Open Source MANO by ETSI

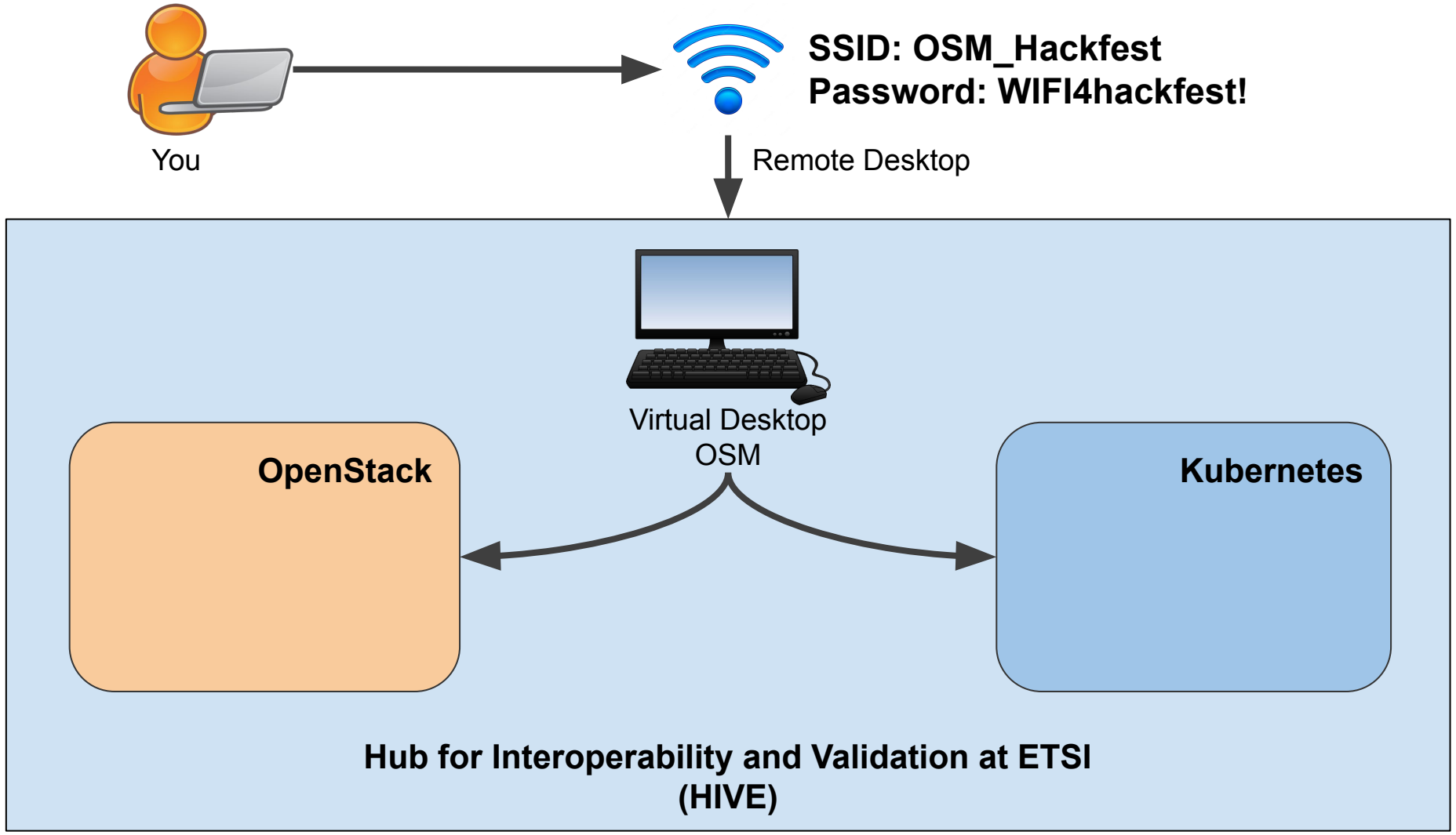
VNF Orchestration: Virtual Machines, Public and Private Cloud

Preethika P. (Tata Elxsi)

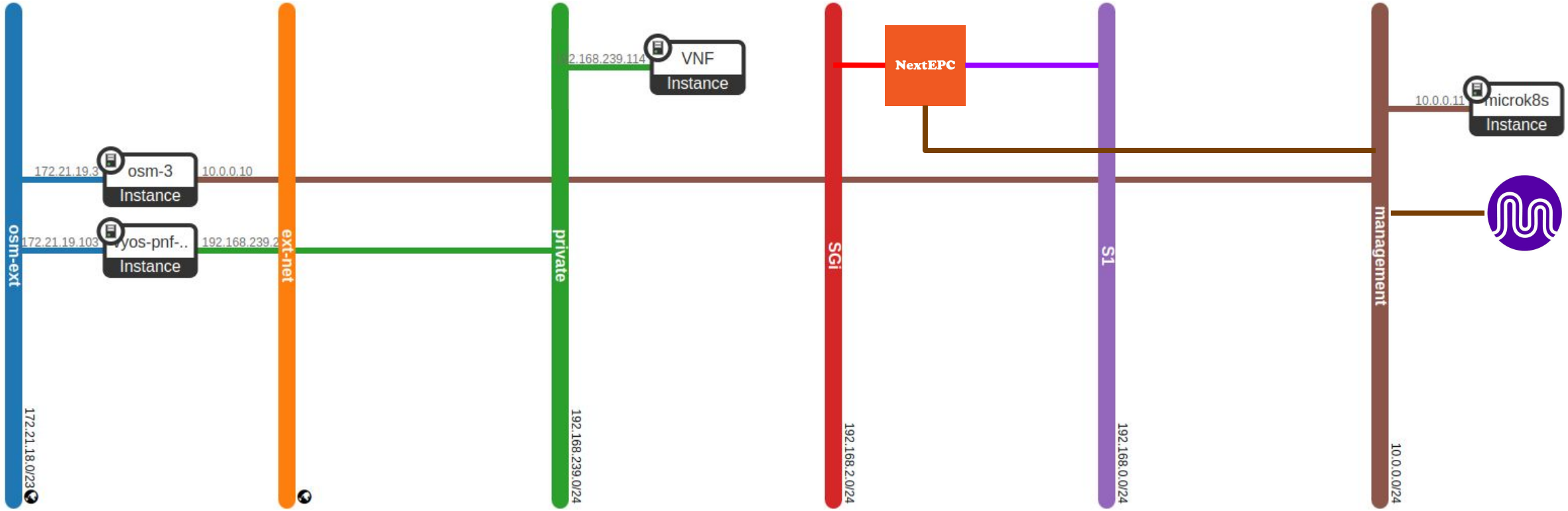
October 18, 2022

Welcome to the Hackfest

Hackfest Environment



Your Openstack Tenant



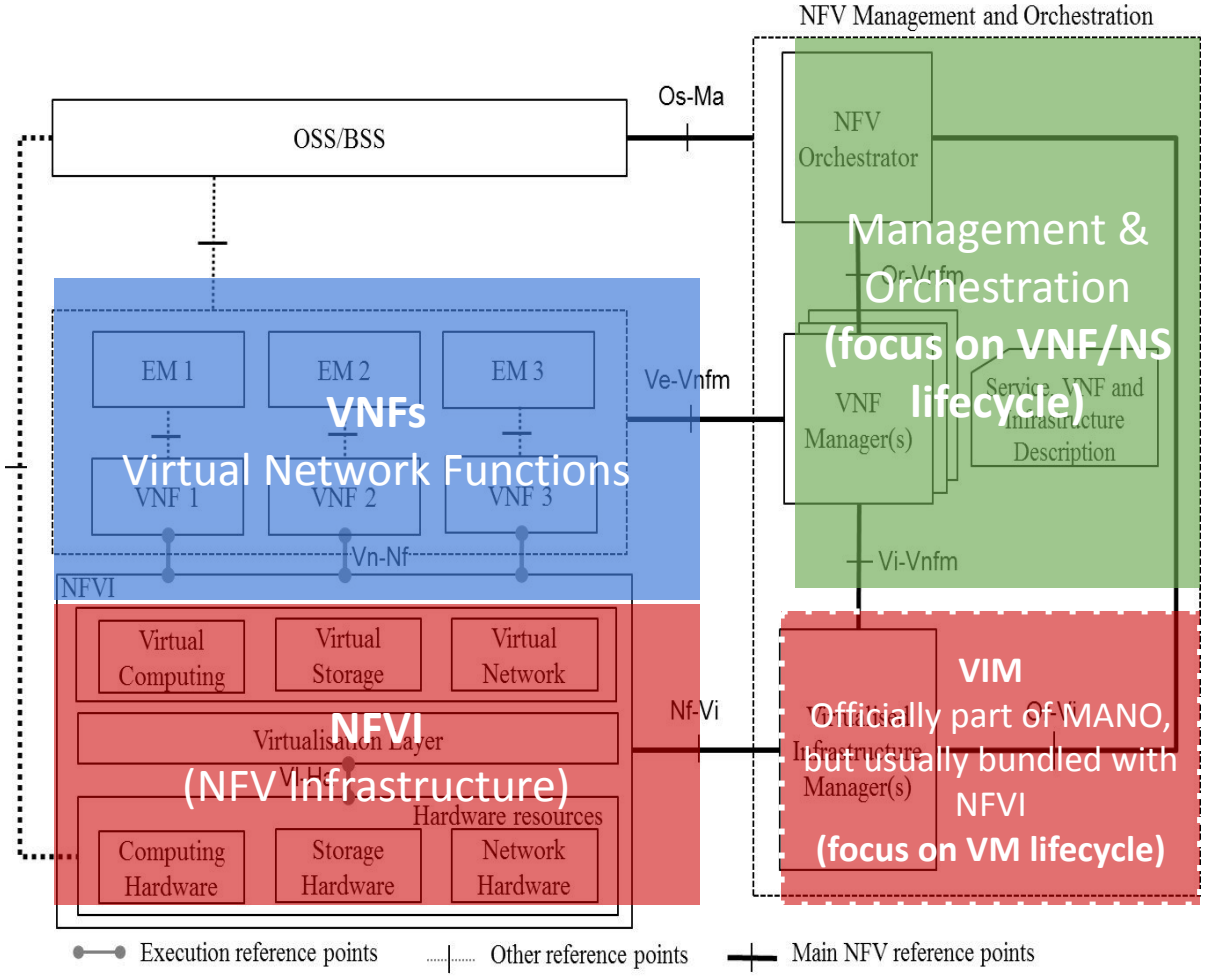
Agenda

- Overview of OSM
- VNF Onboarding Workflow
- NextEPC Modelling
- NextEPC Deployment

Overview Of OSM

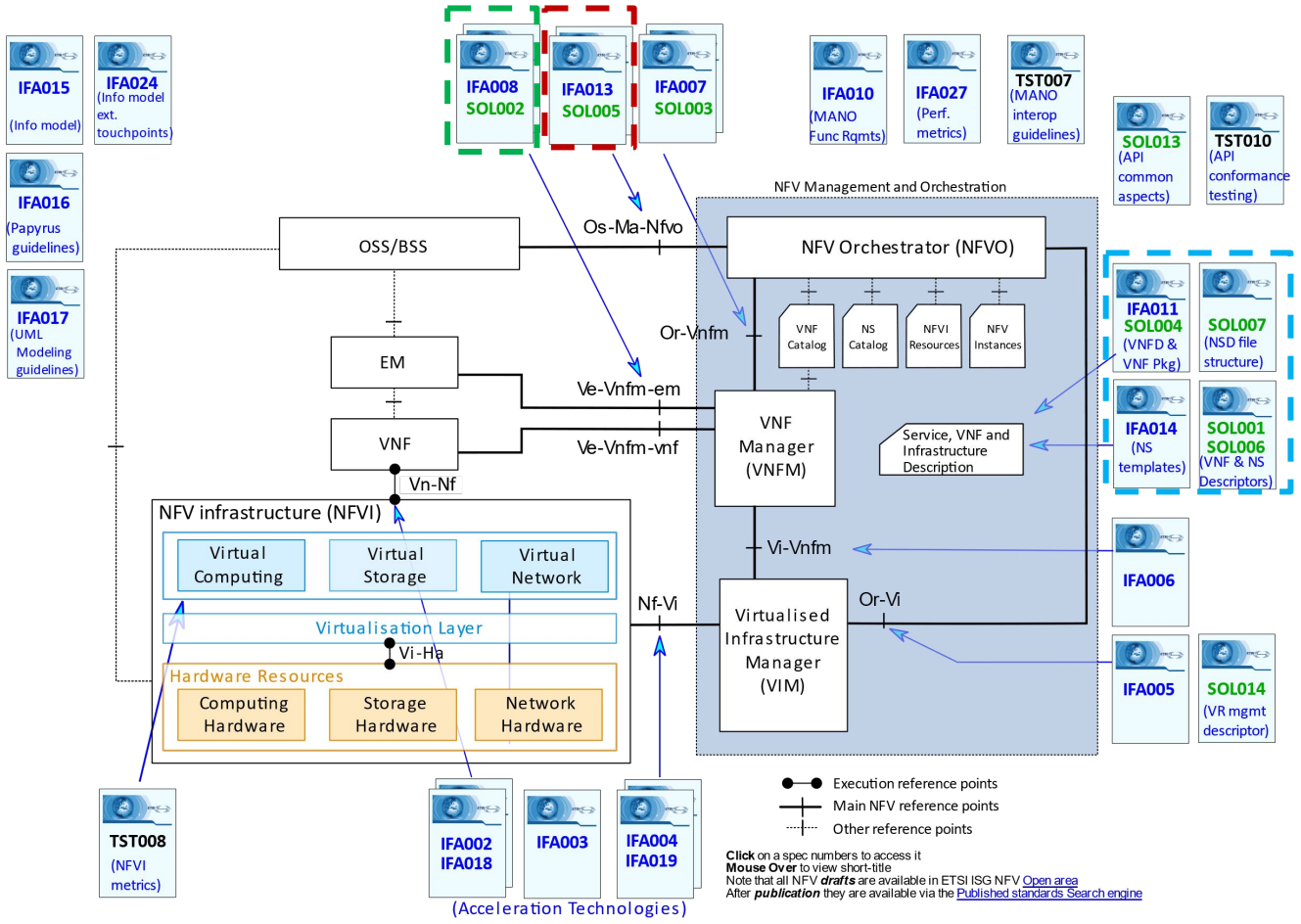
The ETSI NFV Architecture

Open Source MANO



ETSI NFV Architecture and Specifications

All you need is a map



Os-Ma-Nfvo reference point (interface between OSS/BSS and NFVO)

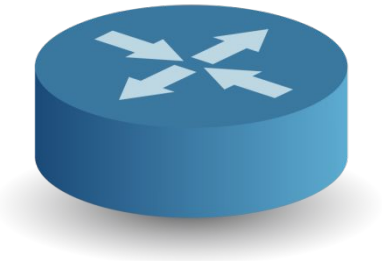
Ve-Vnfm-em/vnf reference points (interface between VNFM and EM/VNF)

VNF and NS descriptors and packages

Source: ETSI. Web: https://www.etsi.org/images/articles/NFV%20Architecture.svg

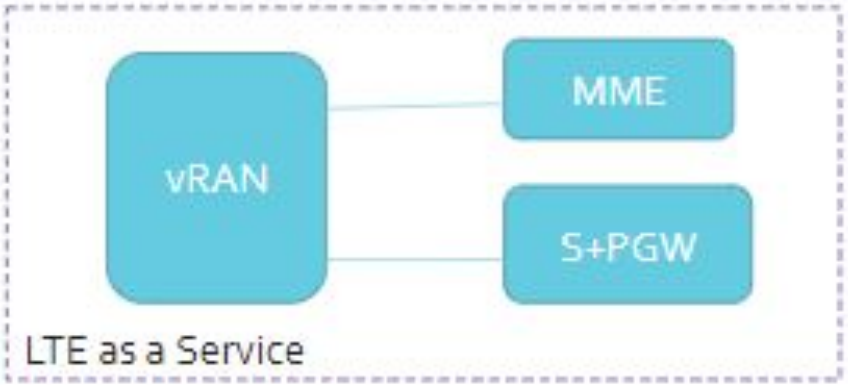
Network Functions

- One or many nodes in a Network Infrastructure that has well defined interfaces and functional networking capability.
- Examples: Firewall, Router, EPC, IMS, etc
- Different Flavours of Network Functions
  - Virtual Network Function
  - Cloud Native/Container Network Function
  - Physical Network Function



Network Services

- The Network Service (NS)is a topology of interconnected NF.
  - ABSTRACTION (NS definition): The topology is agnostic from the place where NF will be deployed.
  - PARTICULARIZATION (NS instance): When instantiating it, parameters are provided specific for those NF instances.
- It is deployed and operated as a whole.
- Examples: LTE, VPN, LAN internet, etc.



