PROJEKT

Multitenantné operačné centrum kybernetickej bezpečnosti riešené ako otvorená cloudová služba s prvkami strojového učenia

Previazané s balíkom KPB1 – Návrh virt. riešenia

Previazané s výstupom
V3 – popis vybraného riešenia spĺňajúceho stanovené kritériá

OBSAH

ÚVOD

V súčasnom digitálnom prostredí predstavuje efektívne ukladanie a správa dát zásadný predpoklad technologického rozvoja a prevádzky moderných informačných systémov. Vývoj v oblasti dátových úložísk prešiel výraznými zmenami – od počiatočných zariadení s nízkou kapacitou po súčasné rozsiahle dátové infraštruktúry, spracovávajúce objemy údajov v rozsahu petabajtov a vyššie. Takýto exponenciálny nárast dát kladie zvýšené nároky na kapacitu úložných systémov, ich dostupnosť a zachovanie integrity údajov.

Tradičné centralizované architektúry vykazujú obmedzenú schopnosť reagovať na rastúce požiadavky, najmä z dôvodu fyzických a geografických limitácií. Nedostatky v oblasti škálovateľnosti a redundancie spôsobujú problémy pri zabezpečení vysokodostupných služieb v kontexte globálne distribuovaných cloudových systémov.

V tejto súvislosti sa ako efektívne riešenie javí implementácia distribuovaného úložiska, ktoré umožňuje rozloženie dát na viaceré uzly v rôznych lokalitách. Takýto prístup poskytuje vyššiu dostupnosť, zvýšenú odolnosť voči výpadkom a zlepšený výkon pri paralelnom prístupe k dátam.

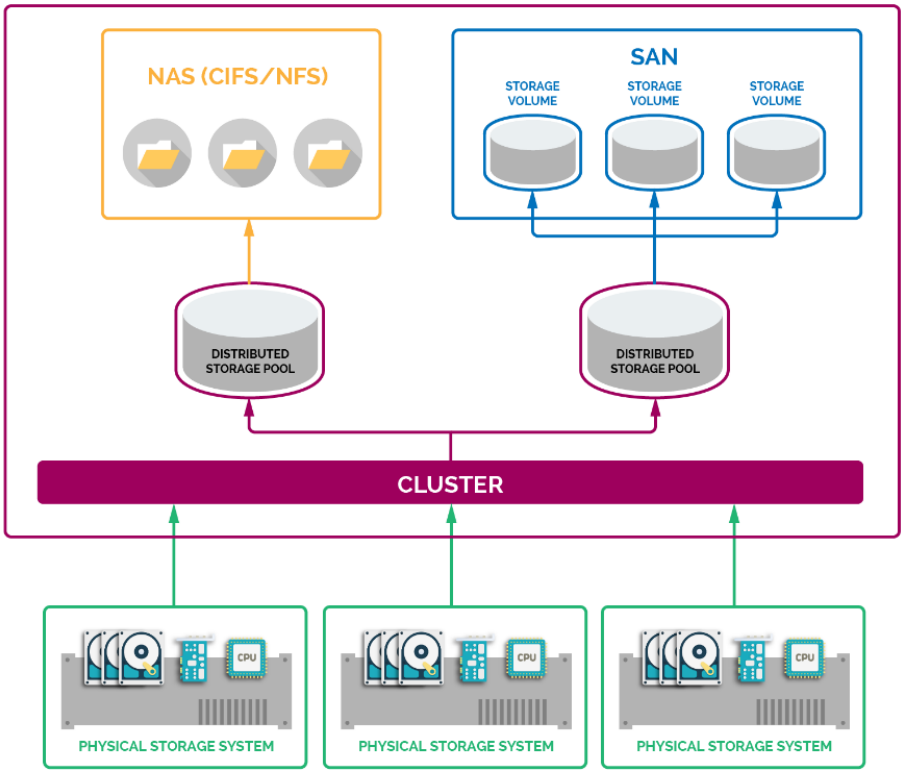
Teoretická časť sa venuje charakteristike distribuovaných úložných systémov a ich integrácii s cloudovou platformou OpenStack. Na základe analytického prehľadu možností bol identifikovaný vhodný softvér, pričom bola detailne preskúmaná jeho architektúra, funkčné moduly a základné koncepty zabezpečujúce jeho správnu funkcionalitu.

Praktická časť je zameraná na testovanie zvoleného riešenia s dôrazom na nasadenie v reálnom prostredí. Inicializačné kroky zahŕňajú manuálne nasadenie systému, nasledované využitím orchestračných nástrojov OpenStack Charms, ktoré zefektívňujú proces nasadenia a uľahčujú integráciu so samotnou cloudovou infraštruktúrou. Súčasťou analýzy sú aj konfiguračné možnosti, správa systému a úprava parametrov pre špecifické prevádzkové požiadavky prostredníctvom nástrojov Charms.

Záverečná časť obsahuje návrh metodiky a odporúčaní pre integráciu zvoleného softvérového riešenia do existujúcej cloudovej architektúry, vrátane špecifikácie požiadaviek na zabezpečenie spoľahlivosti, škálovateľnosti a výkonu distribuovaného úložiska.

1. ČO JE DISTRIBUOVANÉ ÚLOŽISKO?

V rýchlo sa meniacom digitálnom prostredí prešlo ukladanie, prístup k dátam a správa dát významnými zmenami. Jednou z najdôležitejších zmien v posledných rokoch bol prechod na distribuované systémy ukladania – **distribuované úložiská**. Tento prístup k ukladaniu údajov sa vyznačuje **decentralizáciou**, **škálovateľnosťou** a **spoľahlivosťou** a ponúka komplexné riešenie výziev, ktoré prináša exponenciálny rast digitálnych informácií.



Obrázok 1 Všeobecná schéma systému distribuovaného úložiska [1]

1.1 Princípy a výhody distribuovaného úložiska

Distribuované úložisko zabezpečuje decentralizáciu rozložením dát na viacero fyzických serverov, ktoré sa často nachádzajú na rôznych geografických miestach. Tieto servery sú prepojené cez sieť. Táto metóda je v protiklade s tradičnými modelmi ukladania, ktoré centralizujú dáta na jednom mieste. Distribuovaný model zvyšuje rýchlosť prístupu k dátam, spoľahlivosť, škálovateľnosť a odolnosť voči chybám.

Na Obrázku 1 vyššie je zobrazená základná schéma distribuovaného úložiska, kde je decentralizácia znázornená vo forme viacerých fyzických serverov („physical storage system“) zjednotených do tzv. **klastra** („cluster“). Prístup do distribuovaného úložiska je možné zabezpečiť pripojením do klastra vo forme súborového sieťového úložiska **NAS (Network Attached Storage)**, alebo blokového sieťového úložiska **SAN (Storage Area Network)** [1][2][3].

Základom distribuovaného úložiska sú jeho schopnosti bezproblémového rozširovania kapacity (škálovateľnosti), rozdeľovania a replikovania údajov na viaceré servery. Táto konfigurácia umožňuje efektívnu správu dát, paralelný prístup k dátam a ich spracovaniu, čo výrazne znižuje bottleneck-y (úzke hrdlá). Rozdelenie dát alebo „**sharding**“ je rozdelenie dát na menšie segmenty, ktoré sú distribuované do rôznych serverov úložiska. Okrem toho stratégie replikácie zabezpečujú, že sa v rôznych serveroch udržiava viac kópií uložených dát, čo poskytuje robustnú vrstvu ochrany a dostupnosti údajov, čím sa zabezpečuje spoľahlivosť. Táto replikácia zaručuje, že systém má prístup k dátam z iných nepoškodených serverov aj v prípade zlyhania jedného alebo viacerých serverov. Faktor replikácie môžu nakonfigurovať administrátori, aby určili počet kópií dát, ktoré systém udržiava. Vyššie faktory replikácie zvyšujú bezpečnosť dát, ale vyžadujú aj viac úložného priestoru [3][4].

1.2 Využitie distribuovaného úložiska

Distribuované úložisko má široké uplatnenie v rôznych odvetviach vrátane médií, zdravotníctva a analýzy veľkých objemov dát. Umožňuje efektívnu manipuláciu s rozsiahlymi knižnicami multimediálneho obsahu, bezpečné ukladanie citlivých údajov pacientov a správu veľkých súborov dát na analytické spracovanie [3][5].

Cloudové platformy vo veľkej miere využívajú distribuované úložisko vďaka jeho škálovateľnosti a odolnosti voči chybám. **Amazon S3** je najlepším príkladom distribuovaného úložiska v praxi, ktoré ponúka používateľom vysoko dostupné a bezpečné riešenia úložísk. Distribuované úložisko je obzvlášť užitočné pri riešení rozsiahlych potrieb ukladania a spracovania dát a podporuje širokú škálu aplikácií. S rastúcim objemom dát bude význam distribuovaného

ukladania ešte dôležitejší, čo povedie k pokroku a vylepšeniam v technológiách správy dát [5][6].

1.3 Dôvody nasadenia distribuovaného úložiska

Dôvody nasadenia distribuovaného úložiska pre cloud prakticky kopírujú všetky už vyššie zmienené výhody distribuovaného úložiska. Cloud je nasadený nad platformou OpenStack. Hlavnou výhodou cloudových platforiem je ich všestrannosť, možnosti migrovania virtuálnych zariadení a ich dostupnosť v prípade výpadku zariadenia v klastri. Tieto vlastnosti sú však značne limitované v prípade, že úložisko pre každé dané zariadenie je riešené lokálne. To má za následok, že v prípade prístupu viacerých klientov sa zvyšujú nároky na daný konkrétny server, čím sa okrem pomalšieho spracovania požiadaviek časom zvyšuje riziko zlyhania daného servera. V prípade takého zlyhania vďaka lokálnemu ukladaniu dát, je strata dát a znemožnenie poskytovania služieb, ktoré sa na danom serveri nachádzali, veľmi pravdepodobné.

Nasadením distribuovaného úložiska do cloudu sa týmto riziko straty dát alebo prerušenia poskytovania služieb výrazne znižuje. Práve vlastnosti distribuovaného úložiska zabezpečujú paralelný prístup k dátam a ich výraznú redundanciu. Úložisko už nie je lokálne a vďaka replikácií dát na viaceré serveri nedôjde k strate dát v prípade ich zlyhania. Jednoduchá škálovateľnosť zabezpečuje, že prípadné rozširovanie katedrového klastra je výrazne jednoduchšie. V neposlednom rade sa zjednodušuje aj spravovanie a monitorovanie úložiska. Softvéry distribuovaného úložiska poskytujú nástroje na detailnú správu a monitorovanie, ktoré v takom rozsahu v OpenStack-u nie je možné.

2. VÝBER SOFTVÉRU DISTRIBUOVANÉHO ÚLOŽISKA PRE OPENSTACK

Výber vhodného softvéru pre distribuované úložisko v OpenStack-u je rozhodujúci z niekoľkých dôvodov. Správny backend úložiska je nevyhnutný pre zabezpečenie optimálneho výkonu klastra OpenStack. Ak úložisko nedokáže zabezpečiť potrebné vstupno-výstupné operácie pre vyžadované zaťaženie klastra, vedie to k problémom s výkonom, ktoré priamo ovplyvňujú fungovanie aplikácií. Dobre zvolené riešenie úložiska zvyšuje efektivitu prevádzkovania úložiska, a tým aj poskytovaných služieb. Taktiež môže zjednodušiť procesy údržby, zálohovania a riešenia problémov, zatiaľ čo nesprávne zvolené riešenie môže zvýšiť pracovné zaťaženie administrátorského tímu [7].

Rovnako ako poskytnutie potrebného výpočtového výkonu, zabezpečenie spoľahlivosti riešenia úložiska priamo súvisí so spoľahlivosťou celého klastra. Preto musí byť spoľahlivosť dát najvyššou prioritou pri navrhovaní privátneho cloudu. Výber nesprávneho riešenia úložiska môže tento aspekt ohroziť. Okrem toho ďalším kľúčovým faktorom je nákladová efektívnosť, ktorú treba zohľadniť. Riešenie úložiska by malo vyvažovať atribúty výkonu a nákladov. Plytvanie cennými zdrojmi môže byť dôsledkom riešenia, ktoré zvyšuje celkové náklady bez toho, aby primerane spĺňalo požiadavky na výpočtový výkon určených v príslušných SLA (Service Level Agreement) [7][8].

Ďalším dôležitým aspektom pri výbere softvéru je pochopenie povahy údajov a spôsob, akým je potrebné k nim pristupovať alebo ich distribuovať. Napríklad štruktúrované a neštruktúrované údaje si vyžadujú rôzne riešenia ukladania, čo výrazne ovplyvňuje prístup k údajom a ich ukladanie. Okrem toho je nevyhnutné zvážiť škálovateľnosť a potenciál budúceho rastu riešenia úložiska. Vybrané úložisko by malo nielen vyhovovať súčasným potrebám, ale malo by byť aj škálovateľné, aby vyhovovalo budúcemu rastu požiadaviek na kapacitu a výkon [8].

Výber vhodného softvéru distribuovaného úložiska pre OpenStack si vyžaduje strategický rozhodovací proces, ktorý ovplyvňuje výkon, spoľahlivosť, prevádzkovú a nákladovú efektívnosť celej cloudovej infraštruktúry. Je dôležité zvážiť špecifické potreby a okolnosti každého nasadenia OpenStack-u.

2.1 Kritéria pre výber softvéru

Vypracovanie špecifického súboru kritérií pre výber softvéru je nevyhnutným krokom pre zabezpečenie, aby nami vybraný softvér zodpovedal našim stanoveným cieľom. To nám umožní kvantifikovať a porovnať vlastnosti a schopnosti rôznych softvérov.

Vybraný softvér musí spĺňať nasledujúce kritériá:

- voľne dostupný na použitie
- podpora objektového, blokového i súborového typu úložiska
- škálovateľnosť
- zálohovateľnosť, vhodná redundancia uložených dát
- kompatibilita s platformou OpenStack
- kompatibilita s nástrojmi OpenStack Charms

Tieto kritéria tvoria základný rámec pre výber softvéru s ohľadom na požiadavky, infraštruktúru a architektúru privátneho cloudu postaveného nad OpenStackom.

2.2 Dostupné možnosti

V prípade softvéru pre OpenStack, sú tri významné možnosti v tejto oblasti: **Ceph**, **GlusterFS** a **MinIO**, pričom každá z nich má svoje výhody i nevýhody.

Ceph, je široko rozšírená voľba, ktorá vyniká svojou bezproblémovou integráciou s OpenStack-om a ponúka robustné blokové, objektové a súborové úložisko v jednotnom systéme, vďaka čomu je vysoko škálovateľné a odolné. Podobne aj samoregeneračné a samosprávne schopnosti softvéru Ceph výrazne znižujú administratívnu záťaž [9][10].

GlusterFS je obľúbenou možnosťou, vďaka svojej jednoduchosti a použiteľnosti, pričom poskytuje škálovateľný sieťový súborový systém vhodný pre úlohy náročné na dáta. Jeho schopnosť integrovať sa s OpenStack-om

prostredníctvom modulov **Cinder** a **Manila** na ukladanie blokov, resp. súborov z neho robí univerzálnu voľbu pre rôznorodé pracovné zaťaženia [10][11][12].

MinIO sa špecializuje na vysoko výkonné objektové úložisko kompatibilné so systémom S3. Je to výpočtovo nenáročné, ale výkonné riešenie, ktoré je ideálne pre prostredia uprednostňujúce jednoduchosť a efektívnosť pri spracovaní veľkých objemov neštruktúrovaných údajov. Jeho kompatibilita s rozhraním S3 API umožňuje bezproblémovú integráciu do rôznych cloudových aplikácií, vrátane platformy OpenStack [13].

Každá z možností, Ceph, GlusterFS a MinIO predstavuje jedinečný súbor funkcií. Sumárne zobrazenie daných kritérií a ich naplnenie jednotlivých softvérov je zobrazené v Tabuľke 1 nižšie.

Kritériá	Ceph	GlusterFS	MinIO
Free/OpenSource	Áno/Áno	Áno/Áno	Bez podpory/Áno
Podporovaný typ úložiska	Objektové, Blokové, Súborové	Objektové, Blokové, Súborové	Objektové
Škálovateľnosť	Áno	Áno	Áno
Podpora API	RADOS, S3	S3	S3
Snapshot/Klonovanie	Áno	Nie	Neaplikovateľné
OpenStack Driver	Áno	Nie	Nie
Podpora Charms	Plná	Nie	Čiastočná

Tabuľka 1 Porovnanie kritérií softvérov Ceph, GlusterFS a MinIO

2.3 Výber softvéru

Po dôkladnom vyhodnotení kritérií a dostupných možností sme sa rozhodli zvoliť si ako softvér pre distribuované úložisko **Ceph**. Jedným z hlavných dôvodov tohto výberu je, že Ceph je open-source platforma, čo zodpovedá našim preferenciám bezplatných a komunitou podporovaných riešení. Okrem toho je významnou výhodou škálovateľnosť, pretože dokáže efektívne spracovávať dáta v až petabajtovom rozsahu, čím zabezpečuje, že infraštruktúra úložiska môže rásť spolu s budúcimi možnými potrebami bez toho, aby došlo k výraznému zníženiu výkonu. Dôležitým faktorom pri našom rozhodovaní boli aj integrované funkcie redundancie softvéru Ceph. Tieto funkcie sú nevyhnutné na zachovanie

integrity a dostupnosti údajov, najmä v distribuovanom prostredí. Schopnosť bezproblémovo replikovať a obnovovať dáta poskytuje odolnosť, ktorá je pre poskytovanie cloudových služieb potrebná. Ďalším dôvodom výberu Ceph-u je jeho natívna integrácia s platformou OpenStack, ktorá je kľúčovou súčasťou katedrovej cloudovej infraštruktúry. Ceph podporuje nástroje **OpenStack Charms**, ktoré uľahčujú správu a nasadenie v prostredí OpenStack.

V prípade ďalších potenciálnych možností ako napr. GlusterFS, nie je v súčasnosti podporovaný platformou OpenStack, od vydania jej verzie Ocata v roku 2017. Neexistencia podpory pre GlusterFS môže mať za následok problémy s integráciou a spôsobiť potenciálnu nekompatibilitu. Navyše nástroje OpenStack Charms nepodporujú GlusterFS, čo je jedným z najdôležitejších kritérií pre výber softvéru. GlusterFS aj MinIO majú svoje silné stránky, ale pre použitie na katedrovom cloude nie sú najvhodnejšie. Napriek tomu, že sa návrh MinIO zameriava predovšetkým na vysoko výkonné objektové úložisko kompatibilné so systémom S3, neobsahuje vo svojej podstate úroveň redundancie, ktorá je požadovaná. Toto obmedzenie by mohlo predstavovať riziko pre odolnosť a dostupnosť dát. Taktiež kompatibilita s nástrojmi OpenStack Charms je veľmi diskutabilná, nakoľko oficiálne dokumentácie neobsahujú túto variantu.

3. CEPH A ARCHITEKTÚRA SOFTVÉRU CEPH

Nasledujúca kapitola sa venuje softvéru **Ceph** a jeho architektúre. Kapitola predstavuje, čo je Ceph, základnú architektúru softvéru Ceph s dôrazom na jej kľúčové komponenty, technológiu **RADOS** a algoritmus **CRUSH**, ktoré sú základom **efektívnosti** a **škálovateľnosti** [14].

3.1 Softvér Ceph

Ceph je open-source systém distribuovaného úložiska, ktorý spravuje nadácia **Ceph Foundation** v rámci nadácie **Linux Foundation**. Príspevky širokého spektra organizácií vrátane spoločností **Red Hat** a **IBM** zabezpečujú jeho neustály vývoj a zlepšovanie. Poskytuje škálovateľné a jednotné úložisko pre blokové, objektové a súborové údaje a je navrhnutý tak, aby fungoval na komoditnom hardvéri, čo z neho robí nákladovo efektívne a voči chybám odolné riešenie pre rôzne potreby ukladania dát. Od januára 2023 je firma IBM zodpovedná za vývoj, údržbu a podporu Ceph-u, ktorý bol premenovaný na **IBM Storage Ceph** pre podnikové – „**enterprise**“ použitie. Táto verzia obsahuje podnikové funkcie a podporné služby s dohodami SLA a je nasadená pomocou linuxových kontajnerov pre jednoduchšiu správu [15][16].

Každý rok, s cieľovým dátumom v marci, sa uvádza nová stabilná verzia, ktorá je pomenovaná v abecednom poradí. Čísla verzií sa používajú vo formáte **x.y.z,** kde "x" označuje cyklus vydania (poradie písmena v abecede), "y" označuje typ vydania (0 pre vývoj, 1 pre kandidátske vydania, 2 pre stabilné/opravné vydania) a "z" označuje verzie opráv. Stabilné verzie sa aktualizujú každých 4 až 6 týždňov a sú podporované **2 roky** [17].

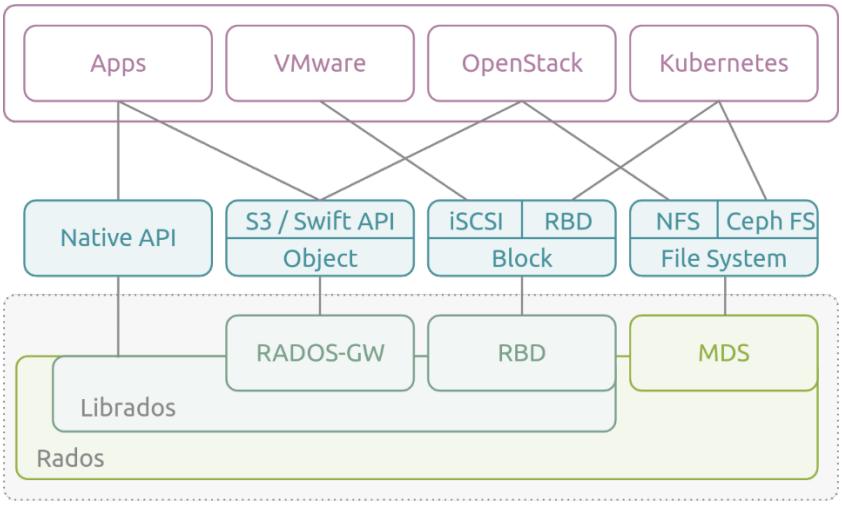
Podľa spoločnosti **HG Insights**, používajú Ceph ako úložisko aj významné firmy akými sú **NVIDIA**, logistický gigant **Maersk Line**, alebo telekomunikačná firma **Ericsson** [18].

3.2 Kľúčové komponenty softvéru Ceph

Architektúra Ceph sa skladá z niekoľkých kľúčových komponentov, z ktorých každý zohráva významnú úlohu vo funkčnosti a výkonnosti systému. Tieto komponenty spolupracujú, aby poskytli riešenie úložiska, ktoré možno ľahko škálovať a prispôsobovať potrebám používateľa, od terabajtov po exabajty a ďalej bez akéhokoľvek narušenia. V nasledujúcich podkapitolách vysvetlíme ako jednotlivé komponenty pracujú, akú majú úlohu v **klastri** a aké poskytujú výhody [14].

3.2.1 Ceph klaster

Základom celého systému Ceph je **klaster**. Klaster sa skladá z **node**-ov, ktoré tvoria jednotlivé zariadenia – servery, na ktorých je nainštalovaný softvér Ceph. Na node-och sa následne nasadzujú jednotlivé **komponenty** klastra. Vytvorenie klastra Ceph je zložitý proces, ktorý integruje rôzne komponenty a vytvára robustné riešenie úložiska. Sú nimi **Ceph Monitor, Ceph Manager, OSD a MDS**, pričom každý z nich zohráva v architektúre klastra jedinečnú úlohu. Orchestrácia týchto prvkov umožňuje efektívnu správu blokového úložiska, súborového systému a objektového úložiska v rámci distribuovaného sieťového prostredia. Spolupráca týchto komponentov vytvára komplexný a prispôsobivý systém ukladania dát, ktorý dokáže splniť zložité požiadavky na ukladanie dát v súčasných dátových prostrediach [14].

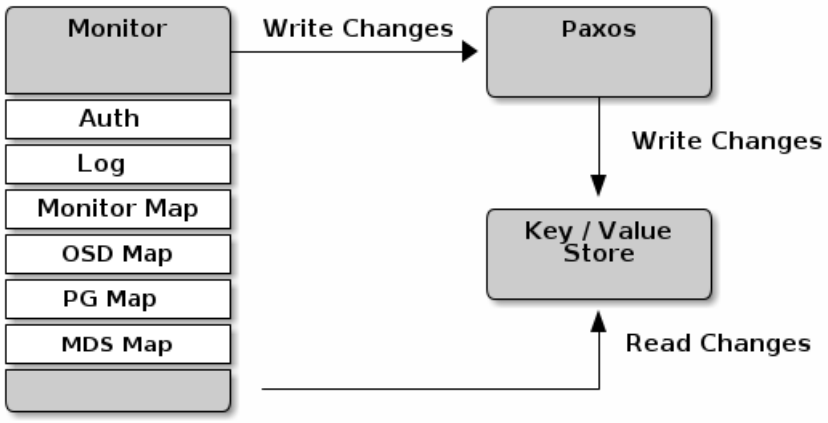


Obrázok 2 Architektúra klastra Ceph [19]

Obrázok 2 zobrazuje architektúru klastra Ceph, jeho komponenty a flexibilitu využitia. Poskytuje široké možnosti pre komunikáciu s klientami, či už vo forme **API rozhraní,** alebo pomocou **gateway**-ov, ktorými poskytuje úložisko pre iné platformy, akými sú VMware, Kubernetes alebo práve OpenStack.

3.2.2 Ceph Monitor

Ceph Monitor (označovaný aj ako „*ceph-mon*“) je dôležitou súčasťou pri udržiavaní funkčného stavu a stability klastra Ceph. Jeho hlavnou funkciou je poskytovať spoľahlivý a aktuálny pohľad na stav klastra a udržiavať mapy klastra. Konkrétne sú to **mapy monitora**, **administratívne mapy**, **mapy OSD**, **mapy MDS** a **mapy CRUSH**, ktoré sú nevyhnutné pre koordináciu a efektívnu správu klastra. Vďaka tomu sa klaster môže prispôsobiť zmenám, napríklad pridaniu alebo odstráneniu node-ov bez prerušenia poskytovania služieb. Monitory taktiež zohrávajú kľúčovú úlohu pri správe klastra, vrátane autentifikácie a autorizácie klientov a démonov, čím sa zvyšuje jeho bezpečnosť [14][20][21].



Obrázok 3 Zobrazenie úloh Ceph monitora a rozhodovacieho procesu [21]

Obrázok 3 zobrazuje jednotlivé úlohy monitora, ktorými zabezpečuje hladký chod klastra s pomocou konsenzuálneho algoritmu **Paxos**. Tento algoritmus umožňuje monitorom dohodnúť sa na stave klastra a prijímať rozhodnutia spoločne, čím sa zvyšuje odolnosť voči poruchám, a tým aj spoľahlivosť klastra. Práve konsenzuálnosť algoritmu zaručuje, že klaster dokáže odolávať zlyhaniam a zachováva integritu aj dostupnosť dát tým, že pre prijatie rozhodnutia je potrebné, aby viac ako polovica monitorov bola v zhode [14][21].

Využitie viacerých monitorov zaručuje, že klaster zostane funkčný aj v prípade zlyhania takmer polovice celkového počtu monitorov. Monitory uľahčujú škálovateľnosť klastra, čo umožňuje prijať viac dát a klientov bez výrazného zvýšenia nákladov na správu. Mechanizmus konsenzu umožňuje klastru udržiavať konzistentný stav aj v prípade rozdelenia siete, alebo zlyhania hardvéru. Dohliadajú na autentifikačný systém **cephx**, ktorý overuje používateľov a démonov, a chráni pred útokmi napr. typu man-in-the-middle [14][20].

3.2.3 Ceph Manager

Ceph Manager, alebo „*ceph-mgr*“, je jedným z kľúčových komponentov, ktorý poskytuje lepšie možnosti monitorovania a správy klastra. Od verzie 12.x sa stal povinnou súčasťou pre bežnú prevádzku klastra. Démon pracuje popri démonoch monitora, čím poskytuje ďalšie funkcie monitorovania spolu s rozhraniami pre externé systémy monitorovania a správy. Nevyžaduje žiadnu ďalšiu konfiguráciu okrem zabezpečenia svojej prevádzky. Na nastavenie démonov ceph-mgr na monitorovacích node-och sa zvyčajne používajú nástroje na nasadenie, ako napríklad „*ceph-ansible*“, alebo „*cephadm*“, hoci sa nemusia nevyhnutne nachádzať na rovnakých node-och ako monitory [22].

