OSM Release EIGHT Release Notes

Open Source MANO
Technical Steering Committee
July 2020

ETSI
06921 Sophia Antipolis CEDEX, France
Tel +33 4 92 94 42 00
info@etsi.org
www.etsi.org

Authors

Name	Organization
Alfonso Tierno	Telefónica
Andy Reid	BT
Antonio Marsico	BT
Barath Kumar R.	Tata ELXSI
David García	Canonical
Felipe Vicens	Atos
Francisco-Javier Ramón	Telefónica
Francisco Rodriguez	Indra
Gerardo García de Blas	Telefónica
Gianpietro Lavado	Whitestack
José Miguel Guzmán	Whitestack
Lars-Göran Magnusson	Arctos Labs
Mark Beierl	Canonical
Mark Shuttleworth	Canonical
Pål Grønsund	Telenor
Preethika P	Tata ELXSI
Ramesh Ramanathan	Tata ELXSI
Silvia Almagia	ETSI
Subhankar Pal	Altran

Content

Introduction

OSM Release EIGHT is the result of more than 4 years of evolution in the development of the most complete open source NFV orchestrator in the market.

In Release SEVEN, OSM started to orchestrate CNFs (Containerized Network Functions), by leveraging existing or new kubernetes clusters. This was a major milestone in an industry that considers Kubernetes as the underlying infrastructure for next generation 5G services.

Release EIGHT brings along a number of new features allowing to orchestrate a broader range of network functions and production environments.

This is part of the greater evolution the industry is seeing, which is converging to a clearer structure enabling an orchestrated combination of virtual, physical and containerized network functions..

This new release would not have been possible without the contributions of the energetic community behind OSM. A community integrated by operators, vendors and academia, that leverage their expertise in each one of these angles. Thanks to all the organizations and individuals that made this release posible!

OSM and Kubernetes

Since Release SEVEN, OSM has evolved to support Kubernetes in two dimensions

- **As the underlying technology to run OSM**: Upstream OSM runs in Docker containers (images available in Docker Hub), which can be deployed in a Kubernetes cluster.
- **As the underlying technology to run KNFs**: OSM is able to deploy Kubernetes-based Network Functions (KNFs) or (equivalently) Containerized Network Functions (CNFs) in a Kubernetes cluster, and orchestrate the KNFs life-cycle.

Using Kubernetes-based VNFs

OSM support of Kubernetes-based Network Functions (KNF) has unlocked over 20.000 Kubernetes Apps that can now be deployed with OSM in addition to VNFs and PNFs.

Deploying a KNF with OSM requires an operational Kubernetes cluster. From an OSM perspective, the Kubernetes cluster is not an isolated element, but a technology that enables the deployment of microservices in a cloud-native way. To handle the networks and facilitate the connection to the infrastructure, the cluster needs to be associated with a VIM.

OSM can deploy KNFs on any Kubernetes cluster, as long as it includes:

1. a **Kubernetes load balancer**, to expose KNFs to the network
2. a **Kubernetes default storage class**, to support persistent volumes.

The Kubernetes cluster can be a self-managed cluster running in a VIM or directly in bare metal servers. In order to facilitate the Kubernetes cluster deployment (out of the scope of OSM), OSM documentation provides different procedures, including the capability to deploy a Kubernetes cluster using VNF and NS packages.

Kubernetes support allows the deployment of more advanced network services combining VNFs, PNFs and KNFs. A good example of this, is Magma (an open source vEPC) that in some commercial distributions provides descriptors for its orchestration components as KNFs, and the access components as VNFs. For more information, please visit: Using Kubernetes-based VNFs

New Features in Release EIGHT

Release EIGHT brings a set of new features allowing to improve the orchestration of diverse virtualization environments, including PNFs, a number of different VIMs for VNFs, and Kubernetes for KNFs. Among these new features we find:

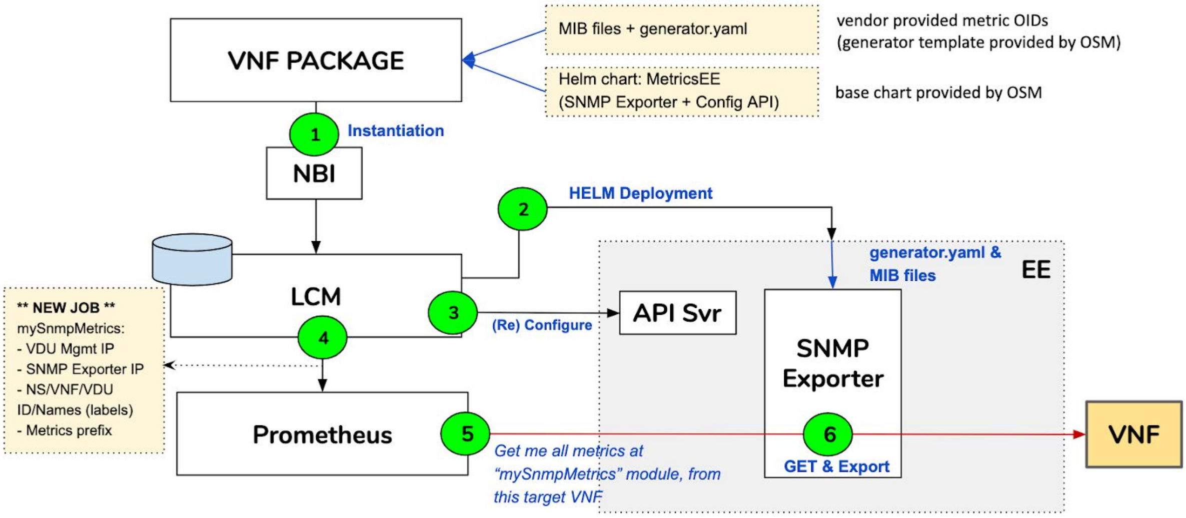
- The new VNF monitoring functionality, based on Prometheus exporters provides the flexibility of using traditional SNMP as well as the ability to adopt newer technologies.
- An improved architecture, allowing to communicate with a Highly Available VCA cluster, which makes the whole system more robust against unexpected failures.
- Proxy charms on Kubernetes allow now Juju to manage the lifecycle of KNFs in a similar way to VNFs, and improve the performance of VNF deployments.
- A brand new NG-UI offers an evolved user interface, based on the well-adopted Angular framework, that brings a fresh look to Release EIGHT.
- The new Placement Optimization Module (PLA) adds for the first time in OSM, an optimization engine, able to decide where to deploy the xNFs (VNFs or KNFs), based on technical criteria. This is an important aspect that the project wants to evolve in subsequent releases.
- Manageability has also been improved, by implementing per tenant quotas in OSM (in addition to the quotas already provided by the VIM), and better integration with OSS/BSS through a subscription mechanism.
- Supporting a wider range of SDN Controllers is also important, as the options for deploying a data center fabric have significantly increased in recent years, and OSM remains able to leverage the different options using the SDN Assist capability.
- VNF Repositories facilitate the distribution of VNFs by using a simple mechanism that allows users to find and download VNFs from vendors' repositories conforming to the OSM guidelines.

The following sections describe in further detail those major new features coming with Release EIGHT.

Advanced VNF Monitoring

VNF-level monitoring has always been a challenge, due to the diversity of protocols and APIs that VNFs expose to present their metrics and indicators, either directly or through Element Managers. However, having the capability of collecting VNF KPIs at the MANO layer is key for achieving self-healing and event correlation, which ultimately impact the Network Services availability.

OSM Release EIGHT introduces the collection of VNF indicators via Prometheus exporters, in a way that allows different “exporter components” that perform the collection to be plugged-in dynamically into the system on a per-VNF basis and feed OSM’s Prometheus time-series database with real-time performance information.



The first use case being covered introduces the possibility of modelling an SNMP Exporter as part of the VNF Package through a Helm Chart, so it can be instantiated by OSM and have metrics predefined by the VNF package builder automatically collected. A key characteristic of the implementation is that these collectors are instantiated in new, dockerized “Execution Environments”, which are modeled through the aforementioned Helm Chart, implementing at the same time a lightweight API server that is able to configure (or reconfigure) the rest of the elements included in the environment in the form of OSM Day-1 and Day-2 primitives. For example, in this first use case, a Day-1 primitive initializes the

SNMP Exporter configuration so that it has all the information required to collect the desired metrics from the VNF.

