

# OPENSTACK

(Hazırlayan: @HalilGÖKSEL)

## İçerik

1. Openstack Nedir.....
2. Openstack Kurulum Gereksinimi.....
3. Openstack Kurulumu.....
4. Openstack Horizon Menüleri.....
6. Openstack API Servisleri.....
  - 6.1 Cinder.....
  - 6.2 Swift.....
  - 6.3 Nova.....
  - 6.4 Flavor.....
  - 6.5 Zun.....
  - 6.6 Neutron.....
  - 6.7 Designate.....
  - 6.8 Keystone.....
  - 6.9 Glance.....
7. Horizon Yönetimi.....
  - 7.1 Proje oluşturma.....
  - 7.2 Instance oluşturma.....
  - 7.3 Image Oluşturma.....
  - 7.4 Flavor Oluşturma.....
  - 7.5 Instance Connectivity.....
  - 7.6 Instance oluşturma 1.2.....
8. Openstack Ölçeklendirme.....
9. Openstack Genişletme.....
10. Openstack Logs.....
11. Openstack CLI.....

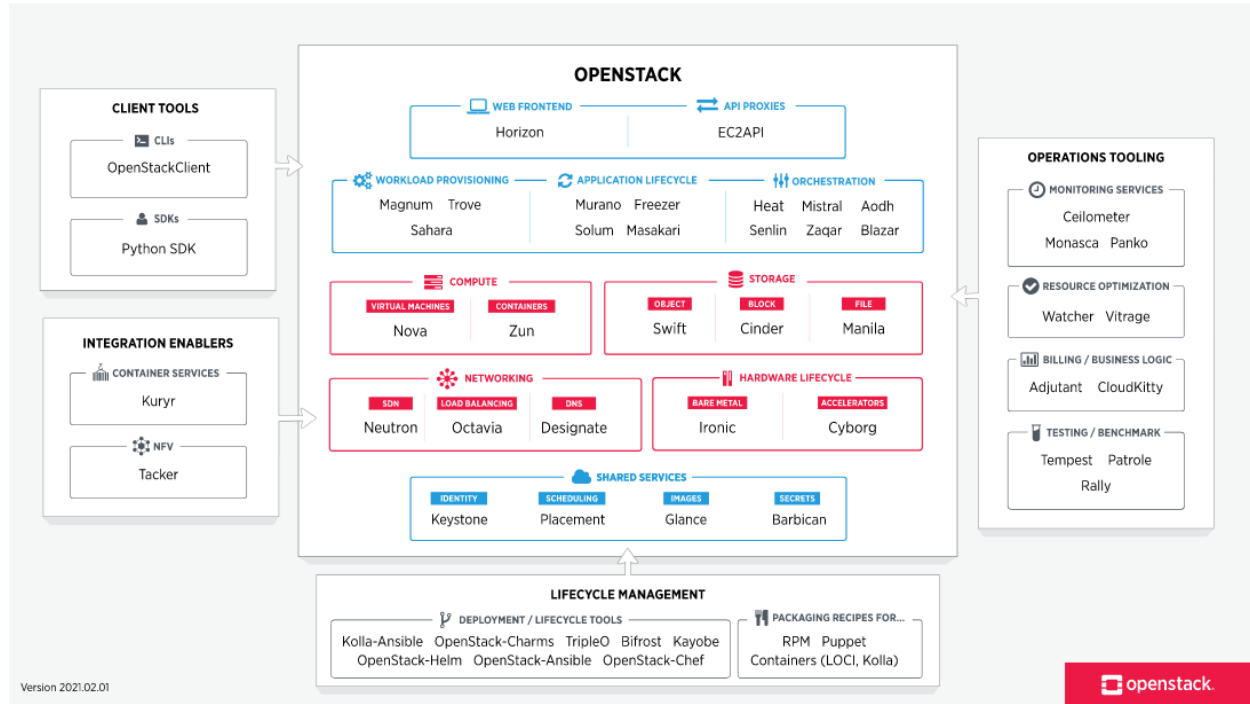
## Openstack nedir

OpenStack, ilk olarak 2010 yılında Amerikan Ulusal Uzay Ajansı (NASA) tarafından sağlanan hesaplama bileşeni Nova ve bulut servis sağlayıcısı Rackspace tarafından sağlanan depolama bileşeni Swift'in birleşmesi ile ortaya çıkan ve hızla gelişerek açık kaynak kodlu bulut çözümlerinde liderliğe yükselen bir bulut yönetim platformudur.

Cisco,IBM,HP,Dell vs.gibi dünya çapında büyük sağlayıcılar OpenStack kullanmakta ve bazıları OpenStack üzerine bileşenler geliştirerek kendi ürünlerini ortaya koyabilmektedirler.

Ülkemizde de daha çok devlet kurumları(TÜBİTAK-ULAKBİM tarafından)ve bazı özel kuruluşların kullanıldığı yerlere aşağıdakileri örnek verebiliriz: Fatih Projesi-T.C.Milli Eğitim Bakanlığı TÜİK Projesi-Türkiye İstatistik Kurumu Bazı Üniversiteler(Sakarya Üni. ,Süleyman Demirel Üni. ,İstanbul Üni. vs) Türksat İç projeleri Turkcell(Nesne depolama çözümü Swift) (Kaynak: openstackturkiye.com ,AB 2018 Karabük ULAKBİM sunumu)

OpenStack'ın 9 temel bileşeni (API) OpenStack topluluğu tarafından temel 9 bileşen olarak tanımlanan ve herhangi bir OpenStack sürümüyle beraber gelen 9 adet bileşen vardır.



## Openstack Kurulum Gereksinimi

Openstack, 2013 yılında RedHat tarafından başlatılan bir openstack dağıtımıdır. RDO Openstack, RedHat'ın RPM Dağıtımı olarak da bilinir. RDO Openstack, Red Hat Enterprise Linux(RHEL) ve CentOS, Fedora gibi diğer linux dağıtımları için başlatıldı.

### Packstack ve Devstack farkı

Packstack, çoğunlukla CentOS ve Fedora gibi Red Hat Distribution Linux için uygundur. Temelde Openstack Bileşenlerinin çeşitli kısımlarını ssh aracılığıyla dağıtmak için kukla modülleri kullanır. b) Devstack, Openstack'i dizüstü bilgisayarda da kurmak için kullanılabilen, Openstack minimal kurulumu ile bir ortam oluşturmak için yazılmış bir komut dosyasıdır

## Requirements

- One Host Machine
  - One physical or virtual host machine running CentOS, RedHat or Fedora
  - Clean OS installation highly recommended
  - This is the machine on which you will install and run all the OpenStack cloud components. The machine must meet the following hardware requirements.
- Minimum of 16 GB of RAM.
- Minimum of 20 GB disk space is recommended.
- Add more memory or disk space if you plan on adding additional instances or volumes after this deployment
- 64-bit x86 processor with support for the Intel 64 or AMD64 CPU extensions, and the AMD-V or Intel VT hardware virtualization extensions enabled.
- Network Access
  1. Fiziksel yada Sanal Centos, Redhat, Fedora işletim sistemi olmalı.
  2. Kaynak değerleri minimum 16 memory, 20 GB Disk alanı olmalı.
  3. 65-bit işlemci tiği olmalıdır ve CPU'da sanallaştırma açık olmalıdır.

## Openstack Kurulumu

İlk olarak sıfırdan sanallaştırma üzerine bir Centos7.6 VM kurulumu yapıyoruz. Sanal VM'imimizin internete açık olacak şekilde switch yapılandırıyoruz. 16 GB memory ve 20 GB tek disk verecek şekilde yapılandırıyoruz.

Kurulum sırasında KDUMP ve Security polisy devre dışı bırakıyoruz, Hostname ve network konfigre ediyoruz.

### 1. Adım

# /etc / environment dosyanızı aşağıdaki yerel ayarlarla doldurun

`vi /etc/environment`

`LANG=en_US.utf-8`

`LC_ALL=en_US.utf-8`

```
[root@openstack ~]# vi /etc/environment
[root@openstack ~]# cat /etc/environment
LANG=en_US.utf-8
LC_ALL=en_US.utf-8
[root@openstack ~]#
```

#vi editor'a aşına değilseniz; bir dosyayı düzenlemeye başlamak için "i" tuşuna basabilirsiniz. Basın düzenlemeyi tamamladığınızda "esc" ve ardından dosyayı kaydetmek ve vi düzenleyicisi'nden çıkmak için ":wq".

### 2. Adım

# firewalld hizmetinin durumunu kontrol edin. Etkinleştirirseniz durdurun ve devre dışı bırakın

`systemctl status firewalld`

`systemctl stop firewalld`

`systemctl disable firewalld`

```
[root@openstack ~]# systemctl status firewalld
• firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset: enabled)
   Active: inactive (dead)
     Docs: man:firewalld(1)
[root@openstack ~]#
```

Openstack NetworkManager'i desteklemediği için ve SELinux'u desteklemediği için bunları disable etmemiz gerekiyor.

### 3. Adım

# NetworkManager hizmetinin durumunu kontrol edin. Etkinleştirirseniz durdurun ve devre dışı bırakın

```
systemctl status NetworkManager
```

```
systemctl stop NetworkManager
```

```
systemctl disable NetworkManager
```

```
NetworkManager.service - Network Manager
Loaded: loaded (/usr/lib/systemd/system/NetworkManager.service; disabled; vendor preset: enabled)
Active: inactive (dead) since Wed 2022-08-31 12:06:49 +03; 10s ago
Docs: man:NetworkManager(8)
Main PID: 764 (code=exited, status=0/SUCCESS)
```

### 4. Adım

#ağ hizmetini etkinleştirme ve başlatma

```
systemctl enable network
```

```
systemctl start network
```

```
[root@openstack ~]# systemctl status network
● network.service - LSB: Bring up/down networking
   Loaded: loaded (/etc/rc.d/init.d/network; bad; vendor preset: disabled)
   Active: active (exited) since Wed 2022-08-31 12:08:17 +03; 8min ago
     Docs: man:systemd-sysv-generator(8)
   Process: 1972 ExecStop=/etc/rc.d/init.d/network stop (code=exited, status=0/SUCCESS)
   Process: 2199 ExecStart=/etc/rc.d/init.d/network start (code=exited, status=0/SUCCESS)
```

### 5. Adım

"enp0s3" ü arayüz adınızla değiştirin ve mevcut ayarlarını kontrol edin, IP, DNS, Gateway static olarak yapılandırılmaları yapın

```
cat /etc/sysconfig/network-scripts/ifcfg-enp0s
```

### 6. Adım

selinux'u yapılandırma dosyasından devre dışı bırak /etc/selinux/config

```
vi /etc/selinux/config
```

```
SELINUX=disabled
```

```

This file controls the state of SELinux on the system.
SELINUX= can take one of these three values:
    enforcing - SELinux security policy is enforced.
    permissive - SELinux prints warnings instead of enforcing.
    disabled - No SELinux policy is loaded.
SELINUX=disable
SELINUXTYPE= can take one of three values:
    targeted - Targeted processes are protected,
    minimum - Modification of targeted policy. Only selected processes are protected.
    mls - Multi Level Security protection.
SELINUXTYPE=targeted

```

## 7. Adım

Sistemi yeniden başlatmak gerekebilir. # yeniden başlatıldıktan sonra selinux'un durumunu kontrol edin, devre dışı bırakılmalıdır

### Reboot

```

[root@openstack ~]# uptime
12:19:29 up 1:01, 3 users, load average: 0.00, 0.01, 0.02
[root@openstack ~]#

```

## 8. Adım

#Centos'ta openstack packag'ın en son sürümünü yükleyin

```
sudo yum install -y centos-release-openstack-train
```

```
sudo yum install yum-utils
```

```
sudo yum-config-manager --enable openstack-train
```

## 9. Adım

Paket güncellerini yapın

```
sudo yum update -y
```

## 10. Adım

#packstack yükleyicisini yükle

```
sudo yum install -y openstack-packstack
```

```

Complete!
[root@openstack ~]# sudo yum install -y openstack-packstack

```

## 11. Adım

Makinamızın IP adresini control ediyoruz.

```
ip address show
```

```
[root@openstack ~]# ip address show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: ens192: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:50:56:01:28:7f brd ff:ff:ff:ff:ff:ff
    inet 172.18.156.25/24 brd 172.18.156.255 scope global ens192
        valid_lft forever preferred_lft forever
    inet6 fe80::250:56ff:fe01:287f/64 scope link
        valid_lft forever preferred_lft forever
3: ens224: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:50:56:01:28:86 brd ff:ff:ff:ff:ff:ff
    inet 172.18.156.6/24 brd 172.18.156.255 scope global ens224
        valid_lft forever preferred_lft forever
    inet6 fe80::250:56ff:fe01:2886/64 scope link
        valid_lft forever preferred_lft forever
```

## 12. Adım

# packstack yükleyicisini aşağıdaki parametreyle çalıştırın

```
packstack --allinone --provision-demo=n --os-neutron-ovs-bridge-mappings=extnet:br-ex --
os-neutron-ml2-mechanism-drivers=openvswitch --os-neutron-l2-agent=openvswitch -- os-
neutron-ovs-bridge-interfaces=br-ex:enp0s3 --os-neutron-ml2-type-drivers=vxlan,flat --os-
neutron-ml2-tenant-network-types=vxlan
```

```
[root@openstack ~]# packstack --allinone --provision-demo=n --os-neutron-ovs-bridge-mappings=extnet:br-ex --os-neutron-ml2-mechanism-drivers=openvswitch --os-neutron-l2-agent=openvswitch -- os-neutron-ovs-bridge-interfaces=br-ex:enp0s3 --os-neutron-ml2-type-drivers=vxlan,flat --os-neutron-ml2-tenant-network-types=vxlan
Welcome to the Packstack setup utility

The installation log file is available at: /var/tmp/packstack/20220831-123701-ZkUXaO/openstack-setup.log
Packstack changed given value  to required value /root/.ssh/id_rsa.pub

Installing:
Clean Up [ DONE ]
Discovering ip protocol version [ DONE ]
Preparing Horizon entries [ DONE ]
Preparing Swift builder entries [ DONE ]
Preparing Swift proxy entries [ DONE ]
Preparing Swift storage entries [ DONE ]
Preparing Gnocchi entries [ DONE ]
Preparing Redis entries [ DONE ]
Preparing Ceilometer entries [ DONE ]
Preparing Aodh entries [ DONE ]
Preparing Puppet manifests [ DONE ]
Copying Puppet modules and manifests [ DONE ]
Applying 172.18.156.6_controller.pp
Testing if puppet apply is finished: 172.18.156.6_controller.pp [ - ]
```

Burada yükleme işlemi 30-60 DK arasında sürebilmektedir, kahvemizi alıp bekleyelim.

```
Applying 172.18.156.6_controller.pp [ DONE ]
Applying 172.18.156.6_controller.pp: [ DONE ]
Applying 172.18.156.6_network.pp [ DONE ]
Applying 172.18.156.6_network.pp: [ DONE ]
Applying 172.18.156.6_compute.pp [ DONE ]
Applying 172.18.156.6_compute.pp: [ DONE ]
Applying Puppet manifests [ DONE ]
Finalizing [ DONE ]

**** Installation completed successfully ****

Additional information:
* A new answerfile was created in: /root/packstack-answers-20220831-183112.txt
* Time synchronization installation was skipped. Please note that unsynchronized time on server instances might be problem for some OpenStack components.
* File /root/keystonerc_admin has been created on OpenStack client host 172.18.156.6. To use the command line tools you need to source the file.
* To access the OpenStack Dashboard browse to http://172.18.156.6/dashboard .
Please, find your login credentials stored in the keystonerc_admin in your home directory.
* Because of the kernel update the host 172.18.156.6 requires reboot.
* The installation log file is available at: /var/tmp/packstack/20220831-183111-b4lYiS/openstack-setup.log
* The generated manifests are available at: /var/tmp/packstack/20220831-183111-b4lYiS/manifests
You have new mail in /var/spool/mail/root
[root@localhost ~]#
```

Bitti.

### 13. Adım

# ethernet arabirim ayarlarınızın böyle görüldüğünden emin olun. Eğer varsa yapmalısın  
IP adresini kaldır  
Arabirim

```
vi /etc/sysconfig/network-scripts/ifcfg-enp0s3
TYPE=OVSPort
NAME=enp0s3
DEVICE=enp0s3
DEVICETYPE=ovs
OVS_BRIDGE=br-ex
ONBOOT=yes
BOOTPROTO=none
```

### 14. Adım

# harici köprü ayarlarınızın aşağıdaki gibi görüldüğünden emin olun

```
DEVICE=br-ex
NAME=br-ex
DEVICETYPE=ovs
TYPE=OVSBridge
OVSBOOTPROTO="none"
IPADDR=<your_IP>
PREFIX=<your_prefix>
GATEWAY=<your_gateway_IP>
IPV4_FAILURE_FATAL=no
IPV6INIT=no
DNS1=<DNS_Server_IP>
ONBOOT=yes
```

### 15. Adım

#bu komut size openstack yönetici ayrıcalıkları sağlar

```
source keystone_admin
```

### 16. Adım

örnekleriniz için sağlayıcı ağını oluşturmak için bu komutu çalıştırın, böylece sağlayıcılar  
# dış dünyayla iletişim kurun

```
neutron net-create external_network --provider:network_type flat -- provider:physical_network
extnet --router:external
```

### 17. Adım

#bu komut sağlayıcı ağına bağlı alt ağı oluşturur. Yapmalısın  
yapıyor olmak



```
neutron subnet-create --name public_subnet --enable_dhcp=False --allocation-pool start=,end= --
gateway= external_network
```

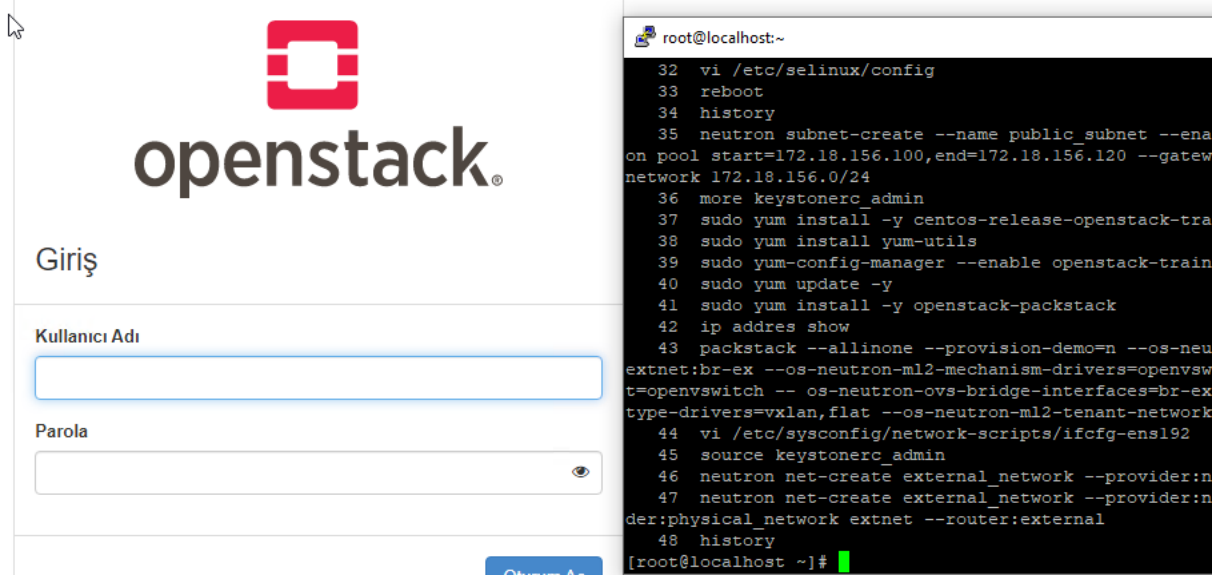
neutron subnet-create --name public\_subnet --enable\_dhcp=False --allocationpool start=172.18.156.100,end=172.18.156.120 --gateway=172.18.156.1 external\_network 172.18.156.0/24

sonrasında 16. Adımdaki komutu çalıştırabiliriz, network yapılandırmasının doğru olduğunu teyit ederiz.

