Open Cloud Reference Architecture

Version 1.8 – February 2020

Abstract

The OpenNebula Cloud Reference Architecture is a blueprint to guide IT architects, consultants, cloud 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 document includes references to software products, specific configurations, and requirements of infrastructure platforms recommended for a **smooth OpenNebula installation**. Three optional functionalities complete this architecture: high availability, cloud bursting for workload outsourcing, and federation of geographically dispersed data centers.

This document describes the reference architecture for Basic (small to medium-size) and Advanced (medium-size to large) OpenNebula clouds, providing recommended software for main architectural components, including support for **KVM-based virtual machines and LXD system containers**. Each section 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. There are of course 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 (e.g. configuration management and automation tools for managing cloud infrastructure and large numbers of devices).

Contents

Glossary

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

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. The development of OpenNebula follows a bottom-up approach driven by the real need of sysadmins, devops and users.

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 for storage related operations.

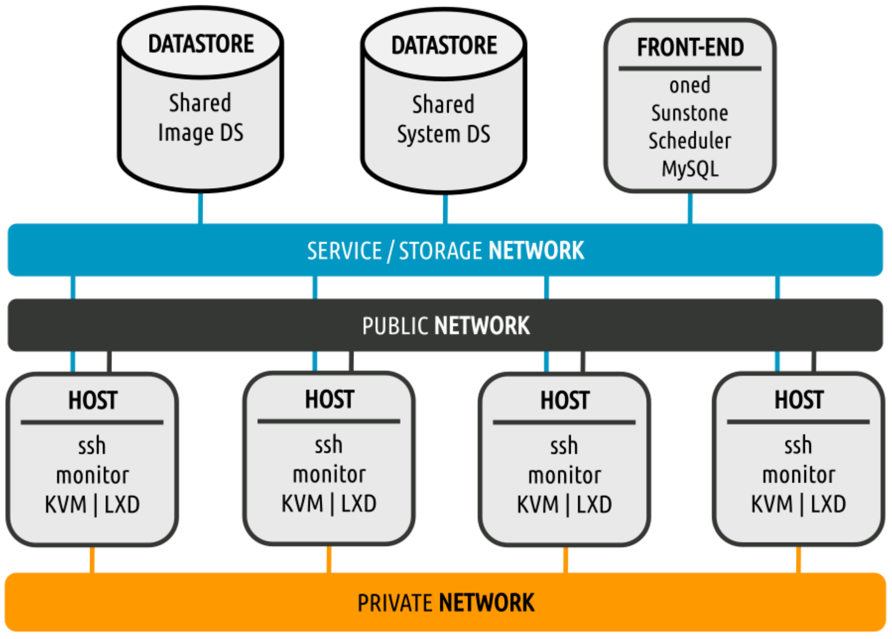

Figure 1. Reference Architecture, a bird's eye view.

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 types 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. Additionally, some VMs

[1] http://opennebula.io/discover/

need to communicate to the world so access to a Public Network connected to the Internet is recommended for some hosts.

Based on our broad experience deploying clouds for our customers in a 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 pre-sets 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. 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 a special node—a physical server or VM—devoted to orchestrating all cloud resources. The recommended Operating Systems for the front-end are CentOS/RHEL and Ubuntu, and the hardware recommendations can be checked in Table 2. Please take into account that these recommendations are meant only as a guidance.

Memory	8 GB
CPU	2 CPU (4 cores)
Disk size	200 GB
Network	2 NICs

Table 2. Front-end hardware recommendations.

The front-end provides the following services:

- OpenNebula management daemon
- Scheduler
- MySQL DB
- Administration and User GUI and APIs
- Optional 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. a dedicated MySQL cluster or separated Sunstone or OneFlow servers).

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 should be 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), both being delivered by the platform OS.

The configuration of the nodes will be homogenous in terms of the installed software components, OpenNebula administration user, and accessible storage. Characteristics of virtualization nodes are the same for the Basic and Advanced architectures. The recommendation is to minimize the number of nodes whilst maximizing the number of cores per node.

A key task when defining a cloud infrastructure is to correctly 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 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.

Apart from KVM and LXD, OpenNebula also comes with native support of 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 sets of datastores:

- The system datastore, which sustains the disks of the running VMs and other files associated with the VM, such as context CD images and VM checkpoints (i.e. for suspended VMs).
- The image datastore, which contains the catalog of images suitable to build new VMs.

