



Palvelinvirtualisointi Openstack- alustalla

Timo Henriksson

Opinnäytetyö
Marraskuu 2014
Tietotekniikan koulutusohjelma
Tietoliikennetekniikka ja
tietoverkot

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tietotekniikan koulutusohjelma
Tietoliikennetekniikka ja tietoverkot

TIMO HENRIKSSON:
Palvelinvirtualisointi Openstack-alustalla

Opinnäytetyö 65 sivua, joista liitteitä 16 sivua
Marraskuu 2014

Tämä opinnäytetyö kertoo Openstack-alustan käytöstä palvelinvirtualisoinnissa. Työssä esitellään virtualisoinnin ja pilvipalveluiden teoriaa sekä käsitellään tarkemmin Openstackin rakennetta ja toimintaa. Käytännön osuus kertoo, kuinka Openstack-ympäristö asennetaan toimimaan yhdessä laitteessa. Tämä työ perustuu osittain Tampereen yliopiston yhteydessä toimivan Yhteiskuntatieteellisen tietoarkiston Openstack-pilven perustamisesta saatuihin kokemuksiin.

Openstack on Linux-järjestelmissä toimiva ohjelmistokokonaisuus, jota voidaan käyttää muun muassa virtualisointiin ja pilvipalveluiden perustamiseen. Openstack ei ole työpöytäsovellus, vaan se asennetaan palvelimeen. Openstack on avointa lähdekoodia, ja se on julkaistu Apache 2.0 -lisenssillä. Tässä työssä käsitelty Openstackin versio on huhtikuussa 2014 julkaistu Icehouse, jonka asennukset on tehty Ubuntu-käyttöjärjestelmään.

Tämän opinnäytetyön ohjeilla asennettava yhden laitteen Openstack-asennus sopii hyvin tilanteeseen, jossa halutaan nopeasti testata Openstackin toimintaa. Lisäksi yhden laitteen Openstack-ympäristö on sopiva todelliseen käyttöön tilanteissa, joissa monilaitteinen palvelinympäristö on tarpeeton. Esimerkiksi Yhteiskuntatieteellinen tietoarkisto siirsi soveluskehitystään tämän opinnäytetyön ohjeen mukaiseen yhden laitteen Openstack-pilveen.

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in ICT Engineering
Telecommunication and Networks

TIMO HENRIKSSON
Server Virtualization Using Openstack Platform

Bachelor's thesis 65 pages, appendices 16 pages
November 2014

This bachelor's thesis shows how Openstack platform can be used for server virtualization. The first part of this work tells about theory of virtualization and cloud services. The second part tells is about structure and services of Openstack. The last part is practical and it contains guide telling how Openstack can be installed to work on a single server (all-in-one installation). This work is partially based on experiences got from building Openstack cloud for Finnish Social Science Data Archive which is a separate unit of the University of Tampere.

Openstack consists of several services that can be used to create cloud environments and virtualize computers. The services work on Linux systems. Openstack is not desktop application but it is installed on servers. Openstack is released as open source under Apache 2.0 licence. Version of Openstack discussed in this bachelor's thesis is Icehouse which was released in April 2014. Operating system used was Ubuntu.

Installation guide of this work is well suited for quickly testing Openstack. The single server installation can also be useful when power of multiple servers is not necessary. For example software development and testing of Finnish Social Science Data Archive was partially moved to a single device Openstack cloud.

Key words: openstack virtualization cloud linux servers

SISÄLLYS

1	JOHDANTO.....	7
2	VIRTUALISOINTI.....	9
	2.1 Virtualisoinnin tekniikka	9
	2.2 Virtualisoinnin edut ja ongelmat.....	11
3	PILVIPALVELUT	13
	3.1 Mikä on pilvipalvelu	13
	3.2 Pilvipalveluiden luokittelu	13
	3.2.1 Infrastruktuuri palveluna (IaaS)	14
	3.2.2 Sovellusalusta palveluna (PaaS)	15
	3.2.3 Sovellus palveluna (SaaS).....	16
	3.3 Pilvipalveluiden ongelmat	16
4	OPENSTACK	18
	4.1 Mikä on Openstack	18
	4.2 Openstackin rakenne	18
	4.3 Nova.....	21
	4.4 Keystone	22
	4.5 Neutron	24
	4.6 Glance	25
	4.7 Horizon	26
5	OPENSTACKIN ASENNUSOHJE.....	28
	5.1 Taustaa.....	28
	5.2 Taustapalveluiden asennus	29
	5.3 Keystonen asennus.....	32
	5.4 Glancen asennus	34
	5.5 Novan asennus	36
	5.6 Neutronin asennus.....	39
	5.7 Horizonin asennus.....	42
	5.8 Verkkojen ja virtuaalikoneiden luonti	43
6	POHDINTA.....	46
	LÄHTEET.....	47
	LIITTEET	50
	Liite 1. Openstakin palvelut ja niiden tehtävät.....	50
	Liite 2. Openstackin palveluiden yhteydet	51
	Liite 3. Keystonen toiminta	52
	Liite 4. MySQL:n konfiguraatio.....	53
	Liite 5. Keystonen konfiguraatio.....	55

Liite 6. Glancen konfiguraatiot.....	56
Liite 7. Novan konfiguraatio	59
Liite 8. Neutronin konfiguraatiot.....	61
Liite 9. Horizonin konfiguraatiot.....	64

ERITYISSANASTO

AMI	Amazon Machine Image, levykuva
API	Application programming interface, ohjelmointirajapinta
DHCP	Dynamic Host Configuration Protocol, ip-osoitteita jakava protokolla
Hypervisor	Virtualisointia ohjaava ohjelmisto
IaaS	Infrastructure as a Service, pilvipalvelumalli
Isäntä	Laite jossa ajetaan virtuaalikoneita
KVM	Kernel-based Virtual Machine, Linuxin kerneliin kuuluva hypervisor
NaaS	Network as a Service, pilvipalvelumalli
PaaS	Platform as a Service, pilvipalvelumalli
Puppet	Ohjelmisto automaattisiin asennuksiin ja konfiguraationhallintaan
SaaS	Software as a Service, pilvipalvelumalli
SSH	Secure Shell, protokolla tietoliikenteen salaukseen
SSL	Secure Sockets Layer, verkkoliikenteen salausprotokolla
SQL	Structured Query Language, kyselykieli tietokantakäyttöön
VDI	Virtualbox Disk Image, levykuva
VMM	Virtual Machine Monitor, toinen nimi hypervisorille
VNC	Virtual Network Computing, etätyöpöytäyhteys

1 JOHDANTO

Tämä opinnäytetyö kuvaa, kuinka Openstack-alustaa voidaan käyttää palvelinvirtualisointiin. Openstack on avoimen lähdekoodin alusta laitteistoriippumattomien yksityisten ja julkisten pilvipalveluiden perustamiseen ja ylläpitoon. Openstackin palvelut toimivat Linux-käyttöjärjestelmissä (Openstack-säätiö 2014c). Tämän työn kuvaama Openstack-pilven asennus oli osa työtäni Tampereen yliopistossa toimivassa Yhteiskuntatieteellisessä tietoarkeistossa.

Valitsin opinnäytetyöni aiheeksi Openstackin, koska pidin aiheesta kiinnostava ja pilvipalvelut sekä virtualisointi ovat nykyään erittäin merkittävä ja nopeasti kasvava osa it-alan liiketoimintaa. Esimerkiksi Vuoden 2014 ensimmäisellä neljänneksellä Euroopassa, Lähi-idässä ja Afrikassa hankituista palvelimista 32 % oli virtualisoituja. Virtualisoitujen palvelinten määrä kasvoi 11,5 % edellisestä neljänneksestä, vaikka fyysisen laitteiden myynti laski 4,7 %. (IDC 2014.) Lisäksi virtuaalisissa palvelimissa suoritettujen laskennan määrä ohitti fyysisissä laitteissa tapahtuvan laskennan jo vuonna 2012 (Hernandez 2013). Pilvipalveluiden ennustetaan olevan it-alan merkittävämpiä ilmiöitä tulevina vuosina (Columbus 2014).

Yhteiskuntatieteellisellä tietoarkeistolla oli useita syitä palvelinten laajamittaiselle virtualisoinnille. Laitemäärän kasvaessa palvelintila oli alkanut käydä ahtaaksi ja suuren laitemäärän ylläpito vaatii paljon resursseja. Arkiston toiminta on laajenemassa, joten myös palvelinmäärän odotetaan kasvavan. Fyysisiin palvelimiin perustuvassa järjestelmässä monien palvelinten käyttöaste oli matala. Virtualisoinnin avulla palvelinlaitteita voitaisiin hyödyntää tehokkaammin, jolloin fyysisiä palvelimia tarvittaisiin vähemmän. Samalla laitteiston käyttöaste kasvaisi, mikä vähentäisi tarvetta uuden laitteiston hankintaan ja toisi rahallisia säästöjä. Virtuaalisia palvelimia voitaisiin siirtää fyysisten laitteiden välillä esimerkiksi huoltojen ajaksi, mikä parantaisi palveluiden luotettavuutta ja helpottaisi ylläpitoa. Jos virtuaalikoneita voitaisiin siirtää helposti toisessa rakennuksessa sijaitseviin palvelintiloihin, saataisiin suojaa vakavia ongelmia, kuten vesivahinkoja tai tulipaloja vastaan.

Palvelinvirtualisointiin on tarjolla useita kaupallisia sekä avoimeen lähdekoodiin perustuvia ratkaisuja. Näistä tarkempaan selvitykseen valittiin Openstack, koska sen käyttö on

ilmaista ja sen ominaisuudet vaikuttivat lupaavilta. Lisäksi Openstackin kehitystyö on edennyt nopeasti ja sillä on aktiivinen käyttäjäyhteisö. Monet merkittävät it-alan yritykset, kuten HP, IBM, Intel ja Red Hat tukevat Openstackin kehitystyötä, joten projektin tulevaisuus vaikuttaa turvatulta (Openstack-säätiö 2014a).

Openstackin soveltuvuutta selvitettiin Tietoarkistossa kesän 2014 aikana. Dokumentaation tutkiminen ja testiasennukset osoittivat, että Openstack täyttää vaatimukset ja mahdollistaa toivotut asiat. Ensimmäinen vaihe Openstackin käyttöönotossa oli yhdessä palvelimessa toimivan Openstack-pilven luominen sovelluskehitystä ja testausta varten. Tämä työ esittää, miten vastaavan yhteen laitteeseen perustuvan Openstack-ympäristön voi asentaa.

Tämän opinnäytetyön alussa on teoreettinen osuus, jossa käsitellään virtualisointia ja pilvipalveluita. Siinä kerrotaan miten nykyiset pilvipalvelut perustuvat virtualisointiin. Sen jälkeen käsitellään pilvipalveluiden ominaisuuksia ja luokittelua. Sitten esitellään Openstack sekä kerrotaan sen toiminnasta ja siihen liittyvistä palveluista. Käytännön osuudessa kerrotaan, kuinka Openstackin palvelut asennetaan toimimaan yhdessä laitteessa.

Tässä työssä käsitelty Openstackin versio on huhtikuussa 2014 julkaistu Icehouse. Työssä esiteltävät asennukset on tehty Ubuntu 14.04 -käyttöjärjestelmää käyttäviin palvelimiin, mutta siinä esiteltyt asiat ovat sovellettavissa myös muihin Linux-käyttöjärjestelmiin. Liitteissä esitellään toimivan Openstack-ympäristön konfiguraatitiedostoja.

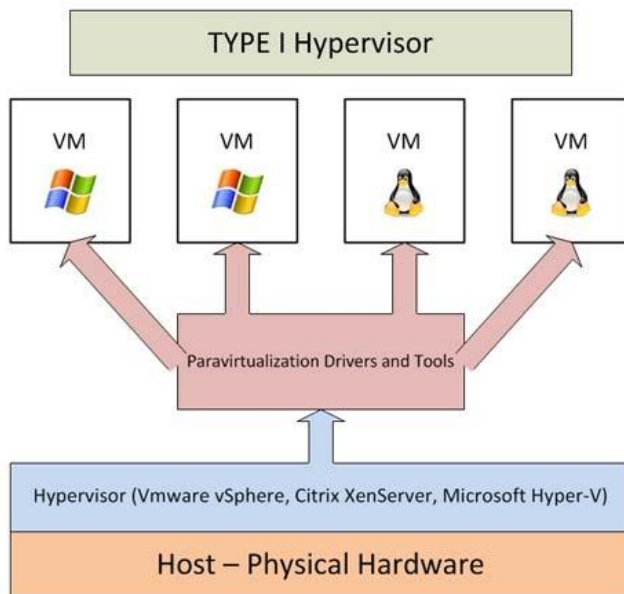
2 VIRTUALISOINTI

2.1 Virtualisoinnin tekniikka

Virtualisointi tarkoittaa, että järjestelmä on abstrakti niin, että sen loogisten osien määrä ei ole suoraan riippuvainen fyysisten osien määrästä. Esimerkiksi yhdessä fyysisessä palvelimessa voi olla useita virtuaalisia palvelimia. Näissä virtuaalisissa palvelimissa toimivat palvelut eivät välttämättä ole lainkaan tietoisia virtualisoinnista. Virtualisointi ei rajoitu vain palvelimiin ja tietokoneisiin, vaan muun muassa tietoverkot voivat olla virtuaalisia. (Portnoy 2012, 2.)

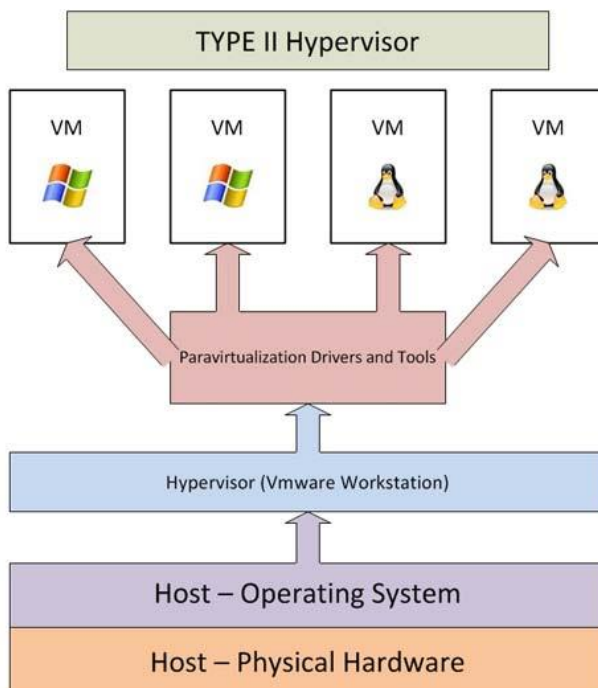
Tietokoneita ja palvelimia virtualisoivia ohjelmistoja kutsutaan yleisnimellä hypervisor tai virtual machine monitor (VMM). Hypervisor jakaa fyysisten laitteiden resursseja virtuaalikoneille. Riippuen siitä, miten virtualisointi toteutetaan, puhutaan tyyppien 1 ja 2 hypervisoreista. (Kleyman 2012.) Jako tyyppisiin 1 ja 2 ei ole ehdoton, vaan joissain hypervisoreissa yhdistyy molempien tyyppien ominaisuuksia. Tällainen on esimerkiksi avoimeen lähdekoodiin perustuva Kernel-based Virtual Machine eli KVM, joka on osa Linux-käyttöjärjestelmien ydintä eli kerneliä. (Day 2012.)

Tyypissä 1 hypervisor toimii suoraan laitteistotason päällä, ilman välissä olevaa käyttöjärjestelmää (kuvio 1). Koska hypervisorilla on suora pääsy laitteistoon, virtualisointi on nopeaa ja tehokasta. Virtuaalikoneille tarjottavat palvelut optimoidaan virtualisointikäyttöön sopiviksi eli ne paravirtualisoidaan. Tyypin 1 virtualisointia käyttävät muun muassa Citrixin XenServer ja Microsoftin Hyper-V. (Kleyman 2012.)