OSM IM

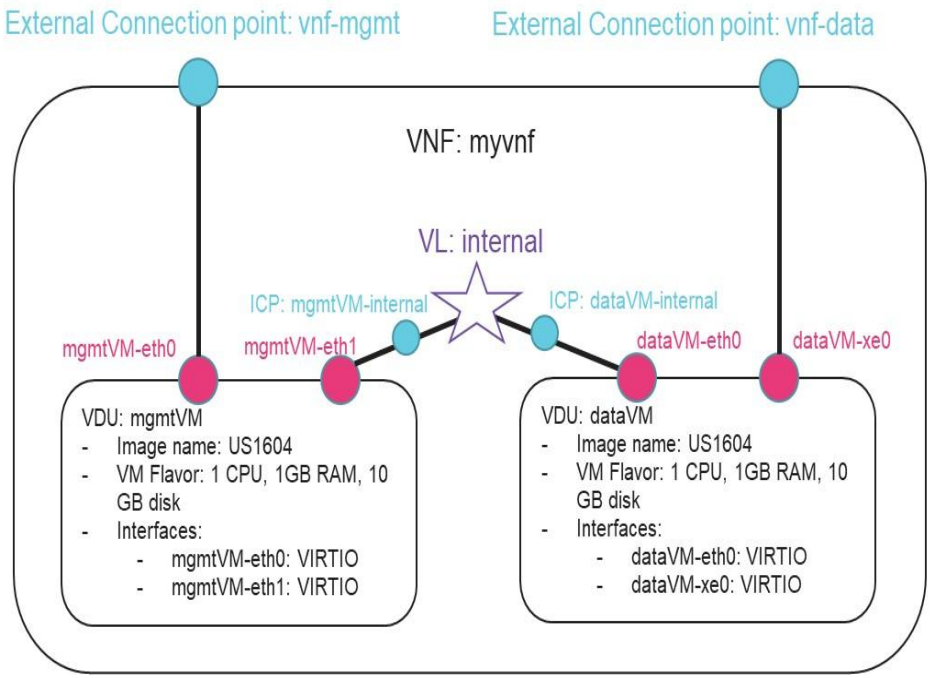
- Information model (IM) to define the different descriptor templates:
  - Network Function (NF)
  - Network Service (NS)
  - Network Slice (NST)

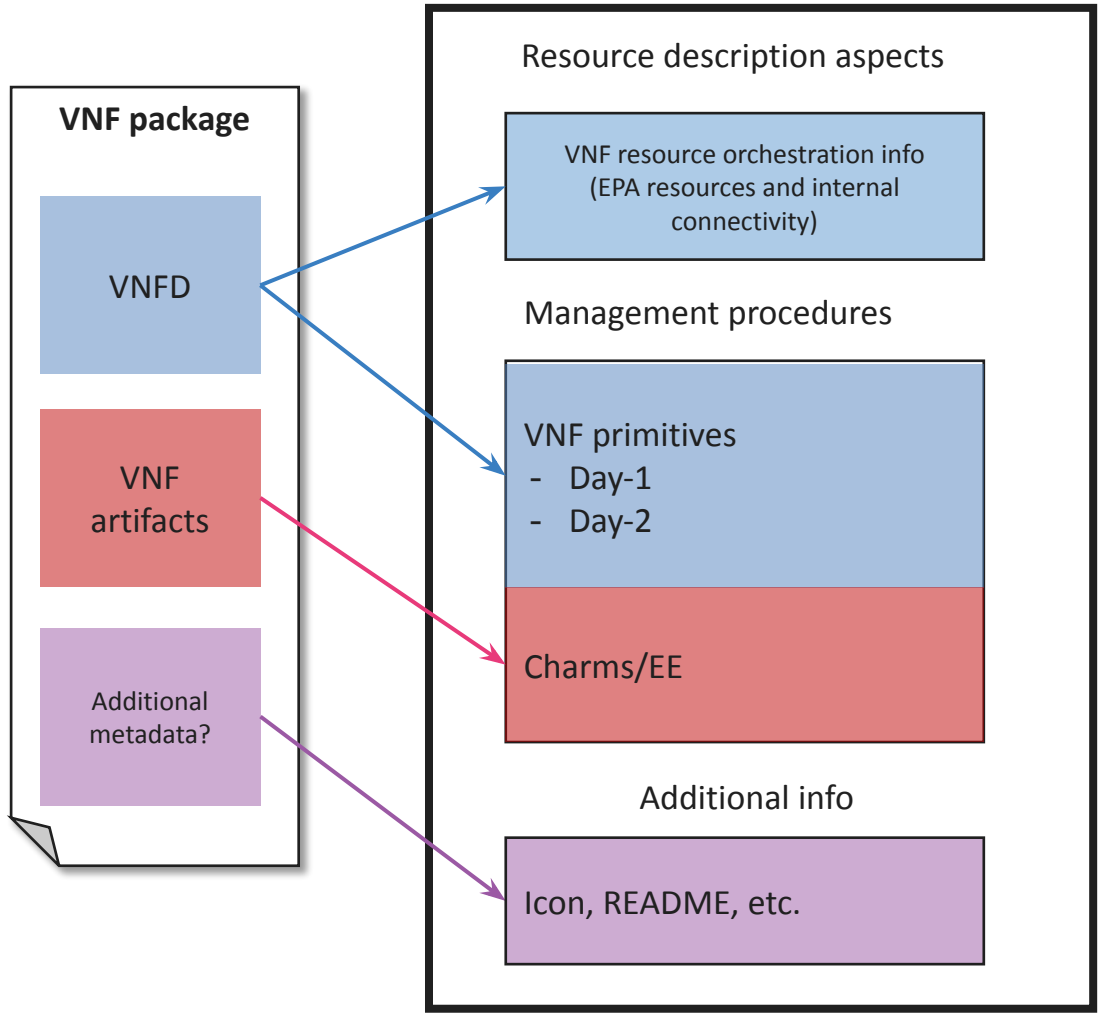
- OSM IM is based on YANG*.

- OSM IM aligned with ETSI NFV, derived from SOL006 (which, in turn, derives from IFA011 and IFA014).

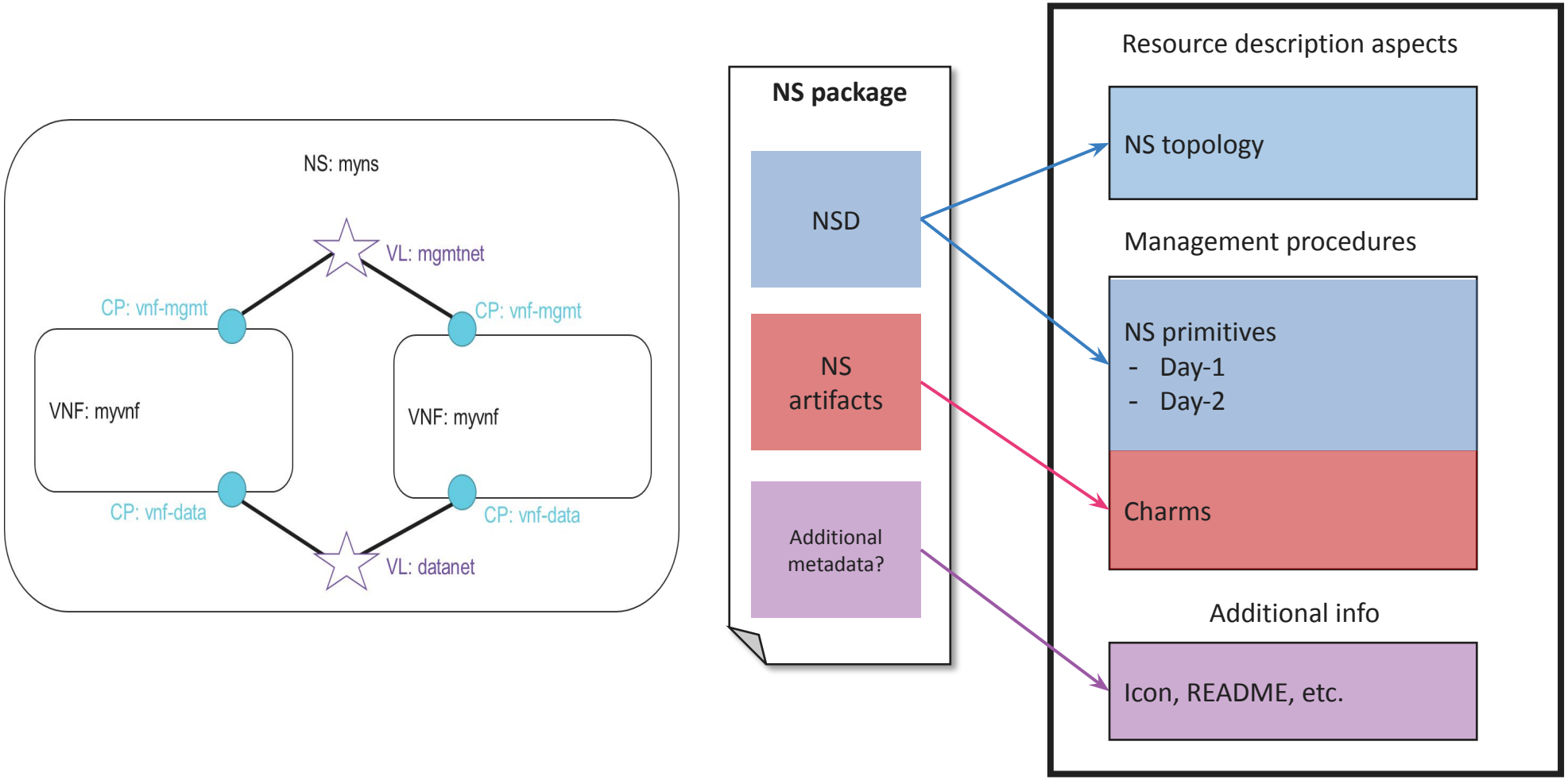
- IFA011 describes the VNF descriptor specification whereas IFA014 on NS descriptor.

Modeling NF
NF Package vs NF Descriptor





Modeling NS
NS Package vs NS Descriptor



Different Stages

Design Time

- Building NF and NS descriptors by referencing the OSM Information Model.
- Building Day-1/Day-2 logic to complete the packages
- Testing your packages

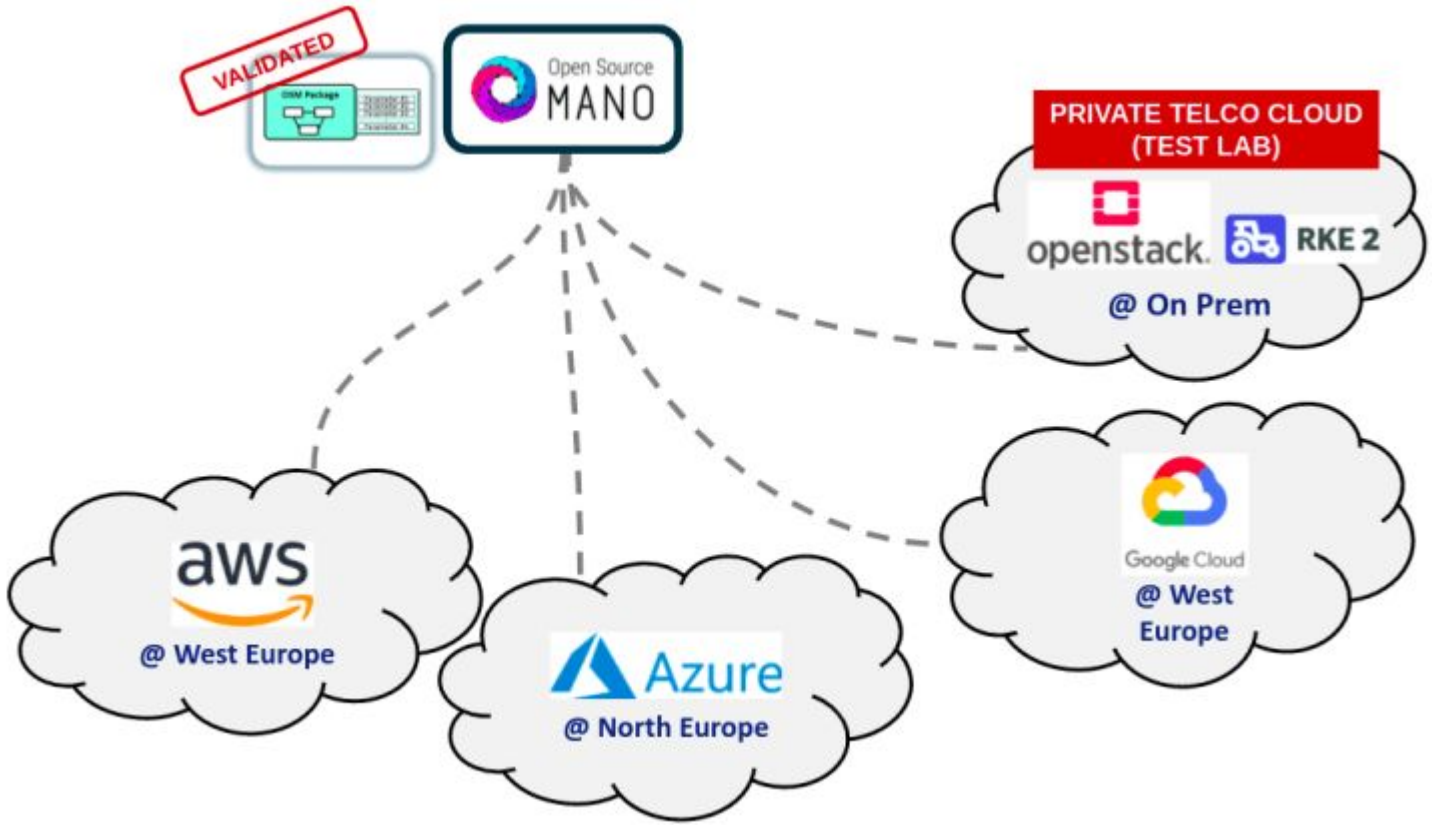
Provisioning Time

- Onboarding: package validation and uploading

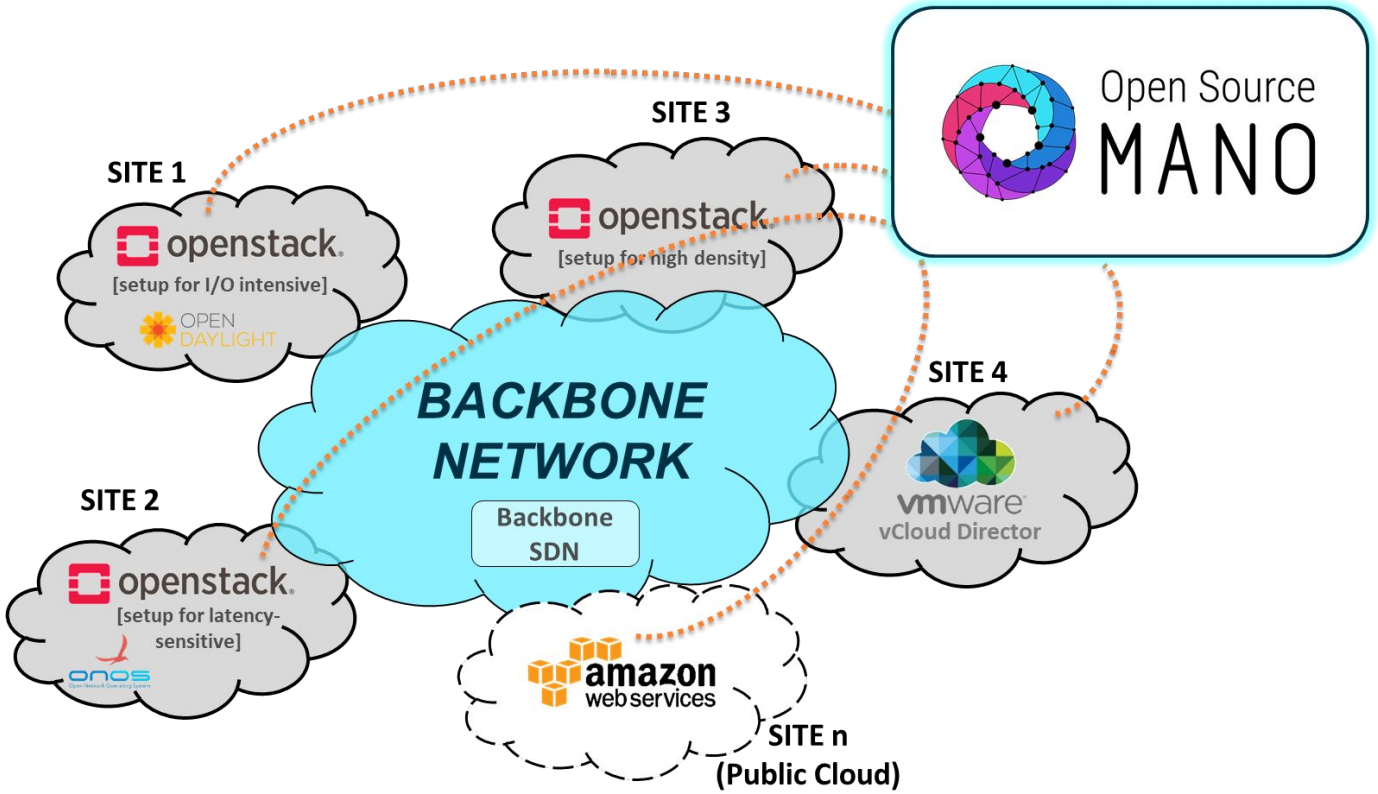
Run Time (see your VNF and NS in action)

- Instantiation (Deployment + Day 0 + Day 1). Here is where the operator decides:
- In which VIM to deploy
- What instantiation parameters to provide (specific IP addresses, configuration params)
- Operation phase (Day 2)

Multi Cloud Orchestration



Multi Site Orchestration



VNF Onboarding workflow

- Network Functions Virtualisation will only scale if all of the functions can be automated.

