

# OpenStack: The Path to Cloud

Considerations and recommendations for  
businesses adopting cloud technology



# Table of Contents

Executive Overview	1
Enterprise Cloud Strategy	2
Approaches to an OpenStack Private Cloud	5
Forming the OpenStack Team	9
Organization and Process Considerations	13
Choosing Workloads for Your Cloud	16
Implementation Phases	22
Post-deployment	30
Summary	32
References	33
Glossary	34

\***Underlined gray bold** words and concepts are defined in the Glossary at the end.

## CONTRIBUTORS

Carol Barrett, *Cloud Software Planner*, Intel Corporation

Tyler Britten, *Technical Advocate*, Blue Box, an IBM company

Kathy Cacciatore, *Consulting Marketing Manager*, OpenStack Foundation

Pete Chadwick, *Senior Product Manager*, SUSE

Paula Phipps, *Senior Manager, Infrastructure Software Marketing*, Hitachi Data Systems

Gerd Prüßmann, *Director Cloud Solutions*, Mirantis

Megan Rossetti, *Cloud Infrastructure Operations*, Walmart

Yih Leong Sun, PhD, *Senior Software Cloud Architect*, Intel Corporation

Shamail Tahir, *Offering Manager*, IBM

Heidi Joy Tretheway, *Senior Marketing Manager*, OpenStack Foundation

Susan Wu, *Director of Technical Marketing*, Midokura

# Executive Overview

This book is written to help enterprise architects implement an OpenStack® cloud. With architects with one foot in information technology and the other in business operations in mind, we want to offer insights and best practices to help you achieve multiple (and sometimes competing) goals.

If you're looking for vendor-neutral answers about planning your path to an OpenStack cloud, you're in the right place.

Members of the OpenStack community—technologists, business leaders and product managers—collaborated on this book to explain how to get started with an OpenStack cloud. We've included pros and cons to help you make better choices when setting up your cloud, along with anticipated investments of both time and money.

In this book, we'll discuss the considerations involved and how to make OpenStack cloud decisions about models, forming your team, organization and process changes, choosing workloads, and implementation from proof-of-concept through ongoing maintenance. Topics include:

- Your technology options and their pros and cons.
- What to expect—in support, level of investment, and customization—from each type of cloud and consumption model.
- Operational models for a cloud, including staffing, plus how to manage consumption of cloud services in your business.
- How to assess the cloud's value to your business.

After reading this book, you'll understand the process of building an OpenStack cloud, various cloud models, and operational and application approaches. You'll understand what decisions to make before building your cloud, and their effects on cost, resources, and capabilities.

# Enterprise Cloud Strategy

There are several considerations when determining what cloud deployment model is best for your organization. You will need to determine your organization's regulatory compliance needs, financial resources, preferred spending model, users' geographic locations, and predictability of demand. Not all cloud models can equally address these requirements. Your criteria will provide an initial guide to determining if a private, public, hybrid, or community cloud is best for your organization.

*Figure 1: Cloud deployment models*



## PRIVATE

- A single organization can be multiple consumers
- Can be a single or multi-site deployment
- Can be on- or off-premises
- Can include multiple consumption models such as managed, distribution (distro), and build your own system (DIY)
- Offers subscription, buy, or build options



## PUBLIC

- Shares infrastructure, multi-tenant
- Is generally multi-site and off-premises
- Normally includes pay-as-you-go pricing



## HYBRID

- Can be single or multi-tenant
- Can include any combination of two (or more) distinct clouds bound together
- Is multi-site



## COMMUNITY

- A cloud that is provisioned for a specific community (e.g. research institutions) that wants to pool resources and facilitate collaboration
- Can be shared infrastructure, and be multi-tenant
- Can be on- or off-premises

The importance of the considerations will vary based on your organization's objectives and use cases for the cloud.

Before we begin to discuss the considerations, let's briefly review various cloud deployment models. For more information about these cloud deployment models, check out the *National Institute of Standards and Technology (NIST) definition of cloud computing*<sup>1</sup>.

## Regulatory and compliance requirements

Your regulatory compliance needs might dictate your choice of a cloud deployment model. Not all models meet the needs of organizations with industry-specific regulations, such as PCI, HIPAA, data sovereignty, etc. Private, hybrid, and community models offer the most flexibility for implementing the necessary controls to meet your compliance needs. Some public cloud providers have specialized cloud regions for industry-specific compliance needs.

## Initial investment amount

Financial resources can influence your decision about which cloud deployment model is right for your organization. Generally, private and hybrid clouds have higher initial costs than public or community clouds. Public clouds use shared infrastructure, which removes much of the startup costs and typically offers “pay-as-you-go” pricing. Similarly, community clouds have lower entry costs than private or hybrid clouds because the cost is divided among multiple organizations. Private and hybrid clouds are generally offered by a single organization, which require you to purchase or lease all components. Some providers offer subscription pricing for private or hybrid clouds depending on the consumption model [[See the Approaches to an OpenStack Private Cloud chapter](#)].

## Geographic access patterns

Your cloud services should be available to your consumers whether they are internal or external to your organization. However, the number of locations and types of consumers can impact your cloud deployment model selection. For example, if you don't have data centers in all the locations near your consumers, you might leverage the hybrid cloud model to have private cloud(s) in your own facilities augmented by public cloud regions in other areas.

If you have regulatory or latency/performance constraints that prohibit using a regional private cloud across the globe, then a public cloud could allow you to maintain geographic proximity at a lower overall cost. For example, a company that has presence in Australia but lacks a local data center in the

region might be required to store data inside the country. The company could leverage a public cloud from Australia as a “virtual data center” at a lower cost than building out a data center for the company itself. Sometimes content and workloads that are latency-sensitive can be impacted by long transmission times that affect delivery of the output. For example, you want the lowest latency you can afford for transmitting video to users.

## Elasticity and demand

In some business models or use cases, usage is predictable. In others, demand can be very uncertain. Cloud computing offers better elasticity than traditional infrastructure models. However, certain cloud deployment models require more attention to capacity management to ensure resources are available as service demands scale. Public clouds generally offer the most elasticity and can handle scenarios where demand is completely uncertain or unpredictable. Hybrid clouds offer a balance between the control offered by private cloud with the elasticity of a public cloud, because new workloads can be started on the public cloud as needed to meet temporary increases in demand. More than 150 OpenStack public cloud data centers are available from several global *public cloud providers*<sup>2</sup>.