Architektúra manager-a umožňuje rozšírenie jeho funkcií prostredníctvom modulov, ako sú **dashboard**, **prometheus** a **balancer**. Modul dashboard poskytuje zabudované webové rozhranie pre správu, prometheus ponúka možnosti monitorovania a balancer optimalizuje rozdelenie **Placement Groups (PGs)** medzi OSD démony. Okrem modulov pre prometheus, manager poskytuje aj moduly pre Zabbix, posielanie upozornení, moduly pre orchestráciu a RESTful API rozhranie [22].

Medzi najväčšie výhody manager-a pre klaster Ceph patrí práve modularita a zjednodušené rozhranie pre správu klastra. Tento modulárny prístup umožňuje administrátorom prispôsobiť funkcie pre správu svojim špecifickým potrebám, čím sa optimalizuje výkon a spoľahlivosť klastra ako celku [22].

3.2.4 OSD démony

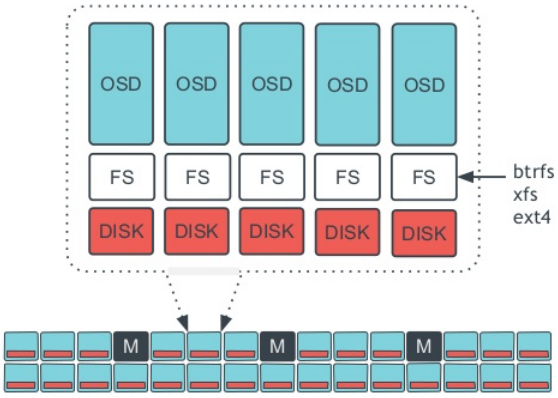
Ceph **Object Storage Daemons** (**OSDs**) – **OSD démony** sú najdôlejžitejšími komponentmi architektúry a prevádzky klastra Ceph. Slúžia ako základ pre ukladanie dát, replikáciu, obnovu a monitorovanie stavu klastra. OSD démony alebo aj „*ceph-osd*“, ukladajú dáta v rámci logických **pool**-ov, zabezpečujú dostupnosť dát a odolnosť systému prostredníctvom svojich úloh pri vyvažovaní dát a kontrole stavu. Sú nevyhnutné pre dosiahnutie vysokej dostupnosti a odolnosti voči chybám, čo uľahčuje schopnosť efektívne spracovať obrovské objemy dát [9][14].

Každý fyzický disk alebo logická partícia určená pre ukladanie dát má zvyčajne vlastný OSD démon. Ukladanie a replikácia dát v klastri sa realizuje práve pomocou OSD démonov, ktoré ukladajú dáta ako **objekty** v rámci **PGs**. Na zabezpečenie odolnosti a dostupnosti dát používa Ceph stratégiu replikácie, ktorá zahŕňa replikáciu objektov vo viacerých OSD démonoch. Ich počet je špecifikovaný **faktorom replikácie**, ktorého hodnota je uvedená v konfigurácii. Distribúciu a správu údajov OSD node-ov efektívne rieši algoritmus **CRUSH**. CRUSH umožňuje dynamické umiestňovanie objektov a dokáže sa prispôsobiť zmenám v topológii klastra bez toho, aby sa vyžadoval manuálny zásah [14][23].

OSD démony obsahujú aj funkcie monitorovania. V pravidelných intervaloch si posielajú tzv. „***heartbeat*“** správy navzájom aj s monitorom, aby oznámili svoj súčasný stav. Tento monitorovací mechanizmus pomáha klastru rýchlo zistiť a reagovať na prípadné zlyhania OSD démonov. V prípade zlyhania OSD démona Ceph automaticky spustí procesy obnovy dát, ktoré majú za úlohu replikovať dotknuté dáta z iných OSD démonov. V prípade pridania nových OSD démonov do klastra, Ceph vyvažuje rozloženie dát s cieľom optimalizovať využitie úložiska a jeho výkon [14][23].

Implementácia OSD démonov v klastri poskytuje množstvo výhod v oblasti škálovateľnosti. Pomocou OSD démonov architektúra Ceph umožňuje plynulé navyšovanie kapacity a výkonu úložiska. Pridanie ďalších diskov do klastra zvyšuje celkovú kapacitu úložiska a rozdeľuje pracovné zaťaženie na väčší súbor zdrojov, čím sa zabezpečuje vysoká dostupnosť. Navyše replikácia dát medzi jednotlivými jednotkami OSD démonov zabezpečuje, že žiadny jednotlivý bod zlyhania nemôže ohroziť dostupnosť dát. Z tohoto dôvodu má klaster schopnosť

pokračovať vo fungovaní aj v prípade zlyhania niektorých OSD démonov. Ceph využíva replikáciu alebo „erasure coding“ pre zvýšenie odolnosti dát a ochranu pred stratou dát v dôsledku zlyhania hardvéru. Disponujú samoregeneračnými schopnosťami, ktoré umožňujú automatickú obnovu po zlyhaní, redistribúciou dát s cieľom zachovať redundanciu a výkon bez nutnosti zásahu správcu [14][23].



Obrázok 4 Zobrazenie OSD démonov na node serveri [24]

3.2.5 MDS a Ceph File System

Metadata Server (MDS) je kľúčovou súčasťou **distribuovaného súborového systému CephFS (Ceph File System)**. Jeho hlavnou úlohou je spravovať metadáta pre operácie so súbormi, ako je otváranie, čítanie, zápis a výpis súborov. Na rozdiel od tradičných systémov ukladania dát, ktoré ukladajú metadáta súborov v rovnakých node-och ako dáta, Ceph používa na správu metadát vyhradené node-y **MDS**. Táto voľba zlepšuje výkonnosť operácií s metadátami tým, že umožňuje node-om úložiska (OSD) sústrediť sa predovšetkým na ukladanie dát bez dodatočnej réžie správy metadát [25].

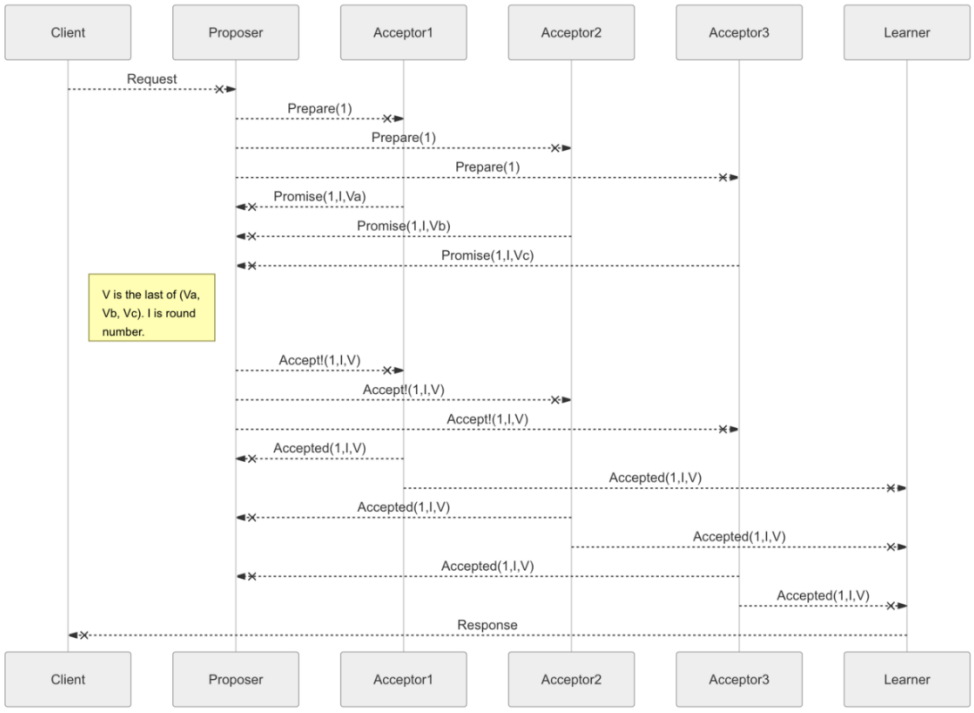
Je dôležité poznamenať, že server MDS je relevantný len pri používaní zdieľaného súborového systému, ako napr. **CephFS** alebo **NFS (Network File System)**. CephFS je distribuovaný súborový systém, ktorý poskytuje vrstvu **file storage** nad klastrom Ceph. MDS je špeciálne navrhnutý na správu metadát pre systém CephFS, čo z neho robí kritický prvok pre nasadenia, ktoré vyžadujú distribuovaný súborový systém. Pre iné typy úložísk Ceph, ako je blokové úložisko (RBD) alebo objektové úložisko (RGW), MDS servery nie sú potrebné, preto sa bez systému CephFS nevyužívajú [14][25].

Servery MDS spravujú „***namespace***“ CephFS a koordinujú prístup k samotnému úložisku poskytovaného OSD démonmi. Spracúvajú požiadavky klientov týkajúce sa metadát súborov, čím umožňujú efektívne operácie súborového systému v distribuovanom prostredí. Metadáta sú uložené vo vyhradenej časti v rámci klastra Ceph, čo umožňuje rýchly prístup a správu. Toto nastavenie umožňuje systému Ceph poskytovať klientom rozhranie súborového systému, ktoré je v súlade so štandardmi **POSIX** a podporovať širokú škálu operácií so súbormi v distribuovanom systéme [26].

Oddelenie ukladania metadát a dát umožňuje klastru s využitím CephFS výrazne optimalizovať operácie založené na súboroch. Oddeleným spracovaním metadát sa znižuje zaťaženie OSD diskov, čo vedie k lepšiemu výkonu a škálovateľnosti operácií s údajmi aj metadátami. Architektúra Ceph-u umožňuje rozširovanie správy metadát pridaním ďalších node-ov MDS, čo je nevyhnutné pri rozsiahlych nasadeniach. Táto schopnosť zabezpečuje, že systém dokáže efektívne spracovať rastúci počet súborov a adresárov. CephFS taktiež podporuje vysokú dostupnosť prostredníctvom nasadenia viacerých serverov MDS v konfigurácii **active/standby**, čím zabezpečuje nepretržitý prístup k súborom a minimalizuje výpadky v prípade zlyhania MDS. Servery MDS zvyšujú efektívnosť operácií s metadátami prostredníctvom techník, ako je **caching** a **journaling**. Journaling je mimoriadne dôležitý na zabezpečenie konzistentnosti a rýchlej obnovy, pretože zaznamenáva zmeny metadát pred ich aplikovaním [14][25][26].

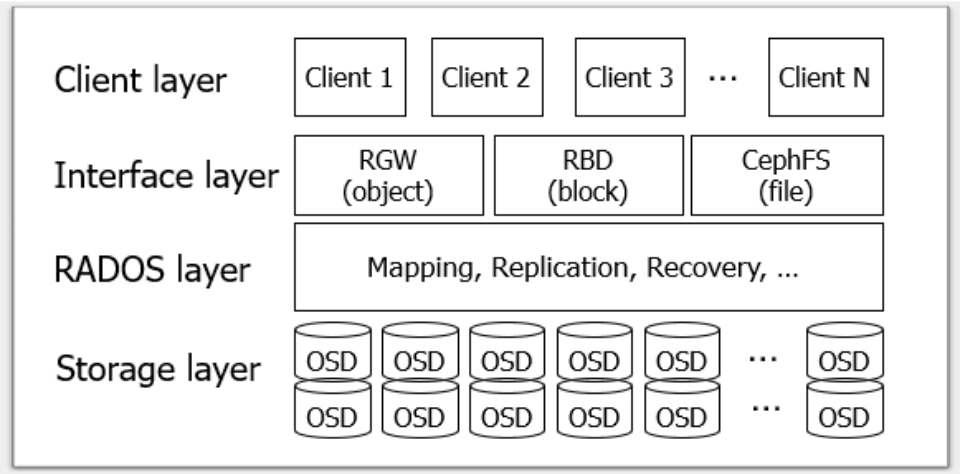
3.3 RADOS

RADOS (Reliable Autonomic Distributed Object Store) je technológia, ktorá tvorí základ distribuovaného systému Ceph a ponúka škálovateľné, spoľahlivé a efektívne riešenia ukladania dát. Je to distribuovaný systém ukladania objektov, ktorý je špeciálne navrhnutý na ukladanie veľkého množstva dát v klastri štandardného hardvéru. RADOS tvorí jadro softvéru Ceph a podporuje rozhrania blokového, súborového a objektového úložiska. Systém je navrhnutý pre vysoký výkon, rozširovateľnosť a spoľahlivosť. Riadi replikáciu, obnovu a distribúciu údajov bez jediného bodu zlyhania [14][27].



Obrázok 5 Princíp dosiahnutia konsenzu pomocou algoritmu Paxos [28]

Nakoľko architektúra klastra Ceph je primárne postavená na RADOS-e s pridaním ďalších funkcií, klastre RADOS podobne pozostávajú z dvoch typov démonov, a to **OSD** a **Monitor**. Práve RADOS je technológia, ktorá zabezpečuje takmer všetky známe výhody Ceph-u pre distribuované úložisko. Používa algoritmy **CRUSH** a **Paxos** na efektívnu distribúciu a správu dát v rámci klastra, ktorého princíp je zobrazený na Obrázku 5 vyššie. RADOS taktiež zabezpečuje trvanlivosť a dostupnosť údajov prostredníctvom replikácie a v neposlednom rade vlastnosť samoregenerácie. V prípade zlyhania node-u, RADOS automaticky prerozdelí dáta do iných node-ov, čím zabezpečí zachovanie replikačného faktora. Tento proces je pre používateľa transparentný a zabezpečuje tak nepretržitú dostupnosť dát [14][27][29].



Obrázok 6 Zobrazenie jednotlivých vrstiev Ceph architektúry [30]

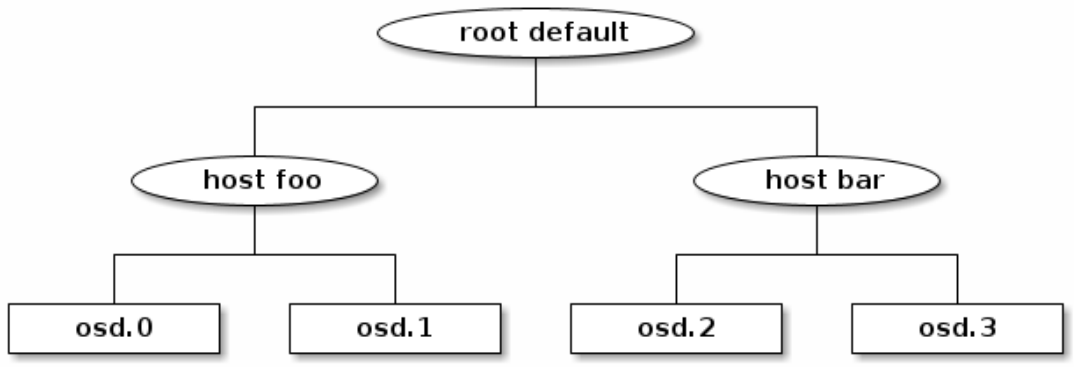
Logické delenie Ceph klastra na vrstvy je zobrazené na Obrázku 6 vyššie. V klientskej vrstve (**Client layer**) sa nachádzajú aplikácie, zariadenia alebo platformy, ktoré využívajú distribuované úložisko klastra Ceph. Prístup k dátam zabezpečuje vrstva rozhraní (**Interface layer**). Ceph ponúka **Ceph Rados Gateway (RGW)** - rozhranie objektového úložiska, ktoré poskytuje aplikáciám rozhranie RESTful API, **Ceph Rados Block Device (RBD)** pre ukladanie blokov a image-ov, napr. pre virtuálne zariadenia a virtualizačné platformy a **CephFS**, ktorý je možné mount-núť ako bežný disk do operačného systému v podobe zdieľaného úložiska. **RADOS layer** - vrstva RADOS je riadiaca vrstva, ktorá má na starosti všetky vyššie spomínané vlastnosti v predošlom odseku. Poslednou vrstvou je úložná vrstva (**Storage layer**), ktorá sa skladá zo všetkých OSD démonov v klastri [14].

Ceph ponúka navyše vo vrstve rozhraní aj priamy prístup pre natívne API pomocou knižnice „librados“, ako je zobrazené na Obrázku 2. Knižnica **librados** poskytuje API rozhranie, ktoré umožňuje priamu interakciu s backendom úložiska RADOS. Vytvorené aplikácie sú tak schopné vykonávať operácie priamo na klastri Ceph, čo vedie k efektívnemu ukladaniu a vyhľadávaniu údajov bez potreby použitia klientskej medzivrstvy. Po pripojení môžu aplikácie vykonávať rôzne operácie, ako je čítanie a zápis dát, správa rozšírených atribútov (XATTR) a výpis objektov v rámci úložiska. Librados ponúka podporu asynchrónnych I/O operácií a poskytuje knižnice pre viaceré programovacie jazyky, ako C, C++, Python, Java a PHP [14][31].

3.4 Algoritmus CRUSH

CRUSH (Controlled Replication Under Scalable Hashing) je pokročilý algoritmus na umiestňovanie dát, ktorý efektívne určuje, ako by sa mali údaje ukladať v úložných node-och klastra. Vďaka tomu sú klastre Ceph vysoko odolné a škálovateľné, pretože distribuujú dáta spôsobom, ktorý maximalizuje dostupnosť a minimalizuje potrebu opätovného vyváženia dát, keď sa klaster zväčšuje alebo zmenšuje. Na rozdiel od tradičných metód prideľovania údajov, ktoré vyžadujú centrálny adresár pre zaznamenávanie pozície jednotlivých dát, CRUSH umožňuje klientom Ceph vypočítať umiestnenie údajov priamo pomocou mapy klastra. Tým sa výrazne znižujú režijné náklady a riziká vzniku potencionálnych bottleneck-ov, čo umožňuje systému Ceph dosiahnuť spoľahlivosť a výkon pri spracovaní obrovského množstva dát v mnohých node-och úložiska [14][20].

Algoritmus CRUSH využíva hierarchickú štruktúru pozostávajúcu z tzv. „*bucket*-ov“ a OSD pre mapovanie dátových objektov na fyzické úložné miesta. Každá úroveň hierarchie môže obsahovať viacero zariadení alebo iných bucket-ov. Tie následne tvoria stromovú štruktúru odrážajúcu fyzické usporiadanie prostredia úložiska. Táto štruktúra je zakódovaná v tzv. **CRUSH mape**, ktorá je dodávaná všetkým klientom Ceph a OSD démonom. Príklad CRUSH mapy je možné vidieť na Obrázku 7 [14][32].



Obrázok 7 Príklad štruktúry CRUSH mapy [32]

Keď klient Ceph-u potrebuje zapisovať údaje, použije algoritmus CRUSH a najnovšiu mapu CRUSH na výpočet, ktorý OSD by mal uložiť konkrétny dátový objekt. Pri tomto výpočte sa zohľadňuje hierarchia, váhy zariadení a pravidlá replikácie alebo erasure coding pravidlá definované pre dáta, čím sa zabezpečí rovnomerné rozloženie dát v rámci klastra a dodržanie stanovených požiadaviek na redundanciu [14][32].

Distribúciou údajov predvídateľným, ale náhodným spôsobom medzi viacerými OSD, zvyšuje CRUSH dostupnosť údajov a odolnosť voči chybám. Umožňuje klastru dynamicky vyvažovať údaje pri pridávaní alebo odstraňovaní OSD, čo je kľúčové pre rozsiahle systémy ukladania dát, ktoré sa musia škálovať, aby sa prispôsobili rastúcim objemom dát bez narušenia prebiehajúcich operácií [14][20][32].

CRUSH umožňuje klastrom Ceph horizontálne sa rozširovať s minimálnym dopadom na výkon rovnomerným rozložením pracovného zaťaženia na všetky dostupné zdroje. Znižuje latenciu a zvyšuje priepustnosť tým, že poskytuje priamu komunikáciu klienta s OSD bez centralizovanej vyhľadávacej tabuľky. Administrátori majú možnosť definovať vlastné hierarchické štruktúry, ktoré zodpovedajú ich skutočnému rozloženiu hardvéru, čím sa optimalizuje výkon a trvanlivosť dát [14][20][32].

3.5 Pool-y a PGs

Pool-y sú základnými jednotkami, v ktorých sa nachádzajú objekty v rámci klastra Ceph. Zohrávajú kľúčovú úlohu pri orchestrácii distribúcie a odolnosti dát. **PGs (Placement Groups)** sú kľúčovým prvkom architektúry Ceph, ktorý zefektívňuje organizáciu, prideľovanie a prístup k údajom v distribuovanom prostredí. Tento mechanizmus umožňuje Ceph-u ľahko škálovať pri zachovaní dostupnosti a vyváženej distribúcie dát v rámci klastra. Porozumenie fungovania PGs je nevyhnutné pre pochopenie, ako Ceph dosahuje redundanciu a efektívnosť pri správe dát [14][33][34].

3.5.1 Pool

V klastri Ceph-u sú pool-y **logické partície**, ktoré obsahujú objekty. Fungujú ako rozhranie pre klientov na ukladanie a načítanie dát. Ceph podporuje dva typy pool-ov: „**replicated**“ a „**erasure coded**“. Replicated pool-y poskytujú redundanciu tým, že ukladajú viacero kópií objektu na rôznych OSD. Erasure coded pool-y používajú na ochranu dát paritu, čím poskytujú mechanizmus redundancie, ktorý je efektívnejší z hľadiska úložiska ako replikované pool-y [14][33].

Fungovanie pool-ov je úzko spojené so schopnosťou klastra spravovať a distribuovať dáta. Tie sa distribuujú na OSD využitím PGs a algoritmu CRUSH [14].

Pool-y Ceph-u sú konfigurovateľné, čo umožňuje administrátorom upravovať nastavenia, ako je počet replík objektov, pravidlá CRUSH a parametre súvisiace s výkonom. Umožňujú logické zoskupovanie dát, čo uľahčuje ich správu a prístup k nim. Pre rôzne typy údajov, ako sú obrázky, videá alebo zálohy, možno vytvoriť rôzne pool-y, pričom každý z nich má vlastné nastavenia [14][33].

Je dôležité však pochopiť nasledujúci koncept. Po vytvorení pool-u, Ceph automaticky spravuje distribúciu dát medzi dostupné OSD na základe konfigurácie a algoritmu CRUSH. Preto sú pool-y nakonfigurované tak, aby riadili stratégie replikácie a umiestňovania dát, a nie aby zahŕňali konkrétne node-y. V predvolenom nastavení sa všetky OSD podieľajú na ukladaní dát pre všetky pool-y. OSD démony nie sú konkrétne priradené ku konkrétnemu pool-u. Namiesto toho algoritmus CRUSH určuje, ako sa údaje distribuujú medzi OSD na základe konfigurácie pool-u a topológie klastra [33].

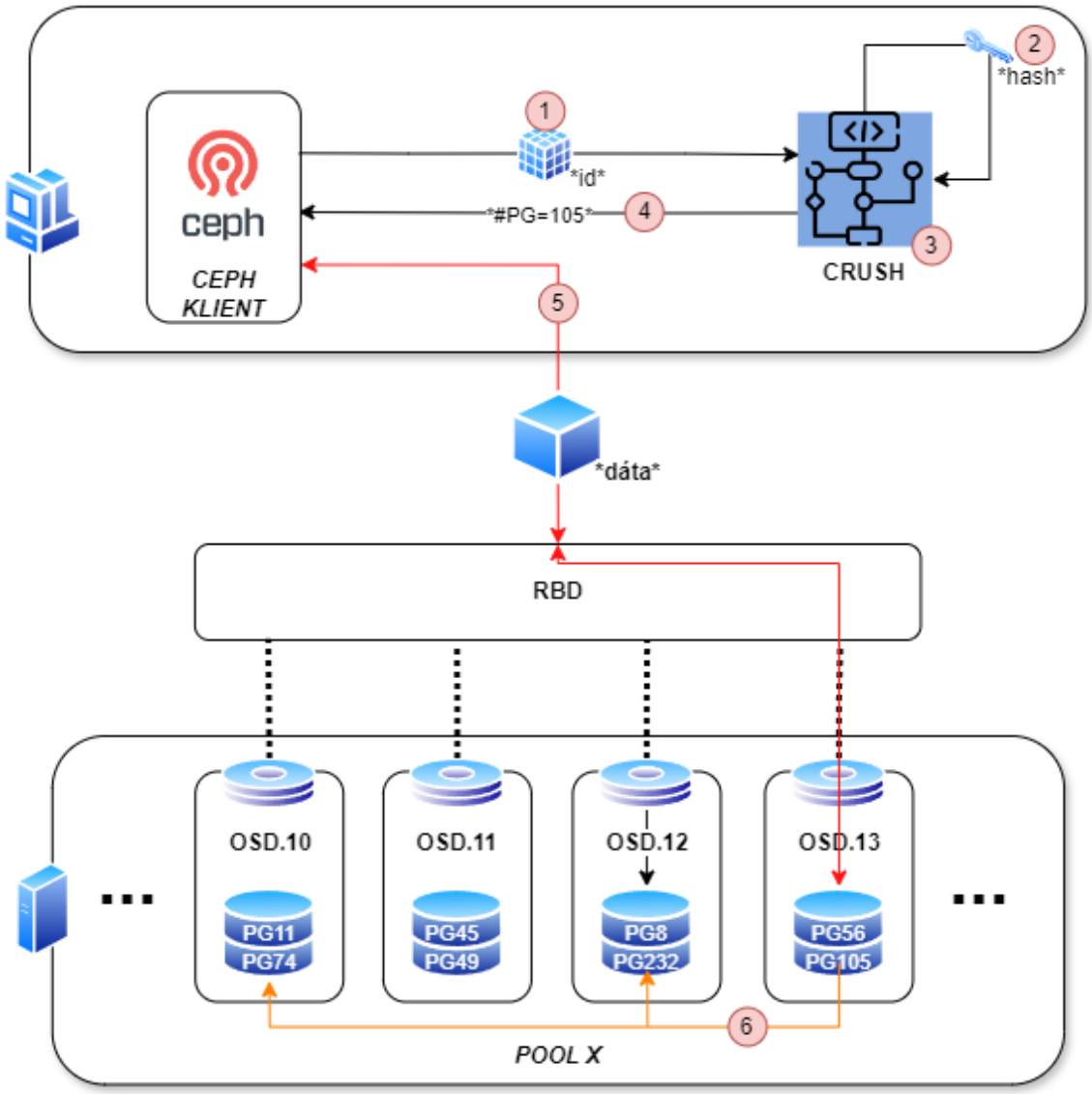
V klastri Ceph, rozhodnutie o tom, kde bude objekt uložený – konkrétne, v ktorom pool-e, sa vykonáva v čase zápisu objektu do klastra. Toto rozhodnutie je založené na aplikácii alebo používateľovi, ktorý špecifikuje cieľový pool pre dáta [14].

3.5.2 PG

Placement Groups (PGs) sú logické partície v rámci Ceph pool-ov. Združujú objekty, čím zefektívňujú prístup k objektom. PGs znižujú réžiu spojenú so sledovaním umiestnenia objektov a metadát, čo je čoraz náročnejšie, keď počet objektov narastá na milióny. Každá PG je potom priradená k jednotlivým démonom OSD v rámci pool-u, pričom jedna PG slúži ako primárna (na koordináciu operácií zápisu alebo čítania pre klienta) a ostatné sekundárne pre redundanciu dát [14][34].

Proces sa začína tým, že klient Ceph identifikuje PG, do ktorej objekt patrí, zvyčajne na základe jeho mena alebo iného jedinečného identifikátora. Táto identifikácia sa vykonáva pomocou funkcie hash-ovania. Výsledkom hash-ovacej funkcie je číslo a toto číslo sa použije na výber jednej z dostupných PGs. Tento proces je navrhnutý pre nahradenie klasického spôsobu ukladania metadát napr. stromovou štruktúrou. Ak by sme zobrali do úvahy situáciu, kedy chceme nájsť jeden konkrétny objekt bez predošlého záznamu v cache medzi miliónom objektov, určenie presného miesta, kde sa objekt nachádza, je mimoriadne časovo náročné, a tým sa znižuje výkon klastra. Použitím PGs sa minimalizuje čas potrebný na lokalizovanie umiestnenia objektu, keďže na základe vypočítanej hash-e dokáže algoritmus CRUSH takmer okamžite určiť OSD a PG, v ktorom je objekt uložený. Na základe týchto údajov algoritmus CRUSH nasmeruje klienta na konkrétne OSD, kam dáta zapíše alebo dáta prečíta [35].

Ceph minimalizuje výpočtovú a pamäťovú réžiu zoskupovaním objektov do PGs. Tie umožňujú efektívnu distribúciu a replikáciu dát v potenciálne veľkom počte OSD démonov. Odolnosť a dostupnosť dát je zabezpečená replikáciou dát na viacerých OSD v rámci PGs, a to aj v prípade zlyhania OSD. Ceph má funkciu nazývanú automatické škálovanie PGs, ktorá dokáže upraviť počet PGs v poole na základe využitia úložiska a definovaných pravidiel. To pomáha udržiavať výkon a využitie zdrojov bez manuálneho zásahu. Automatické škálovanie berie do úvahy celkovú kapacitu úložiska, rozloženie dát medzi rôzne úložné zariadenia a očakávanú veľkosť dát, aby podľa potreby upravilo počet PGs [14][36].



