A Journey in the OpenStack shining world

Stefano Cozzini

eXact lab srl

stefano.cozzini@exact-lab.it

Adapted for MHPC from the original one presented at INAF ICT meeting

Agenda (part 1)

- Introducing Cloud and OpenStack
  - What is ?
  - Why and when we need it ?
- Overview of the OpenStack services
- ~~Examining OpenStack fundamental services~~
  - ~~Basic bricks of an openStack infrastructure~~
- OpenStack for the users
  - A few easy steps to start your infrastructure

Agenda (part 2)

- Where/when to use OpenStack and why ?
  - Standard info
  - Some comparison
- ~~OpenStack from the sys.adm. P.O.V.~~
  - ~~A more detailed view of the basic components~~
  - ~~Tips to install / manage it~~
- eXact-lab and CNR/IOM cloud infrastructures
  - Our ideas and our efforts

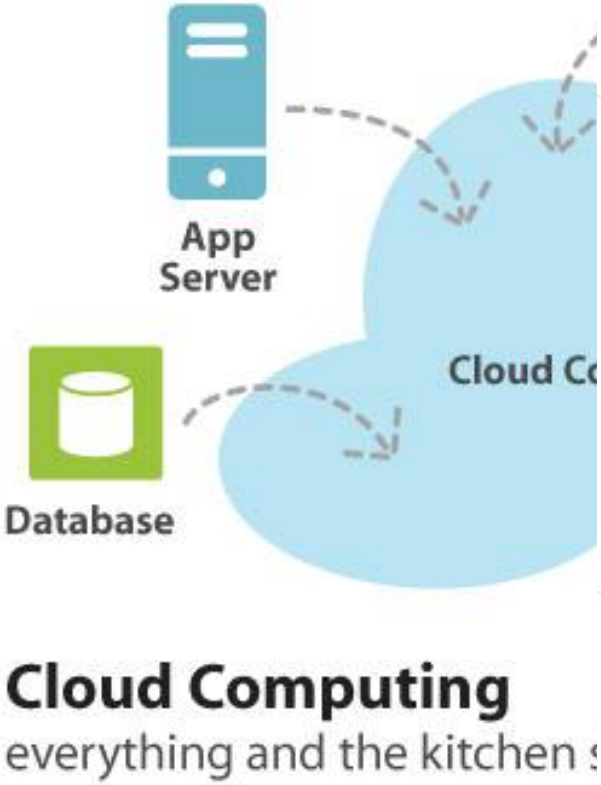
Acknowledge

Slides and materials to arrange them are mainly coming from openstack site, presentation/documents around the net and from contribution of a few people

- Antonio Messina (UZH, Zurich)
- Arturo Sandrigo/Francesco De Giorgi (eXact-lab)
- Giuseppe Brandino (eXact-lab)

Cloud Computing 101

Cloud computing 101



TECNOLOGIA

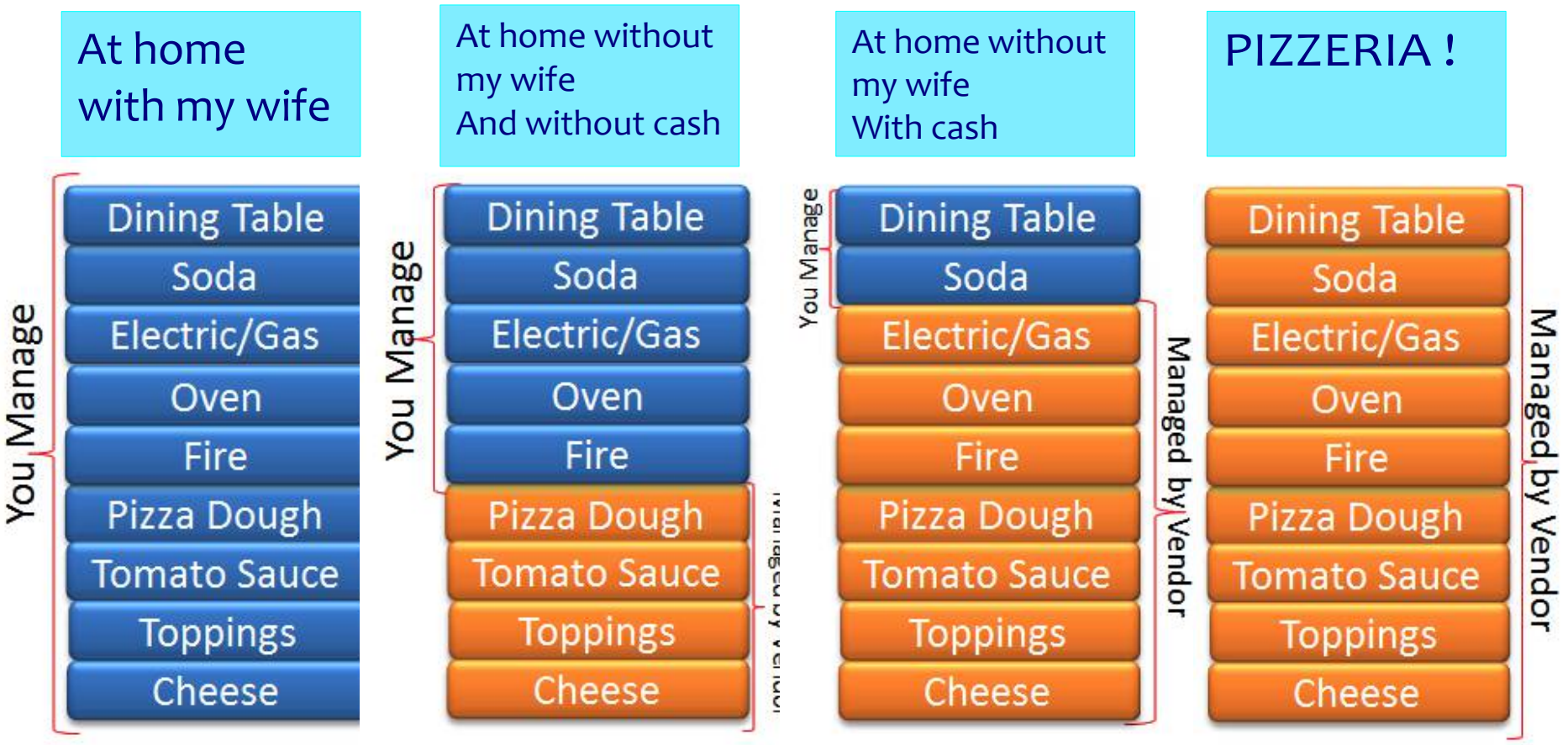
Ora l' Europa punta sul "clound computing"

Ecco l'agenda digitale Ue:
con standard comuni, obblighi chiari e compatibilità provider sono previsti 2,5 milioni di posti di lavoro entro il 2020

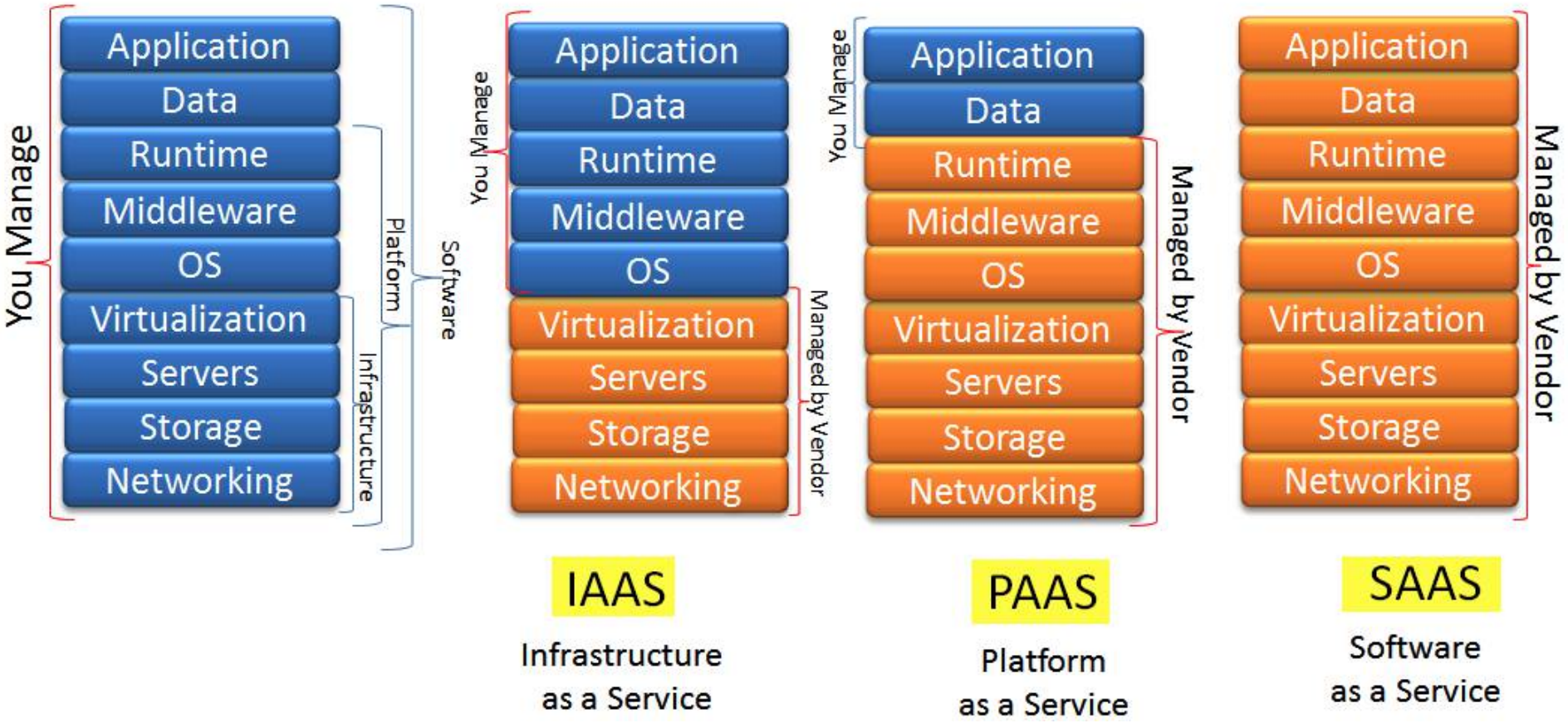
Cloud: basic definition:

- A model of computation and data storage based on “pay as you go” access to “unlimited” remote data center capabilities
- A cloud infrastructure provides a framework to manage scalable, reliable, on-demand access to applications
- Cloud services provide the “invisible” backend to many of our mobile applications
- High level of elasticity in consumption
- Historical roots in today’s Internet apps
  - Search, email, social networks
  - File storage (Live Mesh, Mobile Me, Flicker, …)

When I want pizza..



When I manage IT stuff...



Taken and Slightly changed from
https://cloudcelebrity.wordpress.com/2011/11/22/introduction-to-cloud-services-iaas-paas-saas/

What is cloud ?

- A cloud infrastructure is:

  An infrastructure to provide users with the most flexible way to allocate computational power and storage space (and any other IT services)

quickly provide a customized IT infrastructure

integrate infrastructure provisioning in your data-analysis.

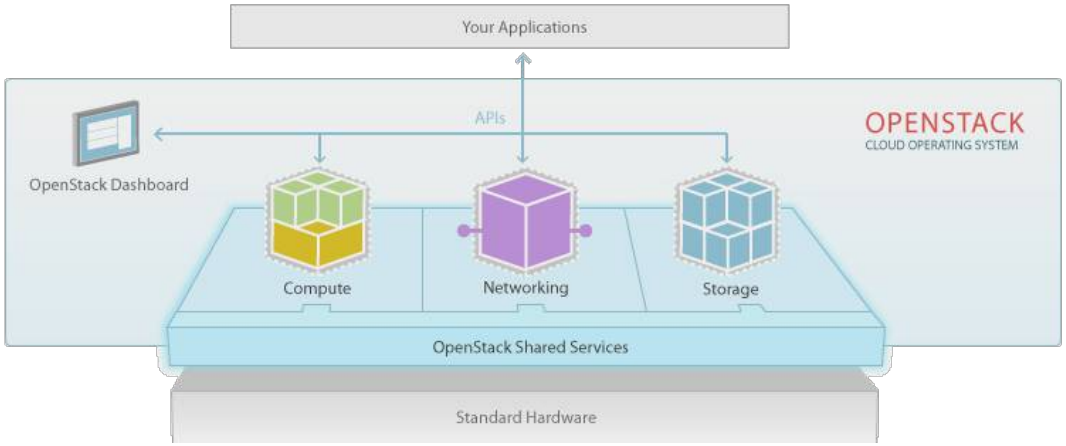
Virtualization vs Cloud

"You are providing cloud services when your users do not bother your sys. Adm. to setup his/her virtual machine.."

"The difference is that a true cloud provides self-service capability, elasticity, automated management, scalability and pay-as you go service that is not inherent in virtualization."

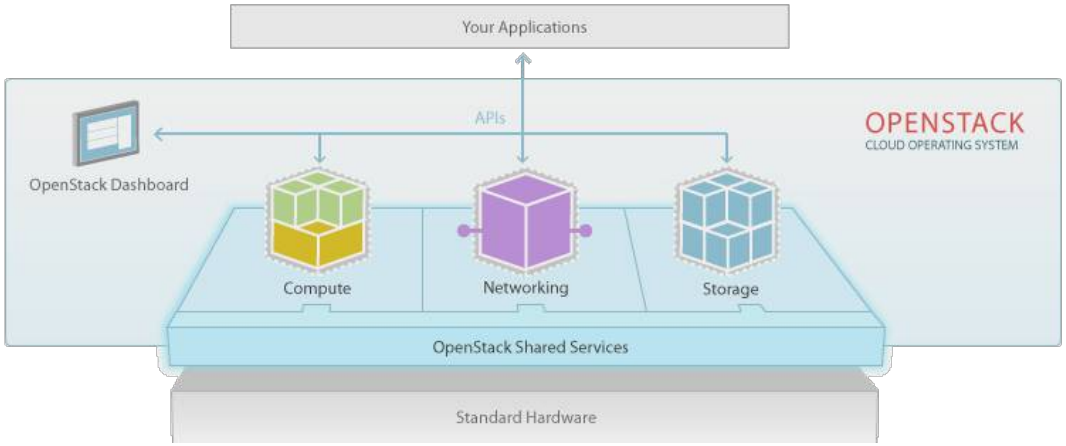
Introducing OpenStack

What is OpenStack?



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

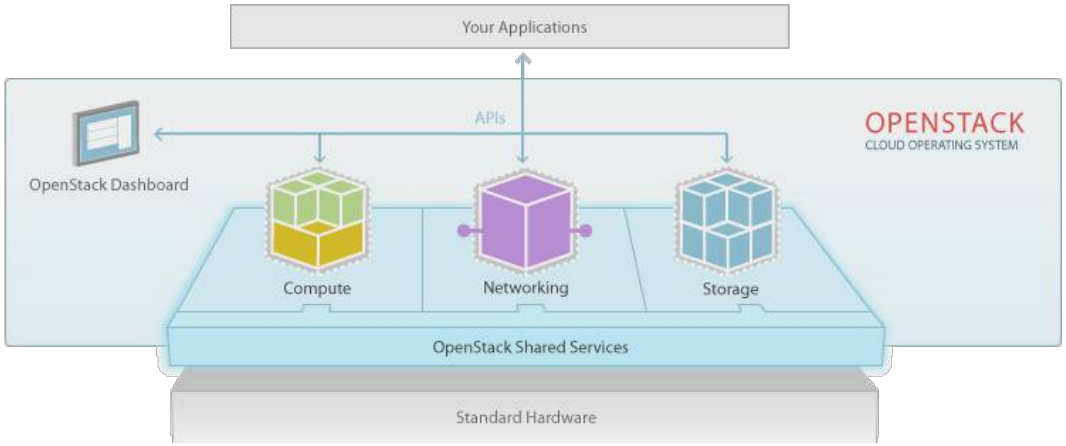
What is OpenStack?



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

It is a complex piece of software plus a buzzword

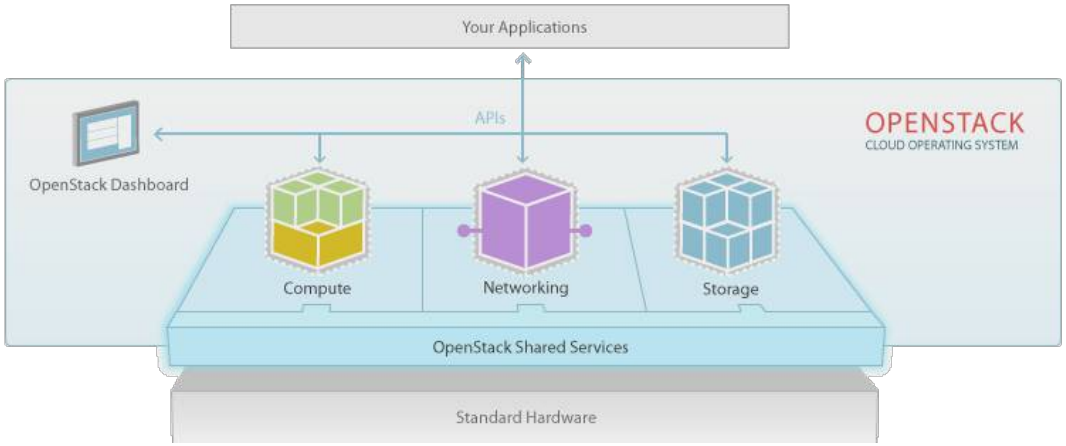
What is OpenStack?



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

It is scalable (thousands of nodes)

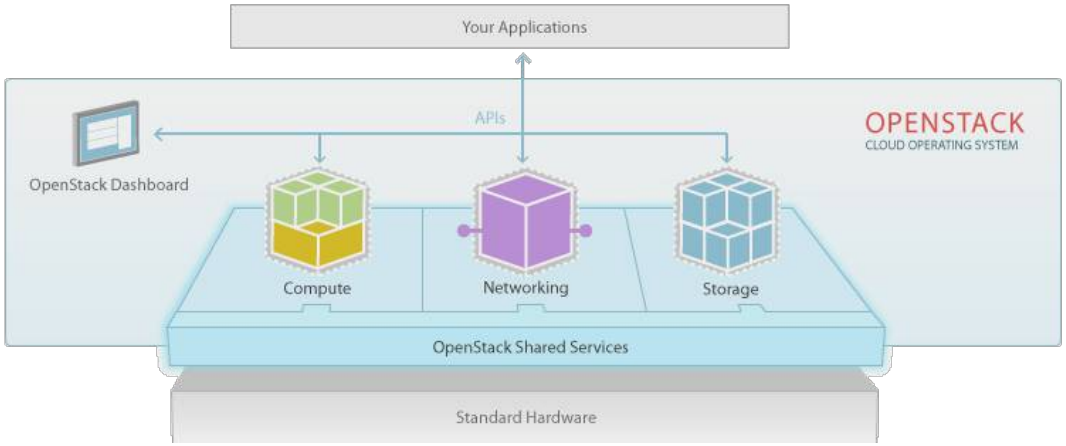
What is OpenStack?



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

Not only CPUs but also storage and network

What is OpenStack?



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

User-centric

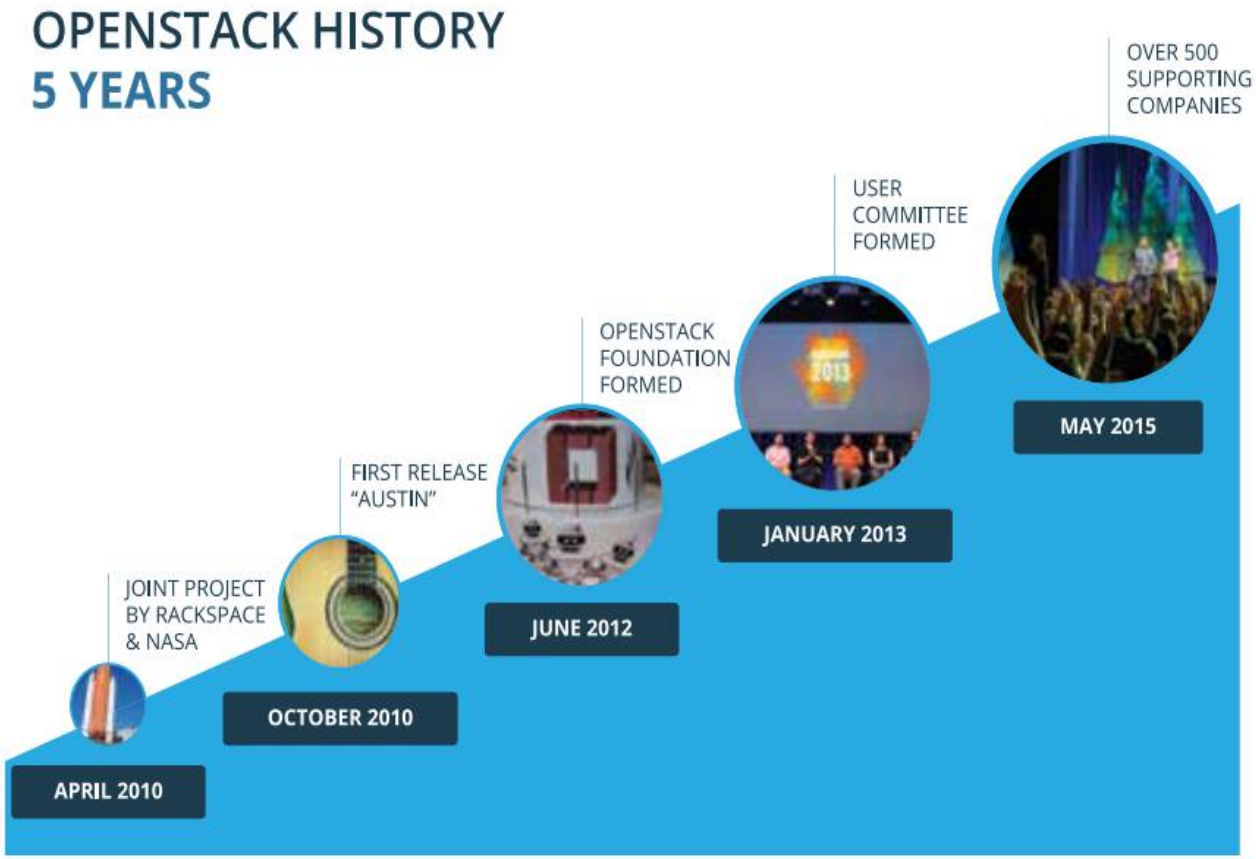
OpenStack basic ideas

- The OpenStack project is an open source cloud computing platform that supports all types of cloud environments.
- The project aims for simple implementation, massive scalability, and a rich set of features.
- Cloud computing experts from around the world contribute to the project.
- OpenStack provides an Infrastructure-as-a-Service (IaaS) solution through a variety of complemental services.
- Each service offers an application programming interface (API) that facilitates this integration

OpenStack Capabilities

- Virtual machines (VMs) on demand
  - provisioning
  - snapshotting
- Networks
- Volumes
- Storage for VMs and arbitrary files
- Multi-tenancy
  - quotas for different projects, users
  - user can be associated with multiple projects

OpenStack history



Release Series

OpenStack is developed and released around 6-month cycles. After the initial release, additional stable point releases will be released in each release series. You can find the detail of the various release series here on their series page. Subscribe to the combined release calendar for continual updates.

