Felhő megoldás Ceph elosztott adattároló platform felett

Készítette:

Szalai László,	NYME INGA	szalai@inf.nyme.hu
Major Kálmán,	NYME INGA	majork@gain.nyme.hu

2015. március.

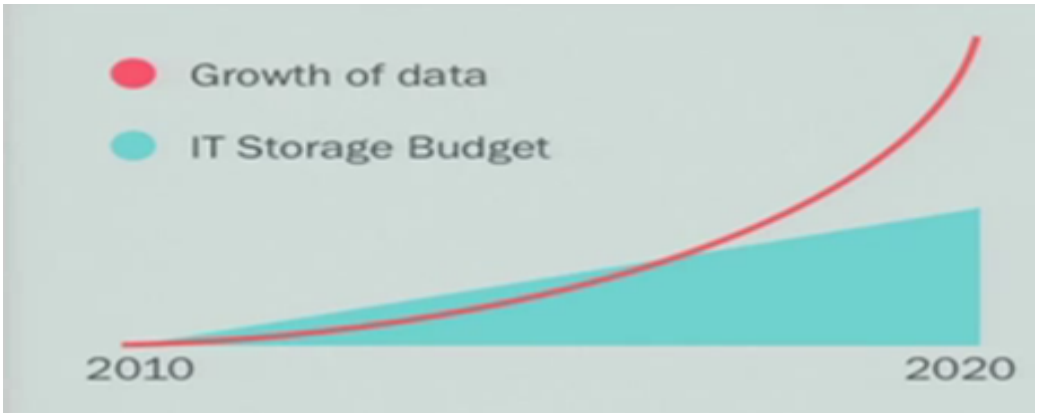
Tartalomjegyzék

1. Bevezetés

Célunk volt egy alapszintű (egygépes) ingyenes privát felhő megoldás telepítése és vizsgálata, illetve elosztott adattároló rendszer alá való felkészítése. A dokumentáció nem kimerítő jellegű, csupán tapasztalatainkat és problémáinkat írtjuk le benne.
Az egyes felhő szolgáltatások klaszteresítése nem része a dokumentációnak, így ezt a részt nem vizsgáltuk.

2. Ceph, mint elosztott adattároló megoldás

Az előrejelzések szerint 2020-ra közel 15 ZB (zetabyte) adatot fogunk tárolni. Jelenleg körülbelül 1.5 ZB adatot tárolunk.