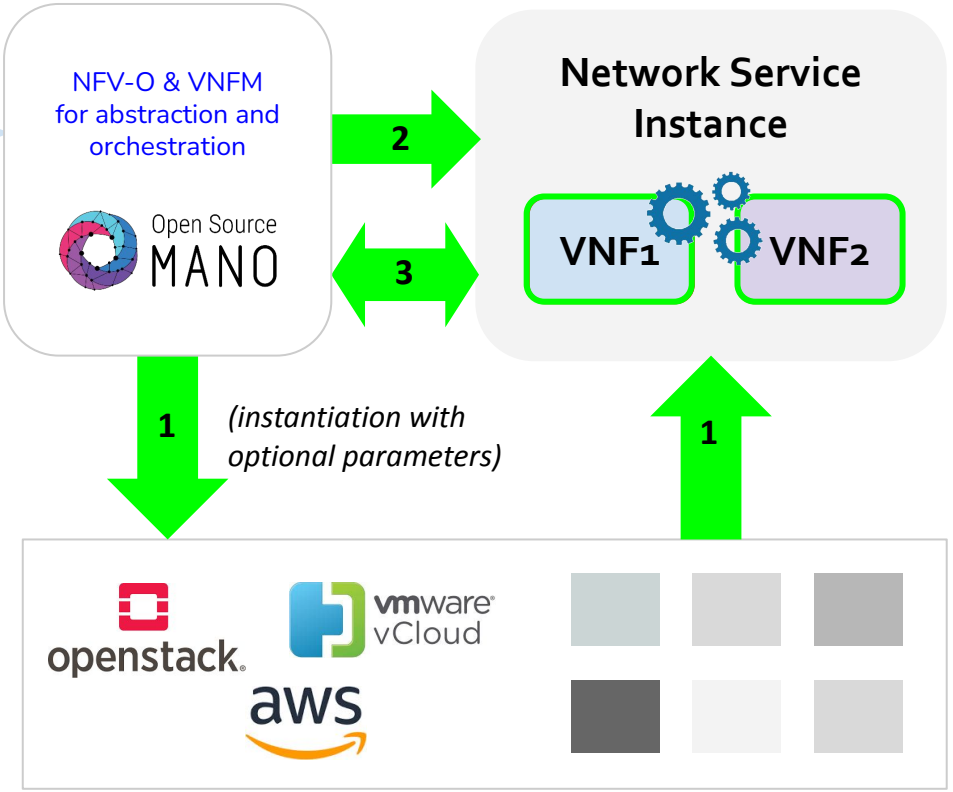
...specially true for 5G!

VNF Packages
(unique)

1. Instantiate Network Services/Slices, making VNFs manageable **("Day 0")**

2. Initialize VNFs so they provide the expected service **("Day 1")**

3. Operate the service: monitoring, reconfigurations and (closed-loop) actions **("Day 2")**

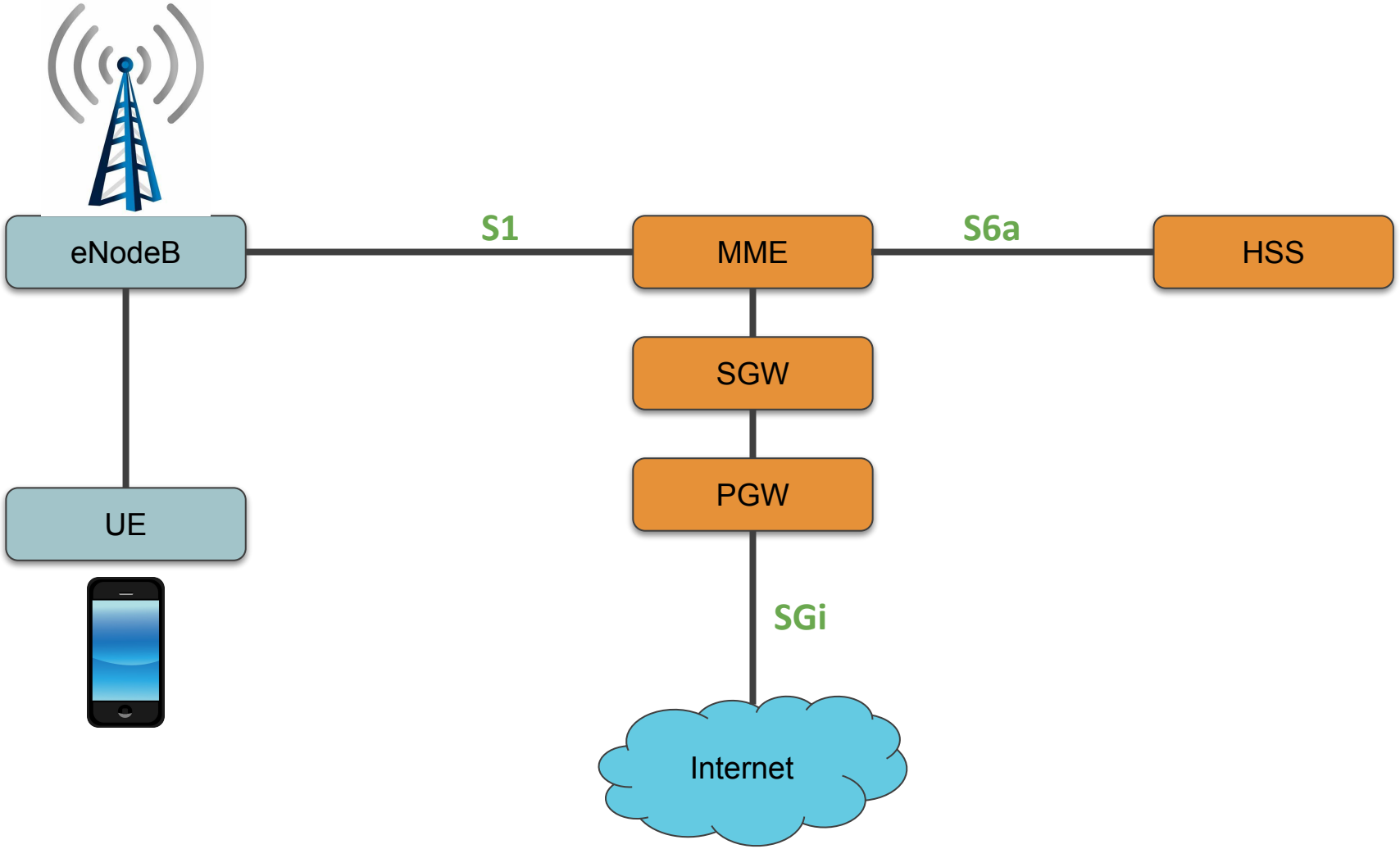


Deployment of NextEPC

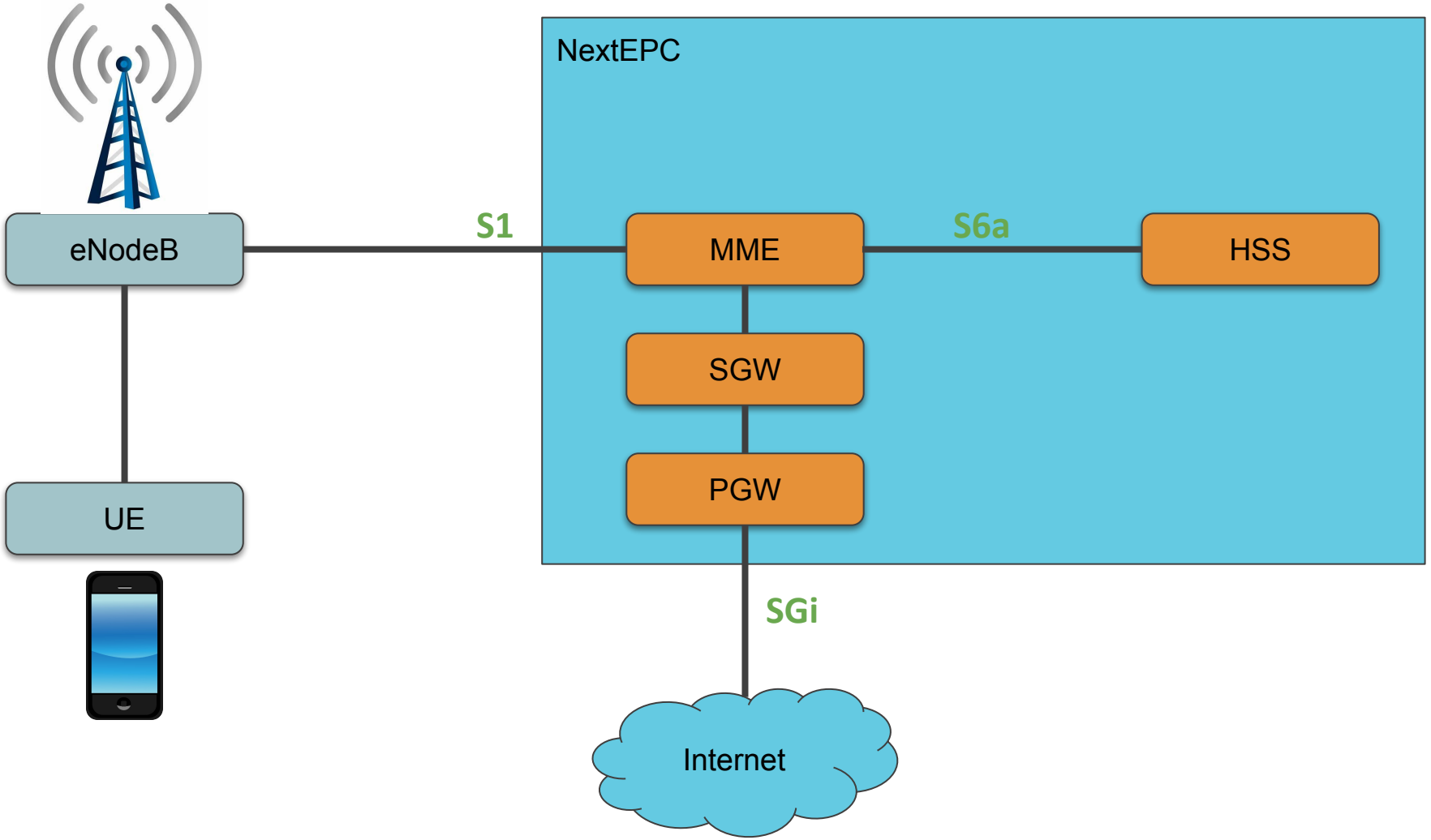
Next EPC Overview

- MME | Mobility Management Entity
  - Keeps track of User Equipment registered on LTE network
  - Handles requests for network access - setting up and tearing down data sessions
- SGW | Serving Gateway
  - IP router with GTP support and charging functionality
  - Module for signalling between PGW and MME
- PGW | Packet Data Network Gateway
  - Provides access to external Packet Data Networks (i.e: internet)
- HSS | Home Subscriber Service
  - Stores subscriber keys and permitted services

