Hyperconverged Cloud Architecture with OpenNebula and StorPool

Version 3.0 – May 2020

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 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

ACL	Access Control List
AD	Active Directory
DC	Datacenter
ONE	OpenNebula
QCOW	QEMU Copy on Write image forma
SAN	Storage Area Network
vCPU	Virtual CPU
VDC	Virtual Datacenters
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.** OpenNebula enables businesses to embrace private, hybrid and edge cloud computing and meet the constant needs of developers for new development tools and frameworks, like containers or Infrastructure as a Code, while ensuring enterprise requirements for DevOps practices. It combines existing virtualization and containerization technologies with advanced features for multi-tenancy, automatic provision and elasticity.

OpenNebula is a single fully **open source product** with a healthy and active community that is commercially supported by OpenNebula Systems. The development of OpenNebula follows a bottom-up approach driven by the real need of sysadmins, devops, developers and users. 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. What is StorPool?

StorPool is a **software-defined storage solution** for building high-performance public and private clouds.[2] It runs on standard servers, drives and network and turns them into an outstanding storage system. StorPool is block-storage software that turns standard servers into a shared storage pool. Compared to traditional SANs, all-flash arrays, and other SDS products, StorPool is faster, more reliable and more scalable due to its distributed architecture. StorPool is designed from the ground up to provide cloud builders with the fastest and most resource-efficient storage software in the market. It is tailored for companies building public or private clouds. It provides you with a fast and reliable shared storage for your cloud and while reducing significantly the total costs of building and operating a cloud.

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 blocks.

[1] http://opennebula.io/discover/
[2] https://storpool.com/get-started/

- **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 a very large CPU, RAM and storage size and performance.

4. High Level Reference Architecture

A cloud architecture is defined by components in four planes: cloud orchestration, computing, storage and networking. Figure 1 shows an overview of the cloud. OpenNebula services run in a cloud orchestration plane and communicate 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.