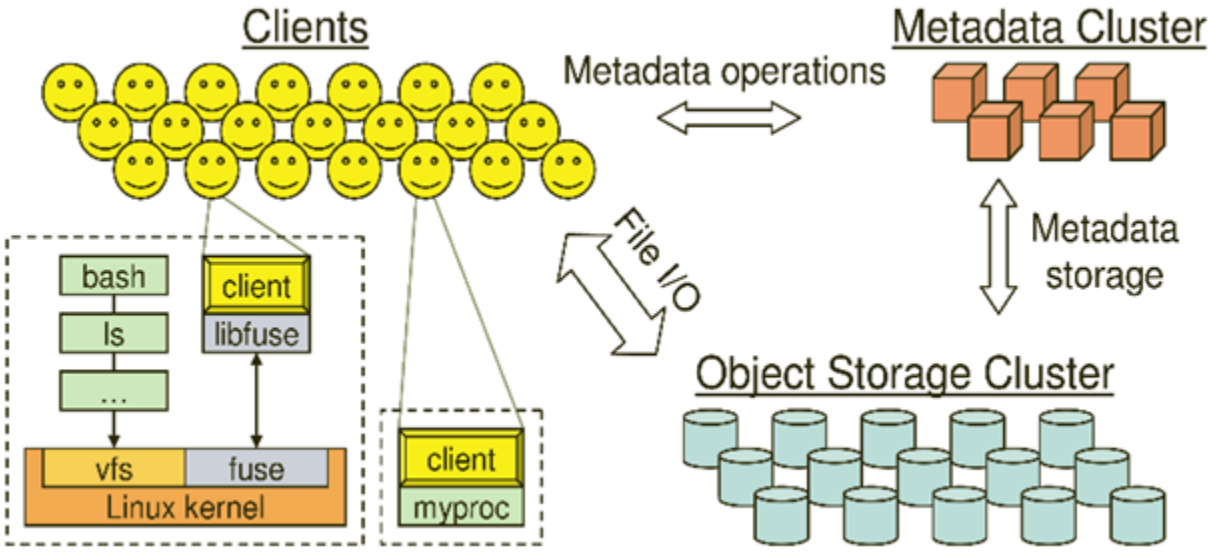
1. ábra Adatok növekedési üteme

A manapság létező rendszerek nem nagyon bővíthetőek. Az áruk folyamatosan emelkedik és egyre bonyolultabbak. Ezért még időben be kell fektetni új platformokba. Mivel egy nagyméretű adatot nehéz, lassabb mozgatni ezért egy lehetséges megoldás az, hogy kisebb darabokban többfelé másoljuk az adatainkat. Erre kínál jó megoldást a Ceph.

2. ábra Adattárolás megoldások

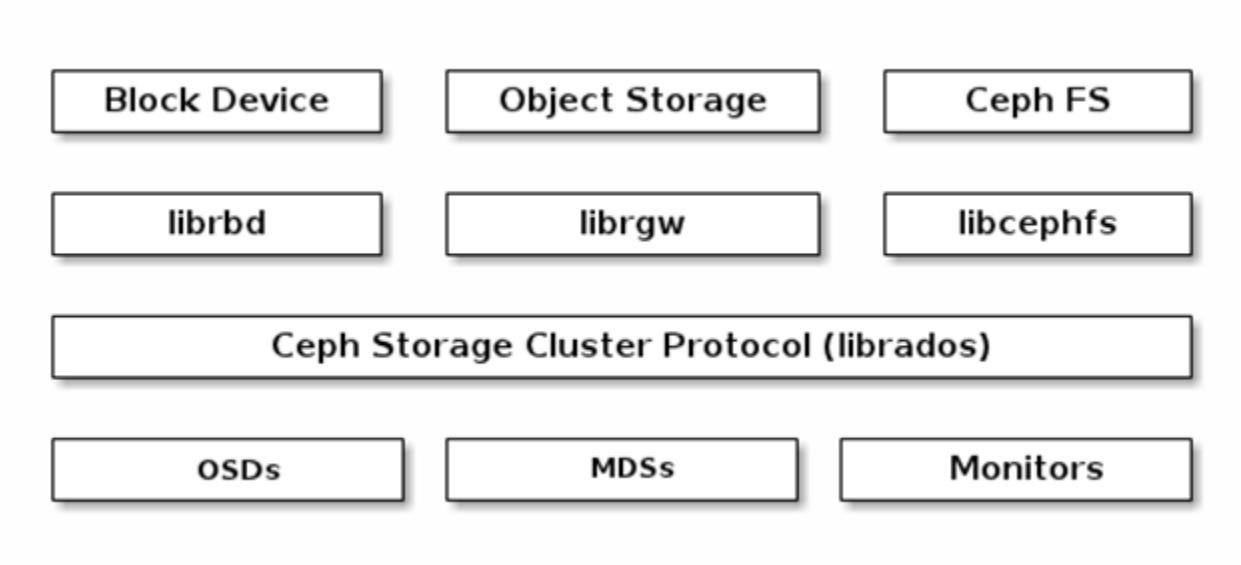
2.1 Ceph felépítése és működési elve

A Ceph egyedülálló módon kézbesít objektum, blokk és fájl storage-t egy egyesített rendszerben. Lényegében számítógépek tömegén darabolva és duplikálva tárolja adatainkat, így könnyen skálázható és lényegében failover.
Ceph Node: egyetlen számítógép vagy szerver a klaszterben. Egy Ceph klaszter több nodeból áll. Az alábbi ábrán láthatjuk a Ceph működését.



