Giuseppe Paternò

OpenStack Explained

Learn OpenStack architecture and the secret of a successful cloud project

An initiative to bring water to African children.

OpenStack

Table of Contents

About Giuseppe Paternò

Giuseppe Paternò is an **OpenStack consultant.** Through his extensive 20+ years experience in IT, he helps customers to **embrace cloud easily, securely and risk-free**.
Most European telecommunications, government and finance organizations are amongst his customers.

Giuseppe has a strong background in the Open Source world, where he played an important role for Canonical (the company behind **Ubuntu**), **RedHat**, **Sun Microsystems,** and **IBM**.

He also has an extensive knowledge in information security: this is reason behind his role as Visiting Professor and Researcher at the University of Trinity College Dublin in 2009.

Web Site: http://www.gpaterno.com
Twitter: @gpaterno
Facebook Page: https://www.facebook.com/gpaterno.pro
E-mail: gpaterno@gpaterno.com

Acknowledgments

A big thank you goes to my wife Maria, without her support I wouldn't even being able to achieve what I did up to now. A thank you also goes to Roberto Argentini from Telecom Italia, without him this paper would not even exist.
Thanks to the reviewers Guido Bolognesi, Sab Knight, Francesco Vollero, Nicolas Barcet and the designers Leonardo and Fausto Nenci.

Testimonials

"Giuseppe is paving the way for enterprises to embrace OpenStack. Telecom Italia is, nonetheless, among these enterprises. Should you be willing to embrace the cloud, I would strongly recommend this book. I am very pleased Giuseppe is going to offer his OpenStack expertise to raise money for the disadvantaged children in Africa."

Gianluca Pancaccini, *CIO of Telecom Italia*

"Giuseppe has done a great job of creating an important source of information on OpenStack technology while also raising money for disadvantaged children.
I would encourage anyone involved in the IT industry to download this ebook and in so doing support this great initiative."

Jeff Cotten, *Managing Director, Rackspace International*

"The OpenStack Project is one of the great examples for the dynamic of open source solutions and makes cloud technology available to organizations regardless of their size and location. Being one of the major contributors for the project, we at SUSE appreciate Giuseppe Paternó clear and concise explanation of OpenStack and it's architecture. For those looking for an introduction to OpenStack, this will be a valuable resource. We also highly appreciate Giuseppe's charitable support to provide clean drinking water for Children in Africa."

Ralf Flaxa, *Vice President of Engineering, SUSE*

1

Cloud: the undiscovered country

OpenStack

I've been sitting in front of many European IT Managers and CTOs and when they want to hear from me about OpenStack or Cloud, most of the times they mean something different: customers want a VMware replacement for virtualization. The most audacious ones are willing to have a nice web interface to access their virtual machines and that's it.

Cloud sounds like yet another marketing buzzword, it can mean just about anything or nothing at all. We are not discussing here what is the reason of walking away from VMWare, but the idea of the equation "Cloud=Virtualization" is pretty spread across all the customers. This is actually what some vendors tried to let you think of cloud.

While Cloud implies a virtualized environment, virtualization is not a cloud. Let me define Cloud using the NIST definition: "*Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (ex: networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort*".

The NIST definition goes on to define the essential characteristics of clouds (i.e., on-demand, network access, multitenancy, elasticity, and metering). It continues by defining three service models: **software as a service** (SaaS), **platform as a service** (PaaS), and **infrastructure as a service** (IaaS). It also identifies four main deployment models: private cloud, public cloud, hybrid cloud, and community cloud.

IaaS is the infrastructure layer that orchestrates the work typically done by system administrators to host the applications, including server provisioning, network management, and storage allocation.

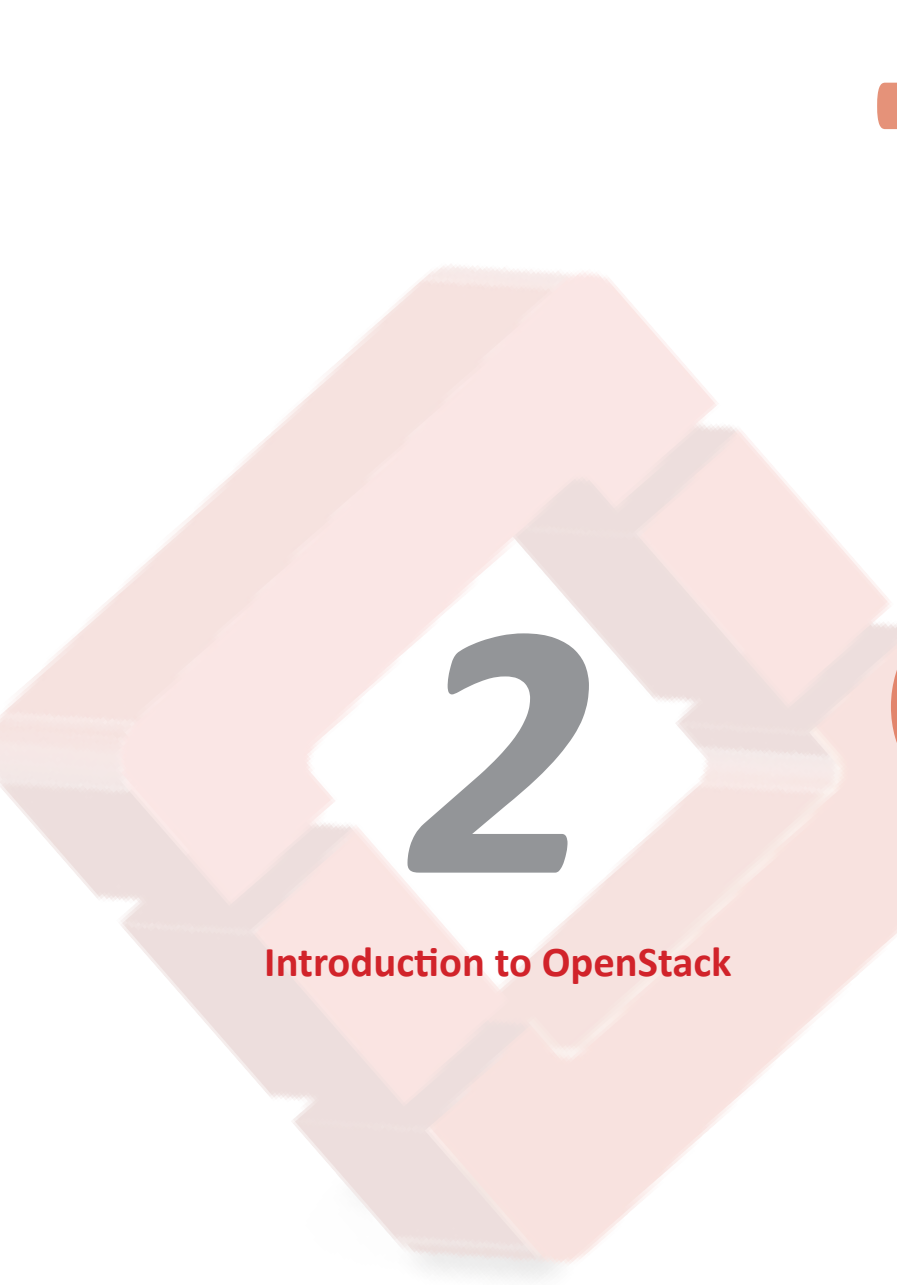
PaaS represents what we used to call middleware, and makes the link between the end-user application and the underlying infrastructure that it is running on. A PaaS solution is aimed at developers who do not want to worry about the infrastructure. PaaS is still a growing area and interesting players such as Openshift, CloudFoundry (and derivatives) and Cloudify.

