

Open Source MANO

OSM MR Hackfest – Hack 1 OSM Architecture & Installation

Gianpietro Lavado (Whitestack)
Cyndi Alarcón (Whitestack)
Francisco Rodriguez (Indra)



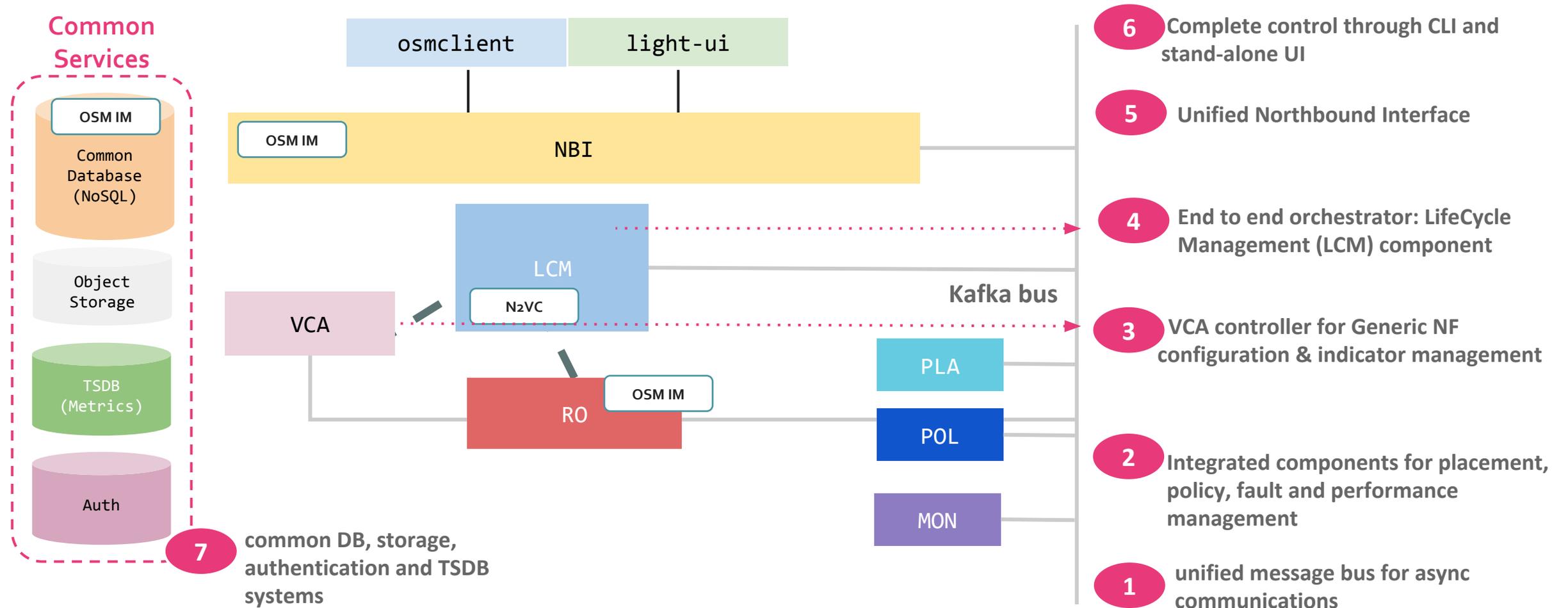


Open Source
MANO

OSM Architecture Review

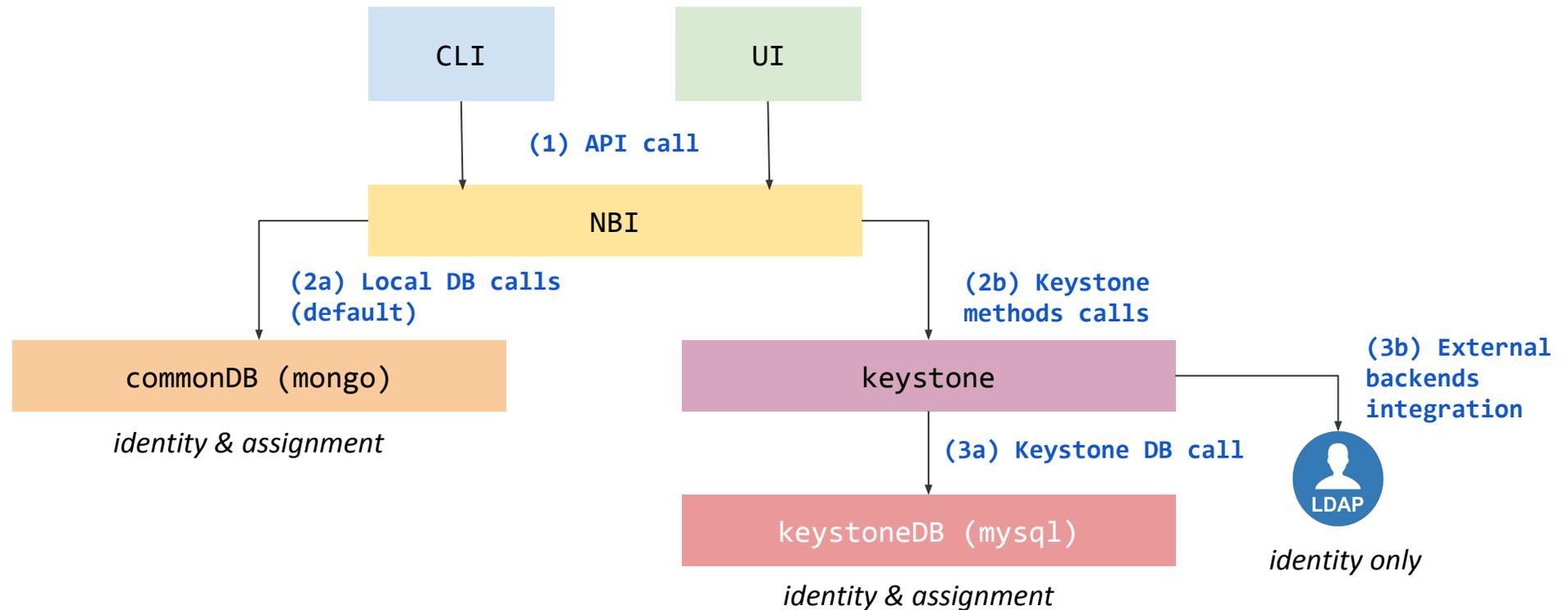


OSM Architecture overview



Identity & Assignment Operations

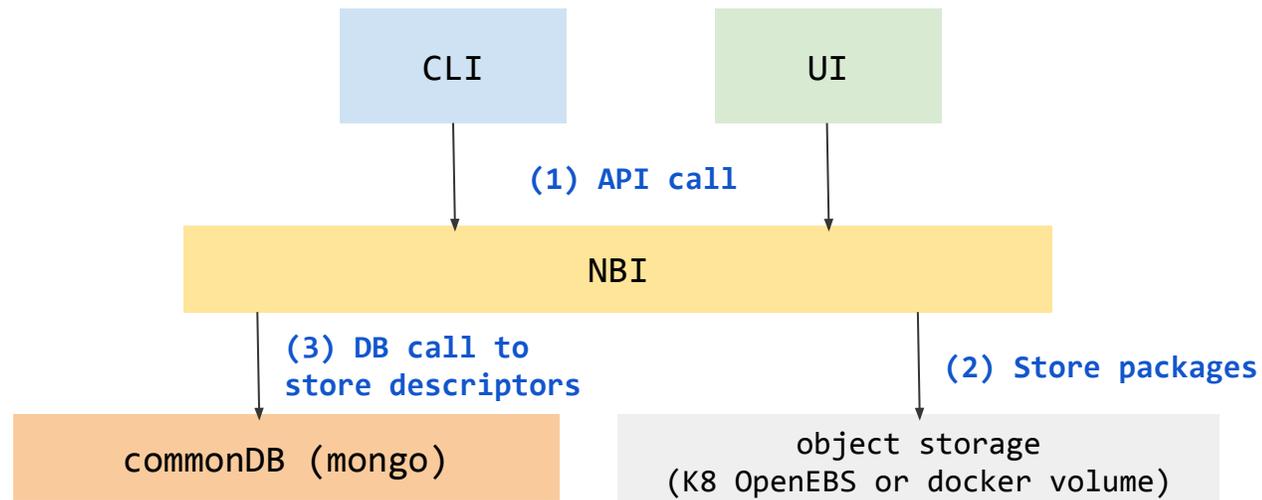
When dealing with the creation, modification or deletion of users, projects and roles, the interacting components vary according to the selected backend.



Uploading Packages

When reading, uploading, modifying and deleting a Network Slice Template, Network Service Package or VNF Package, the following components interact.

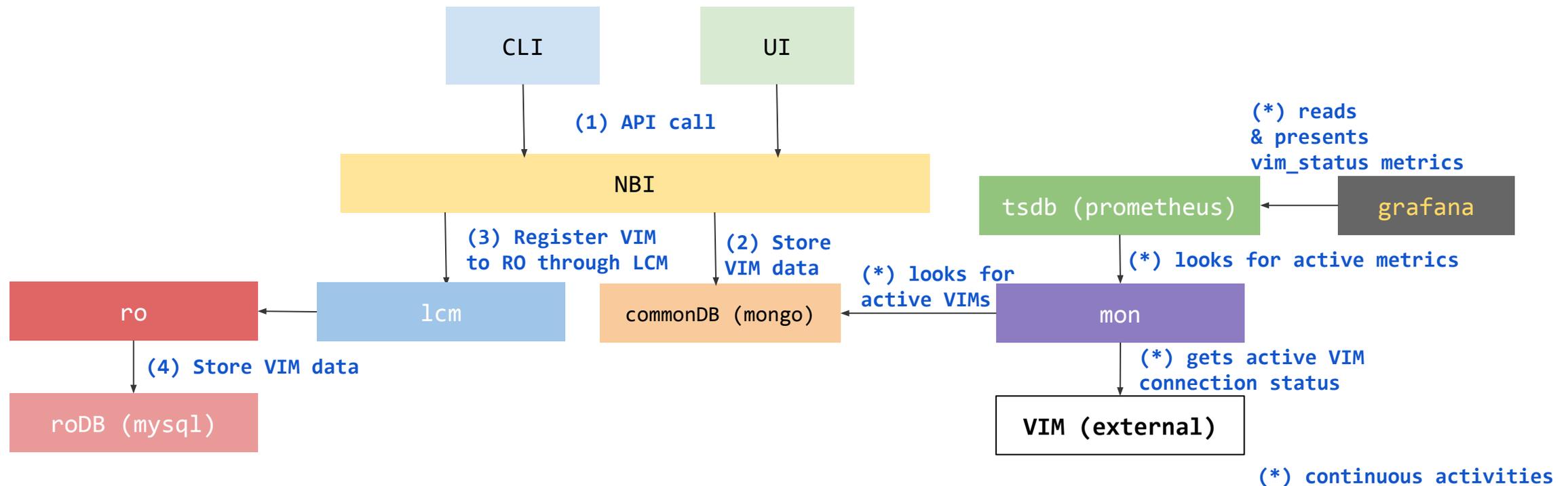
CLI Example: `osm vnfpkg-create myvnfpackage.tar.gz`



Adding VIM/SDNC Sessions

When registering new sessions with VIMs or SDN Controllers, the following components interact.

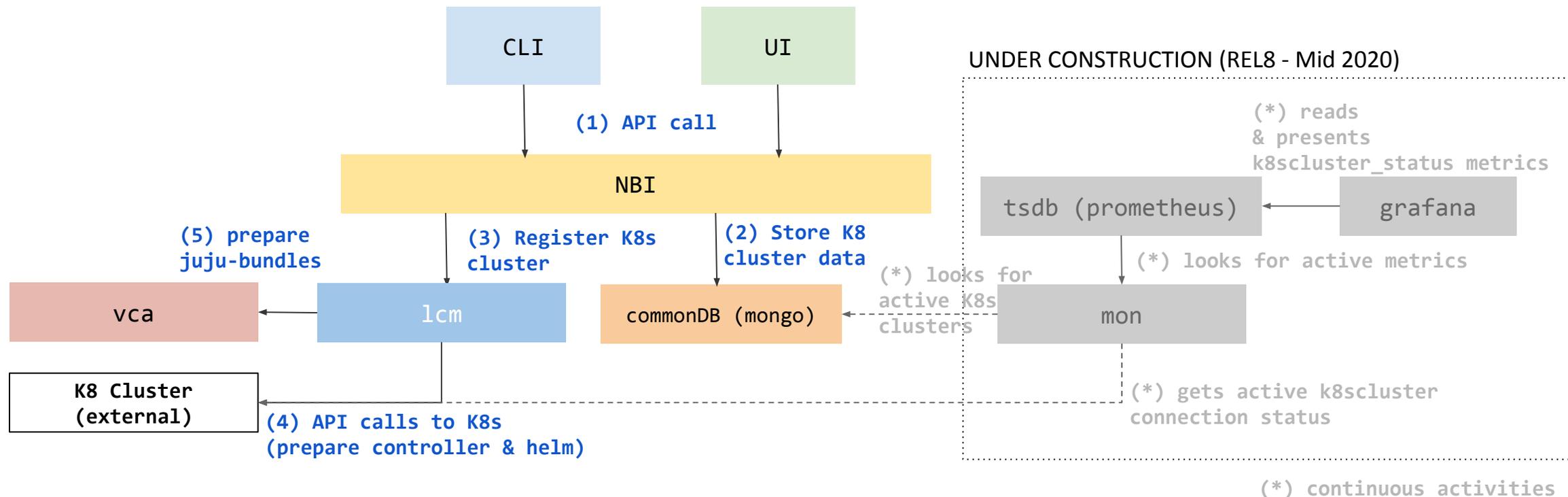
CLI Example: `osm vim-create --name myVIM --user myuser --password myprecious --auth_url http://172.21.7.5:5000/v3 --tenant mytenant --account_type openstack`



Adding a K8 Cluster

When registering new sessions with Kubernetes clusters, the following components interact.