3.ábra – Ceph architektúra

A Ceph Storage klaszterek két típusú daemonból állnak, az egyik a Ceph OSD Daemon (OSD) a másik pedig a Ceph Monitor. Az OSD tárolja az adatokat objektumokként a storage node-okon. A Ceph Monitor figyeli a klaszter különböző mapjeit, beleértve a monitor map-ot,az OSD map-ot, a Placement Groupok (PG) map-eit, és a CRUSH map-ot. Az alábbi ábrán láthatjuk a Ceph felépítését.



3.ábra Ceph felépítése

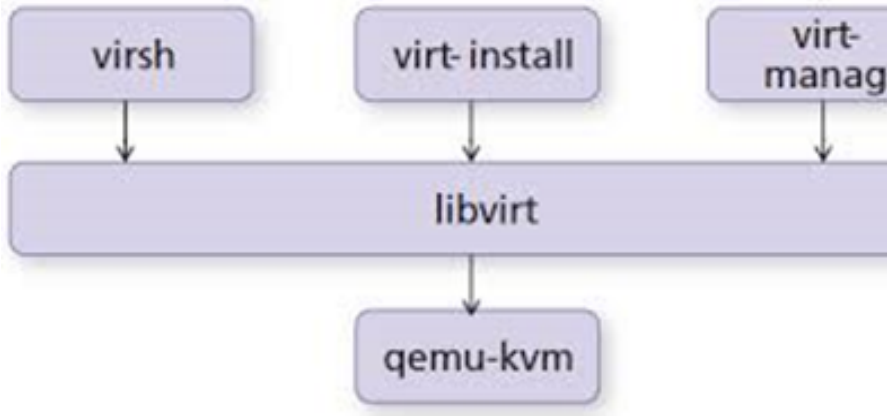
Lényegében az adataink darabolva és duplikálva egy klaszteren helyezkednek el, amely kiszolgálja az egyes klienseket. A megoldás jól skálázható és failover, node-ok kiesése esetén a storage szolgáltatás nem áll le.

3. Hypervisor / Libvirt

A libvirt csomag segítségével többféle virtualizációs technológiát összetudunk kapcsolni. Libvirttel a fejlesztők és a rendszergazdák fókuszálni tudnak a közös menedzsment keretrendszerre, a közös API-ra, illetve közös shell interfésszel (pl. virsh) a sok különböző hypervisorhoz, többek között:

- QEMU/KVM
- XEN
- LXC
- Virtualbox

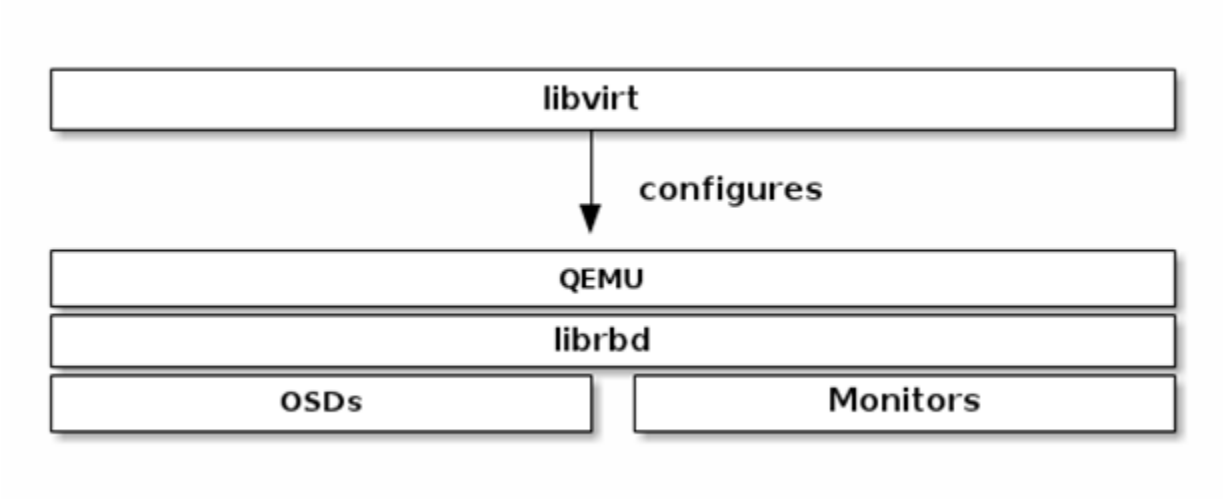
Mivel a felhő megoldások „alatt” többnyire virtualizáció van, emiatt mi is választottunk egyet. Korábban Xen alatt dolgoztunk, a mostani választás a a KVM lett, amely már a mainline kernel része a Linux-okban.