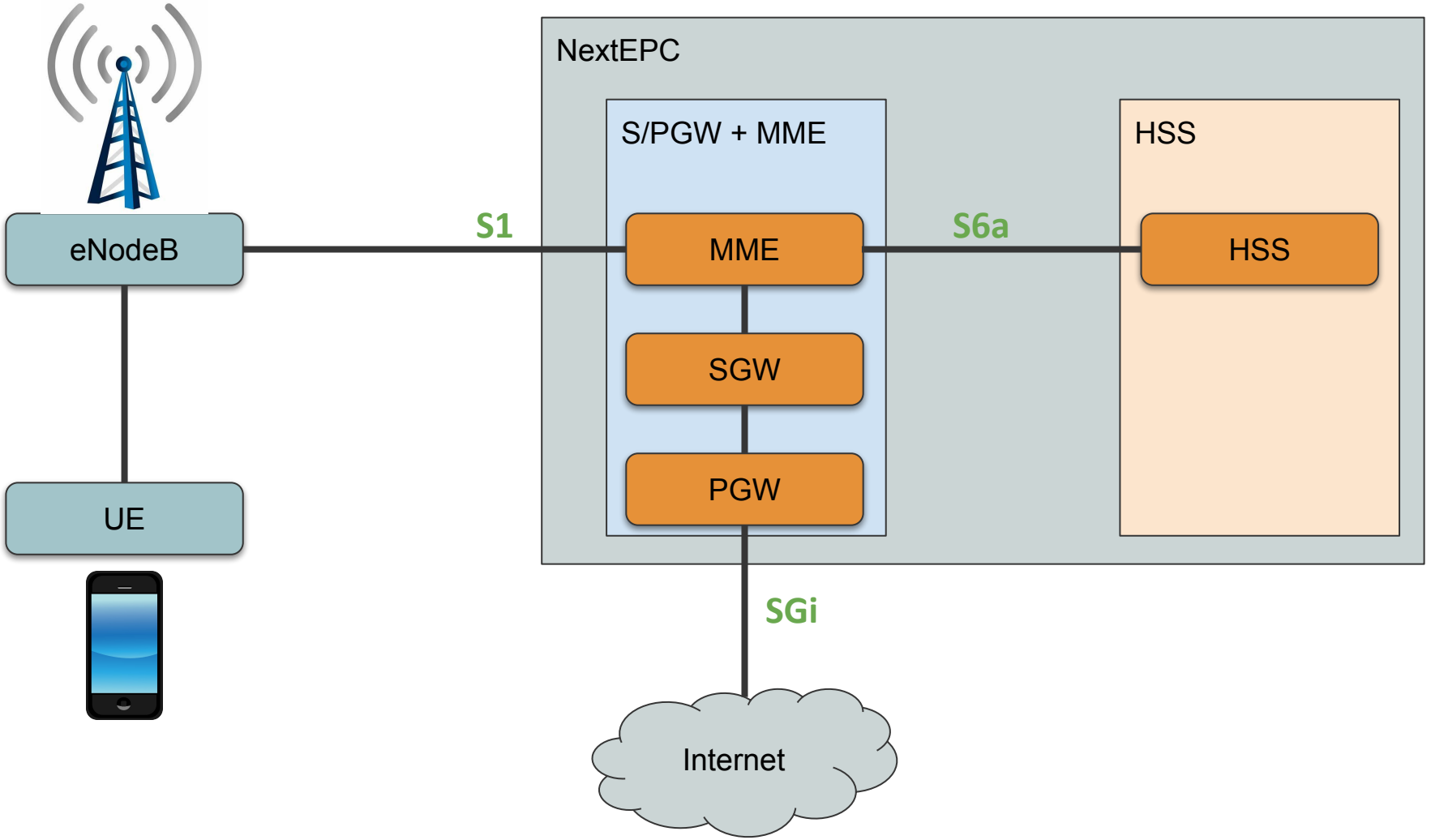
EPC Abbreviated Version



NextEPC Software View



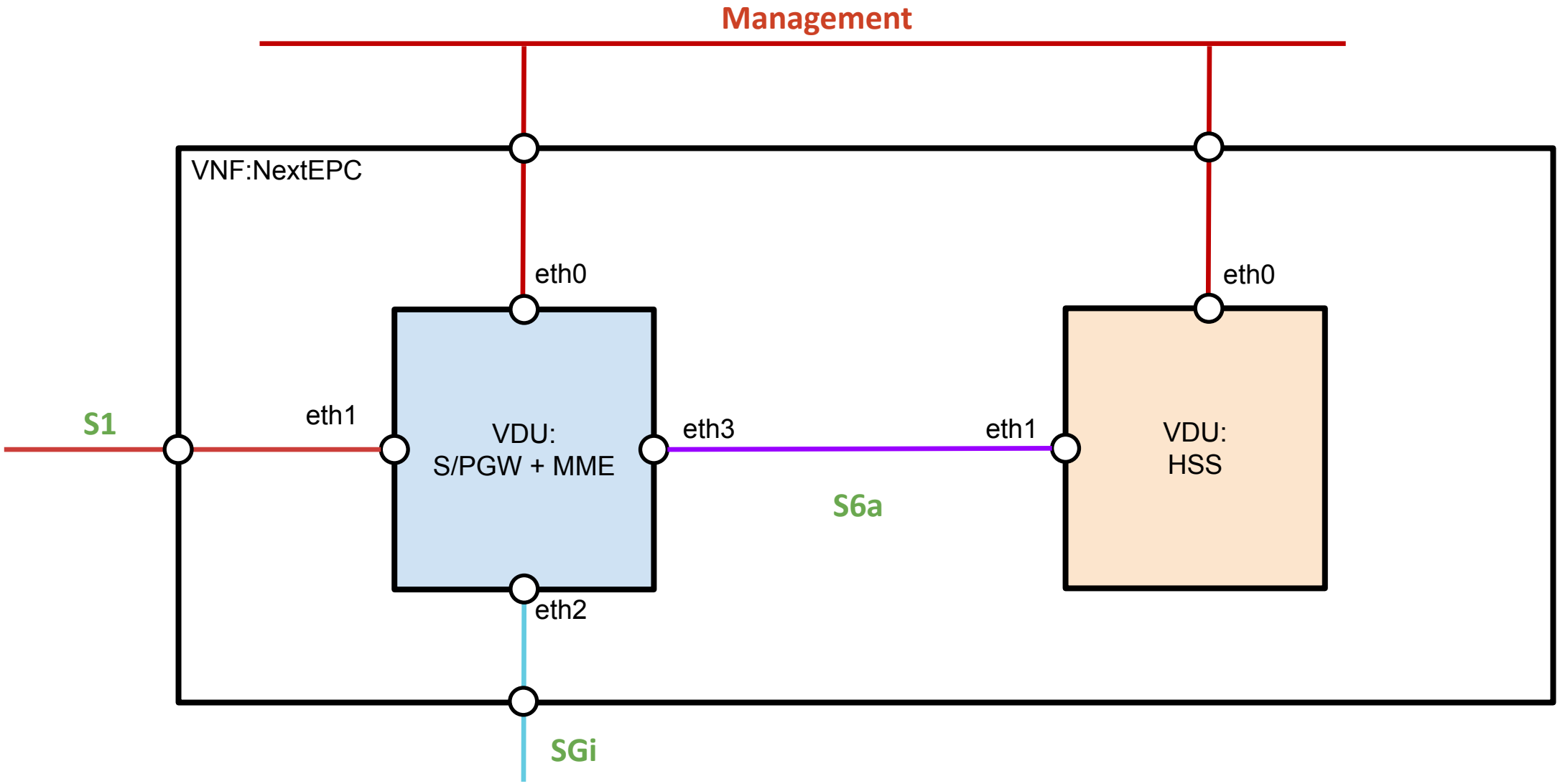
NextEPC Deployment Sample - OSM



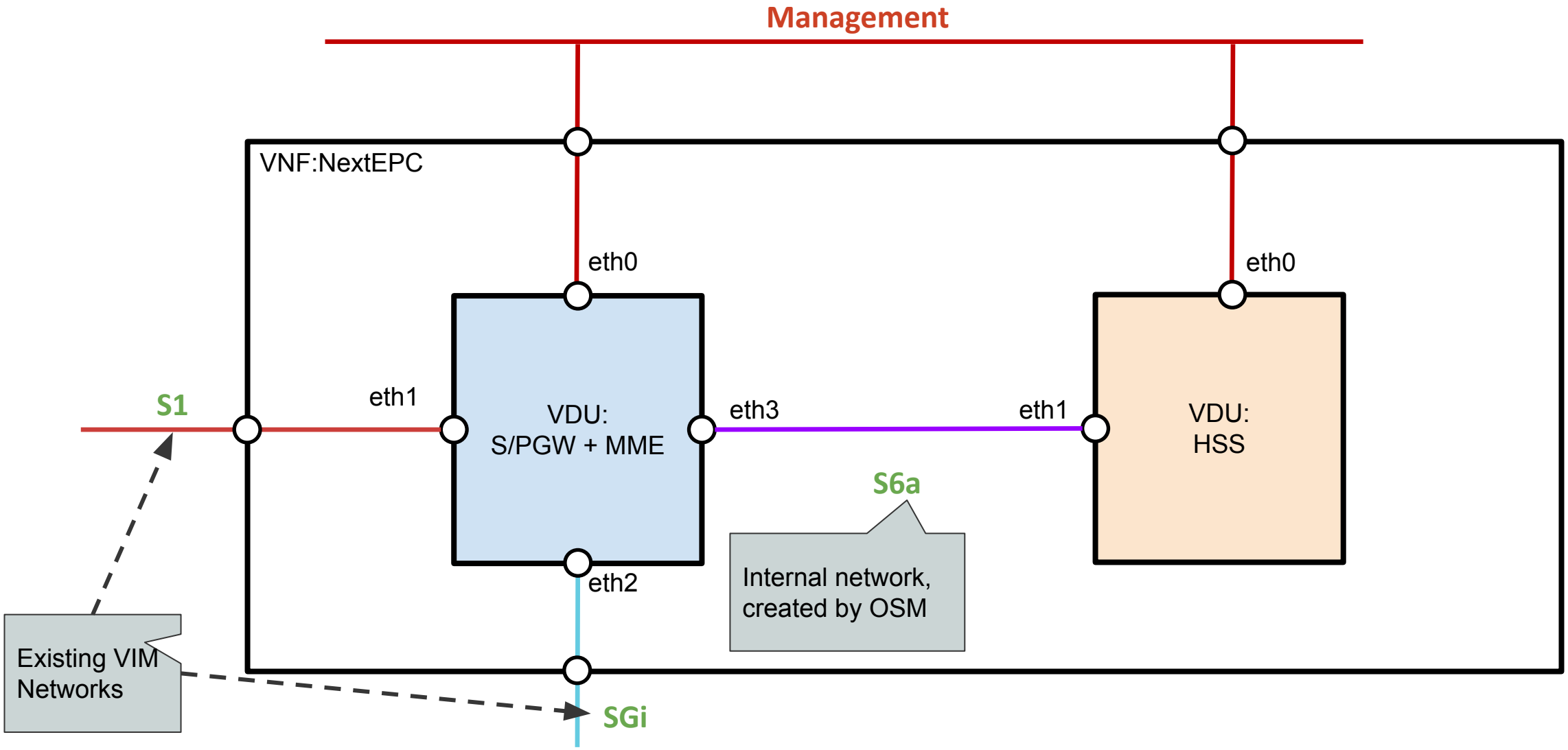
Day 0 Requirements

- Objectives of Day 0
  - Provide the guidelines for including all the necessary elements in the VNF Package for its successful instantiation and management setup, so it can be further configured at later stages.
- Requirements
  - Images for OpenStack
  - Networks (already existing in VIM, or need to be created)
  - Hostnames and IP addresses if needed

NextEPC VNF Model



NextEPC VNF Model

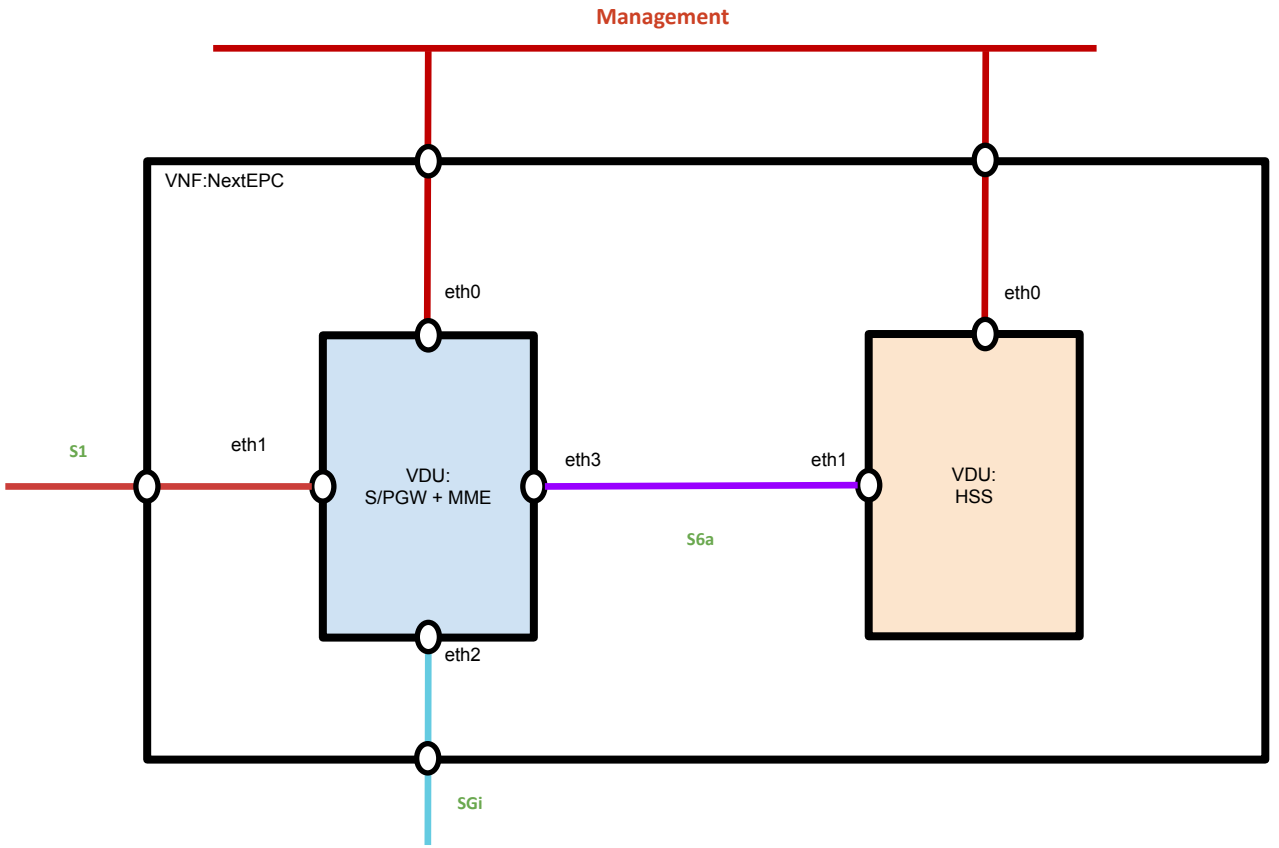


Day1 Requirements

- Objectives of Day 1
  - Provide the guidelines to include all necessary elements in the VNF Package.
  - Allows the exposed services inside the VNF to be automatically initialized right after the VNF instantiation.
- Requirements
  - Identify dependencies between components
  - Define required configuration for service initialization
  - Identify the need for instantiation parameters

Day1 Requirements

- S/PGW + MME requires:
  - 4 network interfaces and addresses
  - SSH keys for management
  - Update of MME configuration with IP address of HSS
- HSS requires:
  - 2 network interfaces and addresses
  - SSH keys for management
  - Update of HSS configuration with IP address of SPGW

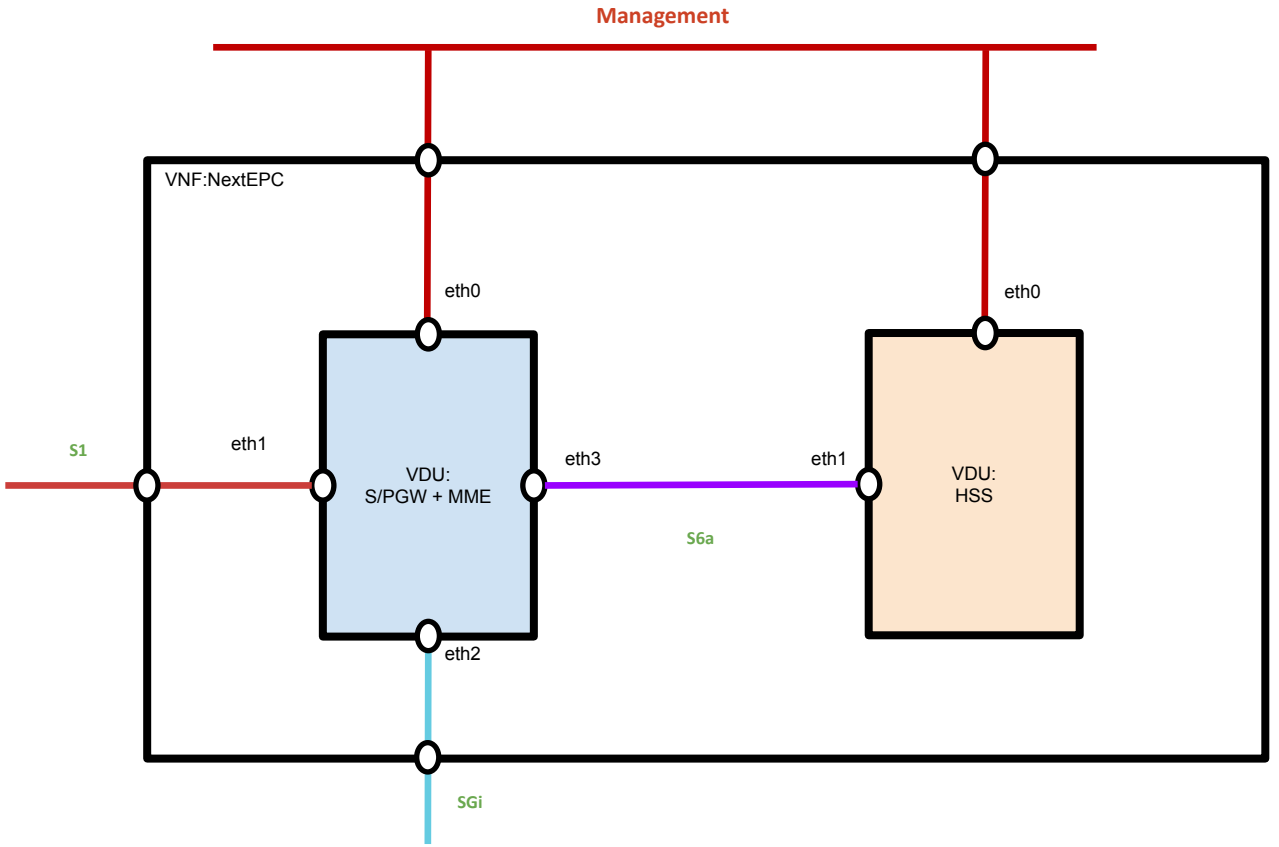


Day2 Requirements

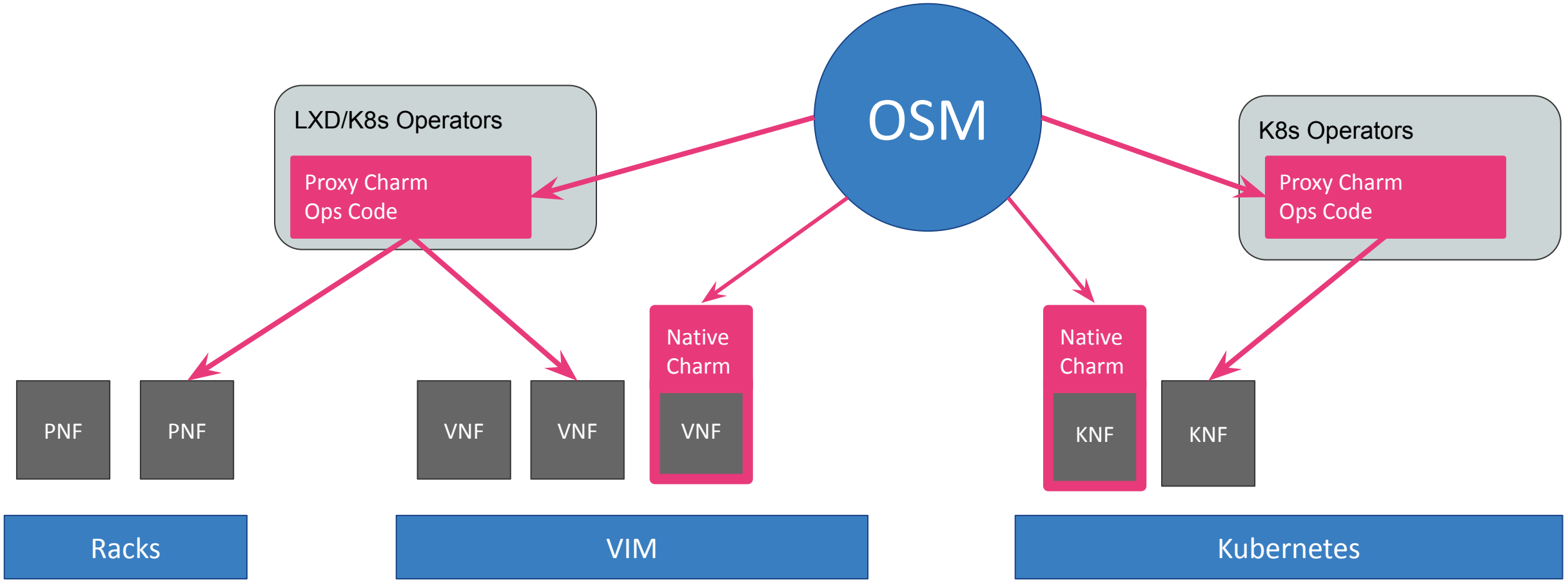
- Objectives of Day 2:
  - Re-configure the VNF so its behavior can be modified at runtime
  - Monitor its main KPIs and raise alarms
- Requirements:
  - Identify dependencies between components
  - Define possible configurations for runtime operations
  - Define key performance indicators
    - VIM: CPU, memory, network or storage usage
    - VNF/EM: Active sessions, users, application status
  - Define closed loop operations - Scaling or healing