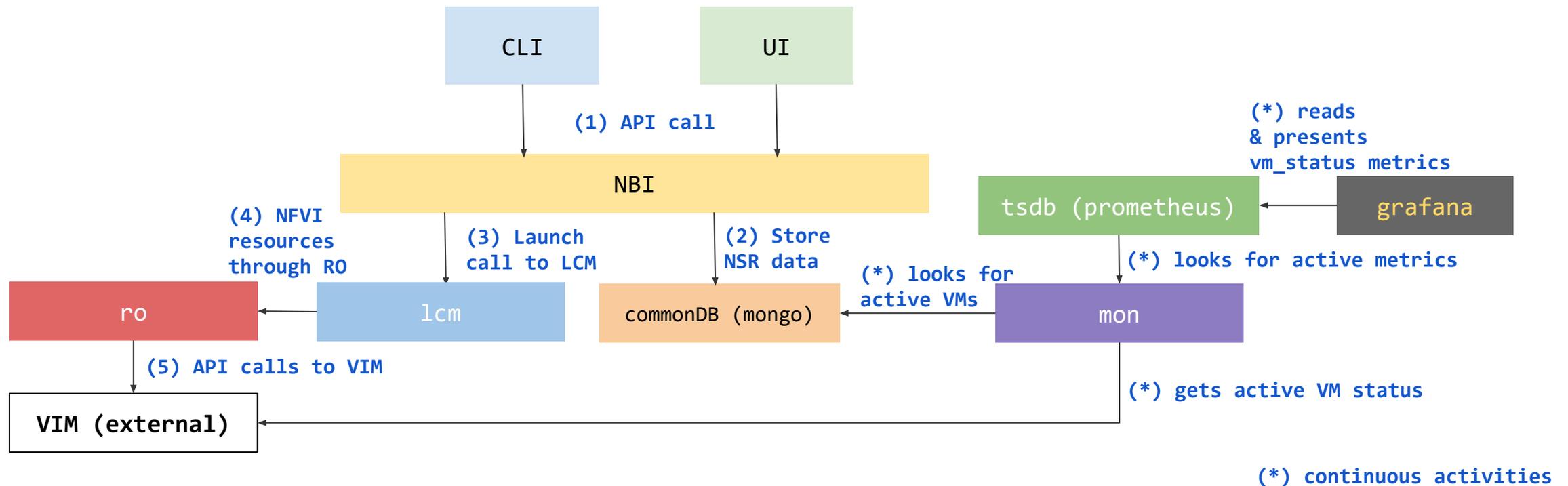
CLI Example: `osm k8scluster-add --creds myCredentials.yaml --version '1.15' --vim myVIM --description "My K8s cluster" --k8s-nets '{"net1": "myVIMnet"}' myK8Cluster`



VNF Instantiation

When launching a new instance of a Network Service or Slice Instance (n x VNFs), the following components interact.

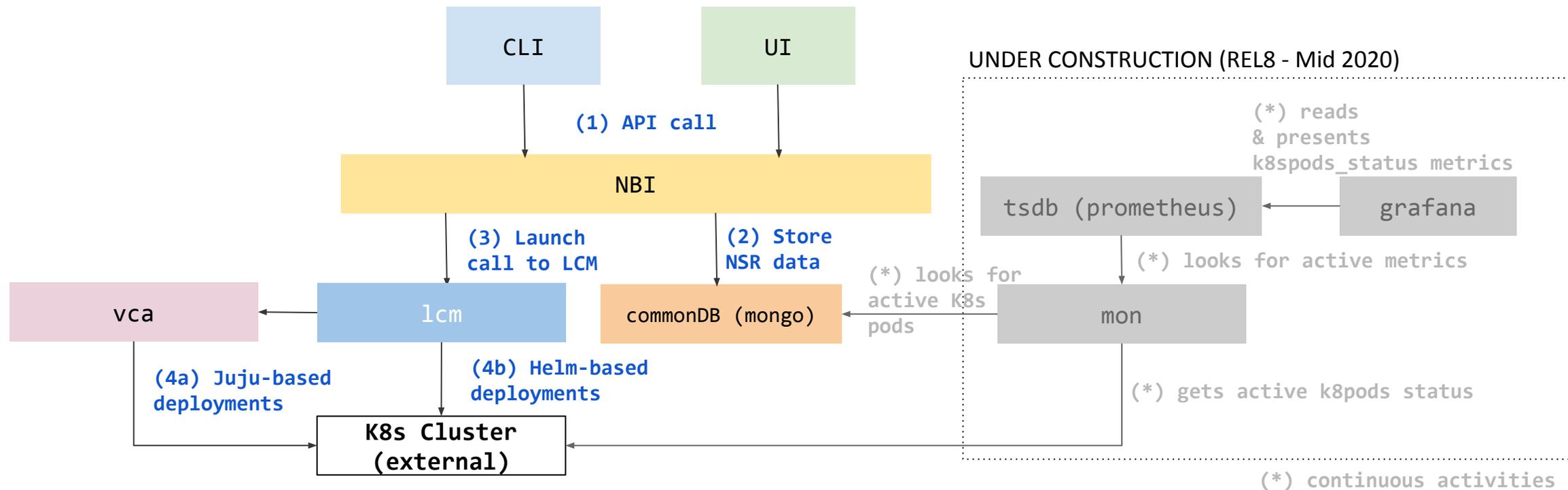
CLI Example: `osm ns-create --ns_name myNS --nsd_name myNSD --vim_account myVIM`



KNF Instantiation

When launching a new instance of a Network Service or Slice Instance (n x VNFs), the following components interact.

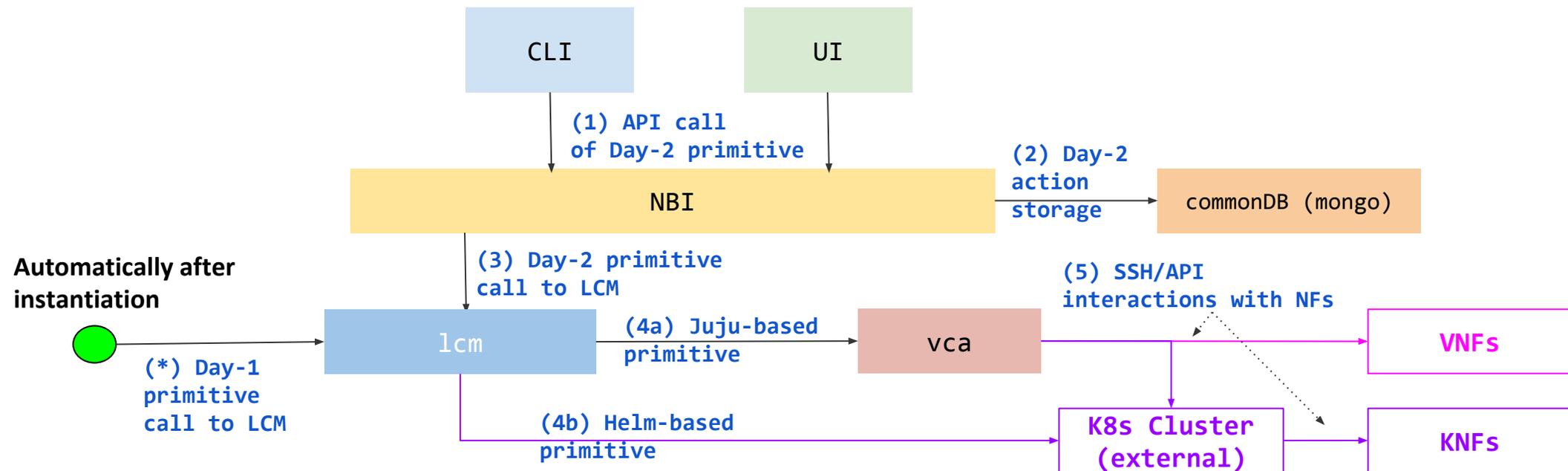
CLI Example: `osm ns-create --ns_name myNS --nsd_name myNSD --vim_account myVIM`



Instantiating with Primitives

When launching a new instance of a Network Service or Slice Instance (n x VNFs), with Day-1/2 automation, direct interaction with the NF is needed, so the following components interact.

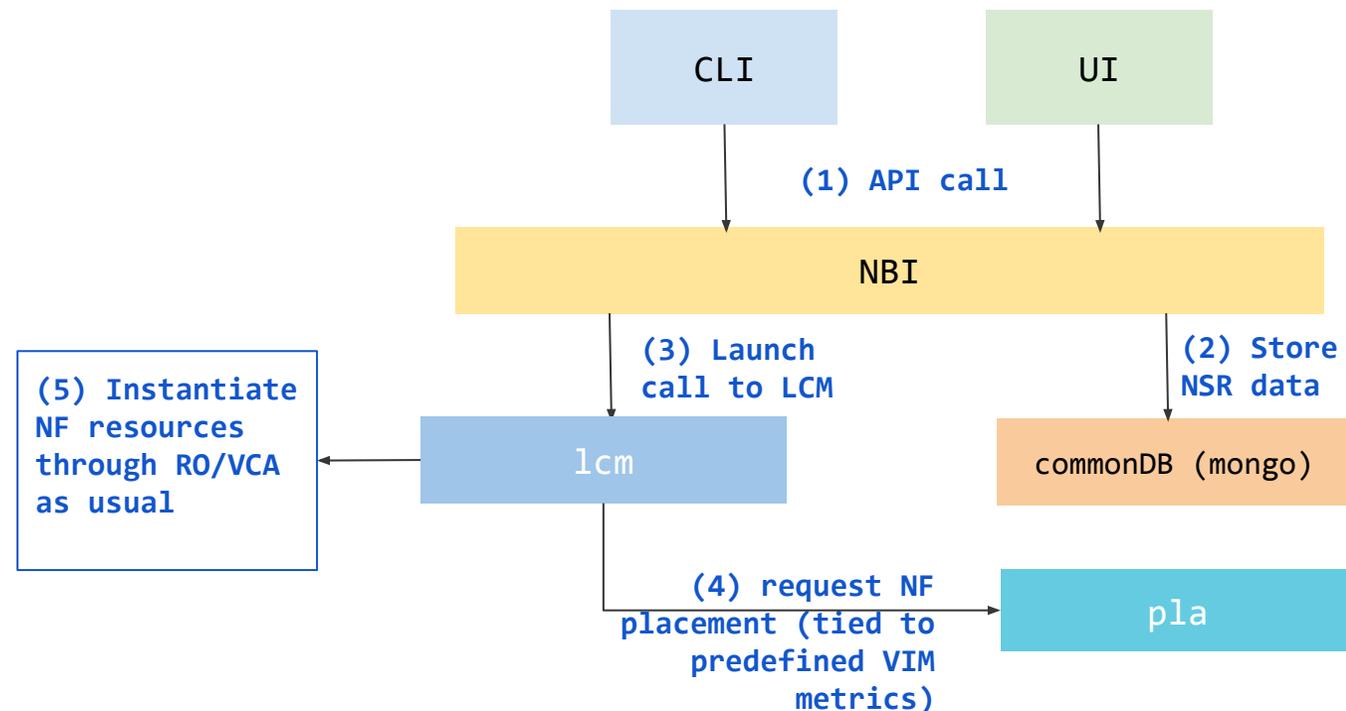
CLI Example of Day-2 primitive: `osm ns-action myNS --vnf_name 1 --action_name myAction`



Instantiating with Placement

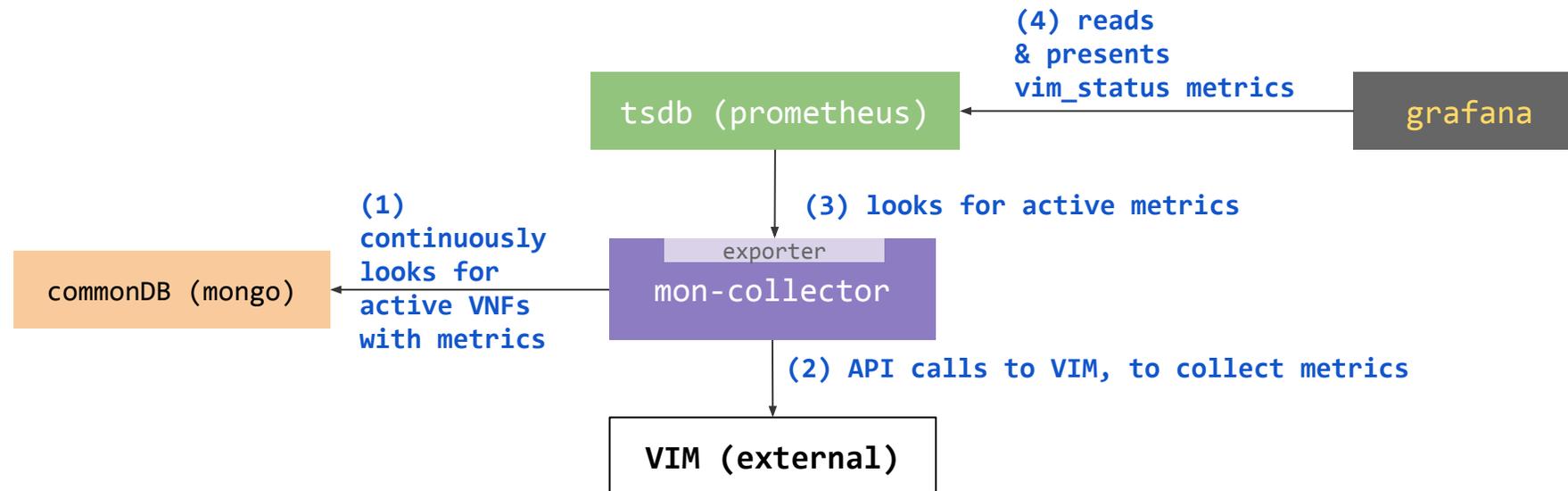
When launching a new instance of a Network Service or Slice Instance (n x VNFs), with placement support, the following components interact.