## CapEx, OpEx, and contract commitments

Private, hybrid, and community cloud models can be procured using either capital (CapEx) or operating (OpEx) expenses. The decision between CapEx and OpEx is based on your organization’s budget model and whether your organization prefers to have assets capitalized or be treated as a monthly expense. A public cloud model, for example, works best for an OpEx arrangement because it offers pay-as-you-go pricing and eliminates the need to purchase physical assets.

Your objectives will help you determine whether the public, private, hybrid, or community cloud deployment model is the best fit. For the remainder of this book, we will focus on the various consumption models for private cloud.



# Approaches to an OpenStack Private Cloud

If, after the considerations outlined above, you've decided to deploy an OpenStack cloud that is dedicated completely to your organization, congratulations! Next, you'll need to decide how to consume your cloud. There are three primary choices: pay someone to manage OpenStack for you (managed cloud), use an OpenStack distribution (distro), or download software from OpenStack.org and build your own system (DIY).

## Turnkey cloud computing: Managed cloud

A managed cloud represents the most hands-off approach to cloud computing. While another business creates, manages, and supports the cloud, your IT organization will still need **domain operators**. This requires personnel to manage the relationship between the end users of the cloud—your employees or developers—and the cloud itself. They perform administrative tasks such as creating users, defining the quota of cloud computing services users can consume, managing costs associated with your cloud, and **business partner** troubleshooting. Learn more about the OpenStack *managed private cloud providers*<sup>3</sup>.

## Supported implementation: OpenStack distributions (distros)

You can build and control a substantial portion of your cloud without having to do it alone. *Cloud distribution providers*<sup>4</sup> develop tools to make the most of the OpenStack platform, such as testing, bug fixing, and packaging to prepare OpenStack for your cloud architects. Effectively, you're not paying for software. You're paying for OpenStack implementation and back-end services to support cloud resources that you control and manage.

If you elect to use a distro, your IT organization will need domain operators, similar to organizations with managed clouds do. You'll also need **cloud architects**, who select your cloud's hardware and software infrastructure,

design your cloud's configuration, and determine what services—such as storage, networking, or compute—your cloud will offer.

The cloud architect works with the distribution company to install and configure the cloud. He or she hands off day-to-day management to the **cloud operator** who is responsible for performance and capacity monitoring, troubleshooting and security.

A team or one person, depending on the size of your organization and the demands on your cloud, can fulfill these roles.

## DIY Cloud

A DIY cloud is great for organizations that want unlimited flexibility and customization. In addition to the resources described above, you'll need an engineering team to handle tasks that distribution vendors would otherwise tackle, such as advanced support, implementation, **packaging**, and testing.

The quickest way to a DIY cloud is to use a distribution vendor's software release, but not its paid support services. This offers pros and cons. While you'll be able to get up and running more quickly by taking advantage of the vendor's free distribution code, you'll be limited to the OpenStack components offered, which can be a fraction of the total OpenStack portfolio. And you'll be subject to the vendor's release cycle and bug fixes.

In the *OpenStack Project Navigator*<sup>5</sup>, you can find out which projects are most broadly supported by the vendor.

*Figure 2: Considerations leading to consumption model decisions*

IS THIS CLOUD GOING TO BE CORE TO MY BUSINESS?			
Yes	DIY/Self-Support	Distro	Managed
No	Distro	Managed	

  

DOES MY CLOUD NEED TO BE IN PRODUCTION WITHIN 3-6 MONTHS?		
Yes	Distro	Managed
No	DIY/Self-Support	

**DO I NEED TO TAKE ADVANTAGE OF NEW OPENSTACK CAPABILITIES AS SOON AS THEY ARE AVAILABLE?**

**Yes**                      DIY/Self-Support

**No**                              Distro                              Managed

**DO I WANT TO INFLUENCE THE DIRECTION OF OPENSTACK BY DIRECTLY CONTRIBUTING TO UPSTREAM DEVELOPMENT OR BY PROVIDING MY REQUIREMENTS TO A VENDOR?**

**Contrib**                      DIY/Self-Support

**Vendor**                              Distro                              Managed

**DO I HAVE ANY EXISTING VENDOR PARTNERSHIPS THAT I WANT TO EXPAND?**

**Yes**                              Distro

**No**                              DIY/Self-Support                              Managed

**ARE THE OPTIONS PRESENTED BY CLOUD VENDORS SUFFICIENT TO MEET THE CONFIGURATION I REQUIRE?**

**Yes**                              Distro                              Managed

**No**                              DIY/Self-Support

**DOES THE VALUE I GAIN FROM HAVING GREATER CONTROL OVER MY CLOUD RESULT IN A SUBSTANTIALLY GREATER VALUE TO MY BUSINESS?**

**Yes**                              DIY/Self-Support

**No**                              Distro                              Managed

**DOES MY ORGANIZATION WANT TO INVEST CAPEX IN BUILDING THE ENVIRONMENT OR DO WE WANT TO ONLY PAY FOR A SERVICE?**

**Yes**                              DIY/Self-Support                              Distro

**No**                              Managed

**DOES MY BUSINESS HAVE THE RESOURCES TO SUPPORT THE RIGHT AMOUNT OF IT PERSONNEL TO RUN MY CLOUD?**

<b>Yes</b>	DIY/Self-Support	Distro
<b>No</b>	Distro	Managed

**ARE THE REQUIRED ENGINEERING SKILLS ASSOCIATED WITH THE MODEL I CHOOSE SOMETHING ALREADY IN MY ORGANIZATION?**

<b>Yes</b>	DIY/Self-Support	Distro
<b>No</b>	Managed	

*Figure 3: Approximate business values by consumption models*

	MANAGED	SUPPORTED DISTRO	DIY
Time to production	Fastest	Moderate	Slowest
Configuration flexibility	Low	High	Highest
Software Customization	None	Depends on vendor policy	Unrestricted
Support	Vendor	Vendor	Community
External Costs	High	Med	Low
Internal Resource Requirements	Low	Med	Highest
Level of Resource Skill	Low	Med	High



# Forming the OpenStack Team

Staffing is a critical component of a successful OpenStack deployment. You'll need a staffing plan that identifies and defines the appropriate skill sets needed during each phase of implementation. Think of this as a map that outlines the staffing requirements for your project for current and future needs.

The creation of the staffing plan can be broken down into three straightforward activities: the **staffing assessment**, **staffing profile**, and **staffing resources**. Some factors to consider when developing a staffing plan include:

- Engaging with company and department managers to clarify the work being performed for OpenStack adoption.
- Developing a working group of employees interested in defining the overall strategy.
- Determining the type and quantity of anticipated workloads to deploy. How “elastic” are these workloads?
- Knowing what is the scheduled timeline to complete each phase.
- Understanding the overall size of the platform.
- Assessing the desired end state of your deployment.
- Developing a vision statement for the end state at some future point—how big will it be and how quickly do you want to get there.