SaaS refers to online application hosting: users will access the application interface over the Internet, all the work that happens in the background to make the application available and scalable is hidden from the end user. Gmail (and most Google services, including Calendar and Docs), SalesForce and SecurePass are typical SaaS examples.

There are several open source solutions which can be used to build our own IaaS Cloud. The three major projects in my personal order of importance are: **OpenStack**, **CloudStack** and **OpenNebula**. Successful private and public clouds are currently operational all over the world using these solutions.

Cloud is a huge shift in your organization and will change forever your way of working in the IT projects, improving your IT dramatically and cutting down costs. If you follow me, I will reveal to you the "secret ingredient" of a successful cloud project and you will be able to achieve incredible results.
If you feel you're not ready for it or if all you want is to find an alternative virtualization solution, I can give you some names such as RedHat oVirt, Citrix XenServer or Google's Ganeti.

But if you are willing to embrace the change, understand what actually are the impacts of OpenStack in your organization or just curious about what OpenStack is, read on. We will go into details of OpenStack, the most successful Open Source project after Linux itself, we will understand the strength and how it accomplished on-demand network access, multitenancy, elasticity, and metering.



2

Introduction to OpenStack

OpenStack

The great advantage of OpenStack is that the **end customer can choose** whether to use the reference implementations for each project or a vendor-specific implementation for each one. **The promise of OpenStack is the interoperability amongst different components** from different vendors or open source projects, giving the customer the choice to find out what is the best solution for their own needs.

OpenStack could bring the following benefits to you:

- Self-service Instance life cycle management: run, reboot, suspend, resize and terminate instances.
- Management of compute resources: CPU, memory, disk, and network interfaces.
- Management of Local Area Networks (Flat, Flat DHCP, VLAN DHCP and IPv6) through programmatically allocated IPs and VLANs.
- API with rate limiting and authentication to manage who has access to compute resources and prevent users from impacting each other with excessive API utilization.
- Distributed and asynchronous architecture for massively scalable and highly available system.
- Virtual Machine (VM) image management i.e. store, import, share, and query images.
- Floating IP addresses i.e. Ability to assign (and re-assign) IP addresses to VMs.
- Security Groups i.e. flexibility to assign and control access to VM instances by creating separation between resource pools.
- Role Based Access Control (RBAC) to ensure security by user, role and project.
- Projects & Quotas i.e. ability to allocate, track and limit resource utilization.
- REST-based API which is key to automation, a key characteristic of clouds, and which is very simple to implement using any computer language.
- Class of services: each type of resource can be available with different features, eventually linked to pricing, such as reliability, performance or resiliency
- Measurability of every action and consuption of resource s to allow audability and usage reporting.


3
OpenStack components
OpenStack

OpenStack is a collection of open source technologies delivering a massively scalable cloud operating system.

OpenStack cloud operating system controls large pools of compute, storage and networking resources throughout a datacenter, all managed through a dashboard that gives administrators control while empowering their users to provision resources through a web interface.

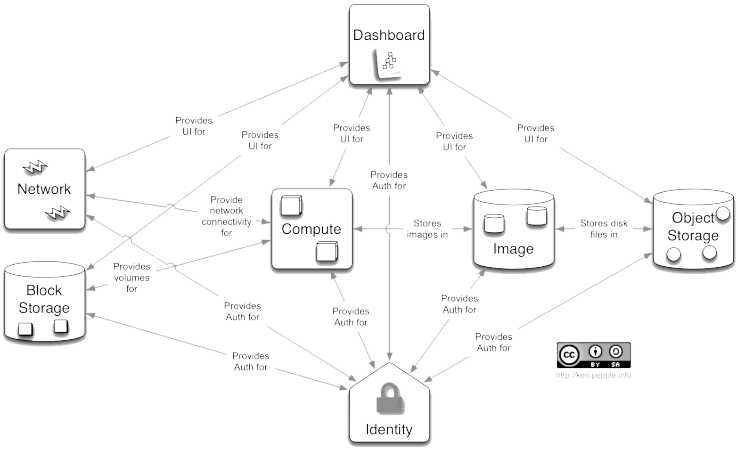
We can think of it as software to power our own Infrastructure as a Service (IaaS) offering, like the one behind Amazon Web Services.

