Open Cloud Reference Architecture

Version 1.6, March 2019

Abstract

The OpenNebula Cloud Reference Architecture is a blueprint to guide IT architects, consultants, administrators and field practitioners in the design and deployment of public and private clouds fully based on open-source platforms and technologies. It has been created from the collective information and experiences from hundreds of users and cloud client engagements. Besides main logical components and interrelationships, this reference documents software products, configurations, and requirements of infrastructure platforms recommended for a smooth OpenNebula installation. Three optional functionalities complete the architecture: high availability, cloud bursting for workload outsourcing, and federation of geographically dispersed data centers.

The document describes the reference architecture for Basic (small to medium-scale) and Advanced (medium to large-scale) OpenNebula Clouds and provides recommended software for main architectural components, including support for virtual machines or system containers, and the rationale behind them. Each section also provides information about other open-source infrastructure platforms tested and certified by OpenNebula to work in enterprise environments. To complement these certified components, the OpenNebula add-on catalog can be browsed for other options supported by the community and partners. Moreover, there are other components in the open cloud ecosystem that are not part of the reference architecture, but are nonetheless important to consider at the time of designing a cloud, like for example Configuration Management and Automation Tools for configuring cloud infrastructure and manage large number of devices.

Contents

1. What is OpenNebula?
2. High Level Reference Architecture
3. OpenNebula Front-End
4. Virtualization Nodes
5. Storage
6. Networking
7. Virtual Machines & Guest Support
8. Authentication
9. Provisioning Model
10. Datacenter Federation
11. Cloud Bursting
112. High Availability
13. Conclusions

Glossary

VM	Virtual Machine
DC	Datacenter
NIC	Network Interface Card
NFS	Network File System
COW	Copy on Write
AD	Active Directory
DB	Database
VDC	Virtual Datacenters
HA	High Availability

1. What is OpenNebula?

Enterprise cloud computing is the next step in the evolution of data center (DC) virtualization. **OpenNebula is a simple, feature-rich and flexible solution to build and manage enterprise clouds and virtualized DCs** that combines existing virtualization technologies with advanced features for multi-tenancy, automatic provision and elasticity. OpenNebula follows a bottom-up approach driven by sysadmins, devops and users real needs.

OpenNebula is a single fully open-source product with a healthy and active community that is commercially supported by OpenNebula Systems. OpenNebula releases are produced on a regular basis and delivered as a single package with a smooth migration path. More information on the benefits of running an OpenNebula cloud can be checked on the key features page[1].

2. High Level Reference Architecture

A cloud architecture is defined by three components: storage, networking and virtualization. Figure 1 shows the bird's eye view of an OpenNebula cloud. OpenNebula services run in the Front-end, connected to the hypervisors through the Service Network. The front-end uses this network to monitor the status of the hypervisors and VMs (virtual machines when using KVM or system containers when using LXD), as well as to initiate VM or storage related operations.

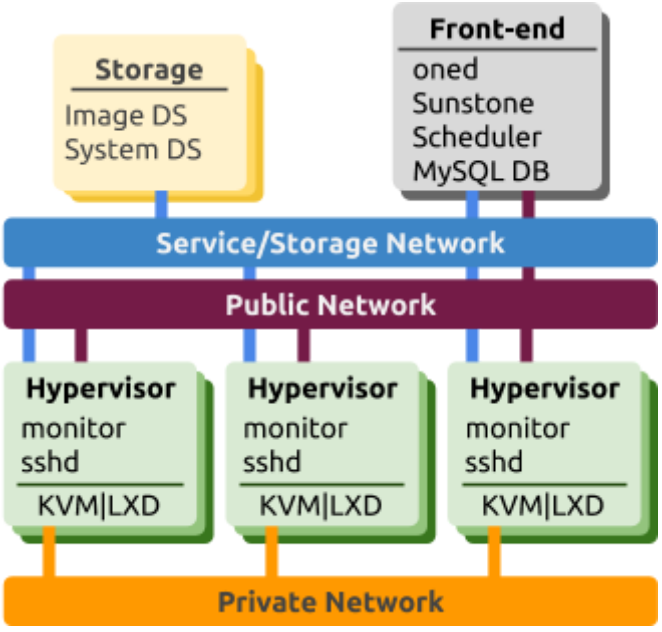

Figure 1. Reference Architecture, a bird's eye view

Virtualization or container hypervisors are also connected to the storage back-end of the cloud through the Storage Network. Given the low network traffic required by OpenNebula to operate, this network can be the same as the dedicated Storage Network. The storage backend is responsible for providing storage support for the running VMs (System Datastore) and for the image repositories (Image Datastore).