## Staffing assessment

A staffing assessment reviews the overall project goals and identifies the skill sets needed during each implementation phase.

An OpenStack implementation frequently utilizes a phased plan approach. They include:

- **Proof-of-concept (PoC)** – focusing on planning the overall deployment and the initial goals.
- **Pilot** – stresses the system, works on expected edge cases, and zeros in on where you need things to be to begin production.
- **Production** – focusing on the program deployment and long-term goals.

During each phase, a staffing assessment should be performed to ensure goals are being met and to anticipate future changes. Enterprise users find it useful to reevaluate staffing needs on a regular basis as their platforms continue to grow and develop. Considerations from current production enterprises include:

- **External support** – Organizations with large initial deployments and quick production ramps have benefited from external support, either through a managed or supported distribution consumption model.
- **Shared resources** – Depending on the skill level of available resources, you can bring in a shared resource to help train and develop your teams for a specified time period. In turn, your team can help identify and define necessary skills for future team members, and train and develop those new hires.
- **Shifting resources** – When existing and future applications are moved onto your OpenStack cloud, headcount can be shifted from those application teams to the cloud team to assist in developing the infrastructure for their success.

## Staffing profile

A staffing profile defines the roles needed during each phase of implementation according to the implementation goals and needs.

Based on feedback from production enterprise users of OpenStack, the chart below covers the most crucial roles during the three phases of implementation. The roles listed below are not indicative of the number of personnel. Generally, a person can fill multiple roles, depending on experience and adopted consumption models. Staffing needs vary from enterprise to enterprise, and there is no one standard that applies to all cloud adoption strategies. A staffing profile should be performed for the roles you are looking to fill during each phase.

Figure 4: Role map for OpenStack deployment phases

PHASES	CONSUMPTION MODELS		
	MANAGED OPENSTACK CLOUD	SUPPORTED OPENSTACK DISTRIBUTION	DIY OPENSTACK CLOUD
<b>PROOF-OF-CONCEPT</b>	Domain Operations	Domain Operations Cloud Architect Cloud Operations	Domain Operations Cloud Architect Cloud Operations Support
<b>PILOT</b>	Cloud Architect	Domain Operations Cloud Architect Cloud Operations Cloud Evangelist	Domain Operations Cloud Architect Cloud Operations Support Packaging Cloud Evangelist
<b>PRODUCTION</b>	Domain Operations Cloud Architect Cloud Evangelist	Domain Operations Cloud Architect Cloud Operations Support <u>Upstream Contributions</u> Cloud Evangelist	Domain Operations Cloud Architect Cloud Operations Support Packaging Upstream Contributions (optional) Cloud Evangelist

## Staffing resources

Identify available skills and staffing resources as existing staff, new staff, outside consulting services, and/or tools that augment the capacities of current staff. Consider who has expressed interest in joining or working on your cloud team and whether or not they will be split or dedicated during the proof-of-concept and pilot phases.

The following skills can be integral in building your cloud teams:

- Linux/UNIX – Command line interface (CLI) and system administration proficiency.

- Cloud computing and virtualization concepts – ability to differentiate between the concepts and technologies of cloud platforms and virtualization platforms.
- Configuration management (e.g. Chef, Puppet) – experience working with scripts, fixing bugs, and automation.
- Development – experience creating design specifications and software updates for cloud platforms.
- Engineering – experience with operating systems, hypervisors, network protocols, and platform patching and maintenance.
- Cloud operations – experience ensuring overall platform health, perform troubleshooting, and maintain capacity expectations.
- Support – experience with automation and monitoring systems.

These steps should be performed during each phase of deployment and re-evaluated regularly.

The OpenStack website provides channels for sourcing and training skilled resources such as *Certified OpenStack Administrator*<sup>6</sup> or *training providers*<sup>7</sup>.

## Finalize and implement a staffing plan

A staffing plan can provide a rational basis for developing and funding programs needed to support your OpenStack implementation and growth plans. As we mentioned above, you should revisit this regularly and update as needed.

## Projecting future staffing requirements

To build the long-term health of your resources, consider the following for staffing:

- Build an internal pipeline of candidates for future growth and possible turnover.
- Train and develop junior employees.
- Institute a company internship.
- Enable employee growth and development.

The *OpenStack Job Board*<sup>8</sup> is a good resource to help define positions in your organization as well as post your positions to help find qualified candidates to fill them.



# Organization and Process Considerations

When organizations deploy private clouds, they are not only investing in a technology project, they are also redefining the relationship between IT and the lines of business. A successful private cloud deployment requires the groups to work together and agree on business challenges prior to implementation. These challenges include the following:

## **Aligning business processes to maximize private cloud value**

Increased business agility from faster delivery of IT services is a primary benefit of deploying a private cloud. However, existing business processes might not align with the speed of the private cloud process. For example, through the automation of a private cloud, a virtual machine can be provisioned in several minutes, but the approval process might still take several days or weeks. For an organization to maximize the value of its investment in a private cloud, IT and the line of business should identify and reassess the processes that impede agility.

## **Aligning the IT organization to cloud realities**

From an infrastructure point of view, the cloud will affect how you approach implementation and ongoing operations across the board. Many large organizations have two separate IT divisions: developers and operations (infrastructure support and management of servers, storage, or networking). In a cloud model, these overlapping teams need to cooperate for successful deployment. A siloed approach should be revisited as you transition to the cloud.

To take full advantage of the increased deployment speed and agility that cloud promises to deliver, consider adopting a **DevOps** model for the creation and roll-out of new business services. Through a DevOps approach, both groups should be able to closely align development with operations through shared goals, metrics, and processes.

## Defining the meaning of self-service in your organization

One of the defining characteristics of cloud computing is **end user self-service** by enabling on-demand access to resources through an automated portal and services catalog. Traditionally, IT has been responsible for provisioning, maintaining, and de-provisioning service requests. But a complete self-service environment shifts these roles and responsibilities, such as enforcing workload compliance and security and the resulting costs, from IT to the lines of business. To help ensure a smooth private cloud rollout, IT and the lines of business must clearly define the division of labor and associated costs.

Operators might see self-service as mixed access to applications, compute, storage, and networking infrastructure. It is more prudent to think of self-service as a set of capabilities for a given organization rather than a mix of technologies. It is important to understand the willingness of end users to engage in self-service capabilities.