There are two different storage set-ups 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 Network Attached Storage (NAS) sustaining both the images and system datastore for 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 image datastore.

A GlusterFS cluster is also possible for medium-sized clouds. In this case the recommendation is to have separate servers, and 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 and the servers should be configured in distributed-replicated mode.

In both cases the images will be stored using qcow2 format to optimize disk usage and deployment time, since they will only occupy the space in use and will grow up to the maximum space only 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 is recommended 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 of 1TB each, 16Gb of RAM, 2 CPUs of 4 cores each, and at least 2 NICs.

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

6. Networking

Networking needs to be carefully designed to ensure reliability of 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 on 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 supporting VLAN trunking to sustain these networks.

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 via SSH protocol. Isolation between different private virtual networks can be achieved using 802.1Q VLAN tagging, with different VLANs for different private 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)

Table 3. Proposed networks for Basic Architecture.

Advanced Architecture

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

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

Table 4. Proposed networks for Advanced Architecture.

The Advanced architecture assumes the use of VXLAN, a network virtualization technology designed for dealing with large cloud deployments, encapsulating Ethernet frames within UDP packets and thus solving the 4096 VLAN limit problem. The requirement for this is to be able to use the multicast address for the Service Network. It is worth noting that there is a limitation in the Linux kernel by which only the handling of 20 different VXLAN ids in the same hypervisor is currently allowed.

OpenNebula comes with native support to other network technologies, suitable for additional security and flexibility requirements. The available options include, for example, ebtables for Layer 2 isolation, the use of OpenvSwitch for advanced network functionalities, 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 for OpenNebula to be able to correctly pass the necessary network and configuration details onto the running VMs. OpenNebula Contextualization packages allows configuration and information sharing between the OpenNebula interface and the guest Operating System of the VM (e.g. scripts can be passed to the VM so they are run at boot time).

The following list contains a sample of guest OSs that are supported by the OpenNebula Contextualization packages on KVM (an exhaustive list is available in the official documentation):[2]

- CentOS >= 6
- Red Hat Enterprise Linux >= 7
- Debian >= 8
- Ubuntu >= 14.04
- Windows >= 7
- Windows Server >= 2008

LXD can only create Linux instances given that host and containers must be able to share the same kernel.

8. MarketPlace

OpenNebula has access to its own MarketPlace, which enables the user to import images from a public repository (containing images of common use that have been tested and certified by OpenNebula Systems) or from private repositories. These images can be added to a datastore and used to add them to existing VM templates or instances.

OpenNebula public marketplace features images in qcow2 format. In this way images can be used by KVM hypervisors, LXD system containers and vCenter Servers.

As for private repositories, there are two possibilities:

- HTTP MarketPlace, where images are accessible through a HTTP server (e.g. Apache, Nginx).
- S3 MarketPlace, where images are accessible through an Amazon S3 API.

9. Authentication

Either the native OpenNebula subsystem or a LDAP/Active Directory (AD) server can be used for authentication purposes. In both cases, the OpenNebula cloud will be accessible to users through the CLI and the Sunstone GUI. With the native OpenNebula authentication subsystem, users' details and credentials (username/password) will be kept in the OpenNebula database and groups will be generated as needed.

Alternatively, users can be authenticated against a corporate LDAP/Active Directory (AD) server, which has to be accessible through the service network. Users are created and added to the appropriate OpenNebula user DB table after the first use. Groups of users will be created as needed, and access to resources will be assigned to them through the definition of Virtual Data Centers (VDCs).

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

10. 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

[2] https://github.com/OpenNebula/addon-context-linux

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 in other environments are called Projects, Domains or Tenants). A Group is an authorization boundary that can be seen as a business unit—if you are considering it as a private cloud—or as a completely separate company—if it is a 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 the cloud administrator to create “logical” pools of Physical Resources (which can belong to different Clusters and Zones) and allocate them to specific user Groups, so enabling their consumption only by users in those Groups (see Figure 2).