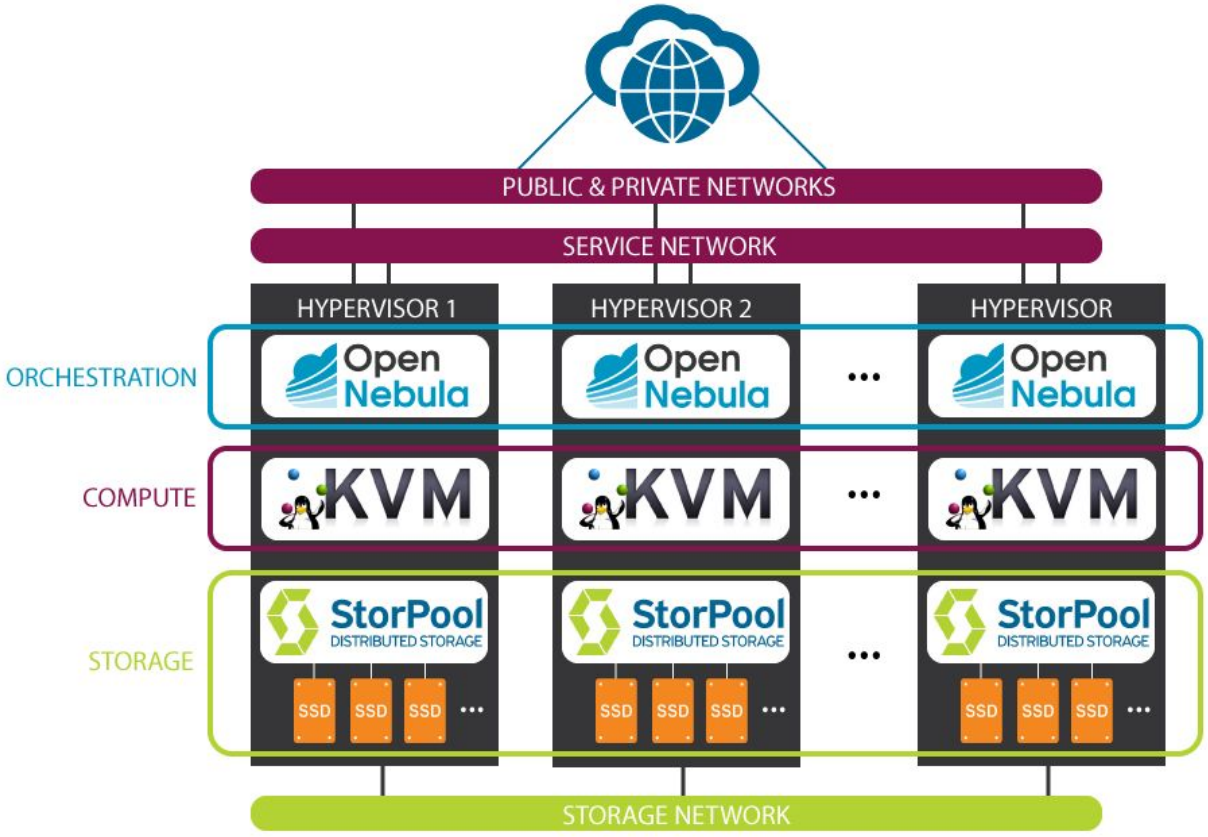

Figure 1. Hyperconverged Architecture, a bird's eye view.

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 three types of networks, that is, instance, service and storage networks.

The cloud can provide virtual machines access to dedicated private virtual networks, allowing closed isolated communication only between a set of virtual machines, or to a shared public virtual network. Access to each virtual network can be restricted to different users or groups or applied to 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 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

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

6. Hyperconverged Nodes

In an hyperconverged cloud architecture, each node implements all cloud functions: Virtualization, Storage and Networking. This provides a 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 homogeneous 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 comes approximately with 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 oversubscription, 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 oversubscription, 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 four network interfaces per node. Two interfaces used for the storage functions and two additional interfaces used for the 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.

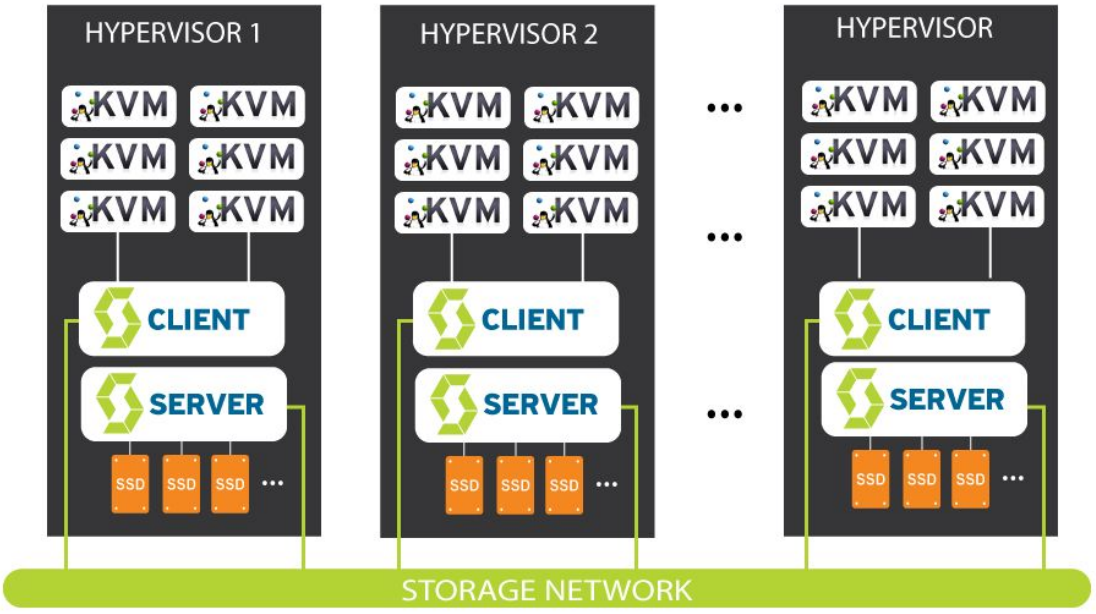

Figure 2. Hyperconverged Architecture, Storage.