## Choosing suitable workloads

When evaluating cloud models, some users assume that deploying a private cloud means transferring all data center applications and services into the cloud. However, not all use cases are an ideal fit for cloud. IT should work with the targeted users to explain the nature and capabilities of the cloud and discuss the services and workloads that provide the most value in terms of time and cost. A particular emphasis should be placed on workloads and use cases that benefit from speed, require a scale-out environment, are dynamic, can be standardized, or require frequent provisioning and de-provisioning. This could include workloads such as dev/test, scenario modeling, or financial applications at the close of a quarter.

When assessing legacy workloads, IT and application owners should consider the system's current lifecycle stage. Migrating and re-architecting might be the best solution to optimize the value of transitioning to OpenStack. Legacy workloads that are built to scale up and change infrequently might not be immediate candidates for the cloud. Perhaps more importantly, it will be critical to ensure that the initial cloud projects are chosen to highlight the value of cloud-based delivery and to show quick "time-to-value."

## Speed application delivery

To get the full benefit of a cloud infrastructure, enterprises must follow best practices in agile development, **continuous integration** and **continuous delivery**. Organizations should consider changing development methodologies and processes with the infrastructure transition. This can include automating test and deployment of applications in development, staging, and production

environments to rapidly deliver new production code. This requires new investments in technology and staff. Training development teams on the use of new tools and the IT support organization on how OpenStack works is critical to overall project success.

### To bill or not to bill

Cloud computing can provide unprecedented insight into IT resource consumption to the lines of business. Implementing **chargeback** or **showback** might require a cultural shift in the organization. Historically, the business unit allocated IT costs, while business personnel and IT justified deployments. With the metering capabilities of a private cloud, use can be directly assigned and charged to consumers' organizations. Additionally, self-service can lead to an erosion of business justification and result in over-consumption of IT services and increased server sprawl. IT and the line of business should implement billing or showback to control costs and demonstrate business value. IT should be prepared to demonstrate how billing benefits the business and how it is a necessary component of a successful private cloud deployment.

### Consider third-party fees

While OpenStack is free for anyone to use, other costs are associated with the support and maintenance of the code base, whether through in-house staff, a managed provider, or a distribution. A larger consideration involves the impact on the license costs for third-party software that you plan to run on the cloud. In a traditional data center deployment, the operations team controls which servers are assigned software. But in a flexible cloud environment, workloads can end up on any server in the cloud. Some software vendors have introduced new license models for cloud deployments that could require a license for every server in the cloud cluster, which could result in significant cost increases. Use of OpenStack features, such as cells and careful configuration of scheduler rules and image metadata, can limit this exposure. However, you should explore the policies of any critical third-party code that you plan to use in cloud deployments.



# Choosing Workloads for Your Cloud

Enterprises that want to transition from virtualization to OpenStack have to decide which applications to transition first. Failure to properly understand the implications and contingency planning could result in higher costs and potential loss of business.

To help with this process, here are some steps to ensure successful migration of existing applications and/or implementation of new application projects for OpenStack:

1. Take inventory of the applications and characterize the workloads: understand the characteristics; decide which applications are suitable for virtual machine, bare metal, and/or container deployments.
2. Depending on the workload characteristics, implement selection criteria based on expected efforts for migration and transformation to the OpenStack cloud.
3. Choose the workloads to prove out the architecture on OpenStack.
4. Measure OpenStack's value, taking short-term/long-term value and tangible and intangible benefits into consideration.

## Characterize workloads

As a first step, inventory and characterize all of the workloads that can be moved to the OpenStack environment. This includes finding the best-fit workloads for an OpenStack environment. All applications should be rated against various criteria to determine if it's first strategically and technically possible, and then reasonable to move them to the OpenStack environment.

Even if an application is suitable for the cloud, operators have to make decisions about whether the deployment models (virtual machine, bare metal, and/or container) required for the application are supported. It often makes sense to start with low-risk, modular applications—those with minimal customer data, legacy touch points, and/or sensitive information—or applications that can take advantage of elasticity and self-service.

*Figure 5: Less complex applications for initial OpenStack deployment*

STANDALONE, LESS COMPLEX APPLICATIONS	TECHNICAL ASPECTS OF LESS COMPLEX WORKLOADS
Limited database access to company's management systems (e.g. web applications, basic streaming)	<ul style="list-style-type: none"> <li>• Scaling paradigm: Scale-out</li> <li>• Automatic and horizontal scaling for each service and component of the application</li> </ul>
Run infrequently but require significant compute resources; benefit from elasticity (e.g. batch processing)	<ul style="list-style-type: none"> <li>• Modular, loosely-coupled distributed application architecture; APIs for each service</li> <li>• Shared-nothing architecture</li> </ul>
Run in a time zone different from IT; benefit from self-service (e.g. internal-facing applications, seasonal workloads)	<ul style="list-style-type: none"> <li>• Resiliency in app, not reliant on infrastructure</li> <li>• Deals gracefully with timeouts</li> </ul>
Applications running out of capacity; benefit from scaling	<ul style="list-style-type: none"> <li>• Services providing active-active redundancy</li> <li>• Use of distributed storage</li> </ul>
Development and testing (e.g. custom applications)	<ul style="list-style-type: none"> <li>• Broad consumption of infrastructure and platform services</li> <li>• Replication of data done on software level</li> </ul>
Back-office applications (e.g. email, project management, expense reporting)	<ul style="list-style-type: none"> <li>• Asynchronous communication</li> <li>• Commodity hardware building blocks</li> </ul>

Figure 6: More complex applications for later cloud enablement

MORE COMPLEX APPLICATIONS, OFTEN REQUIRING ENTERPRISE INTEGRATION	TECHNICAL ASPECTS OF MORE COMPLEX WORKLOADS
Frequent and/or high volume transactions against a DB that cannot be migrated to cloud (e.g. stock trading)	<ul style="list-style-type: none"> <li>• Scaling paradigm: Scale-up</li> <li>• Important complex and centralized systems</li> </ul>
Resource-intensive (memory, IO) or require specific hardware support (e.g. Big Data, Hadoop, database)	<ul style="list-style-type: none"> <li>• Big failure domains</li> <li>• Services protected by failover pairs</li> </ul>
Require integration with company management information databases (e.g. ERP, human resources)	<ul style="list-style-type: none"> <li>• Dedicated servers or virtual machines managed manually by administrators</li> </ul>
High security and compliance requirements (e.g. electronic health records)	<ul style="list-style-type: none"> <li>• Consumes large SANs or persistent block storage</li> <li>• Consumes high CPU (GPU) or high-speed SSD storage</li> </ul>
Performance-sensitive (e.g. Business Intelligence)	<ul style="list-style-type: none"> <li>• All infrastructure components expected to be 100% available</li> </ul>
Runs on legacy platforms or require specialized hardware (e.g. mainframe or specialized encryption hardware)	<ul style="list-style-type: none"> <li>• Requires high-performance hardware to make infrastructure highly available.</li> </ul>

## Selection criteria

Moving traditional, static, and monolithic workloads to an infrastructure-as-a-service (IaaS) platform can be challenging. To reliably host these legacy apps on an IaaS platform, you might first have to transform the structure or architecture of the application. There are a wide range of options between a simple “lift and shift” approach versus a strategy to partially or completely restructure an application during migration.