Day2 Requirements

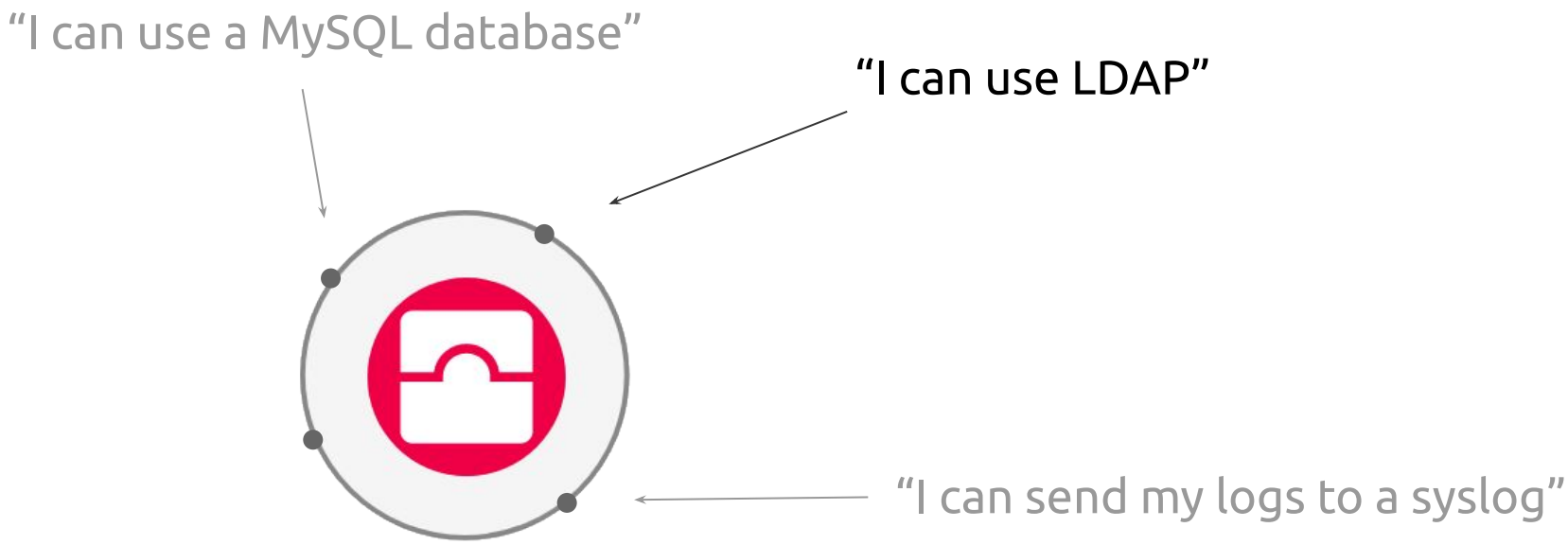
- Add or remove subscribers
- Collect CPU and Memory metrics of the VDU



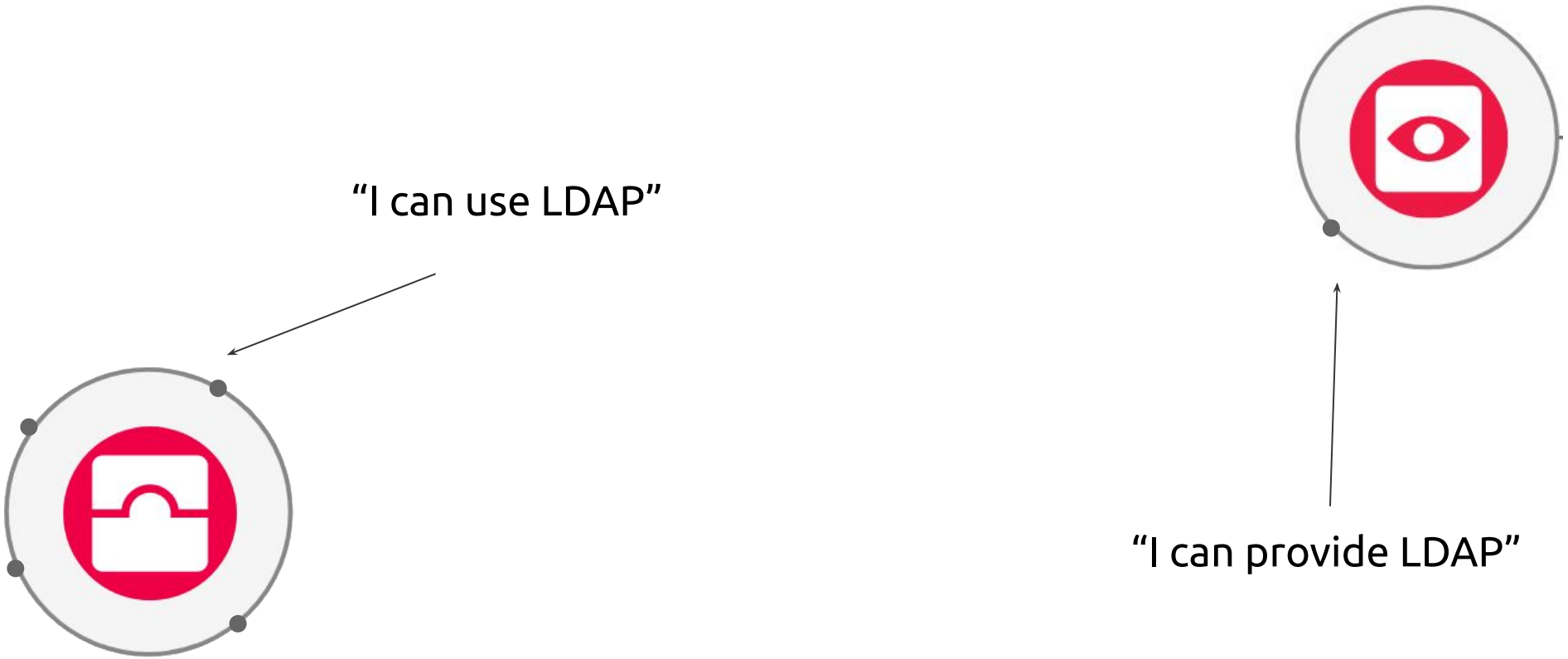
Charms in Action



Charms in Action - Integration

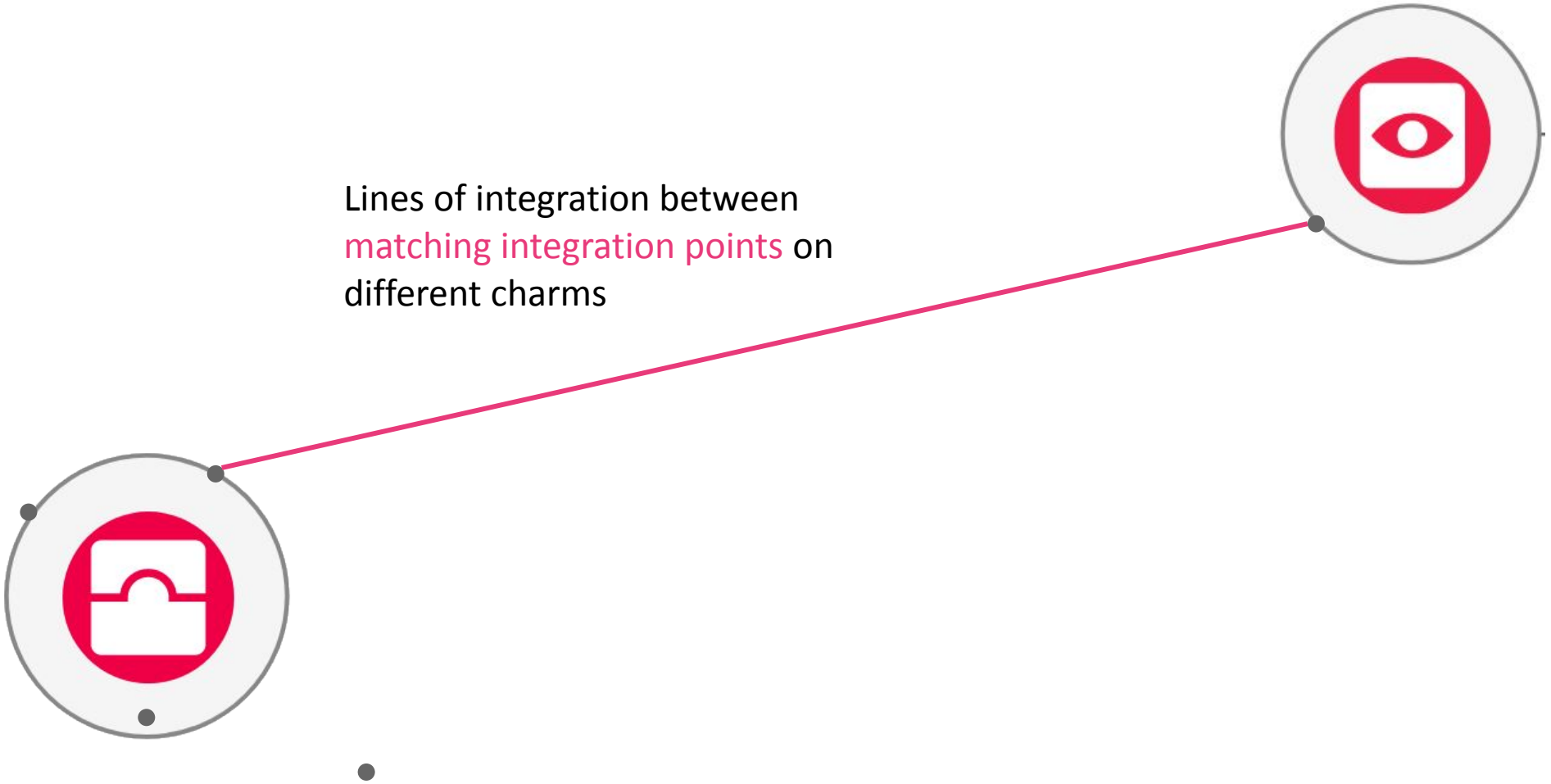


Charms in Action - Integration

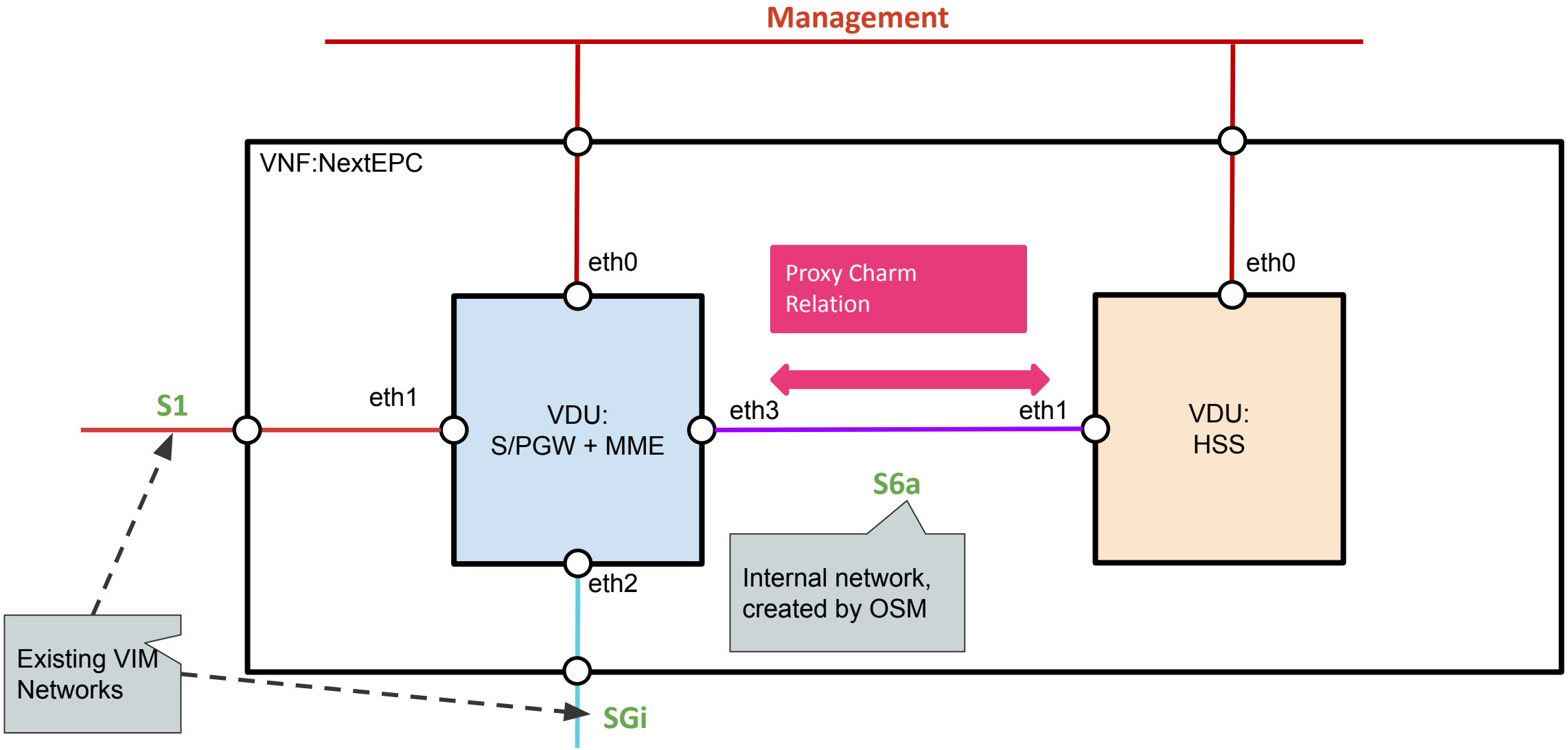


Charms in Action - Integration

Lines of integration between matching integration points on different charms



Charms in Action



Hands On

- Ensure om-packages repo is cloned

```
$ git clone --recursive
https://osm.etsi.org/gitlab/vnf-onboarding/osm-packages.git
```

- Move to Hackfest MR#13

```
$ cd osm-packages/Hackfest_Demos/OSM-MR13/
```

- Package and Onboard the packages

```
$ osm nfpkg-create vEPC_vnf

$ osm nspkg-create vEPC_ns
```

Verify Onboarding

```
$ osm nspkg-list
```

```
osm@osm:~/packages/osm-packages/Hackfest_Demos/OSM-MR13/NextEPC$ osm nsd-list
+----------+--------------------------------------+
| nsd name | id                                   |
+----------+--------------------------------------+
| vEPC-ns  | 5caa5954-e848-478d-9509-db1cfd115de0 |
+----------+--------------------------------------+
```

```
$ osm vnfpkg-list
```

```
osm@osm:~/packages/osm-packages/Hackfest_Demos/OSM-MR13/NextEPC$ osm nfpkg-list
+------------+--------------------------------------+-----------+
| nfpkg name | id                                   | desc type |
+------------+--------------------------------------+-----------+
| vEPC-vnf   | 8f8d0aed-b998-4c3e-b29e-1ec6782a6d21 | sol006    |
+------------+--------------------------------------+-----------+
```

Instantiate NextEPC Service

- Instantiate the wiki network service

```
$ osm ns-create --ns_name EPC --nsd_name vEPC-ns --config_file
params.yaml --vim_account openstack
```

- Verify the network service is deployed properly. Same can be checked via UI

```
$ osm ns-list
```

```
osm@osm:~/packages/osm-packages/Hackfest_Demos/OSM-MR13/NextEPC$ osm ns-list
+-----------------+--------------------------------------+---------------------+---------+-------------------+---------------+
| ns instance name | id                                   | date                | ns state | current operation | error details |
+-----------------+--------------------------------------+---------------------+---------+-------------------+---------------+
| EPC_1           | 7004ac93-cf81-4c7d-86ca-98de477bb6a5 | 2022-10-13T05:44:37 | READY    | IDLE (None)       | N/A           |
+-----------------+--------------------------------------+---------------------+---------+-------------------+---------------+
```

Verification of Service

- Check the juju status

```
$ juju status --relations -m <NS_ID>
```

- List VMs deployed in the Openstack VIM. There should be

```
$ openstack server list
```

- NS reaches the ‘configured’ state, ssh to the HSS and check if the HSS - MME session is established.

```
$ sudo netstat –anlp| grep 10.0.6
```

```
sctp        0      0 host-10-0-6-75.openstacklocal:3868 host-10-0-6-187.openstacklocal:37846 ESTABLISHED
```

Day 2 Action

- Test Day-2 reconfiguration operations by running the primitive that adds routes to the SPGW (no host routes allowed in this example). This can be done through the OSM UI or the CLI,

```
$ osm ns-action --vnf_name vnf-epc --vdu_id spgwmme --action_name
add-route --params '{external-prefix: "8.8.8.0/24", next-hop:
"192.168.2.1"}' EPC
```