Series	Status	Initial Release Date	EOL Date
Queens	*Future*	TBD	TBD
Pike	*Future*	TBD	TBD
Ocata	*Under Development*	2017-02-22 (planned)	TBD
Newton	Current stable release, security-supported	2016-10-06	TBD
Mitaka	Security-supported	2016-04-07	2017-04-10
Liberty	EOL	2015-10-15	2016-11-17
Kilo	EOL	2015-04-30	2016-05-02
Juno	EOL	2014-10-16	2015-12-07

First technicalities

- written in Python (plus auxiliary shell scripts)
- Built around independent components
- Highly distributed architecture
  - designed for very big installations
- intrinsic HA of most OpenStack services
  - (MySQL and RabbitMQ have to be properly configured)
- *SQL database used to store persistent data
- RabbitMQ used for RPC and notification
- RESTful APIs for all the services

VM provisioning

- creation is done via Web GUI, CLI or network APIs

  => it can be scripted

- Actual provisioning of the VMs can be delegated to the user.

  => no need to fill up a form or open an issue to create a VM.

- Time to provision a VM can be very short (depending on the type of setup!!)
  - 10 seconds to create
  - 20- 50 seconds to login (depending on the image)

OpenStack philosophy

OpenStack believes in open source, open design, open development, all in an open community so anyone can participate.

The long-term vision for OpenStack is to produce a ubiquitous open source cloud computing platform that meets the needs of public and private cloud providers regardless of size.

OpenStack services control large pools of compute, storage, and networking resources throughout a data center

OpenStack principles

- Open development model: Code reviews and roadmaps are public.The code for OpenStack is freely available under the Apache 2.0 license.
- Open design process: Every six months the development community holds a design summit to gather requirements and write specifications for the upcoming release. The summit is open to the public and attendees include users, operators, developers, and upstream project personnel.
- Open community: OpenStack is dedicated to producing a healthy,vibrant, and active developer and user community. Most decisions are made by consensus. All processes are documented, open and transparent.

OpenStack community

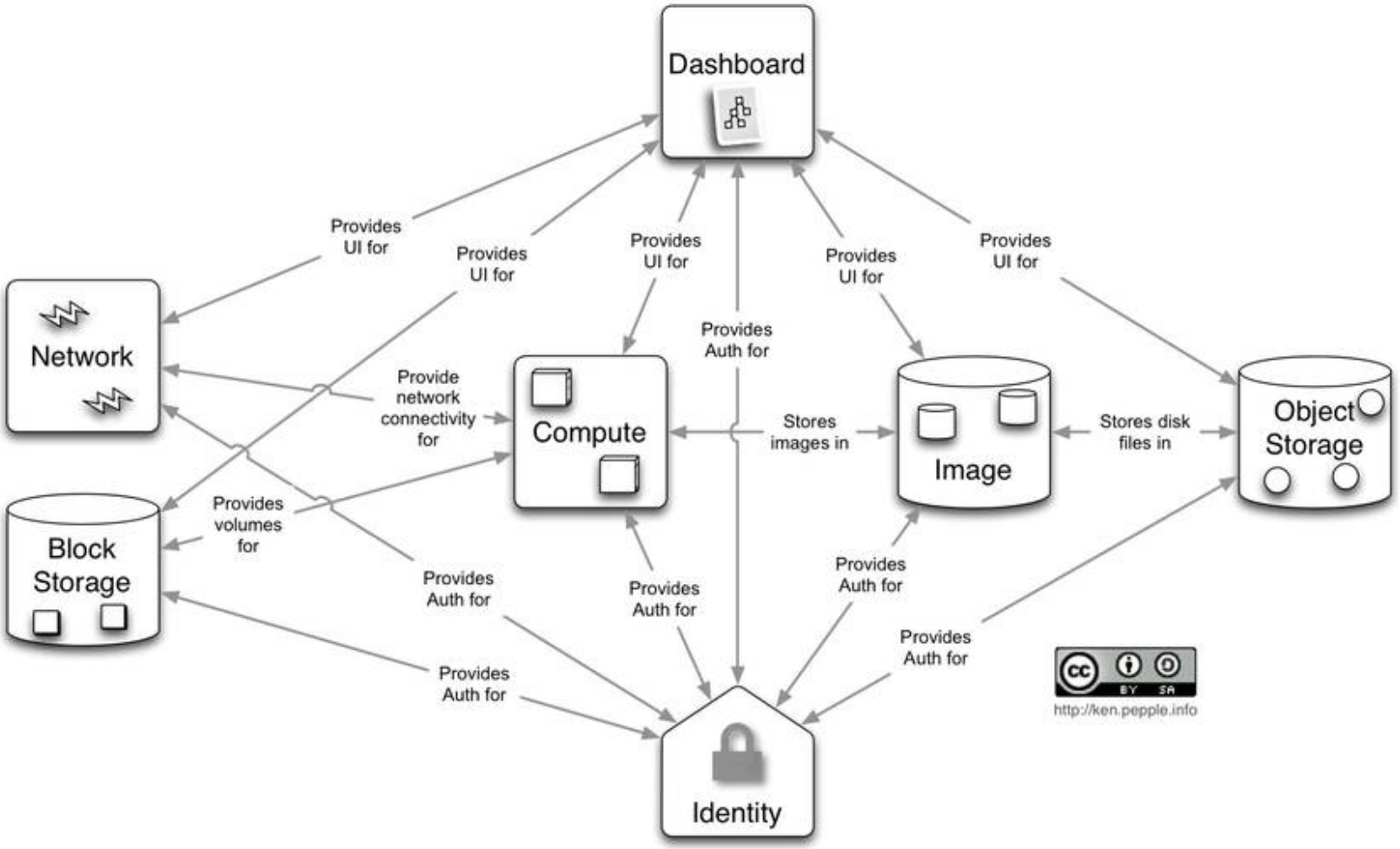
- Created by a large growing community
  - 24,000 people
  - 495 supporting companies
- over 20 million lines of code
  - One recent ("Kilo") release stats:
    - 1,500 developers
    - 170 different companies
    - 20,000 patches

OpenStack introduction recap

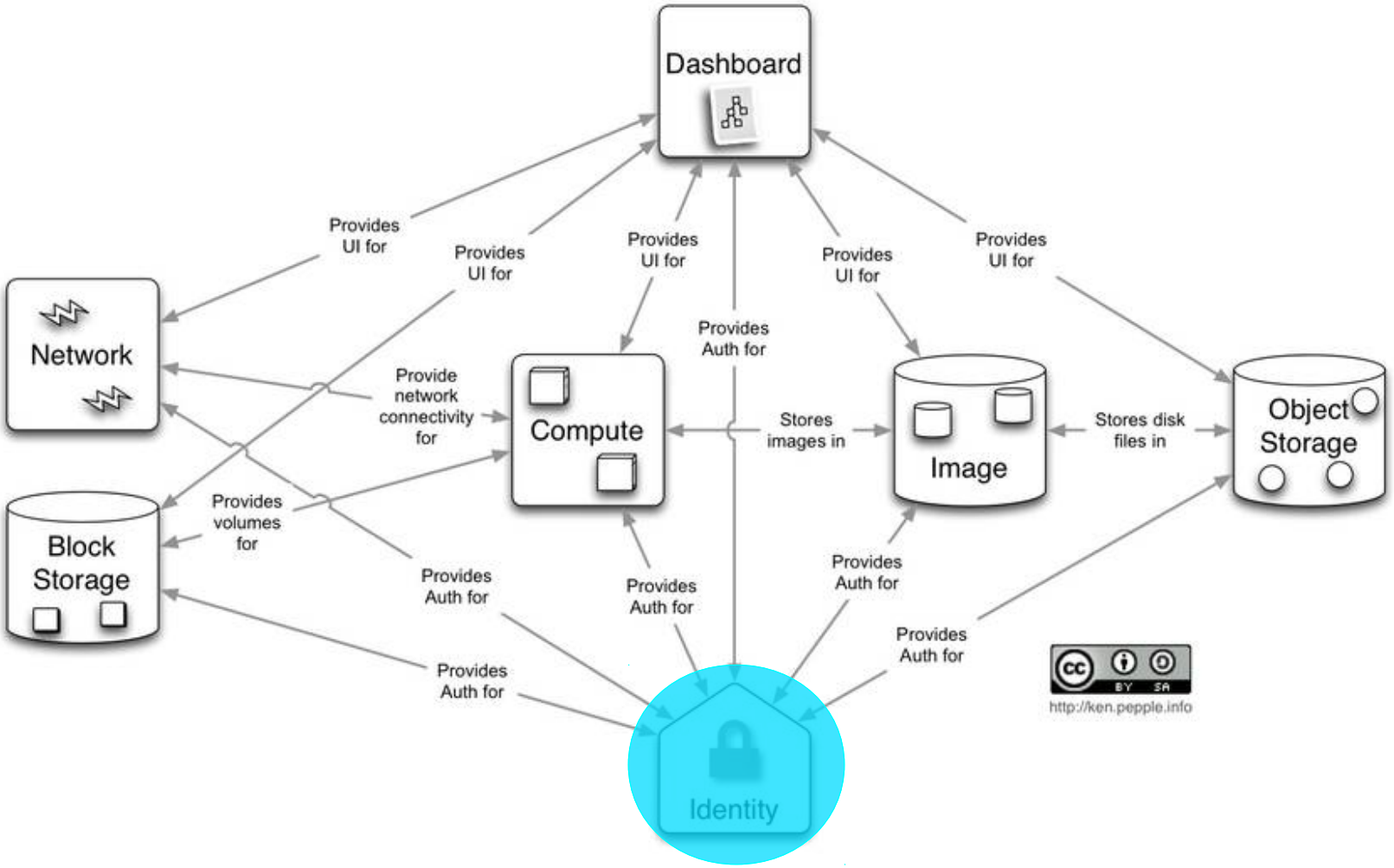
- open source software for building clouds
- release cycle is every 6 months
  - Last one (Newton) out on 15/10/16
- an umbrella over multiple independent programs (components)
- All components talk via RESTful API
- Most components have dedicated DB (SQL) and Message Queuing system
- some talk to 3rd party components using their native APIs