These are just some of the strategies for moving applications to an OpenStack cloud:

- **Redeployment** – An application and its components are redeployed and moved, without modification.
- **Cloudification** – An application is moved to the cloud as-is but consumes public cloud resources or services to replace some of the applications components and services, e.g. from platform as a service (PaaS) and software as a service (SaaS) solutions.

- **Relocation with optimization** – An application is redeployed (relocated) and modified to consume IaaS functionality and services wherever possible.

The amount of effort required depends on the migration strategy and the extent of transformation. The move to OpenStack has to pay off as well. So, the expected cost to move a workload to the OpenStack environment should be determined as criteria for workload selection.

## Workload and OpenStack's project dependencies

Before finalizing your workload selection, examine the workloads' dependencies and the maturity for **OpenStack projects** involved.

When selecting a workload suitable for OpenStack, you also have to decide which infrastructure services are needed. Understanding these dependencies can help you determine the best fit between a workload and the required characteristics and functionality of the OpenStack platform. To leverage the experience of the OpenStack community in this area, review the *OpenStack Project Navigator*<sup>9</sup> and *Sample Configurations*<sup>9</sup>.

The Project Navigator provides information about the primary OpenStack projects and helps users make informed decisions about how to consume the software. Each project is rated by data provided by the OpenStack Technical and User Committees. The rating is based on the level of adoption, maturity, and age of the specific projects.

Based on OpenStack case studies and real-world reference architectures, the samples configurations provide information on core and optional OpenStack projects for popular workloads. This information is essential to finalizing the workloads host on the platform.

*Figure 7: Sample configurations available*



**High-throughput computing**



**Web hosting**



**Public cloud**



**Web services and eCommerce**



Compute starter kit



Big Data



DBaaS



Video processing and content delivery



Container optimized

Figure 8. Example Sample Configuration: Web Services and Ecommerce

Core Services included in this configuration (5 of 6)

**CINDER**  
Block Storage

Provides persistent block storage to running instances. Its pluggable driver architecture facilitates the creation and management of block storage devices.

86% Adoption   7 of 8 Maturity   4 yrs Age

MORE DETAILS

**GLANCE**  
Image Service

Stores and retrieves virtual machine disk images. OpenStack Compute makes use of this during instance provisioning.

94% Adoption   5 of 8 Maturity   6 yrs Age

MORE DETAILS

**KEYSTONE**  
Identity

Provides an authentication and authorization service for other OpenStack services. Provides a catalog of endpoints for all OpenStack services.

96% Adoption   7 of 8 Maturity   4 yrs Age

MORE DETAILS

**NEUTRON**  
Networking

Enables network connectivity as a service for other OpenStack services, such as OpenStack Compute. Provides an API for users to define networks and the attachments into them. Has a pluggable architecture that supports many popular networking vendors and technologies.

89% Adoption   6 of 8 Maturity   4 yrs Age

MORE DETAILS

**NOVA**  
Compute

Manages the lifecycle of compute instances in an OpenStack environment. Responsibilities include spawning, scheduling and decommissioning of machines on demand.

96% Adoption   8 of 8 Maturity   6 yrs Age

MORE DETAILS

**SWIFT**  
Object Storage

Scores and retrieves arbitrary unstructured data objects via a RESTful, HTTP based API. It is highly fault-tolerant with its data replication and scale-out architecture. Its implementation is not like a file server with mountable directories.

62% Adoption   7 of 8 Maturity   6 yrs Age

MORE DETAILS

Optional Services included in this configuration (2 of 13)

NAME	SERVICE	MATURITY	AGE	ADOPTION	DETAILS
Horizon	Dashboard	6 of 8	4 Yrs	95 %	More Details
Trove	Database	2 of 8	2 Yrs	27 %	More Details

## Measuring OpenStack's value

Given the change-averse culture of IT, it is important to prove out the architecture and the value of OpenStack, in addition to delivering the same or better service levels for the cloud-based applications as compared to current service levels. This requires that you measure current service levels delivered as a baseline up front. Planned service disruptions will be unavoidable but can be justified by the benefits and improvements in service levels.

Some enterprises have used the following criteria for comparison:

- Target to raise the percentage of virtualization.
- Improve the admin-to-server ratio.
- Reduce the cost per virtual machine (VM).
- Reduce overall infrastructure costs (by going SaaS).
- Increase (double or triple) number of applications.
- Enable faster delivery of infrastructure resources (as measured in days versus weeks).
- Determine the number of minutes it takes to restore business operations using DR site.
- Improve response times or user satisfaction levels for initial users.
- Achieve buy-in from internal customer acquisition as measured by increase of deployed applications.

Now that you have determined the suitable workloads for your cloud, the next step will be deploying these workloads in phases, from PoC through production.



# Implementation Phases

Implementing a cloud can be a challenging process for enterprises—one that depends on different factors including budget, people, and consumption model. For most enterprises, OpenStack is implemented in three phases: PoC, pilot, and production. In this section, we highlight things to consider for each phase.

## OpenStack services to support the selected workloads

OpenStack offers many services such as compute, storage, application catalog, and database. Depending on the use cases, workload requirements, and implementation phase, you might not use all OpenStack services at once. Most PoC deployments start with a simple set of core services and slowly include additional services after your team has gained familiarity with OpenStack. For example, some might start with only the compute capability (Nova) to provide a self-service VM provisioning feature without any persistent storage capability (Cinder or Swift). Refer to the [Choosing Workloads for Your Cloud](#) chapter to determine the initial use cases and applications, and consult your preferred vendor or OpenStack expert to understand which services should be included in your deployment.

The following table provides a brief summary of commonly used OpenStack services. Please refer to *OpenStack Project Navigator*<sup>5</sup> for a list of primary projects. Visit the OpenStack Project Teams web page for a *complete list of all OpenStack projects*<sup>10</sup> governed by the OpenStack Technical Committee.

