第四章 传统云计算系统构成概述——OpenStack

2021年9月

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Contents

1 Design Philosophy

2 Industrial Standard

3 Modularization

4 Safety Considerations

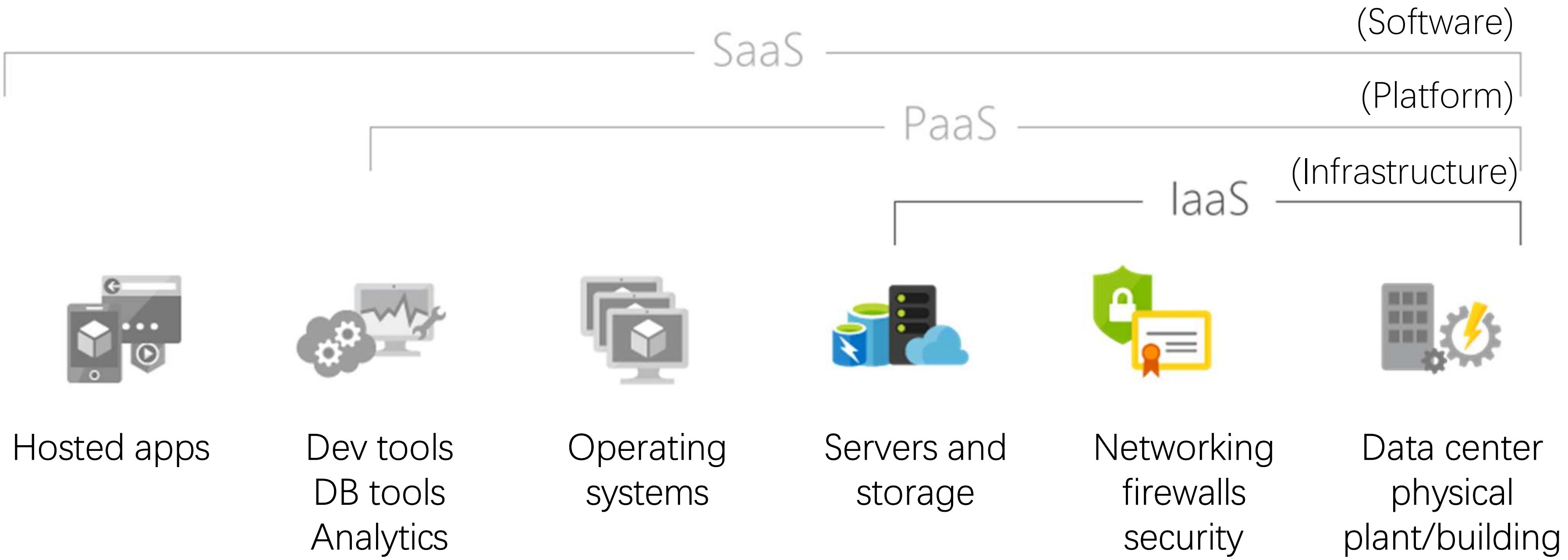
5 Deployment

Design Philosophy

- Cloud computing is a model for enabling:
  - Ubiquitous, on-demand access
  - A shared pool of configurable computing resources

- Massive scale
- Agility \ Elasticity
- Abstraction
- Automation
- Infinite capacity
- Converged API's
- Quick provisioning of resources
- On demand service
- Metering (billing)
- Pay as you go

“X as a service” (XaaS) model



SaaS (Software)

PaaS (Platform)

IaaS (Infrastructure)

Hosted apps

Dev tools
DB tools
Analytics

Operating
systems

Servers and
storage

Networking
firewalls
security

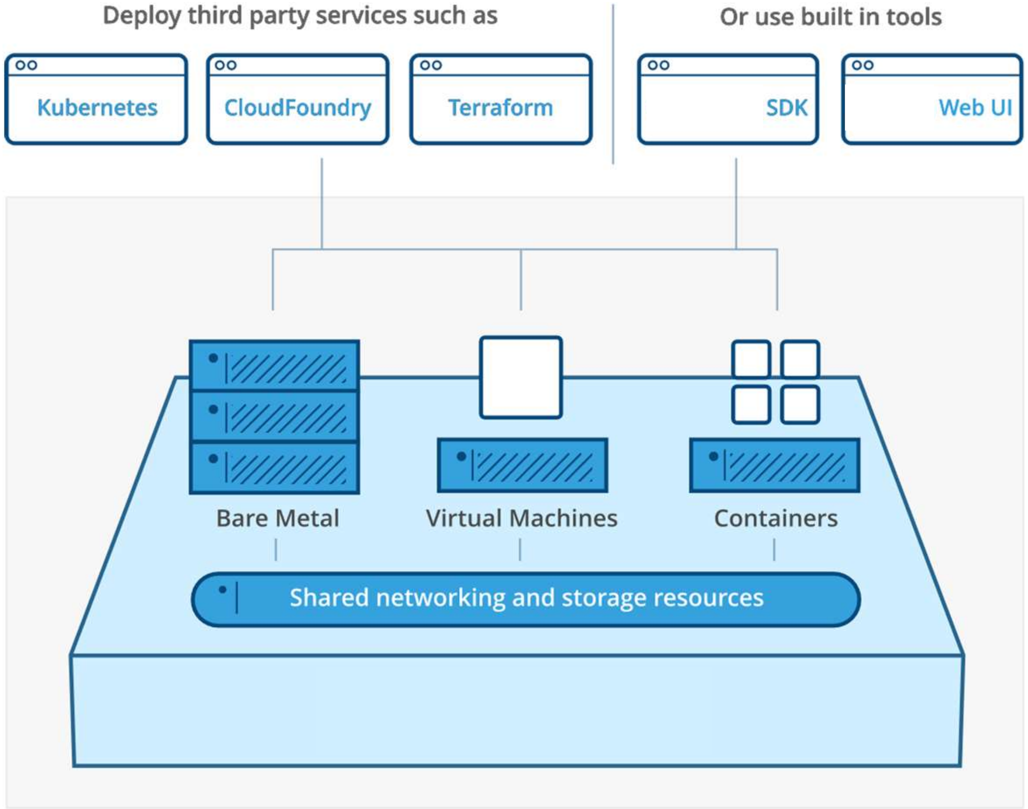
Data center
physical
plant/building

Other services:

FaaS (Function); BMaaS (Bare Metal); DBaaS (Database); AIaaS (Deep Learning)

IaaS: the basic building block

- VM on demand
- VM management
- Storage for VM and files
- Multi-tenancy
- Metering
- Orchestration



Virtualization v.s. Cloud

	Virtualization	Cloud
Definition	Technology	Methodology
Purpose	Create multiple simulated environments from 1 physical hardware system	Pool and automate virtual resources for on-demand use
Use	Deliver packaged resources to specific users for a specific purpose	Deliver variable resources to groups of users for a variety of purposes
Configuration	Image-based	Template-based
Lifespan	Years (long-term)	Hours to months (short-term)
Cost	High capital expenditures (CAPEX), low operating expenses (OPEX)	Private cloud: High CAPEX, low OPEX Public cloud: Low CAPEX, high OPEX
Scalability	Scale up	Scale out
Workload	Stateful	Stateless
Tenancy	Single tenant	Multiple tenants

Public / Private / Hybrid cloud

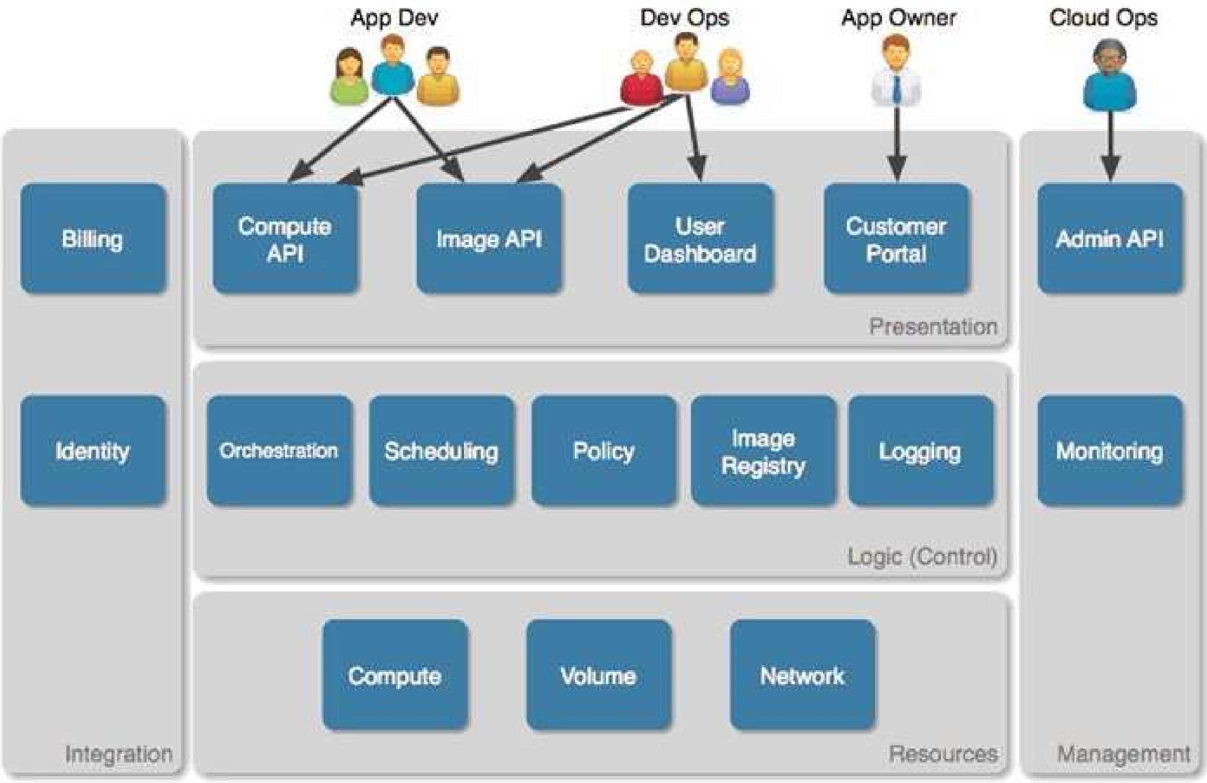
Public Cloud	Private Cloud	Hybrid Cloud
No maintenance costs	Dedicated, secure	Policy-driven deployment
High scalability, flexibility	Regulation compliant	High scalability, flexibility
Reduced complexity	Customizable	Minimal security risks
Flexible pricing	High scalability	Workload diversity supports high reliability
Agile for innovation	Efficient	Improved security

Public / Private / Hybrid cloud

Public Cloud	Private Cloud	Hybrid Cloud
Potential for high TCO	Expensive with high TCO	Potential for high TCO
Decreased security and availability	Minimal mobile access	Compatibility and integration
Minimal control	Limiting infrastructure	Added complexity

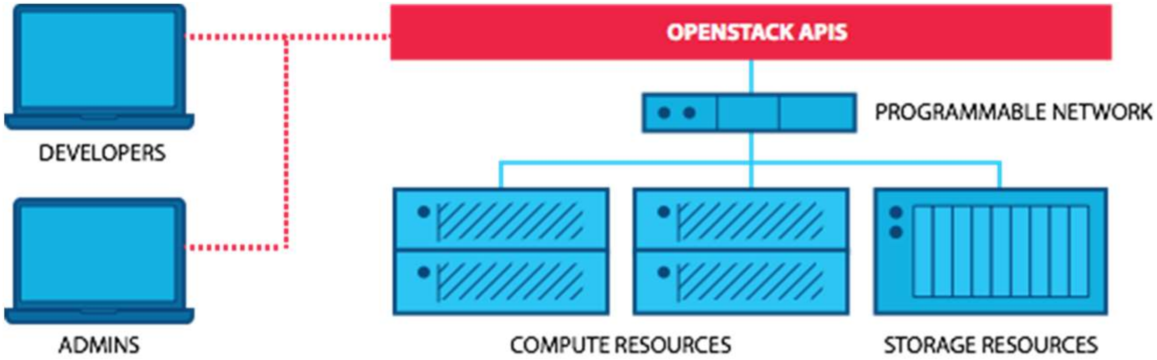
Cloud platform architecture

- Presentation layer: components interact with users
- Logic layer: control, deployment, scheduling, rules, registers, logging

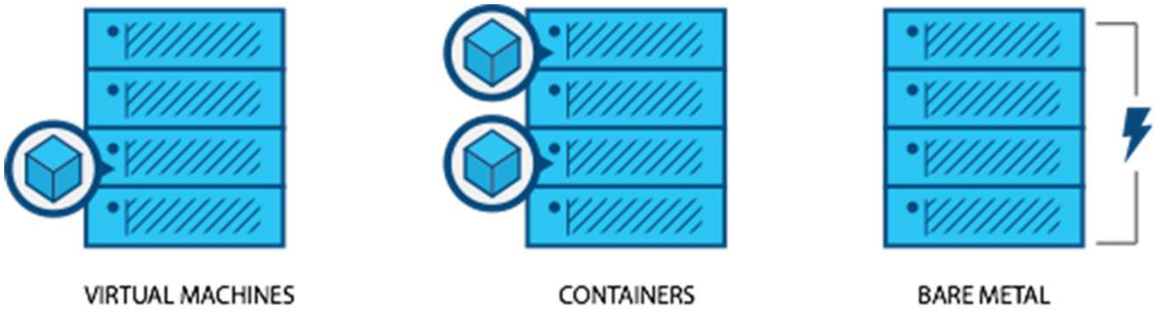


OpenStack intro

- Programmable infrastructure that lays a common set of APIs on top of compute, network, and storage resources.

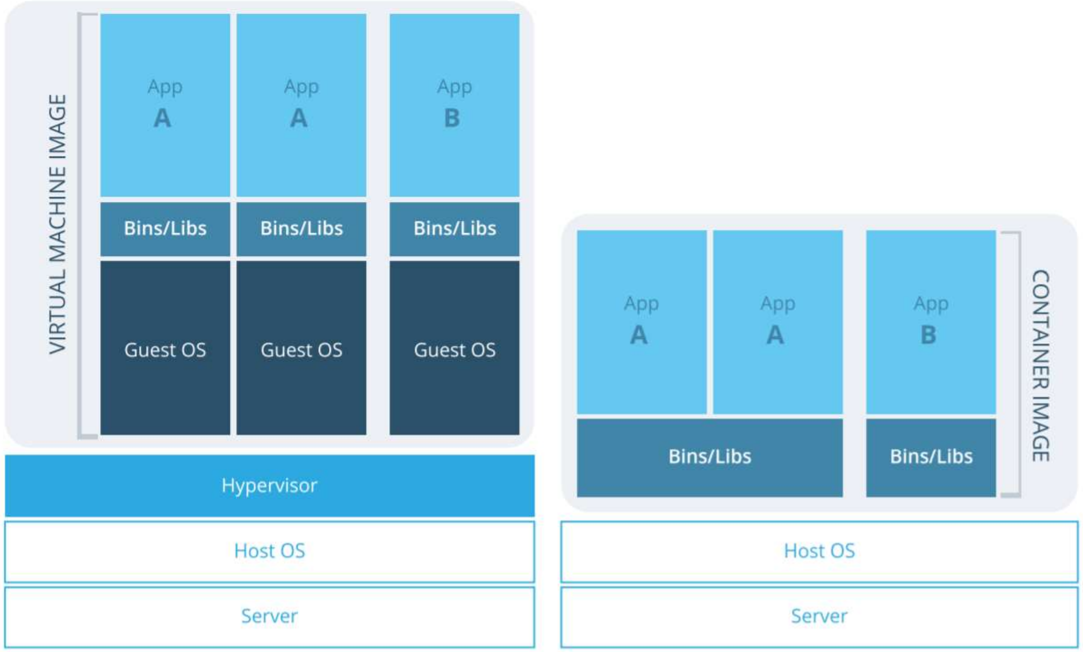


- One platform for VM, containers, and bare metal



Container and VM

- Container: lightweight, high packing density, fewer resource consumption, migrate easily
  - Potential security risks
- VM: Isolated, hardware virtualization, take up more resources



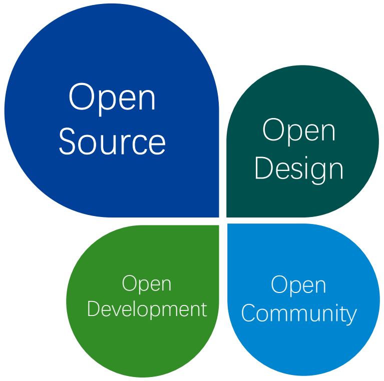
OpenStack intro

- Business drivers:

# 1 – Avoid vendor lock-in

# 2 – Accelerate innovation

# 3 – Operational efficiency



81,000+
MEMBERS

670+
ORGANIZATIONS

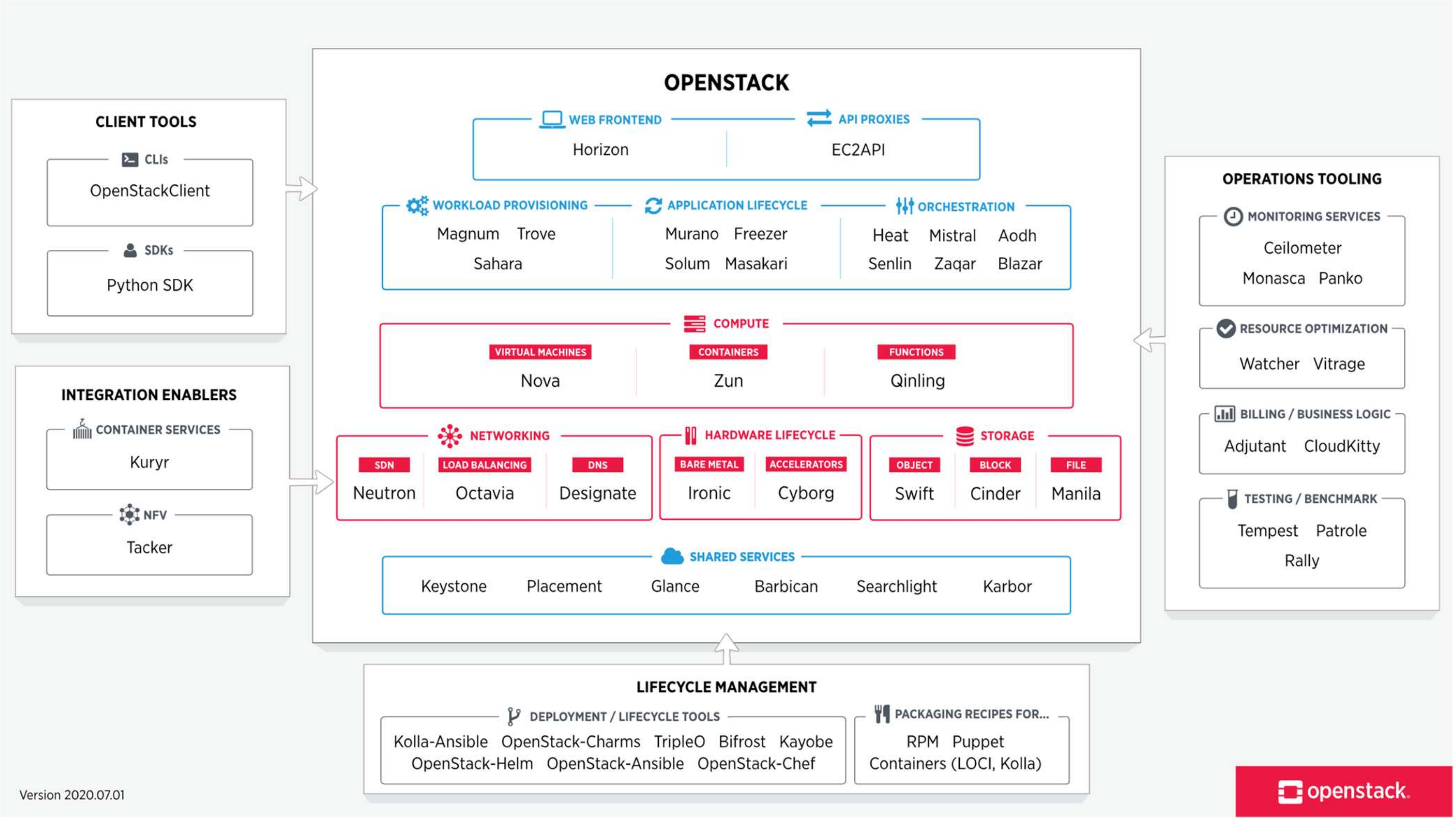
Retail / E-commerce — BEST BUY, Walmart, ebay, overstock

