OSM#13 Hackfest
Magma 5G Core Onboarding Challenge

Team 1: VESUVIUS
Enrique Lluesma Martí (ATOS R&D), Mauricio Alexander Vargas Rodríguez (SENA - CEET), Michail Dalgitsis (VICOMTECH), Rasoul Nikbakht (CTTC)
Table of contents

- Amazon cloud Infrastructure
- Onboard Magma Orchestrator and instantiate it
- Configure AWS Route53
- Configure Magma Orchestrator through the UI
- Onboard Magma AGW and instantiate it
- Configure Magma AGW
- Onboard srsLTE eNB & UE simulator and instantiate it
- Conclusions
Infrastructure provided by the organizers

Team1: VESUVIUS

Enrique  Mauricio  Michail  Rasoul
Steps to access the Amazon cloud workspace (AWS)

- Go to Amazon console https://aws.amazon.com/es/console/
- Sign-in with IAM user
  - Account ID: 891210054201
  - IAM user: team1
  - Password: password123!
- Install AWS CLI in a local machine
  - `curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
  - `sudo unzip awscliv2.zip`
  - `sudo ./aws/install`
- Configure AWS
  - `aws configure`

```
$ aws configure
AWS Access Key ID [None]: AKIA47ACCH144VB5FKF7
AWS Secret Access Key [None]: ro0scGnP6W719r1jXrfqmBuWTsbblnmbMjtdjCcRVOh
Default region name [None]: us-east-1
Default output format [None]: json
```
Configuration scenario & overview

1. Onboard Magma Orchestrator KNF into OSM to deploy Magma Orchestrator into the EKS cluster
   Configure AWS Route53
   Install admin_operator.pfx in the web browser
   Configure Magma Orchestrator through web UI

2. Onboard Magma AGW PNF into OSM to orchestrate Magma AWG

3. Onboard srsLTE eNB and UE simulator VNF into OSM to deploy the eNB
Deploy Magma Orchestrator
Access Kubernetes cluster - EKS

- Install eksctl [https://docs.aws.amazon.com/eks/latest/userguide/eksctl.html](https://docs.aws.amazon.com/eks/latest/userguide/eksctl.html)
- Since the K8s cluster is already provided, we need to export the configuration file that corresponds to Team1
  - Make a folder .kube: `mkdir ~/.kube`
  - Copy – paste in .kube the kube_config file from google drive
  - Export configuration: `export KUBECONFIG=~/.kube/kube_config`
  - Check the nodes of the cluster: `kubectl get nodes`

```
NAME     STATUS    ROLES     AGE       VERSION
ti-192-168-48-58.ec2.internal Ready     <none>  5d20h  v1.22.6-eks-7d68063
ti-192-168-49-86.ec2.internal  Ready     <none>  5d20h  v1.22.6-eks-7d68063
```
- Other commands to check what is in the EKS cluster:
  - Get a list of all the running pods: `kubectl get pods -A`
  - Get a list of all namespaces: `kubectl get namespaces`
Access OSM EC2 instance (1/2)

- sudo ssh -i team1-key.pem ubuntu@184.73.40.173

- With osm -h we can see all the available commands we can execute through osm machine
  - Check the added VIMs: `osm vim-list`
  - Check the added Kubernetes clusters: `osm k8scluster-list`

- Also, since OSM has been installed upon another Kubernetes cluster with microk8s.kubectl we can interact with it: `microk8s.kubectl get namespaces`
Lastly, we can access OSM GUI's on:

https://ui.184.73.40.173.nip.io/login

- Username: admin
- `ubuntu@ip-192-168-32-40:$ env | grep OSM_PASSWORD`
  
  OSM_PASSWORD=8975766776cb4a543213b6ec122bdc91
On the OSM instance we have created a directory called packages and uploaded all the files provided for Magma orchestrator, AGW and srsLTE though SSH:
Magma orchestrator Onboarding to OSM (2/3)