VMs usually require two type of network interconnections: private and public. The Private Network implements isolated virtual networks (VLAN) for the internal communication of VMs. Access to each virtual network can be restricted to different users or groups or limited through quotas. Also some VMs needs to communicate to the world so access to a Public Network connected to Internet is recommended for some hypervisors.

[1] http://opennebula.org/about/key-features/

Based on our broad experience deploying clouds for our customers in variety of scenarios, we recommend two cloud implementations of the previous architecture: basic for medium sized clouds (from a few to tens of hypervisors); and advanced for medium to large sized clouds (starting from tens of virtualization hosts up to hundreds). This is not a hard limit, and particularities of the expected workload and hardware characteristics may impact the instantiation of this architecture. Table 1 shows the presets for each implementations.

	Basic	Advanced
Operating System	Supported OS (Ubuntu for KVM|LXD or CentOS/RHEL for KVM) Specific OpenNebula packages installed	
Hypervisor	KVM | LXD	
Networking	VLAN 802.1Q	VXLAN
Storage	Shared file system (NFS/GlusterFS) using qcow2 format for Image and System Datastores	Ceph Cluster for Image and System Datastores
Authentication	Native authentication or Active Directory	

Table 1. Summary of the Basic and Advanced implementations

Each implementation Basic or Advanced can be customized by adding specific authentication methods, access to external providers or even setting up multiple zones in geographically distributed datacenters; as described in the next sections. The lines separating the two architectures are flexible, for instance it may be the case that VXLAN applies to set ups with few nodes and the other way round, VLAN may apply to large scale infrastructures, which also may prefer glusterFS over Ceph due to its VM characteristics.

3. OpenNebula Front-End

The OpenNebula front-end is an special node -a compute server or a VM- devoted to orchestrate all cloud resources. The recommended Operating Systems for the front-end are CentOS/RHEL and Ubuntu.

This node provides the following services:

- OpenNebula management daemon and scheduler
- MySQL DB
- Administration and User GUI and API's
- OpenNebula services like OneFlow or OneGate

Note that some of these services are optional and can be also deployed in a different host (e.g. dedicated MySQL cluster or separated Sunstone or OneFlow servers). The server that will act as front-end needs to have at least 8GB of memory and 4 cores, please take into account that these recommendations are meant as a guidance and may be relaxed or increased depending on the size and workload of your cloud.

The maximum number of servers (virtualization hosts) that can be managed by a single OpenNebula instance strongly depends on the performance and scalability of the underlying platform infrastructure, mainly the storage subsystem. The general recommendation is that no more than 2,500 servers and 10,000 VMs managed by a single instance.

4. Virtualization Nodes

Compute nodes are responsible for providing VMs (KVM) or system containers (LXD) with execution resources (e.g. CPU, memory, or network access). The recommended Operating Systems for virtualization nodes are CentOS/RHEL and Ubuntu. The recommended hypervisors in the reference architecture are KVM (virtual machines) and LXD (system containers) hypervisors delivered by the platform OS. The configuration of the nodes will be homogenous in terms of software components installed, oneadmin administration user and accessible storage. characteristics of virtualization nodes are the same for the Basic and Advanced architecture. The recommendation is to have the least number of nodes with the most number of cores possible.

