Hyperconverged Cloud Architecture with OpenNebula and StorPool

Version 1.0, January 2018

Abstract

The Hyperconverged Cloud Architecture with OpenNebula and StorPool is a blueprint to aid IT architects, consultants, administrators and field practitioners in the design and deployment of a hyperconverged cloud infrastructure. In this architecture, nodes combine computing, storage and virtual networking functions to achieve better efficiency and simplified management. The Hyperconverged Cloud Architecture is a useful guide for all companies willing to launch fast, easy to use and cost-effective clouds.

This document describes the architecture of a complete IT infrastructure including storage, computing, networking and cloud orchestration. There are other components in the open cloud ecosystem that are not part of the 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

Glossary

ACL	Access Control List
AD	Active Directory
DC	Datacenter
ONE	OpenNebula
QCOW	QEMU Copy on Write image format
SAN	Storage Area Network
vCPU	Virtual CPU
VDC	Virtual Datacenters
VM	Virtual Machine
VXLAN	Virtual Extensible LAN

1. What is OpenNebula?

Enterprise cloud computing is the next step in the evolution of data center (DC) virtualization. OpenNebula is a simple but feature-rich and flexible solution to build and manage enterprise clouds and virtualized DCs that enhances and extends 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.

No two DCs are the same so there is not a one-size-fits-all in the cloud. OpenNebula does not try to provide a turnkey solution that imposes fixed requirements on DC infrastructure. OpenNebula makes cloud an evolution by leveraging existing IT infrastructure, protecting your investments, and avoiding vendor lock-in. Despite the flexibility provided by OpenNebula architecture, this document aims to help those users having doubts regarding the infrastructure environment they should be setting up to build a hyperconverged cloud.

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.

2. What is StorPool?

StorPool is a software-defined storage solution for building high-performance public and private clouds. It runs on standard servers, drives and network and turns them into an outstanding storage system.

StorPool is block-storage software installed on the servers, which creates a shared storage pool from the local drives in these servers. Compared to traditional SANs, all-flash arrays, or other storage software StorPool is faster, more reliable and scalable. StorPool is designed form the ground up to provide cloud builders with the fastest and most resource-efficient storage software on the market. It is tailored for companies which build public or private clouds. It provides you with a fast and reliable shared storage for your cloud and can reduce total costs of building and operating the cloud several times.

3. Benefits of a Hyperconverged Cloud

The main goals when deploying a hyperconverged cloud are usually to gain simplicity, performance and efficiency, without sacrificing features or reliability. The benefits of the proposed hyperconverged cloud architecture are as follows:

Optimized infrastructure costs - Hyperconverged infrastructure provides a low-resource profile without sacrificing performance levels or centralized control. The hyperconverged cloud reduces the need for expensive, specialized staff, because it is easy to manage.

Better performance and reliability - A well-designed hyperconverged infrastructure has a lot of performance available to be shared by all applications. In the hyperconverged concept, there's usually one fast tier of storage and computing, so all applications benefit.

Simplified management - The fewer systems to manage you have, the easier it would be for your company. Especially if you do not have a huge team of dedicated system administrators or infrastructure engineers.

Simplified hardware inventory - Repeated identical building block.

Scalability – Scale compute and storage together. Capacity and performance can be scaled by adding a new server to the cluster. Each VM can be scaled up to very large CPU, RAM and storage size and performance.

4. High Level Architecture

A cloud architecture is defined by components in four planes: cloud orchestration, computing, storage and networking. Figure 1 shows the bird's eye view of the cloud. OpenNebula services run in a cloud orchestration plane and communicates with other components over the Service Network. OpenNebula services monitor the status of the hypervisors and Virtual Machines (VMs), and initiate VM or storage related operations.