OpenStack is an umbrella project that can be divided into many sub-components:

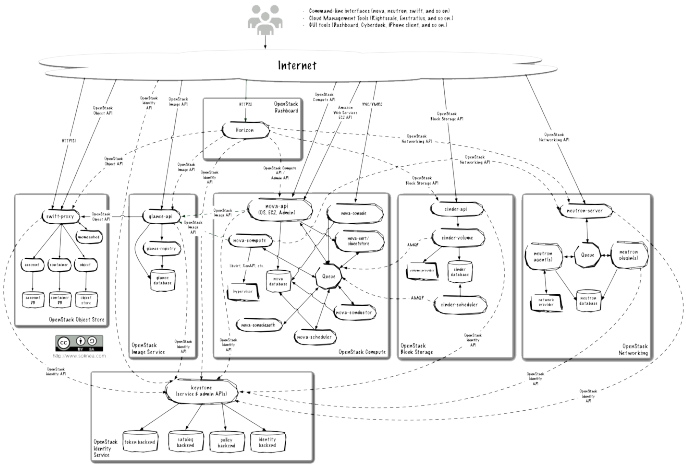
Project	Code Name
Dashboard	**Horizon**
Compute	**Nova**
Identity	**Keystone**
Network	**Neutron**
Image Service	**Glance**
Block Storage	**Cinder**
Object Storage	**Swift**
Telemetry	**Ceilometer**
Orchestration	**Heat**
Database	**Trove**
DNS service	**Designate**
Bare Metal	**Ironic**
Queue Service	**Zaqar**

3.1 - OpenStack logical architecture

There are currently seven core components of OpenStack, how they conceptually interact with each other is shown below:

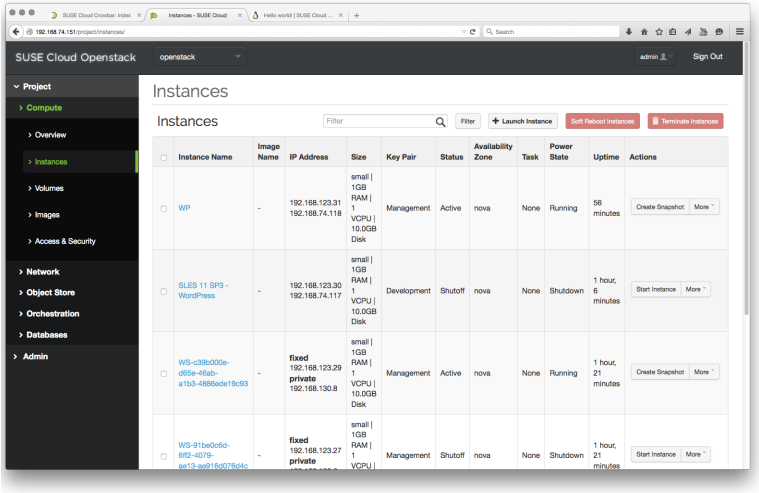


All these components and how they relate each other are shown in the simplest way in the below OpenStack logical architecture:



3.2 - Horizon – Dashboard

Horizon provides a modular web-based user interface for all the OpenStack services. With this web GUI, you can perform most operations on your cloud like launching an instance, assigning IP addresses and setting access controls.

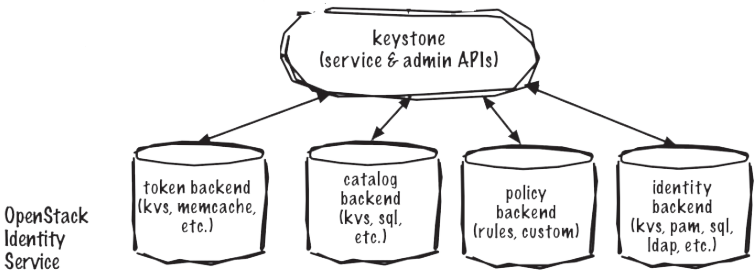


3.3 - Keystone – Identity

Keystone is a central component for **authentication and authorization** for all the OpenStack services. Keystone handles API requests and provides configurable catalog, policy, token and identity services.

Keystone also implements the ability to add users to groups (also known as tenants or projects) and to manage permissions between users and groups. Permissions include the ability to launch and terminate instances.

Keystone can be integrated with LDAP and other external authentication providers as well as be part of an authentication federation using SAML or OpenID with external identity providers, ex: SecurePass (www.seure-pass.net).

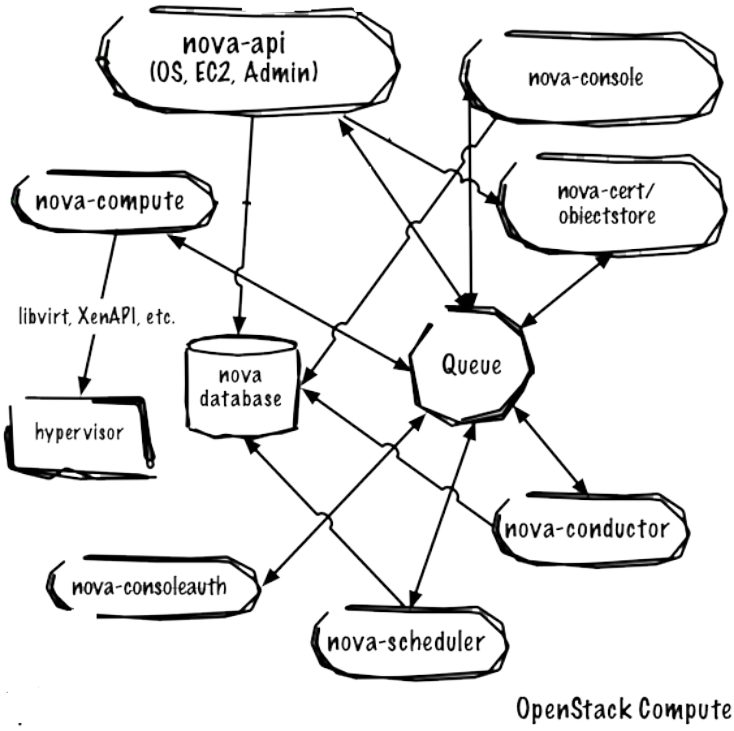


3.4 - Nova – Compute

Probably the most known among the projects, it provides virtual servers upon demand. Nova is the most complicated and distributed component of OpenStack. A large number of processes cooperate to turn end user API requests into running virtual machines.

These are the Nova components and their functions:

- ***nova-api*:** a REST API web service which accepts incoming commands to interact with the OpenStack cloud
- ***nova-compute*:** a worker daemon which creates and terminates virtual machine instances via specific Hypervisor's APIs. There are many different Hypervisors implemented as plugins in OpenStack.
- ***nova-scheduler:*** takes a request from the queue and determines which compute server host it should run on
- ***nova-conductor:*** provides services for nova-compute, such as completing database updates and handling long-running tasks
- ***nova database:*** stores most of the build-times and run-time states for a cloud infrastructure
- The ***message queue*** provides a central hub for passing messages between daemons. This is usually implemented using a pluggable architecture based on the Oslo project, but is most commonly with RabbitMQ
- Nova also provides console services to allow end users to access their virtual instances console through a proxy. This involves several daemons (***nova-console, nova-novncproxy*** *and* ***nova-consoleauth***)
- ***nova-network:*** a worker daemon very similar to *nova-compute*. It accepts networking tasks from the queue and then performs tasks to manipulate the network (such as setting up bridging interfaces or changing iptables rules). This functionality is being migrated to **Neutron**, a separate OpenStack service
- ***nova-volume:*** Manages creation, attaching and detaching of persistent volumes to compute instances. This functionality has been migrated to Cinder, a separate OpenStack service.