An overview of OpenStack services

Logical view

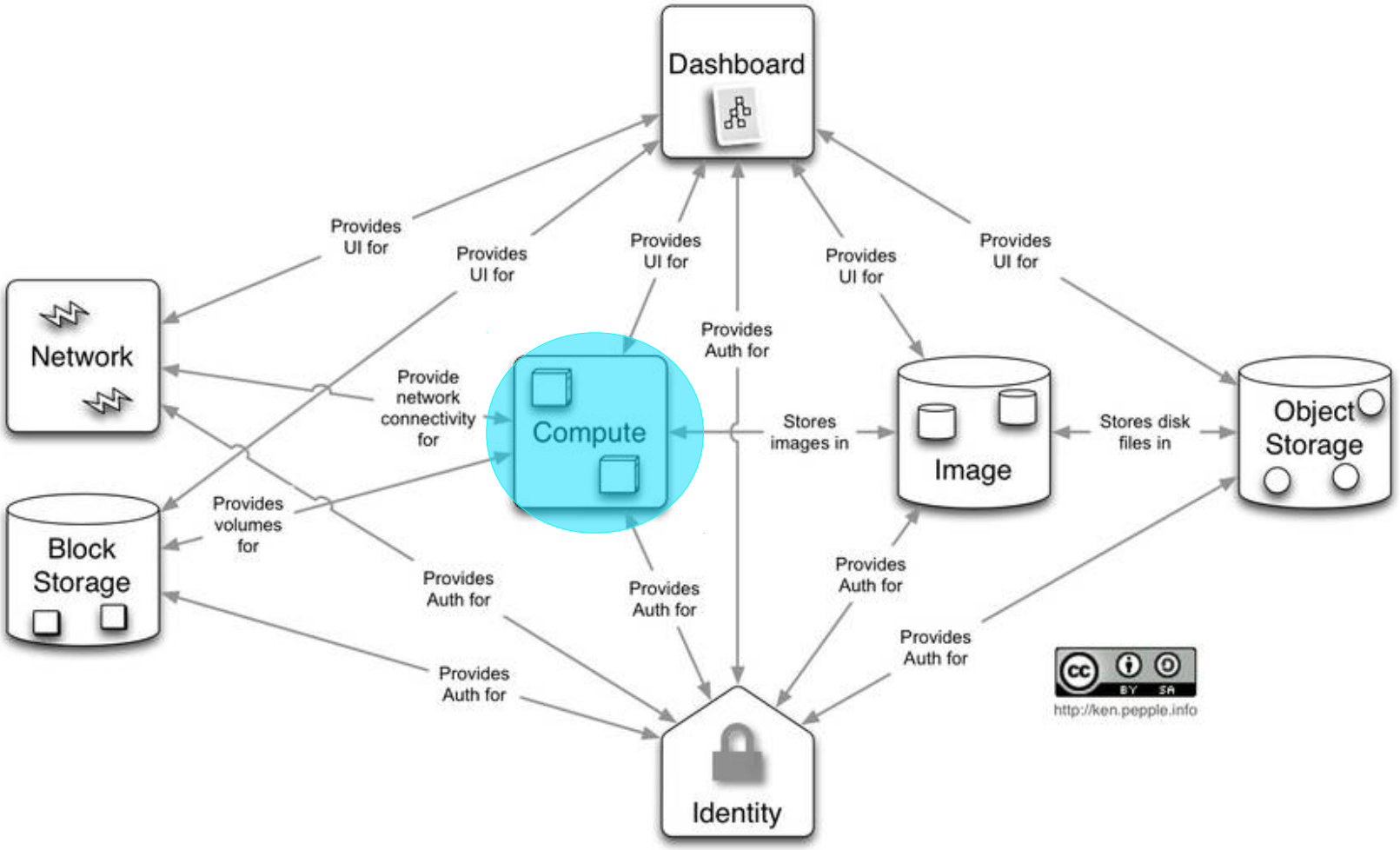


Logical view



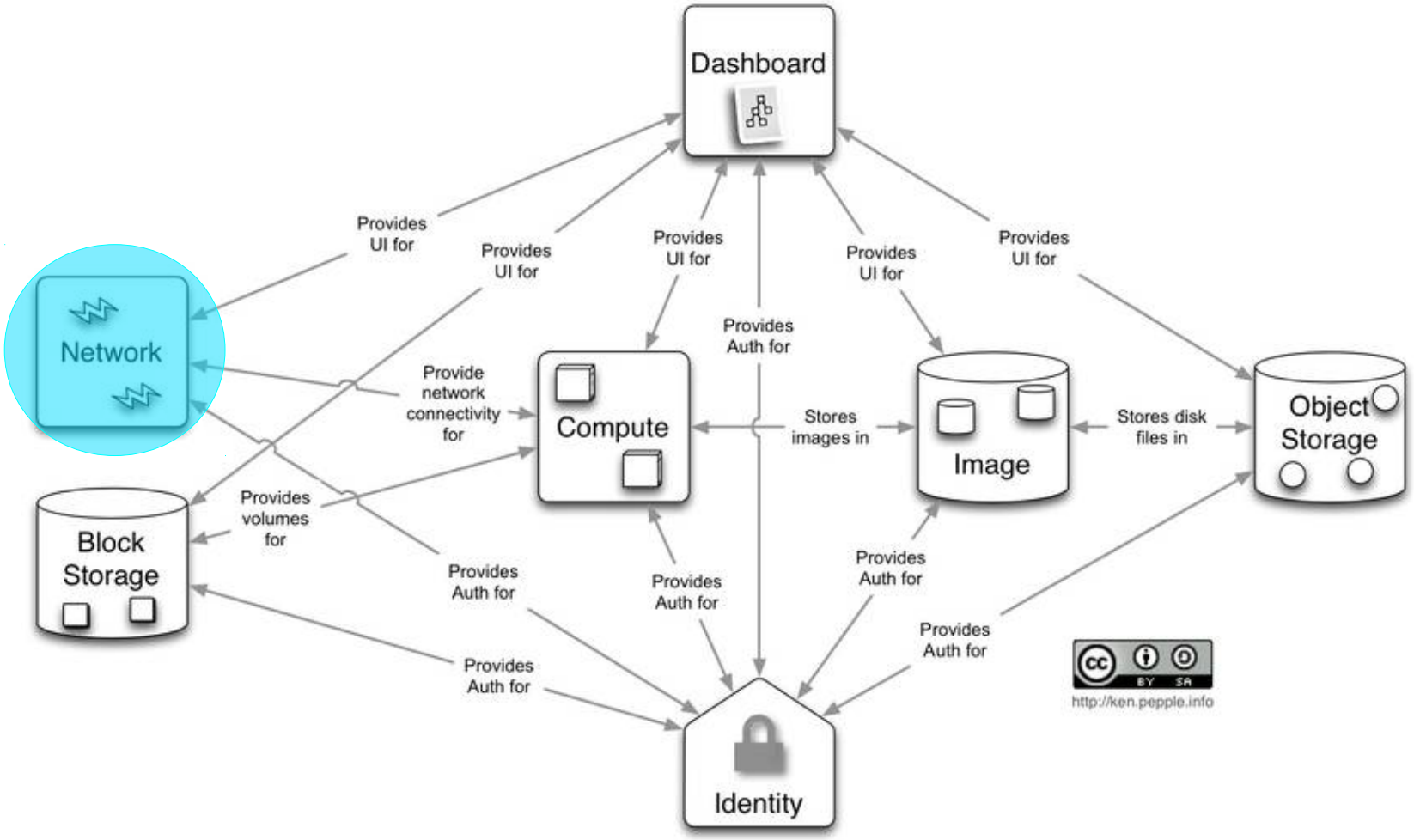
KEYSTONE: authentication service

Logical view



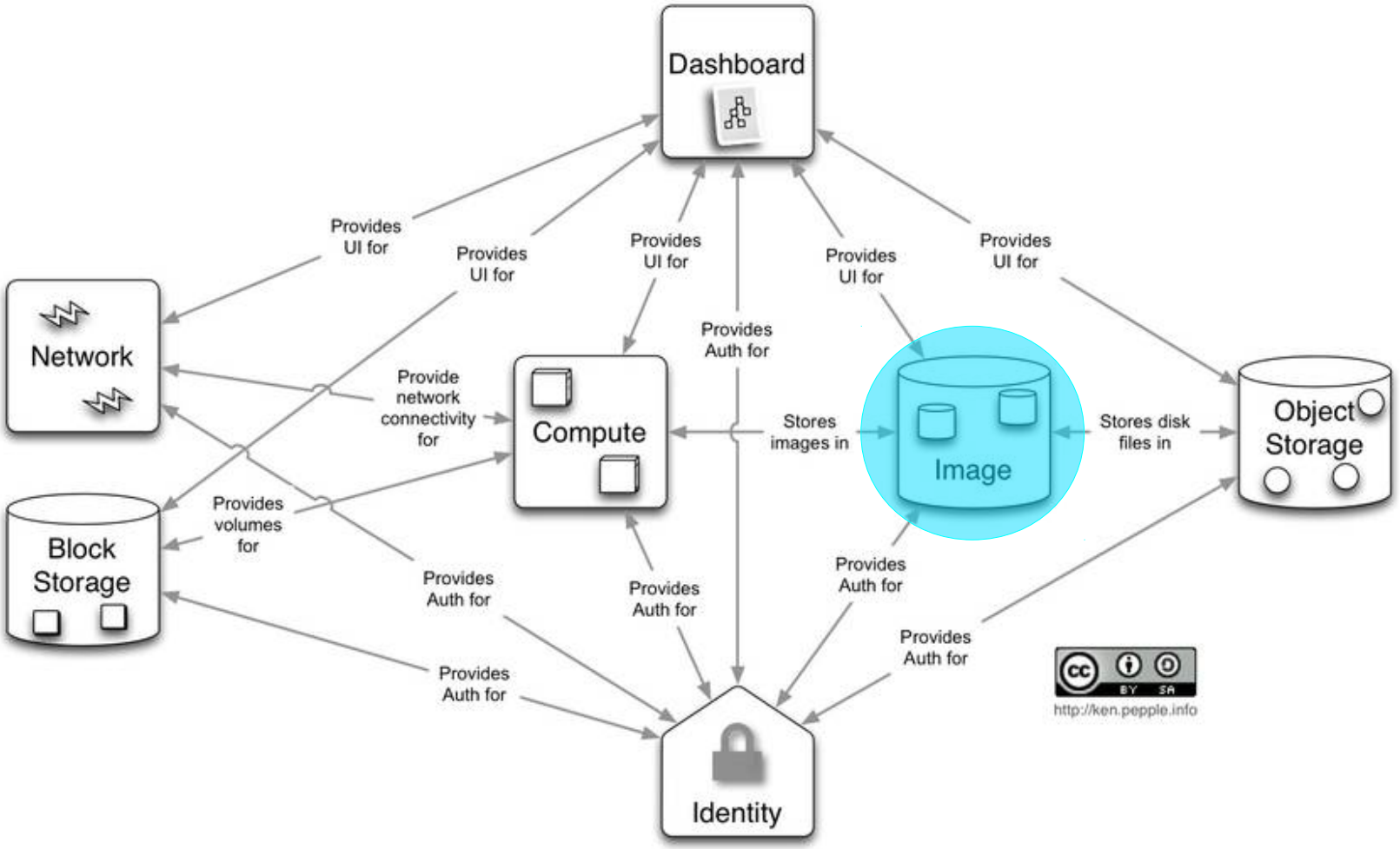
NOVA: computational services

Logical view



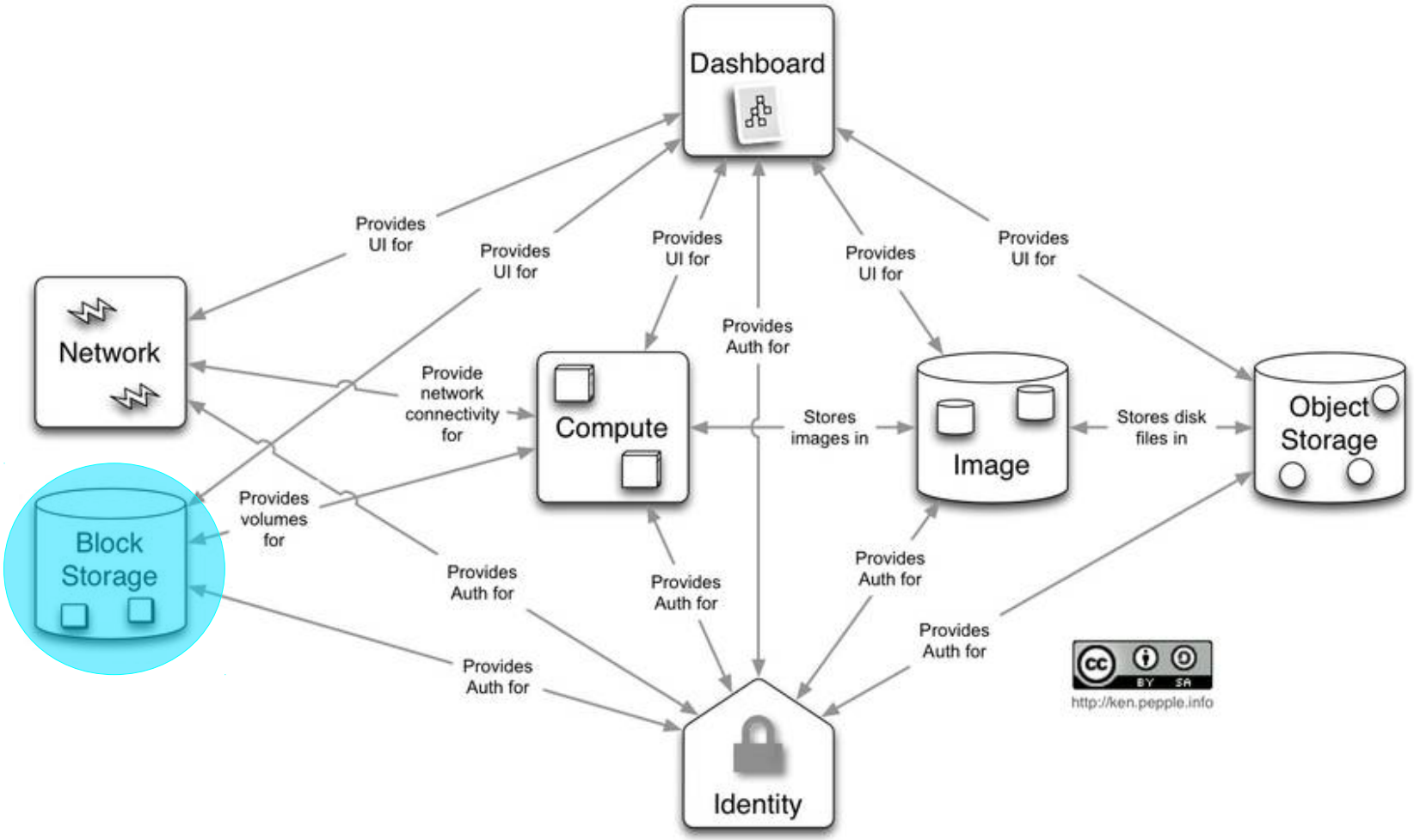
NEUTRON: network services

Logical view



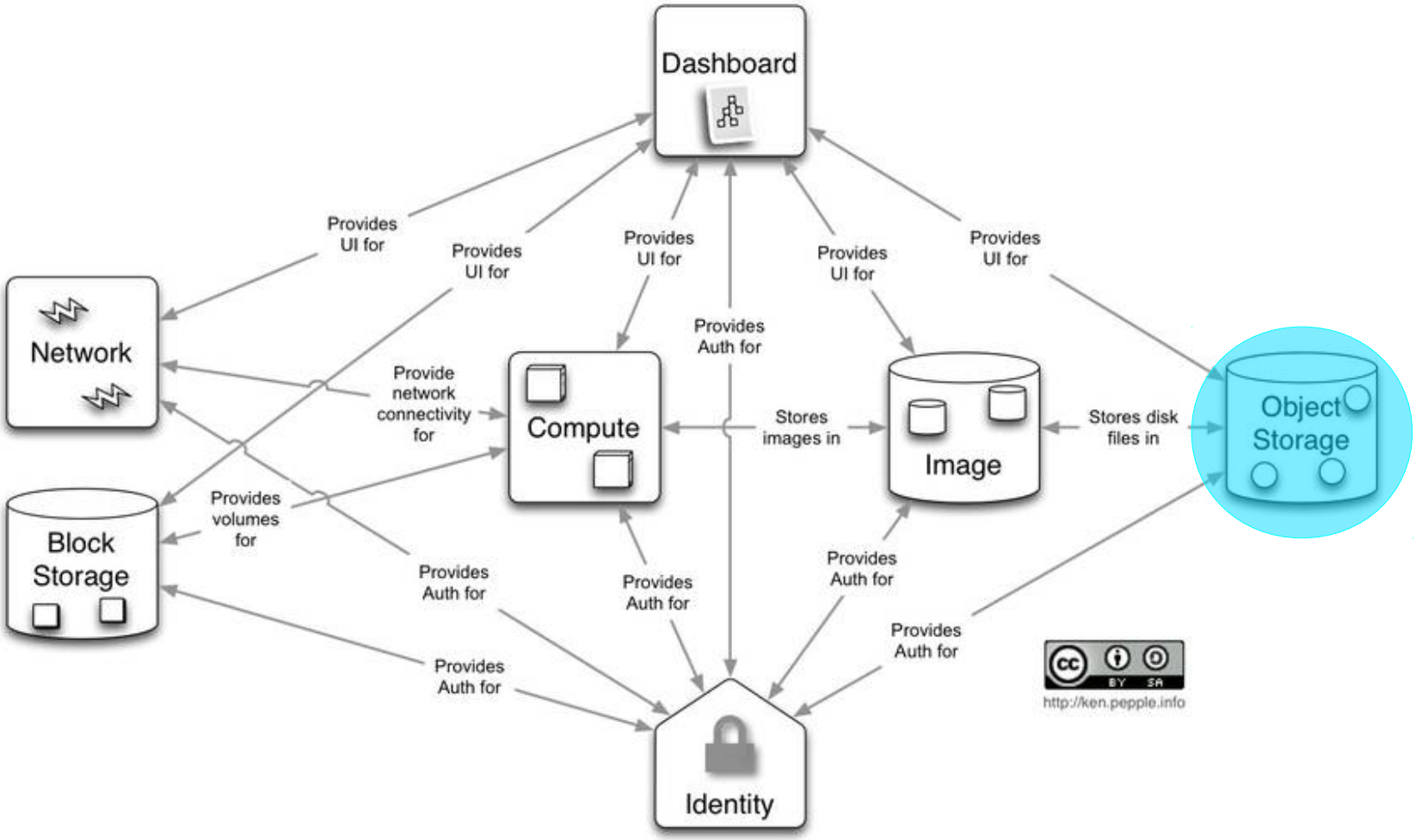
GLANCE: image services

Logical view



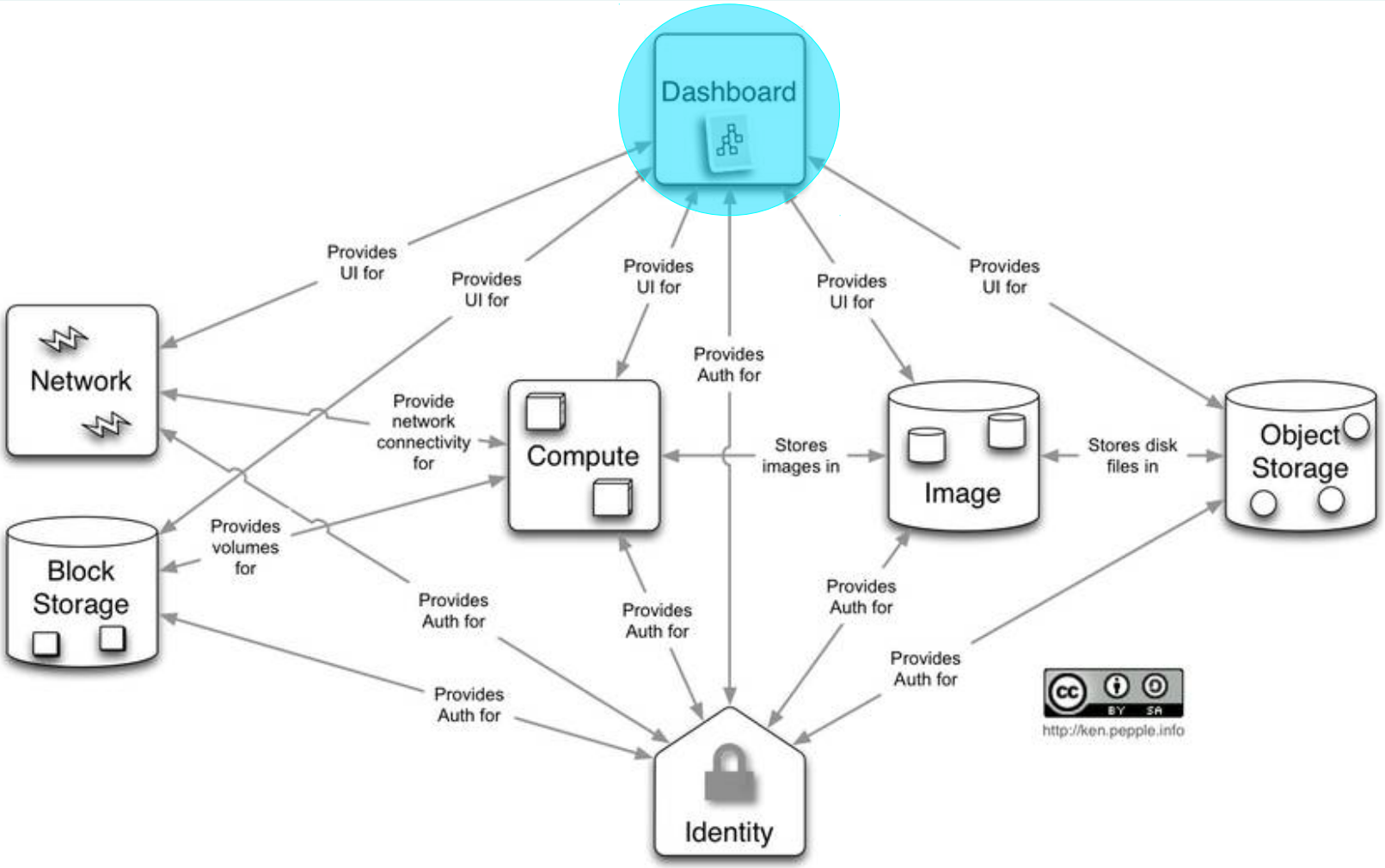
CINDER: block storage services

Logical view



SWIFT: object storage services

Logical view

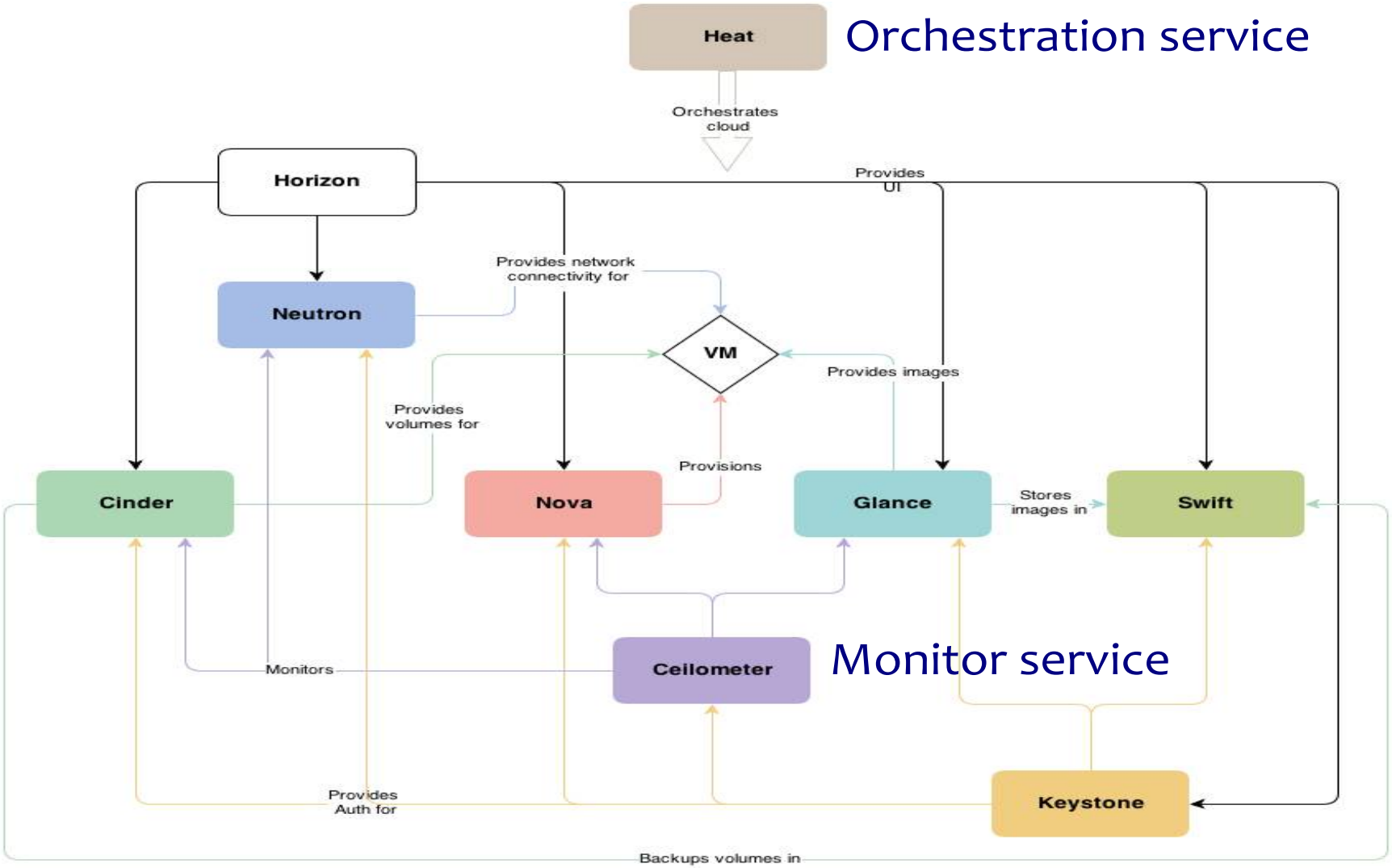


HORIZON: web user interface

List of services so far.

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

There are more (Kilo release)



Components (Mitaka Release)

- Telemetry – Ceilometer
- Orchestration - Heat
- Database - Trove
- Data Processing - Sahara
- Bare-Metal Provisioning – Ironic

From the installation guide

Table 1.1. OpenStack services

Service	Project name	Description
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.
Compute	Nova	Manages the lifecycle of compute instances in an OpenStack environment. Responsibilities include spawning, scheduling and decommissioning of virtual machines on demand.
Networking	Neutron	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.
Storage		
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. In this case, it writes objects and files to multiple drives, ensuring the data is replicated across a server cluster.
Block Storage	Cinder	Provides persistent block storage to running instances. Its pluggable driver architecture facilitates the creation and management of block storage devices.
Shared services		
Identity service	Keystone	Provides an authentication and authorization service for other OpenStack services. Provides a catalog of endpoints for all OpenStack services.
Image service	Glance	Stores and retrieves virtual machine disk images. OpenStack Compute makes use of this during instance provisioning.
Telemetry	Ceilometer	Monitors and meters the OpenStack cloud for billing, benchmarking, scalability, and statistical purposes.
Higher-level services		
Orchestration	Heat	Orchestrates multiple composite cloud applications by using either the native HOT template format or the AWS CloudFormation template format, through both an OpenStack-native REST API and a CloudFormation-compatible Query API.
Database service	Trove	Provides scalable and reliable Cloud Database-as-a-Service functionality for both relational and non-relational database engines.
Data processing service	Sahara	Provides capabilties to provision and scale Hadoop clusters in OpenStack by specifying parameters like Hadoop version, cluster topology and nodes hardware details.

Examining OpenStack fundamental services

Keystone

- entry point for OpenStack API.
- stores authentication information (*users, passwords, tokens, projects, roles*)
- Holds a catalog of available services and their endpoints.
- Can use different backends (SQL database, LDAP)

Keystone: Identity Management

- User
- Credentials
- Token
  - Associated with a user, an arbitrary bit of text that is used to access resources
- Group of users
- Project
  - Synonym to tenant
- Role
  - Assigned to users or groups for projects
- Domain
  - Higher level of hierarchy – users and projects belong to domains

Keystone: Service Catalog

- Service
  - OpenStack service, such as Compute(Nova), Object storage (swift), Image Service (Glance).
- Endpoint
  - A network accessible address, usually described by URL, from where you can access an OpenStack service
- Role
  - A set of requirements to perform an action over the end point.

Nova: OpenStack compute

Manages the lifecycle of compute instances in an OpenStack environment.

Responsibilities include spawning, scheduling and decommissioning of virtual machines on demand.

Nova components

- Nova-api:

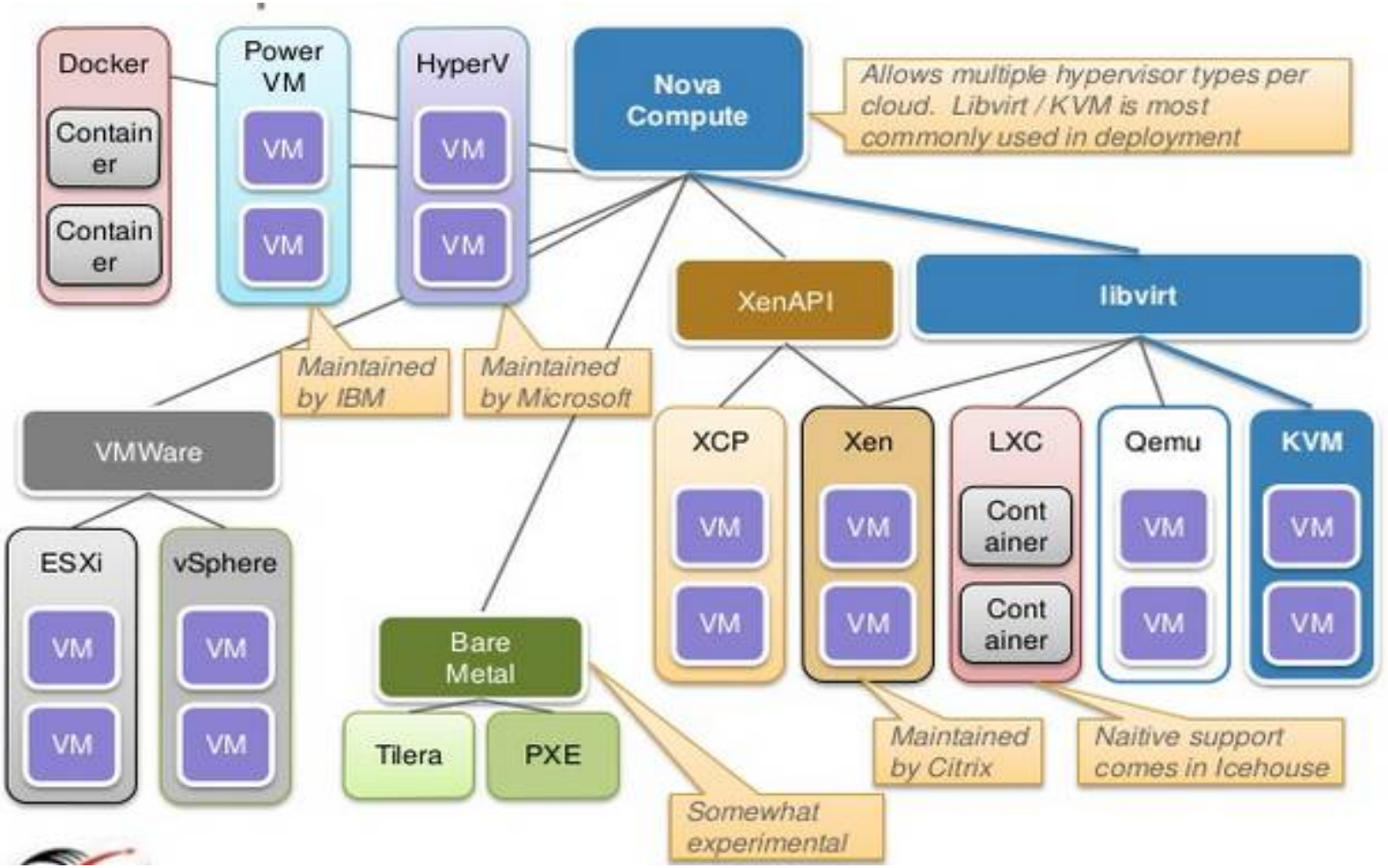
  Web API frontend, accepts requests,validates them and contact other services if needed.

- Nova-scheduler
  - decides where to start an instance

- Nova-compute
  - running on each compute node, interacts with the hypervisor and actually starts the vm.