Obrázok 8 Princíp zápisu/čítania dát s použitím klienta

Pre priblíženie procesu, ako funguje zápis, alebo čítanie dát použijeme Obrázok 8. Proces sa skladá zo šiestich krokov. V prvom kroku klient pošle identifikátor algoritmu CRUSH. Ten z identifikátora v druhom kroku vypočíta hash. V treťom kroku sa z vypočítanej hash-e vypočíta podľa predurčeného algoritmu číslo PG. Štvrtý krok je odoslanie čísla PG klientovi. Keďže klient disponuje celou aktuálnou mapou CRUSH, v piatom kroku na základe čísla PG nadviaže spojenie priamo s príslušným OSD z vybraného pool-u, na ktorom sa PG nachádza. Podľa operácie sa buď dáta zapíšu alebo prečítajú. Posledným šiestym krokom je v prípade zápisu dát replikácia na sekundárne OSD, resp. do ich príslušných PGs.

4. NASADENIE KLASTRA CEPH V TESTOVACOM PROSTREDÍ

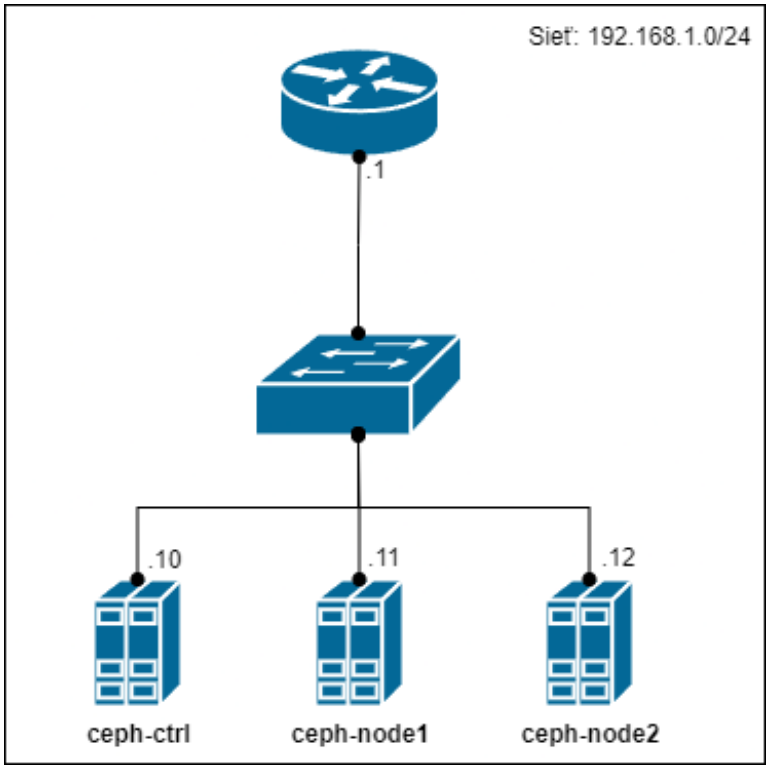
Na inštaláciu softvéru Ceph existuje niekoľko odporúčaných metód. **Cephadm** a **Rook** sú nástroje, ktoré sú navrhnuté primárne pre nasadenie pomocou kontajnerov. Cephadm pracuje priamo s kontajnermi a Rook je navrhnutý pre nasadenie klastrov Kubernetes. Medzi ďalšie metódy patria **ceph-ansible**, **ceph-salt**. Pre platformu OpenStack je možná inštalácia pomocou nástroja OpenStack Charms - **Juju** alebo ako **Puppet** modul. Konkrétna metóda závisí od prostredia a preferencií. Každá metóda je navrhnutá s dôrazom na jednoduchosť použitia, čím sa zabezpečuje kompatibilita s rôznymi infraštruktúrami [37].

Možnosťou je aj manuálna inštalácia. Poskytuje najvyššiu úroveň kontroly a možnosti konfigurácie počas inštalácie. Z tohoto dôvodu sme sa rozhodli použiť práve manuálnu inštaláciu pre nasadenie v testovacom prostredí, ktorá je popísaná v tejto kapitole. Využitím manuálnej inštalácie môžeme lepšie pochopiť, ako softvér Ceph funguje a ako sa konfiguruje [37].

4.1 Testovacie prostredie

Pre nasadenie softvéru Ceph použijeme virtuálne testovacie prostredie s využitím virtuálnych serverov s operačným systémom **Ubuntu 22.04** . Na základe hardvérových odporúčaní [38], každý server disponuje dostatočným množstvom výpočtových prostriedkov.

Topológiu virtuálneho testovacieho prostredia môžeme vidieť na Obrázku 9 nižšie.



Obrázok 9 Topológia virtuálneho testovacieho prostredia

Prostredie sa skladá z troch serverov. Jeden server – „ceph-ctrl“ bude slúžiť ako monitorovací a manažmentový node. Ostatné dva servery sú určené ako OSD node-y, ktoré disponujú navyše jedným 32 GB diskom.

4.2 Príprava serverov

Pred samotným začiatkom nasadenia je potrebné nainštalovať softvér Ceph na každý server, ktorý je súčasťou klastra. Inštalácia sa dá rozdeliť na dva kroky:

1. Pridanie kľúčov a repozitárov

```
wget -q -O- 'https://download.ceph.com/keys/release.asc' | sudo apt-key add –
apt-add-repository 'deb https://download.ceph.com/{release}/ {codename} main'
```

Premenné `release` a `codename` je potrebné nahradiť konkrétnymi údajmi. Premenná `release` označuje inštalovanú verziu softvéru Ceph a premenná `codename` označuje kódové meno distribúcie, na ktorej pridávame repozitár. V tomto prípade premenná `release` sa nahradí verziou “*reef*” a `codename` sa nahradí menom “*jammy*” [39].

2. Nainštalovanie potrebných balíčkov

```
apt-get update && apt-get install ceph ceph-mds
```

Následne prejdeme k aktualizovaniu repozitárov a nainštalujeme balíčky ceph a ceph-mds [40].

4.3 Nasadenie monitorovacieho node-u

Samotné nasadenie softvéru Ceph začína nasadením monitorovacieho node-u. Je potrebné nasadiť minimálne jeden monitor a zabezpečiť dostatočný počet diskov pre OSD démony na ukladanie kópií objektov. Počiatočné nastavenie monitora je dôležité, pretože stanovuje počiatočné kľúčové parametre, ako sú faktor replikácie, počet PGs na OSD, intervaly a požiadavky na overovanie [41].

Pre nasadenie monitora, je potrebné na konkrétnom serveri nakonfigurovať jedinečné ID klastra – tzv. „FSID“. Vytvoríme konfiguračný súbor, v ktorom nastavíme FSID, názov monitora, sieťové a ďalšie nastavenia. Názov konfiguračného súboru určuje názov klastra. Nasleduje vytvorenie kľúča monitora pre zabezpečenú komunikáciu a administratívny kľúč pre konzolový prístup (CLI). Ďalším krokom je vygenerovanie mapy monitora, vytvorenie požadovaných priečinkov a súborov, a nastavenie ich oprávnení. Predposledným krokom je inicializovanie démona monitora s danou konfiguráciou a vytvorenými kľúčmi. Nakoniec spustíme služby monitora [41].

Proces nasadenia monitora s konkrérnymi príkazmi je opísaný v nasledujúcich bodoch:

- Overenie existencie priečinka /etc/ceph [41].

```
ls /etc/ceph
```

- Vytvorenie FSID [41].

```
uuidgen
```

Vygenerované UUID skopírujeme a následne ho použijeme ako FSID v konfiguračnom súbore.

- Vytvorenie konfiguračného súboru a nastavenie potrebných premenných [41].

```
sudo nvim /etc/ceph/ceph.conf
```

Vo vnútri konfiguračného súboru nastavíme FSID v **[global]** kontexte a hodnoty ďalších premenných.

```
[global]
fsid = {uuid}
mon_initial_members = cephmon2
mon_host = 192.168.1.10
public_network = 192.168.1.0/24
auth_cluster_required = cephx
auth_service_required = cephx
auth_client_required = cephx
osd_pool_default_size = 3
osd_pool_default_min_size = 2
osd_pool_default_pg_num = 333
osd_crush_chooseleaf_type = 1
```

Premennou **mon_inintial_members** určíme doménové mená monitorovacích node-ov. V tomto prípade nasadzujeme iba jeden monitorovací node, ale v prípade viacerých je potrebné doménové mená oddeliť čiarkou. Rovnaký princíp platí aj pri premennej **mon_host**. Premennou **public_network** určujeme sieť, cez ktorú budú komunikovať jednotlivé servery klastra s klientami, a pokiaľ nie je určená dedikovaná samostatná sieť, tak aj pre komunikáciu jednotlivých serverov klastra medzi sebou, pre dátovú aj administratívnu komunikáciu. Redundancia resp. počet kópií objektov v klastri je určená premennou **osd_pool_default_size**. Premenná **osd_pool_default_min_size** nastavuje minimálny počet replík, ktoré musia byť k dispozícii, aby klaster umožnil operáciu čítania alebo zápisu objektu. Premenná **osd_crush_chooseleaf_type** nastavuje typ distribuovania replík pomocou CRUSH algoritmu. V tomto prípade sme ju nastavili na hodnotu „1“, čo zodpovedá redistribuovanie replík na konkrétne fyzické servery.

- Vytvorenie privátneho kľúča pre klaster a administratívneho kľúča [41].

```
ceph-authtool --create-keyring /tmp/ceph.mon.keyring --gen-key -n mon. --cap mon 'allow *'
ceph-authtool --create-keyring /etc/ceph/ceph.client.admin.keyring --gen-key -n client.admin --cap
mon 'allow *' --cap osd 'allow *' --cap mds 'allow *' --cap mgr 'allow *'
```

- Nastavenie oprávnení a vlastníctva pre administratívny kľúč.

```
chmod 644 /etc/ceph/ceph.client.admin.keyring
chown ceph:ceph /etc/ceph/ceph.client.admin.keyring
```

Tento krok nie je súčasťou oficiálnej dokumentácie, ale bez tohoto kroku nie je možné spustiť démona ani službu monitorovania.

- Vygenerovanie OSD kľúča, vytvorenie používateľa **client.bootstrap-osd** a priradenia kľúča používateľovi [41].

```
ceph-authtool --create-keyring /var/lib/ceph/bootstrap-osd/ceph.keyring --gen-key -n client.bootstrap-osd --cap mon 'profile bootstrap-osd' --cap mgr 'allow r'
```

- Pridanie administratívneho a OSD kľúča ku kľúčom klastra [41].

```
ceph-authtool /tmp/ceph.mon.keyring --import-keyring /etc/ceph/ceph.client.admin.keyring
ceph-authtool /tmp/ceph.mon.keyring --import-keyring /var/lib/ceph/bootstrap-osd/ceph.keyring
```

- Nastavenie oprávnení a vlastníctva pre kľúč klastra [41].

```
chown ceph:ceph /tmp/ceph.mon.keyring
```

- Vytvorenie monitorovacej mapy [41].

```
monmaptool --create --add {hostnames} {addresses} --fsid {fsid} /tmp/monmap
```

V tomto príkaze pre premenné **hostnames** a **addresses** aplikujeme rovnaké pravidlá ako v prípade konfiguračného súboru. V prípade tohoto nasadenia používame len jeden monitorovací server, takže použijeme iba jedno doménové meno a jednu IP adresu. V prípade viacerých monitorovacích node-ov by sa doménové mená a IP adresy oddelili čiarkou.

- Vytvorenie predvoleného dátového priečinku [41].

```
sudo -u ceph mkdir /var/lib/ceph/mon/{monitor-hostname}
```

Daný príkaz je potrebné aplikovať na každom monitorovacom node-e zvlášť. Premennú **monitor-hostname** je potrebné nahradiť príslušným doménovým menom daného servera. V našom konkrétnom prípade vytvárame iba jeden monitorovací node, takže ho aplikujeme na rovnakom zariadení ako doteraz a premennú **monitor-hostname** nahradíme doménovým menom servera – „*cephmon2*“. V tomto prípade je dôležité použiť príkaz **sudo**, nakoľko daný priečinok je nutné vytvoriť ako „**ceph**“ používateľ.

- Pridanie monitorovacej mapy a kľúčov do démona [41].

```
sudo -u ceph ceph-mon [--cluster {cluster-name}] --mkfs -i {hostname} --monmap /tmp/monmap --keyring /tmp/ceph.mon.keyring
```

Rovnako ako v predchádzajúcich prípadoch, za premennú **hostname** doplníme doménové meno serveru/serverov, a je potrebné ho vykonať ako používateľ „*ceph*“. Možnosť **--cluster** je nutné použiť iba v prípade, že meno klastra nie je „*ceph*“, čo je predvolená hodnota. V našom prípade je, takže táto možnosť nebola použitá.

- Spustenie monitorovacej služby [41].

```
systemctl start ceph-mon@{hostname}
```

Príkaz pre spustenie služby je potrebné takisto ako v prípade vytvárania predvolených priečinkov spustiť na všetkých serveroch.

Pokiaľ je na serveroch spustený firewall, je potrebné sa uistiť, že nastavenia umožňujú komunikáciu monitora/monitorov. Týmto procesom sa vytvorí základ klastra Ceph, ktorý umožní ďalšie rozširovanie o OSD a ďalšie komponenty. Následne pomocou príkazu `ceph -s` môžeme overiť správne spustenie a nasadenie monitorovacieho node-u [41].

```
kysel@cephmon2:~$ sudo ceph -s
  cluster:
    id:     1f624cfb-1bb8-4b53-a9e9-4c18ed4b3938
    health: HEALTH_WARN
            mon is allowing insecure global_id reclaim
            1 monitors have not enabled msgr2

  services:
    mon: 1 daemons, quorum cephmon (age 17h)
    mgr: no daemons active
    osd: 0 osds: 0 up, 0 in

  data:
    pools:   0 pools, 0 pgs
    objects: 0 objects, 0 B
    usage:   0 B used, 0 B / 0 B avail
    pgs:

kysel@cephmon2:~$_
```

Obrázok 10 Zobrazenie výpisu príkazu ceph -s po nasadení klastra

4.4 Nasadenie manager-a

Pri nasadzovaní a správe klastra Ceph je potrebné nakonfigurovať manažmentový modul **ceph-mgr**. To sa dosiahne konfiguráciou modulu **Ceph Dashboard**, čo je zabudovaný webový ovládací panel na monitorovanie a správu [42].

Hoci nie je povinné, aby bol v klastri Ceph spustený dashboard, odporúča sa na efektívnu správu a monitorovanie klastra mať aspoň jednu inštanciu ceph-mgr s aktivovaným dashboard modulom. Dashboard ponúka grafické rozhranie, ktoré sa ľahko používa na monitorovanie stavu a výkonu klastra, ako aj na správu funkcií softvéru Ceph. Je však dôležité poznamenať, že klaster Ceph môže

fungovať aj bez dashboard-u a namiesto toho sa pri správe a monitorovaní spoliehať na nástroje CLI [42].

Pri manuálnom nasadení démona ceph-mgr je potrebné vytvoriť autentifikačný kľúč, umiestniť ho do určeného priečinku, a následne spustiť démona ceph-mgr. Nasadenie management démona prebehne na tom istom serveri, ako predošlé nasadenie monitorovacieho node-u [42].

Pre vytvorenie autentifikačného kľúča použijeme príkaz,

```
ceph auth get-or-create mgr.$name mon 'allow profile mgr' osd 'allow *' mds \ 'allow *'
```

vytvoríme súbor s názvom „*keyring*“, vložíme doň vygenerovaný kľúč predošlým príkazom, presunieme súbor „*keyring*“ do príslušného priečinka príkazom, `mv keyring /var/lib/ceph/mgr/ceph-{hostname}/keyring` a spustíme službu ceph-mgr príkazom, `ceph-mgr -i {hostname}` , kde v oboch prípadoch nahradíme `hostname` za doménové meno servera. Príkazom `ceph mgr module enable dashboard` spustíme modul dashboard-u [42].

Toto je odporúčaný proces pre nasadenie manager-a podľa oficiálnej dokumentácie v čase písania tejto práce. Po vykonaní daných príkazov sa však manager nespustí a dashboard nefunguje.

Modul dashboard-u pracuje len na HTTPS protokole, a tým vyžaduje dve veci naviac – certifikát pre HTTPS protokol a prihlasovacie údaje pre konkrétny modul. Preto v procese nasadenia je potrebné pred spustením démona vykonať nasledujúce príkazy.

```
ceph dashboard create-self-signed-cert
```

Daný príkaz vygeneruje self-signed certifikát SSL pre dashboard, čím zabezpečí šifrovanú komunikáciu medzi ovládacím panelom a jeho používateľmi. Je možné použiť aj vlastný certifikát [43].

```
ceph mgr module enable dashboard
```

Modul dashboard nie je predvolene povolený. Týmto príkazom sa aktivuje modul v rámci manager-a, čím sa sprístupní dashboard. V procese nasadenia podľa oficiálnej dokumentácie bol tento krok ako posledný. V tomto prípade je nutné modul povoliť ešte pred nastavením prihlasovacích údajov [43].

```
ceph dashboard set-login-credentials <username> <password>
```

Tento príkaz nastaví používateľské meno a heslo pre prístup do ovládacieho panelu. Heslo odporúčame napísať do súboru a cestu k tomuto súboru nahradiť

za premennú **password**. V niektorých prípadoch totižto dochádza k nesprávnemu nastaveniu hesla kvôli formátovaniu príkazu [43].

Po zadaní týchto príkazov je možné spustiť démona ceph-mgr. Pre overenie funkčnosti použijeme príkaz **ceph status** alebo **ceph -s**. Tieto príkazy sú rovnocenné. Pre získanie adresy ovládacieho panela použijeme príkaz **ceph mgr services**, kde pre modul „dashboard“ nájdeme IP adresu a port. Pokiaľ nie je určené inak, predvolený port je 8443 [42][43].

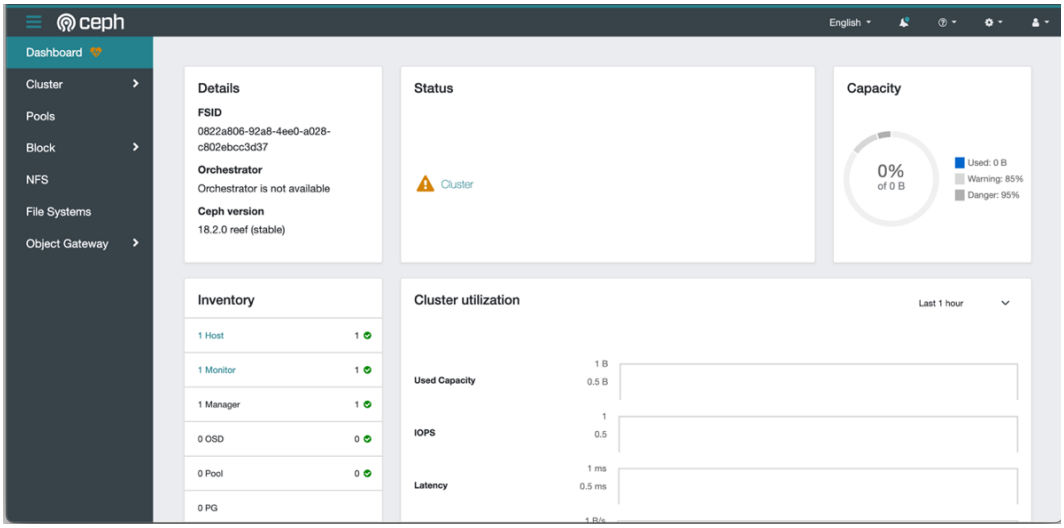
```
kysel@cephmon2:~$ ceph status
  cluster:
    id:     0822a806-92a8-4ee0-a028-c802ebcc3d37
    health: HEALTH_WARN
            mon is allowing insecure global_id reclaim
            1 monitors have not enabled msgr2
            1 daemons have recently crashed
            OSD count 0 < osd_pool_default_size 3

  services:
    mon: 1 daemons, quorum cephmon2 (age 19m)
    mgr: cephmon2(active, since 7m)
    osd: 0 osds: 0 up, 0 in

  data:
    pools:   0 pools, 0 pgs
    objects: 0 objects, 0 B
    usage:   0 B used, 0 B / 0 B avail
    pgs:

kysel@cephmon2:~$
```

Obrázok 11 Výpis príkazu ceph status po nasadení monitorovacieho démona

Rovnako ako na predošlom Obrázku 10, na Obrázku 11 môžeme vidieť, že klaster obsahuje jeden monitor. Navyše vidíme, že naozaj je spustený a pridaný aj jeden manager node resp. služba.

```
kysel@cephmon2:~$ ceph mgr services
{
    "dashboard": "https://192.168.1.13:8443/"
}
kysel@cephmon2:~$
```

Obrázok 12 Výpis príkazu ceph mgr services



Obrázok 13 Ovládací panel Ceph po prihlásení

4.5 Nasadenie OSD node-ov

Proces pridávania OSD zahŕňa skopírovanie súboru **/var/lib/ceph/bootstrap-osd/ceph.keyring** z monitorovacieho node-u na OSD node server. Následne využijeme nástroj **ceph-volume**, ktorý dokáže pripraviť logické alebo fyzické partície pre úložisko. Dôležité je podotknúť, že na príslušných diskoch je potrebné mať už vytvorený súborový systém. Na obidvoch node-och teda rozdelíme 32GB disk na štyri 8GB partície, za účelom simulácie viacerých diskov a vytvoríme na nich ext4 súborový systém [41].

Po vytvorení súborového systému je možné vytvárať OSD dvoma spôsobmi, s prípravou alebo bez prípravy. Nakoľko tieto postupy sú rovnocenné, použijeme jednoduchší postup bez prípravy. Nasadenie prebieha pomocou jediného príkazu:

```
ceph-volume lvm create --data /dev/{#disk}
```

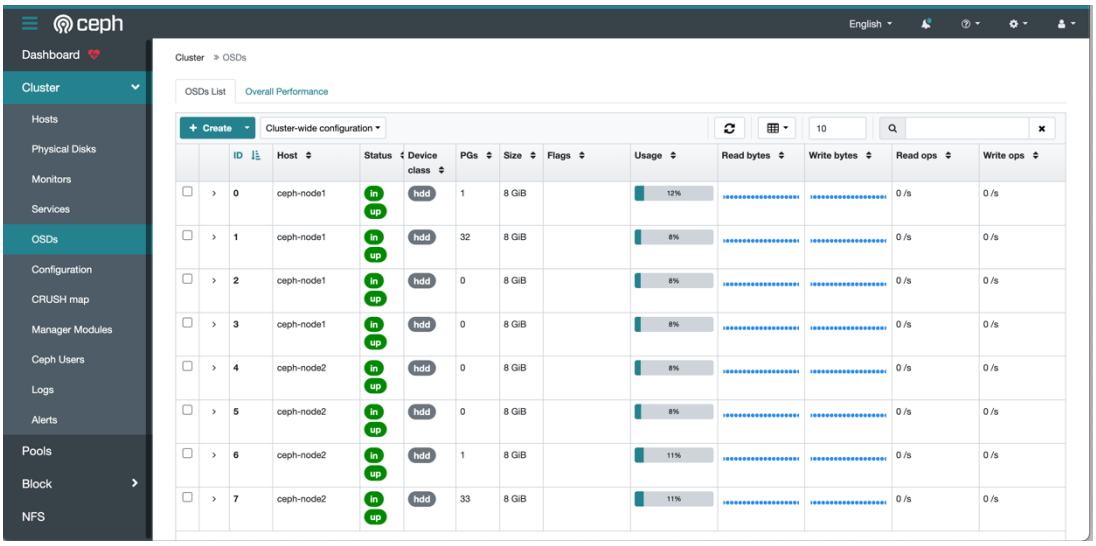
Premennú **#disk** nahradíme označením disku a číslom partície diskov/partícií, ktoré chceme pridať. Príkaz je potrebné zadať pre každý disk/partíciu zvlášť a na každom OSD node-e. V našom prípade na obidvoch serveroch to boli disky označené ako „sdb“ s číslami partícií 1 až 4 (sdb1, sdb2,..) [41].

Overenie môžeme vykonať pomocou príkazu **ceph status**, **ceph -s** alebo v ovládacom paneli v menu **Cluster -> OSDs**.

```
kysel@cephmon2:~$ ceph status
  cluster:
    id:     0822a806-92a8-4ee0-a028-c802ebcc3d37
    health: HEALTH_ERR
            mon is allowing insecure global_id reclaim
            Module 'devicehealth' has failed: table Device already exists

  services:
    mon: 1 daemons, quorum cephmon2 (age 12d)
    mgr: cephmon2(active, since 11d)
    osd: 8 osds: 8 up (since 12d), 8 in (since 12d); 1 remapped pgs

  data:
    pools:   2 pools, 33 pgs
    objects: 1 objects, 320 KiB
    usage:   2.7 GiB used, 61 GiB / 64 GiB avail
    pgs:     1/3 objects misplaced (33.333%)
             32 active+clean
             1  active+clean+remapped

kysel@cephmon2:~$
```

Obrázok 14 Zobrazenie výpisu ceph status po pridaní OSD node serverov

V sekcií „**services:**“ v časti „**osd:**“ na Obrázku 14 môžeme vidieť, že v klastri sa momentálne nachádza osem OSD, a v sekcií „**data:**“ v časti „**usage:**“ a „**pgs:**“, že klaster je funkčný, nakoľko je využitých 6GB priestoru z dostupných 64GB a všetky PGs sú v stave *„active+clean“* prípadne *„active+clean+remapped“*.



Obrázok 15 Zobrazenie menu Cluster -> OSDs z ovládacieho panelu

Obrázok 15 poskytuje detailnejšie zobrazenie OSD v klastri. Stĺpec „Host“ poskytuje informácie o tom, na ktorom zariadení sa jednotlivé OSD nachádzajú. Okrem toho ovládací panel zobrazuje aj kapacitu, využitie, aktuálny stav alebo počet PGs.

5. NASADENIE A INTEGROVANIE KLASTRA CEPH POMOCOU NÁSTROJOV OPENSTACK CHARMS

OpenStack Charms sú špecializované tzv. „charmed“ operátory (všeobecne označované ako **charms** - „charm-y“), určené na efektívne nasadenie a správu OpenStack-u v rámci ekosystému Juju. **Juju** je open-source nástroj, ktorý zjednodušuje nasadenie, konfiguráciu a správu cloudových služieb prostredníctvom intuitívneho rozhrania. Jednotlivé charm-y sú enkapsulované skripty, ktoré sa používajú na automatizáciu nasadenia a správy konkrétnej služby/modulu OpenStack-u, spolu so základnými službami, ktoré nie sú súčasťou OpenStack-u (RabbitMQ, MySQL a pod.) [44][45].

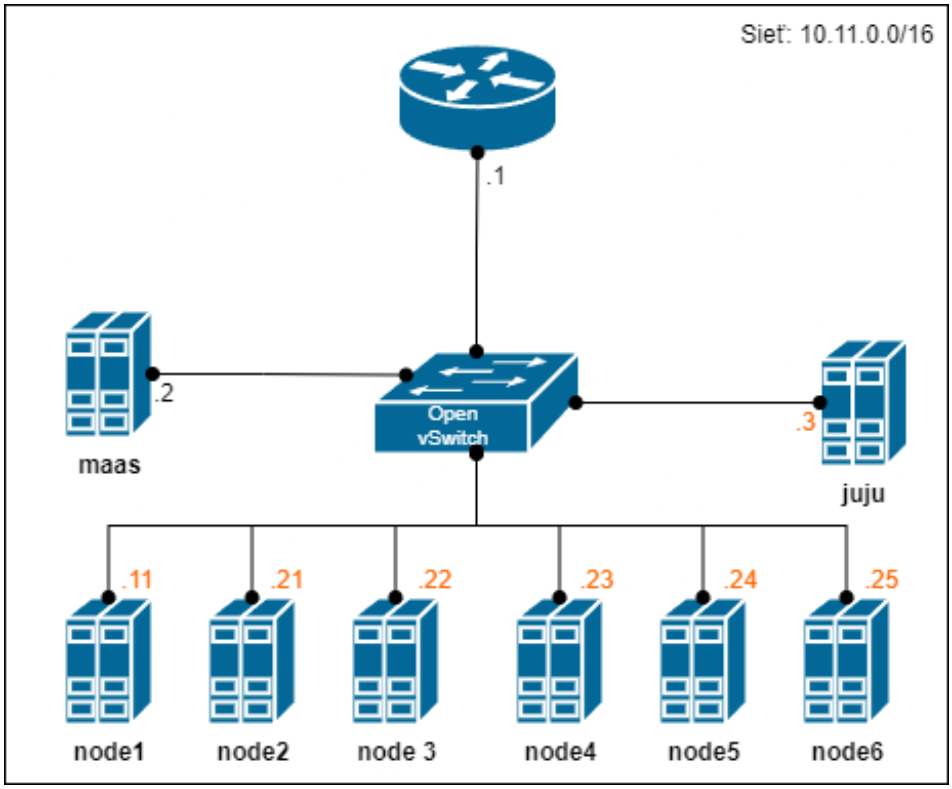
Integrácia **MAAS** (Metal as a Service) s charm-ami OpenStack-u umožňuje poskytovať a spravovať fyzické servery, použité v cloudovej infraštruktúre. Táto bezproblémová interakcia medzi vrstvou poskytovania fyzických služieb, ktorú poskytuje MAAS a možnosťami správy aplikácií Juju, ponúka komplexné riešenie pre nasadenie flexibilných cloudových riešení nad platformou OpenStack. Tento prístup znižuje čas a zložitosť nasadenia a umožňuje vytvoriť škálovateľné cloudové prostredie, ktoré možno prispôsobiť rôznym potrebám bez zvýšenia prevádzkovej réžie [44][45].