Nova also interacts with many other OpenStack services: **Keystone** for authentication, **Glance** for images and **Horizon** for the web interface. The Glance interactions are central to OpenStack. The API process can upload and query Glance while nova-compute will download images for launching images.

Historically, most OpenStack development is done with the most community supported hypervisor: KVM. This allows you to refer to Internet forums to find help on your issues. All the features that are currently supported in KVM are also supported in QEMU.

Microsoft Hyper-V and VMware ESXi too are gaining much support, with Hyper-V now being available with a free license. ESXi can also be used with a free license however API support is limited to READ ONLY without vCenter or an Enterprise license. Nova has support for **XenServer** and XCP through the XenAPI virt layer. Note that this does not imply support for other Xen-based platforms such as those shipped with RHEL 6 or SUSE, which is provided via the libvirt layer (i.e. Xen via libvirt).

Nova also supports bare metal provisioning through the Ironic project, that means it is possible to deploy to hardware in the same way the end user deploys virtual machines. By default, it will use PXE and IPMI in concert to provision and turn on/off machines, but Ironic also supports vendor-specific plugins which may implement additional functionality. Some vendors, most notably HP Helion, use Ironic to deploy OpenStack itself.

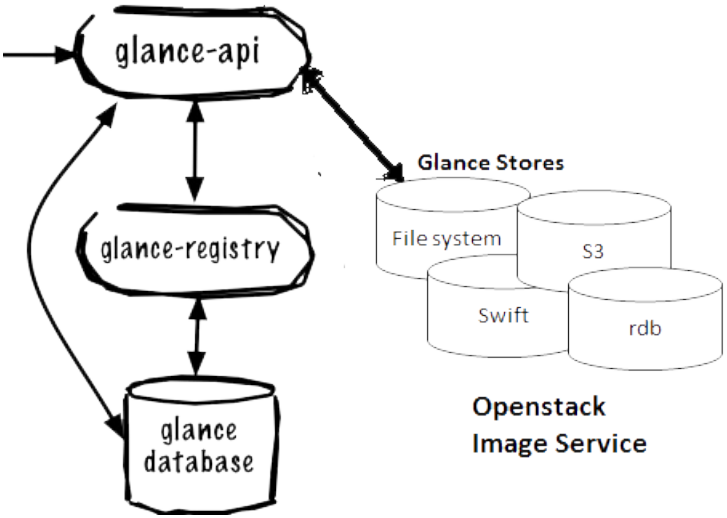
3.5 - Glance – Image Store

It provides discovery, registration and delivery services for disk and server images.

List of components and their functions:

- ***glance-api:*** accepts Image API calls for image discovery, image retrieval and image storage
- ***glance-registry:*** stores, processes and retrieves metadata about images (size, type, etc.)
- ***glance database:*** a database to store the image metadata
- A ***storage repository*** for the actual image files. Glance supports normal filesystems, Ceph block devices, Amazon S3, HTTP and Swift.

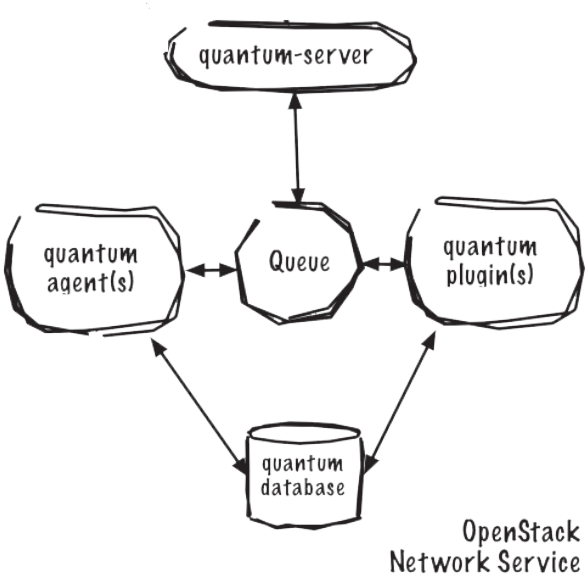
Glance accepts API requests for images (or image metadata) from end users or Nova components and can store its disk files in the object storage service, Swift or other storage repository.



3.6 - Neutron – Network

Neutron provides "*network connectivity as a service*" between network interface devices, routers and more (e.g., vNICs) managed by other OpenStack services (e.g., Nova). The service works by allowing users to create their own networks and then attach interfaces to them. Neutron has a pluggable architecture to support many popular software defined networking (SDN) vendors and other central network control plane technologies.

- ***neutron-server*** accept API requests and routes them to the correct neutron plugin
- ***plugins and agents*** perform actual actions, like plug/unplug ports, creating networks, subnets and IP addressing
- it also uses the same Oslo messaging as Nova and other OpenStack projects to route info between *neutron-server* and various agents
- it has a ***neutron database*** to store networking state for particular plugins

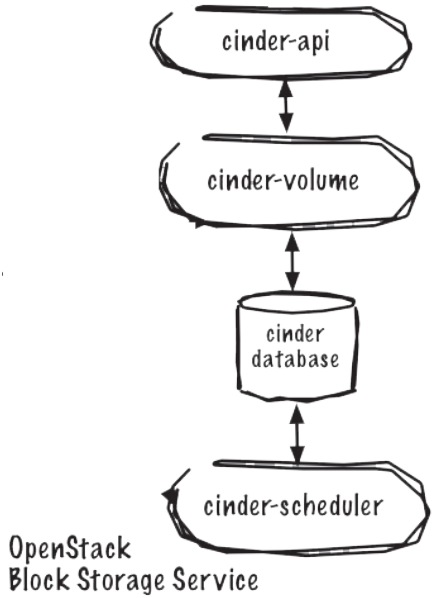


Neutron will interact mainly with Nova, where it will provide networks and connectivity for its instances.

3.7 - Cinder – Block Storage

Cinder allows block devices to be exposed and connected to compute instances for expanded storage & better performance.