KUVIO 1. Tyypin 1 virtualisointi (Kleyman 2012)

Tyypin 2 hypervisor on ohjelma, joka toimii käyttöjärjestelmän päällä (kuvio 2). Tyypin 2 hypervisor ei yllä samaan tehokkuuteen kuin tyyppi 1, koska käyttöjärjestelmä on ylimääräinen askel virtuaalikoneiden ja laitteiston välillä. Nykyajan tehokkaalla laitteistolla ja parantuneella ohjelmistolla myös tyypillä 2 voidaan saavuttaa erittäin hyvä tehokkuus. Esimerkiksi Virtualbox ja Vmwaren Workstation käyttävät tyypin 2 hypervisoria. (Kleyman 2012.)



KUVIO 2. Tyypin 2 virtualisointi (Kleyman 2012)

2.2 Virtualisoinnin edut ja ongelmat

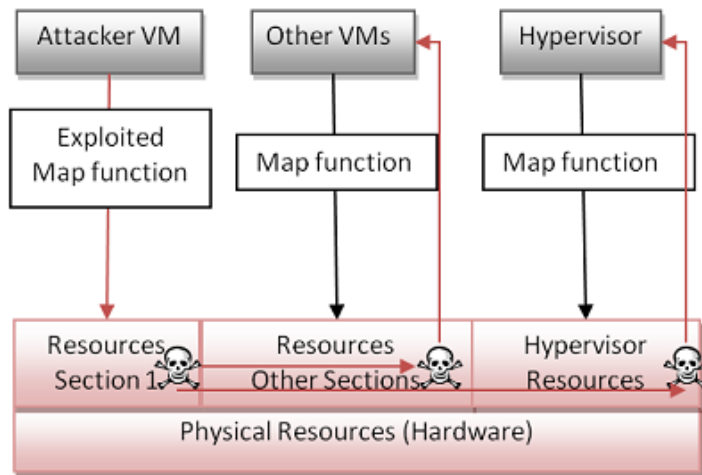
Aikana ennen virtualisointia yritysten oli yleensä hankittava erillinen palvelin jokaista palvelua varten. Palvelimet oli mitoitettava tarvetta suuremmiksi, jotta palvelut selviäsivät rasituspiikeistä ja käytön kasvusta. Kun palvelinten määrä kasvaa, kasvaa samalla hukkaan menevä ylimääräinen kapasiteetti. 10 % käyttöaste palvelinkeskuksessa ei ollut epätavallinen. Koska palvelinten rasitus jakautuu epätasaisesti, joiden laitteiden resurssit saattavat olla loppumassa kesken, kun taas toisissa käyttöaste on vain 5 % tai vielä vähemmän. Lisäksi suurten palvelinmäärien ylläpito on hankalaa ja kallista. (Portnoy 2012, 8–9.)

Virtualisointi tarjoaa ratkaisun näihin ongelmiin. Sen tuoman joustavuuden ansiosta voidaan nopeasti tarjota lisää resursseja niille palveluille, jotka niitä tarvitsevat. Samalla fyysisten laitteiden käyttöaste saadaan korkeammaksi. Kun laitteistoa tarvitaan vähemmän, myös kulut laskevat, sillä esimerkiksi sähköä ja lattiapinta-alaa kuluu vähemmän. Yksi fyysinen palvelin voi ylläpitää helposti kymmeniä virtuaalisia palvelimia ja joskus jopa yli sataa virtuaalista palvelinta. (Portnoy 2012, 10–11.)

Palvelinkeskusten vikasietoisuus paranee virtualisoinnin ansiosta. Kun palvelimet eivät ole enää sidottuja tiettyyn laitteeseen, palvelu on helppo siirtää toiseen laitteeseen huollon ajaksi niin, että palvelussa ei esiinny käyttäjille näkyvää katkosta. Tarpeen vaatiessa kokonaisen palvelinkeskuksen palvelut voidaan siirtää toiseen paikkaan minuuteissa tai tunneissa. Fyysisten palveluiden vastaava siirto veisi viikkoja. (Portnoy 2012, 12–13.) Virtualisointi johtaa myös suurempaan riippumattomuuteen käyttöjärjestelmistä. Yhden virtuaali-isännän virtuaalikoneiden käyttöjärjestelminä voi olla Windowsia, Linuxia ja muita Unixeja riippumatta siitä, mikä on isäntäkoneen käyttöjärjestelmä.

Virtualisointiin liittyy myös ongelmia, jotka on ratkaistava, jotta järjestelmästä saadaan turvallinen. Koska virtuaalikoneita on helppo luoda, niiden määrä kasvaa helposti. Ilman harkintaa luodut virtuaalikoneet voivat jäädä päivittämättä tai konfiguroimatta ja voivat muodostaa tietoturvariskin. Koska virtuaaliset koneet ja verkot eivät ole nähtävissä, käyttäjien voi olla vaikea ymmärtää virtualisoidun datakeskuksen toimintaa ja loogista rakennetta. Nämä ongelma voidaan ratkaista muuttamalla organisaation toimintatapoja muuttuneeseen ympäristöön sopiviksi. (Shackleford 2012, 4–5.)

Toiset ongelmat ovat teknisempiä. Jos hypervisorissa on turva-aukko, jonka avulla virtuaalikoneen kautta voi suorittaa koodia isäntäkoneessa, yhtä virtuaalikonetta hallitseva hyökkääjä voi onnistua kaappaamaan tai saastuttamaan isäntäkoneen lisäksi kaikki siinä ajettavat virtuaalikoneet. Hyökkäyksestä käytetään nimiä VM escape ja Hypervisor escape. Tämä tilanne on esitetty kuviossa 3. Lievemässä tapauksessa hyökkääjä saattaisi kyetä kaatamaan virtuaali-isännän, jolloin myös kaikki muut samassa laitteessa olevat palvelut kaatuisivat. (Shackleford 2012, 6–7.)



KUVIO 3. Hypervisorin turva-aukko altistaa järjestelmän hyökkäykselle (Zheng 2011)

Virtualisoidussa järjestelmässä on enemmän erilaisia osia verrattuna järjestelmään, jota ei ole virtualisoitu. Perinteisiin tasoihin laitteisto, käyttöjärjestelmä, verkko ja sovellukset tulee lisäksi hypervisor ja virtuaalikoneet. Koska järjestelmä on monimutkaisempi, siinä on enemmän osia, joissa saattaa olla turva-aukkoja, joiden havaitseminen on vaikeampaa kuin yksinkertaisemmassa järjestelmässä. (Shackleford 2012 10–12.)

3 PILVIPALVELUT

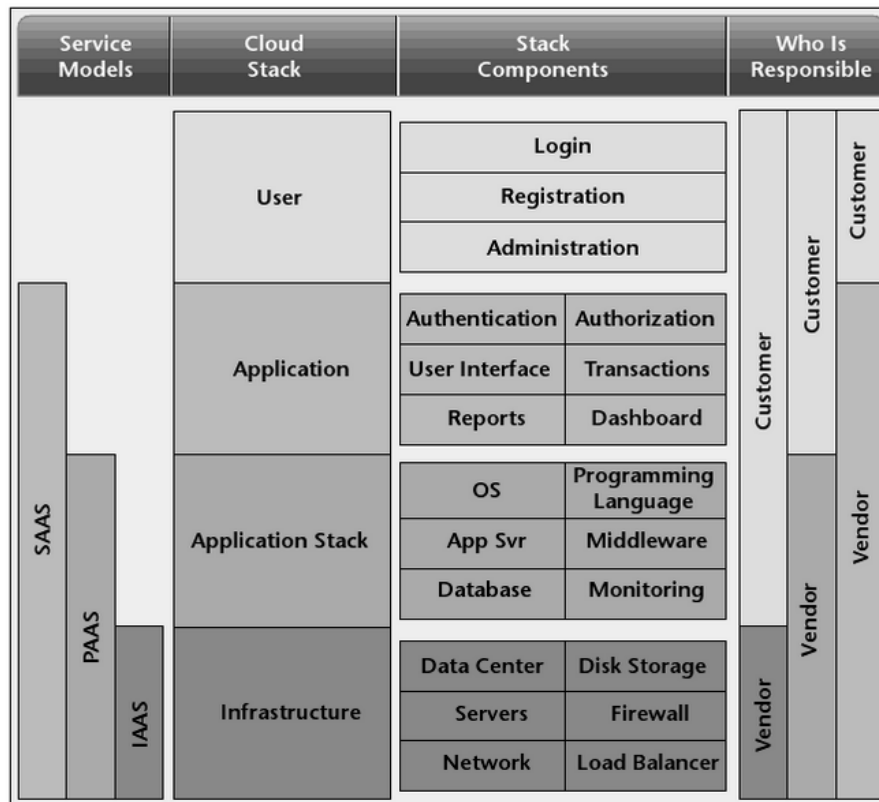
3.1 Mikä on pilvipalvelu

Pilvipalvelut ja pilvilaskenta ovat virtualisointia seuraava askel palvelinkeskusten kehityksessä (Portnoy 2012, 14). Pilvipalvelut eivät kuitenkaan liity pelkästään palvelimiin, vaan melkein mikä tahansa it-alan toiminto voidaan nykyään hankkia pilvipalveluna. Ei ole olemassa yhtä tarkkaa määrittelyä sille, mikä on pilvipalvelu, mutta pilvipalveluihin liitetään seuraavia ominaisuuksia (Mell & Grance 2011):

- Palvelusta maksetaan käytön mukaisesti ja hinnoittelu on automaattista.
- Palvelua käytetään verkkoyhteyden yli eikä sen fyysisellä sijainnilla ole merkitystä.
- Palveluntarjoajan resurssit jaetaan kysynnän mukaisesti useiden asiakkaiden kesken.
- Palvelu skaalautuu automaattisesti niin, että käyttäjän näkökulmasta palvelun resurssit ovat rajattomat.
- Palveluiden käyttöä mitataan ja tilastoidaan niin, että tiedot ovat myös asiakkaan käytettävissä.

3.2 Pilvipalveluiden luokittelu

Pilvipalvelut jaetaan kolmeen päätyyppiin. Näitä ovat sovellusten hankkiminen palveluna eli SaaS (Software as a Service), sovellusalustan hankkiminen palveluna eli PaaS (Platform as a Service) ja infrastruktuurin ostaminen palveluna eli IaaS (Infrastructure as a Service). (Kavis 2014, 13.) Joskus käytetään myös termiä Naas (Network as a Service) kuvaamaan verkkoyhteyden hankintaa skaalautuvana palveluna (Fagan 2013). Kuvio 4 esittää tarkemmin, miten palvelut ja vastuut jakautuvat eri malleissa asiakkaan ja palveluntarjoajan välille. Sama palvelu voi edustaa useaa eri tyyppiä, riippuen siitä, minkä käyttäjäryhmän näkökulmasta asiaa katsotaan. Esimerkiksi ylläpitäjille palvelu voi olla IaaS-mallin mukainen, kun taas loppukäyttäjät katsovat käyttävänsä SaaS-mallin pilvisovellusta.



KUVIO 4. Pilvipalveluiden tasot (Kavis 2014, 14)

Lisäksi pilvipalvelut voidaan luokitella sen mukaan, mikä on palvelun kohderyhmä. Yksityinen pilvipalvelu on tarkoitettu ainoastaan organisaation tai yrityksen omaan käyttöön. Palvelinten ei kuitenkaan tarvitse sijaita yhteisön omissa tiloissa, vaan ne voivat olla ulkopuolisen palveluntarjoajan konesalissa. Yhteisöllinen pilvipalvelu (community cloud) on kuin yksityinen pilvipalvelu, mutta sen käyttäjiä voivat olla omistajan lisäksi esimerkiksi yrityksen yhteistyökumppanit. Julkisella pilvipalvelulla on useita käyttäjiä, joilla ei ole mitään yhteyttä toisiinsa. Sen laitteisto sijaitsee palveluntarjoajan tiloissa. On myös mahdollista, että palvelussa yhdistyy useiden tyyppien ominaisuuksia, jolloin käytetään termiä hybrid cloud. (Mell & Grance 2011.)

3.2.1 Infrastruktuuri palveluna (IaaS)

Infrastruktuurin ostaminen palveluna on pilvipalveluiden ensimmäinen taso. Siinä asiakas ulkoistaa palvelinkeskuksen ylläpidon palveluntarjoajalle. Palveluntarjoajan vastuulle siirtyviä asioita ovat muun muassa palvelinlaitteet, laitteiston yhteydet internetiin, tallennustila ja laitteistopalomuurit. Asiakkaan vastuulle jäävät laitteiden käyttöjärjestelmät, sekä käyttöjärjestelmien päällä ajettavat palvelut ja palvelinten tilan tarkkailu. IaaS

toteutetaan usein yksityisenä pilvipalveluna. Amazon on kenties tunnetuin IaaS-palveluntarjoaja. (Kavis 2014, 14–15.)

Siirtymällä IaaS-mallin käyttöön ylläpidon ei tarvitse keskittyä rautatason ongelmien ratkomiseen eikä mekaanisiin ylläpitotehtäviin. Näin asiakas voi keskittää enemmän resursseja palveluiden ja sovellusten kehittämiseen. Ylläpitäjä hallitsee vuokrattuja palvelimia etäyhteyden kautta ja on edelleen vastuussa ohjelmistotason ylläpitotehtävistä, kuten päivityksistä ja varmuuskopioinnista. (Kavis 2014, 14–15.)

Mikäli palvelimet omistetaan itse, lisäkapasiteetin ostaminen voi viedä viikkoja tai jopa kuukausia ja kysynnän pienentyessä voi olla mahdotonta keksiä järkevää käyttöä ylimääräiselle laitteistolle. IaaS-mallin ansiosta palveluiden määrä voidaan skaalata sopivaksi minuuteissa ja skaalautuminen voidaan automatisoida. Lisäksi organisaatio voi vähentää kiinteitä kuluja, kuten sähkölaskua ja tilavuokria, kun palvelusta maksetaan käytön mukaan eikä omille laitteille tarvitse varata tilaa. (Kavis 2014, 14–15.)

3.2.2 Sovellusalusta palveluna (PaaS)

Seuraava pilvipalveluiden taso on koko ohjelmistoalustan hankkiminen palveluna. Siinä palvelinten lisäksi abstrakteiksi ja joustaviksi muuttuvat myös käyttöjärjestelmät ja tietokannat. Samalla asiakas siirtää entistä enemmän vastuuta palveluntarjoajalle, vähentäen merkittävistä perinteisistä ylläpidon tehtäviä. (Havainnollistettu aiemmin kuviossa 4.) PaaS tarjoaa sovelluskehitykseen valmiit ympäristöt ja kirjastot. Muiden muassa Google Apps Engine ja Microsoft Azure tarjoavat PaaS-pilveä. (Kavis 2014, 15–16.)

Sovellusalustan hankkiminen palveluna kaventaa palveluiden kehittäjien valinnanvaraa, sillä käytettävät työkalut ja ohjelmistopinot (software stack) on valittava palveluntarjoajan valikoimasta. Kehittäjillä on vain vähän mahdollisuuksia vaikuttaa alemman tason ohjelmistojen toimintaan, esimerkiksi muistinhallintaan, säikeistykseen tai välimuistin käyttöön. Käyttäjien vapaus valita työkalunsa ja vaikuttaa ympäristöön vaihtelee paljon eri palveluntarjoajien välillä. PaaS voidaan toteuttaa myös yksityisenä pilvipalveluna, jolloin palvelusta vastaa organisaation oma ylläpito. (Kavis 2014, 16–17.)

PaaS-palveluntarjoajat tarjoavat usein valmiita sovelluskomponentteja, jotka voidaan liittää organisaation omiin palveluihin tai tuotteisiin. Näitä ovat esimerkiksi hakutyökalut, analytiikka ja maksujärjestelmät (esimerkiksi verkkokaupan perustamista varten). Nämä työkalut voidaan nopeasti yhdistää rajapintojen avulla toimivaksi palveluksi. Siirtämällä sovellusalustat pilveen organisaatiot pysyvät näin keskittymään sovellustensa kehittämiseen ja käyttämään markkinoiden parhaita työkaluja. (Kavis 2014, 16–17.)