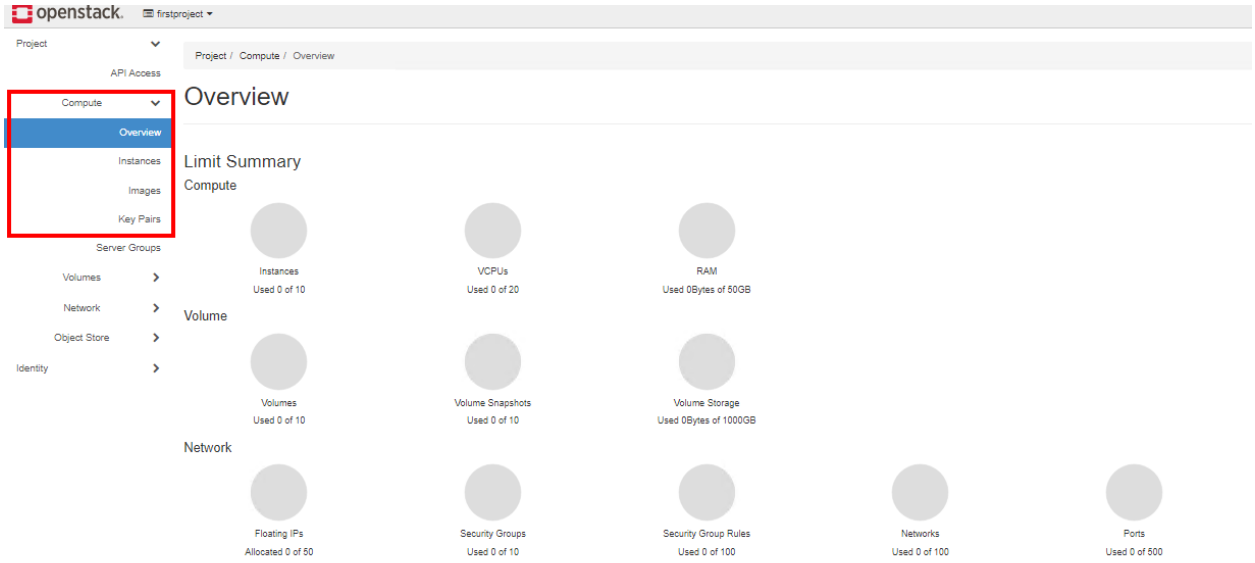
### 18. Adım

More keystone\_rc\_admin

Daha sonra bir web tarayıcıdan openstack'e admin, password bilgilerimiz ile giriş yapabiliriz.



## Openstack – Horizon Menüler



Aşağıdaki açıklamalarda yeşil alanlar standart ve admin kullanıcılar için yer verilmiştir Kırmızı alanlar admin içindir.

### COMPUTE

**Overview** – Openstack bulut ortamımızın üzerinde bulunan kaynakları, makinaleri, Grupları, Network yapılandırmalarının özetle gösterdiği ortamdır

**Instance** – vmware karşılığı sanal makina olan instance alanı, sanal makina oluşturmak, silmek, duraklatmak, snapshot almak için kullandığımız alandır.

**Images** – Projede oluşturulan imajları ve snapshotları görüntülediğimiz alandır. Burada imaj oluşturabilir, düzenleyebilir, storage'daki imajları başlatabiliriz.

**Key Pairs** – Oluşturduğumuz instancalar SSH ile erişebilmek adına keypairs oluşturduğumuz kısımdır.

**Server Groups** – Instance gruplarını yönettiğimiz alandır, Toplu yönetim için kontainer gibi düşünebilir, Vmware karşılığı vApp'dir.

### VOLUMES

**Volumes**: Birimleri oluşturmayı ve görüntülemeye ve silme aksiyonlarını yapabileceğimiz alandır, Kalıcı birimlerdir.

**Snapshot**: Burada volume'lerin snapshotlarını alabileceğimiz bir alandır.

**Group**: Volumeri group halinde yönetmek için kullandığımız alandır.

**Group Snapshot**: Volume gruplarının snapshotlarını yönettiğimiz alandır.

## Network

**Network Topology:** Ağımızın görsel olarak network topolojisini oluşturabildiğimiz alandır.

**Network:** Network kısmı bizim public ve private ağları düzenleyebileceğimiz ve oluşturabileceğimiz alandır.

**Routers:** Altağlar arasında yönlendirmeyi yapılandırmak için kullandığımız alandır.

**Security Group:** Güvenlik grupları, bulut sunucumuza gelen ve gelen trafiği yöneten erişim control listeleri oluşturur.

**Floating IPs:** Bulun sunucumuzun public IP'si yani static IP yönettiğimiz alandır.

**Trunks:** OVS (Open virtual switch) – OVN(Open virtual Newtork) yapılandırabilir, Farklı vlanlar arasında bağlantı ilişkisi yapılandırabilir. Birincil olarak subnet oluşturulup ardından trunk yapılır,

## Object Store

**Containers** OpenStack Nesne Depolama (openstack-swift), nesnelerini (verilerini) iç içe olamamalarına rağmen bir dosya sistemindeki dizinlere benzeyen kaplarda depolar . Kapsayıcılar, kullanıcıların her türlü yapılandırılmamış veriyi depolaması için kolay bir yol sağlar; örneğin nesneler fotoğraflar, metin dosyaları veya resimler içerebilir.

File Storage'dan farklı HTTP üzerinden servis imkanı sunmaktadır.

## Admin

**Overview:** Temel kullanım raporlarının bir görünümünü sağlayan yönetici genel bakışını burada inceleyip indirebiliriz.

## Compute

**Hypervisors** Hipervizor hostumuzun Kaynak grafiklerini buradan inceleyebiliriz.

**Host Aggregates:** Tüm kaynak yönetimini Nova hostu üzerinden alır görüntüleriz, Burada cluster'da bulunan tüm hostlar yer verilmektedir.

**Instance:** Yönetici kullanıcısının instanceler üzerinden, başlatma, duraklatma, restart etme, Askıya alma, Migrate etme gibi aksiyonlarını ele alır. Bu instance yönetimi projelerin tüm instancelerini görüntülemek ve yönetmektir.

**Flavors:** bir bulut yapılandırmasındaki sanal makine örneklerine otomatik olarak atanabilen işlem kaynaklarının boyutunu (sanal CPU sayısı, bellek ve depolama kapasitesi) tanımlar .

**Images:** Projeler dahil tüm imajların yönetimini sağlar, Oluşturabilir, silebilir, düzenleyebiliriz.

## Volume

**Volumes / Snapshots / Volume Types :** VolumeVolume grupları oluşturma silmek ve düzenlemek ve snapshotını almak için kullandığımız menülerdir.

**Groups / Group Snapshots / Groyp Types :** Bu menüde grup türleri gruplandırılmış volume yönetimi içinidir.

## Network

**Networks:** Private ağlar oluşturmak ve düzenlemek için kullandığımız menüdür.

**Routes:** Router'leri yönetici olarak görüntülediğimiz ve düzenleyebildiğimiz alandır, Farklı projeler arası network trafiğini konttrol etmek için kullanmamız mümkün kılar.

**Floating IPs:** Türçe karşılığı kayan IP'nin mantığı HA sağlamaktadır, Network'de oluşan herhangi bir sorunda Floating IPs devreye girer.

**Trunks:** Bulun yöneticisi olarak trunk portalrını oluşturabilir ve yönetebiliriz.

**RBAC Policies:** Belirli projeler için kaynaklara erişim imkanı sunar.

## System

**Defaults:** Sistem tarafından atanan kaynakları güncellemek için kullandığımız alandır.

**Metadata Definitions:** Satıcıların, yöneticilerin, hizmetlerin ve kullanıcıların farklı kaynak türlerinde (images, artifacts, volumes, flavors, aggregates, and other resources) kullanılabilecek kullanılabilir anahtar/değer çifti meta verilerini anlamlı bir şekilde **tanımlamaları** için ortak bir API sağlar.

**System Information:** Bulut servisimizin çalışmakta olduğu servisleri ve üzerinde çalıştıkları sunucuları görebiliriz.

## Identity

**Projects:** Admin user: Yeni projeler yaratabilir, düzenleyebilir, yetkilendirebiliriz. / Standart user: Burada üzerimize atanan projelerimizi görebiliriz., Yada proje atayabiliriz.

**Users** Admin user: Openstack tüm userları yönettiğimiz alandır/ Standart user: Standart kullanıcımızın görüldüğü alandır

**Groups:** Kullanıcıları gruplandırmak için kullandığımız menüdür.

**Roles:** Kullanıcılarımıza için ayrıcalık, yani yetki tanımlayacağımız alandır.

**Application Credentials** Uygulamalar için kimlik bilgileri oluşturmak mümkündür. Projedeki uygulamalar için kimlik bilgisi atar.

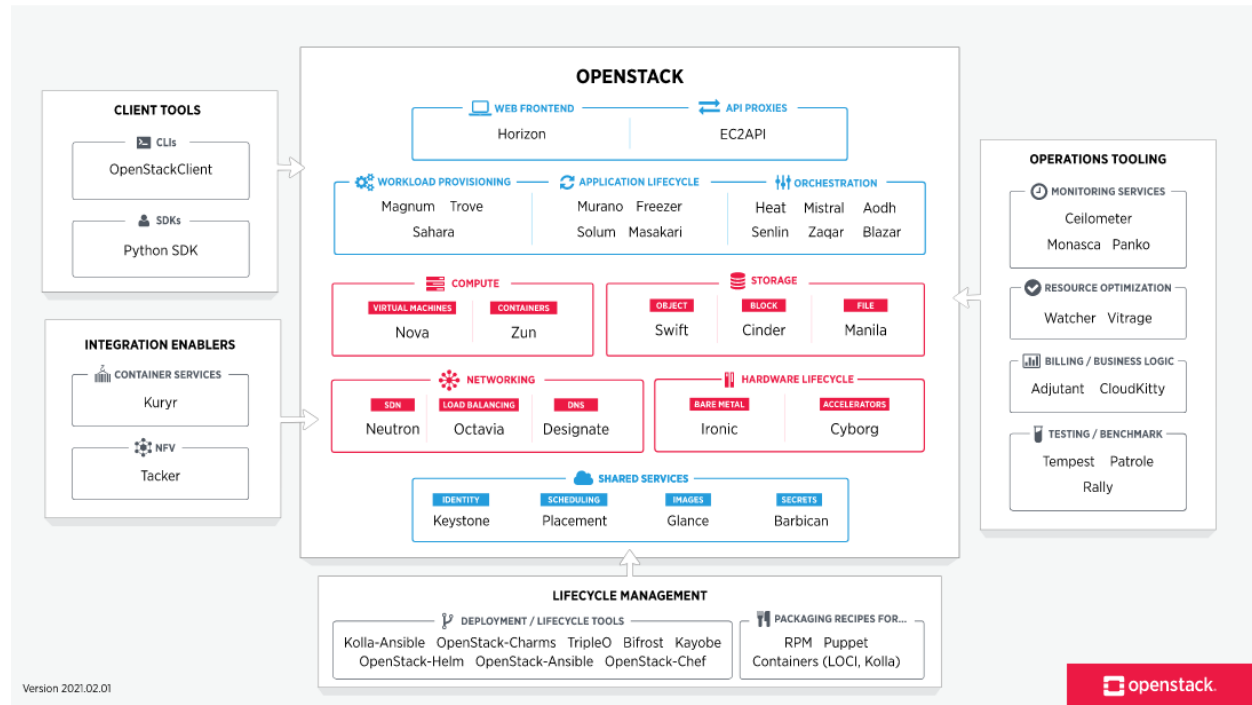
## Project

**API Access:** Buradan, API erişim menüsüne sahibiz, ortamımızdaki tüm API uçlarını görüntüleyebiliriz.

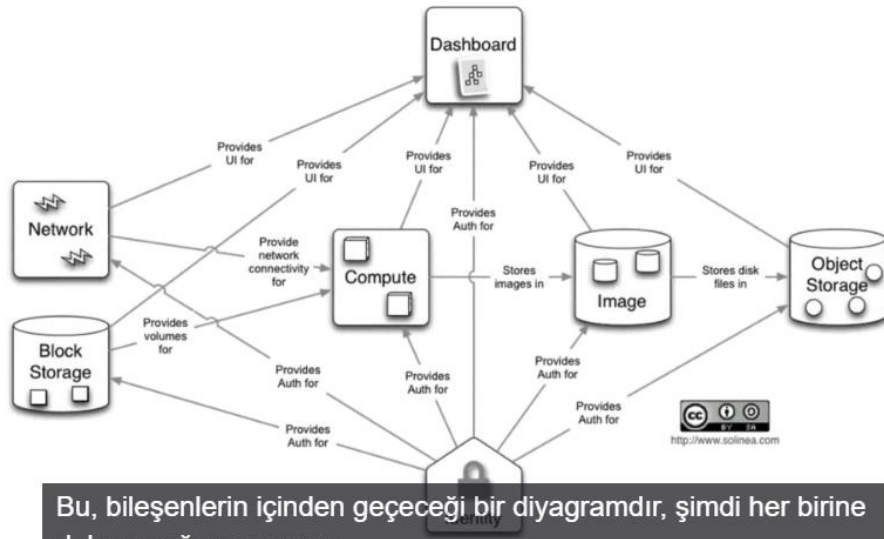
## Openstack API Servisleri

Bir OpenStack dağıtımı, altyapı kaynaklarına erişmek için API'ler sağlayan bir dizi bileşen içerir. Bu sayfada, bulut son kullanıcılarına bu tür kaynakları sağlamak için dağıtılabilecek çeşitli hizmetler listelenmektedir.



### Openstack servis ve API'leri





# Overview





## Compute

	NOVA	Compute Service
	ZUN	Containers Service




## Hardware Lifecycle

	IRONIC	Bare Metal Provisioning Service
	CYBORG	Lifecycle management of accelerators




## Storage

	SWIFT	Object store
	CINDER	Block Storage
	MANILA	Shared filesystems







## Networking

	NEUTRON	Networking
	OCTAVIA	Load balancer
	DESIGNATE	DNS service




## Shared Services

	KEYSTONE	Identity service
	PLACEMENT	Placement service
	GLANCE	Image service
	BARBICAN	Key management





## Orchestration

	HEAT	Orchestration
	SENLIN	Clustering service
	MISTRAL	Workflow service
	ZAQAR	Messaging Service
	BLAZAR	Resource reservation service
	AODH	Alarming Service

## Workload Provisioning

	MAGNUM	Container Orchestration Engine Provisioning
	SAHARA	Big Data Processing Framework Provisioning
	TROVE	Database as a Service


## Application Lifecycle

	MASAKARI	Instances High Availability Service
	MURANO	Application Catalog
	SOLUM	Software Development Lifecycle Automation
	FREEZER	Backup, Restore, and Disaster Recovery

## API Proxies

	EC2API	EC2 API proxy
---	--------	---------------

## Web frontends

	HORIZON	Dashboard
	SKYLINE	Next generation dashboard (tech preview)

## Storage

Openstack bulut kullanıcılarına farklı depolama türleri sunar, Bunlar grafikte yer verildiği gibi geçici depolama, blok depolama, nesne depolama, paylaşımlı dosya sistemi depolama, kısa ömürlü depolama seçenekleridir.

## Storage Types

	Ephemeral Storage	Block Storage	Object Storage	Shared File System Storage
Used to	run operating system and scratch space	add additional persistent storage to a virtual machine	store data/files including VM images	add additional shared persistent storage to a VM shared
accessed through	a file system	a block device that can be partitioned, formatted and mounted (such as dev/vdc)	REST API	a shared file systems service share that can be partitioned, formatted and mounted
accessible from	within a VM	within a VM	anywhere	within a VM
persists until	VM is terminated	deleted by user	deleted by user	deleted by user
managed by	openstack compute	openstack block storage	openstack object storage	openstack shared file system service
typical usage example	10 gb first disk, 30 gb second disk	1 TB disk	10s of TBs of dataset storage	1 TB shared disk

### Ephemeral Storage:

Geçici depolama bir örnek için tahsis edilir ve örnek silindiğinde silinir. İşletim sistemini çalıştırmak için kullanılır. Varsayılan olarak, Hesaplama geçici sürücüler dosya olarak saklar hesaplama düğümündeki yerel diskler \* /var/lib/nova/ örnekleri \* yalnızca sanal makine geçici disk görüntüsünü diğerine taşır hesaplama düğümü (Nova SSH üzerinden kopyalar)

### Block Storage:

Sanal makineye ek kalıcı depolama alanı ekleme • Bu olabilir bir blok cihaz üzerinden erişilir bölümlenmiş, biçimlendirilmiş ve takılı • Yeniden boyutlandırılabilir • Kullanıcı silene kadar devam eder • Şifrelenebilir • Kullanım durumu: uzun süreli çalışma için kalıcı depolama sağlayın güçlü tutarlılık ve düşük gecikme süresi gerektiren hizmetler bağlantı (örn. veritabanları)

### Object Storage:

Nesne depolama, ikili nesneleri depolamak içindir. Kullanıcılar bu ikili nesnelere bir API aracılığı ile erişebilir. Nesne depolama sisteminin iyi bilinen örneklerinden biri Amazon S3'e aşına olabiliriz. Dropbox, Google Driver gibi bulut tüketicilerinin dosyalarımızın sakladığı farklı bir nesne depolama örneğidir.

Nesne depolama, openstack object storage swift project tarafından uygulanır.

Ayrıca, Openstack sanal makine görüntülerimiz bir nesne depolama sistemi içinde saklayabiliriz.

Kullanım durumları: Yedekleme dosyaları veritabanı dökümleri için depolama ve



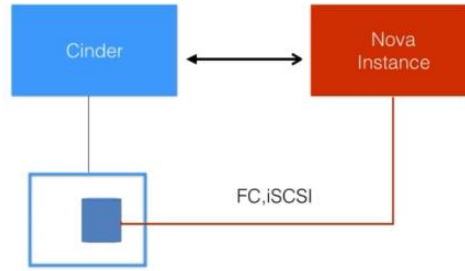
günlük dosyaları; Büyük veri kümeleri (ör. multimedia dosyaları); arka uç görüntü depolama  
Shared File System Storage:

Çoklu kiracılı bir bulut ortamında kullanıcılara bağlantı sağlayarak paylaşılan Dosya system depolaması hizmetiyle etkileşim kurarlar.

## Cinder (Blok Depolama Servisi)

# Cinder Block Storage

- Persistent block level storage devices for use with compute instances
- Block Device Lifecycle Management
- One to one relationship between instances and block storage device
- Manages snapshots
- Manages volume types
- Fully integrated into Nova and Horizon allowing self service



Cinder, Openstack için blok depolama hizmetidir, sonuna kadar depolama kaynaklarını sunmak için tasarlanmıştır. Kullanıcılar tarafından tüketilecek açık yığın ise Project nova'dır.

Son kullanıcılara, bu kaynakları gerektirmeden istemeleri ve kullanmaları için bir self servis API sağlar. Cinder kalıcı sürücülerimiz yaşılan döngüsü yönetimi yapar, yani ciner'ı oluşturabilir, yeniden boyutlandırabilir ve taşıyabiliriz.

Cinder'ı düşündüğümüzde bir sadece USB sürücüsü gibi düşünebiliriz.

Diyelimki bir VM çalıştırmak istiyoruz, sonlandırma işleminden sonra data kaybetmek istemiyoruz, Cinder bunun için kolaylık sağlayacaktır, yeni bir VM çalıştırıp, Cinder'ı mount ettiğimizde bilgilere kolaylıkla erişebiliriz.

Cinder anlık görüntü birimleri, yani snapshot'larında işlemektedir.

Cinder aracılığı ile volume'ler oluşturmak, silmek, genişletmek yada eklemek mümkündür.