- Verify the status of action from OSM

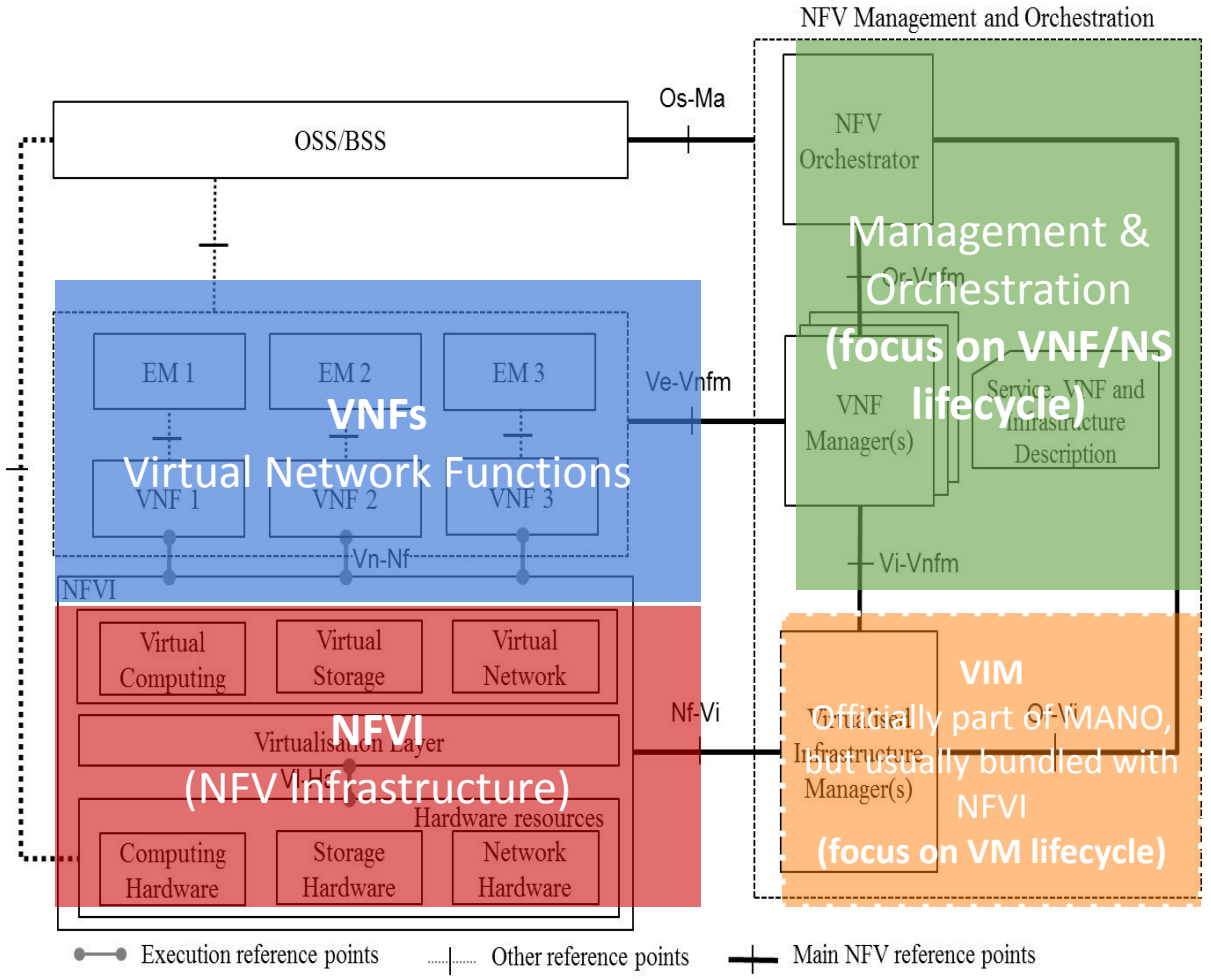
```
$ osm ns-op-list <NS_Name>
```

```
$ osm ns-op-show <day2_action_id>
```

Agenda

- Re-visiting VNF Onboarding Workflow
- Real World Use Case
- Hands on –Wiki Webserver
- Monitoring
- Auto Healing
- Alerts
- NS Policy Update
- Auto Scaling

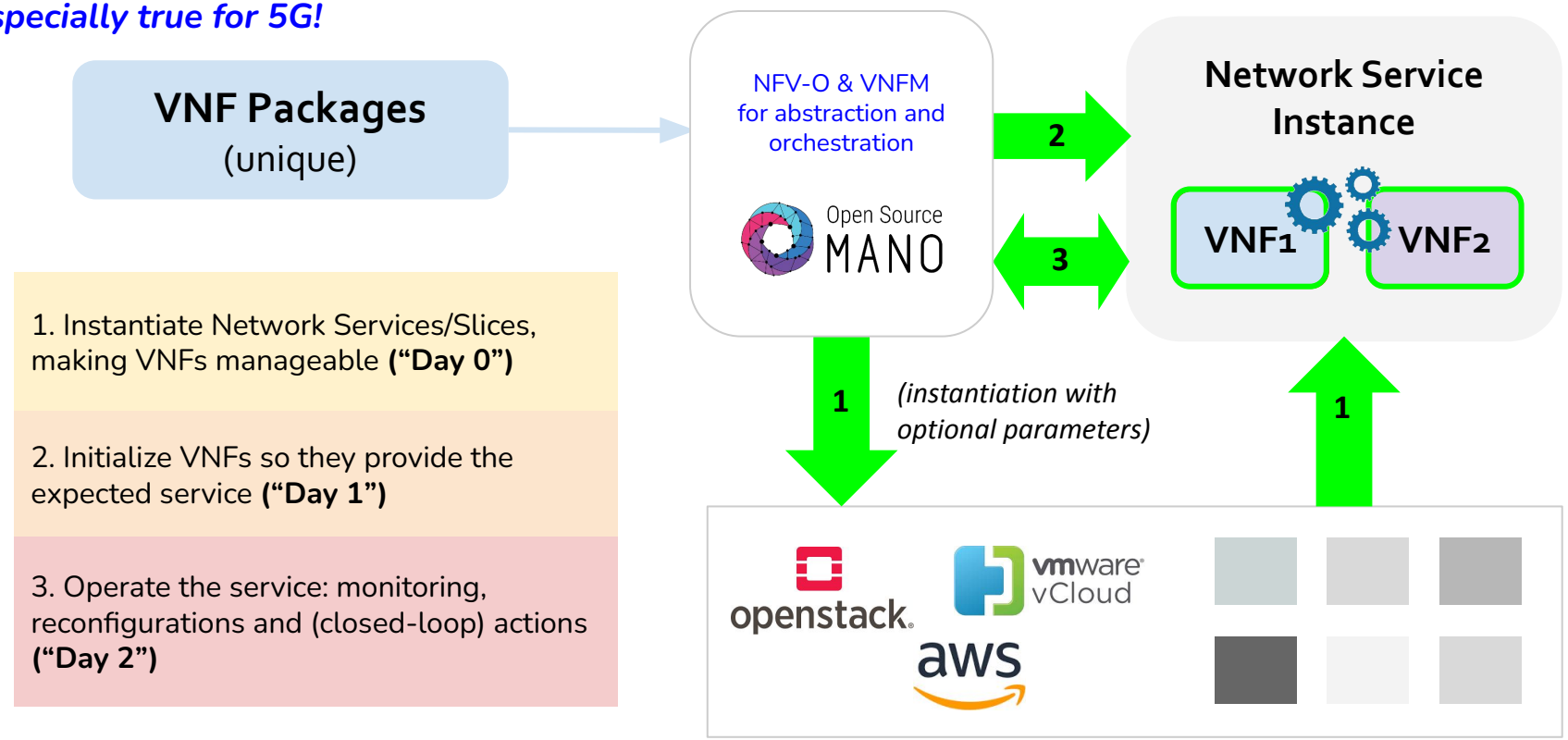
The ETSI NFV Architecture



Re-visiting VNF Onboarding workflow

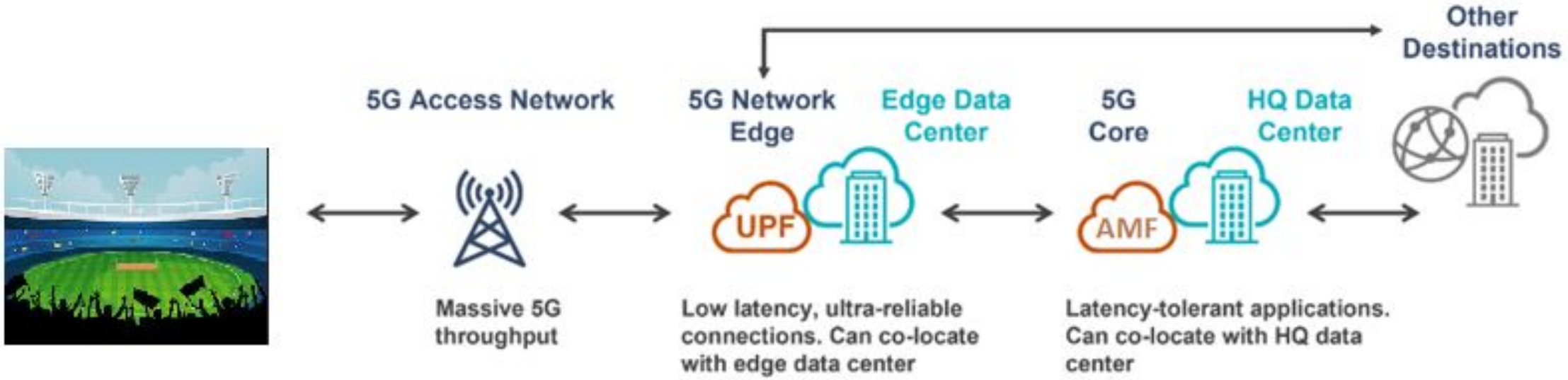
- Network Functions Virtualisation will only scale if all of the functions can be automated.

...specially true for 5G!

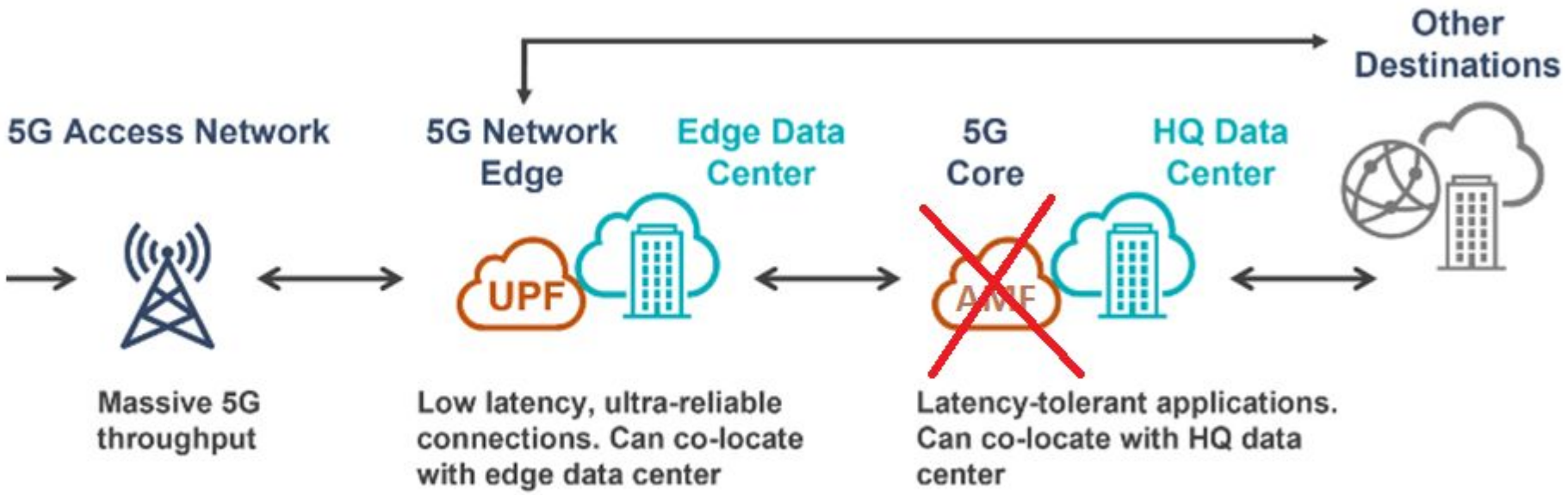


1. Instantiate Network Services/Slices, making VNFs manageable **("Day 0")**
2. Initialize VNFs so they provide the expected service **("Day 1")**
3. Operate the service: monitoring, reconfigurations and (closed-loop) actions **("Day 2")**

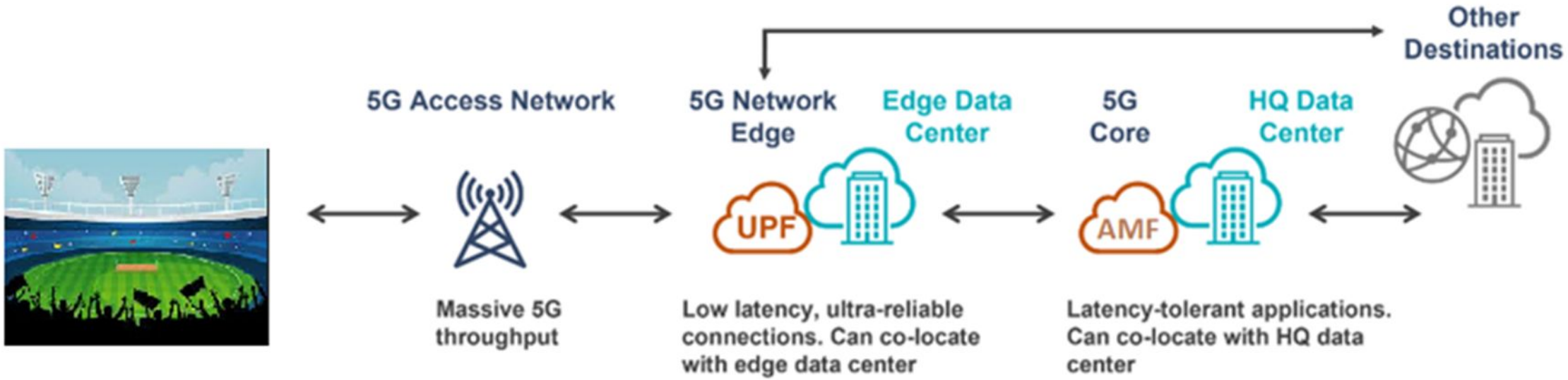
Real World Use Case



Real World Use Case



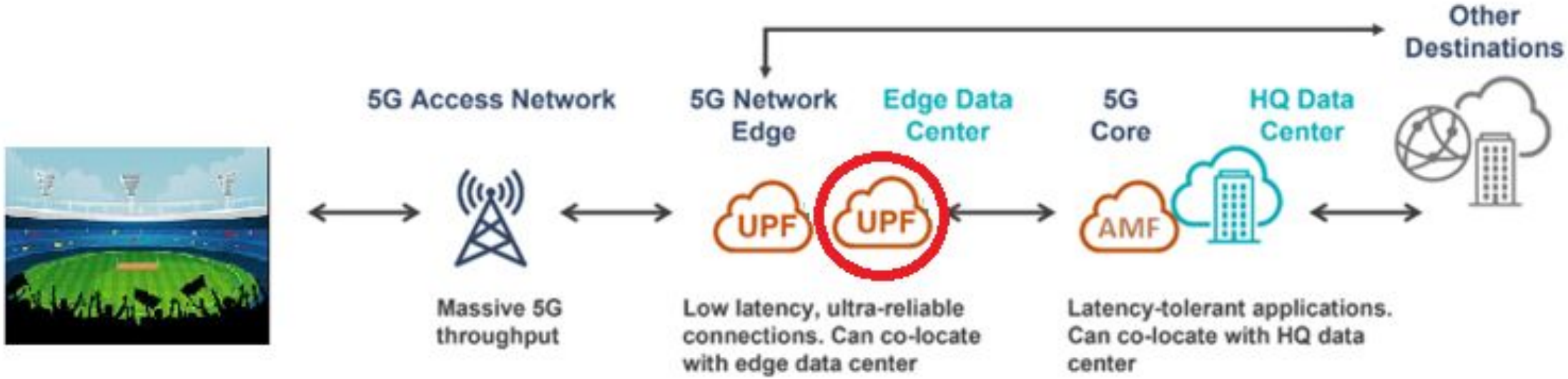
Real World Use Case



=

Scaling Policy update required!!
NS update feature to the rescue

Real World Use Case



Service Assurance

Open Source MANO

Main Components

Auxiliary/ Optional

MON

- Covers the basic uses cases, with a solid architecture to expand them easily.
- Opportunities to enhance usability.

POL

- Designed around the autoscaling use case.
- Starting to cover VNF alarms.

PLA

- Provides computation of optimal placement of NFs over VIMs
- Considers cost of compute/network

Prometheus

- OSM's TSDB for metrics since REL5
- Opportunities to enhance multi-tenancy to match new RBAC capabilities.

Grafana

- Integrates seamlessly with Prometheus.
- Great tool for enhancing usability of the system's Service Assurance

ELK

- Proved seamless integration with OSM.
- Main use case remains at log processing where stack is used.