Energy and manufacturing — STATE GRID CORPORATION OF CHINA, VW

Financial — TD Bank, PayPal, UnionPay 银联, AMERICAN EXPRESS, CommonwealthBank

Telecom,Insurance,Entertainment, Acedemic, Research, ……
See more at: openstack.org/user-stories

OpenStack landscape



OPENSTACK

WEB FRONTEND
Horizon

API PROXIES
EC2API

WORKLOAD PROVISIONING
Magnum Trove
Sahara

APPLICATION LIFECYCLE
Murano Freezer
Solum Masakari

ORCHESTRATION
Heat Mistral Aodh
Senlin Zaqar Blazar

COMPUTE
- VIRTUAL MACHINES: Nova
- CONTAINERS: Zun
- FUNCTIONS: Qinling

NETWORKING
- SDN: Neutron
- LOAD BALANCING: Octavia
- DNS: Designate

HARDWARE LIFECYCLE
- BARE METAL: Ironic
- ACCELERATORS: Cyborg

STORAGE
- OBJECT: Swift
- BLOCK: Cinder
- FILE: Manila

SHARED SERVICES
Keystone Placement Glance Barbican Searchlight Karbor

CLIENT TOOLS

CLIs
OpenStackClient

SDKs
Python SDK

INTEGRATION ENABLERS

CONTAINER SERVICES
Kuryr

NFV
Tacker

OPERATIONS TOOLING

MONITORING SERVICES
Ceilometer
Monasca Panko

RESOURCE OPTIMIZATION
Watcher Vitrage

BILLING / BUSINESS LOGIC
Adjutant CloudKitty

TESTING / BENCHMARK
Tempest Patrole
Rally

LIFECYCLE MANAGEMENT

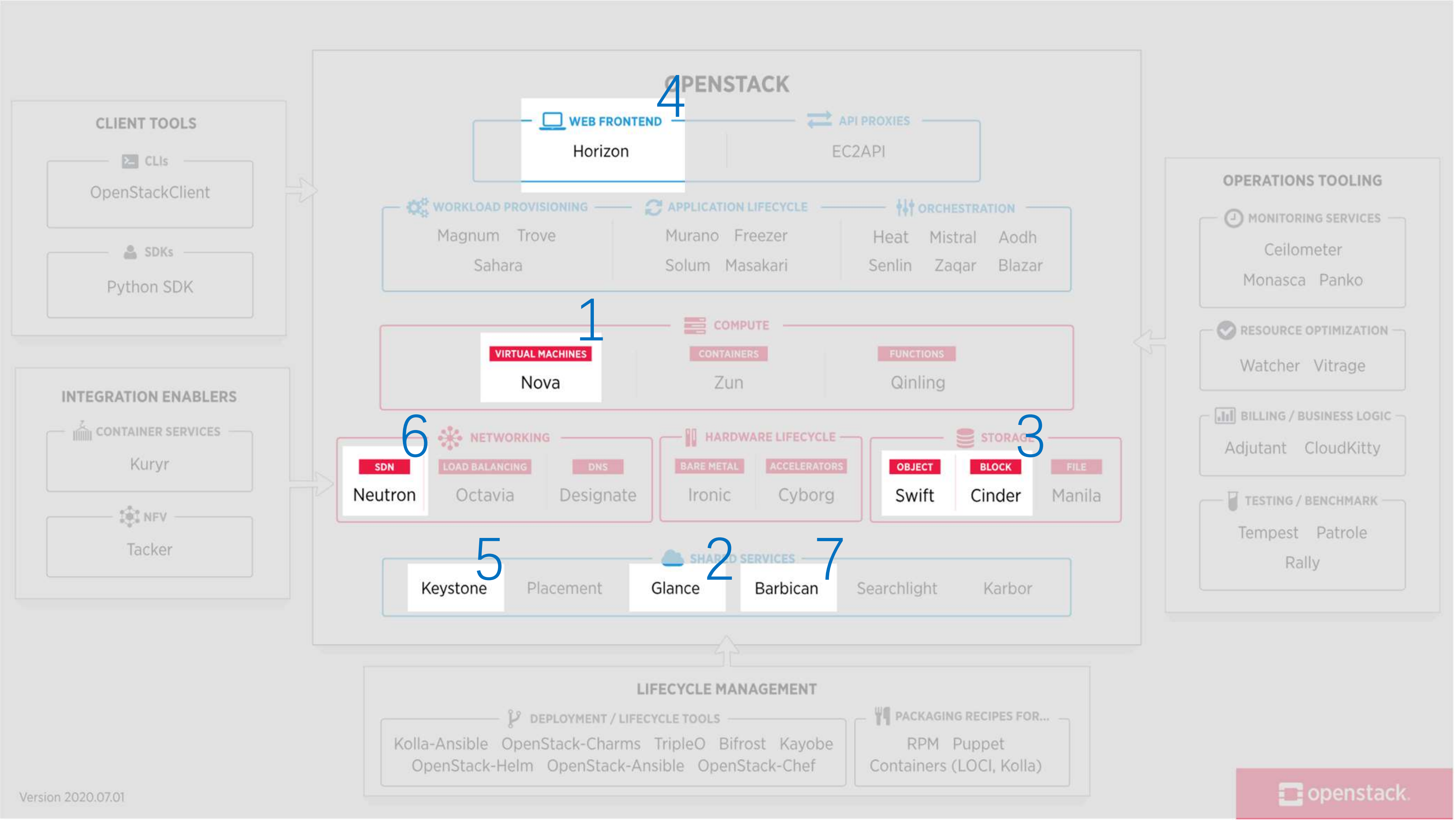
DEPLOYMENT / LIFECYCLE TOOLS
Kolla-Ansible OpenStack-Charms TripleO Bifrost Kayobe
OpenStack-Helm OpenStack-Ansible OpenStack-Chef

PACKAGING RECIPES FOR...
RPM Puppet
Containers (LOCI, Kolla)

Version 2020.07.01

openstack.

OpenStack landscape



OPENSTACK

WEB FRONTEND 4
Horizon

API PROXIES
EC2API

WORKLOAD PROVISIONING
Magnum Trove
Sahara

APPLICATION LIFECYCLE
Murano Freezer
Solum Masakari

ORCHESTRATION
Heat Mistral Aodh
Senlin Zaqar Blazar

COMPUTE
1 VIRTUAL MACHINES
Nova
CONTAINERS
Zun
FUNCTIONS
Qinling

6 NETWORKING
SDN
Neutron
LOAD BALANCING
Octavia
DNS
Designate

HARDWARE LIFECYCLE
BARE METAL
Ironic
ACCELERATORS
Cyborg

STORAGE 3
OBJECT
Swift
BLOCK
Cinder
FILE
Manila

SHARED SERVICES
5 Keystone
Placement
2 Glance
7 Barbican
Searchlight
Karbor

CLIENT TOOLS
CLIs
OpenStackClient
SDKs
Python SDK

INTEGRATION ENABLERS
CONTAINER SERVICES
Kuryr
NFV
Tacker

OPERATIONS TOOLING
MONITORING SERVICES
Ceilometer
Monasca Panko
RESOURCE OPTIMIZATION
Watcher Vitrage
BILLING / BUSINESS LOGIC
Adjutant CloudKitty
TESTING / BENCHMARK
Tempest Patrole
Rally

LIFECYCLE MANAGEMENT
DEPLOYMENT / LIFECYCLE TOOLS
Kolla-Ansible OpenStack-Charms TripleO Bifrost Kayobe
OpenStack-Helm OpenStack-Ansible OpenStack-Chef
PACKAGING RECIPES FOR...
RPM Puppet
Containers (LOCI, Kolla)

Version 2020.07.01

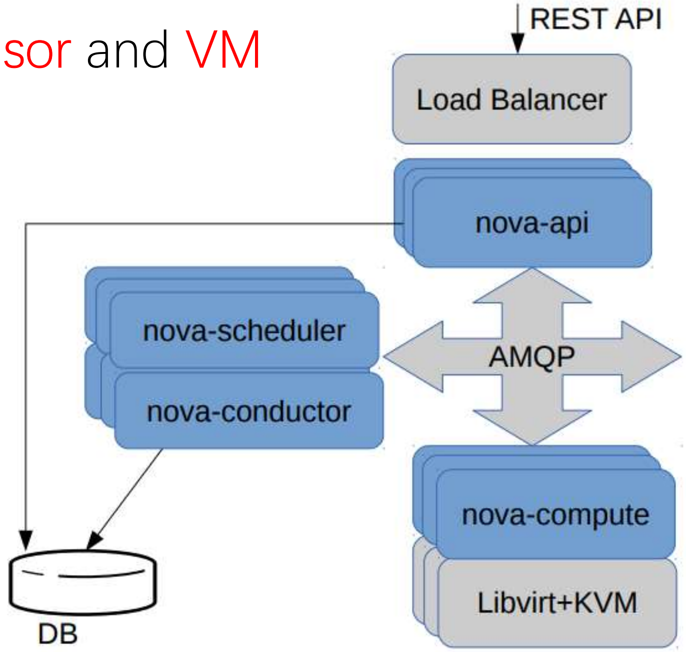
openstack.

Nova: compute resources

- Responsible for managing compute resources
- Nova is virtualization agnostic:
  - Libvirt (KVM, QEMU, Xen, LXC), XenAPI, Hyper-V, Vmware ESX, PowerVM, etc.
- Provides massively scalable, on demand, self service access to compute resources.
- Features:
  - VM scheduling by defining drivers that interact with underlying virt mechanism
  - Authenticated instance and database access
  - Libvirt driver libvirtd support that uses KVM as the hypervisor

Nova components

- nova-api: receives HTTP requests, converts commands, and call other components via message queue or HTTP
- nova-scheduler: decides which host gets each instance
- nova-conductor: handles coordination (build/resize), acts as DB proxy
- nova-compute: manages comm. with hypervisor and VM



Glance: image service

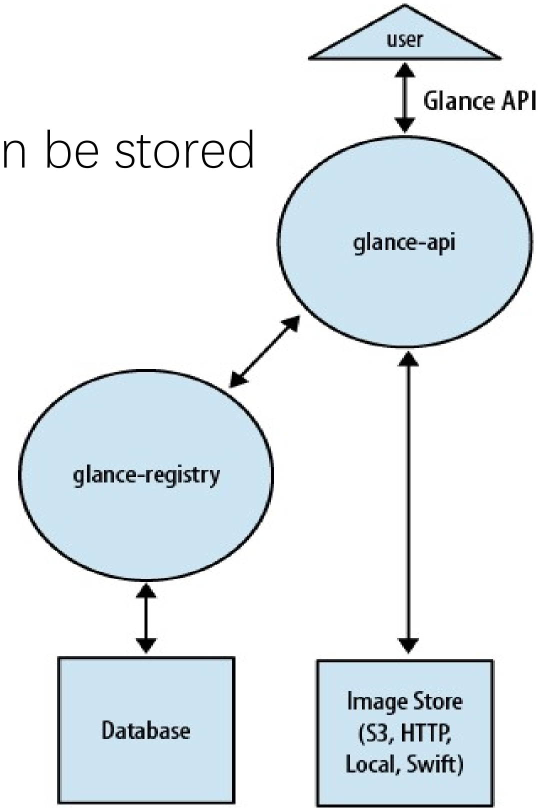
- Responsible for managing VM images
- Provides an API for disk and service image management and registration
- Supports multiple image formats:
  - ISO
  - QCOW2 (for QEMU), Raw (for QEMU/KVM and Xen)
  - VDI (for VirtualBox), VHD (for Hyper-v), VMDK (for Vmware)
  - AKI, AMI, ARI (for Amazon, including kernel, machine, ramdisk images)
  - OVF (for Open Virtualization Format)
- Supports image conversion: qemu-img

```
$ qemu-img convert -f raw -O qcow2 image.img image.qcow2
```

Glance components

- glance-api: accepts image API calls
- glance-registry: stores, processes and retrieves image metadata
- Database: stores image metadata
- Image Store: variety of locations where an image can be stored

Image status	
Queued	Upload not finished
Saving	Uploading image
Active	Image is fully available
Killed	Upload error occurred
Deleted	Image is no longer available
Pending_delete	Non-recoverable image



Cinder: block storage

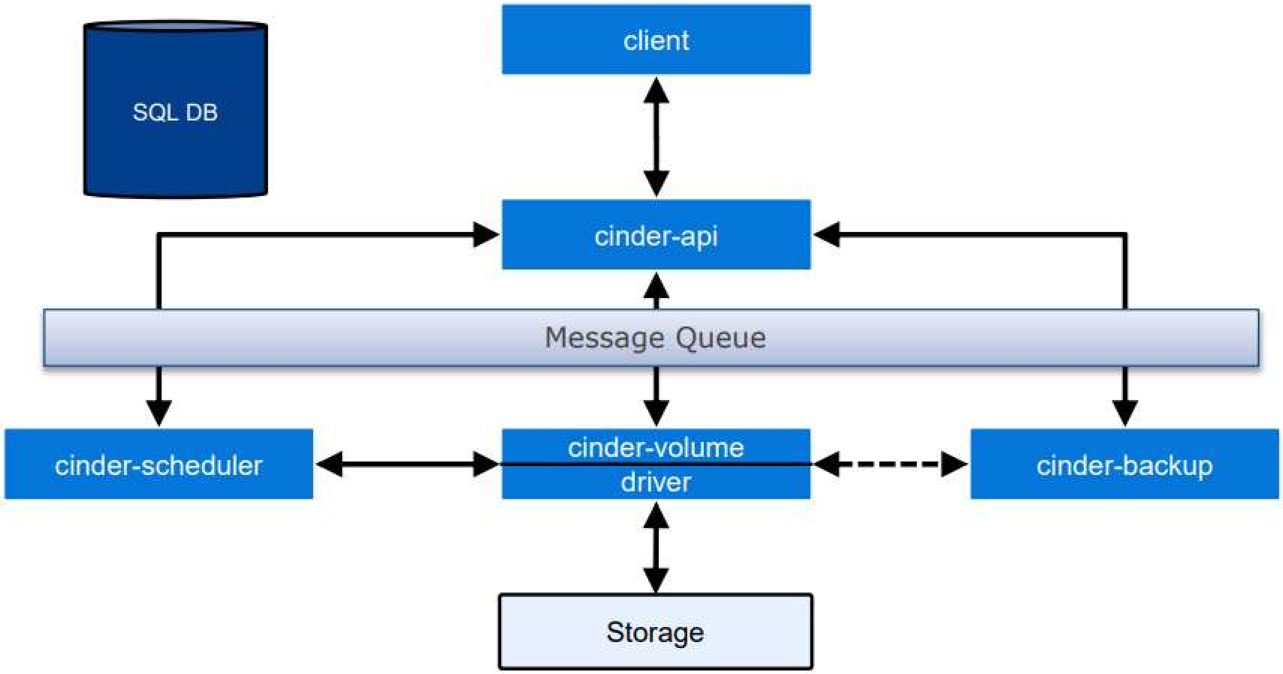
- Responsible for block device provisioning of VMs
- Provides an API for various storage array vendors to manage their block device and translate commands between Nova and other services
- Best used for performance-sensitive scenarios, such as database storage or expandable file systems
- Features:
  - Volumes, persistent R/W Block Storage devices
  - Snapshots, can be used to create a new instance
  - Backups, an archived copy of a volume

Cinder & Swift: block & object store

	BLOCK Cinder	OBJECT Swift
Objectives	• Storage for running VM disk volumes on a host • Ideal for perf. apps • Enables Amazon EBS-like service	• Ideal for cost effective, scale-out storage • Fully distributed, API-accessible • Ideal for backup, archiving, data retention • Enables Dropbox-like service
Workloads	• High change content • Smaller, random R/W • Higher / Bursty IO	• More static content • Larger, sequential R/W • Lower IOPS

Cinder components

- cinder-api: Authenticates and routes requests
- cinder-scheduler: Scheduling/routing volume requests to the service
- cinder-volume: Managing block storage devices

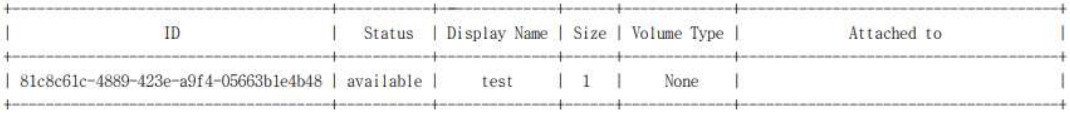


How it works?