• Onboarding to OSM
  • osm nfpkg-create magma_orc_cnf.tar.gz

  magma_orc_cnf    e957263-a973-44f6-b235-1004f1976666  1.0  Canonical  KiS container deployment of Magma Orchestrator

  • osm nspkg-create magma_orc_ns.tar.gz

  magma_orc_cnf.ns  76db5e65-fa4f-4c58-9ade-59db4f93e910  1.0  Canonical  NS with 1 KDU connected to the mgmtnet VL

• Instantiating
  • osm ns-create --ns_name magma_orc_cnfNs --nsd_name magma_orc_cnf_ns --vim_account aws-site
  • Magma ORC takes around 15 min to be deployed:

    juju status -m magma-orc-kdu-898e9ce1-f59b-4f9a-4a02e766c5 --watch 10s
After onboarding the KNF and NS packages and instantiating the network service, the Magma Orchestrator is running on the EKS Kubernetes cluster.
After instantiating Magma orchestrator, there are services running in the EKS cluster related to it.

With `kubectl get services -n magma-orc-kdu-898e9ce1-f59b-4fea-a309-4a4022e766c5` we can see all these services.

However we are interested in the LoadBalancer service type.

- `kubectl get services -n magma-orc-kdu-898e9ce1-f59b-4fea-a309-4a4022e766c5 | grep LoadBalancer`

Here we can find the Internal and External IP addresses.
To configure AWS Route53 appropriately we need to create A records.

One way is through the AWS console and another is through the AWS CLI.

Through the AWS CLI, we need to create an arecords.json file like the one on the figure.

The DNSName field must be filled according to the services from the previous slide using the external IP.

We apply this file with the command:

```
aws route53 change-resource-record-sets --hosted-zone-id Z0309839BG6MW88K4E1V --change-batch file://arecords.json
```

**NOTE**: We execute this command from our local terminal, neither from EKS nor from any EC2 instance.
AWS Route53 configuration (3/3)

- We can confirm that all run well either by AWS CLI or by AWS Console

Confirm by AWS CLI

Confirm by AWS Console
• Magma Orchestrator installation provides us a GUI. To access it we need to use a certificate

• We need to check the models and download the Orchestrator’s certificates for our browser from the OSM machine with the juju command:

• Export it to our local environment
• Install the admin_operator.pfx certificate on the browser with the password: password123
Day-2 actions to get credentials for Magma Orchestrator GUI (1/2)

- Get the GUI credentials
- `$ osm ns-action magma_orc_cnf_ns \n  --vnf_name magma_orc_cnf \n  --kdu_name magma-orc-kdu \n  --action_name get-admin-credentials
- `$ osm ns-op-show 989205c5-c081-4957-9cbd-c11e532578c3
Day-2 actions to get credentials for Magma Orchestrator GUI (2/2)

- Access the [https://master.nms.team1.osmhackfest.com](https://master.nms.team1.osmhackfest.com) with the credentials

```
"worker": "619bc1924b80"
"Code": "9",
"admin-password": "eFFMGLqoEv",
"admin-username": "admin@team1.osmhackfest.com"
```

![Magma Orchestrator GUI](image1)

![Magma Login](image2)
• Create new tenant in the NMS
Configuration of Magma Orchestrator (2/7)

- Create NMS user for tenant:
  User: user@team1.osmhackfest.com
  Password: password1234
  Role: superuser
Configuration of Magma Orchestrator (3/7)

• Log in to the NMS https://vesuvius.nms.team1.osmhackfest.com
Configuration of Magma Orchestrator (4/7)

• Create an LTE network in tenant’s NMS
• Configure PLMN (MCC: 722, MNC: 17)
• Create APN (name: default)
Configuration of Magma Orchestrator (7/7)