Nova components could run on different physical server

Nova compute drivers



Nova Compute Notes

- Functionality provided by drivers is not 100% similar.
- Exact "run_instance" flow depends on the driver implementation.
- Most features are developed and tested on KVM.
- Our experience is limited to KVM (no need to explore other so far..).
- Our next step: docker (see later)

Glance: OpenStack image

Service responsible of storing image information and, optionally, image files.

- Holds information about available images.
- Optionally allow to download and upload images.
- Images can be stored on different backends
- (RDB, S3, Swift, filesystem)

Cinder: block device services

- Creates and export volumes via iSCSI to the compute node.
- Volumes are mounted transparently from the virtual machines.
- Supports multiple storage backends (NFS, LVM, Ceph, GlusterFS but also SAN/NAS devices from IBM, NetApp etc. . .)

OpenStack storage concepts

- Ephimeral storage:
  - Persists until VM is terminated
  - Accessible from within VM as local file system
  - Used to run operating system and/or scratch space
  - Managed by Nova
- Block storage:
  - Persists until specifically deleted by user
  - Accessible from within VM as a block device (e.g. /dev/vdc)
  - Used to add additional persistent storage to VM and/or run operating system
  - Managed by Cinder
- Object storage:
  - Persists until specifically deleted by user
  - Accessible from anywhere
  - Used to add store files, including VM images
  - Managed by Swift

Horizon: the dashboard

- The graphical interface to access, provision and automate cloud-based resources.
- easy to plug in and expose third party products and services, such as billing, monitoring and additional management tools.
- also brandable for service providers and other commercial vendors who want to make use of it.
- A self-service portal to provision their own resources within the limits set by administrators.
- it provides to cloud administrator an overall view of the size and state of your cloud. You can create users and projects, assign users to projects and set limits on the resources for those projects.

Neutron: network service

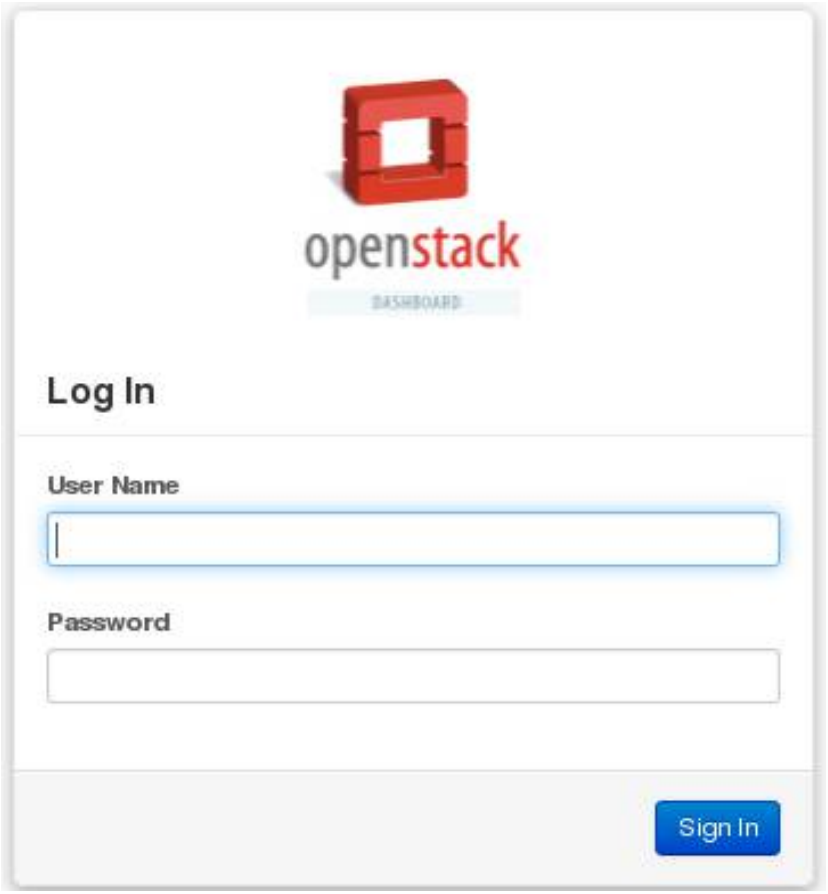
Service responsible of creating and managing networks.

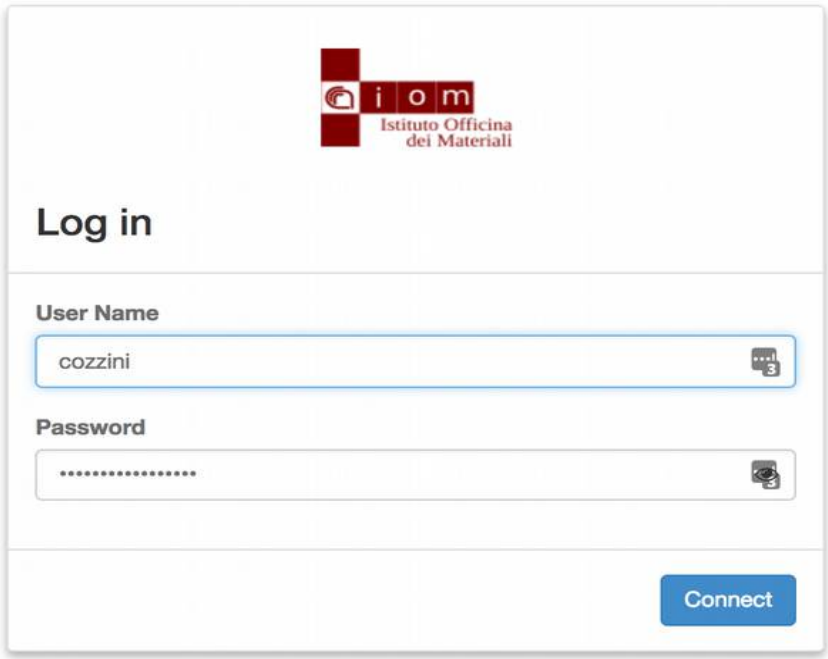
Very feature rich:

VERY COMPLEX TO MANAGE

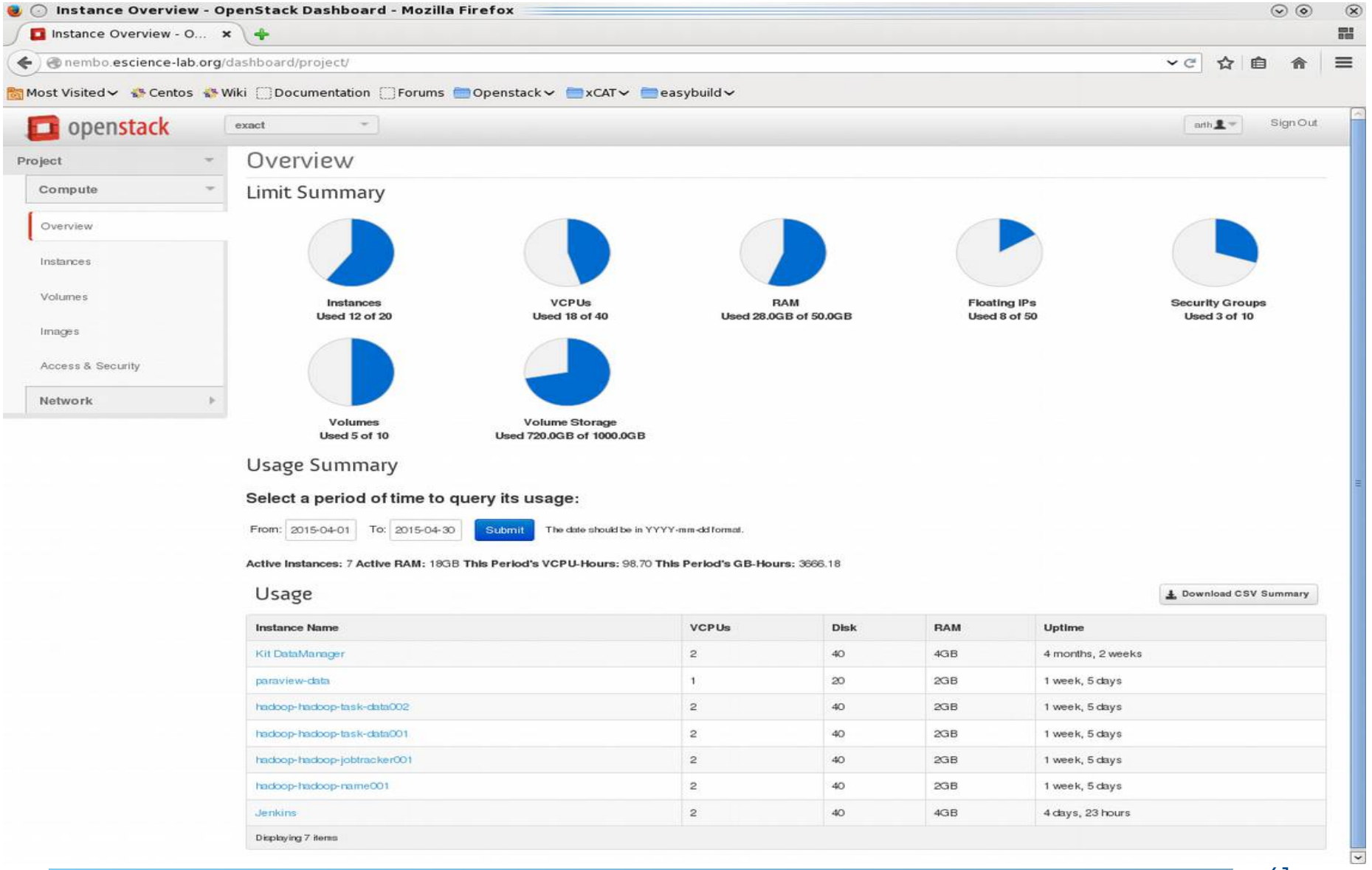
- L2 and L3 networks.
- Allow creation of multiple networks and subnets.
- Plugin architecture.
- Supports advanced network services (Load Balancer, Firewall, DNS as a service)
- Integrates with network devices (Cisco, Brocade. . .)

OpenStack dashboard login





OpenStack dashboard login

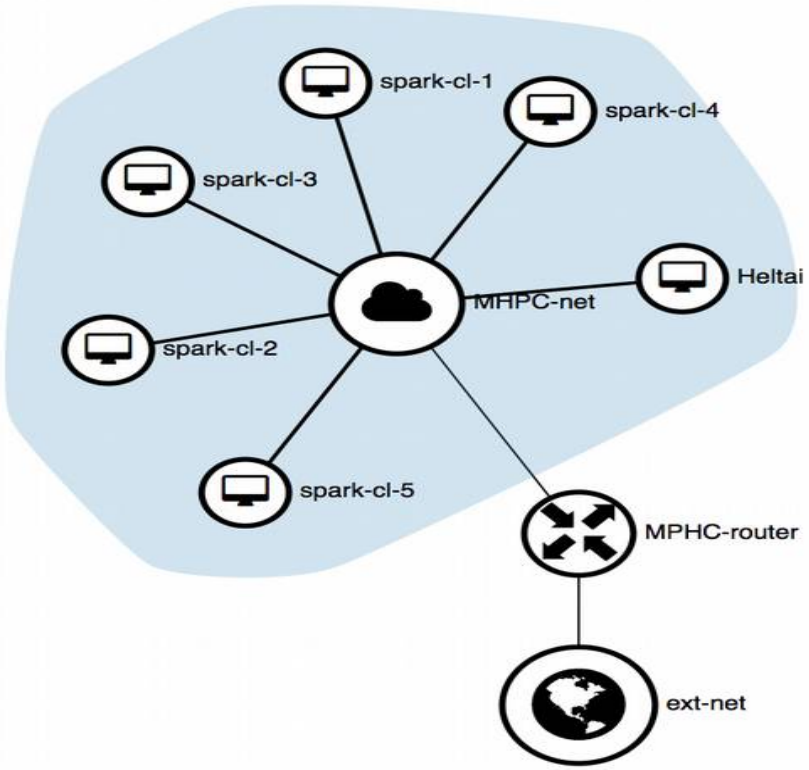


Using OpenStack

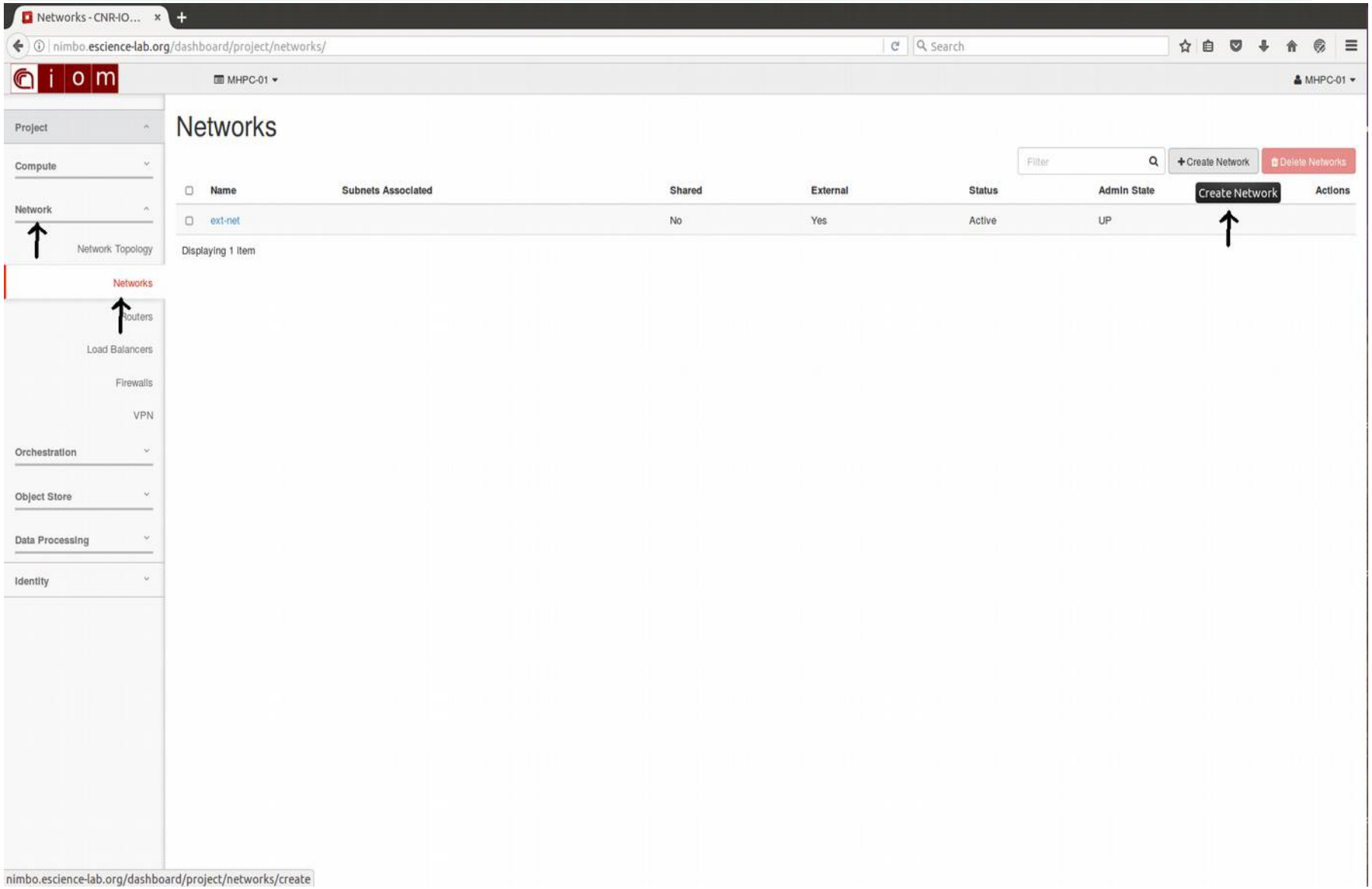
Create a VM

- Request Provisioning From UI
- A three steps action:
  - Login to Horizon
  - Specify parameters of VM
    - VM Name
    - Image (OS type)
    - Flavor (specifies CPU, Memory, Disk)
    - Network (required for Neutron)
    - Select/generate SSH keys for remote login
    - Optional (Persistent volumes, comments, etc.)
  - Select/Press "Launch" button