One of the most important tasks defining a cloud infrastructure is to dimension the virtualization nodes according to the expected workload. Memory wise, the recommendation is having at least 1GB per core, but this also depends on the expected workload, that is, the characteristics of the VMs that are going to be run in the cloud. For instance, to run 80 VMs with 2GB of RAM and 1 vCPU each, the cloud would need 2 KVM servers with 128GB each (96GB will be free to be distributed among these 80 VMs in any way seem fit) and 80 execution threads which can be delivered by two processors with 10 cores each, and two execution threads per core. Network wise, the recommendation is having 4 NICs present in each virtualization node. Regarding system containers nodes, since LXD avoids using virtual hardware, the demand of resources for the hypervisor is much lower when compared to KVM

Besides KVM and LXD, OpenNebula supports natively VMware vCenter.

5. Storage

Storage is one of the most critical aspects of a cloud infrastructure, and needs to be carefully planned to avoid bottlenecks. OpenNebula works with two different set of datastores, the system datastore to sustain the disks of the running VMs and other files associated with the VM like context CD images and VM checkpoints (for suspended VMs); and the image datastore to contain the catalog of images suitable to build new VMs. There are two different storage setups corresponding to the Basic and Advanced architecture.

Basic Architecture

The proposed storage for the Basic architecture is based on a shared file system like NFS or GlusterFS. NFS can be served by a NAS sustaining both the images and system datastore in OpenNebula. Regarding the NAS capacity, the rule of thumb is that the disks of the VMs running simultaneously must fit on the disk, subtracting the space of the images present on the catalog of the images datastore.

A GlusterFS cluster is also possible for medium sized clouds, in this case the recommendation is to have separated servers, not mixing them with the virtualization nodes. At least two servers are needed with at least 1TB of disk, 16Gb of RAM, 2 CPUs of 4 cores each and at least 2 NICs; the interconnection network in this case should be at least 10Gb/s. The servers should be configured in distributed-replicated.

In both cases the images will be stored using *qcow2* format to optimize the disk usage and deployment time, since they will only occupy the space used and will grow up to the maximum space when required. Image clonation uses the *qcow2* backing storage in a COW (Copy on write) fashion. OpenNebula leverages the CoW capabilities of the *qcow2* format to speed-up VM instantiation and optimize datastore usage.

Advanced Architecture

For larger clouds, a Ceph cluster will be used as the storage backend, using its own network for storage distribution. Ceph pools will back the OpenNebula images datastores to hold golden images as well as the

system datastores to hold runtime VM disks.

The Ceph cluster will provide high availability making the data accessible even when one of the storage nodes is down. The recommendation is to have separate servers for the Ceph cluster, not mixing them with the virtualization nodes. At least three servers are needed with 5 disks each of 1TB, 16Gb of RAM, 2 CPUs of 4 cores each and at least 2 NICs.

OpenNebula supports various storage systems -for instance, SAN cabinets exported by Fibre Channel, NetApp and other NAS models, local storage managed by SSH, etc) and various file systems or block devices -for instance LVM, VMFS, etc-.

6. Networking

Networking needs to be carefully designed to ensure reliability in the cloud infrastructure. Depending on the size of the cloud, two recommended configurations are given for Basic and Advanced OpenNebula clouds. Both proposals enable the use of Security Groups, allowing inbound/outbound traffic in VMs network interfaces.

Basic Architecture

The proposed network for Basic architecture is based in three networks. The virtualization nodes are connected to all the networks that are part of the cloud infrastructure. The recommendation is to use at least 10 Gb/s switches that support VLAN trunking to sustain these networks.

Private Network	Communication between VMs. It is important to assign contiguous VLAN identifiers to ports in the switch connected to this network, since the network configuration of OpenNebula will be using 802.1Q VLAN tagging
Public Network	To serve VMs that need internet access
Service Network	For front-end and virtualization node communication -including inter node communication for live migration-, as well as for storage traffic
Storage Network	To serve the shared file system to the virtualization nodes ***(optional)***

Only one interface connected to the Service Network is needed in the front-end. This interface would be used by OpenNebula to connect to the shared file system server and the virtualization nodes through the SSH protocol. Isolation between different private virtual networks can be achieved using 802.1Q VLAN tagging, using different VLANs for different private networks.

Advanced Architecture

For larger clouds, a dedicated Storage Network is advised. The virtualization nodes are connected to all the networks that are part of the cloud infrastructure. The recommendation is to use 10GB/s switches to sustain the Storage, the Private, Public and Service Network.