3.2.3 Sovellus palveluna (SaaS)

Sovellusten hankkiminen palveluna on pilvipalveluiden pisimmälle viety muoto. Siinä kaikki palveluiden käyttöön liittymätön on ulkoistettu palveluntarjoajan vastuulle ja asiakas saa käyttöönsä valmiin palvelun. (Esitetty aiemmin kuviossa 4.) Asiakkaan on ainoastaan konfiguroitava palvelun asetukset ja hallittava käyttäjätilejä. Asiakkaan ei tarvitse välittää siitä, millä tekniikalla palvelu on toteutettu. Tyypillinen esimerkki SaaS-pilvipalvelusta on verkkosähköposti, esimerkiksi Googlen Gmail. (Kavis 2014, 17–18.) Melkein mitä tahansa verkkoyhteyden yli käytettävää palvelua voidaan pitää SaaS-mallin mukaisena pilvipalveluna.

Koska asiakkailta on vain vähän vaikutusvaltaa SaaS-palveluiden toimintaan, niitä käytetään yleensä vain vähemmän kriittisten palveluiden toteuttamiseen. Tärkeimmät palvelut puolestaan halutaan pitää vahvemmin organisaation omassa hallinnassa. Palveluna hankitun ohjelmiston ylläpito vaatii vain hyvin vähän organisaation omaa työtä, sillä palvelun käyttötukikin saattaa sisältyä palvelun hintaan. (Kavis 2014, 18.)

3.3 Pilvipalveluiden ongelmat

Vaikka pilvipalveluissa on paljon hyviä puolia, ne tuovat mukanaan myös uusia ongelmia. Suurin muutos koetaan, kun siirrytään julkisiin pilvipalveluihin. Julkisessa pilvessä asiakas ei voi vaikuttaa palvelun luotettavuuteen muuten kuin sopimalla tietystä palvelutasosta ja luottamalla siihen, että palveluntarjoaja pystyy pitämään lupauksensa. Mikäli palvelu lakkaa toimimasta, asiakas ei voi kuin odottaa, että palveluntarjoaja saa järjestelmänsä jälleen toimimaan. (Kavis 2014, 19–20.) Tunnettu esimerkki tästä on Amazonin

pilvipalveluita vuoden 2011 huhtikuussa vaivannut ongelma, joka kaatoi suuren osan palvelua useiksi tunneiksi ja joidenkin asiakkaiden kohdalla jopa päiviksi. Suhteellisen yksinkertainen tiedon varmistukseen liittynyt ongelma levisi Amazonin järjestelmässä vaikuttaen tuhansiin asiakkaisiin. (CNR 2011.) Siksi voi olla hyvä idea hankkia palveluita ainakin kahdesta erillisesti palvelinkeskuksesta, tai jopa kahdelta eri palveluntarjoajalta.

Käytettäessä julkista pilveä asiakas joutuu myös luottamaan tietonsa palveluntarjoajalle. Kaiken tiedon salaaminen ei ole käytännöllistä, joten on voitava luottaa siihen, että tiedot pysyvät luottamuksellisina eivätkä esimerkiksi samoja laitteita vuokraavat muut asiakkaat pääse vahingossa näkemään tietoja. Lisäksi lait saattavat rajoittaa organisaation mahdollisuuksia tallentaa tietoja omien konesaliensa ulkopuolelle (Kavis 2014, 20). Esimerkiksi Suomessa henkilötietolaki rajoittaa organisaatioiden oikeutta viedä suomalaisten tietoja EU:n alueen ulkopuolella sijaitseviin palvelinkeskukseen (Kuntaliitto 2013).

Asiakkaalla ei myöskään ole suurta valinnanvaraa julkisen pilvipalvelun laitteiston suhteen, vaan on tyydyttävä tarjolla olevaan rajoitettuun valikoimaan. Normaalisti palveluntarjoajat käyttävät mahdollisimman yleiskäyttöistä laitteistoa, mikä ei välttämättä palvele kaikkia erikoistarpeita. Mikäli organisaatiolla on erityistarpeita laitteiston suhteen, yksityinen pilvipalvelu voi olla parempi vaihtoehto. (Kavis 2014, 20.)

Kaikki pilvipalveluiden toiminnot eivät ole standardoituja eivätkä kaikki standardit ole avoimia, mikä voi vaikeuttaa palveluntarjoajan vaihtamista. Yhden palveluntarjoajan pilvessä toimiva järjestelmä ei välttämättä siirry kovin helposti toiseen pilvipalveluun. Oman järjestelmän muuttaminen toisen palvelun kanssa yhteensopivaksi saattaa vaatia paljon käsityötä ja tulla siksi kalliiksi. Esimerkiksi, jos käytetyn pilvipalvelun ohjelmointirajapinnat eivät ole yleisen standardin mukaisia, kaikki niitä käyttävät sovellukset toimivat vain kyseisessä pilvipalvelussa. Nämä rajoitteet voivat pilata pilvipalveluiden alkuperäisen idean palveluiden joustavuudesta. (Germain 2013.)

4 OPENSTACK

4.1 Mikä on Openstack

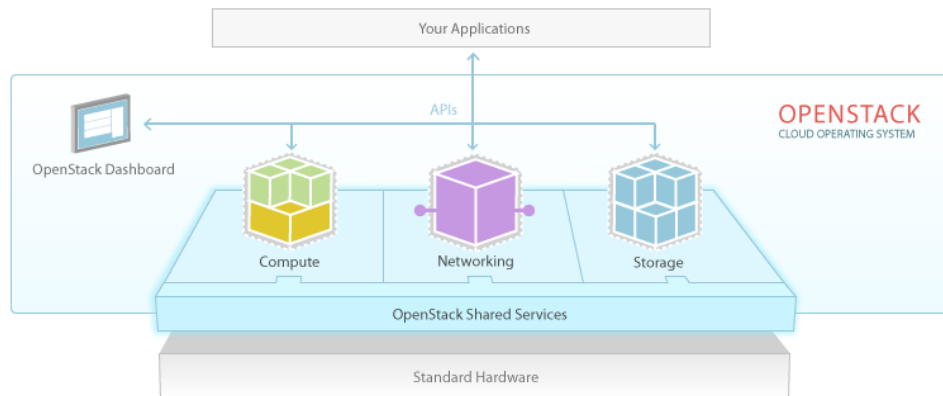
Openstack on Apache 2.0 -lisenssillä julkaistu avoimen lähdekoodin ohjelmistoalusta yksityisten ja julkisten pilvipalveluiden perustamiseen ja ylläpitoon. Toisin kuin esimerkiksi Virtualbox, Openstack ei ole työpöytäsovellus, vaan se asennetaan palvelimeen ja graafinen käyttöliittymä saadaan vain selaimen kautta. Projektin aloittivat NASA ja Rackspace vuonna 2010 (Jackson 2012, 1). Nykyään Openstackin kehitystä ohjaa voittoa tavoittelematon Openstack-säätiö. Säätiön jäsenenä on yli 9500 yksityishenkilöä ja 850 organisaatiota. Jäsenyys on ilmaista yksityishenkilöille, mutta organisaatioiden on tuettava säätiötä rahallisesti. (Openstack-säätiö 2014g.) Openstack asennetaan Linux-käyttöjärjestelmää käyttäviin laitteisiin ja siinä voidaan ajaa Windows- ja Linux-virtuaalikoneita sekä eräitä muita Unix-käyttöjärjestelmiä. Ubuntu ja Red Hat ovat yleisimmät alustat Openstack-asennuksille (Yegulalp 2014).

Openstackistä julkaistaan kaksi versiota vuodessa, yleensä huhtikuussa ja syys- tai loka-kuussa. Päivitykset taataan kahdelle viimeksi julkaistulle versiolle, mutta myös vanhemmat versiot saattavat saada turvapäivityksiä. (Openstack-säätiö 2014e.) Openstack-palveluita tarjoavat yritykset voivat tarjota pidempiä tukiaikoja käyttäjilleen. Esimerkiksi Canonical lupaa tukea Ubuntu pitkäaikaisesti tuetun version (LTS) mukana tulevaa Openstackin versiota 5 vuotta. (Ubuntu 2014.) Openstack-pilven päivitys versiosta toiseen ei ole ongelmaton, vaan saattaa pahimmillaan vaatia vanhan version poistoa ja uutta puhdasta asennusta (Darrow 2013). Uusien versioiden myötä päivitysprosessi on kuitenkin muuttunut aiempaa helpommaksi. Useat yritykset käyttävät Openstack-alustaa omien pilvipalveluidensa ylläpitoon, hyvinä esimerkkeinä Paypal ja Intel (Openstack-säätiö 2014h).

4.2 Openstackin rakenne

Openstackin yksinkertaistettu rakenne on esitetty kuviossa 5. Kuvion alaosassa on laitteisto, jonka päällä Openstackin palvelut toimivat. Koska Openstack on laitteistoriippumaton, laitteistossa voi olla erilaisia palvelimia, joiden ei tarvitse edes sijaita fyysisesti

samassa paikassa. Openstackin laitteistotuki on laaja, joten ei ole pelkoa lukkiutumisesta tietyn valmistajan laitteisiin. Openstackin palvelut yhdistävät nämä laitteet yhdeksi pilveksi. Openstack tarjoaa käyttäjän sovelluksille kolmea perusresurssia: laskentatehoa, verkkoa ja tallennustilaa. Resursseja tarjoavat palvelut viestivät keskenään yhteisten rajapintojen (API) kautta ja palveluita voi hallita myös graafisen selainkäyttöliittymän kautta. Palveluiden joustavuuden ansiosta Openstack pystyy ylläpitämään jopa kymmeniä tuhansia virtuaalikoneita (Page 2014).



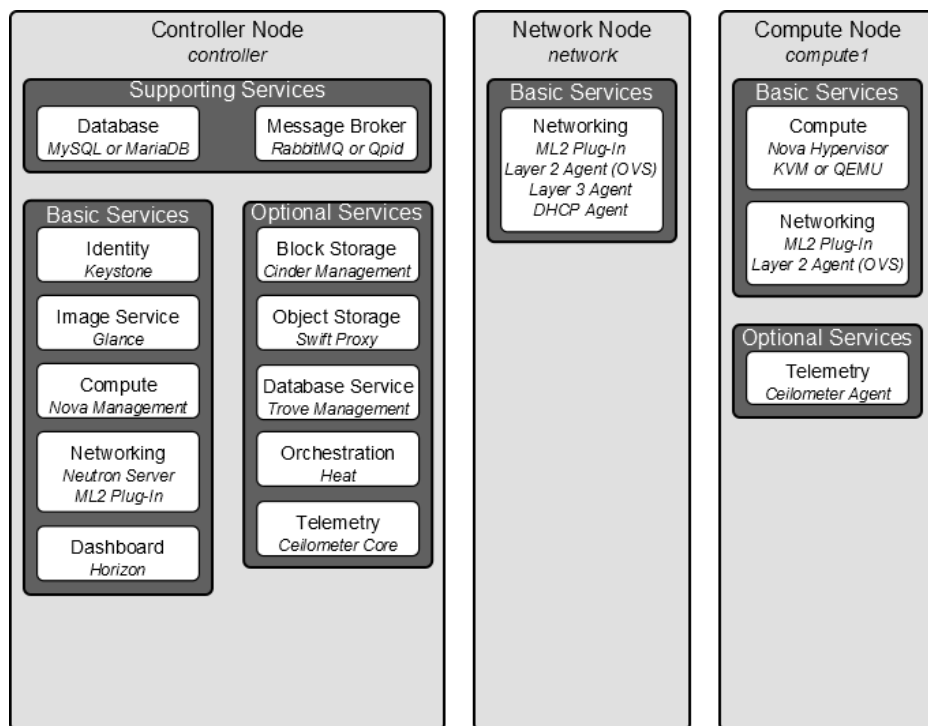
KUVIO 5. Openstack-alustan peruskomponentit (Openstack-säätiö 2014d)

Openstackin monimutkainen ja laaja kokonaisuus on jaettu useisiin erillisiin palveluihin, joista jokaisella on oma selvä tehtävänsä. Tämä helpottaa kehitystyötä ja ylläpitoa. Yhtä palvelua voidaan kehittää melko vapaasti muista riippumatta, kunhan vain sen rajapinta säilyy muiden palveluiden ymmärtämässä muodossa. Vikojen etsiminen helpottuu, kun ongelmasta voi usein päätellä missä palvelussa ongelma on. Lisäksi yhdessä palvelussa oleva vika ei välttämättä haittaa muiden palveluiden toimintaa. Palvelut ja niiden tehtävät on listattu liitteessä 1.

Kaikki palvelut eivät ole välttämättömiä Openstack-asennuksessa. Pakollisia palveluita tyypillisessä asennuksessa ovat virtuaalikoneita hallitseva Nova, verkkoyhteyksistä huolehtiva Neutron, levykuvista vastaava Glance ja käyttöoikeuksista huolehtivat Keystone. Suurten tietomäärien tallennukseen tarkoitettu Swift poikkeaa muista palveluista siinä, että se voi toimia täysin itsenäisesti, ilman muita palveluita. Vikasietoisuuden parantamiseksi sama palvelu voi toimia useassa laitteessa. Lisäksi tarvitaan palveluita, jotka eivät ole osa Openstack-projektia. Nämä ovat viestinvälitysohjelma (message broker), jonka avulla palvelut viestivät keskenään. Suositeltuja viestinvälittäjiä ovat Rabbitmq ja Qpid. Toinen vaadittu ulkopuolinen palvelu on SQL-tietokanta, johon palvelut voivat tallentaa tietojaan. Tyypillinen vaihtoehto on MySQL (Openstack-säätiö 2014c, 18).

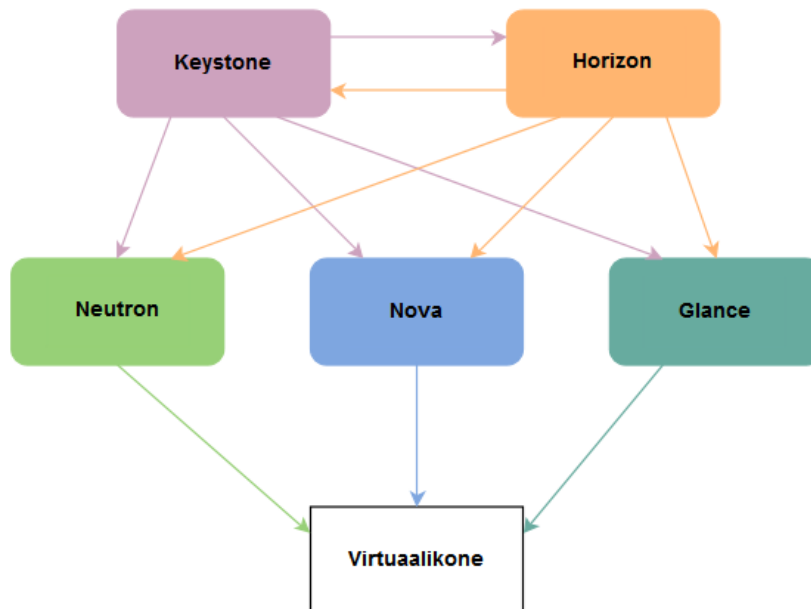
Openstackin kaikki palvelut voidaan asentaa samaan laitteeseen, mutta sen todelliset hyödyt tulevat ilmi vasta, kun useita laitteita liitetään yhdeksi Openstack-pilveksi. Pilven eri laitteita kutsutaan noodeiksi. Yhteen pilveen liitettyjen nooidien ei tarvitse sijaita fyysisesti samassa palvelinkeskuksessa, vaan ainoa vaatimus on nopea verkkoyhteys eri nooidien välillä. Pilven eri palvelinkeskukset voivat sijaita satojen kilometrien päässä toisistaan, esimerkiksi Euroopan hiukkasfysiikan tutkimuskeskuksella (CERN) on Openstack-pilvi, jonka palvelinkeskuksista toinen sijaitsee Sveitsissä ja toinen Unkarissa. (Bell 2014.)

