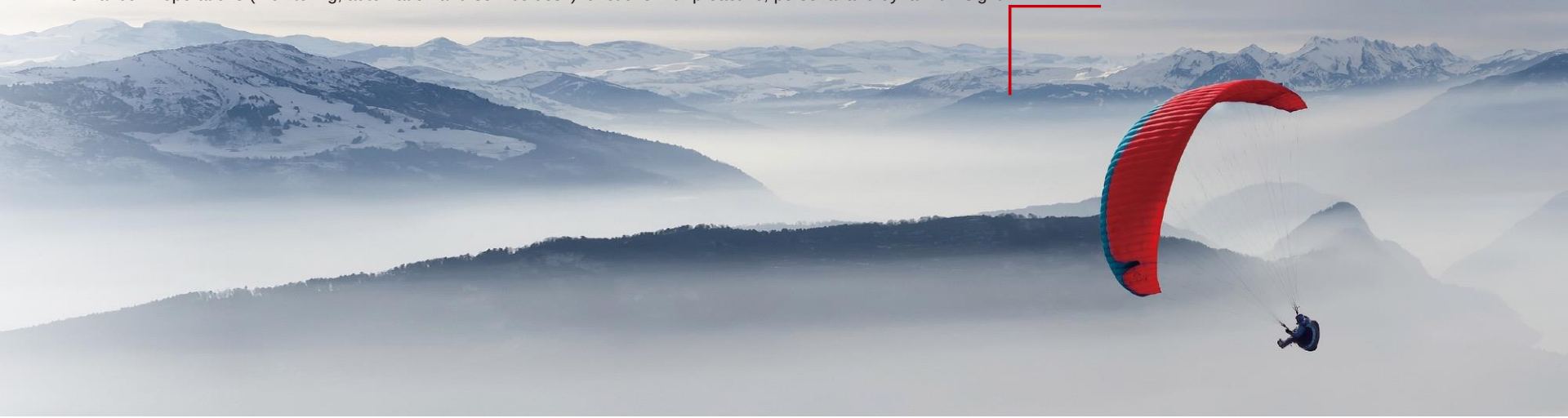
Intelligent Cloud Operations
Part 2. OpenStack Cloud Operating System

Definition (Gartner) [**AIOps**]

AIOps platforms utilize big data, modern machine learning and other advanced analytics technologies to directly and indirectly enhance IT operations (monitoring, automation and service desk) functions with proactive, personal and dynamic insight.



Prof. Jorge Cardoso
E-mail: jorge.cardoso@huawei.com
Large-scale AIOps Lab/SRE Dept.
Munich and Ireland Research Center

2020

Technische Universität Berlin

HUAWEI

Intelligent Cloud Operations
The emerging field of AIOps

The field of **AIOps**, also known as **Artificial Intelligence for IT Operations**, uses advanced technologies to dramatically improve the monitoring, operation, and troubleshooting of distributed systems. Its main premise is that operations can be automated using monitoring data to reduce the workload of operators (e.g., SREs or production engineers). Our current research explores how AIOps – and many related fields such as **deep learning, machine learning, distributed traces, graph analysis, time-series analysis, sequence analysis, advanced statistics, NLP and log analysis** – can be explored to effectively **detect**, **localize**, **predict**, and **remediate failures** in **large-scale cloud infrastructures** (>50 regions and AZs) by analyzing **service management data** (e.g., distributed traces, logs, events, alerts, metrics).

- Planet/large-scale Distributed systems and cloud computing
- Distributed traces, logs, events, alerts, metrics, ….
- Big Data platforms with Kubernetes, Hadoop, Spark, Flink, …
- Deep Learning, machine learning, data mining, advanced statistics, time-series analysis, NLP, ….
- Detect, localize, predict and remediate failures in infrastructures

Dr. Jorge Cardoso is Chief Architect for Planet-scale AIOps at Huawei's Ireland and Munich Research Centers. Previously he worked for several major companies such as SAP Research (Germany) on the Internet of Services and the Boeing Company in Seattle (USA) on Enterprise Application Integration. He previously gave lectures at the Karlsruhe Institute of Technology (Germany), University of Georgia (USA), University of Coimbra and University of Madeira (Portugal). His current research involves the development of the next generation of AIOps platforms, Cloud Operations and Analytics tools driven by AI, Cloud Reliability and Resilience, and High Performance Business Process Management systems. He has a Ph.D. in Computer Science from the University of Georgia (USA).

Interests: **AIOps, Service Reliability Engineering, Cloud Computing, Distributed Systems, Business Process Management**

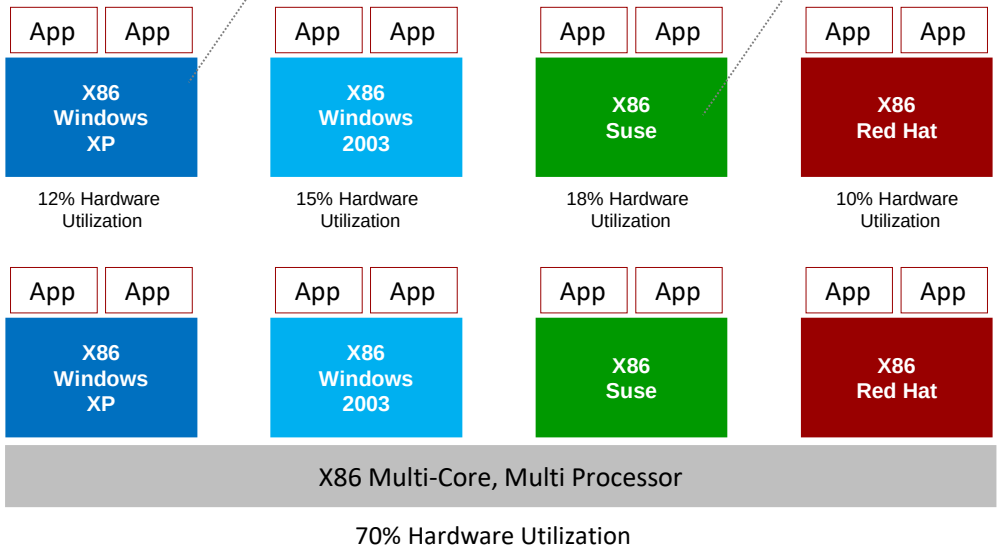
GitHub | Slideshare.net | GoogleScholar

Cloud Computing
Virtualization

Server virtualization is the partitioning of a physical server into multiple smaller virtual servers to maximize resources. The resources of the server are hidden from users.

A virtual machine is an "*efficient, isolated duplicate of a real machine*" (Popek and Goldberg, 1974)

Run the guest operating system as a normal user process on the host. A virtual machine monitor process needs to handle: address space changes, device accesses, system calls,...



Popek and Goldberg defined a set of sufficient conditions for a computer architecture to efficiently support virtualization. The set provides guidelines for the design of virtualized computer architectures

Formal Requirements for Virtualizable Third Generation Architectures

Gerald J. Popek
University of California, Los Angeles
and
Robert P. Goldberg
Honeywell Information Systems and
Harvard University

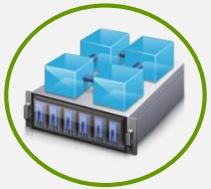
Virtual machine systems have been implemented on a limited number of third generation computer systems, e.g. CP-67 on the IBM 360/67. From previous empirical studies, it is known that certain third generation computer systems, e.g. the DEC PDP-10, cannot support a virtual machine system. In this paper, model of a third-generation-like computer system is developed. Formal techniques are used to derive precise sufficient conditions to test whether such an architecture can support virtual machines.

Communications of the ACM
Vol 17, no 7, 1974, pp.412-421

Cloud Computing Deployment Stages of Enterprises

Virtualization is the base to build cloud services. **Private Cloud** services are used by a single organization and are not exposed to the public. **Public Cloud** platforms are exposed to the public and can be used by anyone. **Hybrid Cloud** services are distributed among public and private clouds. Typically, sensitive applications are kept inside the organization's private network and other services can be hosted in public clouds.

