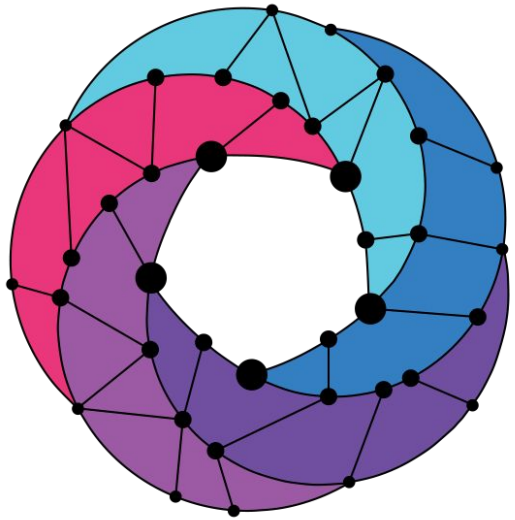


Open Source
MANO

OSM MR Hackfest – Hack 2
Network Services General Instantiation

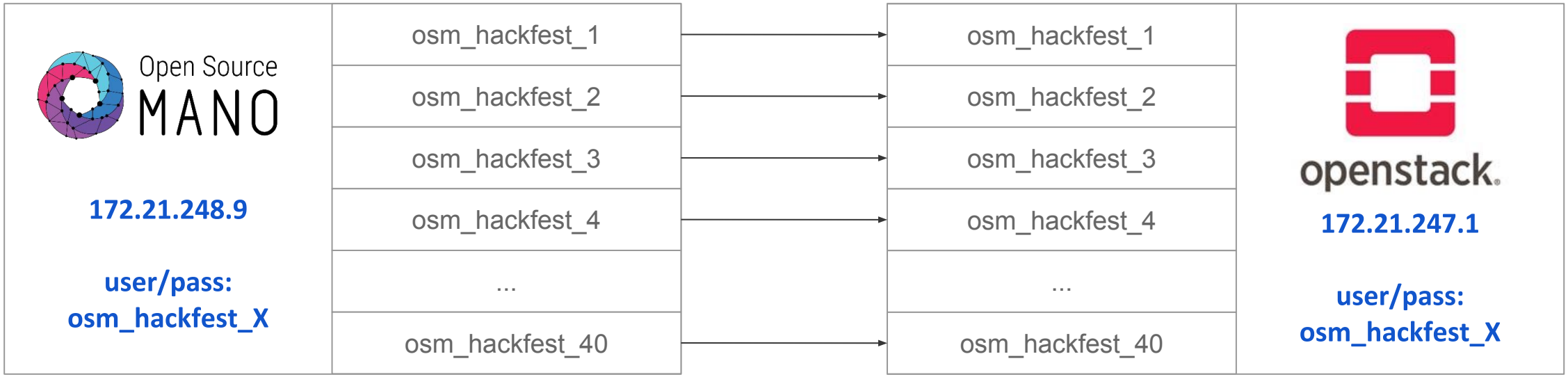
Gianpietro Lavado (Whitestack)
Alfonso Tierno (Telefonica)
Felipe Vicens (Atos)

© ETSI 2020

ETSI

New OSM instance!

This time, we will use a shared OSM instance, leveraging the OSM RBAC capabilities. Don’t forget to create your VIM.




```
osm vim-create --name openstack-site-hackfest-x --user osm_hackfest_x --password
osm_hackfest_x --auth_url http://172.21.247.1:5000/v3 --tenant osm_hackfest_x
--account_type openstack
```

© ETSI 2020

Open Source MANO

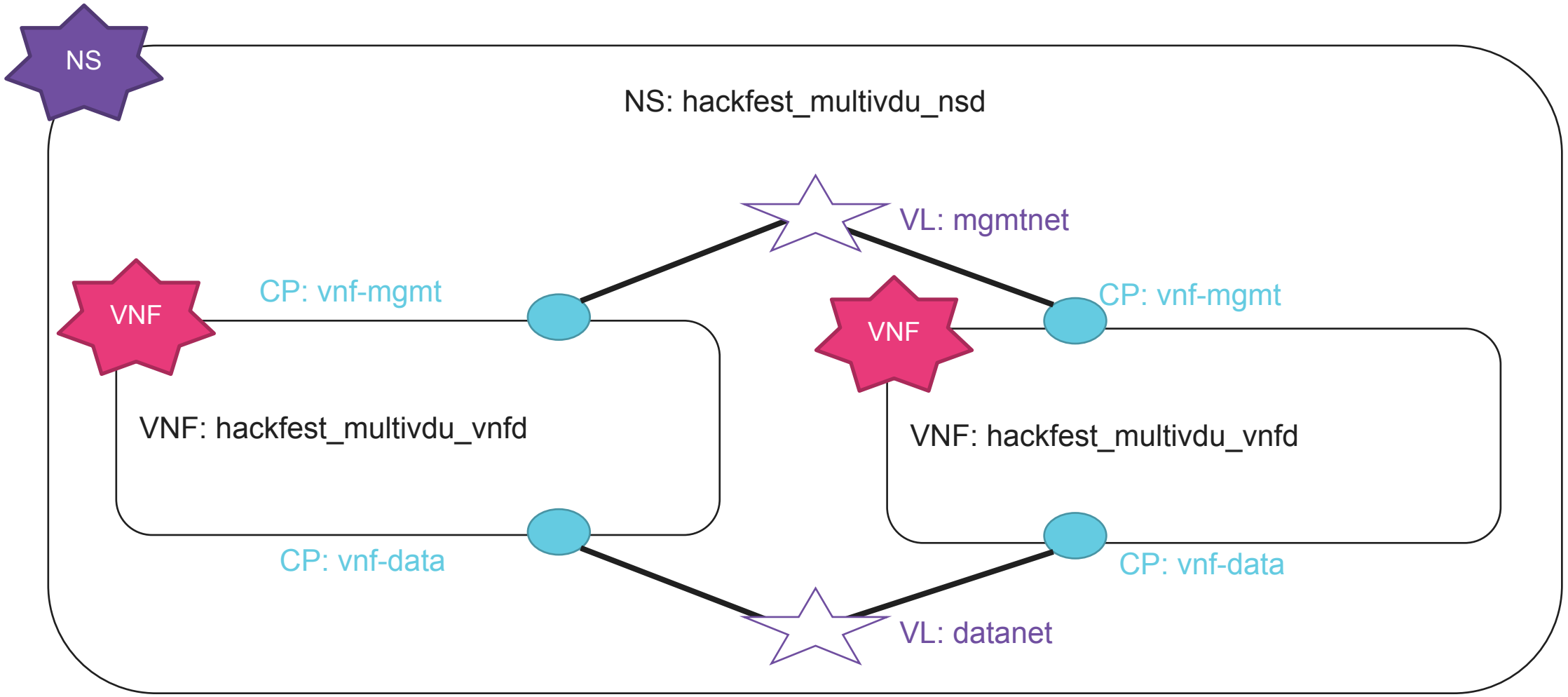
Presentation & Hands-on: Multi-VDU Network Services





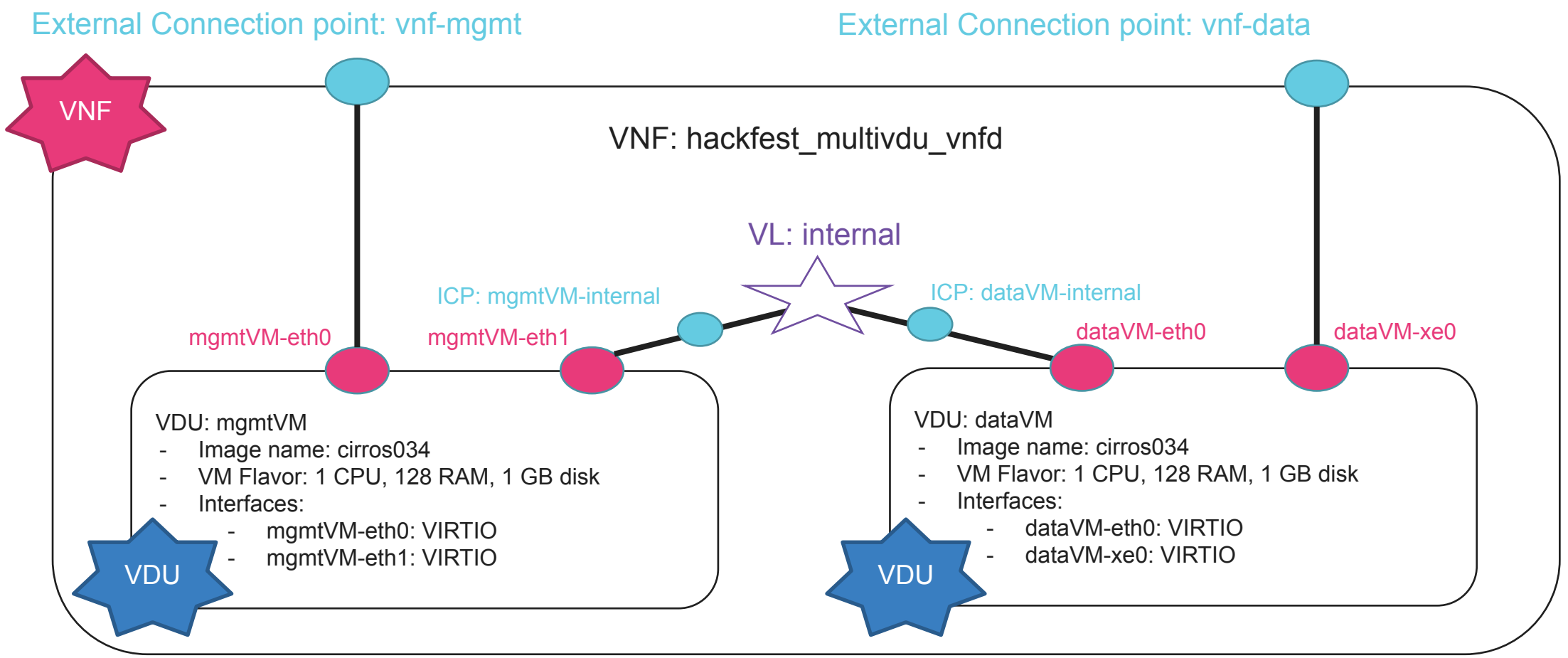
© ETSI 2020

NS diagram



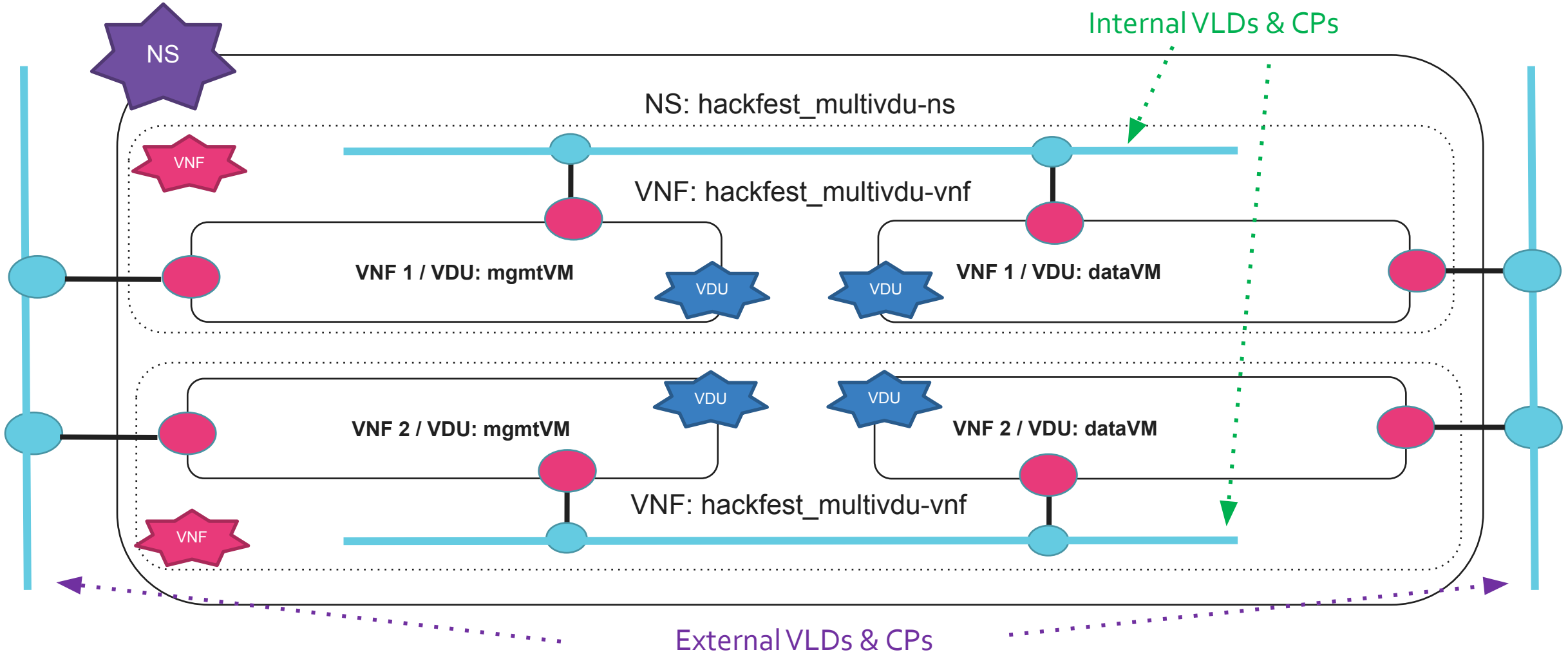
© ETSI 2020

VNF diagram



© ETSI 2020

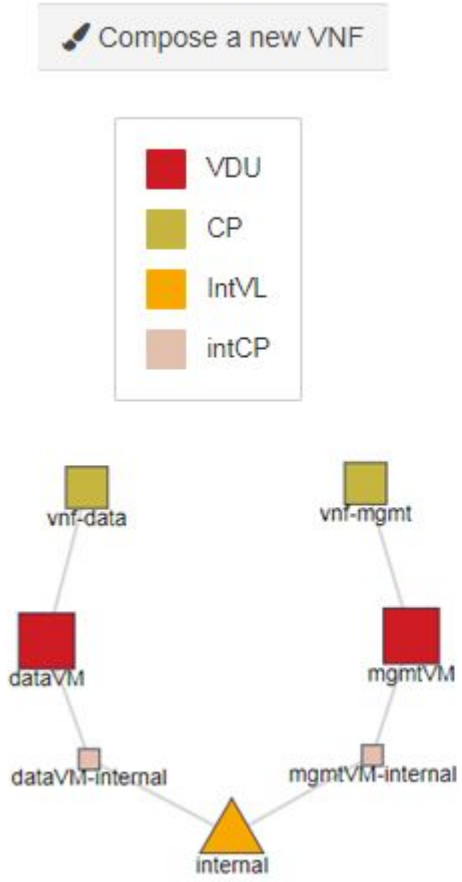
Final Multi-VDU Picture

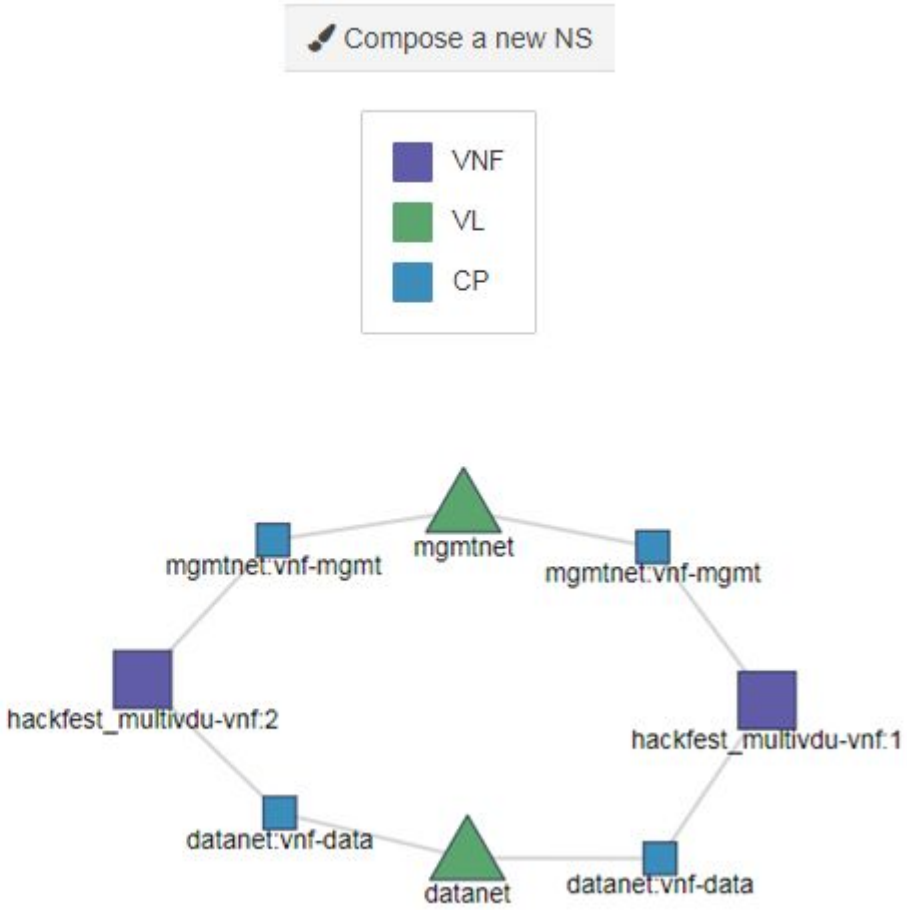


© ETSI 2020

VNF/NS Compose

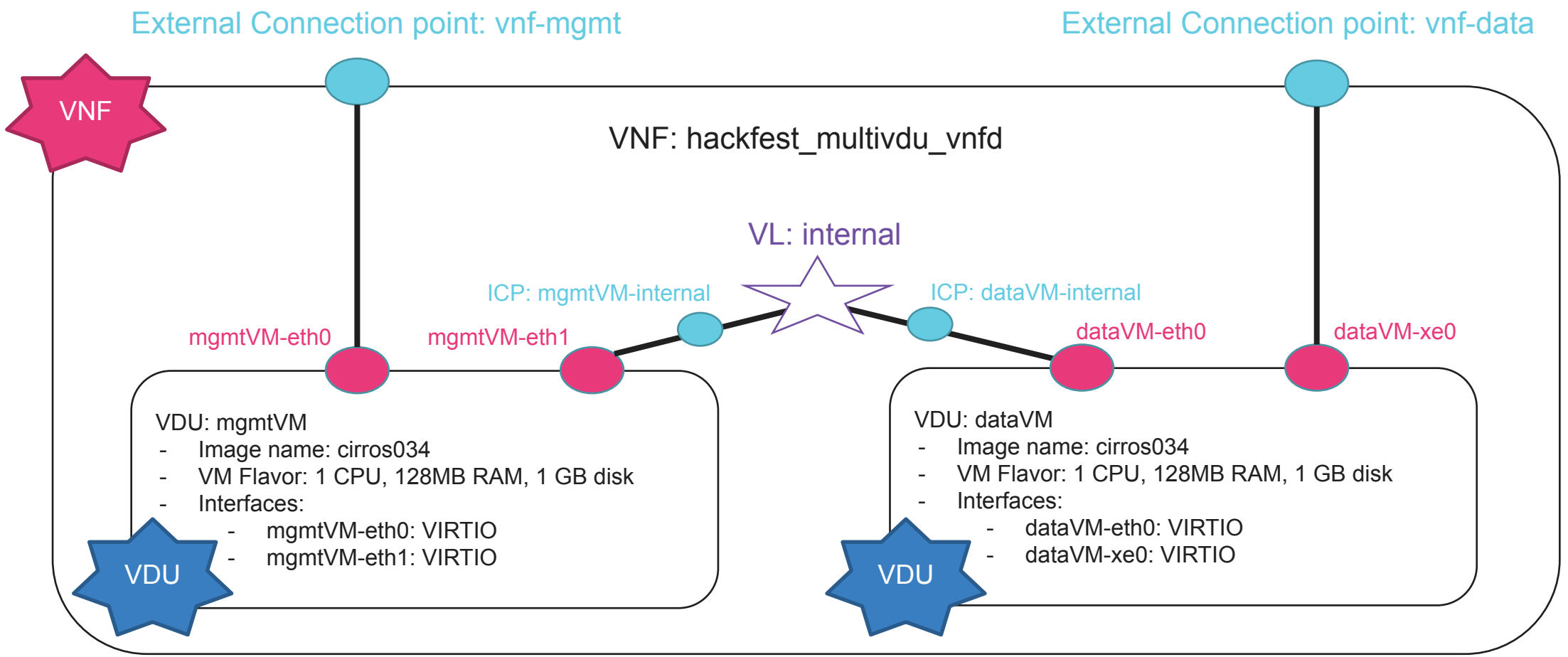
- Compose a VNF or NS graphically.