In the near future and without requiring any change in OSM's code, other use cases will be documented, such as new metrics collectors (i.e. Streaming Telemetry), or even simple VNF configurations made through the API Server component described above.

High Availability for VCA

OSM allows orchestration of multiple network services across different substrates, from deployment to Day-1 and Day-2 operations.

The core component to handle daily operations in network functions has been the VNF Configuration and Abstraction (VCA) since OSM Release ONE. The VCA (Juju) is in charge of deploying, configuring, and relating charms (the operations package of the VNFs), providing the tools to integrate unique pieces of software, and creating a smooth harmony among them.

OSM is a production-ready MANO, which means having resiliency and redundancy across the prime components is fundamental. In Release EIGHT, OSM has improved its architecture, and can now communicate with a Highly Available VCA cluster, which makes the whole system more robust against unexpected failures.

High Availability for Proxy Charms

Charms are code that enable Day-1 and Day-2 primitives in the workloads deployed with OSM. Charms implement the Kubernetes 'operator pattern' but work on both Kubernetes and traditional machine-oriented workloads. A charm is where operations code is defined and it is part of the VNF package. The VCA is responsible for deploying and managing the lifecycle of the charms.

When the charm code cannot run in the same container as the workload, the charm is deployed on a LXD or Kubernetes cluster known by OSM. These particular charms are called proxy charms. With this feature, OSM can deploy more units of a proxy charm by only giving additional parameters at instantiation time. Additionally, an external LXD cluster with multiple nodes can be provided to OSM, and the VCA will place each unit on a different node.

Highly available proxy charms protect the deployments from failures in the charms. In case of an unexpected downtime in one of the units, another one will take the lead, and the operations in the workload will not be impacted. Resiliency and redundancy are the key benefits of this feature.

Kubernetes Proxy Charms

Prior to Release EIGHT, proxy charms that were deployed on a LXD cloud needed to create a container, initialize it and install some software packages before they could run Day-1 and Day-2 configuration. This usually took a couple of minutes, and could delay the deployment of network services.

OSM Release EIGHT allows operating a VNF through a proxy charm deployed in Kubernetes instead of LXD. Charms on LXD and Kubernetes use the same mechanisms for lifecycle and event handling, so LXD machine charms can be ported to Kubernetes. It also provides VCA with the ability to handle two different clouds at the same time, LXD and Kubernetes.

With Kubernetes proxy charms, the time of initialization is reduced drastically to seconds thanks to the containerized nature of Kubernetes. This makes the deployment of the network service much faster, and therefore the user experience is significantly improved.

In addition, OSM takes advantage of the existing benefits of working with Kubernetes, like self-healing. This means that if our charm is not in a healthy state, Kubernetes will spin up a new pod automatically.