Virtualization



- Computing virtualization
- Storage virtualization
- Network and security virtualization

Focus on resources

Private Cloud

- Automatic management
- Elastic resource scheduling
- HA based on large clusters

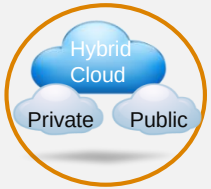
Gradually focus on business

Data Center Consolidation

- Consolidation of multiple DCs
- Multi-level backup and DR
- Software-defined networking (SDN)

Focus on global business

Hybrid Cloud

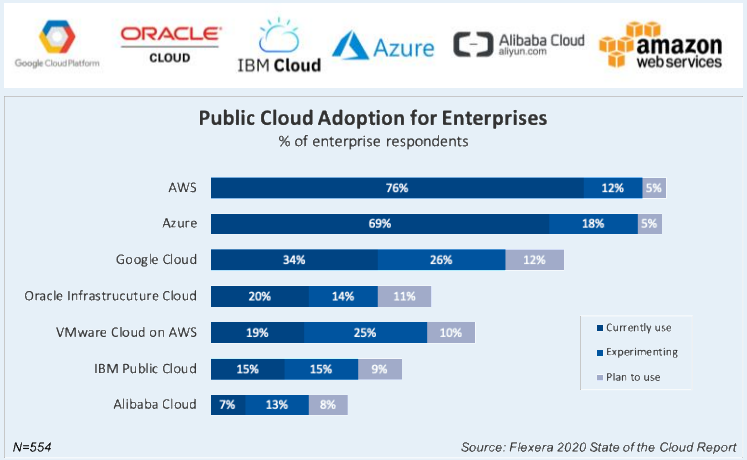


- Unified management
- Optimal resource allocation
- Flexible service migration

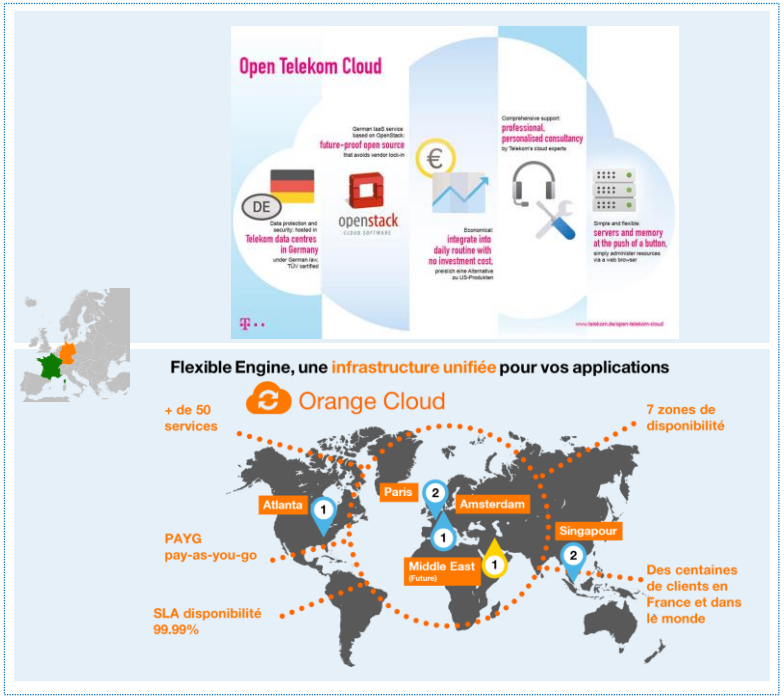
Flexible and service-driven

Cloud Computing
Public Clouds

- Amazon, Azure, Google Cloud, Oracle Cloud, VMWare, IBM, Alibaba Cloud, etc.
- Transforms datacenters into pools of resources
- Provides a management layer for controlling, automating, and efficiently allocating resources
- Adopts a self-service mode
- Enables developers to build cloud-aware applications via standard APIs



- Open Telekom Cloud and Orange Business Services offer public cloud services in Germany and France, respectively.

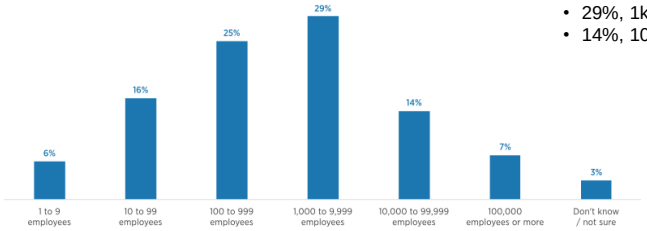


Cloud Computing
OpenStack

- Started by Rackspace and NASA (2010)
- Driven by the emergence of virtualization
- Rackspace wanted to rewrite its cloud servers offering
- NASA had published code for Nova, a Python-based cloud computing controller

Series	Status	Initial Release Date	Next Phase	EOL Date
Wallaby	Development	2021-04-14 *estimated* (schedule)	Maintained *estimated 2021-04-14*	
Victoria	Maintained	2020-10-14	Extended Maintenance *estimated 2022-04-18*	
Ussuri	Maintained	2020-05-13	Extended Maintenance *estimated 2021-11-12*	
Train	Maintained	2019-10-16	Extended Maintenance *estimated 2021-05-12*	
Stein	Maintained	2019-04-10	Extended Maintenance *estimated 2020-11-11*	
Rocky	Extended Maintenance (see note below)	2018-08-30	Unmaintained *TBD*	
Queens	Extended Maintenance (see note below)	2018-02-28	Unmaintained *TBD*	
Pike	Extended Maintenance (see note below)	2017-08-30	Unmaintained *TBD*	
Ocata	Extended Maintenance (see note below)	2017-02-22	Unmaintained *estimated 2020-06-04*	
Newton	End Of Life	2016-10-06		2017-10-25
Mitaka	End Of Life	2016-04-07		2017-04-10
Liberty	End Of Life	2015-10-15		2016-11-17
Kilo	End Of Life	2015-04-30		2016-05-02
Juno	End Of Life	2014-10-16		2015-12-07
Icehouse	End Of Life	2014-04-17		2015-07-02
Havana	End Of Life	2013-10-17		2014-09-30
Grizzly	End Of Life	2013-04-04		2014-03-29
Folsom	End Of Life	2012-09-27		2013-11-19
Essex	End Of Life	2012-04-05		2013-05-06
Diablo	End Of Life	2011-09-22		2013-05-06
Cactus	End Of Life	2011-04-15		
Bexar	End Of Life	2011-02-03		
Austin	End Of Life	2010-10-21		

What size organizations use OpenStack?

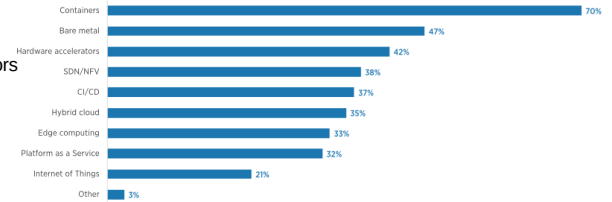


2018 results include survey responses among those who who logged at least one deployment and cannot be compared to prior years.
Sample size of 685.

- 25%, 100-1k employees
- 29%, 1k-10k employees
- 14%, 10k-100k employees

Which emerging technologies interest OpenStack users?

- Containers
- Base metal
- Hardware accelerators
- SDN/NFV
- CI/CD
- Hybrid Cloud
- Edge Computing
- PaaS



Results will not add to 100% as multiple responses were accepted.
2018 results include survey responses among those who who logged at least one deployment and cannot be compared to prior years.

What types of clouds are running OpenStack

- On-premises Private Cloud
- Public Cloud
- Off-premises Private Cloud

Sample size of 677.