Figure 9: Primary OpenStack services

OPENSTACK SERVICE	OPENSTACK PROJECT	DESCRIPTION	DEPLOYMENT CHOICE
Compute	Nova	Manages the lifecycle of compute instances in an OpenStack environment. Responsibilities include spawning, scheduling, and decommissioning machines on demand.	Required
Identity service	Keystone	Provides an authentication and authorization service for OpenStack services. Provides a catalog of endpoints for all OpenStack services.	Required
Networking	Neutron	Enables network connectivity as a service for other OpenStack services, such as OpenStack Compute. Has a pluggable architecture that supports many popular networking vendors and technologies. Provides an API for users to define networks and advanced services such as <b>FWaaS</b> , <b>LBaaS</b> .	Required
Block storage	Cinder	Provides persistent block storage to running instances. Its pluggable driver architecture facilitates the creation and management of block storage devices.	Optional unless requiring persistent storage attachment to VM.
Object storage	Swift	Stores and retrieves arbitrary unstructured data objects via a RESTful, HTTP based API. It is highly fault tolerant with its data replication and scale-out architecture. Its implementation is not like a file server with mountable directories.	Optional but very useful cloud-based persistent storage.
Image service	Glance	Stores and retrieves virtual machine disk images. OpenStack Compute makes use of this during instance provisioning.	Required

Dashboard	Horizon	Provides a web-based self-service portal to interact with underlying OpenStack services, such as launching an instance, assigning IP addresses and configuring access controls.	Optional but generally included in most default installations.
Orchestration	Heat	Orchestrates multiple composite cloud applications by using either the native Heat Orchestration Template (HOT) format or the AWS CloudFormation template format, through both an OpenStack-native REST API and a CloudFormation-compatible Query API.	Optional but very useful feature.

## Cloud capacity and performance

Planning for enough capacity to support the use cases or workloads is important to the success of your cloud deployment. In OpenStack, every virtual machine is associated with a virtual hardware template called a **flavor**, which defines the vCPUs, memory, and **ephemeral storage** for each virtual machine. The following table illustrates the default flavors in most OpenStack installations. You can customize these or define new ones that suit your environment.

*Figure 10: Default OpenStack flavors*

FLAVOR NAME	VCPU	MEMORY	EPHEMERAL DISK
m1.tiny	1	512 MB	1 GB
m1.small	1	2048 MB	20 GB
m1.medium	2	4096 MB	40 GB
m1.large	4	8192 MB	80 GB
m1.xlarge	8	16384 MB	160 GB

The total amount of vCPU, memory, or ephemeral storage consumed by all the VMs of your workload will determine your hardware architecture. Most deployments tend to overcommit the CPU (e.g. 1:5 ratio of physical CPUs to virtual CPUs) but not overcommit the memory (1:1 ratio of physical memory to virtual memory). You should also consider extra CPU and memory usage required by the OpenStack services and local system processes, and the amount of disk space required to provision the VM's ephemeral disk. SSD drives can be used on the compute hosts for faster performance. This

information will help you design the hardware architecture of your cloud. You might refer to the *OpenStack Operations Guide*<sup>11</sup> for more details on estimating the hardware requirements.

In OpenStack, storage can be ephemeral or persistent. Ephemeral storage will persist with the lifespan of the associated VM. Anything written to that storage will be lost if the associated VM is deleted. In most default installations, the virtual disks associated with the VMs are ephemeral. If your use cases require saving the data regardless of the VM lifespan, you need to consider using persistent storage. OpenStack offers a few types of persistent storage such as block storage and object storage. Data written to these persistent storage types will remain available regardless of the state of the associated VM. This information will be useful to estimate your storage requirements. If your use case requires high disk I/O performance, consider high-speed SSD drives here as well.

Networking is also an important aspect when it comes to OpenStack deployments. Use case requirements such as latency, throughput, and security will influence your network architecture design. Consider using high-speed 10G network equipment or network bonding for better performance and redundancy. Different vendor distributions might use different network topologies in their architecture. You should work closely with the enterprise network architect and information security architect to determine how to integrate OpenStack network with your corporate network environment such as vlan, firewall, and proxy.

## Automation and tooling

It is highly recommended to incorporate automation capabilities, even in the early phase of your cloud implementation. Running an OpenStack cloud requires a new operating model, as compared to the conventional IT model. Adopting a DevOps mindset and CI/CD model allows you to get the best benefits from your OpenStack cloud and accelerate the time-to-value.

While you can deploy OpenStack manually (either DIY or using distro), this approach quickly becomes burdensome beyond just a few hosts. It is not recommended to deploy OpenStack services manually given its services dependency complexity. Automation and configuration management are the keys to successfully deploying and maintaining a healthy OpenStack cloud.

Most OpenStack distributions have their own deployment tools. If you are not using a distribution and choose to build a DIY cloud, there are tools available such as: OpenStack Ansible, Chef OpenStack, Puppet OpenStack, Kolla, TripleO, and Fuel. If these options don't work or are missing features needed for your

DIY cloud, you can build something yourself. However, you need to sort out the package dependency issues for a successful installation. It is key to do some sort of automation to get your cloud up and running quickly in a stable and manageable manner.

Other tasks should be considered for day-to-day cloud operation such as host and VM monitoring and alerts, logging, accounting, disaster recovery, and user management. Most enterprise IT organizations have existing tooling to support data center operation. If you prefer to use this tooling, ask your vendor if they provide integration with OpenStack. Other open-source toolsets and plugins are available that provide similar capabilities to operate your cloud.

From an enterprise integration perspective, OpenStack provides capabilities to integrate with existing authentication systems such as Active Directory or LDAP. You might also consider federated authentication in your cloud. Unfortunately, OpenStack today does not have a comprehensive solution to integrate with other enterprise systems such as CMDB, Asset Management, or ITIL. You can discuss these with your vendors, and customize the relevant software to satisfy your needs. The OpenStack community is actively seeking solutions to fill these gaps.

## Engage with users

One of the common failures of cloud deployment is the lack of user engagement. Educate users on the business value and benefits they will realize from the new cloud deployment. Keep the cloud simple for your users and remove any unnecessary steps or procedures. Make your cloud services self-explanatory and provide good documentation and training. Request feedback from your users and continuously improve the user experience.

## Evolving from PoC to pilot to production

The first phase of deploying your OpenStack cloud is the proof-of-concept (PoC). Be aware that your budget could influence your implementation strategy. You might consider running your PoC on a virtualized environment without dedicated physical servers. However, that will not provide the best performance experience. You can choose to deploy OpenStack on bare metal resources from a public cloud provider, using a subscription-based model. If the PoC is for an initial feasibility study or prototyping, you might consider running an all-in-one deployment on a single physical host. Many of the OpenStack distros provide a quick and easy “all-in-one” installation. This helps familiarize your team with OpenStack concepts before the company decides to invest more resources into the next phase.