CLI Example: `osm ns-create --ns_name myNS --nsd_name myNSD --vim_account myVIM --config '{placement-engine: PLA, placement-constraints: {...}}'`



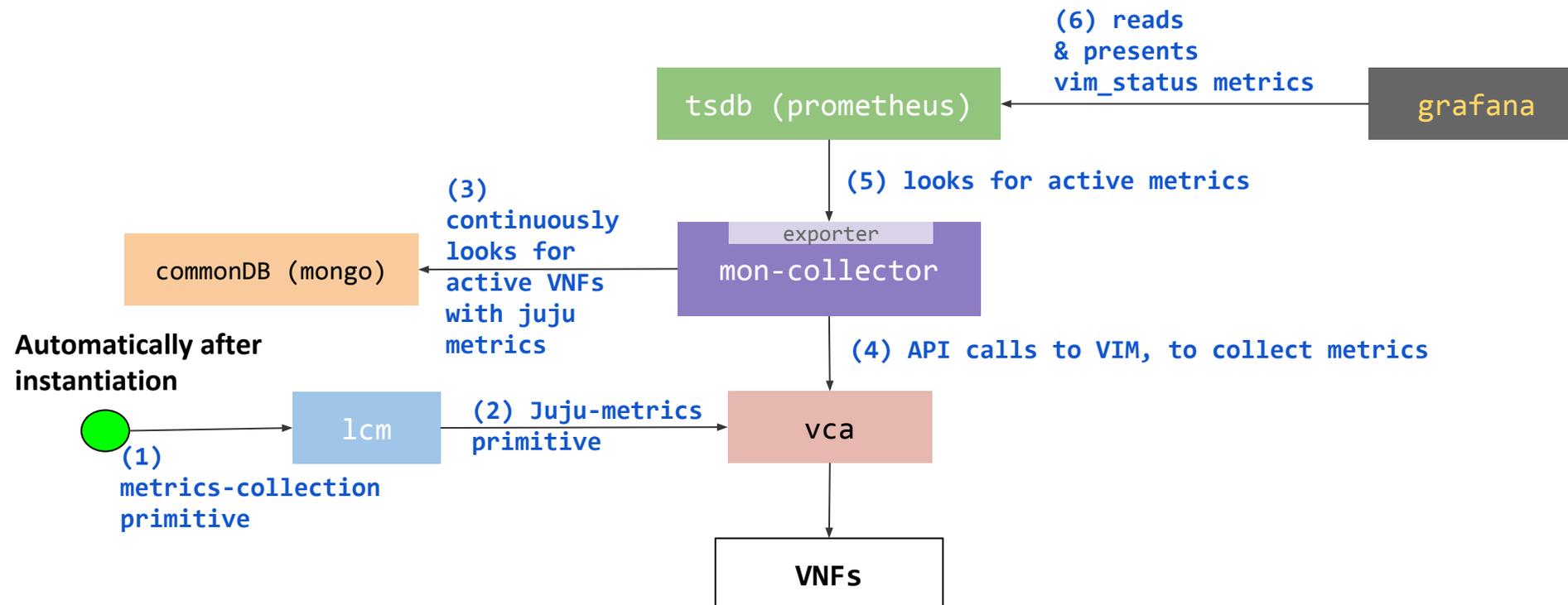
Collecting VNF Metrics (NFVI)

When launching a new instance of a Network Service or Slice Instance (n x VNFs) which is described with the collection of VNF Metrics that come from infrastructure (NFVI), the following components interact.



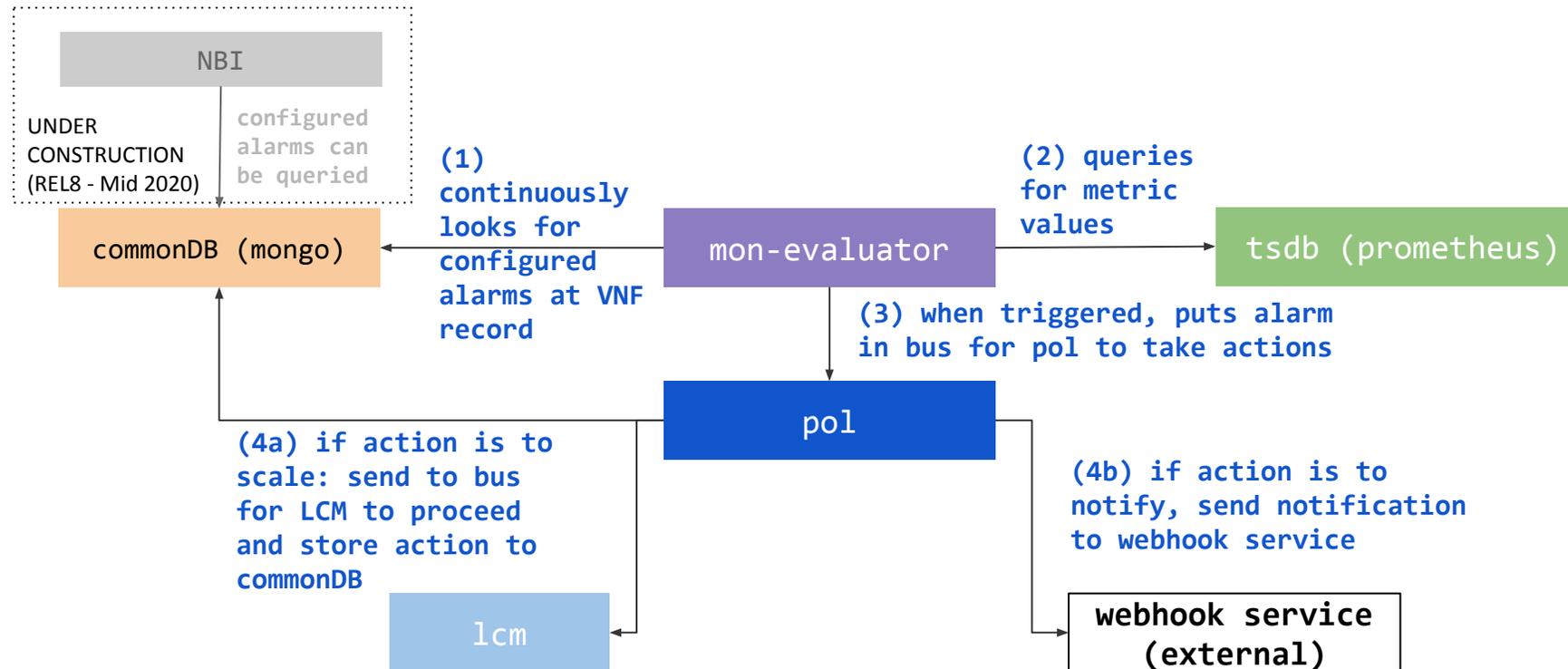
Collecting VNF Metrics (VNF)

When launching a new instance of a Network Service or Slice Instance (n x VNFs) which is described with the collection of VNF Metrics that come from the VNF itself, the following components interact.



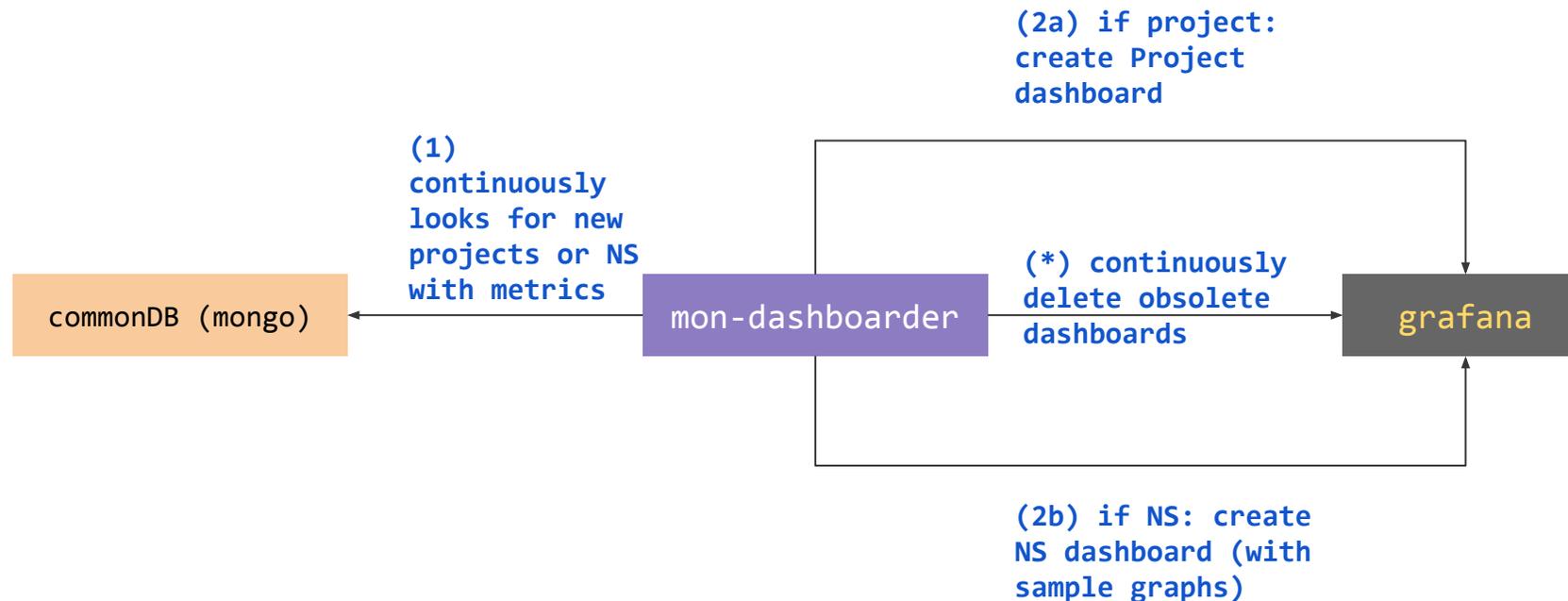
Alarms & AutoScaling

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



Automatic Dashboards

When creating Projects or Network Services, Grafana dashboards are created automatically and the following elements interact.

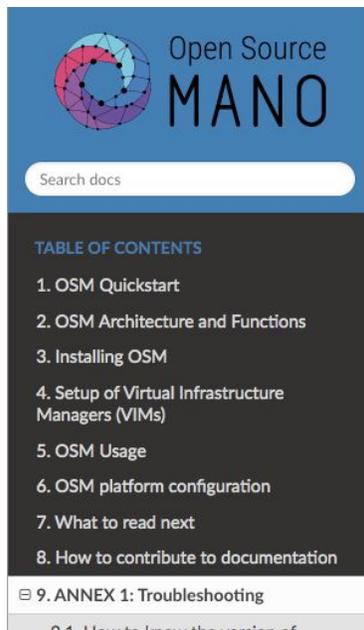


A general approach for OSM Troubleshooting is to first look for error messages in “show” commands, as in:

```
osm ns-show [ns]
osm vim-show [vim]
```

Besides that, knowing which components interact for each operation, you can troubleshoot by looking at the logs of each component. All troubleshooting tips are being documented in the user guide, here:

<https://osm.etsi.org/docs/user-guide/09-troubleshooting.html>



Open Source
MANO

Search docs

TABLE OF CONTENTS

- 1. OSM Quickstart
- 2. OSM Architecture and Functions
- 3. Installing OSM
- 4. Setup of Virtual Infrastructure Managers (VIMs)
- 5. OSM Usage
- 6. OSM platform configuration
- 7. What to read next
- 8. How to contribute to documentation
- 9. ANNEX 1: Troubleshooting

Docs » 9. ANNEX 1: Troubleshooting

[View page source](#)

9. ANNEX 1: Troubleshooting

9.1. How to know the version of your current OSM installation

Run the following command to know the version of OSM client and OSM NBI:

```
osm version
```

In some circumstances, it could be useful to search the `osm-devops` package installed in your system, since `osm-devops` is the package used to drive installations:

```
dpkg -l osm-devops
```

ii	Name	Version	Architecture	Description
ii	osm-devops	7.0.0-1	all	



Open Source
MANO

OSM Installation methods



OSM Installation methods

1. OSM can be installed in a single server or VM with the following requirements:

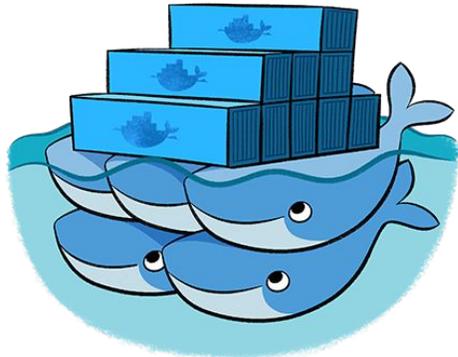
	CPU	RAM	DISK	NIC	Internet	SO
MINIMUM	2	4GB	20GB	1	Yes	Ubuntu18.04 (64-bit variant required)
RECOMMENDED	2	8GB	80GB	1	Yes	Ubuntu18.04 (64-bit variant required)

2. Once you have prepared the host with the previous requirements, all you need to do is:

```
wget https://osm-download.etsi.org/ftp/osm-7.0-seven/install\_osm.sh  
chmod +x install_osm.sh
```

OSM Installation methods

OSM R7 can be installed using these main options:



Docker Swarm

```
./install_osm.sh
```

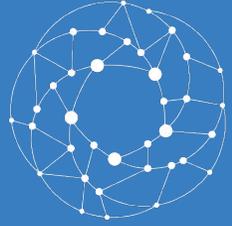


kubernetes

```
./install_osm.sh -c k8s
```

```
./install_osm.sh -c charmed
```

For more information go to <https://osm.etsi.org/docs/user-guide/01-quickstart.html#installing-osm>