5.1 Koncept nasadenia

Na rozdiel od predošlej kapitoly, kde bol Ceph nainštalovaný ako samostatný klaster, v tomto prípade je súčasťou cloudovej platformy OpenStack. Nakoľko aktuálny cloud je nasadený pomocou nástrojov OpenStack Charms, nasadenie a integrácia Ceph-u bude rovnako riešená pomocou už vyššie spomínaných nástrojov MAAS a Juju. Pre nasadenie OpenStack-u a Ceph-u použijeme tzv. „**bundle**“, čo je konfiguračný **YAML** súbor, ktorý zjednocuje jednotlivé kroky nasadenia OpenStack-u do jedného kompaktného súboru, v ktorom sú zadefinované všetky potrebné parametre a nastavenia.

5.1.1 Topológia

Samotné nasadenie bude realizované vo virtuálnom prostredí. Virtuálne prostredie poskytuje flexibilitu a umožňuje jednoducho implementovať zmeny v topológií bez potreby fyzického prístupu k zariadeniam. Na Obrázku 16 nižšie je zobrazená topológia, s ktorou budeme následne pracovať.



Obrázok 16 Topológia pre nasadenie OpenStacku a Ceph-u pomocou charm-ov

Na zobrazenej topológií vidíme, že pre nasadenie použijeme celkovo 8 zariadení, ktoré predstavujú plánované rozšírenie súčasného cloudu. Všetky zariadenia sa nachádzajú v rovnakej subsieti 10.11.0.0 a virtuálne servery majú nastavené statické IPv4 adresy z danej podsiete. Posledné oktety zobrazené oranžovou farbou označujú, že napriek tomu, že tieto IP adresy sú statické, tak za prideľovanie týchto IPv4 adries je zodpovedný MAAS. Nakoľko pre nasadenie OpenStack-u sú použité VLAN-y, je potrebné na MAAS servery nakonfigurovať tzv. „bridge“ rozhrania typu Open vSwitch (OVS). Podrobnosti sú opísané neskôr v podkapitole 5.2 .

5.1.2 Virtuálne servery

Všetky zariadenia použité pre nasadenie sú virtuálne servery. Logicky môžeme servery rozdeliť na manažmentové a výpočtové servery. Manažmentové servery potrebujú menšie výpočtové zdroje ako výpočtové servery. V našom prípade sú manažmentové servery „*maas*“ a „*juju*“. Výpočtové servery tvoria zariadenia „*node1*“ až „*node6*“.

Zariadenie	CPU	RAM	Disky
maas	4	4 GB	60 GB
juju	4	4 GB	60 GB
node1	6	16 GB	2x 100 GB
node2	6	16 GB	60 GB, 100 GB
node3	6	16 GB	60 GB, 100 GB
node4	6	16 GB	60 GB, 100 GB
node5	6	16 GB	60 GB, 100 GB
node6	6	16 GB	60 GB, 100 GB

Tabuľka 2 Hardvérové prostriedky pridelené virtuálnym serverom

Detailnejšie pridelenie prostriedkov je zobrazené v Tabuľke 2 vyššie. Pri prideľovaní prostriedkov pre jednotlivé servery bolo potrebné zvážiť minimálne požiadavky, ktoré sú uvedené pre Juju a compute node-y v sekcii „Getting started“ na stránke sprievodcu charm-ami [44], a pre MAAS na oficiálnej stránke [46]. Pre MAAS server vzhľadom na veľkosť spravovanej infraštruktúry, aj napriek odporúčaným minimálne 4.5 GB operačnej pamäte RAM, sú po uvážení 4 GB dostačujúce. V prípade compute node-ov servery disponujú okrem 60GB disku, ktoré budú použité pre nasadenie operačného systému a jednotlivých komponentov OpenStack-u, navyše jedným diskom o veľkosti 100 GB. Tieto disky budú použité ako disky pre OSD démony Ceph-u. Výnimkou je „node1“, kde systémový disk je o veľkosti 100GB, nakoľko na ňom spustíme väčšie množstvo charm-ov.

5.2 Príprava serverov MAAS a Juju

Proces nasadenia začína prípravou serverov MAAS a Juju. Prvým krokom je nainštalovanie operačného systému Ubuntu 22.04 na server „maas“. Pre nasadenie OpenStack-u, vrátane príprav serverov MAAS a Juju, postupujeme presne podľa návodu v dokumentácii OpenStack Charms miernymi úpravami. Ako bundle súbor použijeme upravený bundle,podľa ktorého je nasadený aktuálny produkčný cloud.

Prvou zmenou je alokovanie externých sietí. V pôvodnej príprave subnet-y „EXT1“ a „EXT2“ majú pridelený rozsah z verejného rozsahu. Tieto siete v tomto prípade nemajú využitie a nie sú potrebné. Pre zachovanie jednoduchosti a konzistentnosti bundle-u však tieto siete je nutné ponechať, a preto došlo k zmene z verejného rozsahu na privátny (Tabuľka 3). Kroky pre vytvorenie sietí OpenStack-u pre komunikáciu s vonkajšími sieťami vynecháme.

Subnet	Pôvodný adresný rozsah	Pridelený adresný rozsah
EXT1	10.255.153.0/24	10.193.153.0/24
EXT2	10.255.154.0/24	10.193.154.0/24

Tabuľka 3 Zmena adresného rozsahu pre subnet-y EXT1 a EXT2

Ďalšia úprava postupu súvisí s rozhraniami serverov. Pôvodne proces zahŕňa vytvorenie tzv. „bond“ rozhraní na compute node-och. Nakoľko však všetky použité virtuálne servery disponujú len jedným sieťovým rozhraním, tento krok je zbytočný. Preto len pôvodné meno rozhrania nahradíme za názov „*bond*“.

Pre rozlíšenie jednotlivých Juju serverov upravíme taktiež meno Juju controller-a. V našom prípade sa controller nazýva „kysel-dp-juju-controller“.

Posledná zmena je vynechanie inštalácie dashboard-u pre Juju. Tento komponent nie je potrebný pre účely tejto práce, a preto tento krok bol vynechaný.

Verzie nainštalovaných softvérov MAAS a Juju sú taktiež odlišné. Nainštalovaná verzia MAAS je v našom prípade 3.4.1 a Juju je vo verzií 3.4.2 .

5.3 Komponenty Ceph-u v bundle súbore

Nasadenie klastra Ceph je súčasťou inštalácie OpenStack-u, ktorého cieľom je nasadiť základný klaster Ceph integrovaného do platformy OpenStack, vo forme poskytovania „backend“ funkcií pre jednotlivé moduly úložiska. Ceph v tomto prípade bude poskytovať všetky typy úložísk – objektové, blokové aj súborové, pre moduly **Glance**, **Cinder** a **Manila**.

Juju ponúka celkovo 10 charm-ov pre Ceph v rámci projektu Openstack Charms. Pre nasadenie klastra použijeme 7 z nich. Nepoužitými charm-ami sú **Ceph Proxy**, **Ceph Rbd Mirror** a **Ceph Iscsi**. Ceph Proxy sa používa pre pripojenie externého klastra Ceph do OpenStack-u. Nakoľko našim cieľom je integrovaný klaster priamo na compute node-och, je tento charm nevyužitý. Rovnako nevyužitými sú aj Ceph Rbd Mirror, pretože neexistuje ďalší klaster Ceph, do ktorého by sa údaje zrkadlovo ukladali a Ceph Iscsi - charm, ktorý umožňuje exportovanie obrazov blokových zariadení RADOS (RBD) ako iSCSI disky cez sieť, ktorý navyše už nie je ďalej vyvíjaný [48][49].

V nasledujúcich podkapitolách sa zameriame na parametre, ktoré je potrebné alebo možné zadefinovať pre nasadenie klastra Ceph s požadovanými funkciami v rámci bundle súboru (sekcie “options:” a “bindings:”). Samotné definovanie charm-ov sa nachádza v sekcií “applications:”. Povinné parametre ako “charm:” a “channel:” popisované nebudú, nakoľko tieto parametre sú povinné parametre pre každú aplikáciu a dajú sa nájsť na stránke charmhub.io [48]. Channel v našom prípade bude pre všetky aplikácie Ceph-u “reef/stable”. Pre parameter “source:” použijeme “cloud:focal-yoga”. Pre uľahčenie konfigurácie použijeme funkciu premenných, ktorý bundle ponúka [47].

```
variables:
  openstack-origin: &openstack-origin cloud:focal-yoga
  data-port: &data-port brEX:bond.153 brEX2:bond.154 brVXLAN:bond.500
  osd-devices: &osd-devices /dev/sdb
  expected-osd-count: &expected-osd-count 5
  expected-mon-count: &expected-mon-count 3
```

Premenné `expected-osd-count` a `expected-mon-count` nastavíme na hodnoty 5 a 3, z čoho vyplýva, že plánujeme nasadiť aspoň 5 OSD démonov a 3 monitory.

Pri vypísaných možnostiach v tvare <možnosť,..>, možnosť označená „*“ znamená predvolenú hodnotu.

5.3.1 Ceph Mon

Takisto, ako v prípade manuálnej inštalácie Ceph klastra, je potrebné začať monitorom. Na rozdiel od manuálnej inštalácie však monitor už nebude len jeden, ale tri, a tak v prípade poruchy jedného monitora nedôjde k výpadku prístupu k dátam. V tomto prípade nebudeme nastavovať žiadne iné parametre okrem očakávaného počtu OSD démonov a počtu monitorov.

Ceph Mon charm však nenasadzuje len monitor samotný. Pri nasadení každej inštancie monitora sa vo vnútri inštancie vytvára zároveň aj jedna inštancia ceph-mgr. Preto počet nasadených manager-ov je ovplyvnený počtom monitorov v takto nasadenom klastri.

5.3.1.1 Aplikovaná konfigurácia

```
ceph-mon:
  bindings:
    "": openstack-mgmt
    public: openstack-mgmt
    cluster: openstack-mgmt
  charm: ch:ceph-mon
  channel: reef/stable
  num_units: 3
  options:
    expected-osd-count: *expected-osd-count
    monitor-count: *expected-mon-count
    source: *openstack-origin
  to:
  - lxd:21
  - lxd:22
  - lxd:23
```

V sekcií „bindings:“ definujeme tzv. „space“, v ktorom bude aplikácia komunikovať. V našom prípade všetka komunikácia prebieha v space-y „openstack-mgmt“. Rovnaký space sme nastavili aj pre parametre „public:“ a „cluster:“. Parameter „ **"":** “ predstavuje predvolenú sieť. Tieto parametre fungujú rovnako ako v prípade manuálnej inštalácie. To znamená, že nimi dokážeme nastaviť sieť, cez ktorú bude klaster komunikovať s klientami, prípadne oddelenú sieť určenú pre administratívnu komunikáciu. Pre parametre v sekcií „options:“, nastavíme príslušné premenné.

5.3.1.2 Ďalšie možnosti [48]

Okrem možností použitých v bundle-i, je možné nastaviť aj ďalšie parametre, ktoré sme nepoužili, ale môžu byť užitočné v reálnom nasadení.

- **monitor-data-available-critical:** (int) – hodnota dostupnosti úložiska monitora v %, ktorá signalizuje HEALTH_ERR status klastra <*5>
- **monitor-data-available-warning:** (int) – hodnota dostupnosti úložiska monitora v %, ktorá signalizuje HEALTH_WARN status klastra <*30>
- **pg-autotune:** (string) – automatické nastavovanie PGs <*auto, true, false>
- **pgs-per-osd:** (int) – manuálne nastavenie počtu PGs podľa PGcalc [50]
- **prefer-ipv6:** (bool) – zapnutie podpory IPv6
- **use-syslog:** (bool) – logovanie do systémového syslogu

Všetky dostupné možnosti nájdeme na stránke charmhub.io v časti Ceph Mon [48].

5.3.2 Ceph Osd

Nasadenie OSD démonov je ďalším krokom, ktorý je nevyhnutný pre nasadenie Ceph klastra. Ako už bolo spomenuté vyššie, pre nasadenie použijeme momentálne 5 serverov, konkrétne 100GB disky **/dev/sdb**, ktoré sme zadefinovali v sekcií premenných ako premennú **osd-devices**. Dôležité je poznamenať, že na disk na controller node-e (node1) OSD démony nenasadzujeme, a to z dôvodu neskoršieho demonštrovania pridávania OSD do už vytvoreného klastra pomocou charm-ov.

5.3.2.1 Aplikovaná konfigurácia

```
ceph-osd:
  charm: ch:ceph-osd
  channel: reef/stable
  num_units: 5
  options:
    osd-devices: *osd-devices
    source: *openstack-origin
```

```
to:
- '21'
- '22'
- '23'
- '24'
- '25'
```

Použijeme základnú konfiguráciu. Špecifikujeme názvy diskov, ktoré sa majú použiť pre nasadenie OSD démonov (premenná **osd-devices**) a nasadíme ich na servery označené 21 - 25.

5.3.2.2 Ďalšie možnosti [48]

V prípade charm-u pre OSD démony Ceph-u existuje aj pár ďalších možností, ktoré môžu byť vhodné pre nastavenie.

- **bdev-enable-discard:** (string) – zapnutie „TRIM“ funkcie pre SSD disky <*auto, enable, disable>
- **ignore-device-errors:** (bool) – ak je nastavený na „true“ a charm nie je schopný inicializovať disk, namiesto chyby hlási iba varovanie
- **max-sectors-kb:** (int) – vytvára možnosť použitia väčších I/O veľkostí (napr. pri použití RAID kariet) <*1048576>
- **osd-encrypt:** (bool) – nastavenie šifrovania OSD diskov
- **osd-encrypt-keymanager:** (string) – nastavenie manažéra zodpovedného za uchovávanie šifrovacích kľúčov <vault, *ceph>
- **osd-journal:** (string) – nastavenie zariadenia, ktoré sa má používať ako zdieľaný journaling disk pre všetky OSD v node-e <*(každý OSD disk má vlastný)>
- **prefer-ipv6:** (bool) - zapnutie podpory IPv6
- **use-direct-io:** (bool) – nastavenia použitia priameho I/O prístupu pre journaling OSD disky
- **use-syslog:** (bool) - logovanie do systémového syslogu

Všetky konfiguračné možnosti sú dostupné na stránke charmhub.io v sekcií Ceph Osd [48].

5.3.3 Ceph Fs

Charm Ceph Fs pri nasadení nevytvára iba pool pre CephFS súborový systém, ale rovnako nasadzuje aj MDS servery s každou nasadenou inštanciou. Tie následne poskytujú rozhranie pre pripojenie sa k danému pool-u ako k zdieľanému úložisku.

5.3.3.1 Aplikovaná konfigurácia

```
ceph-fs:
   charm: ch:ceph-fs
   channel: reef/stable
   num_units: 5
   options:
     source: *openstack-origin
   bindings:
     "": openstack-mgmt
   to:
   - lxd:21
   - lxd:22
   - lxd:23
   - lxd:24
   - lxd:25
```

V tomto prípade použijeme znovu len základnú konfiguráciu charmu. Rovnako ako v prípade OSD démonov nasadíme MDS servery na zariadenia 21 – 25. Na server „node1“ nenasadíme žiadny MDS server, a to z dôvodu rozloženia záťaže, nakoľko „node1“ je zodpovedný za viac riadiacich inštancií OpenStack-u.

5.3.3.2 Ďalšie možnosti [48]

Ceph Fs poskytuje iba zopár užitočných nastavení pre naše využitie.

- **ceph-osd-replication-count:** (int) – nastavenie počtu replík ukladaných objektov <*3>
- **mds-cache-memory-limit:** (string) - maximálna veľkosť pre cache MDS serverov v bajtoch <*4Gi>
- **metadata-pool:** (string) - názov metadátového pool-u, ktorý sa má vytvoriť/použiť
- **pool-type:** (string) - nastavenie typu pool-u <*replicated,erasure-coded>
- **prefer-ipv6:** (bool) - zapnutie podpory IPv6

- **rbd-pool-name:** (string) – názov dátového pool-u, ktorý sa má vytvoriť/použiť
- **use-syslog:** (bool) - logovanie do systémového syslogu

V prípade, že by sme nastavili **pool type:** parameter na „erasure-coded“, existujú ďalšie možnosti konfigurácie, špecifické pre tento typ pool-u, ktoré môžeme nájsť v dokumentácií charm-u Ceph Fs na stránke charmhub.io [48].

5.3.4 Ceph Radosgw

Ceph Radosgw charm nasadzuje a spravuje RADOS Gateway (RGW) v klastri Ceph. V testovacom prostredí sme tento komponent nenasadzovali kvôli zjednodušeniu nasadenia, ale pre integráciu s OpenStack-om je nevyhnutný.

5.3.4.1 Aplikovaná konfigurácia

```
ceph-radosgw:
   bindings:
      "": openstack-mgmt
      public: openstack-mgmt
      internal: openstack-mgmt
      admin: openstack-mgmt
   charm: ch:ceph-radosgw
   channel: reef/stable
   num_units: 1
   options:
     source: *openstack-origin
   to:
   - lxd:11
```

Základná konfigurácia je momentálne rozšírená o **admin:** sieť, ktorá slúži na správu RGW.

5.3.4.2 Ďalšie možnosti [48]

Charm ponúka viacero relevantných možností konfigurácie už v bundle súbore.

- **cache-size:** (int) – počet kľúčov modulu keystone v lokálnej cache
- **ceph-osd-replication-count:** (int) - nastavenie počtu replík ukladaných objektov <*3>

- **pool-type:** (string) - nastavenie typu pool-u <*replicated,erasure-coded>
- **port:** (int) - nastavenie portu, na ktorom RGW počúva <*80, 443>
- **prefer-ipv6:** (bool) - zapnutie podpory IPv6
- **ssl_ca:** (string) – nastavenie vlastnej certifikačnej autority (CA) v prípade použitia HTTPS protokolu pre API.
- **ssl_cert:** (string) – nastavenie SSL certifikátu pre HTTPS API
- **ssl_key:** (string) – nastavenie kľúča SSL certifikátu pre HTTPS API
- **use-syslog:** (bool) - logovanie do systémového syslogu

Ďalšie konfiguračné možnosti môžeme nájsť na stránke charmhub.io [48].

5.3.5 Ceph Dashboard

Ceph Dashbord rovnako ako v prípade manuálnej inštalácie nevyžaduje žiadnu konkrétnu konfiguráciu pre svoje fungovanie. Poskytuje však pár konfiguračných možností, ktoré by mohli byť použité pre nasadenie na katedrový cloud.

5.3.5.1 Aplikovaná konfigurácia

```
ceph-dashboard:
  charm: ch:ceph-dashboard
  channel: reef/stable
```

Aplikovaná konfigurácia v prípade charm-u pre dashboard je jednoduchá. Nevyžaduje žiadnu konfiguráciu okrem nevyhnutných parametrov – meno charm-u a channel-u. Keďže dashboard ceph-u sa inicializuje na serveri, kde je spustený ceph-mgr, nie je potrebné uvádzať na aký server dashboard nasadiť, keďže charm ho nasadí automaticky – na inštanciu monitora.

5.3.5.2 Ďalšie možnosti [48]

Možnosti nastavenia modulu ceph-dashboard v bundle súbore nie sú rozsiahle. Všetky sa týkajú buď zabezpečenia hesla, resp. požiadaviek na vytvorenie hesla pre prihlásenie do dashboard-u, alebo nastavenia privátnych

SSL (Secure Socket Layer) certifikátov pre API rozhrania. Pre využitie na cloude sú najužitočnejšie politiky pre nastavenie hesla na prihlásenie.

- **password-policy-min-length:** (int) – nastavenie minimálnej dĺžky hesla
- **password-policy-check-username:** (boolean) – nedovolí vytvoriť heslo, ktoré obsahuje meno používateľa
- **password-policy-check-length:** (bool) – zamietne vytvorenie hesla, pokiaľ má dĺžku menšiu ako nastavené minimum

Tak ako v predošlých prípadoch všetky možnosti s podrobnosťami sú uvedené na stránke charmhub.io pre charm Ceph Dashboard [48].

5.3.6 Cinder Ceph

Prechádzame k charm-om, ktoré nie sú priamo súčasťou Ceph-u, ale sú nevyhnutné pre integráciu s OpenStack-om. Charm Cinder Ceph integruje Ceph s modulom **Cinder**, čím umožňuje Cinder-u používať blokové zariadenia Ceph-u (RBD).

5.3.6.1 Aplikovaná konfigurácia

```
cinder-ceph:
   charm: ch:cinder-ceph
   channel: yoga/stable
   num_units: 0
```

Pre nasadenie tohto charm-u použijeme takisto základnú konfiguráciu pre nasadenie s predvolenými hodnotami. Dôvodom je následné pridávanie rôznych pool-ov ako „backend“ Cinder-u pre OpenStack.

5.3.6.2 Ďalšie možnosti [48]

Pri opisovaní modulov OpenStack-u, ktoré poskytujú možnosti integrácie s Ceph-om sa zameriame len na možnosti, ktoré sú relevantné pre nastavenie správania pool-ov alebo Ceph-u samotného.

- **ceph-osd-replication-count:** (int) - nastavenie počtu replík ukladaných objektov <*3>

- **pool-type:** (string) - nastavenie typu pool-u <*replicated,erasure-coded>
- **use-syslog:** (bool) - logovanie do systémového syslogu
- **volume-backend-name:** (string) – nastavenie mena pre backend pool

Cinder Ceph charm ponúka viac nastavení najmä pre konfiguráciu BlueStore formátu diskov, ktoré využivajú OSD disky pre priamu komunikáciu s fyzickými diskami, alebo nastavenia erasure-coded pool-ov. Tieto nastavenia nie sú relevantné pre použitie na cloude, nakoľko optimalizácia na takejto úrovni je zbytočná. Všetky detailné nastavenia sú dostupné opäť na stránke charmhub.io [48].

5.3.7 Glance

Pre používanie klastra Ceph na ukladanie obrazov virtuálnych zariadení v prostredí OpenStack-u, je nutné použiť charm modulu **Glance**. Tento modul nemá špeciálny charm, ktorý by integroval Glance a Ceph napriek tomu, že Ceph je odporúčaným úložiskom pre Glance modul [48].

5.3.7.1 Aplikovaná konfigurácia

```
glance:
  bindings:
    "": openstack-mgmt
    public: openstack-mgmt
    internal: openstack-mgmt
    admin: openstack-mgmt
    shared-db: openstack-mgmt
  charm: ch:glance
  channel: yoga/stable
  num_units: 1
  options:
    openstack-origin: *openstack-origin
  to:
  - lxd:11
```

Použitá konfigurácia opäť obsahuje iba základné parametre nutné pre daný charm. Dôvodom je aj fakt, že najdôležitejším aspektom pri integrácií Ceph-u s modulom Glance, je vzťah medzi týmito dvoma komponentami, ktoré budú bližšie popísané v kapitole 5.4 .

5.3.7.2 Ďalšie možnosti [48]

Glance samozrejme obsahuje viacero konfiguračných možností pre fungovanie modulu samotného. Nasledujúce možnosti však priamo ovplyvňujú Ceph a jeho komponenty.

- **ceph-osd-replication-count:** (int) - nastavenie počtu replík ukladaných objektov <*3>
- **pool-type:** (string) - nastavenie typu pool-u <*replicated,erasure-coded>
- **prefer-ipv6:** (bool) - zapnutie podpory IPv6
- **use-syslog:** (bool) - logovanie do systémového syslogu

Rovnako ako v predošlých prípadoch sa viaceré možnosti opakujú, keďže pre každý modul sa vytvára vlastný pool s vlastnými nastaveniami. Podrobné možnosti konfigurácie modulu Glance sú popísané na stránke charmhub.io [48].

5.3.8 Manila Ganesha

Posledným charm-om pre integráciu Ceph-u je Manila Ganesha. Podobne ako Cinder Ceph charm, tento charm integruje CephFS s modulom Manila nasadením NFS-Ganesha. NFS-Ganesha poskytuje rozhranie pre protokol NFS (Network File System), ktorý dokáže obsluhovať súborové systémy z rôznych úložísk. Funguje ako most, ktorý poskytuje prostredníctvom systému NFS zdieľané úložisko inštanciám OpenStack-u. Tieto inštancie používajú ako backend-ové úložisko systém CephFS a bezproblémovo ho tak integrujú do cloudovej infraštruktúry OpenStack-u [48][51].

Tento charm neposkytuje žiadne možnosti nastavenia v bundle súbore, ktoré by ovplyvňovali klaster Ceph. Z tohoto dôvodu aplikovaná konfigurácia ani ďalšie možnosti nebudú zahrnuté.

5.4 Vzťahy modulov

V Juju sú vzťahy – „relations“ mechanizmom, ktorý umožňuje vzájomné prepojenie služieb a zdieľanie informácií o konfigurácii, službách a správe. Môžu byť definované v bundle súbore a opisujú, ako by mali byť rôzne aplikácie navzájom prepojené. Vzťahy sú kľúčové, pretože určujú, ako rôzne časti cloudového prostredia komunikujú, zdieľajú údaje a odovzdávajú si riadiace pokyny.

Všetky vzťahy sú v bundle súbore definované v sekcií „relations:“. Rovnako ako v predošlých prípadoch pri opisovaní konfiguračných možností pre jednotlivé charm-y v bundle súbore, tak i v prípade vzťahov budeme riešiť vzťahy len tých komponentov, ktoré priamo ovplyvňujú Ceph klaster.

Začneme vzťahmi, ktoré súvisia s autentifikáciou. Pre autentifikáciu sa v OpenStack-u sa používa modul Keystone a taktiež softvér Vault.

```
relations:
- - ceph-radosgw:identity-service
  - keystone:identity-service
- - vault:certificates
  - ceph-radosgw:certificates
- - ceph-dashboard:certificates
  - vault:certificates
```

Vzťah medzi `ceph-radosgw:identity-service` a `keystone:identity-service` spája Ceph RGW s modulom Keystone pre autentifikačné služby. Vzťahy medzi `vault:certificates` a `ceph-radosgw:certificates`, ako aj `ceph-dashboard:certificates` riadia bezpečnú komunikáciu a zdieľanie certifikátov medzi komponentmi Vault a Ceph.

Nasledujúce vzťahy popisujú, ako ceph-mon spolupracuje s ostatnými službami a komponentmi v klastri s cieľom zabezpečiť efektívnu prevádzku a konektivitu.

```
- - ceph-mon:client
  - nova-compute:ceph
- - ceph-mon:client
  - cinder-ceph:ceph
- - ceph-mon:client
  - glance:ceph
- - ceph-mon:client
  - manila-ganesha:ceph
- - ceph-osd:mon
  - ceph-mon:osd
- - ceph-radosgw:mon
  - ceph-mon:radosgw
- - ceph-mon:mds
  - ceph-fs:ceph-mds
- - ceph-dashboard:dashboard
  - ceph-mon:dashboard
```

Vzťahy zahŕňajúce **ceph-mon:client** umožňujú modulom Nova, Cinder, Glance a Manila komunikovať s klastrom Ceph. Komunikácia medzi OSD démonmi a monitormi je nastavená pomocou **ceph-osd:mon** a **ceph-mon:osd**. Ceph RGW je prepojený s monitormi klastra, prostredníctvom **cephradosgw:mon** a **ceph-mon:radosgw**. Monitory sú prepojené s MDS a CephFS inštanciami prostredníctvom vzťahu medzi **ceph-mon:mds** a **ceph-fs:ceph-mds**. Ceph Dashboard je prepojený s Ceph Mon zadefinovaním vzťahu **ceph-dashboard:dashboard** a **ceph-mon:dashboard**.