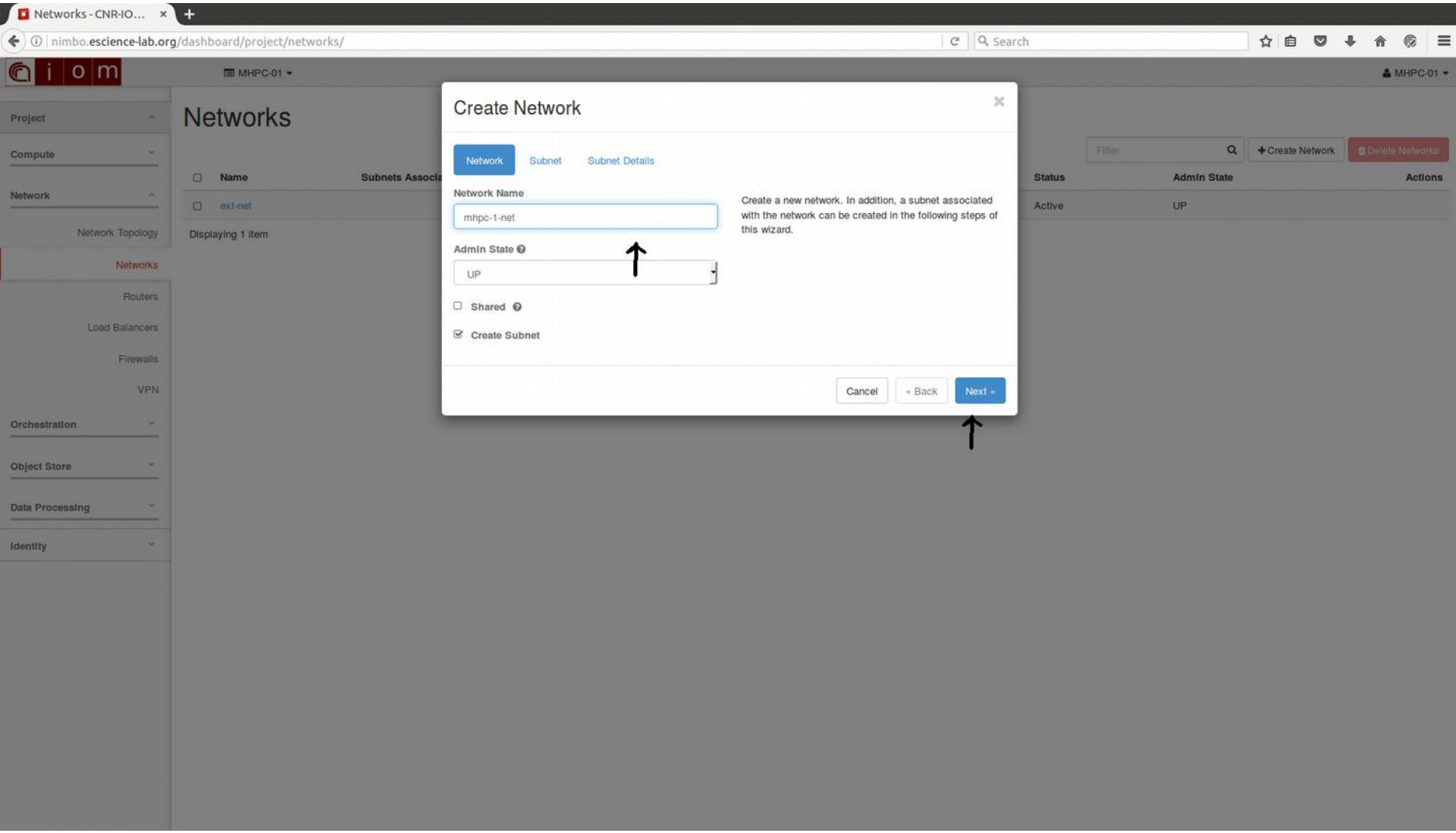
Network topology



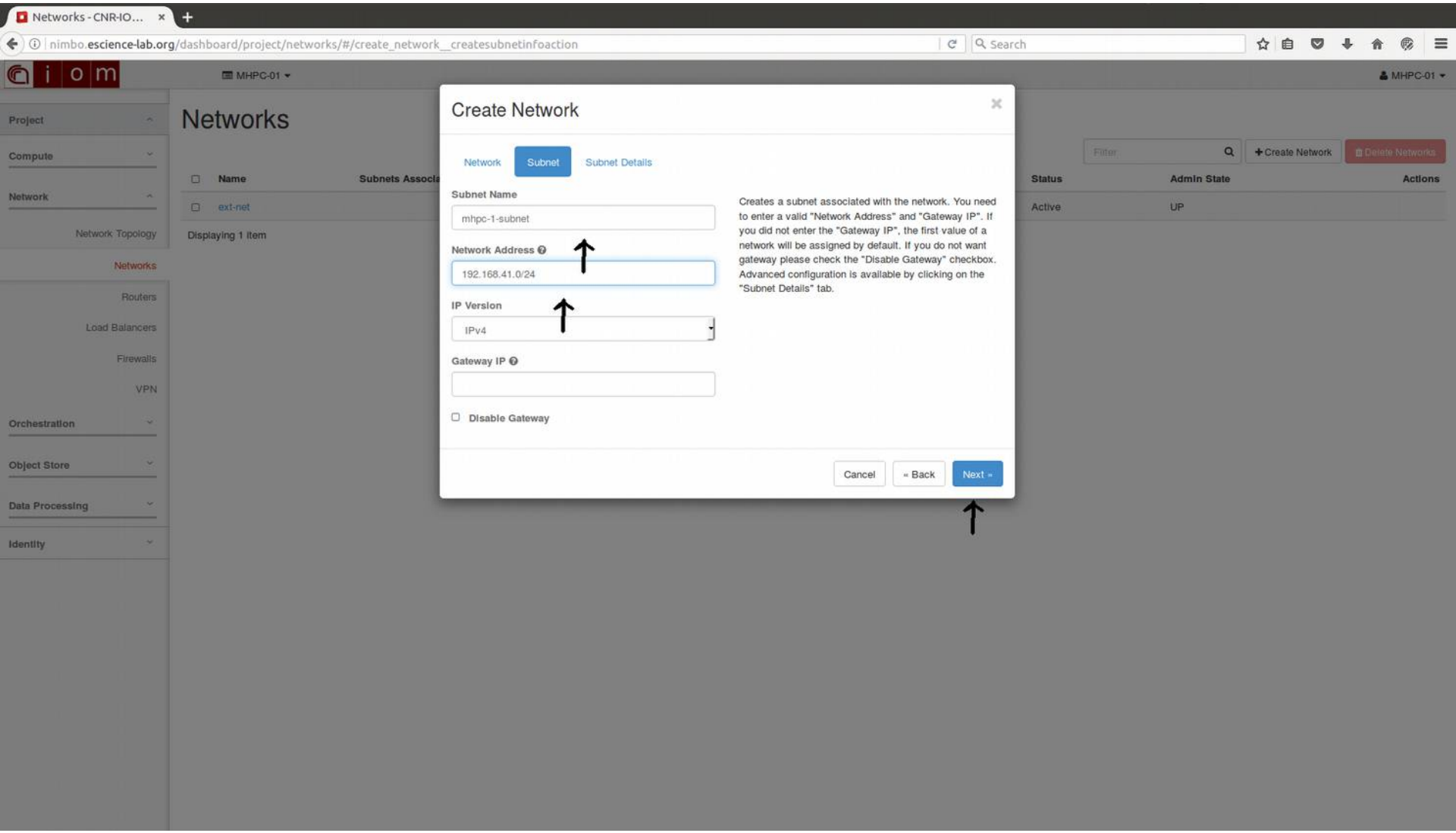
Create a private network: step1



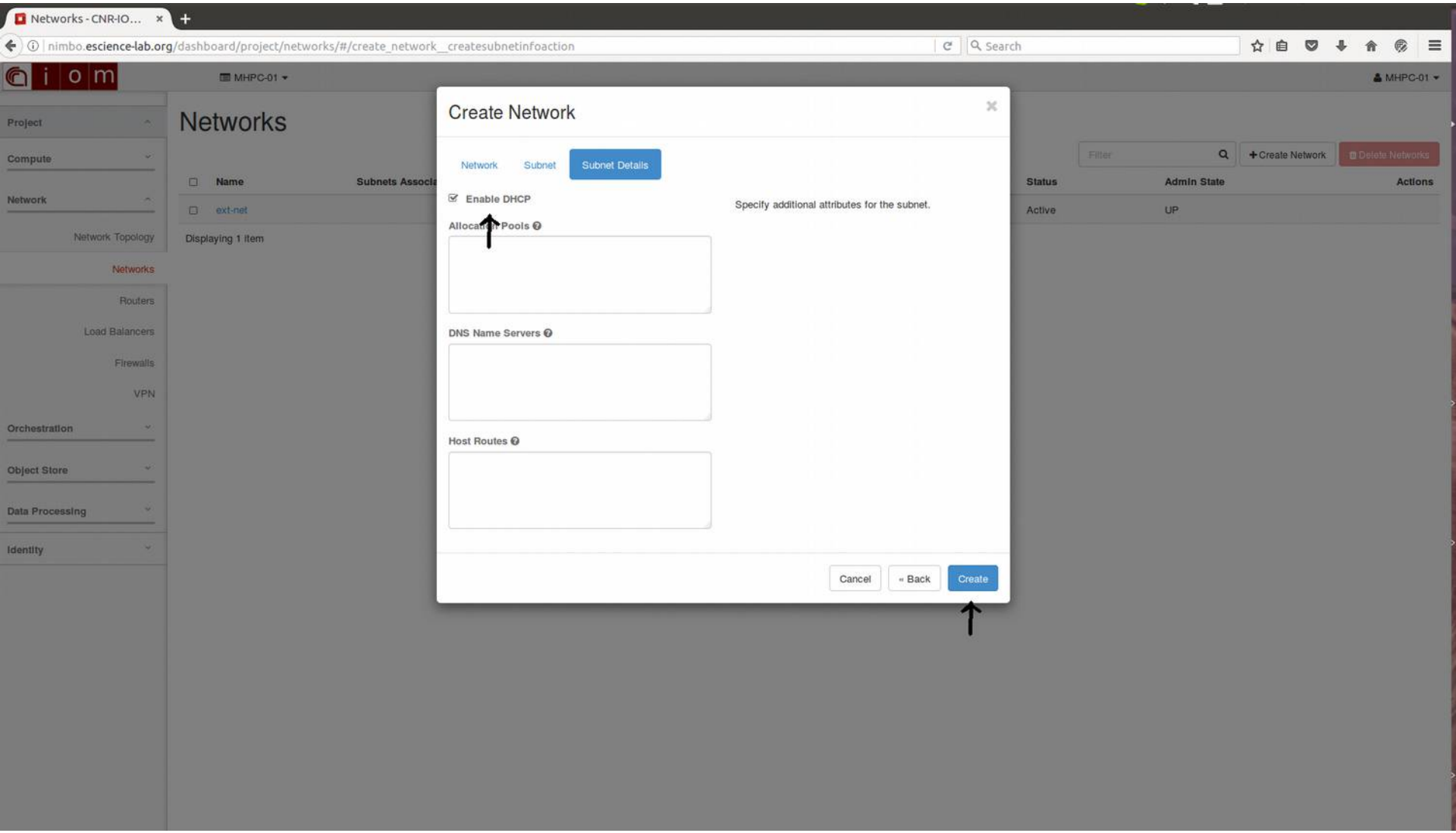
Create a private network: step2



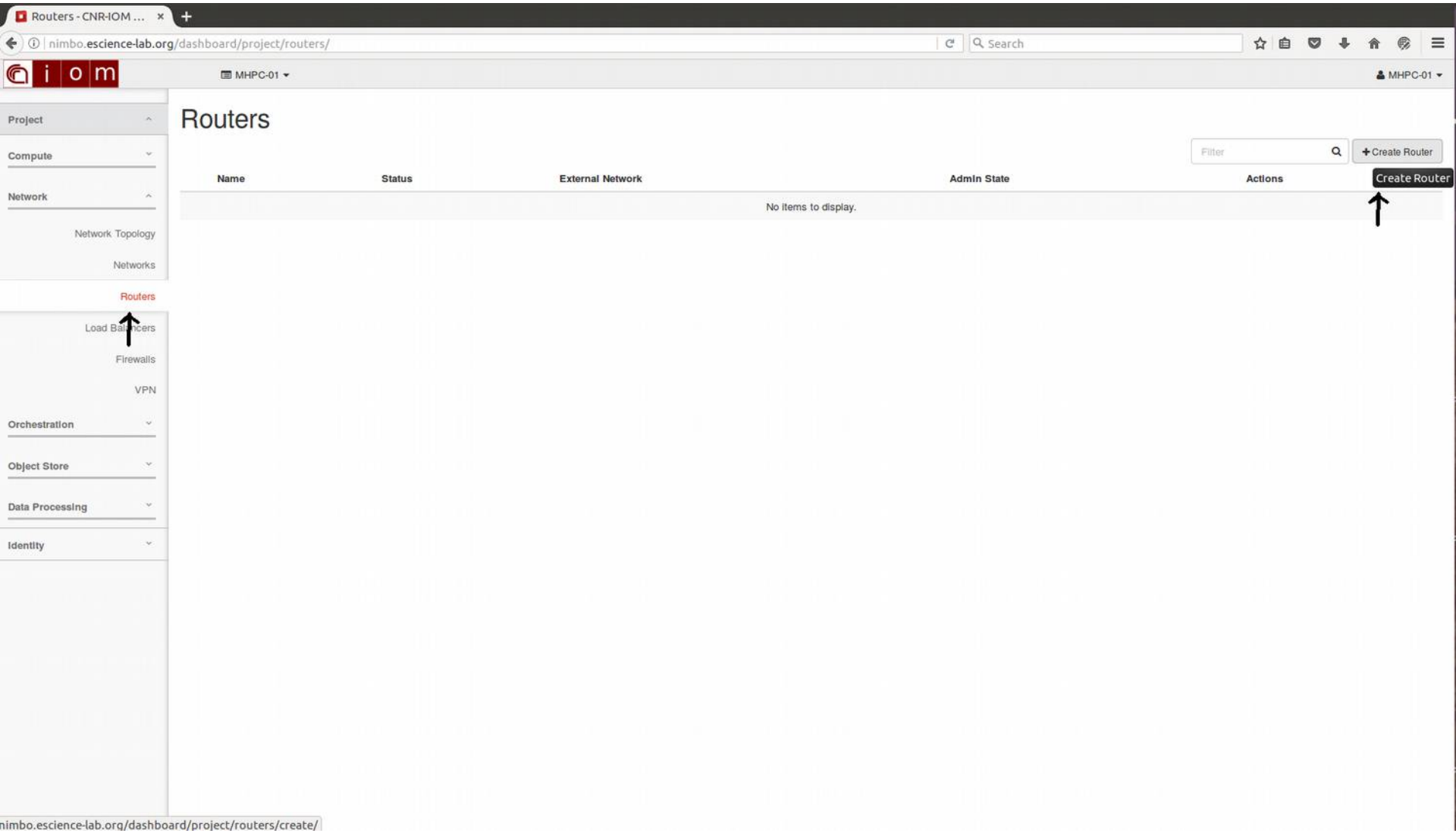
Create a private network: step3



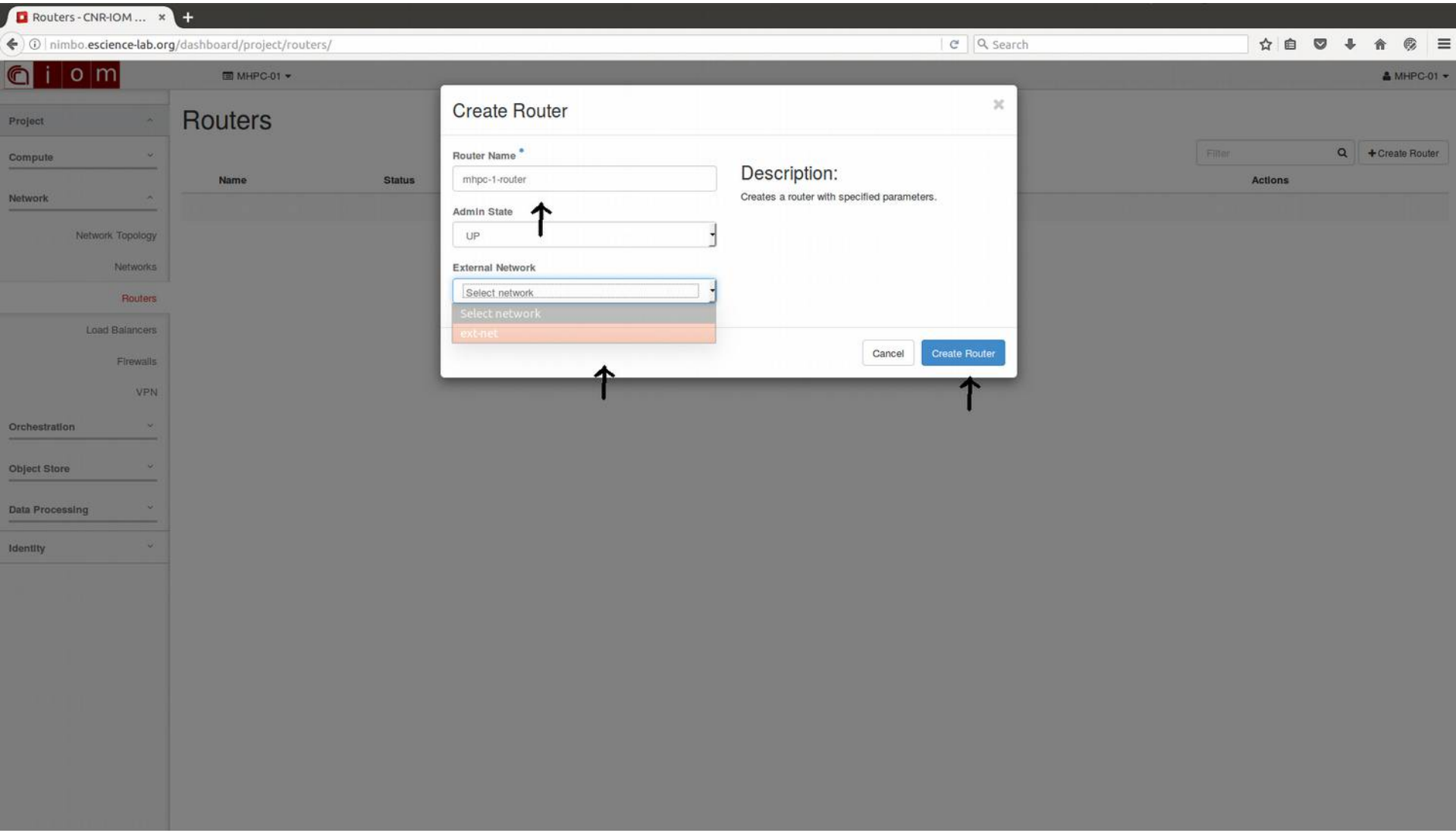
Create a private network: step4



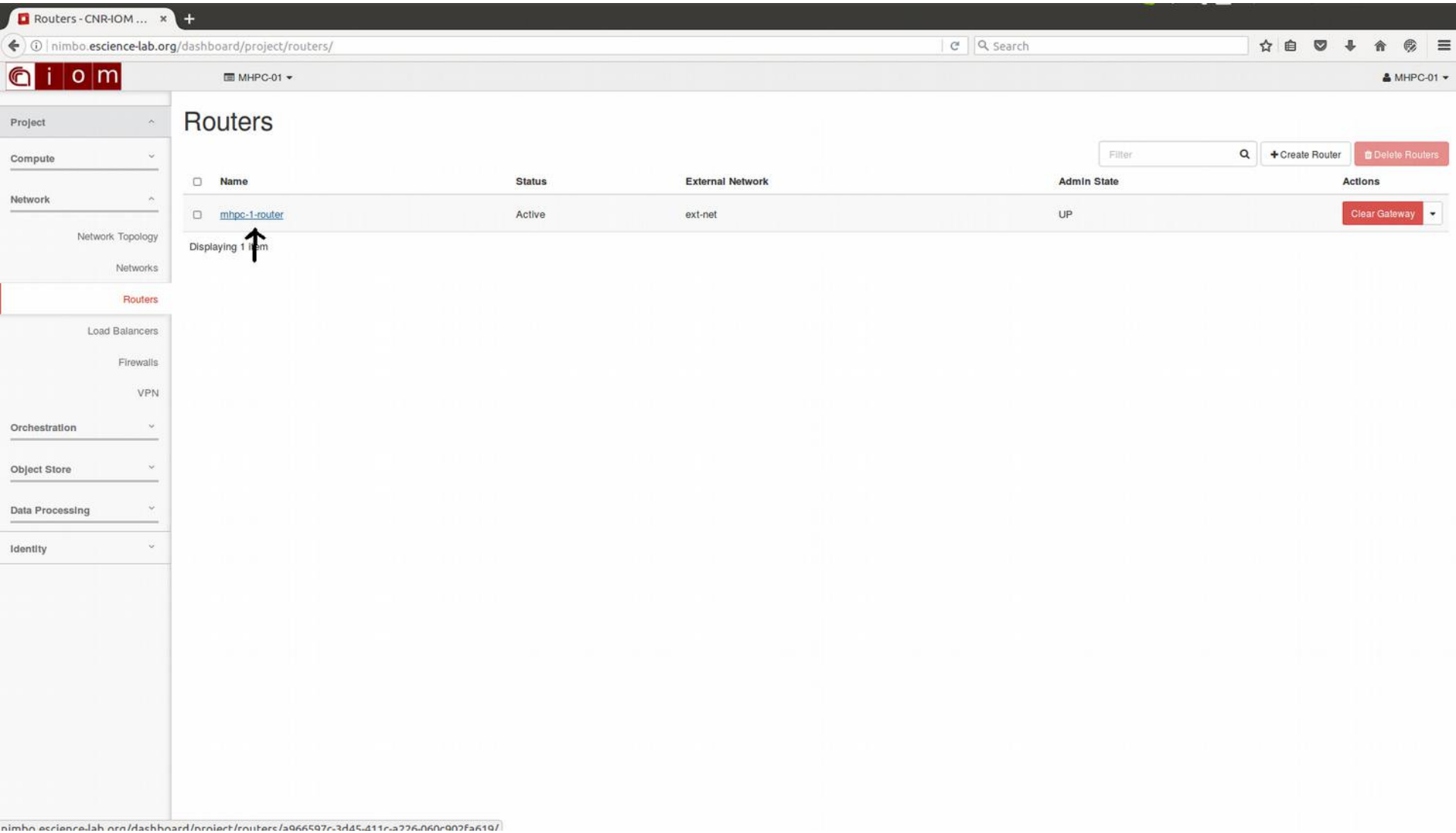
Create a private network: step5



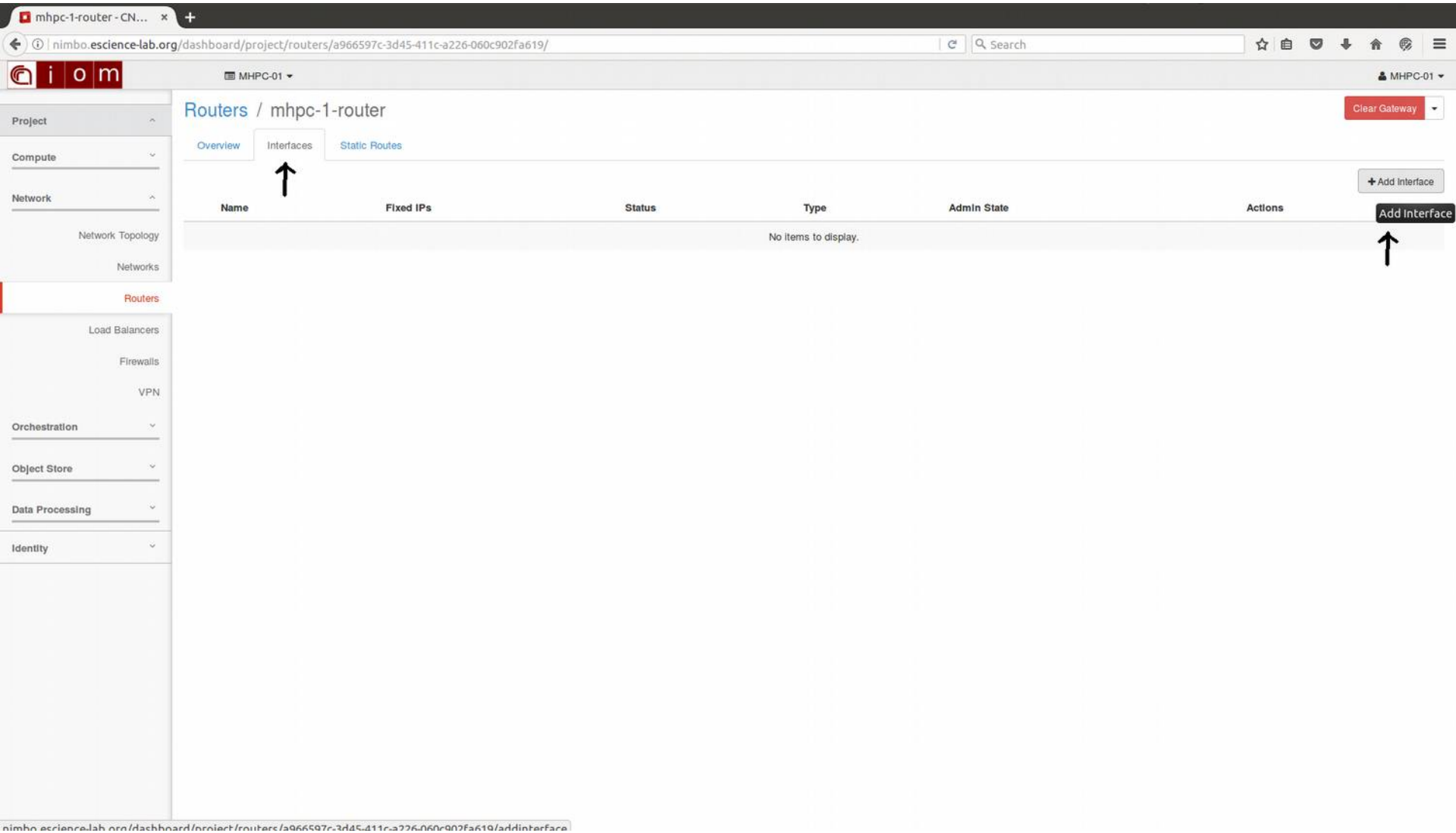
Create a private network: step6



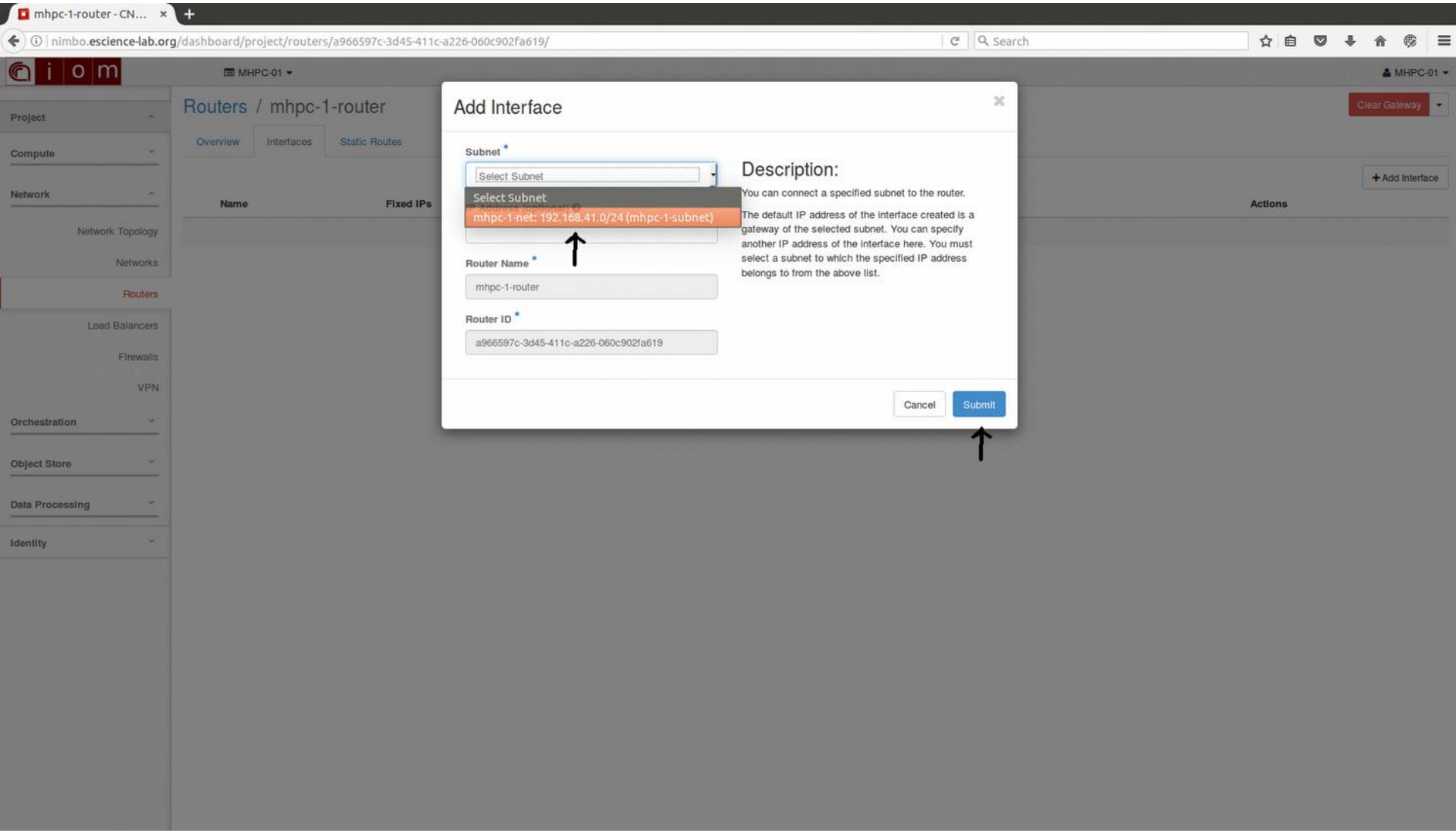
Create a private network: step7



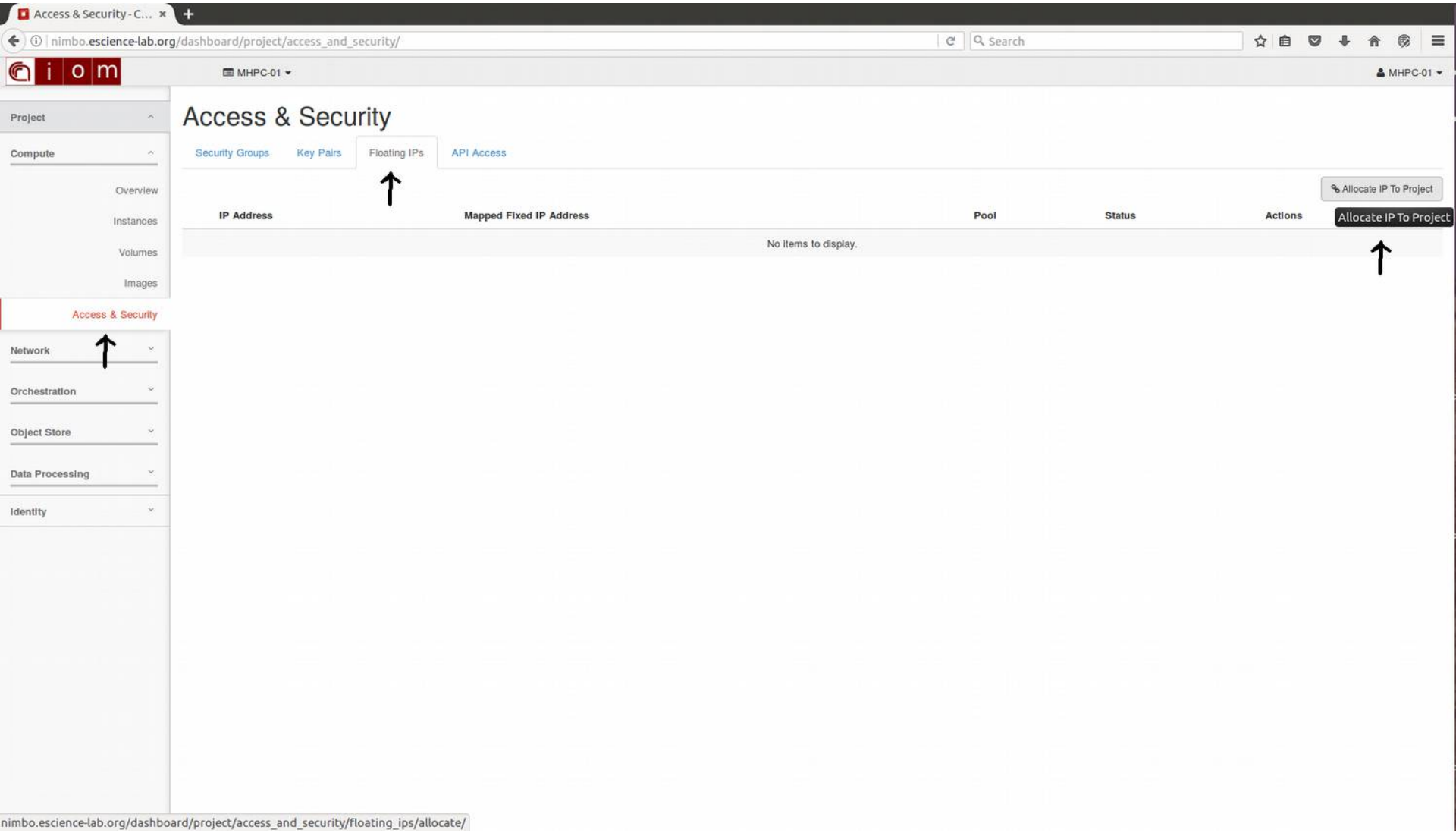
Create a private network: step8



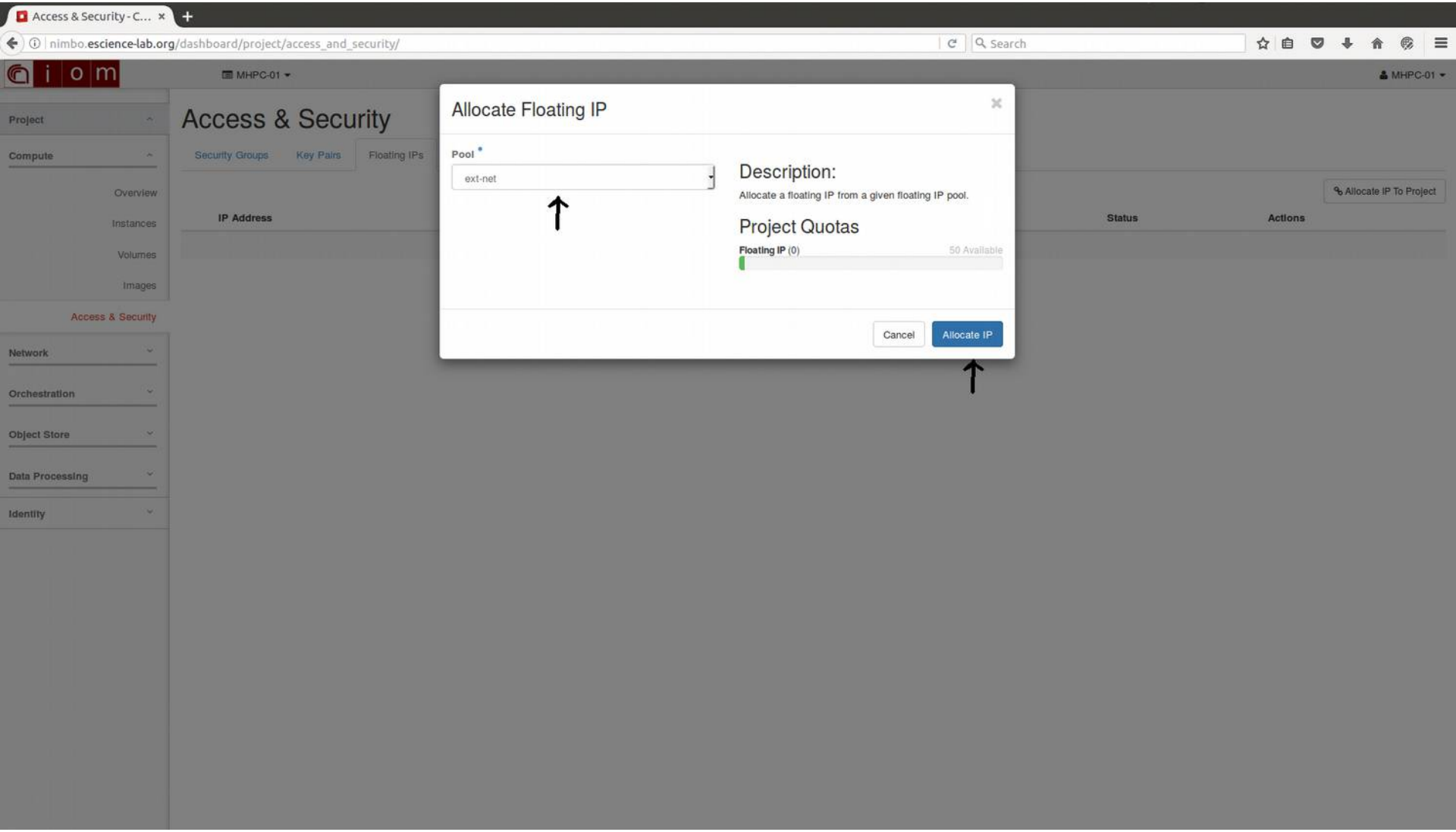
Create a private network: step9



Create a private network: step10



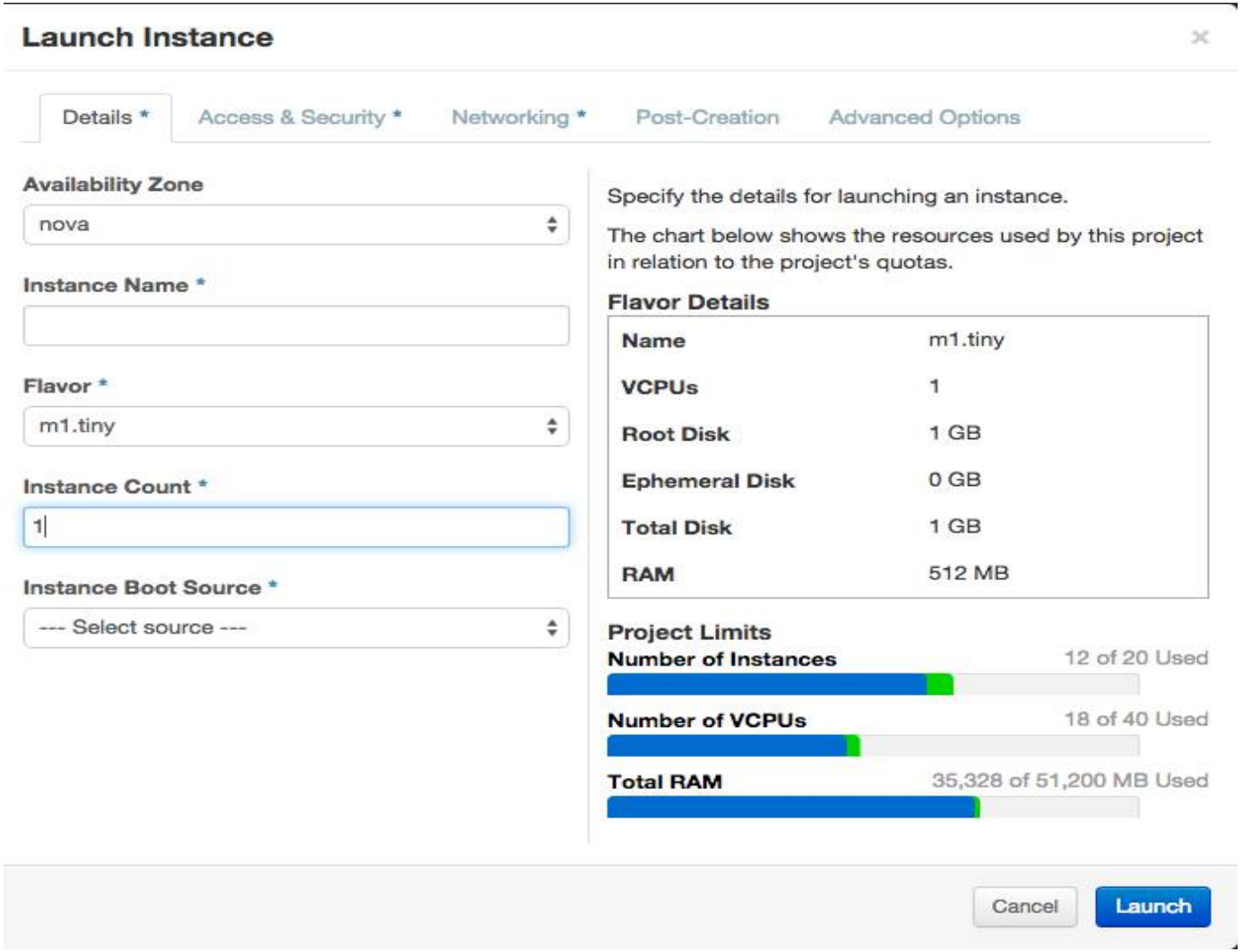
Create a private network: step11



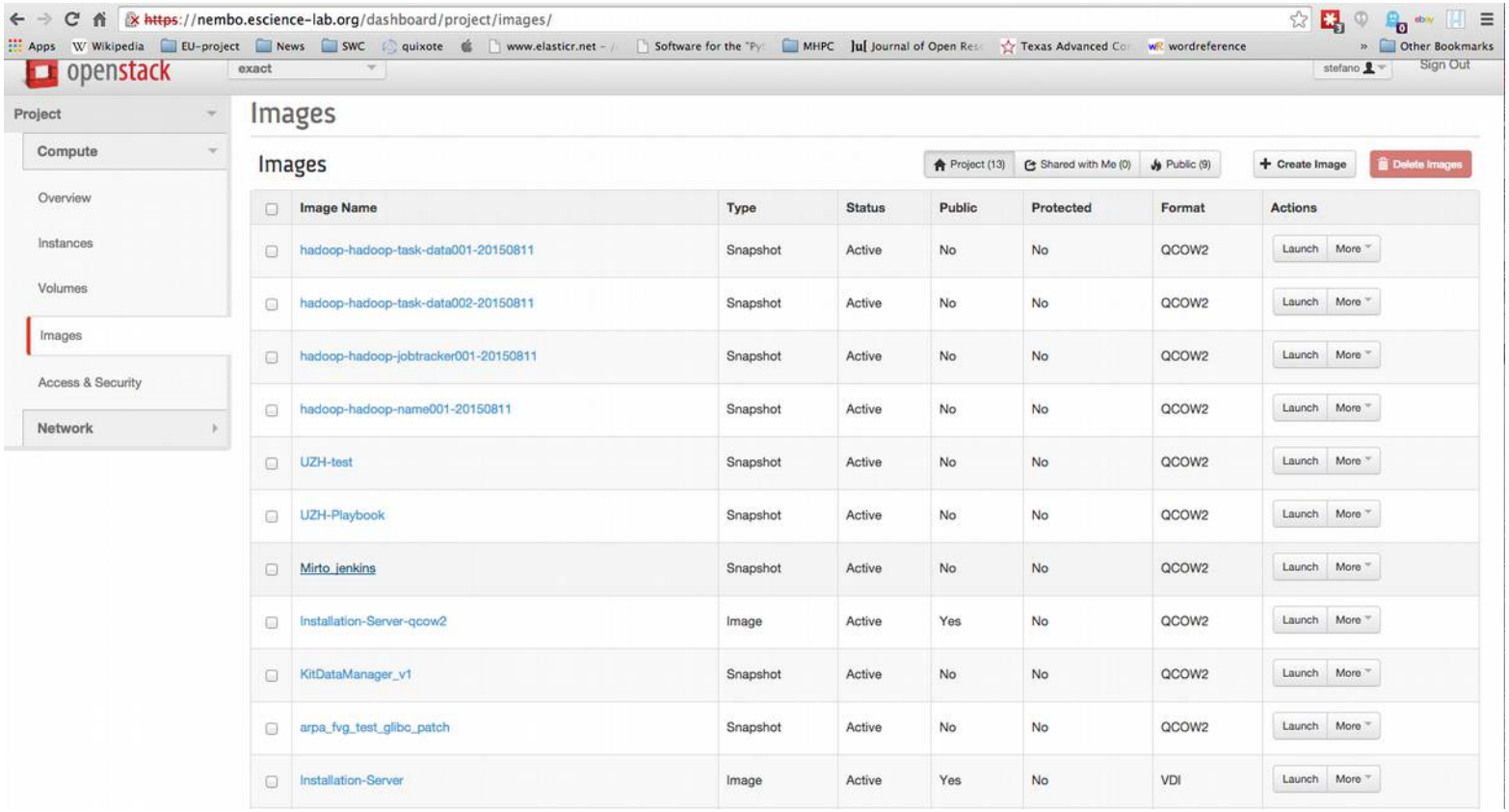
Create a VM

- Request Provisioning From UI
- A three steps action:
  - Login to Horizon
  - Specify parameters of VM
    - VM Name
    - Image (OS type)
    - Flavor (specifies CPU, Memory, Disk)
    - Network (required for Neutron)
    - Select/generate SSH keys for remote login
