

OPENSTACK

(Hazırlayan: @HalilGÖKSEL)

İçerik

1. Openstack Nedir.....
2. Openstack Kurulum Gereksinimi.....
3. Openstack Kurulumu.....
4. Openstack Horizon Menüler.....
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.....

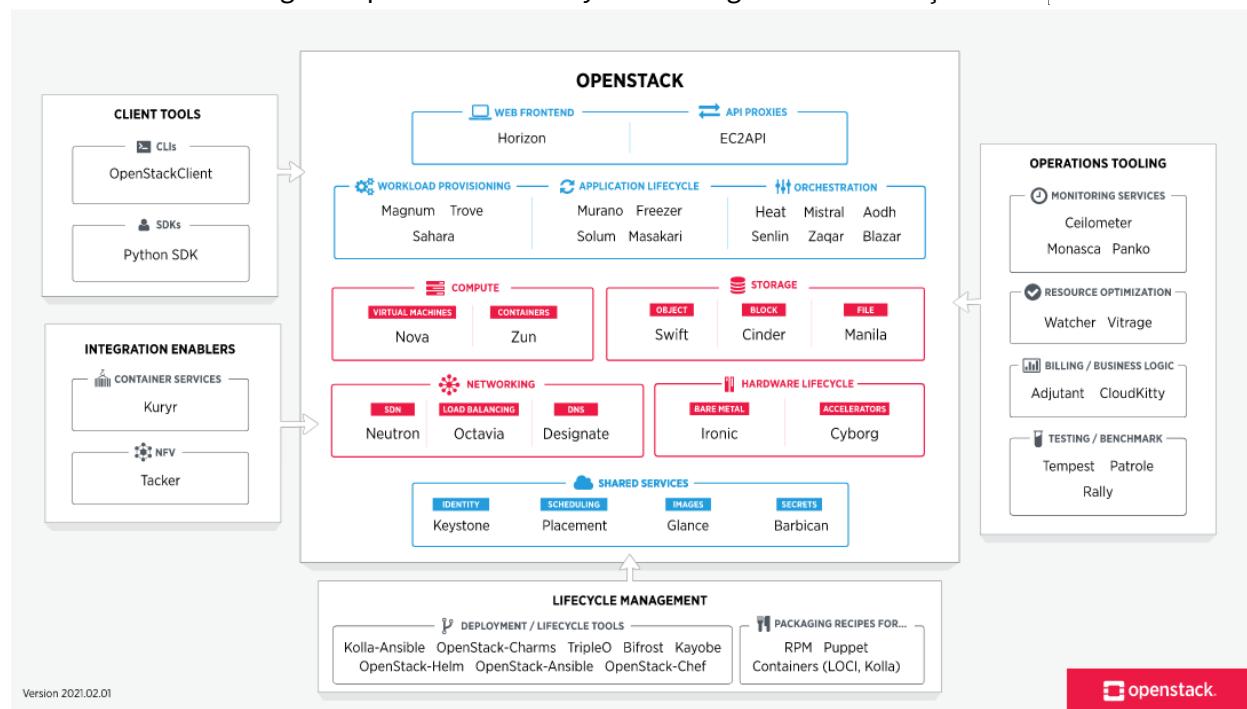
Openstack nedir

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 geliştirmektedirler.

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'in 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 dağıtımidır. RDO Openstack, RedHat'ın RPM Dağıtımını 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 türü olmalıdır ve CPU'da sanallaştırma açık olmalıdır.

Openstack Kurulumu

İlk olarak sıfırdan sanallaştırma üzerine bir Centos7.6 VM kurulumu yapıyoruz. Sanal VM'imizin interne 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
```

#vi editor'a aşağına degilseniz; 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)
```

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
```

```
[root@openstack ~]# systemctl status 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 yapılandırımları 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=disabled  
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
[root@openstack ~]#
```

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-ZxUXa0/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.

```
[root@openstack ~]# Applying Puppet modules and manifests [ DONE ]
Applying 172.18.156.6_controller.pp
172.18.156.6_controller.pp: [ DONE ]
Applying 172.18.156.6_network.pp
172.18.156.6_network.pp: [ DONE ]
Applying 172.18.156.6_compute.pp
172.18.156.6_compute.pp: [ DONE ]
Applying Puppet manifests
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-b41YiS/openstack-setup.log
* The generated manifests are available at: /var/tmp/packstack/20220831-183111-b41YiS/manifests
You have new mail in /var/spool/mail/root
[root@localhost ~]#
```

Bitti.

13. Adım

```
# ethernet arabirim ayarlarınızın böyle göründüğünden 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 göründüğünden 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 keystonerc_admin
```

16. Adım

```
örnekleriniz için sağlayıcı ağınızı oluşturmak için bu komutu çalıştırın, böylece sağlayıcılar  
# dış dünyaya iletişim kurun
```

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

17. Adım

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

```
# linux makinenizin bağlı olduğu lan'a göre yapılandırma
neutron subnet-create --name public_subnet --enable_dhcp=False --allocation-pool start=,end=-
gateway=external_network
```

example:

```
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.
```

```
[root@localhost ~]# neutron subnet-create --name public_subnet --enable_dhcp=False --allocation pool start=172.18.156.100,end=172.18.156.120 --gateway=172.18.156.1 external_network 172.18.156.0/24
neutron CLI is deprecated and will be removed in the future. Use openstack CLI instead.
Auth plugin requires parameters which were not given: auth_url
[root@localhost ~]#
```

18. Adım

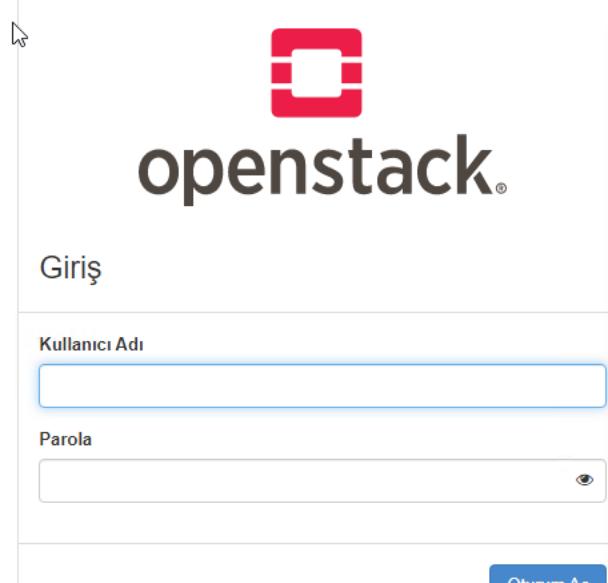
Openstack admin panaline giriş için keystone RC'den admin password bilgilerini alabiliriz.

More kestonerc_admin

```
Auth plugin requires parameters which were not given: auth_url
[root@localhost ~]# more kestonerc_admin
unset OS_SERVICE_TOKEN
export OS_USERNAME=admin
export OS_PASSWORD='f75ab3ae2316434a'
export OS_REGION_NAME=RegionOne
export OS_AUTH_URL=http://172.18.156.6:5000/v3
export PSL='[\u0009 \W(keystone_admin)]\$'

export OS_PROJECT_NAME=admin
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_DOMAIN_NAME=Default
export OS_IDENTITY_API_VERSION=3
```

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



Giriş

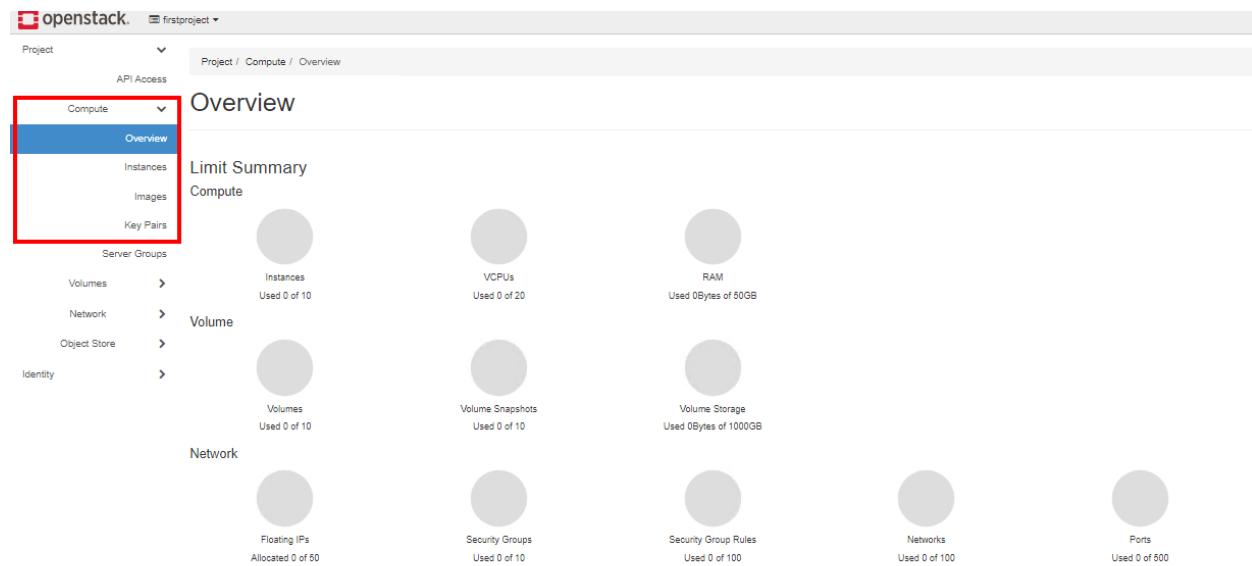
Kullanıcı Adı

Parola

Oturum Aç