Private Network	Communication between VMs
Public Network	To serve VMs that need internet access
Service Network	For front-end and virtualization node communication -including inter node communication for live migration-, as well as for storage traffic

Storage Network	To serve the Ceph pools to the virtualization nodes

The Advanced architecture assumes the use of VXLAN, a network virtualization technology aimed to solve large cloud deployments problems, encapsulating Ethernet frames within UDP packets, and thus solving the 4096 VLAN limit problem. The requirement is being able to use the multicast address for the Service Network. There is a current limitation that needs to be noted in the Linux kernel that only allows the handling of 20 different VXLAN ids in the same hypervisor.

OpenNebula natively supports other network mechanisms with different functionalities, to be chosen using security and flexibility requirements. The available options are using ebtables to achieve layer 2 isolation, the use of OpenvSwitch for advanced network functionality, port-group and vSwitch support for VMware, or the use of a Virtual Router for RADVD, DNS, DHCP, port forwarding, etc.

7. Virtual Machines & Guest Support

Virtual Machine images must contain the OpenNebula contextualization packages in order for OpenNebula to correctly pass needed network and configuration information onto the running VMs. OpenNebula Contextualization packages allows configuration and information sharing between the OpenNebula interface and the guest OS of the VM (e.g scripts can be passed to the VM so they are run at boot time). The following Guest OSs are supported by the for OpenNebula Contextualization packages on KVM:

- CentOS/RedHat >= 6
- Debian >= 6
- Ubuntu >= 11.10
- Windows >= 7
- Windows Server >= 2008
- FreeBSD >= 11.2

LXD can only create Linux instances due to the condition of the host and containers kernel sharing.

8. Authentication

Depending on the corporation, a native OpenNebula authentication subsystem or an Active Directory (AD) server can be used. In both cases, the OpenNebula cloud will be accessible to users by both the CLI and the Sunstone web interface. In the native OpenNebula authentication subsystem, users will be kept in the OpenNebula database as native ones and authentication is done using user/password. Groups will be generated as needed for the different users that will use the system.

Alternatively, users can be authenticated against a corporate AD, which has to be accessible through the service network. Users are created in the OpenNebula user DB table after the first use. Groups of users will be created as needed for the different users that will use the system, and assign them access to resources through Virtual Datacenters (VDCs).

OpenNebula natively supports several authentication mechanisms, like for instance SSH keys, LDAP authentication and X509 credentials.

9. Provisioning Model

The Provisioning Model in OpenNebula is based on VDCs (Virtual Data Centers). A VDC is a fully-isolated virtual infrastructure environment where a Group of users (or optionally several Groups of users), under the

control of a Group Admin, can create and manage compute and storage capacity. The users in the Group, including the Group admin, would only see the virtual resources and not the underlying physical infrastructure. The Physical Resources allocated to the Group are managed by the cloud administrator through a VDC. These resources grouped in the VDC can be dedicated to the Group, providing isolation at the physical level too.

Users are organized into Groups (similar to what other environments call Projects, Domains, Tenants...). A Group is an authorization boundary that can be seen as a business unit if you are considering it as private cloud or as a complete new company if it is public cloud. While Clusters are used to group Physical Resources according to common characteristics such as networking topology, or physical location, Virtual Data Centers (VDCs) allow to create “logical” pools of Physical Resources (which could belong to different Clusters and Zones) and allocate them to user Groups, so enabling their consumption, see Figure 2.

Different authorization scenarios can be enabled with the powerful and configurable ACL system provided, from the definition of Group Admins to the privileges of the users that can deploy virtual machines. Each Group can execute different types of workload profiles with different performance and security requirements.

The following are common enterprise use cases in large cloud computing deployments:

- On-premise Private Clouds Serving Multiple Projects, Departments, Units or Organizations. On-premise private clouds in large organizations require powerful and flexible mechanisms to manage the access privileges to the virtual and physical infrastructure and to dynamically allocate the available resources. In these scenarios, the Cloud Administrator would define a VDC for each Department, dynamically allocating resources according to their needs, and delegating the internal administration of the Group to the Department IT Administrator.
- Cloud Providers Offering Virtual Private Cloud Computing. Cloud providers providing customers with a fully-configurable and isolated environment where they have full control and capacity to administer its users and resources. This combines a public cloud with the control usually seen in a personal private cloud system.