# cinder create --display_name test 1

- ➢ Creates an Logic Volume into the Volume Group

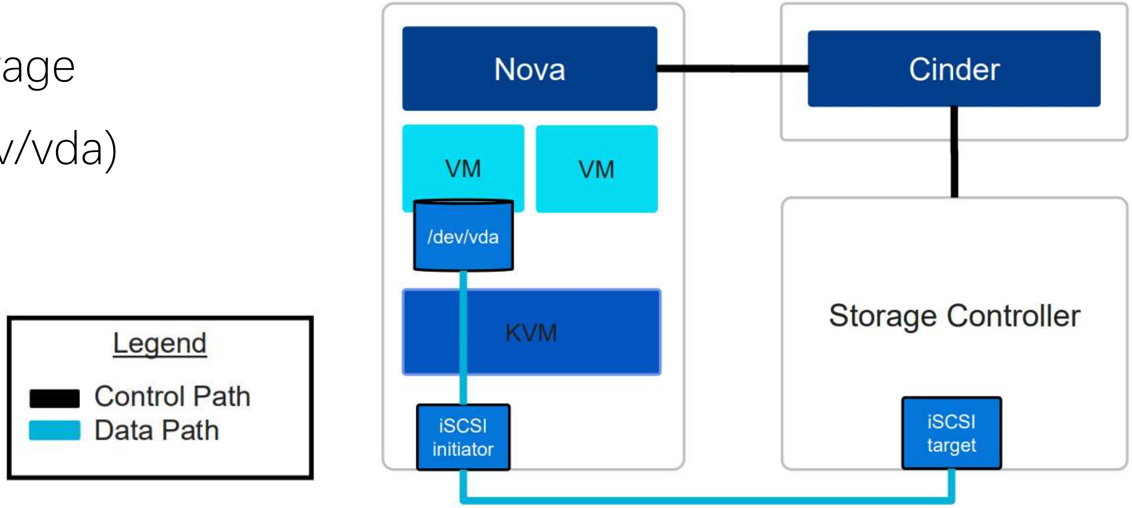
# cinder list



```
+--------------------------------------+-----------+--------------+------+-------------+--------------------------------------+
|                  ID                  |   Status  | Display Name | Size | Volume Type |             Attached to              |
+--------------------------------------+-----------+--------------+------+-------------+--------------------------------------+
| 81c8c61c-4889-423e-a9f4-05663b1e4b48 | available |     test     |  1   |     None    |                                      |
+--------------------------------------+-----------+--------------+------+-------------+--------------------------------------+
```

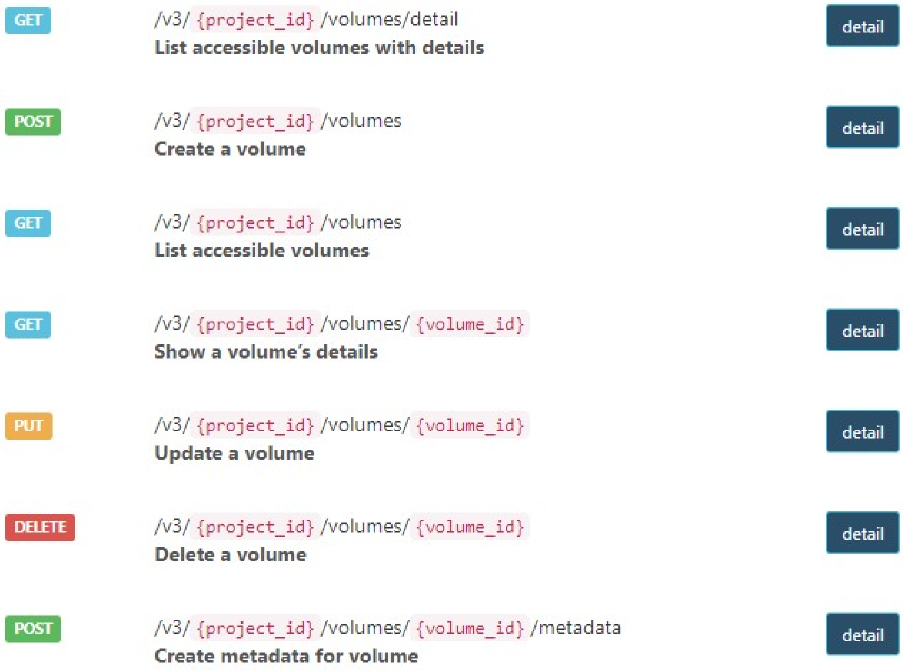
# nova volume-attach vm1 81c8c61c-4889-423e-a9f4-05663b1e4b48 /dev/vda

- ➢ Creates a unique iSCSI IQN exposed to the compute node
- ➢ Compute node has an active iSCSI session
- ➢ Libvirt uses the local storage
- ➢ VM gets a new disk (/dev/vda)



Cinder APIs

- Volume types / actions / extension / snapshots / transfer / backups
- Groups creation / replication/ snapshots / types
- Quota / QoS, and more……



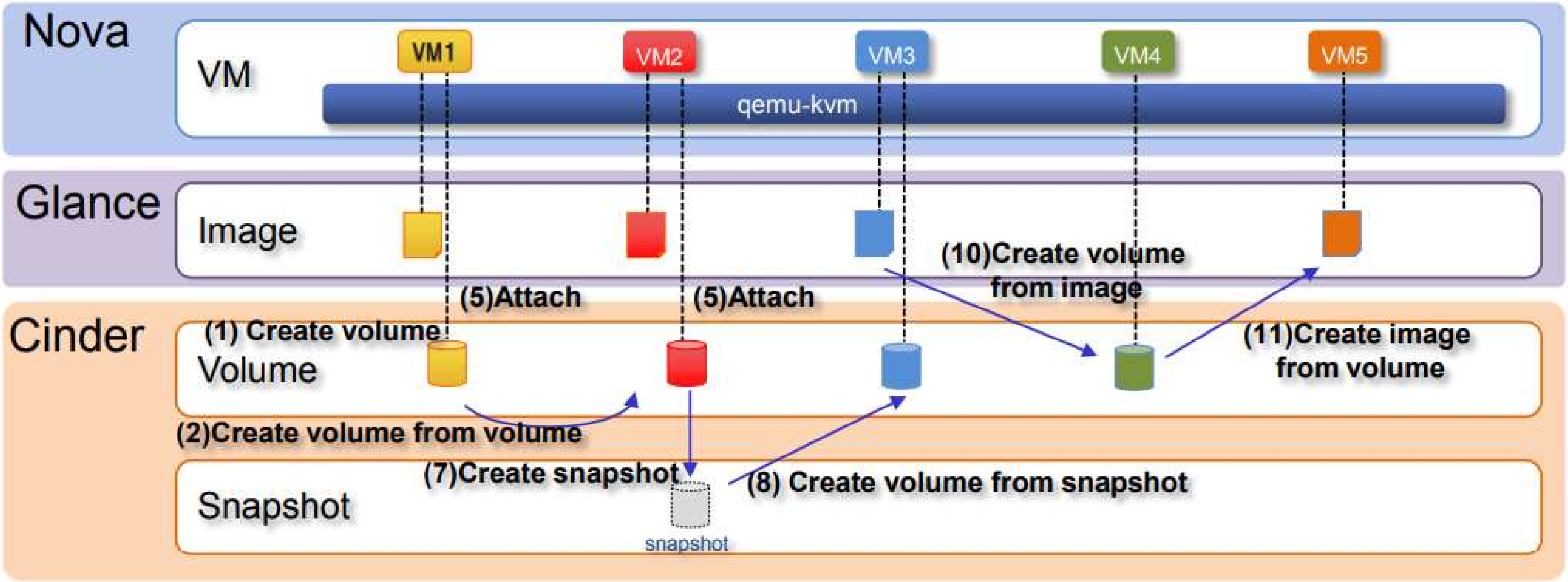
GET	/v3/ {project_id} /volumes/detail **List accessible volumes with details**	detail
POST	/v3/ {project_id} /volumes **Create a volume**	detail
GET	/v3/ {project_id} /volumes **List accessible volumes**	detail
GET	/v3/ {project_id} /volumes/ {volume_id} **Show a volume's details**	detail
PUT	/v3/ {project_id} /volumes/ {volume_id} **Update a volume**	detail
DELETE	/v3/ {project_id} /volumes/ {volume_id} **Delete a volume**	detail
POST	/v3/ {project_id} /volumes/ {volume_id} /metadata **Create metadata for volume**	detail

View full API at:

https://docs.openstack.org/api-ref/block-storage/v3/index.html

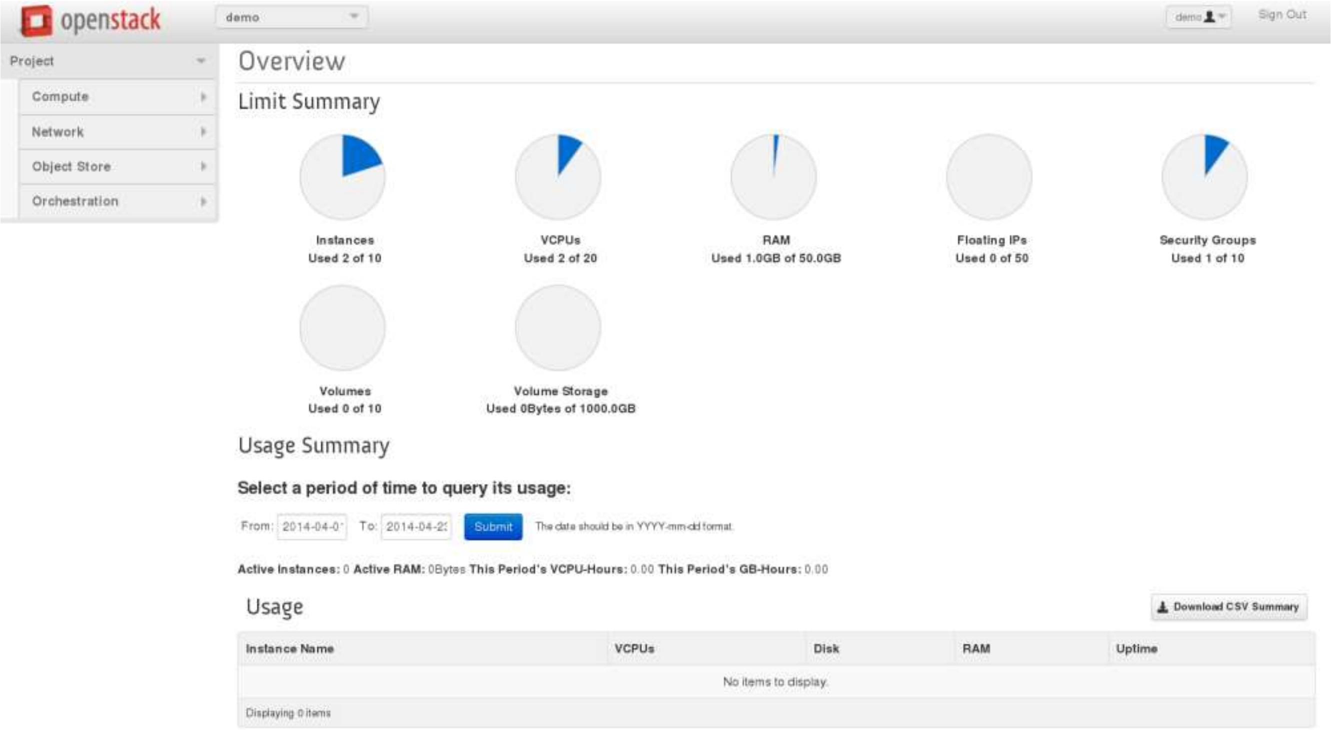
Cinder APIs w/ Glance

- Connects with Glance to support volume creation from image



Horizon

- Self service UI, a python WSGI application
- Interact with all other services (nova, cinder, glance, swift, neutron)



Overview

Limit Summary

Instances
Used 2 of 10

VCPUs
Used 2 of 20

RAM
Used 1.0GB of 50.0GB

Floating IPs
Used 0 of 50

Security Groups
Used 1 of 10

Volumes
Used 0 of 10

Volume Storage
Used 0Bytes of 1000.0GB

Usage Summary

Select a period of time to query its usage:

From: 2014-04-0 To: 2014-04-2 The date should be in YYYY-mm-dd format.

Active Instances: 0 **Active RAM:** 0Bytes **This Period's VCPU-Hours:** 0.00 **This Period's GB-Hours:** 0.00

Usage

Instance Name	VCPUs	Disk	RAM	Uptime
No items to display.				

Displaying 0 items

Keystone: safety first !!

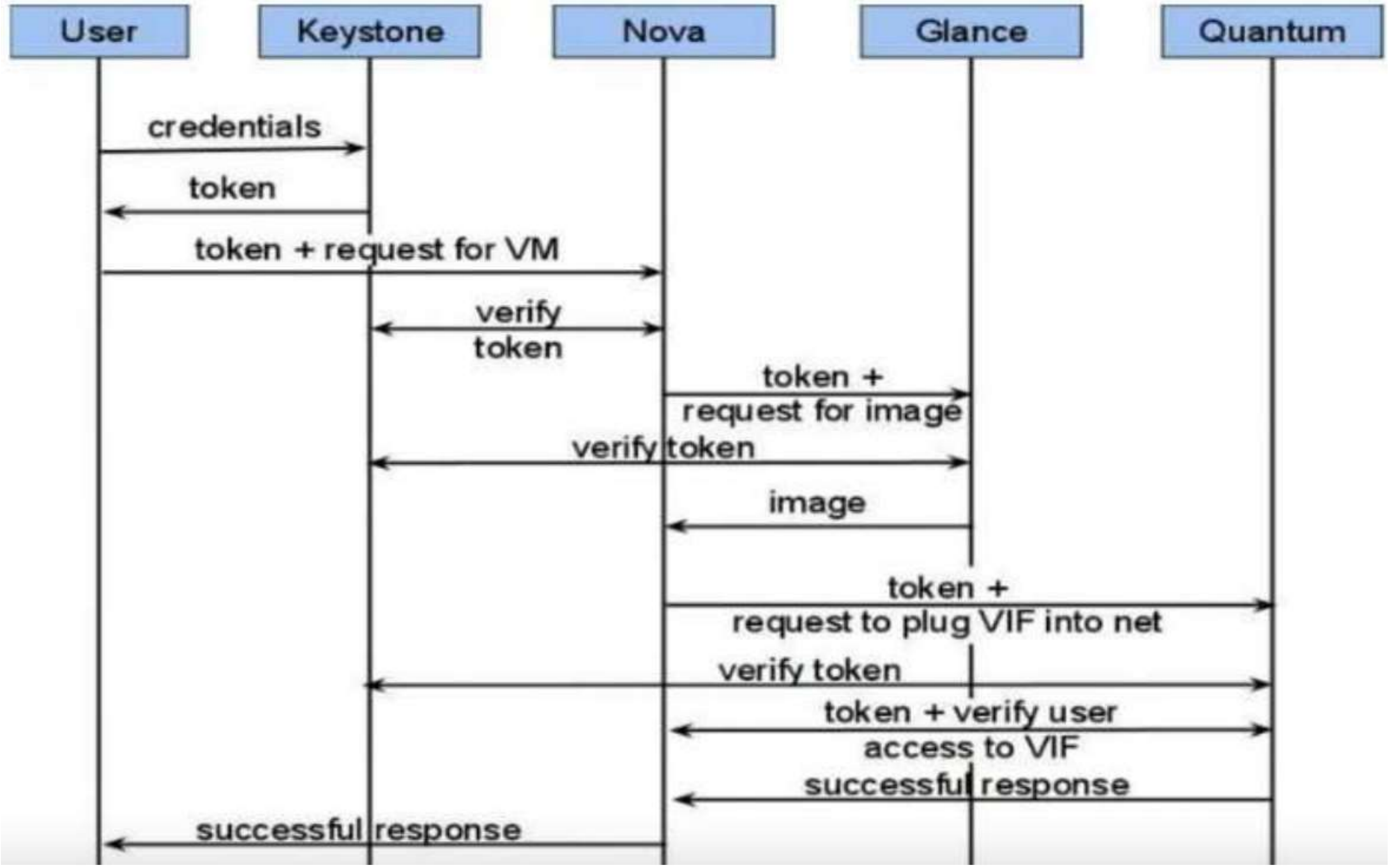
- Many OpenStack services, many API endpoints
  - (endpoint = a network-accessible address, described by URL)
  - How to authenticate them?
  - Who manages the authoritzation?
  - How can I know which endpoint that I want to access?
- OpenStack Keystone identity service for authentication & authorization
- Usually installed as the first service
- Mainly two primary functions: user management + service catalog

> "Keystone provides Identity, Token, Catalog and Policy services for use specifically by projects in the OpenStack family.“

At the core of OpenStack

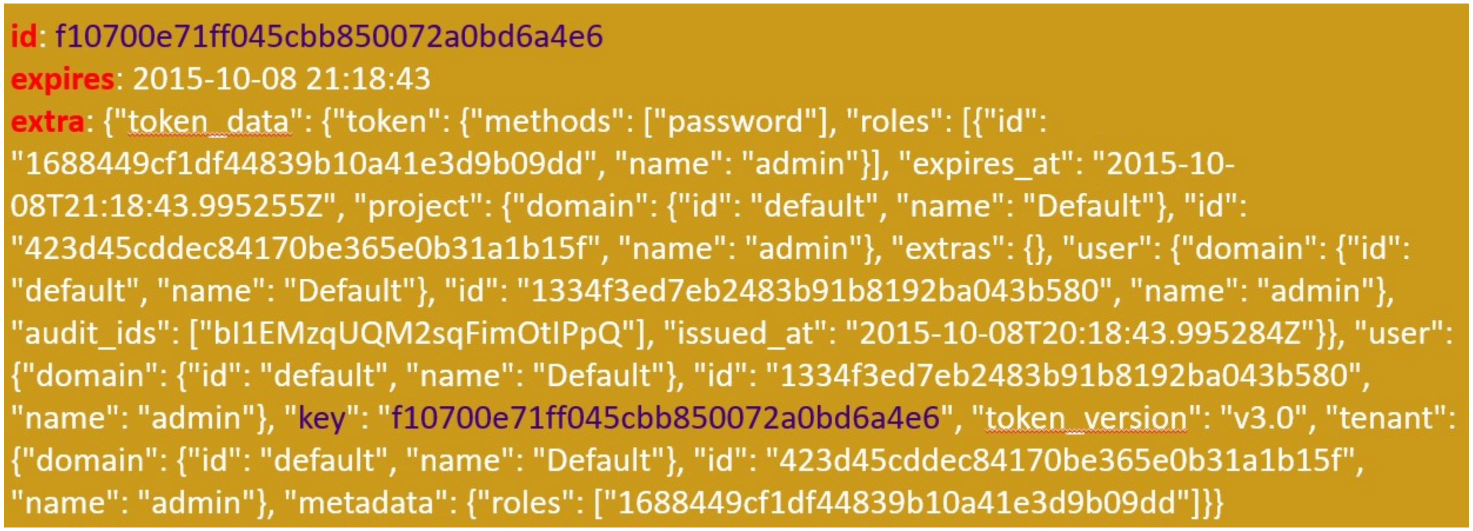
- As a user:
  - Get a token
  - Get the service catalog
- As an admin, defines:
  - Users, Projects, Roles, Roles for users on a project
  - Services, Endpoints for services
  - (roles=assigned rights and privileges)
- As a service
  - Validate a token
  - Tracks installed services and where to locate them
  - Get a trust to impersonate user

Keystone sequence diagram



Token formats - UUID

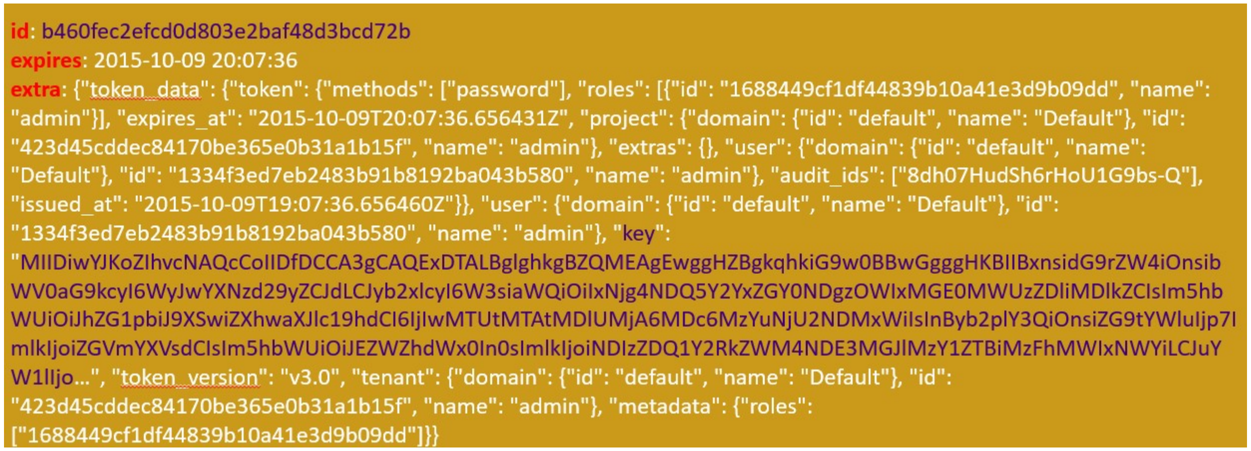
- Randomly generated UUID4 hexadecimal values provide uniqueness
- Pros: better user experience, as the simplest and smallest token format
- Cons: need go back to Keystone server for validation



