UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Amir Huskić

Možnosti zagotavljanja povezljivosti navideznih in resničnih računalnikov

MAGISTRSKO DELO

PODIPLOMSKI ŠTUDIJSKI PROGRAM INFORMACIJSKI SISTEMI IN ODLOČANJE

MENTORICA: doc. dr. Mojca Ciglarič

Ljubljana, 2016

Univerza *v Ljubljani*
Fakulteta *za računalništvo in informatiko*

Večna pot 113
1000 Ljubljana, Slovenija
telefon: 01 47 98 100
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Številka: 162-MAG-ISO/2016
Datum: 06. 04. 2016

Amir HUSKIĆ, univ. dipl. soc. del.

Ljubljana

Fakulteta za računalništvo in informatiko Univerze v Ljubljani izdaja naslednjo magistrsko nalogo

Naslov naloge: **Možnosti zagotavljanja povezljivosti navideznih in resničnih računalnikov**

Options for providing connectivity of virtual and real computers

Tematika naloge:

Virtualizacija je omogočila razvoj računalništva v oblaku, ki je v zadnjih letih med najmočnejšimi trendi na področju informacijskih tehnologij. Zaradi številnih prednosti virtualizacije in računalništva v oblaku uporabniki virtualizirajo vedno več obstoječe informacijske infrastrukture. Obstajajo pa primeri, ko tega ni možno narediti in takrat je pogosto potrebno zagotoviti vsaj omrežno povezljivost oblačne infrastrukture s fizično infrastrukturo na način, kot da se obe nahajata v istem krajevnem omrežju.
V magistrski nalogi naredite pregled in analizo področja povezovanja virtualnih računalnikov v oblaku s fizičnimi računalniki. Preglejte tudi obstoječe tehnologije skupaj z morebitnimi rešitvami na trgu in njihovim namenom. Identificirajte različne arhitekturne modele ter predlagajte nadaljnje možne arhitekture varnega povezovanja oblačne infrastrukture, predvsem njenih virtualiziranih komponent, s fizično informacijsko infrastrukturo. Podajte pregled mehanizmov za izolacijo uporabnikov v oblaku na 2. in 3. plasti komunikacijskega sklada ter pregled mehanizmov za izolacijo in varnost omrežnega prometa z uporabo tunelskih protokolov. Raziščite možne postopke orkestracije in avtomatizacije postavljanja takšnih arhitektur. Identificirajte omejitve in pomanjkljivosti različnih obstoječih modelov in arhitektur ter kritično ovrednotite produkcijsko zrelost možnih rešitev.

Mentorica:

doc. dr. Mojca Ciglarič

Dekan:

prof. dr. Nikolaj Zimic

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Zahvaljujem se mentorici doc. dr. Mojci Ciglarič za nasvete, odzivnost in strokovno vodenje pri nastajanju magistrske naloge.

Zahvaljujem se staršem, prijateljem ter vsem ostalim, ki so na različne načine nesebično prispevali in pomagali.

Posebna zahvala življenjski sopotnici Tanji ter otrokoma Siji in Maku za vso podporo in razumevanje.

KAZALO

Seznam uporabljenih kratic

kratica	angleško	slovensko
IaaS	Infrastructure as a service	Infrastruktura kot storitev
PaaS	Platform as a service	Platforma kot storitev
SaaS	Software as a service	Programska oprema kot storitev
API	Application program interface	Programski vmesnik
SDN	Software defined networking	Programsko določena omrežja
NV	Network virtualization	Omrežna virtualizacija
NFV	Network function virtualization	Virtualizacija omrežnih funkcij
ACL	Access control list	Seznam za kontrolo dostopa
VLAN	Virtual local area network	Virtualna lokalna omrežja
KVM	Kernel virtualized machines	KVM hipervizor
Xen	Xen hypervisor	Xen hipervizor
QEMU	Quick emulator	Hitri posnemovalnik
LXC	Linux containers	Linux vsebniki
VXLAN	Virtual extensible local area network	Navidezno razširljivo lokalno omrežje
VNI	Virtual network identifier	Navidezni omrežni identifikator
OVS	Open virtual switch	Odprto navidezno stikalo
L2	Layer 2	Druga plast komunikacijskega sklada
L3	Layer 3	Tretja plast komunikacijskega sklada
VPC	Virtual private cloud	Virtualni zasebni oblak
REST	Representational state transfer	Predstavitveni prenos stanj
DHCP	Dynamic host configuration protocol	Protokol za dinamično dodeljevanje IP naslovov v omrežju
GRE	Generic routing encapsulation	Inkapsulacija pri generičnem usmerjanju
LVM	Logical volume manager	Upravljalnik logičnih particij
BYOD	Bring your own device	Prinesi svojo napravo
LLDP	Link layer discovery protocol	Protokol za odkrivanje na povezovalnem sloju
STP	Spanning tree protocol	Protokol vpetega drevesa
ICMP	Internet control message protocol	Protokol internetnega krmilnega sporočila
JSON	JavaScript object notation	Objektni JavaScript zapis
VRF	Virtual routing and forwarding	Virtualno usmerjanje in posredovanje
VTEP	Virtual tunnel endpoint	Virtualna tunelska končna točka
NVGRE	Network virtualization using generic routing encapsulation	Navidezno omrežje z uporabo generične enkapsulacije
STT	Stateles transport tunneling	Transportno tuneliranje brez stanja
LSO	Large segment offload	Mehanizem za povečanje pretoka omrežnega izhodnega prometa
LRO	Large receive offload	Mehanizem za povečanje pretoka omrežnega