4. ábra Libvirt

3.1 Libvirt és a Ceph RBD

A Ceph blokk eszközök támogatják a Qemu/KVM-et és fordítva. A blokk eszközöket használni tudjuk szoftveresen a libvirt interfész segítségével.
Az alábbi ábrán látható, hogy a libvirt és a qemu hogyan használja a Ceph Block Device-t (librbd).

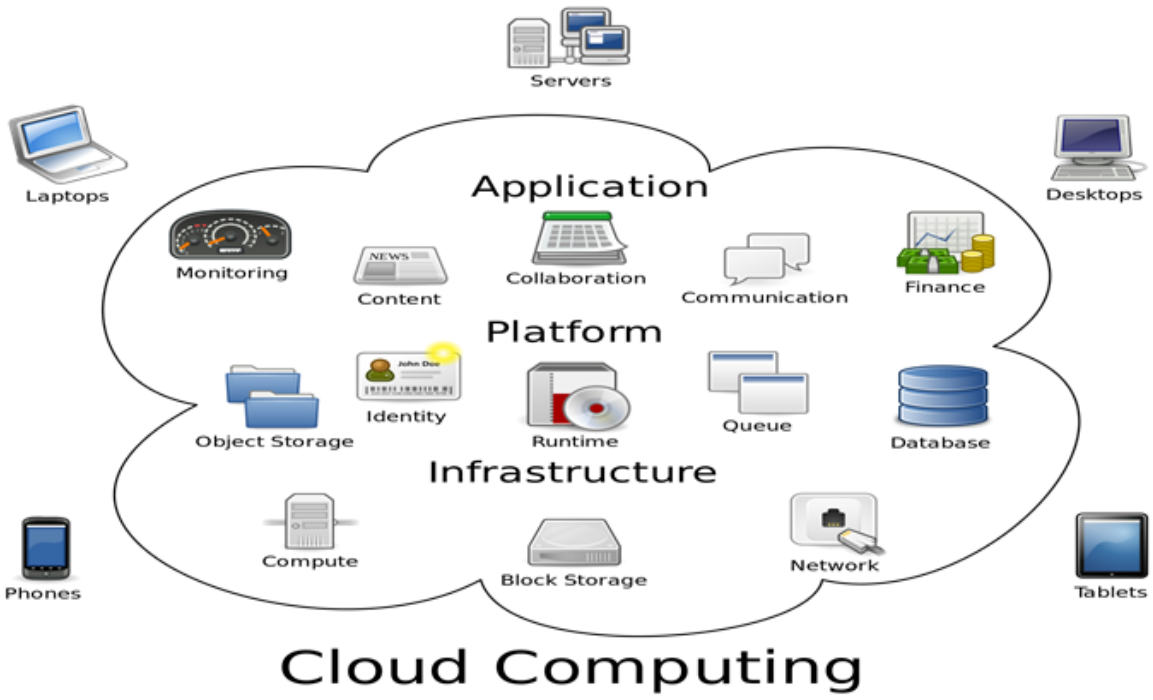


5.ábra libvirt és az rbd működése

A libvirt segítségével használni tudja a Ceph által biztosított blokk eszközt a felhő szolgáltatásunk (pl. Cloudstack, OpenStack).