- ***cinder-api*** accepts requests and routes them to cinder-volumes for action
- ***cinder-volume*** reports reading or writing to the cinder database to maintain state, interacts with other processes (like cinder-scheduler, see below) through a message queue and directly on block storage providing hardware or software
- ***cinder-scheduler*** picks the optimal block storage node to create the volume on
- the ***messages queue*** routes information between Cinder processes
- a ***cinder database*** stores volumes state



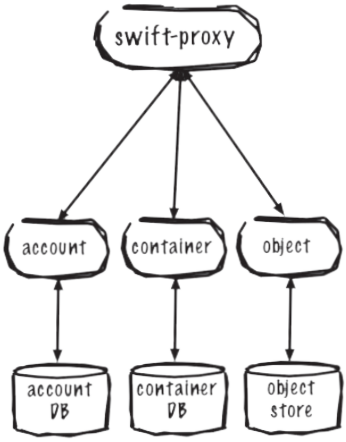
Like Neutron, Cinder will mainly interact with Nova, providing volumes for its instances.

3.8 - Swift – Object Storage

Object store allows you to store or retrieve files as individual unitx. It provides a fully distributed, API-accessible storage platform that can be directly integrated into applications or **used for backup, archiving** and data retention.

Object Storage is not a traditional file system, but rather a distributed storage system for static data such as virtual machine images, or photos, e-mails, backups and archives. Unlike filesystems, there is no hierarchy of files nor is it possible to "open" a file and modify it online. If you want to modify a file, you have to pull it, modify and push it back. This may seem a huge restriction but it allows for much improved scalability (by multiple orders of magnitudes) than traditional shared file systems, ex: in multi-datacenter deployments.

- Proxy server (***swift-proxy-server***) accepts incoming requests, like files to upload, modifications to metadata or container creation; it also serves files and provides container listing
- ***Accounts server*** manage accounts defined within the object storage service.
- ***Container servers*** manage mapping of containers, folders, within the object store service.
- ***Object servers*** manage actual objects, files, on the storage nodes.



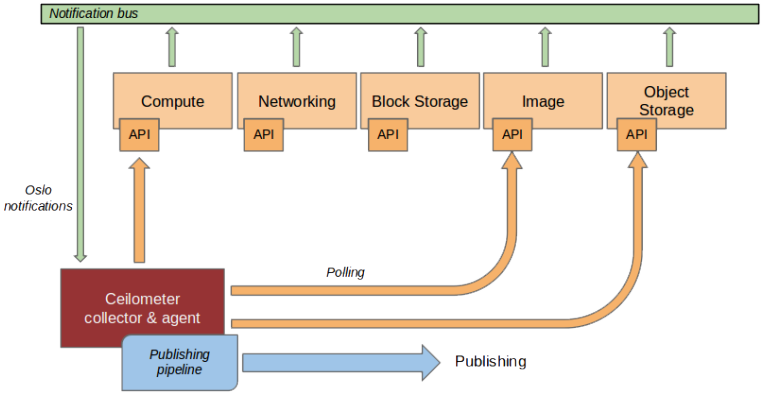
Also replication services run to provide consistency and availability across the cluster, audit and update.

OpenStack Object Store

3.9 - Ceilometer - Telemetry

The required steps to bill for usage in a cloud environment are **metering, rating and billing**. Because the provider's requirements may be far too specific for a shared solution, rating and billing solutions cannot be designed as a common module that satisfies all possible scenarios. Providing users with measurements on cloud services is required to meet the "measured service" definition of cloud computing.

The Telemetry module was originally designed to support billing systems for OpenStack cloud resources. This project only covers the metering portion of the required processing for billing. The module **collects information about the system usage** and stores it in the form of samples in order to provide data about anything that can be billed. It does so by both listening to event on the message queue and by polling information out of the various components.