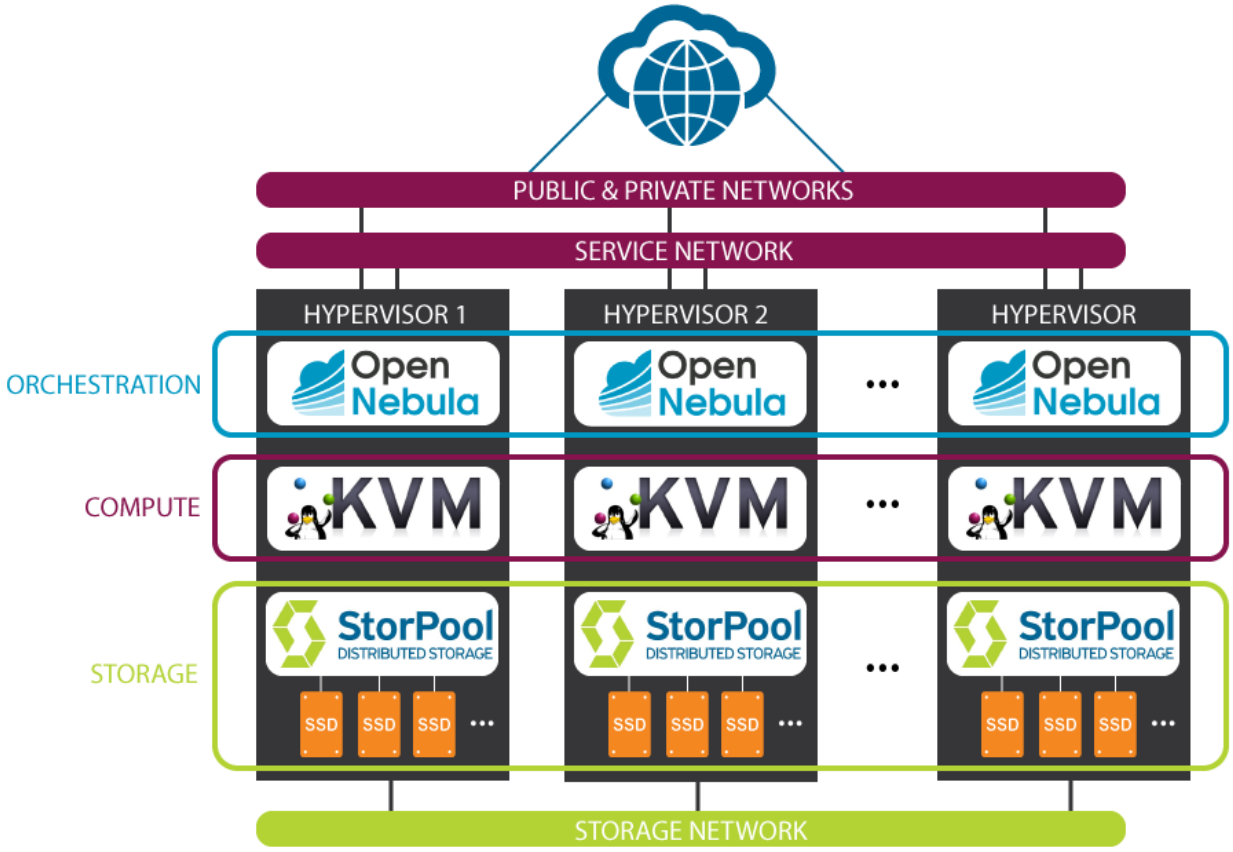


Figure 1. Hyperconverged Architecture, a bird's eye view

Hypervisors nodes have identical configurations running the same set of components in all planes. This architecture provides efficient resource utilization, high level of redundancy and simplified management. Each node contains CPU, memory, disk and networking resources, to act as a storage and compute node simultaneously. It runs the KVM hypervisor, StorPool storage and OpenNebula drivers to manage local resources. Each hypervisor is connected to all 3 networks - instance, service and storage networks.

VMs require two types of network interconnections: private and public. The Private Network implements

isolated virtual networks (VLAN) for the internal communication between VMs. Access to each virtual network can be restricted to different users or groups or applied different QoS settings and rate limits. The Public Network is used to connect VMs to the Internet.

Operating System	Supported host OS (Ubuntu, CentOS or RHEL) on all hosts
Hypervisor	KVM
Networking	Redundant 10Gbps storage network, 1Gbps or 10Gbps network shared by service network, public and private virtual networks
Storage	StorPool
Authentication	Native authentication or Active Directory

Table 1. Summary of the implementation

5. Cloud Orchestration

All resources of the hyperconverged cloud are orchestrated by the OpenNebula front-end. In the Hyperconverged Architecture, all OpenNebula front ends are running in virtual machines. There is an option for one of the OpenNebula front ends to be on a physical server. The recommended Operating Systems for the front-end are CentOS, RHEL or Ubuntu.