## Volumes

- Persistent R/W Block storage
- Attached to instances as secondary storage
- Can be used as root volume to boot instances
- Volume lifecycle management
  - Create, delete, extend volumes
  - Attach/detach Volumes
- Manages volume management

Project / Compute / Volumes

### Volumes

Volume Snapshots

Filter

Create Volume Attach Transfer Detach Volumes

Name	Description	Size	Status	Type	Attached To	Availability Zone	Bootable	Encrypted	Actions
snapshot-01	-	1GB	Available	iscsi	none	none	Yes	No	Edit Volume
snapshot-02	-	1GB	In-use	iscsi	Attached to 1 instance on availability	none	Yes	No	Edit Volume

Displaying 2 items

## Snapshots

A read only copy of a volume

Create/delete snapshots

Create a volume out of a snapshot

Volumes Volume Snapshots

Filter

Delete Volume Snapshots

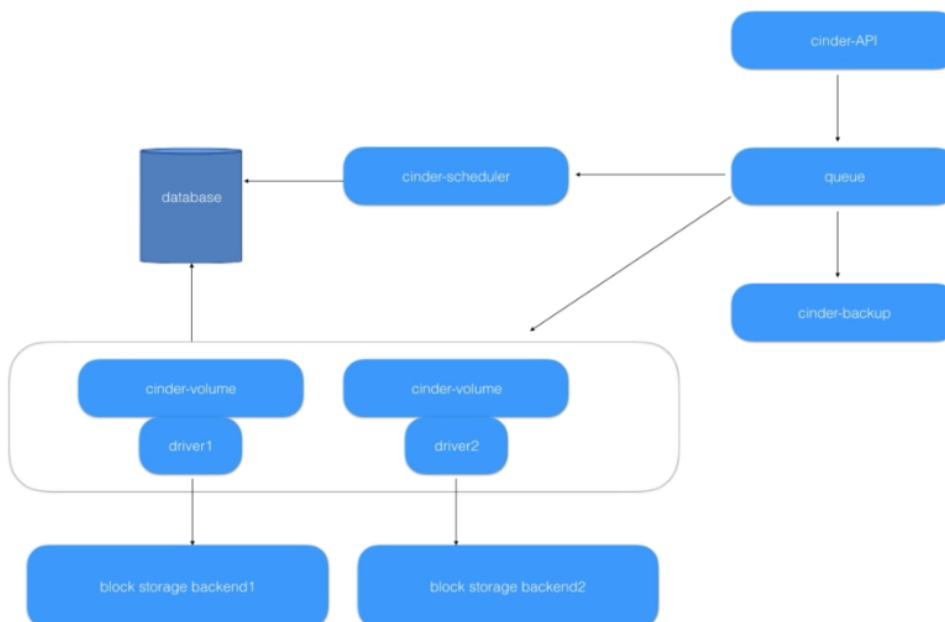
Name	Description	Size	Status	Volume Name	Actions
snapshot-redhat03-01_15_2017	-	1GiB	Available	persistent1	Create Volume
snapshot-cirros-01_15_2017	-	1GiB	Available	persistent1	Create Volume

Cinder ayrıca anlık görüntüde işler, Anlık görüntü birimin zaman kopyasında salt okunur bir noktadır., Blok birimlerinin anlık görüntülerini alabilir.

## Cinder Mimarisi

cinder, birimleri yönetmek için bir altyapı sağlar ve openstack hesaplama ile iletişime geçer.

## Architecture



**Cinder API:** istemcilerden gelen dinleme tabanlı json veya xml istekleri Kabul eden ve doğrulayan WSI uygulamasıdır.

API üzerine gelen istekler;

Volume create/delete/list/show

Create image or snapshot

Snapshot create, delete, list, show

Volume attach, detach ve kota, yedekleme isteklerini karşılar.

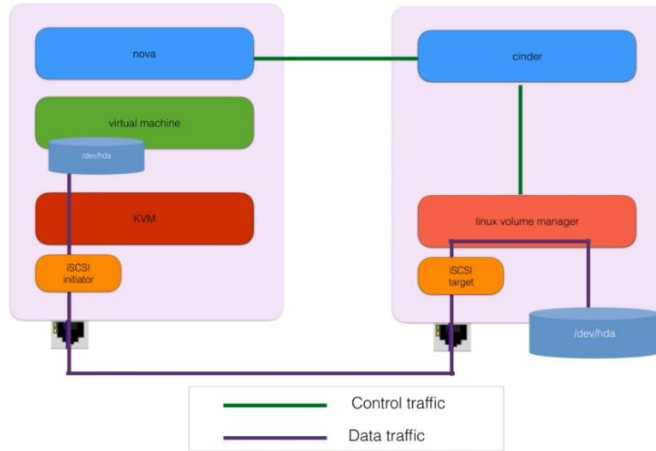
**Cinder volume:** zamanlayıcı gibi diğer gönderen işlemlerden gelen istekleri Kabul eder, Blok depolama hizmetine gönderilen okuma ve yazma isteklerine verdiği yanıt ile çeşitli depolama sağlayıcılarıyla etkileşime geçer.

**Cinder scheduler:** Birimin oluşturacağı en uygun depolama sağlayıcısını seçer.

**Driver:** Çeşitli depolama türleriyle iletişim kurmak için belirli kod'lar içerir. Bunlar EMC, Netapp, LVM depolama denetleyicilerine örnek verebiliriz.

**Cinder Backup:** Her bir türdeki birimleri yedekleme depolama sağlayıcısına yedeklemeyi sağlar. Gönderen birim hizmeti diyebiliriz. Bu sürücü mimarisi aracılığıyla çeşitli depolama sağlayıcılarıyla iletişime geçer.

### Example of Data and Control Traffic For Cinder



Cinder, kalıcı hacimler sağlamak için nova ile iletişime girer.

Bunun yapmanın yolu, her bir bileşenin API'leri aracılığı ile etkileşim kurmaktadır.

## Cinder – CLI ile yönetme

Cinder service list / cinder servislerini listeleriz.

Cinder service-disable / enable (service durdurmak başlatmak için komut)

```

[root@localhost ~(keystone_admin)]# cinder service-list
+-----+-----+-----+-----+-----+-----+-----+
| Binary | Host | Zone | Status | State | Updated_at |
+-----+-----+-----+-----+-----+-----+
| cinder-backup | localhost.localdomain | nova | enabled | up | 2022-09-06T09:27:41.000000 |
| cinder-scheduler | localhost.localdomain | nova | enabled | up | 2022-09-06T09:27:41.000000 |
| cinder-volume | localhost.localdomain@lvm | nova | enabled | up | 2022-09-06T09:27:46.000000 |

```

Openstack command list | grep openstack.volume -A 40 (cinder CLI komutlarını listeler)

Openstack volume create -h (volume oluşturma için komutları listeler)

```

Create new volume
positional arguments:
  <name>                Volume name
optional arguments:
  -h, --help            show this help message and exit
  --size <size>         Volume size in GB
  --type <volume-type>  Set the type of volume
  --image <image>       Use <image> as source of volume (name or ID)
  --snapshot <snapshot> Use <snapshot> as source of volume (name or ID)
  --source <volume>     Volume to clone (name or ID)
  --description <description>
                        Volume description
  --user <user>         Specify an alternate user (name or ID)
  --project <project>   Specify an alternate project (name or ID)
  --availability-zone <availability-zone>
                        Create volume in <availability-zone>
  --property <key=value>
                        Set a property to this volume (repeat option to set
                        multiple properties)

```

openstack volume create --size 1 vol1  
openstack volume list

Openstack volume create --size 1 vol1 çalıştırıyoruz

Openstack volume list

```

[root@localhost ~(keystone_admin)]# openstack volume list
+-----+-----+-----+-----+-----+
| ID | Name | Status | Size | Attached to |
+-----+-----+-----+-----+-----+
| 7b8304c1-8bb2-4261-918b-a8a5815938cf | vol1 | available | 1 | |

```

**Volumumuzu eklemek için**

Openstack server add volume “instance01” “vol1” komutunu çalıştırıyoruz.

Diskimizi instance ekledikten sonra OS’den biçimlendirme yapmak gerekir

```

sudo mkfs.ext3 /dev/vdc
sudo mkdir /mydisk
sudo mount /dev/vdc /mydisk

$ sudo mkdir /mydisk
$ sudo mount /dev/vdc /mydisk
$ ls -al /mydisk
total 21
drwxr-xr-x  3 root  root    4096 Jan 24 14:36 .
drwxr-xr-x 23 root  root   1024 Jan 24 14:37 ..
drwx----- 2 root  root   16384 Jan 24 14:36 lost+found

```

Backup volume oluşturmak istersek ve volume yedeklemek istersek,

openstack volume backup create --name backup1 --force vol1

openstack volume backup show backup1 (backup1 volume bilgisini görüyoruz)

Openstack backup volume'ler swift 'de nesne depolama'da tutulur, Dosya olarak saklanır.

Openstack snapshot create --name snap1 --force "vol1" (volume snapshot almak için komut)

## Swift (Nesne Depolama Servisi)

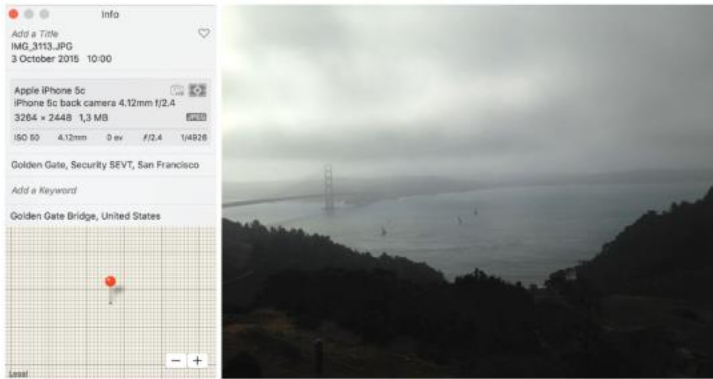
Bir birim VM'e disk olarak hizmet sunarken, Ciner'ın aksiyone Swift içinden içerik geçirmeniz için bir API sunar. Çok temel dosya depolanması ve çok güçlü olabilmektedir.

Vikipedi, Comcast veya Time gibi tüm sistemleri çalıştırmak için nesne depolamayı kullanan web siteleri vardır.

Yüksek eşzamanlılık sağlar, yani aynı anda binlerce kullanıcıya hizmet verebiliriz.

## What Is An Object?

Object = File + Metadata

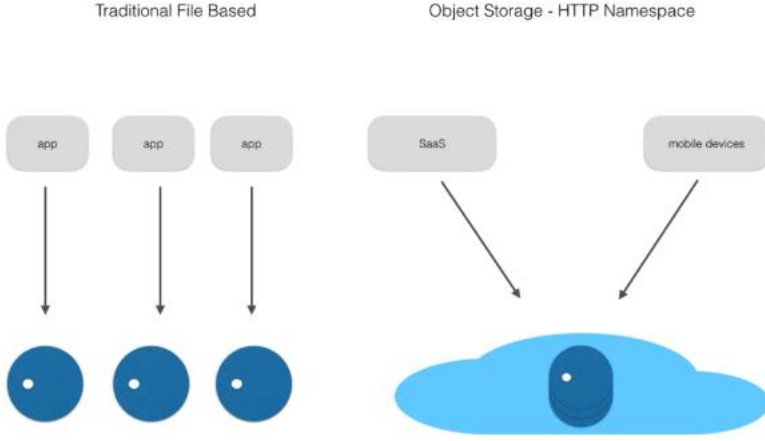


Swift nesneleri içinde saklar, nesne veri parçası olan metaverilere diyebiliriz.

Gördüğünüz gibi jpeg dosyası ile birlikte gelen bir çok bilgi içeriyor.

Resmin kendisiyle birleştirilen tüm bu meta verileri bir nesneyi oluşturur.

# Need for Object Storage



Swift'in normal storage'lardan ayıran özelliği swift'in yerel arayüzü HTTP'dir, yani internet dilini konuşmaktadır.

Diğer büyük fark, Swift'in saf yazılım tabanlı bir çözüm olmasıdır.

Çalıştırmak istediğiniz donanımı seçme ve bu donanımı herhangi bir zamanda ölçeklendirme esnekliği sağlar.

## Swift'in Özellikleri

Geleneksel depolama sistemlerinde genellikle kesinlik tutarlıdır, Örneği bir veri payçası yazdığından birden fazla konuma yazması gerekebilir, Doğru işlemin tamamlanması için, işlemlerin sırayla başarılı şekilde tamamlanması gerekir, Bir konuma yazarken hata varsa tüm doğru işlemlerde başarısız olarak Kabul edilir. Tipik bir depolama örneğidir.

Swift openstack bağlı değildir, Ancak keystone ile entegre olması gerekmektedir.

Swift tutarlı sistemlerden biraz farklıdır. Bir yazma yada birden fazla kopyalama işlemi başlattığınızda, system üç nüsha yazacak şekilde yapılandırılan ve bunlardan ikisini hakların yarısından fazlasını tamamlayan hak olarak Kabul edilir.

Bu son başarısız parça daha sonra asenkron sonradan olarak yazacaktır. 3 parçayı beklememize gerek duymaz. Örneğin, sahip olduğumuz üçüncü bir veri merkezinde bir ağ kesintisi veya denediği bir disk sürücüsü varsa yasmak istediğiniz bozuk diskin değiştirilmesi gereklidir. Sistem kendini iyileştirecektir.

Yüksek kullanılabilirlik için veriyi çoğaltmak için depolama servisi düğümleri arasında rsync protokolü kullanılır. Ayrıca, vekil servisi, istemci sonlandırma noktası ile bulut ortamı arasında ileri geri veri iletirken depolama servisi ile iletişim kurar.

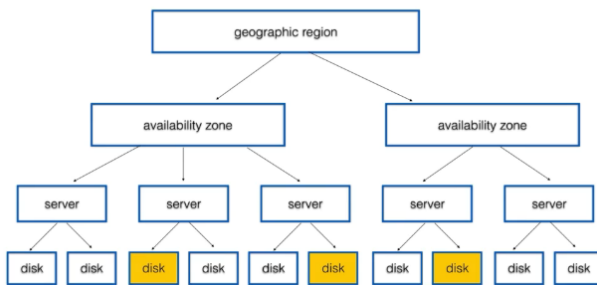
## Durability with Replicas

- Swift stores multiple replicas
- 3 replicas is the default
  - Good balance between durability and cost effectiveness
  - Can be changed
- Swift stores MD5 checksums with each object
  - Returned in header so client can check

Sonuç olarak swift çoğaltılmış bir depolama sistemi olduğudur. Dayanıklılık ve high availability sağlar. Çok fazla CPU gereksinimi olmadan kolay okuma yazma sağlar. Web ve mobil içerikler için tercih edilir.

Swift'in var sayılan çoğaltması 3'dür bu değiştirilebilir.

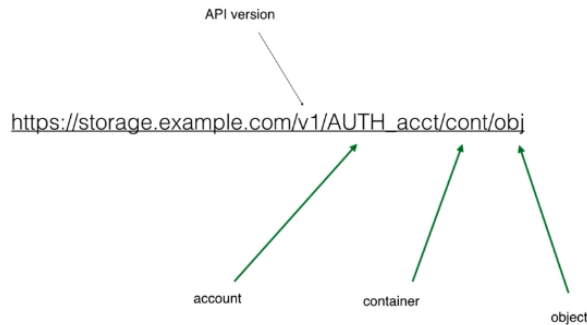
## Data Placement in Swift



Örnek bir mimari sol tarafta görebiliriz.

Swift mimarisi için bir bölgede 2 kopya diğer bölge 3. Kopya mimarisi mevcuttur. (SQL üzerinde alwayson gibi düşünebiliriz)

## Swift essentials: Account, Container and Object



Swift'in dinleme ile konuşmak için bir API kullanmalıdır. Açık, ve static bir ortamda iletişim kurmanın standart bir yolu gibi.

Swift ile konuşmak için standart yanıt kodları kullanır.

Swift'de okumak için http, https protokollerini kullanabiliriz. Hesapların üç önemli parçası vardır, container, nesne ve hesap. Hesabın kullanıcı kimliği olması gerekmez, Kullanım amacına göre oluşturulabilir

## Swift Mimarisi

### The Swift API

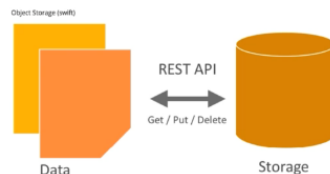
Swift yapmak için PUT, okumak için GET protokolünü kullanırız.

Write an object:

PUT /v1/account/container/object

Read an object:

GET /v1/account/container/object



Swift ile konuşmak için web tarayıcı kullanabiliriz.

Sağ taraftaki yapıda yeşil olan kısım donanım ve yazılım, mavi olan kısım swift'in yapısıdır.

LB: tüm iletişimler nedeniyle çok sayıda isteğe hizmet vermek için yük dengeleme yapmak gerekir.

SSL: Güvenlik amaçlıdır, Kritik verilerinizi korumak için SSL sonlandırma özelliğine ihtiyacı vardır.

Authentication: Swift verilere ulaşmak için kimlik doğrulama ihtiyacı vardır.

Proxy: Veri isteklerini el alır okur veya yazar.

Container: nesnelerin gruplandırılması ve listelenmesi için servis sağlar

Object: Proxy sunucuları ile disk depolama alanı arasında arabirim sağlar.

Replication: swift kümeleri, tek bir düğümden birden fazla veri merkezine kapsayan ve taşıyan servistir.



## Swift – CLI Yönetimi

Openstack object store account show (yönetici hesabımıza ilişkili hesabımızda her obje bu account altına yerleşecektir)

Openstack container list (swift container listeler.)

Openstack container create “container adı” / container oluşturur.

```
[root@localhost ~](keystone_admin)]# openstack container list
+-----+
| Name |
+-----+
| container2 |
| volumebackups |
+-----+
```

openstack object create -h / obje oluşturmak için yardım çıktısını verir

Openstack object create “container adı” “dosya adı”

Openstack object create container2 keystone\_admin / keystone dosyasını container2 içine obje olarak atıyoruz.

```
[root@localhost ~](keystone_admin)]# openstack object create container2 keystone_admin
+-----+
| object | container | etag |
+-----+
| keystone_admin | container2 | 94a73b0938c7437440fc90428a7f4dcd |
+-----+
[root@localhost ~](keystone_admin)]#
```

Horizonta control ettiğimizde sorun şekilde erişebildiğimizi görüyoruz.