Palveluita voidaan jakaa eri nooidien välillä melko vapaasti, mutta tyypillinen ratkaisu on jako kolmeen eri noodityyppiin. Näitä ovat ohjausnoodi (controller), verkkonoodi (network) ja tietokoneiden virtualisoinnin hoitava laskentanoodi (compute). Tyypillinen palveluiden jako nooidien välillä on esitetty kuviossa 6. Koska ohjaus- ja verkkonoodeihin kohdistuva rasitus on melko kevyttä ja pääosa laskentatehosta ja muistista vaaditaan laskentanoodilta, tyypillisessä pienen mittakaavan asennuksessa on yksi ohjausnoodi ja yksi verkkonoodi muutamia laskentanooideja. Jos vikasietoisuutta halutaan parantaa, ohjaus- ja verkkonoodit voidaan kahdentaa. Jos taas laitteiston määrä halutaan pitää pienenä, verkko- ja ohjausnoodit voidaan yhdistää.



KUVIO 6. Palveluiden jako noodeihin (Openstack-säätiö 2014c, 4, muokattu)

Openstackin tärkeimpien palveluiden väliset yhteydet näkyvät kuviossa 7. Kyseisten palveluiden toiminta on esitelty tarkemmin omissa alaluvuissaan. Kuviosta nähdään, miten Openstackin palvelut toimivat yhteistyössä tarjoten palveluita toisilleen ja virtuaalikoneille. Virtuaalikone saa tarvitsemansa resurssit (muun muassa prosessoriajan, verkkoyhteyden ja levykuvan) Novaalta, Neutronilta ja Glancelta. Ylemmän tason palvelut Keystone ja Horizon eivät ole suoraan yhteydessä virtuaalikoneeseen, vaan palvelevat muita palveluita. Keystone autentikoi muut palvelut oikeita tunnuksia vastaan. Horizon yhdistää muut palvelut graafiseen käyttöliittymään. Yksinkertaisuuden vuoksi kuviossa 7 on jätetty pois osa Openstackin palveluista, mutta kokonaiskuva kaikkien palveluiden välisistä yhteyksistä on esitelty liitteessä 2.



KUVIO 7. Openstackin peruspalveluiden välinen toiminta

4.3 Nova

Nova eli Openstack Compute on Openstackin laskentapalvelu, joka luo ja ylläpitää virtuaalikoneita ja tarjoaa niille tarvittut fyysisten laitteiden resurssit. Nova yhdistää Openstackin käytössä olevat laitteet yhdeksi pilveksi, josta virtuaalikoneet saavat muistia, prosessoriaikaa ja tallennustilaa joutumatta välittämään siitä, mistä laitteesta nämä resurssit tulevat. (Jackson 2013, 52.) Kun virtuaalikone perustetaan, Nova varaa sille tarvittun määrän resursseja, käytön aikana Nova muuttaa resurssien määrää tarpeen mukaan ja kun virtuaalikone poistetaan, Nova vapauttaa käytössä olleet resurssit. Novan virtuaalikoneita

hallitaan SSH-yhteydellä tai graafisesti VNC:llä. Lisäksi Nova kykenee tarjoamaan virtuaalikoneille yksinkertaisen verkkoyhteyden, jos verkkopalvelu Neutronin laajempia ominaisuuksia ei haluta tai voida käyttää.

Vaikka Nova hallitseekin virtuaalikoneita, se ei ole hypervisor, vaan käyttää virtuaalikoneiden ohjaamiseen käyttäjän valitsemaa hypervisoria. Nova kykenee toimimaan useiden eri hypervisoreiden kanssa, mutta hypervisoreiden tuetuissa ominaisuuksissa on suuria eroja. Xenserver, KVM ja QEMU ovat parhaiten tuettuja ja myös Xenissä ja VMwaressa on melko laajat ominaisuudet, mutta esimerkiksi Docker on vielä kokeellinen, eikä sen tuotantokäyttöä suositella. (Openstack-säätiö 2014b.)

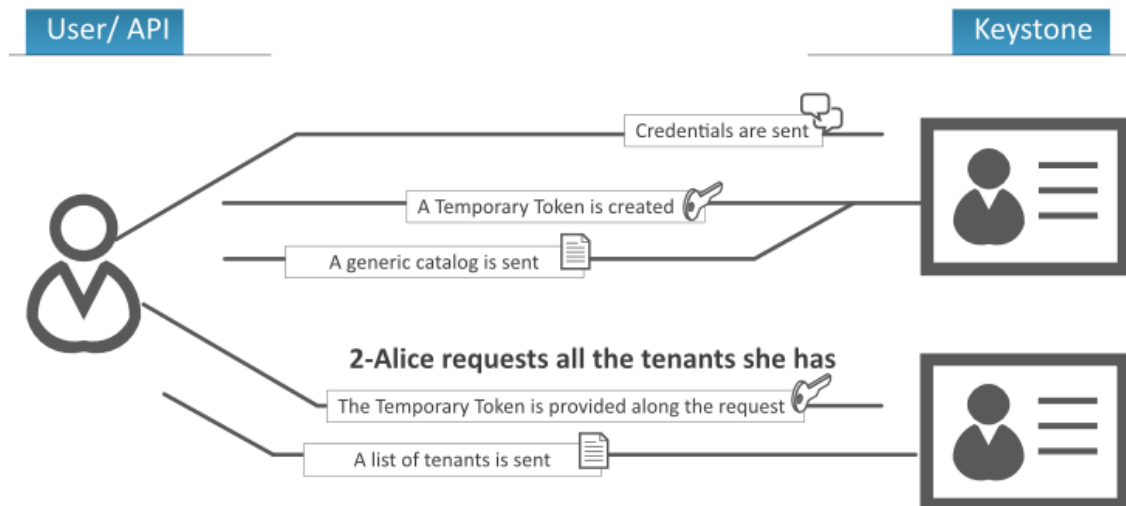
Nova koostuu joukosta käyttöjärjestelmän palveluina ajettavia ohjelmia, jotka on toteutettu pääosin Pythonilla. Laskentanoodissa sijaitseva Nova-compute ohjaa virtuaalikoneiden toimintaa hypervisorin kautta. Muut Novan palvelut sijaitsevat yleensä ohjausnoodissa. Muita tärkeitä palveluita ovat Nova-api, joka kuljettaa viestejä Novan palveluiden, käyttäjien ja virtuaalikoneiden välillä, Nova-scheduler, joka päättää mihin laitteeseen virtuaalikone perustetaan, ja Nova-conductor, joka huolehtii Nova-computen ja tietokannan välisestä viestinnästä. (Openstack-säätiö 2014c, 41–42.) Kaikkien palveluiden tunteminen ei ole välttämätöntä Novan asennuksessa ja käytössä.

4.4 Keystone

Keystone on Openstackin identiteettipalvelu, joka luo ja autentikoi käyttäjät ja muut palvelut, ja määrittää mitä oikeuksia käyttäjillä on. Openstack-ympäristöä luotaessa Keystone on asennettava ensin, koska muut palvelut tarvitsevat siltä saatavia oikeuksia. Keystone asennetaan yleensä ohjausnoodiin. (Jackson 2013, 5.) Käyttäjiä voidaan jakaa ryhmiin (tenant) ja heille voidaan antaa valmiiksi määriteltyjä rooleja. Keystone pitää myös kirjaa porteista ja ip-osoitteista, joiden kautta muiden palveluiden rajapinnat tavoitetaan. (Openstack-säätiö 2014c, 22–23.) Lisäksi käyttäjille ja ryhmille tarjolla olevia resursseja voidaan rajoittaa, esimerkiksi jokaiselle käyttäjälle voidaan antaa vain yksi julkinen ip-osoite.

Kuvio 8 esittää miten käyttäjä ensin todentaa itsensä palvelulle ja saa sitten palvelulta haluamansa tiedot. Ensin käyttäjä lähettää Keystoneille tunnuksensa ja salasanasensa. Koska

tunnukset ovat oikein, käyttäjä saa vastaukseksi väliaikaisen avaimen (token). Lähettämällä väliaikaisen avaimen Keystoneelle käyttäjä saa haluamansa tiedot, kunhan hänen oikeutensa riittävät niiden saamiseen. Kuviossa käyttäjä pyytää listaa kaikista Keystoneen luoduista ryhmistä, mutta pyydetty tieto voisi olla esimerkiksi tietyn palvelun ip-osoite tai lista käytössä olevista palveluista.



KUVIO 8. Käyttäjän todennus ja tietojen pyytäminen (Openstack-säätiö 2014c, 23, muokattu)

Muut Openstackin palvelut kysyvät tietoa Keystoneelta vastaavaan tapaan. Jos käyttäjä haluaa esimerkiksi perustaa uuden virtuaalikoneen, hän lähettää Novalle komennon sekä tunnuksensa ja salasanansa. Nova tarkistaa Keystoneelta, onko käyttäjällä oikeus perustaa virtuaalikone. Keystoneen vastauksen perusteella joko luo virtuaalikoneen tai palauttaa virheilmoituksen. Lisäksi Keystone varmistaa, että vain hyväksytyt laitteet voivat liittyä osaksi Openstack-pilveä. Koko valtuutusprosessi käyttäjän, Keystoneen ja muiden palveluiden välillä on kuvattu liitteessä 3.

Vaikka Keystone parantaakin Openstackin tietoturvaa, on tärkeää huomata, että oletusasetuksilla tieto palveluiden ja käyttäjien välillä liikkuu salaamattomassa muodossa. Palvelut tukevat SSL-salausta, mutta se on otettava erikseen käyttöön. Jos verkko ei ole turvallinen, perustilassa ulkopuolinen taho voi helposti ohittaa Keystoneen kaappaamalla liikenteestä esimerkiksi salasanat (kuvio 9).

```

Follow TCP Stream (tcp.stream eq 2)
Stream Content
POST /v2.0/tokens HTTP/1.1
Host: 192.168.112.2:35357
Content-Length: 111
Content-Type: application/json
Accept-Encoding: gzip, deflate, compress
Accept: */*
User-Agent: python-keystoneclient
{"auth": {"tenantName": "admin", "passwordCredentials": {"username": "admin",
"password": "admin"} }}HTTP/1.1 200 OK
Vary: X-Auth-Token
X-Distribution: ubuntu
Content-type: application/json
Content-Length: 3150

```

KUVIO 9. Wiresharkilla kaapattu admin-tunnuksen salasana

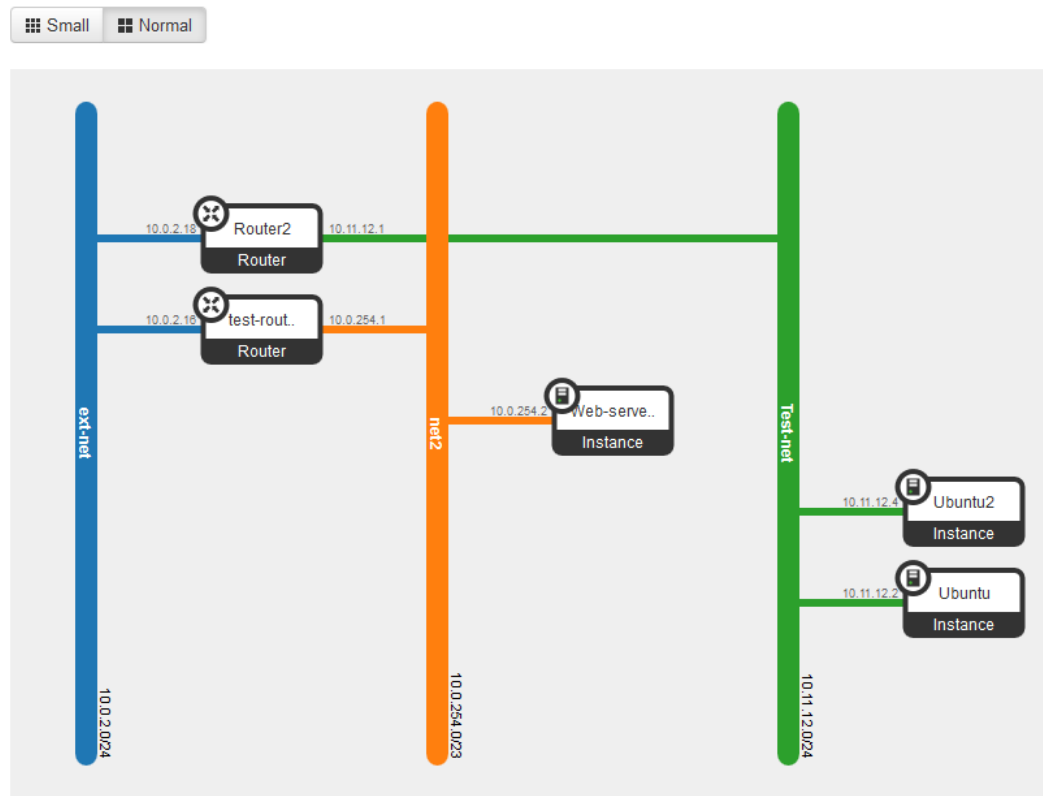
4.5 Neutron

Openstackin verkkopalvelu Neutron luo virtuaalisen verkon, jonka kautta virtuaalikoneet viestivät keskenään ja siirtää dataa virtuaaliverkon ja fyysisen verkon välillä. Aiemmin verkkopalvelun nimi oli Quantum, mutta nimi jouduttiin vaihtamaan tekijänoikeussyistä. Vanhemmissa dokumentaatioissa ja ohjeissa saatetaan yhä käyttää vanhaa nimeä. Neutronin virtuaaliverkoissa voilla muun muassa vlaneja ja virtuaalisia reitittimiä. Lisäksi Neutron tarjoaa DHCP-palvelun ja palomuurin. (Jackson 2013, 168.)

Neutron tukee useita erilaisia virtuaaliverkkoja. Näistä yksinkertaisin on Flat Network, jossa verkot ovat kaikkien ryhmien näkyvissä ja virtuaalikoneilla on vain kiinteitä ip-osoitteita. Yhteys ulkomaailmaan on konfiguroitava manuaalisesti fyysisillä reitittimillä. Kehittyneempiä tyyppisiä ovat muun muassa vlan ja gre. Ne erottelevat virtuaaliverkot toisistaan kapseloimalla (encapsulation) paketit ja tarjoavat enemmän ominaisuuksia, kuten julkisten ip-osoitteiden antamisen virtuaalikoneille. (Openstack-säätiö 2014f.)

Kuvio 10 esittää Neutronilla luotuja virtuaaliverkkoja. Vasemmanpuoleisin verkko (ext-net) on fyysinen verkko, jonka kautta virtuaaliverkkojen laitteet voivat viestiä Openstackin ulkopuolisten laitteiden ja internetin kanssa. Virtuaaliset reitittimet ohjaavat fyysisestä verkosta tulevat yhteydet virtuaaliverkkoihin ja yhteydet virtuaaliverkoista ulkomaailmaan. Reitittimet toimivat myös palomureina verkkojen välillä. Virtuaalikoneita voidaan jakaa useisiin virtuaaliverkkoihin, jos esimerkiksi halutaan erottaa tietyt palvelut toisistaan. Jos virtuaalikoneen on toimittava palvelimena, sen virtuaaliverkon yksityiseen ip-osoitteeseen voidaan liittää julkinen IP, jonka avulla se on tavoitettavissa internetistä.

Network Topology



KUVIO 10. Neutronin virtuaaliverkkoja

4.6 Glance

Glance on Openstackin levykuvapalvelu. Sen tehtäviä ovat levykuvien luominen, valmiiden levykuvien lisääminen järjestelmään sekä levykuvien tarjoaminen Novalle virtuaalikoneiden perustamista varten. Lisäksi käynnissä olevista virtuaalikoneista otetut vedokset (snapshot) tallennetaan Glanceen ja noudetaan sieltä kun virtuaalikone halutaan palauttaa käyttöön. Tästä hyötyä esimerkiksi tehtäessä riskialtista päivitystä, jos palvelin lakkaa toimimasta, sen voi palauttaa aikaisempaan tilaan erittäin nopeasti.