Next Generation User Interface

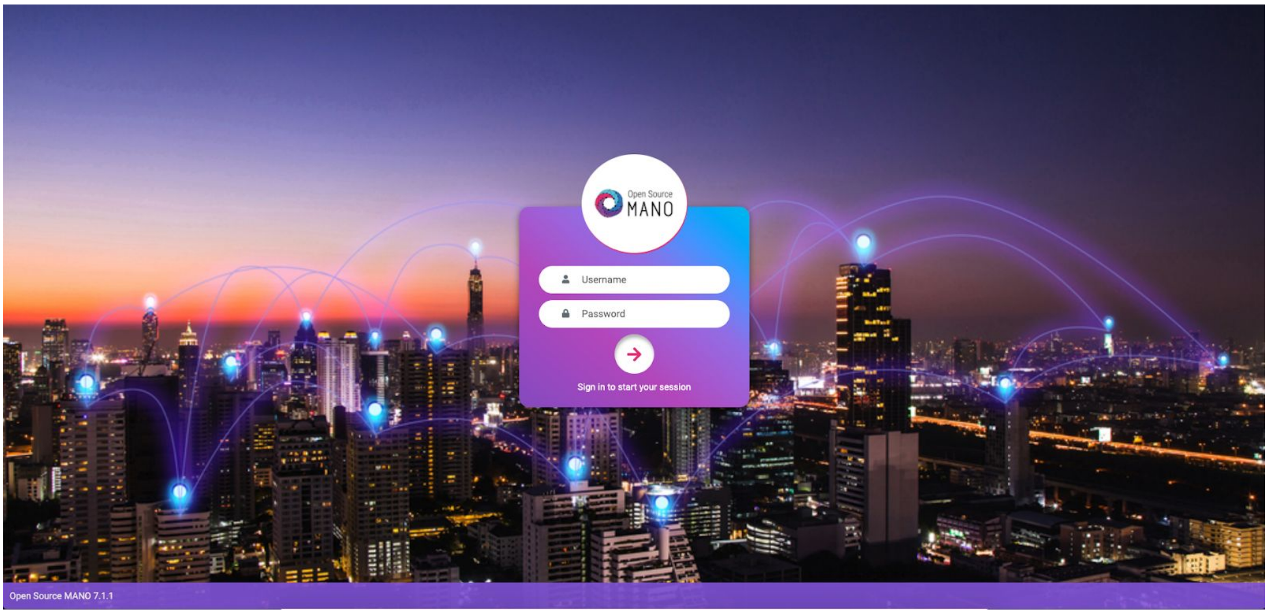
NG-UI is another exciting new addition to Open Source MANO Release EIGHT. Built on Angular, NG-UI brings in the modern notion of Single Page Application in a clean and elegant manner. Being modular, component-driven and based on the MVC design pattern, NG-UI is easy to maintain, extend, re-skin and debug. The portal is fully responsive supporting multiple browsers like IE10/11, Edge, latest Chrome and Firefox. Built ground up to be 5G aware, NG-UI adheres to industry standards in security and code quality.

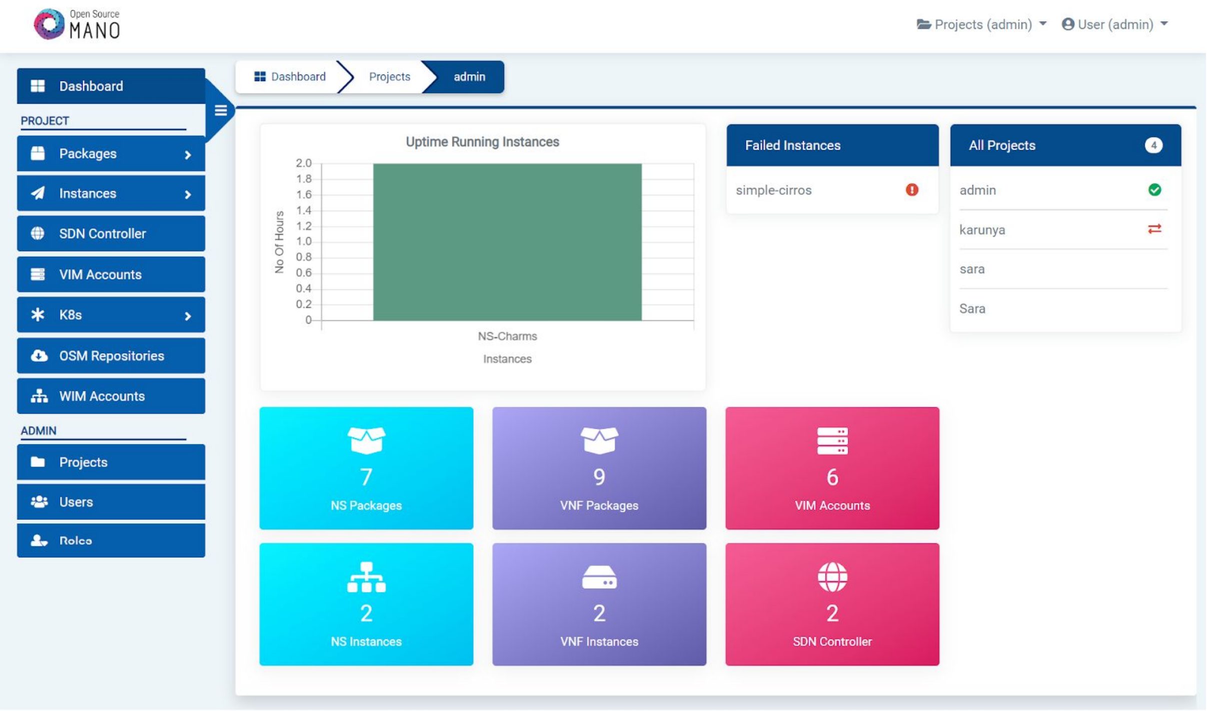
Main features:

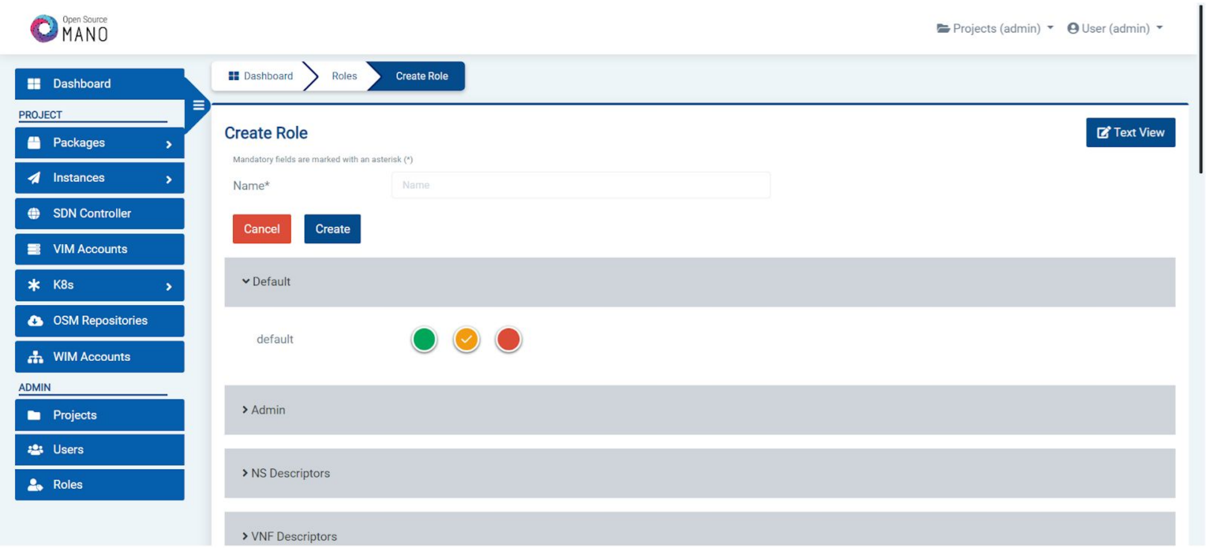
- Intuitive user experience for building 5G networks

- Data centre map view to inspect all running & failed NS instances in a single click
- Granular RBAC design and control with preview mode
- Day 2 configuration support for multi-level primitives, NS, VNF and VDU primitives
- Multilingual support - English, German, Spanish, Portuguese
- Enhanced VIM creation with new configuration support
- Quota limit functionality that limits the number of packages, instances, VIM and projects
- Provision to access the NS metrics

The following are screenshots from the NG-UI:




Open Source MANO
Projects (admin)
User (admin)
Dashboard
Projects
admin
PROJECT
Packages
Instances
SDN Controller
VIM Accounts
K8s
OSM Repositories
WIM Accounts
ADMIN
Projects
Users
Roles
Uptime Running Instances
No Of Hours
2.0
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0
NS-Charms
Instances
Failed Instances
simple-cirros
All Projects
4
admin
karunya
sara
Sara
7
NS Packages
9
VNF Packages
6
VIM Accounts
2
NS Instances
2
VNF Instances
2
SDN Controller


Open Source MANO
Projects (admin)
User (admin)
Dashboard
Roles
Create Role
PROJECT
Packages
Instances
SDN Controller
VIM Accounts
K8s
OSM Repositories
WIM Accounts
ADMIN
Projects
Users
Roles
Create Role
Text View
Mandatory fields are marked with an asterisk (*)
Name*
Name
Cancel
Create
Default
default
Admin
NS Descriptors
VNF Descriptors

Placement Optimization Module

The Placement Optimization module (PLA) will help the user to automatically find the most cost efficient distribution of the xNFs in an NSD over the set of available VIMs when a Network Service is instantiated.