## Containers

[+ Container](#)

container2	
Object Count:	1
Size:	372 bytes
Date Created:	Sep 7, 2022
<input checked="" type="checkbox"/> Public Access:	<a href="#">Link</a>

volumebackups

container2

Displaying 1 item

☐ Name ^☐ keystone\_admin

Displaying 1 item

[←](#) [→](#) [↺](#) [⚠ Güvenli değil | 172.18.156.6:8080/v1/AUTH\\_e01a73d0436744b1acab574e20b55ad7/container2](#)

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<?xml version="1.0" encoding="UTF-8" ?>
<container name="container2">
  <object>
    <name>keystone_admin</name>
    <hash>94a73b0938c7437440fc90428a7f4dcd</hash>
    <bytes>372</bytes>
    <content_type>application/octet-stream</content_type>
    <last_modified>2022-09-07T09:59:45.430220</last_modified>
  </object>
</container>
```

## Compute

### Nova (Openstack işlem servisi)

Bilgi işlem hizmeti, bulut ortamındaki bilgi işlem kaynaklarının korunmasından ve yönetilmesinden sorumlu olan openstack'in temel hizmetlerinden biridir. Openstack projesinde kod adı nova'dır.

Nova'nın kendisi herhangi bir sanallaştırma yeteneği sağlamaz. Bilgi işlem hizmetleri sağlar ve temel alınan Hypervisor (sanal makine yöneticisi) ile etkileşim kurmak için farklı sanallaştırma sürücülerini kullanır. Tüm bilgi işlem örnekleri (sanal sunucular) yaşam döngüsü zamanlaması (başlatma, askıya alma, durdurma, silme vb.) için Nova tarafından yönetilir

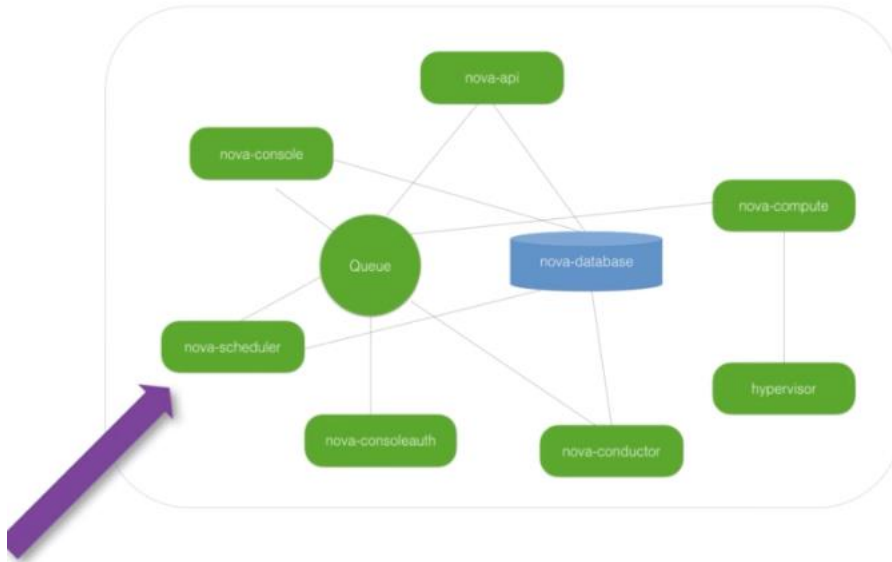
Nova, keystone, glance, neutron, cinder ve swift gibi diğer hizmetlerin desteğine ihtiyaç duyar ve şifreli diskler ve çıplak metal bilgi işlem örnekleri uygulamak için bu hizmetlerle entegre olabilir.

Bilgi işlem hizmeti NOVA, openstack'in en önemli çekirdek servsidir. Nova, sanal makine anlamına gelen zengin API örneği ile anında yaşam döngüsü yönetimini sağlar. Nova bu anlamda Hipervizör yöneticisidir.

Openstack ölçeği büyütmemiz gerektiğinde yapacağımız şey Nova'yı başka bir sunucuya kurmak ve openstack'in bir parçası haline getirmektir.

Nova, ESXI, Hyper-V KVM gibi hipervizörlerle yerleşik bir entegrasyon yeteneğine sahiptir.

### NOVA MİMARİSİ



Queue: Proje içindendeki tüm iletişimi, merkezde görüldüğü gibi bir RPC mesaj aracılığıyla iletir.

Nova-conductor: Tüm işlemler ait istekler nova-conductor'a iletilir. Hesaplama düğümlerini iletken aracılığıyla veritabanına iletir.

Nova-Scheduler: Novadaki zamanlayıcı sürecinin yapısıdır.

Nova-compute: Bir sanal makinasan istek alır ve hangi bilgisayar kodunda olması gerektiği belirtir.

Nova-database: Bir bulut alyyapısı için derleme zamanını var çalışma zamanının varlıklarını depolar.

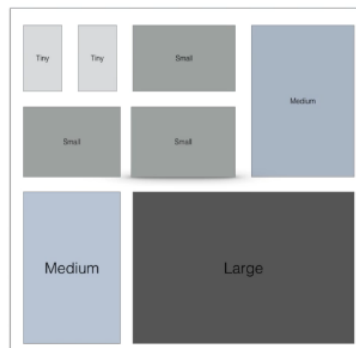
## Flavor (İmage Servisi)

(Instance oluşturken hangi kaynak değerlerince vereceğimizin ön şablonu olarak düşünebiliriz, imajımızı glance'den seçtikten sonra flavor'da seçerek kaynak değerlerini belirtebiliriz.)

Openstack flavor list komutu ile var olan flavor'ları görebiliriz.

### Flavor Selection

- Simplify the process of packing instances onto physical hosts
- Typically each flavor is twice the size(CPU, Disk, RAM) of the smaller one
- Flavors can be customised by the administrator



Flavor oluşturmak için CLI komutu,

Openstack flavor create -h (help) ile inceleyebiliriz.

```
optional arguments:
  -h, --help            show this help message and exit
  --id <id>             Unique flavor ID: 'auto' creates a UUID (default:
                        auto)
  --ram <size-mb>       Memory size in MB (default 256M)
  --disk <size-gb>      Disk size in GB (default 0G)
  --ephemeral <size-gb> Temporary disk size in GB (default 0G)
  --swap <size-mb>      Additional swap space size in MB (default 0M)
  --vcpus <vcpus>       Number of vcpus (default 1)
  --rxtx-factor <factor> RX/TX factor (default 1.0)
  --public              Flavor is available to other projects (default)
  --private             Flavor is not available to other projects
  --property <key=value> Property to add for this flavor (repeat option to set
                        multiple properties)
  --project <project>   Allow <project> to access private flavor (name or ID)
                        (Must be used with --private option)
  --description <description> Description for the flavor. (Supported by API versions
                        '2.55' - '2.latest')
  --project-domain <project-domain> Domain the project belongs to (name or ID). This can
                        be used in case collisions between project names
                        exist.
```

openstack flavor create --id "10" --ram "256" --disk "2" --public "m1.tinier"

flavorumuzu oluşturduk (openstack flavor list)

ID	Name	RAM	Disk	Ephemeral	VCPUs	Is Public
1	ml.tiny	512	1	0	1	True
10	ml.tinier	256	2	0	1	True
2	ml.small	2048	20	0	1	True
3	ml.medium	4096	40	0	2	True
4	ml.large	8192	80	0	4	True
5	ml.xlarge	16384	160	0	8	True

Şimdide CLI ile bir instance oluşturmaya devam edelim

```
openstack server create --image <image_name> --key-name <keypair_name> --flavor  
<flavor_ID> --nic net-id=<network_ID> <instance_name>
```

“openstack server create --image cirros --key-name mykeypair --flavor 10 --nic net-id=e35fc8f5-45ec-48a6-9541-3d686e89bf13 intanece01 “ komutumuz çalıştırarak oluşturduk.

İnstanclarımızın geldiği horizon üzerinden görebiliyoruz.

Admin / Compute / Instances

Overview

Hypervisors

Aggregates

Instances

Flavors

Images

Displaying 2 items

<input type="checkbox"/>	Project	Host	Name	Image Name
<input type="checkbox"/>	admin	-	instance01	cirros
<input type="checkbox"/>	admin	-	intanece01	cirros

Displaying 2 items

## Zun (Container Servisi)

Zun nedir?

Zun bir OpenStack Konteyner servsidir. Sunucuları veya kümeleri yönetmeye gerek kalmadan uygulama kapsayıcılarını çalıştırmak için bir API hizmeti sağlamayı amaçlamaktadır.

Temel işlev için aşağıdaki ek OpenStack servislerini gerektirir:

[Keystone](#)

[Nötron](#)

[Kuryr-libnetwork](#)

Ayrıca aşağıdakileri dahil etmek için diğer hizmetlerle entegre olabilir:

Cinder

Heat

Glance

### Son Kullanıcılar İçin

Zun'un son kullanıcısı olarak, araçlar veya API ile doğrudan kapsayıcı iş yükü oluşturmak ve yönetmek için Zun'u kullanacaksınız. Zun'un tüm son kullanıcı (ve bazı idari) özellikleri, doğrudan tüketilebilen bir REST API aracılığıyla kullanıma sunulur. Aşağıdaki kaynaklar, API'yi doğrudan kullanmaya başlamanıza yardımcı olacaktır.

[API Başvurusu](#)

Alternatif olarak, son kullanıcılar REST API'yi çeşitli araçlar veya SDK'lar aracılığıyla kullanabilir. Bu araçlar aşağıda toplanmıştır.

[Horizon](#): OpenStack Projesi için resmi web kullanıcı arayüzü.

[OpenStack İstemcisi](#): OpenStack Projeleri için resmi CLI.

[Zun İstemcisi](#): Zun'un API'sini kullanan Python istemcisi.

### Operatörler İçin

#### Kurma

Zun için ayrıntılı kurulum kılavuzu. İşlevsel bir Zun ayrıca [Keystone](#), [Neutron](#) ve [Kuryr-libnetwork](#)'ün kurulmasını gerektirecektir. Lütfen önce yükleme kılavuzlarını izlediğinizden emin olun.

# Networking

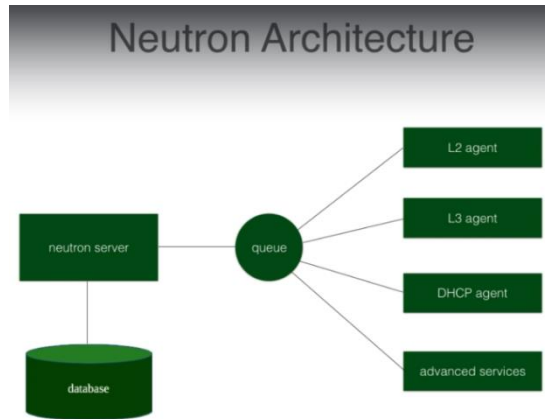
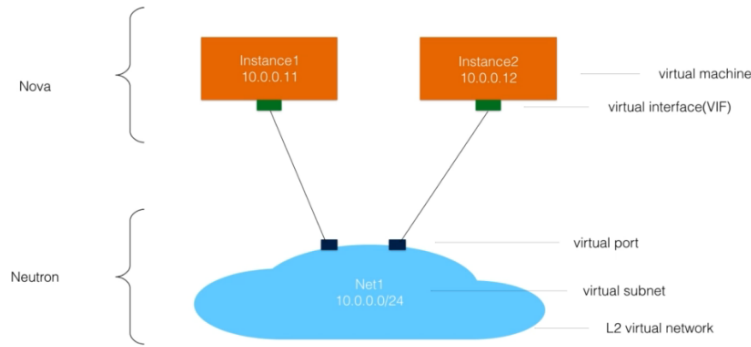
## Neutron (Network servisi)

Neutron servisi, ağ ile ilgili tüm yapılandırmayı yerine getirmekle sorumludur. Neutron'u API'si ile konfigre edebiliriz.

Dış dünya ile haberleşirebilir, Yönlendirme yapabilir, LB, VPN yapısı kurabiliriz, genel olarak tüm firewall'larda olan donanımları kapsar diyebiliriz.

Neutron ile zengin topolijiler oluşturabiliriz.

Nova ile Neutron grafiği aşağıdaki gibidir.



L2 agent: Bağlantıları kablolamakla ve aygıtları birbirine bağlamakla sorumludur.

L3 agent: farklı networkler arasında bağlantı kullanan servistir.

DHCP agent: DHCP protokolünü kullanan servistir

Advanced Services: Firewall, VPN, LB servislerini kullanan servistir.

## Network Technologies Supported

Desteklenen teknoloji segmentleri sol taraftaki gibidir..

- Traffic segmentation is a must!
- For scalability, security and network management
- Large networks degrade performance
- Need to separate tenant traffic
- Choices:
- Local network
- Flat network
- VLAN
- IEEE 802.1Q (up to 4096)
- Underlay must support
- Tunneling(GRE/VxLAN)
- L2 encapsulated in L3
- Underlay independent

D

Local network: Biririne bağlı nodeler arasında iletişim gerçekleşir.

Flat network: Düz ağ, bize herhanbir segmentasyon sağlamayan ağdır, single broadcast domain üzerinden yayınlanır. Ağ üzerinde iyi bir yalıtım sağlanamaz

Vlan network: Flat network aksiyone birden fazla yayın alanını paylaşır. Ölçeklenebilir ve güvenilirdir. 4096 vlana kadar destekler. Farklı vlanlar arasında konusturma yapma için yönlendirme yapılması gerekir

GRE and VXLAN Tunnels: VXLAN, geniş çaplı bulut bilişim dağıtımlarında görülen ağların ölçeklenebilirlik sorunlarına bir çözüm amacıyla tasarlanan yeni bir ağ sanallaştırma tekniği olarak tanımlanmaktadır. VXLAN teknik olarak Ethernet protokolünün tünellenmesi yöntemi ile çalışır. Tünelleme işlemini, UDP paketine veya ilgili cihazlarla uyumlu bir taşıma mekanizması içine eklenmiş yeni bir Ethernet frame (header bilgisi olarak ekstradan birkaç yeni byte eklenmiş şekilde) gerçekleştirir.

### Neutron özellikleri ve işlevselliği

**Security Group:** Güvenlik grupları kurallar, yöneticilerin ve kiracıların trafik türünü belirtebilmeleri olarak tanır. Çakışan IP adresleri desteklenir, kiracılar arasında sorun yaratmaz. IPv6desteklemektedir. Sanal birim oluşturulduğunda bir security grubuyla ilişkilendirilir. Default olarak giriş ve çıkış trafiği izin verilmiş şekildedir.

**NAT:** Ağ adresi çevirisi olarak bilinir, Kaynak adresleri hedef adreslere doğru yazılan kurallar neticesinde yönlendirir. SNAT(source address translation), DNAT (Destination Address translation), PAT(port address translation) gibi nat işlemleri yapılabilir.

**DVR(Distributed Virtual Route):**Neturon dağıtılmış sanal yönlendiricili L3 agent katmanında çalışır. Sanal Yönlendirici Artıklık Protokolü (VRRP) kullanarak büyütme destekler. Bu yapılandırmayı kullanarak, sanal yönlendiriciler hem --dağıtılmış hem de --ha seçeneklerini destekler.

Eski HA yönlendiricilere benzer şekilde, DVR/SNAT HA yönlendiricileri, SNAT hizmetinin farklı bir düğümde çalışan bir l3 aracısındaki yedek DVR/SNAT yönlendiricisine hızlı bir şekilde yük devretmesini sağlar.

**Floating IP Adresses:** Kayan IP, yalnızca özel bir alt ağda (yani ortak ağ arabirimi olmadan) oluşturulan örnekler için ortak, statik bir IP adresidir. Kayan IP kullanarak, böyle bir örnek İnternet'ten gelen bağlantıları kabul edebilecektir.

Kayan IP ile, ortak ağ arabirimi olmadan harici bir ağdan bulut sunucusuna hızlı bir şekilde erişim sağlayabilirsiniz. Kısaca Bulutunuza public IP'nin eklenmesidir.

### **Neutron CLI yönetimi – Network oluşturma**

Openstack network create “network id”

openstack subnet create “subnet adı” --subnet-range “10.5.5.0/24” --dns-nameserver “8.8.8.8” — network “network id”

openstack network list / subnet list (ile oluşturulan network görebiliriz, tabi internete çıkış için rout etmemiz ve kura yazmamışta gerekecektir)

## Designate (DNS Servisi)

OpenStack için bir DNSaaS bileşeni olan Designate API servisidir.

Designate, OpenStack için çok kiracılı bir DNSaaS hizmetidir. Entegre Keystone kimlik doğrulaması ile bir REST API sağlar. Nova ve Neutron eylemlerine dayalı olarak kayıtları otomatik olarak oluşturmak üzere yapılandırılabilir. Designate, Bind9 ve PowerDNS 4 dahil olmak üzere çeşitli DNS sunucularını destekler.



Red Hat Enterprise Linux ve CentOS için kurun ve yapılandırın

## Önkoşullar

DNS hizmetini yükleyip yapılandırmadan önce hizmet kimlik bilgileri ve API uç noktaları oluşturmalısınız.

1. **admin**Yalnızca yönetici CLI komutlarına erişim elde etmek için kimlik bilgilerini kaynaklayın:

2. `$ source admin-openrc`

3. Hizmet kimlik bilgilerini oluşturmak için şu adımları tamamlayın:

- Kullanıcıyı oluşturun **designate**:

- `$ openstack user create --domain default --password-prompt designate`

- **admin**Rolü kullanıcıya ekleyin **designate**:

- `$ openstack role add --project service --user designate admin`

- Belirlenen hizmet varlıklarını oluşturun:

- `$ openstack service create --name designate --description "DNS" dns`

4. DNS hizmeti API uç noktasını oluşturun:

5. `$ openstack endpoint create --region RegionOne \`

```
dns public http://controller:9001/
```

- 
1. Paketleri kurun:

2. `# yum install openstack-designate\*`

3. **designate**Kullanıcının erişebileceği bir veritabanı oluşturun **designate** . **DESIGNATE\_DBPASS**Uygun bir şifre ile değiştirin :

```
4. # mysql
5. MariaDB [(none)]> CREATE DATABASE designate CHARACTER SET utf8 COLLATE utf8_general_ci;
6. MariaDB [(none)]> GRANT ALL PRIVILEGES ON designate.* TO 'designate'@'localhost' \
7. IDENTIFIED BY 'DESIGNATE_DBPASS';
8. MariaDB [(none)]> GRANT ALL PRIVILEGES ON designate.* TO 'designate'@'%' \
9. IDENTIFIED BY 'DESIGNATE_DBPASS';
```

10. BIND paketlerini kurun:

```
11. # yum install bind bind-utils
```

12. Bir RNDC Anahtarı oluşturun:

```
13. # rndc-confgen -a -k designate -c /etc/designate/rndc.key -r /dev/urandom
```

14. **/etc/named.conf** Dosyaya aşağıdaki seçenekleri ekleyin :

```
15. ...
16. include "/etc/designate/rndc.key";
17.
18. options {
19.     ...
20.     allow-new-zones yes;
21.     request-ixfr no;
22.     listen-on port 53 { 127.0.0.1; };
23.     recursion no;
24.     allow-query { 127.0.0.1; };
25. };
26.
27. controls {
28.     inet 127.0.0.1 port 953
29.     allow { 127.0.0.1; } keys { "designate"; };
```

```
30. };
```

31. DNS hizmetini başlatın ve sistem önyüklendiğinde başlayacak şekilde yapılandırın:

```
32. # systemctl enable named
```

```
33.
```

```
34. # systemctl start named
```

35. Dosyayı düzenleyin `/etc/designate/designate.conf` ve aşağıdaki işlemleri tamamlayın:

- Bölümde `[service:api]`, yapılandırın `auth_strategy`:

- `[service:api]`
- `listen = 0.0.0.0:9001`
- `auth_strategy = keystone`
- `enable_api_v2 = True`
- `enable_api_admin = True`
- `enable_host_header = True`
- `enabled_extensions_admin = quotas, reports`

- Bölümde `[keystone_authtoken]`, aşağıdaki seçenekleri yapılandırın:

- `[keystone_authtoken]`
- `auth_type = password`
- `username = designate`
- `password = DESIGNATE_PASS`
- `project_name = service`
- `project_domain_name = Default`
- `user_domain_name = Default`
- `www_authenticate_uri = http://controller:5000/`
- `auth_url = http://controller:5000/`
- `memcached_servers = controller:11211`

Kimlik hizmetinde kullanıcı `DESIGNATE_PASS` için seçtiğiniz parolayla değiştirin `.designate`

- Bu bölümde, mesaj kuyruğu erişimini **[DEFAULT]** yapılandırın **:RabbitMQ**

```
○ [DEFAULT]
○ # ...
○ transport_url = rabbit://openstack:RABBIT_PASS@controller:5672/
```

RabbitMQ'da hesap **RABBIT\_PASS** için seçtiğiniz şifre ile değiştirin **.openstack**

- Bölümde **[storage:sqlalchemy]**, veritabanı erişimini yapılandırın:

```
○ [storage:sqlalchemy]
○ connection = mysql+pymysql://designate:DESIGNATE_DBPASS@controller/designate
```

Veritabanı **DESIGNATE\_DBPASS** için seçtiğiniz parola ile değiştirin **.designate**

- atama veritabanını doldurun

```
○ # su -s /bin/sh -c "designate-manage database sync" designate
```

36. Atanan merkezi ve API hizmetlerini başlatın ve bunları sistem önyüklendiğinde başlayacak şekilde yapılandırın:

```
37. # systemctl start designate-central designate-api
38.
39. # systemctl enable designate-central designate-api
```

40. **/etc/designate/pools.yaml** Aşağıdaki içeriklerle bir pools.yaml dosyası oluşturun :

```
41. - name: default
42.   # The name is immutable. There will be no option to change the name after
43.   # creation and the only way will to change it will be to delete it
44.   # (and all zones associated with it) and recreate it.
45.   description: Default Pool
46.
47.   attributes: {}
```

```
48.
49.  # List out the NS records for zones hosted within this pool
50.  # This should be a record that is created outside of designate, that
51.  # points to the public IP of the controller node.
52.  ns_records:
53.    - hostname: ns1-1.example.org.
54.      priority: 1
55.
56.  # List out the nameservers for this pool. These are the actual BIND servers.
57.  # We use these to verify changes have propagated to all nameservers.
58.  nameservers:
59.    - host: 127.0.0.1
60.      port: 53
61.
62.  # List out the targets for this pool. For BIND there will be one
63.  # entry for each BIND server, as we have to run rndc command on each server
64.  targets:
65.    - type: bind9
66.      description: BIND9 Server 1
67.
68.    # List out the designate-mdns servers from which BIND servers should
69.    # request zone transfers (AXFRs) from.
70.    # This should be the IP of the controller node.
71.    # If you have multiple controllers you can add multiple masters
72.    # by running designate-mdns on them, and adding them here.
73.    masters:
74.      - host: 127.0.0.1
75.        port: 5354
76.
77.    # BIND Configuration options
78.    options:
79.      host: 127.0.0.1
80.      port: 53
```

```
81.      rndc_host: 127.0.0.1
82.      rndc_port: 953
83.      rndc_key_file: /etc/designate/rndc.key
```

84. Havuzları güncelleyin:

```
85. # su -s /bin/sh -c "designate-manage pool update" designate
```

86. atama ve mDNS hizmetlerini başlatın ve bunları sistem önyüklendiğinde başlayacak şekilde yapılandırın:

```
87. # systemctl start designate-worker designate-producer designate-mdns
88.
```

```
# systemctl enable designate-worker designate-producer designate-mdns
```

Kaynak (<https://docs.openstack.org/designate/latest/install/install-rdo.html>)

## Shared Service

### Keystone (Kimlik Servisi)

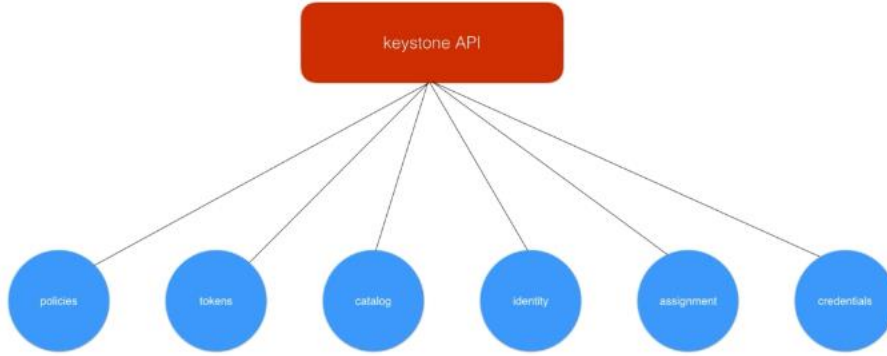
Keystone servisi kimlik doğrulama ve yetkilendirme için kullanılır, Ve tüm geçerli hizmetler openstack üzerindeki API servisleri üzerinden request'lerde doğrulama yaparak işlemlerini gerçekleştirmektedir. Örnek olarak bir user, VM oluşturmak istediğinde Nova üzerinden doğrulama alır, Network oluşturmak istediğinde neutron servisi üzerinden doğrulama alır, gibi düşünebiliriz.

Keystone servisi LDAP desteklemektedir, AD entegrasyonu bu noktada mümkündür.

Keystone servisi aşağıdaki aksiyonlarda rol oynamaktadır.

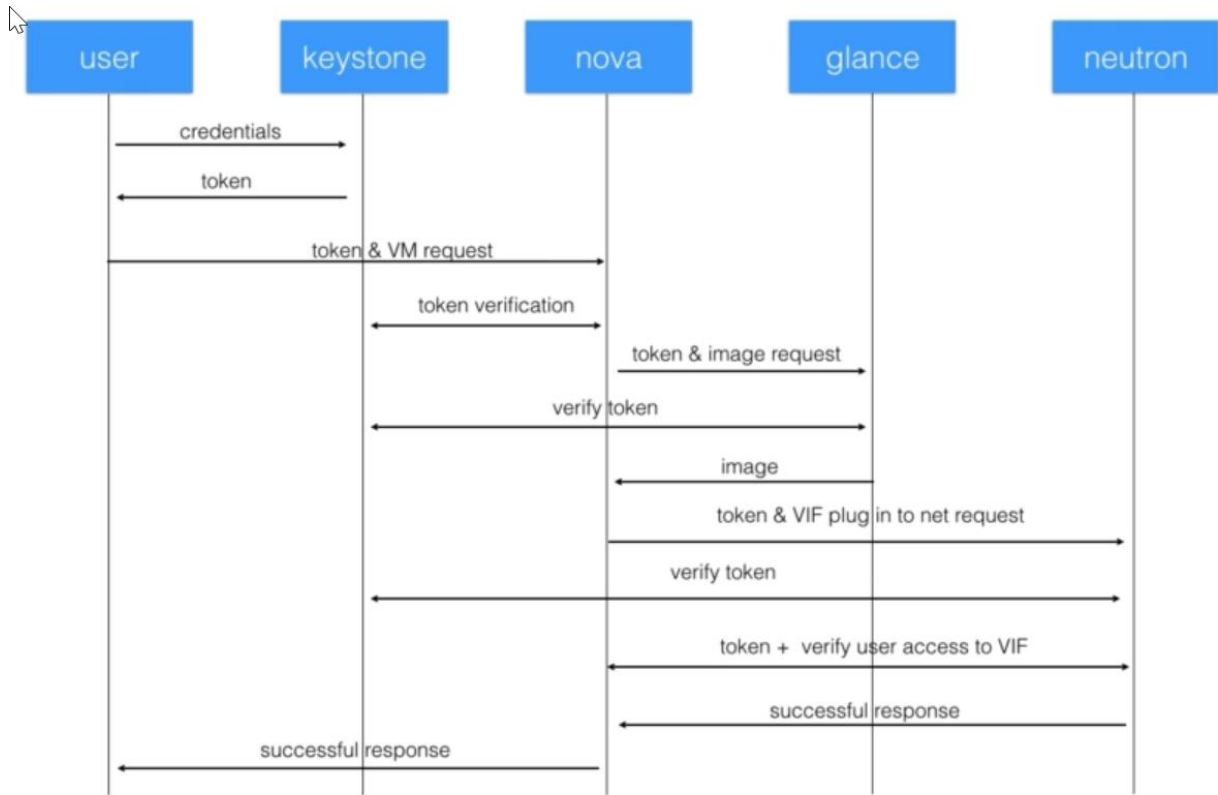
- Kullanıcılar
- Projeler (Yani kiracılar) Hem global bazında yetki , Hemde projeye bazında yetkilendirme
- Roller(Yetkilendirme ve sınırlandırma)

- Token(Openstack erişim doğrulaması), Belirlenen zaman diliminde token ile sınırlı bir sürede erişim sağlayabilir, yada iptal edebilir.
- Katalog (API servisleri keystone üzerinde request işlemlerini gerçekleştirir.)



Keystone aşağıdaki grafikteki gibi çalışır.

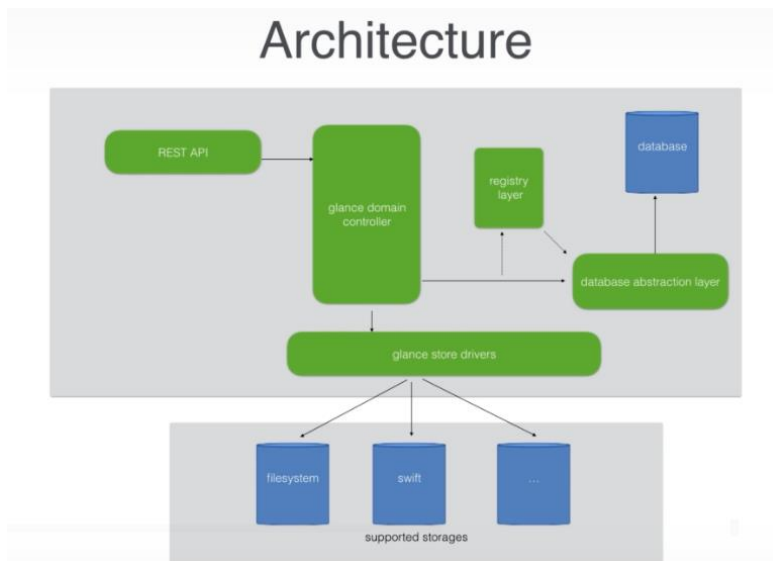
API'ler üzerinden istekler, keystone URL servisine ulaşır ve istek gönderir. Sadece ilk credential erişiminden sonra keystone user tarafından request gitmez onun haricinde diğer servisler her bir istekte keystone gitmektedir.



### Glance (İmaj Servisi)

Nova ile yeni bir VM başlatmak istediğimizde, işletim sisteminin yeni bir yüklemesini yapamayız. Glance üzerinde önceden oluşturduğumuz imajlarımızı kullanmamız gerekir.

Dışarıdan başka bir imaj ekleyebiliriz. RAW, VMDK(Vmware), VDI(virtualbax), VHD(Hyper-V), OVF, Qcow2(KVW) gibi imaj tiplerini desteklemektedir. Glance çalışma mimarisi aşağıdaki grafikteki gibidir.





CLI yönetmek için örnek uygulama aşağıdaki gibidir

openstack command list | grep openstack.image -A 30 (ile glance servisine ait komutlarını listeleyebiliriz.)

```
root@localhost ~(keystone_admin)]# openstack command list | grep openstack.image -A 30
openstack.image.v2
| image add project
| image create
| image delete
| image list
| image member list
| image remove project
| image save
| image set
| image show
| image unset
| openstack.key_manager.v1
| acl delete
| acl get
| acl submit
| acl user add
| acl user remove
| ca get
| ca list
| secret container create
| secret container delete
| secret container get
| secret container list
| secret delete
| secret get
| secret list
| secret order create
| secret order delete
| secret order get
| secret order list
| secret store
| secret update
openstack.load_balancer.v2
| loadbalancer amphora configure
```

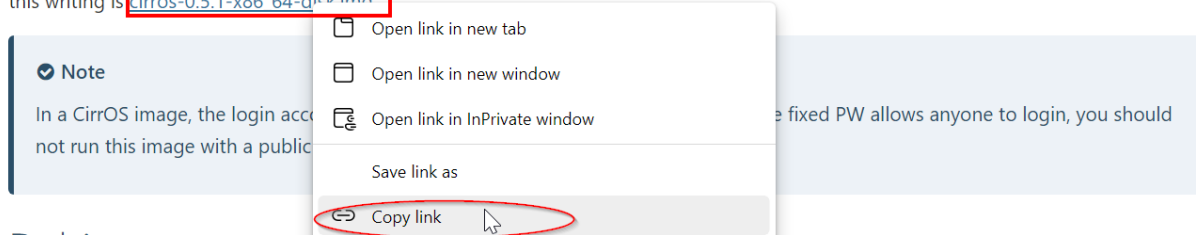
Openstack image list (openstack bulutumuzdaki imajları listelemek için kullandığımız komut)

Google’da Cirros cloud image aratarak hazır olarak bulunan küçük boyutlu imajları elde edebiliriz.

## Cirros (test)

Cirros is a minimal Linux distribution that was designed for use as a test image on clouds such as OpenStack Compute. You can download a Cirros image in various formats from the [Cirros download page](#).

If your deployment uses QEMU or KVM, we recommend using the images in qcow2 format. The most recent 64-bit qcow2 image as of this writing is [cirros-0.5.1-x86\\_64-disk.img](#).



İndirmek için

Curl -o “localfilestorageimage” “imageurl” komutu kullanmalıyız.

İndirme işlemi tamamlandı.

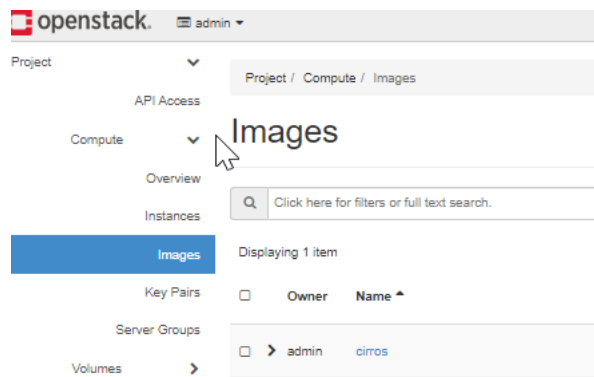
```
[root@localhost ~(keystone_admin)]# ls
1.0.3.tar.gz  anaconda-ks.cfg  keystone-admin  masscan-1.0.3  packstack-answers-20220831-183112.txt
[root@localhost ~(keystone_admin)]# pwd
/root
[root@localhost ~(keystone_admin)]# curl -o /root/cirros-0.3.4.img http://download.cirros-cloud.net/0.5.1/cirros-0.5.1-x86_64-disk.img
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left   Speed
100 273 100 273 0 0 437 0 --:--:-- --:--:-- --:--:-- 438
[root@localhost ~(keystone_admin)]# ls
1.0.3.tar.gz  cirros-0.3.4.img  masscan-1.0.3
anaconda-ks.cfg  keystone-admin  packstack-answers-20220831-183112.txt
[root@localhost ~(keystone_admin)]#
```

Openstack image create -h (-help parametresi ile neler yapabiliriz destek alabiliriz)

openstack image create --min-disk 2 --private --disk-format qcow2 --file /root/cirros-0.3.4.img cirros

```
[root@localhost ~(keystone_admin)]# openstack image create --min-disk 2 --private --disk-format qcow2 --file /root/cirros-0.3.4.img cirros
+-----+
| Field          | Value                                                                 |
+-----+-----+
| checksum       | 25c96942084f61e008d593b6c2cfda00                                   |
| container_format | bare                                                                |
| created_at     | 2022-09-04T19:52:35Z                                               |
| disk_format    | qcow2                                                              |
| file           | /v2/images/1af2d476-006e-47dc-a05a-b16fd433b055/file              |
| id             | 1af2d476-006e-47dc-a05a-b16fd433b055                             |
+-----+-----+
```

Horizondanda teyit ediyoruz.



Openstack image show “image adı” ile imajımıza ait bilgileri görebiliriz.

Openstack image delete “image adı” ile imajımızı silebiliriz

Openstack image list ile imajlarımızı görebiliriz.