This component provides the following services:

- OpenNebula management daemon
- Resource Scheduler
- MySQL database
- Administration and User GUI and API's
- Optional OpenNebula services: OneFlow, OneGate, …

OpenNebula front ends are using a distributed consensus protocol to provide fault-tolerance and state consistency across OpenNebula services.

6. Hyperconverged Nodes

In hyperconverged cloud architecture, each node implements all cloud functions - Virtualization, Storage and Networking. This provides high level of availability with optimum resource utilization.

Compute

Compute nodes are responsible for providing VMs with execution resources (e.g. CPU, memory, or

network access). The recommended Operating Systems for the virtualization nodes are CentOS/RHEL and Ubuntu. The recommended hypervisor in the architecture is the KVM delivered by the platform OS. The configuration of the nodes is homogenous in terms of the software components installed, the oneadmin administration user and accessible storage. To keep the cloud simple and efficient, it is recommended to have a smaller number of nodes with a high number of CPU cores.

One of the most important tasks defining a cloud infrastructure is to dimension the virtualization nodes according to the expected workload. A typical virtualization node, used by cloud service providers in 2017 had approximately 384 GB RAM and 40 CPU cores (80 logical CPUs). This equates to 4.8 GiB RAM per logical CPU. Depending on the expected workload and its requirements in terms of memory vs CPU, as well as resource over-subscription, the optimal amount of RAM per node might be smaller or larger. For example, if the requirement is for 2 GB RAM per vCPU and there will be no vCPU over-subscription, a hypervisor with 80 logical CPUs would need only 160-192 GB RAM. In any case using larger, denser servers like the ones described here is usually much more efficient than using servers with a much smaller number of CPU cores.

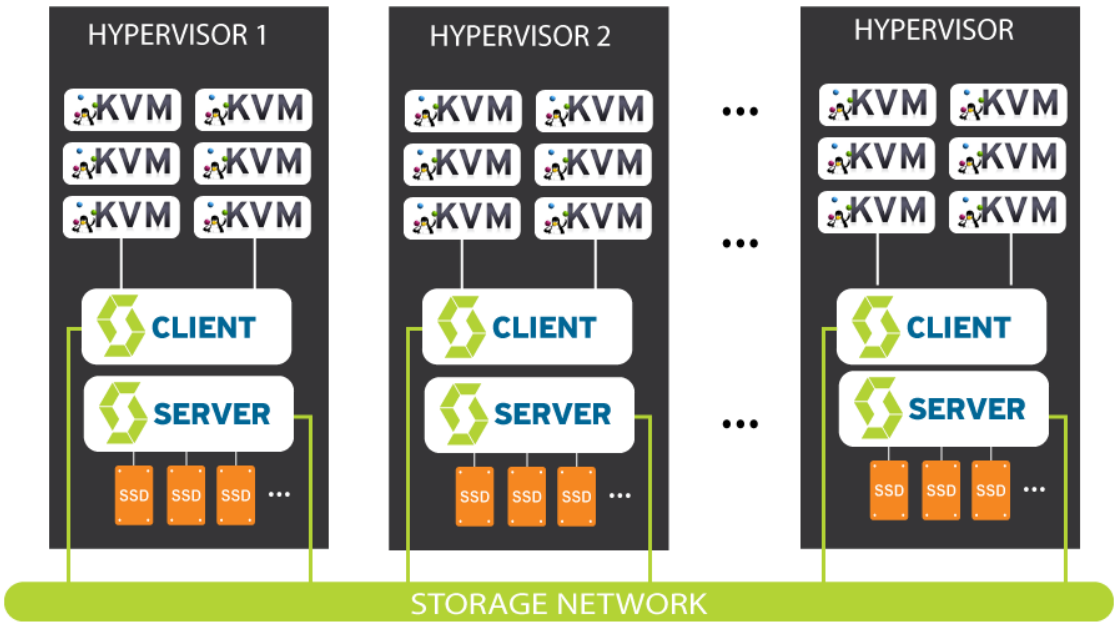
Networking

In terms of networking, we recommend 4 network interfaces per server. 2 interfaces would be used for storage functions and 2 additional interfaces will be used for public, private and service networks.

Storage

Storage is one of the most critical aspects of a cloud infrastructure, and needs to be carefully planned to avoid bottlenecks.