4. Felhő

A felhő szolgáltatások egyre inkább kezdenek elterjedni a világon. Többféle felhő alapú szolgáltatást különböztethetünk meg, a közös bennük az, hogy a szolgáltatásokat nem egy dedikált hardveren üzemeltetik, hanem a szolgáltató eszközein elosztva, a szolgáltatás üzemeltetési részleteit a felhasználótól elrejtve. Ezeket a szolgáltatásokat a felhasználók hálózaton keresztül érhetik el, publikus felhő esetében az interneten keresztül, privát felhő esetében a helyi hálózaton vagy az interneten.



6.ábra Cloud computing

5. Tesztkörnyezet kialakítása

5.1 Teszt Ceph klaszter létrehozása

A Ceph klaszter kialakításához három darab hétköznapi számítógépet használtunk, melyekbe 500Gb winchestert és 60GB-os SSD-t raktunk. Az felhő szoftvert, illetve a libvirtet egy 16 GB memóriával rendelkező szervergépen teszteltük. A hálózat a szerver és a klaszter között gigabites, ahol a szerver gép hálózati kártyái bondingban vannak sebesség növelés és failover funkciók miatt.
A köztes switchen Etherchannel (LACP) volt létrehozva az egyes interfészeknél.

5.2 Libvirt összekapcsolása a Ceph-el

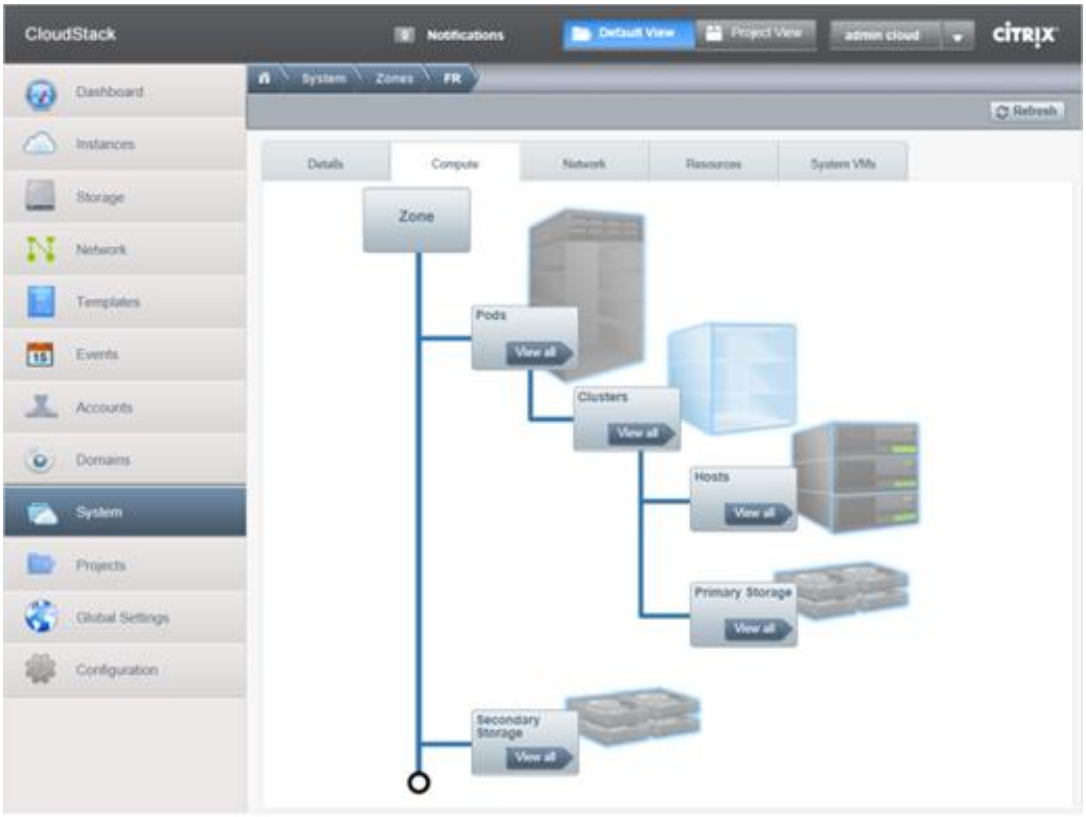
Első lépésben a szervert, melyen a felhő megoldás fut, felkészítettük a Ceph-re.