Ako posledné ostáva definovať vzťahy, ktoré zabezpečujú prístup a ako je úložisko poskytované k pripojeným službám, ktoré to vyžadujú (virtuálne zariadenia alebo kontajnery).

```
- - cinder-ceph:storage-backend
  - cinder:storage-backend
- - nova-compute:ceph-access
  - cinder-ceph:ceph-access
```

Prepojenie **cinder-ceph:storage-backend** s **cinder:storage-backend** konfiguruje Ceph ako backend pre Cinder. Napriek tomu, že modul Nova pri konfigurácií nebol spomenutý, je dôležité, aby vzťah medzi **nova-compute:ceph-access** a **cinder-ceph:ceph-access** bol zadefinovaný. Ten totiž zabezpečuje, že modul Nova má potrebný prístup k úložisku Ceph-u.

5.5 Nasadenie OpenStack-u s klastrom Ceph

Výhodou nasadenia OpenStack-u pomocou bundle súboru je jednoduchosť. Začneme vytvorením modelu príkazom **juju add-model**. Pomocou príkazov **juju switch** a **juju spaces** je potrebné overiť, či sme v ovládaní správneho modulu a jednotlivé space-y majú priradené správne siete. Príkazom **juju deploy** spustíme nasadenie cloudu [47].

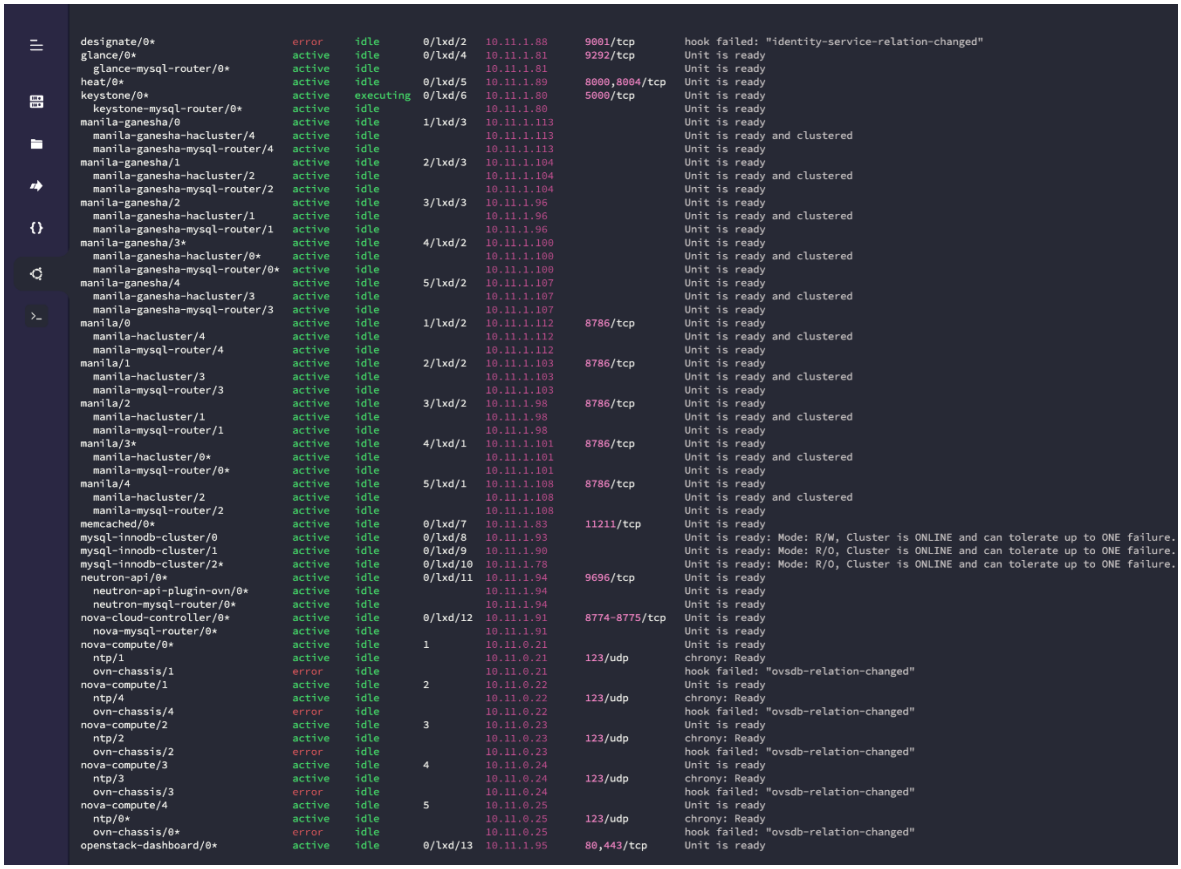
```
juju add-model <názov modelu>
juju deploy <cesta k bundle súboru>
```

Následne je potrebné vykonať úkony pre spustenie charm-u Vault, ktoré sú opísané v dokumentácii OpenStack. Je však nutné podotknúť, že príkazy pre Juju sa výrazne zmenili, a preto pred každým nasadením, je nutné prečítať dokumentáciu k aktuálnej verzií Juju, vrátane príkazov v tejto práci.

5.5.1 Overenie nasadenia

Po vykonaní príkazov pre nasadenie cloudu a charm-u Vault je nutné počkať, kým sa spustia všetky charm-y a ich príslušné kontajnery a služby. Nás konkrétne budú zaujímať najmä komponenty klastra Ceph a moduly, ktoré sú potrebné pre integráciu s OpenStack-om. Informácie, či sú služby aktívne a spustené, získame zadaním príkazu **juju status**.

Počas práce na práci sa vyskytli problémy s virtuálnym prostredím. Pri opakovanom nasadzovaní dochádzalo k nefunkčnosti modulov. V každom novom nasadení zlyhali rôzne moduly z rôznych príčin. Postupom času sa podarilo niektoré problémy vyriešiť, ale nasadenie nebolo možné použiť na testovanie v prípade, že modul „ovn-chassis“ nebol funkčný.

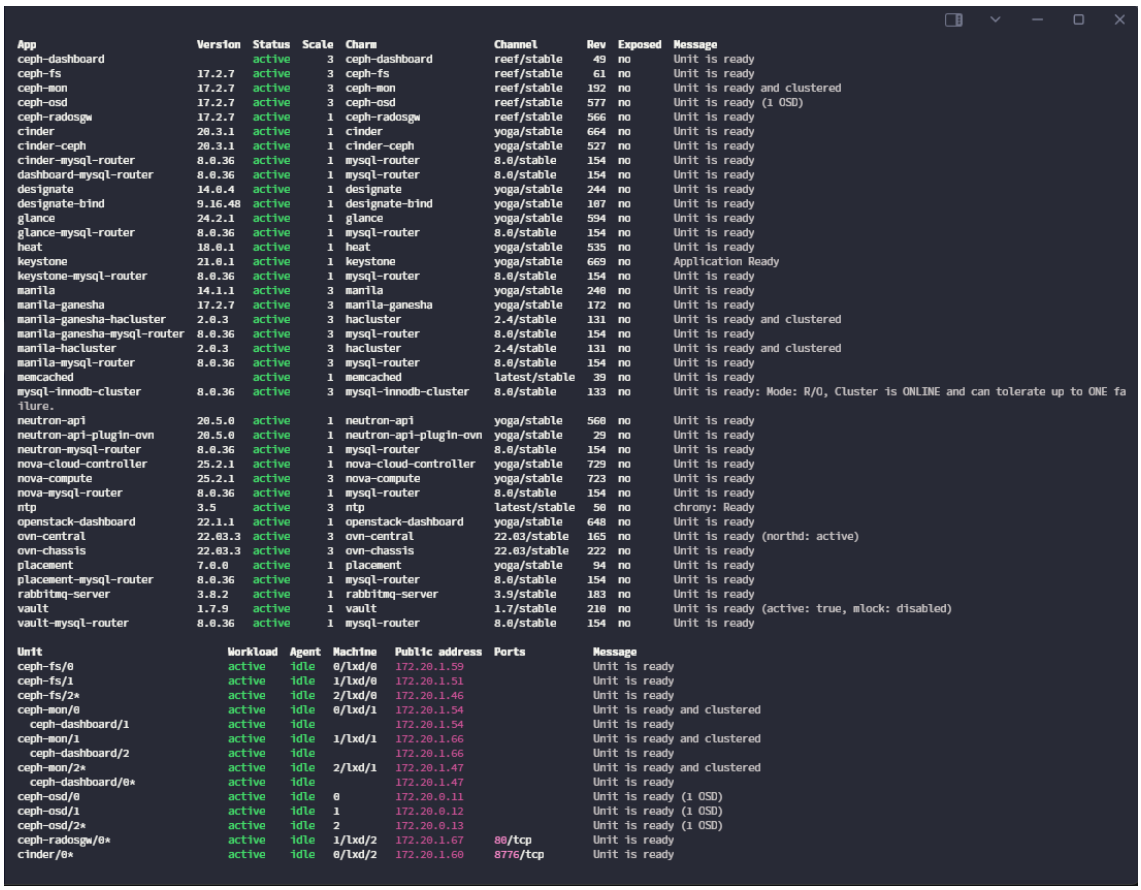


designate/0*	error	idle	0/lxd/2	10.11.1.88	9001/tcp	hook failed: "identity-service-relation-changed"
glance/0*	active	idle	0/lxd/4	10.11.1.81	9292/tcp	Unit is ready
glance-mysql-router/0*	active	idle		10.11.1.81		Unit is ready
heat/0*	active	idle	0/lxd/5	10.11.1.89	8000,8004/tcp	Unit is ready
keystone/0*	active	executing	0/lxd/6	10.11.1.80	5000/tcp	Unit is ready
keystone-mysql-router/0*	active	idle		10.11.1.80		Unit is ready
manila-ganesha/0	active	idle	1/lxd/3	10.11.1.113		Unit is ready
manila-ganesha-hacluster/4	active	idle		10.11.1.113		Unit is ready and clustered
manila-ganesha-mysql-router/4	active	idle		10.11.1.113		Unit is ready
manila-ganesha/1	active	idle	2/lxd/3	10.11.1.104		Unit is ready
manila-ganesha-hacluster/2	active	idle		10.11.1.104		Unit is ready and clustered
manila-ganesha-mysql-router/2	active	idle		10.11.1.104		Unit is ready
manila-ganesha/2	active	idle	3/lxd/3	10.11.1.96		Unit is ready
manila-ganesha-hacluster/1	active	idle		10.11.1.96		Unit is ready and clustered
manila-ganesha-mysql-router/1	active	idle		10.11.1.96		Unit is ready
manila-ganesha/3*	active	idle	4/lxd/2	10.11.1.100		Unit is ready
manila-ganesha-hacluster/0*	active	idle		10.11.1.100		Unit is ready and clustered
manila-ganesha-mysql-router/0*	active	idle		10.11.1.100		Unit is ready
manila-ganesha/4	active	idle	5/lxd/2	10.11.1.107		Unit is ready
manila-ganesha-hacluster/3	active	idle		10.11.1.107		Unit is ready and clustered
manila-ganesha-mysql-router/3	active	idle		10.11.1.107		Unit is ready
manila/0	active	idle	1/lxd/2	10.11.1.112	8786/tcp	Unit is ready
manila-hacluster/4	active	idle		10.11.1.112		Unit is ready and clustered
manila-mysql-router/4	active	idle		10.11.1.112		Unit is ready
manila/1	active	idle	2/lxd/2	10.11.1.103	8786/tcp	Unit is ready
manila-hacluster/3	active	idle		10.11.1.103		Unit is ready and clustered
manila-mysql-router/3	active	idle		10.11.1.103		Unit is ready
manila/2	active	idle	3/lxd/2	10.11.1.98	8786/tcp	Unit is ready
manila-hacluster/1	active	idle		10.11.1.98		Unit is ready and clustered
manila-mysql-router/1	active	idle		10.11.1.98		Unit is ready
manila/3*	active	idle	4/lxd/1	10.11.1.101	8786/tcp	Unit is ready
manila-hacluster/0*	active	idle		10.11.1.101		Unit is ready and clustered
manila-mysql-router/0*	active	idle		10.11.1.101		Unit is ready
manila/4	active	idle	5/lxd/1	10.11.1.108	8786/tcp	Unit is ready
manila-hacluster/2	active	idle		10.11.1.108		Unit is ready and clustered
manila-mysql-router/2	active	idle		10.11.1.108		Unit is ready
memcached/0*	active	idle	0/lxd/7	10.11.1.83	11211/tcp	Unit is ready
mysql-innodb-cluster/0	active	idle	0/lxd/8	10.11.1.93		Unit is ready: Mode: R/W, Cluster is ONLINE and can tolerate up to ONE failure.
mysql-innodb-cluster/1	active	idle	0/lxd/9	10.11.1.90		Unit is ready: Mode: R/O, Cluster is ONLINE and can tolerate up to ONE failure.
mysql-innodb-cluster/2*	active	idle	0/lxd/10	10.11.1.78		Unit is ready: Mode: R/O, Cluster is ONLINE and can tolerate up to ONE failure.
neutron-api/0*	active	idle	0/lxd/11	10.11.1.94	9696/tcp	Unit is ready
neutron-api-plugin-ovn/0*	active	idle		10.11.1.94		Unit is ready
neutron-mysql-router/0*	active	idle		10.11.1.94		Unit is ready
nova-cloud-controller/0*	active	idle	0/lxd/12	10.11.1.91	8774-8775/tcp	Unit is ready
nova-mysql-router/0*	active	idle		10.11.1.91		Unit is ready
nova-compute/0*	active	idle	1	10.11.0.21		Unit is ready
ntp/1	active	idle		10.11.0.21	123/udp	chrony: Ready
ovn-chassis/1	error	idle		10.11.0.21		hook failed: "ovsdb-relation-changed"
nova-compute/1	active	idle	2	10.11.0.22		Unit is ready
ntp/4	active	idle		10.11.0.22	123/udp	chrony: Ready
ovn-chassis/4	error	idle		10.11.0.22		hook failed: "ovsdb-relation-changed"
nova-compute/2	active	idle	3	10.11.0.23		Unit is ready
ntp/2	active	idle		10.11.0.23	123/udp	chrony: Ready
ovn-chassis/2	error	idle		10.11.0.23		hook failed: "ovsdb-relation-changed"
nova-compute/3	active	idle	4	10.11.0.24		Unit is ready
ntp/3	active	idle		10.11.0.24	123/udp	chrony: Ready
ovn-chassis/3	error	idle		10.11.0.24		hook failed: "ovsdb-relation-changed"
nova-compute/4	active	idle	5	10.11.0.25		Unit is ready
ntp/0*	active	idle		10.11.0.25	123/udp	chrony: Ready
ovn-chassis/0*	error	idle		10.11.0.25		hook failed: "ovsdb-relation-changed"
openstack-dashboard/0*	active	idle	0/lxd/13	10.11.1.95	80,443/tcp	Unit is ready

Obrázok 17 Výpis príkazu juju status - problémy vo virtuálnom prostredí

Z tohoto dôvodu sme sa dohodli s vedúcim práce, že bude použité aj ďalšie prostredie, tentokrát s reálnymi zariadeniami a upravenou topológiou aj konfiguráciou. Pokiaľ by sa nepodarilo problémy vyriešiť, testovanie by pokračovalo vo fyzickom prostredí.

Problémy sa však podarilo čiastočne vyriešiť, najmä ten s modulom „ovn-chassis“. Na obrázkoch 18 a 19 nižšie sú zobrazené výpisy príkazu **juju status** z oboch prostredí.



```
App                          Version  Status  Scale  Charm                Channel        Rev  Exposed  Message
ceph-dashboard               17.2.7   active      3  ceph-dashboard       reef/stable     49  no       Unit is ready
ceph-fs                      17.2.7   active      3  ceph-fs              reef/stable     61  no       Unit is ready
ceph-mon                     17.2.7   active      3  ceph-mon             reef/stable    192  no       Unit is ready and clustered
ceph-osd                     17.2.7   active      3  ceph-osd             reef/stable    577  no       Unit is ready (1 OSD)
ceph-radosgw                 17.2.7   active      1  ceph-radosgw         reef/stable    566  no       Unit is ready
cinder                       20.3.1   active      1  cinder               yoga/stable    664  no       Unit is ready
cinder-ceph                  20.3.1   active      1  cinder-ceph          yoga/stable    527  no       Unit is ready
cinder-mysql-router          8.0.36   active      1  mysql-router         8.0/stable     154  no       Unit is ready
dashboard-mysql-router       8.0.36   active      1  mysql-router         8.0/stable     154  no       Unit is ready
designate                    14.0.4   active      1  designate            yoga/stable    244  no       Unit is ready
designate-bind               9.16.48  active      1  designate-bind       yoga/stable    107  no       Unit is ready
glance                       24.2.1   active      1  glance               yoga/stable    594  no       Unit is ready
glance-mysql-router          8.0.36   active      1  mysql-router         8.0/stable     154  no       Unit is ready
heat                         18.0.1   active      1  heat                 yoga/stable    535  no       Unit is ready
keystone                     21.0.1   active      1  keystone             yoga/stable    669  no       Application Ready
keystone-mysql-router        8.0.36   active      1  mysql-router         8.0/stable     154  no       Unit is ready
manila                       14.1.1   active      3  manila               yoga/stable    240  no       Unit is ready
manila-ganesha               17.2.7   active      3  manila-ganesha       yoga/stable    172  no       Unit is ready
manila-ganesha-hacluster     2.0.3    active      3  hacluster            2.4/stable     131  no       Unit is ready and clustered
manila-ganesha-mysql-router  8.0.36   active      3  mysql-router         8.0/stable     154  no       Unit is ready
manila-hacluster             2.0.3    active      3  hacluster            2.4/stable     131  no       Unit is ready and clustered
manila-mysql-router          8.0.36   active      3  mysql-router         8.0/stable     154  no       Unit is ready
memcached                               active      1  memcached            latest/stable   39  no       Unit is ready
mysql-innodb-cluster         8.0.36   active      3  mysql-innodb-cluster 8.0/stable     133  no       Unit is ready: Mode: R/O, Cluster is ONLINE and can tolerate up to ONE fa
ilure.
neutron-api                  20.5.0   active      1  neutron-api          yoga/stable    568  no       Unit is ready
neutron-api-plugin-ovn       20.5.0   active      1  neutron-api-plugin-ovn yoga/stable   29  no       Unit is ready
neutron-mysql-router         8.0.36   active      1  mysql-router         8.0/stable     154  no       Unit is ready
nova-cloud-controller        25.2.1   active      1  nova-cloud-controller yoga/stable   729  no       Unit is ready
nova-compute                 25.2.1   active      3  nova-compute         yoga/stable    723  no       Unit is ready
nova-mysql-router            8.0.36   active      1  mysql-router         8.0/stable     154  no       Unit is ready
ntp                          3.5      active      3  ntp                  latest/stable   50  no       chrony: Ready
openstack-dashboard          22.1.1   active      1  openstack-dashboard  yoga/stable    648  no       Unit is ready
ovn-central                  22.03.3  active      3  ovn-central          22.03/stable   165  no       Unit is ready (northd: active)
ovn-chassis                  22.03.3  active      3  ovn-chassis          22.03/stable   222  no       Unit is ready
placement                    7.0.0    active      1  placement            yoga/stable     94  no       Unit is ready
placement-mysql-router       8.0.36   active      1  mysql-router         8.0/stable     154  no       Unit is ready
rabbitmq-server              3.8.2    active      1  rabbitmq-server      3.9/stable     183  no       Unit is ready
vault                        1.7.9    active      1  vault                1.7/stable     210  no       Unit is ready (active: true, mlock: disabled)
vault-mysql-router           8.0.36   active      1  mysql-router         8.0/stable     154  no       Unit is ready

Unit                     Workload  Agent  Machine  Public address  Ports     Message
ceph-fs/0                active    idle   0/lxd/0  172.20.1.59               Unit is ready
ceph-fs/1                active    idle   1/lxd/0  172.20.1.51               Unit is ready
ceph-fs/2*               active    idle   2/lxd/0  172.20.1.46               Unit is ready
ceph-mon/0               active    idle   0/lxd/1  172.20.1.54               Unit is ready and clustered
  ceph-dashboard/1       active    idle            172.20.1.54               Unit is ready
ceph-mon/1               active    idle   1/lxd/1  172.20.1.66               Unit is ready and clustered
  ceph-dashboard/2       active    idle            172.20.1.66               Unit is ready
ceph-mon/2*              active    idle   2/lxd/1  172.20.1.47               Unit is ready and clustered
  ceph-dashboard/0*      active    idle            172.20.1.47               Unit is ready
ceph-osd/0               active    idle   0        172.20.0.11               Unit is ready (1 OSD)
ceph-osd/1               active    idle   1        172.20.0.12               Unit is ready (1 OSD)
ceph-osd/2*              active    idle   2        172.20.0.13               Unit is ready (1 OSD)
ceph-radosgw/0*          active    idle   1/lxd/2  172.20.1.67  80/tcp       Unit is ready
cinder/0*                active    idle   0/lxd/2  172.20.1.60  8776/tcp     Unit is ready
```

Obrázok 18 Výpis príkazu juju status po nasadení na fyzických zariadeniach

Obrázok 18 vyššie zobrazuje všetky charm-y nasadené v reálnom prostredí. Z výpisu môžeme vidieť, že všetky sú v stave „*active*“. V stĺpci „*Channel*“ vidíme verzie nasadených charm-ov. Obrázok 19 zobrazuje sekciu „Unit“ z výpisu **juju status**. Na ňom sú zobrazené jednotlivé kontajnery, na ktorých bežia aj konkrétne služby daných charm-ov, spolu s IP adresami a identifikátormi, na ktorých zariadeniach sa kontajnery nachádzajú (prvé číslo je číslo zariadenia – „*Machine*“). Vo výpise sú všetky komponenty klastra Ceph v stave „active“.

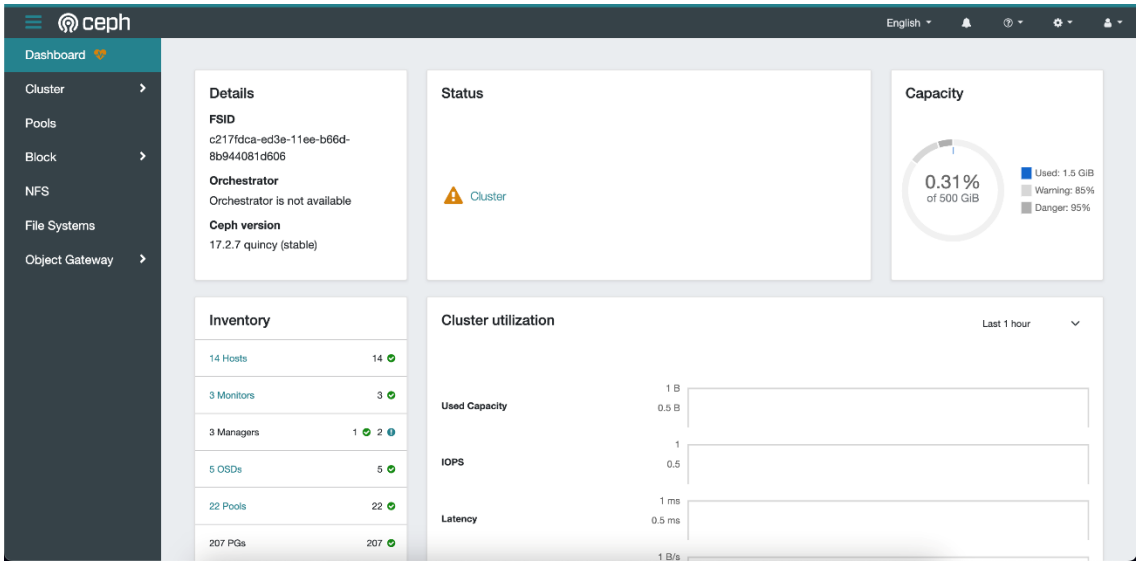
```
Unit                          Workload  Agent  Machine   Public address  Ports           Message
ceph-fs/0                     active    idle   1/lxd/0   10.11.1.141                     Unit is ready
ceph-fs/1                     active    idle   2/lxd/0   10.11.1.149                     Unit is ready
ceph-fs/2*                    active    idle   3/lxd/0   10.11.1.135                     Unit is ready
ceph-fs/3                     active    idle   4/lxd/0   10.11.1.145                     Unit is ready
ceph-fs/4                     active    idle   5/lxd/0   10.11.1.136                     Unit is ready
ceph-mon/0                    active    idle   1/lxd/1   10.11.1.142                     Unit is ready and clustered
  ceph-dashboard/1            active    idle             10.11.1.142                     Unit is ready
ceph-mon/1                    active    idle   2/lxd/1   10.11.1.150                     Unit is ready and clustered
  ceph-dashboard/2            active    idle             10.11.1.150                     Unit is ready
ceph-mon/2*                   active    idle   3/lxd/1   10.11.1.134                     Unit is ready and clustered
  ceph-dashboard/0*           active    idle             10.11.1.134                     Unit is ready
ceph-osd/0*                   active    idle   1         10.11.0.21                      Unit is ready (1 OSD)
ceph-osd/1                    active    idle   2         10.11.0.22                      Unit is ready (1 OSD)
ceph-osd/2                    active    idle   3         10.11.0.23                      Unit is ready (1 OSD)
ceph-osd/3                    active    idle   4         10.11.0.24                      Unit is ready (1 OSD)
ceph-osd/4                    active    idle   5         10.11.0.25                      Unit is ready (1 OSD)
ceph-radosgw/0*               active    idle   0/lxd/0   10.11.1.114     80/tcp          Unit is ready
cinder/0*                     active    idle   0/lxd/1   10.11.1.117     8776/tcp        Unit is ready
  cinder-ceph/0*              active    idle             10.11.1.117                     Unit is ready
  cinder-mysql-router/0*      active    idle             10.11.1.117                     Unit is ready
designate-bind/0*             active    idle   0/lxd/3   10.11.1.131                     Unit is ready
designate/0*                  error     idle   0/lxd/2   10.11.1.118     9001/tcp        hook failed: "identity-service-relation-changed"
glance/0*                     active    idle   0/lxd/4   10.11.1.119     9292/tcp        Unit is ready
  glance-mysql-router/0*      active    idle             10.11.1.119                     Unit is ready
heat/0*                       active    idle   0/lxd/5   10.11.1.122     8000,8004/tcp   Unit is ready
keystone/0*                   active    idle   0/lxd/6   10.11.1.128     5000/tcp        Unit is ready
  keystone-mysql-router/0*    active    idle             10.11.1.128                     Unit is ready
manila-ganesha/0              active    idle   1/lxd/3   10.11.1.139                     Unit is ready
  manila-ganesha-hacluster/2  active    idle             10.11.1.139                     Unit is ready and clustered
  manila-ganesha-mysql-router/2 active  idle             10.11.1.139                     Unit is ready
manila-ganesha/1              active    idle   2/lxd/3   10.11.1.147                     Unit is ready
  manila-ganesha-hacluster/4  active    idle             10.11.1.147                     Unit is ready and clustered
  manila-ganesha-mysql-router/4 active  idle             10.11.1.147                     Unit is ready
manila-ganesha/2*             active    idle   3/lxd/3   10.11.1.132                     Unit is ready
  manila-ganesha-hacluster/0* active    idle             10.11.1.132                     Unit is ready and clustered
  manila-ganesha-mysql-router/0* active idle             10.11.1.132                     Unit is ready
manila-ganesha/3              active    idle   4/lxd/2   10.11.1.146                     Unit is ready
  manila-ganesha-hacluster/3  active    idle             10.11.1.146                     Unit is ready and clustered
  manila-ganesha-mysql-router/3 active  idle             10.11.1.146                     Unit is ready
manila-ganesha/4              active    idle   5/lxd/2   10.11.1.137                     Unit is ready
  manila-ganesha-hacluster/1  active    idle             10.11.1.137                     Unit is ready and clustered
  manila-ganesha-mysql-router/1 active  idle             10.11.1.137                     Unit is ready
manila/0                      active    idle   1/lxd/2   10.11.1.140     8786/tcp        Unit is ready
  manila-hacluster/2          active    idle             10.11.1.140                     Unit is ready and clustered
  manila-mysql-router/2       active    idle             10.11.1.140                     Unit is ready
manila/1                      active    idle   2/lxd/2   10.11.1.151     8786/tcp        Unit is ready
  manila-hacluster/4          active    idle             10.11.1.151                     Unit is ready and clustered
  manila-mysql-router/4       active    idle             10.11.1.151                     Unit is ready
manila/2*                     active    idle   3/lxd/2   10.11.1.133     8786/tcp        Unit is ready
  manila-hacluster/0*         active    idle             10.11.1.133                     Unit is ready and clustered
  manila-mysql-router/0*      active    idle             10.11.1.133                     Unit is ready
manila/3                      active    idle   4/lxd/1   10.11.1.143     8786/tcp        Unit is ready
  manila-hacluster/3          active    idle             10.11.1.143                     Unit is ready and clustered
  manila-mysql-router/3       active    idle             10.11.1.143                     Unit is ready
manila/4                      active    idle   5/lxd/1   10.11.1.138     8786/tcp        Unit is ready
  manila-hacluster/1          active    idle             10.11.1.138                     Unit is ready and clustered
  manila-mysql-router/1       active    idle             10.11.1.138                     Unit is ready
memcached/0*                  active    idle   0/lxd/7   10.11.1.127     11211/tcp       Unit is ready
mysql-innodb-cluster/0        active    idle   0/lxd/8   10.11.1.115                     Unit is ready: Mode: R/O, Cluster is ONLINE and can tolerate up to ONE failure.
mysql-innodb-cluster/1        active    idle   2/lxd/4   10.11.1.148                     Unit is ready: Mode: R/O, Cluster is ONLINE and can tolerate up to ONE failure.
mysql-innodb-cluster/2*       active    idle   4/lxd/3   10.11.1.144                     Unit is ready: Mode: R/W, Cluster is ONLINE and can tolerate up to ONE failure.
neutron-api/0*                active    idle   0/lxd/9   10.11.1.123     9696/tcp        Unit is ready
  neutron-api-plugin-ovn/0*   active    idle             10.11.1.123                     Unit is ready
  neutron-mysql-router/0*     active    idle             10.11.1.123                     Unit is ready
nova-cloud-controller/0*      active    idle   0/lxd/10  10.11.1.116     8774-8775/tcp   Unit is ready
  nova-mysql-router/0*        active    idle             10.11.1.116                     Unit is ready
nova-compute/0*               active    idle   1         10.11.0.21                      Unit is ready
  ntp/1                       active    idle             10.11.0.21      123/udp         chrony: Ready
  ovn-chassis/1               active    idle             10.11.0.21                      Unit is ready
```

Obrázok 19 Výpis príkazu juju status po nasadení vo virtuálnom prostredí

Výpis využijeme na vytvorenie prihlasovacích údajov pre dashboard. Z neho zistíme IP adresu kontajnera, na ktorom služba dashboard beží a zadáme príkaz pre vytvorenie používateľa.

```
juju run ceph-dashboard/0 add-user username=admin role=administrator
```

Príkaz vypíše heslo, ktoré je potrebné si uložiť. Pridávať môžeme viac používateľov a taktiež im priradiť iné práva. Po získaní údajov sa rovnako ako v prípade manuálnej inštalácie pripojíme na adresu https://<IP_adresa>:8443 . Ten istý postup sme uplatnili aj pri nasadení na fyzických zariadeniach.