StorPool distributed storage presents unique characteristics - delivering consistently high performance, low latency and availability using a small and fixed amount of resources. StorPool is built using a distributed, shared-nothing architecture. All functions are performed by all nodes on an equal peer basis. The software consists of two parts - a storage server, and a storage client that are installed on each physical server (host). All storage servers and clients are connected over a dedicated 10G/25G/40G storage network for minimum latency and maximum throughput.



Each storage node is responsible for data stored on its local drives. Storage nodes collaborate to provide the storage service. StorPool provides a shared storage pool combining all the available storage capacity. Every volume (VM disk or ONE image) is distributed on multiple storage nodes and multiple disks, ensuring resilience against server, disk or network failure. StorPool uses synchronous replication across servers. The StorPool client communicates in parallel with all StorPool servers to achieve the aggregate throughput and IOPS of all installed disks in the cluster.

Virtual disks in KVM, correspond to block devices in the host provided by the StorPool client, which correspond directly to volumes in the StorPool storage system. Hence virtual machine images are stored directly in the StorPool volumes using the "raw" format, without a filesystem or a VM image format (such as QCOW) in between, reducing the overhead and maximizing the performance. All operations on the VM images, such as creating snapshots, cloning images and thin provisioning are implemented natively by StorPool.

StorPool volume management is integrated with OpenNebula to allow seamless use of the capabilities of the storage system through OpenNebula GUI, CLI and API interfaces. The end result of the deep integration between the two systems is that creating an image in OpenNebula directly corresponds to creating a volume in StorPool. When provisioning a number VMs based on a VM template, the VMs don't take additional space (they refer to the base image), thus saving space.

Data redundancy is provided by multiple copies (replicas) of the data written synchronously across the nodes in the cluster. We recommend that 3 copies are used for all types of regular data. Only in exceptional circumstances, where data durability is not important, for example when it can be recreated with a simple calculation, it could be stored with fewer than 3 copies.

7. Networking

In hyperconverged cloud, three separate networks are defined - instance networks (public, private), that provides connectivity to the VMs across different hosts, service network - used by OpenNebula front end to manage hypervisors and storage, and storage network used by StorPool storage. In this Hyperconverged Architecture storage network is a dedicated physical 10Gb/s network, and service and instance networks share another physical network.

Instance Network	Private virtual networks for communication between VMs, and public virtual networks for connectivity to external networks and Internet. Implemented with 802.1Q VLAN tagging or VXLAN
Service Network	For front-end and virtualization node communication and cloud management - including inter-node communication for live migrations
Storage Network	Used by StorPool storage for high-speed transfer of data between nodes. A dedicated redundant 10Gb/s network not connected to external networks or elements.

Isolation between different virtual networks can be achieved using 802.1Q VLAN tagging or VXLANs.

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 the 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).

9. Provisioning Model and Multi-tenancy

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 separate 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 cloud computing deployments:

- On-premises Private Clouds Serving Multiple Projects, Departments, Units or Organizations. On-premises 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.