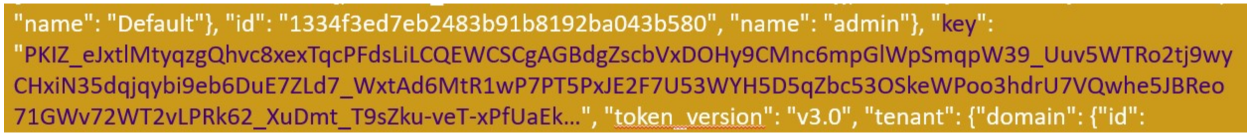
```
id: f10700e71ff045cbb850072a0bd6a4e6
expires: 2015-10-08 21:18:43
extra: {"token_data": {"token": {"methods": ["password"], "roles": [{"id":
"1688449cf1df44839b10a41e3d9b09dd", "name": "admin"}], "expires_at": "2015-10-
08T21:18:43.995255Z", "project": {"domain": {"id": "default", "name": "Default"}, "id":
"423d45cddec84170be365e0b31a1b15f", "name": "admin"}, "extras": {}, "user": {"domain": {"id":
"default", "name": "Default"}, "id": "1334f3ed7eb2483b91b8192ba043b580", "name": "admin"},
"audit_ids": ["bI1EMzqUQM2sqFimOtIPpQ"], "issued_at": "2015-10-08T20:18:43.995284Z"}}, "user":
{"domain": {"id": "default", "name": "Default"}, "id": "1334f3ed7eb2483b91b8192ba043b580",
"name": "admin"}, "key": "f10700e71ff045cbb850072a0bd6a4e6", "token_version": "v3.0", "tenant":
{"domain": {"id": "default", "name": "Default"}, "id": "423d45cddec84170be365e0b31a1b15f",
"name": "admin"}, "metadata": {"roles": ["1688449cf1df44839b10a41e3d9b09dd"]}}
```

Token formats – PKI / PKIZ

- X509 standard cryptographically signed document
- “Z” in PKIZ means compressed PKI
- Pros: token validation w/o Keystone
- Cons: larger than standard HTTP header size, need complex configuration



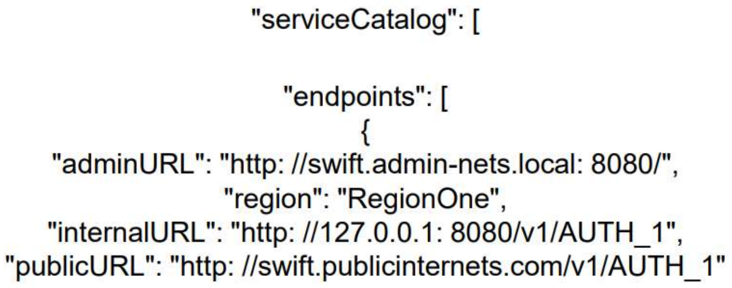
```
id: b460fec2efcd0d803e2baf48d3bcd72b
expires: 2015-10-09 20:07:36
extra: {"token_data": {"token": {"methods": ["password"], "roles": [{"id": "1688449cf1df44839b10a41e3d9b09dd", "name":
"admin"}], "expires_at": "2015-10-09T20:07:36.656431Z", "project": {"domain": {"id": "default", "name": "Default"}, "id":
"423d45cddec84170be365e0b31a1b15f", "name": "admin"}, "extras": {}, "user": {"domain": {"id": "default", "name":
"Default"}, "id": "1334f3ed7eb2483b91b8192ba043b580", "name": "admin"}, "audit_ids": ["8dh07HudSh6rHoU1G9bs-Q"],
"issued_at": "2015-10-09T19:07:36.656460Z"}}, "user": {"domain": {"id": "default", "name": "Default"}, "id":
"1334f3ed7eb2483b91b8192ba043b580", "name": "admin"}, "key":
"MIIDiwYJKoZIhvcNAQcCoIIDfDCCA3gCAQExDTALBglghkgBZQMEAgEwggHZBgkqhkiG9w0BBwGgggHKBIIBxnsidG9rZW4iOnsib
WV0aG9kcyI6WyJwYXNzd29yZCJdLCJyb2xlcyI6W3siaWQiOiIxNjg4NDQ5Y2YxZGY0NDgzOWIxMGE0MWUzZDliMDlkZCIsIm5hb
WUiOiJhZG1pbiJ9XSwiZXhwaXJlc19hdCI6IjIwMTUtMTAtMDlUMjA6MDc6MzYuNjU2NDMxWiIsInByb2plY3QiOnsiZG9tYWluIjp7I
mlkIjoiZGVmYXVsdCIsIm5hbWUiOiJEZWZhdWx0In0sImlkIjoiNDIzZDQ1Y2RkZWM4NDE3MGJlMzY1ZTBiMzFhMWIxNWYiLCJuY
W1lIjo...", "token_version": "v3.0", "tenant": {"domain": {"id": "default", "name": "Default"}, "id":
"423d45cddec84170be365e0b31a1b15f", "name": "admin"}, "metadata": {"roles":
["1688449cf1df44839b10a41e3d9b09dd"]}}
```



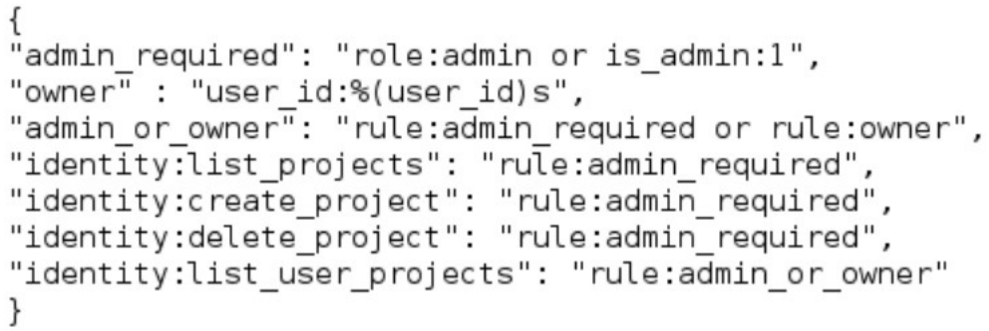
```
"name": "Default"}, "id": "1334f3ed7eb2483b91b8192ba043b580", "name": "admin"}, "key":
"PKIZ_eJxtlMtyqzgQhvc8xexTqcPFdsLiLCQEWCSCgAGBdgZscbVxDOHy9CMnc6mpGlWpSmqpW39_Uuv5WTRo2tj9wy
CHxiN35dqjqybi9eb6DuE7ZLd7_WxtAd6MtR1wP7PT5PxJE2F7U53WYH5D5qZbc53OSkeWPoo3hdrU7VQwhe5JBReo
71GWv72WT2vLPRk62_XuDmt_T9sZku-veT-xPfUaEk...", "token_version": "v3.0", "tenant": {"domain": {"id":
```

Service catalog and policy

- adminURL: for admin users
- internalURL: other services use to talk to each other
- publicURL: everyone else accessing the service endpoint
- Policy provides a rule-based authorization engine and the associated rule management interface, see /etc/keystone/policy.json



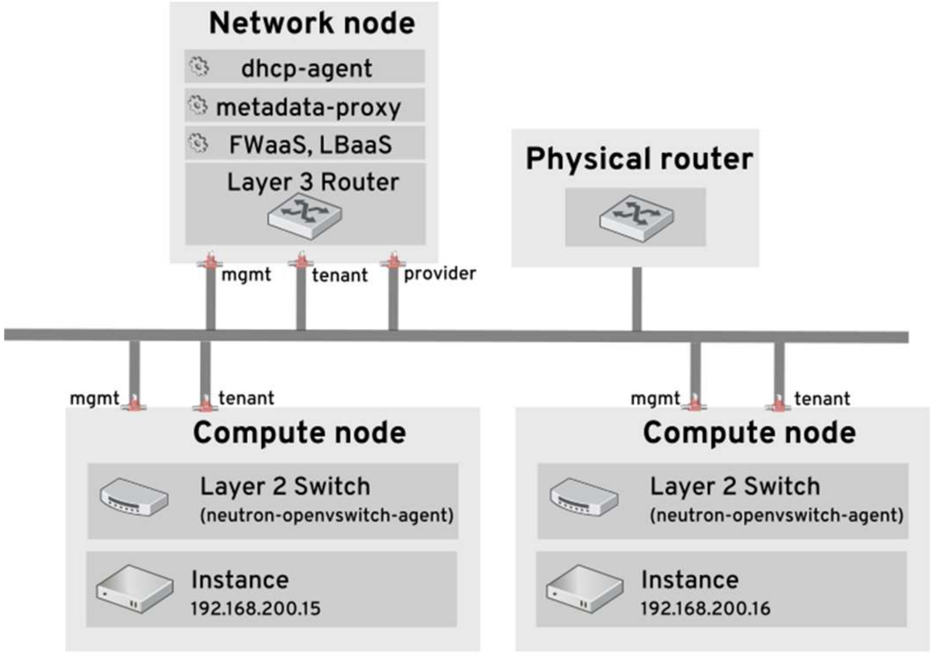
```
"serviceCatalog": [

"endpoints": [
{
"adminURL": "http: //swift.admin-nets.local: 8080/",
"region": "RegionOne",
"internalURL": "http: //127.0.0.1: 8080/v1/AUTH_1",
"publicURL": "http: //swift.publicinternets.com/v1/AUTH_1"
```



```
{
"admin_required": "role:admin or is_admin:1",
"owner" : "user_id:%(user_id)s",
"admin_or_owner": "rule:admin_required or rule:owner",
"identity:list_projects": "rule:admin_required",
"identity:create_project": "rule:admin_required",
"identity:delete_project": "rule:admin_required",
"identity:list_user_projects": "rule:admin_or_owner"
}
```

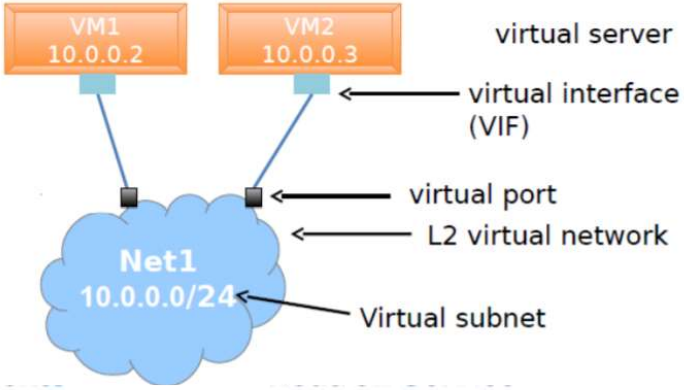
Neutron: network service

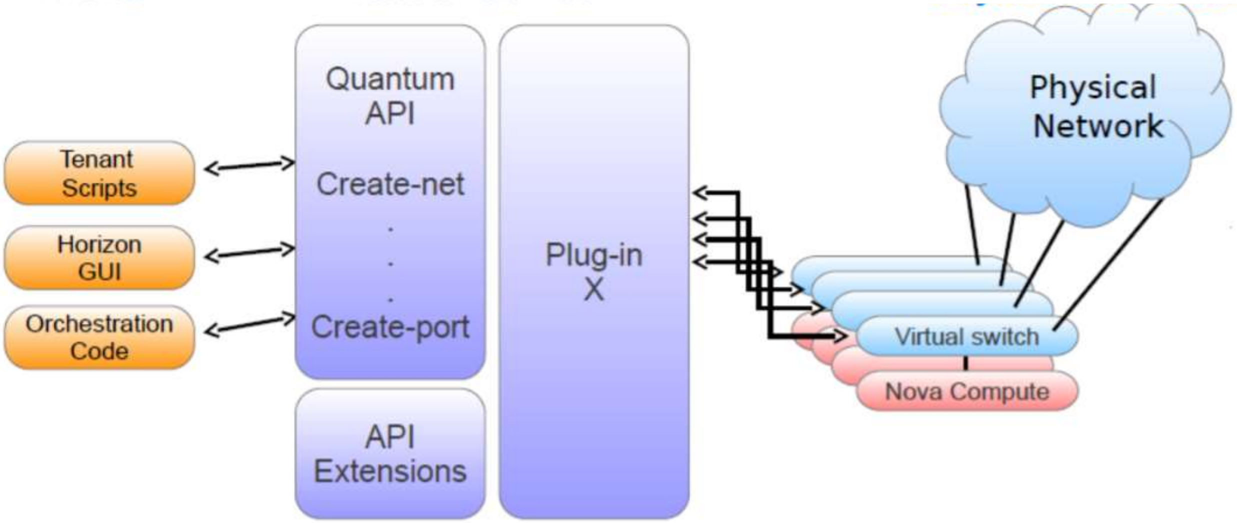
- Supports many network topologies and services
  - L3: self-tenant provisioning
  - Security (ingress + egress rules support)
  - LBaaS (Load Balancing, now Octavia)
  - VPNaaS
- Supports overlay with GRE
- Open to 3rd party solution
  - Vmware NSX plugin
  - LinuxBridge plugin (deprecated)
  - OVS plugin
  - Cisco UCS plugin