The use of automatic placement optimization is optional and invoked by the user at Network Service instantiation time.

The user may express constraints on the VLDs for maximum allowed latency and jitter. The user may also pin selected xNFs to specific VIMs while leaving the decision for optimal placement of other xNFs to the placement optimization module.

Computation of optimal placement of xNFs is based on user-provided models of

- compute and networking costs
- latency and jitter metrics for inter-VIM connectivity.

Optimal placement of xNFs over the VIMs is done by matching the NS-specific requirements to infrastructure availability and metrics, while considering cost of compute and networking.

The placement optimization module is an optional module in OSM.

Quotas Management

This functionality enhances security and provides a more granular control of OSM usage. It allows setting limits for the infrastructure (number of VIM, WIM, SDN controllers, Kubernetes clusters, PDUs, repos), packages (nfpkg, nspkg, netslice templates) and deployments (ns, netslice instances). Once the limit is reached, any attempt to create a new item will be rejected to prevent overloading the system.

Quotas are scoped by projects. The project information contains the assigned quotas, that can be changed by updating the project. In case of no information a default large value is used.

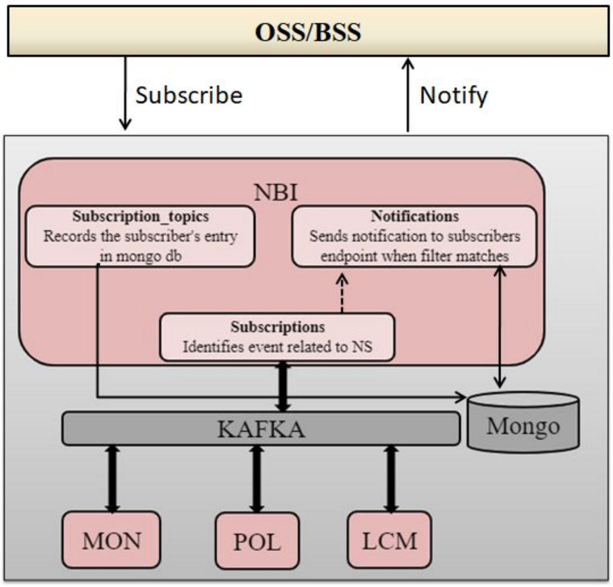
It is important to mention that this quota management functionality is at the OSM level, and therefore it is independent from and unrelated to the VIM quotas.

Subscription API for BSS/OSS

OSS/BSS systems communicate with OSM via the ETSI NFV SOL005 compliant Northbound APIs. To ensure real time monitoring, assurance and predictive maintenance, it is critical for the OSS/BSS systems to be aware of the run-time status of each and every deployed service in the network. OSM Release EIGHT brings in a comprehensive mechanism of subscriptions and notifications to give a single glass pane view to the OSS and to the new age Digital NOCs. The notifications cover the entire range of network service life cycle management operations.

By subscribing to the dynamic changes in the network, operators can now take the necessary steps to assure customers service continuity in real time and in a predictive manner via the Digital NOC or OSS processes. The subscription mechanism is completely configurable from the end-user perspective, in terms of the choice of services to follow and the choice of lifecycle events. During the subscription process, the OSS/BSS or new age NOC registers a callback URI on which it wants to receive the notifications along with the required filter options. These options are stored in MongoDB to persist and can be modified and audited at a later point in time.

When an event is raised in Kafka that matches any of the subscribers' filter criteria, an appropriate notification is generated to the registered URI as shown in the diagram below.