Hands On

- Ensure om-packages repo is cloned

```
$ git clone --recursive
https://osm.etsi.org/gitlab/vnf-onboarding/osm-packages.git
```

- Move to Hackfest MR#13

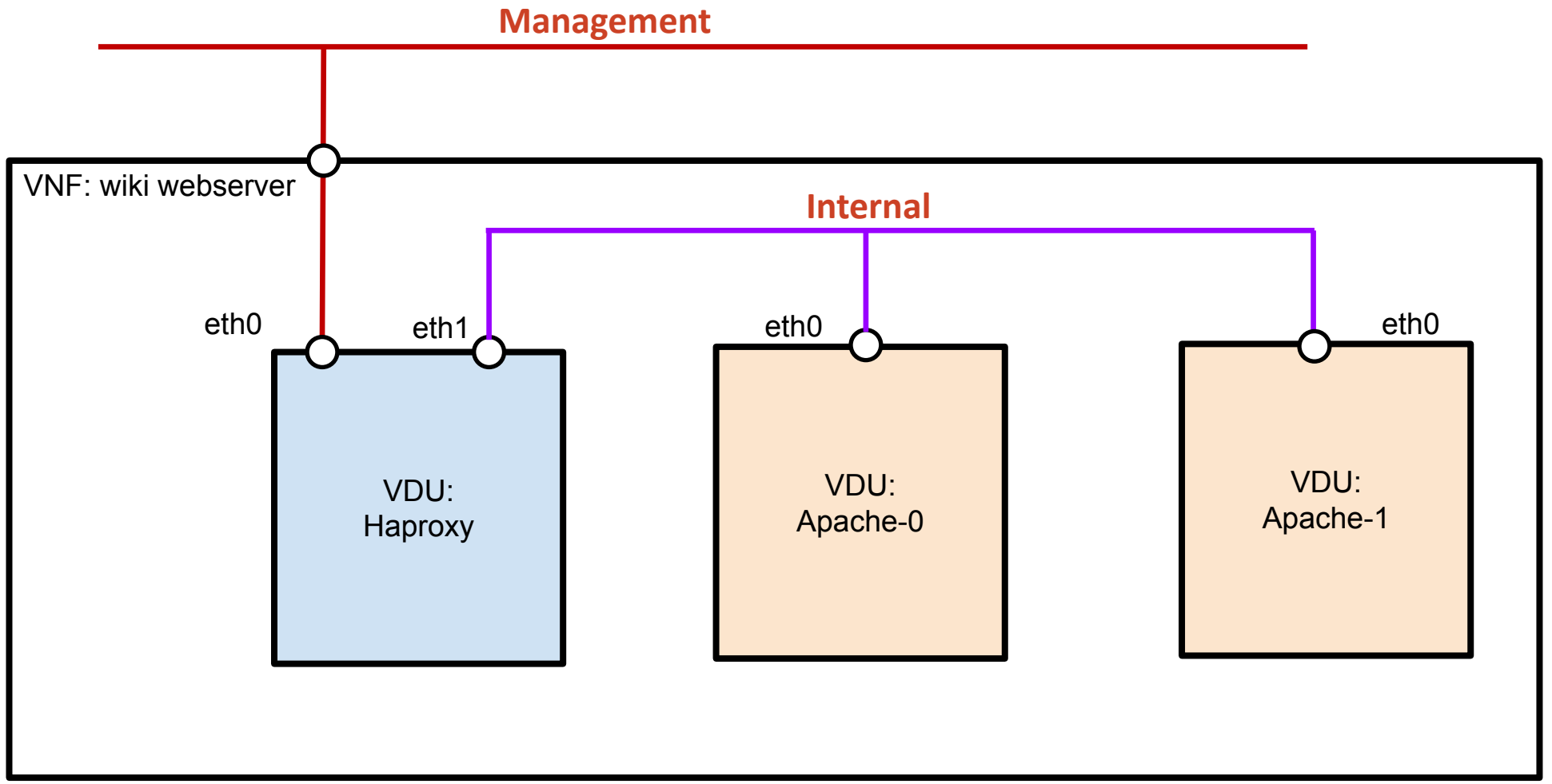
```
$ cd osm-packages/Hackfest_Demos/OSM-MR13/1.2-VNF
```

- Package and Onboard the packages

```
$ osm nfpkg-create wiki_webserver_autoscale_vnfd

$ osm nspkg-create wiki_webserver_autoscale_nsd
```

Wiki Webserver



Verify Onboarding

Open Source MANO

```
$ osm nspkg-list
```

```
osm@osm:~$ osm nspkg-list
+-------------------------------+--------------------------------------+
| nsd name                      | id                                   |
+-------------------------------+--------------------------------------+
| wiki_webserver_autoscale_ns   | fa10780a-431a-4e7a-8f22-91e265490114 |
+-------------------------------+--------------------------------------+
```

```
$ osm vnfpkg-list
```

```
osm@osm:~$ osm vnfpkg-list
+-------------------------------+--------------------------------------+----------+
| nfpkg name                    | id                                   | desc type |
+-------------------------------+--------------------------------------+----------+
| wiki_webserver_autoscale_vnf  | 328ffb4b-885b-4cab-832f-5ce6d710044d | sol006   |
+-------------------------------+--------------------------------------+----------+
```

Instantiate Wiki Webserver Service

- Instantiate the wiki network service

```
$ osm ns-create --ns_name wiki_web_service --nsd_name
wiki_webserver_autoscale_ns --vim_account openstack
```

- Verify the network service is deployed properly. Same can be checked via UI

```
$ osm ns-list
```

```
osm@osm:~$ osm ns-list
+-----------------+--------------------------------------+--------------------+---------+------------------+--------------+
| ns instance name | id                                   | date               | ns state | current operation | error details |
+-----------------+--------------------------------------+--------------------+---------+------------------+--------------+
| wiki_service_1   | c98faa54-3b3f-46a9-af61-672dcec35324 | 2022-10-08T11:52:55 | READY    | IDLE (None)       | N/A           |
+-----------------+--------------------------------------+--------------------+---------+------------------+--------------+
To get the history of all operations over a NS, run "osm ns-op-list NS_ID"
For more details on the current operation, run "osm ns-op-show OPERATION_ID"
```

Verification of Service

- List VMs deployed in the Openstack VIM. There should be

```
$ openstack server list
```

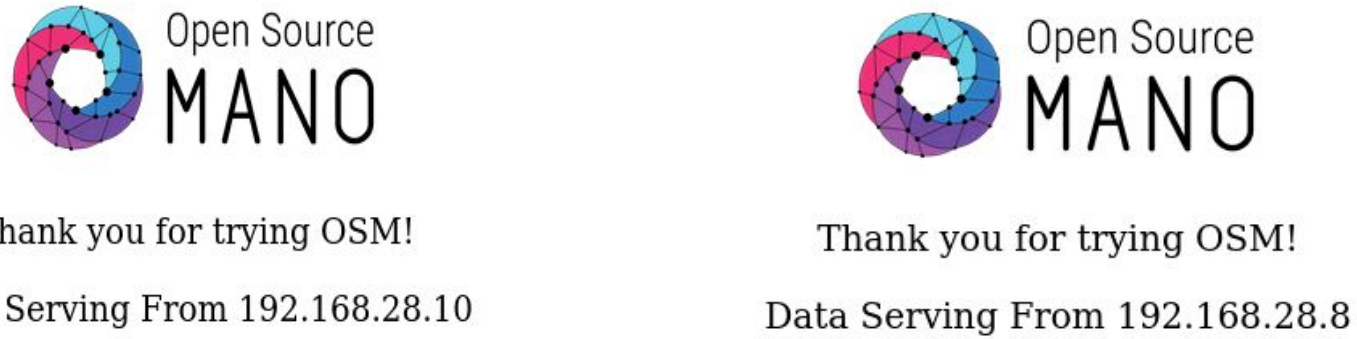
```
osm@osm:~$ openstack server list
+--------------------------------------+--------------------------------+---------+-----------------------------------------------------------------------------------+---------------+-------
-------------------------+
| ID                                   | Name                           | Status  | Networks                                                                          | Image         | Flavor
                         |
+--------------------------------------+--------------------------------+---------+-----------------------------------------------------------------------------------+---------------+-------
-------------------------+
| f28b2456-acfb-402e-ba63-93a462d26267 | wiki_service_1-vnf-apache_vdu-1  | ACTIVE  | wiki_service_1-internal_service=192.168.28.8                                      | apache_ubuntu  | hackfe
st_basic_metrics-VM-flv-3 |
| 02eaec47-bddc-4f1c-a378-151605500019 | wiki_service_1-vnf-apache_vdu-0  | ACTIVE  | wiki_service_1-internal_service=192.168.28.10                                     | apache_ubuntu  | hackfe
st_basic_metrics-VM-flv-3 |
| 408de7fc-7a94-4dae-9d94-050977736172 | wiki_service_1-vnf-haproxy_vdu-0 | ACTIVE  | osm-ext=172.21.248.197; wiki_service_1-internal_service=192.168.28.11 | haproxy_ubuntu | haprox
y_vdu-flv                |
```

- Check if haproxy is working by trying to access the URL

```
http://<HA Proxy External IP>/
http://<HA Proxy External IP>:32700/
```

Verification of Service

- The Following page should be visible



Note: Tr**oubleshooting Tips** If HA proxy is not started automatically, play the following commands from haproxy VM

```
$ service haproxy status
$ sudo service haproxy restart
```

Verification of Service

Note: "NOLB"/"DRAIN" = UP with load-balancing disabled.

haproxynode

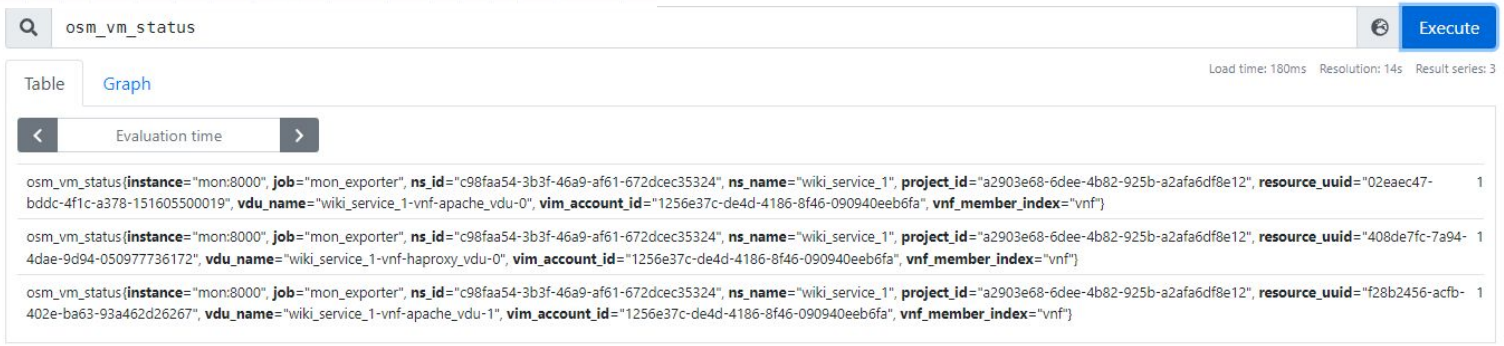
	Queue			Session rate			Sessions						Bytes		Denied		Errors		Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				0	1	-	0	2	2 000	8			4 006	43 233	0	0	0					OPEN							

backendnodes

	Queue			Session rate			Sessions						Bytes		Denied		Errors		Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
node0	0	0	-	0	1		0	1	-	4	1	4m4s	346	213		0		1	0	3	0	4m12s DOWN	L4TOUT in 2001ms	1	Y	-	1	1	4m12s	-
node1	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	0	4m12s DOWN	* L4TOUT in 2001ms	1	Y	-	1	1	4m12s	-
node2	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	0	4m11s DOWN	* L4TOUT in 2002ms	1	Y	-	1	1	4m11s	-
node3	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	0	4m11s DOWN	* L4TOUT in 2001ms	1	Y	-	1	1	4m11s	-
node4	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	0	4m11s DOWN	* L4TOUT in 2001ms	1	Y	-	1	1	4m11s	-
node5	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	0	4m11s DOWN	* L4TOUT in 2001ms	1	Y	-	1	1	4m11s	-
node6	0	0	-	0	1		0	1	-	6	6	1m28s	1 975	2 467		0		0	0	0	0	4m14s UP	L4OK in 0ms	1	Y	-	0	0	0s	-
node7	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	0	4m11s DOWN	* L4TOUT in 2000ms	1	Y	-	1	1	4m11s	-
node8	0	0	-	0	1		0	1	-	5	5	3m5s	1 685	40 365		0		0	0	0	0	4m14s UP	L4OK in 0ms	1	Y	-	0	0	0s	-
node9	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	0	4m12s DOWN	L4CON in 0ms	1	Y	-	1	1	4m12s	-
node10	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	0	4m10s DOWN	* L4TOUT in 2001ms	1	Y	-	1	1	4m10s	-
node11	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	0	4m10s DOWN	* L4TOUT in 2000ms	1	Y	-	1	1	4m10s	-
node12	0	0	-	0	0		0	0	-	0	0	?	0	0		0		0	0	0	0	4m10s DOWN	* L4TOUT in 2001ms	1	Y	-	1	1	4m10s	-
Backend	0	0		0	2		0	2	200	12	12	1m28s	4 006	43 045	0	0		1	0	3	0	4m14s UP		2	2	0		0	0s	

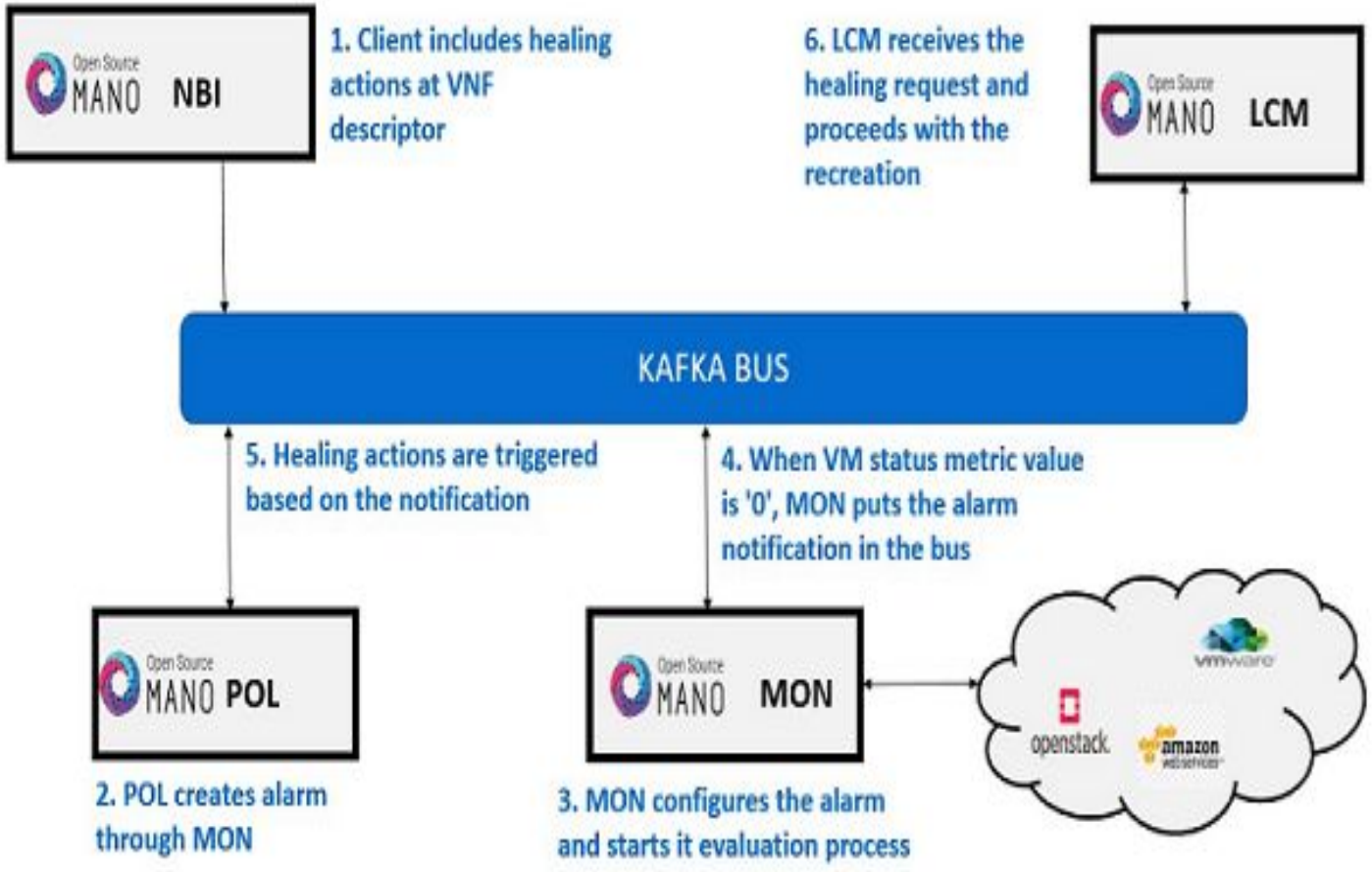
stats

	Queue			Session rate			Sessions						Bytes		Denied		Errors		Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				1	1	-	1	1	2 000	2			352	262	0	0	0					OPEN							
Backend	0	0		0	0		0	0	200	0	0	0s	352	262	0	0		0	0	0	0	4m14s UP		0	0	0		0		



Auto Healing

- Healing descriptors can be included and be tied to automatic reaction to VM metric thresholds.
- An internal alarm manager has been added to MON through the 'mon-evaluator' module, so VM metrics can trigger threshold-violation alarms when VM is in `ERROR/DELETE` state and perform healing actions.