Obrázok 20 Dashboard Ceph-u po nasadení vo virtuálnom prostredí

Obrázok 20 zobrazuje stav klastra Ceph-u po nasadení. Je nasadených 5 OSD démonov, celková kapacita klastra je 500GB a klaster je v stave „*warning*“. Po priblížení kurzora na „*Cluster*“ uvidíme detajly warning-u. V tomto prípade ide o bug, kde raz za čas sa v náhodných intervaloch reštartuje manager, čo spôsobí zmenu „zdravia“ v klastri. Detailnejšie výpisy stavu klastra môžeme získať pomocou nasledujúcich príkazov na pripojenie sa do kontajnera Ceph monitora:

```
juju ssh ceph-mon/leader
sudo -u ceph ceph status
```

Ďalší bug, ktorý aktuálne existuje je, že napriek špecifikovaniu verzie Ceph-u pre nasadenie na „*reef*“, charm nasadí verziu „*quincy*“. Tieto bugy sú prítomné na nasadeniach v oboch prostrediach.

Po zvážení sme sa napokon rozhodli využiť obe prostredia pre následné testovanie konfiguračných možností Ceph-u s OpenStack-om, ktoré použijeme pre rozdielne procesy testovania a ich overenia.

6. KONFIGURAČNÉ MOŽNOSTI CEPH-U

Táto kapitola popisuje možnosti, ktoré je možné vykonať po nasadení klastra. Ceph poskytuje robustné mechanizmy pre optimalizovanie správy dát. Zameriame sa na základné procesy konfigurácie a správy klastra ako celku. Napriek opisu procesov na základe integrácie s OpenStack-om, všetky ďalej popísané procesy je možné uplatniť aj v prípade samostatného klastra.

6.1 Pridávanie OSD diskov

Navýšenie kapacity klastra Ceph si vyžaduje pridanie nových diskov do existujúcich node-ov, prípadne pridanie nových node-ov. Tento proces je kľúčový pre rozšírenie úložných možností klastra, a tým aj možné zvýšenie redundancie dát. Okrem navýšenia kapacity a redundancie dát sa rovnomernejšie rozdeľuje pracovné zaťaženie v rámci klastra. Proces pridávania zahŕňa identifikáciu vhodného node-u pre disk, prípravu disku a jeho integráciu do klastra pomocou nástrojov Ceph OSD.

Pridávanie diskov do klastra, ktorý bol nasadený pomocou Juju, je špecifické. V samostatnom klastri identifikácia servera, do ktorého sme pridali disk, je priamočiará a celú konfiguráciu vykonáme na danom node-e. V prípade Juju je to komplikovanejšie. Pri bundle súbore sme definovali tzv. aplikácie. Aplikácie sú spoločné pre celý model v Juju. Dôležitý pre nás je tzv. „unit“. V prípade Ceph-u, unit môžeme prirovnať k jednému konkrétnemu reálnemu node-u. Napriek tomu, že Juju by dovolilo vytvoriť ďalší unit na tom istom servery (tzv. „machine“), tento postup nie je odporúčaný.

6.1.1 Pridanie nového unit-u

Pridanie nového unit-u pracuje s predpokladom, že máme „machine“ už pripravený. Pre pridanie unitu použijeme príkaz **juju add-unit** s príslušnými flag-mi.

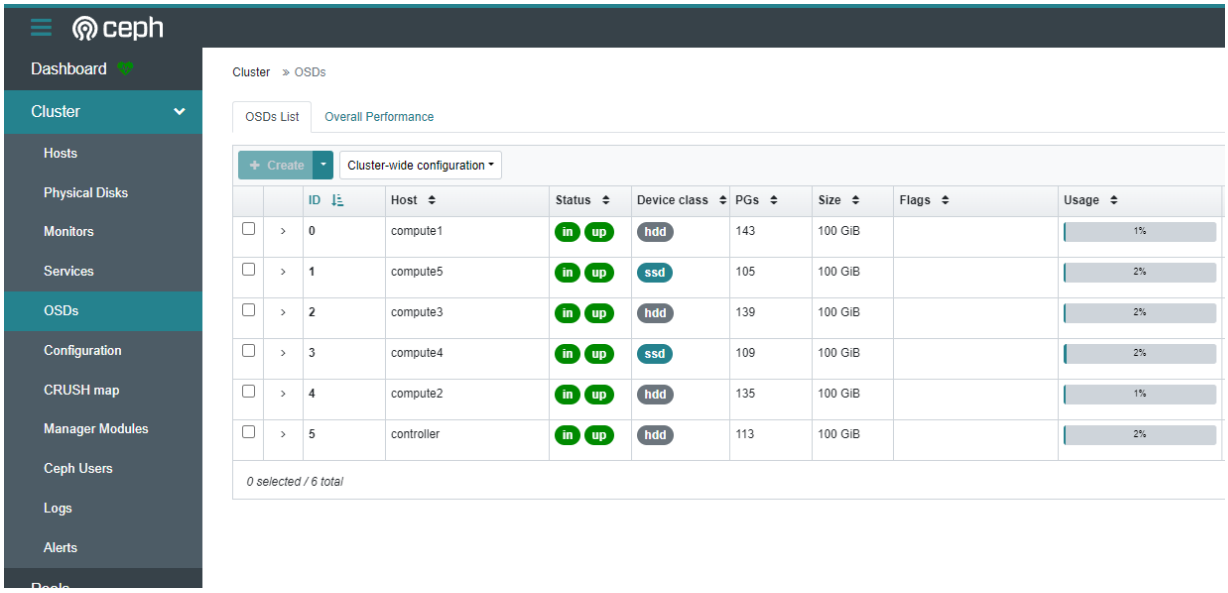
```
juju add-unit ceph-osd --m <nazov_modelu> --n <pocet> --to <id_machine>
```

Flag **--m** priradí model, do ktorého chceme unit pridať, **--n** určuje počet unit-ov a **--to** určí machine, na ktorý sa unit nasadí [52].

```
Unit                      Workload  Agent  Machine   Public address  Ports           Message
ceph-fs/0                 active    idle   1/lxd/0   10.11.1.141                     Unit is ready
ceph-fs/1                 active    idle   2/lxd/0   10.11.1.149                     Unit is ready
ceph-fs/2*                active    idle   3/lxd/0   10.11.1.135                     Unit is ready
ceph-fs/3                 active    idle   4/lxd/0   10.11.1.145                     Unit is ready
ceph-fs/4                 active    idle   5/lxd/0   10.11.1.136                     Unit is ready
ceph-mon/0                active    idle   1/lxd/1   10.11.1.142                     Unit is ready and clustered
  ceph-dashboard/1        active    idle             10.11.1.142                     Unit is ready
ceph-mon/1                active    idle   2/lxd/1   10.11.1.150                     Unit is ready and clustered
  ceph-dashboard/2        active    idle             10.11.1.150                     Unit is ready
ceph-mon/2*               active    idle   3/lxd/1   10.11.1.134                     Unit is ready and clustered
  ceph-dashboard/0*       active    idle             10.11.1.134                     Unit is ready
ceph-osd/0*               active    idle   1         10.11.0.21                      Unit is ready (1 OSD)
ceph-osd/1                active    idle   2         10.11.0.22                      Unit is ready (1 OSD)
ceph-osd/2                active    idle   3         10.11.0.23                      Unit is ready (1 OSD)
ceph-osd/3                active    idle   4         10.11.0.24                      Unit is ready (1 OSD)
ceph-osd/4                active    idle   5         10.11.0.25                      Unit is ready (1 OSD)
ceph-osd/9                active    idle   0         10.11.0.11                      Unit is ready (1 OSD)
ceph-radosgw/0*           active    idle   0/lxd/0   10.11.1.114     80/tcp          Unit is ready
cinder/0*                 active    idle   0/lxd/1   10.11.1.117     8776/tcp        Unit is ready
  cinder-ceph/0*          active    idle             10.11.1.117                     Unit is ready
  cinder-mysql-router/0*  active    idle             10.11.1.117                     Unit is ready
designate-bind/0*         active    idle   0/lxd/3   10.11.1.131                     Unit is ready
designate/0*              error     idle   0/lxd/2   10.11.1.118     9001/tcp        hook failed: "identity-service-relation-changed"
glance/0*                 active    idle   0/lxd/4   10.11.1.119     9292/tcp        Unit is ready
  glance-mysql-router/0*  active    idle             10.11.1.119                     Unit is ready
heat/0*                   active    idle   0/lxd/5   10.11.1.122     8000,8004/tcp   Unit is ready
keystone/0*               active    idle   0/lxd/6   10.11.1.128     5000/tcp        Unit is ready
```

Obrázok 21 Overenie úspešného nasadenia unit-u

Po nasadení unit-u sa aplikuje aktuálna konfigurácia pre Ceph-Osd charm, ktorú si môžeme zobraziť pomocou príkazu **juju config ceph-osd**. Tá obsahuje všetky disky, ktoré sme definovali už v bundle súbore v časti „osd-devices:“. Keďže sme určili zariadenie /dev/sdb a „node1“ tento disk obsahuje, unit automaticky nasadí OSD démon pre daný disk. Ako je zobrazené na Obrázku 21, nasadenie prebehlo v poriadku a môžeme vidieť pridaný unit „ceph-osd/9“. Overenie priamo v klastri Ceph môžeme vykonať pomocou CLI príkazu **sudo -u ceph ceph osd tree** po prihlásení do unit-u monitora, alebo v dashboarde – Obrázok 22.



Obrázok 22 Overenie pridania node-u a disku do klastra v dashboard-e Ceph-u

Pre pridanie diskov nedefinovaných v konfiguračnom súbore charm-u postupujeme ako v prípade pridávania diskov do už existujúceho unit-u.

6.1.2 Pridanie diskov do existujúceho OSD unit-u

V prípade pridávania diskov do už existujúceho unit-u použijeme tzv. „actions“, ktoré môžeme nájsť na stránke charmhub.io [48]. Použijeme príkaz „add-disk“ a zadefinujeme disky pomocou parametra **osd-devices**.

```
juju run ceph-osd/<#osd> add-disk osd-devices=/dev/<označenie_disku/diskov>
```

Po zadaní príkazu **juju status**, by sa mal pri príslušnom unit-e zobraziť stav „*maintenance*“ a správa o inicializovaní zadaného disku (Obrázok 23). Ak sme zadali viac diskov, disky sa inicializujú postupne a v správe sa zobrazuje práve aktuálne inicializovaný disk [52].

Unit	Workload	Agent	Machine	Public address	Ports	Message
ceph-fs/0	active	idle	0/lxd/0	172.20.1.59		Unit is ready
ceph-fs/1	active	idle	1/lxd/0	172.20.1.51		Unit is ready
ceph-fs/2*	active	idle	2/lxd/0	172.20.1.46		Unit is ready
ceph-mon/0	active	idle	0/lxd/1	172.20.1.54		Unit is ready and clustered
ceph-dashboard/1	active	idle		172.20.1.54		Unit is ready
ceph-mon/1	active	idle	1/lxd/1	172.20.1.66		Unit is ready and clustered
ceph-dashboard/2	active	idle		172.20.1.66		Unit is ready
ceph-mon/2*	active	executing	2/lxd/1	172.20.1.47		Unit is ready and clustered
ceph-dashboard/0*	active	idle		172.20.1.47		Unit is ready
ceph-osd/0	maintenance	idle	0	172.20.0.11		Initializing device /dev/sdc
ceph-osd/1	active	idle	1	172.20.0.12		Unit is ready (1 OSD)
ceph-osd/2*	active	idle	2	172.20.0.13		Unit is ready (1 OSD)
ceph-radosgw/0*	active	idle	1/lxd/2	172.20.1.67	80/tcp	Unit is ready
cinder/0*	active	idle	0/lxd/2	172.20.1.60	8776/tcp	Unit is ready
cinder-ceph/0*	active	idle		172.20.1.60		Unit is ready
cinder-mysql-router/0*	active	idle		172.20.1.60		Unit is ready
designate-bind/0*	active	idle	0/lxd/4	172.20.1.44		Unit is ready
designate/0*	active	idle	0/lxd/3	172.20.1.42	9001/tcp	Unit is ready
glance/0*	active	idle	2/lxd/2	172.20.1.45	9292/tcp	Unit is ready
glance-mysql-router/0*	active	idle		172.20.1.45		Unit is ready
heat/0*	active	idle	1/lxd/3	172.20.1.72	8000,8004/tcp	Unit is ready

Obrázok 23 Inicializácia disku po pridaní do unit-u

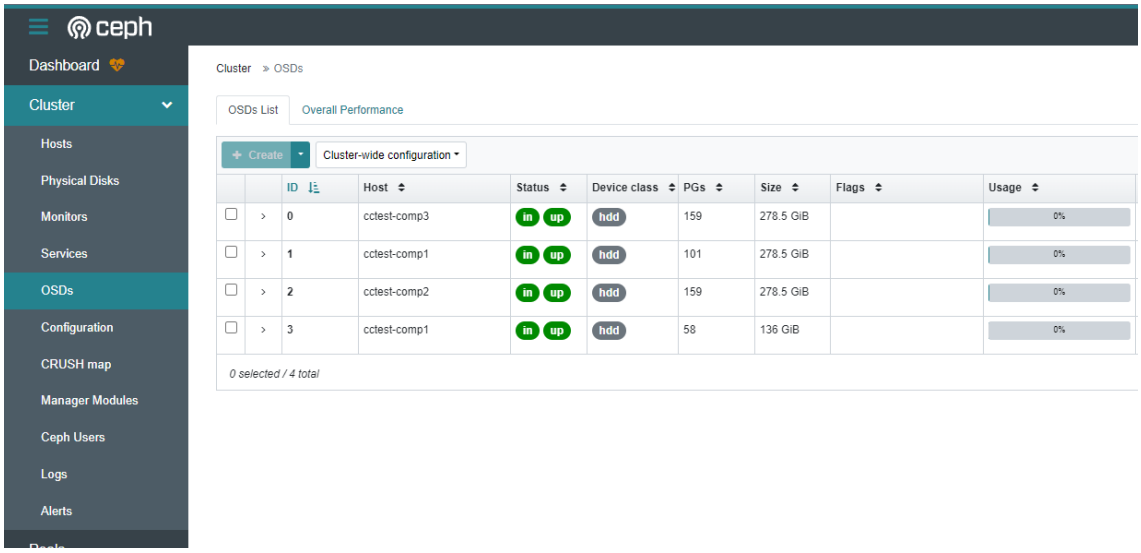
Po úspešnom skončení inicializácie sa stav vráti do stavu „active“. Pri porovnaní obrázkov 24 a 18 v stĺpci „Message“ na Obrázku 18 unit „ceph-osd/0“ uvádza 1 OSD. Na Obrázku 24 rovnaký unit, podľa očakávania, uvádza 2 OSD, čo znamená úspešné pridanie disku.

Unit	Workload	Agent	Machine	Public address	Ports	Message
ceph-fs/0	active	idle	0/lxd/0	172.20.1.59		Unit is ready
ceph-fs/1	active	idle	1/lxd/0	172.20.1.51		Unit is ready
ceph-fs/2*	active	idle	2/lxd/0	172.20.1.46		Unit is ready
ceph-mon/0	active	idle	0/lxd/1	172.20.1.54		Unit is ready and clustered
ceph-dashboard/1	active	idle		172.20.1.54		Unit is ready
ceph-mon/1	active	idle	1/lxd/1	172.20.1.66		Unit is ready and clustered
ceph-dashboard/2	active	idle		172.20.1.66		Unit is ready
ceph-mon/2*	active	idle	2/lxd/1	172.20.1.47		Unit is ready and clustered
ceph-dashboard/0*	active	idle		172.20.1.47		Unit is ready
ceph-osd/0	active	idle	0	172.20.0.11		Unit is ready (2 OSD)
ceph-osd/1	active	idle	1	172.20.0.12		Unit is ready (1 OSD)
ceph-osd/2*	active	idle	2	172.20.0.13		Unit is ready (1 OSD)
ceph-radosgw/0*	active	idle	1/lxd/2	172.20.1.67	80/tcp	Unit is ready
cinder/0*	active	idle	0/lxd/2	172.20.1.60	8776/tcp	Unit is ready
cinder-ceph/0*	active	idle		172.20.1.60		Unit is ready
cinder-mysql-router/0*	active	idle		172.20.1.60		Unit is ready

Obrázok 24 Výpis príkazu juju status po pridaní disku do existujúceho unit-u

Rovnako ako pri pridávaní celého unit-u do klastra môžeme overenie vykonať aj priamo v Ceph-e pomocou CLI príkazov, alebo v dashboard-e.

Rozdielom je, že do tabuľky OSD nepribudne nový node, ale jeden node (cctest-comp1) bude obsahovať 2 OSD démony, ako je zobrazené na Obrázku 25.



Obrázok 25 Overenie pridania disku do klastra v dashboard-e Ceph-u

Pokiaľ pridávame disky do node-ov spôsobom, že do každého node-u pridáme rovnaký počet diskov, pridávanie nie je nutné robiť týmto spôsobom pre každý disk. V takomto prípade môžeme pridávanie nechať na samotné Juju zmenou globálnej konfigurácie. Mená diskov však musia byť na všetkých node-och rovnaké.

```
juju config ceph-osd osd-devices="$disky"
```

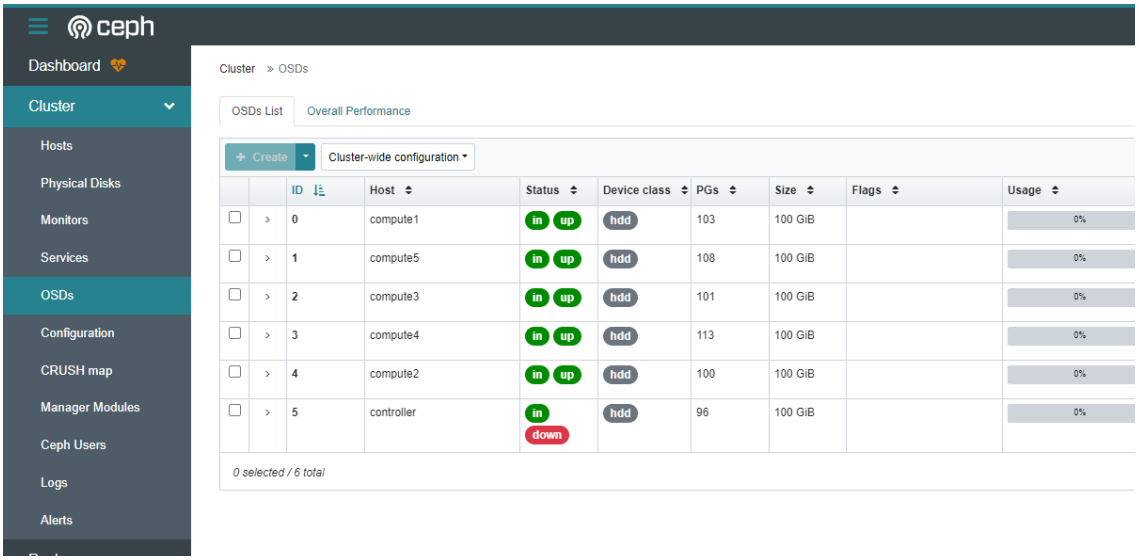
Premennú **disky** nahradíme menami diskov, vrátane už existujúcich, oddelené medzerou [52].

6.2 Výmena a odstránenie OSD diskov

Výmena a odstraňovanie diskov z Ceph klastra pomocou Juju je takmer rovnaké. Vymenenie disku sa líši tým, že v takom prípade sa predpokladá, že fyzický disk zlyhal a v pláne máme tento disk nahradiť. Ako prvé identifikujeme disk, ktorý zlyhal. Pre zistenie môžeme použiť príkaz **sudo -u ceph ceph osd tree** po prihlásení do monitora, alebo dashboard, kde disk ktorý zlyhal, bude v stave „down“ (Obrázok 26). Daný disk označíme v klastri ako „out“, a následne disk odstránime pomocou príkazu **remove-disk**.

```
juju run ceph-osd/<#osd> osd-out osds=<$ids>
juju run ceph-osd/<#osd> remove-disk osd-ids=<$ids>
```

Ak zlyhalo viac OSD, je možné ich označiť a odstrániť naraz. Vtedy je potrebné premennú **$ids** nahradiť za OSD ID diskov oddelených čiarkou [52].



Obrázok 26 Zobrazenie zlyhania OSD disku v dashboard-e

Ak chceme disk len odstrániť, je nutné nastaviť aj parameter purge na „*true*“. Niekedy môže dôjsť k timeout-u a zlyhaniu bezpečného odstránenia. Vtedy je možné pridať parameter force, ktorý nastavíme rovnako na „*true*“ [52].

```
juju run ceph-osd/<#osd> remove-disk osd-ids=<$ids> purge=true /force=true/
```

V situácií, kedy chceme disk nahradiť, tieto parametre neuvádzame. Po odstránení uvedených OSD sa ďalej postupuje ako v prípade klasického pridávania disku do existujúceho unit-u.

6.3 Ceph multi-backend pre Cinder

Cinder je modul poskytujúci blokové úložisko pre OpenStack. Spravuje a zabezpečuje obrazy úložiska tzv. „volumes“, ktoré sú používané ako virtuálne disky pre inštancie OpenStack-u. Implementácia viacerých backend-ov v Cinder-i je dôležitá, pretože umožňuje používať rôzne technológie v rámci jedného cloudového prostredia. Táto konfigurácia umožňuje používateľom splniť rôzne požiadavky na výkon a zároveň zabezpečuje, že úložisko dokáže podporovať širokú škálu aplikácií. Okrem toho môže použitie viacerých backend-ov zlepšiť odolnosť systému rozdelením zdrojov úložiska, čím sa redukuje vplyv akéhokoľvek jednotlivého bodu zlyhania.

Viac pool-ov umožňuje efektívnejšie využiť dostupnú infraštruktúru. Pri vytváraní klastra Ceph, pomocou Juju charm-ov sa vytvára množstvo prednastavených pool-ov, ktoré majú svoju rolu. Dá sa povedať, že každá služba, ktorú sme nakonfigurovali pre integráciu s OpenStack-om (Manila, Glance, Cinder,..), má svoj pridelený pool. Tieto pool-y zdieľajú všetky OSD disky spoločne. V jednoduchej infraštruktúre, akou napr. bolo testovacie prostredie, alebo ktorá využíva rovnaký typ fyzických diskov problém nenastáva. Avšak tento koncept v prípade komplexnejších požiadaviek, napr. nasadenie OpenStack-u alebo využívanie rôznych typov diskov (HDD, SSD), už neposkytuje dostatočnú efektivitu a odolnosť. V prípade rôznych typov diskov, budú SSD disky, (resp. PGs na týchto diskoch) vždy limitované rýchlosťami PGs na HDD diskoch [27]. Zdieľanie OSD pre všetky pool-y v prípade výpadku jedného prípadne viacerých OSD znamená, že dochádza k ovplyvneniu viac než jednej služby.

6.3.1 Vytvorenie pool-ov

Ako demonštráciu použijeme najzákladnejší typ rozdelenia úložiska. Vytvoríme jeden pool, ktorý bude poskytovaný čisto SSD diskami, a druhý pool HDD diskami. Týmto spôsobom je možné efektívnejšie priradiť úložisko na základe požiadaviek danej inštancie. Pre vytvorenie pool-ov je možné použiť Juju actions, ale pre následnú konfiguráciu už actions neposkytujú žiadne možnosti. Z tohoto dôvodu budeme celú konfiguráciu, vrátane vytvorenia pool-u, aplikovať na unit-e hlavného monitor-a priamo v Ceph-e.

Prvým krokom je klasifikácia diskov do tried. Ceph je schopný pri inštalácií rozlíšiť HDD a SSD disky. Avšak klasifikácia môže prebehnúť ľubovoľným spôsobom. Príkazom **sudo -u ceph ceph osd tree** zistíme OSD ID a aktuálnu triedu konkrétnych diskov (Obrázok 27).

```
juju ssh ceph-mon/leader
sudo -u ceph ceph osd tree
```

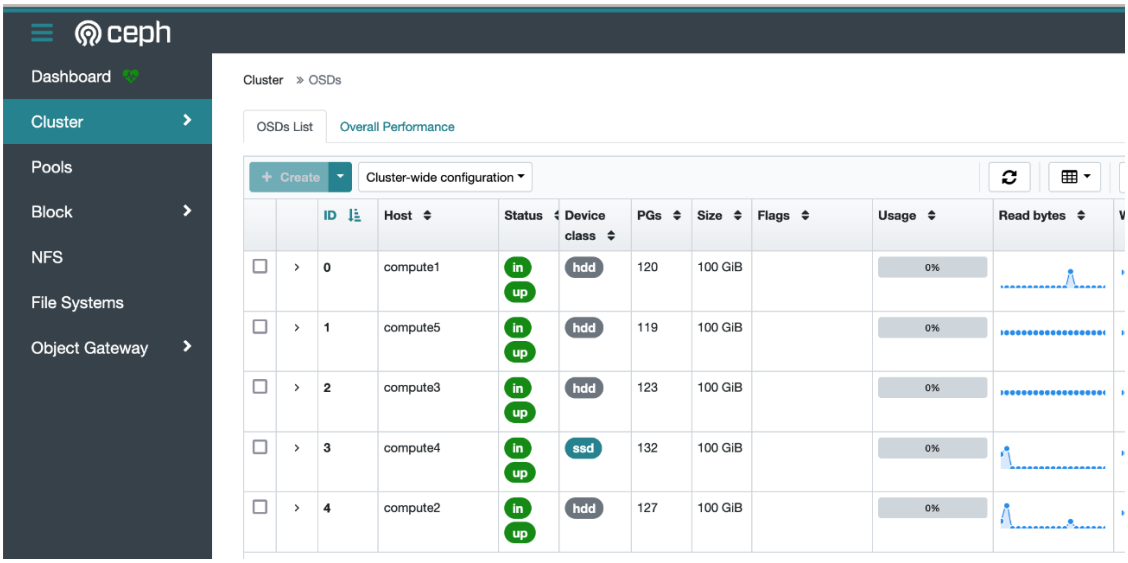
```
ubuntu@juju-befae6-3-lxd-1:~$ sudo -u ceph ceph osd tree
ID   CLASS  WEIGHT   TYPE NAME           STATUS  REWEIGHT  PRI-AFF
 -1         0.58612  root default
 -3         0.09769      host compute1
  0    hdd  0.09769          osd.0           up   1.00000  1.00000
-11         0.09769      host compute2
  4    hdd  0.09769          osd.4           up   1.00000  1.00000
 -7         0.09769      host compute3
  2    hdd  0.09769          osd.2           up   1.00000  1.00000
 -9         0.09769      host compute4
  3    ssd  0.09769          osd.3           up   1.00000  1.00000
 -5         0.09769      host compute5
  1    ssd  0.09769          osd.1           up   1.00000  1.00000
```

Obrázok 27 Výpis príkazu ceph osd tree

Keďže v našom prípade disky sú virtuálne, všetky disky boli označené triedou „hdd“. Pre otestovanie vyhradíme 2 disky, konkrétne z node-ov 4 a 5, ktoré klasifikujeme ako SSD disky. Tieto OSD mali ID 1 a 3. Pre zmenu triedy disku musíme najprv pôvodnú triedu z OSD odstrániť a následne priradiť novú. Ako už bolo spomenuté, názov triedy je ľubovoľný. Pre náš účel zvolíme názov „ssd“.

```
sudo -u ceph ceph osd crush rm-device-class osd.<#osd>
sudo -u ceph ceph osd crush set-device-class <nazov_triedy> osd.<#osd>
```

Overiť klasifikáciu diskov môžeme priamo buď v CLI, alebo v dashboard-e.