A libvirt a librbd (melya Ceph deploy során kerül a gépre) segítségével kezeli a Ceph-et.
A virsh segítségével létrehozott virtuális gépünk egyik diszkjét beállítottuk, hogy libvirttel használja a Ceph blokk eszközét, mely az alábbi xml-ben látható. A ceph klaszter monitor IP címet (vagy címeit) meg kellett adni.

```
<emulator>/usr/bin/kvm-spice</emulator>
<disk type="network" device="disk">
<driver name="qemu"/>
<source name="teszt/tesztwin7" protocol="rbd">
<host name="10.8.2.85" port="6789"/>
<host name="10.8.2.86" port="6789"/>
<host name="10.8.2.87" port="6789"/>
</source>
<target dev="vda" bus="ide"/>
<address type="drive" unit="0" bus="0" target="0" controller="0"/>
</disk>
```

6. Cloudstack

A Cloudstack egy általánosan elterjedt felhő megoldás, ami képes kezelni külső storage-ket is, például a Ceph-et. A 4.2-es verzióval keszdtük el a tesztelést, amely Ubuntu 14.04 server platform alatt futott.



Az alapfelépítés az ábrán látható, storage szinten primary és egy secondary storage adható a felhőhöz. A primary storage a virtuális gépek adatainak tárolásáért felel, a secondary storage pedig a telepítő (ISO) fileok tárolásáért. A Ceph elosztott adattárolót szerettük volna használni a Cloudstack alatt.

A primary storage nevében telepítéskor NFS adható, majd a későbbiekben tudnánk hozzá adni a Ceph-et, RBD alapokon. Sajnálatos módon sem a 4.2.x, sem a 4.4.x alatt nem sikerült működőképesen beüzemelni az RBD alapú adattárolót. A beállítások között felvehető ugyan a Ceph, sőt, sikeresen le is tudja kérdezni a Cloudstack API az adattároló kapacitását, de valami oknál fogva kivétellel elszáll a java folyamat, amikor instance létrehozásra kerülne.

A Cloudstack egyébként egy Apache/Tomcat alapú webes felületet bocsát a felhasználó rendelkezésére, ezen keresztül lehet a beállításokat eszközölni. Rengeteg probléma van a Cloudstack storage kezelésével, nálunk a primary storage csak akkor működött megfelelően, ha egygépes környezetben az NFS szerver is futott és az volt a primary kiszolgáló. Külső NFS szerverek esetében érdekes hibákat tapasztaltunk.
Az egyes VM futtató hosztok a Cloudstack Agent alkalmazás segítségével vannak kapcsolva a központi Cloudstack szerverhez, az egyes hosztokon a libvirt/KVM megoldást választottuk volna.

7. Devstack/Openstack

Az Openstack egy elterjedt felhő megoldás, amelyet sok nagy cég támogat. Megéztük, milyen lehetőségek vannak Openstack alatt a Ceph RBD kezelésére.
Mivel eme felhő megoldás alatt működött a Ceph, emiatt egy kicsit részletesebben foglalkozunk vele.

A Devstack egy OpenStack verzió, amelynél egy számítógépre települ az összes komponens.
Az alábbi komponensekből épül:
-Nova
-Cinder
-Glance
-Horizon
-Keystone

Nova
A számítási felhő vezérlő szerkezete (Az IaaS fő része), amely képes dolgozni széles körben elérhető virtualizációs technológiákkal.