Auto Healing – Hands on

- To replicate auto heal scenario, remove apache-VDU 1 from Openstack VIM

```
$ openstack server list

$ openstack server delete <Name of apache vdu 1>
```

- Check the status of Apache1 VM in Prometheus, the state would have moved to ‘0’

- Check from OSM for healing action taken

```
$ osm ns-op-list <NS_Name>
```

Alert Functions

Alerts

- Alarm section defined within descriptor helps to send notification to external system based on metric monitored.
- An internal alarm manager has been added to MON through the 'mon-evaluator' module, so that both VIM and VNF metrics can also trigger threshold-violation alarms and scaling actions.

NS Policy Update (Lifecycle Management)

- Policy update is a part of NS Update CHANGE_VNFPKG operation and it can be used to update all the parameters related to policies like scaling-aspect, healing-aspect, etc after a NS instance is deployed and running.
- Policy update changes are performed on a running VNF instance unless `software-version` is changed in the new revision of VNFD.

Scaling Group Descriptor – Hands on

```
scaling-aspect:
-   aspect-delta-details:
        deltas:
        -   id: apache_vdu_autoscale-delta
            vdu-delta:
            -   id: apache_vdu
    number-of-instances: 1
    id: apache_vdu_autoscale
    max-scale-level: 10
    name: apache_vdu_autoscale
    scaling-policy:
    -   cooldown-time: 180
    name: apache_cpu_util_above_threshold
    scaling-criteria:
    -   name: apache_cpu_util_above_threshold
        scale-in-relational-operation: LT
        scale-in-threshold: 20
        scale-out-relational-operation: GT
        scale-out-threshold: 40
        vnf-monitoring-param-ref: apache_vnf_cpu_util
    scaling-type: automatic
    threshold-time: 10
```

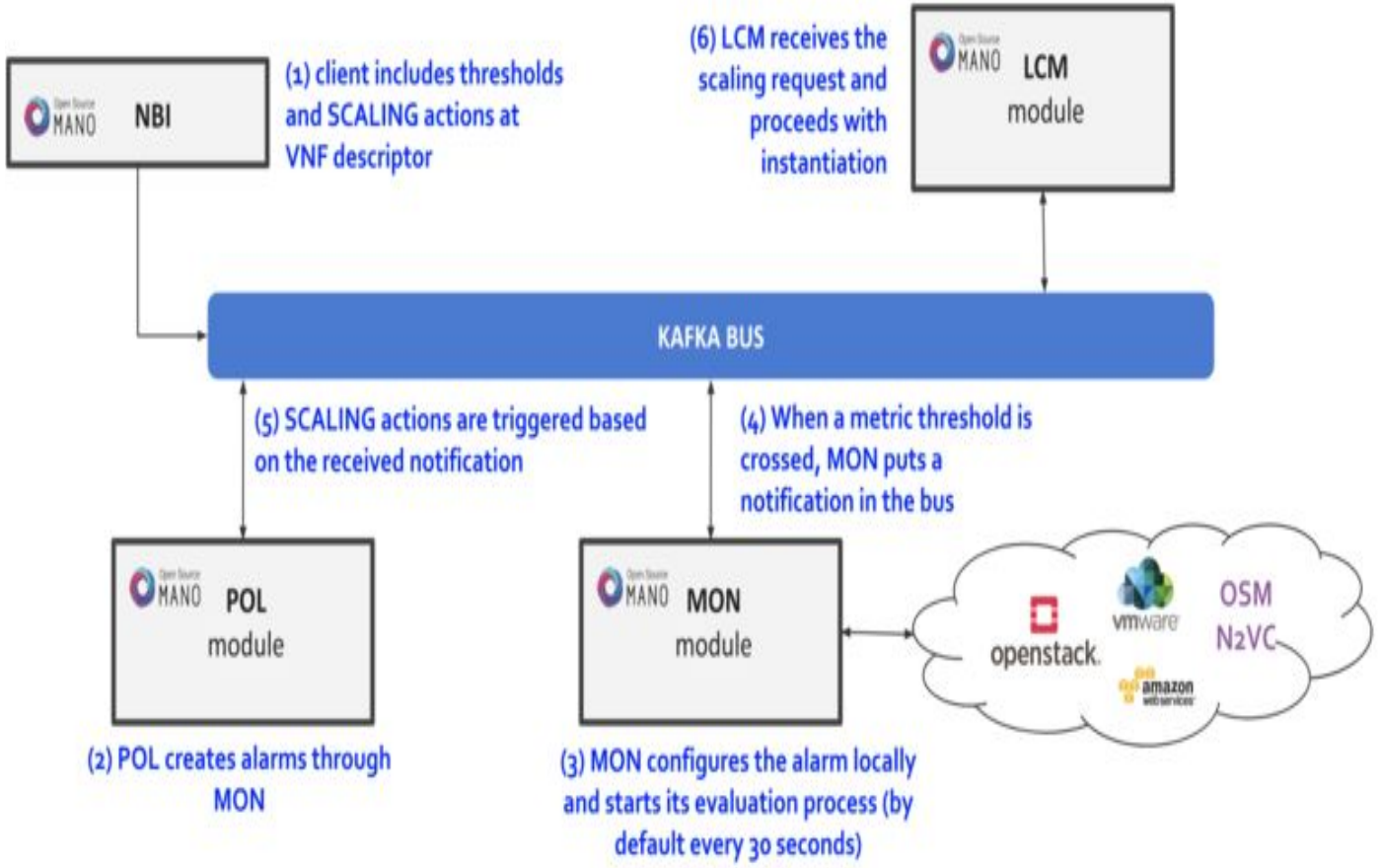
The scaling descriptor is part of a VNFD. Like the example shows, it mainly specifies:

- An existing metric to be monitored, which should be pre-defined in the monitoring-param list (monitoring-parameter).
- The thresholds to monitor (scale-in/out threshold)
- The minimum amount of scaled instances to produce (max-scale-level).
- The minimum time it should pass between scaling operations (cooldown-time).
- The VDU to be scaled and the amount of instances to scale per event.

Auto Scaling

Auto Scaling

- Auto scaling allows to automatically scale VNFs with a VDU granularity and based on any available metric.
- Scaling descriptors can be included and be tied to automatic reaction to VIM/VNF metric thresholds.
- Supported metrics are both VIM and VNF metrics.



Let’s play with wiki service!

Open Source MANO

- Increase http traffic to the Wiki Web service

```
$ sudo apt install apache2-utils

$ ab -n 10000000 -c 50 http://<Haproxy-IP>/test.php
```

- Monitor traffic in Prometheus and Grafana
- Check the autoscale status from OSM

```
$ osm ns-op-list <NS_Name>
```

Verification of Auto Scaling – OSM & VIM

- Check the scaled out instances in Openstack VIM

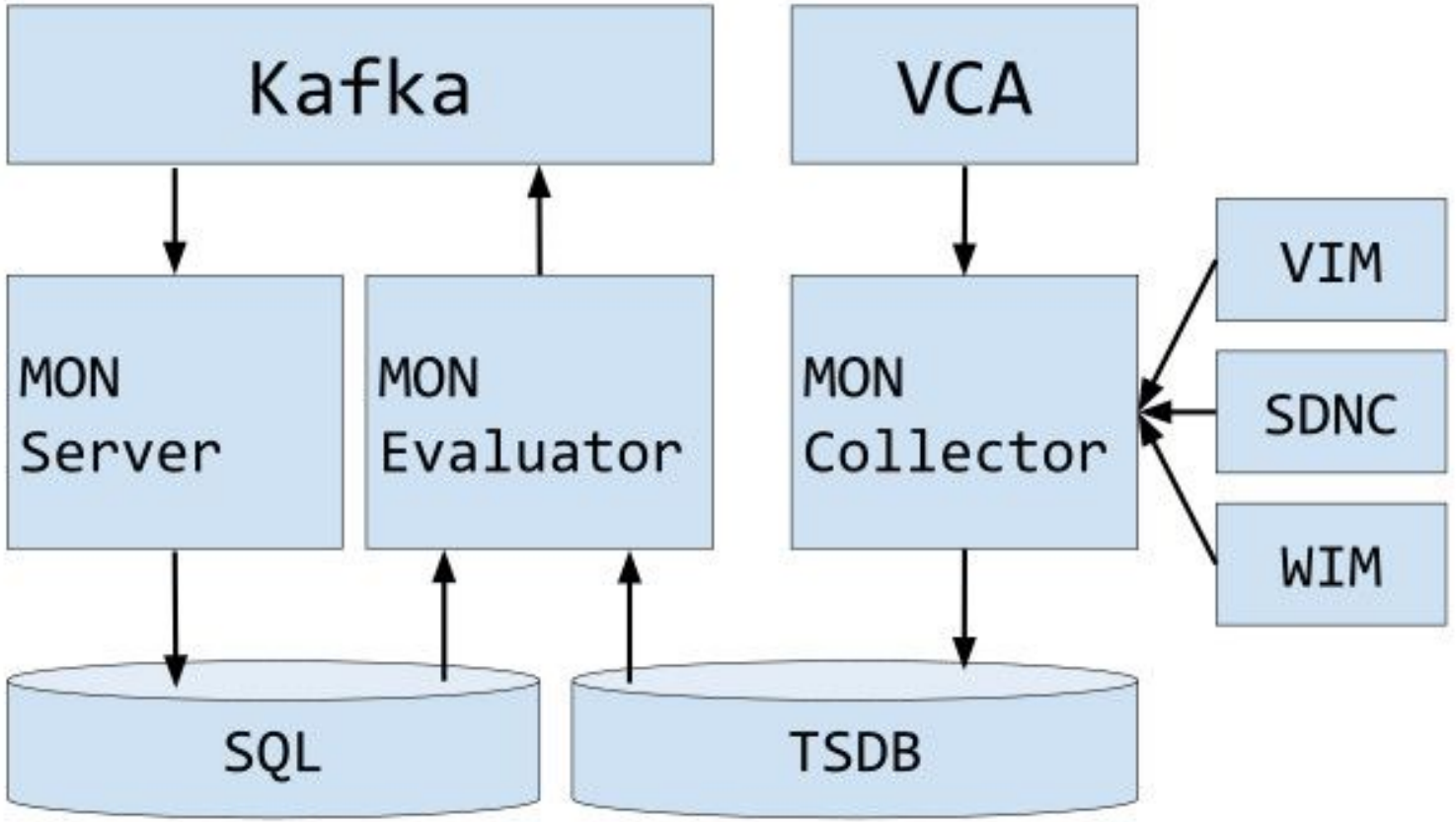
```
$ openstack server list
```

- Stop the traffic generator to see the scale in trigger

```
$ osm ns-op-list <NS_Name>
```

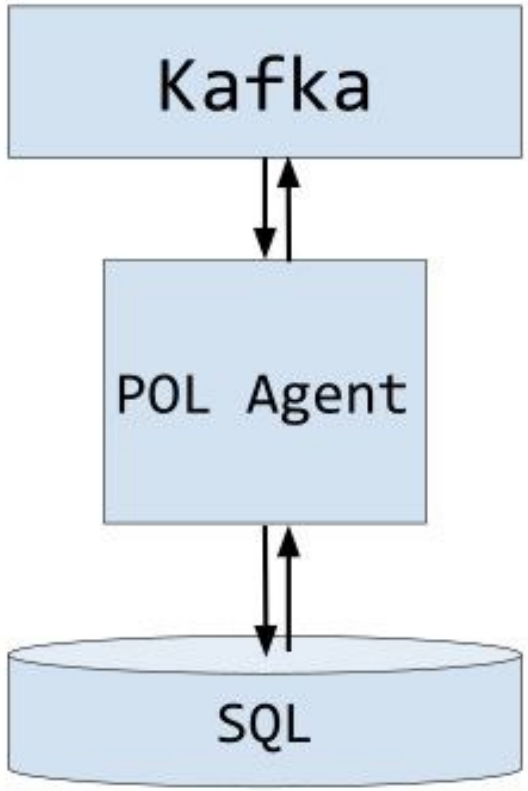
MON Architecture

Formal documentation: https://osm.etsi.org/gitlab/osm-architecture/osm-arch-doc/blob/master/04-mon.md



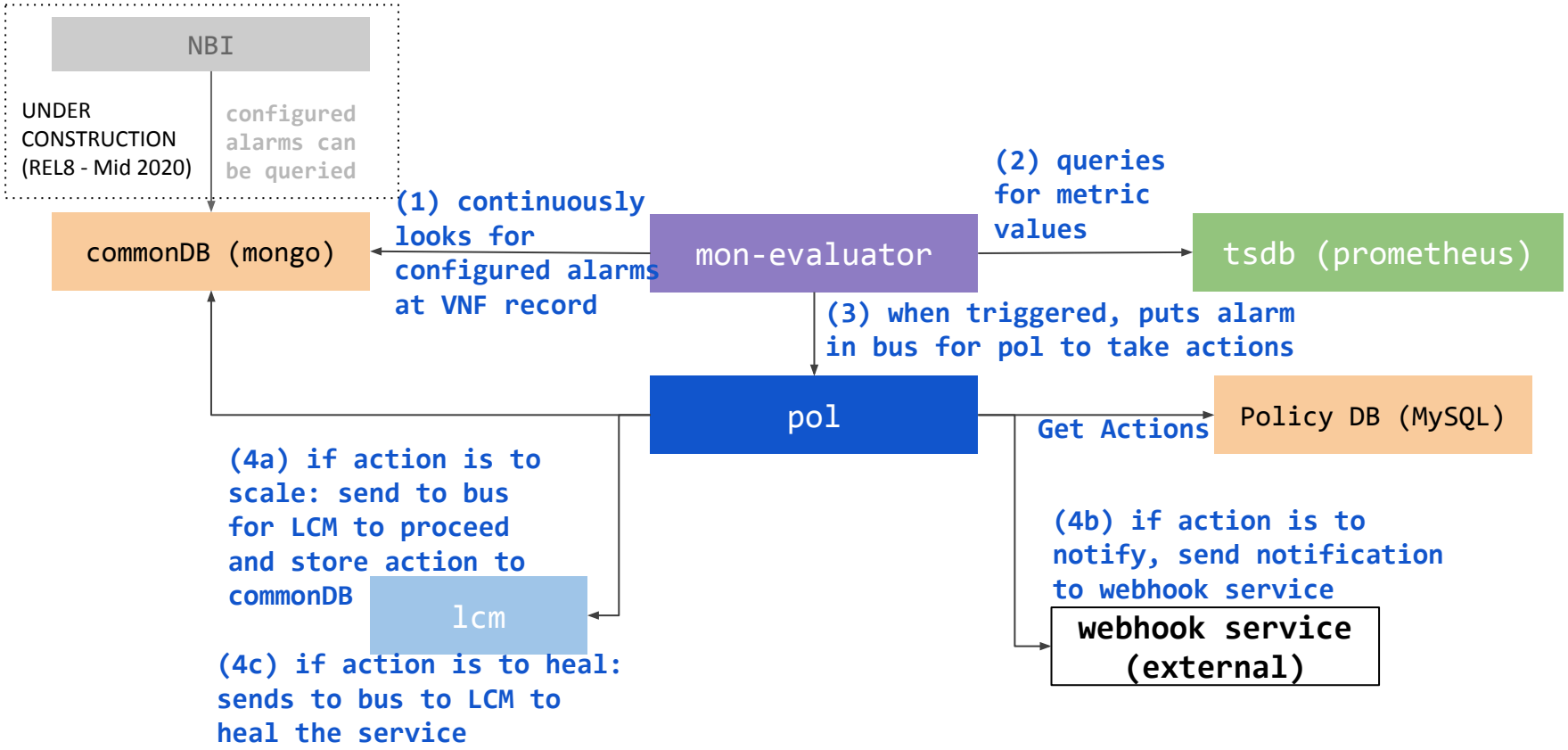
POL Architecture

Formal documentation: https://osm.etsi.org/gitlab/osm-architecture/osm-arch-doc/blob/master/05-pol.md



Auto Healing, Auto Scaling & Alarms

When configuring alarms associated to scaling actions or just webhook notifications (through the VNFD), the following components interact.



Try OSM…

Community installer

```
wget
https://osm-download.etsi.org/ftp/osm-12.0-twelve/install_osm.sh
chmod +x install_osm.sh
  ./install_osm.sh
```

Charmed installer

```
wget
https://osm-download.etsi.org/ftp/osm-12.0-twelve/install_osm.sh
chmod +x install_osm.sh
  ./install_osm.sh --charmed
```

Join us!

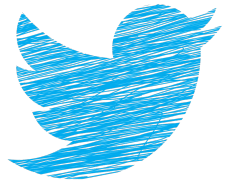
ETSI members, non-members,

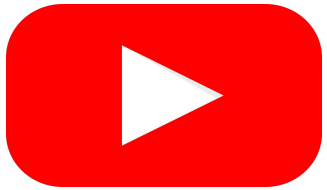
individual developers and users.

Learn how to join

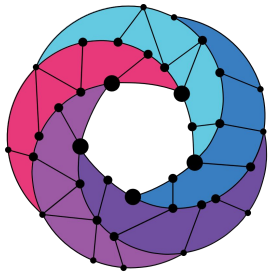
Open Source MANO

Follow us!





OpenSourceMANO



Open Source
MANO
by ETSI

Thank You!