© ETSI 2020

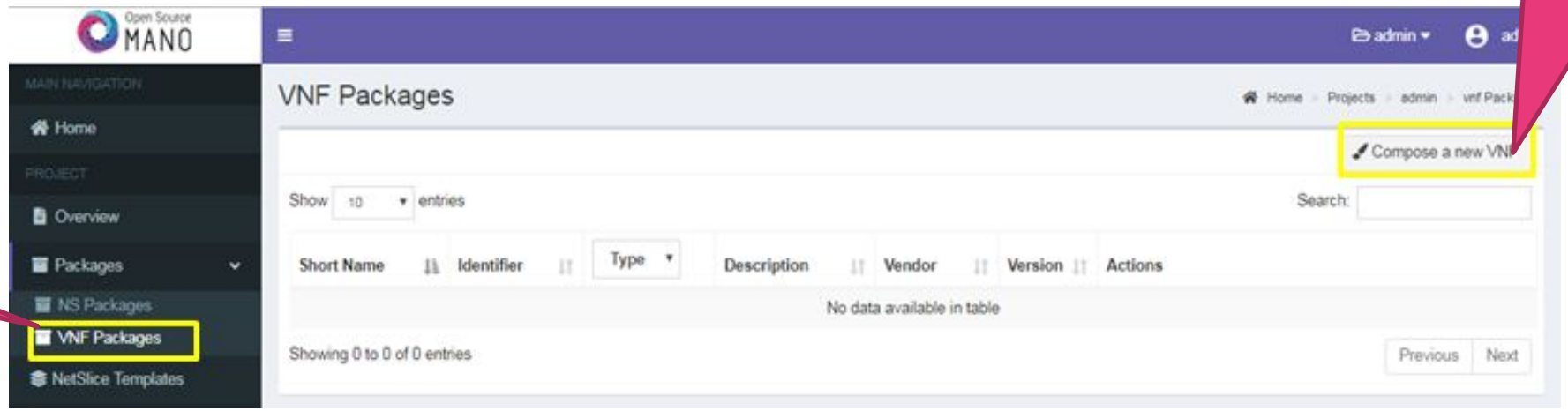
VNF diagram



© ETSI 2020

User Interface

- Steps:
  - Compose a new VNF



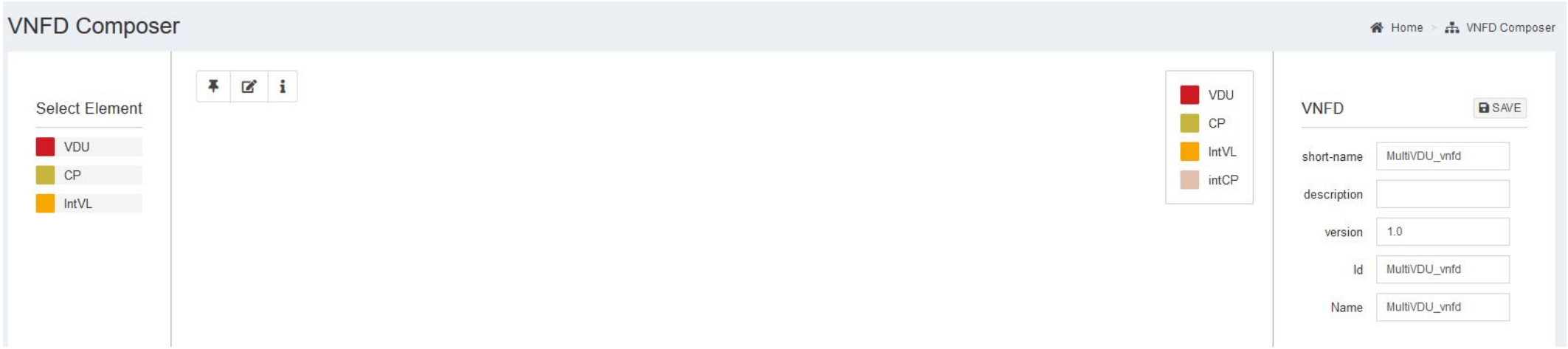
  - Create new Package



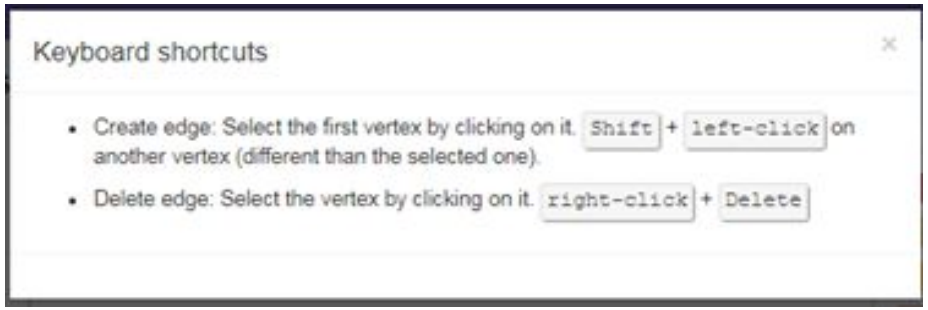
© ETSI 2020

VNFD Composer

- Steps
  - VNFD Composer



  - Keyboard shortcuts



© ETSI 2020

Creating the new multi-VDU VNF (1/4)

- Steps
  - Create VDUs   (Drag and drop)

VDU SAVE

count	1
description	
Name	mgmtVM
image	cirros034
Id	mgmtVM

VDU SAVE

count	1
description	
Name	dataVM
image	cirros034
Id	dataVM

- Edit the descriptor  to add the flavor:
  - `vm-flavor: {memory-mb: '128', storage-gb: '1', vcpu-count: '1'}`

Edit VNFD Descriptor

YAML JSON

```
vnfd:vnfd-catalog:
    vnfd:
    -   connection-point: []
        description: ''
        id: MultiVDU_vnfd
        internal-vld: []
        mgmt-interface:
            cp: ''
        name: MultiVDU_vnfd
        short-name: MultiVDU_vnfd
        vdu:
        -   count: '1'
            description: ''
            id: mgmtVM
            image: cirros034
            interface: []
            internal-connection-point: []
            monitoring-param: []
            name: mgmtVM
            vm-flavor: {}
            count: '1'
            description: ''
            id: dataVM
            image: cirros034
            interface: []
            internal-connection-point: []
            monitoring-param: []
            name: dataVM
            vm-flavor: {}
        version: '1.0'
```

© ETSI 2020

Creating the new multi-VDU VNF (2/4)

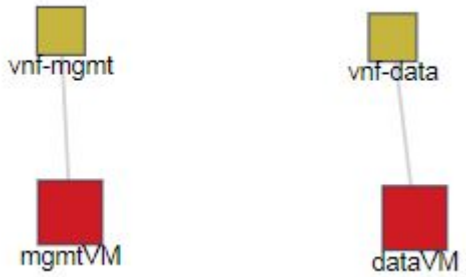
- Steps
  - Create Connection Points: 



(Drag and drop)



  - Link CPs with VDUs (Shift + Left Click)



© ETSI 2020

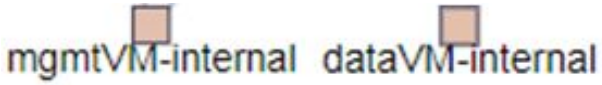
Creating the new multi-VDU VNF (3/4)

- Steps
  - Create Internal VL: 

(Drag and drop)



  - Link internal VL with VDUs (Shift + Left Click)
    - VNFD composer automatically create the internal connection points:



  - Edit the descriptor  to add the CP in mgmt-interface

Edit VNFD Descriptor

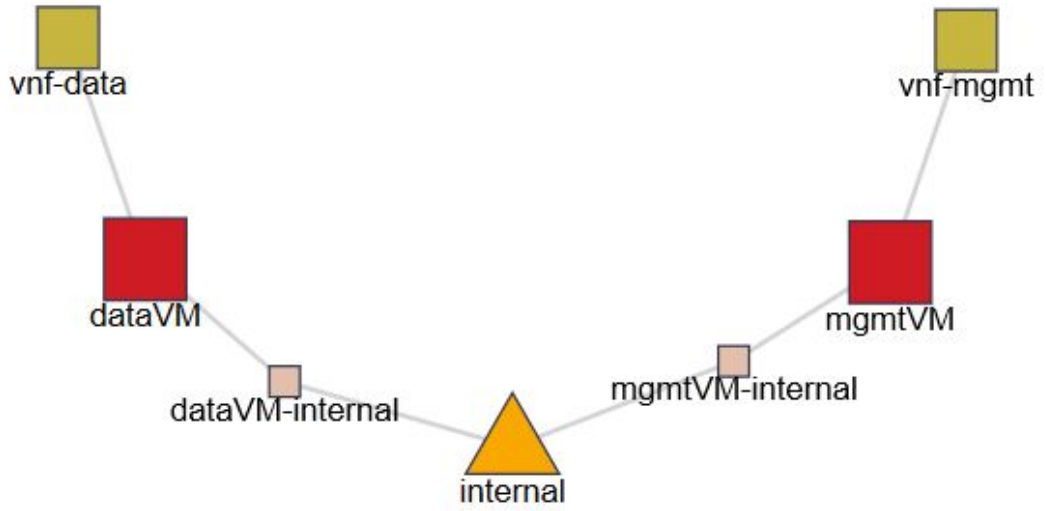
YAML JSON

```
1  vnfd:vnfd-catalog:
2      vnfd:
3      -   connection-point:
4          -   name: vnf-mgmt
5              type: VPORT
6          -   name: vnf-data
7              type: VPORT
8          description: ''
9          id: MultiVDU_vnfd
10         internal-vld:
11         -   id: internal
12             internal-connection-point:
13             -   id-ref: intcp_j5j9
14             -   id-ref: intcp_ih6y
15             ip-profile-ref: ''
16             name: internal
17             short-name: internal
18             type: ELAN
19         mgmt-interface:
20             cp: 'vnf-mgmt'
21         name: MultiVDU_vnfd
22         short-name: MultiVDU_vnfd
```

© ETSI 2020

Creating the new multi-VDU VNF (4/4)

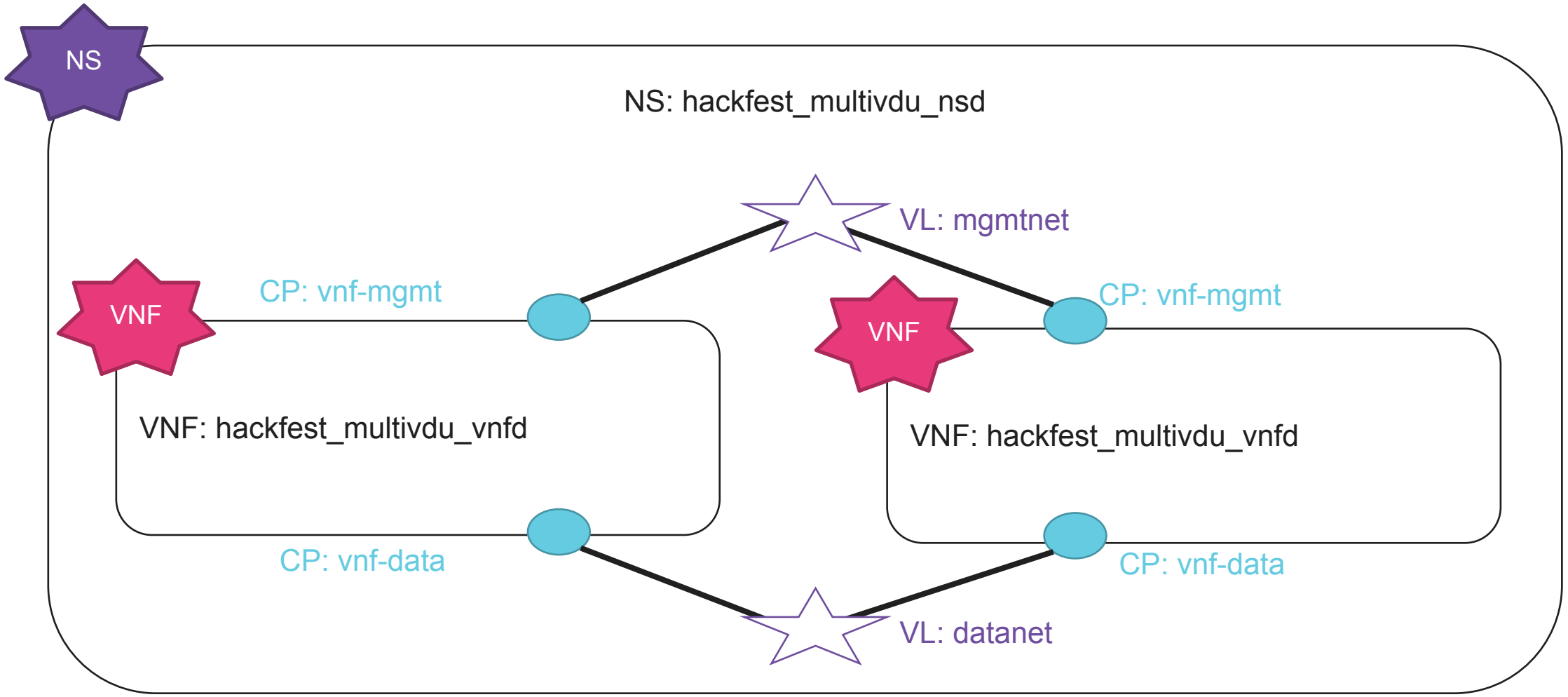
- Final Scenario multiVDU_vnfd



And finally, this is the sample file of Hackfest Multi VDU VNF Descriptor
https://osm-download.etsi.org/ftp/osm-6.0-six/7th-hackfest/packages/hackfest_multivdu_vnf.tar.gz

© ETSI 2020

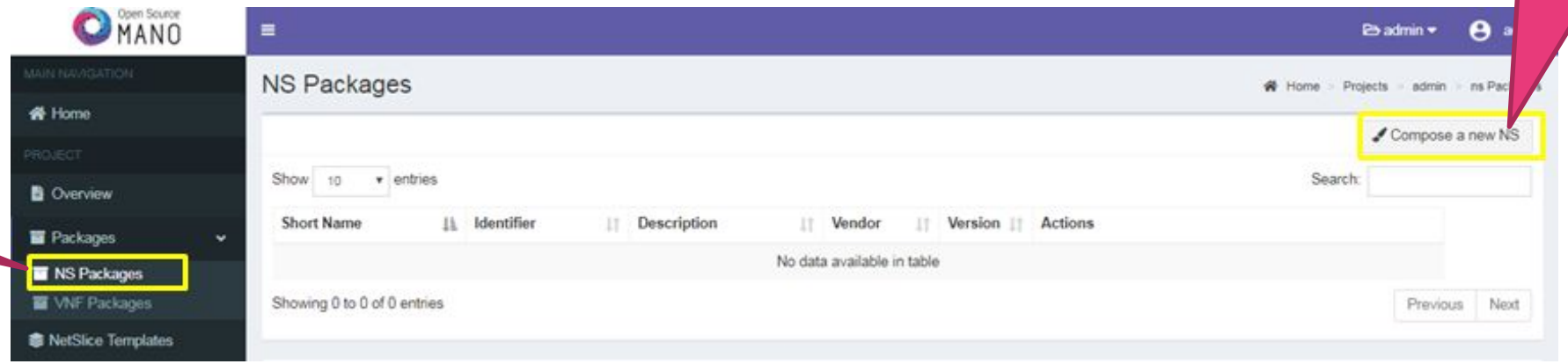
NS diagram



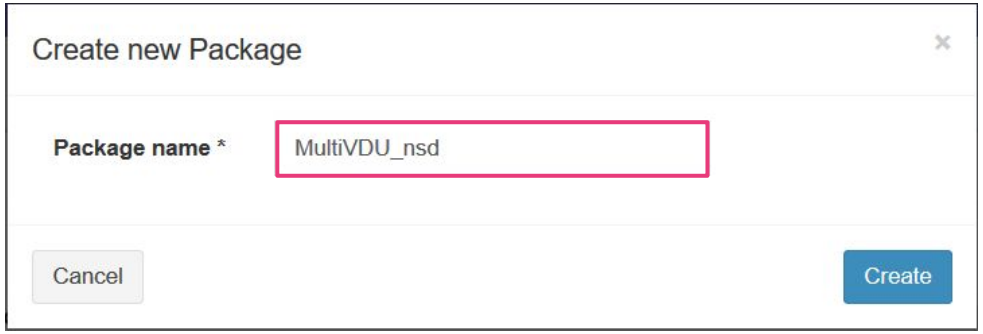
© ETSI 2020

User Interface

- Steps:
  - Compose a new NS



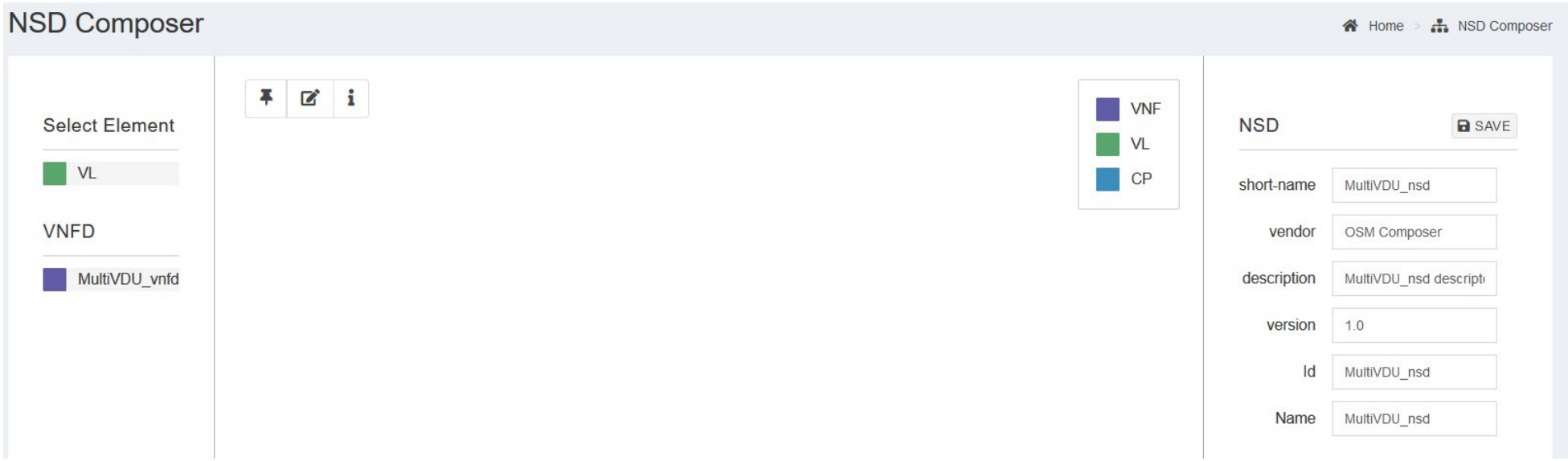
  - Create new Package



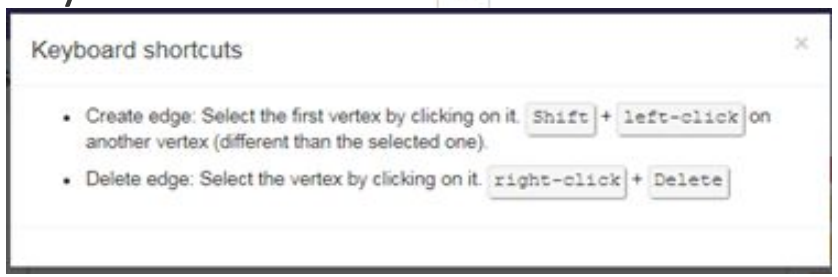
© ETSI 2020

NSD Composer

- Steps
  - NSD Composer



  - Keyboard shortcuts



© ETSI 2020

Creating the NSD (1/3)