Glanceen tallennetut tiedot voivat sijaita paikallisella kiintolevyllä, verkkolevyllä tai Openstackin datantallennuspalvelu Swiftissä. (Jackson 2013, 35.) Glanceen voi lisätä useissa erimuodoissa tallennettuja levykuvia, muun muassa Amazonin pilvipalvelun levykuvia (AMI, Amazon Machine Image) ja Virtualboxin käyttämiä VDI-kuvia sekä isomuotoisia levykuvia (Openstack-säätiö 2014c, 38). Kuvio 11 esittää, miten Glanceen liitetään uusi Ubuntuun pohjautuva levykuva.

```
root@openstack-testi:/home/ubuntu# glance image-create --name=Ubuntu3 --container-format=bare --disk-format=qcow2 < ubuntu14.04.iso
```

Property	Value
checksum	08d25bf879e353686a974b7b14ae7d81
container_format	bare
created_at	2014-10-20T13:10:07
deleted	False
deleted_at	None
disk_format	qcow2
id	8e26fcac-05cc-4bc5-9591-cca2ecd2a477
is_public	False
min_disk	0
min_ram	0
name	Ubuntu3
owner	7b92ae60d3d14c648072c04256f0ccca
protected	False
size	575668224
status	active
updated_at	2014-10-20T13:10:18
virtual_size	None

KUVIO 11. Uuden levykuvan lisääminen

4.7 Horizon

Horizon on Openstackin graafinen käyttöliittymä, joka toimii selaimessa. Vaikka Openstackin käyttö perustuukin komentorivin kautta annettuihin komentoihin, monet asiat on kätevämpää hoitaa graafisen käyttöliittymän kautta. Lisäksi monimutkaisten virtuaaliverkkojen rakenne on paljon helpompi ymmärtää kuvana kuin tekstinä (kuvio 10). Horizonin käyttöliittymässä ovat Openstackin olennaisimmat toiminnot ja tiedot, mutta monimutkaisemmat toiminnot on hoidettava komentorivin kautta. Horizon on toteutettu Apachella ja Django-ympäristöllä. (Jackson 2013, 217.)

Valmiin Openstack-ympäristön käyttö Horizonin avulla on hyvin helppoa. Openstackin peruskäyttö, kuten virtuaalikoneiden perustaminen (kuvio 12), ei vaadi erityistä perehtymistä virtualisointiin tai Openstackiin. Virtuaalikonetta perustettaessa on ainoastaan valittava koneelle nimi, käytettävä levykuva, koneelle annettavat resurssit sekä eräitä muita tietoja. Kun tiedot on annettu, uusi virtuaalikone on käyttövalmis muutamassa minuutissa.

Launch Instance

Launch Instance

Details *
Access & Security *
Networking *
Post-Creation
Advanced Options

Availability Zone:
nova

Instance Name: *
Testikone

Flavor: *
m1.medium

Instance Count: *
1

Instance Boot Source: *
Boot from image

Image Name:
Ubuntu server (549.0 MB)

Specify the details for launching an instance.
The chart below shows the resources used by this project in relation to the project's quotas.

Flavor Details

Name	m1.medium
VCPUs	2
Root Disk	40 GB
Ephemeral Disk	0 GB
Total Disk	40 GB
RAM	4,096 MB

Project Limits

Number of Instances 2 of 10 Used

Number of VCPUs 2 of 20 Used

Total RAM 1,024 of 51,200 MB Used

Launch

KUVIO 12. Uuden virtuaalikoneen perustaminen

5 OPENSTACKIN ASENNUSOHJE

5.1 Taustaa

Tämä asennusohje kertoo, kuinka Openstack asennetaan toimimaan yhdessä laitteessa (all-in-one -asennus). Tämä asennustapa sopii hyvin esimerkiksi Openstackin testaamiseen, koska sen toteuttaminen on yksinkertaista ja nopeaa. Yhdessä palvelinlaitteessa voi helposti ajaa kymmeniä virtuaalikoneita, joten all-in-one -tyypin asennus voi olla täysin riittävä moniin tarpeisiin myös vakavassa käytössä. Tämä ohjeen mukainen asennus sisältää peruspalvelut, joita tarvitaan virtuaalikoneiden luomiseen, käyttöön ja hallintaan, mutta ei lisäpalveluiden tuomia erikoisominaisuuksia. (Palvelut ja niiden ominaisuudet on esitelty liitteessä 1.)

Jos tarkoitus on ainoastaan testata Openstackin toimintaa, ei tarvita kovin tehokasta tietokonetta. 2 gigatavua keskusmuistia, 10 gigatavua vapaata kiintolevytilaa ja 2 prosessoriydintä riittävät hyvin testaukseen. Testausta varten Openstackin voi asentaa myös virtuaalikoneeseen, kunhan hypervisorin verkkoasetuksissa sallitaan promiscuous-tila. Esimerkiksi Virtualbox kelpaa hyvin tähän tarkoitukseen.

Jos Openstackin asennus tehdään todellista käyttöä varten, laitteelta vaaditaan testilaitetta parempaa suorituskykyä. Tarkkoja minimivaatimuksia on vaikea antaa, koska käyttötapa vaikuttaa hyvin paljon syntyvään rasitukseen. Hyvä lähtökohta on kuitenkin 4 virtuaalikonetta yhtä prosessoriydintä kohti ja fyysistä keskusmuistia 70 % virtuaalikoneille annettavasta keskusmuistinmäärästä. Tarvittava kiintolevytilan määrää riippuu täysin käyttötarkoituksesta ja käytettävien käyttöjärjestelmien koosta, mutta Openstackille ja isäntäkoneen käyttöjärjestelmälle kannattaa varata ainakin 30 gigatavua.

Tämä ohje on tehty Ubuntun versiolle 14.04 LTS, mutta asennusprosessi on hyvin samanlainen myös muille Ubuntun tuetuille versioille. Asennettava Openstackin versio on Icehouse. Esimerkeissä tietokoneen nimi on Icehouse ja käyttäjänimi on Ubuntu. Asennuksessa laitteen ip-osoite on 192.168.112.99, tämä tulee muuttaa omaa ip-osoitetta vastaavaksi. Asennuksessa käytetty hypervisor on KVM. Vaikka tämän ohjeen käyttämät salasanat ovatkin heikkoja, julkiseen internetiin kytkettävässä palvelussa tulee käyttää vahvempia salasanvoja.

Kun tässä ohjeessa muokataan konfiguraatitiedostoja, ohje mainitsee ainoastaan rivit, jotka tulee lisätä tiedostoihin. Jos vastaava rivi on jo olemassa tiedostossa, alkuperäinen arvo tulee korvata ohjeen arvolla. Otsikot on merkitty hakasulkein ja tiedot pitää lisätä merkityn otsikon alle, jos otsikko on osoitettu ohjeessa. Liitteissä 4–9 esitellään mallit konfiguraatitiedostoista.

5.2 Taustapalveluiden asennus

Ennen Openstackin omien pakettien asentamista pitää asentaa Openstackin toimintaa tukevia palveluita. Openstackin palvelut tarvitsevat käyttöönsä tietokannan, tässä ohjeessa tietokannaksi on valittu MySQL. Openstackin palveluiden välistä viestintää varten asennetaan viestivälityspalvelimeksi Rabbitmq.

Asennetaan tarvittu paketit (1).

```
1)          sudo apt-get install mysql-server python-mysqldb rabbitmq-server
```

Openstack vaatii tiettyjen MySQL:n perusasetusten muuttamista. Tiedostossa `/etc/mysql/my.cnf` otsikon `[mysqld]` alle lisätään kohdan 2 mukaiset rivit. `Bind-address` saattaa olla valmiiksi määritetty, tässä tapauksessa poista alkuperäinen rivi.

```
2)          bind-address = 192.168.112.99
            default-storage-engine = innodb
            innodb_file_per_table
            collation-server = utf8_general_ci
            init-connect = 'SET NAMES utf8'
            character-set-server = utf8
```

Tämän jälkeen MySQL:n asennus on viimeisteltävä kohdan 3 komennoilla. `Mysql_install` pyytää valitsemaan tietokannan root-käyttäjän salasanan. Seuraaviin `secure_installin` kysymyksiin vastataan kyllä: Poista anonyymit käyttäjät, estä root-käyttäjän etäkirjautuminen, poista testitietokanta ja lataa taulukot uudelleen.

```
3)          sudo mysql_install_db
```

```
sudo mysql_secure_installation
```

Asetetaan Rabbitmq:n salasana komennolla 4. (Oletuskäyttäjä Guestin poistaminen ja korvaaminen toisella tunnuksella parantavat tietoturvaa.)

```
4) rabbitmqctl change_password guest rabbitpass
```

Käynnistetään palvelut uudelleen, jotta muutokset tulevat voimaan (5).

```
5) sudo service rabbitmq-server restart  
sudo service mysql restart
```

Siirrytään MySQL:n root-tilaan (6) kohdassa 3 valitulla salasanalla:

```
6) mysql -u -root -p
```

Luodaan tietokannat ja käyttäjät Openstackin palveluita varten (7).

```
7) CREATE DATABASE keystone;  
GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'localhost' IDENTIFIED BY  
'keystonepass';  
GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'%' IDENTIFIED BY  
'keystonepass';  
  
CREATE DATABASE glance;  
GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'localhost' IDENTIFIED BY  
'glancepass';  
GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'%' IDENTIFIED BY 'glancepass';  
  
CREATE DATABASE nova;  
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'localhost' IDENTIFIED BY  
'novapass';  
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'%' IDENTIFIED BY 'novapass';  
  
CREATE DATABASE neutron;  
GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'localhost' IDENTIFIED BY  
'neutronpass';
```

```
GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@%' IDENTIFIED BY
'neutronpass';
```

Poistutaan tietokannasta (8).

```
8)          exit;
```

Openstackin verkot käyttävät virtuaalisia kytkimiä virtuaalikoneiden väliseen viestintään. Virtuaalisen kytkimen käyttöönotto saattaa katkaista laitteen verkkoyhteyden, mikäli asennuksessa tekee virheen. Siksi tässä vaiheessa kannattaa varmistaa, että laite on käytettävissä paikallisen konsolin kautta, jos SSH-yhteys katkeaa.

Asennetaan tarvittut paketit (9).

```
9)          sudo apt-get install openvswitch-common openvswitch-switch
```

Muokataan tiedostoon `/etc/network/interfaces` kohdan 10 mukaiset rivit. Eth0 vastaa laitteen verkkoliitintä, jolla ollaan yhteydessä internetiin. Tarkista oman laitteesi verkkoliitimen koodi ja tarvittaessa muuta kohdan 10 koodia vastaavasti. Korvaa ip-osoitteet oman verkkosi osoitteilla.

```
10)         auto lo
            iface lo inet loopback

            auto br-ex
            iface br-ex inet static
            address 192.168.112.99
            netmask 255.255.255.0
            network 192.168.112.0
            broadcast 192.168.112.255
            gateway 192.168.112.1
            up ifconfig $IFACE promisc
            dns-nameservers 8.8.8.8 8.8.4.4

            auto eth0
            iface eth0 inet manual
```

```

up ip address add 0/0 dev $iface
up ip link set $IFACE up
up ifconfig $IFACE promisc
up ifconfig $IFACE multicast
down ip link set $IFACE down

```

Luodaan virtuaalinen kytkin kohdan 11 komennoilla. Ensimmäisen komento luo virtuaalisen kytkimen ja toinen liittää sen fyysiseen verkkoliittimen ja käynnistää koneen uudelleen, jotta muutokset tulevat voimaan. Kun laite on käynnistynyt uudelleen, verkkoyhteyden pitäisi toimia normaalisti.

```

11)      sudo ovs-vsctl add-br br-ex
          sudo ovs-vsctl add-port br-ex eth0 && sudo shutdown -r now

```

5.3 Keystone asennus

Asennetaan Keystone isäntäpaketti sekä asiakasohjelma, jolla Keystonea käytetään (12).

```

12)      sudo apt-get install keystone python-keystoneclient

```

Konfiguroidaan Keystone asetustiedoon `/etc/keystone/keystone.conf` kohdan 13 mukaiset tiedot. Parametriin `admin_token` keksitään väliaikainen salasana, jolla Keystonea voi käyttää ennen kuin siihen on luotukäyttäjätunnuksia. Kun Admin-tunnus on luotu ja todettu toimivaksi, kannattaa väliaikainen salasana poistaa tietoturvan parantamiseksi.

```

13)      [default]
          admin_token=temptoken
          rabbit_password=rabbitpass
          [database]
          connection = mysql://keystone:keystonepass@192.168.112.99/keystone
          log_dir=/var/log/keystone

```

Otetaan Keystone käyttöön (14).


```
14)      sudo su -s /bin/sh -c "keystone-manage db_sync" keystone
          sudo service keystone restart
```

Poistetaan asennuksessa syntynyt tarpeeton tietokanta (15).

```
15)      sudo rm /var/lib/keystone/keystone.db
```

Otetaan väliaikainen salasana käyttöön ja luodaan sen avulla admin-käyttäjätunnus, käyttäjäryhmä ja muita tarpeellisia asetuksia. Luodaan myös oma ryhmä palveluille ja rajapinta, jonka kautta Keystonea käytetään. (16).

```
16)      export OS_SERVICE_TOKEN=temptoken
          export OS_SERVICE_ENDPOINT=http://192.168.112.99:35357/v2.0
          keystone user-create --name=admin --pass=adminpass
          keystone role-create --name=admin
          keystone tenant-create --name=admin --description="Admin Tenant"
          keystone user-role-add --user=admin --tenant=admin --role=admin
          keystone user-role-add --user=admin --role=_member_ --tenant=admin
          keystone tenant-create --name=service --description="Service Tenant"
          keystone service-create --name=keystone --type=identity --
          description="OpenStack Identity"
          keystone endpoint-create --service-id=$(keystone service-list | awk '/
          identity / {print $2}') --publicurl=http://192.168.112.99:5000/v2.0 --
          internalurl=http://192.168.112.99:5000/v2.0 --
          adminurl=http://192.168.112.99:35357/v2.0
```

Poistetaan väliaikainen tunnus käytöstä (17).

```
17)      unset OS_SERVICE_TOKEN OS_SERVICE_ENDPOINT
```

Nyt voidaan käyttää kohdassa 16 luotua admin-tunnusta Openstackin palveluiden hallintaan. Käytön helpottamiseksi tunnuksen tiedot kannattaa tallentaa tiedostoon. Tiedosto päätteeksi tulee .sh, esimerkiksi tunnuks.sh (18).

```
18)      export OS_USERNAME=admin
```

```
export OS_PASSWORD=adminpass
export OS_TENANT_NAME=admin
export OS_AUTH_URL=http://192.168.112.99:35357/v2.0
```

Tunnukset ladataan muistiin komennolla 19. (Suorita komento kansiossa jossa tiedosto sijaitsee).

```
19)      source tunnuks.sh
```

Kohdan 19 komennon jälkeen Keystone on valmis käyttöön. Toiminnan voi testata esimerkiksi komennolla 20, joka antaa listan Keystoneen luoduista käyttäjistä. Vastauksena komento antaa listan, jossa on yksi käyttäjätunnus, admin.

```
20)      keystone user-list
```

5.4 Glancen asennus

Asennetaan Glanceen liittyvät paketit (21).

```
21)      sudo apt-get install glance python-glanceclient
```

Lisätään tiedostoihin /etc/glance/glance-api.conf ja /etc/glance/glance-registry.conf kohdan 22 mukaiset tiedot. Jotkut parametrit löytyvät tiedostoista valmiina ja niiden kohdalla pitää varmistaa, ettei sama parametri tule tiedostoon kahdesti. Hakasuluissa on merkitty, minkä otsikon alle parametrit kuuluvat.

```
22)      [default]
          rpc_backend = rabbit
          rabbit_host = 192.168.112.99
          rabbit_password = rabbitpass
          [database]
          connection = mysql://glance:glancepass@192.168.112.99/glance

          [keystone_authtoken]
```

```

auth_uri = http://192.168.112.99:5000
auth_host = 192.168.112.99
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = glance
admin_password = glancepass
[paste_deploy]
flavor=keystone

```

Varmistetaan, että asennus ei ole luonut ylimääräistä tietokantatiedostoa (23) ja otetaan Glance käyttöön (24).

```
23)      sudo rm /var/lib/glance/glance.sqlite
```

```
24)      sudo su -s /bin/sh -c "glance-manage db_sync" glance
```

Luodaan Glanceille käyttäjätunnus ja rajapinta Keystoneen (25) ja käynnistetään palvelut uudelleen, jotta muutokset tulevat voimaan (26).

```
25)      keystone user-create --name=glance --pass=glancepass
          keystone user-role-add --user=glance --tenant=service --role=admin
          keystone service-create --name=glance --type=image --
          description="OpenStack Image Service"
          keystone endpoint-create --service-id=$(keystone service-list | awk '/
          image / {print $2}') --publicurl=http://192.168.112.99:9292 --
          internalurl=http://192.168.112.99:9292 --
          adminurl=http://192.168.112.99:9292

```

```
26)      sudo service glance-registry restart
          sudo service glance-api restart

```

Testataan Glancen toiminta luomalla levykuva (27). Testaukseen sopii hyvin esimerkiksi erittäin pienikokoista Cirros-jakelu. Viimeisen komennon pitäisi palauttaa lista, jossa on yksi Cirros-niminen levykuva. Tarkistetaan tulos (28).