Obrázok 28 Overenie zmeny triedy OSD

Vyhradenie OSD pre jednotlivé pool-y sa realizuje pomocou pravidiel algoritmu CRUSH. Vytvoríme pravidlo „*SSD_allocate*“, ktoré nastavíme tak, aby objekty, ktoré sú určené pre zápis do pool-u, na ktorom je toto pravidlo aplikované, boli zapísané práve na OSD disky, klasifikované do špecifikovanej triedy (ssd). Pool môže mať súčasne aplikované iba jedno pravidlo v danom čase.

```
sudo -u ceph ceph osd crush rule create-replicated <nazov_pravidla> default \ host ssd
```

Príkaz vytvorí replikované úložisko. Parameter **default** určuje, že pravidlo platí globálne a nie je určené len pre konkrétny region, rack a pod. . Parameter **host** určuje prioritu, kde sa snaží algoritmus tvoriť repliky a parameter **ssd** určuje názov triedy (bucket-u).

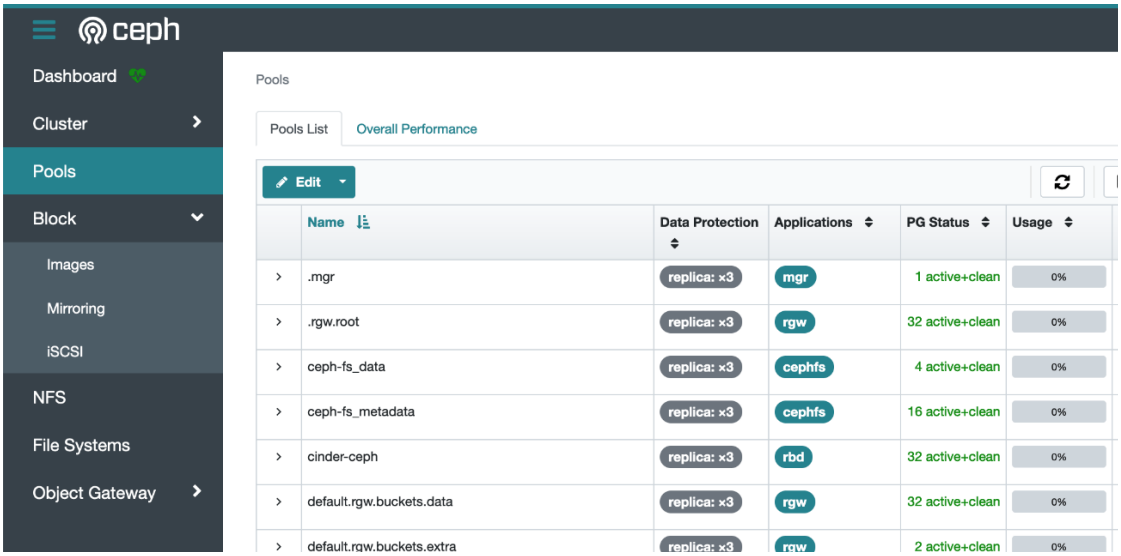
Alternatívou k tomuto postupu je vytvorenie bucket-u. V našom prípade, sme však využili fakt, že Ceph dopredu vytvára predvolené buckety „*default_hdd*“ a „*default_ssd*“. Ak nastavíme triedu na „*ssd*“, disk sa automaticky presunie do predvoleného bucket-u. Administrátor má však možnosť definovať vlastné bucket-y, a tie následne aplikovať na jednotlivé CRUSH pravidlá. Všetky bucket-y je možné exportovať v JSON formáte príkazom **sudo -u ceph ceph osd crush dump**.

Ďalším krokom je vytvorenie pool-u a nakonfigurovanie základných nastavení.

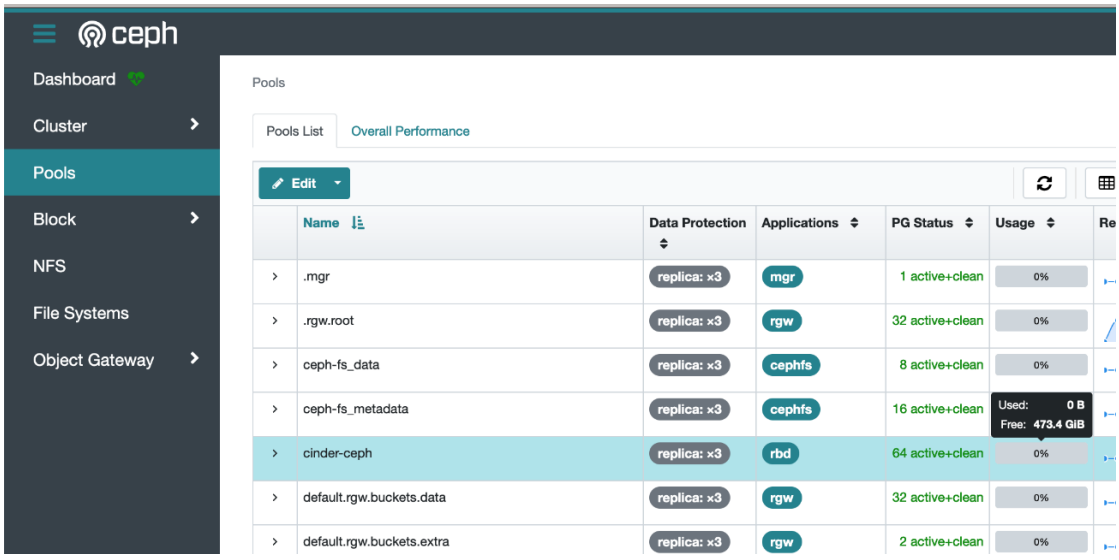
```
sudo -u ceph ceph osd pool create <nazov_poolu> replicated <pravidlo_CRUSH>
sudo -u ceph ceph osd pool set <nazov_poolu> size 2
sudo -u ceph ceph osd pool set <nazov_poolu> pg_autoscale_mode on
sudo -u ceph ceph osd pool application enable <nazov_poolu> <$aplikacia>
```

Príkazom **pool create** vytvoríme samotný pool a parameter **replicated** určuje typ redundancie pool-u. Príkazmi **pool set** nastavujeme jednotlivé nastavenia v rámci pool-u. Parametrom **size** určujeme faktor replikácie. Parameter **pg_autoscale_mode** nastavuje automatické rozširovanie počtu PGs v pool-e. Príkazom **pool application enable** nastavujeme typ úložiska, pre ktorý je pool určený, vo forme rozhraní (rbd - blokové, rgw - objektové, cephfs – súborový) určený premennou **$aplikacia**. Toto nastavenie je povinné a bez neho pool nebude funkčný.

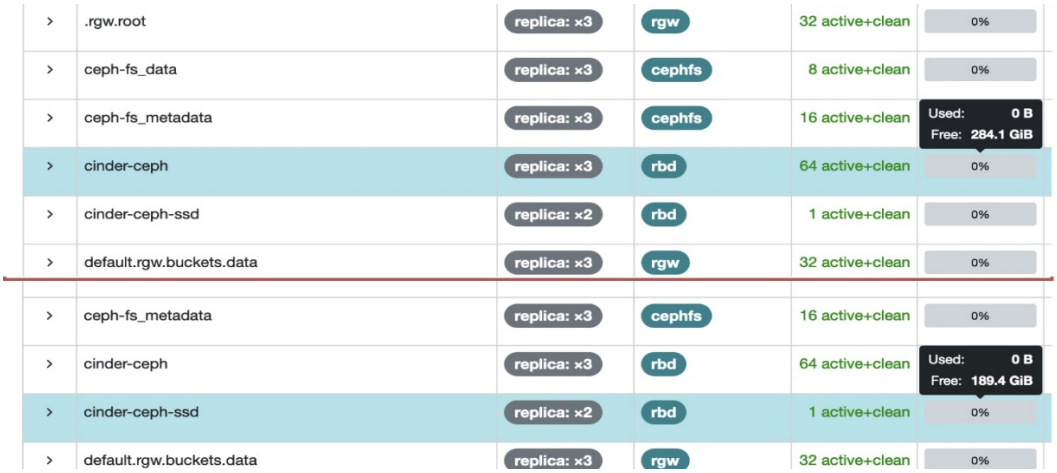
Rovnaký postup použijeme aj pre HDD pool. Konfigurácia HDD pool-u je však uľahčená o zmenu triedy OSD a vytvorenie pool-u, nakoľko trieda už je „hdd“ a pool je vytvorený pomocou Cinder-Ceph charm-u. Overenie vytvorenia pool-ov môžeme vykonať pomocou príkazu **sudo -u ceph ceph osd pool ls**, alebo v dashboard-e v záložke „Pools“.



Obrázok 29 Pool-y v klastri Ceph pred pridaním pool-u



Obrázok 30 Kapacita pool-u cinder-ceph pred použitím CRUSH pravidla



Obrázok 31 Kapacita pool-ov po aplikovaní pravidiel

Obrázok 31 slúži ako overenie vytvorenia a priradenia aplikácie pre pool „cinder-ceph-ssd“ a zmeny kapacity po aplikovaní pravidiel CRUSH pre jednotlivé pool-y.

Je však potrebné poukázať na fakt, že takto vytvorený pool „cinder-ceph-ssd“ má k dispozícii iba jednu PG. Preto je potrebné vhodne nastaviť počet PGs pre daný pool. Hodnoty PGs by mali byť mocniny čísla 2, ale nie je to nutné.

```
sudo -u ceph ceph osd pool set cinder-ceph-ssd pg_num <počet_PGs>
```

Ak by sa jednalo o skutočné SSD disky, je potrebné spustiť funkciu TRIM nasledovným príkazom:

```
sudo -u ceph ceph config set osd bluestore_volume_selection_policy \ spdk_dmcrypt_discard
```

	Name	Data Protection	Applications	PG Status
›	.mgr	replica: x3	mgr	1 active+clean
›	.rgw.root	replica: x3	rgw	32 active+clean
›	ceph-fs_data	replica: x3	cephfs	8 active+clean
›	ceph-fs_metadata	replica: x3	cephfs	16 active+clean
›	cinder-ceph	replica: x3	rbd	64 active+clean
›	cinder-ceph-ssd	replica: x2	rbd	32 active+clean
›	default.rgw.buckets.data	replica: x3	rgw	32 active+clean

Obrázok 32 Overenie zmeny počtu PGs

6.3.2 Priradenie pool-ov k modulu Cinder

Nastavenie multi-backend-u pre Cinder rozdelíme na dva kroky. Prvým krokom je zmena konfiguračného súboru **/etc/cinder/cinder.conf** pre modul Cinder. Ten nájdeme v unit-e „*cinder/0*“. Juju už prednastavilo tento súbor pre OpenStack a vďaka charmu Cinder-Ceph nastavila Ceph ako backend so všetkými parametrami. Tento fakt využijeme pre nastavenie ďalšieho pool-u/pool-ov. Všetky pool-y, ktoré sú vytvorené a povolené v tzv. kontexte v konfiguračnom súbore Cinder-u, sú automaticky dostupné pre Cinder. Kontextom nazývame mená alebo kľúčové slová v hranatých zátvorkách vo vnútri konfiguračného súboru. Skopírujeme už dopredu vytvorený kontext „*cinder-ceph*“, ktorý vytvorilo Juju pri nasadení charm-om Cinder-Ceph a postupne upravíme jednotlivé jeho hodnoty.

```
#pôvodná konfigurácia súboru /etc/cinder/cinder.conf
#...

enabled_backends = cinder-ceph

[backend_defaults]

[cinder-ceph]
volume_backend_name = cinder-ceph
volume_driver = cinder.volume.drivers.rbd.RBDDriver
rbd_pool = cinder-ceph
rbd_user = cinder-ceph
rbd_secret_uuid = 21696ef1-d740-4f59-9eb7-70799c8d9ded
rbd_ceph_conf = /var/lib/charm/cinder-ceph/ceph.conf
report_discard_supported = True
rbd_exclusive_cinder_pool = True
rbd_flatten_volume_from_snapshot = False

#...
```

V našom prípade túto konfiguráciu skopírujeme, vložíme a mierne upravíme, aby obsahovala ďalší kontext.

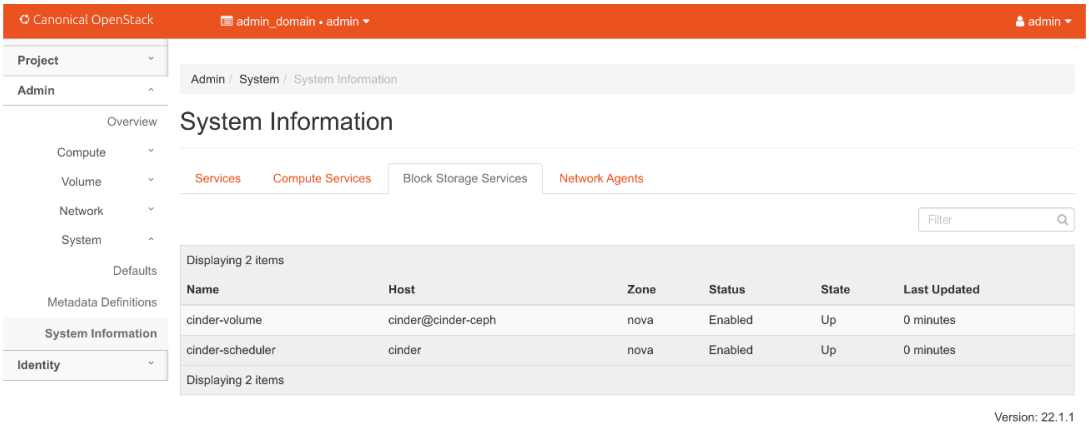
```
#pôvodná konfigurácia súboru /etc/cinder/cinder.conf
#...

enabled_backends = cinder-ceph,<nazov_dalsieho_pool-u>

[backend_defaults]

[cinder-ceph]
volume_backend_name = cinder-ceph
volume_driver = cinder.volume.drivers.rbd.RBDDriver
rbd_pool = cinder-ceph
rbd_user = cinder-ceph
rbd_secret_uuid = 21696ef1-d740-4f59-9eb7-70799c8d9ded
rbd_ceph_conf = /var/lib/charm/cinder-ceph/ceph.conf
report_discard_supported = True
rbd_exclusive_cinder_pool = True
rbd_flatten_volume_from_snapshot = False

[<nazov_noveho_kontextu>]
volume_backend_name = <nazov_pool-u_pre_openstack>
volume_driver = cinder.volume.drivers.rbd.RBDDriver
rbd_pool = <nazov_pool-u_vytvoreneho_v_cephe>
rbd_user = cinder-ceph
rbd_secret_uuid = 21696ef1-d740-4f59-9eb7-70799c8d9ded
rbd_ceph_conf = /var/lib/charm/cinder-ceph/ceph.conf
report_discard_supported = True
rbd_exclusive_cinder_pool = True
rbd_flatten_volume_from_snapshot = False


#...
```

Napriek tomu, že názvy kontextu, a parametrov `volume_backend_name` a `rbd_pool` nemusia byť rovnaké, je to best practice. Rovnaké musia byť len názvy pool-ov v Ceph-e a v parametri `rbd_pool`. V našom prípade teda všetky tri názvy nastavíme ako „*ceph-cinder-ssd*“. Potrebné je upraviť aj parameter `enabled_backends`, kde pridáme názov kontextu. Jednotlivé kontexty oddeľujeme čiarkou bez medzery. Týmto spôsobom môžeme pridať ľubovoľný počet pool-ov

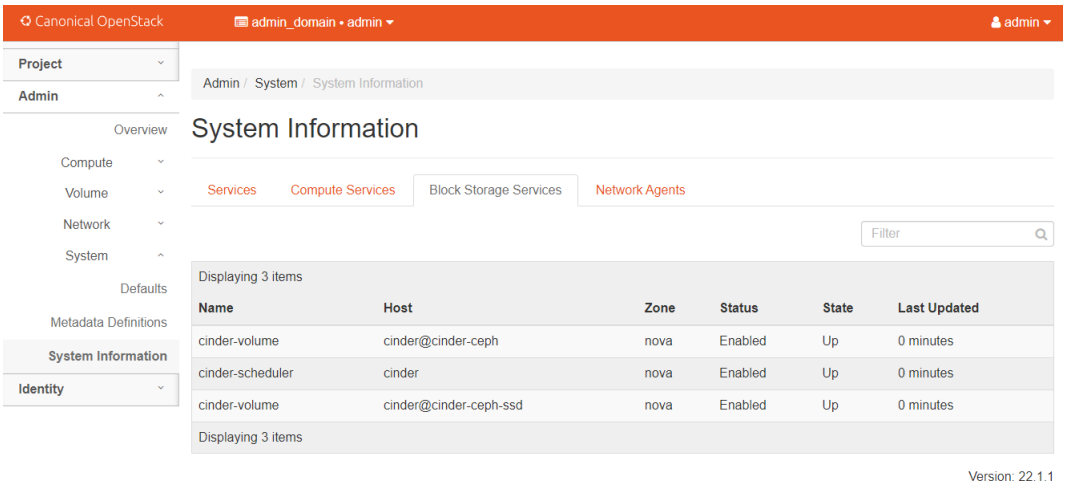
ako backend pre Cinder. Po týchto úpravách je potrebné reštartovať všetky služby modulu Cinder.

```
sudo systemctl restart cinder-volume.service
sudo systemctl restart cinder-scheduler.service
```

Ak po reštartovaní týchto služieb sa obe služby spustia a sú aktívne, konfigurácia je správna. Keby došlo k chybe v konfigurácií, alebo by sme priradili nesprávne údaje pre autentifikáciu, služby cinder-u by vrátili chybu a zlyhali by. Zadanie neexistujúceho názvu pool-u v parametri **rbd_pool** nevráti chybu, ale pri pokuse čítať alebo zapísať dáta do daného pool-u, pokus zlyhá. Tieto zlyhania sú zaznamenávané v logoch služby „*ceph-volume*“. Overenie môžeme vykonať pomocou príkazov v CLI alebo v dashboard-e OpenStack-u [53].



Obrázok 33 Stav pred pridaním backend-u v dashboard-e



Obrázok 34 Overenie pridania backend-u v dashboard-e

```
kysel@kysel-dp-maas:~$ cinder get-pools
+----------+--------------------------------+
| Property | Value                          |
+----------+--------------------------------+
| name     | cinder@cinder-ceph#cinder-ceph |
+----------+--------------------------------+
+----------+----------------------------------------+
| Property | Value                                  |
+----------+----------------------------------------+
| name     | cinder@cinder-ceph-ssd#cinder-ceph-ssd |
+----------+----------------------------------------+
kysel@kysel-dp-maas:~$
```

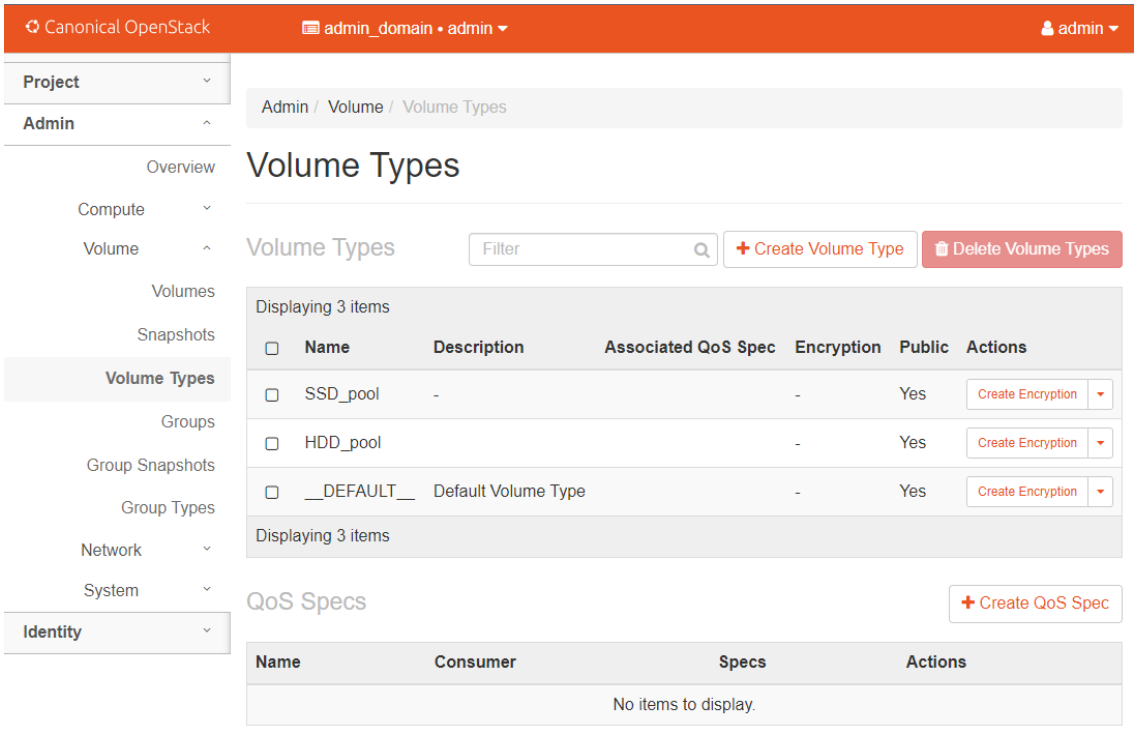
Obrázok 35 Overenie pridania backend-u v CLI

6.3.3 Nastavenie OpenStack-u pre využitie viac pool-ov

Druhým krokom pre nastavenie multi-backend-u pre Cinder je samotné nastavenie OpenStack-u tak, aby bolo možné vybrať si aký pool chceme použiť pre daný volume alebo inštanciu. Pre dosiahnutie tohoto cieľa potrebujeme nastaviť tzv. „*volume types*“. Vytvoriť volume types môžeme pomocou CLI príkazov alebo v dashboard-e. Nastavenie pomocou dashboard-u je mierne prehľadnejšie, preto zvolíme tento spôsob.

Po prihlásení do dashboard-u OpenStack-u prejdeme do záložky **Admin -> Volume -> Volume Types**. Klikneme na tlačidlo **Create Volume Type**. Zadáme názov a overíme, či je zaškrtnutá možnosť „Public“. Klikneme na tlačidlo **Create Volume Type** a počkáme, kým sa vytvorí.

Po vytvorení je potrebné ešte určiť backend pre daný volume type. Napravo klikneme na šípku smerujúcu dole vedľa tlačidla **Create Encryption** a z menu vyberieme možnosť **View Extra Specs**. Klikneme na tlačidlo **Create**. Do poľa „*Key*“ zadáme parameter „*volume_backend_name*“ a do poľa „*Value*“ zadáme názov backend-u, ktorý sme priradili v rovnakom parametri v konfiguračnom súbore Cinder-u. Je dôležité, aby tieto názvy boli rovnaké. Následne klikneme na tlačidlo **Create**. Týmto spôsobom vytvoríme jeden volume type pre každý backend.



Obrázok 36 Overenie nastavenia Volume Types v dashboard-e

Pre overenie môžeme rovnako použiť terminál. Príkazom **openstack volume type list** je možné zobraziť všetky volume type v OpenStack-u. Pre zobrazenie podrobnejších informácií pre daný volume type je k dispozícii príkaz **openstack volume type show <nazov_volume_type>**.

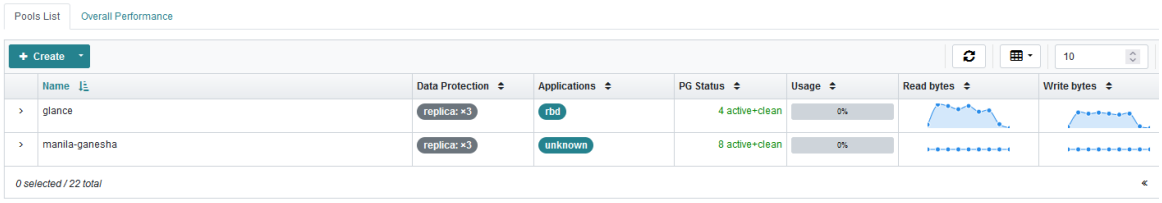
```
kysel@kysel-dp-maas:~$ openstack volume type list
+--------------------------------------+-------------+-----------+
| ID                                   | Name        | Is Public |
+--------------------------------------+-------------+-----------+
| 0c29c9e1-59c8-4a4c-affb-92d8a2314a79 | SSD_pool    | True      |
| 44f44076-fbe5-4d8d-a5c7-c8625bf0209b | HDD_pool    | True      |
| f0343753-da12-4a8c-b2ff-0db5d1aee19a | __DEFAULT__ | True      |
+--------------------------------------+-------------+-----------+
kysel@kysel-dp-maas:~$ openstack volume type show SSD_pool
+--------------------+-----------------------------------------+
| Field              | Value                                   |
+--------------------+-----------------------------------------+
| access_project_ids | None                                    |
| description        | None                                    |
| id                 | 0c29c9e1-59c8-4a4c-affb-92d8a2314a79    |
| is_public          | True                                    |
| name               | SSD_pool                                |
| properties         | volume_backend_name='cinder-ceph-ssd'   |
| qos_specs_id       | None                                    |
+--------------------+-----------------------------------------+
kysel@kysel-dp-maas:~$
```

Obrázok 37 Overenie volume types pomocou CLI

6.3.4 Overenie funkčnosti

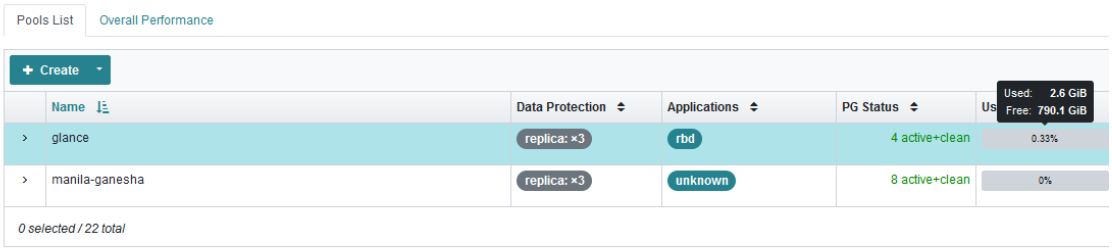
Na overenie použijeme jednoduchý postup. Do OpenStack-u nahráme ľubovoľný cloudový obraz linuxovej distribúcie, čím overíme činnosť modulu Glance. Následne vytvoríme volume v každom pool-e tak, že prekopírujeme tento obraz na daný volume.

V našom prípade sme zvolili cloudový obraz operačného systému Ubuntu vo formáte QCOW2. Pre nahratie obrazu sme použili dashboard OpenStack-u. Počas tohoto procesu sme sa prihlásili do dashboard-u Ceph-u a sústredili sme sa na pool „*glance*“.