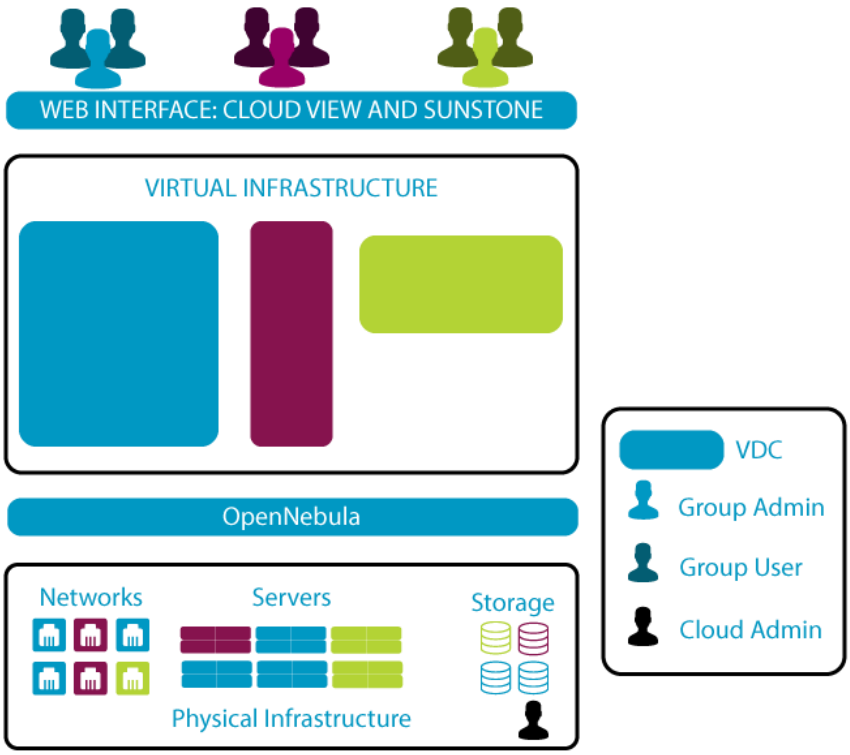


Figure 2: Resource Provisioning Model in OpenNebula

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 each Zone. Federation in an OpenNebula starts with an authentication & authorization federation, where users can consume resources from various OpenNebula Zones. This federation can be extended to the network level, where virtual networks can be made visible across different OpenNebula Zones.

Authentication & Authorization Federation

A typical scenario for an OpenNebula federation is a company with several DCs, distributed in different

geographic locations, as depicted in Figure 3. The requirement to achieve OpenNebula federation is to have DC interconnection with dedicated links with latencies under 300ms.

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 web interface of a Zone will not have to log out and enter the address of another Zone. Sunstone allows to change the active Zone at any time, and it will automatically redirect the requests to the OpenNebula at the selected Zone.

A federation will have a unique *oneadmin* account. That is the federation administrator account. In a trusted environment, each Zone administrator will log in with an account in the *oneadmin* group. In other scenarios, the federation administrator can create a special administrative group with total permissions for one Zone only.

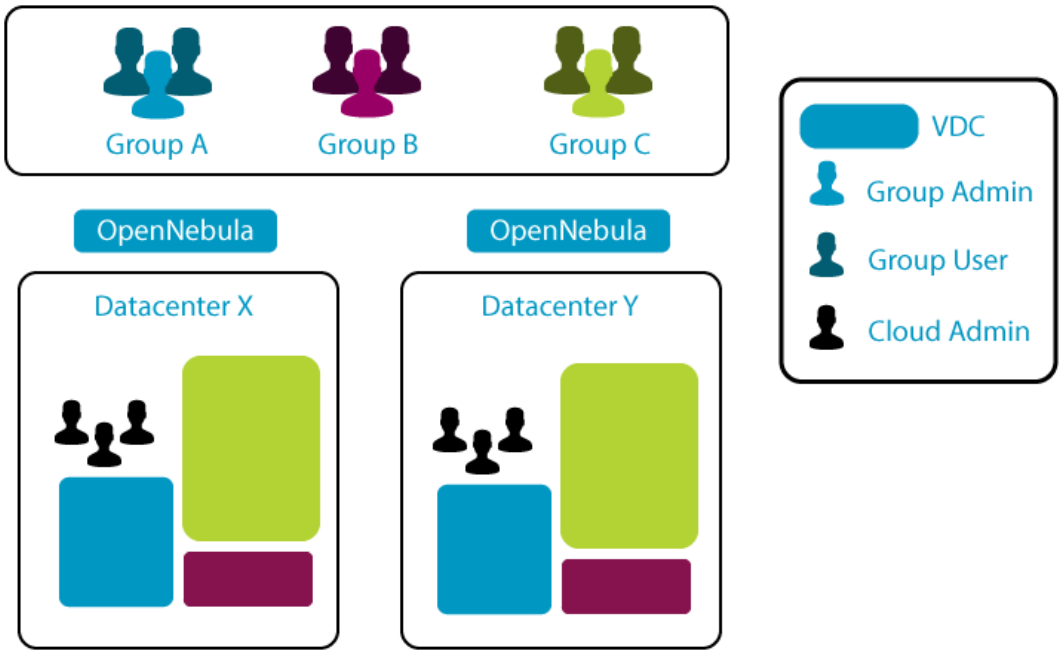


Figure 3: Auth Federation Architecture

Network Federation

Virtual networks can be connected across OpenNebula Zones. This network federation is achieved through inter-DC routers implemented as a network virtualized function. Leveraging OpenNebula functionality the routers can be dynamically configured to interconnect different networks. Also, traffic across virtual networks can be filtered with Security Groups, so traffic from/to different VLAN segments can be allowed/disallowed.