    - Optional (Persistent volumes, comments, etc.)
  - Select/Press "Launch" button

Launch panel



VM parameters: image (OS)



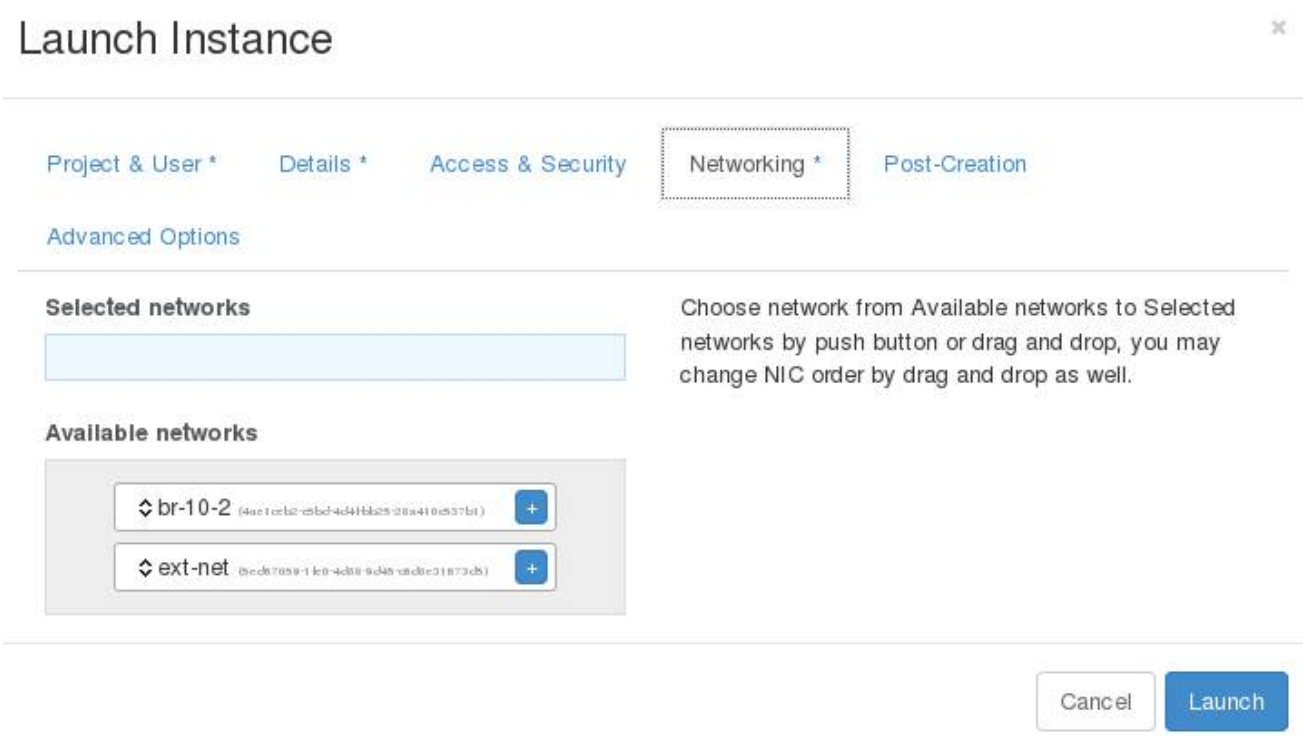
VM parameters: flavor

- Virtual hardware templates are called "flavors" in OpenStack, defining sizes for RAM, disk, number of cores, and so on.

```
$ nova flavor-list
+----+-----------+-----------+------+-----------+\+-------+-\+-------------+
| ID | Name      | Memory_MB | Disk | Ephemeral |/| VCPUs | /| extra_specs |
+----+-----------+-----------+------+-----------+\+-------+-\+-------------+
| 1  | m1.tiny   | 512       | 1    | 0         |/| 1     | /| {}          |
| 2  | m1.small  | 2048      | 10   | 20        |\| 1     | \| {}          |
| 3  | m1.medium | 4096      | 10   | 40        |/| 2     | /| {}          |
| 4  | m1.large  | 8192      | 10   | 80        |\| 4     | \| {}          |
| 5  | m1.xlarge | 16384     | 10   | 160       |/| 8     | /| {}          |
+----+-----------+-----------+------+-----------+\+-------+-\+-------------+
```

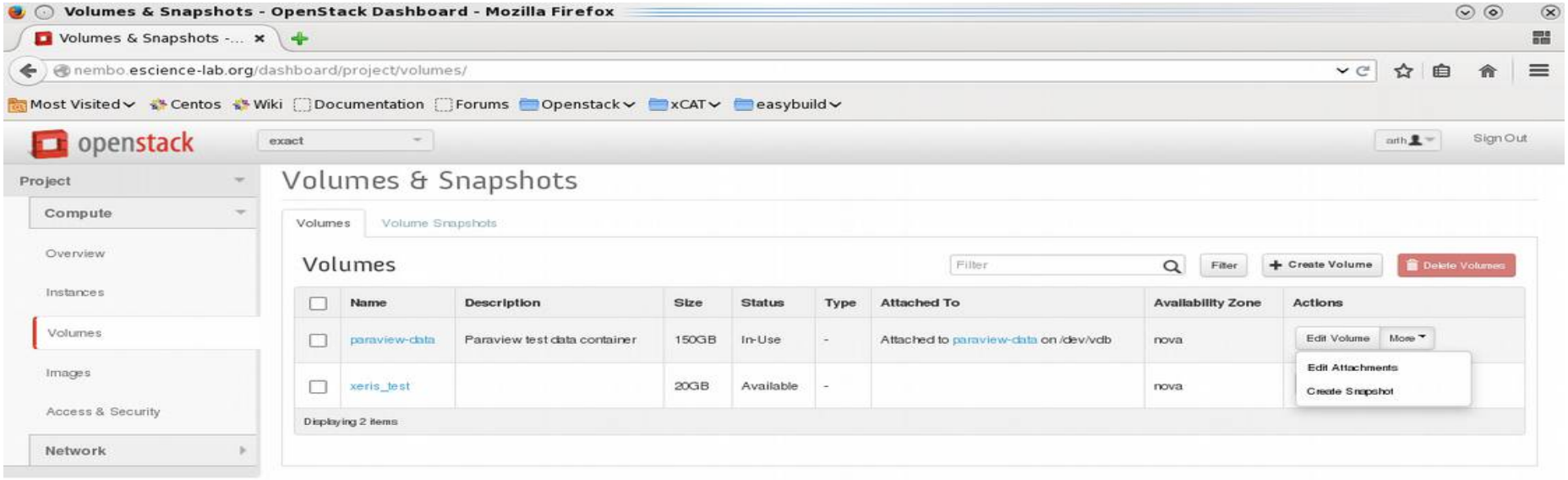
VM parameters: network

- A network should be defined..