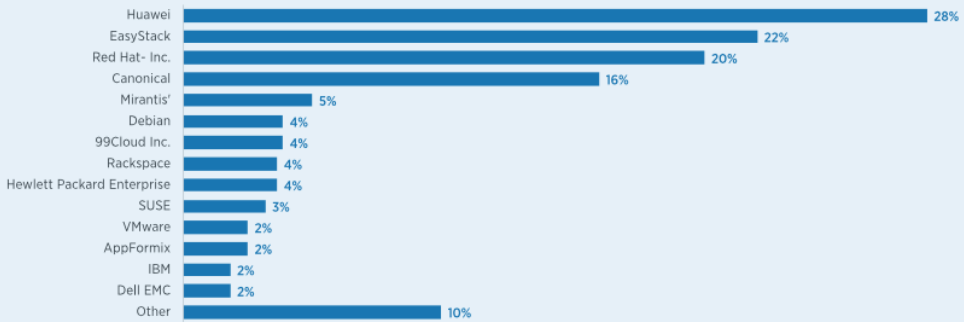
https://www.openstack.org/user-survey/2018-user-survey-report

Cloud Computing
OpenStack Community

- 1,500+ active participants
- 17 countries represented at Design Summit
- 60,000+ downloads
- Worldwide network of user groups (North America, South America, Europe, Asia and Africa)

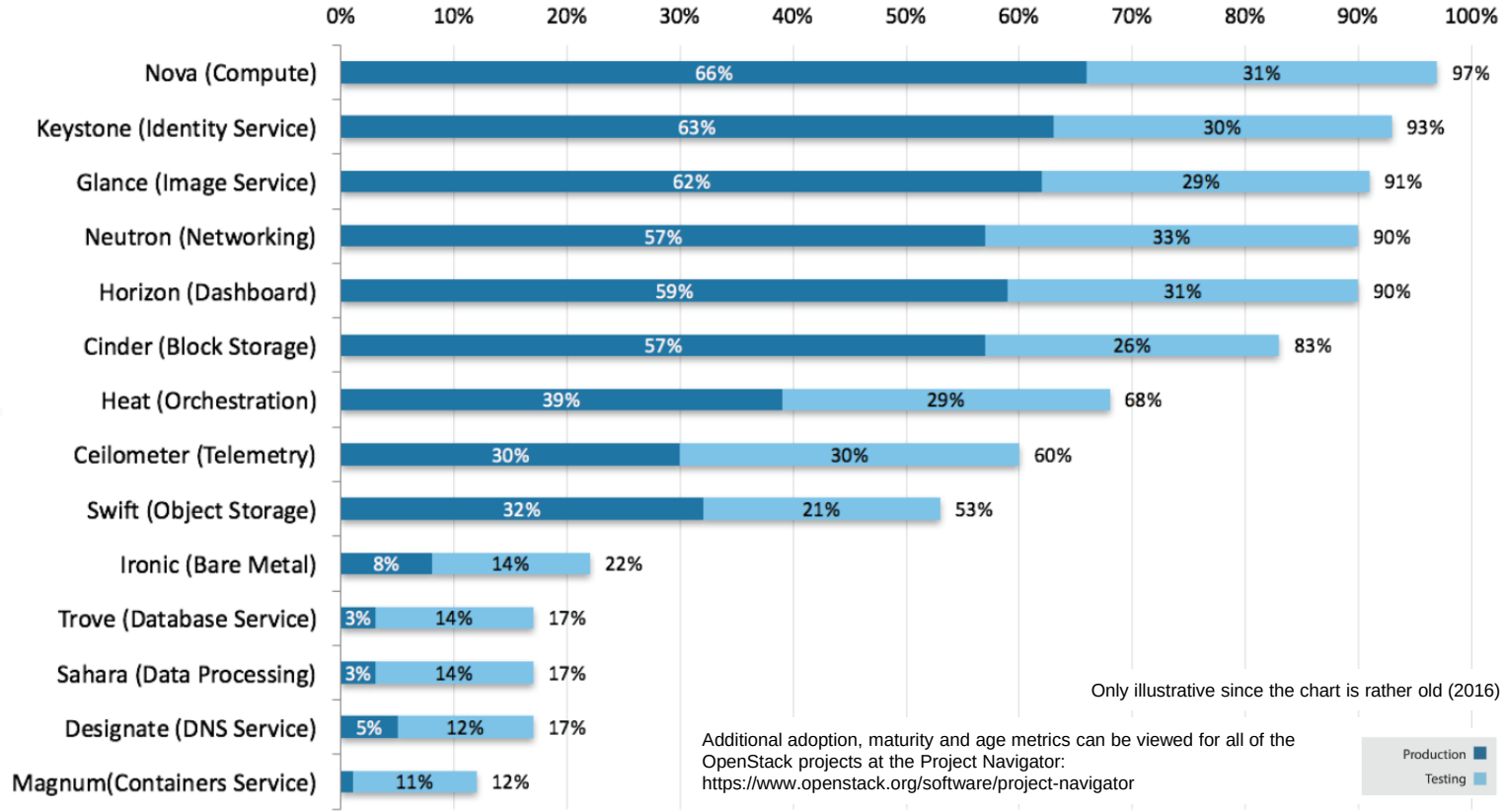
Key stat: 33% increase in community members YoY

Which vendor's OpenStack-related software products power user's clouds?



#	Company	Reviews
1	Red Hat	18232
2	Huawei	5348
3	SUSE	4650
4	Rackspace	2948
5	IBM	2644
	*independent	2458
6	Fujitsu	2258
7	NEC	2232
8	Intel	1574
9	AT&T	1344

OpenStack
Popular Services



Only illustrative since the chart is rather old (2016)

Additional adoption, maturity and age metrics can be viewed for all of the OpenStack projects at the Project Navigator:
https://www.openstack.org/software/project-navigator

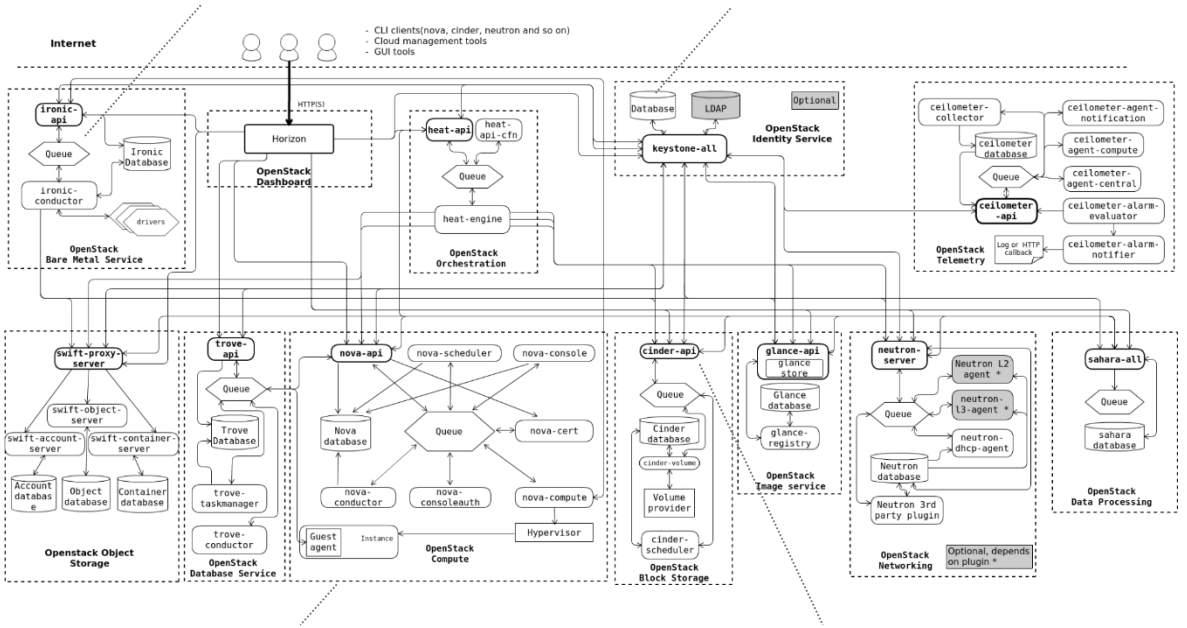
OpenStack User Survey: A snapshot of OpenStack users' attitudes and deployments. April 2016. (https://www.openstack.org/assets/survey/April-2016-User-Survey-Report.pdf). Fig. 4.6, Pag. 31.