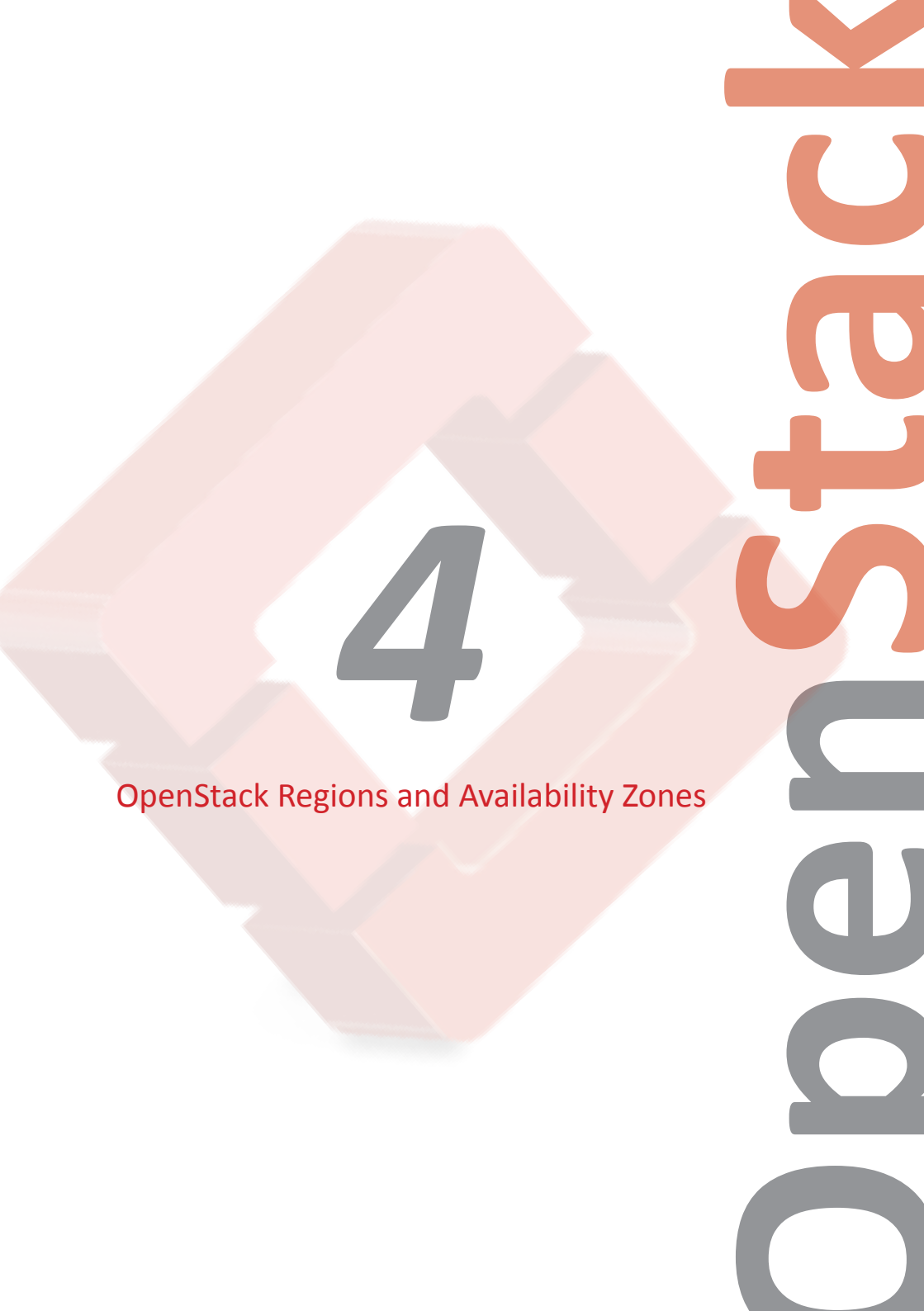


The list of metrics is continuously growing, which makes it possible to use the data collected by Telemetry for many more purposes other than billing. For example Heat can autoscale resources when Ceilometers triggers an alarm, for example adding more front-end web servers when CPU utilization is more than 70% for 5 minutes.

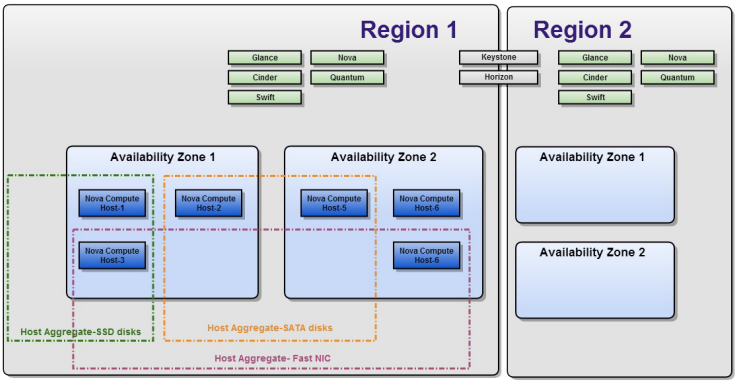
3.10 - Other projects

Although the former ones are the most relevant, there are three other projects worth mentioning:

- **Trove** is a database-as-a-service provisioning relational and non-relational database engines. It allows an agnostic access to databases, currently supports MySQL and PostgreSQL, but vendors like Oracle and Microsoft might provide a Trove plugin for their databases in future.
- **Ironic** (Bare Metal Provisioning), is an incubated OpenStack project that aims to provision bare metal machines instead of virtual machines. Ironic is currently in use by HP Helion.
- **Zaqar** (Multiple Tenant Cloud Messaging), is a multi-tenant cloud messaging service for Web developers. Zaqar was formerly known as Marconi.
- **Designate** provides a DNS as a Service for OpenStack.


4
OpenStack Regions and Availability Zones
OpenStack

OpenStack was made from the ground up to scale to thousands of nodes and span different datacenters and geographical regions. For this reason, Openstack clouds can be divided in three main hierarchical zones:
Regions, **Availability Zones** and **Host Aggregates**.



Region
Each Region has its own full Openstack deployment, including its own API endpoints, networks and compute resources. Different Regions share one set of Keystone and Horizon services, to provide access control and a Web interface.

Availability Zone
Inside a Region, compute nodes can be logically grouped into Availability Zones (AZ): when launching a new VM instance we can specify the AZ we want it instantiated in, or even a specific node inside an AZ to run the VM instance.

Host Aggregates
Besides AZs, compute nodes can also be logically grouped into Host Aggregates.
Host Aggregates have meta-data to tag groups of compute nodes, e.g. all nodes with an SSD disk can belong to one Host Aggregate, while another Host Aggregate may contain all nodes with 10 GB NICs.

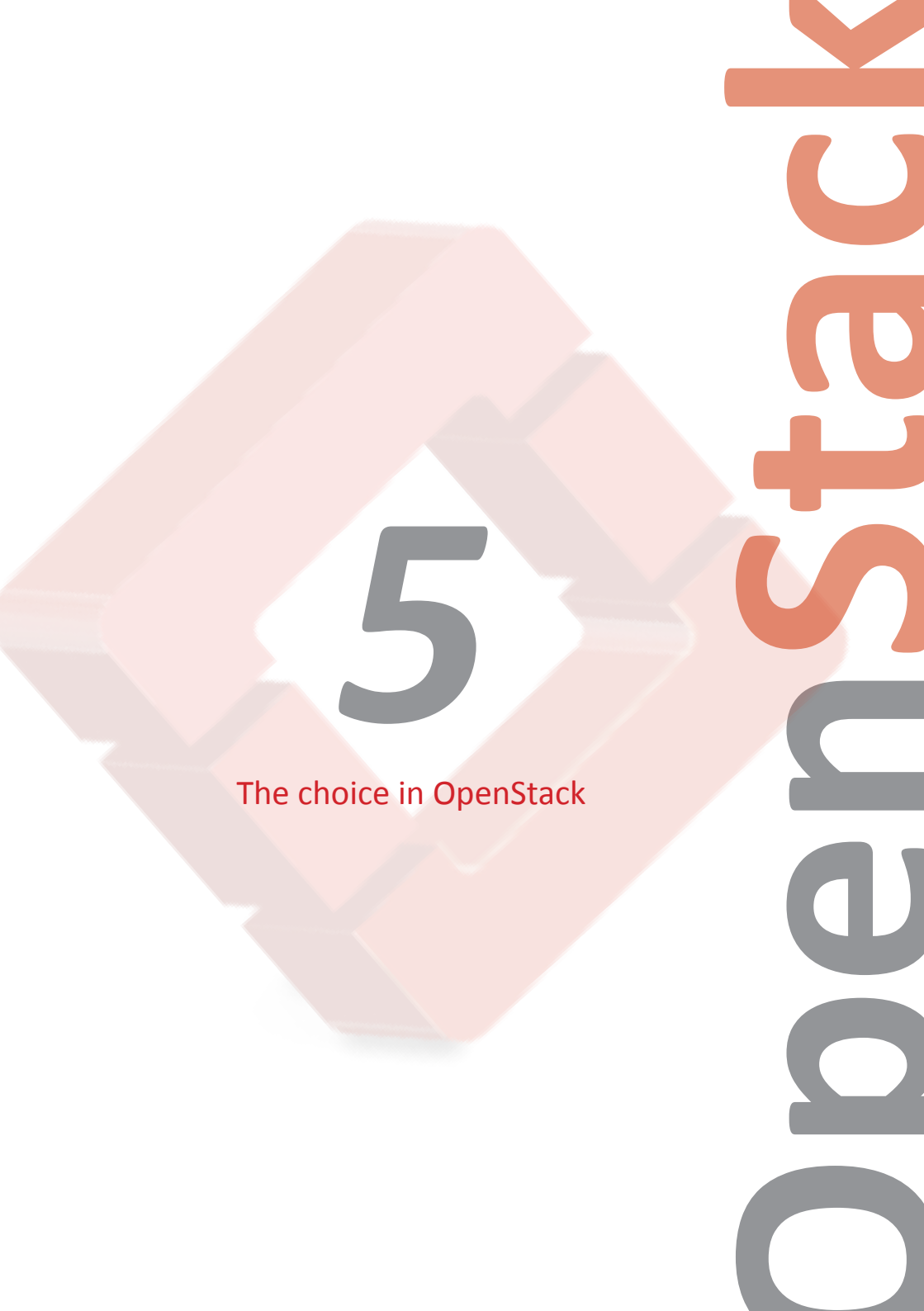
One compute node can be put into both an Host Aggregate and an Availability Zone at the same time, as they do not conflict. Moreover, one compute node can belong to more than one Host Aggregate. Host Aggregates are visible only to the admin and can also be used to mix hypervisors in the same AZ, for example to save license costs: some vendors provide free guests for their hypervisors.