OSS/BSS
Subscribe
Notify
NBI
Subscription_topics
Records the subscriber's entry in mongo db
Notifications
Sends notification to subscribers endpoint when filter matches
Subscriptions
Identifies event related to NS
KAFKA
Mongo
MON
POL
LCM

Support for new SDN Controllers

Continuing the evolution towards a fully integrated Telco Cloud switching fabric, OSM is adding support for more SDN Controllers available in the market.

The SDN Assist functionality allows OSM to take control of the underlying switching fabric in order to provide connectivity between VNFs, for scenarios where this connectivity is out of the scope of the VIM. A typical scenario is when SR-IOV or PCI passthrough is used for optimizing the dataplane, which prevents Openstack from leveraging the OVS infrastructure for creating tenant networks.

With SDN Assist, OSM will instantiate the Network Service through the VIM, and in parallel will deploy the networks (for connecting integrating VNFs) through the corresponding SDN controller, in what is called a "*decoupled switching fabric model*".

In Release EIGHT the support for two new SDN Controllers has been added:

- Arista CloudVision
- Juniper Contrail

VNF Repositories

OSM VNF Repositories is a powerful mechanism that allows organizations (research institutions, VNF vendors, etc..) to expose their VNFs in a consistent way, so that OSM operators can retrieve them from remote locations more easily, by using OSM tools. This feature enables the vision of serving VNFs in distributed “application stores”.

Repositories can be local (exposed in the local filesystem) or remote (exposed by HTTP), and just need to be served with a predefined file structure in order for any OSM system to be able to interact with the VNF it contains.

This structure can be easily created with a command like:

```
osm repo-index --origin osm-packages --destination vnf-catalog
```

The corresponding OSM VNF Repository would look like this example:

```
http://nfv-vendor.repo/vnf-catalog/vnf/
└── index.yaml
    ├── vnf
    │   └── cgnat
    │       ├── 1.2.0
    │       │   ├── metadata.yaml
    │       │   └── cgnat-1.2.0.tgz
    │       └── 1.2.1
    │           ├── metadata.yaml
    │           └── cgnat-1.2.2.tgz
    └── images
        ├── cgnat-traffic-0.2.1
        │   ├── metadata.yaml
        │   └── cgnat-traffic-0.2.1.qcow2
        └── cgnat-balancer-0.3.4
            ├── metadata.yaml
            └── cgnat-balancer-0.3.4.qcow2
```

Given a reachable repository, an operator can:

- **Add the repository**
  osm repo-add --description <description> <repo_name> <url>

- **List packages inside the repository**
  osm nfpkg-repo-list

- **Upload packages from a repository to OSM**
  osm nfpkg-create --repo <repo_name> <package_name>

...and other similar tasks that take VNF Package consistency and distribution to a complete new level.

OSM Installation to OpenStack

In order to simplify OSM installation, the OSM Release EIGHT installer provides an additional option to install OSM to a remote OpenStack infrastructure with an Ansible playbook. The playbook takes care of configuring the OpenStack infrastructure before deploying a VM with OSM, through the following steps:

1. Creation of a new VM flavour (4 CPUs, 8 GB RAM, 40 GB disk)
2. Download of Ubuntu 18.04 image and upload it to OpenStack Glance
3. Generation of a new SSH private and public key pair
4. Setup of a new security group to allow external SSH and HTTP access
5. Deployment of a clean Ubuntu 18.04 VM and installation of OSM to it

Conclusions

Release EIGHT defines a new milestone for OSM, with the introduction of a new set of features helping to orchestrate diverse virtualization environments, including a number of different VIMs for VNFs, and Kubernetes for KNFs.

Release EIGHT is the most comprehensive OSM release so far, leveraging contributions from a strong community of 140 members and participants, and the ETSI vision for standardization, which is positioning OSM as the most flexible open-source NFV orchestrator.

The OSM community is an open community, in its fourth year of growth. If you want to know how to join, please visit: HOW TO GET INVOLVED IN OSM

To go further

For more information, please see:

- Open Source MANO's documentation
- Open Source MANO's developer's guide
- Open Source MANO's VNF Onboarding guide
- OSM Ecosystem
- OSM Members and Participants
- Register for the next OSM-MR9 Hackfest