- Select VNFs: 

(Drag and drop)

VNF

member-vnf-index 1

vnfd-id-ref MultiVDU_vnfd

VNF

member-vnf-index 2

vnfd-id-ref MultiVDU_vnfd

- Create VLs: 

(Drag and drop)

Virtual Link SAVE

Vim network name osm-ext

Name mgmtnet

Mgmt network true

Type ELAN

Id mgmtnet

Virtual Link SAVE

Vim network name

Name datanet

Mgmt network false

Type ELAN

Id datanet

© ETSI 2020

Creating the NSD (2/3)

- Steps
  - Link VLs with VNFs (Shift + Left Click)
    - You need to know the name for the CPs (vnf-data and vnf-mgmt)

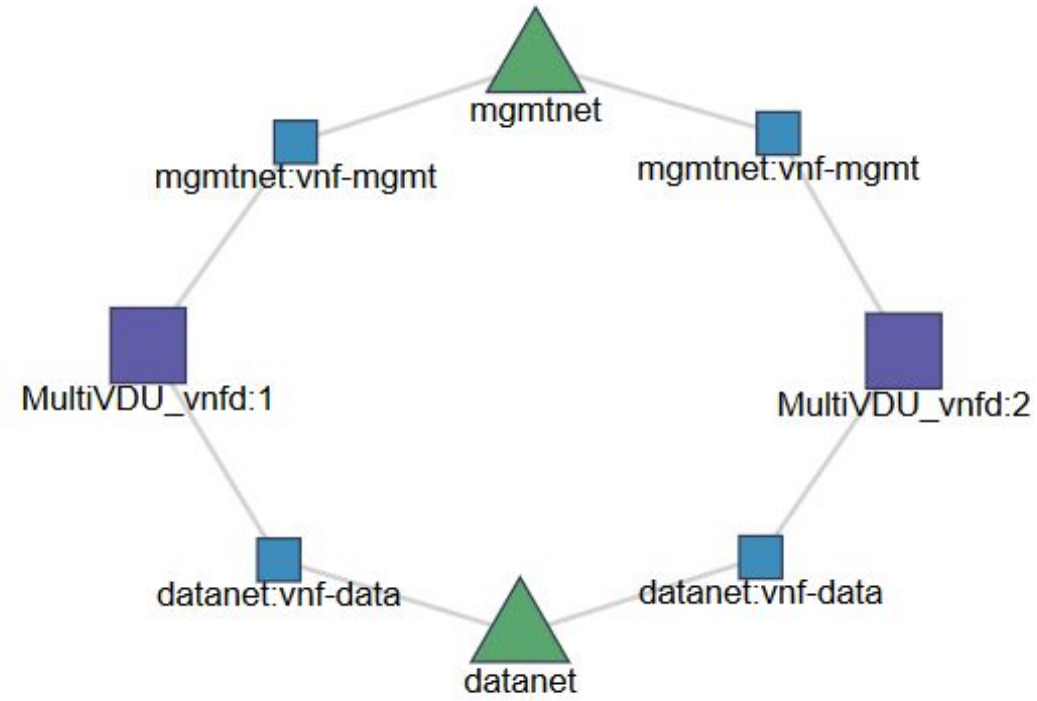




© ETSI 2020

Creating the NSD (3/3)

- Final Scenario multiVDU_nsd



And finally, this is the sample file of Hackfest Multi VDU VNF Descriptor
https://osm-download.etsi.org/ftp/osm-6.0-six/7th-hackfest/packages/hackfest_multivdu_ns.tar.gz

© ETSI 2020

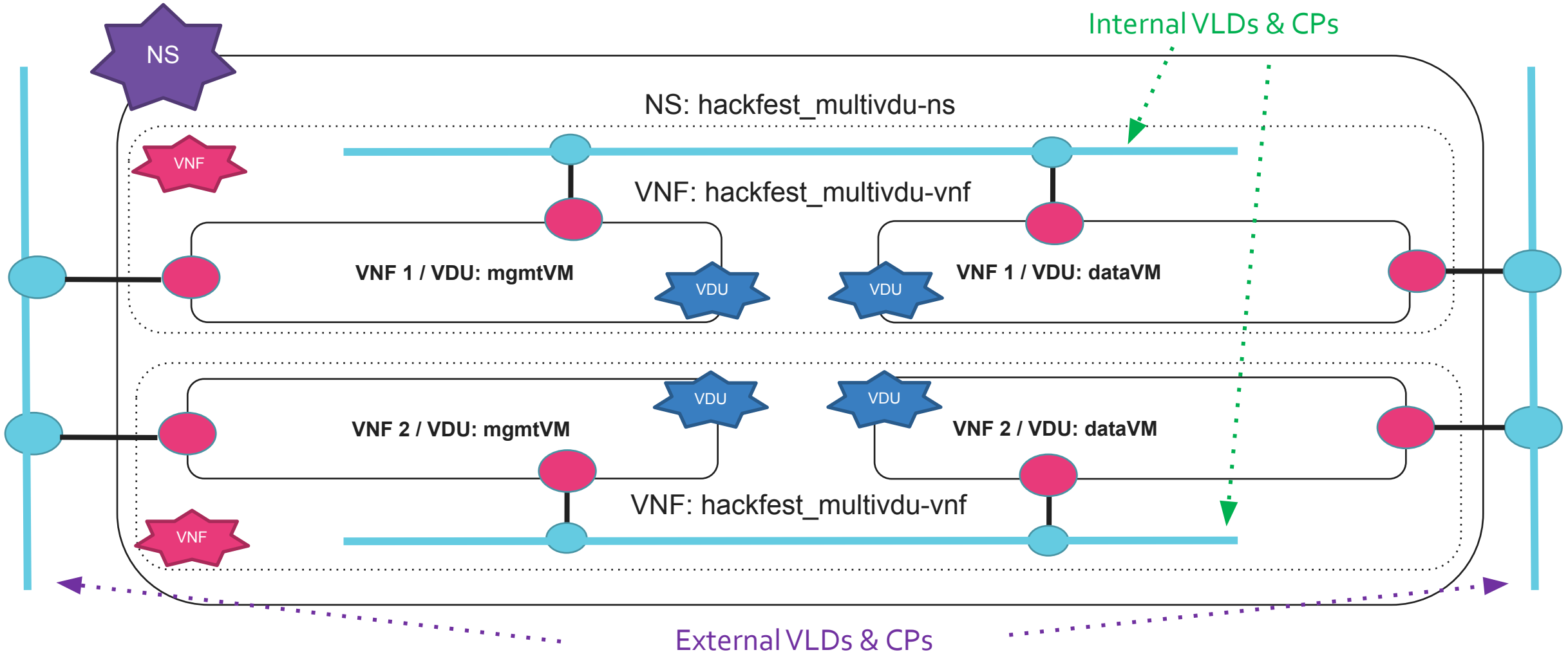
Deploying NS in the UI

- Visit the catalogue and confirm elements are created
- Launch the NS from the UI
  - Depending on the VIM, specify a VIM network name to map mgmtnet
  - If you need to change the VIM, change the network name using config:

    {vld: [{name: mgmtnet, vim-network-name: osm-ext}]}
- Click the info button to see the mgmt IP address of each VNF
- Connect to each VNF:
  - ssh cirros@<IP>
    - password: `cubswin:)`

© ETSI 2020

Final Multi-VDU Picture



© ETSI 2020

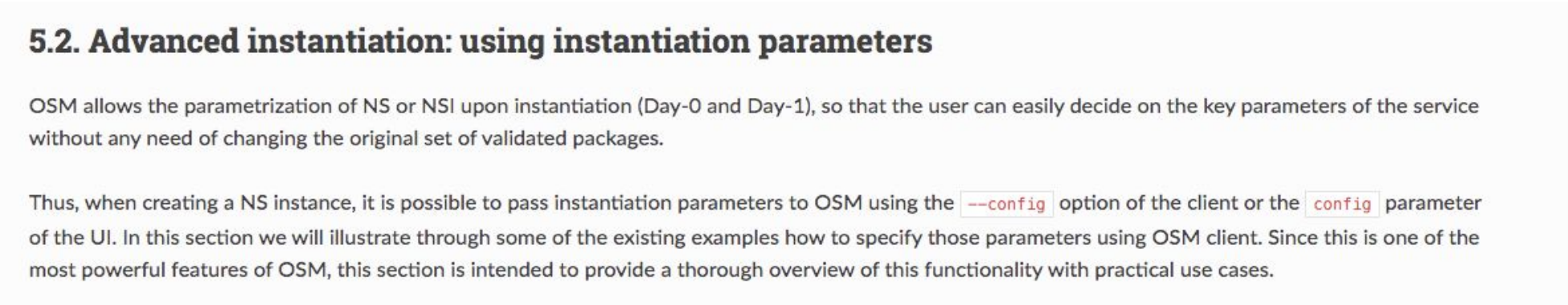
Advanced Network modelling!

The rich OSM Information Model and instantiation-time parameters allow for:

- **IP Profiles** for modifying the subnet parameters
- Assigning **fixed ip addresses** to connection points
- Assigning **fixed MAC addresses** to connection points
- Multi-site deployments

...and more!

More information here:
https://osm.etsi.org/docs/user-guide/05-osm-usage.html#advanced-instantiation-using-instantiation-parameters



5.2. Advanced instantiation: using instantiation parameters

OSM allows the parametrization of NS or NSI upon instantiation (Day-0 and Day-1), so that the user can easily decide on the key parameters of the service without any need of changing the original set of validated packages.

Thus, when creating a NS instance, it is possible to pass instantiation parameters to OSM using the `--config` option of the client or the `config` parameter of the UI. In this section we will illustrate through some of the existing examples how to specify those parameters using OSM client. Since this is one of the most powerful features of OSM, this section is intended to provide a thorough overview of this functionality with practical use cases.

© ETSI 2020

Open Source MANO

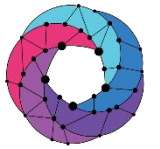
Presentation & Hands-on: Automating Day-0 configuration through cloud-init





© ETSI 2020

What is cloud-init and what can it be used for?



- It is a Linux package used to automate initial configuration of a VM
- VM requirements:
  - Cloud-init package
  - Cloud-init configuration (data source) via /etc/cloud/cloud.cfg
    - Config drive
    - Openstack metadata server
    - …
- What can be done?
  - Setting a default locale
  - Setting an instance hostname
  - Generating instance SSH private keys
  - Adding SSH keys to a user's .ssh/authorized_keys so they can log in
  - Setting up ephemeral mount points
  - Configuring network devices
  - Adding users and groups
  - Adding files
- Docs: http://cloudinit.readthedocs.io/en/latest/

- Cloud-init is available in Linux VMs and might be supported in other OS
- Not all VIMs support cloud-init via a metadata server

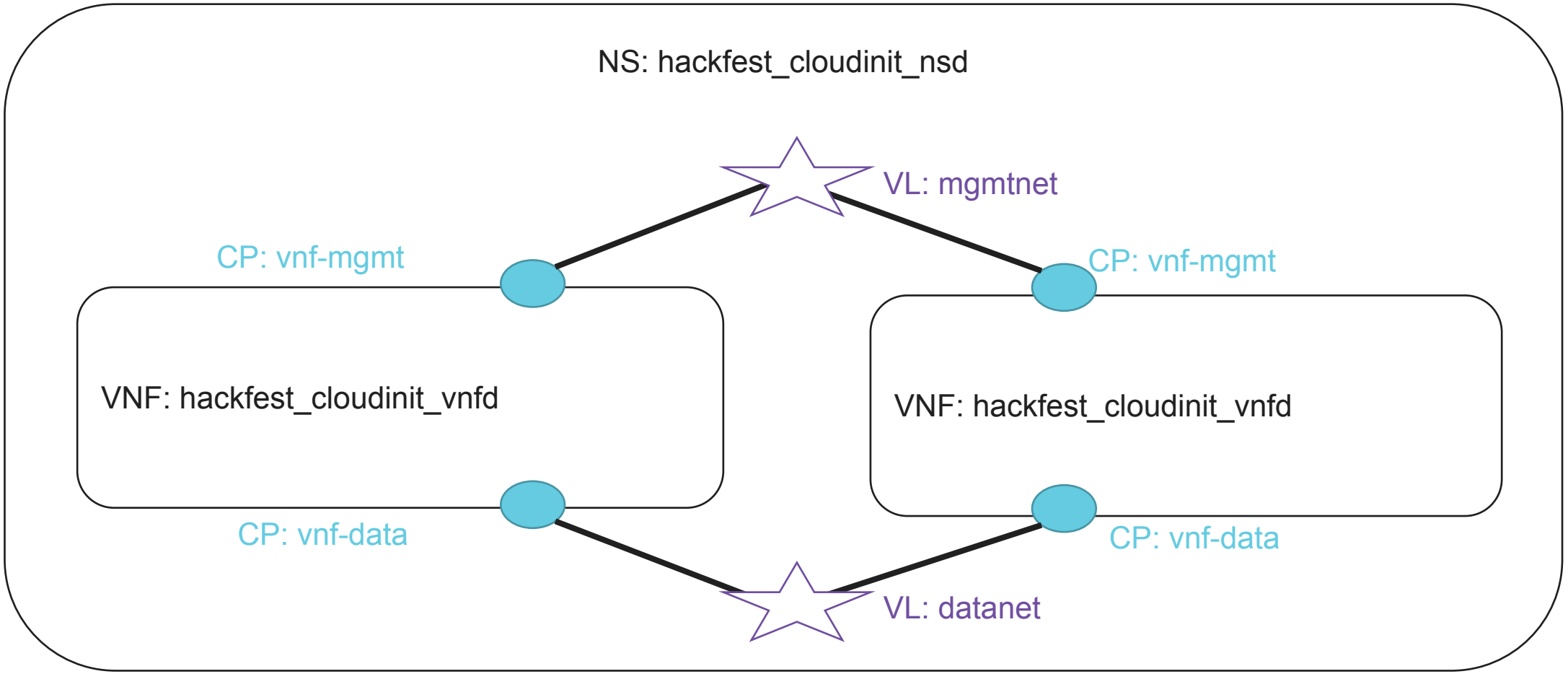
© ETSI 2020

Hands-on: Get and explore the packages

- VNF Package
  - https://osm-download.etsi.org/ftp/osm-6.0-six/8th-hackfest/packages/hackfest_cloudinit_vnf.tar.gz
- NS Package
  - https://osm-download.etsi.org/ftp/osm-6.0-six/8th-hackfest/packages/hackfest_cloudinit_ns.tar.gz

© ETSI 2020

NS diagram



© ETSI 2020

Hands-on: Let's explore the Cloud-init file

- Content:

```
#cloud-config
password: osm4u
chpasswd: { expire: False }
ssh_pwauth: True

write_files:
-   content: |
        # My new helloworld file

    owner: root:root
    permissions: '0644'
    path: /root/helloworld.txt
```

A password is added for the default user (‘ubuntu’).

Put whatever password you want!

A new file ‘/root/helloworld.txt’ will be created at VM creation to illustrate the way this feature works.

© ETSI 2020

Hands on: Deploying NS in the UI

Open Source MANO

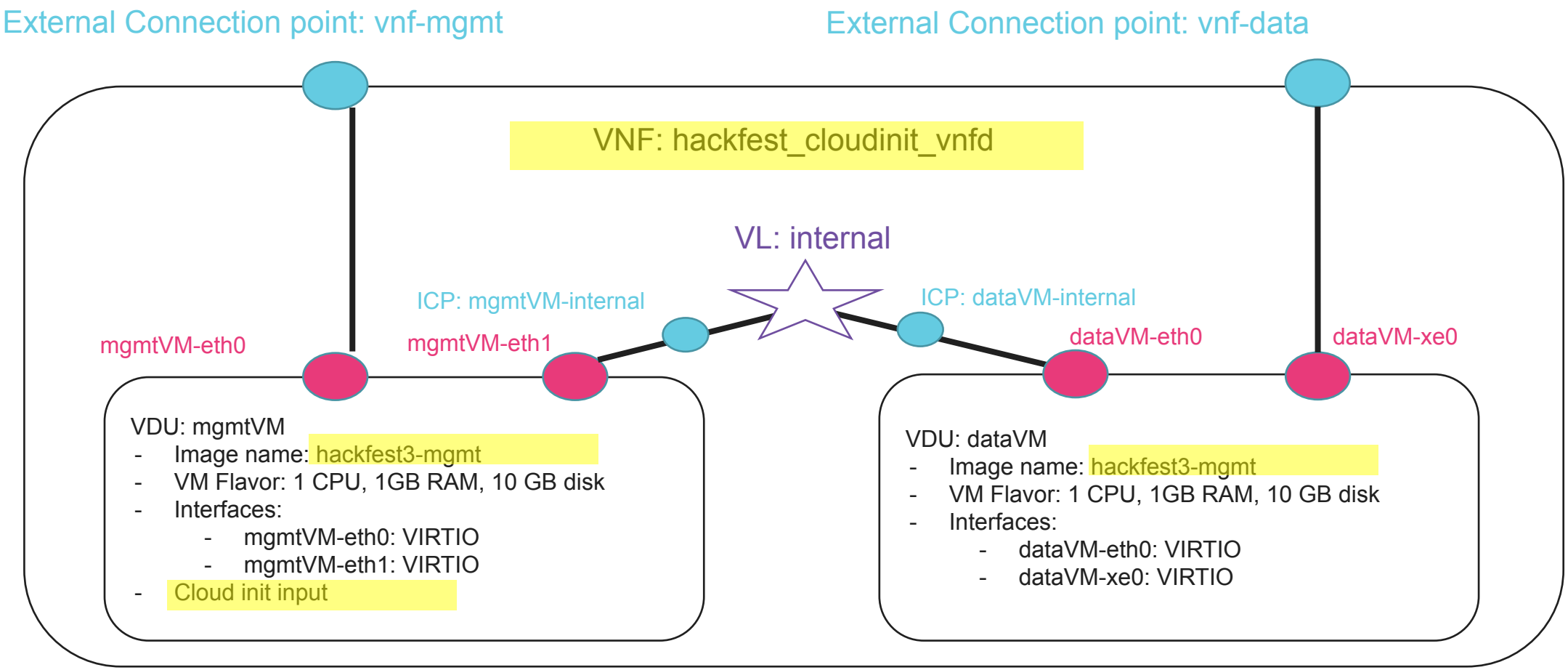
1. Onboard the VNFD and NSD
2. Select hackfest_cloudinit_nsd and instantiate it

   If you need to change the VIM, change the network name using config:

   {vld: [{name: mgmtnet, vim-network-name: osm-ext}]}
3. Complete the form
   a. Add a name to the NS
   b. Select the Datacenter where the NS will be deployed
4. Go to the dashboard to see the instance and get the mgmt IP address of the VNF
5. **Connect to each VNF: ssh ubuntu@<IP>**
6. Check that the cloud-config file was executed by entering your password and checking the file.