OpenStack
Basic Design Tenets

1. **Scalability** and **elasticity** are the main goals
2. Any feature that limits our main goals must be optional
3. Everything should be **asynchronous**
   - *If you can't do something asynchronously, see #2*
4. All required components must be **horizontally scalable**
5. Always use **shared nothing architecture** or **sharding**
   - *If you can't share nothing/shard, see #2*
6. **Distribute everything**
   - *Especially logic. Move logic to where state naturally exists.*
7. Accept **eventual consistency** and use it where it is appropriate.
8. Test everything
   - *We require tests with submitted code. (We will help you if you need it)*

3 *Message queues provide asynchronous communication and coordination for distributed components*

5 **6** *Decentralized data management where each microservice encapsulates its own database*



8 *The system is divided into a number of smaller, individual and independent testable services*

4 *High availability is achieved by expecting that at least 2 servers are utilized to run each service. No single point of failure exists. Service can be replicated to increase scalability*

6 *Each service run as autonomous processes and communicate with one another through APIs*

OpenStack
Conceptual Architecture

OpenStack is comprised of several core projects that form a complete IaaS solution

- Horizon. Dashboard
- Keystone. Identity Service
- Neutron. Networking
- Cinder. Block Storage
- Nova. Compute
- Glance. Image Service
- Swift. Object Storage

1. Accessing the dashboard

2. Calling the identity service for authentication
3. Generating a token to be used for subsequent calls

4. Contacting the image service to list and retrieve a base image

5. Processing the request to the compute service API

6. Processing compute service calls to determine security groups and keys

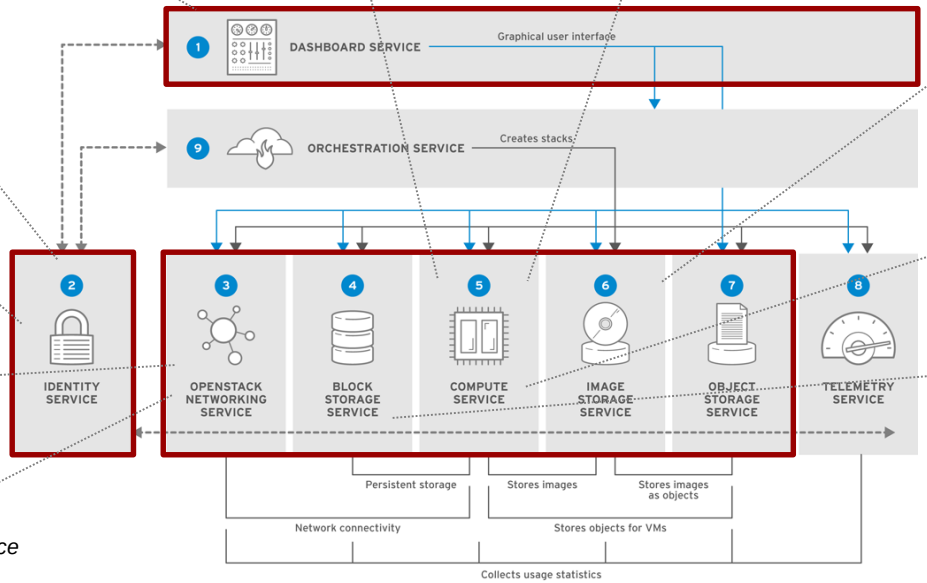
7. Calling the network service API to determine available networks

8. Choosing the hypervisor node by the compute scheduler service

9. Calling the block storage service API to allocate volume to the instance

10. Spinning up the instance in the hypervisor via the compute service API call

11. Calling the network service API to allocate network resources to the instance



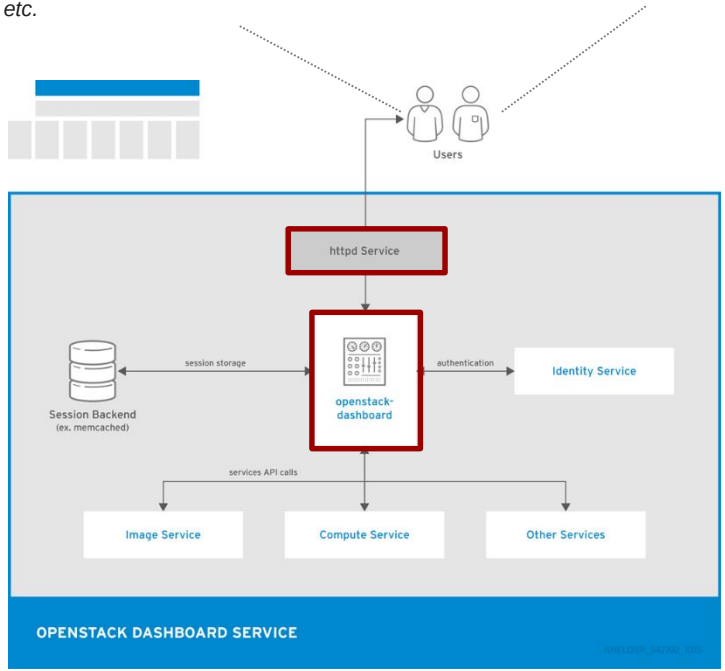
OpenStack Key Services
Dashboard Service

Key Capabilities

- Thin wrapper over APIs, no local state
- **Openstack-dashboard**. Django Web application that provides access to the dashboard from any Web browser.
- **HTTPservice.** Apache HTTP server (httpd service)
- Ships with three central dashboards, a “User Dashboard”, a “System Dashboard”, and a “Settings”
- Out-of-the-box support for all core OpenStack projects
  - Nova, Glace, Switch, Neutron
- Anyone can add a new component as a “first-class citizen”.
  - Follow design and style guide.
- Console Access

Provides a web based user interface to OpenStack services including Nova, Swift, Keystone, etc.

Provides users a self-service portal to manage their own resources



OpenStack Key Services
Identity Management Service

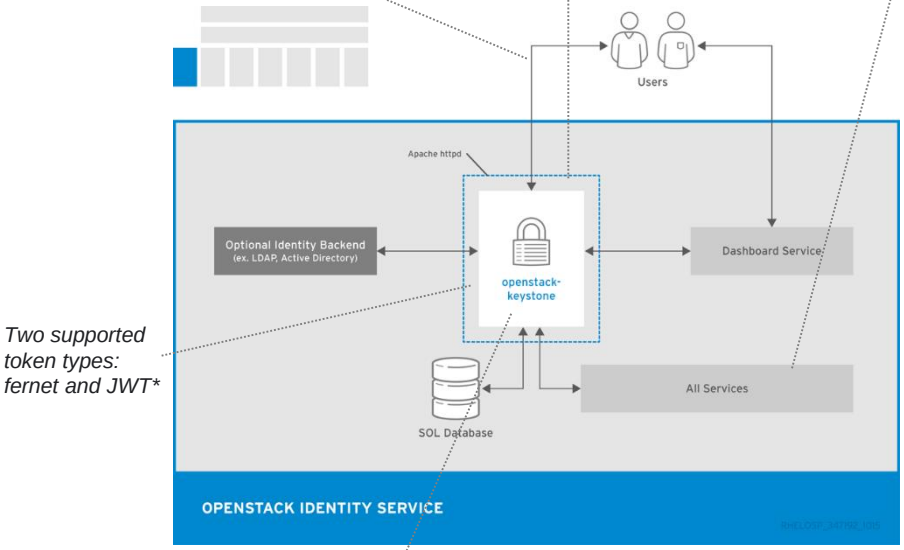
Key Capabilities

- By default, the Identity service uses a MariaDB back end for token, catalog, policy, and identity information.
- Identity service provides auth credential validation and data about Users, Tenants and Roles
- Tenant, or project, management. Tenants can be the user group, project, or organization.
- Role management. Roles determine the user permissions.
- Token service validates and manages tokens used to authenticate requests after initial credential verification
- Catalog service provides an endpoint registry used for endpoint discovery.
- Policy service provides a rule-based authorization engine and the associated rule management interface.
- REST-based APIs

The Identity service generates authentication tokens that enable clients to access OpenStack services REST APIs.

Clients obtain a token and the URL endpoints for other service APIs by supplying their credentials

The token is often passed within the structure of a larger context of an Identity service response



*Two supported token types: fernet and JWT**

A token can have a variable life span; however the default value for expiry is one hour

(*) https://jwt.io , https://cryptography.io/en/2.8/fernet/

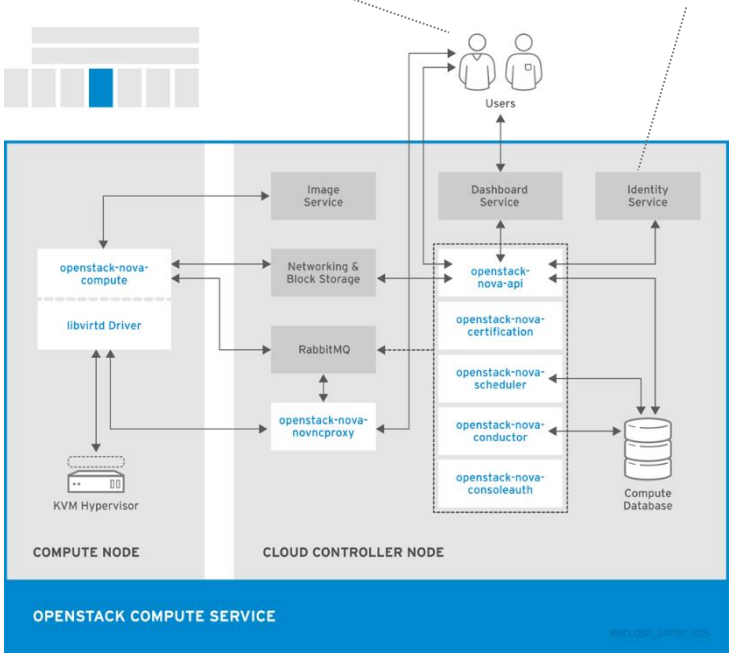
OpenStack Key Services
Compute Service

Key Capabilities

- Manage virtualized server resources
  - CPU/Memory/Disk/Network Interfaces
- API with rate limiting and authentication
- Distributed and asynchronous architecture
  - Massively scalable and highly available system
- Live guest migration
  - Move running guests between physical hosts
- Live VM management (instance)
  - Run, reboot, suspend, resize, terminate instances
- Security Groups
- Role Based Access Control (RBAC)
  - Ensure security by user, role and project
- Projects & Quotas
- VNC Proxy through web browser