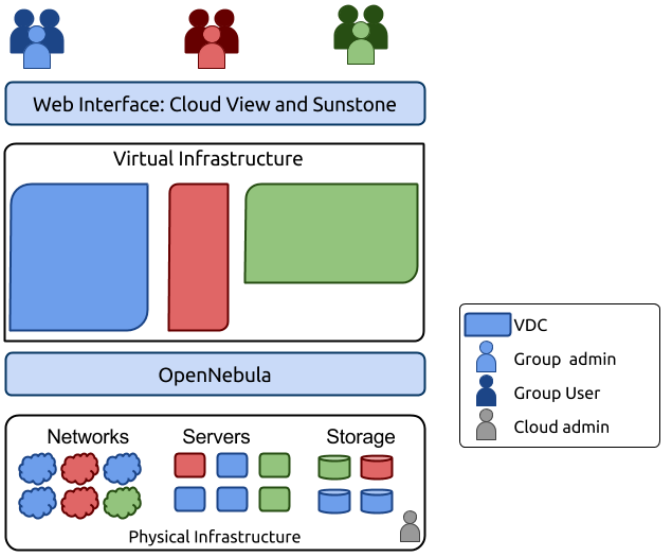

Figure 2: Resource Provisioning Model in OpenNebula

The Cloud will therefore have three different types of users:

- Cloud Admins. Role reserved to the corporation IT staff. They will have admin privileges and will belong to the group "oneadmin".
- Group Admins. These users are allowed to manage virtual resources that belong to that Group, as well as new users. They are allowed to use physical resources associated to each of the VDCs the Group

have access to, in a transparent way.
- Customers or End users. Instantiate and manage VMs based on the predefined setups defined by both the Cloud and Group Admins.

10. Datacenter Federation

Several OpenNebula instances can be configured as a federation. Each instance of the federation is called a Zone, one of them configured as master and the others as slaves. An OpenNebula federation is a tightly coupled integration, all the instances will share the same user accounts, groups, and permissions configuration. Access can be restricted to certain Zones, and also to specific Clusters inside that Zone. Sunstone allows to change the active Zone at any time, and it will automatically redirect the requests to the right OpenNebula at the target Zone.

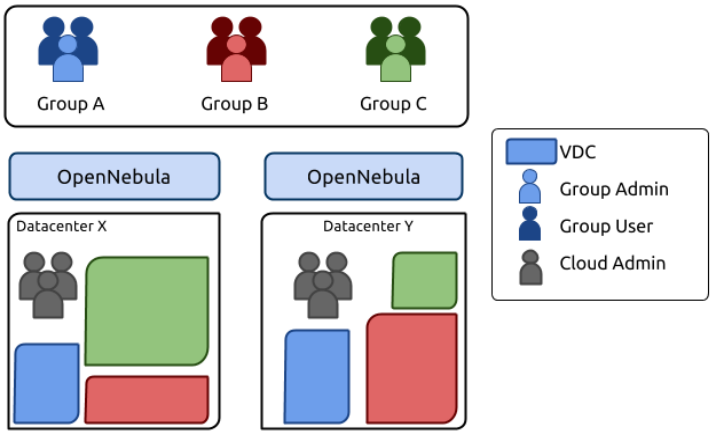

Figure 3: Auth Federation Architecture

11. Cloud Bursting

Cloud bursting is the model in which local resources of a private cloud are combined with resources from remote cloud providers hence creating a hybrid cloud. The remote provider could be a commercial cloud service, such as Amazon EC2, MS Azure, or even other OpenNebula based cloud. This support for cloud bursting enables highly scalable hosting environments. The architecture for hybrid clouds is described in Figure 5.