© ETSI 2020

Bonus Hands-on: Build the VNF!



© ETSI 2020

Bonus Hands-on: Creating the new CloudInit VNF (1/5)

Open Source MANO

- Use the tool to create a new VNFD called "hackfest_cloudinit_vnfd":
  osm package-create --image hackfest3-mgmt vnf hackfest_cloudinit

- Descriptor located at
  hackfest_cloudinit_vnf/hackfest_cloudinit_vnfd.yaml

- Add 2 Connection Points (external):
  - CONNECTION POINT 1:
    - name: vnf-mgmt
  - CONNECTION POINT 2:
    - name: vnf-data
- Add new VLD 'internal' to the VNF:
  - Name: internal
  - TYPE: ELAN
  - Refer to internal CPs we will define later

Internal VLD example

```
...
    mgmt-interface:
      cp: vnf-mgmt
    connection-point:
    - id: vnf-mgmt
      name: vnf-mgmt
      type: VPORT
    - id: vnf-data
      name: vnf-data
      type: VPORT
    internal-vld:
    - id: internal
      name: internal
      short-name: internal
      type: ELAN
      internal-connection-point:
      - id-ref: mgmtVM-internal
      - id-ref: dataVM-internal
...
```

© ETSI 2020

Bonus Hands-on: Creating the new CloudInit VNF (2/5)

- Add VDU1 in the VNF
  - Name: mgmtVM
  - Image: hackfest3-mgmt
  - VM Flavor:
    - VCPU COUNT: 1
    - MEMORY MB: 1024
    - STORAGE GB: 10
  - Add 1 internal connection point:
    - ID: mgmtVM-internal
    - Name: mgmtVM-internal
    - Type: VPORT
  - Add 2 interfaces to the VDU:
    - Interface 1:
      - Name: mgmtVM-eth0
      - Position: 1
      - Connection-point-type: EXTERNAL
      - EXTERNAL-CONNECTION-POINT-REF: vnf-mgmt
      - Virtual-interface:
        - Type: VIRTIO

```
...
      interface:
      - name: mgmtVM-eth0
        position: '1'
        type: EXTERNAL
        virtual-interface:
          type: VIRTIO
        external-connection-point-ref: vnf-mgmt
      - name: mgmtVM-eth1
        position: '2'
        type: INTERNAL
        virtual-interface:
          type: VIRTIO
        internal-connection-point-ref: mgmtVM-internal
      internal-connection-point:
      - id: mgmtVM-internal
        name: mgmtVM-internal
        short-name: mgmtVM-internal
        type: VPORT
...
```

- Interface 2:
  - Name: mgmtVM-eth1
  - Position: 2
  - Connection-point-type: INTERNAL
  - INTERNAL-CONNECTION-POINT-REF: mgmtVM-internal
  - Virtual-interface:
    - Type: VIRTIO

© ETSI 2020

Bonus Hands-on: Creating the new CloudInit VNF (3/5)

Open Source MANO

- Add VDU2 in the VNF
  - Name: dataVM
  - Image: hackfest3-mgmt
  - VM Flavor:
    - VCPU COUNT: 1
    - MEMORY MB: 1024
    - STORAGE GB: 10
  - Add 1 internal connection point:
    - ID: dataVM-internal
    - Name: dataVM-internal
    - Type: VPORT
  - Add 2 interfaces to the VDU:
    - Interface 1:
      - Name: dataVM-eth0
      - Position: 1
      - Connection-point-type: INTERNAL
      - INTERNAL-CONNECTION-POINT-REF: dataVM-internal
      - Virtual-interface:
        - Type: VIRTIO

```
...
      interface:
      - name: dataVM-etho
        position: '1'
        type: INTERNAL
        virtual-interface:
          type: VIRTIO
        internal-connection-point-ref: dataVM-internal
      - name: dataVM-xeo
        position: '2'
        type: EXTERNAL
        virtual-interface:
          type: VIRTIO
        external-connection-point-ref: vnf-data
      internal-connection-point:
      - id: dataVM-internal
        name: dataVM-internal
        short-name: dataVM-internal
        type: VPORT
...
```

- Interface 2:
  - Name: dataVM-xe0
  - Position: 2
  - Connection-point-type: EXTERNAL
  - EXTERNAL-CONNECTION-POINT-REF: vnf-data
  - Virtual-interface:
    - Type: VIRTIO

© ETSI 2020

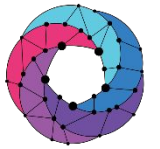
Bonus Hands-on: Creating the new CloudInit VNF (4/5)

- Download cloud-init file:

  https://osm-download.etsi.org/ftp/osm-6.0-six/8th-hackfest/other/cloud-config.txt
- Modify VDU mgmtVM in VNFD:
  - cloud-init-file: cloud-config.txt
- Add cloud-init file to hackfest_cloudinit_vnf/cloud_init folder

© ETSI 2020

Bonus Hands-on: Creating the new CloudInit VNF (5/5)

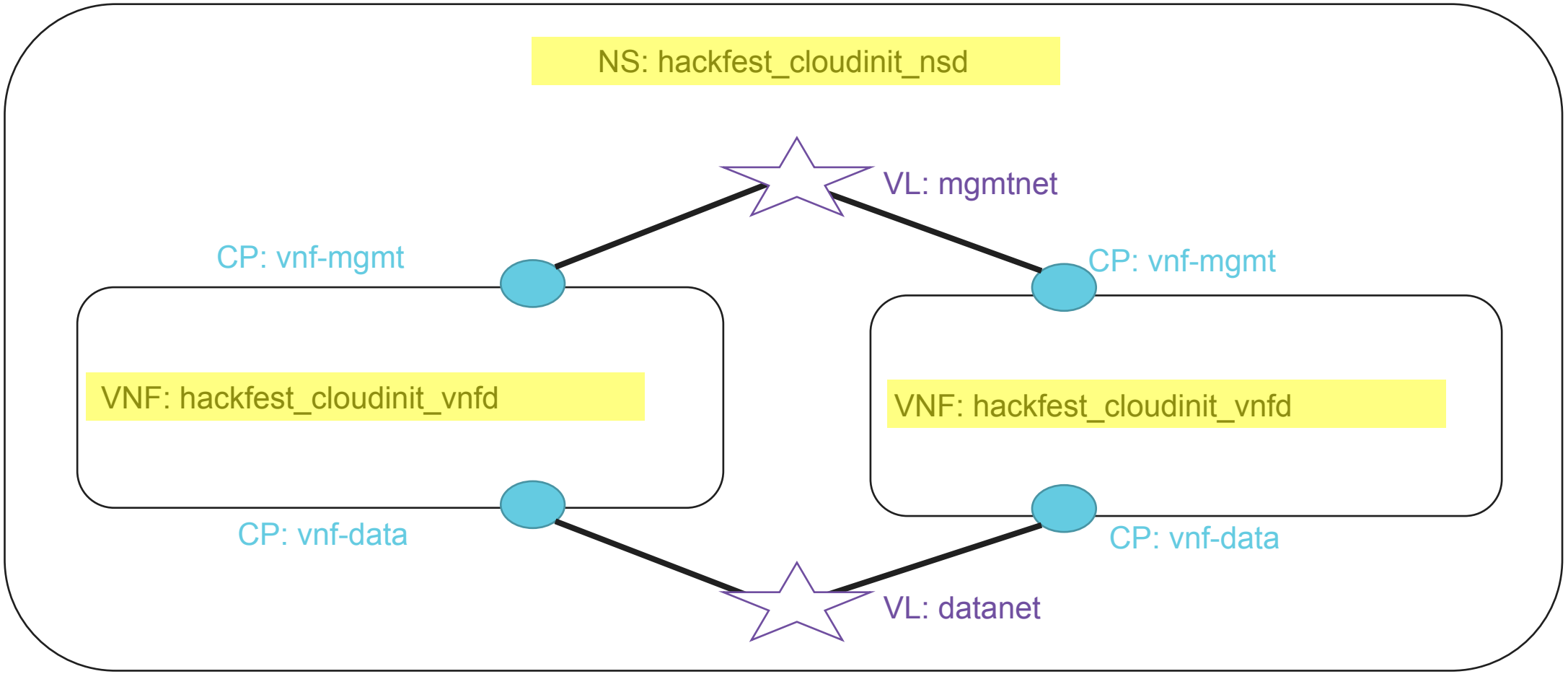


- Validate your descriptor using the tool:
  osm package-validate hackfest_cloudinit_vnf

- Generate VNF package:
  osm package-build hackfest_cloudinit_vnf

© ETSI 2020

Bonus Hands-on: Build the NS!



© ETSI 2020

Bonus Hands-on: Creating the NS (1/3)

- Use the tool to create a new NSD called "hackfest_cloudinit_nsd":
  osm package-create ns hackfest_cloudinit

- The descriptor is located at hackfest_cloudinit_ns/hackfest_cloudinit_nsd.yaml

- Specify constituent VNFs:
  constituent-vnfd:
  - vnfd-id-ref: hackfest_cloudinit-vnf
    member-vnf-index: '1'
  - vnfd-id-ref: hackfest_cloudinit-vnf
    member-vnf-index: '2'

© ETSI 2020

Bonus Hands-on: Creating the NS (1/3)

- Add first VLD:

```
- id: mgmtnet
      name: mgmtnet
      short-name: mgmtnet
      type: ELAN
      mgmt-network: 'true'
      vim-network-name: osm-ext
      vnfd-connection-point-ref:
      - vnfd-id-ref: hackfest_cloudinit-vnf
        member-vnf-index-ref: '1'
        vnfd-connection-point-ref: vnf-mgmt
      - vnfd-id-ref: hackfest_cloudinit-vnf
        member-vnf-index-ref: '2‘
        vnfd-connection-point-ref: vnf-mgmt
```

© ETSI 2020

Bonus Hands-on: Creating the NS (2/3)

- Add second VLD:

```
- id: datanet
      name: datanet
      short-name: datanet
      type: ELAN
      vnfd-connection-point-ref:
      - vnfd-id-ref: hackfest_cloudinit-vnf
        member-vnf-index-ref: '1'
        vnfd-connection-point-ref: vnf-data
      - vnfd-id-ref: hackfest_cloudinit-vnf
        member-vnf-index-ref: '2'
        vnfd-connection-point-ref: vnf-data
```

© ETSI 2020

Bonus Hands-on: Creating the NSD (3/3)

- Validate your descriptor using the tool:
  osm package-validate hackfest_cloudinit_ns

- Generate NS package:
  osm package-build hackfest_cloudinit_ns

© ETSI 2020

Open Source MANO

Presentation & Hands-on: Enabling high performance on VNFs - SDN-Assist





© ETSI 2020

EPA (Enhanced Platform Awareness)

- **EPA features** like the use of large hugepages memory, dedicated CPUs, strict NUMA node placement, the use of passthrough and SR-IOV interfaces, **can be used in OSM's VNF descriptors since Rel ZERO**.

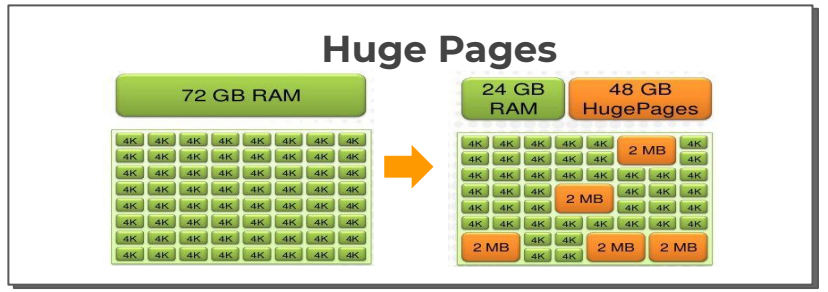
- If your VIM supports EPA, then you don't need to do anything extra to use it from OSM. VIM connectors in OSM take advantage of EPA capabilities if the VIM supports it. All you need to do is build your descriptors and deploy.

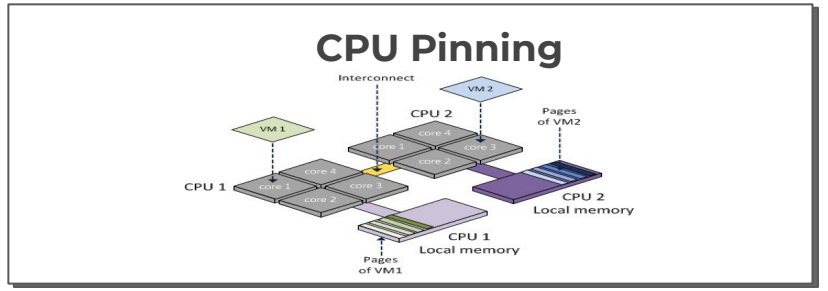
- Openstack configuration for EPA (reference guide)

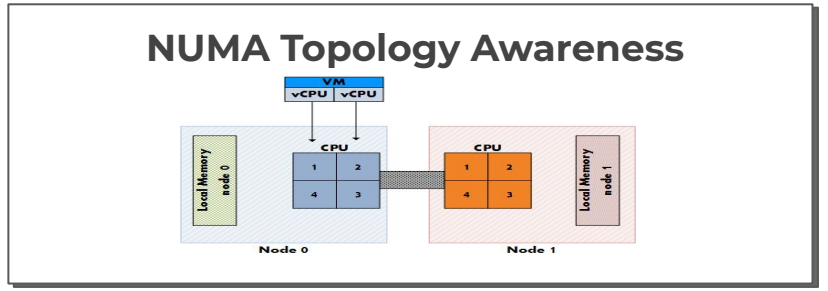
© ETSI 2020

EPA examples

- **Huge pages:** allocates big chunks of memory to avoid having too many entries to look up.
- **CPU Pinning:** binds a set of CPUs to a VM, improving the performance by avoiding degrading events such as cache misses.
- **NUMA (Non-Uniform Memory Access):** it provides separate memory for each processor to avoid the decrease on performance when several processors attempt to address the same memory.
- **SR-IOV (Single Root Input/Output Virtualization):** specification that allows the isolation of a PCI Express resource (e.g. a network interface) for manageability and performance reasons.



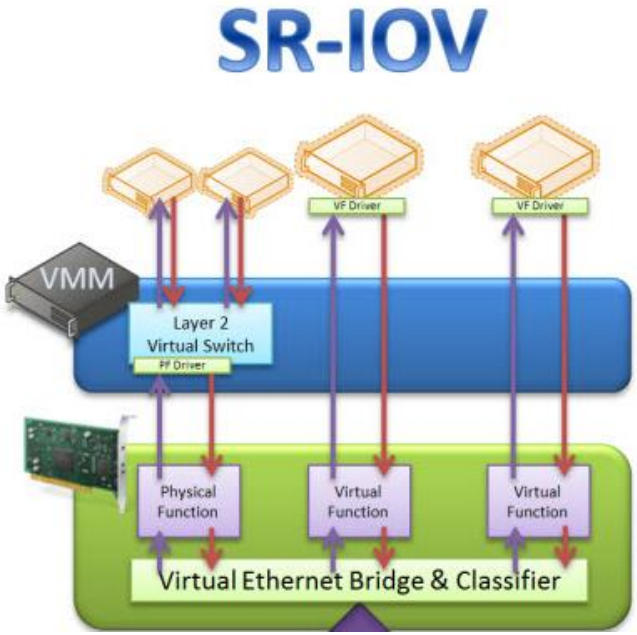


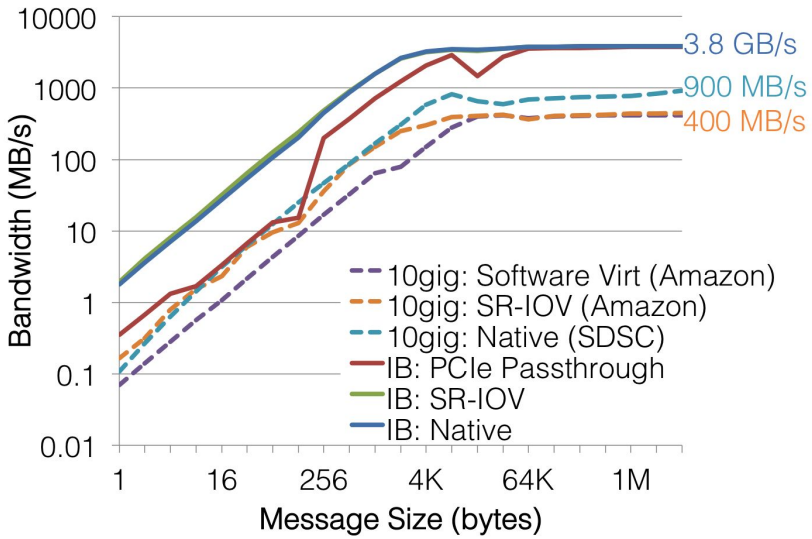


© ETSI 2020

What is SR-IOV?

SR-IOV allows VNFs to have direct access to a virtualized PCI of a NIC, thus giving it better throughput.





SDSC Early SR-IOV results

Note: *Enabling SR-IOV requires a node reload for reconfiguration of the IOMMU virtualization mode. It also requires physical interfaces to be dedicated to this feature.*

© ETSI 2020

Comparing SR-IOV provisioning (per VM/NS)

Open Source MANO

Provisioning SR-IOV with OpenStack

1. Create a network that uses the SR-IOV physnet

```
openstack network create --provider-network-type=vlan
--provider-physical-network=physnet2 --provider-segment=110
sriov-vlan110
```

2. Create the corresponding subnet

```
openstack subnet create --no-dhcp --network=sriov-vlan110
--subnet-range=11.0.0.0/24 sriov-vlan110-subnet
```

3. Create a port with the direct binding at the subnet

```
openstack port create --network sriov-vlan110 --vnic-type=direct
--binding-profile trusted=true sriov-vlan110-port01
```

4. Launch the VM using that port

```
openstack server create --flavor m1.large --image bionic --nic
port-id=sriov-vlan110-port01 vm01
```

5. Repeat as needed, then configure the switches

Automating SR-IOV connectivity for the complete NS with OSM

1. Add the VIM that includes the physnet (once)

```
osm vim-create --name VIMSRIOV --user user --password password --auth_url
http://172.21.7.5:5000/v3 --tenant tenant --account_type openstack --config
'{dataplane_physical_net: physnet2, microversion: 2.32}'
```

2. Model your VNF(s) to use SR-IOV instead of VIRTIO

```
interface:
-   name: dataVDU
    type: EXTERNAL
    virtual-interface:
        type: SR-IOV
```

3. Launch your NS, OSM will take care of everything, including VLD interconnectivity if the fabric was pre-configured with SDN Assist.