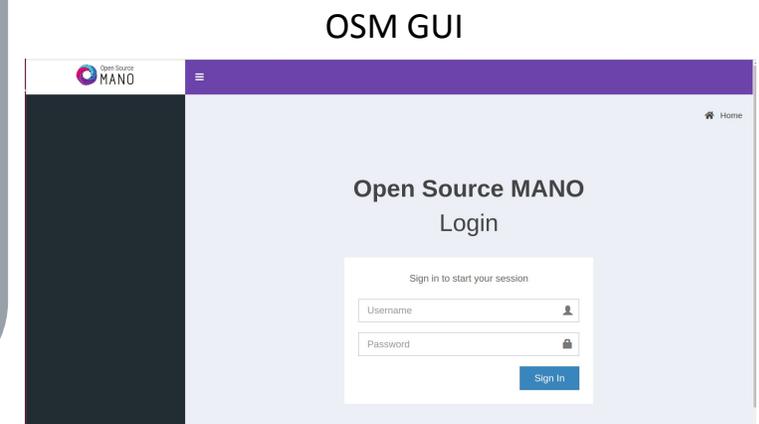
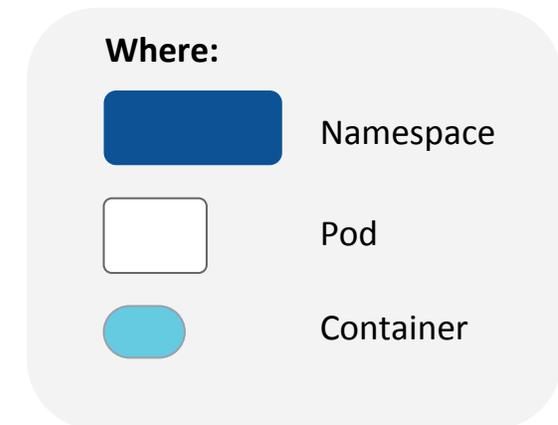
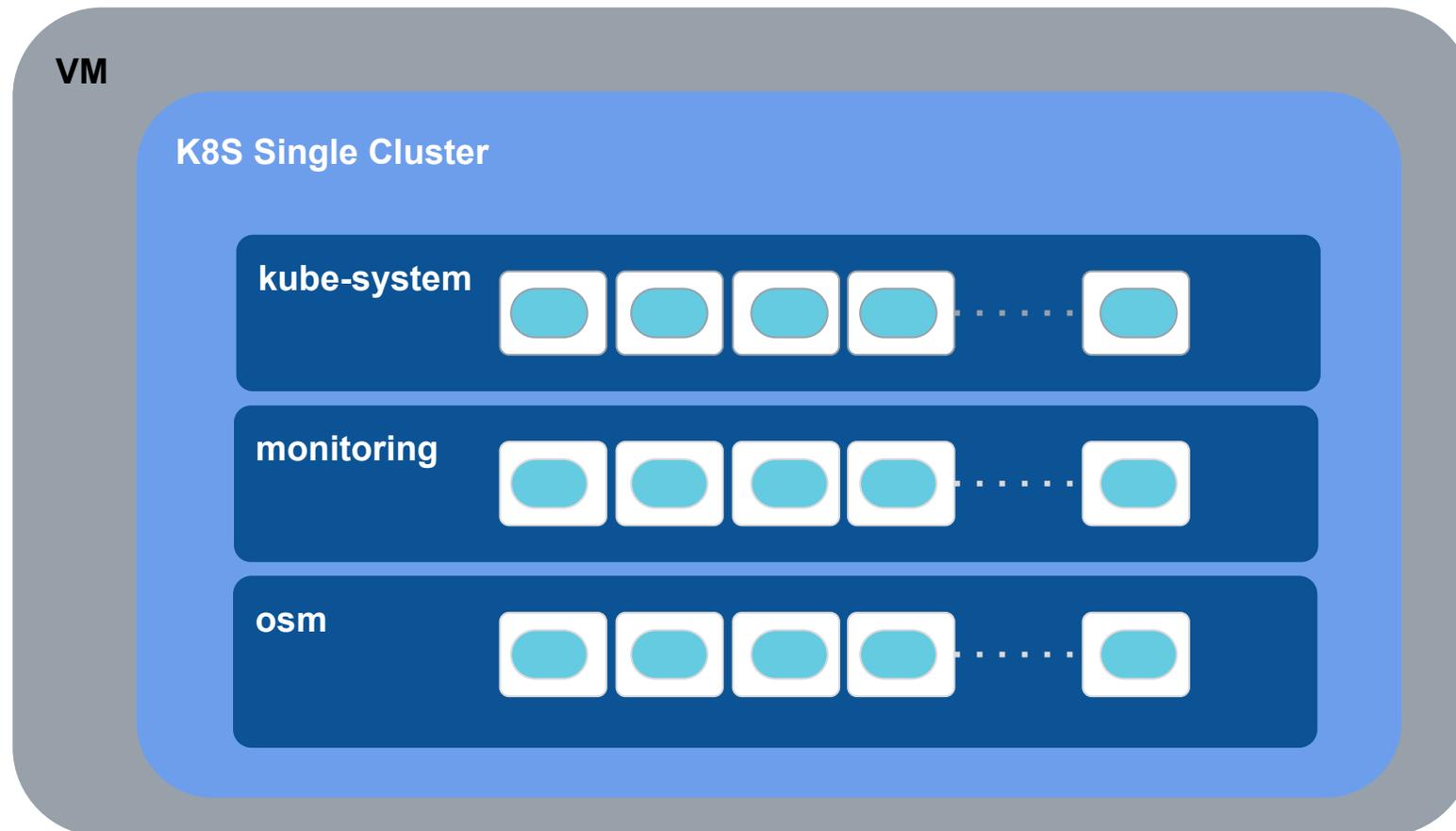
Open Source
MANO

Hands-on: OSM Installation over Kubernetes



Hands-on: OSM Installation over Kubernetes

Scenario



Hands-on: OSM Installation over Kubernetes

1. Take a tenant from <http://bit.ly/OSMHF>
2. Check the IP of your VM at <http://172.21.247.1/project/instances/>, then access it through SSH
user: ubuntu
Password: osm4u
3. Now, let's follow the user-guide at:
<https://osm.etsi.org/docs/user-guide/01-quickstart.html#installing-osm>

```
wget https://osm-download.etsi.org/ftp/osm-7.0-seven/install\_osm.sh
```

4. Make the installer executable

```
chmod +x install_osm.sh
```

Hands-on: OSM Installation over Kubernetes

5. Run the installer with -c k8s flag

```
./install_osm.sh -c k8s
```

6. You will be asked to confirm the installation of the following components:

```
The installation will do the following
  1. Install and configure LXD
  2. Install juju
  3. Install docker CE
  4. Disable swap space
  5. Install and initialize Kubernetes
as pre-requirements.
Do you want to proceed (Y/n)? Y
```

Hands-on: OSM Installation over Kubernetes

7. When installation is finished, execute the following commands to check k8s installation:

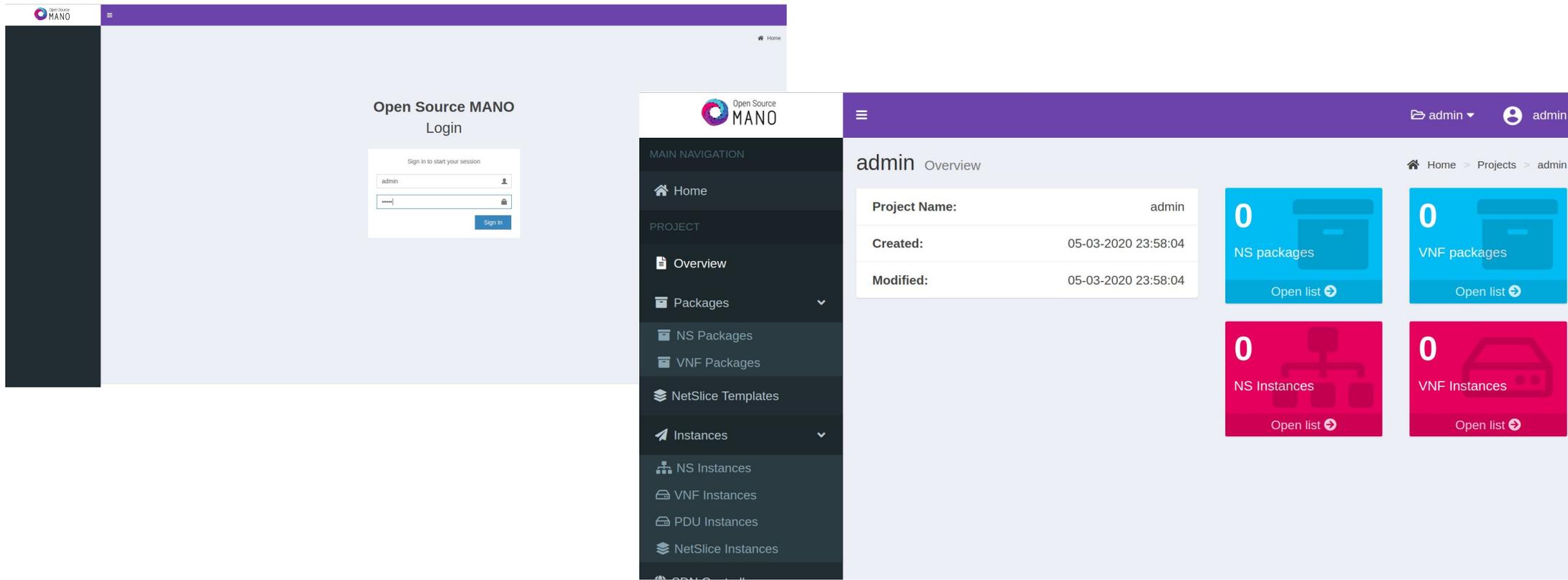
```
kubectl get nodes
kubectl get namespaces
kubectl get pods --all-namespaces
kubectl get all -n kube-system
kubectl get all -n osm
kubectl describe pod light-ui-xyz -n osm
```

8. Test the OSM client:

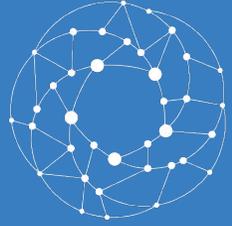
```
osm --help
osm user-list
```

Hands-on: OSM Installation over Kubernetes

9. Go to OSM GUI at `http://<VM-IP>` and access with `admin/admin`



The image displays two screenshots of the Open Source MANO GUI. The left screenshot shows the login page with the title "Open Source MANO Login" and a "Sign in to start your session" form. The form contains fields for "admin" and a password, with a "Sign in" button. The right screenshot shows the "admin Overview" page. It features a sidebar navigation menu with options like Home, Overview, Packages, NS Packages, VNF Packages, NetSlice Templates, Instances, NS Instances, VNF Instances, PDU Instances, and NetSlice Instances. The main content area displays the "admin" project details, including "Project Name: admin", "Created: 05-03-2020 23:58:04", and "Modified: 05-03-2020 23:58:04". Below this, there are four summary cards: "NS packages" (0), "VNF packages" (0), "NS Instances" (0), and "VNF Instances" (0), each with an "Open list" button.



Open Source
MANO

Hands-on: OSM System Monitoring

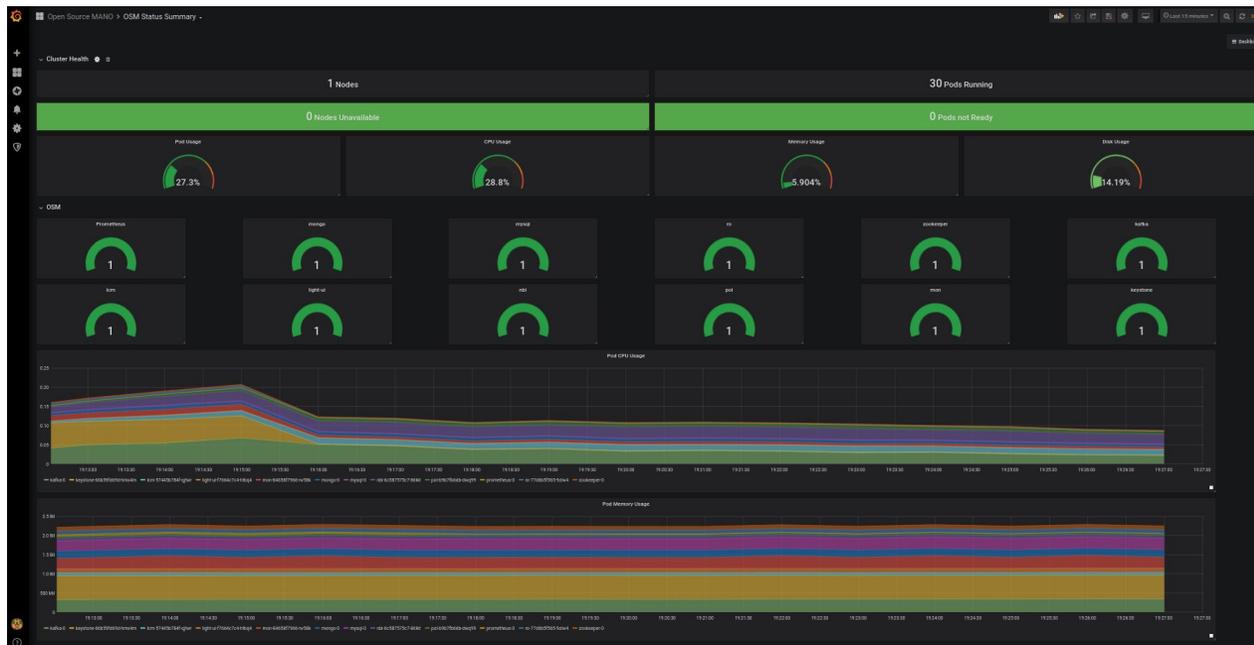


Do not do this yet

The usual way to go

```
./install_osm.sh -c k8s --k8s_monitor
```

Access dashboard: <http://<osm-host>:3000>



Kubernetes health



OSM component
status



OSM component
resource consumption

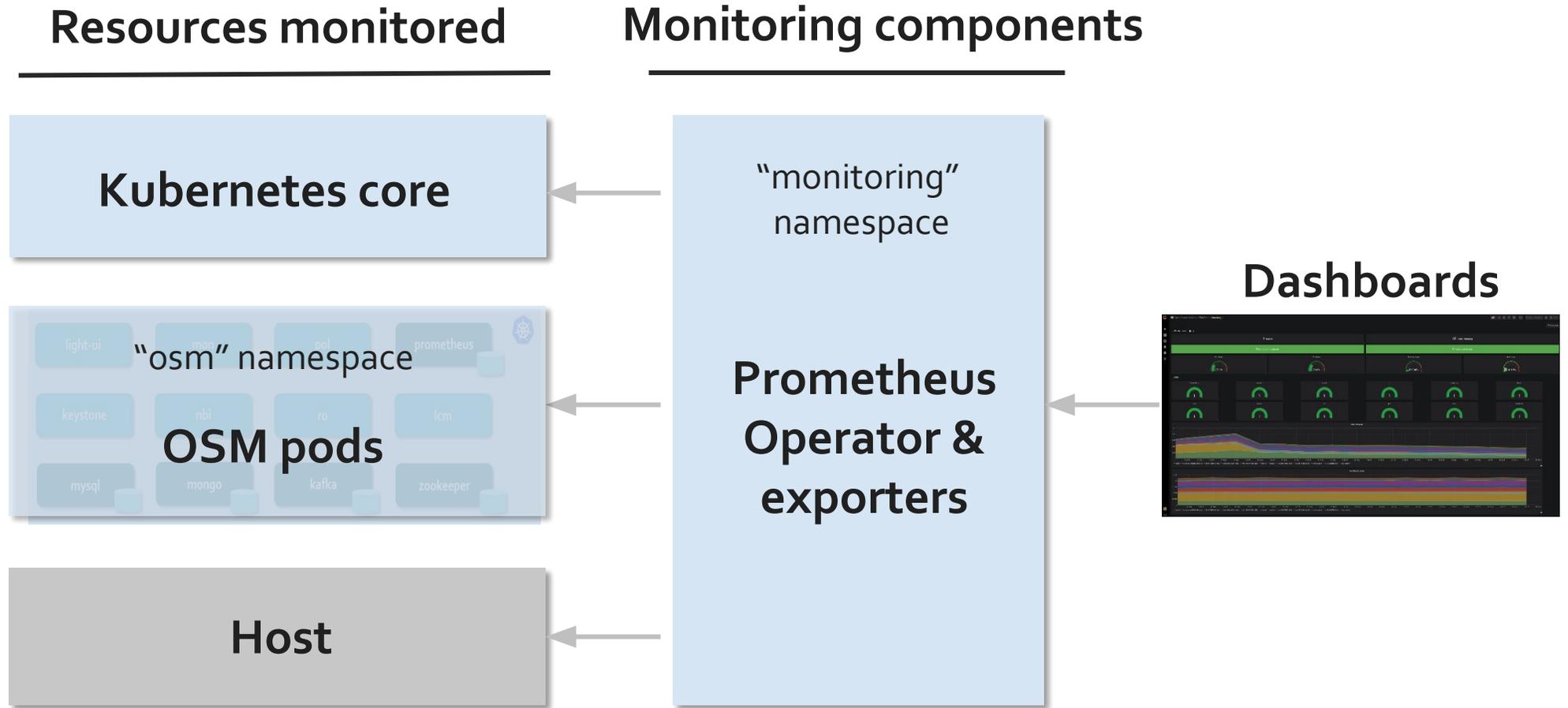
Getting the latest release candidate

- Note1 : We are assuming you did not include the switch “-k8s_monitor” in the previous installation. Otherwise please do now installers/uninstall-k8s-monitoring.sh after step 2
- Note 2: We are assuming you used the switch “-c k8s”