```
[root@localhost ~(keystone_admin)]# openstack image list
+-----+-----+-----+
| ID          | Name    | Status |
+-----+-----+-----+
| 1af2d476-006e-47dc-a05a-b16fd433b055 | cirros  | active |
+-----+-----+-----+
[root@localhost ~(keystone_admin)]#
```

Oluşturulan imaj standart kullanıcılar göremez, bu imajları projelere atamadığımızdan proje yöneticileride bu imajlara sahip olamazlar.

## Web Frontends

### Horizon (Dashboard yönetimi)

Horizon, Nova, Swift, Keystone vb. dahil olmak üzere OpenStack servislerine web tabanlı bir kullanıcı arayüzü sağlayan OpenStack'in Kontrol Panelinin uygulamasıdır.

Şimdi sizlerle horizon üzerinden, gerçek yaşantıda karşılanacak bir senaryo üzerinden giderek uygulama yapalım.

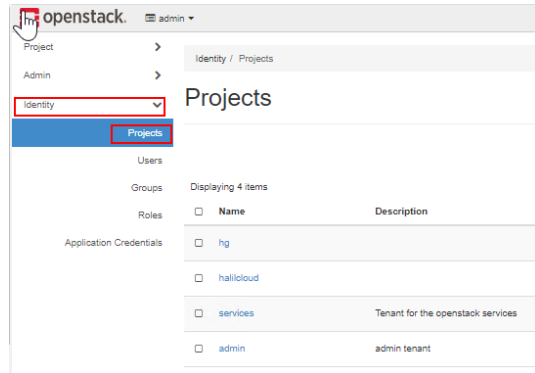
Openstack uygulaması kullanan kuruluşumuzun, yazılım departmanında bir ekibin yeni bir proje adım atması üzerine altyapı hizmetlerini sağlayalım.

### Proje / kullanıcı oluşturma

Yeni bir proje oluşturmak için openstack horizon admin yetkili bir user ile giriş yapıp identity altında project kısmında new project diyerek proje adı vererek yeni bir proje oluşturabiliriz.

İki türlü proje vardır, 1. Yönetici projesi, 2. Hizmet projesidir.

Yönetici projesi sizin openstack hizmetlerinin kaydedileceği alandır. Tüm hizmetler bu projede oluşturulur.



Yönetici projesi oluşturuyoruz.

Create Project

Project Information \* Project Members Project Groups

Domain ID: default

Domain Name: Default

Name \*: halilgoksel.com

Description:

Enabled: ☒

Cancel Create Project

Proje adı belirliyoruz.

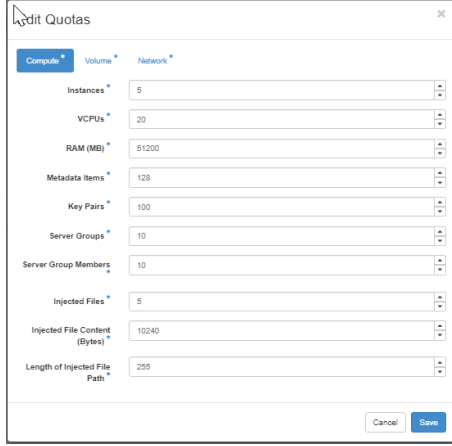
Create Project

Project Information \* Project Members Project Groups

All Users: hg, glance

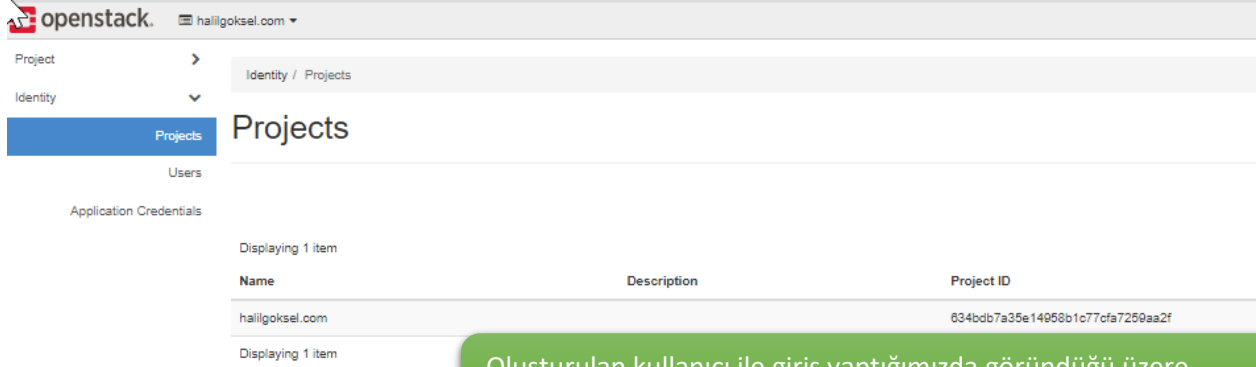
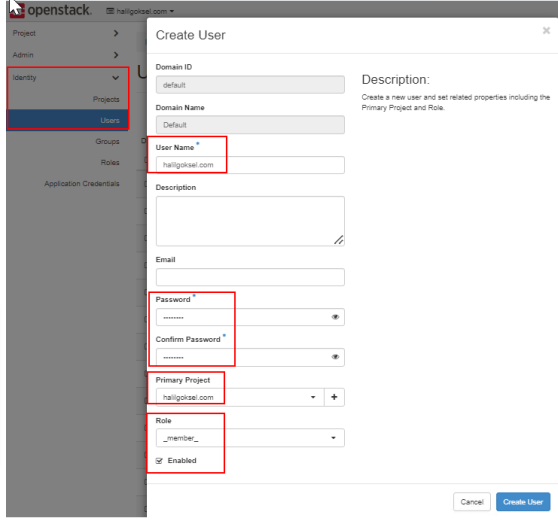
Project Members: admin (role: \_member\_)

Admin kullanıcısı, admin yetkileri ile projeye atıyoruz.



Oluşturduktan sonra set quota diyip, proje kotasını change edebiliyoruz

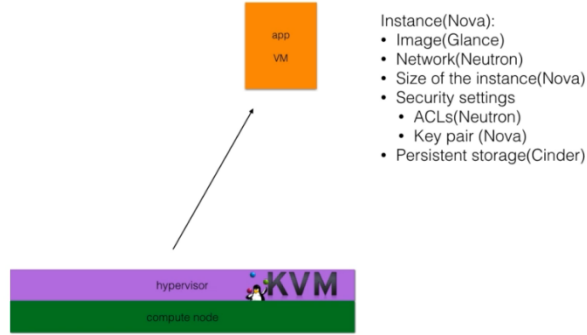
Hemen ardından bir kullanıcı oluşturup, yeni oluşturduğumuz projeye proje yöneticiliği yapacak kullanıcı olduğundan admin olarak atıyoruz.



Name	Description	Project ID
halilgoksel.com		634bdb7a35e14958b1c77cfa7259aa2f

Oluşturulan kullanıcı ile giriş yaptığımızda görüldüğü üzere, proje admini olarak sınırlı kotalar ile giriş yaptığımızı gördük.

### Launching an Instance



Openstack üzerinde Instance oluşturmak için zorunlu olan ve zorunlu olmayan API servislerindeki bunlarla beraber oluşturmanız gereklidir.

Instance (Nova): Openstack'ın bilgiyi işleyen çekirdek birimidir diyebiliriz

Image: Openstack üzerinde ISO insert edip sunucu oluşturulamaz, Glance üzerinde imajlar yaratıp sunucu ayağa kaldırabiliriz.

Network: Instancemizin, internet yada

intranet iletişim için network yapılandırılmalarının yapılacağı servistir. Key pair servisi, uzaktan yönetebileceğimiz servistir, Çift anahtar gereklidir.

Cinder: Instancemizin, kalıcı depolama sağladığı servistir.

Gereksinimleri kategorize edecek olursak instance oluşturmak için, Glance, Neutron, Nova servislerini kullanmalıyız. Diğer keypair ve Cinder isteğe bağlıdır. Elbette işlevsel hale getirmek için tüm servislere ihtiyaç duyulmaktadır.

## Image Oluşturma

Oluşturduğumuz projede bir imaj oluşturmak istiyoruz.

Aşağıdaki gibi, projemize gelip images create diyoruz, imajımızın adını belirtiyoruz, image dosyamızı upload etmek için brows diyoruz, Formatımızı seçiyoruz, minimum memory ve disk veriyoruz, Visibility alanında private yaparsak oluşturulan imaj sadece bizim projemizde yer alacaktır, Create diyerek devam ediyoruz.

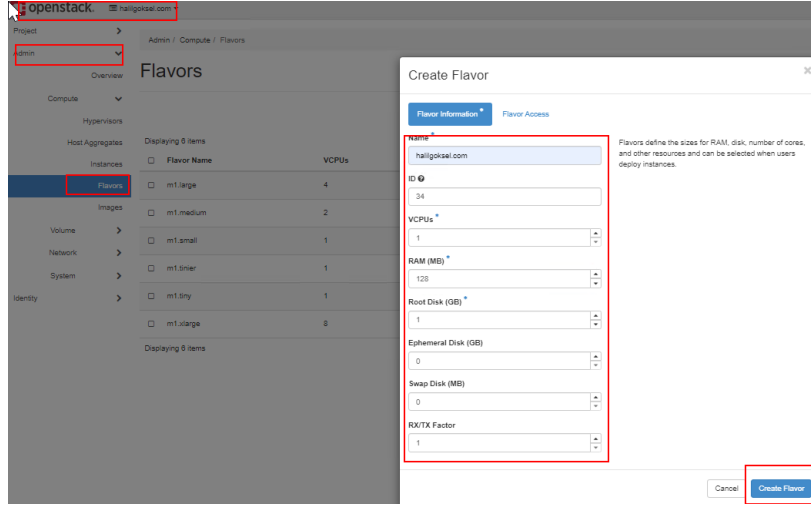
The screenshot shows the OpenStack dashboard with the 'Create Image' dialog box open. The dialog is titled 'Create Image' and has a 'Metadata' tab selected. The 'Image Name' field is set to 'cirros'. The 'Image Source' is set to 'File' with a 'Browse...' button. The 'Format' is set to 'QCOW2 - QEMU Emulator'. The 'Image Requirements' section shows 'Kernel' as 'Choose an image', 'Architecture' as 'Choose an image', 'Minimum Disk (GB)' as '1', and 'Minimum RAM (MB)' as '128'. The 'Image Sharing' section shows 'Visibility' as 'Private' and 'Protected' as 'Yes'. The 'Create Image' button is at the bottom right.

Admin useri giriş yaptığımızda imaj sahibini görebiliriz

The screenshot shows the OpenStack dashboard with the 'Images' tab selected. The list shows one item: 'cirros' with owner 'halilgoksel.com'. The 'Images' tab is highlighted in blue.

## Flavor Oluşturma

Flavors özetçe kaynakların önceden tanımlandığı bir çeşit yapılandırma servisi diyebiliriz.



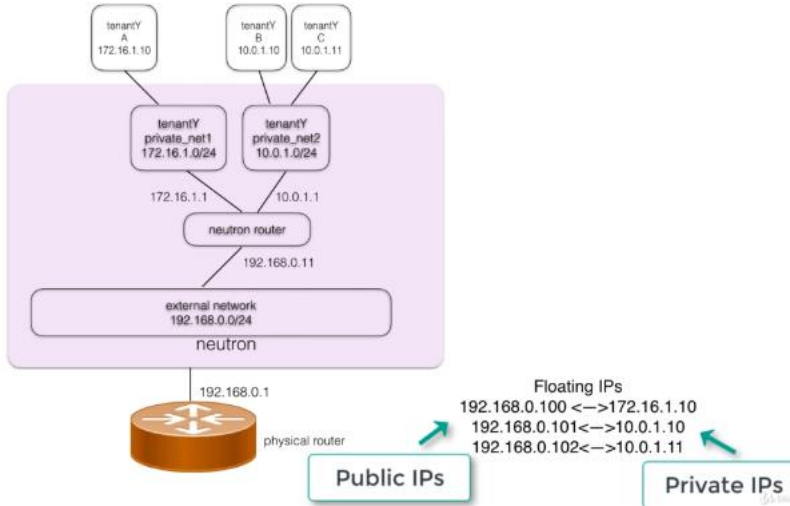
Admin kullanıcısı ile flavors oluşturabiliriz.

Yeni bir flavors oluşturup verilebilecek kaynak değerlerini giriyoruz.

Create diyerek flavors oluşturabiliriz.

## Instance Connectivity

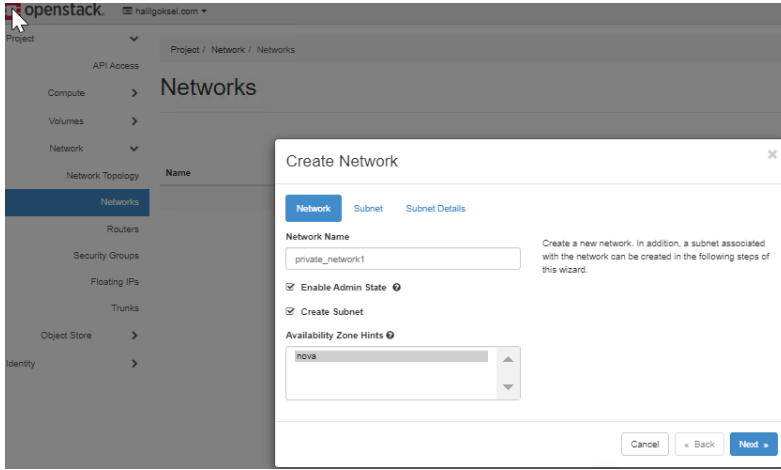
### Instance Connectivity



Şemada yer verildiği üzere 2 farklı vlan  
3 farklı NIC 'de network topolojisi belirtilmiş.

3 Farklı IP'nin NAT yapılmış  
neutron'dan geçerek public adreslerine  
doğru trafiği çıkışı görmekteyiz.

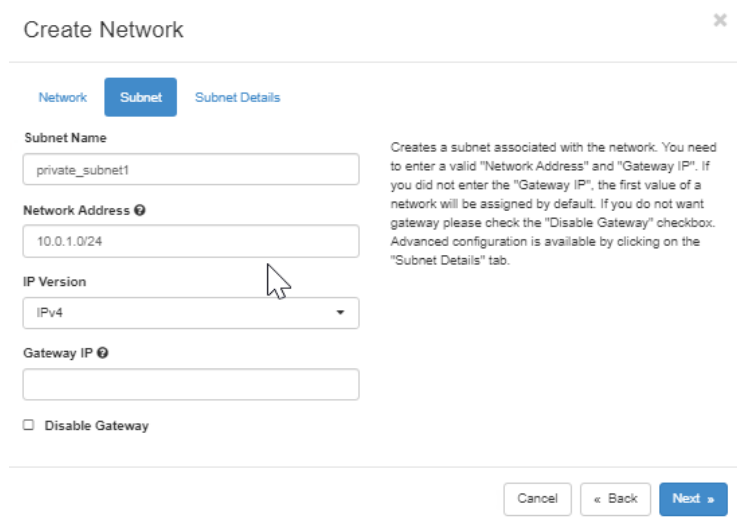
## Network



### Network Oluşturma

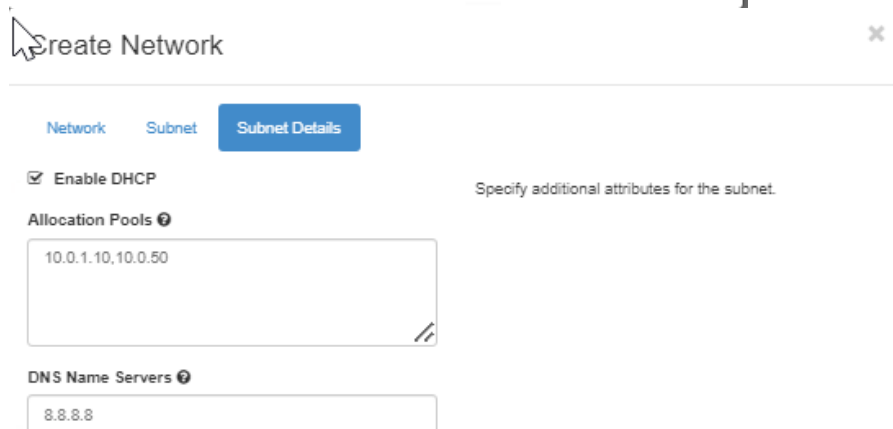
Projemize member kullanıcımızla giriş yapıp network tabında create network diyoruz.

Network adını belirtiyoruz, subnet kısmına geçiyoruz.



Subnet alanında subnet name veriyoruz, Subnet blogunu ekliyoruz, Gateway eklemek zorun değildir auto olarak tanımlanır.

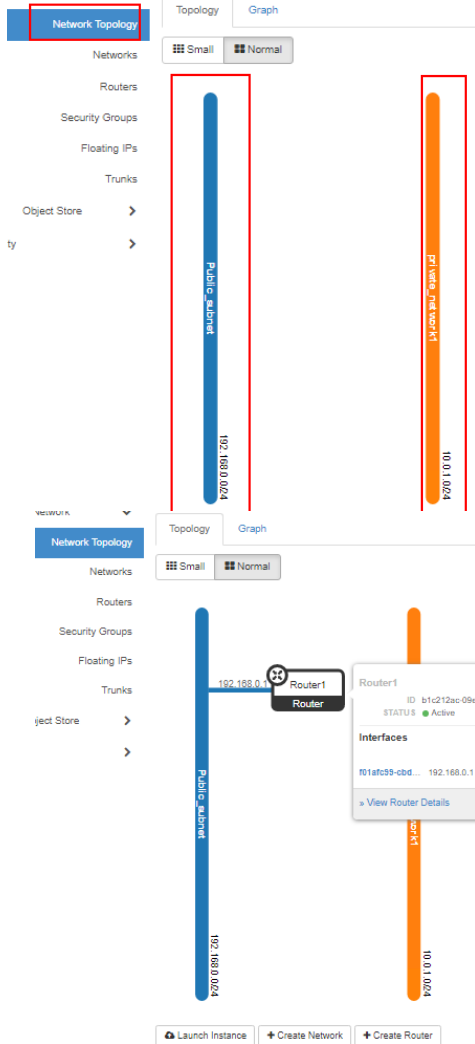
Subnet details kısmında DHCP, DNS ve Route düzenlemelerini yapabiliriz, default olarak DHCP aktiftir, belirli aralıkta dağıtım istiyorsan set edebiliriz.





Create diyoruz, statusunu ve içeriğini görebiliriz.

Networks	Name	Subnets Associated	Shared	External	Status	Admin State
Routers	Public_subnet	Public_Subnet 192.168.0.0/24	No	No	Active	UP
Security Groups	private_network1	private_subnet1 10.0.1.0/24	No	No	Active	UP



İnternete çıkış için ömevcut ağımız ile public networkmu ile etkileşim halinde olmasını sağlamalıyız.

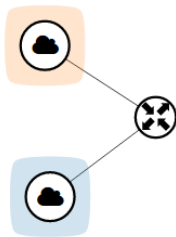
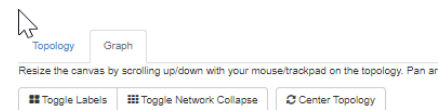
Sağ tarafta görüldüğü üzere bir bağ yok, aralarında bağlantıyı sağlamak adına create add router diyerek router oluşturabiliriz.

Router'ımıza isim veriyoruz, Create diyoruz, Daha sonra public ağımıza bağlamak için router tabında interface ekleyerek bağlayabiliriz.

Public ağımızı route bağlı şekilde geldi, Bu sefer'de internete çıkacak vlanımızı bağlamak için interace add diyoruz.

IP adresini istersek özelleştirebiliriz.

Graph olarak' da topolojimizi görebiliriz.



## Security Group

Vlanlar arası bağlantımızı oluşturduk şimdide firewall izinlerini geldi.

Routers tabının altından security group sekmesine geliyoruz. Default olarak bizi 1 security group 4 firewall rule karşılamaka.

Egress çıkan trafik, İngres giren trafik kontrolüdür.

Default olarak bu kural any any verilmiş görülmekte, Rempte IP prefix 0.0.0.0/0 anlamı tüm subneti kapsamaktadır.

Direction	Ether Type	IP Protocol	Port Range	Remote IP Prefix	Remote Security Group	Description	Actions
Egress	IPv4	Any	Any	0.0.0.0/0	-	-	Delete Rule
Egress	IPv6	Any	Any	::/0	-	-	Delete Rule
Ingress	IPv4	Any	Any	-	default	-	Delete Rule
Ingress	IPv6	Any	Any	-	default	-	Delete Rule

Securty group create edip, bir isim veriyoruz, ardından oluşturduğumuz gruba rule ekliyoruz

Rule Name: securitygroup1  
Description: ALL ICMP  
Direction: Ingress  
Remote: CIDR  
CIDR: 0.0.0.0/0  
Add

Görselde yer verildiği gibi securitygroup1 olarak oluşturduk.

Ardından add rule diyerek bir incoming ICMP protokü ekledir, Burada birçok port bilgisi mevcuttur.

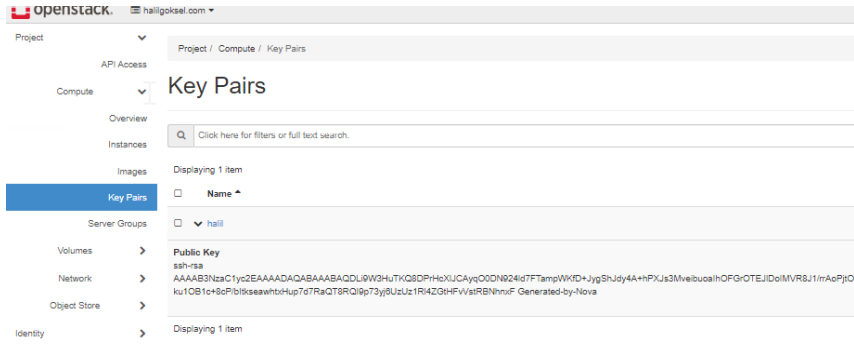
CIDR diyerek tüm vlana kapsayacak şekilde rule girebiliriz yada vlanda belirtebiliriz. Add diyerek kaydettik.

Aynı şekilde incoming trafik için SSH ingress rule girdik

<input type="checkbox"/>	Ingress	IPv4	ICMP	Any	0.0.0.0/0
<input type="checkbox"/>	Ingress	IPv4	TCP	22 (SSH)	0.0.0.0/0

## Key Pairs

Daha sonra instance uzaktan erişim için Key pairs oluşturmak gereklidir.



Daha sonra network tabında Floating IPs tanımlamamız gerecektir. Yani external IP'miz. Biz daha önceden yapılandırma yapmadığımızdan bu alanı es geçiyorum.

Herşeyimizi tamamladık şimdi sıra instance yaratmakta geldi.

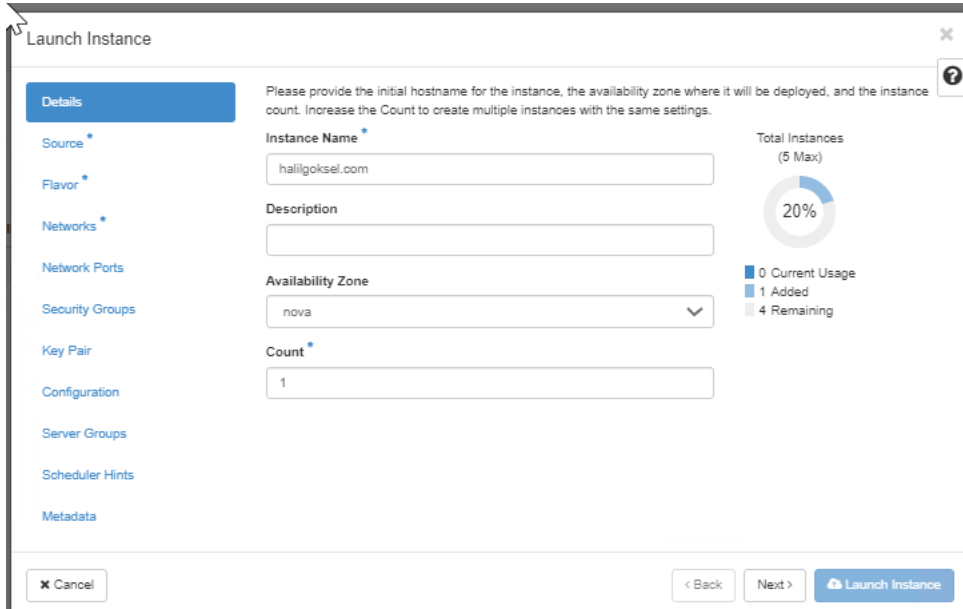
## Instance Oluşturma Part 1

Launch instance diyerek işlemi başlatıyoruz.

Instance Name kısmına instance bir isim veriyoruz.

Avability zone seçebiliriz fakat bizim birtane zone'umuz olduğu için böyle devam ediyoruz.

Count birden fazla instance oluşturmak için kullanabiliriz.



Launch Instance

Details

Source

Flavor \*

Networks \*

Network Ports

Security Groups

Key Pair

Configuration

Server Groups

Scheduler Hints

Metadata

Instance source is the template used to create an instance. You can use an image, a snapshot of an instance (image snapshot), a volume or a volume snapshot (if enabled). You can also choose to use persistent storage by creating a new volume.

Select Boot Source

Image

Create New Volume

Yes No

Delete Volume on Instance Delete

Yes No

Volume Size (GB) \*

1

Allocated

Name	Updated	Size	Type	Visibility
> cirros	9/7/22 3:35 PM	15.58 MB	qcow2	Private

Available 0

Select one

Q Click here for filters or full text search.

Name	Updated	Size	Type	Visibility
No available items				

Cancel

< Back Next > Launch Instance

Buradan Boot source cirros imajımızı kullanacağımız için seçiyoruz.

Avaiable kısmında kullanabileceğimiz imajlarımız yer alıyor, ok işaretleri ile aallocated alanına taşıyoruz.

Kalıcı birim yaratacağımız için create new volume yes demeliyiz, Volume size Cinder içindir, Flavor farklıdır, Burada Cinder size belirtiyoruz.

Hemen ardından flavor seçimini yapıyoruz.

Launch Instance

Details

Source

Flavor

Networks \*

Network Ports

Security Groups

Key Pair

Configuration

Server Groups

Scheduler Hints

Metadata