More info on SDN Assist:
https://osm.etsi.org/wikipub/index.php/EPA_and_SDN_assist

Note: In all cases, the image you use should have the driver for supporting the physical NIC

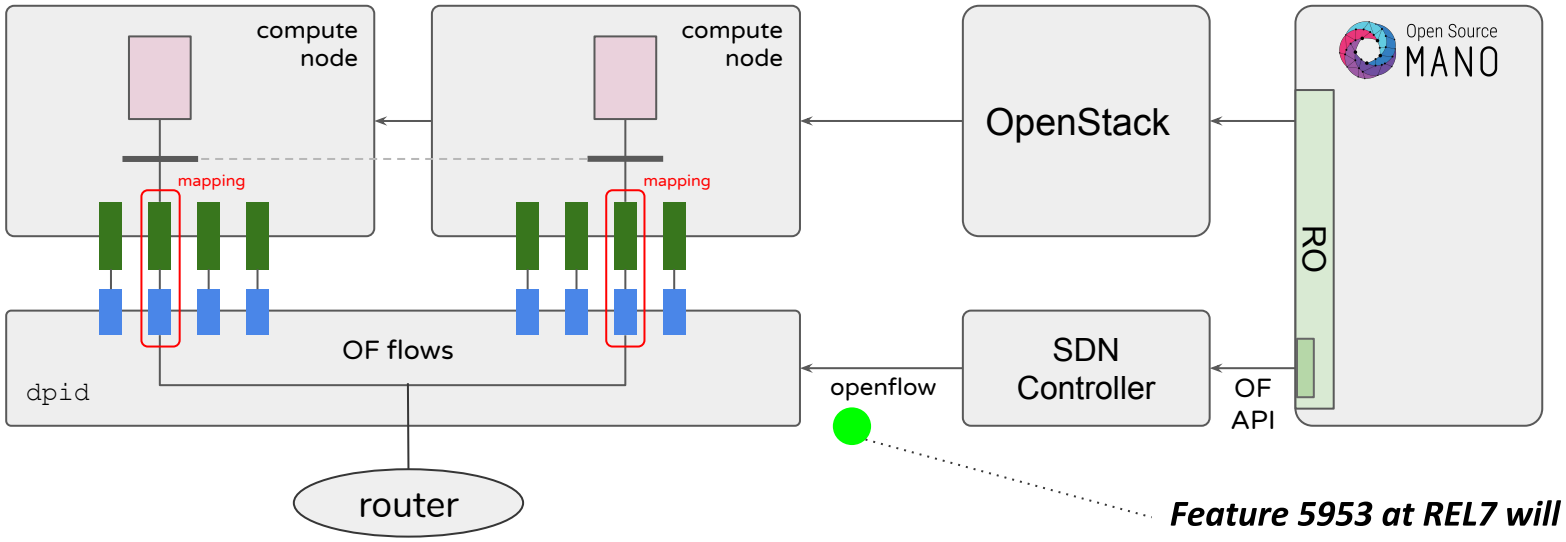
© ETSI 2020

SR-IOV + SDN Assist:
Enabling the chaining of **high performance** VNFs

1. OSM orquestrates SR-IOV

   → Proper assignment of I/O physical interfaces to the VM (VFs = Virtual Functions)

2. OSM SDN Assist gives the ability to create L2 connections between VFs
   - Interconnecting VMs
   - Attaching external traffic sources



Feature 5953 at REL7 will implement higher level protocols

© ETSI 2020

A basic example of SR-IOV in OSM

Open Source MANO

You could clone the Hackfest-basic as the baseline for building:

```
vnfd:vnfd-catalog:
    vnfd:
    -   connection-point:
        -   name: vnf-cp0
        -   name: vnf-cp1
        id: hackfest_basic-vnf-sriov
        mgmt-interface:
            cp: vnf-cp0
        vdu:
        -   cloud-init-file: cloud.config
            count: '1'
            id: hackfest_basic-VM
            image: bionic_wsvf
            interface:
            -   external-connection-point-ref: vnf-cp0
                name: vdu-eth0
                type: EXTERNAL
            -   external-connection-point-ref: vnf-cp1
                name: vdu-eth1
                type: EXTERNAL
                virtual-interface:
                    type: SR-IOV
            name: hackfest_basic-VM
            vm-flavor:
                memory-mb: '2048'
                storage-gb: '30'
                vcpu-count: '2'
```

http://osm-download.etsi.org/ftp/osm-6.0-six/8th-hackfest/packages/hackfest-basic-vnf-sriov.tar.gz

```
nsd:nsd-catalog:
    nsd:
    -   constituent-vnfd:
        -   member-vnf-index: '1'
            vnfd-id-ref: hackfest_basic-vnf-sriov
        id: hackfest_basic-ns-sriov
        name: hackfest_basic-ns-sriov
        version: '1.0'
        vld:
        -   id: mgmtnet
            mgmt-network: 'true'
            name: mgmtnet
            short-name: mgmtnet
            type: ELAN
            vim-network-name: management
            vnfd-connection-point-ref:
            -   member-vnf-index-ref: '1'
                vnfd-connection-point-ref: vnf-cp0
                vnfd-id-ref: hackfest_basic-vnf-sriov
        -   id: sriov_vld
            name: sriov_vld
            short-name: sriov_vld
            type: ELAN
            vnfd-connection-point-ref:
            -   member-vnf-index-ref: '1'
                vnfd-connection-point-ref: vnf-cp1
                vnfd-id-ref: hackfest_basic-vnf-sriov
```

http://osm-download.etsi.org/ftp/osm-6.0-six/8th-hackfest/packages/hackfest-basic-ns-sriov.tar.gz

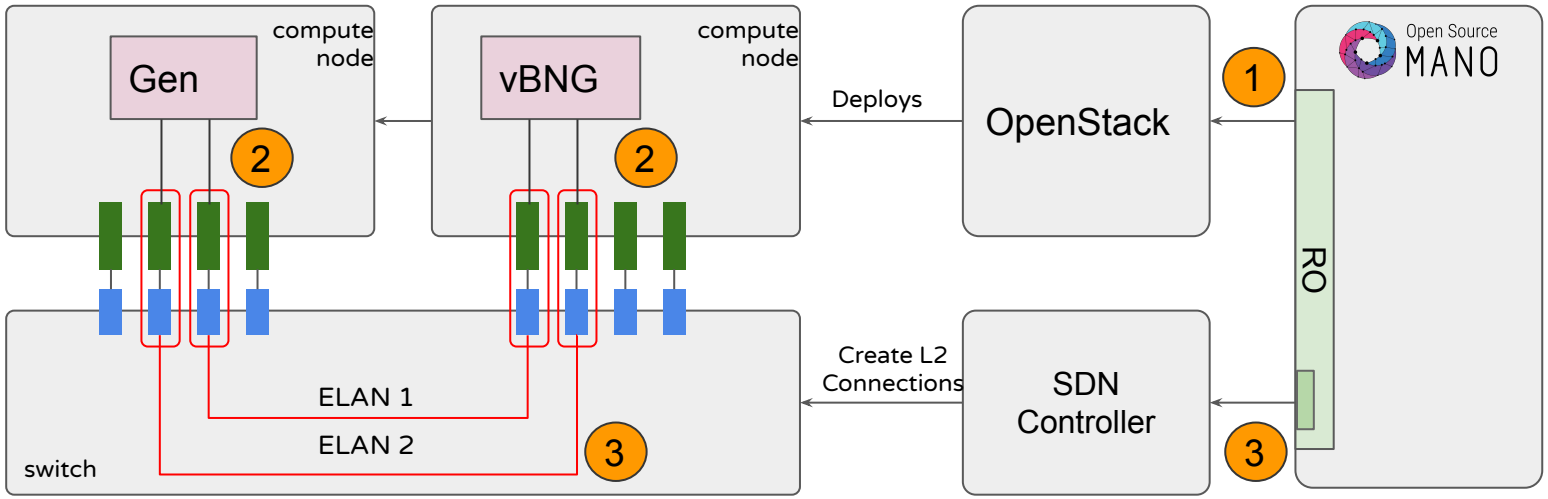
© ETSI 2020

DEMO: Deployment of two VNFs with EPA+Passthrough:
Underlay connectivity with ONOS SDN controller

1. OSM orchestrates two VNF: Gen & vBNG. Each one with:

   → EPA: 10 cores, 32 GB huge pages, 2 pci passthrough interfaces

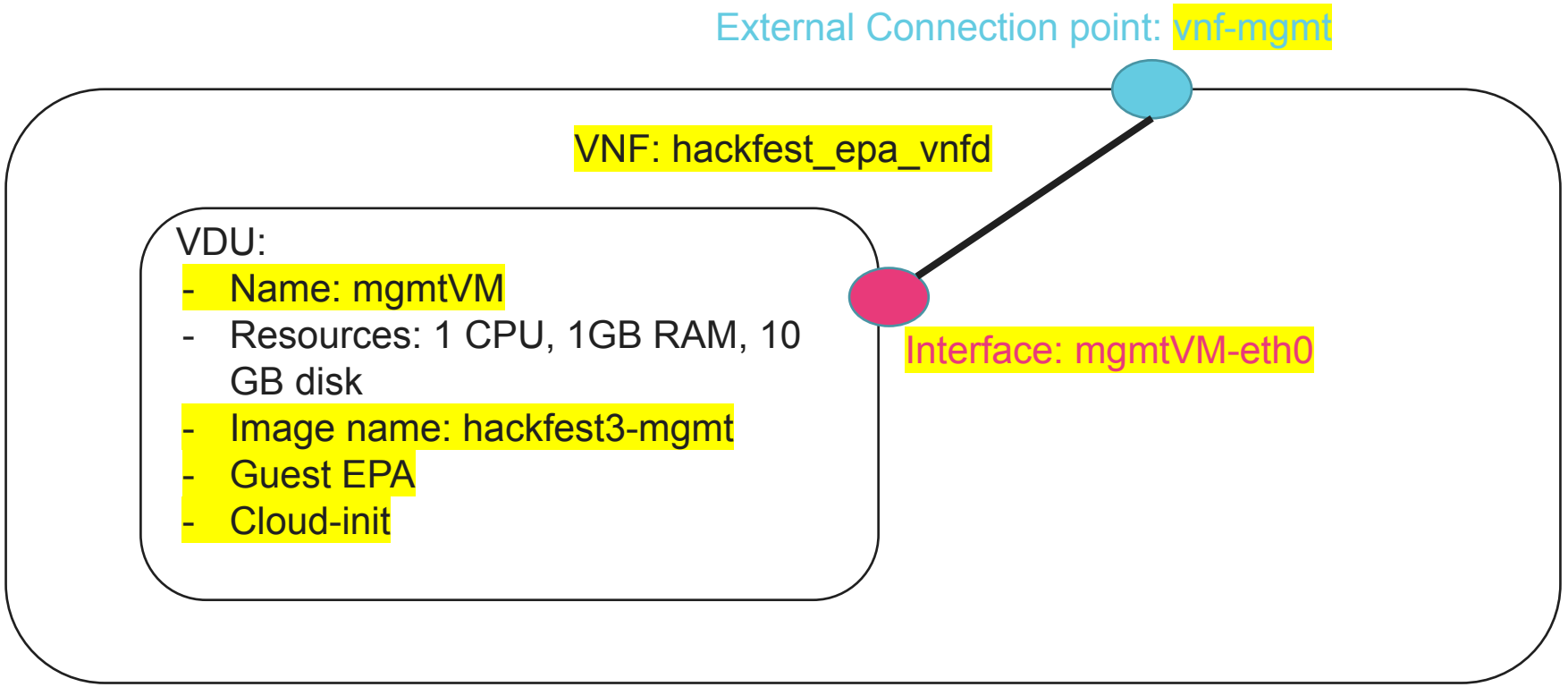
2. VIM deploys and allocate physical interfaces to the VM
3. OSM SDN Assist creates requeted L2 connectivity



© ETSI 2020

Bonus Hands-on: Enabling EPA in a VNF

- Starting from `hackfest_basic_vnfd` we can build the following VNF:



Note: changes highlighted in yellow

© ETSI 2020

Bonus Hands-on: Enabling EPA in a VNF

Open Source MANO

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. From UI, edit the VNFD to **lower the memory to 256MB** (to reduce exhaustion), and include:

```
    vdu:
    -   ...
        guest-epa:
            cpu-pinning-policy: DEDICATED
            mempage-size: LARGE
```

4. Launch it!

© ETSI 2020

Bonus Hands-on: Enabling EPA in a VNF

- In the VIM, the flavor of the VM should be a new custom one:

Instance Name	Image Name	IP Address	Flavour	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
epa-1-mgmtVM-1	hackfest3-mgmt	172.21.248.111	mgmtVM-flv-1	-	Active	nova	None	Running	9 hours, 5 minutes	Create Snapshot

- Authenticate in your VIM using Openstack CLI to show the EPA flavor specs.

```
osm_hackfest_40@osmhf:~$ source osm_hackfest_40.sh
osm_hackfest_40@osmhf:~$ openstack flavor show mgmtVM-flv | grep properties
| properties                  | hw:cpu_policy='dedicated', hw:cpu_sockets='1', hw:cpu_thread_policy='prefer',
hw:mem_page_size='large', hw:numa_mempolicy='strict', hw:numa_nodes='1' |
```

© ETSI 2020

Open Source MANO

Presentation & Hands-on: Monitoring & Auto-scaling VNFs





© ETSI 2020

Service Assurance MDG

Check the Monitoring and autoscaling user guide here!
https://osm.etsi.org/docs/user-guide/05-osm-usage.html#monitoring-and-autoscaling

Main components

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.
- Architecture needs a revisit based on expected use cases.

Prometheus

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

Auxiliary/Optional Tools

becoming default

Grafana

- Integrates seamlessly with Prometheus.
- Great tool for enhancing usability of the system's Service Assurance → should be included by default?

ELK

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

© ETSI 2020

Metrics collection

Prometheus collects the following metrics from “MON Exporter”

Metrics Collection @ OSM				
Metric	**Collection type**	**Behavior**	**KPI**	**Labels**
VIM Status	Infrastructure	By default	status (up/down)	vim_id
SDNC Status			status (up/down)	sdnc_id
VM Status	VNF		status (up/down)	nsr_id, vnf_member_index, vdu_name
VDU CPU Utilization		Enabled by descriptor	utilization, rate, etc.	
VDU Memory Utilization				
VDU Packet forwarding				
VNF Metrics through Juju *(to be migrated to Prometheus exporters)*				

© ETSI 2020

Metrics collection

- VDU Metric Collection from VIM

```
vdu:
   id: hackfest_basic-VM
   ...
   monitoring-param:
         - id: "cpu_util"
           nfvi-metric: "cpu_utilization"
...
monitoring-param:
-    id: "vnf_cpu_util"
     name: "vnf_cpu_util"
     aggregation-type: AVERAGE
     vdu-monitoring-param:
       vdu-ref: "hackfest_basic-VM"
       vdu-monitoring-param-ref: "cpu_util"
```

nfvi-metric corresponds to a OSM metric name which maps to the corresponding metric in each supported VIM

© ETSI 2020

Autoscaling

- Scaling descriptors can be included and be tied to automatic reaction to VIM/VNF metric thresholds. An internal alarm manager is supported, so that both VIM and VNF metrics can trigger threshold-violation alarms and scaling actions.

```
scaling-group-descriptor:
-   name: "vdu_autoscale"
    min-instance-count: 0
    max-instance-count: 10
    scaling-policy:
    -   name: "cpu_util_above_threshold"
        scaling-type: "automatic"
        threshold-time: 10
        cooldown-time: 120
        scaling-criteria:
        -   name: "cpu_util_above_threshold"
            scale-in-threshold: 10
            scale-in-relational-operation: "LT"
            scale-out-threshold: 60
            scale-out-relational-operation: "GT"
            vnf-monitoring-param-ref: "vnf_cpu_util"
    vdu:
    - vdu-id-ref: hackfest_basic_metrics-VM
      count: 1
```

© ETSI 2020

VNF Alarms (new)

- Alarms based on metric thresholds can be sent to webhooks

```
vdu:
-   alarm:
      -   alarm-id: alarm-1
          operation: LT
          value: 20
          actions:
            alarm:
              - url: https://webhook.site/1111
            ok:
              - url: https://webhook.site/2222
            insufficient-data:
              - url: https://webhook.site/3333
          vnf-monitoring-param-ref: vnf_cpu_util
```

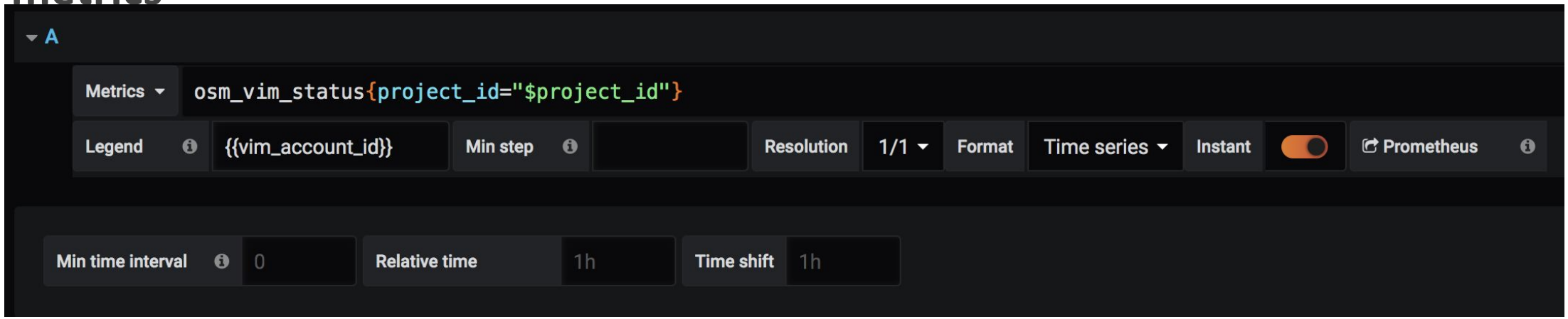
© ETSI 2020

Project-scoped VIM/VNF Metrics (new)

Open Source MANO

Objective: Follow RBAC structure for metric consumption.

- Prometheus does not support multi-tenancy, other projects need to be explored (e.g. Cortex)
- **Short-term implementation is a label for project_id in all Prometheus metrics**



Find your dashboard! (beta) → http://172.21.248.9:3000/dashboards

© ETSI 2020

Grafana Dashboard Automation

Open Source MANO

Objective: adding to the previous feature, a new “MON Dashboarder” component will take care of dashboard “lifecycle”.

Updates in...	...automates these dashboards...	...and these Grafana resources
OSM installation	System Metrics, Admin Project-scoped → ***beta available***	Admin-privileges
OSM Projects	Project-scoped (Grafana “team” privileges) → ***beta available***	Grafana “team” privileges
OSM Users	-	Grafana users to teams
OSM Network Services	NS-scoped → ***beta available***	-

...Let’s play with this!

© ETSI 2020

Hands-on: Metrics and (auto)dashboards!