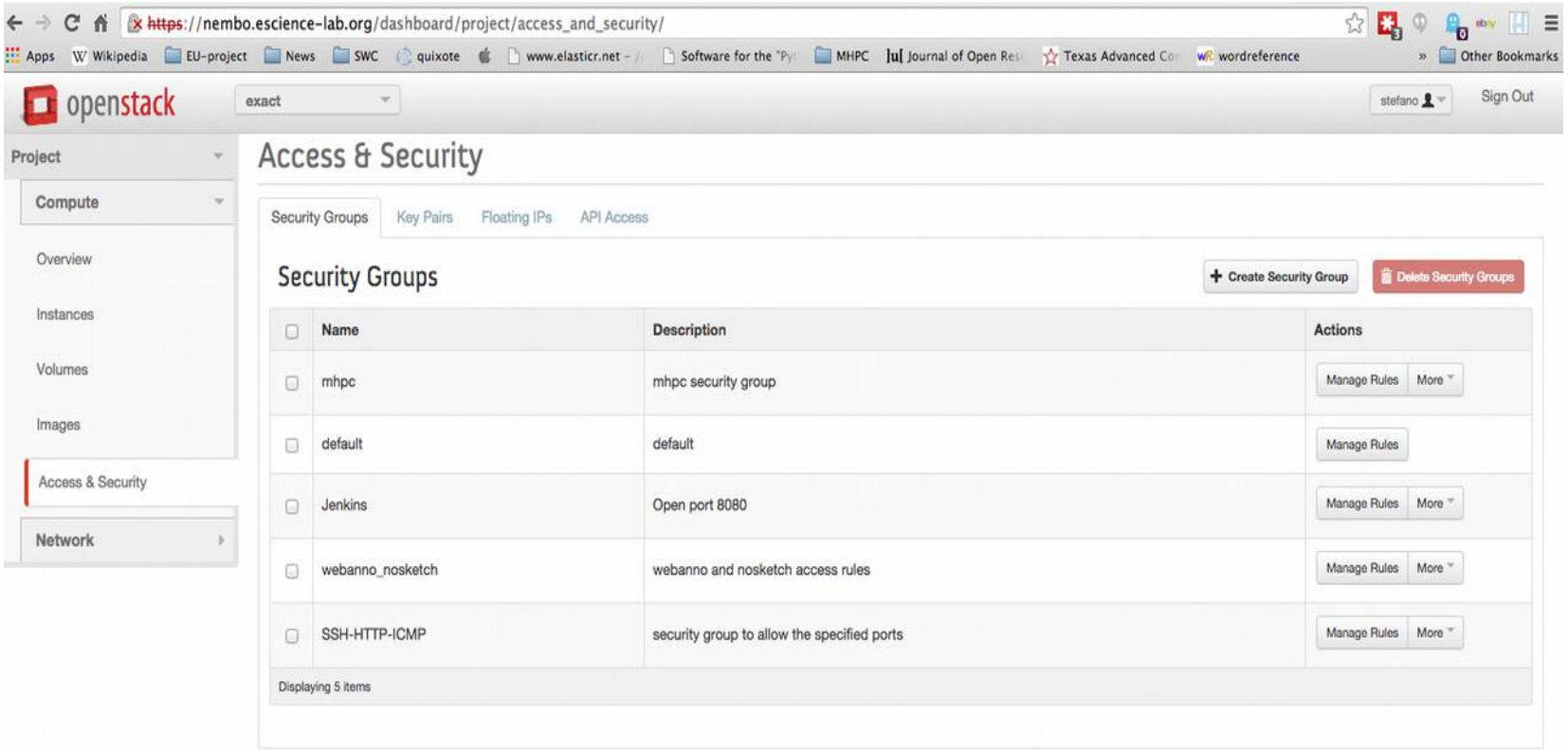
VM parameters: volumes

- Optionally Volumes can be attached



VM parameters: security

- Optionally some security settings can be defined..



VM life cycle

AFTER pressing “LAUNCH” :

1. Authentication is performed
2. nova-api is contacted and a new request is created:
3. nova-scheduler find an appropriate host
4. nova-compute reads the request and start an instance:
5. (if requested) nova-compute contacts cinder to provision the volume
6. neutron/nova-network configure the network
7. nova-compute starts the virtual machine
8. horizon/nova poll nova-api until the VM is ready.

Where/when to use OpenStack and why?

OpenStack possible use-cases?

- Service providers offering an IaaS compute platform or services higher up the stack (PaaS)
- IT departments acting as cloud service providers for business units and project teams
- Processing big data with tools like Hadoop
- Scaling compute up and down to meet demand for web resources and applications
- High-performance computing (HPC) environments processing diverse and intensive workloads

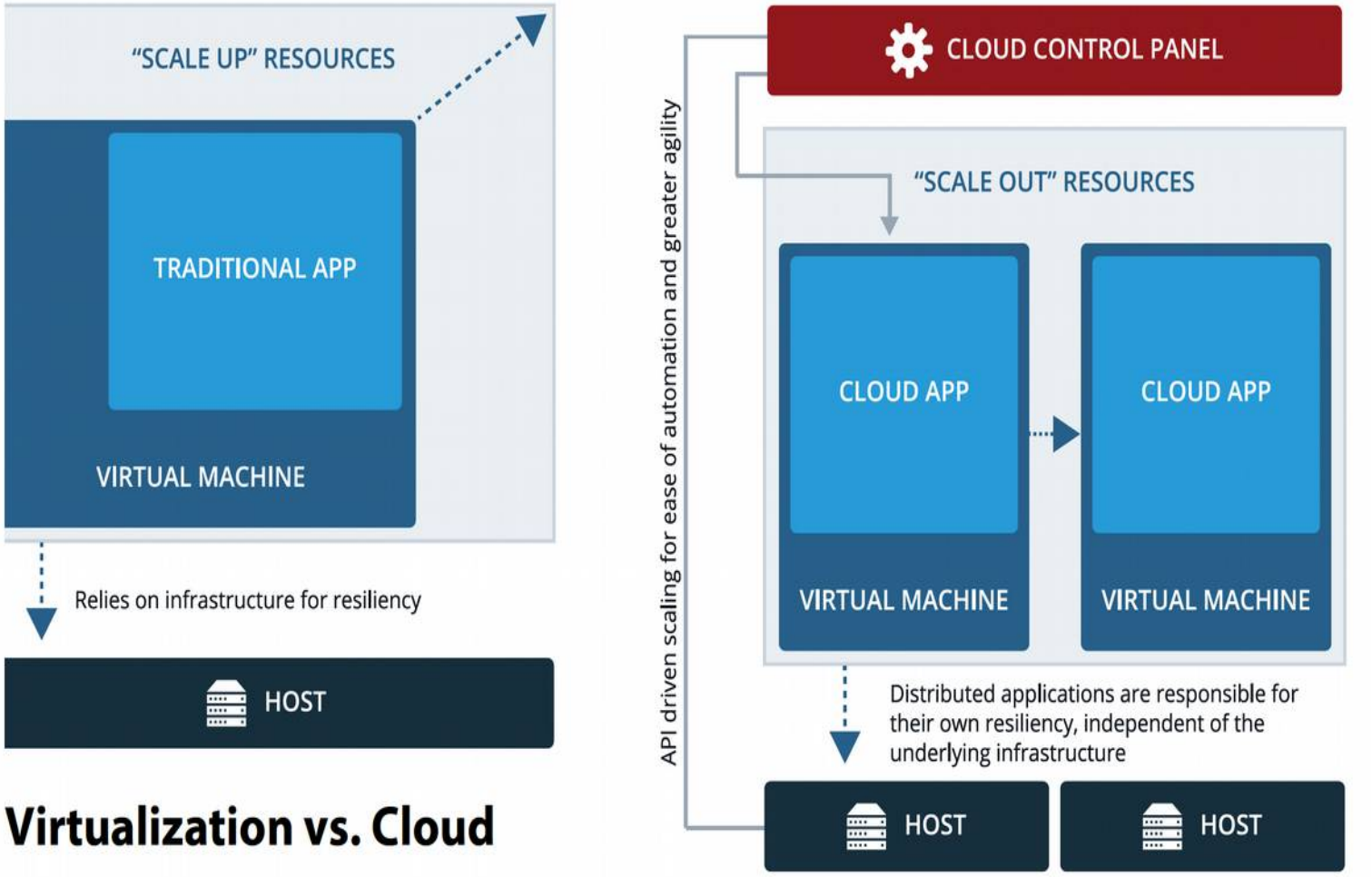
Cattle vs Pets

- pets are given names like
  - cicci.example.org
- you care about them
- they are unique, you check on them every day
- when they get ill, you nurse them back to health

- cattle are given names like
  - `vm-001.example.org`
- they are all the same
- when they get ill, you shoot them and get another one

Borrowed from @randybias at Cloudscaling http://www.slideshare.net/randybias/the--cloud-- revolution--cyber--press--forum--philippines

Scale up vs scale out



Which cloud?

- General purpose: Uses common components that address 80% of common use cases.
- Compute focused: For compute intensive workloads such as high performance computing (HPC).
- Storage focused: For storage intensive workloads such as data analytics with parallel file systems.
- Network focused: For high performance and reliable networking, such as a content delivery network (CDN).
- Multi-site: For applications that require multiple site deployments for geographical, reliability or data locality reasons.
- Hybrid cloud: Uses multiple disparate clouds connected either for failover, hybrid cloud bursting, or availability.
- Massively scalable: For cloud service providers or other large installations
- Specialized cases: Architectures that have not previously been covered in the defined use cases.

From http://docs.openstack.org/arch-design/arch-design.pdf

What expectation ?

According to a survey by ZDNet, respondents perceived benefits from the ability to deploy:

- applications faster (54 percent);
- reduced effort to deploy applications (40 percent);
- Streamlined development and testing (38 percent);
- reduced application deployment costs (31percent);
- server consolidation (25 percent).

OpenStack vs HPC/HTC clusters

- quota management vs. fairsharing
- software installation (user vs. admin)
- scalability vs. performance
- shared vs. exclusive resource
- continuous growth vs. fixed size

Cluster provisioning in openstack

- Do you need a cluster ?
- Buy it !
  - buy the machines
  - find a room
  - setup air conditioning and ensure you have enough power
  - hire a system administrator
- Run on someone else's cluster
  - it may not have all the software you need
  - need to negotiate policies
  - resource usage conflicts
- Make your own cluster in the Cloud
  - you choose the software and the configuration
  - You create it when you need it

What about performance ?

- Not so big concern
  - hypervisors can reduce around 5-10% the performance.
  - linpack and stream small reduction: ~5%.
  - Networking: bandwidth about 5%, for latency about 10% reduction.
- Numbers are above are perfectly acceptable if you reduce the provisioning time from months to minutes..

Tools to provide a cluster

- Many around
  - Starcluster
  - Virtualcluster etc..
- Our experience:
  - elasticluster (UZH)
    - https://github.com/gc3-uzh-ch/elasticluster/
  - Command line tool
    1. creates virtual machines in a cloud
    2. installs and configures the software you want
    3. add and remove nodes if needed

Elasticluster features

- Different kind of computational clusters:
  - Batch systems:
    - SLURM
    - OpenGridEngine
    - Torque+MAUI
  - Hadoop
  - Matlab Distributed Computing Servers
- Multiple distributed filesystems:
  - OrangeFS/PVFS
  - GlusterFS
  - Ceph
  - HDFS

Are you still concerned ?

- IRONIC service could be the solution
  - bare metal (physical server) provisioning in cloud;
- CONTAINERS is the other option

Ironic is promising..

- A few use-cases for bare metal (physical server) provisioning in cloud:
  - High-performance computing clusters
  - Computing tasks that require access to hardware devices which can’t be virtualized
  - Database hosting (some databases run poorly in a hypervisor)
  - Single tenant, dedicated hardware for performance, security, dependability and other regulatory requirements
  - Or, rapidly deploying a cloud infrastructure

What are containers ?

Containers are isolated, portable environments where you can run applications along with all the libraries and dependencies they need.

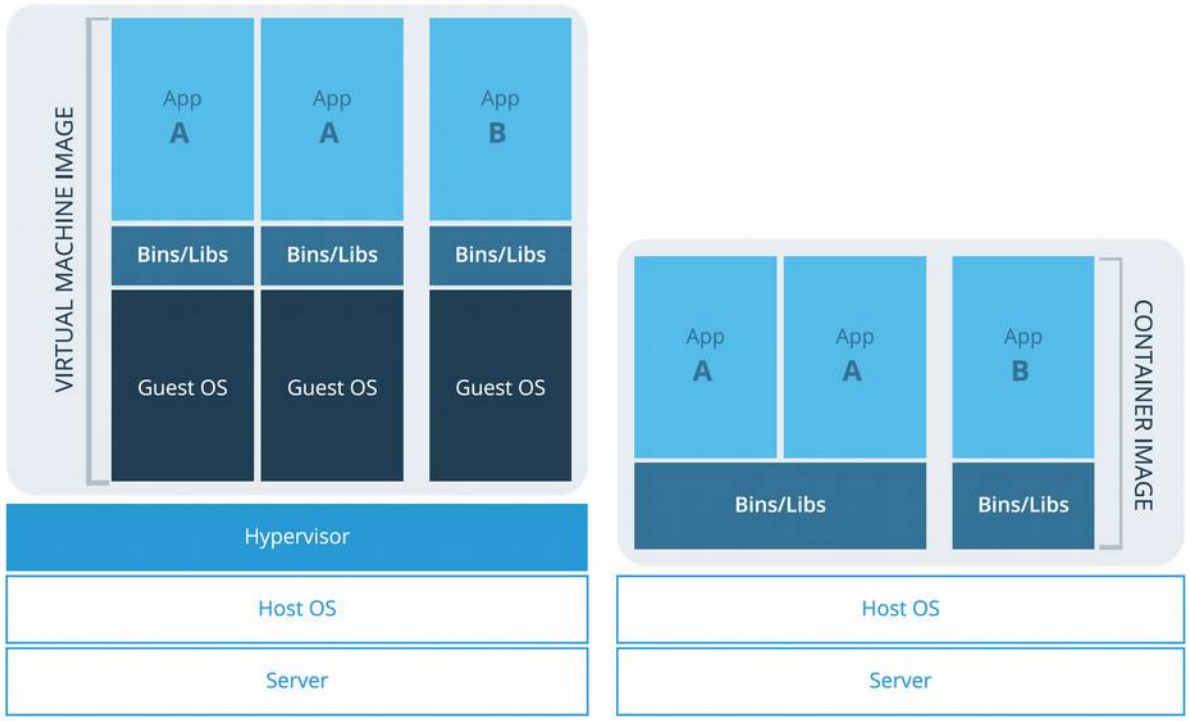


Figure 1: Containers vs. VMs

Containers&openstack

A Containers team active from May 2014

Integrate container technology in OpenStack, and providing new services and tools

The objective is to allow users to create and manage containers with an experience consistent with what they expect from using the Nova service to get virtual machines.

The aim is to offer developers a single set of compatible APIs to manage their workloads, whether those run on containers, virtual machines or bare metal.

Openstack project on containers

- Magnum is designed to offer container specific APIs for multi-tenant containers-as-a-service with OpenStack
- Kolla is designed to offer a dynamic OpenStack control plane where each OpenStack service runs in a Docker container.
- Murano is an application catalog solution that allows for packaged applications to be deployed on OpenStack
- With the Liberty release, Magnum and Murano ~~will~~ should be production-ready.

What about overhead for HPC

TABLE I. RESULTS FOR PXZ, LINPACK, STREAM, AND RANDOMACCESS. EACH DATA POINT IS THE ARITHMETIC MEAN OF TEN RUNS. DEPARTURE FROM NATIVE EXECUTION IS SHOW WITHIN PARENTHESES "()". THE STANDARD DEVIATION IS SHOWN WITHIN SQUARE BRACKETS "[]".

Workload		Native	Docker	KVM-untuned	KVM-tuned
PXZ (MB/s)		76.2 [±0.93]	73.5 (-4%) [±0.64]	59.2 (-22%) [±1.88]	62.2 (-18%) [±1.33]
Linpack (GFLOPS)		290.8 [±1.13]	290.9 (-0%) [±0.98]	241.3 (-17%) [±1.18]	284.2 (-2%) [±1.45]
RandomAccess (GUPS)		0.0126 [±0.00029]	0.0124 (-2%) [±0.00044]	0.0125 (-1%) [±0.00032]	Tuned run not warranted
Stream (GB/s)	Add	45.8 [±0.21]	45.6 (-0%) [±0.55]	45.0 (-2%) [±0.19]	
	Copy	41.3 [±0.06]	41.2 (-0%) [±0.08]	40.1 (-3%) [±0.21]	
	Scale	41.2 [±0.08]	41.2 (-0%) [±0.06]	40.0 (-3%) [±0.15]	
	Triad	45.6 [±0.12]	45.6 (-0%) [±0.49]	45.0 (-1%) [±0.20]	

Taken From "An Updated Performance Comparison of Virtual Machines and Linux Containers"

A few more observation

1. Cloud systems are generally about maximizing tenancy/utilization of resources

2. Not so much about individual application performance.

As services provider point 1 should be taken into account

Individual application performance is important but at a second stage

eXact-lab/CNR_IOM OpenStack infrastructures

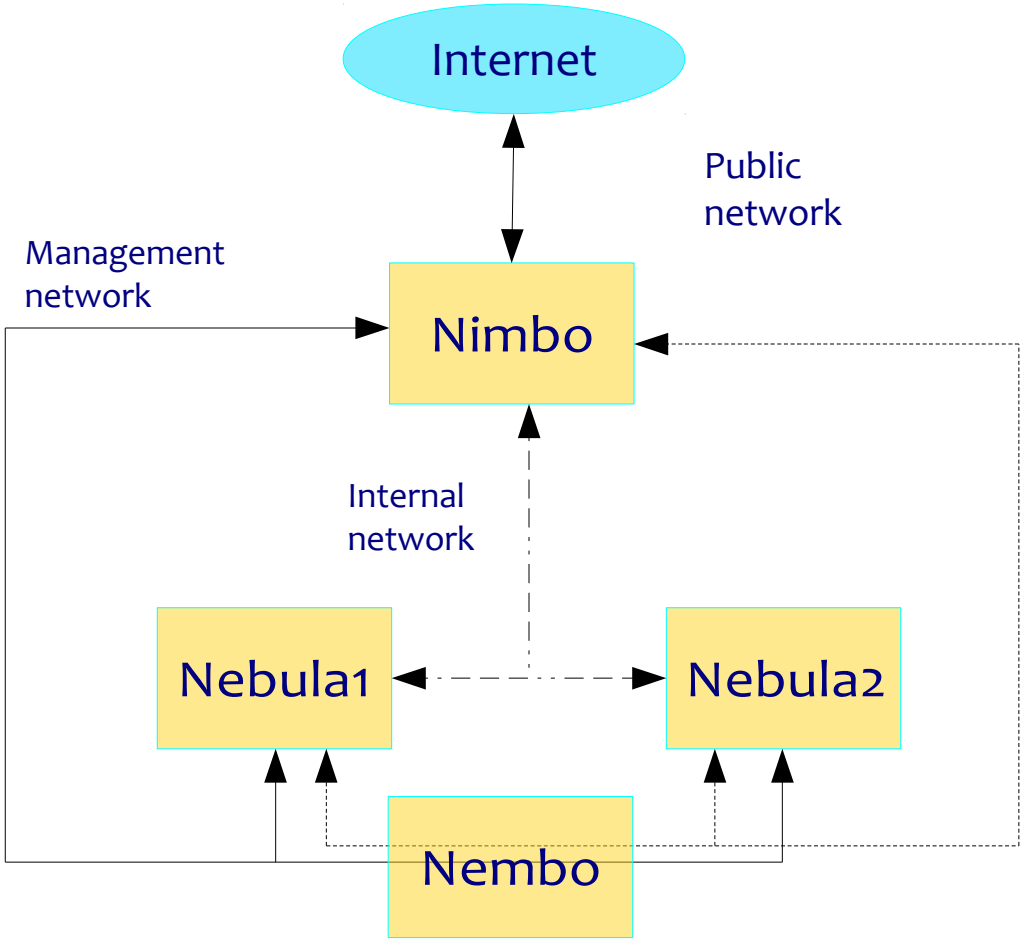
eXact lab cloud infrastructures

- eXact lab manages:
  - CNR/IOM cloud infrastructure
    - production
  - eXact lab testbeds
    - Experimental ones
  - COSILT cloud infrastructure
    - Production level