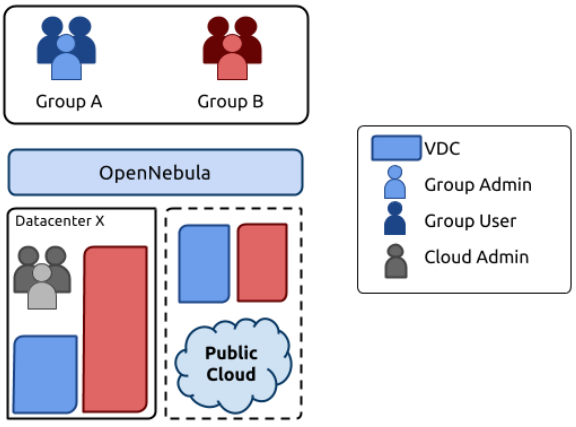

Figure 5: Hybrid Cloud architecture enabling cloud bursting

OpenNebula approach to cloud bursting is unique. The reason for this uniqueness is the transparency to

both end users and cloud administrators to use and maintain the cloud bursting functionality. Transparency to cloud administrators comes from the fact that a public cloud region is modelled as any other host (albeit of potentially a much bigger capacity), so the scheduler can place VMs in cloud region.

12. High Availability

OpenNebula uses a distributed consensus protocol, based on Raft, to provide fault-tolerance and state consistency across OpenNebula services. To preserve a consistent view of the system across servers, modifications to system state are performed through a special node, the leader. The servers in the OpenNebula cluster elects a single node to be the leader. The leader periodically sends heartbeats to the other servers, the followers, to keep its leadership. If a leader fails to send the heartbeat, followers promote to candidates and start a new election. Read-only operations can be performed through any oned server in the cluster; this means that reads can be arbitrarily stale but generally within the round-trip time of the network.

To ensure a proper functioning of the HA cluster, a minimum of three front-end needs to be deployed, to support one node failure. This number can be increased provided the target number is odd. For instance, a 5 front-end configuration can be deployed to support two node failures

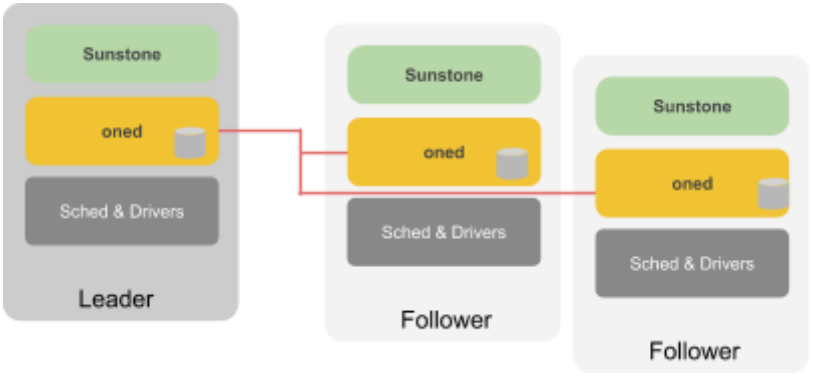

Figure 6: Overview of the HA architecture and main components

HA can also be configured for VMs, to be re-launched if they enter a fail state, or it can be configured for virtualization nodes, so all the VMs running in a crashed node get moved automatically to another virtualization node.

13. Conclusions

The reference architecture described in this document has been created from the collective information and experiences from hundreds of users and cloud client engagements to help in the design and deployment of open cloud infrastructures. The document recommends software products and configurations for a smooth OpenNebula installation. However, in many cases, there are other aspects to be taken into account, like infrastructure platforms and services preexisting in the datacenter as well as the provisioning processes of the corporation. In these scenarios, OpenNebula can be easily adapted to fit into your data center and policies. Contact us, we look forward to helping you at any stage of cloud computing adoption.

LET US HELP YOU DESIGN, BUILD AND OPERATE YOUR CLOUD

CONSULTING & TRAINING

Help you design, integrate, build and operate an OpenNebula cloud infrastructure

SUPPORT SUBSCRIPTION

From OpenNebula developers with product influence and privacy guarantee at a competitive cost

JUMPSTART PACKAGES

Help you springboard your productivity, speed time to deployment, and reduce business and technical risks

Sign up for updates at OpenNebula.org/getupdated

Rev1.6_20190301