Neutron components

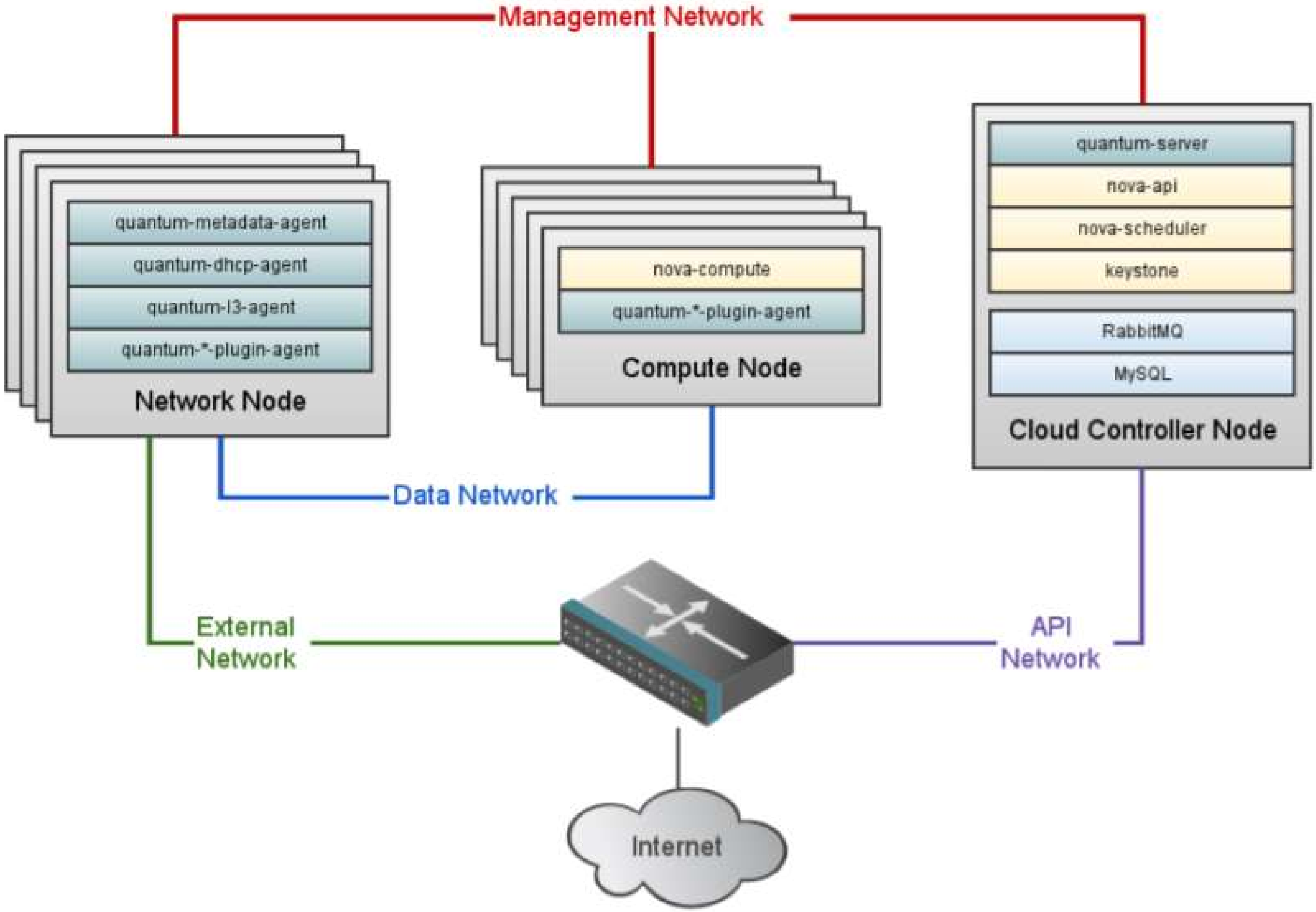
- Nova
- Neutron



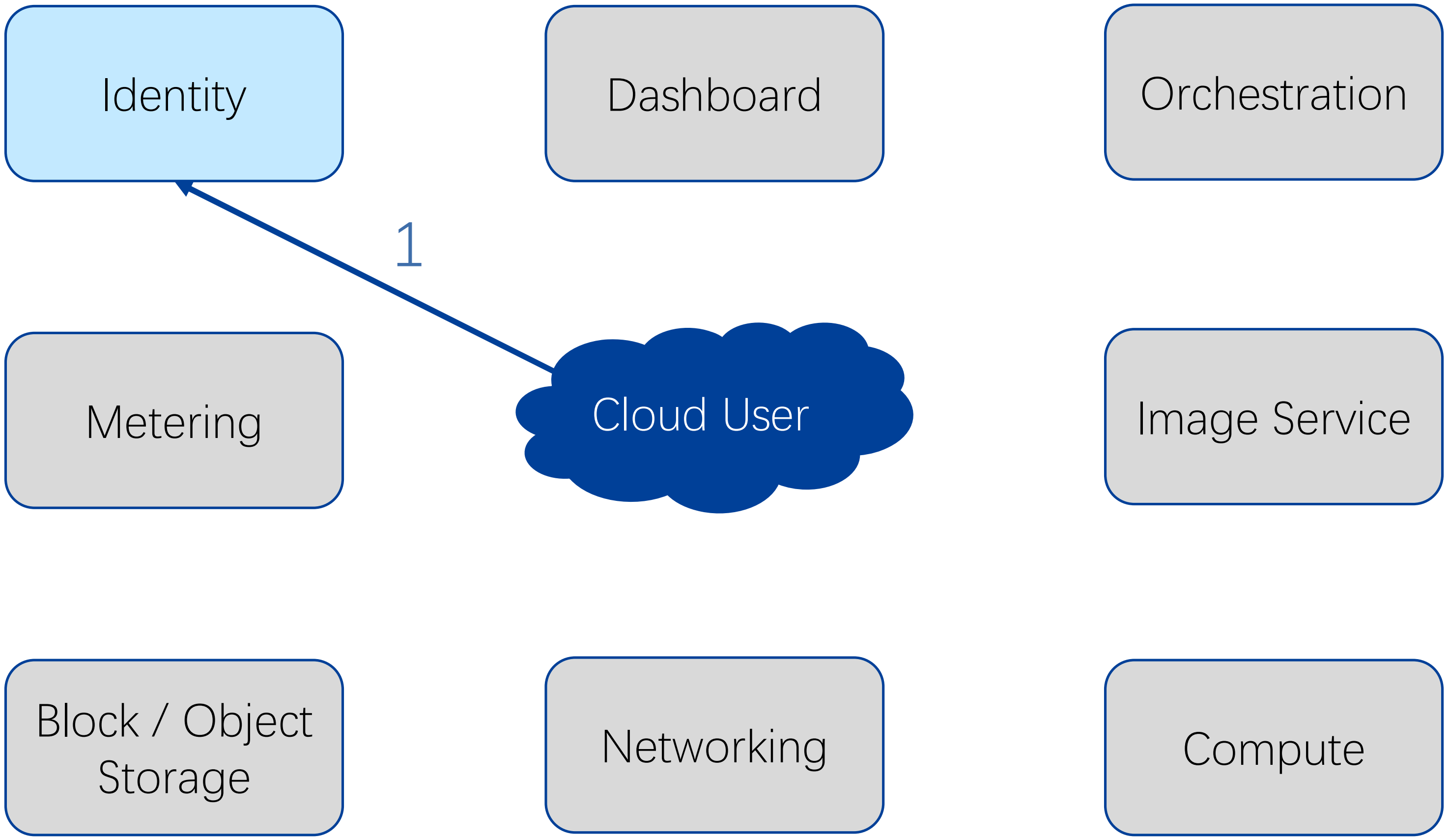


Clients Neutron Service Backend Networks

OpenStack network connectivity



Put together: instance boot step



Identity

Dashboard

Orchestration

1

Metering

Cloud User

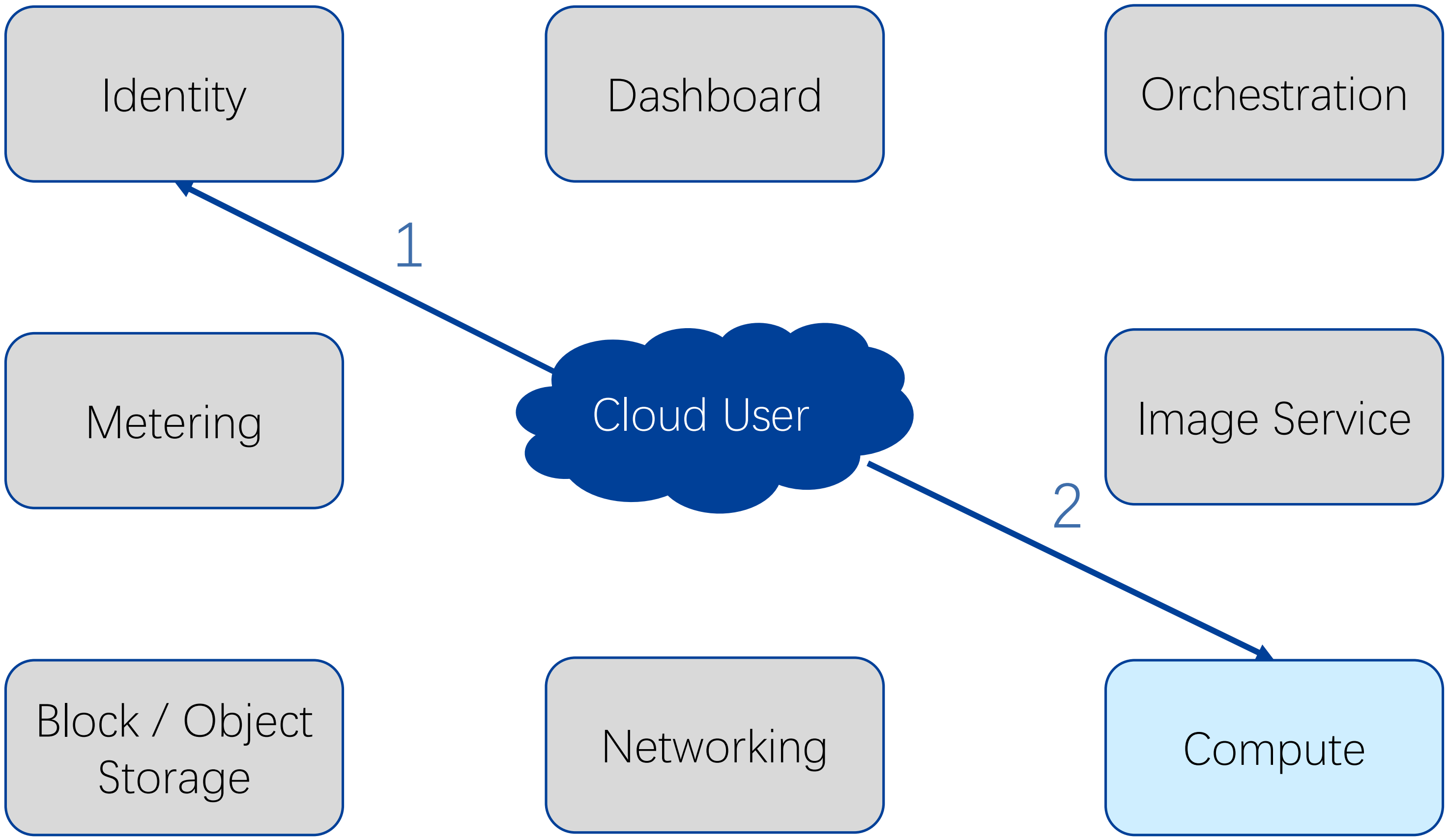
Image Service

Block / Object Storage

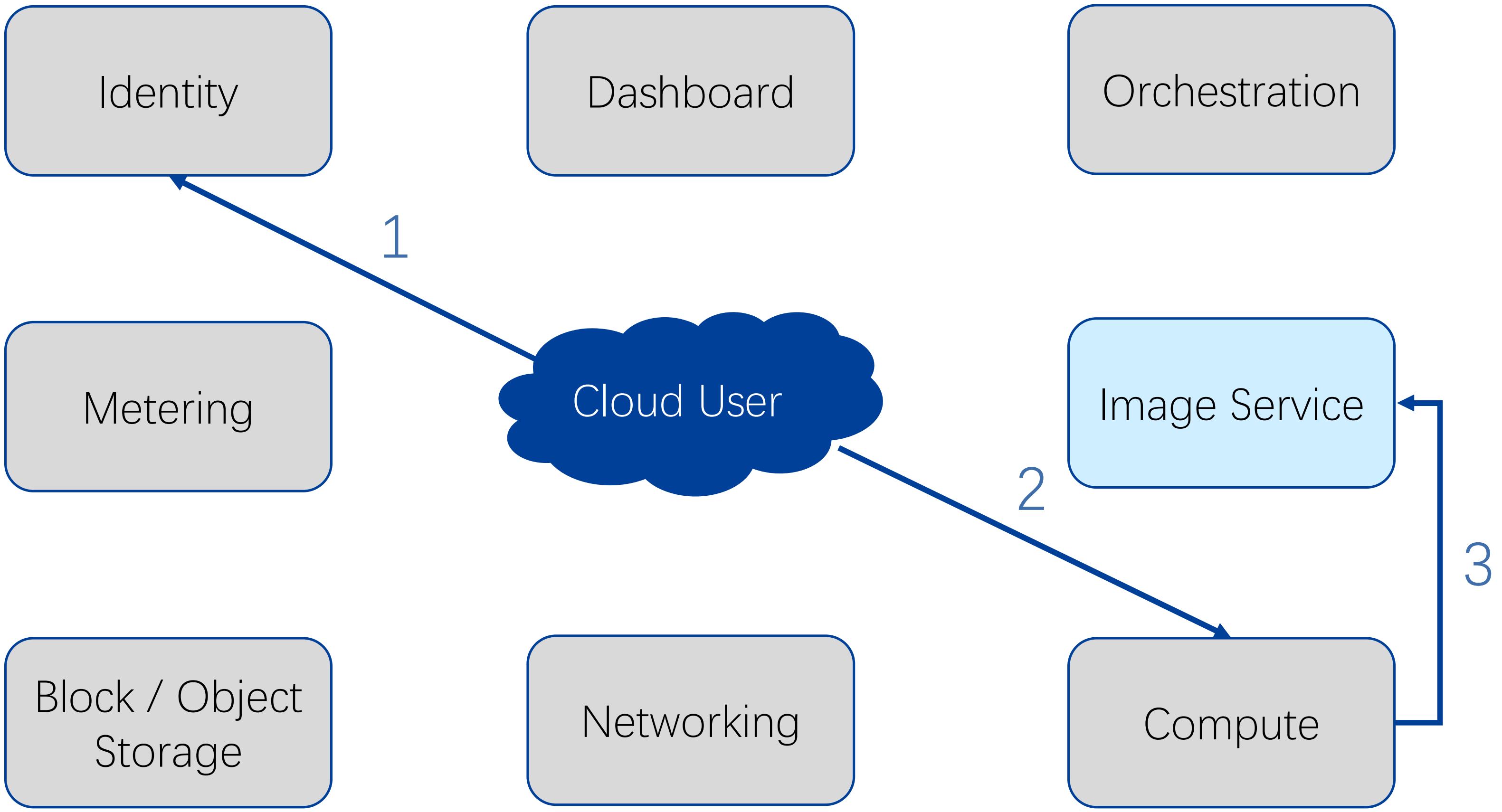
Networking

Compute

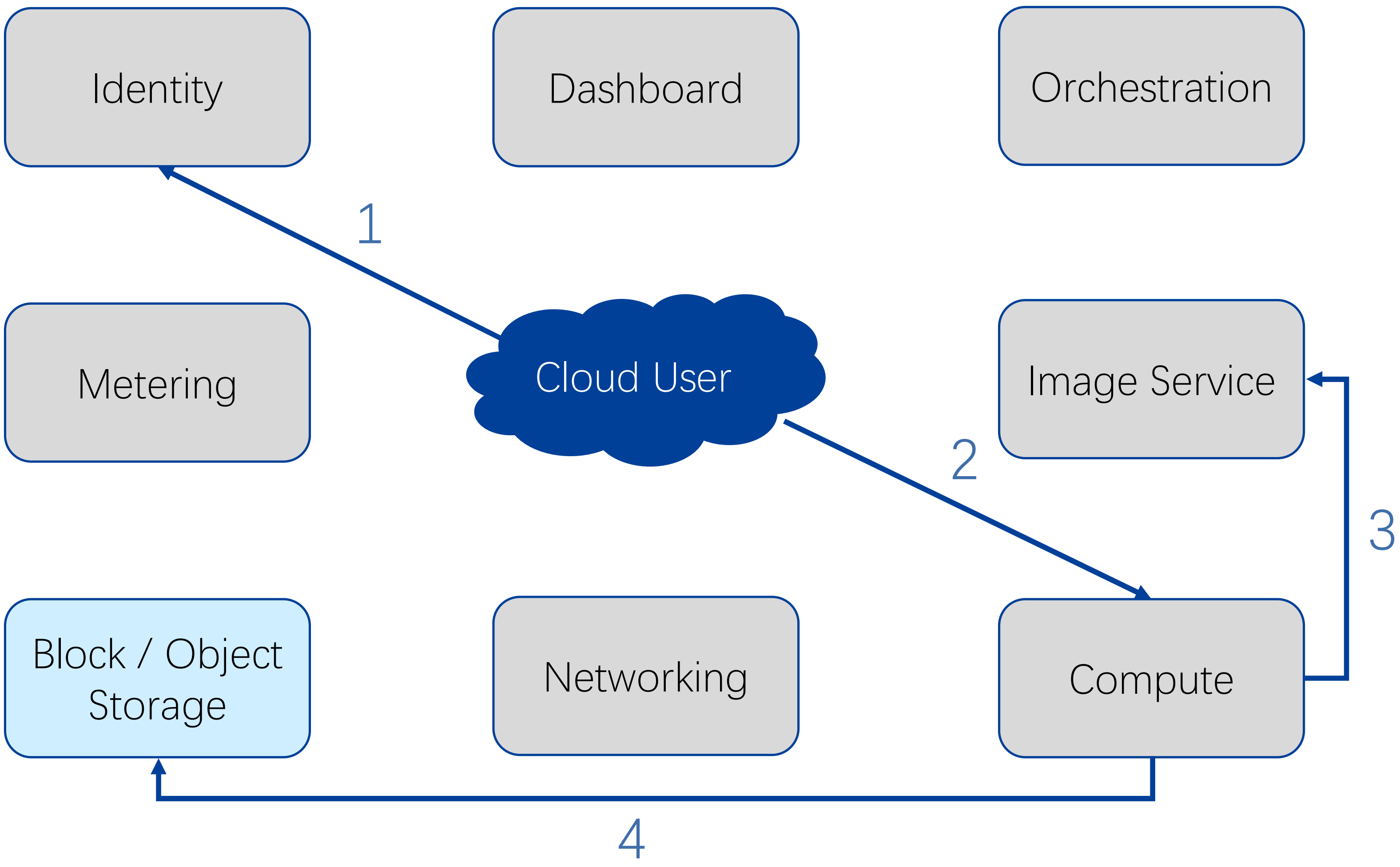
Put together: instance boot step



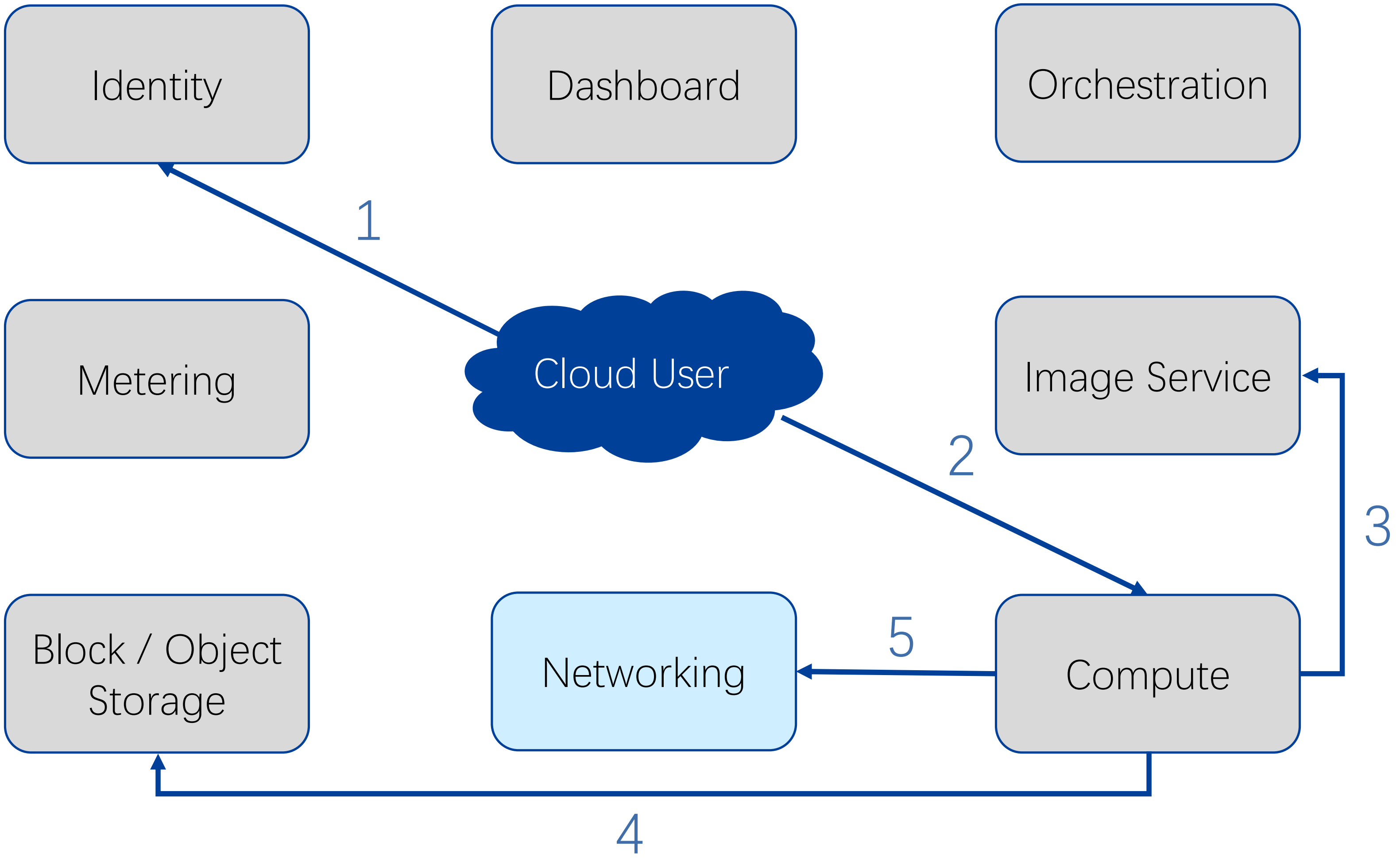
Put together: instance boot step



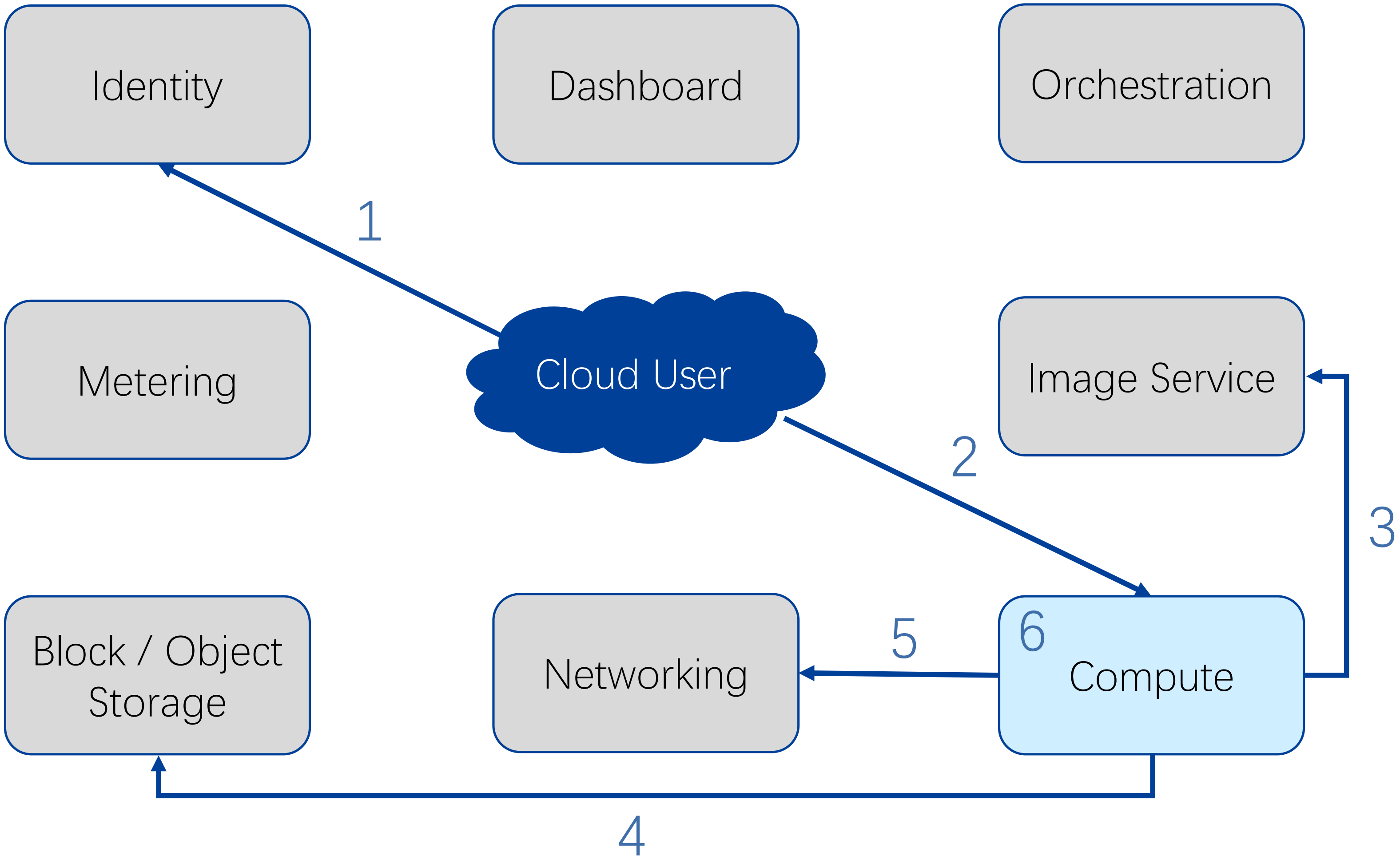
Put together: instance boot step



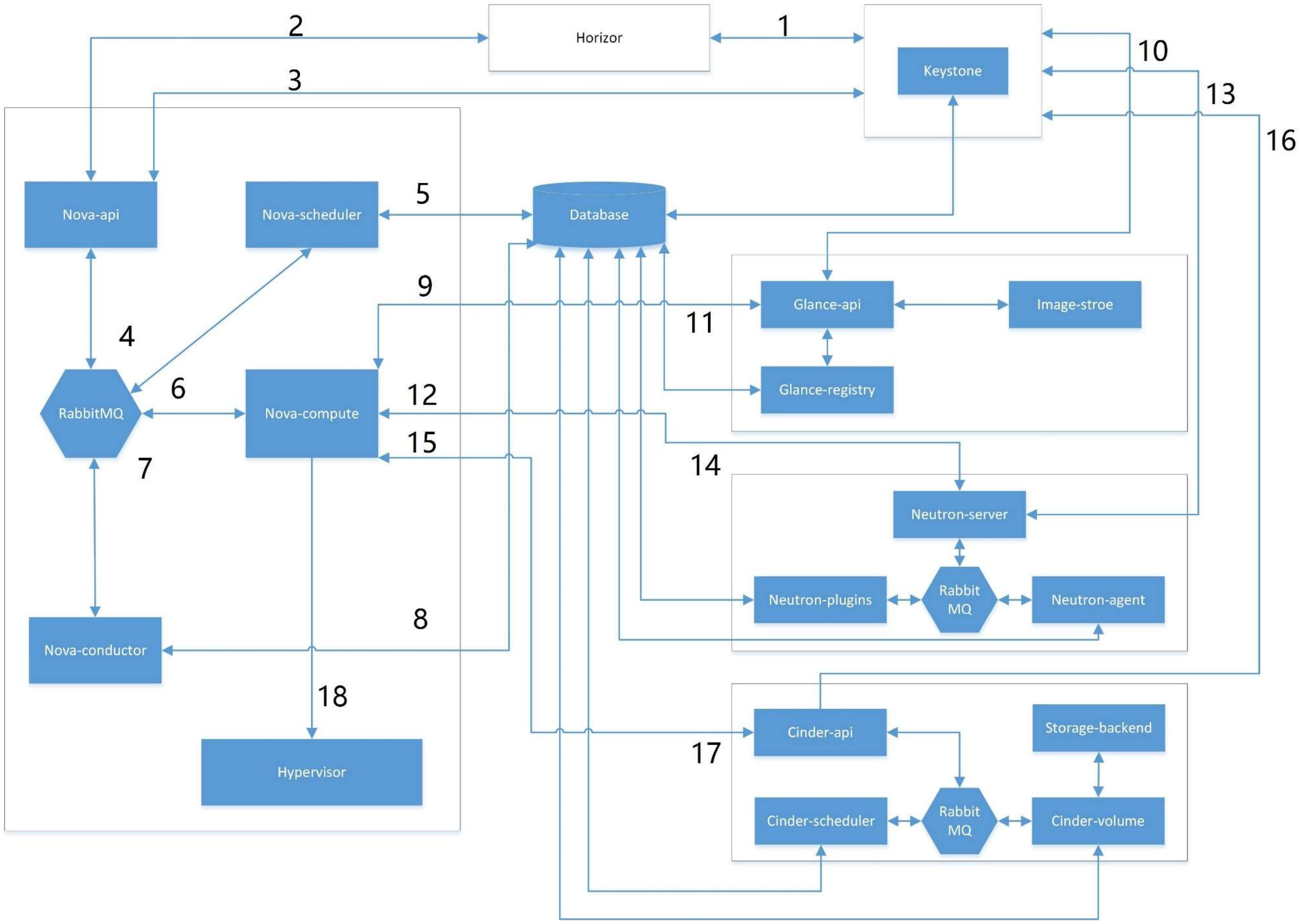
Put together: instance boot step

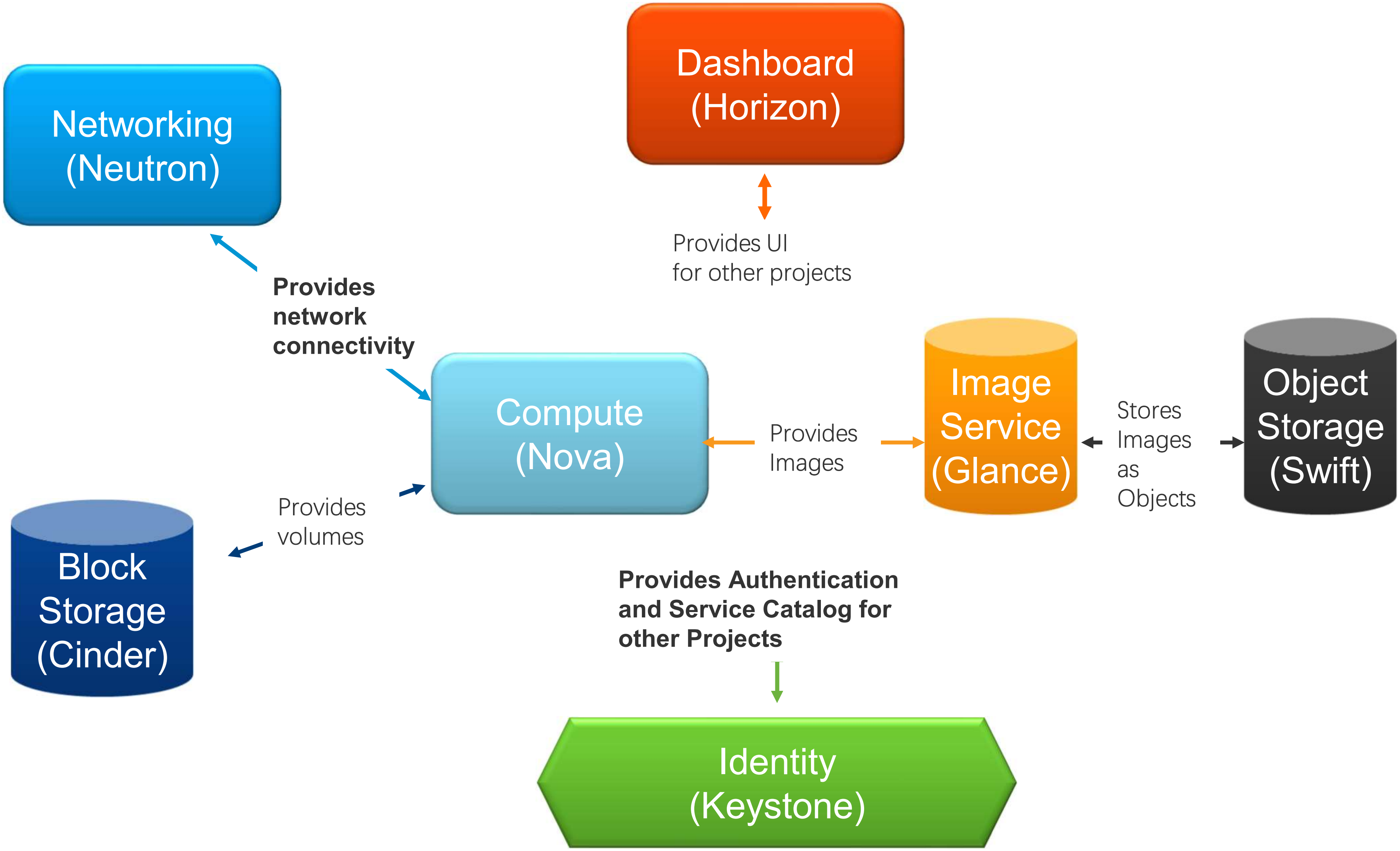


Put together: instance boot step



More details ……


Horizor
1
2
3
Keystone
10
13
16
Nova-api
Nova-scheduler
5
Database
4
6
9
11
Glance-api
Image-stroe
Glance-registry
RabbitMQ
Nova-compute
12
15
7
14
Neutron-server
Neutron-plugins
Rabbit MQ
Neutron-agent
8
Nova-conductor
18
Cinder-api
Storage-backend
17
Cinder-scheduler
Rabbit MQ
Cinder-volume
Hypervisor



Dashboard (Horizon)

Provides UI for other projects

Networking (Neutron)

Provides network connectivity

Compute (Nova)

Provides Images

Image Service (Glance)

Stores Images as Objects

Object Storage (Swift)

Provides volumes

Block Storage (Cinder)

Provides Authentication and Service Catalog for other Projects

Identity (Keystone)

Advanced topic: key protection

- Encryption plays a key role in cloud platform
  - Protect data against leaks
  - Personal Health Information (PHI)
  - Credit Card Payment Data (PCI)
  - AI training data
  - Intellectual Property

- In shared hosting environments, each tenant must only have access to their own stuff
  - Per-Tenant or Per-Volume encryption keys facilitate this