Nova provides a way to provision compute instances (i.e., virtual machines or servers)

It requires the following additional services:
- *Keystone for identity and authentication*
- *Glance for compute images repository*
- *Neutron for virtual or physical networks*
- *Placement for tracking resources*



https://access.redhat.com/documentation/en/red-hat-openstack-platform/8/paged/architecture-guide/chapter-1-components
Egan Ford, OpenStack Technology Review & Demo, IBM

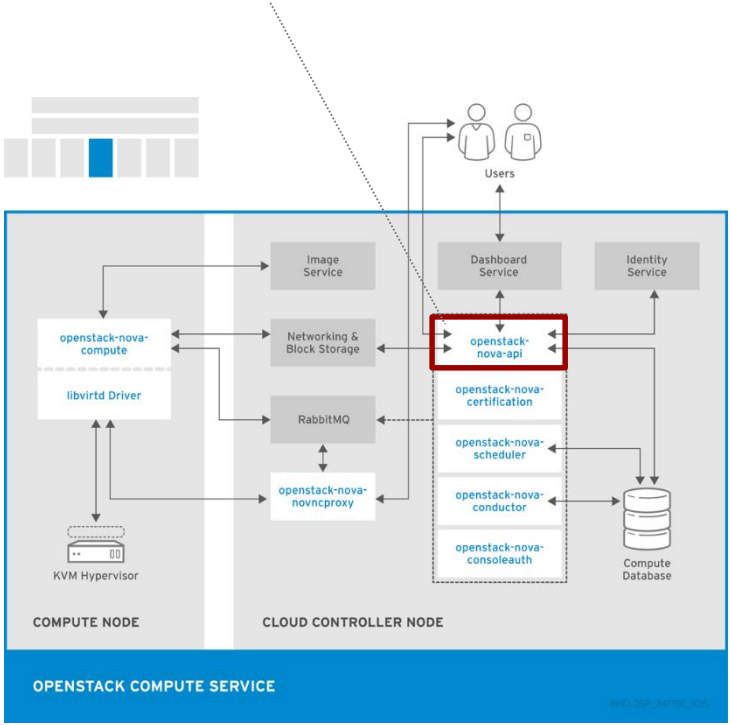
OpenStack Key Services
Compute Service: Nova-API

Key Capabilities

- APIs supported
  - OpenStack Compute API (REST-based)
    - Similar to RackSpace APIs
  - EC2 API (subset)
    - Can be excluded
  - Admin API (nova-manage)

- Robust extensions mechanism to add new capabilities

End users and components communicate with nova-api interface to create instances via the OpenStack API or EC2 API.



OpenStack Key Services
Compute Service: RabbitMQ and Database

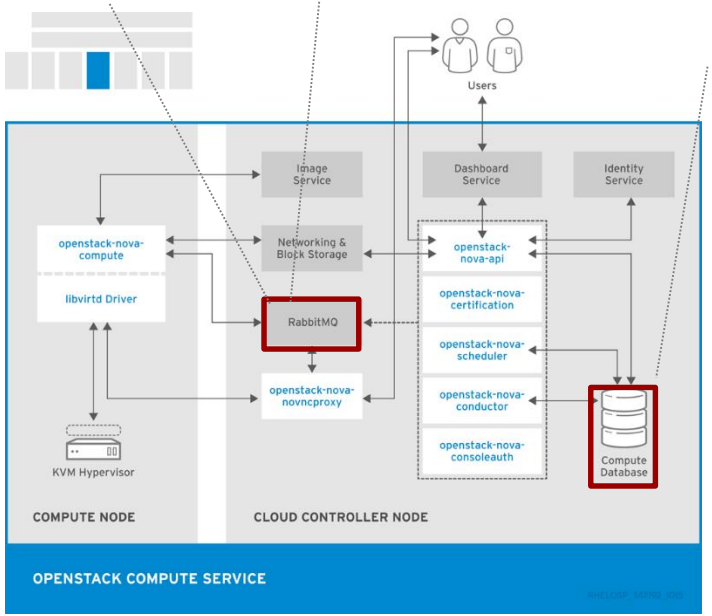
Key Capabilities

- Responsible for providing communications hub and managing data persistence
- RabbitMQ is default queue, MySQL DB
  - Documented HA methods
  - ZeroMQ implementation available to decentralize queue
- Single “cell” (1 Queue, 1 Database) typically scales from 500 – 1000 physical machines
  - Cells can be rolled up to support larger deployments
- Communications route through queue
  - API requests are validated and placed on queue
  - Workers listen to queues based on role or role + hostname
  - Responses are dispatched back through queue

Advantages: buffer requests, decoupling, unicast and group-based communication

Message Queue provides a central hub to pass messages between different services in an asynchronous way

Stores build-time and run-time states, including available instance types, instances in use, available networks…



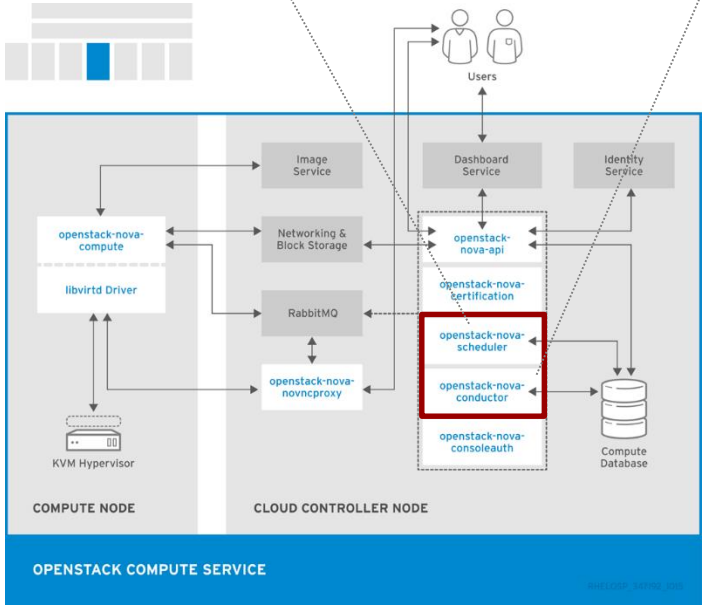
OpenStack Key Services
Compute Service: Scheduler

Key Capabilities

- Determines which physical hardware to allocate to a virtual resource
- Default scheduler uses a series of filters to reduce set of applicable hosts and uses costing functions to provide Weight
  - Not a focus point for OpenStack
  - Default implementation finds first fit
- Shorter the workload lifespan, less critical the placement decision
- If default does not work, often deployers have specific requirements and develop custom

nova-scheduler takes a VM instance's request from the queue and determines which compute host it should run on

nova-conductor prevents direct database access from the compute nodes to enhance database security



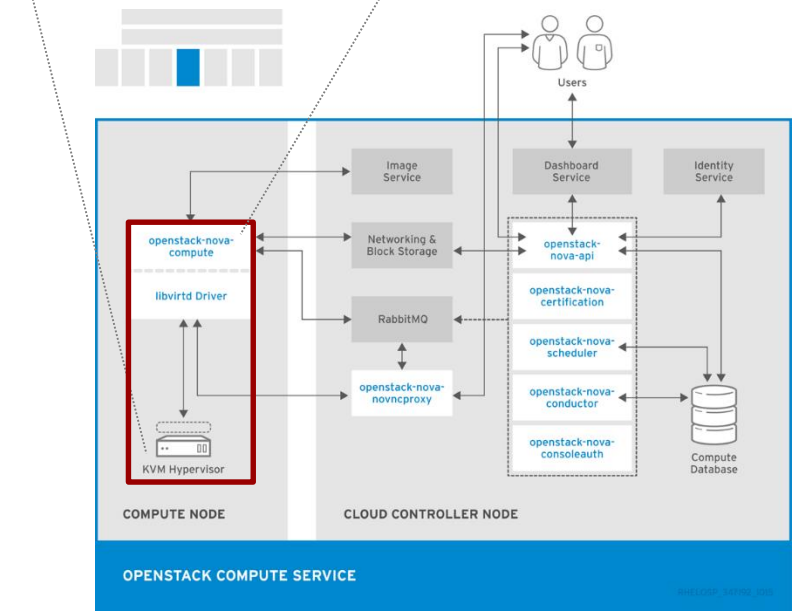
OpenStack Key Services
Compute Service: Node