```
root@localhost:~
32  vi /etc/selinux/config
33  reboot
34  history
35  neutron subnet-create --name public_subnet --ena
on pool start=172.18.156.100,end=172.18.156.120 --gatew
network 172.18.156.0/24
36  more kestonerc_admin
37  sudo yum install -y centos-release-openstack-tr
38  sudo yum install yum-utils
39  sudo yum-config-manager --enable openstack-train
40  sudo yum update -y
41  sudo yum install -y openstack-packstack
42  ip address show
43  packstack --allinone --provision-demo=n --os-neu
extnet:br-ex --os-neutron-ml2-mechanism-drivers=openvsw
t=openvswitch -- os-neutron-ovs-bridge-interfaces=br-ex
type-drivers=vxlan,flat --os-neutron-ml2-tenant-network
44  vi /etc/sysconfig/network-scripts/ifcfg-ens192
45  source kestonerc_admin
46  neutron net-create external_network --provider:n
47  neutron net-create external_network --provider:n
der:physical_network extnet --router:external
48  history
[root@localhost ~]#
```

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 instanceler SSH ile erişebilmek adına keypairs oluşturduğumuz kısımdır.

Server Groups – Instance guruplarını yönettiğimiz alandır, Toplu yönetim için kontainer gibi düşününebilir, Vmware karşılığı vApp'dır.

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) yapılandırılabilir, Farklı vlanlar arasında bağlantı ilişkisi yapılandırılabilir. 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: Hipervizör 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 yapılandırmásındaki 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 : Volume Volume grupları oluşturma silmek ve düzenlemek ve snapshotını almak için kullandığımız menülerdir.

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

Network

Networks: Private ağlar oluşturmak ve düzenlemek için kullandığımız menüdür.

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

Floating IPs: Türkçe karşılığı kayan IP'nin mantığı HA sağlamaktadır, Network'de oluşan herhangibir sorunda Floating IPs devreye girer.

Trunks: Bulun yönetici olarak trunk portalını 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 çalıştıktarı sunucuları görebiliriz.

Identity

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

Users Admin user: Openstack tüm userları yönettiğimiz alandır/ Standart user: Standart kullanıcımızın göründüğü 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.

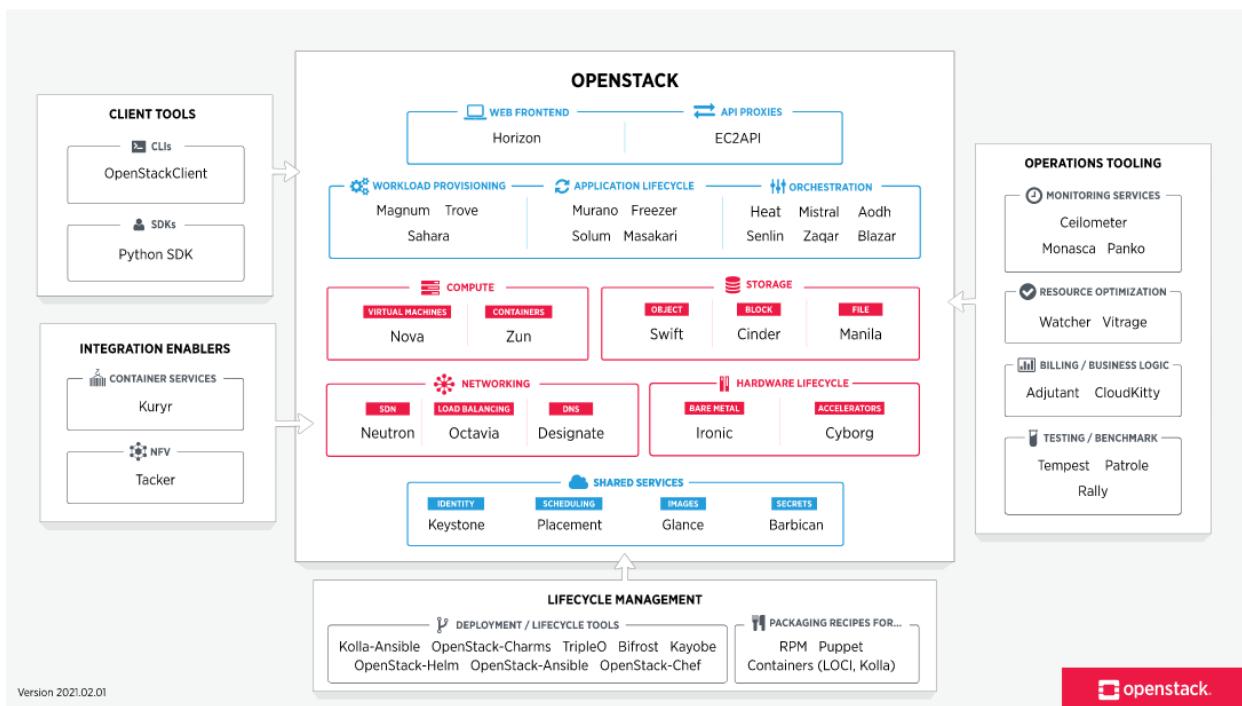
Project

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

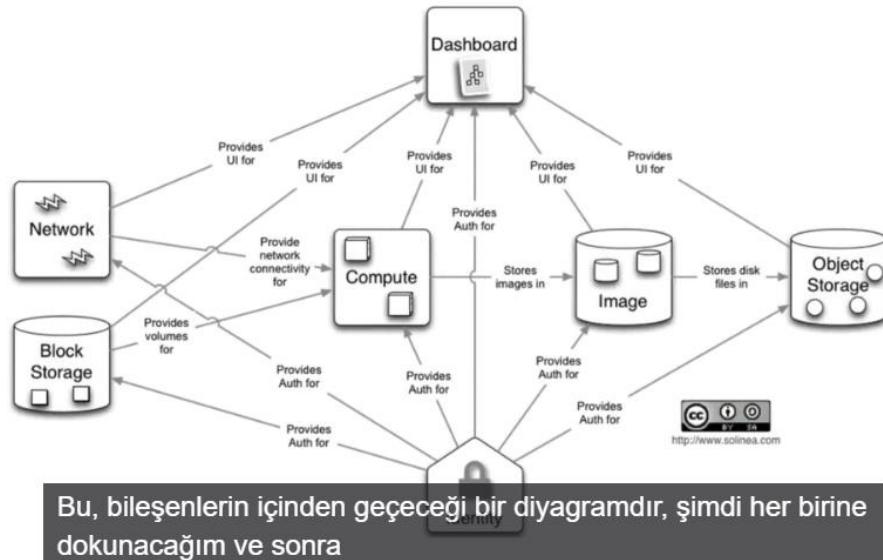
Openstack API Servisleri

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, paylaşımı dosya sistemi deplama, 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 çalıştırma için kullanılır. Varsayılan olarak, Hesaplama geçici sürücülerini dosya olarak saklar hesaplama düğümündeki yerel diskler * /var/lib/nova/ örnekleri * yalnızca sanal makine geçiş disk görüntüsünü diğerine taşırı hesaplama düğümü (Nova SSH üzerinden kopyalar

Block Storage:

Sanal makineye ek kalıcı depolama alanı ekleme • Bu olabilir bir blok cihaz üzerinden erişilir bölümlemiş, 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 makina görüntülerimiz bir nesne depolama sistemi içinde saklayabiliriz. Kullanım durumları: Yedekleme dosyaları veritabanı dökümleri için depolama ve

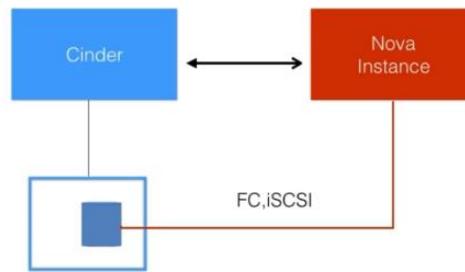
günlük dosyaları; Büyük veri kümeleri (ör. multimedya dosyaları); arka uç görüntü depolama
Shared File System Storage:

Çoklu kiracılı bir bulut ortamında kullanıcılarla 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 tüketebilecek açık yığın ise Project nova'dır.

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

Cinder'ı düşündüğümüzde bir sadece USB sürücüsü gibi düşünebiliriz.

Diyelimki bir VM çalıştırma 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'larıda işlemektedir.

Cinder aracı 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

The screenshot shows the 'Volumes' section of the OpenStack Compute interface. It lists two volumes: 'snapshot-001' and 'persister'. Both volumes are 1GB in size and are attached to local instances. The 'snapshot-001' volume is available, while 'persister' is in use.

Snapshots

A read only copy of a volume

Create/delete snapshots

Create a volume out of a snapshot

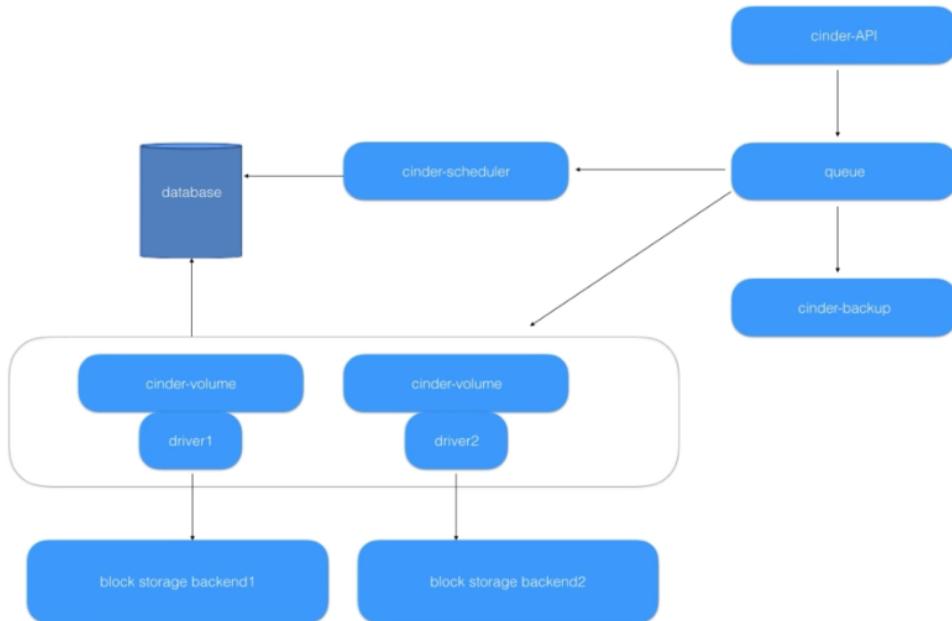
Cinder ayrıca anlık görüntüde işler, Anlık görüntü birimin zaman kopyasında salt okunur bir noktadır., Blok birimlerinin anlık görüntülerini alabilir.

The screenshot shows the 'Volume Snapshots' section of the OpenStack Compute interface. It lists two snapshots: 'snapshot-redhat03-01_15_2017' and 'snapshot-cirros-01_15_2017', both 1GiB in size and available. They were created from the volume 'persistent1'.

Cinder Mimarisi

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

Architecture



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

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.

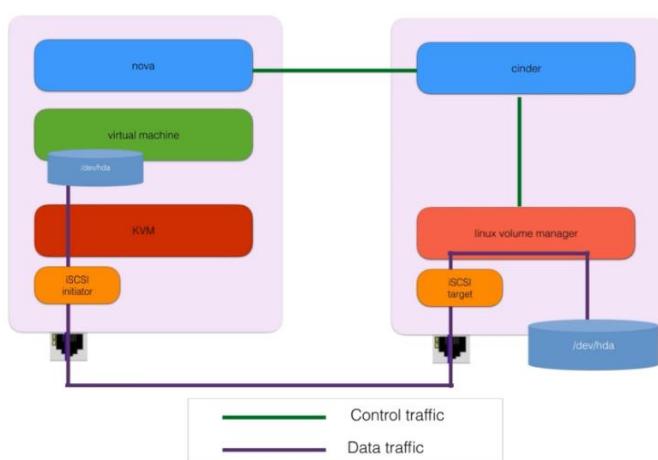
Cinder scheduler: Birimin oluşturacağı en uygun depolama sağlayacısını seçer.

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 sağlayacısına 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 yapnanın 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)

cinder service-list						
Binary	Host	Zone	Status	State	Updated_at	
Cluster	Disabled Reason	Backend State				
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	
		up				

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>
--source <volume> Use <snapshot> as source of volume (name or ID)
--volume <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

openstack volume list			
	Name	Status	Size
	7b8304c1-8bb2-4261-918b-a8a5815938cf	available	1

Volumumuza eklemek için

Openstack server add volume "instance01" "vol1" komutunu çalıştırıyoruz.

Diskimizi instance ekledikten sonra OS'dan biçimlendirme yapmak gereklidir