- Security Best Practice
  - Save keys away from your encrypted data, even away from yourself

Barbican: key management system (KMS)

- Provides:
  - RESTful API for Secrets Management
  - Pluggable Backends: Crypto, PKCS#11, KMIP, SGX, etc
  - Integration with Nova, Cinder, and Swift, Neutron, Heat, etc
  - Built to Scale

Store a payload to Barbican:

```
    $ openstack secret store --name testSecret --payload 'TestPayload’
```

Fetch the stored secret:

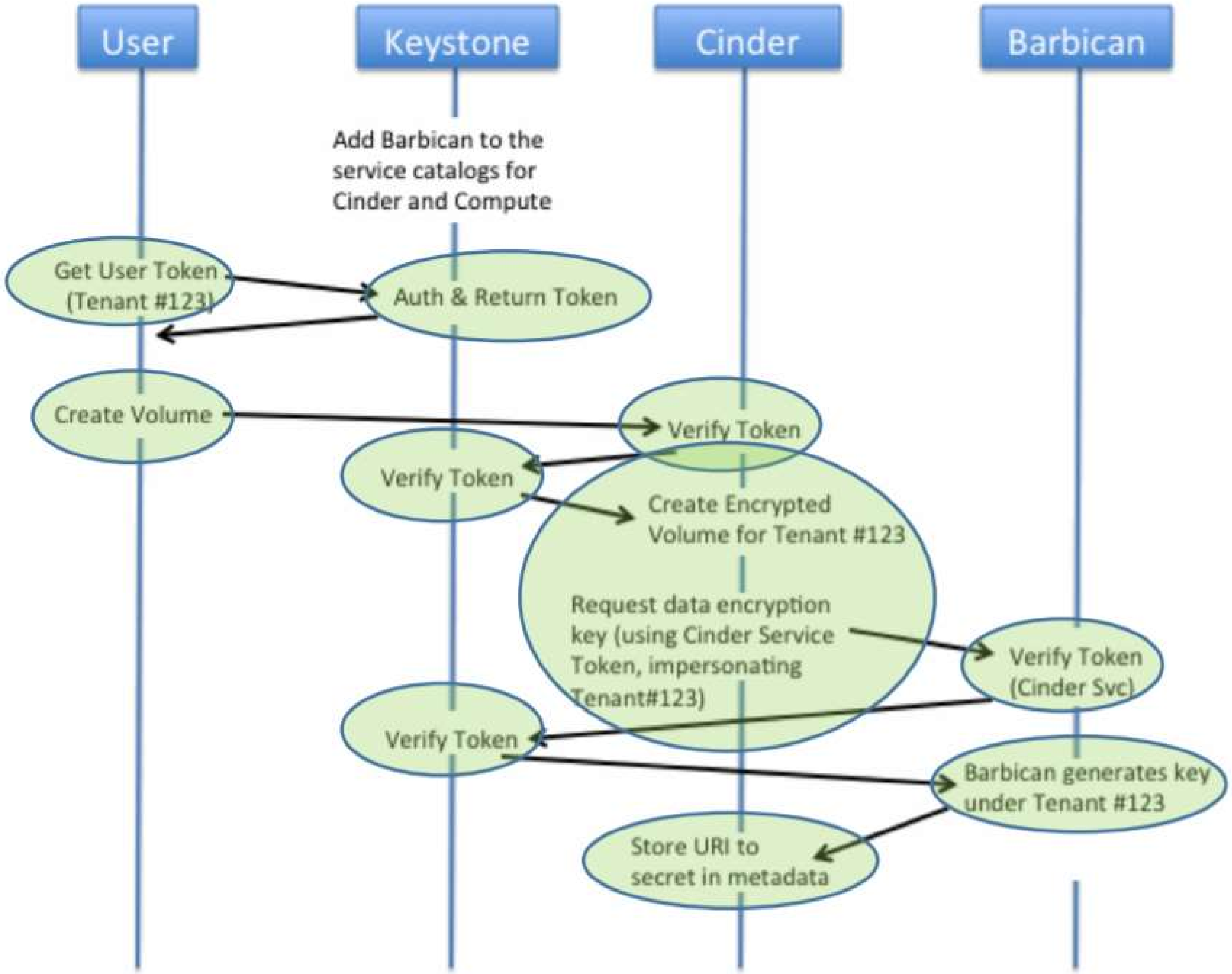
```
    $ openstack secret get
https://192.168.123.173:9311/v1/secrets/efcfec49-b9a3-4425-a9b6-5ba69cb18719
```

Use case: Cinder encryption

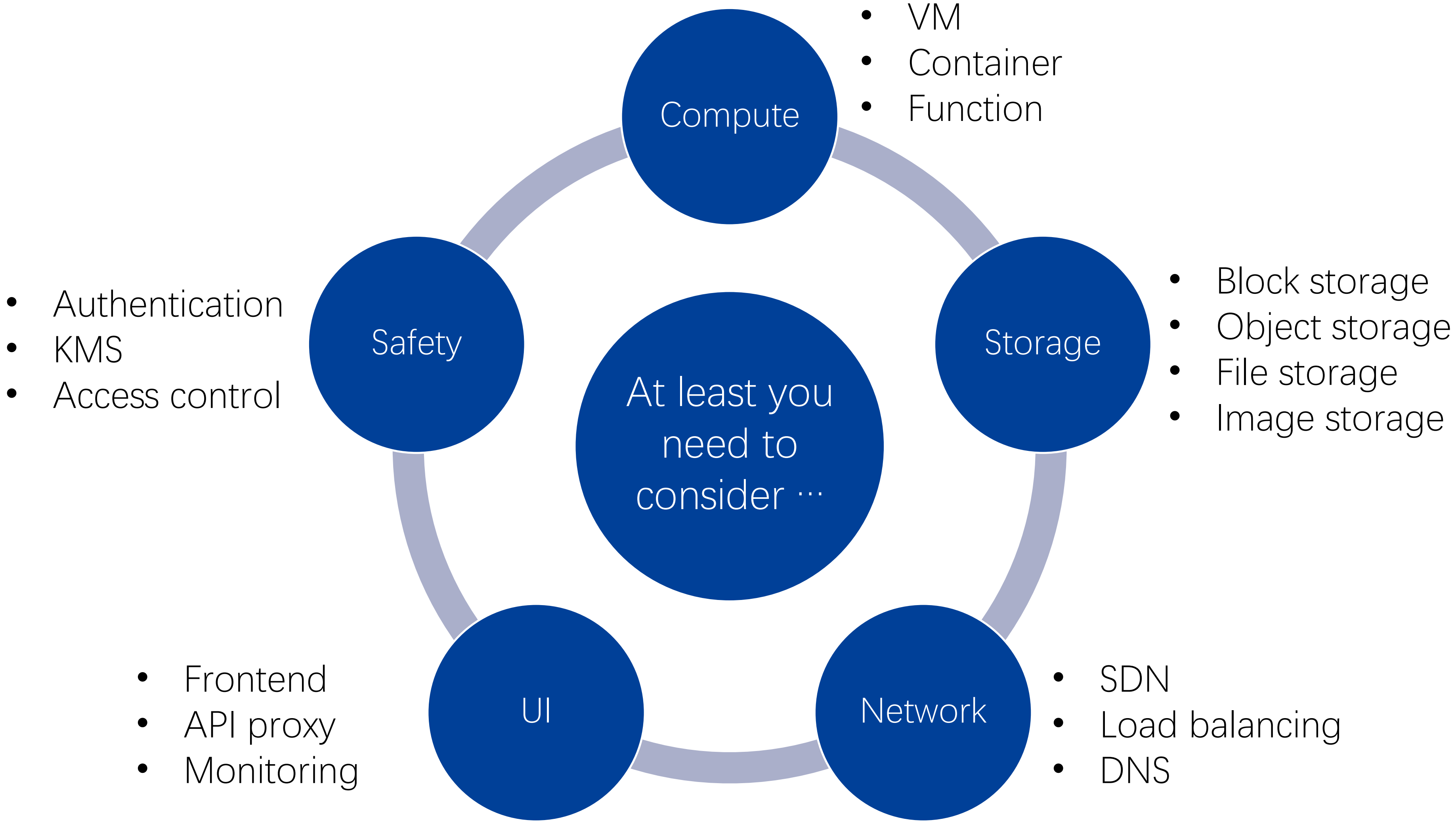
- Volume decrypted on the hypervisor (with Cinder) instead of the guest OS
  - No agent in VM required
  - Works with any operating system and works with bootable volumes
  - Protects data at rest and in-transit to your hypervisor
  - Every volume is protected by it's own unique key
- How to protect Barbican itself?
  - Deploy KMS and DB securely in a locked cabinet with limited physical access
  - Set private Barbican instance not accessible to tenants
  - Use SSL to protect key requests in-transit to hypervisors
  - Even more advanced, use Trusted Execution Environment (TEE) such as SGX *

* Somnath Chakrabarti et al., "Intel SGX Enabled Key Manager Service with OpenStack Barbican", in arXiv, 2017

Creating an encrypted volume


User
Keystone
Cinder
Barbican
Add Barbican to the service catalogs for Cinder and Compute
Get User Token (Tenant #123)
Auth & Return Token
Create Volume
Verify Token
Verify Token
Create Encrypted Volume for Tenant #123
Request data encryption key (using Cinder Service Token, impersonating Tenant#123)
Verify Token (Cinder Svc)
Verify Token
Barbican generates key under Tenant #123
Store URI to secret in metadata