Cells

OpenStack Compute cells allow you to run the cloud in a distributed fashion. Hosts in a cloud are partitioned into groups called *cells*. Cells are configured in a tree. The top-level cell ("API cell") has a host that runs the nova-api service, but no nova-compute services.

This allows for a single API server being used to control access to multiple cloud installations. Introducing a second level of scheduling (the cell selection), in addition to the regular nova-scheduler selection of hosts, provides greater flexibility to control where virtual machines are run.

Unlike having a single API endpoint, regions have a separate API endpoint per installation, allowing for a more discrete separation. Users wanting to run instances across sites have to explicitly select a region. However, the additional complexity of running a new service is not required.


5
The choice in OpenStack
OpenStack

Earlier in this publication I mentioned that the promise of OpenStack is the interoperability among different components from different vendors or open source projects. As a matter of fact, **each of the components** described in the previous page **can be easily replaced** with projects or products from each vendor.

At the time of writing, the only project that has no valuable alternative among vendors is Keystone. Keystone acts as a service registry and user repository, therefore plays an important role in OpenStack. While it was conceived to have internal users like Amazon does, the development is shifting towards an HTTP interface to existing identity systems, such as LDAP or SAML.

Also Horizon, the web dashboard, has few chances to be replaced, as its colors and logos can easily be customized to be adapted for everyone. Some other dashboards exist for OpenStack, but usually the company who needs a different web interface goes for a customized development on top of the OpenStack APIs.

Projects in which it makes sense to adopt **a plugin approach** are **Nova, Neutron, Swift and Cinder**. Let us review in a table what are the most relevant open source and proprietary technology for each component (please keep in mind that this list can vary).

5.1 - Nova

Open Source	Proprietary
KVM	VMWare ESX/ESXi
XenServer	Microsoft Hyper-V
LXC	
Docker	

5.2 - Cinder

Open Source	Proprietary
LVM	NetApp
Ceph	IBM (Storwize family/SVC, XIV)
Gluster	Nexenta
NFS (any compatible)	SolidFire
	HP LeftHand/3PAR/MSA
	Dell EqualLogic/Storage Center
	EMC VNX/XtremIO

5.3 - Neutron

Open Source	Proprietary
Linux Bridge	VMWare NSX
Open vSwitch	Brocade
Midonet	Big Switch
OpenContrail (Juniper OpenSource)	Alcatel Nuage
	Cisco Nexus

5.4 - Swift

Open Source	Proprietary
Swift project	EMC Isilon OneFS
Ceph	NetApp E-Series
Gluster	Nexenta
Hadoop with SwiftFS/Sahara	

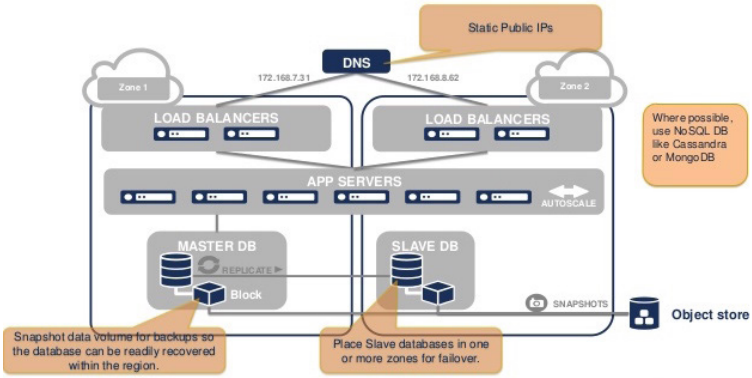

6
Applications in the cloud
OpenStack

So you have read about OpenStack and you are really eager to implement it. But let us step back and understand why you are willing to embrace the cloud. You might think of several reasons, but -- judging by my experience -- everything comes down to two root causes:

- You are looking to take advantage of the **fast provisioning of the infrastructure**, either for savings, speed, or both
- Your applications may have varying demand patterns, resulting in the **need of increased computing power during some periods**. You may want to take advantage of the scaling capabilities of Cloud to fire up new instances of key modules at peak periods, shutting them down when not needed, freeing up infrastructure resources for other tasks and reducing the TCO

Most of the customers just want a fast provisioning mechanism of the infrastructure. Do not get me wrong, this is perfectly fine and OpenStack gets the job done.

But you will get the full benefit of the cloud when you'll have an application that might be in need of resources on-demand. Think about a sports news portal when the World Cup is on, the invoicing and billing at the end of the month or a surge in the need to process a data from devices.