```
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)

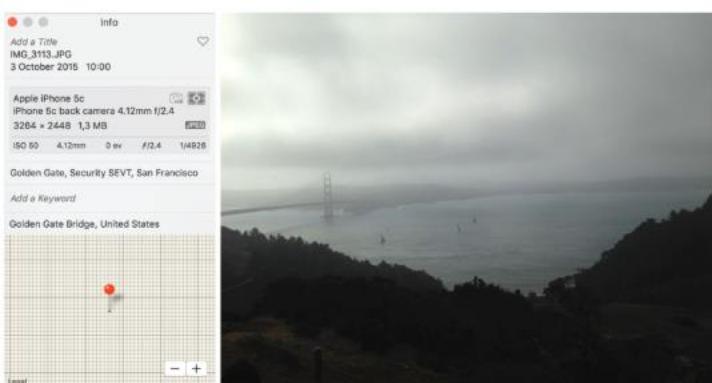
Bir birim VM'e disk olarak hizmet sunarken, Ciner'in aksiyone Swift içinden içerik geçirmeniz için bir API sunar. Çok temel dosya depolanması ve çok güçlü olabilmektedir.

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

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

What Is An Object?

Object = File + Metadata

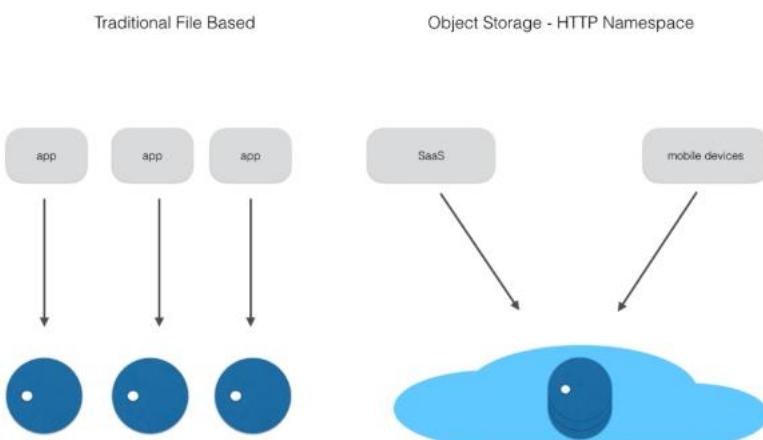


Swift nesneleri içinde saklar, nesne veri parçası olan metaverilere diyebiliriz.

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 yazdığını birden fazla konuma yazması gerekebilir, Doğru işlemin tamamlanması için, işlemlerin sırayla başarılı şekilde tamamlanması gereklidir, 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 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 ileterken depolama servisi ile iletişim kurar.

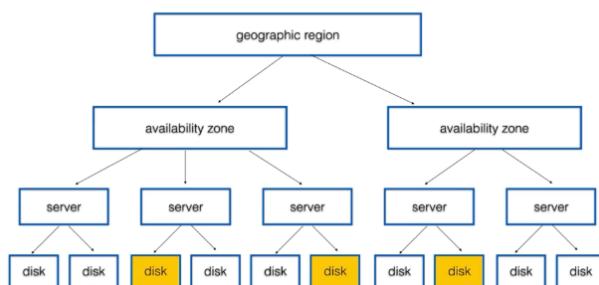
Durability with Replicas

- 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

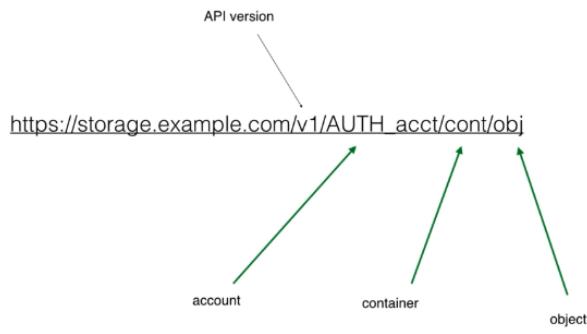
Sonuç olarak swift çoğaltılmış bir depolama sistemi olduğunu söyleyebiliriz. Dayanıklılık ve high availability sağlar. Çok fazla CPU gereksinimi olmadan kolay okuma yazma sağlar. Web ve mobil içerikler için tercih edilir.