To help estimate the bill of materials, revisit the cloud capacity and plan for future expansion. Many moving parts (e.g. CPU, memory, storage, networking) are needed to run an OpenStack cloud. Understanding your use case scenario will help avoid over-provisioning one element and under-provisioning another. Refer to the *Cloud Capacity and Performance* section earlier in this chapter and consult your vendor or OpenStack expert for the most cost-effective and scalable hardware configuration while expanding your cloud from phase to phase.

OpenStack can be deployed in high availability mode, which requires a more complex configuration and more physical nodes. Depending on your use case requirements, budget, and objectives, a simple approach is to deploy a functional OpenStack cloud in non-HA mode using the fewest, minimally configured physical nodes during the PoC phase before you move into HA mode in pilot or production.

It is recommended that you engage users to validate the workload deployment on your cloud. Continuously evaluate the cloud-based workload for impact and potential changes to existing organizational processes and operating model. Plan for team expansion and include other domain experts whenever necessary from phase to phase. Establish new operating procedures before you move into pilot phase. Once you have gained sufficient knowledge in running your OpenStack cloud, you might consider supporting more use cases. Do not rush into pilot or jump into production if your team is not ready. Spend more time in PoC and educate your team by experimenting with various use case scenarios. Operating an OpenStack cloud will be totally different than a conventional virtualization environment.

The following table recommends a few deployment scenarios for the PoC and pilot phases. Configurations can vary depending on the use cases and objectives.

Figure 11: Deployment scenario recommendations for PoC and pilot phases

DEPLOYMENT SIZE & TOPOLOGY	BENEFITS	RECOMMENDED USE CASE / WORKLOAD CHARACTERISTICS	RECOMMENDED OPENSTACK PROJECTS
<b>1 node all-in-one</b>	Easy to start	<ul style="list-style-type: none"> <li>Gaining familiarity with key OpenStack concepts</li> <li>Simple VM provisioning with very limited capacity</li> </ul>	<ul style="list-style-type: none"> <li>Nova, Neutron, Keystone, Glance, Horizon</li> <li>Most distro's all-in-one also include other services such as Cinder/Swift/Heat by default</li> </ul>
<b>4 nodes, multi-node non-HA: 1 for controller node, 3 for compute nodes</b>	Small cloud capacity for PoC applications  Fast experiments	<ul style="list-style-type: none"> <li>PoC or pilot cloud projects</li> <li>Simple VM provisioning</li> <li>Limited storage</li> <li>Experimental greenfield application or conventional 3-tier app.</li> </ul>	<ul style="list-style-type: none"> <li>Nova, Neutron, Keystone, Glance</li> <li>Optional: Cinder/Swift</li> <li>Consider Heat &amp; Ceilometer</li> </ul>
<b>12 nodes multi-node HA: 3 for controller nodes, 3 for Compute nodes (Nova), 3 for Cinder storage nodes, 3 for Swift storage nodes.</b>	High availability  Larger cloud capacity  Dedicated compute and storage	<ul style="list-style-type: none"> <li>PoC or small production workloads</li> <li>SLA requirements</li> <li>Big data application</li> <li>Scalable cloud-aware app</li> </ul>	<ul style="list-style-type: none"> <li>Nova, Neutron, Keystone, Glance, Cinder, Swift, Horizon, Heat &amp; Ceilometer</li> <li>Optional: Swift</li> <li>Consider Murano, Trove, Sahara</li> </ul>

Moving the OpenStack cloud into the production phase requires thorough architecture planning. One of the most frequently asked questions is how to scale your cloud in a production environment. The OpenStack cloud is designed to be a modular and scalable system that allows services to be distributed across multiple nodes or clusters. Most production deployments choose an HA architecture to provide reliable, high quality service to mission-critical applications. Some areas require more planning and design for an HA deployment.

Various OpenStack services are **stateless**, where every request is treated as an independent transaction, unrelated to any previous or subsequent request. This service does not need to maintain the session or status information for each communication. Examples include nova-api, nova-conductor, glance-api, keystone-api, neutron-api. To provide HA, you can simply provide redundant instances of the service and load balance them.

In OpenStack, there are also stateful services that need to be handled with care in an HA configuration. In a stateful service, a request depends on the results of another request to complete a transaction or action. This requires that the session or status information for each communication is maintained. Examples include backend databases and message queues. Clustering technologies such as MySQL Galera, RabbitMQ Clustering, Pacemaker, and Corosync are generally used in OpenStack HA deployment, which requires more complex configurations.

OpenStack provides mechanisms that segregate your cloud and allow you to deploy applications in a distributed or redundant site, eliminating single points of failure. In OpenStack, cells and regions are used to distribute a single cloud across different geographical locations. Availability zones or host aggregates are used to partition a deployment within a single site. They can be used to arrange a set of hosts into logical groups with physical isolation. For example, grouping the compute hosts so they use different power sources, where a failure of one will not bring down both. Host aggregates are used to group a set of hosts that share common features, for example, two types of hypervisor (e.g. KVM vs ESXi) or different classes of hardware (e.g. SSD vs. HDD storage). These segregation methods allow you to design the cloud infrastructure as common logical groups based on physical characteristics or capabilities, and to ensure that specific workloads can be distributed and placed on relevant resources. The *OpenStack Architecture Design Guide*<sup>12</sup> provides good documentation on how to plan, design, and architect your OpenStack Cloud.



# Post-deployment

After successfully deploying your OpenStack cloud, you will need to adopt a set of processes and procedures for maintenance and upgrades. A best practice is to keep your OpenStack cloud homogeneous in terms of **release version**<sup>13</sup> and take measures to prevent portions of your cloud from operating under different versions. Version homogeneity will be key to your ability to continue to automate and operate routine processes.

## Upgrade cycle

OpenStack is committed to an open design and development process. The community operates around a six-month, time-based release cycle with frequent development milestones. During the planning phase of each release, the community gathers for a *Design Summit*<sup>14</sup> to facilitate live developer working sessions and assemble the *roadmap*<sup>15</sup>. You should participate in the semiannual Design Summit to help plan the next OpenStack release, and attend the semiannual *OpenStack Summit*<sup>16</sup> to understand and evaluate new features. At the OpenStack Summit, you are encouraged to share your knowledge about architecting and operating your OpenStack cloud. At the same time, you will benefit from the knowledge shared by others.