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. StorPool scales linearly in capacity and performance. The bigger the cluster grows, the higher is the performance and the throughput it can provide.

StorPool software that is installed on each node (host) consists of two parts: a storage server and a storage client/driver. All storage servers and clients are connected over a dedicated redundant 10G/25G/40G storage network for minimum latency and maximum throughput.

StorPool servers on all nodes collaborate to create a distributed service, where each StorPool server stores part of the user data. Redundancy is guaranteed through a synchronous replication algorithm. This can be thought of as a very advanced software RAID between servers. By combining all available drives of all servers in a single storage pool, the user is not limited by the capacity or performance of a single disk. Every volume (virtual disk or OpenNebula image) is distributed on multiple storage nodes and multiple disks, ensuring resilience against server, disk or network failures.

The StorPool client communicates in parallel with all StorPool servers to achieve the aggregate throughput and IOPS of all installed disks in the cluster. Volumes defined in the storage system are presented by the StorPool client as standard block devices to the operating system.

With the OpenNebula integration, every virtual disk or image corresponds to a volume in StorPool that is presented as a block device in the host where it is used.

Hence virtual machine images are stored directly in the StorPool volumes using "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 that automates all tasks in the storage system.

Deep integration between OpenNebula and StorPool allows efficient use of the features of the storage system. For example when a new virtual machine is instantiated from a template, the virtual disk is created as a clone of the master image. Being a copy-on-write system, StorPool creates clones without consuming additional space. This can result in significant savings in large clusters with many identical virtual machines.

7. Networking

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

Isolation between different virtual networks can be achieved using 802.1Q VLAN tagging or VXLANs. OpenNebula supports two technologies to implement virtual networks in Linux hypervisors: Linux bridge and Open vSwitch.

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.

Table 3. Hyperconverged Architecture, Networking.

8. 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).

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 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 3).

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.

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.

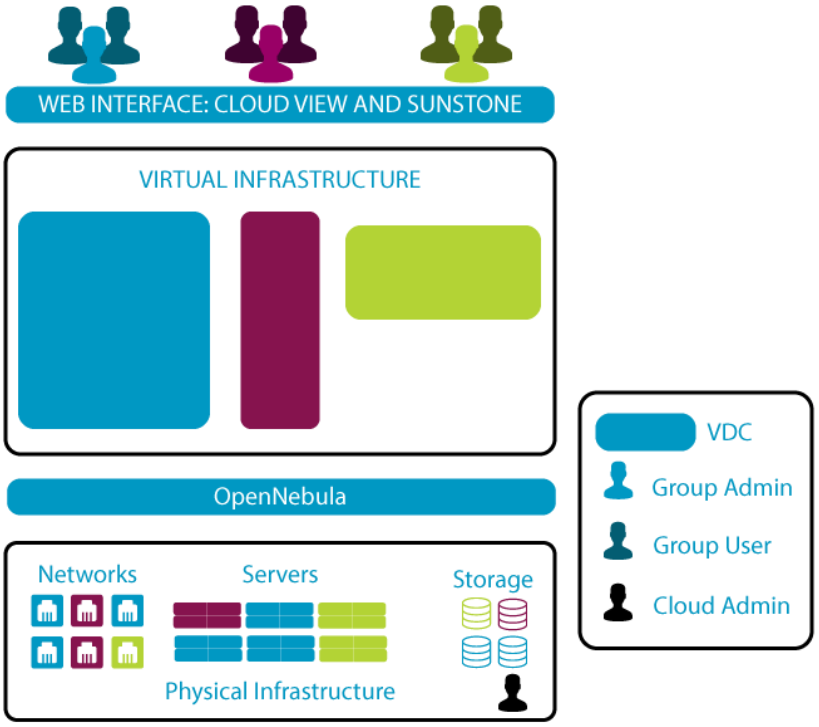

Figure 3. Resource Provisioning Model in OpenNebula.

The Cloud will therefore have three different types of users:

- Cloud Admins: Role reserved to the company's IT staff with 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.

10. 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 4).

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.