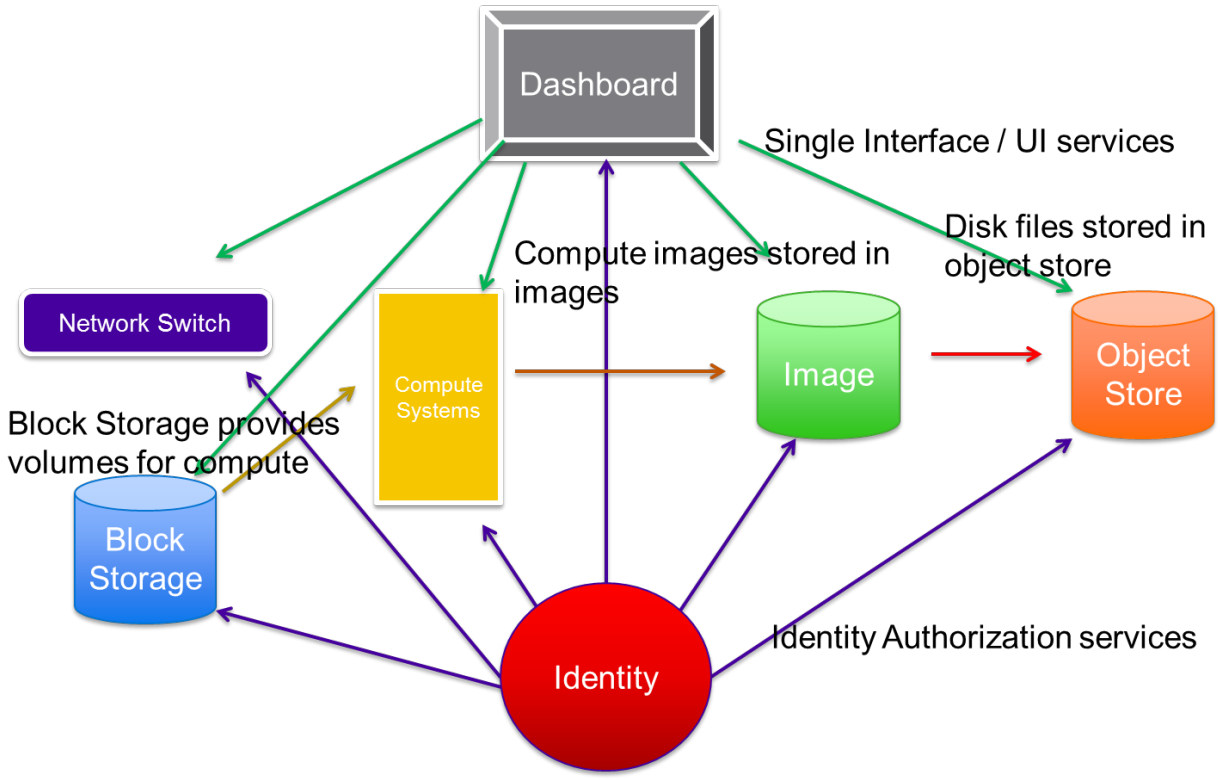
Cinder
A Cinder egy blokk storage szolgáltatás az OpenStack számára. A blokk storage rendszer kezeli a blokk eszközök csatolását, leválasztását, készítését.

Glance
Az OpenStack image szolgáltatása. Tárolja az imageket, melyeket templateként tudjuk használni. A blokk storage kötetek teljesen beépülnek a Novába, illetve a dashboard-on engedélyezi a felhő felhasználóinak, hogy menedzseljék a saját tárhelyüket, melyekre szükségük van.

Horizon

Az OpenStack grafikus webes felülete, ami biztosítja az openstack szolgáltatásokat beleértve a Novát, Switftet, Keystone-t, stb.

Keystone
A Keystone a felhasználók azonosítására szolgáló szolgáltatás.



7. ábra OpenStack API működése

7.1. Devstack felkészítése a Ceph-re

A Ceph esetében storageről van szó, ezért a Devstack több szolgáltatását is fel kellett készíteni a fogadására.
Létrehoztunk a Ceph alatt két új poolt – images, volumes – melyeken tárolni fogjuk a templateket és a köteteket.