Open Source MANO

1. From your SSH console, download new descriptors and upload them to OSM

```
wget http://osm-download.etsi.org/ftp/osm-6.0-six/8th-hackfest/packages/hackfest_basic_metrics_vnfd.tar.gz
wget http://osm-download.etsi.org/ftp/osm-6.0-six/8th-hackfest/packages/hackfest_basic_metrics_nsd.tar.gz
osm vnfd-create hackfest_basic_metrics_vnfd.tar.gz
osm nsd-create hackfest_basic_metrics_nsd.tar.gz
```

2. Instantiate your NS

```
osm ns-create --ns_name hfmetrics_XX --nsd_name hackfest_basic-ns-metrics --vim_account
openstack-site-hackfest-X --config '{vld: [ {name: mgmtnet, vim-network-name: osm-ext} ] }'
```

3. Look for a new dashboard dedicated to your NS! → http://172.21.248.9:3000/dashboards

© ETSI 2020

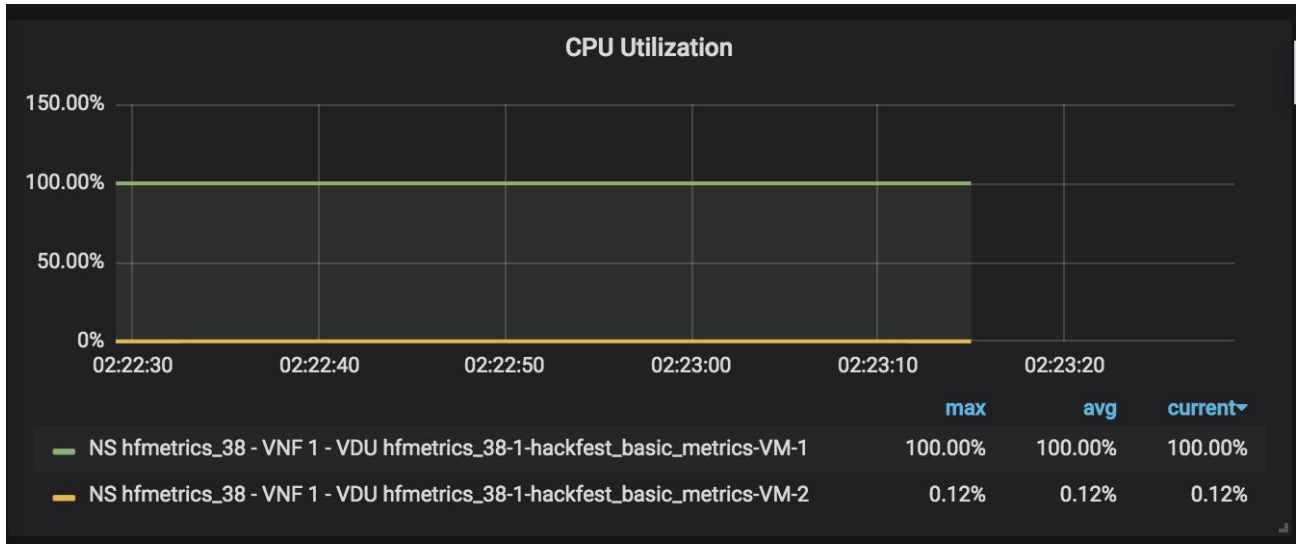
Hands-on: Autoscaling

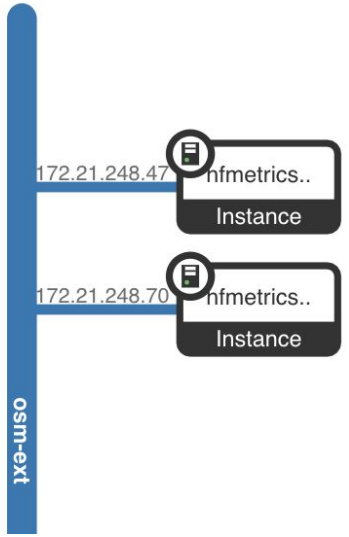
1. Access your VM and stress it out!

```
osm vnf-list                  # to find the IP address
ssh ubuntu@172.21.248.XX      # password: osm4u

yes > /dev/null &    # 4 or 5 times!
```

2. Wait for a bit (5 to 10 minutes due to current collection period), and watch it scale!





© ETSI 2020

Open Source MANO

Presentation & Hands-on: Orchestrating Network Slices with OSM



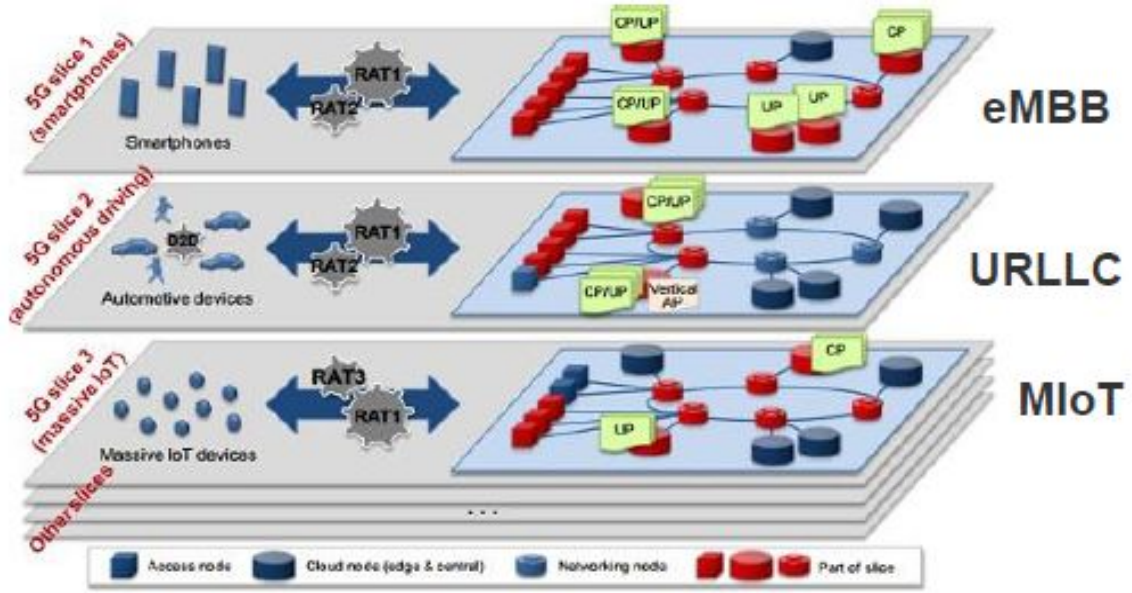


© ETSI 2020

Network Slicing Overview

In the scope of 5G, network slicing allows the network operator to deploy independent PLMNs with different capabilities

Different network slices addressing different types of usage requiring different levels of functionality, performance and reliability.



Source: NGMN

© ETSI 2020

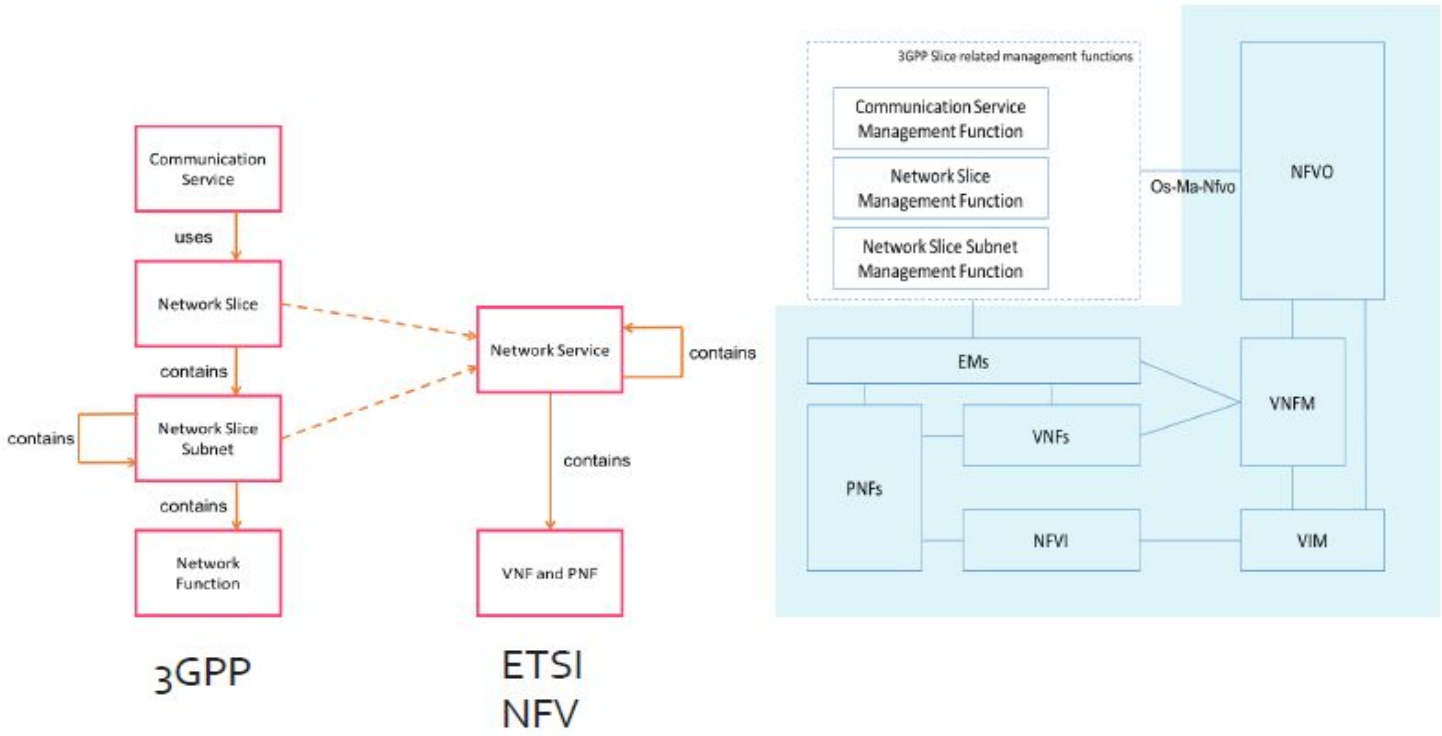
ETSI NFV Framework for Network Slicing

Analysis of 3GPP and alignment of NFV architecture in the ETSI GR NFV-EVE 012

Considerations:

- Network Slice Subnet can be considered as an NFV Network Service.

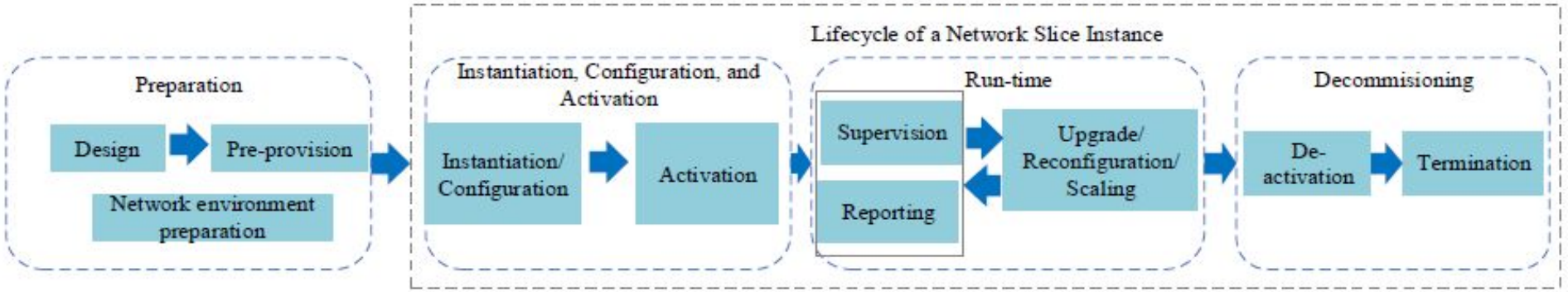
- Network Functions can be described as VNF and PNF



© ETSI 2020

Network Slicing in OSM

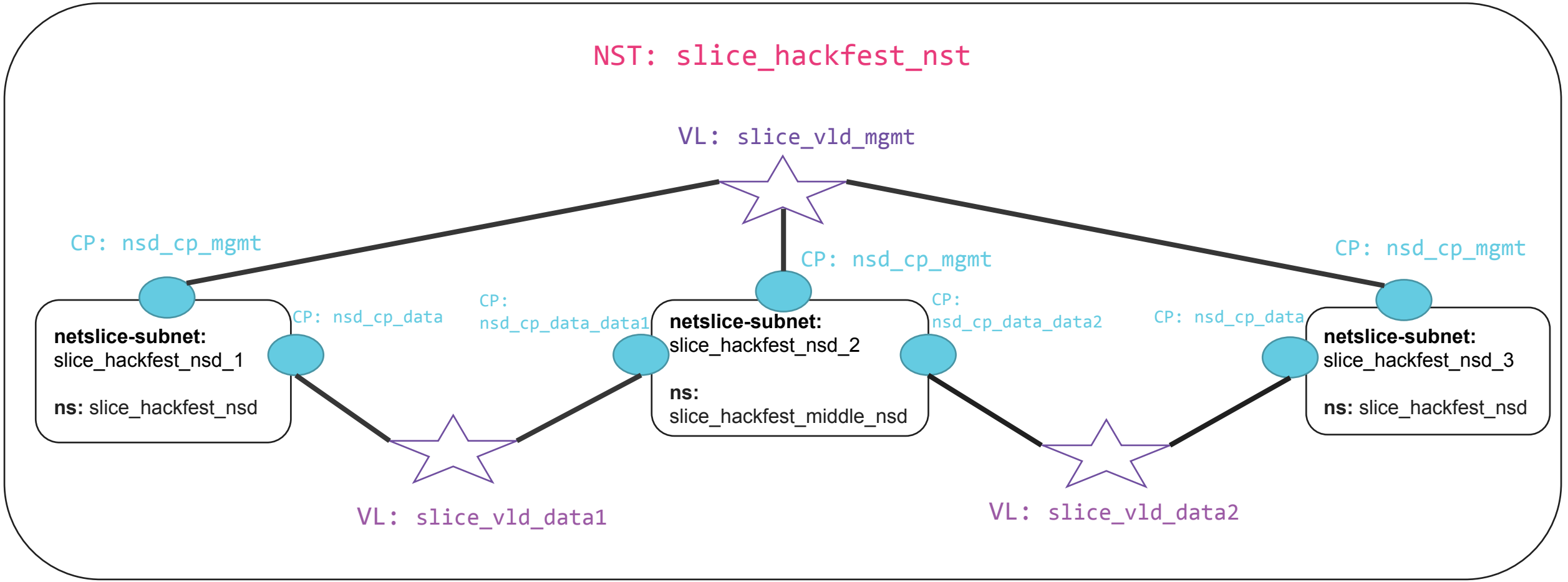
- **Network slice instance**: a set of network functions and the resources for these network functions which are arranged and configured, forming a complete logical network to meet certain network characteristics.



- The NSI contains:
  - NFs (e.g. belonging to AN and CN)
  - Information relevant to the interconnections between these NFs like topology of connections.
  - Individual link requirements (e.g. QOS attributes)
    - **The NSI is defined by a Network Slice Template.**

© ETSI 2020

NST diagram



© ETSI 2020

Network Slice Template Requirements

VNFD:

- slice_hackfest_vnf.tar.gz
- slice_hackfest_middle_vnf.tar.gz

NSD:

- slice_hackfest_ns.tar.gz
- slice_hackfest_middle_ns.tar.gz

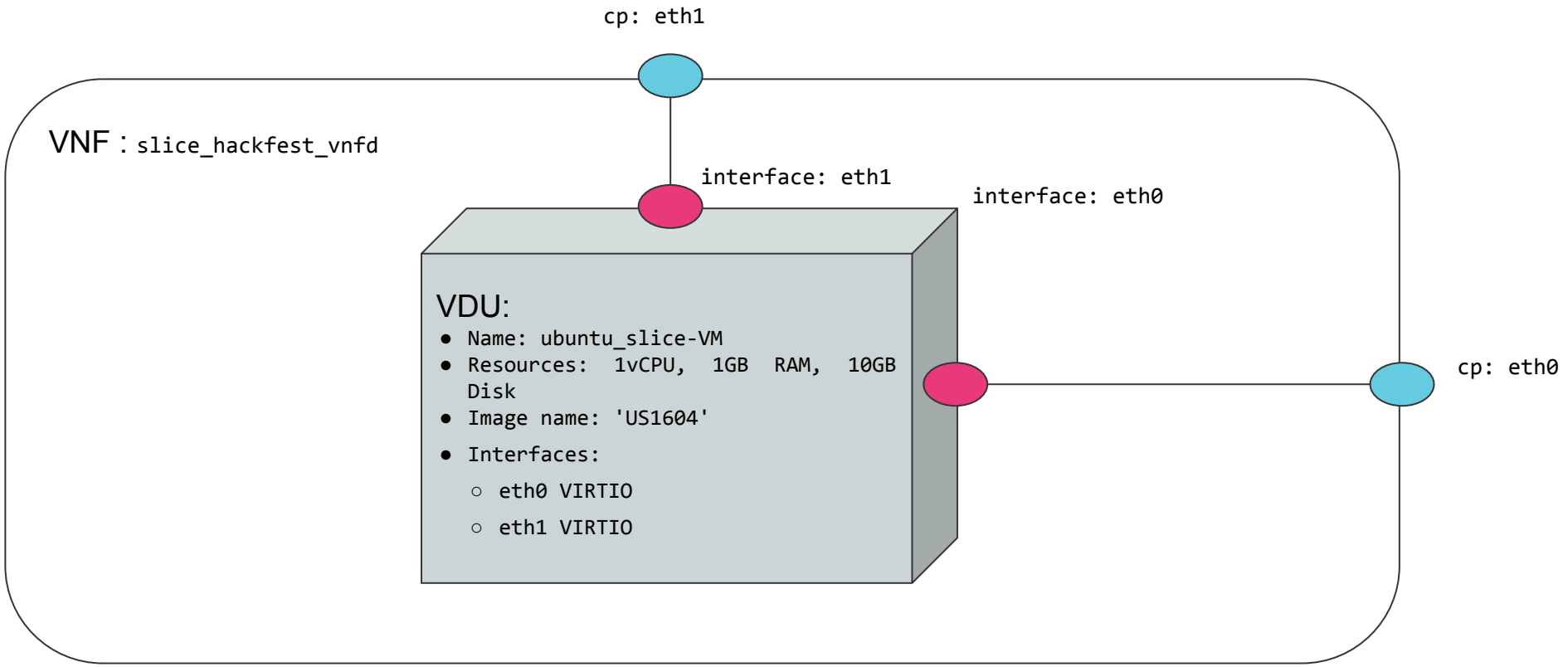
NST:

- slice_hackfest_nst.yaml
- slice_hackfest2_nst.yaml

- **The descriptors for the slice hackfest session are available:**
  - **https://osm-download.etsi.org/ftp/osm-6.0-six/8th-hackfest/packages/**