Flavors manage the sizing for the compute, memory and storage capacity of the instance.

Allocated

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
> halligoksel.com	1	128 MB	1 GB	1 GB	0 GB	Yes

Available 0

Select one

Q Click here for filters or full text search.

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
> m1.tinier	1	256 MB	2 GB	2 GB	0 GB	Yes
> m1.tiny	1	512 MB	1 GB	1 GB	0 GB	Yes
> m1.small	1	2 GB	20 GB	20 GB	0 GB	Yes
> m1.medium	2	4 GB	40 GB	40 GB	0 GB	Yes
> m1.large	4	8 GB	80 GB	80 GB	0 GB	Yes
> m1.xlarge	8	16 GB	160 GB	160 GB	0 GB	Yes

Cancel

< Back Next > Launch Instance

Network kısmında instance ait network seçiyoruz.

The screenshot shows the 'Launch Instance' dialog box with the 'Networks' tab selected. The left sidebar contains links for Details, Source, Flavor, Networks (selected), Network Ports, Security Groups, Key Pair, Configuration, Server Groups, Scheduler Hints, and Metadata. The main content area has a header 'Networks provide the communication channels for instances in the cloud.' and a sub-header 'Select networks from those listed below.' Below this, there are two sections: 'Allocated' and 'Available'. The 'Allocated' section shows a table with columns: Network, Subnets Associated, Shared, Admin State, and Status. It contains one entry: 'private\_network1' with 'private\_subnet1' subnets, 'No' shared, 'Up' admin state, and 'Active' status. The 'Available' section has a search bar and a table with the same columns. It contains one entry: 'Public\_subnet' with 'Public\_Subnet' subnets, 'No' shared, 'Up' admin state, and 'Active' status. At the bottom, there are buttons for 'Cancel', '< Back', 'Next >', and 'Launch Instance'.

Network	Subnets Associated	Shared	Admin State	Status
private_network1	private_subnet1	No	Up	Active

Network	Subnets Associated	Shared	Admin State	Status
Public_subnet	Public_Subnet	No	Up	Active

Network port extra birbağlantı noktasına ihtiyacımız olmadığından atlıyoruz.

Secury group'dan oluşturduğumuz security group seçiyoruz.

The screenshot shows the 'Launch Instance' dialog box with the 'Security Groups' tab selected. The left sidebar contains links for Details, Source, Flavor, Networks, Network Ports, Security Groups (selected), Key Pair, Configuration, Server Groups, Scheduler Hints, and Metadata. The main content area has a header 'Select the security groups to launch the instance in.' and a sub-header 'Select one or more'. Below this, there are two sections: 'Allocated' and 'Available'. The 'Allocated' section shows a table with columns: Name and Description. It contains two entries: 'default' with 'Default security group' and 'securitygroup1' with 'Default security group'. The 'Available' section has a search bar and a table with the same columns. It contains one entry: 'No available items'. At the bottom, there are buttons for 'Cancel', '< Back', 'Next >', and 'Launch Instance'.

Name	Description
default	Default security group
securitygroup1	Default security group

Name	Description
No available items	

Key Pair kısmında sunucumuza dışarıdan güvenli bir şekilde erişmek için oluşturduğumuz Key pair seçiyoruz ve Launch Instance ile devam ediyoruz.

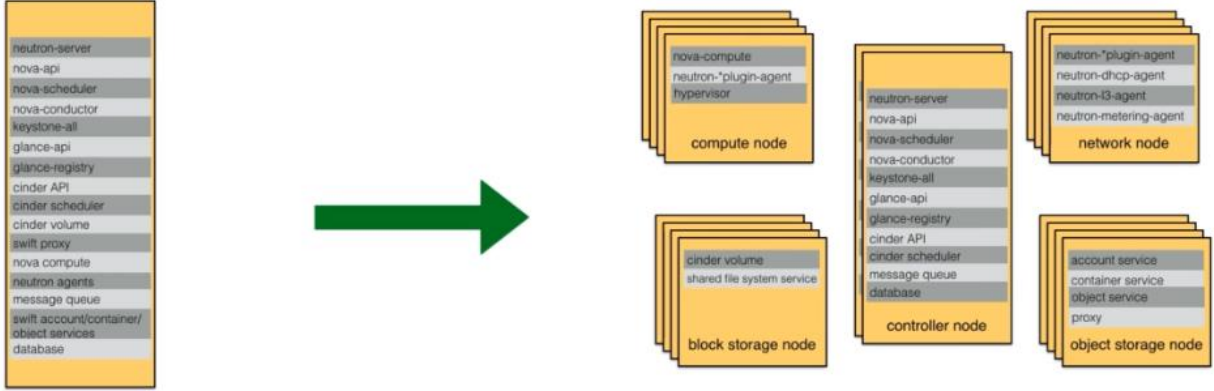


Bir süre sonra instance oluştuğunu görüyoruz. Action tabında instance üzerinde Instance düzenleyebilir, NIC ekleyebilir, Floating IP ekleyebilir, suspend, shutdown, snapshot gibi işlemler yapabiliriz.

## Openstack Ölçeklendirme

### Nova Node

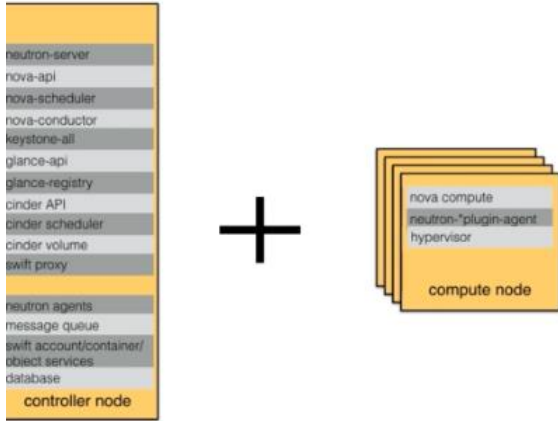
## Deployment Scale



Openstack dağıtımları ölçek olarak farklılık gösterir, Örneklerle yaptığımız gibi hepsi tek bir düğüm üzerinde çalışabilir yada onbinlerde node kadar çıkabilir.

Hangi servisleri nerede çalıştırmayalım sorusuna ölçeklendirmede sizler topolojiye uygun karar verebilirsiniz, bunu yapmakta özgürsünüz.

## Compute Node



Openstack bulutumda daha fazla instance çalıştırabilmek istiyorum

Bunun için üzerinden hypvizör yüklü daha fazla makineye yani compute node'a ihtiyacım olacak.

Onların bulutumun bir parçacı olmasını istiyorum.

Bunun için compute node'larda ek düğümlerin çalışıyor olması gereklidir.

Neutron ajanları, Nova işlemcisi, hipervizör çalıştırır.

# Storage for Instances

Non-compute node based shared file system:

- Disks storing the running instances are hosted in servers outside of the compute nodes
- Compute hosts are stateless. In case of node failure, instances are easily recoverable

On compute node storage—shared file system

- Each compute node is specified with a significant amount of disk space
- A distributed file system ties the disks from each compute node into a single mount
- Data locality is lost
- Complicated instance recovery

On compute node storage—nonshared file system

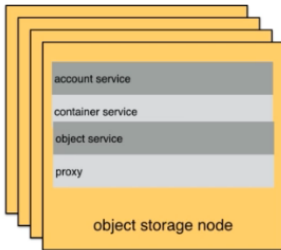
- Local disks on compute nodes used
- Heavy I/O usage on one compute node does not affect instances on other compute nodes
- When you lose a node, data on it gets lost as well
- Instance migration is complicated

Instance'lar için depolama alanı sağlamaya yönelik yaklaşım, Nova düğümü tabanlı olmayan paylaşımlı sistemler olarak adlandırılır. Yani hypervizor altında verileri depolayan diskler, işlem düğümleri olmayan sunucularda barındırılır.

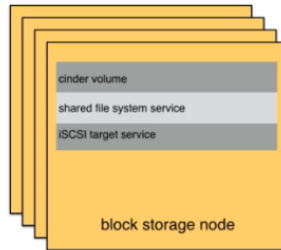
Sonuç olarak işlem ve depolama konakları ayrıdır. Compute node'lar için herhan bir bakım gerektiğinde başka bir düğüme geçirerek bakım için çevrim dışı moduna (vmware: maintance mode) getirebiliriz. İkinci depolama yaklaşımı, paylaşılan dosya sistemine sahip hesapla düğümü depolamasıdır. Bu seçenekte her bir hesaplama düğümü önemli miktarda disk alanıyla belirtilir, ancak dağıtılmış dosya sistemine bağlanır, Bu hesaplama düğümü diskleri tek bir bağlama dönüştürür.

Bu seçneğin temel avantajı diskleri tek bir bağlama dönüştürür, ek gereksinip duyduğunda depolama ölçeklendirilebilir.

## Storage Node



tek bir depolama hizmetini çalıştırır.



Depolama düğümleri, depolama hizmetlerini çalıştırır

Depolama düğümleri, volume snapshot'larda dağıl olmak üzere ortam için gereken tüm verileri üzerinde barındırır.

İmage servisi, blok depolama, nesne depolama ve paylaşılan dosya depolamayı işler.

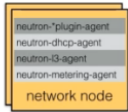
Sağ tarafda görüldüğü gibi Genellikle bir depolama düğümü



Glance, bir nesne depolama düğümündeki görüntüler için, depolama hizmeti sağlayan düğümde çalıştırılmalıdır.

## Network Node

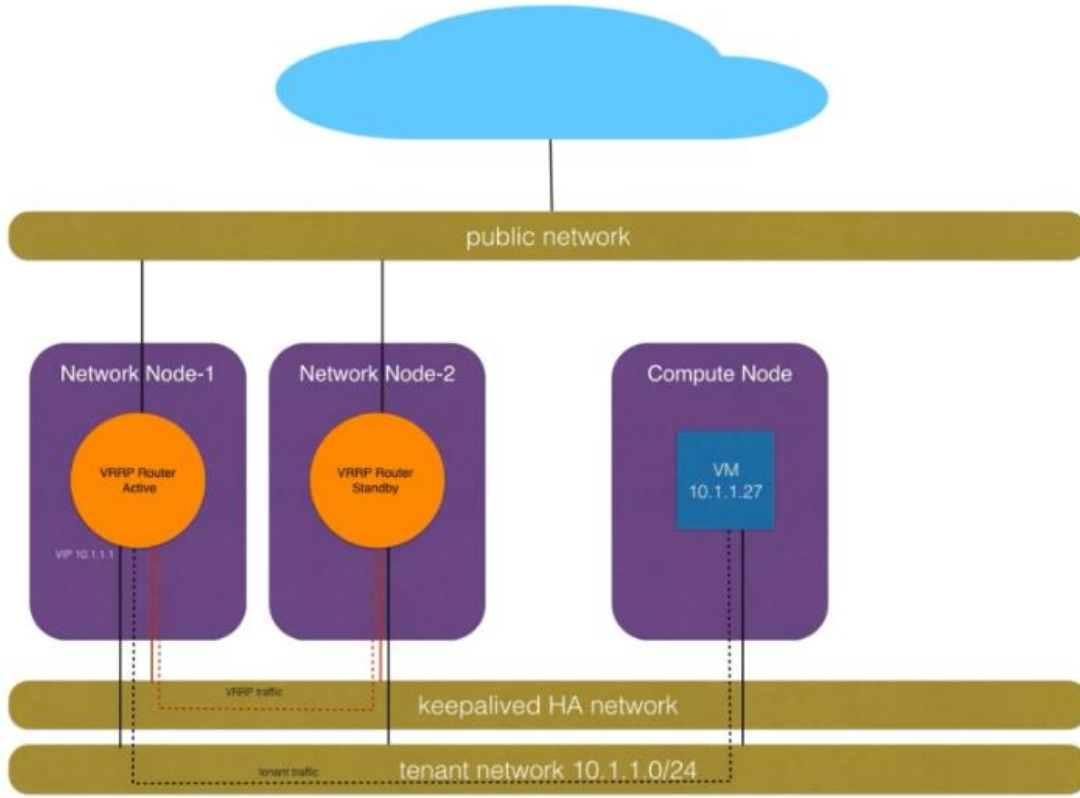
# Network Node



- Handles virtual networking needs of cloud consumers
  - Routing
  - NAT/Floating IPs
  - Creates logical routers as instance gateway
  - Creates and manages namespaces
- L4-L7 advanced services
  - Load balancer as a service
  - Firewall as a service
- DVR can offload Network Node

Network Node’umuz Bulutumuzdan neutronun 3. Katman(network) ve yönlendirme ağı gibi hizmetlerini sağladığı yerdir. Network ve route haricinde hizmet olarakda yük dengeleyici servisleride bu düğümde çalışmaktadır.

# L3 Agent HA



Ağ düğümünde boşaltmak istersen dağıtılmış sana modunda kullanabiliriz. Bu durumda ağ düğümü görevlerini diğer ağ node'larına aktaracaktır.

Bu şekilde bir ölçeklendirmede yedeklilik sağlamamız gereklidir.

İki düğüm arasında sanal yönlendiriciler VRRP protokolünü kullanır.

Bu şekilde bir ağ düğümünde en az iki NIC gereklidir.

1. NIC tüm düğümlere ve internete ulaşmak için ağ geçiği olarak kullanır, 2. NIC ise, Paket yükleme, güvenlik güncelleştirmeleri gibi node yönetimi için kullanılır. Performan gereksinip yada güvenlik politikalarına göre trafiği dahada segmentlere ayırabiliriz, Trafik modellerine bağlı daha fazla NIC'de kullanılabilir.

## Controller Node

### Controller Node



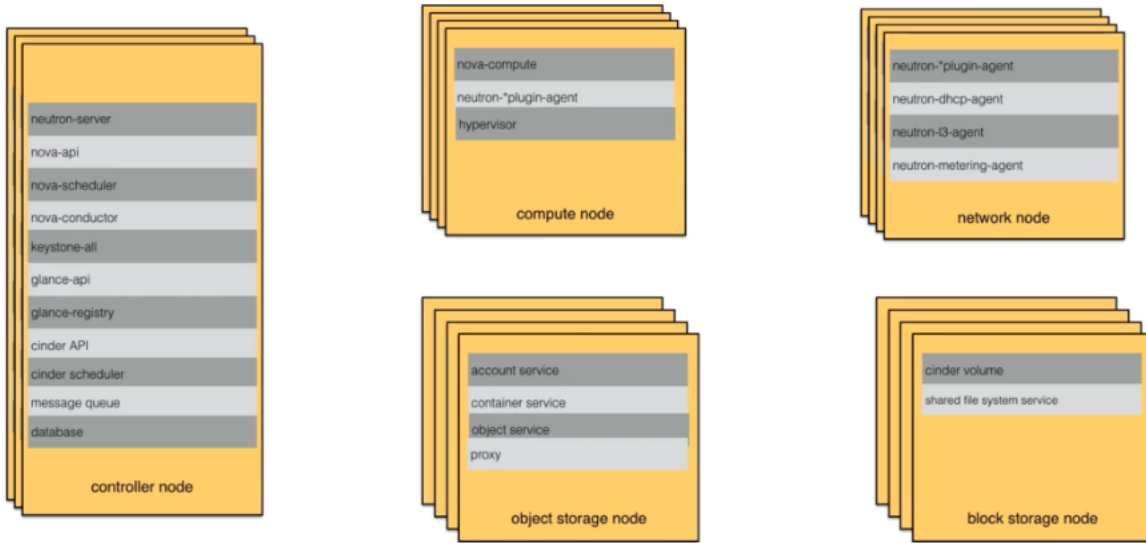
Farklı düğümlerde çalışacak bileşenler ana düğümümüzde ayrılır.

Bu Ana düğüm controller node olarak geçmektedir. İnsanların eriştiği ön kapı ve diğer tüm hizmet bileşenleri controller node üzerinden geçer ve diğer düğümlere API servislerinde ulaşır.

Tüm bu hizmetler, bir dinleme servisi sağlar, HTTP üzerinden iletişime geçerler. Glance API, Sender API, Nova API veya Neutron API gibi API uç noktası denetleyici düğümünde çalışır. Bir istek gönderdiğimizde, bize cevapla yanıt verirler. Ya evet, istenen işlemi gerçekleştirdim ya da belirli bir nedenden dolayı yerine getiremiyor. Yerine getirilip getirilmediğine dair bir yanıt alacağız. Genellikle denetleyici düğümü bu rest API uç noktalarını tutar ve isteği verilen kişiye iletir.

## Multi Node Tasarımı

# Multi Node Deployment



Şimdiye kadar projemizde oluşturduğumuz tek düğüm(node) üzerinde, yapılandırmalar ve controller test etmek için kullanırken, Üretici ortamlarda mimari ve tasarım multi node'lar üzerinde yapılandırılmalıdır.

Multi node'lar ölçeklendirebilirliğin yanı high availability ve performans imkanı sağlar.

Her bir hizmet ayrıştırılır ve hizmetler arasındaki tüm iletişim dinlenen API servisler aracılığı ile yapılmaktadır.

Avantajları, multi node tasarımı konusunda muazzam bir esnekliğe sahip olmasıdır. Ön görülen bir dağıtım modeline bağlı değiliz.

Tek bir controller üzerinden node'larımızı aktif yada pasif bir yapılandırma oluşturabiliriz.

Note:Pacemaker kullanarak linux platformu için HA ve LB işlevlerini yürütebiliriz. Openstack özgü bir servis değildir.

[Pacemaker](#) küme yığını, Linux platformu için son teknoloji ürünü bir yüksek kullanılabilirlik ve yük dengeleme yığıdır. Pacemaker, OpenStack altyapısını yüksek oranda kullanılabilir hale getirmek için kullanılır

## Node Requirements



Bulut servisi ortamına bağlı olarak genelde 4 lü Node yapılandırmanın olduğunu görürüz.

Kullanım durumlarında, düğümlerin donanım özellikleri disk sayısı ve ram miktarı değişkenlik gösterebiliriz.

Grafikteki görselde düğümler için minimum miktarı ele alabiliriz.

## Openstack Genişletme

### Node Ekleme

#### 1. Denetleyici düğümündeki yanıt dosyasını değiştirme

Denetleyici düğümünde kök olarak oturum açın, mevcut **answers.txt** dosyasını yedekleyin:

```
[root@controller ~]# cp /root/answers.txt /root/answers.txt.backup
```

Mevcut **answers.txt** dosyasında aşağıdaki parametreleri aşağıdaki gibi görünecek şekilde değiştirin:

```
[root@controller ~]# vim /root/answers.txt
```

```
EXCLUDE_SERVERS=192.168.2.4,192.168.2.5
```

```
CONFIG_COMPUTE_HOSTS=192.168.2.5,192.168.2.8
```

Not: **Mevcut düğümlerin yanlışlıkla yeniden yüklenmesini önlemek** için **EXCLUDE\_SERVERS** parametresinde doğru IP'leri ayarladığınızdan emin olun!

İşte **answer.txt** dosyası, yeni **Compute0** düğümü eklerken kullandık.

#### 2. OpenStack dağıtımı için yeni düğüm hazırlayın

Yeni donanım hazırlayın:

– CentOS7 64bit'i yeni donanıma

yükleyin – NetworkManager hizmetini  
devre dışı bırakın ve durdurun – RDO'dan openstack/juno deposunu yükleyin

**Not: Openstack dağıtımı için düğümün nasıl hazırlanacağı hakkında ayrıntılar için OpenStack Juno'yu CentOS 7 / RHEL 7'ye yükleme makalesini kontrol edin.**

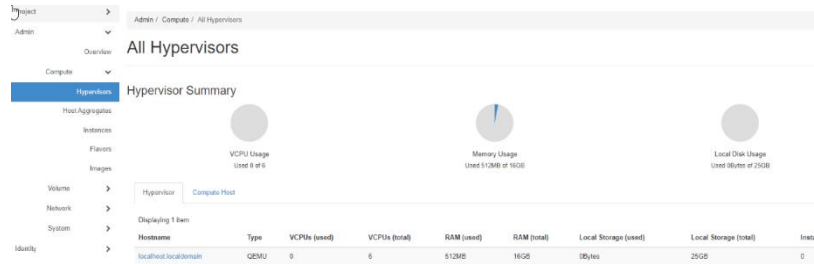
### 3. Mevcut OpenStack'e yeni düğüm ekleyin

Packstack yükleyici betiğini kullanarak yeni **Compute0** düğümü ekleyin:

```
root@controller ~]# packstack --answer-file=/root/answers.txt
```

OpenStack'in yeni düğümüne dağıtımı biraz zaman alabilir:

Eklediğimiz node openstack üzerinde hypervisor'de görünecektir



Eklediğimiz ve üzerinde hangi API servisinin bulunduğu node'ları görmek için system informan kısmında multi node üzerine yapılandırılmış openstack sağlayacağımızın node bilgileri yer verilecektir.

## Openstack Logs

### Logs in Openstack

- OpenStack services use the standard logging levels, at increasing severity: TRACE, DEBUG, INFO, AUDIT, WARNING, ERROR, and CRITICAL
- To enable logs:
  - For Neutron, edit `/etc/neutron/neutron.conf` → `debug=true`
  - For Nova, edit `/etc/nova/nova.conf` → `debug=true`
  - For Keystone, edit the `/etc/keystone/logging.conf` file and look at the `logger_root` and `handler_file` sections
  - For Horizon, edit `/etc/openstack_dashboard/local_settings.py`.
  - For Cinder, edit the configuration file on each node with Cinder role.
  - For Glance, edit two configuration files: `/etc/glance/glance-api.conf` and `/etc/glance/glance-registry.conf`

Openstack servisi, standart loglama düzeyi haricinde, Trace, Debug, Info, Audit, Warning, Error ve Critical günlük kaydını yapılandırdığınızda iletiler, sadece bilirdiğiniz iletiler eşliğindeyse görünür.

Neutron için hata ayıklama düzeyi günlük etkinleştirme üzere `etc/neutron/neutron.conf` dosyasını `vi` ile editleyip `debug=true` ile netron servislerinde loglama işlemi gerçekleştir.

```
# If set to true, the logging level will be set to DEBUG instead of the default
# INFO level. (boolean value)
# Note: This option can be changed without restarting.
#debug = false
debug=True
```

Log aramak içinde `grep` komutlarıyla hata kaynaklanan api log dosyalarında aratabiliriz.

```
a559-6575-48d1-9b95-907045d1fadc.
[root@localhost ~]# tail -f /var/log/nova/* |grep -i error
```

Nova içinde aynı şekilde `/etc/nova/nova.conf` içerisinde yer alır.

Keystone biraz daha farklı şekilde ele alır, günlük düzeyini değiştirmek için keystone günlük dosyasını düzenlememiz ve günlüğe bakmamız gerekir.

Horizon için günlük yapılandırma dosyası `/etc/openstack_dashboard/local_settings.py` içerisinde yer alır.

Diğer servislerde görüntüde yer verilmiştir.

Aşağıdaki görüldüğü üzere log'ların yazıldığı dosya sistemleri aşağıdaki dizinlerdeki gibidir.

## Log Files

Node type	Service	Log location
Cloud controller	nova-*	/var/log/nova
Cloud controller	glance-*	/var/log/glance
Cloud controller	cinder-*	/var/log/cinder
Cloud controller	keystone-*	/var/log/keystone
Cloud controller	neutron-*	/var/log/neutron
Cloud controller	horizon	/var/log/apache2/
All nodes	misc (swift, dnsmasq)	/var/log/syslog
Compute nodes	libvirt	/var/log/libvirt/libvirtd.log
Compute nodes	Console (boot up messages) for VM instances:	/var/lib/nova/instances/instance-<instance id>/console.log
Block Storage nodes	cinder-volume	/var/log/cinder/cinder-volume.log

## Openstack – CLI

Not:

- 1.Openstack CLI yönetiminde horizon'dan erişim sağlayamayız.
- 2.İşletim sistemi (Centos7 üzerinde koşuyor) ile CLI arayüzü birbiri ile karıştırmamalıdır. İşlem sistemi kimlik bilgileri openstack için birşey ifade etmez, Sadece OS üzerinde koşan bir servistir.
- 3.CLI'dan kullandığımız komutlar openstack'de API çağrılarına çevirir.

Openstack CLI geçiş için openstack sunucusuna bağlanıyoruz,

Source keystonerc\_admin ile, admin yetkileriyle CLI oturumunu açıyoruz.

Horizon'da yaptığımız tüm işlemleri CLI'dan yürütebiliriz, dökümandan faydalanabilirsiniz.

[“ https://docs.openstack.org/python-openstackclient/pike/cli/command-list.html”](https://docs.openstack.org/python-openstackclient/pike/cli/command-list.html)

## Openstack – CLI Komutları

**source keystonerc\_admin** / Openstack CLI geçiş komutudur.

more keystonerc\_admin / Openstack erişim bilgileri

openstack endpoint list / Openstack API servisleri erişim bilgileri (URL, Servis adı, Servis portu)

openstack command list / (openstack CLI yönetimi için komutları listeler)

openstack command list | grep openstack.image -A 30 (imaj komutlarını listeler)

openstack endpoint list / API servislerini listeler

openstack endpoint show “service id” / API servisi ile ilgili bilgileri listeler

openstack project create “proje adı” / Proje oluşturur

openstack project show “proje adı” / proje hakkında bilgi verir

openstack user create –project “proje adı” –password-prompt “kullanıcı adı” / bir kullanıcı oluşturur ve belirlediğimiz projeye tanımlanır.

Openstack role list

Openstack role assignment list –project “proje adı” –user “kullanıcı adı”



Openstack image list (openstack bulutumuzdaki imajları listelemek için kullandığımız komut)

Openstack console url show --novnc "instance" / web console'dan erişmek için

## Command List Link

UPDATED: 2019-09-09 16:24

- [access token](#)
  - [access token create](#)
- [address scope](#)
  - [address scope create](#)
  - [address scope delete](#)
  - [address scope list](#)
  - [address scope set](#)
  - [address scope show](#)
- [aggregate](#)
  - [aggregate add host](#)
  - [aggregate create](#)
  - [aggregate delete](#)
  - [aggregate list](#)
  - [aggregate remove host](#)
  - [aggregate set](#)
  - [aggregate show](#)
  - [aggregate unset](#)
- [availability zone](#)
  - [availability zone list](#)
- [backup](#)
  - [backup create](#)
  - [backup delete](#)
  - [backup list](#)
  - [backup restore](#)
  - [backup show](#)
- [catalog](#)
  - [catalog list](#)
  - [catalog show](#)
- [command](#)
  - [command list](#)
- [complete](#)
  - [complete](#)
- [compute agent](#)
  - [compute agent create](#)
  - [compute agent delete](#)

- [compute agent list](#)
  - [compute agent set](#)
- [compute service](#)
  - [compute service delete](#)
  - [compute service list](#)
  - [compute service set](#)
- [configuration](#)
  - [configuration show](#)
- [consistency group](#)
  - [consistency group add volume](#)
  - [consistency group create](#)
  - [consistency group delete](#)
  - [consistency group list](#)
  - [consistency group remove volume](#)
  - [consistency group set](#)
  - [consistency group show](#)
- [consistency group snapshot](#)
  - [consistency group snapshot create](#)
  - [consistency group snapshot delete](#)
  - [consistency group snapshot list](#)
  - [consistency group snapshot show](#)
- [console log](#)
  - [console log show](#)
- [console url](#)
  - [console url show](#)
- [consumer](#)
  - [consumer create](#)
  - [consumer delete](#)
  - [consumer list](#)
  - [consumer set](#)
  - [consumer show](#)
- [container](#)
  - [container create](#)
  - [container delete](#)
  - [container list](#)
  - [container save](#)
  - [container set](#)
  - [container show](#)
  - [container unset](#)
- [credential](#)
  - [credential create](#)
  - [credential delete](#)
  - [credential list](#)
  - [credential set](#)
  - [credential show](#)
- [domain](#)

- [domain create](#)
  - [domain delete](#)
  - [domain list](#)
  - [domain set](#)
  - [domain show](#)
- [ec2 credentials](#)
  - [ec2 credentials create](#)
  - [ec2 credentials delete](#)
  - [ec2 credentials list](#)
  - [ec2 credentials show](#)
- [endpoint](#)
  - [endpoint create](#)
  - [endpoint delete](#)
  - [endpoint list](#)
  - [endpoint set](#)
  - [endpoint show](#)
- [extension](#)
  - [extension list](#)
  - [extension show](#)
- [federation protocol](#)
  - [federation protocol create](#)
  - [federation protocol delete](#)
  - [federation protocol list](#)
  - [federation protocol set](#)
  - [federation protocol show](#)
- [flavor](#)
  - [flavor create](#)
  - [flavor delete](#)
  - [flavor list](#)
  - [flavor set](#)
  - [flavor show](#)
  - [flavor unset](#)
- [floating ip](#)
  - [floating ip create](#)
  - [floating ip delete](#)
  - [floating ip list](#)
  - [floating ip set](#)
  - [floating ip show](#)
  - [floating ip unset](#)
- [floating ip pool](#)
  - [floating ip pool list](#)
- [group](#)
  - [group add user](#)
  - [group contains user](#)
  - [group create](#)
  - [group delete](#)

- [group list](#)
  - [group remove user](#)
  - [group set](#)
  - [group show](#)
- [host](#)
  - [host list](#)
  - [host set](#)
  - [host show](#)
- [hypervisor](#)
  - [hypervisor list](#)
  - [hypervisor show](#)
- [hypervisor stats](#)
  - [hypervisor stats show](#)
- [identity provider](#)
  - [identity provider create](#)
  - [identity provider delete](#)
  - [identity provider list](#)
  - [identity provider set](#)
  - [identity provider show](#)
- [image](#)
  - [image add project](#)
  - [image create](#)
  - [image delete](#)
  - [image list](#)
  - [image remove project](#)
  - [image save](#)
  - [image set](#)
  - [image show](#)
  - [image unset](#)
- [ip availability](#)
  - [ip availability list](#)
  - [ip availability show](#)
- [ip fixed](#)
  - [ip fixed add](#)
  - [ip fixed remove](#)
- [ip floating](#)
  - [ip floating add](#)
  - [ip floating create](#)
  - [ip floating delete](#)
  - [ip floating list](#)
  - [ip floating remove](#)
  - [ip floating show](#)
- [ip floating pool](#)
  - [ip floating pool list](#)
- [keypair](#)
  - [keypair create](#)

- [keypair delete](#)
  - [keypair list](#)
  - [keypair show](#)
- [limits](#)
  - [limits show](#)
- [mapping](#)
  - [mapping create](#)
  - [mapping delete](#)
  - [mapping list](#)
  - [mapping set](#)
  - [mapping show](#)
- [module](#)
  - [module list](#)
- [network](#)
  - [network create](#)
  - [network delete](#)
  - [network list](#)
  - [network set](#)
  - [network show](#)
  - [network unset](#)
- [network agent](#)
  - [network agent add network](#)
  - [network agent add router](#)
  - [network agent delete](#)
  - [network agent list](#)
  - [network agent remove network](#)
  - [network agent remove router](#)
  - [network agent set](#)
  - [network agent show](#)
- [network auto allocated topology](#)
  - [network auto allocated topology create](#)
  - [network auto allocated topology delete](#)
- [network flavor](#)
  - [network flavor add profile](#)
  - [network flavor create](#)
  - [network flavor delete](#)
  - [network flavor list](#)
  - [network flavor remove profile](#)
  - [network flavor set](#)
  - [network flavor show](#)
- [network flavor profile](#)
  - [network flavor profile create](#)
  - [network flavor profile delete](#)
  - [network flavor profile list](#)
  - [network flavor profile set](#)
  - [network flavor profile show](#)

- [network meter](#)
  - [network meter create](#)
  - [network meter delete](#)
  - [network meter list](#)
  - [network meter show](#)
- [network meter rule](#)
  - [network meter rule create](#)
  - [network meter rule delete](#)
  - [network meter rule list](#)
  - [network meter rule show](#)
- [network qos policy](#)
  - [network qos policy create](#)
  - [network qos policy delete](#)
  - [network qos policy list](#)
  - [network qos policy set](#)
  - [network qos policy show](#)
- [network qos rule](#)
  - [network qos rule create](#)
  - [network qos rule delete](#)
  - [network qos rule list](#)
  - [network qos rule set](#)
  - [network qos rule show](#)
- [network qos rule type](#)
  - [network qos rule type list](#)
- [network rbac](#)
  - [network rbac create](#)
  - [network rbac delete](#)
  - [network rbac list](#)
  - [network rbac set](#)
  - [network rbac show](#)
- [network segment](#)
  - [network segment create](#)
  - [network segment delete](#)
  - [network segment list](#)
  - [network segment set](#)
  - [network segment show](#)
- [network service provider](#)
  - [network service provider list](#)
- [object](#)
  - [object create](#)
  - [object delete](#)
  - [object list](#)
  - [object save](#)
  - [object set](#)
  - [object show](#)
  - [object unset](#)

- [object store account](#)
  - [object store account set](#)
  - [object store account show](#)
  - [object store account unset](#)
- [policy](#)
  - [policy create](#)
  - [policy delete](#)
  - [policy list](#)
  - [policy set](#)
  - [policy show](#)
- [port](#)
  - [port create](#)
  - [port delete](#)
  - [port list](#)
  - [port set](#)
  - [port show](#)
  - [port unset](#)
- [project](#)
  - [project create](#)
  - [project delete](#)
  - [project list](#)
  - [project set](#)
  - [project show](#)
  - [project unset](#)
- [project purge](#)
  - [project purge](#)
- [quota](#)
  - [quota list](#)
  - [quota set](#)
  - [quota show](#)
- [region](#)
  - [region create](#)
  - [region delete](#)
  - [region list](#)
  - [region set](#)
  - [region show](#)
- [request token](#)
  - [request token authorize](#)
  - [request token create](#)
- [role](#)
  - [role add](#)
  - [role create](#)
  - [role delete](#)
  - [role list](#)
  - [role remove](#)
  - [role set](#)

- [role show](#)
- [role assignment](#)
  - [role assignment list](#)
- [router](#)
  - [router add port](#)
  - [router add subnet](#)
  - [router create](#)
  - [router delete](#)
  - [router list](#)
  - [router remove port](#)
  - [router remove subnet](#)
  - [router set](#)
  - [router show](#)
  - [router unset](#)
- [security group](#)
  - [security group create](#)
  - [security group delete](#)
  - [security group list](#)
  - [security group set](#)
  - [security group show](#)
- [security group rule](#)
  - [security group rule create](#)
  - [security group rule delete](#)
  - [security group rule list](#)
  - [security group rule show](#)
- [server](#)
  - [server add fixed ip](#)
  - [server add floating ip](#)
  - [server add port](#)
  - [server add security group](#)
  - [server add volume](#)
  - [server create](#)
  - [server delete](#)
  - [server dump create](#)
  - [server list](#)
  - [server lock](#)
  - [server migrate](#)
  - [server pause](#)
  - [server reboot](#)
  - [server rebuild](#)
  - [server remove fixed ip](#)
  - [server remove floating ip](#)
  - [server remove port](#)
  - [server remove security group](#)
  - [server remove volume](#)
  - [server rescue](#)



- [server resize](#)
  - [server restore](#)
  - [server resume](#)
  - [server set](#)
  - [server shelve](#)
  - [server show](#)
  - [server ssh](#)
  - [server start](#)
  - [server stop](#)
  - [server suspend](#)
  - [server unlock](#)
  - [server unpause](#)
  - [server unrescue](#)
  - [server unset](#)
  - [server unshelve](#)
- [server backup](#)
  - [server backup create](#)
- [server event](#)
  - [server event list](#)
  - [server event show](#)
- [server group](#)
  - [server group create](#)
  - [server group delete](#)
  - [server group list](#)
  - [server group show](#)
- [server image](#)
  - [server image create](#)
- [service](#)
  - [service create](#)
  - [service delete](#)
  - [service list](#)
  - [service set](#)
  - [service show](#)
- [service provider](#)
  - [service provider create](#)
  - [service provider delete](#)
  - [service provider list](#)
  - [service provider set](#)
  - [service provider show](#)
- [snapshot](#)
  - [snapshot create](#)
  - [snapshot delete](#)
  - [snapshot list](#)
  - [snapshot set](#)
  - [snapshot show](#)
  - [snapshot unset](#)

- [subnet](#)
  - [subnet create](#)
  - [subnet delete](#)
  - [subnet list](#)
  - [subnet set](#)
  - [subnet show](#)
  - [subnet unset](#)
- [subnet pool](#)
  - [subnet pool create](#)
  - [subnet pool delete](#)
  - [subnet pool list](#)
  - [subnet pool set](#)
  - [subnet pool show](#)
  - [subnet pool unset](#)
- [token](#)
  - [token issue](#)
  - [token revoke](#)
- [trust](#)
  - [trust create](#)
  - [trust delete](#)
  - [trust list](#)
  - [trust show](#)
- [usage](#)
  - [usage list](#)
  - [usage show](#)
- [user](#)
  - [user create](#)
  - [user delete](#)
  - [user list](#)
  - [user set](#)
  - [user show](#)
- [user role](#)
  - [user role list](#)
- [volume](#)
  - [volume create](#)
  - [volume delete](#)
  - [volume list](#)
  - [volume migrate](#)
  - [volume set](#)
  - [volume show](#)
  - [volume unset](#)
- [volume backup](#)
  - [volume backup create](#)
  - [volume backup delete](#)
  - [volume backup list](#)
  - [volume backup restore](#)

- [volume backup set](#)
  - [volume backup show](#)
- [volume host](#)
  - [volume host failover](#)
  - [volume host set](#)
- [volume qos](#)
  - [volume qos associate](#)
  - [volume qos create](#)
  - [volume qos delete](#)
  - [volume qos disassociate](#)
  - [volume qos list](#)
  - [volume qos set](#)
  - [volume qos show](#)
  - [volume qos unset](#)
- [volume service](#)
  - [volume service list](#)
  - [volume service set](#)
- [volume snapshot](#)
  - [volume snapshot create](#)
  - [volume snapshot delete](#)
  - [volume snapshot list](#)
  - [volume snapshot set](#)
  - [volume snapshot show](#)
  - [volume snapshot unset](#)
- [volume transfer request](#)
  - [volume transfer request accept](#)
  - [volume transfer request create](#)
  - [volume transfer request delete](#)
  - [volume transfer request list](#)
  - [volume transfer request show](#)
- [volume type](#)
  - [volume type create](#)
  - [volume type delete](#)
  - [volume type list](#)
  - [volume type set](#)
  - [volume type show](#)
  - [volume type unset](#)