Key Capabilities

- Responsible for managing all interactions with individual endpoints providing compute resource, e.g.
  - Attach iSCSI volume to physical host, map to guest as additional HDD
- Implementations direct to native hypervisor APIs
  - Avoids abstraction layers that bring least common denomination support
  - Enables easier exploitation of hypervisor differentiators
- Service instance runs on every physical compute node, helps to minimize failure domain
- Support for security groups that define firewall rules
- Support for
  - KVM, LXC, VMware ESX/ESXi (4.1 update 1), Xen (XenServer 5.5, Xen Cloud Platform), Hyper V

The majority of nova deployments use libvirt/kvm, but other compute drivers can be used

nova-compute is a daemon that creates and terminates VM instances via the hypervisor's APIs



OpenStack Key Services
Network Service

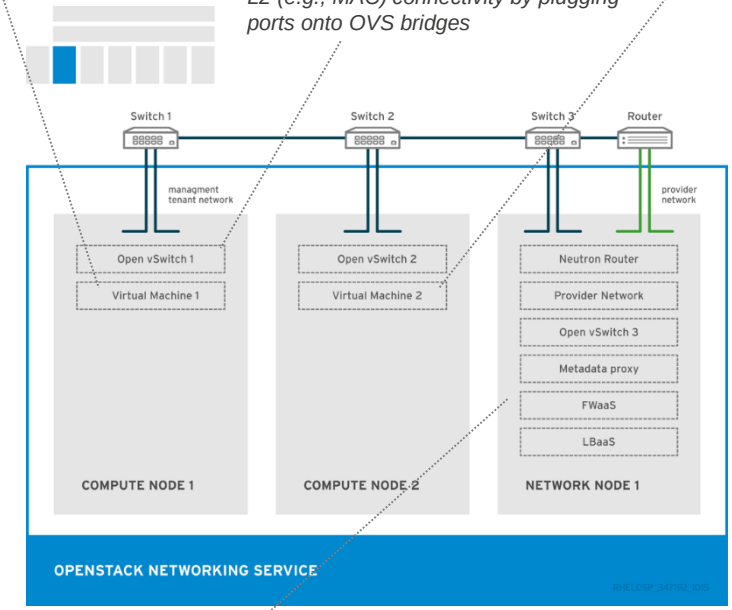
Key Capabilities

- Responsible for managing networks, ports, and attachments on infrastructure for virtual resources
- Create/delete tenant-specific L2 networks
- L3 support (Floating IPs, DHCP, routing)
- NAT including services such as FWaaS and LBaaS
- Attach / detach host to network
- Support for
  - Open vSwitch
  - OpenFlow (NEC & Floodlight controllers)
  - Cisco Nexus
  - Niciria

Network as a Service (NaaS) capability between interface devices that are managed by nova

Users can create their own networks and attach server interfaces to them

Open vSwitch (OVS) agent implements L2 (e.g., MAC) connectivity by plugging ports onto OVS bridges

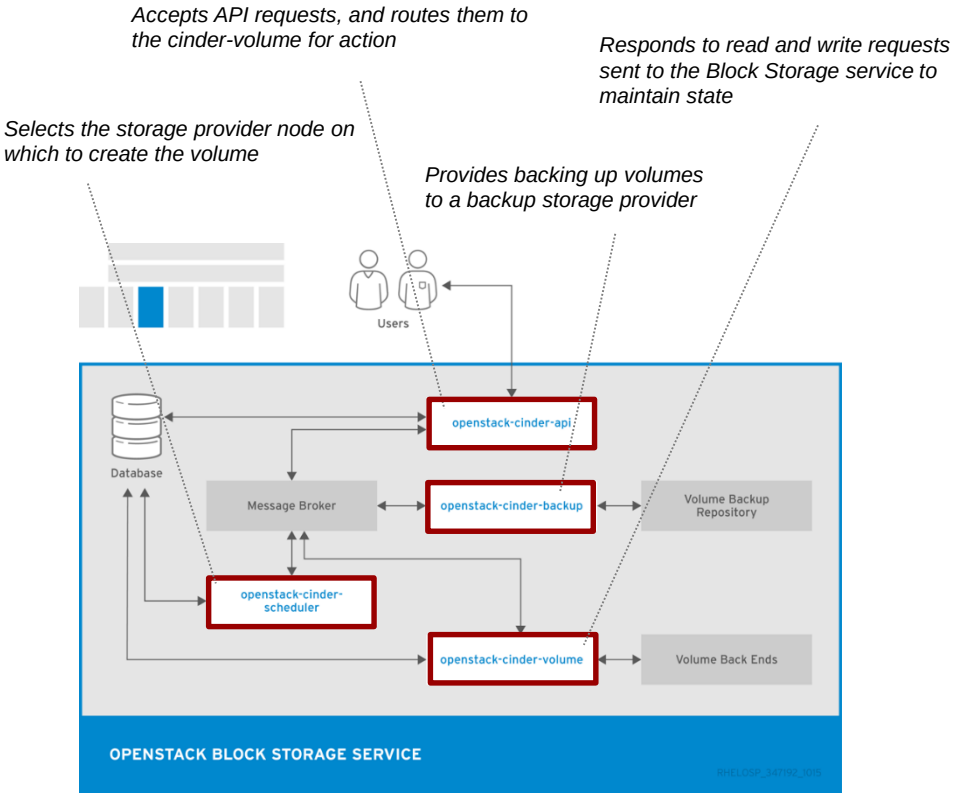


L3 (e.g., IP) agents run only on network nodes and provide routing and NAT services

OpenStack Key Services
Block Storage Service

Key Capabilities

- Responsible for managing lifecycle of volumes and exposing for attachment
- Structure is a copy of Compute (Nova), sharing same characteristics and structure in API server, scheduler, etc.
- Enables additional attached persistent block storage to virtual machines
- Support for booting virtual machines from nova-volume backed storage
- Allows multiple volumes to be attached per virtual machine
- Backs up a Block Storage volume to an external storage repository.
- Supports
  - NFS, Ceph distributed file system, etc.



OpenStack Key Services
Image Storage Service

Key Capabilities

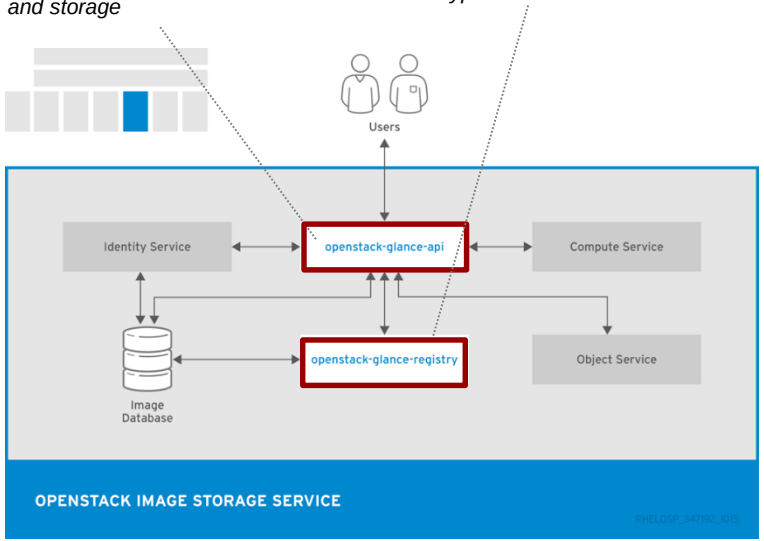
- REST-based APIs
- Query for information on public and private disk images
- Register new disk images
- Disk images can be stored in and delivered from a variety of stores (e.g. SoNFS, Swift)
- Supported formats
  - aki/ami/ari (Amazon kernel, ramdisk, or machine image)
  - iso (archive format for optical discs, such as CDs)
  - qcow2 (Qemu/KVM, supports Copy on Write)
  - raw (unstructured format)
  - vhd (Hyper-V, common for virtual machine monitors from vendors such as VMware, Xen, Microsoft, and VirtualBox)