If you have deployed OpenStack using a do-it-yourself approach, you will want to track OpenStack software versions at a project level. You will need to make decisions on when to adopt which branches of code and package for deployment using techniques and resources mentioned earlier in the *Choosing Workloads for Your Cloud* and *Implementation Phases* chapters. Use tools that provide capabilities to support your upgrade strategy and execution. Further, you will want to stay on top OpenStack primary projects using the *Project Navigator*<sup>5</sup> and the community-provided *Multi-release Roadmap*<sup>17</sup>.

If you have implemented your OpenStack cloud using a particular distro, you will need to stay informed around versioning and time your upgrades according to distro releases. Distro providers are active members and active in the OpenStack community. So it will be beneficial for you to follow their movements and actively participate in the OpenStack community.

## Ongoing work

As you have discovered, your OpenStack cloud processes will likely need to change from your pre-cloud processes. As part of your ongoing work, continue to calibrate your processes to the cloud model you have deployed. For example, in your new OpenStack cloud, policies might be set wherein failing devices automatically switch over to working devices. Where in the past a technician might have been deployed to replace the failed device, in your new OpenStack cloud you might want to set periodic sweeps that fix or remove all failed devices and replace them in concert. To take advantage of patches and bug fixes, more frequent updates to applications might be required than pre-OpenStack cloud. These frequent updates will need to become part of your ongoing work routine.

Ideally, your organization will contribute to and leverage OpenStack community resources, such as the *OpenStack Cloud Administrator Guide*<sup>18</sup>.



# Summary

You are now ready—or already have begun—to plan the details of your OpenStack cloud. We hope this book detailed important steps to consider, from initial choices, to building the team, to implementation phases from proof-of-concept through production.

The OpenStack community is the most valuable resource from which to learn more. Visit our *community website*<sup>19</sup> to read the welcome guide, ask questions, join mailing lists, and find user groups near you.

You can also watch presentations from users who have workloads and business challenges similar to yours, and learn more from OpenStack developers contributing to platform development and building apps. Check out all of these from OpenStack Summit presentations on our *OpenStack YouTube channel*<sup>20</sup>.

We hope to hear your success story soon!

# References

1. NIST definition of cloud: <http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf>
2. Available OpenStack public clouds: <http://www.openstack.org/marketplace/public-clouds/>
3. Managed or hosted OpenStack private cloud providers: <http://www.openstack.org/marketplace/hosted-private-clouds/>
4. Available OpenStack distributions: <https://www.openstack.org/marketplace/distros/>
5. OpenStack Project Navigator: <http://www.openstack.org/software/project-navigator>
6. Certified OpenStack Administrator: <https://www.openstack.org/coa>
7. OpenStack Training Providers: <http://www.openstack.org/marketplace/training/>
8. OpenStack Job Board : <https://www.openstack.org/community/jobs/>
9. OpenStack Sample Configurations: <http://www.openstack.org/software/sample-configs>
10. OpenStack Projects: <http://governance.openstack.org/reference/projects/>
11. OpenStack Operations Guide: <http://docs.openstack.org/openstack-ops/content/index.html>
12. OpenStack Architecture Design Guide: <http://docs.openstack.org/arch-design/>
13. OpenStack Releases: <http://releases.openstack.org/>
14. OpenStack Design Summit: [https://wiki.openstack.org/wiki/Design\\_Summit](https://wiki.openstack.org/wiki/Design_Summit)
15. OpenStack Software Roadmap: <https://www.openstack.org/software/roadmap/>
16. OpenStack Summit: <https://www.openstack.org/summit/>
17. Community-provided Multi-release Roadmap: [https://wiki.openstack.org/wiki/ProductTeam/MultiRelease\\_Roadmap](https://wiki.openstack.org/wiki/ProductTeam/MultiRelease_Roadmap)
18. OpenStack Cloud Administrator Guide: <http://docs.openstack.org/admin-guide-cloud/>
19. OpenStack Community website: <http://www.openstack.org/community/>
20. OpenStack Foundation YouTube channel: <https://www.youtube.com/user/OpenStackFoundation>

# Glossary

**Business partner:** An entity that a business group is working or collaborating with, such as a 3rd-party company, vendor or other internal business group.

**Chargeback:** Billing the cost of cloud usage

**Cloud architect:** An individual who is responsible for the architecture design of the cloud.

**Cloud operator:** An individual who is responsible for day-to-day IaaS operation including performance and capacity monitoring, troubleshooting, and security.

**Continuous Integration/Continuous Delivery (CI/CD):** CI is the software engineering practice of merging all developer working copies to a shared repository several times a day. With CD, teams produce software in short cycles, ensuring that the software can be reliably released at any time. It aims at building, testing, and releasing software faster and more frequently.

**DevOps:** A culture, movement, or practice that emphasizes the collaboration and communication of both software developers and other information-technology (IT) professionals or operation team to automate the process of software delivery and infrastructure changes.

**Domain operator:** An individual who is responsible for supporting the cloud consumers/users on using the OpenStack cloud.

**Ephemeral storage:** The storage will be deleted when the associated VM is deleted.

**Flavor:** A virtual hardware template that defines the vCPUs, memory, and ephemeral storage for each virtual machine.

**FWaaS:** Firewall-as-a-Service

**LBaaS:** Load Balancing-as-a-Service

**OpenStack project:** Software development implementation of an OpenStack service.

**Packaging:** Wrapping individual software files and resources—sometimes hundreds of them—together to create a software collection. As with nearly all software, OpenStack relies on other software to function.

**Persistent storage:** The storage remains available regardless of the state of the VM.

**Release version:** OpenStack is developed and released around 6-month cycles. After the initial release, additional stable point releases will be released in each release series.

**Showback:** Showing the cost of cloud usage but not billing them.

**Staffing assessment:** Reviews the overall project goals and identifies the skills needed during each implementation phase.

**Staffing profile:** Defines the roles needed during each phase of implementation according to the implementation objectives.

**Staffing resources:** Available resources including existing staff, potential new staff, outside consulting services and/or tools that augment the capacities of current resources.

**Stateless:** Each request is an independent transaction that is unrelated to any previous request so that the communication consists of independent pairs of request and response, and does not require the server to retain session information or status about each communications partner for the duration of multiple requests.

**Upstream contribution:** Contributing to the development of OpenStack software in the master code repository.

OpenStack, the OpenStack Word Mark and OpenStack Logo are registered trademarks of the OpenStack Foundation in the United States, other countries or both.

All other company and product names may be trademarks of their respective owners.



*This work is licensed under the Creative Commons Attribution-NonCommercial 4.0 International License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc/4.0/>*

