN configuration in XOS

```
<table>
<thead>
<tr>
<th>Actions:</th>
<th>Description</th>
<th>Enb id</th>
<th>Id</th>
<th>Ip addr</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>accelleran</td>
<td>1</td>
<td>1</td>
<td>192.168.87.151</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>Actions:</th>
<th>Adm control</th>
<th>Cell individual offset</th>
<th>Di alloc r b rate</th>
<th>Di sched type</th>
<th>Di Alloc rb rate</th>
<th>Di wifi rate</th>
<th>End</th>
<th>Enodeb Id</th>
<th>Handover Id</th>
<th>Id</th>
<th>Mmeip</th>
<th>Mmeport</th>
<th>Name</th>
<th>Owner Id</th>
<th>Start</th>
<th>Status</th>
<th>Subs profile</th>
<th>Ul Alloc rb rate</th>
<th>Ul Sched type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>RR</td>
<td>0</td>
<td>0</td>
<td>1569826800</td>
<td>1</td>
<td>barcelonahandover</td>
<td>68</td>
<td>10.90.0.131</td>
<td>36412</td>
<td>downloadslice</td>
<td>1567321200</td>
<td>false</td>
<td>IMSI1</td>
<td>5</td>
<td>RR</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>50</td>
<td>RR</td>
<td>0</td>
<td>0</td>
<td>1569826800</td>
<td>1</td>
<td>barcelonahandover</td>
<td>69</td>
<td>10.90.0.131</td>
<td>36412</td>
<td>videoslice</td>
<td>1567321200</td>
<td>false</td>
<td>IMSI2</td>
<td>50</td>
<td>RR</td>
<td></td>
</tr>
</tbody>
</table>
```
Thank You

Follow Up Links:
https://guide.opencord.org