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

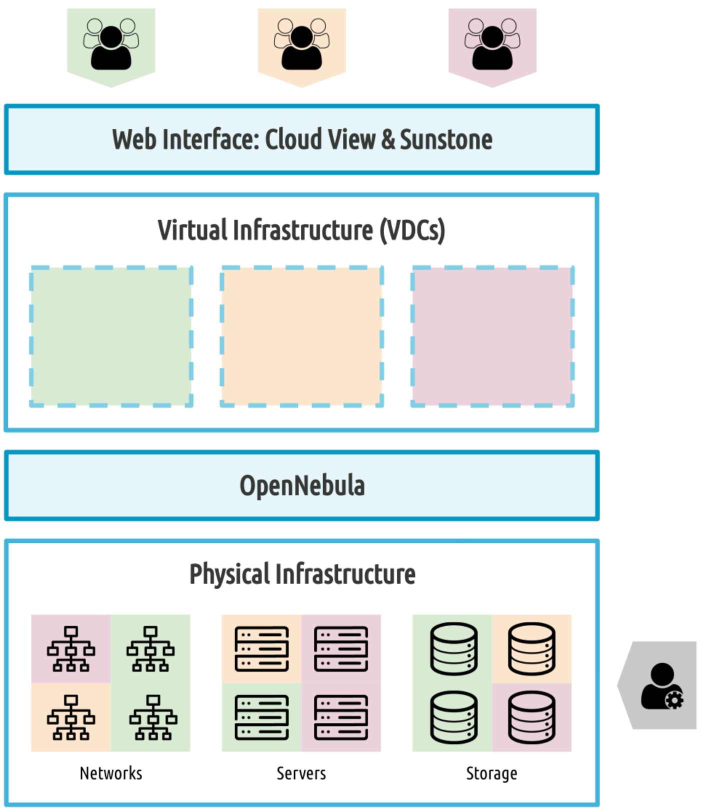
Web Interface: Cloud View & Sunstone
Virtual Infrastructure (VDCs)
OpenNebula
Physical Infrastructure
Networks
Servers
Storage

Figure 2. Resource Provisioning Model in OpenNebula.

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 their virtual and physical infrastructure, and to dynamically allocate the available resources among different projects and departments. 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’s IT administrator.

- **Cloud providers offering Virtual Private Clouds.** Cloud providers provide 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 an enterprise private cloud system.

The Cloud will therefore have three different types of users:

- Cloud Admins: Role reserved to the company's IT staff will full admin privileges.
- Group Admins: These users are allowed to manage virtual resources that belong to a specific Group, as well as its users. They are allowed to use physical resources associated with each of the VDCs the Group has access to, in a transparent way.
- Customers / End users: Allowed to instantiate and manage VMs based on the predefined set-ups defined by both the Cloud and Group Administrators.

11. Datacenter Federation

If administration domains need to be isolated, or the interconnection between datacenters does not allow a single controlling entity, OpenNebula can be configured in a federation. Each OpenNebula instance of the federation is called a Zone, one of them configured as primary and the others as secondaries. An OpenNebula federation is a tightly coupled integration, with all the instances sharing the same user accounts, groups and permission configurations (see Figure 3).

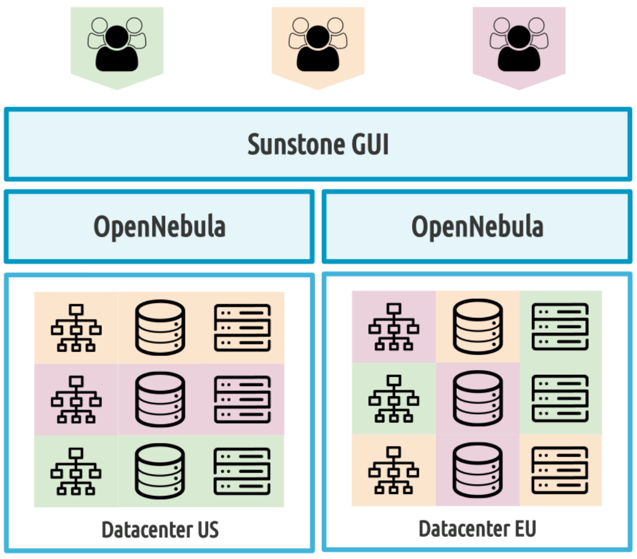

Figure 3. Federation Architecture.

Federation allows end users to consume resources allocated by the federation administrators regardless of their geographic location. The integration is seamless, meaning that a user logged into the Sunstone GUI of a Zone will not have to log out and enter the address of another Zone. Sunstone allows users to change the active Zone at any time, and it will automatically redirect the requests to the right OpenNebula instance.

12. Cloud Bursting

Cloud bursting is the model in which local resources of a private cloud are combined with resources from a remote cloud provider hence creating a hybrid cloud. The remote provider could be a commercial cloud service, such as Amazon Web Services (AWS), Microsoft Azure, or even some other OpenNebula cloud.