- 27) `wget http://cdn.download.cirros-cloud.net/0.3.2/cirros-0.3.2-x86_64-disk.img`
`glance image-create --name "cirros-0.3.2-x86_64" --disk-format qcow2 --container-format bare --is-public True --progress < cirros-0.3.2-x86_64-disk.img`
- 28) `glance image-list`

5.5 Novan asennus

Novaan kuuluvien pakettien asennus (28).

- 28) `sudo apt-get install nova-api nova-cert nova-conductor nova-consoleauth novanovncproxy nova-scheduler python-novaclient nova-compute-kvm python-guestfs`

Openstackin palveluille pitää antaa lukuoikeus kerneliin (29).

- 29) `sudo dpkg-statoverride --update --add root root 0644 /boot/vmlinuz-$(uname -r)`

Lisäksi on tehtävä skripti, joka antaa lukuoikeudet myös päivityksissä tuleviin uudempiin kernelin versioihin. Skripti tallennetaan sijaintiin `/etc/kernel/postinst.d/` nimellä `statoverride` (30). Lisäksi skriptitiedostoon on annettava suoritusoikeudet (31).

- 30) `#!/bin/sh`
`version="$1"`
`[-z "${version}"] && exit 0`
`dpkg-statoverride --update --add root root 0644 /boot/vmlinuz-${version}`
- 31) `sudo chmod +x /etc/kernel/postinst.d/statoverride`

Konfiguroidaan Novan asetukset tiedostoon `/etc/nova.nova.conf` (32). Otsikon database alainen connection ja keystone_authotokenin alaiset parametrit ovat todennäköisesti tiedostossa valmiina, tällöin kohdan 32 mukaiset arvot lisätään vanhojen arvojen tilalle.

```

32) [default]
    rpc_backend = rabbit
    rabbit_host = 192.168.112.99
    rabbit_password = rabbitpass
    my_ip = 192.168.112.99
    vncserver_listen = 192.168.112.99
    vncserver_proxycient_address = 192.168.112.99
    vnc_enabled=True
    vncserver_listen=0.0.0.0
    novncproxy_base_url=http://192.168.112.99:6080/vnc_auto.html
    glance_host = 192.168.112.99
    auth_strategy = keystone

[database]
    connection = mysql://nova:novapass@192.168.112.99/nova

[keystone_authtoken]
    auth_uri = http://192.168.112.99:5000
    auth_host = 192.168.112.99
    auth_port = 35357
    auth_protocol = http
    admin_tenant_name = service
    admin_user = nova
    admin_password = novapass

```

Varmistetaan, että rautatason tukivirtualisoinnille löytyy (33). Jos tulos on yli nollan, laitteisto tukee virtualisointia. Jos tulos on 0, KVM ei sovellu hypervisoriksi, vaan on käytettävä Qemua. Tällöin muokataan tiedostoon `/etc/nova/nova-compute.conf` kohdan 34 mukainen koodi. Qemua tarvitaan lähinnä siinä tilanteessa, että Openstack-asennus tehdään virtuaalikoneeseen.

```
33) egrep -c '(vmx|svm)' /proc/cpuinfo
```

```
34) [libvirt]
    virt_type = qemu
```

Luodaan Novalle käyttäjätunnus ja rajapinta Keystoneen (35).

```
35)      keystone user-create --name=nova --pass=novapass
        keystone user-role-add --user=nova --tenant=service --role=admin
        keystone service-create --name=nova --type=compute --
        description="OpenStack Compute"
        keystone endpoint-create --service-id=$(keystone service-list | awk '/
        compute / {print $2}') --
        publicurl=http://192.168.112.99:8774/v2/%\(tenant_id\)s --
        internalurl=http://192.168.112.99:8774/v2/%\(tenant_id\)s --
        adminurl=http://192.168.112.99:8774/v2/%\(tenant_id\)s
```

Poistetaan oletustietokanta, mikäli se on olemassa (36) ja otetaan Nova käyttöön (37).

```
36)      rm /var/lib/nova/nova.sqlite
```

```
37)      sudo su -s /bin/sh -c "nova-manage db sync" nova
```

Käynnistetään palvelut uudelleen, jotta muutokset tulevat voimaan (38).

```
38)      sudo service nova-api restart
        sudo service nova-cert restart
        sudo service nova-consoleauth restart
        sudo service nova-scheduler restart
        sudo service nova-conductor restart
        sudo service nova-novncproxy restart
        sudo service nova-compute restart
```

Testataan Novan toiminta komennolla 39. Tuloksena pitäisi olla lista, jossa näkyy Glancen asennuksen lopussa luotu Cirros-levykuva.

```
39)      nova image-list
```

5.6 Neutronin asennus

Asennetaan Neutroniin kuuluvat paketit (40) ja luodaan Keystoneen rajapinta ja käyttäjätunnus (41).

- ```
40) sudo apt-get install neutron-server neutron-plugin-ml2 neutron-plugin-
 openswitch-agent neutron-l3-agent neutron-dhcp-agent

41) keystone user-create --name=neutron --pass=neutronpass
 keystone user-role-add --user=neutron --tenant=service --role=admin
 keystone service-create --name=neutron --type=network --
 description="OpenStack Networking"
 keystone endpoint-create --service-id=$(keystone service-list | awk '/
 network / {print $2}') --publicurl=http://192.168.112.99:9696 --
 internalurl=http://192.168.112.99:9696 --
 adminurl=http://192.168.112.99:9696
```

Otetaan Keystoneen palveluiden tunnus talteen (42).

- ```
42)      keystone tenant-get service
```

Konfiguroidaan Neutronin asetustiedosto `/etc/neutron/neutron.conf` (43). Huomiotava, että parametrejä tulee usean eri otsikon alle.

- ```
43) [DEFAULT]
 rpc_backend = neutron.openstack.common.rpc.impl_kombu
 rabbit_host = 192.168.112.99
 rabbit_password = rabbitpass
 notify_nova_on_port_status_changes = True
 notify_nova_on_port_data_changes = True
 nova_url = http://192.168.112.99:8774/v2
 nova_admin_username = nova
 nova_admin_tenant_id = TÄHÄN KOHDAN 42 ID
 nova_admin_password = novapass
 nova_admin_auth_url = http://192.168.112.99:35357/v2.0
```

```
core_plugin = ml2
service_plugins = router
allow_overlapping_ips = True
auth_strategy = keystone
verbose = True
[database]
connection = mysql://neutron:neutronpass@192.168.112.99/neutron
[keystone_authtoken]
auth_uri = http://192.168.112.99:5000
auth_host = 192.168.112.99
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = neutron
admin_password = neutronpass
```

Konfiguroidaan tiedosto /etc/neutron/plugins/ml2/ml2\_conf.ini (44).

```
44) [ml2]
type_drivers = gre
tenant_network_types = gre
mechanism_drivers = openvswitch
[ml2_type_gre]
tunnel_id_ranges = 1:1000
[securitygroup]
firewall_driver =
neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver
enable_security_group = True
[ovs]
local_ip = 192.168.112.99
tunnel_type = gre
enable_tunneling = True
```

Konfiguroidaan tiedosto /etc/nova/nova.conf (45).



```

45) [DEFAULT]
network_api_class = nova.network.neutronv2.api.API
neutron_url = http://192.168.112.99:9696
neutron_auth_strategy = keystone
neutron_admin_tenant_name = service
neutron_admin_username = neutron
neutron_admin_password = neutronpass
neutron_admin_auth_url = http://192.168.112.99:35357/v2.0
linuxnet_interface_driver =
nova.network.linux_net.LinuxOVSIInterfaceDriver
firewall_driver = nova.virt.firewall.NoopFirewallDriver
security_group_api = neutron
service_neutron_metadata_proxy = true
neutron_metadata_proxy_shared_secret = METADATA_SECRET

```

Konfiguroidaan tiedosto `/etc/sysctl.conf` (46) ja aktivoidaan muutokset (47).

```

46) net.ipv4.conf.all.rp_filter=0
net.ipv4.conf.default.rp_filter=0
net.ipv4.ip_forward=1

```

```

47) sudo sysctl -p

```

Konfiguroidaan tiedosto `/etc/neutron/l3_agent.ini` (48).

```

48) interface_driver = neutron.agent.linux.interface.OVSIInterfaceDriver
use_namespaces = True

```

Konfiguroidaan tiedosto `/etc/neutron/dhcp_agent.ini` (49).

```

49) [DEFAULT]
interface_driver = neutron.agent.linux.interface.OVSIInterfaceDriver
dhcp_driver = neutron.agent.linux.dhcp.Dnsmasq
use_namespaces = True

```

Konfiguroidaan tiedosto /etc/neutron/metadata\_agent.ini (50).

```
50) auth_url = http://192.168.112.99:5000/v2.0
 auth_region = regionOne
 admin_tenant_name = service
 admin_user = neutron
 admin_password = neutronpass
 nova_metadata_ip = 192.168.112.99
 metadata_proxy_shared_secret = METADATA_SECRET
```

Käynnistetään Neutronin palvelut uudelleen (51).

```
51) sudo service nova-api restart && sudo service nova-scheduler restart &&
 sudo service nova-conductor restart && sudo service neutron-server
 restart && sudo service neutron-plugin-openvswitch-agent restart &&
 sudo service neutron-l3-agent restart && sudo service neutron-dhcp-agent
 restart && sudo service neutron-metadata-agent restart
```

Testataan Neutronin toiminto komennolla 52. Komento antaa listan, jossa on ryhmä nimeltä default.

```
52) neutron security-group-list
```

## 5.7 Horizonin asennus

Asennetaan tarvittavat paketit (53).

```
53) sudo apt-get install apache2 memcached libapache2-mod-wsgi openstack-dash-
 board
```

Poistetaan Ubuntun lisäämä ulkoasu, joka ei toimi kunnolla (54).

```
54) sudo apt-get remove openstack-dashboard-ubuntu-theme
```

Muokataan tiedostoon `/etc/memcached.conf` kohdan (55) mukainen rivi.

```
55) -l 192.168.112.99
```

Muokataan tiedostoon `/etc/openstack-dashboard/local_settings.py` kohtiin `LOCATION` ja `OPENSTACK_HOST` ip-osoite `192.168.112.99` (56).

```
56) CACHES = {
 'default': {
 'BACKEND': 'django.core.cache.backends.memcached.MemcachedCache',
 'LOCATION': '192.168.112.99:11211',
 }
 }

 OPENSTACK_HOST = "192.168.112.99"
```

Käynnistetään palvelut uudelleen, jotta muutokset tulevat voimaan (57).

```
57) sudo service apache2 restart && sudo service memcached restart
```

Horizon on nyt käytettävissä selaimella osoitteessa `http://192.168.112.99/horizon/` käyttäjätunnuksella `admin` ja salasanalla `adminpass`.

## 5.8 Verkkojen ja virtuaalikoneiden luonti

Luodaan SSH-avain virtuaalikoneita varten (58) ja lisätään se Novaan (59) sekä varmistetaan, että avaimen lisäys onnistui (60).

```
58) ssh-keygen
```

```
59) nova keypair-add --pub-key /root/.ssh/id_rsa.pub test-key
```

```
60) nova keypair-list
```

Seuraavaksi luodaan ulkoverkko Neutroniin. Tämän verkon on vastattava verkkoa, johon Openstackiä ajava laite on kytketty ja oletusyhdykäytävän on oltava oikea. Ip-alue on määritettävä siten, että siihen kuuluvat ip:t ovat vapaita, koska Openstackin virtuaalikoneet käyttävät tämän alueen ip-osoitteita, kun ne kytkentään ulkoverkkoon. (61 ja 62).

```
61) neutron net-create ext-net --shared --router:external=True
neutron subnet-create ext-net --name ext-subnet --allocation-pool
start=192.168.112.101,end=192.168.112.250 --disable-dhcp --gateway
192.168.112.1 192.168.112.99/24
```

Luodaan virtuaalinen sisäverkko, ip-osoitteen voi valita vapaasti. Esimerkiksi 10-alueen osoitteet ovat hyvä vaihtoehto. (62)

```
62) neutron net-create int-net
neutron subnet-create int-net --name int-subnet --gateway 10.0.0.1
10.0.0.0/24
```

Luodaan virtuaalinen reititin yhdistämään kohtien 61 ja 62 verkot toisiinsa (63). Sen jälkeen liitetään ulkoverkko ja sisäverkko reitittimeen. Ulkoverkko määritetään oletusyhdykäyväksi, jonka kautta virtuaalikoneilla on pääsy internetiin.

```
63) neutron router-create R1
neutron router-interface-add R1 int-subnet
neutron router-gateway-set R1 ext-net
```

Muutetaan turvallisuusasetuksia niin, että virtuaalikoneet voivat vastata ping-kutsuihin ja sallitaan SSH-yhteydet (64).

```
64) nova secgroup-add-rule default icmp -1 -1 0.0.0.0/0
nova secgroup-add-rule default tcp 22 22 0.0.0.0/0
```

Tarkistetaan mikä on sisäverkon id ja otetaan se talteen (65).

```
65) neutron-net list
```

Luodaan ensimmäinen virtuaalikone komennolla 66. Net-id on kohdassa 65 saatu tunnus. Jos levykuvan nimi ei täsmää, oikea nimi löytyy komennolla 28.

```
66) nova boot --flavor=m1.tiny --image=cirros-0.3.2-x86_64 --nic net-id=ID --
security-group=default --key-name=test-key virtuaalikone1
```

Tarkistetaan, että virtuaalikoneen luonti onnistui (67). Tällöin virtuaalikoneen tilan tulee olla active. Käynnistykseen voi mennä muutama minuutti. Jos virtuaalikone jää tilaan build, täytyy Novan tila päivittää (68) ja yrittää käynnistystä sen jälkeen uudelleen.

```
67) nova list
```

```
68) sudo su -s /bin/sh -c "nova-manage db sync" nova
sudo service nova-compute restart
```

Openstackin perusasennus on nyt valmis ja toimintakunnossa. Jatkossa virtuaalikoneita voi luoda, muokata ja hallita graafisen käyttöliittymän kautta selaimessa. Lisätietoa Openstackin käytöstä ja hallinnasta antavat käyttäjän opas (69) ja ylläpitäjän opas (70). Lisäpalveluiden asentamisesta kertoo virallinen asennusohje (71).

```
69) http://docs.openstack.org/user-guide/content/
```

```
70) http://docs.openstack.org/user-guide-admin/content/
```

```
71) http://docs.openstack.org/icehouse/install-guide/install/apt/content/
```

## 6 POHDINTA

Openstackin Icehouse-versio on tarpeeksi kehittynyt ja vakaa sopiakseen tuotantotasois-  
ten pilvipalveluiden perustamiseen. Puolen vuoden käytön aikana Openstackin palvelut  
eivät ole kaatuneet kertaakaan, enkä ole törmännyt yhteenkään vakavaan bugiin.  
Openstackin perustoimintojen asennukseen ja käyttöön on hyvät dokumentaatiot, mutta  
kehittyneempiä ratkaisuja varten on vaikea löytää ajan tasalla olevia ohjeita ja siksi monet  
ratkaisut on etsittävä kokeilemalla. Varsinkin Openstackin verkkopalvelu Neutron on mo-  
nimutkainen ja siihen liittyvien ongelmien ratkominen voi olla erittäin vaikeaa.  
Openstack-ympäristön asennus käsityönä on melko työlästä, joten voi olla hyvä idea au-  
tomatisoida asennusprosessia automaatiotyökaluilla, esimerkiksi Puppetilla, joka tarjoaa  
melko hyvät työkalut Openstackin asennukseen ja konfigurointiin.