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

Data Placement in Swift



Swift essentials: Account, Container and Object



Örnek bir mimari sol tarafta görebiliriz.

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

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

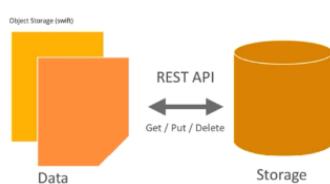
Swift yasmak için PUT, okumak için GET protokolünü kullanırız.

Write an object:

PUT /v1/account/container/object

Read an object:

GET /v1/account/container/object



Swift ile konuşmak için web tarayıcı kullanabiliriz.

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

LB: tüm iletişimler nedeniyle çok sayıda istege hizmet vermek için yük dengeleme yapmak gereklidir.

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

Authentication: Swift verilere ulaşmak için kimlik doğrulamaya ihtiyaç 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 alına yerleşicektir)

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 kestonerc_admin / kestoner dosyasını container2 içine obje olarak atıyoruz.

```
[root@localhost ~ (keystone_admin)]# openstack object create container2 kestonerc_admin
+-----+-----+-----+
| object      | container | etag          |
+-----+-----+-----+
| kestonerc_admin | container2 | 94a73b0938c7437440fc90428a7f4dd |
+-----+-----+-----+
[root@localhost ~ (keystone admin)]#
```

Horizonta control ettiğimizde sorun şekilde erişebildiğimizi görüyoruz.

Containers

The screenshot shows the OpenStack Object Store interface under the 'Containers' section. On the left, there's a sidebar with a '+ Container' button and a search bar. The main area displays a table for 'container2'. The table includes columns for 'Object Count' (1), 'Size' (372 bytes), 'Date Created' (Sep 7, 2022), and 'Public Access' (checkbox checked, with a 'Link' button). To the right, there's a search bar and a list of objects. The first object is 'keystonerc_admin' with the same details as the container. A message at the bottom states: 'This XML file does not appear to have any style information associated with it. The document tree is shown below.' Below this message is the XML document tree.

container2

Click here for filters or full text search.

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

volumebackups

Displaying 1 item

Name ▲

keystonerc_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"?>
<container name="container2">
  <object>
    <name>keystonerc_admin</name>
    <hash>94a73b0938c7437440fc90428a7f4dc</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

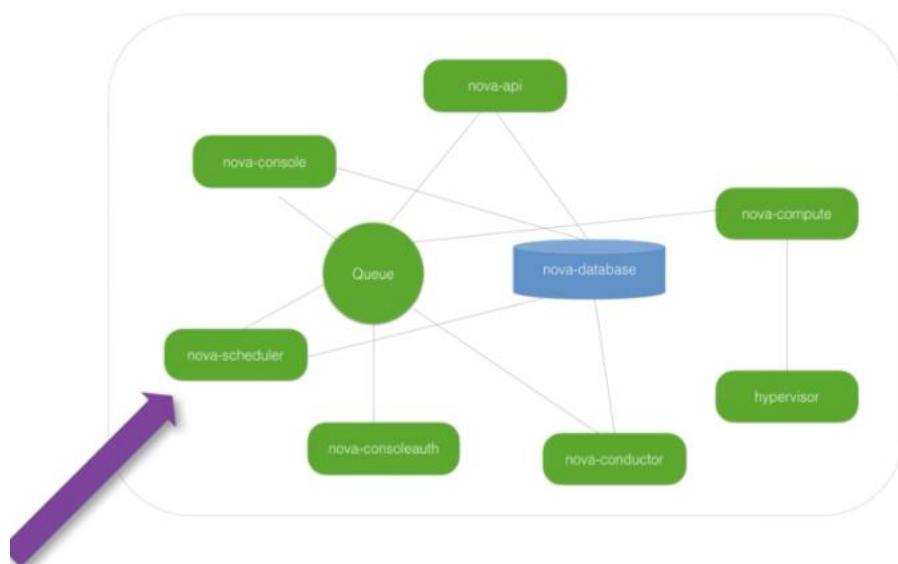
Nova, keystone, glance, neutron, cinder ve swift gibi diğer hizmetlerin desteğine ihtiyaç duyar ve şifreli diskler ve çiplak metal bilgi işlem örnekleri uygulamak için bu hizmetlerle entegre olabilir.

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

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

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

NOVA MİMARİ



Queue: Proje içindendeki tüm iletişimimizi, merkezde göründüğ 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 zmamanını var çalışma zamanının varlıklarını depolar.

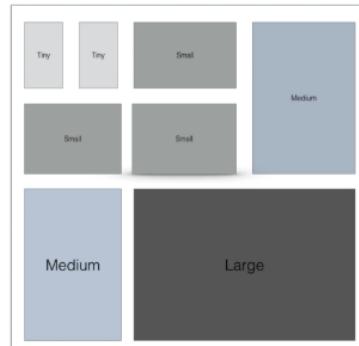
Flavor (Image Servisi)

(Instance oluşturan hangi kaynak değerlerince vereceğimiz ön şablonu olarak düşünübiliriz, imajımızı glance'den seçtikten sonra flover'da secerék kaynak değerlerini belirtebiliriz.)

Opensak flavor list komutu ile var olan flovar'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



Flovar oluşturmak için CLI komutu,

Openstack flavor create -h (help) ile inceleyebiliriz.

```
 positional 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>
                        Ephemeral 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	m1.tiny	512	1	0	1	True
10	m1.tinier	256	2	0	1	True
2	m1.small	2048	20	0	1	True
3	m1.medium	4096	40	0	2	True
4	m1.large	8192	80	0	4	True
5	m1.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.

Instancelarımızın geldiği horizon üzerinden görebiliyoruz.

The screenshot shows the OpenStack Horizon interface under the 'Compute' tab. On the left, there's a navigation sidebar with 'Overview', 'Instances', 'Hypervisors', 'Aggregates', and 'Images'. The 'Instances' link is highlighted with a blue box. The main content area is titled 'Instances' and displays two entries:

Project	Host	Name	Image Name
admin	-	instance01	cirros
admin	-	intanece01	cirros

Zun (Container Servisi)

Zun nedir?

Zun bir OpenStack Konteyner servisidir. Sunucuları veya kümeleri yönetmeye gerek kalmadan uygulama kapsayıcılarını çalıştırma 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 kullanıcıı 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 tüketilebilir 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

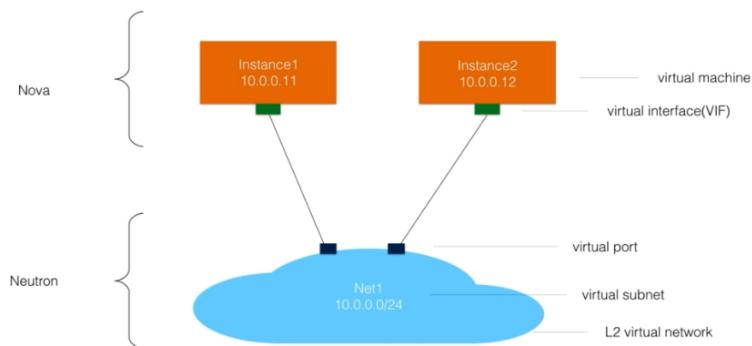
Neutron (Network servisi)

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

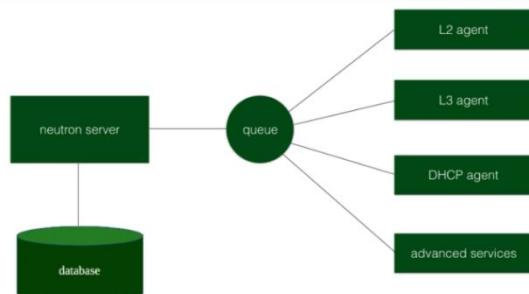
Dış dünya ile haberleşirebilir, Yönlendirme yapabilir, LB, VPN yapısı kurabiliriz, genel olarak tüm firewall'larda olan donanımları kapsar diyebiliriz.

Neutron ile zengin topolijiler oluşturabiliriz.

Nova ile Neutron grafiği aşağıdaki gibidir.



Neutron Architecture



L2 agent: Bağlantıları kablolamakla ve aygıtları birbirine bağlamakla sorumludur.

L3 agent: farklı networkler arasında bağlantı kullanan servisdir.

DHCP agent: DHCP protokolünü kullanan servisdir

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

Network Technologies Supported

- 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

Desteklenen teknoloji segmentleri sol tarafdaki gibidir..

D

Local network: Biririne bağlı nodeler arasında iletişim gerçekleşir.

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

Vlan network: Flat network aksiyone birden fazla yayın alanını paylaşır. Ölçeklenebilir ve güvenilirdir. 4096 vlan'a kadar destekler. Farklı vlanlar arasında konursturma yapma için yönlendirme yapılması gereklidir.

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 tünenmesi 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 işlevselligi

Security Group: Güvenlik grupları kurallar, yöneticilerin ve kiracıların trafik türünü belirtebilmeleri olarak tanır. Çaklısan IP adresleri desteklenir, kiracılar arasında sorun yaratmaz. IPv6 desteklemektedir. Sanal birim oluşturulduğunda bir security grubuya 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 yönlendirici L3 agent katmanında çalışır. Sanal Yönlendirici Artıklık Protokolü (VRRP) kullanarak büyütmeye 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 I3 aracısındaki yedek DVR/SNAT yönlendiricisine hızlı bir şekilde yük devretmesini sağlar.

Floating IP Adresses: Kayan IP, yalnızca özel bir alt ağda (yani ortak ağ arabirimini 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 arabirimini 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, tabii internețe çıkış için rout etmemiz ve kura yazmamışsa 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:

- o Kullanıcıyı oluşturun **designate**:

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

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

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

- o Belirlenen hizmet varlıklarını oluşturun:

```
o $ 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**'ının seçtiğiniz şifre ile değiştirin .**openstack**

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

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

Veritabanı **DESIGNATE_DBPASS**'ının 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)

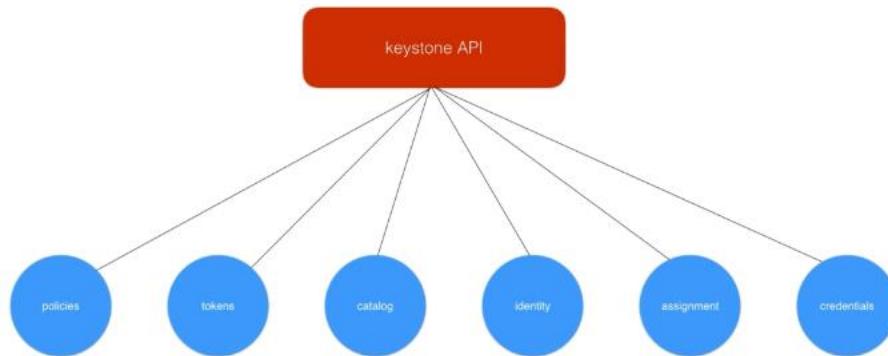
Keystone servisi kimlik doğrulama ve yetkilendirme için kullanır, Ve tüm geçerli hizmetler openstack üzerindeki API servisleri üzerinden request'lerde doğrulama yaparak işlemlerini gerçekleştirmektedir. Örnek olarak bir user, VM oluşturmak istediğiNova üzerinden doğrulama alır, Network oluşturmak istediği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

Keystone servisi LDAP desteklemektedir, AD entegrasyonu bu noktada mümkündür.

Keystone servisi aşağıdaki aksiyonlarda rol oynamaktadır.

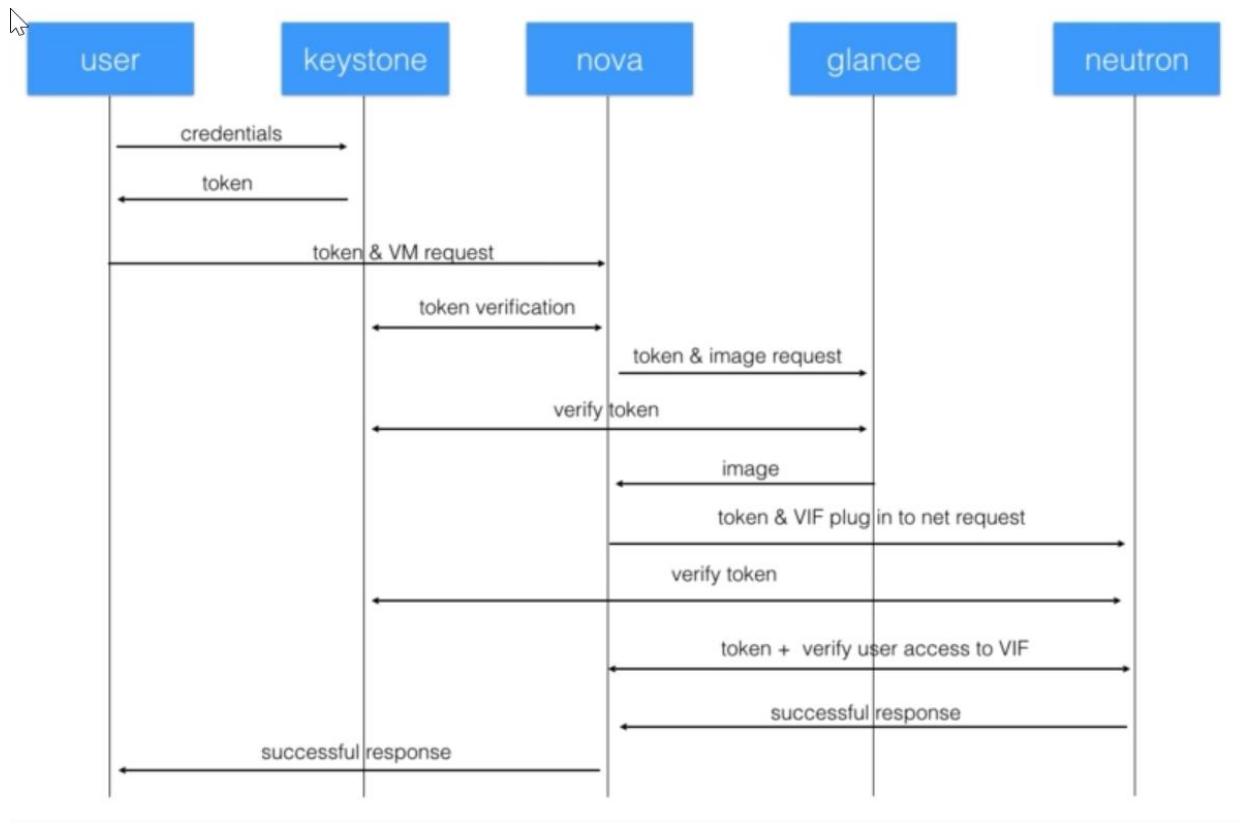
- Kullanıcılar
- Projeler (Yani kiracılar) Hem global bazında yetki , Hemde projere bazında yetkilendirme
- Roller(Yetkilendirme ve sınırlandırma)

- 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 grafikdeki 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.



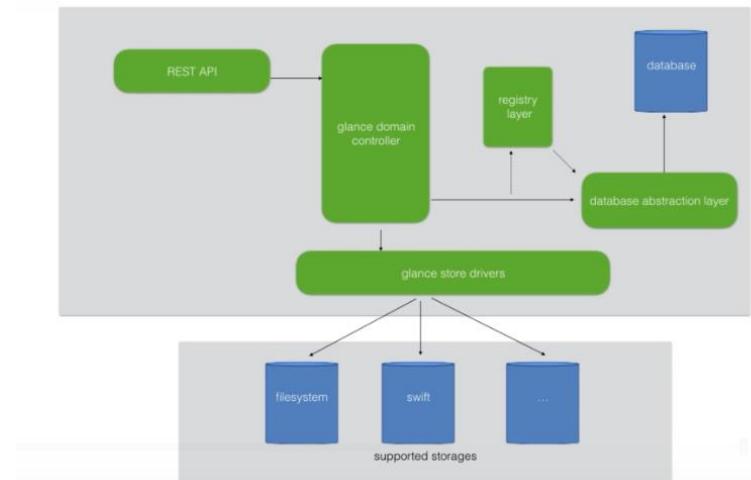
Glance (İmaj Servisi)

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

Dışarıdan başka bir imaj ekleyebiliriz. RAW, VMDK(Vmware), VDI(virtualbox), VHD(Hyper-V), OVF, Qcow2(KVM) gibi imaj tiplerini desteklemektedir.

Glance çalışma mimarisi aşağıdaki grafikdeki gibidir.