Users can add new images or take a snapshot of an existing server for storage

```
$ openstack image list
+--------------------------------------+--------+--------+
| ID                                   | Name   | Status |
+--------------------------------------+--------+--------+
| 38047887-61a7-41ea-9b49-27987d5e8bb9 | cirros | active |
+--------------------------------------+--------+--------+
```

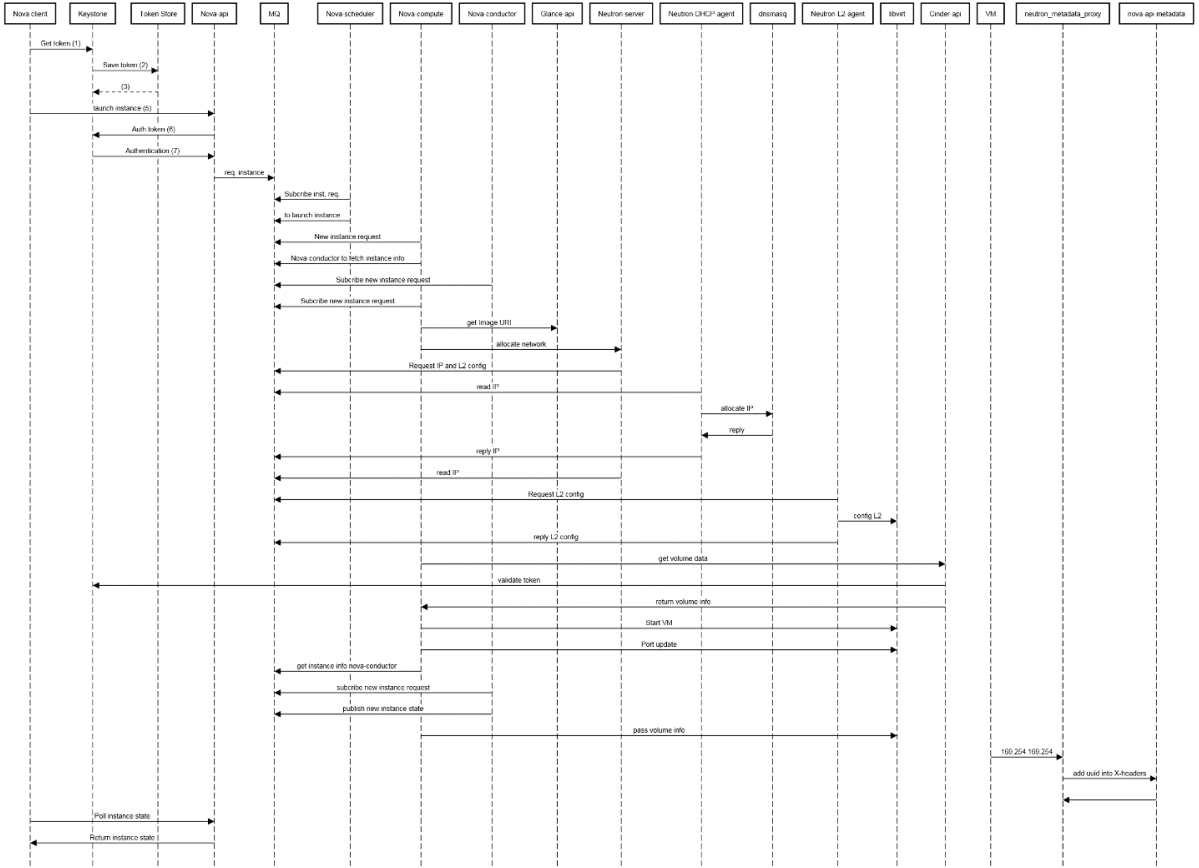
Accepts API calls for image discovery, retrieval, and storage

Stores, processes, and retrieves metadata about images. Metadata includes items such as size and type



OpenStack Operation
Server Creation Workflow

```
Nova client-> Keystone: Get token (1)
Keystone-> Token Store: Save token (2)
Token Store--> Keystone: (3)
Keystone-->-Nova client: Auth token (4)

Nova client-> Nova-api: launch instance (5)
Nova-api-> Keystone: Auth token (6)
Keystone->Nova-api: Authentication (7)

Nova-api->MQ: req. instance

Nova-scheduler->MQ: Subscribe inst. req.
Nova-scheduler->MQ: to launch instance

Nova-compute->MQ: New instance request
Nova-compute->MQ: Nova-conductor to fetch instance info
Nova-conductor->MQ: Subscribe new instance request

Nova-compute->MQ: Subscribe new instance request

Nova-compute-> Glance-api: get Image URI
Glance-api->-Nova-compute: Return image URI

Nova-compute->Neutron-server: allocate network
Neutron-server->MQ: Request IP and L2 config

Neutron-DHCP-agent->MQ: read IP
Neutron-DHCP-agent->dnsmasq: allocate IP
dnsmasq->Neutron-DHCP-agent: reply
Neutron-DHCP-agent->MQ: reply IP

Neutron-server->MQ:read IP
Neutron-L2-agent->MQ: Request L2 config
Neutron-L2-agent->libvirt: config L2
Neutron-L2-agent->MQ: reply L2 config
Neutron-server->-Nova-compute: net info

Nova-compute->Cinder-api: get volume data
Cinder-api->Keystone: validate token
Keystone-->-Cinder-api: updated auth headers with roles and acl
Cinder-api->Nova-compute: return volume info

Nova-compute->libvirt: Start VM
Nova-compute->libvirt: Port update
Nova-compute->MQ: get instance info nova-conductor
Nova-conductor->MQ: subcribe new instance request
Nova-conductor->MQ: publish new instance state

Nova-compute->libvirt: pass volume info

VM->neutron_metadata_proxy: 169.254.169.254
neutron_metadata_proxy->nova-api-metadata: add uuid into X-headers
nova-api-metadata->neutron_metadata_proxy:
neutron_metadata_proxy->-VM-instance: return metadata

Nova client->Nova-api: Poll instance state
Nova-api-> Nova client: Return instance state
```



https://sequencediagram.org/

OpenStack
DevStack Deployment

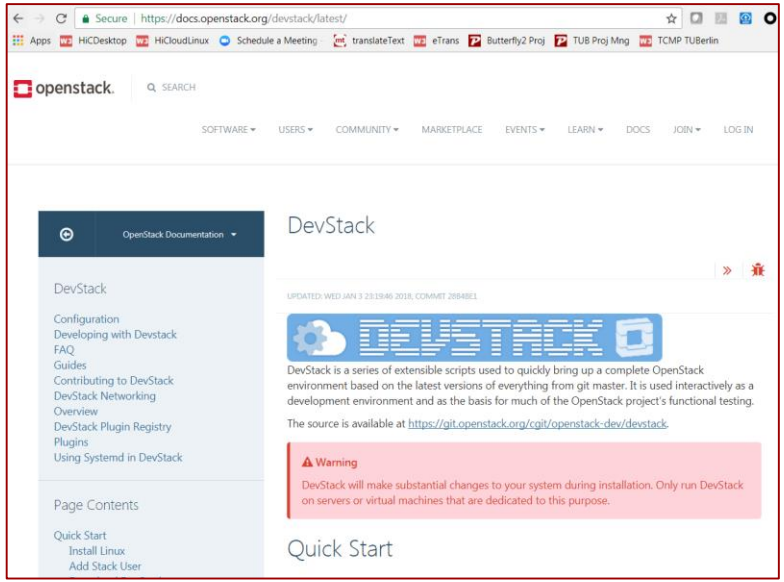
Create a VM at LRZ TU Munich 1

16 GB, image: Ubuntu_20G, 1 vCPU, (optional: access to the Internet)

- Generate ssh key pairs: lrz and lrz.pub
- Copy the public key lrz.pub in the field SSH_PUBLIC_KEY
- Take a note of the public IP

Access the LRZ VM

- ssh-keygen -R 141.40.254.130 (in case you rebuild your VM)
- ssh -i lrz -v root@141.40.254.130



Install Openstack 2

Follow the instructions at https://docs.openstack.org/devstack/latest/

Add Stack User

- $ sudo useradd -s /bin/bash -d /opt/stack -m stack
- $ echo "stack ALL=(ALL) NOPASSWD: ALL" | sudo tee /etc/sudoers.d/stack
- $ sudo su - stack