STEPS

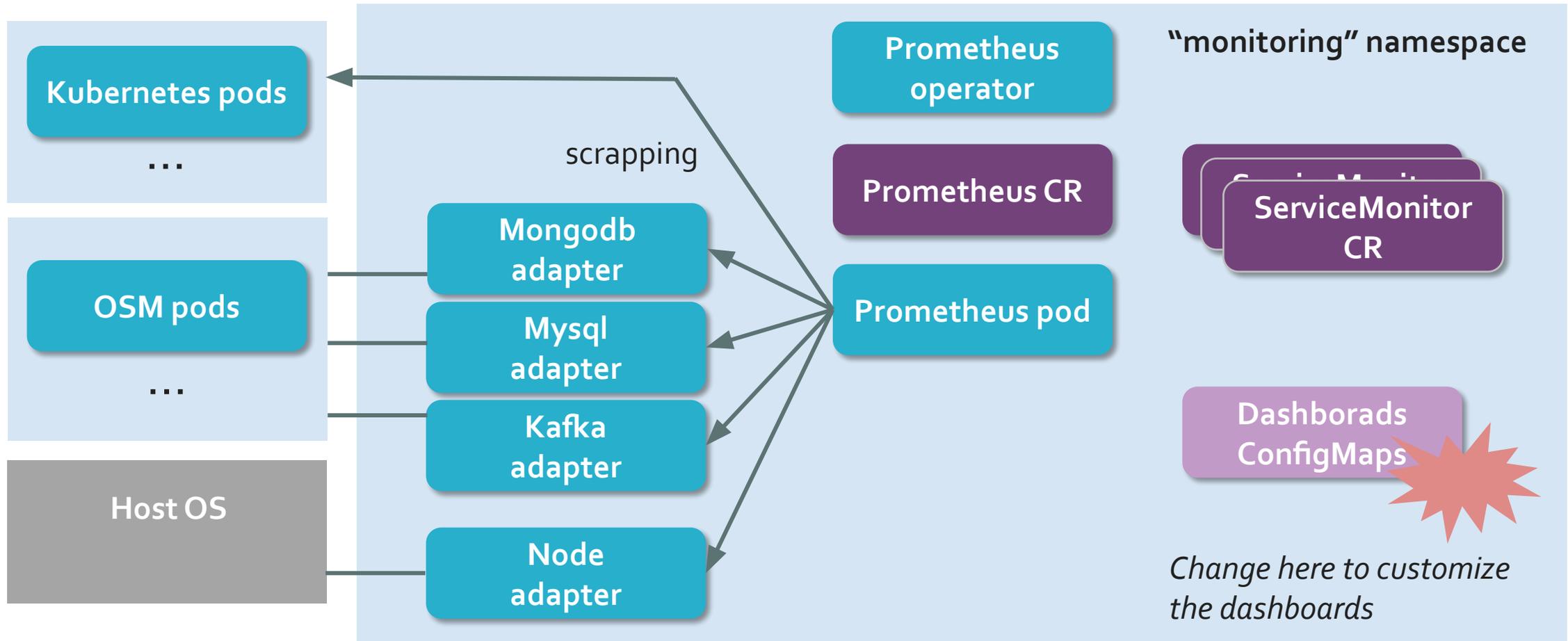
1. `git clone "https://osm.etsi.org/gerrit/osm/devops"`
2. `cd ~/devops/`
3. `git pull "https://osm.etsi.org/gerrit/osm/devops"
refs/changes/72/8372/10`
4. `cd ~/devops/installers/`
5. `./full_install_osm.sh -o k8s_monitor -D $HOME/devops`

What is installed



- Available in the k8s deployment of OSM.
- There is a similar feature for the docker swarm (classic) deployment of OSM (not to be discussed here)
- Aimed at monitoring OSM infrastructure, NOT the VNF/NS deployed
- Implementation based on Prometheus operator (Helm chart), plus some Prometheus exporters (node, Kafka, mysql, mongodb), in “monitoring” namespace

More implementation details



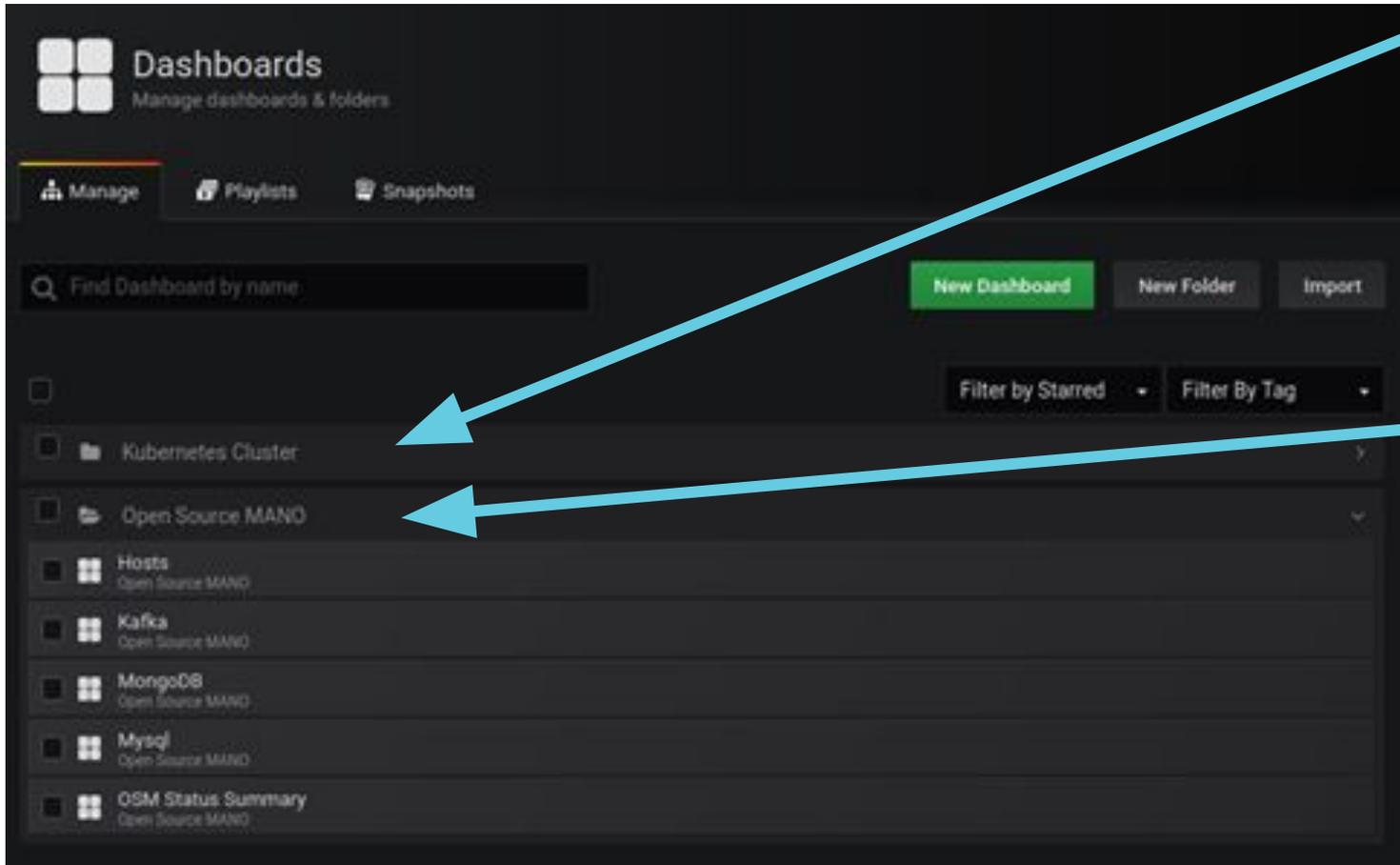
Resources monitored

Adapters to Prometheus

Monitoring pods

Configuration

http://ip-address>:3001 (admin:prom-operator)



- *Kubernetes cluster* upstream dashboards in Prometheus operator helm chart
- *Open Source MANO* Specific dashboards for OSM
 - OSM Status summary
 - Hosts
 - Kafka, mongoddb, mysql

OSM Status summary



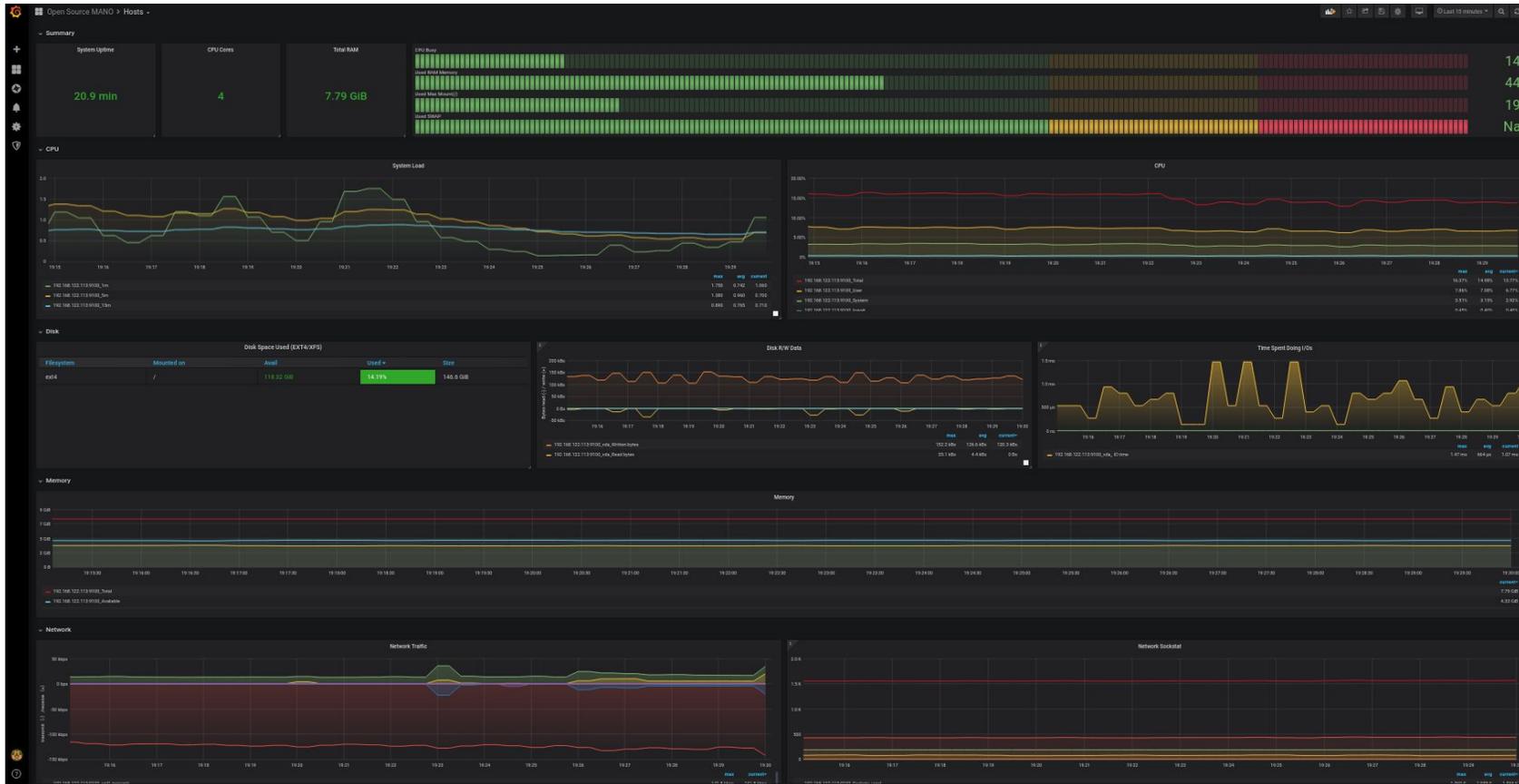
Failed pods / Failed nodes (if any)

K8s resources requested

OSM components status (up/down)

CPU/Memory per OSM component

Hosts status



Summary (uptime, used memory, CPU, disk)