CNR/IOM infrastructure

The production environment is composed by

- One central node:
  - provides the central services (Keystone, Nova controller, Neutron, Glance, Cinder)
  - 2 computes nodes (nova-compute)
  - 1 storage node (glance+cinder)

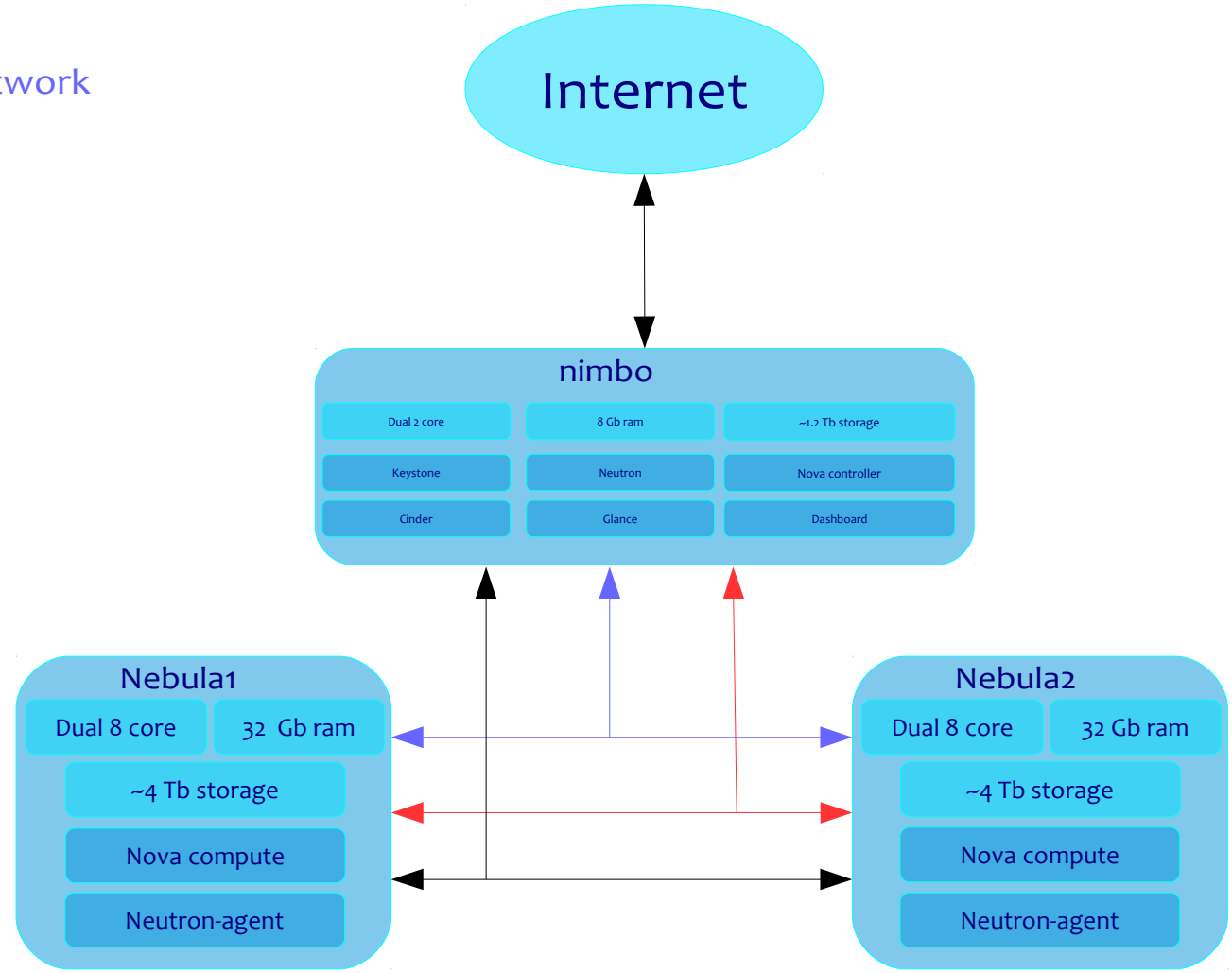


CNR/IOM infrastructure

Management network

Public network

Internal network



CNR/IOM infrastructure

3 different networks:

- the management network
  - physical network
- the internal network (high speed network for data transfer)
  - Physical infiniband network + internal virtual network
- OpenStack “public” network:
  - Virtual network managed by neutron

CNR/IOM infrastructure usage

- Server Consolidation
- Simple testing of virtual machine/services for CNR/IOM and eXact users
- Release: Mitaka
- High availability under development
- Integration on CNR/IOM HPC cluster at later stage..(thesis available..)

eXact lab cloud testbeds

Testbeds to:

- test the new releases of OpenStack
- plan the migration from one release to another.
- Experimenting/testing new services and/or specific customizations

- On relatively old HW nodes hosted at CNR/IOM premises.

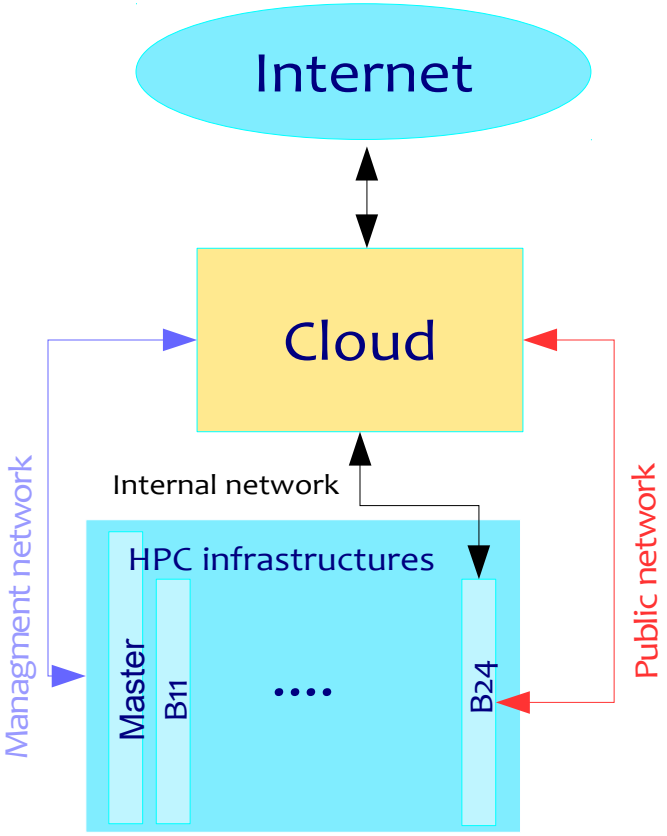
C3E cloud infrastructure

- COSilt setup an HPC infrastructure to provide HPC services to SMEs..
- Such approach requires that HPC resources should be easily used/provided on demand
- eXact-lab is providing its competence to COSINT to help the consortium to identify
- Openstack is the tool we are using..

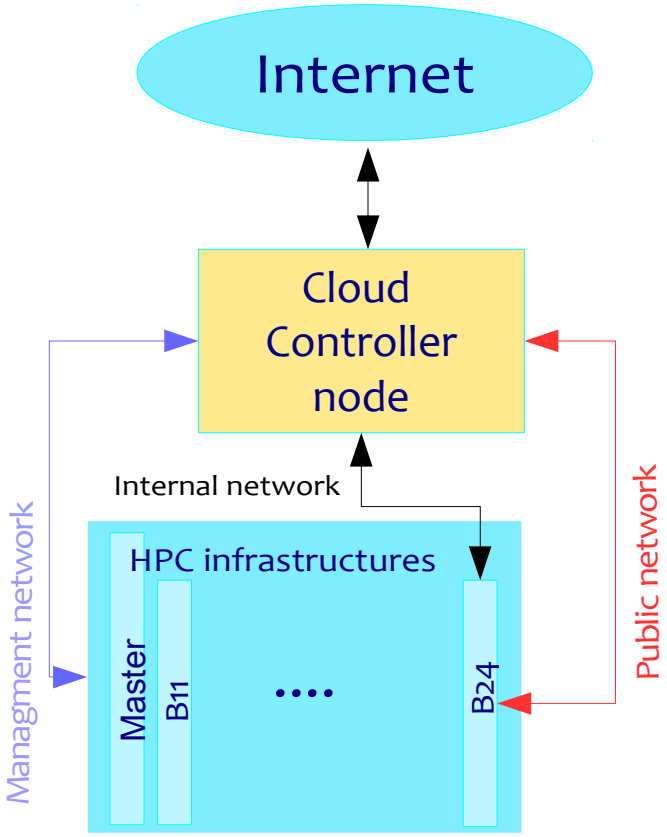
cosint
CONSORZIO PER LO SVILUPPO INDUSTRIALE DI TOLMEZZO

Cosint Cloud Infrastructure

Cloud: Dual Intel
40 core
@2.3GHz
32 Gb ram
2 Tb storage
B24: Dual Intel Xeon
E5 -2697/v2
24 core
@2.7GHz
64 Gb ram
1 Tb storage



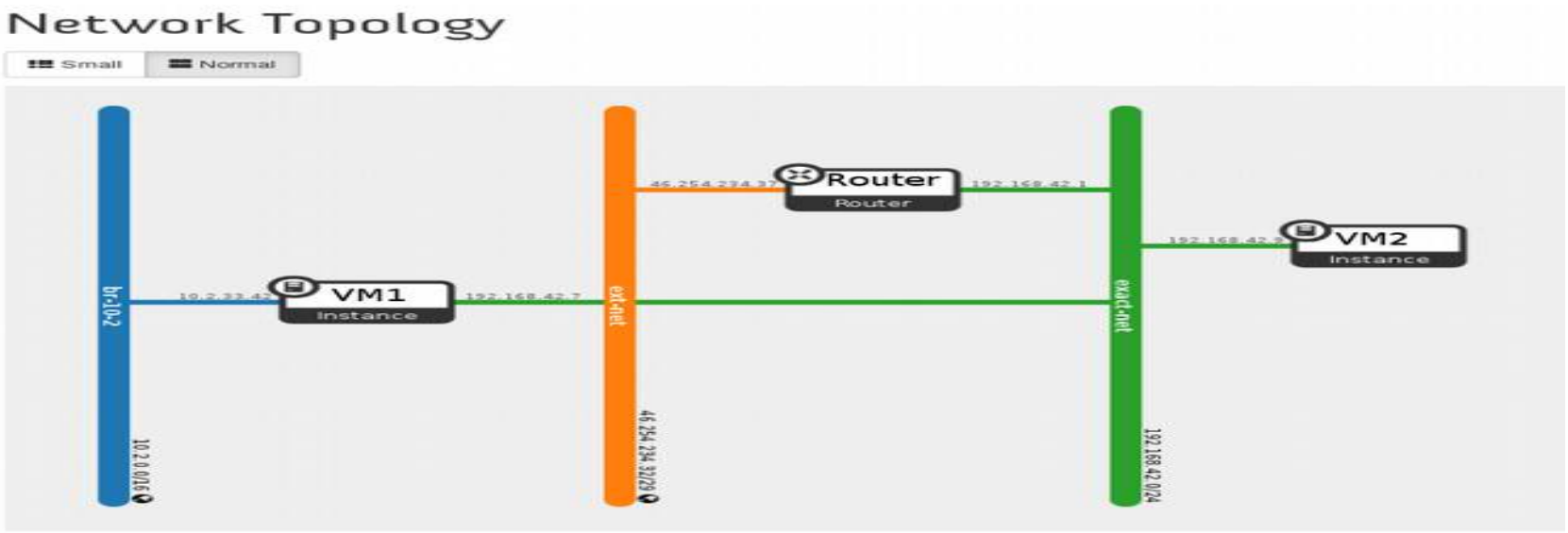
About the network



- The internal network is where the virtual network are spawned.
- A virtual network belongs to a tenant and can be shared.
- It is possible to assign a floating IP to a VM spawned on the virtual network
- The security of the VM is managed by the security groups (it is still possible to have a firewall on the VM, but only the connections allowed by the security groups will arrive to the VM)

Network topology example (COSINT):

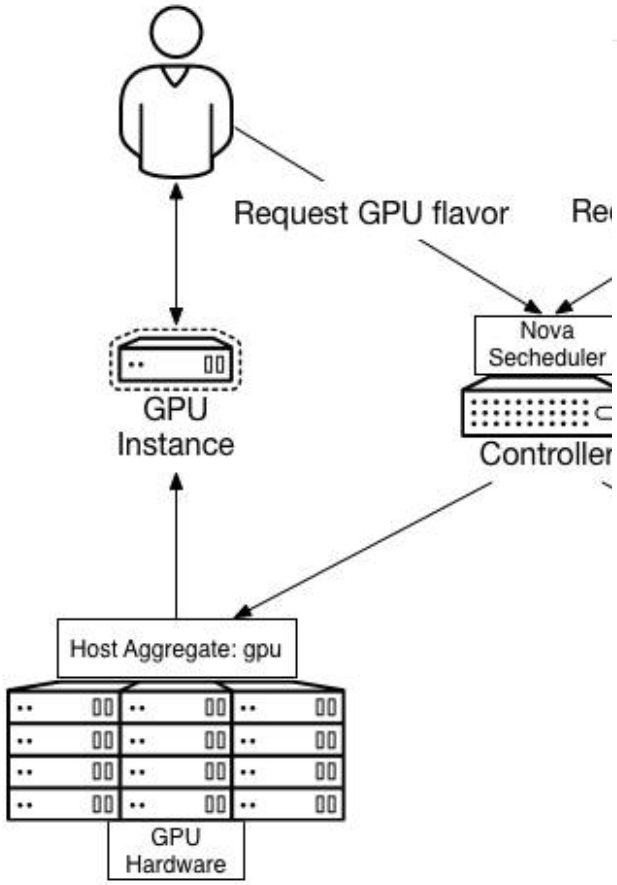
- br-10-2 and ext-net are external network, shared between all the tenants.
- br-10-2 is configured without a router and the VMs connected to this network have only (private) fixed IP
- on ext-net floating IPs are assigned to the Vms (this allow the Vms to be reached from internet).
- The exact-net is the tenant network and is connected to the ext-net via a router (exact-router)

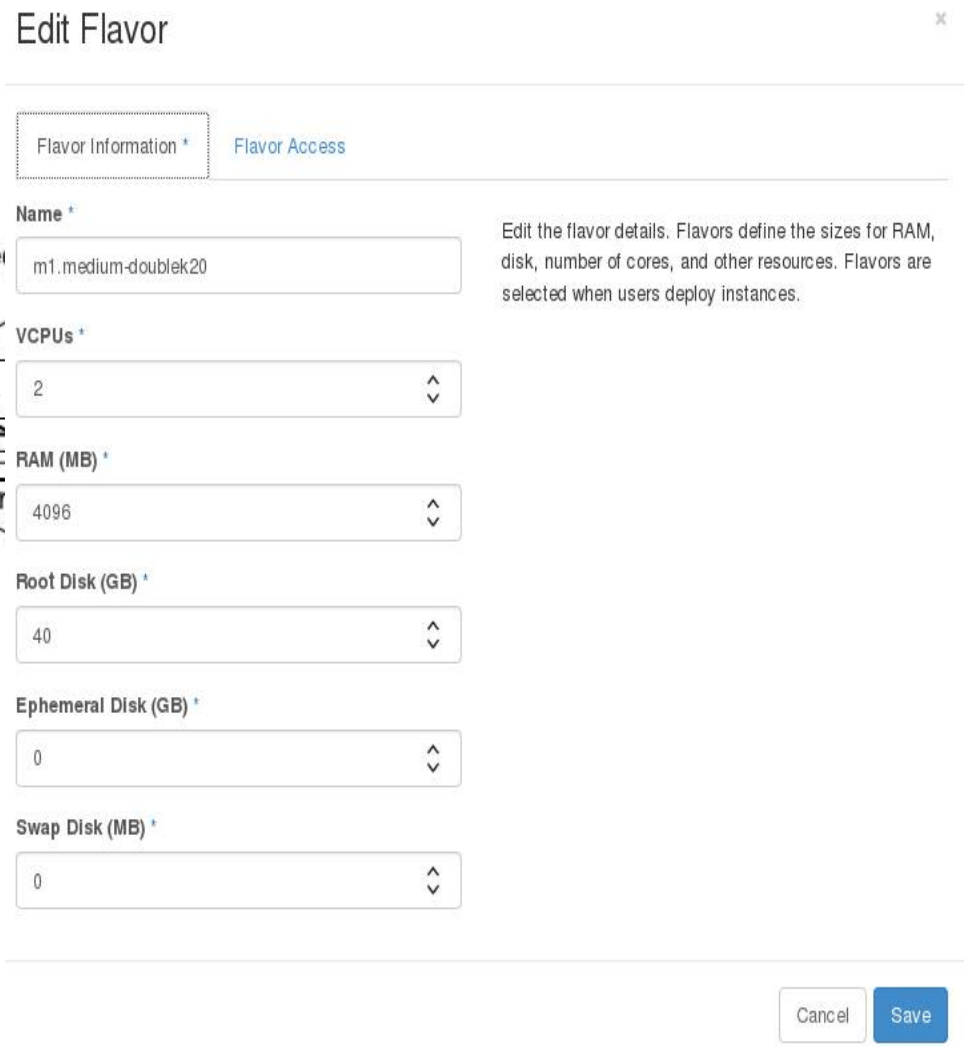


eXact lab testing on COSINT

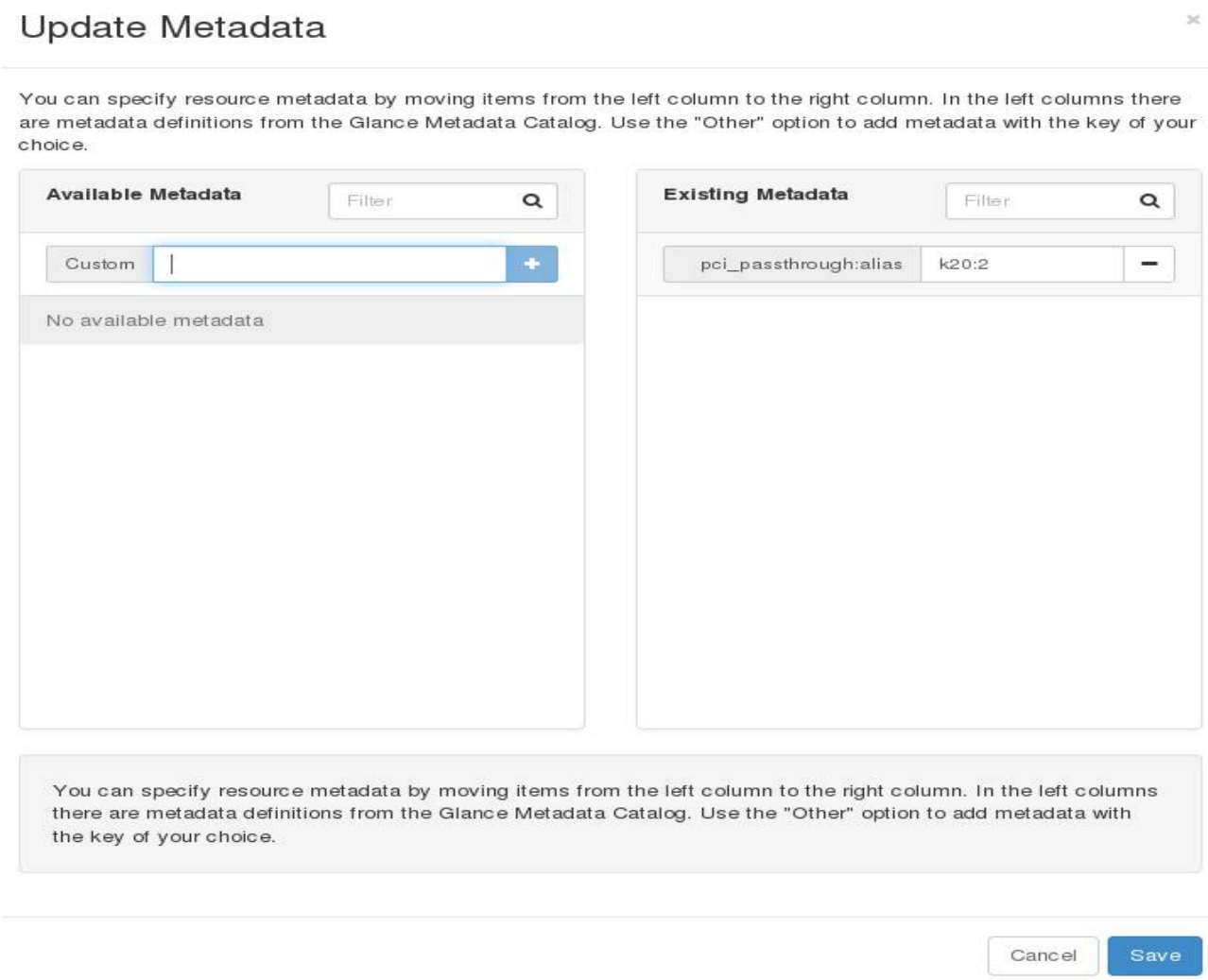
- Provide VM with GPGPU facilities
  - a VM can have direct access to the GPU on the compute node and so provide hardware acceleration
- Done through enabling at nova level “PCI passthrough” mechanism.
- Not yet officially supported by mainstream openstack distribution

Get the GPU flavor





GPU flavor



Conclusions

- OpenStack complex software not easy to master rapidly
- Complex to use, learning curve is steep
- Stability/Usability is still an issue sometime
- Many interesting projects/components that promise great applications and smart usage for HPC/scientific community