Architecture



CLI yönetmek için örnek uygulama aşağıdaki gibidir

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

```
[root@localhost ~]# 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
```

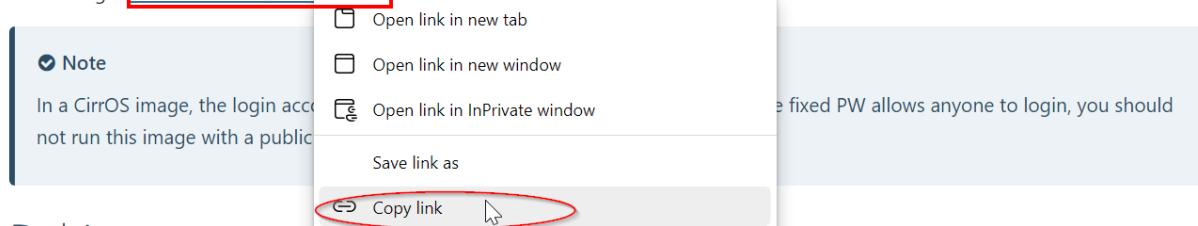
Openstack image list (openstack bulutumuzdaki imajları listelemek için kullandığımız komut)

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 OFMU 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](#)



Indirmek için

Curl -o "localfilestorageimage" "imageurl" komutu kullanmalıyız.

İndirme işlemi tamamlandı.

```
[root@localhost ~]# openstack.load_balancer.v2 | loadbalancer amphora configure
[root@localhost ~]# ls
1.0.3.tar.gz anaconda-ks.cfg keystonerc_admin masscan-1.0.3 packstack-answers-20220831-183112.txt
[root@localhost ~]# pwd
/root
[root@localhost ~]# 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 ~]# ls
1.0.3.tar.gz cirros-0.3.4.img masscan-1.0.3
anaconda-ks.cfg keystonerc_admin packstack-answers-20220831-183112.txt
[root@localhost ~]#
```