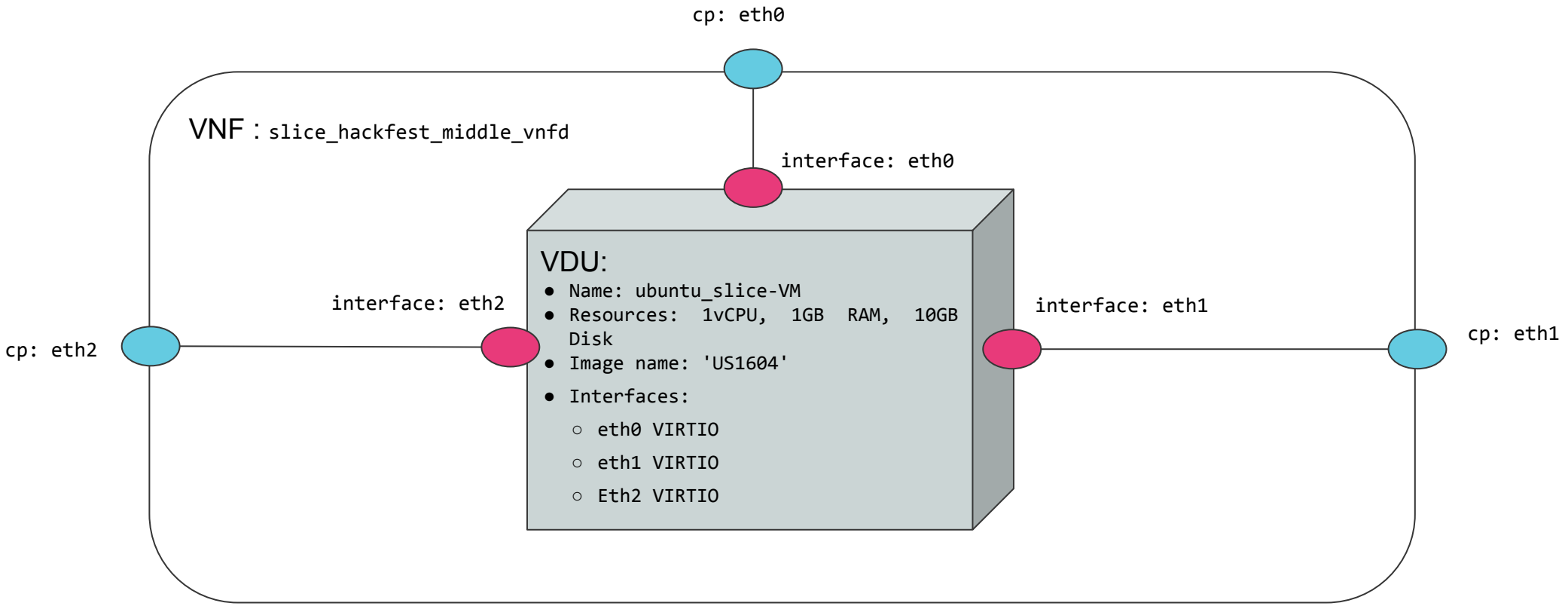
© ETSI 2020

VNFD - slice_hackfest_vnfd



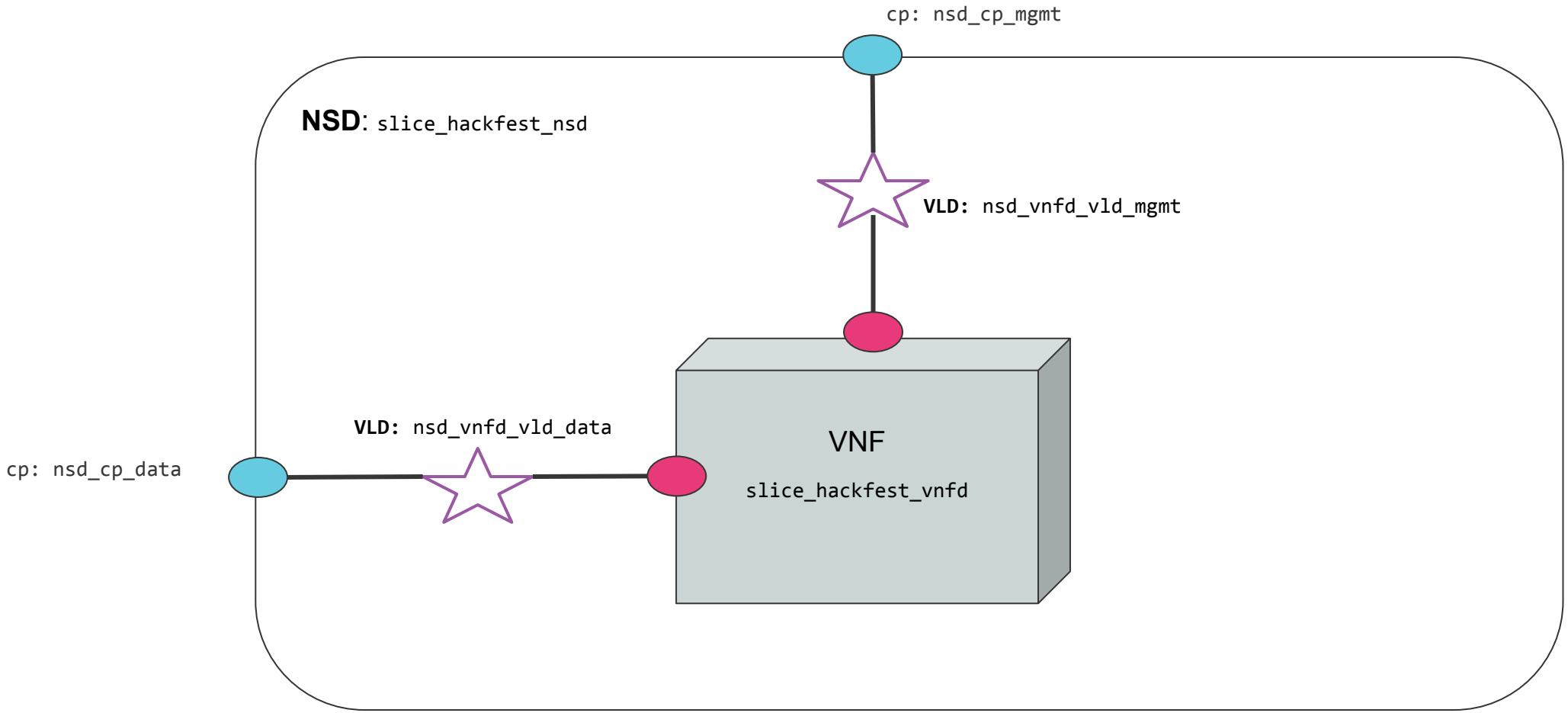
© ETSI 2020

VNFD - slice_hackfest_middle_vnfd



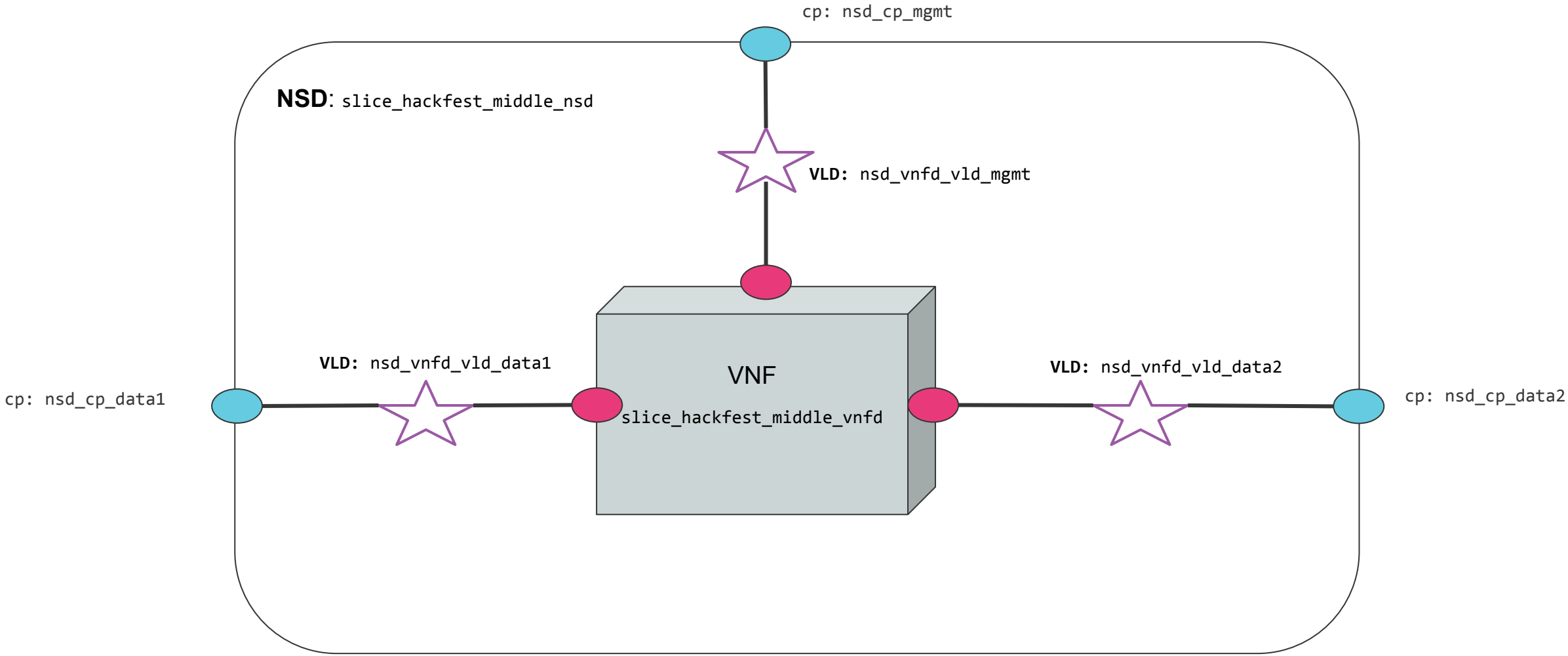
© ETSI 2020

NSD - slice_hackfest_nsd



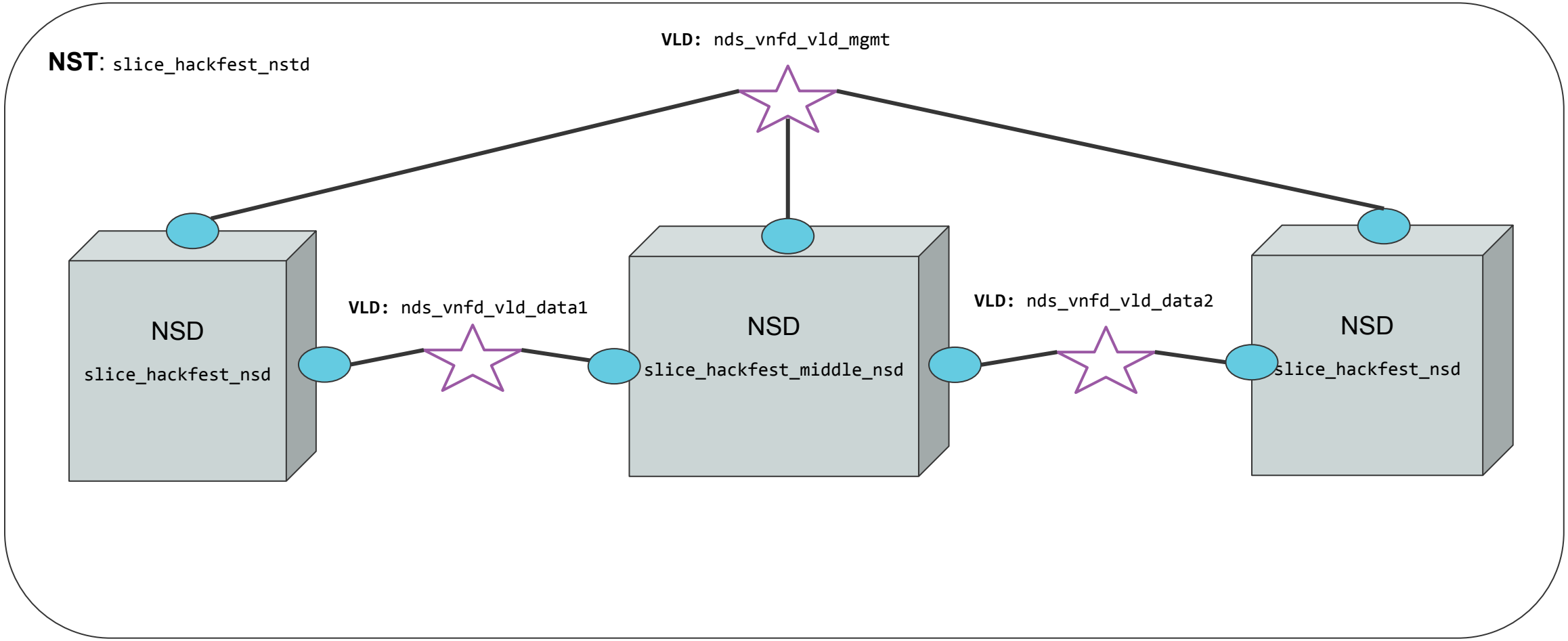
© ETSI 2020

NSD - slice_hackfest_middle_nsd



© ETSI 2020

NST - slice_hackfest_nst



© ETSI 2020

Network Slice Template

```
nst:
-   id: slice_hackfest_nstd
    name: slice_hackfest_nstd
    SNSSAI-identifier:
        slice-service-type: eMBB
    quality-of-service:
        id: 1

    netslice-subnet:
    -   id: slice_hackfest_nsd_1
        is-shared-nss: false
        description: NetSlice Subnet (service) composed by 1 vnf with 2 cp
        nsd-ref: slice_hackfest_nsd
    -   id: slice_hackfest_nsd_2
        is-shared-nss: true
        description: NetSlice Subnet (service) composed by 1 vnf with 3 cp
        nsd-ref: slice_hackfest_middle_nsd
    -   id: slice_hackfest_nsd_3
        is-shared-nss: false
        description: NetSlice Subnet (service) composed by 1 vnf with 2 cp
        nsd-ref: slice_hackfest_nsd

    netslice-vld:
    -   id: slice_vld_mgmt
        name: slice_vld_mgmt
        type: ELAN
        mgmt-network: true
        nss-connection-point-ref:
        -   nss-ref: slice_hackfest_nsd_1
            nsd-connection-point-ref: nsd_cp_mgmt
        -   nss-ref: slice_hackfest_nsd_2
            nsd-connection-point-ref: nsd_cp_mgmt
        -   nss-ref: slice_hackfest_nsd_3
            nsd-connection-point-ref: nsd_cp_mgmt
-   id: slice_vld_data1
     name: slice_vld_data1
     type: ELAN
     nss-connection-point-ref:
     -   nss-ref: slice_hackfest_nsd_1
         nsd-connection-point-ref: nsd_cp_data
     -   nss-ref: slice_hackfest_nsd_2
         nsd-connection-point-ref: nsd_cp_data1
 -   id: slice_vld_data2
     name: slice_vld_data2
     type: ELAN
     nss-connection-point-ref:
     -   nss-ref: slice_hackfest_nsd_2
         nsd-connection-point-ref: nsd_cp_data2
     -   nss-ref: slice_hackfest_nsd_3
         nsd-connection-point-ref: nsd_cp_data
```

© ETSI 2020

https://osm-download.etsi.org/ftp/osm-6.0-six/8th-hackfest/packages/slice_hackfest_nst.yaml

Creating a Network Slice Template (NST)

Open Source MANO

Information Model

```
module: nst
  +--rw nst* [id]
     +--rw id                              string
     +--rw name                            string
     +--rw SNSSAI-identifier
     |  +--rw slice-service-type           network-slice-type
     |  +--rw slice-differentiator?        string
     +--rw quality-of-service
     |  +--rw id                           uint16
     |  +--rw resource-type?               resource-type
     |  +--rw priority-level?              uint16
     |  +--rw packet-delay-budget?         uint16
     |  +--rw packet-error-rate?           uint16
     |  +--rw default-max-data-burst?      uint16
```



NST - id, name, and slice parameters section

```
nst:
-   id: slice_hackfest_nst
    name: slice_hackfest_nst
    SNSSAI-identifier:
        slice-service-type: eMBB
    quality-of-service:
        id: 1
```

- The Information model for the NST is available:
  http://osm-download.etsi.org/repository/osm/debian/ReleaseSIX/docs/osm-im/osm_im_trees/nst.html

© ETSI 2020

Creating a Network Slice Template (NST)

Information Model

```
+--rw netslice-subnet*                [id]
 |  +--rw id                          string
 |  +--rw description?                string
 |  +--rw is-shared-nss?              boolean
 |  +--rw nsd-ref                     -> /nsd:nsd-catalog/nsd/id
 |  +--rw instantiation-parameters
    | +--.....
```



NST - netslice-subnet section

```
netslice-subnet:
    -   id: slice_hackfest_nsd_1
        is-shared-nss: false
        description: NetSlice Subnet (service) composed by 1 vnf with 2 cp
        nsd-ref: slice_hackfest_nsd

    -   id: slice_hackfest_nsd_2
        is-shared-nss: true
        description: NetSlice Subnet (service) composed by 1 vnf with 3 cp
        nsd-ref: slice_hackfest_middle_nsd

    -   id: slice_hackfest_nsd_3
        is-shared-nss: false
        description: NetSlice Subnet (service) composed by 1 vnf with 2 cp
        nsd-ref: slice_hackfest_nsd
```

- The Information model for the NST is available:
  http://osm-download.etsi.org/repository/osm/debian/ReleaseSIX/docs/osm-im/osm_im_trees/nst.html

© ETSI 2020

Creating a Network Slice Template (NST)

Open Source MANO

NST - netslice-subnet section

```
netslice-subnet:
  -  id: slice_hackfest_nsd_1
     is-shared-nss: false
     description: NetSlice Subnet (service) composed by 1 vnf with 2 cp
     nsd-ref: slice_hackfest_nsd

  -  id: slice_hackfest_nsd_2
     is-shared-nss: true
     description: NetSlice Subnet (service) composed by 1 vnf with 3 cp
     nsd-ref: slice_hackfest_middle_nsd

  -  id: slice_hackfest_nsd_3
     is-shared-nss: false
     description: NetSlice Subnet (service) composed by 1 vnf with 2 cp
     nsd-ref: slice_hackfest_nsd
```

NSD - id, name, and NS parameters section

```
nsd-catalog:
  nsd:
    -  id: slice_hackfest_nsd
       name: slice_hackfest_nsd
       short-name: slice_hackfest_ns
       description: NetServiceDescriptor with 2 vnfs and 2 vld (mgmt and
data networks)
       vendor: OSM
       version: '1.0'
       logo: osm_2x.png
```

- The Information model for the NST is available:
  http://osm-download.etsi.org/repository/osm/debian/ReleaseSIX/docs/osm-im/osm_im_trees/nst.html

© ETSI 2020

Creating a Network Slice Template (NST)

Open Source MANO

Information Model

```
+--rw netslice-vld* [id]
|  +--rw id                          string
|  +--rw name?                       string
|  +--rw short-name?                 string
|  +--rw vendor?                     string
|  +--rw description?                string
|  +--rw version?                    string
|  +--rw type?                       manotypes:virtual-link-type
|  +--rw root-bandwidth?             uint64
|  +--rw leaf-bandwidth?             uint64
|  +--rw provider-network
|  |  +--rw physical-network?        string
|  |  +--rw segmentation_id?         uint32
|  +--rw mgmt-network?               boolean
|  +--rw nss-connection-point-ref* [nss-ref nsd-connection-point-ref]
|     +--rw nss-ref                  -> /nst/netslice-subnet/id
|     +--rw nsd-connection-point-ref -> /nsd:nsd-catalog/
                                        nsd/connection-point/name
|     +--rw ip-address?              inet:ip-address
```



NST - netslice-vld section

```
netslice-vld:
   -   id: slice_vld_mgmt
       name: slice_vld_mgmt
       type: ELAN
       mgmt-network: true
       nss-connection-point-ref:
       -   nss-ref: slice_hackfest_nsd_1
           nsd-connection-point-ref: nsd_cp_mgmt
       -   nss-ref: slice_hackfest_nsd_2
           nsd-connection-point-ref: nsd_cp_mgmt
       -   nss-ref: slice_hackfest_nsd_3
           nsd-connection-point-ref: nsd_cp_mgmt
   -   id: slice_vld_data1
       name: slice_vld_data1
       type: ELAN
       nss-connection-point-ref:
       -   nss-ref: slice_hackfest_nsd_1
           nsd-connection-point-ref: nsd_cp_data
       -   nss-ref: slice_hackfest_nsd_2
           nsd-connection-point-ref: nsd_cp_data1
```