- Import subscribers to the NMS (csv file available on each team’s Google Drive)
Deploy Magma Access GW
Before we onboard the PNF into OSM, we need to tell OSM about the PDU. We need the VIM-ID:

```bash
ubuntu@ip-192.168.32.48:~$ osm vim list | grep aws | awk '{print $4}'
afdd0ec-bb75-4b98-8f39-9ab2a136a624
ubuntu@ip-192.168.32.48:~$`

We create a pdu.yaml descriptor file in our working directory. The pdu.yaml file must have a specific structure:

- `vim_account`: afd90edc-bb75-4b98-8f39-9ab2a136a624
- Magma AGW SGi IP: 192.168.51.193
- Magma AGW S1 IP: 192.168.34.52

```bash
ubuntu@ip-192.168.32.48:~$: cd packages/
ubuntu@ip-192.168.32.48:~$ packages$ cat pdu.yaml
name: MagmaGW
description: Magma Access GW
type: gateway
vim_accounts: [afdd0ec-bb75-4b98-8f39-9ab2a136a624]
shared: False
interfaces:
  - name: eth0
    address: 192.168.51.193
    mgmt: True
  - name: eth1
    address: 192.168.34.52
    mgmt: False
ubuntu@ip-192.168.32.48:~$ packages$
```
Onboarding Magma Access GW

• Creating packages

$ osm nfpkg-create magma_agw_pnf.tar.gz
$ osm nspkg-create magma_agw_ns.tar.gz
Instantiating Magma Access GW

• Instantiating the Network Service

    ubuntu@ip-192-168-32-40:/packages$ osm ns-create --ns_name magma_agw_ns \
    > --nsd_name magma_agw_ns \
    > --vim_account aws-site

• We can now orchestrate the Magma AGW EC2 machine
After instantiating the Magma AGW Network Service, from the OSM machine, we have to download a key with the following command:

```bash
nuju scp -m $model-name --container="magma-orc8r-certifier"
orc8r-certifier/0:/var/opt/magma/certs/..data/rootCA.pem \rootCA.pem
```
Download certificates – Part 2 (2/2)

• Then, we have to copy & paste it into the Magma AGW EC2 machine.
Access Magma AGW EC2

• ssh -i team1-key.pem **ubuntu@54.161.100.52**

• Paste the rootCA.pem file

• Then, run the command:
  - *magma-access-gateway.configure --domain team1.osmhackfest.com --root-ca-pem-path ./rootCA.pem*

• Magma AGW configuration starts
  - Copying files
  - Configures Control Proxy
  - Restart services

• When the configuration is done, we must add the Access Gateway to Magma Orchestrator
Magma AGW configuration

root@ip-192-168-51-193:/home/ubuntu# magma-access-gateway.configure
> magma-agw-configure -d domain.team1.com --domain.team2.com -p /rootCA.pem

Magma AGW Configuration: Starting magma AGW configuration...
Magma AGW Configuration: Copying Root CA PEM from /rootCA.pem to /var/opt/magma/tmp/certs...
Magma AGW Configuration: Controlling Config Proxy...
Magma AGW Configuration: Restarting magma Agw services...
Magma AGW Configuration: Stopping magma* service...
Magma AGW Configuration: Restarting magma@mgwrd service...
Magma AGW Configuration: magma Access Gateway configuration done!
Magma AGW Configuration: To add this Access Gateway to an Orchestrator please use hardware secrets printed below:

Magma AGW Configuration: Hardware ID
Magma AGW Configuration: 7eed1da2-2152-4f6c-8c6b-fc69a1f24f07c

Magma AGW Configuration: Challenge key
Magma AGW Configuration: Note: Hardware ID is this gateway's unique identifier
Magma AGW Configuration: Challenge key is this gateway's long-term keypair used for bootstrapping a secure connection to the cloud
Magma AGW Configuration: Build info shows git commit information of this build
Magma AGW Configuration: Once Access Gateway is integrated with the Orchestrator, run magma-access-gateway.post-install to verify the Installation.

STEP1

STEP2
STEP1 - Integrate Access gateway with magma Orchestrator (1/2)

- Access the Magma Orchestrator through web GUI and select equipment, then select gateways
STEP1 - Integrate Access gateway with magma Orchestrator (2/2)

• Add the new gateway with the hardware ID and challenge key from the previous AGW configuration
STEP2 - Install magma-access-gateway.post-install