Wouldn't it not be nice, given the detected increasing loads, to have the **application scale automatically to cope with the requests**? Believe it or not, it is not magic and it is totally feasible. Netflix did it and I can name a lot of other SaaS systems that are doing it. There is only one constraint: you have to be in control of the source code of your applications. If you bought your application "as is", contact your vendor, but there are yet few chances that you can follow this pattern.

Being cloud aware for an application also means that it need to be implemented with self resiliency in mind.

In case you have the source code, you can adapt your application to take full advantage of your new environment. In this scenario, you will have to intervene more into your code as you will need to ensure that **the application can take full advantage of the environment**, reconfiguring load balancers, dynamically allocating resources and etc. There are some "tricks" that an application has to adopt to be "cloudish", but is outside of the scope of this publication.

It's quite common that a customer might decide to have a phased approach to the cloud, starting to take advantage of the fast provisioning and then transforming the application to adapt it to the cloud. The cloud is a long journey and it can be successful, are you ready for it?

7

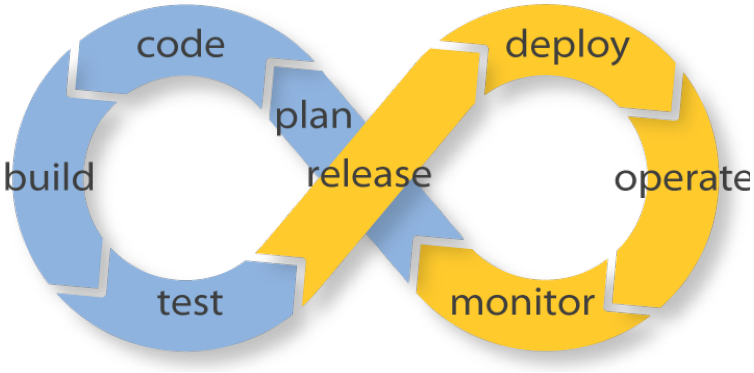
DevOps: to the infinity and beyond

OpenStack

So you understood OpenStack, its components and how applications play an important role in the cloud. Before revealing to you how to be successful with OpenStack, there is another important piece I want you to understand.

The next marketing buzzword that everybody mentions nowadays is DevOps. As you might understand, DevOps is an acronym that stands for “Development” and “Operations”.

The goal of DevOps is to improve service delivery agility, promoting communication, collaboration and integration between software developers and IT operations. Rather than seeing these two groups as silos who pass things along but do not really work together, **DevOps recognizes the interdependence of software development and IT operations**.



In an ideal world, through the use of continuous integration tools and automated tests, a group of developers could bring a new application on-line without any operations team. For example, Flickr developed a DevOps approach to support a business requirement of ten deployments per day. Just for your own information, this kind of approach is also referred as continuous deployment or continuous delivery.

Discussing development and agile methodologies is not within the scope of this publication, but this is one thing you have to understand and keep in mind, no matter if you are an IT manager, developer or system administrator.

If you decided to embrace the cloud in full and you are thinking of adapting your application to take advantage of it, then **every single aspect of IT have to be carefully analyzed**. Development, whether you do it internally or outsourced, must be taken into consideration. Also the way your company has been organized has to change: did I mention before **Cloud is a huge shift**?

8

The “secret ingredient” of a successful project

OpenStack

I promised you at the beginning that I would have revealed the "secret ingredient" to deploy a successful OpenStack project. Let me begin with two examples.

The first one belongs to a well-known European telecommunication provider. Like every other telco, they have a complex internal structure and when someone from the internal team proposed OpenStack as a possible solution, the upper management decided that it was not "enterprise enough" and that they had to stick to a certified stack that included, amongst others, VMware and Oracle. The time needed to deploy a single virtual machine was around 40 days, because of all the process it had go through.

A system engineer was receiving daily complaints from the developers, who in turn were under pressure from the marketing team to deploy new campaigns faster to the market. This person decided to **form an "unofficial team" of very skilled people**, stole some decommissioned old machines and created an OpenStack cluster in two regions**.** He made some **internal meetups to educate the developers** and the internal people on how to embrace the OpenStack philosophy and how an application can leverage the underlying platform to take advantage of elasticity and scalability.

The unofficial experiment had an unexpected success and most of the new applications were deployed on this platform. Although this bypassed all the internal rules and processes, the upper management could only take note of the status-quo and approve it officially. The system administrator is now the leader of this "SWAT group", that is composed of only 11 people and runs now 50% of the company internal applications.

The second story I want to share with you is about an American financial institution. Following a conversation with the CEO of a premier vendor, the CTO of this bank decided to embrace OpenStack and switch over from VMWare. He asked his management to go and execute the change. The office in charge of the IT architectures had to review all the processes in place and start talking with the departments of the company.

Like many big companies, **this enterprise is divided in several departments** (network, security, system administrators, middleware, …) and in the following months a large number of meetings went through all the departments. OpenStack was deployed in 8 months using the existing corporate policies and following the standards and best practices that were in place. Despite all the efforts, the processes of delivering a new application to internal customer went from 90 days to 75 days. As OpenStack itself was perfectly working, **the upper management did not understand what went wrong and could not justify the investment to the CTO**.

What is the moral of the story then? The reality is that OpenStack is just a technology and it enables you to do more if you embrace its philosophy. This requires a **company to change deeply in the way IT is conceived, and to become even more productive and agile**. If you are not willing to change your internal processes and department divisions, you will not enjoy the full benefits of OpenStack. Meanwhile, if you look at the other example, the telco was so successful because nobody believed on the project at the beginning, therefore the system administrators were free to bypass all the internal schemas. This approach, with a proper internal awareness program, made the project a great success.

Of course it is not all black and white, there are several shades of gray in between and not each company was created equal. I hope I gave you the tools to have a clean and vendor-neutral idea of what cloud is and what benefits it can bring you, but you will have to find your own recipe to be successful in deploying OpenStack.

9

Donation

OpenStack

Giuseppe Paternò will donate all the revenues of this publication to bring drinking water for school children in Benin: many children in Benin have no choice but to drink contaminated water and illness is just around the corner.

The donation will help the charity HELVETAS Swiss Intercooperation to build drinking water wells in schoolyards, and educating children about hygiene. With such project, the children stay healthy and are able to attend school. For more information visit Helvetas website www.helvetas.org