- The Information model for the NST is available:
  http://osm-download.etsi.org/repository/osm/debian/ReleaseSIX/docs/osm-im/osm_im_trees/nst.html

© ETSI 2020

Creating a Network Slice Template (NST)

Open Source MANO

NST - netslice-subnet section

```
netslice-subnet:
  - id: slice_hackfest_nsd_1
    nsd-ref: slice_hackfest_nsd
```

NSD - id, name, and NS parameters section

```
nsd-catalog:
   nsd:
-    id: slice_hackfest_nsd
     name: slice_hackfest_nsd
     short-name: slice_hackfest_ns
```

NST - netslice-vld

```
netslice-vld:
-    id: slice_hackfest_vld_mgmt
     name: slice_hackfest_vld_mgmt
     type: ELAN
     mgmt-network: true
     nss-connection-point-ref:
-    nss-ref: slice_hackfest_nsd_1
     nsd-connection-point-ref: nsd_cp_mgmt
```

NSD - connection-point section

```
connection-point:
-    name: nsd_cp_mgmt
     vld-id-ref: nsd_vnfd_vld_mgmt
```

- The Information model for the NST is available:
  http://osm-download.etsi.org/repository/osm/debian/ReleaseSIX/docs/osm-im/osm_im_trees/nst.html

© ETSI 2020

Before the netslice instantiation, Adding VNF, NS packages and NST

- VNF package:
  - `osm vnfd-list`
  - `osm vnfd-create slice_hackfest_vnf.tar.gz`
  - `osm vnfd-create slice_hackfest_middle_vnf.tar.gz`
  - `osm vnfd-show slice_hackfest_vnfd`
  - `osm vnfd-show slice_hackfest_middle_vnfd`
- NS package:
  - `osm nsd-list`
  - `osm nsd-create slice_hackfest_ns.tar.gz`
  - `osm nsd-create slice_hackfest_middle_ns.tar.gz`
  - `osm nsd-show slice_hackfest_nsd`
  - `osm nsd-show slice_hackfest_middle_nsd`
- NST:
  - `osm nst-list`
  - `osm nst-create slice_hackfest_nst.yaml`
  - `osm nst-create slice_hackfest2_nst.yaml`
  - `osm nst-show slice_hackfest_nst`

© ETSI 2020

Creating a Network Slice Instance (CLI)

- `osm nsi-create --help`

```
Usage: osm nsi-create [OPTIONS]
  creates a new Network Slice Instance (NSI)

Options:
  --nsi_name TEXT      name of the Network Slice Instance
  --nst_name TEXT      name of the Network Slice Template
  --vim_account TEXT   default VIM account id or name for the deployment
  --ssh_keys TEXT      comma separated list of keys to inject to vnfs
  --config TEXT        Netslice specific yaml configuration:
                       netslice_subnet: [
                       id: TEXT, vim_account: TEXT,
                       vnf: [member-vnf-index:
                       TEXT, vim_account: TEXT]
                       vld: [name: TEXT, vim-network-
                       name: TEXT or DICT with vim_account, vim_net entries]],
                       netslice-vld: [name: TEXT, vim-network-name: TEXT or
                       DICT with vim_account, vim_net entries]
  --config_file TEXT   nsi specific yaml configuration file
  --help               Show this message and exit.
```

- `osm nsi-create --nsi_name my_first_slice --nst_name slice_hackfest_nst --vim_account <replace_vim_account_name> --config 'netslice-vld: [{ "name": "slice_vld_mgmt", "vim-network-name": <replace_vim_external_network> }]'`

© ETSI 2020

Listing and Deleting a Network Slice Instance (CLI)

- List Network Slice Instances
  - `osm nsi-list`
- Delete Network Slice Instance
  - `osm nsi-delete <nsi_name> or <nsi_id>`

© ETSI 2020

Managing Network Slice Templates (via CLI)



- Creates a new Network Slice Template
  - `netslice-template-create / nst-create`
- Deletes a Network Slice Template
  - `netslice-template-delete / nst-delete`
- List all Network Slice Templates
  - `netslice-template-list / nst-list`
- Shows the content of a Network Slice Template
  - `netslice-template-show / nst-show`
- Updates a Network Slice Template
  - `netslice-template-update / nst-update`

© ETSI 2020

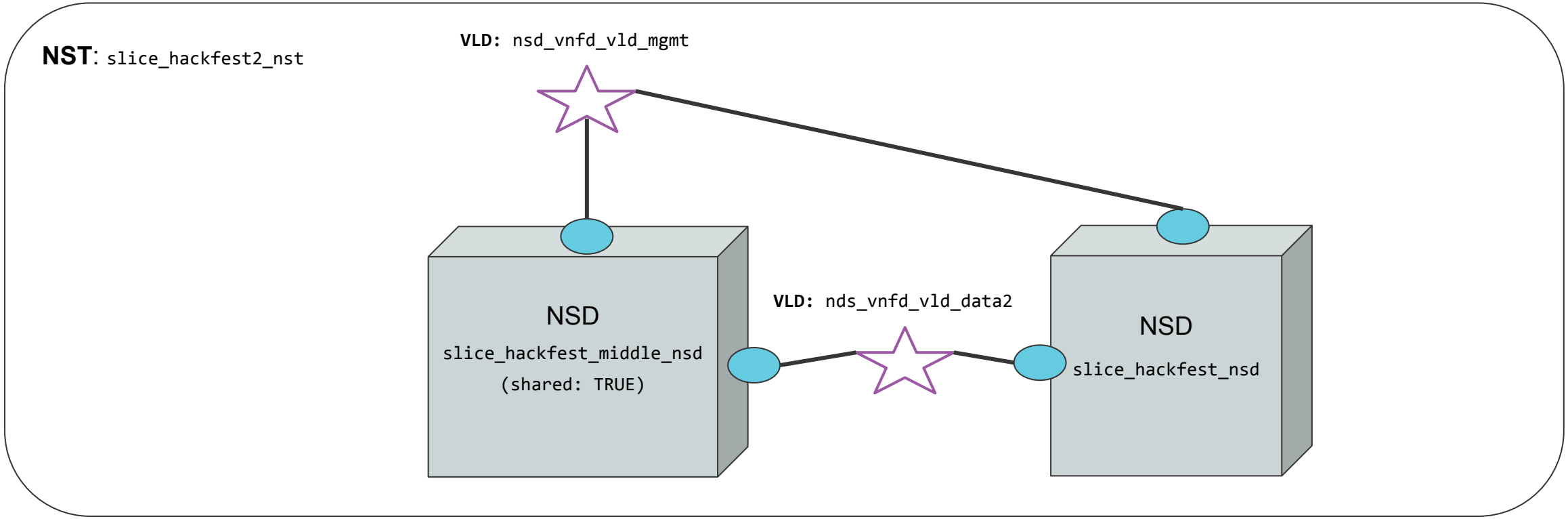
Managing Network Slice Instances (via CLI)

- Creates a new Network Slice Instance
  - `netslice-instance-create / nsi-create`
- Deletes a Network Slice Instance
  - `netslice-instance-delete  / nsi-delete`
- List all Network Slice Instances (NSI)
  - `netslice-instance-list / nsi-list`
- Shows the history of operations over a
  - `netslice-instance-op-list / nsi-op-list`
- Shows the info of an operation over a Network Slice Instance(NSI)
  - `netslice-instance-op-show / nsi-op-show`
- Shows the content of a Network Slice Instance (NSI)
  - `netslice-instance-show / nsi-show`

© ETSI 2020

Shared Network Slices

- Let’s share the middle NS using a new NST



© ETSI 2020

Network Slice Template 2

```
nst:
-   id: slice_hackfest2_nst
    name: slice_hackfest2_nst
    SNSSAI-identifier:
        slice-service-type: eMBB
    quality-of-service:
        id: 1

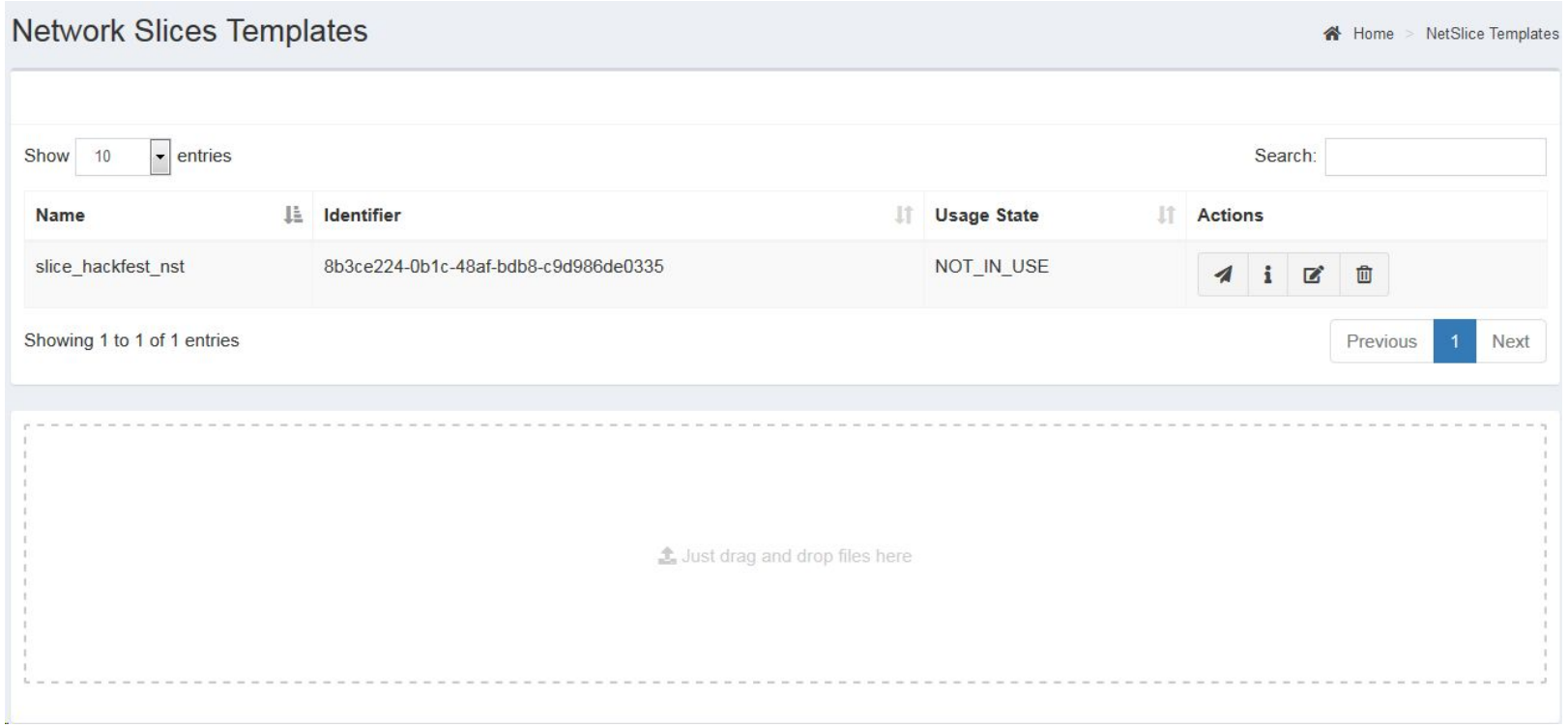
    netslice-subnet:
    -   id: slice_hackfest_nsd_2
        is-shared-nss: true
        description: NetSlice Subnet (service) composed by 1 vnf with 3 cp
        nsd-ref: slice_hackfest_middle_nsd
    -   id: slice_hackfest_nsd_3
        is-shared-nss: false
        description: NetSlice Subnet (service) composed by 1 vnf with 2 cp
        nsd-ref: slice_hackfest_nsd

    netslice-vld:
    -   id: slice_vld_mgmt
        name: slice_vld_mgmt
        type: ELAN
        mgmt-network: true
        nss-connection-point-ref:
        -   nss-ref: slice_hackfest_nsd_2
            nsd-connection-point-ref: nsd_cp_mgmt
        -   nss-ref: slice_hackfest_nsd_3
            nsd-connection-point-ref: nsd_cp_mgmt
-   id: slice_vld_data2
    name: slice_vld_data2
    type: ELAN
    nss-connection-point-ref:
    -   nss-ref: slice_hackfest_nsd_2
        nsd-connection-point-ref: nsd_cp_data2
    -   nss-ref: slice_hackfest_nsd_3
        nsd-connection-point-ref: nsd_cp_data
```

© ETSI 2020
https://osm-download.etsi.org/ftp/osm-6.0-six/8th-hackfest/packages/slice_hackfest2_nst.yaml

Creating a Network Slice Instance (UI)

- List Network Slice Templates
  - In the left panel, click on the Netslice Templates icon 



© ETSI 2020

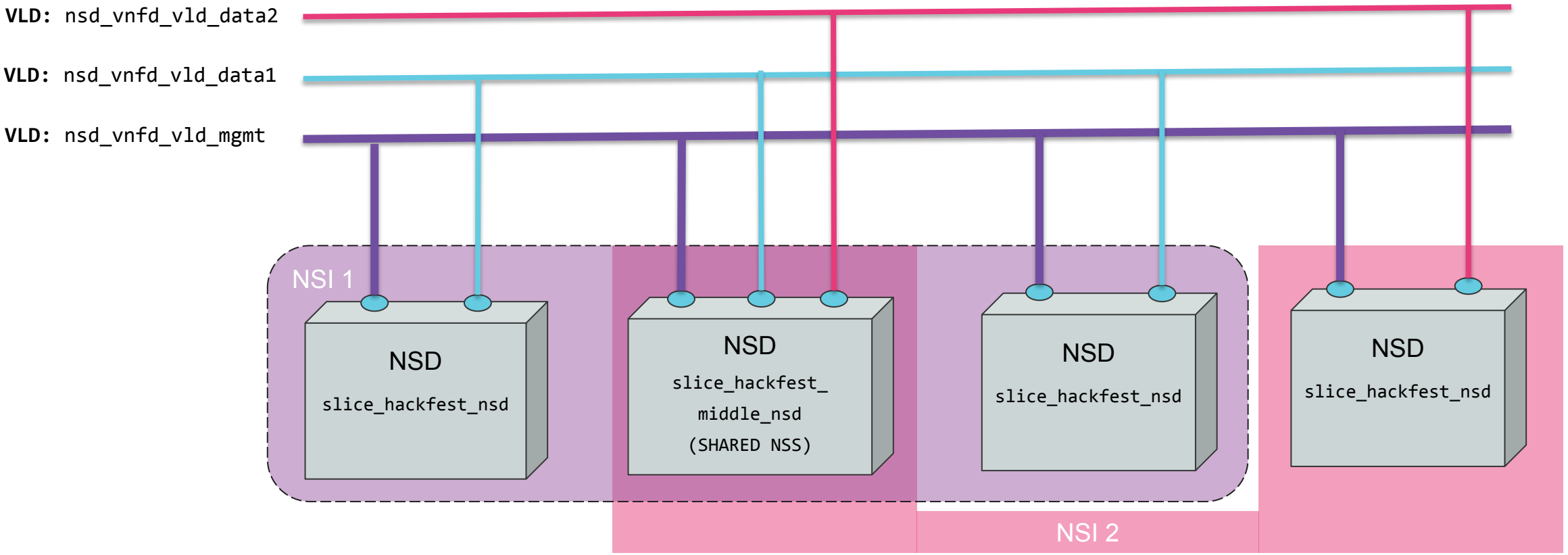
Creating a Network Slice Instance (UI)



© ETSI 2020

Shared Network Slices

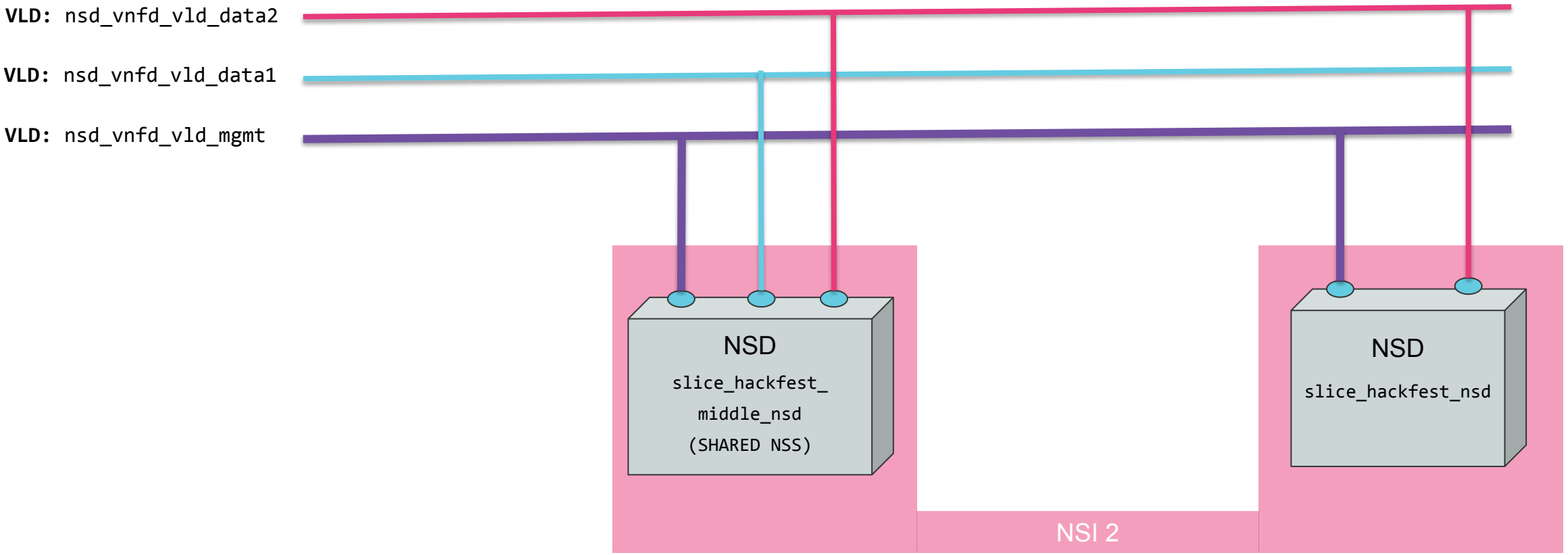
- Final deployment



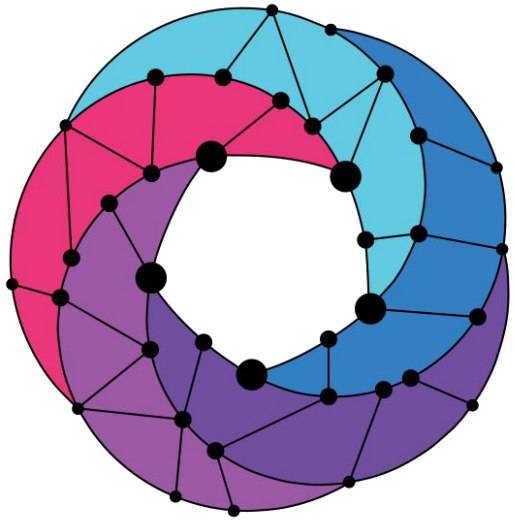
© ETSI 2020

Shared Network Slices

- Result of deleting NSI 1



© ETSI 2020



Open Source MANO

Find us at:

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

© ETSI 2020