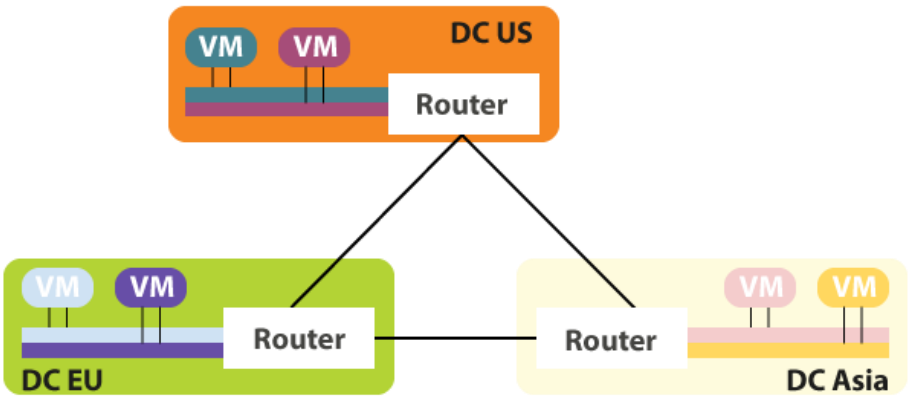


Figure 4: Networking Architecture for cross-DC interconnections

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 or Microsoft Azure. This support for cloud bursting enables highly scalable hosting environments. The architecture for hybrid clouds is described in Figure 5.

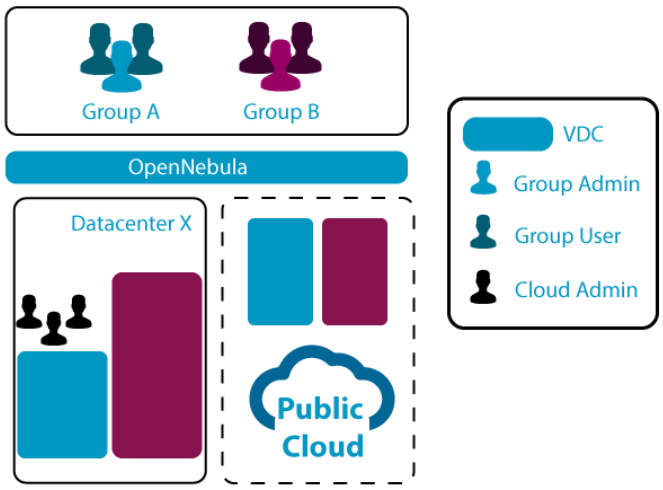


Figure 5: Hybrid Cloud architecture enabling cloud bursting

OpenNebula approach to cloud bursting is quite unique. The reason behind 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 an AWS EC2 region is modelled as any other host (albeit of potentially a much bigger capacity), so the scheduler can place VMs in EC2.

On the other hand, the transparency to end users is offered through the hybrid template functionality: the same VM template in OpenNebula can describe the VM if it is deployed locally and also if it gets deployed

in Amazon EC2. So users just have to instantiate the template, and OpenNebula will transparently choose if that is executed locally or remotely.

12. High Availability

High availability design is implemented in each plane of the Hyperconverged Cloud Architecture.

Cloud Orchestration

In the Cloud Orchestration plane, 3 OpenNebula front ends are running distributed consensus protocol to provide fault-tolerance and state consistency across OpenNebula services. A consensus algorithm is built around two concepts:

- System State, in OpenNebula the system state is the data stored in the database tables (users, ACLs, or the VMs in the system).
- Log, a sequence of SQL statements that are consistently applied to the OpenNebula database in all servers to evolve the system state.

To preserve a consistent view of the system across servers, modifications to system state are performed through a special node, the leader, elected by the cluster. 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.

Whenever the system is modified (e.g. a new VM is added to the system), the leader updates the log and replicates the entry in a majority of followers before actually writing it to the database. The latency of DB operations is thus increased, but the system state is safely replicated, and the cluster can continue its operation in case of node failure.

In OpenNebula, 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.