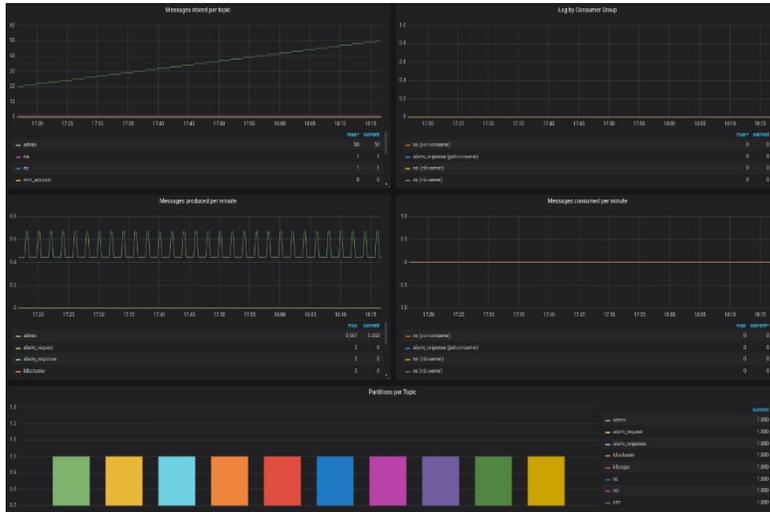
CPU usage

Disk usage

Memory usage

Network usage

Mongo, mysql and Kafka dashboards



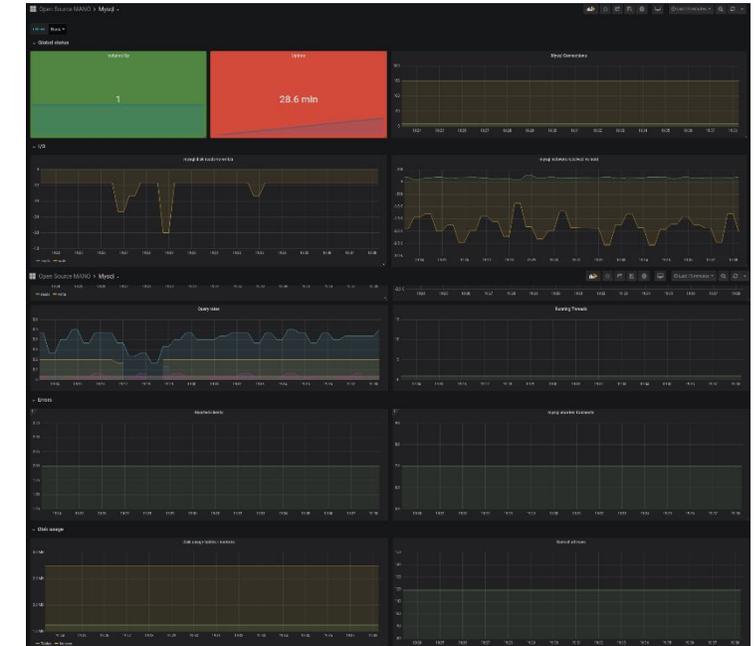
Kafka

Messages produced/consumed
Lag by consumer group
Partitions per topic



Mongodb

Connections
Document operation stats
Network operations



Mysql

Connections
Disk occupation (indexes, tables)
Network operations

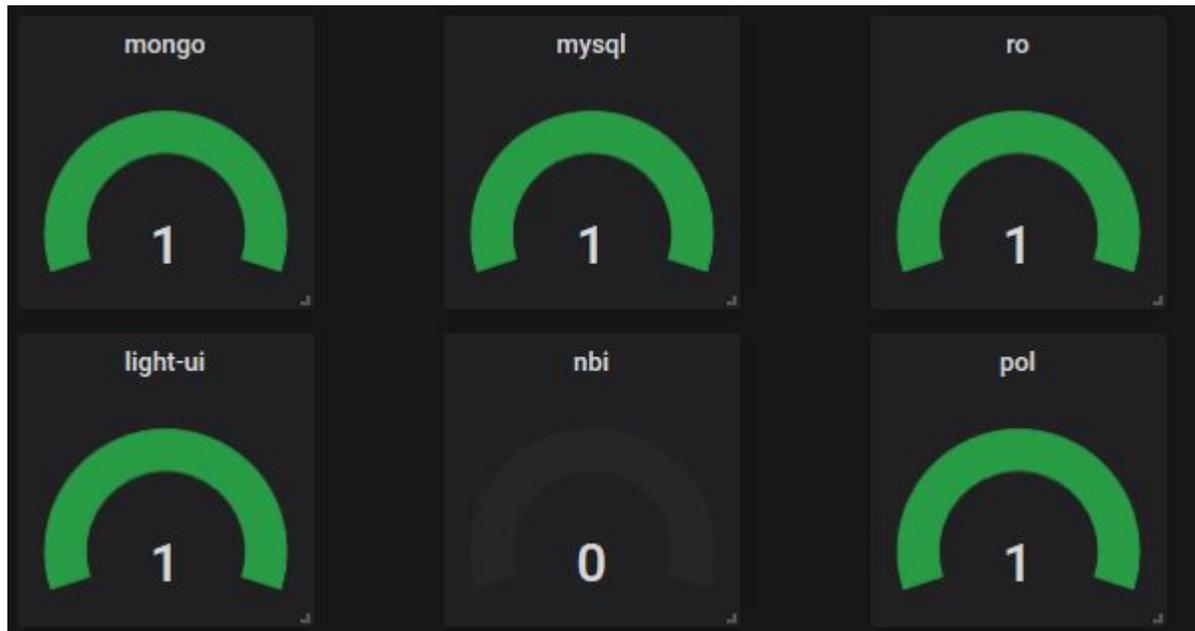
Inspecting the “monitoring” namespace

- See all the objects deployed in the monitoring namespace
 - `kubectl --namespace monitoring get all`
- In particular, the dashboards are stored as configmaps
 - `kubectl --namespace monitoring get configmap`
- Servicemonitors specify what is to be scrapped by Prometheus
 - `kubectl --namepsace monitoring get servicemonitor`

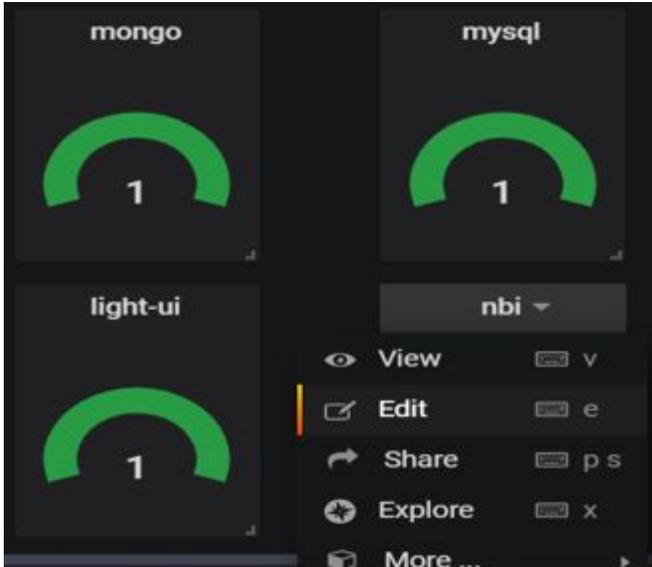
Let's play a little

Force no pods running nbi

```
kubectl scale --namespace osm --replicas=0  
deployment/nbi
```



We are going to improve the dashboard



Go to Edit ->Visualization

Coloring: Activate "value"

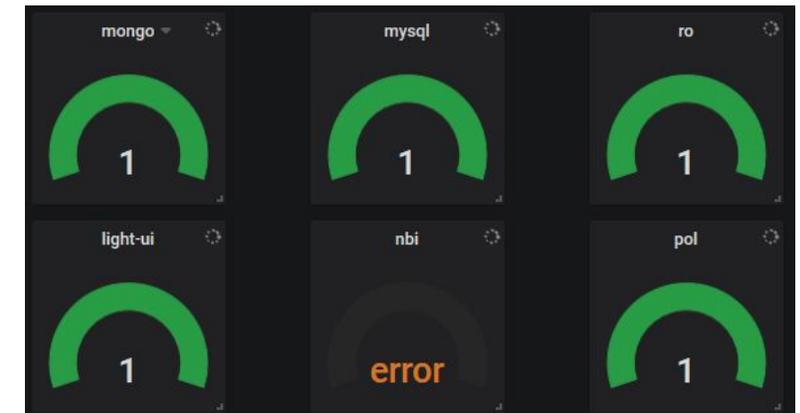
Gauge: Deactivate "show"

Value Mappings: Set value mappings

null -> error

0 -> error

1 -> ok



And make the change persistent

- Get the summary dashboard configmap definition to your computer

```
scp  
ubuntu@<ip-addr>:/home/ubuntu/devops/installers/k8s/summary-dashboard.yaml .
```

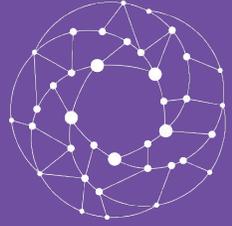
- In grafana, “export” “json”, and copy in the data contents of the .yaml file defining the configmap

- Upload the modified file

- ```
scp summary-dashboard.yaml
ubuntu@<ip-addr>:/home/ubuntu/devops/installers/k8s
```

- Update the dashboard

- ```
kubectl -n monitoring apply -f summary-dashboard.yaml
```



Open Source
MANO

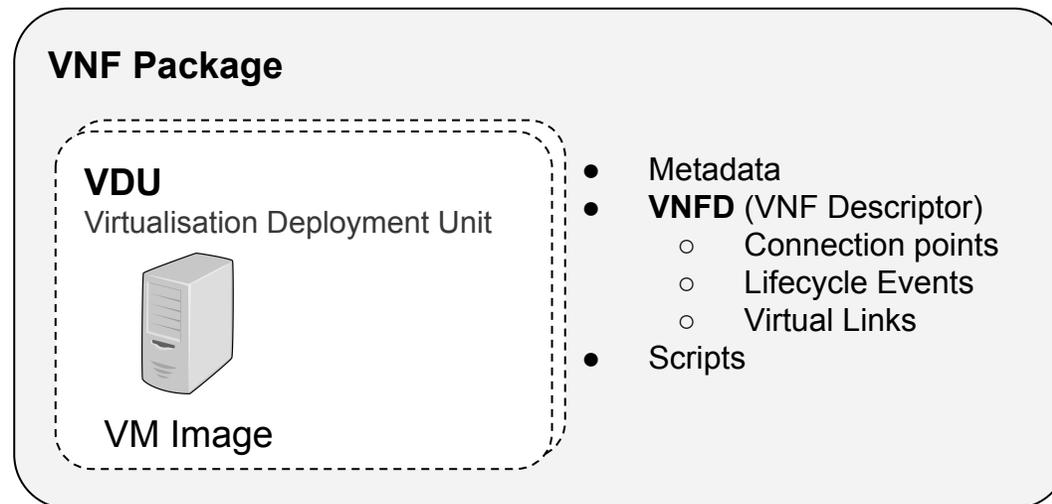
OSM Packages overview



What is a package in NFV?

Packages contain the information that orchestrators need to launch a network service. There are basically two types of packages.

The VNF Package

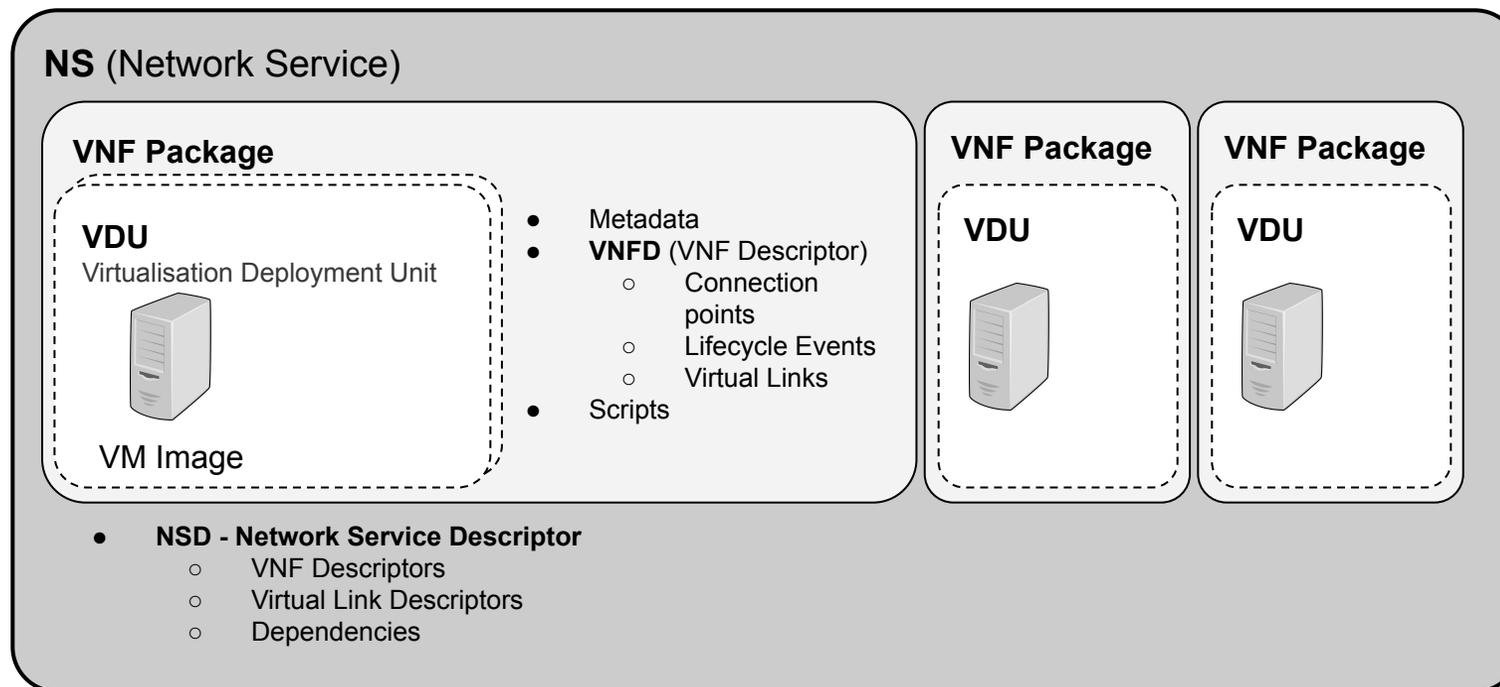


- It contains the characteristics of the VNF, for example:
 - The software image(s) it needs.
 - Compute resources.
 - Network connections between its components (Internal Virtual Links)
 - Performance requirements.
 - Automation scripts.
- Its main element is the VNF Descriptor (VNFD)
- It is built and provided by the VNF vendor.
- This applies in a similar way to new conceptual kinds of Network Functions (NFs), like a Physical NF (PNF), a Containerized NF (CNF), a Kubernetes-based NF (KNF), and Hybrid Network Package (HNF), etc.

What is a package in NFV?

Packages contain the information that orchestrators need to launch a network service. They are basically two types of packages.

The Network Service Package



- It contains the characteristics of the Network Service, for example:
 - The VNF(s) it needs.
 - Network connections between VNFs (external Virtual Links)
- Its main element is the NS Descriptor (NSD)
- It is built by the operator from VNFs that conform the Network Service that needs to be provided.

Package descriptors in OSM are modeled in an increasing alignment to ETSI NFV standards (SOL006) Everything that can be put in a descriptor to model a VNF or NS, is present at OSM's Information Model, maybe the richest model of the NFV MANO industry.