Recap: towards a minimum cloud



At least you need to consider …

Compute

- VM
- Container
- Function

Storage

- Block storage
- Object storage
- File storage
- Image storage

Network

- SDN
- Load balancing
- DNS

UI

- Frontend
- API proxy
- Monitoring

Safety

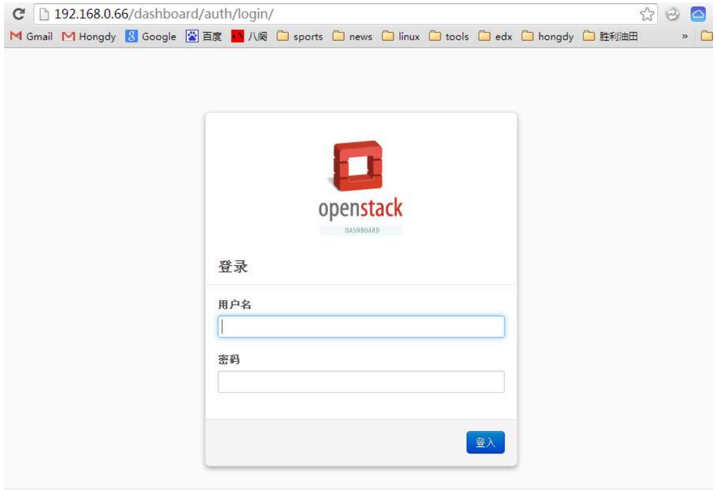
- Authentication
- KMS
- Access control

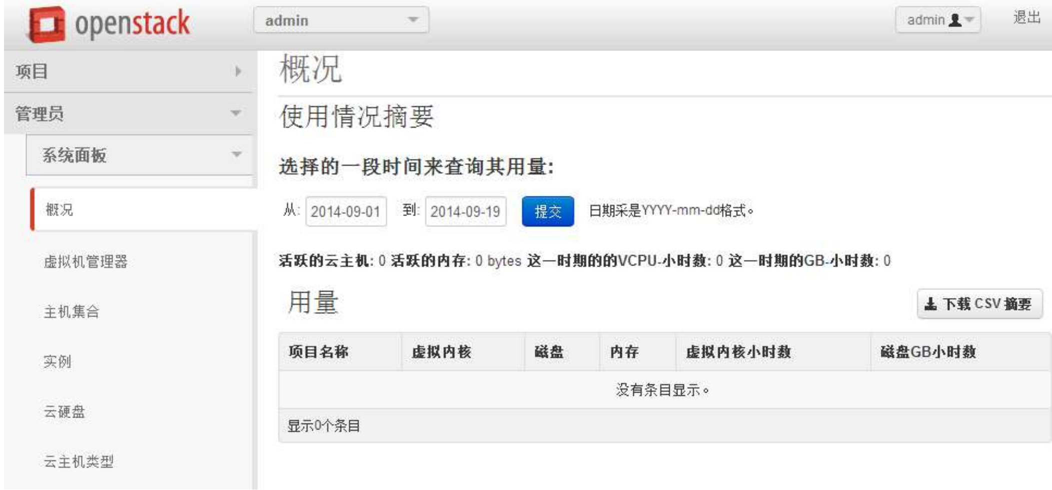
OpenStack Liberty deployment

- Default host os: centos 7.2.x
- Install OpenStack via “packstack”:
  - $ yum –y install openstack-packstack
- Generate configuration:
  - $ packstack --gen-answer-file=/root/myanswer.txt
- Modify configuration file according to:
  - Network interface
  - DB admin password
  - Control, compute, network node IP addresses in a cluster deployment
  - VLAN configuration for ML2 and OVS

OpenStack Liberty deployment

- Configure network interface:
  - OVSPort interface at "/etc/sysconfig/network-scripts/ifcfg-eth0"
  - OVSBridge for outside at "/etc/sysconfig/network-scripts/ifcfg-br-ex"
- Login into OpenStack dashboard
  - Username and password defined in "/root/keystonerc_admin"





概况

使用情况摘要

选择的一段时间来查询其用量:

从: 2014-09-01 到: 2014-09-19 日期采是YYYY-mm-dd格式。

活跃的云主机: 0 活跃的内存: 0 bytes 这一时期的的VCPU-小时数: 0 这一时期的GB-小时数: 0

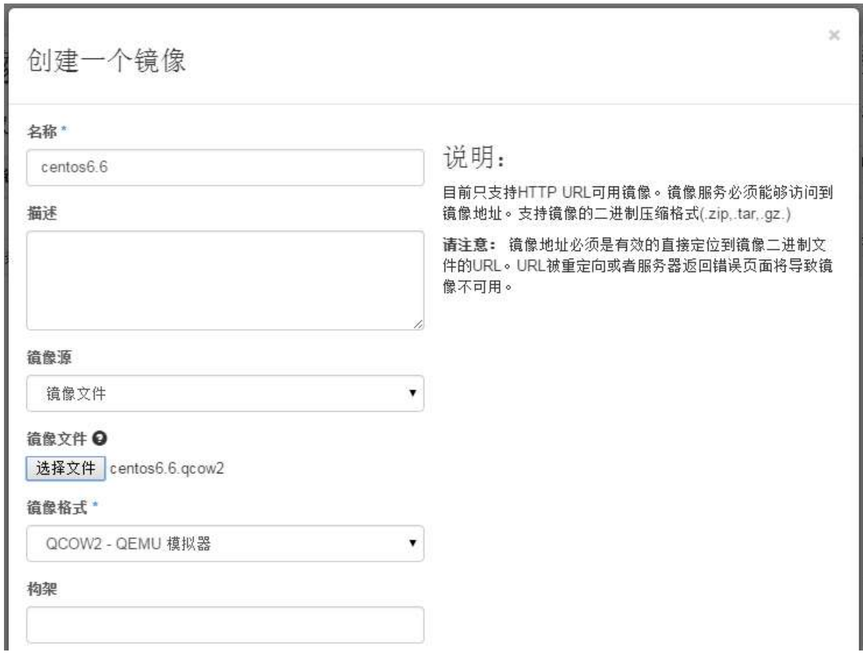
用量

项目名称	虚拟内核	磁盘	内存	虚拟内核小时数	磁盘GB小时数
没有条目显示。					

显示0个条目

OpenStack Liberty deployment

- Image creation and network creation
  - Upload image with QCOW2 format (mentioned before)
  - Choose network supplier VXLAN



创建一个镜像

名称 *
centos6.6

描述

镜像源
镜像文件

镜像文件
选择文件 centos6.6.qcow2

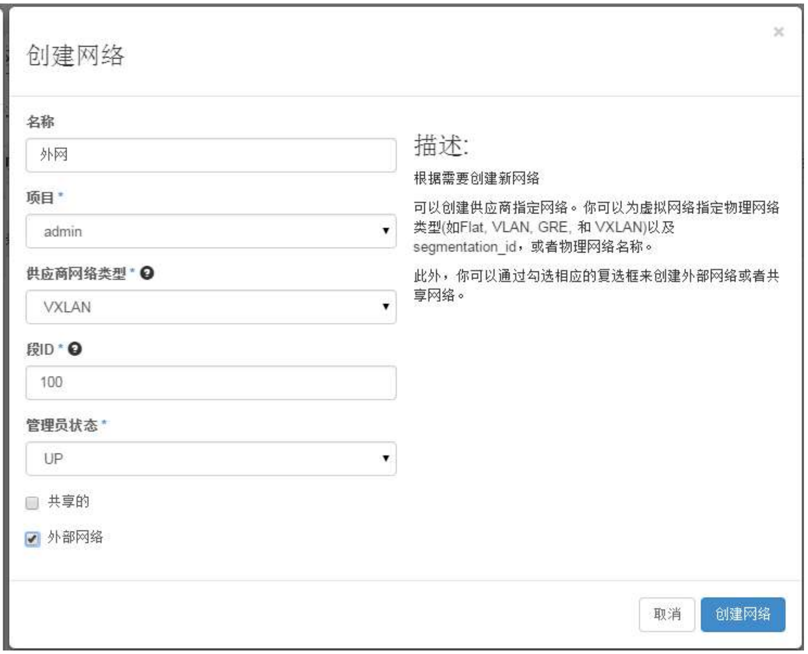
镜像格式 *
QCOW2 - QEMU 模拟器

构架

说明：

目前只支持HTTP URL可用镜像。镜像服务必须能够访问到镜像地址。支持镜像的二进制压缩格式(.zip,.tar,.gz.)

请注意： 镜像地址必须是有效的直接定位到镜像二进制文件的URL。URL被重定向或者服务器返回错误页面将导致镜像不可用。



创建网络

名称
外网

项目 *
admin

供应商网络类型 *
VXLAN

段ID *
100

管理员状态 *
UP

□ 共享的
☑ 外部网络

描述:

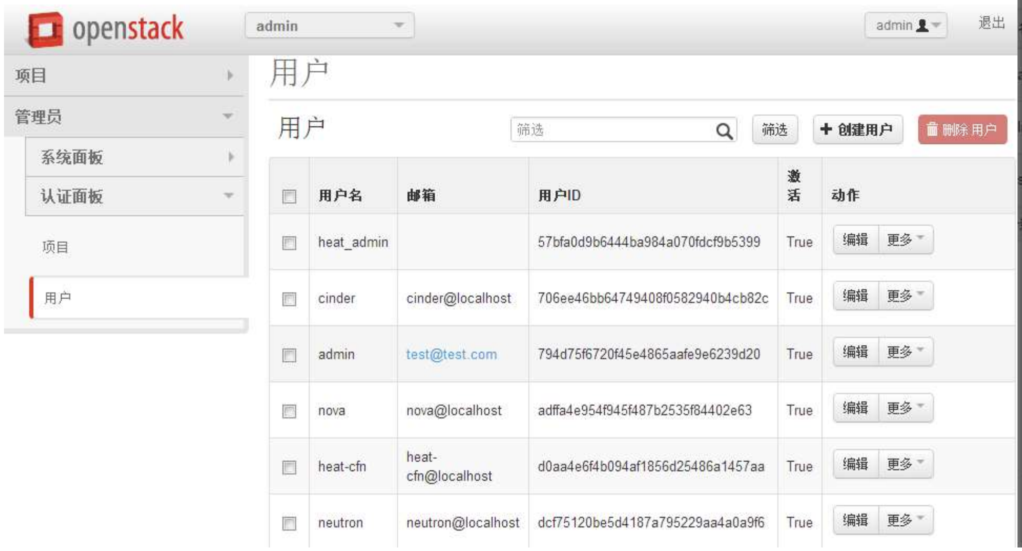
根据需要创建新网络

可以创建供应商指定网络。你可以为虚拟网络指定物理网络类型(如Flat, VLAN, GRE, 和 VXLAN)以及segmentation_id，或者物理网络名称。

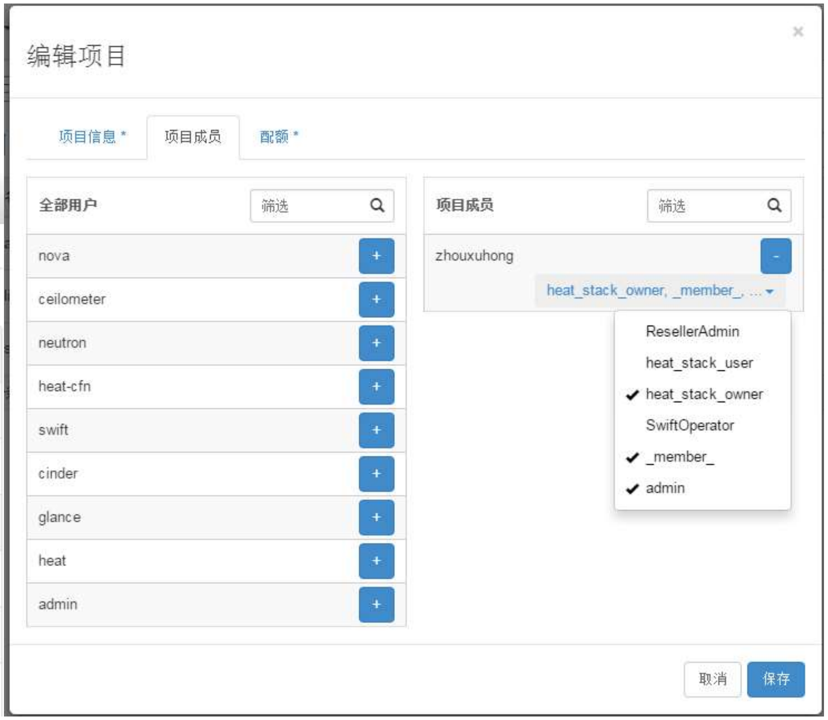
此外，你可以通过勾选相应的复选框来创建外部网络或者共享网络。

OpenStack Liberty deployment

- User management and project management
  - Assign role for each user
  - Assign privilege for each user in the project

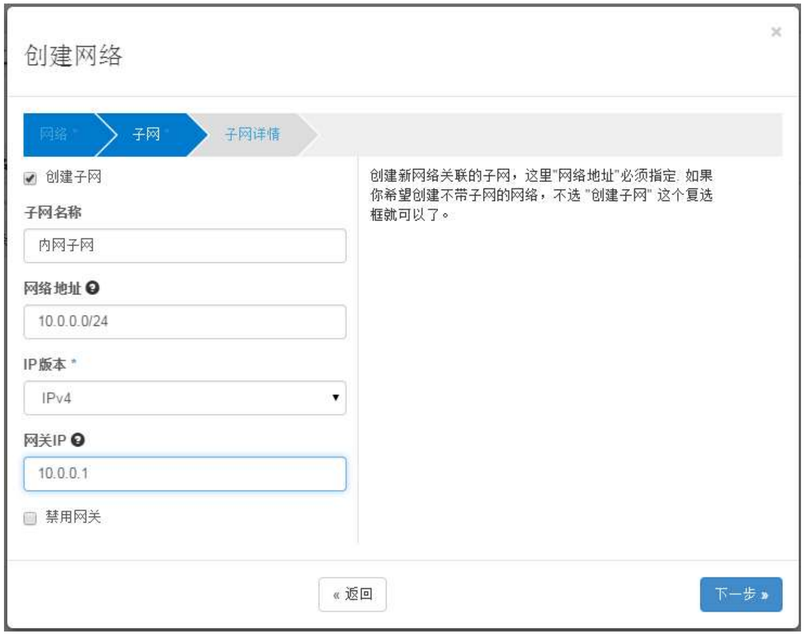


□	用户名	邮箱	用户ID	激活	动作
□	heat_admin		57bfa0d9b6444ba984a070fdcf9b5399	True	编辑 更多
□	cinder	cinder@localhost	706ee46bb64749408f0582940b4cb82c	True	编辑 更多
□	admin	test@test.com	794d75f6720f45e4865aafe9e6239d20	True	编辑 更多
□	nova	nova@localhost	adffa4e954f945f487b2535f84402e63	True	编辑 更多
□	heat-cfn	heat-cfn@localhost	d0aa4e6f4b094af1856d25486a1457aa	True	编辑 更多
□	neutron	neutron@localhost	dcf75120be5d4187a795229aa4a0a9f6	True	编辑 更多



From user’s perspective

- Setup internal network and access control
  - Connect the created internal network with router to enable outside comm.
  - Add ICMP and TCP, verify with “ping”



创建网络

☑ 创建子网

子网名称: 内网子网

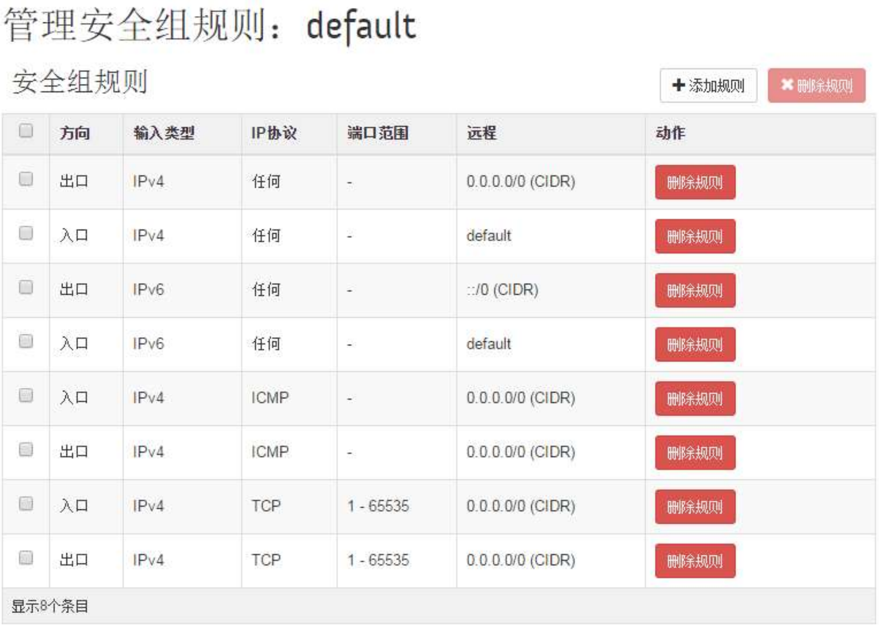
网络地址: 10.0.0.0/24

IP版本 *: IPv4

网关IP: 10.0.0.1

□ 禁用网关

创建新网络关联的子网，这里"网络地址"必须指定. 如果你希望创建不带子网的网络，不选 "创建子网" 这个复选框就可以了。



管理安全组规则：default

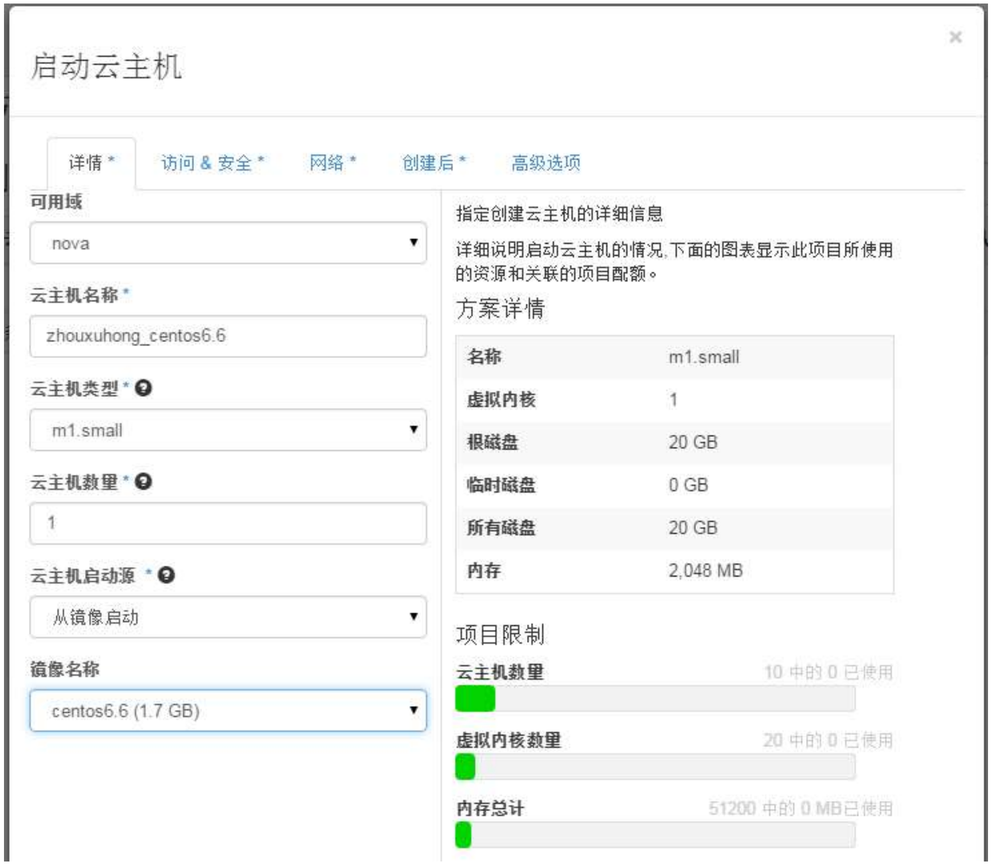
安全组规则

□	方向	输入类型	IP协议	端口范围	远程	动作
□	出口	IPv4	任何	-	0.0.0.0/0 (CIDR)	删除规则
□	入口	IPv4	任何	-	default	删除规则
□	出口	IPv6	任何	-	::/0 (CIDR)	删除规则
□	入口	IPv6	任何	-	default	删除规则
□	入口	IPv4	ICMP	-	0.0.0.0/0 (CIDR)	删除规则
□	出口	IPv4	ICMP	-	0.0.0.0/0 (CIDR)	删除规则
□	入口	IPv4	TCP	1 - 65535	0.0.0.0/0 (CIDR)	删除规则
□	出口	IPv4	TCP	1 - 65535	0.0.0.0/0 (CIDR)	删除规则

显示8个条目

From user’s perspective

- Create VM with assigned access control and network interface



启动云主机

可用域
nova

云主机名称 *
zhouxuhong_centos6.6

云主机类型 *
m1.small

云主机数量 *
1

云主机启动源 *
从镜像启动

镜像名称
centos6.6 (1.7 GB)