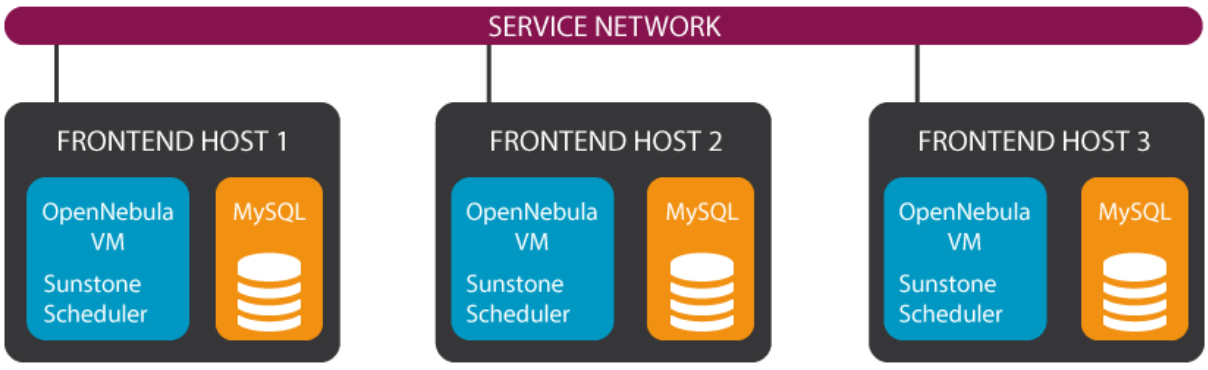


Figure 6: Overview of the OpenNebula HA architecture and main components

Compute plane

In case of a failure of a hypervisor, all VMs are automatically relaunched on remaining operational

hypervisors resulting only a short, predictable downtime for the virtual machines. Planned maintenance operations can be performed without service downtime, by using live migration of the virtual machines off the hypervisor, before it is shut down or restarted. This allows non-service affecting hardware or software upgrades of individual nodes or entire cluster.

Individual virtual machine failures can be handled too, and VMs can be automatically relaunched if needed.

Storage plane

Storage is the most critical component of the cloud in terms of availability. In the storage plane, high availability is built in several layers.

Disk failures are handled by multiple redundant data copies (usually 3) always on different disks and different nodes. In case of a disk failure StorPool storage will continue normal operation using the remaining active copies of the data.

With end-to-end data integrity and self-healing, StorPool automatically recovers from data loss caused by disk or node failures, bit rot, undetected disk, memory or network errors, by restoring missing or corrupted copies to preserve the configured level of data redundancy.

Host failures are mitigated by implementing shared-nothing distributed architecture. All the components inside a hypervisor (client, server, etc.) are replicated across all the available hypervisors to ensure data redundancy and high availability of the system. In case of an issue with a hypervisor, the system continues to work uninterrupted, without downtime or loss of data.

Unlike the StorPool server component which uses an active-active load-balancing redundancy scheme, the StorPool API (used for provisioning and monitoring functions) uses an active-passive redundancy scheme with a floating IP address.

Networking

All hardware nodes are using redundant physical connections to two network switches with automatic fallback mechanisms for host, network card, interface, cable or switch failure. On the public, private and service networks the system uses Linux Bonding to provide a highly-available network connection. Depending on capabilities of the switches the Bond interface will be configured for LACP (active-active load-balancing) or active-backup mode. On the storage network side the network architecture uses StorPool's integrated multi-path functionality to provide HA and use both of the interfaces for increased throughput when they are available.

13. Conclusions

The Architecture described in this document has been created from the collective information and experiences from tens of users and cloud client engagements to help in the design and deployment of hyperconverged infrastructures. The document recommends software products and configurations for a smooth OpenNebula with StorPool cloud installation. However, in many cases, there are other aspects to be taken into account, like infrastructure platforms and services pre-existing in the datacenter as well as the provisioning processes of the corporation.

About OpenNebula

OpenNebula is a turnkey enterprise-ready solution that includes all the features needed to provide an on-premises (private) cloud offering, and to offer public cloud services. With tens of thousands of deployments, OpenNebula is parked in industry leaders like Activision, CA Technologies, Unity, Akamai, CentOS, BBC, Alcatel-Lucent and BlackBerry, and research leaders like FermiLab, ESA and SurfSARA. OpenNebula Systems develops OpenNebula, supports its community, and provides support subscriptions, consulting, and training. OpenNebula Systems has a global presence with offices in Europe and the United States.

About StorPool

StorPool is a software-defined storage solution for building high-performance public and private clouds. It runs on standard servers, drives and network and turns them into an outstanding storage system. StorPool is block-storage software installed on the servers, which creates a shared storage pool from the local drives in these servers. Compared to traditional SANs, all-flash arrays, or other storage software StorPool is faster, more reliable and scalable.

Rev. 20180131