Openstackin tietoturva on hyvä, mutta laajan Openstack-pilven kaikkien osien toiminnan  
ymmärtäminen voi olla vaikeaa. Mikäli järjestelmässä on osia, joiden toimintaa ei ym-  
mällä, voi tietoturva vaarantua hyvin helposti, vaikka itse ohjelmistossa ei olisi turva-  
aukkoja. Pilvipalveluihin liittyy uudenlaisia tietoturvauhkia, joiden torjumiseen ei riitä  
pelkästään perinteisten järjestelmien tietoturvan tunteminen.

Openstackin laaja ja aktiivinen kehittäjäyhteisö, kehitystä rahoittavat merkittävät it-alan  
yritykset sekä pilvipalveluiden kysynnän kasvu tekevät Openstackin tulevaisuudesta lu-  
paavan. Yritysten suuri rooli palvelun tukijoina ja kehittäjinä voi kuitenkin olla myös  
uhka, sillä vaikka Openstack on ilmainen, monet jäsenyrityksen myyvät Openstackiin pe-  
rustuvia pilvipalveluita ja virtualisointiratkaisuja. Tästä voi seurata eturistiriita, sillä jois-  
sain tilanteissa yrityksille voi olla kannattavaa jättää ominaisuuksia pois Openstackin  
yleisestä versiosta, jotta oman tuotteen kilpailukyky olisi parempi. Tämä saattaisi johtaa  
Openstack-projektin kehityksen hidastumiseen.

Kirjoitushetkellä Yhteiskuntatieteellinen tietoarkisto on siirtänyt osan sovelluskehitys-  
tään yhdessä palvelimessa toimivaan Openstack-pilveen ja kehityksen siirtoa pilveen on  
tarkoitus jatkaa. Myös osa ylläpidon testiympäristöistä on siirretty samaan Openstack-  
pilveen. Myöhemmin tavoitteena on rakentaa suurempi, useasta laitteesta koostuva  
Openstack-ympäristö ja siirtää sinne Tietoarkiston tuotantopalveluita. Visiona on, että  
vuonna 2015 aikana suuri osa Tietoarkiston palveluista toimisi Openstack-pilvessä.

## LÄHTEET

Bell T. 2014. Our Cloud in Havana. 24.2.2014. Luettu 7.10.2014. <http://openstack-in-production.blogspot.fi/2014/02/our-cloud-in-havana.html>

CNR. 2011. Amazon Cloud Outage Aftermath: Questions, Concerns Linger. 11.5.2011. Luettu 4.10.2014. <http://www.crn.com/news/cloud/229500034/amazon-cloud-outage-aftermath-questions-concerns-linger.htm>

Columbus L. 2014. Roundup Of Cloud Computing Forecasts And Market Estimates, 2014. 14.3.2014. Luettu 21.10.2014. <http://www.forbes.com/sites/louiscolumbus/2014/03/14/roundup-of-cloud-computing-forecasts-and-market-estimates-2014/>

Darrow, B. 2013. "Backbreaking" OpenStack migrations hinder enterprise upgrades. 20.12.2013. Luettu 10.10.2014. <https://gigaom.com/2013/12/20/backbreaking-openstack-migrations-hinder-enterprise-upgrades/>

Day, M. 2012. KVM Myths - Uncovering the Truth about the Open Source Hypervisor. 3.7.2012. Luettu 25.9.2014. [https://www.ibm.com/developerworks/community/blogs/ibmvirtualization/entry/kvm\\_myths\\_uncovering\\_the\\_truth\\_about\\_the\\_open\\_source\\_hypervisor?lang=en](https://www.ibm.com/developerworks/community/blogs/ibmvirtualization/entry/kvm_myths_uncovering_the_truth_about_the_open_source_hypervisor?lang=en)

Fagan, J. 2013. Network-as-a-Service, NaaS: Bridging the gap between the cloud and the network. 23.10.2014. Luettu 2.10.2014. <http://blog.pacnet.com/network-as-a-service-naas-bridging-the-gap-between-the-cloud-and-the-network/>

Germain, J. 2013. How to Avoid Cloud Vendor Lock-In. 13.11.2013. Luettu 7.10.2014. <http://www.linuxinsider.com/story/79417.html>

Hernandez, P. 2013. Survey: 51% of x86 Servers Now Virtualized. 17.1.2014. Luettu 28.9.2014. <http://www.serverwatch.com/server-trends/survey-51-of-x86-servers-now-virtualized.html>

IDC. 2014. Increasing Virtualization Rates in CEMA Approaching Those of WE. Lehdistöiedote. 17.7.2014. Luettu 28.9.2014. <http://www.idc.com/getdoc.jsp?containerId=prCZ24997714>

Jackson, K. 2012. OpenStack Cloud Computing Cookbook. 1. painos. Birmingham: Packt Publishing.

Jackson, K. 2013. OpenStack Cloud Computing Cookbook. 2. painos. Birmingham: Packt Publishing.

Kavis, M. 2014. Architecting the Cloud: Design Decisions for Cloud Computing Service Models (SaaS, PaaS, and IaaS). Somerset: Wiley.

Kleyman, B. 2012. Hypervisor 101: Understanding the Virtualization Market. 1.9.2012. Luettu 25.9.2014. <http://www.datacenterknowledge.com/archives/2012/08/01/hypervisor-101-a-look-hypervisor-market/>

- Kuntaliitto. 2013. Tietosuoja ja pilvipalvelut henkilötietolain näkökulmasta. Muistio. 4.7.2013. Luettu 4.10.2014.  
[http://www.kunnat.net/fi/asiantuntijapalvelut/laki/hallintojuridiikka/julkisuus\\_tietosuoja/tietosuoja-pilvipalvelut/Documents/Tietosuoja%20ja%20pilvipalvelut%20henkil%C3%B6tietolain%20n%C3%A4k%C3%B6kulmasta\\_Muistio.pdf](http://www.kunnat.net/fi/asiantuntijapalvelut/laki/hallintojuridiikka/julkisuus_tietosuoja/tietosuoja-pilvipalvelut/Documents/Tietosuoja%20ja%20pilvipalvelut%20henkil%C3%B6tietolain%20n%C3%A4k%C3%B6kulmasta_Muistio.pdf)
- Mell, P & Grance, T. 2011. The NIST Definition of Cloud Computing. National Institute of Standards and Technology, Yhdysvallat. Luettu 3.10.2014.  
<http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf>
- Openstack-säätiö. 2014a. Companies Supporting the OpenStack Foundation. Luettu 30.9.2014 <http://www.openstack.org/foundation/companies/>
- Openstack-säätiö. 2014b. Hypervisor Support Matrix. Luettu 13.10.2014.  
<https://wiki.openstack.org/wiki/HypervisorSupportMatrix>
- Openstack-säätiö. 2014c. OpenStack Installation Guide for Ubuntu 12.04/14.04 (LTS) (Icehouse). Päivitetty 14.9.2014. Luettu 28.9.2014.  
<http://docs.openstack.org/icehouse/install-guide/install/apt/openstack-install-guide-apt-icehouse.pdf>
- Openstack-säätiö. 2014d. OpenStack: The Open Source Cloud Operating System. Luettu 28.9.2014. <https://www.openstack.org/software/>
- Openstack-säätiö. 2014e. Releases. Luettu 7.10.2014.  
<https://wiki.openstack.org/wiki/Releases>
- Openstack-säätiö. 2014f. Tenant and provider networks. 15.10.2014. Luettu 20.10.2014.  
<http://docs.openstack.org/admin-guide-cloud/content/tenant-provider-networks.html>
- Openstack-säätiö. 2014g. The OpenStack Foundation. Luettu 30.9.2014.  
<http://www.openstack.org/foundation/>
- Openstack-säätiö. 2014h. User Stories. Luettu 3.11.2014.  
<http://www.openstack.org/user-stories/>
- Page, J. 2014. How we scaled OpenStack to launch 168,000 cloud instances. 18.6.2014. Luettu 1.10.2014. <http://javacruft.wordpress.com/2014/06/18/168k-instances/>
- Portnoy, M. 2012. Virtualization essentials. Indianapolis: John Wiley & Sons, Inc.
- Shackleford, D. 2012. Virtualization Security: Protecting Virtualized Environments. Indianapolis: John Wiley & Sons, Inc.
- Ubuntu. 2014. Openstack - Why build your cloud on Ubuntu. Luettu 7.10.2014.  
<http://www.ubuntu.com/cloud/openstack>
- Yegulalp S. 2014. User survey: Ubuntu is the top OpenStack OS. 19.5.2014. Luettu 7.11.2014. <http://www.infoworld.com/article/2608163/openstack/user-survey--ubuntu-is-the-top-openstack-os.html>



Zheng, M. 2011. Virtualization Security in Data Centers and Clouds. 27.11.2011. Lu-  
ettu 26.9.2014. <http://www.cse.wustl.edu/~jain/cse571-11/ftp/virtual/>

## LIITTEET

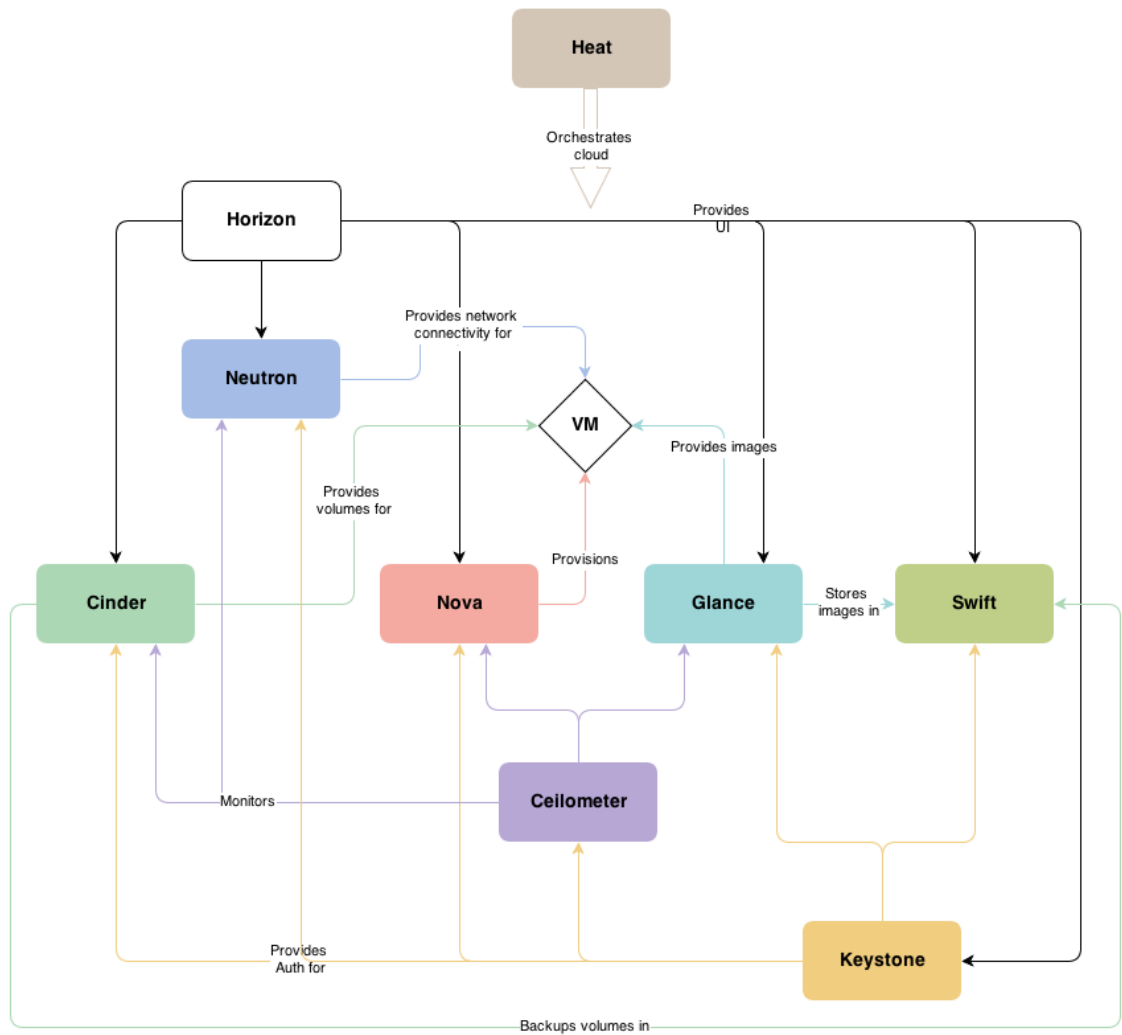
### Liite 1. Openstackin palvelut ja niiden tehtävät

Openstackin palvelut (Openstack-säätiö 2014c, 1)

| <b>OpenStack services</b>    |                     |                                                                                                                                                                                                                                                                             |
|------------------------------|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Service</b>               | <b>Project name</b> | <b>Description</b>                                                                                                                                                                                                                                                          |
| Dashboard                    | Horizon             | Provides a web-based self-service portal to interact with underlying OpenStack services, such as launching an instance, assigning IP addresses and configuring access controls.                                                                                             |
| Compute                      | Nova                | Manages the lifecycle of compute instances in an OpenStack environment. Responsibilities include spawning, scheduling and decommissioning of virtual machines on demand.                                                                                                    |
| Networking                   | Neutron             | Enables network connectivity as a service for other OpenStack services, such as OpenStack Compute. Provides an API for users to define networks and the attachments into them. Has a pluggable architecture that supports many popular networking vendors and technologies. |
| <b>Storage</b>               |                     |                                                                                                                                                                                                                                                                             |
| Object Storage               | Swift               | Stores and retrieves arbitrary unstructured data objects via a RESTful, HTTP based API. It is highly fault tolerant with its data replication and scale out architecture. Its implementation is not like a file server with mountable directories.                          |
| Block Storage                | Cinder              | Provides persistent block storage to running instances. Its pluggable driver architecture facilitates the creation and management of block storage devices.                                                                                                                 |
| <b>Shared services</b>       |                     |                                                                                                                                                                                                                                                                             |
| Identity service             | Keystone            | Provides an authentication and authorization service for other OpenStack services. Provides a catalog of endpoints for all OpenStack services.                                                                                                                              |
| Image Service                | Glance              | Stores and retrieves virtual machine disk images. OpenStack Compute makes use of this during instance provisioning.                                                                                                                                                         |
| Telemetry                    | Ceilometer          | Monitors and meters the OpenStack cloud for billing, benchmarking, scalability, and statistical purposes.                                                                                                                                                                   |
| <b>Higher-level services</b> |                     |                                                                                                                                                                                                                                                                             |
| Orchestration                | Heat                | Orchestrates multiple composite cloud applications by using either the native HOT template format or the AWS CloudFormation template format, through both an OpenStack-native REST API and a CloudFormation-compatible Query API.                                           |
| Database Service             | Trove               | Provides scalable and reliable Cloud Database-as-a-Service functionality for both relational and non-relational database engines.                                                                                                                                           |