A Glance kezeli az image-ket az Openstack alatt, ennek a felkészítése Ceph-re sikeres volt.

A /etc/glance/glance-api.conf fájlban az alábbi sorokat szúrtuk be:

Az alapértelmezett tárolási formát átírtuk rbd-re, illetve a ceph.conf elérési útvonalát megadtuk az alábbi látható módon. Megadtuk, hogy melyik poolt használj, mi az images poolt definiáltuk a klaszteren.

```
default_store = rbd

[ ... ]

# ============ RBD Store Options =============================

# Ceph configuration file path
# If using cephx authentication, this file should
# include a reference to the right keyring
# in a client. section
rbd_store_ceph_conf = /etc/ceph/ceph.conf

# RADOS pool in which images are stored
rbd_store_pool = images

# Images will be chunked into objects of this size (in megabytes).
# For best performance, this should be a power of two
rbd_store_chunk_size = 8
```

A Cinder az érdemi blokkeszköz szolgáltatás, lényegében itt tárolódnak az instance-ok nevében futó VM-ek, diszk szinten is.

A következő lépésben a cinder-t konfiguráltuk be az alábbi módon a /etc/cinder/cinder.conf-ban:

```
volume_driver=cinder.volume.drivers.rbd.RBDDriver
rbd_pool=volumes
glance_api_version=2
rbd_ceph_conf=/etc/ceph/ceph.conf
rbd_flatten_volume_from_snapshot=false
rbd_max_clone_depth=5
```

Az imageket sikeresen létre tudjuk hozni a blokk eszközünkre a beállított konfigurációval.

A Devstack esetében két lehetőségünk van arra, hogy a Ceph által biztosított blokk eszközünkről fussanak a virtuális gépeink:

- Első opció: A fapados megoldás, az instance könyvtár eredeti helyére becsatoljuk a kernel szintű RBD modul segítségével a logikai RBD kötetet, amelyet a Ceph szolgáltat. Enek vizsgálatát azért végeztük el, mert kezdetben nem működött a sztandard út.
  Ez az /opt/stack/data/nova/instances könyvtár, ez alá felmountoljuk az RBD logikai köteteket, melyek segítségével már a mi általunk definiált eszközünkről futnak a virtuális gépeink.

A jogosultságokat természetesen a stack felhasználó által olvasható és írhatóvá kell tenni.
Ez a megoldás működik, a Nova szolgáltatás a futtatás során nem tud róla, hogy alatta Ceph RBD kötet van.

- Második opció: A normál megoldás, azaz a felületen történő RBD alapú instance létrehozás és futtatás. Ehhez a Nova szolgáltatást kell felkészíteni.
A /etc/nova/nova.conf fájlban az alábbi sorokat szúrtuk be, ahol megadtuk többek között az image típusát, mely köteten és a ceph.conf elérési útvonalát:

```
[libvirt]
inject_partition = -2
live_migration_uri = qemu+ssh://stack@%s/system
use_usb_tablet = False
cpu_mode = none
virt_type = kvm
libvirt_images_type=rbd
libvirt_images_rbd_pool=volumes
libvirt_images_rbd_ceph_conf=/etc/ceph/ceph.conf
libvirt_inject_password=false
libvirt_inject_key=false
libvirt_inject_partition=-2
inject_password= True
enable_instance_password = True
```

8. Összefoglalás

A dokumentum célja az, hogy bemutassa, a Ceph adattároló milyen mértékben használható a meglévő ismert ingyenes IAAS felhő szoftver megoldásokkal. A Cloudstack és a Devstack/Openstack viszgálata során azt tapasztaltuk, hogy stabilan csak ez utóbbi képes használni a Ceph RBD tárhelyet a virtuális gépek tárolására, futtatására.
Kiaknázandó még az egyéb Openstack funkciók (failover, migráció, snapshot, stb.) használata a Ceph alatt, illetve a Ceph extra funkciók (BTRFS – CoW, deduplikáció, stb.) integrálása is. Ezekkel egy későbbi időpontban fogunk foglalkozni.