		vhodnega prometa
GENEVE	Generic network virtualization encapsulation	Generično navidezno omrežje z enkapsulacijo
IPSsec	Internet protocol security	Protokol za varovanje IP prometa
SSL	Secure socket layer	Sloj varnih vtičnic
VPN	Virtual private network	Navidezno zasebno omrežje
DPI	Deep packet inspection	Temeljit pregled paketa
QoS	Quality of service	Kakovost storitve
SLA	Service level agreement	Sporazum o nivoju storitve
NGN	Next generation network	Omrežje naslednje generacije
SBC	Session border controller	Robni nadzornik sej
IMS	IP multimedia subsystem	IP multimedijski podsistem
ETSI	European telecommunication standard institute	Evropski inštitut za telekomunikacije
DSLAM	Digital subscriber line access multiplexer	Dostopovni multipleksor digitalne naročniške linije
OPEX	Operating expense	Stroški poslovanja
CAPEX	Capital expense	Naložbe v osnovna sredstva
VNF	Virtual network functions	Virtualne omrežne funkcije
WAN	Wide area network	Prostrano omrežje
vPE	Virtual edge routers	Virtualni robni usmerjevalniki
BSS/OSS	Business and operation support systems	Poslovni in operacijski podporni sistemi
TCO	Total cost of ownership	Celotni stroški lastništva
VIM	Virtualization infrastructure manager	Virtualni infrastrukturni upravljalnik
MANO	Management and orchestration	Upravljanje in orkestracija
VRRP	Virtual router redundancy protocol	Virtualni usmerjevalni redundantni protokol
NAT	Network address translation	Prevajanje omrežnih naslovov
SNAT	Source NAT	Prevajanje izvornih omrežnih naslovov
DNAT	Destination NAT	Prevajanje ponornih omrežnih naslovov
SSH	Secure shell	Varnostna lupina

Povzetek

Virtualizacija je omogočila razvoj računalništva v oblaku, ki je v zadnjih letih med najmočnejšimi trendi na področju informacijskih tehnologij. Zaradi številnih prednosti virtualizacije in računalništva v oblaku uporabniki virtualizirajo vedno več obstoječe informacijske infrastrukture. Obstajajo pa primeri, ko tega ni možno narediti, in takrat je pogosto treba zagotoviti vsaj omrežno povezljivost oblačne infrastrukture s fizično infrastrukturo na način, kot da se obe nahajata v istem krajevnem omrežju.

V magistrski nalogi smo najprej naredili pregled področja ter obstoječih tehnologij: virtualizacija, SDN, NV, NFV in odprtokodne oblačne platforme. Naredili smo pregled mehanizmov za izolacijo uporabnikov na 2. in 3. plasti komunikacijskega sklada z uporabo virtualnih omrežij, programskih virtualnih stikal, različnih programskih vtičnikov in tunelskih protokolov. Raziskali smo različne modele in arhitekture varnega povezovanja oblačne infrastrukture, predvsem njenih virtualiziranih komponent s fizično informacijsko infrastrukturo. Na praktičnih primerih smo preverili, na kakšen način se lahko omogoči povezovanje in komunikacija med virtualnimi računalniki in fizično infrastrukturo ter raziskali možnosti orkestracije in avtomatizacije postopkov postavljanja takšnih arhitektur. Pri delu smo bili osredotočeni na odprtokodno oblačno platformo OpenStack, ki je trenutno najbolj razširjena. Na koncu smo identificirali omejitve in pomanjkljivosti različnih obstoječih modelov in arhitektur ter ovrednotili produkcijsko zrelost posameznih možnih rešitev povezovanja fizične in virtualne infrastrukture.

Ključne besede: virtualizacija, SDN, NV, NFV, OpenStack, Neutron

Abstract

Virtualization has enabled the development of cloud computing, which during the last couple of years, has increasingly become one of the most important Information Technology trends. With the numerous advantages afforded by virtualization and cloud computing, users can virtualize an increasing percentage of the existing IT infrastructure. However, there are cases where this cannot be done. Should this happen, it is often imperative to provide at least network connectivity for cloud infrastructures with the physical infrastructure in such a way as to guarantee that they are both located in the same local area network at these times.