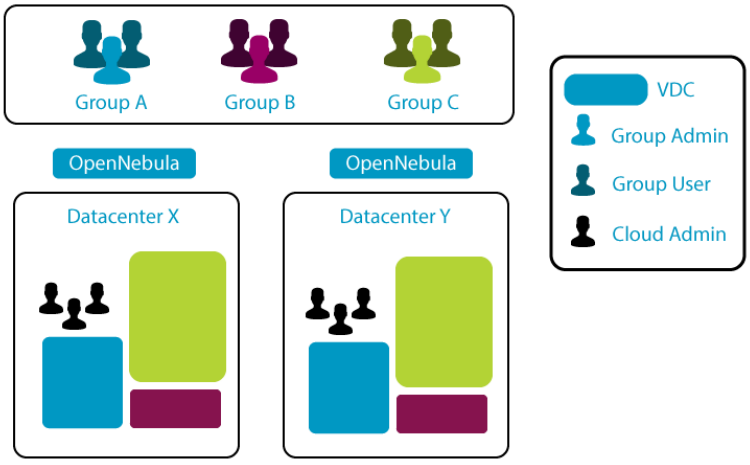

Figure 4. Federation Architecture.

11. Cloud Bursting

The architecture for hybrid clouds is described in Figure 5. 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.

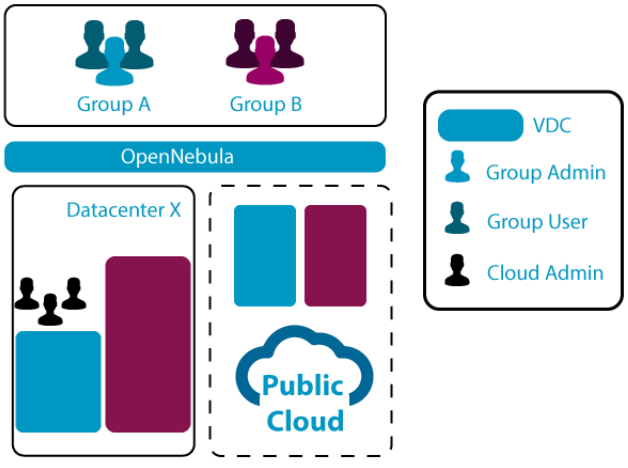

Figure 5. Hybrid Cloud architecture enabling cloud bursting.

12. High Availability

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

Cloud Orchestration

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 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 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 (e.g. 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.

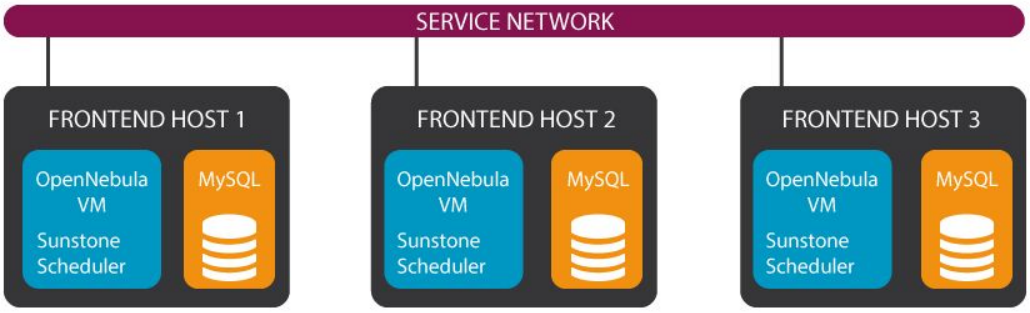

Figure 6. Overview of the OpenNebula HA architecture and main components.

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 an 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 three) 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 server, 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 Hyperconverged 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 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

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.** OpenNebula enables businesses to embrace private, hybrid and edge cloud computing and meet the constant needs of developers for new development tools and frameworks, like containers or Infrastructure as a Code, while ensuring enterprise requirements for DevOps practices. It combines existing virtualization and containerization technologies with advanced features for multi-tenancy, automatic provision and elasticity.

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, and other SDS products, StorPool is faster, more reliable and more scalable due to its distributed architecture

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

Rev3.0_20200528