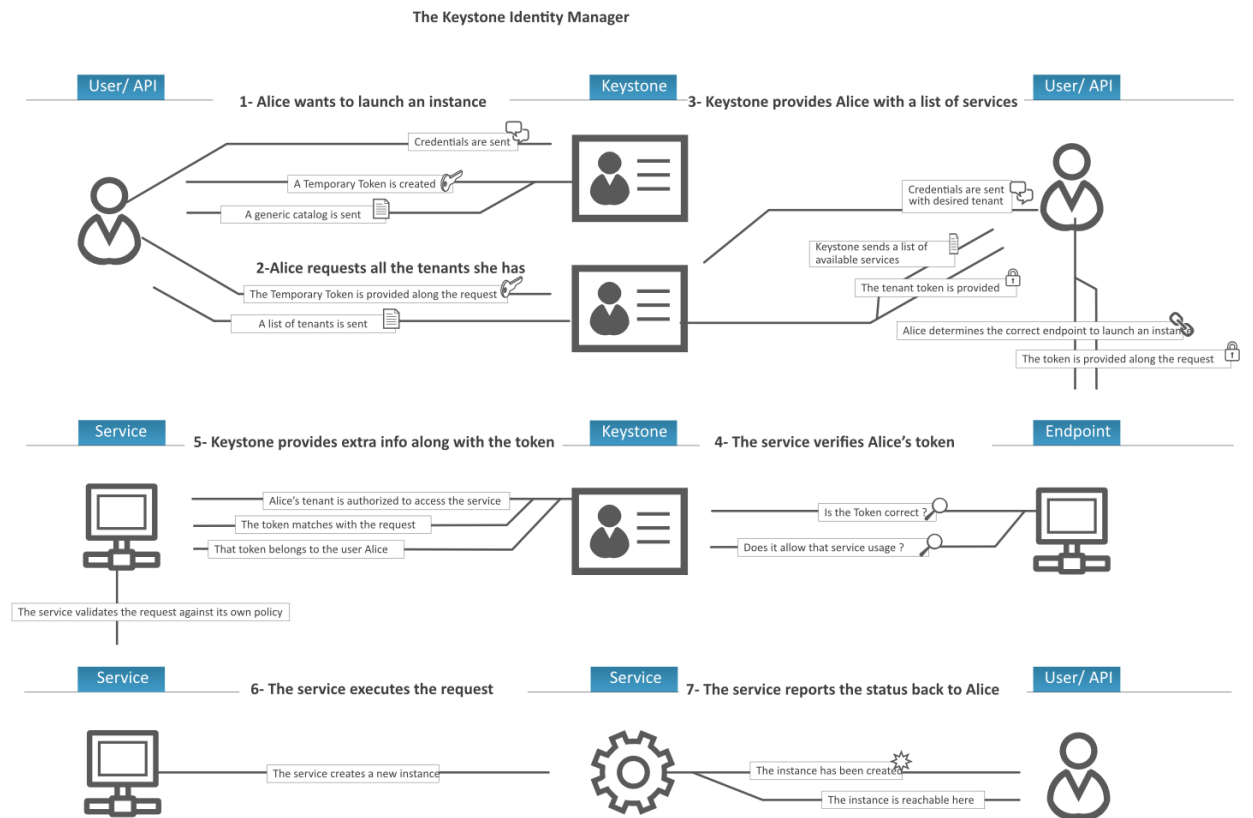
## Liite 2. Openstackin palveluiden yhteydet

### Openstackin palveluiden väliset toiminnot (Openstack-säätiö 2014c, 2) )



## Liite 3. Keystonein toiminta

## Keystonein toiminta virtuaalikonetta luotaessa (Openstack-säätiö 2014c, 23)



## Liite 4. MySQL:n konfiguraatio

1 (2)

Tästä ja seuraavista liitteistä on luettavuuden parantamiseksi jätetty pois kommentit.

**/etc/mysql/my.cnf**

[client]

port = 3306

socket = /var/run/mysqld/mysqld.sock

[mysqld\_safe]

socket = /var/run/mysqld/mysqld.sock

nice = 0

[mysqld]

user = mysql

pid-file = /var/run/mysqld/mysqld.pid

socket = /var/run/mysqld/mysqld.sock

port = 3306

basedir = /usr

datadir = /var/lib/mysql

tmpdir = /tmp

lc-messages-dir = /usr/share/mysql

skip-external-locking

bind-address = 192.168.112.99

default-storage-engine = innodb

innodb\_file\_per\_table

collation-server = utf8\_general\_ci

init-connect = 'SET NAMES utf8'

character-set-server = utf8

key\_buffer = 16M

max\_allowed\_packet = 16M

thread\_stack = 192K

thread\_cache\_size = 8

myisam-recover = BACKUP

```
query_cache_limit = 1M
query_cache_size = 16M
log_error = /var/log/mysql/error.log
expire_logs_days = 10
max_binlog_size = 100M
```

```
[mysqldump]
```

```
quick
```

```
quote-names
```

```
max_allowed_packet = 16M
```

```
[mysql]
```

```
[isamchk]
```

```
key_buffer = 16M
```

```
!includedir /etc/mysql/conf.d/
```

## Liite 5. Keystoneen konfiguraatio

**/etc/keystone.keystone.conf**

[DEFAULT]

admin\_token=temptoken

rabbit\_password=rabbitpass

log\_dir=/var/log/keystone

[assignment]

[auth]

[cache]

[catalog]

[credential]

[database]

connection = mysql://keystone:keystonepass@192.168.112.99/keystone

[ec2]

[endpoint\_filter]

[federation]

[identity]

[kvs]

[ldap]

[matchmaker\_ring]

[memcache]

[oauth1]

[os\_inherit]

[paste\_deploy]

[policy]

[revoke]

[signing]

[ssl]

[stats]

[token]

[trust]

[extra\_headers]

Distribution = Ubuntu

## Liite 6. Glancen konfiguraatiot

1 (3)

**/etc/glance/glance-registry.conf**

[DEFAULT]

rpc\_backend = rabbit

rabbit\_host = 192.168.112.99

rabbit\_password = rabbitpass

bind\_host = 0.0.0.0

bind\_port = 9191

log\_file = /var/log/glance/registry.log

backlog = 4096

api\_limit\_max = 1000

limit\_param\_default = 25

[database]

connection = mysql://glance:glancepass@192.168.112.99/glance

sqlite\_db = /var/lib/glance/glance.sqlite

backend = sqlalchemy

[keystone\_authtoken]

auth\_uri = http://192.168.112.99:5000

auth\_host = 192.168.112.99

auth\_port = 35357

auth\_protocol = http

admin\_tenant\_name = service

admin\_user = glance

admin\_password = glancepass

[paste\_deploy]

flavor=keystone



**/etc/glance/glance-api.conf**

```
[DEFAULT]
rpc_backend = rabbit
rabbit_host = 192.168.112.99
rabbit_password = rabbitpass
default_store = file
bind_host = 0.0.0.0
bind_port = 9292
log_file = /var/log/glance/api.log
backlog = 4096
workers = 1
registry_host = 0.0.0.0
registry_port = 9191
registry_client_protocol = http
rabbit_host = localhost
rabbit_port = 5672
rabbit_use_ssl = false
rabbit_userid = guest
rabbit_password = guest
rabbit_virtual_host = /
rabbit_notification_exchange = glance
rabbit_notification_topic = notifications
rabbit_durable_queues = False
qpid_notification_exchange = glance
qpid_notification_topic = notifications
qpid_hostname = localhost
qpid_port = 5672
qpid_username =
qpid_password =
qpid_sasl_mechanisms =
qpid_reconnect_timeout = 0
qpid_reconnect_limit = 0
qpid_reconnect_interval_min = 0
qpid_reconnect_interval_max = 0
qpid_reconnect_interval = 0
qpid_heartbeat = 5
qpid_protocol = tcp
qpid_tcp_nodelay = True
filesystem_store_datadir = /var/lib/glance/images/
swift_store_auth_version = 2
swift_store_auth_address = 127.0.0.1:5000/v2.0/
```

```
swift_store_user = jdoe:jdoe
swift_store_key = a86850deb2742ec3cb41518e26aa2d89
swift_store_container = glance
swift_store_create_container_on_put = False
swift_store_large_object_size = 5120
swift_store_large_object_chunk_size = 200
swift_enable_snet = False
s3_store_host = 127.0.0.1:8080/v1.0/
s3_store_access_key = <20-char AWS access key>
s3_store_secret_key = <40-char AWS secret key>
s3_store_bucket = <lowercased 20-char aws access key>glance
s3_store_create_bucket_on_put = False
sheepdog_store_address = localhost
sheepdog_store_port = 7000
sheepdog_store_chunk_size = 64
delayed_delete = False
scrub_time = 43200
scrubber_datadir = /var/lib/glance/scrubber
image_cache_dir = /var/lib/glance/image-cache/
```

[database]

```
connection = mysql://glance:glancepass@192.168.112.99/glance
sqlite_db = /var/lib/glance/glance.sqlite
backend = sqlalchemy
```

[keystone\_authtoken]

```
auth_uri = http://192.168.112.99:5000
auth_host = 192.168.112.99
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = glance
admin_password = glancepass
```

[paste\_deploy]

```
flavor=keystone
```

## Liite 7. Novan konfiguraatio

1 (2)

**/etc/nova.nova.conf**

[DEFAULT]

dhcpbridge\_flagfile=/etc/nova/nova.conf

dhcpbridge=/usr/bin/nova-dhcpbridge

logdir=/var/log/nova

state\_path=/var/lib/nova

lock\_path=/var/lock/nova

force\_dhcp\_release=True

iscsi\_helper=tgtadm

libvirt\_use\_virtio\_for\_bridges=True

connection\_type=libvirt

root\_helper=sudo nova-rootwrap /etc/nova/rootwrap.conf

verbose=True

ec2\_private\_dns\_show\_ip=True

api\_paste\_config=/etc/nova/api-paste.ini

volumes\_path=/var/lib/nova/volumes

enabled\_apis=ec2,osapi\_compute,metadata

rpc\_backend = rabbit

rabbit\_host = 192.168.112.99

rabbit\_password = rabbitpass

my\_ip = 192.168.112.99

vncserver\_listen = 192.168.112.99

vncserver\_proxyclient\_address = 192.168.112.99

vnc\_enabled=True

novncserver\_listen=0.0.0.0

novncproxy\_base\_url=http://192.168.112.99:6080/vnc\_auto.html

glance\_host = 192.168.112.99

auth\_strategy = keystone

network\_api\_class = nova.network.neutronv2.api.API

neutron\_url = http://192.168.112.99:9696

neutron\_auth\_strategy = keystone

neutron\_admin\_tenant\_name = service

```
neutron_admin_username = neutron
neutron_admin_password = neutronpass
neutron_admin_auth_url = http://192.168.112.99:35357/v2.0
linuxnet_interface_driver = nova.network.linux_net.LinuxOVSIInterfaceDriver
firewall_driver = nova.virt.firewall.NoopFirewallDriver
security_group_api = neutron
service_neutron_metadata_proxy = true
neutron_metadata_proxy_shared_secret = METADATA_SECRET

[database]
connection = mysql://nova:novapass@192.168.112.99/nova

[keystone_authtoken]
auth_uri = http://192.168.112.99:5000
auth_host = 192.168.112.99
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = nova
admin_password = novapass
```

## Liite 8. Neutronin konfiguraatiot

1 (3)

**/etc/neutron/neutron.conf**

## [DEFAULT]

```
rpc_backend = neutron.openstack.common.rpc.impl_kombu
rabbit_host = 192.168.112.99
rabbit_password = rabbitpass
notify_nova_on_port_status_changes = True
notify_nova_on_port_data_changes = True
nova_url = http://192.168.112.99:8774/v2
nova_admin_username = nova
nova_admin_tenant_id = c9209ba77dae44dfbf60d526b9031b18
nova_admin_password = novapass
nova_admin_auth_url = http://192.168.112.99:35357/v2.0
core_plugin = ml2
service_plugins = router
allow_overlapping_ips = True
auth_strategy = keystone
verbose = True
state_path = /var/lib/neutron
lock_path = $state_path/lock
core_plugin = neutron.plugins.ml2.plugin.Ml2Plugin
notification_driver = neutron.openstack.common.notifier.rpc_notifier
[quotas]
```

## [agent]

```
root_helper = sudo /usr/bin/neutron-rootwrap /etc/neutron/rootwrap.conf
```

## [keystone\_authtoken]

```
auth_uri = http://192.168.112.99:5000
auth_host = 192.168.112.99
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
```

```
admin_user = neutron
admin_password = neutronpass
[database]
connection = mysql://neutron:neutronpass@192.168.112.99/neutron
```

```
[service_providers]
service_provider=LOADBALANCER:Haproxy:neutron.services.loadbalancer.drivers.h
aproxy.plugin_driver.HaproxyOnHostPluginDriver:default
service_provider=VPN:openswan:neutron.services.vpn.service_drivers.ipsec.IPsecVPN
Driver:default
```

### **etc/neutron/plugins/ml2/ml2\_conf.ini**

```
type_drivers = gre
tenant_network_types = gre
mechanism_drivers = openvswitch
[ml2_type_flat]
[ml2_type_vlan]
[ml2_type_gre]
tunnel_id_ranges = 1:1000
[ml2_type_vxlan]
[securitygroup]
firewall_driver =
neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver
enable_security_group = True
[ovs]
local_ip = 192.168.112.99
tunnel_type = gre
enable_tunneling = True
```

### **/etc/sysctl.conf**

```
net.ipv4.conf.all.rp_filter=0
net.ipv4.conf.default.rp_filter=0
net.ipv4.ip_forward=1
```

**/etc/neutron/l3\_agent.ini**

[DEFAULT]

interface\_driver = neutron.agent.linux.interface.OVSInterfaceDriver

use\_namespaces = True

**/etc/neutron/dhcp\_agent.ini**

[DEFAULT]

interface\_driver = neutron.agent.linux.interface.OVSInterfaceDriver

dhcp\_driver = neutron.agent.linux.dhcp.Dnsmasq

use\_namespaces = True

**/etc/neutron/metadata\_agent.ini**

[DEFAULT]

auth\_url = http://192.168.112.99:5000/v2.0

auth\_region = regionOne

admin\_tenant\_name = service

admin\_user = neutron

admin\_password = neutronpass

nova\_metadata\_ip = 192.168.112.99

metadata\_proxy\_shared\_secret = METADATA\_SECRET

auth\_url = http://localhost:5000/v2.0

auth\_region = RegionOne

admin\_tenant\_name = %SERVICE\_TENANT\_NAME%

admin\_user = %SERVICE\_USER%

admin\_password = %SERVICE\_PASSWORD%

## Liite 9. Horizonin konfiguraatiot

1 (2)

**/etc/memcached.conf**

```
-d
logfile /var/log/memcached.log
-m 64
-p 11211
-u memcache
-l 192.168.112.99
```

**/etc/openstack-dashboard/local\_setting.py**

```
import os
from django.utils.translation import ugettext_lazy as _
from openstack_dashboard import exceptions
DEBUG = False
TEMPLATE_DEBUG = DEBUG
HORIZON_CONFIG = {
 'dashboards': ('project', 'admin', 'settings'),
 'default_dashboard': 'project',
 'user_home': 'openstack_dashboard.views.get_user_home',
 'ajax_queue_limit': 10,
 'auto_fade_alerts': {
 'delay': 3000,
 'fade_duration': 1500,
 'types': ['alert-success', 'alert-info']
 },
 'help_url': "http://docs.openstack.org",
 'exceptions': {'recoverable': exceptions.RECOVERABLE,
 'not_found': exceptions.NOT_FOUND,
 'unauthorized': exceptions.UNAUTHORIZED},
}
LOCAL_PATH = os.path.dirname(os.path.abspath(__file__))
from horizon.utils import secret_key
```



2 (2)

```
SECRET_KEY = secret_key.generate_or_read_from_file('/var/lib/openstack-
dashboard/secret_key')
 CACHES = {
 'default': {
 'BACKEND' : 'django.core.cache.backends.memcached.MemcachedCache',
 'LOCATION' : '192.168.112.99:11211',
 }
 }
try:
 from ubuntu_theme import *
except ImportError:
 pass
LOGIN_URL='/horizon/auth/login/'
LOGOUT_URL='/horizon/auth/logout/'
LOGIN_REDIRECT_URL='/horizon'
COMPRESS_OFFLINE = True
ALLOWED_HOSTS = '*'
EMAIL_BACKEND = 'django.core.mail.backends.console.EmailBackend'
OPENSTACK_HOST = "192.168.112.99"

```

(Tiedoston loppu on jätetty pois, koska siinä on satoja rivejä koodia, johon ei tarvitse tehdä muutoksia.)