Visit this link to navigate the model: <https://osm.etsi.org/docs/user-guide/11-osm-im.html>



The screenshot shows the Open Source MANO documentation website. It features a blue header with the logo and a search bar. Below the header is a dark sidebar with a 'TABLE OF CONTENTS' section listing seven items: 1. OSM Quickstart, 2. OSM Architecture and Functions, 3. Installing OSM, 4. Setup of Virtual Infrastructure Managers (VIMs), 5. OSM Usage, 6. OSM platform configuration, and 7. What to read next.

Docs » 11. ANNEX 3: OSM Information Model

[View page source](#)

11. ANNEX 3: OSM Information Model

11.1. YANG model in OSM repos

YANG models can be found in the IM repo under the models folder: <https://osm.etsi.org/gitweb/?p=osm/IM.git;a=tree>

OSM uses `pyang` and `pyangbind` to generate Python classes used by the different OSM components.

11.2. OSM IM tree representation

Below you can find tree representations of the VNFD (VNF Descriptor), NSD (Network Service Descriptor), NST (Network Slice Template), VNFR (VNF Record), NSR (Network Service Record), NSI (Network Slice Instance), both in navigable and plain text formats.

Packages in OSM

The NS Package is the one actually being launched in OSM.
It requires constituent VNF Packages to be present in the system.

```
nsd:nsd-catalog:
  nsd:
  - id: hackfest_basic-ns
    name: hackfest_basic-ns
    short-name: hackfest_basic-ns
    description: Simple NS with a single VNF and a single VL
    version: '1.0'
    logo: osm.png
    constituent-vnfd:
    - vnfd-id-ref: hackfest_basic-vnf
      member-vnf-index: '1'
    vld:
    - id: mgmtnet
      name: mgmtnet
      short-name: mgmtnet
      type: ELAN
      mgmt-network: 'true'
      vnfd-connection-point-ref:
      - vnfd-id-ref: hackfest_basic-vnf
        member-vnf-index-ref: '1'
        vnfd-connection-point-ref: vnf-cp0
```

Network Service “hackfest_basic-ns”

It needs VNF “hackfest_basic-vnf” to be present

It will put the VNF in a new network called ‘mgmtnet’

Packages in OSM

The VNF Package is the one describing a given Network Function.
It requires VIM/NFVIs to support whatever characteristic is being required through its descriptor.

```
vnfd:vnfd-catalog:
  vnfd:
  - id: hackfest_basic-vnf
    name: hackfest_basic-vnf
    short-name: hackfest_basic-vnf
    version: '1.0'
    description: A basic VNF descriptor w/ one VDU
    logo: osm.png
    connection-point:
    - name: vnf-cp0
      type: VPORT
    vdu:
    - id: hackfest_basic-VM
      name: hackfest_basic-VM
      image: ubuntu1604
      alternative-images:
      - vim-type: aws
        image: ubuntu/images/hvm-ssd/ubuntu-artful-17.10-amd64-server-20180509
      count: '1'
      vm-flavor:
        vcpu-count: '1'
        memory-mb: '1024'
        storage-gb: '10'
      interface:
      - name: vdu-eth0
        type: EXTERNAL
        virtual-interface:
          type: PARAVIRT
        external-connection-point-ref: vnf-cp0
    mgmt-interface:
      cp: vnf-cp0
```

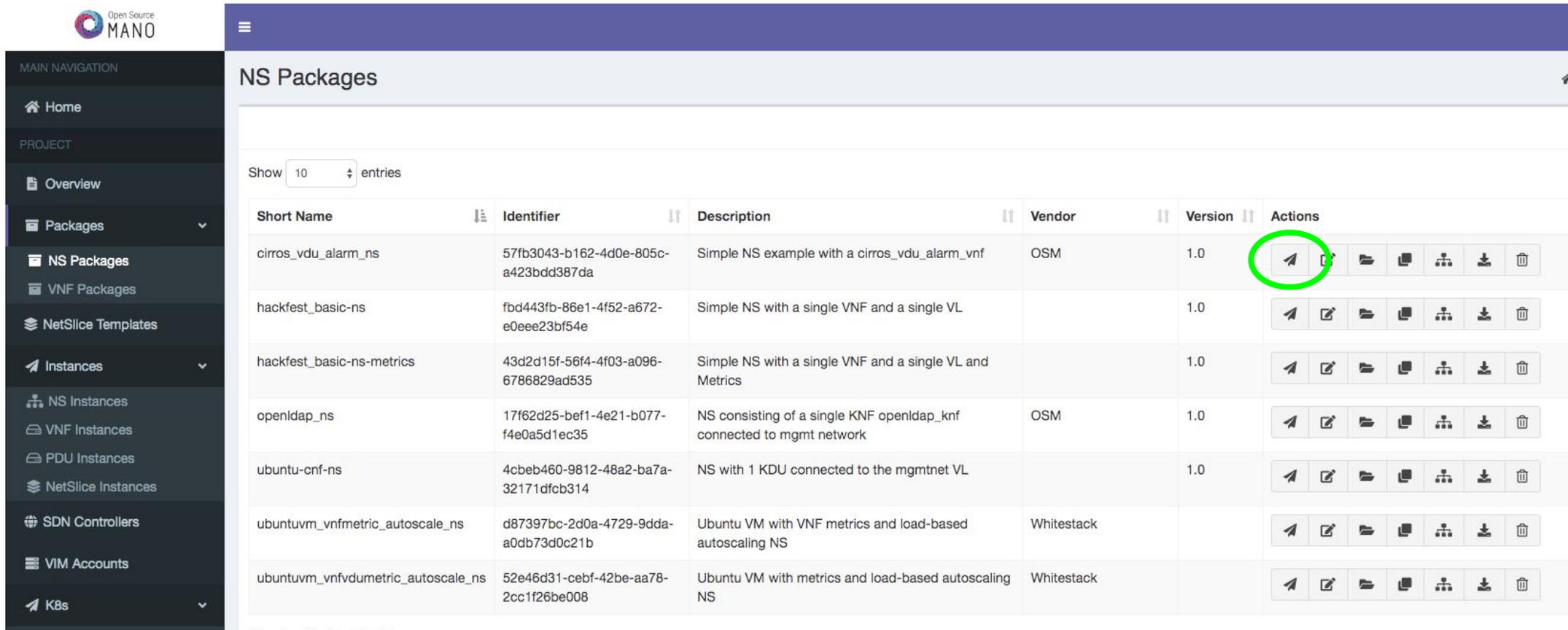
VNF “hackfest_basic-vnf”

It has one VDU (VM) that requires an image called ‘ubuntu1604’, and a flavor with 1 vCPU, 1GB RAM and 10GB of storage.

It has one interface, exposed to the Network Service as external Connection Point “vnf-cp0”

Packages in OSM

Once NS Packages and their constituent VNF Packages are present in the system, and at least a VIM is registered, a Network Service can be launched.



Open Source MANO

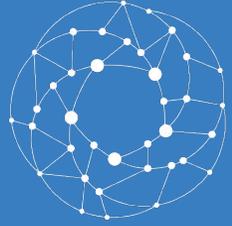
MAIN NAVIGATION

- Home
- PROJECT
- Overview
- Packages
- NS Packages
- VNF Packages
- NetSlice Templates
- Instances
- NS Instances
- VNF Instances
- PDU Instances
- NetSlice Instances
- SDN Controllers
- VIM Accounts
- K8s

NS Packages

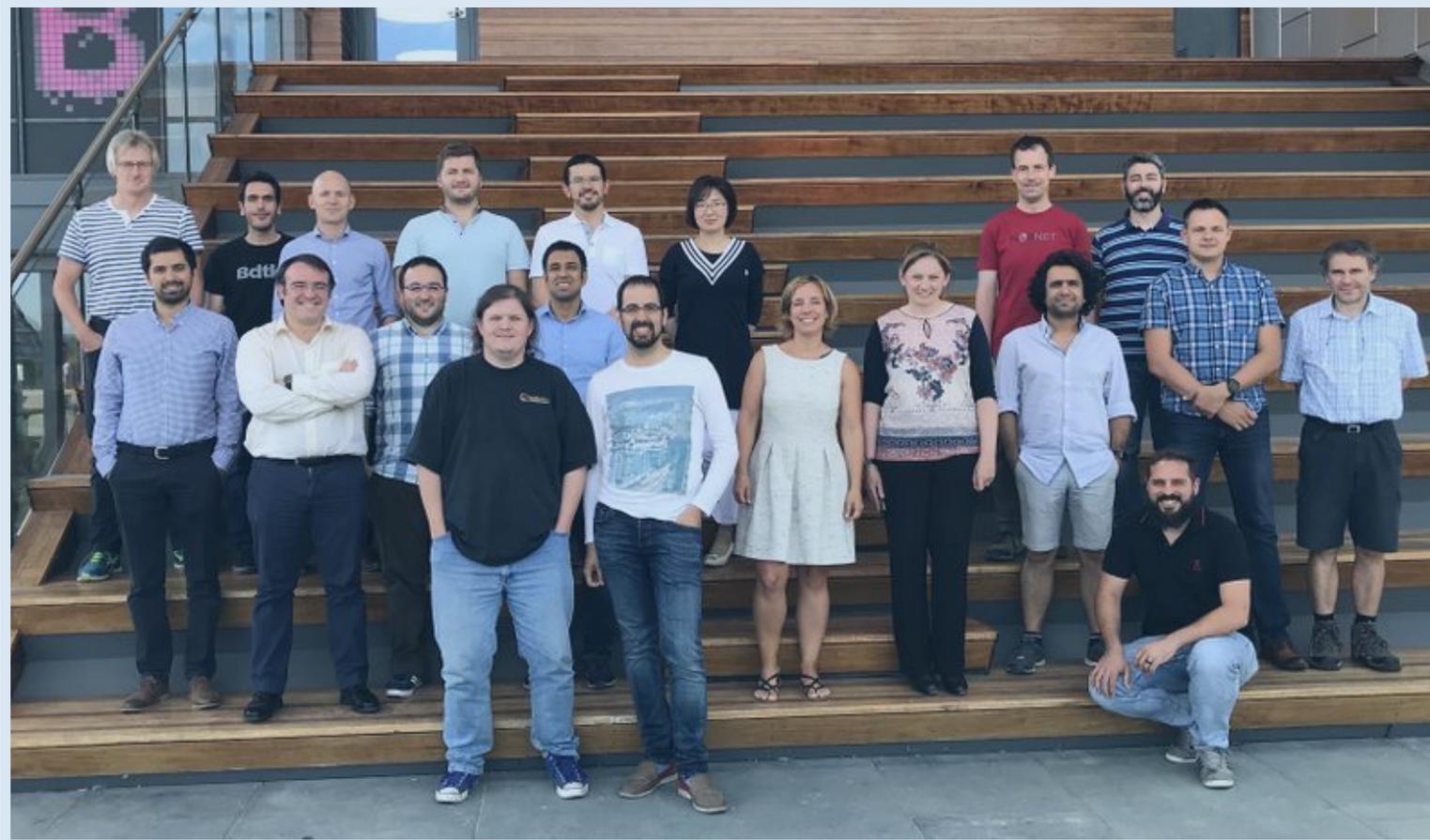
Show 10 entries

Short Name	Identifier	Description	Vendor	Version	Actions
cirros_vdu_alarm_ns	57fb3043-b162-4d0e-805c-a423bdd387da	Simple NS example with a cirros_vdu_alarm_vnf	OSM	1.0	
hackfest_basic-ns	fb4443fb-86e1-4f52-a672-e0eee23bf54e	Simple NS with a single VNF and a single VL		1.0	
hackfest_basic-ns-metrics	43d2d15f-56f4-4f03-a096-6786829ad535	Simple NS with a single VNF and a single VL and Metrics		1.0	
openldap_ns	17f62d25-bef1-4e21-b077-f4e0a5d1ec35	NS consisting of a single KNF openldap_knf connected to mgmt network	OSM	1.0	
ubuntu-cnf-ns	4cbeb460-9812-48a2-ba7a-32171dfcb314	NS with 1 KDU connected to the mgmtnet VL		1.0	
ubuntum_vnfmetric_autoscale_ns	d87397bc-2d0a-4729-9dda-a0db73d0c21b	Ubuntu VM with VNF metrics and load-based autoscaling NS	Whitestack		
ubuntum_vnfvdmetric_autoscale_ns	52e46d31-cebf-42be-aa78-2cc1f26be008	Ubuntu VM with metrics and load-based autoscaling NS	Whitestack		



Open Source
MANO

Hands-on: Integrating a VIM & Instantiating a basic Network Service



Hands-on: Integrating a VIM

1. Create a VIM in OSM via CLI

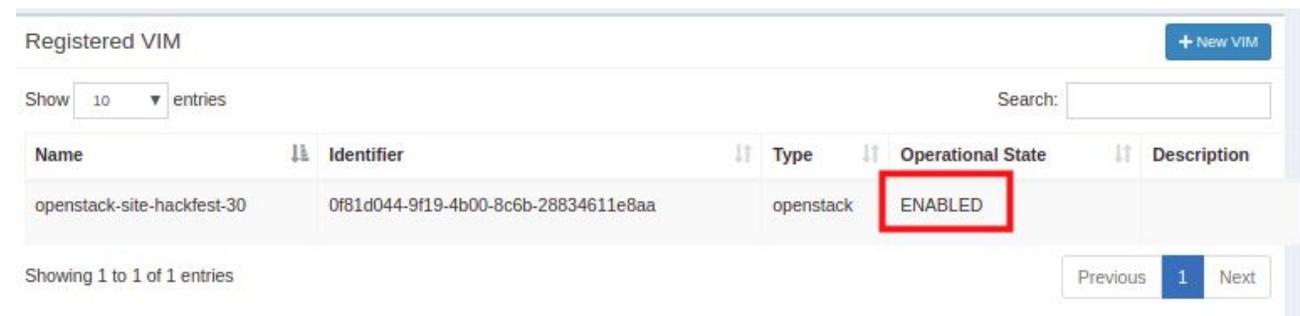
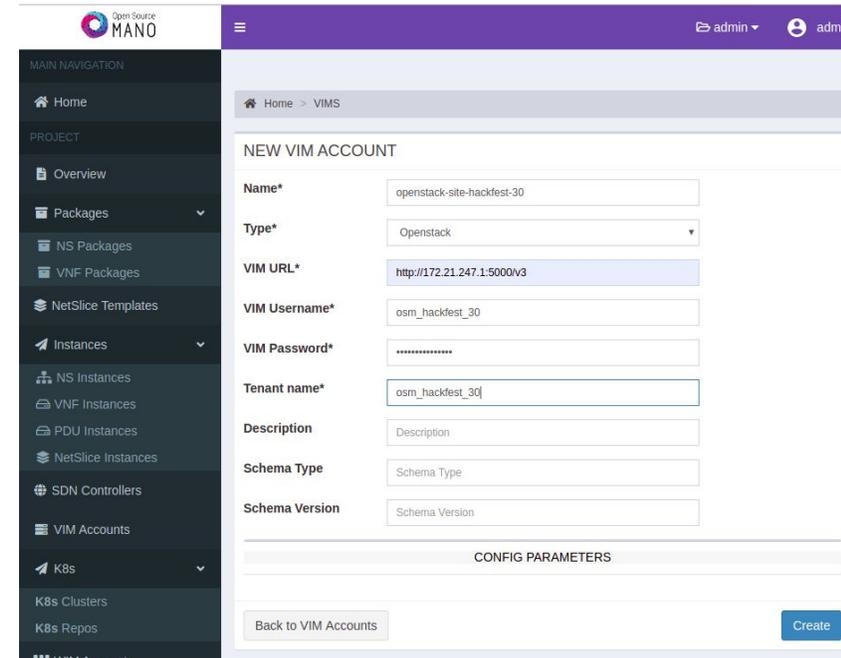
```
osm vim-create --name openstack-site-hackfest-x --user osm_hackfest_x --password  
<Pass> --auth_url http://<VIM-IP>:5000/v3 --tenant osm_hackfest_x --account_type  
openstack --config='{security_groups: default}'
```