指定创建云主机的详细信息

详细说明启动云主机的情况.下面的图表显示此项目所使用的资源和关联的项目配额。

方案详情

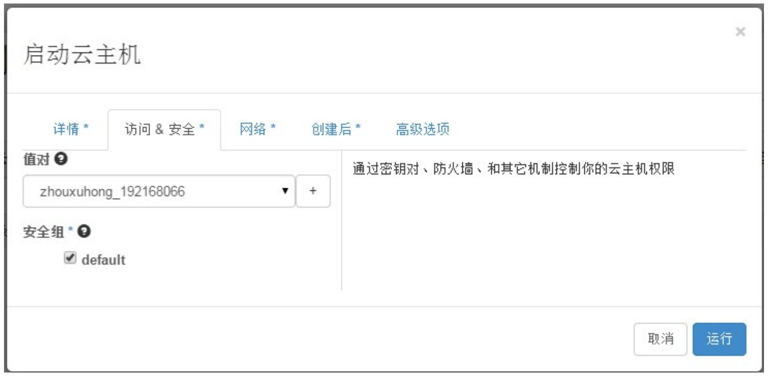
名称	m1.small
虚拟内核	1
根磁盘	20 GB
临时磁盘	0 GB
所有磁盘	20 GB
内存	2,048 MB

项目限制

云主机数量 10 中的 0 已使用

虚拟内核数量 20 中的 0 已使用

内存总计 51200 中的 0 MB已使用

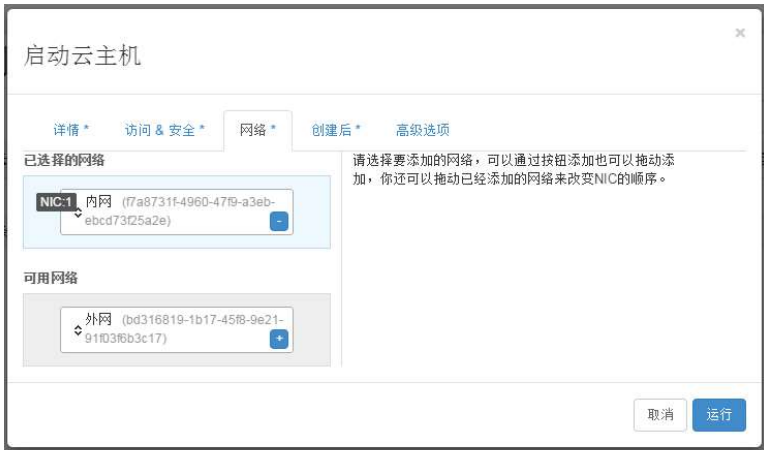


启动云主机

值对
zhouxuhong_192168066 +

安全组 *
☑ default

通过密钥对、防火墙、和其它机制控制你的云主机权限



启动云主机

已选择的网络
NIC:1 内网 (f7a8731f-4960-47f9-a3eb-ebcd73f25a2e) -

可用网络
外网 (bd316819-1b17-45f8-9e21-91f03f6b3c17) +

请选择要添加的网络，可以通过按钮添加也可以拖动添加，你还可以拖动已经添加的网络来改变NIC的顺序。

From user’s perspective

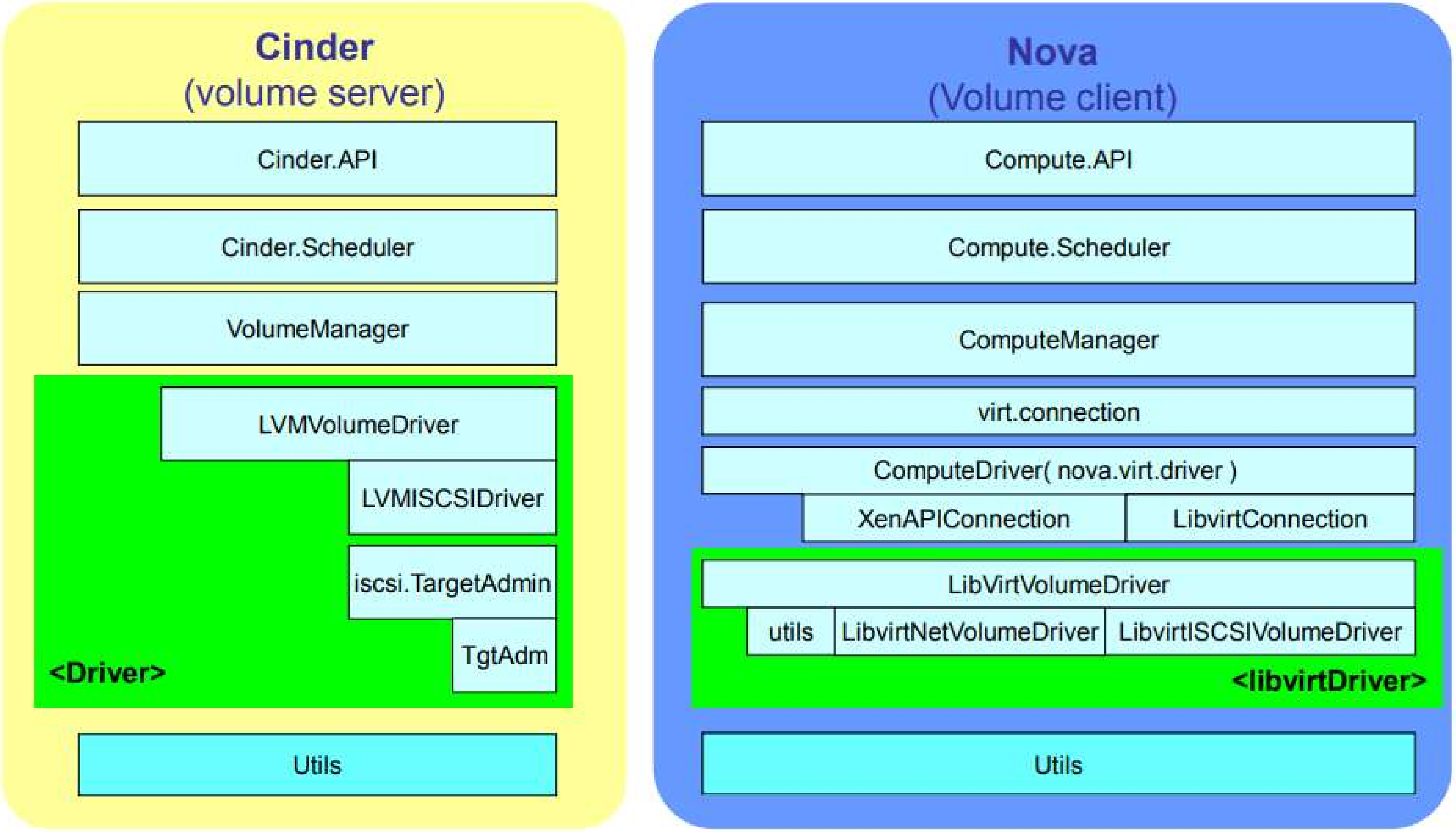
- Now you may SSH into your VM
- Other advanced functionality includes:
  - Attach network storage to your VM
  - Load balancing among multiple VMs
  - Stack deployment
  - Dashboard customization
  - Creating docker containers

Reference

- https://cloudarchitectmusings.com/2013/11/18/laying-cinder-block-volumes-in-openstack-part-1-the-basics/
- https://events.static.linuxfound.org/sites/events/files/slides/CloudOpenJapan2014-Kimura_0.pdf
- https://www.slideshare.net/prk1980/cloud-orchestration-major-tools-comparision
- https://www.linux-kvm.org/images/7/7b/Kvm-forum-2013-openstack.pdf
- https://www.redhat.com/en/topics/cloud-computing/cloud-vs-virtualization
- https://www.programmersought.com/article/20663670268/
- https://www.slideshare.net/eprasad/keystone-openstack-identity-service
- https://www.cisco.com/c/dam/global/en_ca/assets/ciscoconnect/2014/pdfs/open_stack_deployment_in_the_enterprise_josh_kaya_mike_perron.pdf
- https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux_openstack_platform/7/html/networking_guide/openstack_networking_concepts
- https://www.slideshare.net/CodeOps/containers-and-openstack-a-happy-marriage-madhuri-intel-cc18
- https://www.slideshare.net/devananda1/ods-havana-provisioning-bare-metal-with-open-stack
- https://object-storage-ca-ymq-1.vexxhost.net/swift/v1/6e4619c416ff4bd19e1c087f27a43eea/www-assets-prod/pdf-downloads/Containers-and-OpenStack.pdf
- https://object-storage-ca-ymq-1.vexxhost.net/swift/v1/6e4619c416ff4bd19e1c087f27a43eea/www-assets-prod/presentation-media/OSSummitAtlanta2014-NovaLibvirtKVM2.pdf

谢谢！

Cinder & Nova collaboration



Advanced token format – Fernet

- Symmetric, encrypt with Primary Key, decrypt with a list of Fernet keys
- Key size 256b = SHA256 HMAC Signing Key (128b) + AES Key (128b)
- Primary key: encrypt and decrypt, key file named with the highest index
- Secondary key: only decrypt, key file named not the highest or the lowest
- Staged key: key file named with the lowest index (0)

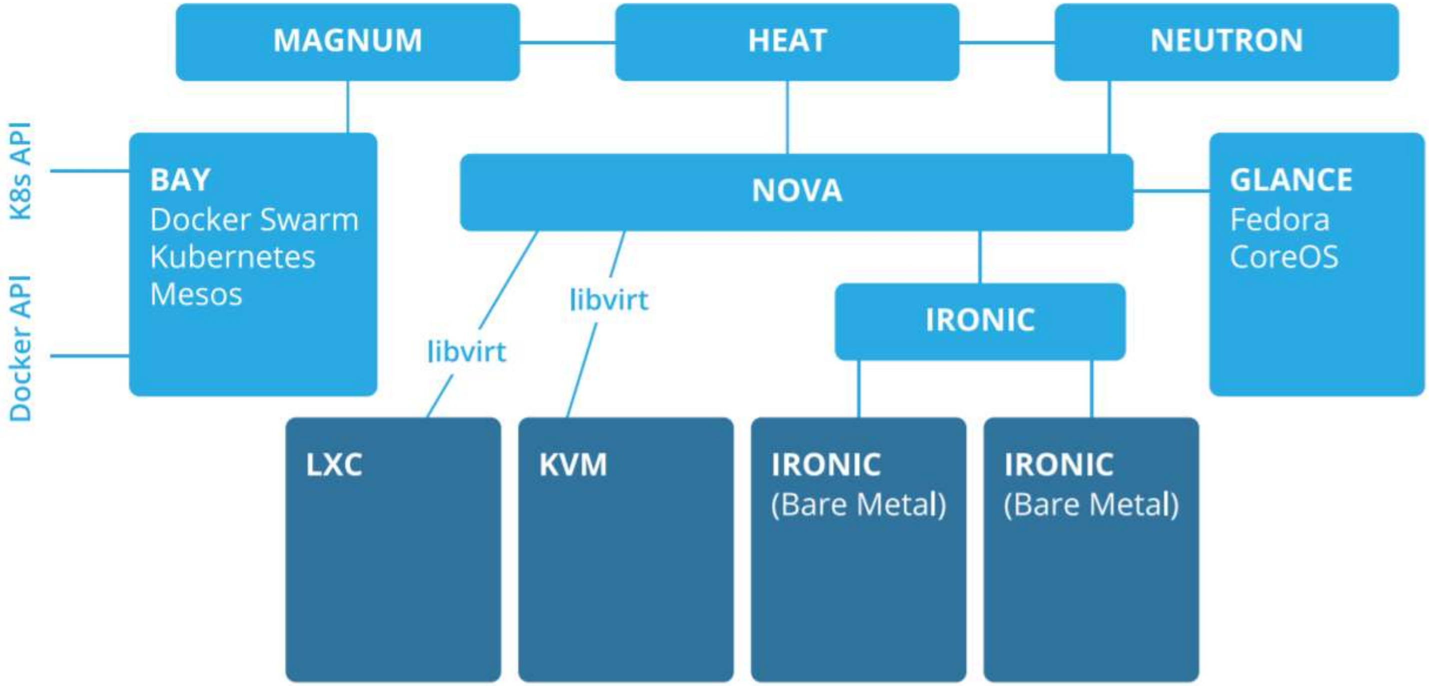
- Pros: no persistence, multiple data center deployment
- Cons: Validation performance impacted by #revocation events

Alternatives to OpenStack

	EUCALYPTUS	apachecloudstack	openstack
Weakness	• Installation requirements • Configurable but not very easily customizable • Community inclusion	• Very clean GUI • Single Java code • Weak AWS integration	• Young Codebase • Uncertain future • Initial configuration
Strengths	• Excellent commercial support • Fault tolerance • Offers Hybrid solution with AWS	• Well round GUI • Stack is fairly simple • Customization of the storage backend	• Single Codebase • Growing community • Corporate support

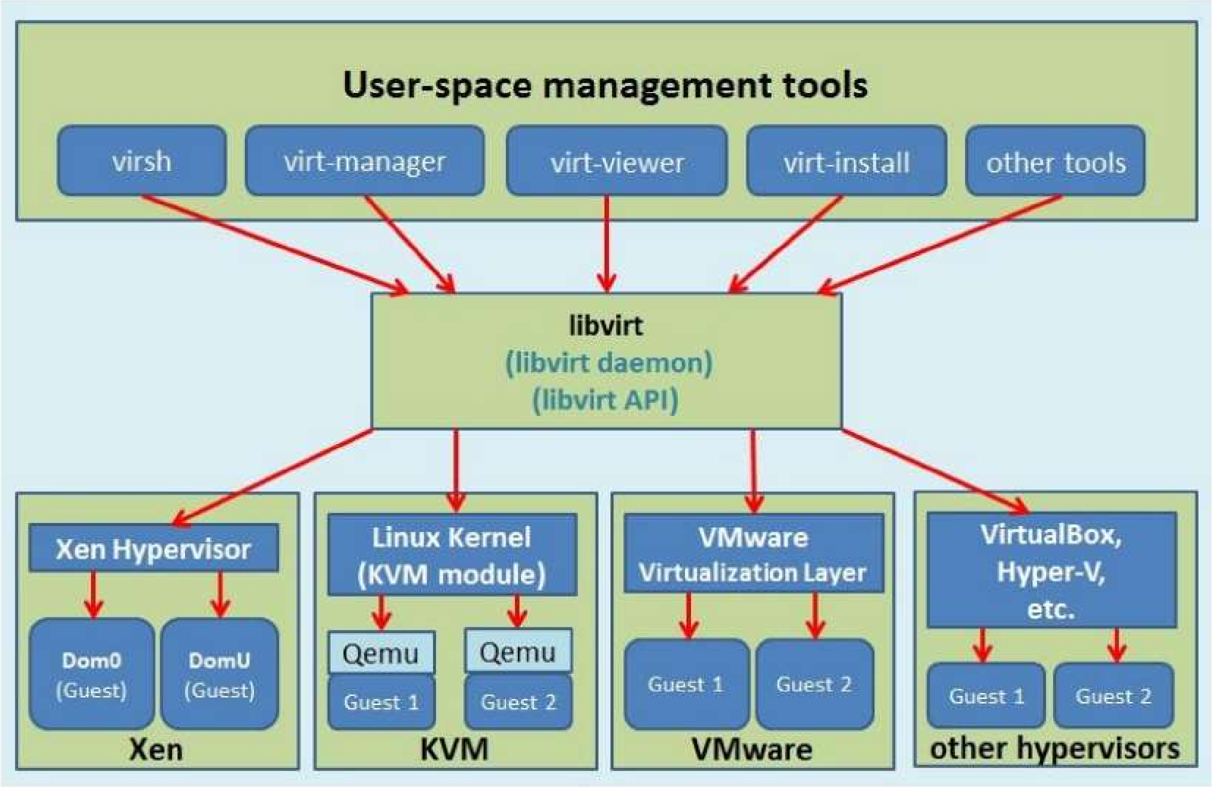
Modules for containers

- Magnum: Container specific APIs for multi-tenant containers-as-a-service
- Kolla: dynamic OpenStack control plane, services runs in containers
- Murano: catalog allowing deploying packaged Kubernetes applications



附录1：Libvirt 介绍

- Libvirt是一个支持多种hypervisor的标准虚拟化管理框架
- 支持Xen，KVM（常用），Vmware，Hyper-V等多种hypervisor



附录1：Libvirt 介绍

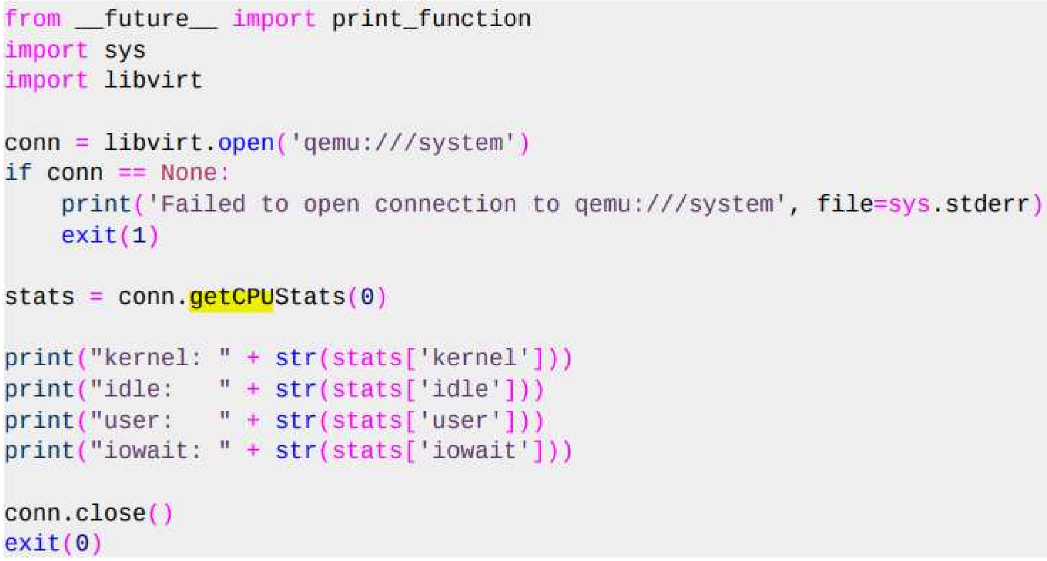
- Libvirt支持了许多常用的功能：
  - Libvirtd：最主要的守护进程，与其他 API 沟通
  - Virt-manager：图形化管理器
  - Guestfish：虚拟机（客户机）文件系统管理
  - Virsh (cli for libvirt)：虚拟化命令行
  - Virt-install / virt-clone / virt-convert
  - Qemu-img：磁盘管理
- Libvirt 的局限性:
  - 目前没有易用的网页接口（web interface），依赖命令行操作
  - Virt-manager 可以与远端（remote）hypervisor 通信，但是 virt-manager 仅能在 linux 下运行
  - 其使用的 XML 格式与其他平台不通用，不易从头构建

附录1：Libvirt 介绍

- 安装libvirt及python支持libvirt-python：

```
$ sudo apt install  pkg-config libvirt-dev
$ pip3 install libvirt-python
```

- 以下示例的目的是获取一个vCPU的运行状态：



```
from __future__ import print_function
import sys
import libvirt

conn = libvirt.open('qemu:///system')
if conn == None:
    print('Failed to open connection to qemu:///system', file=sys.stderr)
    exit(1)

stats = conn.getCPUStats(0)

print("kernel: " + str(stats['kernel']))
print("idle:   " + str(stats['idle']))
print("user:   " + str(stats['user']))
print("iowait: " + str(stats['iowait']))

conn.close()
exit(0)
```

更多libvirt API**请参考官方文档：**

https://libvirt.org/docs/libvirt-appdev-guide-python/en-US/pdf/Version-1.1-Libvirt_Application_Development_Guide_Using_Python-en-US.pdf