OSM#9 Hackfest
Hack 1: OSM System Monitoring and NS Instantiation
Hands-on: OSM System Monitoring
How to install K8 System Monitoring

```
./install_osm.sh -c k8s --k8s_monitor
Access dashboard: http://<osm-host>:3001
```

- Kubernetes health
- OSM component status
- OSM component resource consumption
What is installed

Resources monitored

- Kubernetes core
- "osm" namespace
- OSM pods
- Host

Monitoring components

- "monitoring" namespace
- Prometheus Operator & exporters
- Dashboards
Monitoring OSM

• 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, while VNF/NS metrics are also available since Release 7.1.

• 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

- Kubernetes pods
- OSM pods
- Host OS

- Prometheus operator
- Prometheus CR
- Prometheus pod

- Scraping
- Mongodb adapter
- Mysql adapter
- Kafka adapter
- Node adapter

“monitoring” namespace

- ServiceMonitor CR
- Dashboards
- ConfigMaps

Change here to customize the dashboards

© ETSI 2020
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, mongodb, 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 --namespace monitoring get servicemonitor
Let’s play a little

Force no pods running nbi

```sh
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`
OSM Packages overview
What is a package in NFV?

Packages contain the information that orchestrators need to launch a network service. The 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. There 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.
Packages in OSM

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

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’
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.

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”

```
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
          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
```
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.
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
Hands-on: Launching your first NS

VNFD Descriptor and Diagram

VNF name: hackfest_basic-vnf

VNF descriptor:
- 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'
- name: hackfest_basic-VM
  - Image: ubuntu1604
  - Flavor:
    - 1 CPU
    - 1GB RAM
    - 10 GB Disk

VDU:
- Name: hackfest_basic-VM
- Image: ubuntu1604
- Flavor:
  - 1 CPU
  - 1GB RAM
  - 10 GB Disk

© ETSI 2020
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'
vl:
  - id: mgmtnet
gmt-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
```

**VDU**
- **Name:** hackfest_basic-VM
- **Image:** ubuntu1604
- **Flavor:**
  - 1 CPU
  - 1GB RAM
  - 10 GB Disk

© ETSI 2020
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

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

2. **Create the NS Descriptor**
   
   ```bash
   osm package-create ns hackfest-basic
   ```

3. **Build the packages**
   
   ```bash
   osm package-build hackfest-basic_vnf
   osm package-build hackfest-basic_ns
   ```

4. **Upload VNFD and NSD to OSM**
   
   ```bash
   osm vnfd-create hackfest-basic_vnf.tar.gz
   osm nsd-create hackfest-basic_ns.tar.gz
   ```
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

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