Obrázok 38 Aktivita diskov pri nahrávaní obrazu

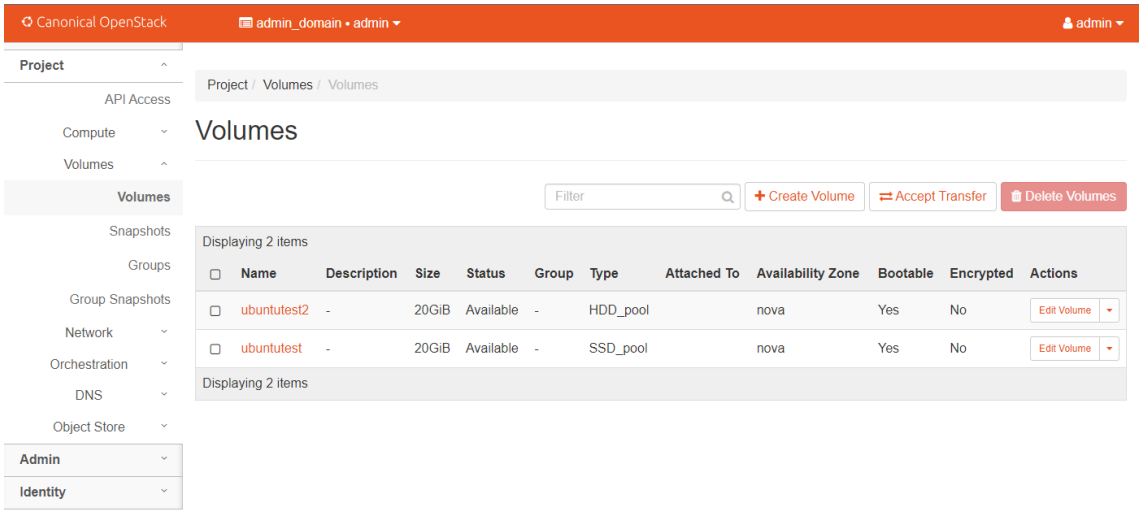
Na Obrázku 38 vidíme, že na disku prebiehala aktivita počas procesu nahrávania obrazu. Na Obrázku 39 môžeme vidieť využitie pool-u „glance“ po nahratí obrazu. Využitie pool-u však zobrazuje celkovo využitý priestor na diskoch, vrátane vytvorených replík. Ak využívame faktor replikácie 3, kapacitu pool-u je nutné vydeliť tromi. Pre replikačný faktor 5 ju vydelíme piatimi a pod. .



Obrázok 39 Využitie pool-u glance

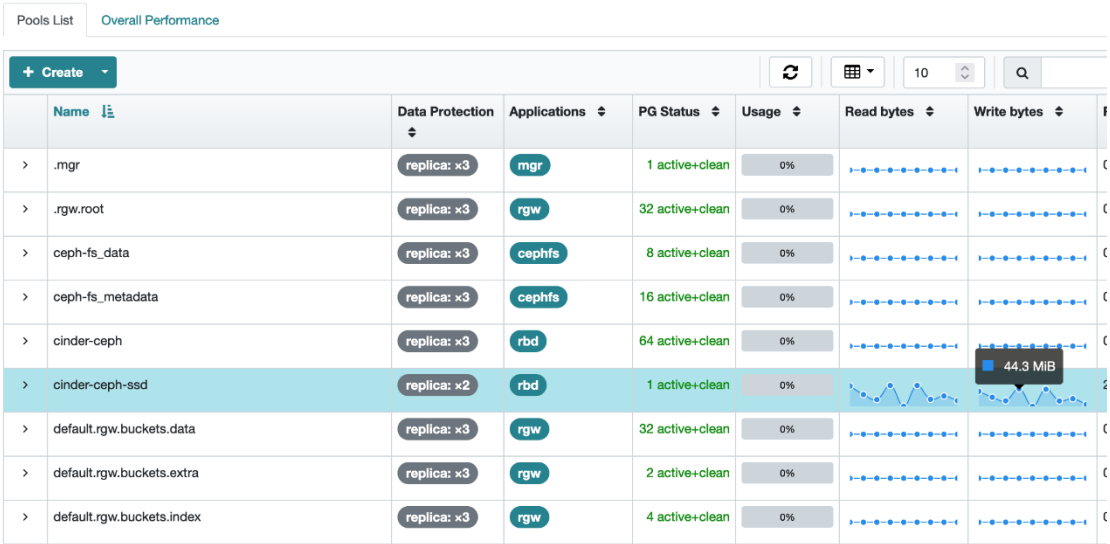
Pre overenie backend-u sme vytvorili volume pre každý volume type pomocou dashboardu OpenStack-u. Pri vytváraní sme použili pre „Volume Service“ možnosť „Image“ a ako obraz sme zvolili nahratý cloudový obraz Ubuntu. Nastavili sme požadovaný „Type“ na SSD_pool alebo HDD_pool volume

type. Rovnako ako v predošlom prípade sme tento proces monitorovali v dashboard-e Ceph-u.



Obrázok 40 Vytvorené volumes v dashboard-e OpenStack-u

Takisto ako v prípade pool-u „glance“ aj počas testovania týchto dvoch pool-ov sme pozorovali aktivitu na diskoch pri vytváraní jednotlivých volumes (Obrázok 41).



Obrázok 41 Aktivita na diskoch pri vytváraní volume

Po dokončení vytvárania volumes, na Obrázku 42 môžeme vidieť, že využitie disku sa zvýšilo v oboch prípadoch. To znamená že multi-backend je nastavený správne a jednotlivé volume types sú taktiež správne nastavené. Takto nakonfigurované pool-y je možné naplno využívať v OpenStack-u ako úložisko pre inštancie.

Obrázok 42 Overenie zápisu volumes do príslušných pool-ov

6.4 Manuálne stiahnutie obrazu z pool-u

Správa virtualizovaných prostredí OpenStack-u s úložiskom Ceph ako backend môže vyžadovať export volume-u z rôznych dôvodov, napr. z dôvodu zálohovania, migrácie alebo archivácie. Aktuálne nie je dostupný žiadny nástroj, ktorý by umožňoval automatizovaný export. Táto podkapitola opisuje proces exportovania prostredníctvom CLI rozhrania OpenStack-u alebo priamo z klastra Ceph prostredníctvom Ceph klienta. Tieto procesy potom môžu byť použité v na mieru vytvorených skriptoch, ktoré následne môžu slúžiť pre chýbajúcu automatizáciu.

6.4.1 Exportovanie s použitím OpenStack-u

Proces s využitím OpenStack-u začneme zistením ID volume-u, ktorý chceme exportovať. Následne vytvoríme snapshot, získame ID snapshot-u a z daného snapshot-u vytvoríme nový volume. Získame ID nového volume-u a vytvoríme z neho obraz. Tento obraz následne uložíme ako súbor, ktorý môžeme stiahnuť.

```
openstack volume list
openstack volume snapshot create --volume <id_volume-u> <nazov_snapshot-u>
openstack volume snapshot list
openstack volume create --snapshot <id_snapshot-u> --size <velkost> \ <nazov_noveho_volume-u>
openstack volume list
openstack image create --volume <nazov_noveho_volume-u> <nazov_obrazu>
```

```
openstack image save --file <nazov_suboru> <nazov_obrazu>
```

S vytvoreným súborom môžeme narábať ako s klasickým súborom, takže ho môžeme preniesť napr. pomocou nástroja scp [54].

6.4.2 Exportovanie s použitím Ceph klienta

Exportovať volume môžeme aj priamo z klastra Ceph-u. V tomto prípade je však dôležité, aby sme mali administratívny prístup ku klastru, keďže pre využitie tohoto spôsobu potrebujeme nakonfigurovať Ceph klienta.

Ako prvé nainštalujeme balíček „*ceph-common*“. Následne z kontajnera monitora prenesieme kľúče klienta a konfiguračný súbor na klientské zariadenie. Overíme funkčnosť klienta a zobrazíme si všetky volumes v poole. Následne tento volume exportujeme do zariadenia, na ktorom je nainštalovaný klient Ceph-u.

```
sudo apt update
sudo apt install ceph-common
```

Po nainštalovaní balíčka sa pripojíme do unit-u monitora a prekopírujeme klientský kľúč a konfiguračný súbor ceph-u na klienta.

```
sudo scp /etc/ceph/ceph.conf <username_klienta>@<IP_klienta>:/etc/ceph/
sudo scp /etc/ceph/ceph.client.admin.keyring \ <username_klienta>@<IP_klienta>:/etc/ceph/
```

Na klientskom zariadení zadáme nasledujúce príkazy:

```
ceph -s #overenie prístupu ku klastru
rbd ls <nazov_pool-u>
rbd export <nazov_poolu>/volume-<volume_id> <nazov_suboru_na_klientovi>
```

Veľkým rozdielom medzi exportovaním cez OpenStack a s exportovaním priamo z klastra je, že v prípade exportu priamo z Ceph-u dostávame ako výsledný súbor tzv. „raw“ blokový obraz. V prípade exportu z OpenStack-u obraz dostávame ako súbor v určitom formáte, napr. QCOW2. Ďalším rozdielom je, že proces exportovania cez OpenStack je „pracnejší“, ale je aj bezpečnejší, vďaka vytvoreniu snapshot-u a následne s prácou s ním. V prípade exportu priamo z klastra môže nastať situácia, že už prenesený sektor vo vnútri blokového obrazu sa zmení a novo prenesené sektory počítajú s touto zmenou, kdežto obraz obsahuje neaktuálne dáta. Hoci táto situácia nie je častá, v ojedinelých prípadoch môže dôjsť k znehodnoteniu volume-u.

7. ODPORÚČANIA PRE NASADENIE KLASTRA CEPH

V tejto kapitole sa budeme venovať vytvoreniu odporúčaní pre nasadenie klastra Ceph integrovaného do platformy OpenStack. Odporúčania sú založené na oficiálnych odporúčaniach z oficiálnej dokumentácie Ceph-u a na základe vedomostí získaných počas testovania a práce na vypracovaní tejto práce.

7.1 Koncept nasadenia

Odporúčania budú zamerané na konfiguráciu Ceph-u, resp. manažovanie klastra ako celku, aby spĺňali tri dôležité podmienky. Prvou podmienkou je efektívne využitie dostupných zdrojov klastra. Druhou podmienkou je zabezpečenie dostatočnej redundancie dát v prípade výpadku node-u alebo disku. Poslednou podmienkou je návrhnúť odporúčania tak, aby zahŕňali rozširovanie cloud-u, a tým aj rozširovanie klastra Ceph-u, či už vo forme pridávania node-ov alebo diskov do už existujúcich node-ov.

7.2 Prostredie nasadenia

Prostredie, v ktorom bude cloud nasadzovaný, bude tvoriť momentálne plánovaných 6 node-ov. Z hľadiska počtu a kapacity diskov sú tieto node-y rovnaké. Každý node bude disponovať dvomi SSD diskami o veľkosti 1,92 TB a piatimi HDD diskami. Spomedzi týchto piatich diskov je jeden určený ako systémový disk s veľkosťou 300 GB a ostatné štyri HDD disky ako dátové disky pre Ceph, každý o veľkosti 1,2 TB.

7.3 Konfigurácia Ceph-u

Konfiguráciu Ceph-u pri integrácií s OpenStack-om rozdelíme na dve časti. V prvej časti sa venujeme konfigurácií v rámci bundle súboru. Druhá časť je venovaná konfigurácií klastra po nasadení, vrátane logického členenia klastra.

7.3.1 Konfigurácia bundle súboru

Pre konfiguráciu v bundle súbore odporúčame vykonať len mierne zmeny. Na základe skúseností nadobudnutých pri testovaní, odporúčam nasadiť Ceph-Osd na všetky node-y, ale inicializovať iba HDD disky, ktoré sú k dispozícií.

Dôvodom je prehľadnosť v rámci OSD diskov. Keďže OSD diskom sú pridelené ID od nuly v poradí akom sa inicializovali, môžeme si tým uľahčiť následnú konfiguráciu.

Odporúčame použitie „*replicated*“ pool-ov. Dôvodom je, že „*erasure-coded*“ pool-y neposkytujú dostatočnú úroveň redundancie. Výhody týchto pool-ov, ktorými sú využívanie takmer plnej kapacity klastra a rýchlosť obsluhy, v cloudovom prostredí nevyvážia nedostatok redundancie, ktorá je jednou z hlavných podmienok.

Napriek tomu, že viacero konfiguračných parametrov má nastavené vhodné prednastavené hodnoty, odporúčame ako best practice nasledovné parametre uviesť ručne. Pri zmenách verzií môže dôjsť k zmenám prednastavených hodnôt, a to následne spôsobiť nepredvídateľné správanie. Tieto parametre sú uvedené v Tabuľke 4 nižšie.

V aplikácii Ceph-Osd odporúčame nastaviť jeden dôležitý parameter. Parameter **ignore-device-errors:** predchádza nefunkčnosti unit-u, v prípade zlyhania inicializovania jedného disku. Nakoľko v unit-e sa bude nachádzať viac ako jedno OSD, na základe skúseností z testovania odporúčame túto možnosť zapnúť.

Pre každú aplikáciu odporúčame nastaviť parameter **use-syslog** na „*true*“. Napriek tomu, že to môže signifikantne zvýšiť počet logov v systéme, tak logy nemusíme hľadať v rôznych súboroch, ale sú na jednom mieste – v syslogu. V prípade vhodnej agregácie logov to vedie k ľahšej správe a odhaľovaniu porúch alebo chýb v klastri.

Aplikácia	Parameter	Hodnota
Ceph-Mon	expected-osd-count: monitor-count: pg-autotune:	11 5 true
Ceph-Osd	ignore-device-errors:	true
Ceph Fs	ceph-osd-replication-count: pool-type:	3 replicated
Ceph Radosgw	ceph-osd-replication-count:	3

	pool-type: port:	replicated 80
Cinder Ceph	ceph-osd-replication-count: pool-type: volume-backend-name:	3 Replicated cinder-ceph-hdd
Glance	ceph-osd-replication-count: pool-type:	3 replicated

Tabuľka 4 Odporúčané parametre a ich hodnoty pre bundle súbor

7.3.2 Konfigurácia nasadeného klastra

Po nasadení klastra bundle súborom odporúčame logicky vyčleniť jednotlivé OSD do troch až piatich bucket-ov, nakoľko predvolene využívajú všetky pool-y všetky OSD démony. Jeden bucket budú tvoriť len SSD disky, ktoré budú určené pre SSD pool pre Cinder. Druhý bucket bude využívať HDD disky pre HDD pool ako druhý back-end pre Cinder. Podľa uváženia môžeme následne postupovať dvomi spôsobmi:

- Vytvoriť tretí bucket, ktorý bude obsahovať vyčlenené OSD disky, a tento bucket aplikujeme na ostatné vytvorené pool-y, ktoré vytvorilo Juju vrátane pool-ov pre moduly Glance a Manila.
- Vyhradiť disky a vytvoriť navyše bucket-y pre moduly Glance a Manila a ako v prvom prípade vytvoriť všeobecný bucket pre ostatné pool-y.

Obidva prístupy majú svoje výhody a nevýhody. Vytvorenie len troch bucket-ov je v prípade zlyhania OSD disku vo „všeobecnom“ bucket-e ovplyvnených viac služieb naraz. V prípade piatich bucketov sa toto riziko výrazne znižuje, čo je výhodou tohto prístupu. Nevýhodou sú ale zvýšené nároky na disky, a tým aj kapacitu jednotlivých pool-ov, čo v prvom postupe problém nie je.

Po uvážení viacerých faktorov odporúčame pre nasadenie na 6 node-ov prvý prístup s vyhradením jedného OSD disku na každom node-e pre spoločný bucket. V budúcnosti pri rozšírení o ďalšie compute node-y alebo pri dostatočnom navýšení počtu diskov odporúčame postupovať vytvorením piatich bucket-ov.

ZÁVER

Cieľom tejto práce bola analýza dostupných možností a výber vhodného softvéru pre nasadenie distribuovaného úložiska pre cloud postavený na platforme OpenStack. Na začiatku boli preskúmané bezplatné riešenia softvérov poskytujúcich služby nasadenia distribuovaného úložiska. Na základe vopred stanovených kritérií bol vybraný softvér Ceph.

Pre lepšie pochopenie jeho fungovania bola analyzovaná architektúra systému Ceph a jeho hlavné komponenty. Vysvetlené boli aj základné koncepty ako pool-y, PGs či algoritmy CRUSH a Paxos, ktorých pochopenie je zásadné pre správne nasadenie a konfiguráciu klastra Ceph.

V praktickej časti sa realizovalo manuálne nasadenie klastra, čo umožnilo detailne porozumieť činnosti jednotlivých komponentov počas reálneho nasadenia a ich konfigurácii. Následne sa pristúpilo k inštalácii a integrácii klastra Ceph s platformou OpenStack pomocou nástrojov OpenStack Charms, ktoré zjednodušujú proces nasadenia do cloudového prostredia. Počas tohto postupu boli detailne preskúmané a zdokumentované konfiguračné možnosti pri použití tzv. bundle súboru.

Po overení funkčnosti klastra sa pozornosť sústredila na konfiguráciu základných komponentov. Z dôvodu technických problémov s virtuálnym aj fyzickým prostredím nebolo možné implementovať cache-ovanie dát. V prílohách je však priložená dokumentácia konfigurácie cache-ovania, ktorá zatiaľ nebola otestovaná, a preto nie je súčasťou hlavného postupu.

Na záver boli vypracované odporúčania pre nasadenie softvéru Ceph v prostredí katedrového cloudu, vychádzajúc z poznatkov získaných počas testovania manuálneho nasadenia, použitia charm-ov a následnej konfigurácie. Cieľom bolo vytvoriť spoľahlivé distribuované úložisko pre cloudovú platformu OpenStack, ktoré spĺňa všetky zadané podmienky.

Monitorovanie OpenStack-u a klastra Ceph pomocou softvéru Prometheus nebolo realizované, keďže presahovalo rozsah projektu. Výsledky práce však môžu slúžiť ako podklad pre ďalšiu optimalizáciu a zavedenie monitorovania s využitím cache-ovania a softvéru Prometheus.

ZOZNAM POUŽITEJ LITERATÚRY

1. BMC SOFTWARE, INC. Distributed Storage Model - Concept and Principles - Documentation for BMC Discovery content reference - BMC Documentation [online]. [Dátum 29.3.2024]. Dostupné na internete: https://docs.bmc.com/docs/discovery/contentref/distributed-storage-model-concept-and-principles-997880678.html
2. NUTANIX. What is Distributed Storage? Explore Distributed Cloud | Nutanix [online]. [Dátum 29.3.2024]. Dostupné na internete: https://www.nutanix.com/info/distributed-storage
3. TELNYX. What is distributed storage, and why does it matter? [online]. [Dátum 29.3.2024]. Dostupné na internete: https://telnyx.com/resources/what-is-distributed-storage
4. WEKA.IO. Distributed File Systems Explained | Features and Advantages [online]. [Dátum 29.3.2024]. Dostupné na internete: https://www.weka.io/learn/distributed-file-systems/distributed-file-system/
5. HOSTKEY. What is Distributed Storage? Types and Examples [online]. [Dátum 21.3.2024]. Dostupné na internete: https://hostkey.com/blog/53-what-is-distributed-storage/
6. CLOUDIAN, INC. Distributed Storage: What's Inside Amazon S3? [online]. [Dátum 21.3.2024]. Dostupné na internete: https://cloudian.com/guides/data-backup/distributed-storage/
7. JULIO'S BLOG. Selecting the right storage backend for your OpenStack cloud [online]. [Dátum 9.10.2023]. Dostupné na internete: https://www.juliosblog.com/selecting-the-right-storage-backend-for-your-openstack-cloud/
8. STORAGE SWISS. 2015. Selecting the right OpenStack storage module [online]. [Dátum 9.10.2023]. Dostupné na internete: https://storageswiss.com/2015/10/14/selecting-the-right-openstack-storage-module/
9. CEPH DOCUMENTATION. Reef [online]. [Dátum 9.10.2023]. Dostupné na internete: https://docs.ceph.com/en/reef/
10. OPENSTACK. 2023. OpenStack Administration Guide [online]. [Dátum 9.10.2023]. Dostupné na internete: https://docs.openstack.org/2023.2/admin/
11. GLUSTER DOCS. Latest documentation [online]. [Dátum 19.10.2023]. Dostupné na internete: https://docs.gluster.org/en/latest/
12. RED HAT. 3.3. Red Hat Gluster Storage 3.3 Administration Guide [online]. [Dátum 19.10.2023]. Dostupné na internete: https://access.redhat.com/documentation/en-us/red_hat_gluster_storage/3.3/html-single/administration_guide/index#sect-Managing_Block_Storage
13. MINIO. MinIO [online]. [Dátum 19.10.2023]. Dostupné na internete: https://min.io/
14. CEPH. Architecture — Ceph Documentation [online]. [Dátum 15.2.2024]. Dostupné na internete: https://docs.ceph.com/en/reef/architecture/
15. CEPH FOUNDATION. Ceph.io — About the foundation [online]. [Dátum 21.3.2024]. Dostupné na internete: https://ceph.io/en/foundation/about/
16. CEPH. Red Hat's Ceph team is moving to IBM [online]. [Dátum 21.3.2024]. Dostupné na internete: https://ceph.io/en/news/blog/2022/red-hats-ceph-team-is-moving-to-ibm/
17. CEPH AUTHORS AND CONTRIBUTORS. Ceph Releases (general) — Ceph Documentation [online]. [Dátum 21.3.2024]. Dostupné na internete: https://docs.ceph.com/en/latest/releases/general/
18. HG INSIGHTS. Companies Using Ceph, Market Share, Customers and Competitors [online]. [Dátum 11.3.2024]. Dostupné na internete: https://discovery.hgdata.com/product/ceph
19. Canonical Ltd. What is Ceph? | Ubuntu [online]. [Dátum 18.2.2024]. Dostupné na internete: https://ubuntu.com/ceph/what-is-ceph

20. CEPH AUTHORS AND CONTRIBUTORS. Intro to Ceph — Ceph Documentation [online]. [Dátum 15.2.2024]. Dostupné na internete: https://docs.ceph.com/en/latest/start/intro/
21. CEPH. Monitor Config Reference — Ceph Documentation [online]. [Dátum 15.2.2024]. Dostupné na internete: https://docs.ceph.com/en/reef/rados/configuration/mon-config-ref/
22. CEPH. Ceph Manager Daemon — Ceph Documentation [online]. [Dátum 15.2.2024]. Dostupné na internete: https://docs.ceph.com/en/reef/mgr/
23. CEPH. OSD Service — Ceph Documentation [online]. [Dátum 21.2.2024]. Dostupné na internete: https://docs.ceph.com/en/reef/cephadm/services/osd/
24. CICIMOV, Igor. Ceph cluster on Ubuntu-14.04 - DevOps [online]. [Dátum 21.1.2024]. Dostupné na internete: https://icicimov.github.io/blog/storage/Ceph-cluster-on-Ubuntu-14.04/
25. CEPH. ceph-mds -- ceph metadata server daemon — Ceph Documentation [online]. [Dátum 16.2.2024]. Dostupné na internete: https://docs.ceph.com/en/reef/man/8/ceph-mds/
26. CEPH. MDS Journaling — Ceph Documentation [online]. [Dátum 21.2.2024]. Dostupné na internete: https://docs.ceph.com/en/reef/cephfs/mds-journaling/
27. WEIL, Sage, LEUNG, Andrew, BRANDT, Scott, MALTZAHN, Carlos, 2007. RADOS: A scalable, reliable storage service for petabyte-scale storage clusters. In: Proceedings of the 2nd International Workshop on Petascale Data Storage. pp. 35-44. [Dátum 22.2.2024]. DOI: 10.1145/1374596.1374606.
28. FLYDEAN. 04 Multi Paxos [online]. [Dátum 22.2.2024]. Dostupné na internete: https://docs.flydean.com/www.flydean.com/distribution/04-multi-paxos
29. CEPH AUTHORS AND CONTRIBUTORS. Ceph Storage Cluster — Ceph Documentation [online]. [Dátum 22.2.2024]. Dostupné na internete: https://docs.ceph.com/en/reef/rados/
30. CHUM, Sopanhapich, PARK, Heekwon and CHOI, Jongmoo, 2021. Supporting SLA via Adaptive Mapping and Heterogeneous Storage Devices in Ceph. Electronics. Vol. 10, p. 847. [Dátum 22.2.2024]. DOI: 10.3390/electronics10070847.
31. CEPH AUTHORS AND CONTRIBUTORS. Introduction to librados — Ceph Documentation [online]. [Dátum 23.2.2024]. Dostupné na internete: https://docs.ceph.com/en/latest/rados/api/librados-intro/
32. CEPH AUTHORS AND CONTRIBUTORS. CRUSH Maps — Ceph Documentation [online]. [Dátum 19.2.2024]. Dostupné na internete: https://docs.ceph.com/en/latest/rados/operations/crush-map/
33. CEPH AUTHORS AND CONTRIBUTORS. Pools — Ceph Documentation [online]. [Dátum 13.3.2024]. Dostupné na internete: https://docs.ceph.com/en/latest/rados/operations/pools/
34. CEPH AUTHORS AND CONTRIBUTORS. Placement Groups — Ceph Documentation [online]. [Dátum 14.3.2024]. Dostupné na internete: https://docs.ceph.com/en/reef/rados/operations/placement-groups/
35. D'ATRI, Anthony, Vaibhav BHAMBRE, and Karan SINGH. Learning Ceph - Second Edition. Birmingham: Packt Publishing, 2017. ISBN 978-1787127913.
36. CEPH AUTHORS AND CONTRIBUTORS. PG (Placement Group) notes — Ceph Documentation [online]. [Dátum 14.3.2024]. Dostupné na internete: https://docs.ceph.com/en/latest/dev/placement-group/
37. CEPH DOCUMENTATION. Install [online]. [Dátum 14.11.2023]. Dostupné na internete: https://docs.ceph.com/en/reef/install/
38. CEPH DOCUMENTATION. Hardware recommendations [online]. [Dátum 14.11.2023]. Dostupné na internete: https://docs.ceph.com/en/reef/start/hardware-recommendations/
39. CEPH DOCUMENTATION. Get packages [online]. [Dátum 20.11.2023]. Dostupné na internete: https://docs.ceph.com/en/reef/install/get-packages/
40. CEPH DOCUMENTATION. Install storage cluster [online]. [Dátum 14.11.2023]. Dostupné na internete: https://docs.ceph.com/en/reef/install/install-storage-cluster/

41. CEPH DOCUMENTATION. Manual deployment [online]. [Dátum 26.11.2023]. Dostupné na internete: https://docs.ceph.com/en/reef/install/manual-deployment/

42. CEPH DOCUMENTATION. Mgr administrator guide [online]. [Dátum 8.12.2023]. Dostupné na internete: https://docs.ceph.com/en/reef/mgr/administrator/#mgr-administrator-guide

43. GITHUB. 2018. Issue comment on ceph-container [online]. [Dátum 10.1.2024]. Dostupné na internete: https://github.com/ceph/ceph-container/issues/1158#issuecomment-412949409

44. OPENSTACK AUTHORS. OpenStack Charm Guide — charm-guide 0.0.1.dev804 documentation [online]. [Dátum 31.2.2024]. Dostupné na internete: https://docs.openstack.org/charm-guide/latest/

45. OPENSTACK AUTHORS. OpenStack Charms Deployment Guide — charm-deployment-guide 0.0.1.dev519 documentation [online]. [Dátum 31.2.2024]. Dostupné na internete: https://docs.openstack.org/project-deploy-guide/charm-deployment-guide/latest/

46. Canonical MAAS. MAAS installation technical reference (snap/2.8/CLI) - Docs - Canonical MAAS | Discourse [online]. [Dátum 3.3.2024]. Dostupné na internete: https://discourse.maas.io/t/maas-installation-technical-reference-snap-2-8-cli/4523

47. REŠKO, Dominik, 2023. Návrh katalógu cloudových služieb pre akademické prostredie. Žilinská univerzita v Žiline.

48. CHARMHUB. Charmhub | The Open Operator Collection [online]. [Dátum 21.03.2024]. Dostupné na internete: https://charmhub.io/

49. CEPH AUTHORS AND CONTRIBUTORS. Ceph iSCSI Gateway — Ceph Documentation [online]. [Dátum 21.3.2024]. Dostupné na internete: https://docs.ceph.com/en/latest/rbd/iscsi-overview/

50. CEPH AUTHORS AND CONTRIBUTORS. PG Calc — Ceph Documentation [online]. [Dátum 25.3.2024]. Dostupné na internete: https://docs.ceph.com/en/latest/rados/operations/pgcalc/

51. NFS-GANESHA PROJECT. nfs-ganesha.github.io [online]. [Dátum 26.3.2024]. Dostupné na internete: https://nfs-ganesha.github.io/

52. Canonical Ltd. How-to guides | Ubuntu [online]. [Dátum 26.3.2024]. Dostupné na internete: https://ubuntu.com/ceph/docs/howtos

53. OPENSTACK AUTHORS. Configure multiple-storage back ends — cinder 24.1.0.dev10 documentation [online]. [Dátum 27.3.2024]. Dostupné na internete: https://docs.openstack.org/cinder/latest/admin/multi-backend.html

54. VINCHIN. How to Export and Import Image or Instance in OpenStack? | Vinchin Backup [online]. [Dátum 11.4.2024]. Dostupné na internete: https://www.vinchin.com/vm-backup/openstack-export-import-image-instance-vm-snapshot-volume.html