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
```

Field	Value
checksum	25c96942084f61e008d593b6c2cfda00
container_format	bare
created_at	2022-09-04T19:52:35Z
disk_format	qcow2
file	/v2/images/laf2d476-006e-47dc-a05a-b16fd433b055/file
id	laf2d476-006e-47dc-a05a-b16fd433b055

Horizontanda teyit ediyoruz.

The screenshot shows the OpenStack Horizon dashboard. The user is navigating through the Compute section, specifically the Images page. The 'Images' tab is active. A single image entry is displayed: 'cirros' (Owner: admin). There are also tabs for 'Key Pairs', 'Server Groups', and 'Volumes'. A search bar and filter options are present at the top of the list.

Openstack image show "image adı" ile imajımıza ait bilgileri görebiliriz.

Openstack image delete "image adı" ile imajımızı silebiliriz

Openstack image list ile imajlarımızı görebiliriz.

ID	Name	Status
laf2d476-006e-47dc-a05a-b16fd433b055	cirros	active

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 yaştıda karşılaşacak 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.

Proje adı belirliyoruz.

Create Project

Project Information * Project Members Project Groups

Domain ID: default
Domain Name: Default
Name: halilgoksel.com
Description:
Enabled:

Cancel Create Project

Create Project

Project Information * Project Members Project Groups

All Users Filter Project Members Filter

User	Role
hg	+ <input type="button"/>
glance	+ <input type="button"/>
admin	_member_admin <input type="button"/>

Admin kullanıcıyı, admin yetkileri ile projeye atıyoruz.

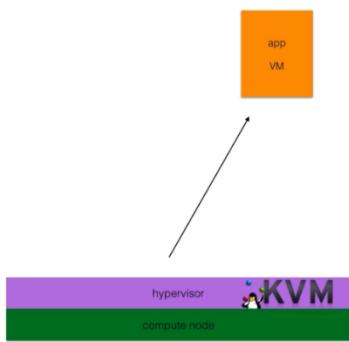
Oluşturuktan sonra set quota diyip, proje kotasını change edebiliyoruz

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

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

Instance Oluşturma

Launching an Instance



- Instance(Nova):
- Image(Glance)
 - Network(Neutron)
 - Size of the Instance(Nova)
 - Security settings
 - ACLs(Neutron)
 - Key pair (Nova)
 - Persistent storage(Cinder)

Openstack üzerinde Instance oluşturmak için zorunlu olan ve zorunlu olmayan API servislerinide bununla beraber oluşturmanız gereklidir.

Instance (Nova): Openstack'in bilgisi işleyen çekirdek birimidir diyebiliriz

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

Network: İnstancemizin, internet yada

instranet iletişim için network yapılandırımlarının yapılacaksı servistir. Key pair servisi, uzaktan yönetebileceğimiz servistir, Çift anahtar gereklidir.

Cinder: İnstancemizin, kalıcı depolama sağladığı servistir.

Gereksinimleri katagorize edecek olursak instance oluşturmak için, Glance, Neutron, Nova servislerini kullanmalıyız. Diğer keypair ve Cinder isteği 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 dierek devam ediyoruz.

The screenshot shows the 'Create Image' dialog box in the OpenStack Horizon interface. The 'Image Details' section has 'Image Name' set to 'cirros'. The 'Image Source' section shows a file named 'cirros-0.5.1-x86_64-disk.img' selected. The 'Format' dropdown is set to 'QCOW2 - QEMU Emulator'. In the 'Image Requirements' section, 'Kernel' and 'Architecture' dropdowns are empty. Under 'Image Sharing', 'Visibility' is set to 'Private'. The 'Protected' checkbox is checked. At the bottom right, there are 'Create Image' and 'Next >' buttons.

Admin useri giriş yaptığımızda imaj sahibini görebiliriz

The screenshot shows the 'Images' tab in the OpenStack Horizon interface for the 'admin' user. A single item, 'cirros', is listed under the 'Owner' column, which shows 'halligoksel.com'. The 'Name' column also shows 'cirros'. The row for 'cirros' is highlighted with a red box.

Flavor Oluşturma

Flavors özetçe kaynakların önceden tanımlandığı bir çeşit yapılandırma servisi diyebiliriz.

The screenshot shows the OpenStack Compute interface with the 'Flavors' tab selected. A modal window titled 'Create Flavor' is open, showing fields for 'Name' (halilgoksel.com), 'ID' (34), 'VCPU' (1), 'RAM (MB)' (128), 'Root Disk (GB)' (1), 'Ephemeral Disk (GB)' (0), 'Swap Disk (MB)' (0), and 'RX/TX Factor' (1). The 'Create Flavor' button at the bottom right is highlighted with a red box.

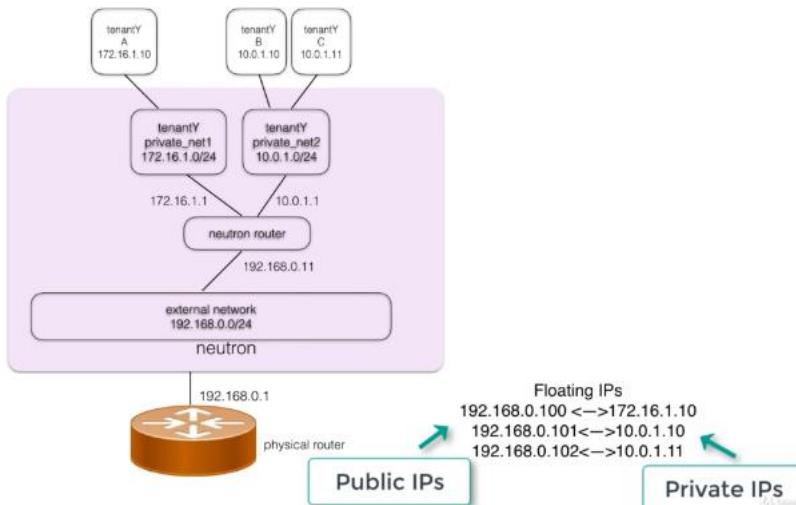
Admin kullanıcı ile flavors oluşturabiliriz.

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 toplojisi belirtilmiş.

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

Network

The screenshot shows the Openstack dashboard with the URL halilgoksel.com. The left sidebar is collapsed. The main navigation bar shows 'Project / Network / Networks'. A modal window titled 'Create Network' is open. The 'Network' tab is selected. The 'Network Name' field contains 'private_network1'. Two checkboxes are checked: 'Enable Admin State' and 'Create Subnet'. Under 'Availability Zone Hints', 'nova' is listed. At the bottom of the modal are 'Cancel', '< Back', and 'Next >' buttons.

Network Oluşturma

Projemize member kullanıcımızla giriş yapıp network tabında create network diyoruz.

Network adını belirtiyoruz, subnet kısmına geçiyoruz.

The screenshot shows the 'Create Network' wizard at the 'Subnet' step. The 'Subnet' tab is selected. The 'Subnet Name' field contains 'private_subnet1'. The 'Network Address' field contains '10.0.1.0/24'. The 'IP Version' dropdown is set to 'IPv4'. The 'Gateway IP' field is empty. A checkbox for 'Disable Gateway' is unchecked. At the bottom are 'Cancel', '< Back', and 'Next >' buttons.

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

Subnet details kısmında DHCP, DNS ve Route düzenlemelerini yapabiliyoruz, default olark DHCP aktiftir, belirli aralıkta dağıtım istiyorsan set edebiliyoruz.

The screenshot shows the 'Create Network' wizard at the 'Subnet Details' step. The 'Subnet Details' tab is selected. A checkbox for 'Enable DHCP' is checked. A note says 'Specify additional attributes for the subnet.' Below it, an 'Allocation Pools' field contains '10.0.1.10,10.0.50'. A 'DNS Name Servers' field contains '8.8.8.8'. At the bottom are 'Cancel', '< Back', and 'Next >' buttons.

Create diyoruz, statusunu ve içeriğini görebiliriz.

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

Sağ tarafta görüldüğü üzere bir bağ yok, aralarında bağlantıyi 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 interface 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 karşılamakta.

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	-	-	<button>Add Rule</button> <button>Delete Rule</button>
Egress	IPv6	Any	Any	::/0	-	-	<button>Add Rule</button> <button>Delete Rule</button>
Ingress	IPv4	Any	Any	-	default	-	<button>Add Rule</button> <button>Delete Rule</button>
Ingress	IPv6	Any	Any	-	default	-	<button>Add Rule</button> <button>Delete Rule</button>

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

The screenshot shows the 'Add Rule' dialog for a security group named 'securitygroup1'. The 'Rule' dropdown is set to 'ALL ICMP'. The 'Description' field contains 'some any'. The 'Direction' dropdown is set to 'Ingress'. The 'Remote IP Prefix' dropdown is set to 'CIDR 0.0.0.0'. The 'Add' button at the bottom right is highlighted with a red box.

Görselde yer verildiği gibi securitygroup1 olarak oluşturduk.

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

CIDR diyerek tüm viana 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.

The screenshot shows the OpenStack Compute interface under the 'Compute' tab. In the 'Key Pairs' section, a search bar contains the text 'halil'. Below the search bar, a table displays a single key pair entry:

Name	Public Key
ssh-nsa	AAAAAB3NzaC1yc2EAAAQABAAQDl9W3HuTKQ8DRHcXUCAysODN924ldFTtempWkID+jygbJdy4A+hPXJr3Mreibua... ku1QB1c+8oPbtkseawhtx4up7d7RaQT8RQjg73y8JuzUz1RiAZGhfVtsRBVhnsf Generated-by-Nova

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

Herşeyimizi tamamladık şimdi sıra instance yaratmakta geldi.

Instance Oluşturma Part 2

Launch instance diyerken işlemi başlatıyoruz.

Instance Name kısmına instance bir isim veriyoruz.

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

Count birden fazla instance oluşturmak için kullanabiliriz.

The screenshot shows the 'Launch Instance' dialog box. The 'Details' tab is active. The form fields include:

- Source: halilgoksel.com
- Flavor: nova
- Networks: (empty)
- Network Ports: (empty)
- Security Groups: nova
- Key Pair: ssh-nsa
- Configuration: Count: 1

On the right side, there is a progress bar indicating 20% completion and a legend for 'Total Instances (5 Max)':

- 0 Current Usage (blue)
- 1 Added (blue)
- 4 Remaining (light grey)

At the bottom, there are buttons for 'Cancel', 'Back', 'Next >', and a large blue 'Launch Instance' button.

Launch Instance

Source

Flavor *

Networks *

Allocated

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

Available

No available items

Metadata

Cancel **Next >** **Launch Instance**

Buradan Boot source cirros imajımızı kullanacağımız için seçiyoruz.

Avaible 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

Flavor

Available

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
halilgoksel.com	1	128 MB	1 GB	1 GB	0 GB	Yes
m1.tiny	1	512 MB	2 GB	2 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 **Next >** **Launch Instance**

Network kısmında instance ait network seçiyoruz.

Launch Instance

Details

Source

Flavor

Networks

Network Ports

Security Groups

Key Pair

Configuration

Server Groups

Scheduler Hints

Metadata

Allocated 1

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

Available 0

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

Click here for filters or full text search.

< Back Next > Launch Instance

Network port extra birbağlantı noktasına ihtiyacımız olmadığından atlıyoruz.

Security group'dan oluşturduğumuz security group seçiyoruz.

Launch Instance

Details

Source

Flavor

Networks

Network Ports

Security Groups

Key Pair

Configuration

Server Groups

Scheduler Hints

Metadata

Select the security groups to launch the instance in.

Allocated 2

Name	Description
default	Default security group
securitygroup1	

Available 0

Name	Description
No available items	

Click here for filters or full text search.

< Back Next > Launch Instance

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

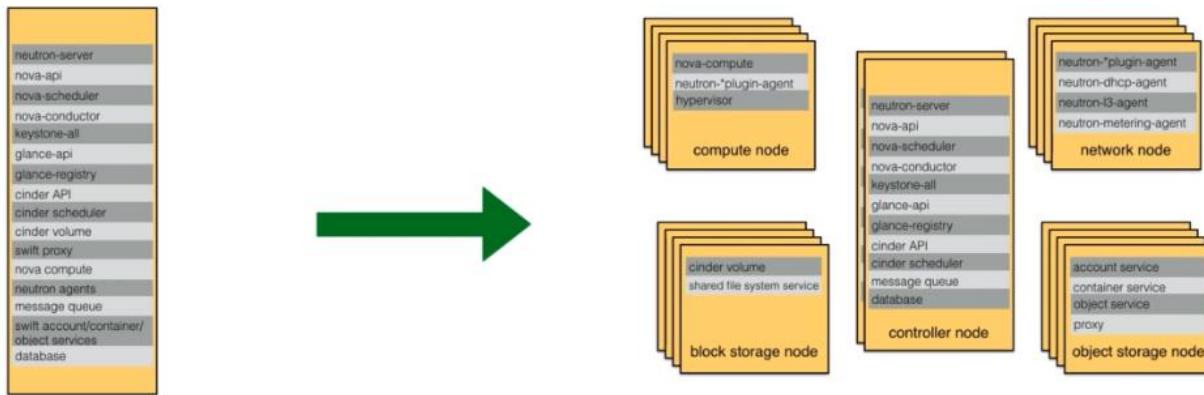


Bir süre sonra instance olduğunu görüyoruz. Action tabında instance üzerinde Instance düzenleyebilir, NIC ekleyebilir, Floating IP ekleyebilir, suspend, shutdown, snapshot gibi işlemler yapabiliriz.

Openstack Ölçeklendirme

Nova Node

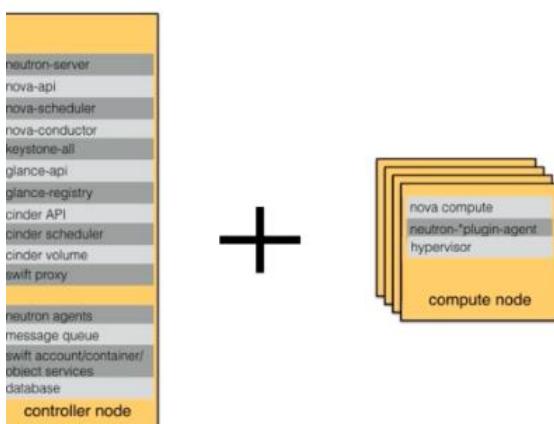
Deployment Scale



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

Hangi servisleri nerede çalıştırırmalım sorusuna ölçeklendirmede sizler topolojiye uygun karar verebilirisiniz, 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 Node

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

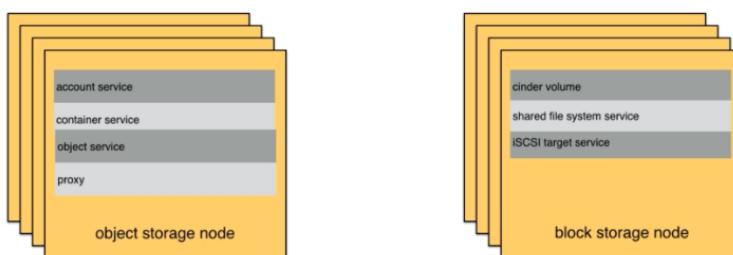
- Local disks on compute nodes used
- Heavy I/O usage on one compute node does not affect instances on other compute nodes
- When you lose a node, data on it gets lost as well
- Instance migration is complicated

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

Sonuç olarak İşlem ve depolama konakları ayrıdır. Compute node'lar için herhangi bir bakım gereğinde başka bir düğüme geçirerek bakım için çevrim dışı moduna (vmware: maintenance mode) getirebiliriz. İkinci depolama yaklaşımı, paylaşılan dosya sisteme 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 baglama dönüştürür.

Bu seçeneğin temel avantajı diskleri tek bir baglama 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 gerekli tüm verileri üzerinde barındırır.

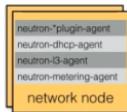
İmage servisi, blok depolama, nesne depolama ve paylaşılan dosya depolamayı işler.

Sağ tarafda göründüğü gibi Genellikle bir depolama düğümü

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

Network Node

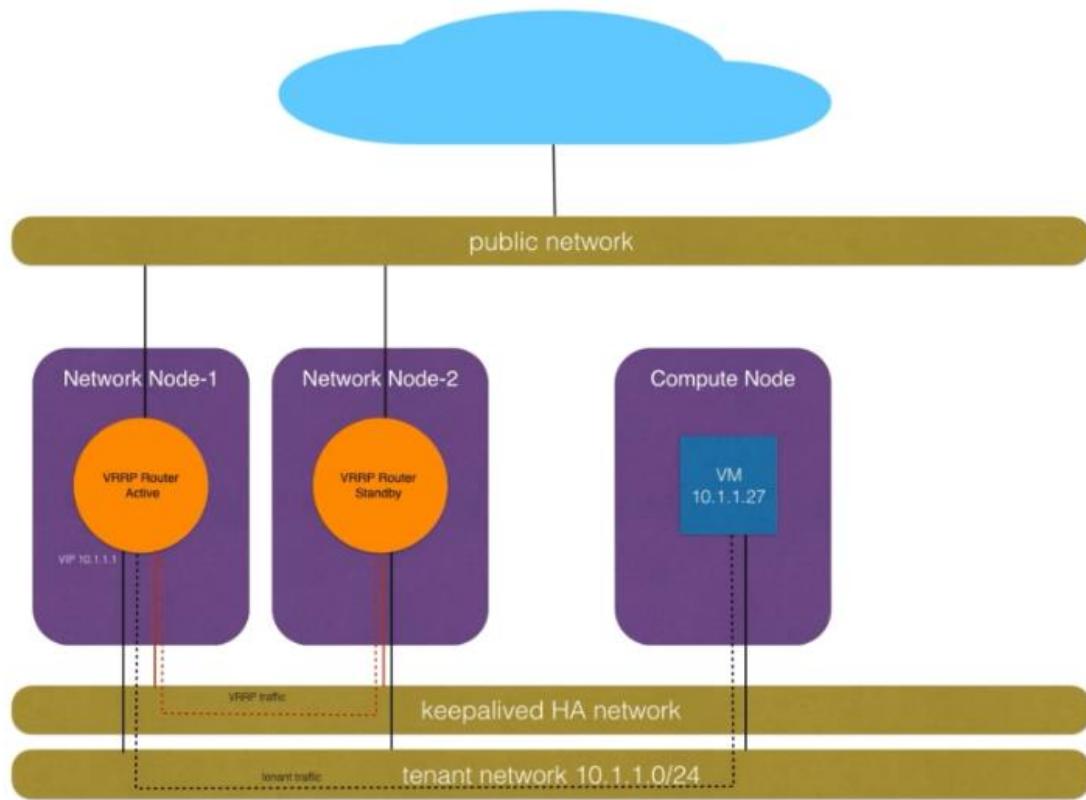
Network Node



- 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 modunuda 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 güncellemeleri gibi node yönetimi için kullanılır.
Performan gereksinip yada güvenlik politikalarına göre trafiği dahada segmentlere ayıralabiliriz, Trafik modellerine bağlı daha fazla NIC'de kullanılabilir.

Controller Node

Controller Node



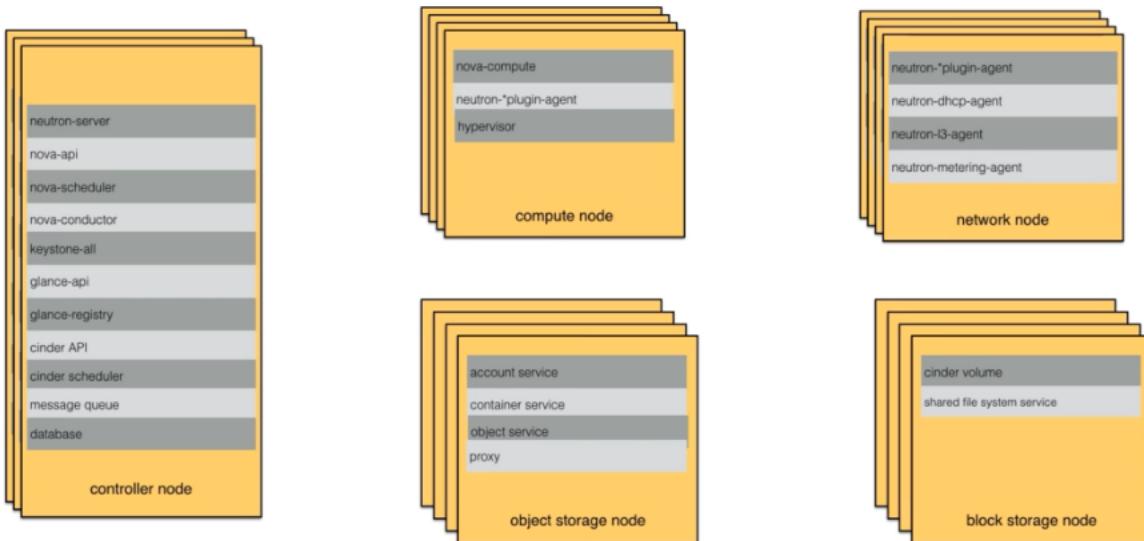
Farklı düğümlerde çalışacak bileşenler anadüğüümüzde ayrılır.

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 getirilmemişine 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 ölçelendirilebilirliğin yanı high availability ve performans imkanı sağlar.

Her bir hizmet ayırtılır ve hizmetler arasındaki tüm iletişim dinlenen API servisler aracılı 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 oluşturabiliz.

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

Pacemaker küme yiğini, Linux platformu için son teknoloji ürünü bir yüksek kullanılabilirlik ve yük dengeleme yiğinidir. Pacemaker, OpenStack altyapısını yüksek oranda kullanılır 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.

Grafiktedeki 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 dağıtıımı 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 dağıtımlı için düğümün nasıl hazırlanacağı hakkında ayrıntılar için OpenStack Juno'yu CentOS 7 / RHEL 7'ye yükleme makalesini kontrol edin.

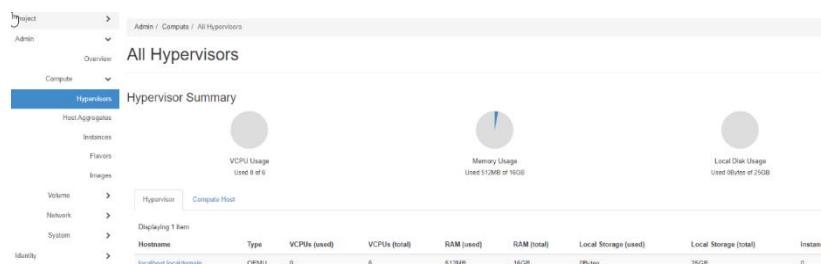
3. Mevcut OpenStack'e yeni düğüm ekleyin

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

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

OpenStack'in yeni düğümüne dağıtımlı biraz zaman alabilir:

Eklediğimiz node openstack üzerinde hypervizör'de görünecektir



Eklediğimiz ve üzerinde hangi API servisinin bulunduğu node'ları görmek için system informaın kısmında multi node üzerine yapılandırılmış openstack sağlayacımızın 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, standartdart loglama düzeyi haricinde, Trace, Debug, Info, Audit, Warning, Error ve Critical günlük kaydını yapılandırdığınızda iletiler, sadece bilirlediğiniz iletide 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.

```
[a559-6575-48d1-9b95-907045d1fadec]
[root@localhost ~]# tail -f /var/log/nova/* |grep -i error
```

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

Keysone biraz daha farklı şekilde ele alır, günlük düzeyini değiştirmek için keystone günlük dosyasını düzenlememiz ve günlüğe baktmamız gereklidir.

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

Diger servislerde görüntüde yer verilmiştir.

Aşağıdaki göründüğü 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 servisdir.
- 3.CLI'dan kullandığımız komutlar openstack'de API çağrılarına çevirir.

Openstacn CLI geçiş için openstack sunucusuna bağlanıyoruz,

Source kestonerc_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>

[Openstack – CLI Komutları](#)

source kestonerc_admin / Openstack CLI geçiş komutudur.

more kestonerc_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 "projeadi" --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](#)