```
root@ip-192-168-51-193:/home/ubuntu# magma-access-gateway.post-install
Magma AGW Post-Install: Starting Magma AGW post-installation checks...
Magma AGW Post-Install: Checking network interfaces configuration...
Magma AGW Post-Install: Checking eth0's internet connectivity...
Magma AGW Post-Install: Checking whether required services are running...
Magma AGW Post-Install: Checking whether required packages are installed...
Magma AGW Post-Install: Checking whether Root Certificate exists...
Magma AGW Post-Install: Checking Control Proxy configuration...
Magma AGW Post-Install: Checking AGW connectivity with Orchestration...
Magma AGW Post-Install: Magma AGW post-installation checks finished successfully.
root@ip-192-168-51-193:/home/ubuntu#
```
STEP3 – Confirm AG with Magma Orchestrator (Optional)

• Check the AGW status by clicking on the gateway ID
Deploy SRS LTE
Onboarding srsLTE to OSM

• We start creating the packages

$ osm nfpkg-create srs-lte-enb_vnfd.tar.gz
$ osm nspkg-create srs-lte-enb_nsd.tar.gz
Deploy (Instantiate) the eNB

• Execute the command (Takes some time to deploy the enb instance)

```bash
$ osm ns-create --ns_name enb --nsd_name srs-lte-enb_nsd --vim_account aws-site --config "{"vid: {
    (name: mgmt, vim-network-name: subnet-098c0d6260c1f0a95
}
],
additionalParamsForVnf: [
    {member-vnf-index: 'srsLTE',
    additionalParams: {
        bind_address_subnet: '192.168.32.0/19',
        mme_addr: '192.168.34.52',
        enb_mcc: '722',
        enb_mnc: '17'}}]"
```
Attach UE to Magma (from CLI)

• The attachment of users is a Day-2 OSM action

• Execute the command

```
$ osm ns-action enb --vnf_name "srsLTE" --vdu_id srsLTE-vdu --action_name attach-ue --params '{
    usim-imsi: "722170000000008",
    usim-k: "00112233445566778899aabbccddeeff",
    usim-opc: "63bfa50ee6523365ff14c1f45f88737d"
}
```

Output: bbb7becc-4a2b-456e-b733-940e829da171

• We can also attach users through the OSM dashboard (UI)
- Select “Exec primitive” on the eNB instance on OSM web GUI
• Select “VDU Level”, VNF Profile ID, VDU primitive and Add Primitive params with the previous UE values, the execute:
Attach UE to Magma (from UI) (3/4)

- Check the History on the enb NS instance:
• Check the UE connected on the Magma Orchestrator:
One step forward (1/4)

- **Access one active UE**
  - Description: Access a UE and generate traffic to a web site or simply ping it.
  - osm vnf-list
  - ssh 54.80.0.209
  - User: Ubuntu
  - Pass: osm2022
  - Check tun_srsue
One step forward (2/4)

- **Access one active UE**
  - Check the `rr.conf` file to get the Cell ID and TAC to configure on Magma-Orc
• Detach specific UE

• Description: The current deployment with the detach action, removes all the active users. The detach_specific_ue action could detach a specific UE based on its IMSI number.
• Detach specific UE

• Description: The current deployment with the detach action, removes all the active users. The detach_specific_ue action could detach a specific UE based on its IMSI number.
Conclusions (1/2)

Team1: VESUVIUS

- Enrique
- Mauricio
- Michail
- Rasoul
Conclusions (2/2)
Thank you!