Download DevStack¶

- $ git clone https://git.openstack.org/openstack-dev/devstack
- $ cd devstack

Create a local.conf

Start the install

- ./stack.sh

Wait 15 - 20 minutes

Inspecting the services installed

- sudo systemctl status "devstack@*"

Inspecting the code of the services

- ls –al /opt/stack/

Inspecting the dashboard (Horizon)

- http://141.40.254.130/
- User Name: admin
- Password: admin (specified in the file local.conf)

https://docs.openstack.org/install-guide/get-started-logical-architecture.html

OpenStack
RDO Deployment

https://www.rdoproject.org/install/quickstart/

If you are using non-English locale make sure your /etc/environment is populated:

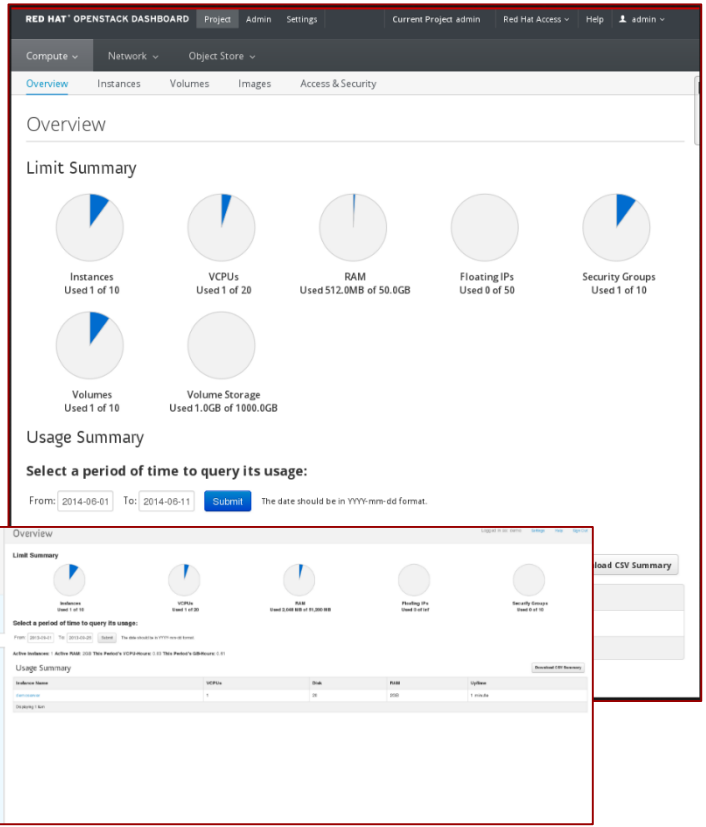
- LANG=en_US.utf-8
- LC_ALL=en_US.utf-8

On RHEL:

```
$ sudo yum install -y
https://www.rdoproject.org/repos/rdo-release.rpm
$ sudo yum update –y
$ sudo yum install -y openstack-packstack
$ sudo packstack --allinone
```

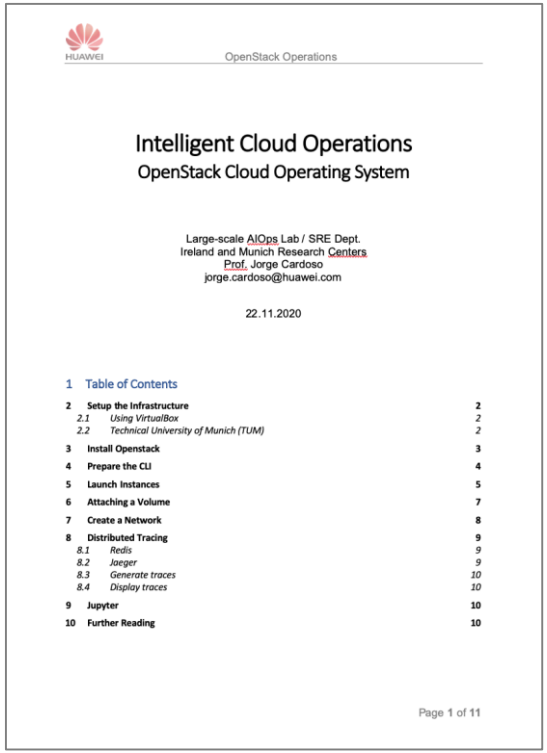
On CentOS:

```
$ sudo yum install -y centos-release-openstack-pike
$ sudo yum update –y
$ sudo yum install -y openstack-packstack
$ sudo packstack --allinone
```



OpenStack
Hands-on Exercises

- The following document contains the exercises for this lecture
  - Setup the Infrastructure
  - Install Openstack
  - Prepare the CLI
  - Launch Instances
  - Attaching a Volume
  - Create a Network



HUAWEI OpenStack Operations

Intelligent Cloud Operations

OpenStack Cloud Operating System

Large-scale AIOps Lab / SRE Dept.
Ireland and Munich Research Centers
Prof. Jorge Cardoso
jorge.cardoso@huawei.com

22.11.2020

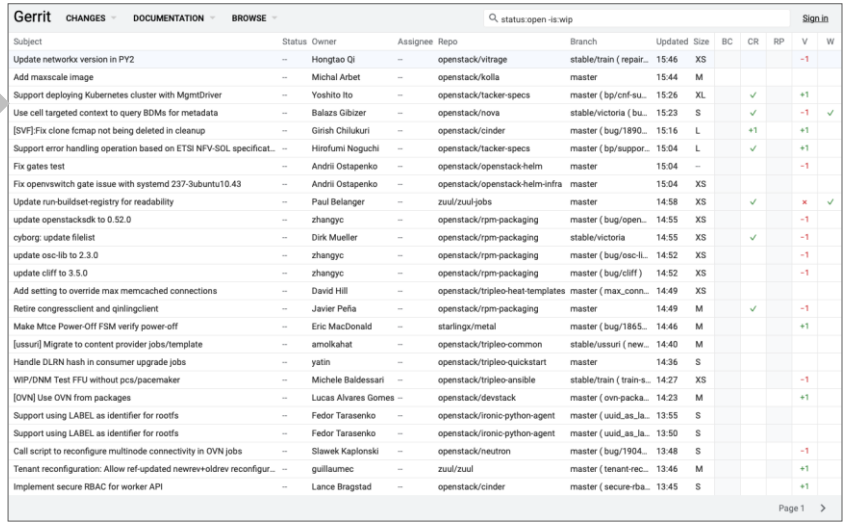
1 Table of Contents

Page 1 of 11

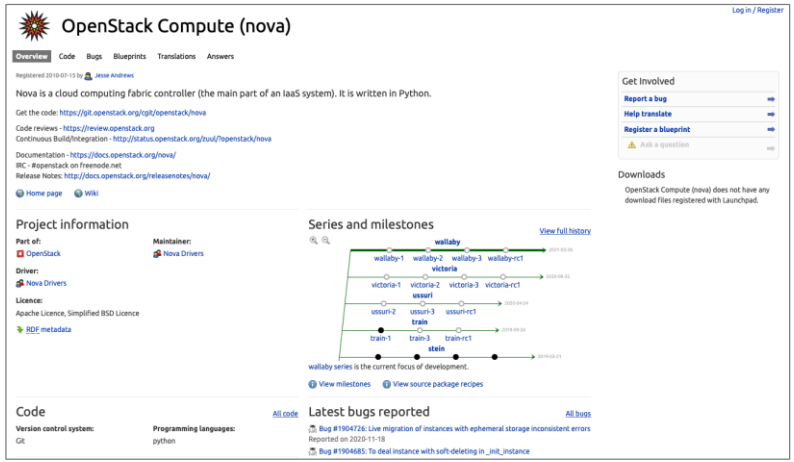
OpenStack Resources

- **Website:** http://openstack.org
- **Q&A**: https://ask.openstack.org/en/questions/
- **Wiki**: http://wiki.openstack.org/
- **Documentation**: http://docs.openstack.org/
- **Code Review**: https://review.openstack.org/
- **Mailing Lists**: http://wiki.openstack.org/MailingLists
- **Projects**: https://launchpad.net/openstack
- **Blogs**: http://planet.openstack.org
- **Real-time chat room**: #openstack and #openstack-dev on irc://freenode.net (443 users currently logged in)

Code Review



Projects



Thank you.

Bring digital to every person, home and organization for a fully connected, intelligent world.