2. Validate the VIM creation . The status should be ENABLED

```
osm vim-list  
osm vim-show openstack-site-hackfest-x
```

Hands-on: Integrating a VIM

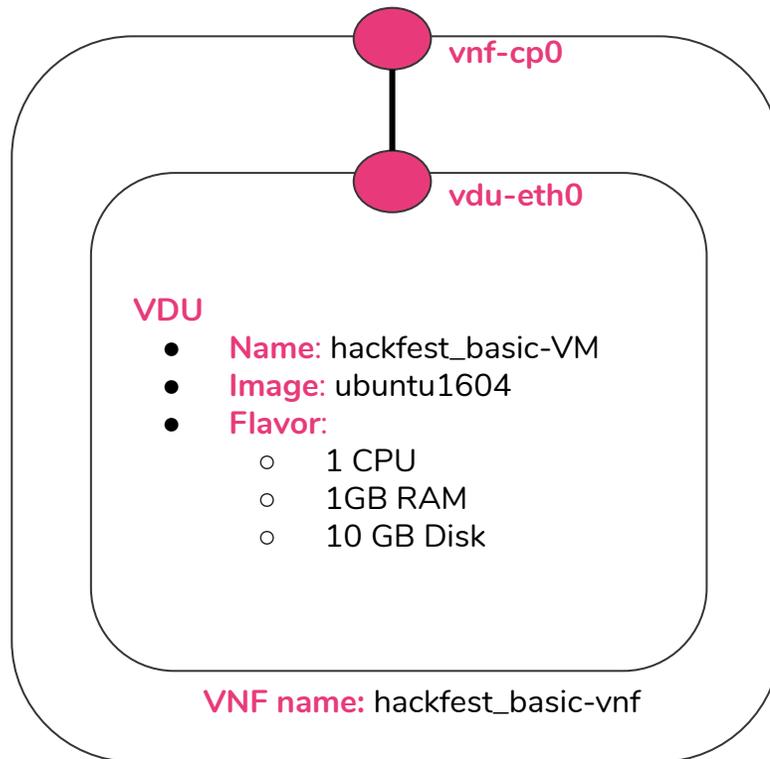
1. Create a VIM in OSM via GUI
2. Go to VIM accounts -> add new VIM
 - Name: openstack-site-hackfest-**x**
 - Type: Openstack
 - VIM URL: http://<VIM-IP>:5000/v3
 - VIM Username: osm_hackfest_**x**
 - VIM Password: *****
 - Tenant name: osm_hackfest_**x**
3. Click in Create button
4. Validate the VIM creation .
The status should be ENABLED



Name	Identifier	Type	Operational State	Description
openstack-site-hackfest-30	0f81d044-9f19-4b00-8c6b-28834611e8aa	openstack	ENABLED	

Hands-on: Launching your first NS

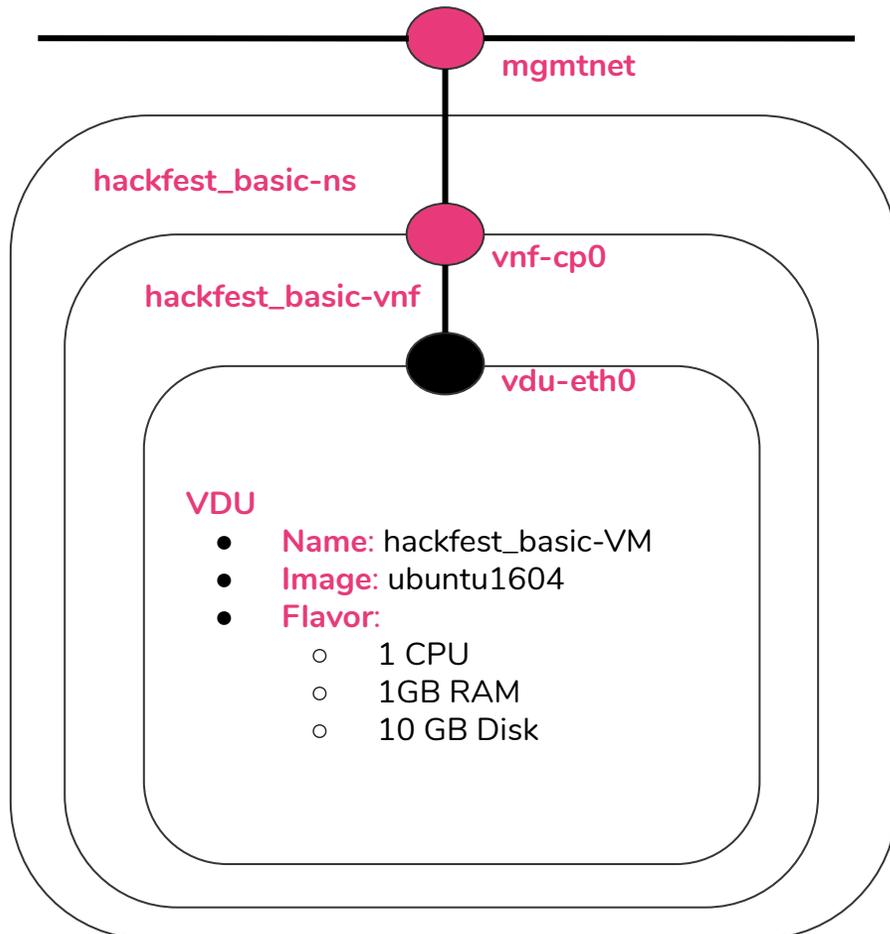
VNFD Descriptor and Diagram



```
vnfd:vnfd-catalog:
vnfd:
- connection-point:
  - name: vnf-cp0
    type: VPORT
  description: A basic VNF descriptor w/ one VDU
  id: hackfest_basic-vnf
  logo: osm.png
  mgmt-interface:
    cp: vnf-cp0
  name: hackfest_basic-vnf
  short-name: hackfest_basic-vnf
vdu:
- alternative-images:
  - image: ubuntu/images/hvm-ssd/ubuntu-artful-17.10-amd64-server-20180509
    vim-type: aws
  count: '1'
  id: hackfest_basic-VM
  image: ubuntu1604
  interface:
  - external-connection-point-ref: vnf-cp0
    name: vdu-eth0
    type: EXTERNAL
  virtual-interface:
    type: PARAVIRT
  name: hackfest_basic-VM
  vm-flavor:
    memory-mb: '1024'
    storage-gb: '10'
    vcpu-count: '1'
  version: '1.0'
```

Hands-on: Launching your first NS

NSD Descriptor and Diagram



```
nsd:nsd-catalog:  
  nsd:  
    - constituent-vnfd:  
      - member-vnf-index: '1'  
        vnfd-id-ref: hackfest_basic-vnf  
      description: Simple NS with a single VNF and a single VL  
      id: hackfest_basic-ns  
      logo: osm.png  
      name: hackfest_basic-ns  
      short-name: hackfest_basic-ns  
      version: '1.0'  
    vld:  
      - id: mgmtnet  
        mgmt-network: 'true'  
        name: mgmtnet  
        short-name: mgmtnet  
        type: ELAN  
        vnfd-connection-point-ref:  
          - member-vnf-index-ref: '1'  
            vnfd-connection-point-ref: vnf-cp0  
            vnfd-id-ref: hackfest_basic-vnf
```

Hands-on: Launching your first NS

1. Download the nsd and vnfd packages

```
wget http://osm-download.etsi.org/ftp/osm-5.0-five/6th-hackfest/packages/hackfest_basic_vnf.tar.gz  
wget http://osm-download.etsi.org/ftp/osm-5.0-five/6th-hackfest/packages/hackfest_basic_ns.tar.gz
```

2. Create the NSD and VNFD in OSM

```
osm vnfd-create hackfest_basic_vnf.tar.gz  
osm nsd-create hackfest_basic_ns.tar.gz
```

3. Create an SSH key

```
ssh-keygen
```

4. Create the Network Service in OSM

```
osm ns-create --ns_name hackfest1 --nsd_name hackfest_basic-ns --vim_account openstack-site-hackfest-x  
--ssh_keys .ssh/id_rsa.pub --config '{vld: [ {name: mgmtnet, vim-network-name: osm-ext} ] }'
```

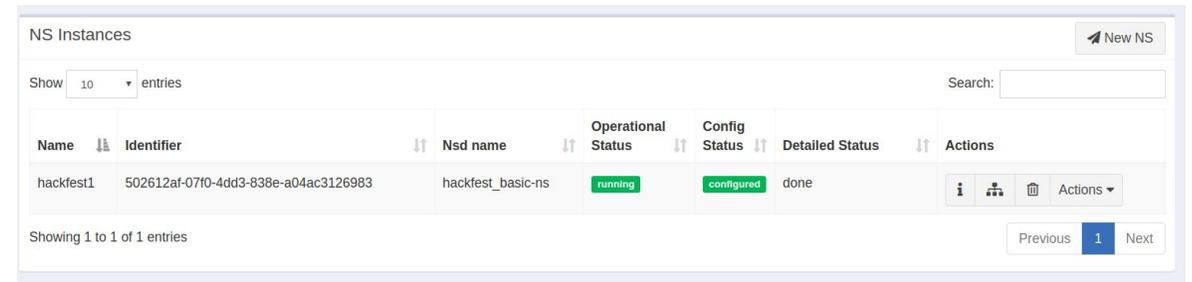
Hands-on: Launching your first NS

5. Validate NS creation in OSM via CLI

```
osm ns-list  
osm ns-show hackfest1
```

6. Validate NS creation in OSM via GUI

- Go to Instances -> NS Instances



The screenshot shows the 'NS Instances' page in the OSM GUI. It features a table with columns for Name, Identifier, Nsd name, Operational Status, Config Status, Detailed Status, and Actions. A single instance named 'hackfest1' is listed with a 'running' operational status and 'configured' config status. The page also includes a search bar, a 'New NS' button, and pagination controls.

Name	Identifier	Nsd name	Operational Status	Config Status	Detailed Status	Actions
hackfest1	502612af-07f0-4dd3-838e-a04ac3126983	hackfest_basic-ns	running	configured	done	   Actions

7. Access to the VM created in Openstack VIM

```
ssh -i .ssh/id_rsa ubuntu@<MGMT_IP>
```

8. Delete NS , NSD and VNFD

```
osm ns-delete hackfest1  
osm vnfd-delete hackfest_basic_vnf  
osm nsd-delete hackfest_basic_ns
```

Bonus Hands-on: Creating VNF & NS Descriptors

1. Create the VNF Descriptor

```
osm package-create vnf hackfest-basic
```

2. Create the NS Descriptor

```
osm package-create ns hackfest-basic
```

3. Build the packages

```
osm package-build hackfest-basic_vnf  
osm package-build hackfest-basic_ns
```

4. Upload NFD and NDS to OSM

```
osm vnfd-create hackfest-basic_vnf.tar.gz  
osm nsd-create hackfest-basic_ns.tar.gz
```

Bonus Hands-on: Creating VNF & NS Descriptors

5. Create the Network Service

```
osm ns-create --ns_name hf-basic --nsd_name hackfest-basic_nsd --vim_account openstack-site-hackfest-x  
--ssh_keys ~/.ssh/id_rsa.pub --config '{vld: [ {name: mgmt, vim-network-name: osm-ext} ] }'
```

6. Validate NS creation in OSM via GUI

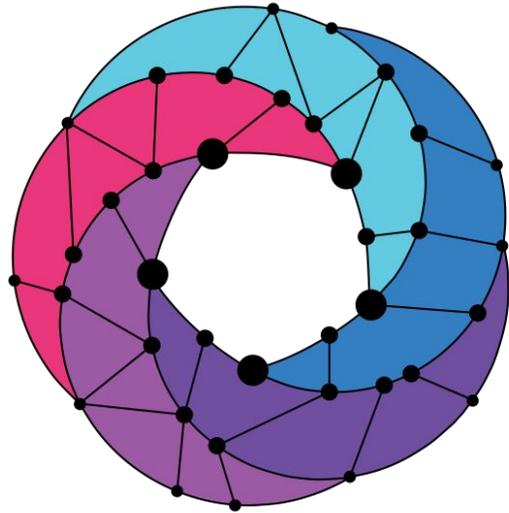
- Go to Instances -> NS Instances



Name	Identifier	Nsd name	Operational Status	Config Status	Detailed Status	Actions
hf-basic	c1b16f36-ad89-4252-a60c-7ddb7fc83f4a	hackfest-basic_nsd	failed	failed	ERROR Waiting for tasks to be finished: b"---\nerror:\n code: 400\n description: 'VIM Exception vimconnException Image not found at VIM with filter:\n "[\"name\": \"image-name\"]\". (True, \" Rollback successful.\")\n type: 400 Bad Request\n"	i 👤 🗑️ Actions

Showing 1 to 1 of 1 entries

7. Compare the VNFD of this example with the previous Hands-On, find the difference and fix it



Open Source MANO

Find us at:

osm.etsi.org
osm.etsi.org/wikipub