The architecture for hybrid clouds is described in Figure 4. The support that OpenNebula provides for cloud bursting enables highly scalable hosting environments. In fact, OpenNebula's 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 at any cloud region.

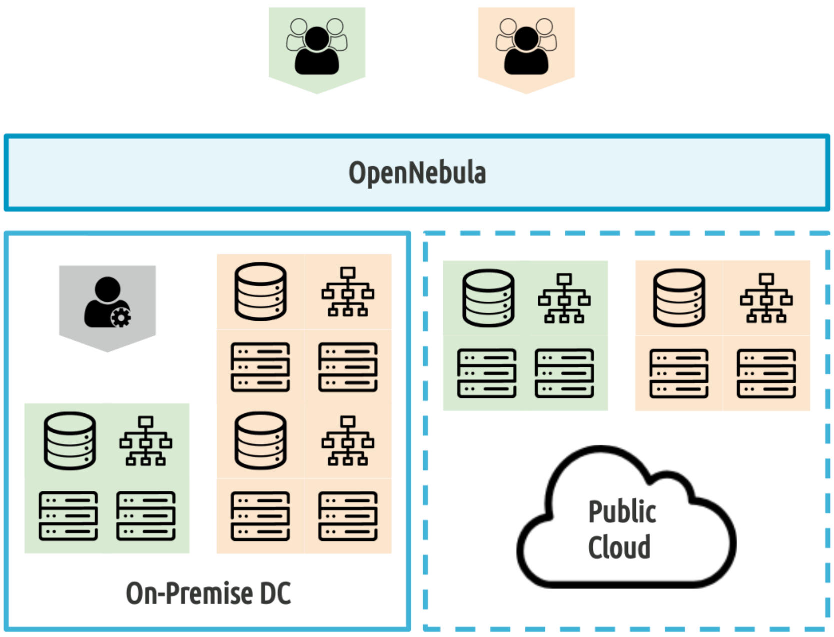

Figure 4. Hybrid Cloud architecture enabling cloud bursting.

13. High Availability

OpenNebula uses a distributed consensus protocol, based on Raft, to provide fault-tolerance and state consistency across OpenNebula services.

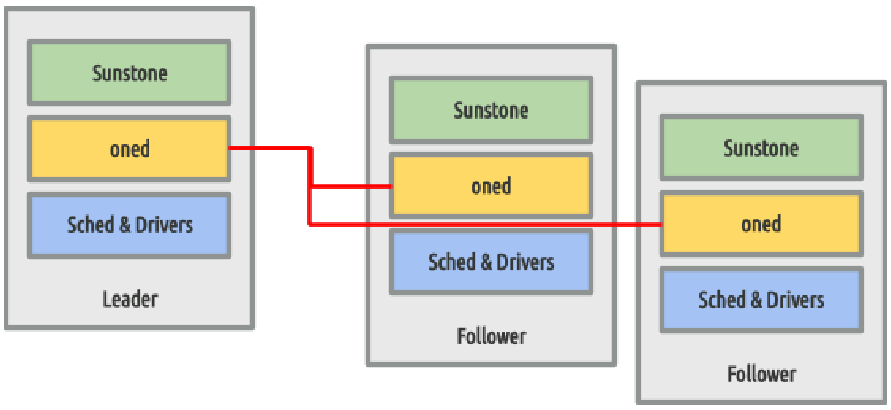

Figure 5. Overview of the HA architecture and main components.

To preserve a consistent view of the system across servers, modifications to the system state are performed through a special node—the leader. The servers in the OpenNebula cluster elect 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 OpenNebula server (oned) in the cluster; this means that reads can be arbitrarily stale but generally within the round-trip time of the network. A minimum of three front-end needs to be deployed in order to support one node failure. HA can also be configured for VMs (i.e. 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 node.

14. Ready for a Test Drive?

You can evaluate OpenNebula on your vCenter environment and build a cloud in only 5 minutes by downloading miniONE,[3] our deployment tool for installing a single-node KVM or LXD cloud inside a virtual machine or physical host.

15. 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. This 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 pre-existing services in the data center, as well as specific provisioning processes within the company. In these scenarios, OpenNebula can be easily adapted to fit into your data center and corporate 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 private or hybrid 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 risks

Sign up for updates at OpenNebula.io/getupdated

Rev1.8_20200229

[3] https://github.com/OpenNebula/minione