The Master's Thesis first examines the field itself and the matter of existing technologies: virtualization, SDN, NV, NFV, as well as open-source cloud platforms. We have drawn up an overview of the user isolating mechanisms used for isolating the Layer 2 and Layer 3 networks through the use of virtual networks, software virtual switches, various software plugins and tunnelling protocols. We looked at various models and architectures which would enable a safe integration with the cloud infrastructure, especially its virtualized component, with physical IT infrastructure. We used practical examples to ascertain how such interfaces and communication can be allowed for networking and communication between virtual computers and the physical infrastructure, also looking into the orchestration and automation of the procedures employed in setting up such architectures. Our work mainly centred on the most frequently used open-source cloud platform, i.e. the OpenStack. In the end, we identified the limitations and shortcomings of various existing models and architecture, and evaluated the production maturity of individual possible solutions for linking the physical and virtual infrastructure.

Key Words: virtualization, SDN, NV, NFV, OpenStack, Neutron

1 Uvod

Programsko opremo, ki jo uporabljamo za simulacijo strojne platforme, imenujemo virtualizacija. Virtualizacija deluje na enak način kot strojna oprema, le da se simulira s pomočjo virtualnih instanc, ki pa so v celoti ločene od fizične opreme. Zasnova virtualizacije na centralnih računalnikih sega v šestdeseta leta 20. stoletja. IBM je bil v tistem času vodilni pri razvoju rešitev, ki omogočajo razporejanje virov večjemu številu odjemalcev v istem času. Posameznikom in podjetjem zato ni bilo treba kupovati računalnikov, da bi jih lahko uporabljali [19]. Prav s tem razlogom je tudi dandanes uporaba virtualizacije v velikem porastu. Procesorska arhitektura x86 v preteklosti ni imela podpore za strojno virtualizacijo [64]. Dodana je bila leta 2006 (Intel VT-x, AMD-SVM), kar je skupaj s programskimi hipernadzorniki, kot so komercialni VMware Workstation in ESXi ter Microsoft Hyper-V ali odprtokodni KVM, Xen in Oracle VirtualBox, omogočilo širši dostop do virtualizacije podjetjem in posameznikom [62, 63, 65]. Virtualizacija je tako omogočila razvoj računalništva v oblaku, ki je zaradi svojih lastnosti, zlasti zmožnosti ponujati prilagodljivo in dinamično informacijsko infrastrukturo, programsko okolje in storitve, že leta 2007 prehitelo mrežno računalništvo in razpršeno računanje (*ang. grid computing*) [67].

Model računalništva, ki omogoča relativno enostaven dostop do omrežja in deljenih računalniških virov, imenujemo računalništvo v oblaku. Dostopi do omrežij, strežnikov, podatkovnih hramb in aplikacij so na voljo preko interneta ne glede na lokacijo uporabnika. Deljeni računalniški viri so zaradi lastnosti dinamičnega prilagajanja učinkovito in hitro rezervirani, sproščeni in posredovani s strani ponudnika storitev h končnemu uporabniku [45]. Zaradi prakse večuporabniškega okolja in večnajemniškega modela je poleg standardnih varnostnih mehanizmov in sistemov zaščite v oblaku treba zagotoviti tudi različne mehanizme izolacije na različnih nivojih omrežnega dostopa [66]. Problem je, da tradicionalne omrežne tehnologije niso bile zasnovane za današnje uporabnike oblačnih storitev [2]. Navedene pomanjkljivosti poskušajo odpraviti [5, 24, 25, 26, 31] novi omrežni koncepti, kot so programsko določena omrežja (*ang. Software Defined Networking*), omrežna virtualizacija (*ang. Network Virtualization*) ter virtualizacija omrežnih funkcij (*ang. Network Function Virtualization*) skupaj z oblačnimi platformami.

Zaradi številnih prednosti virtualizacije (fleksibilnost, upravljanje, avtomatizacija itd.) poskušajo uporabniki virtualizirati čimveč obstoječe informacijske infrastrukture. Vendar pa obstaja nekaj primerov, kjer popolna virtualizacija ni mogoča. Nekaterih funkcij in storitev zaradi tehničnih omejitev ali potreb ni možno virtualizirati, na primer:

- zastarelih (*ang. legacy*) aplikacij in sistemov,
- specializirane namenske strojne opreme,

- uporabe obstoječe omrežne opreme (požarne pregrade, usmerjevalniki) pri fizični in virtualizirani informacijski infrastrukturi,
- povezovanja fizičnih in virtualnih podatkovnih centrov.

Vse zgoraj naštete ovire trenutno predstavljajo velik izziv številnim odprtokodnim skupnostim (*ang. Open Source Communities*) in tradicionalnim podjetjem za razvoj omrežne opreme. Obenem pa se pojavljajo nova podjetja, ki v teh izzivih vidijo nove poslovne priložnosti.

V takih primerih je treba zagotoviti omrežno povezljivost virtualizirane oblačne infrastrukture z obstoječo fizično infrastrukturo in informacijskimi sistemi.

2 Oblak

Računalništvo v oblaku je model, ki ne glede na lokacijo uporabnika storitev omogoča relativno enostaven dostop na zahtevo do omrežja in deljenih računalniških virov (npr. omrežij, strežnikov, hrambe podatkov, aplikacij in storitev). Z minimalnim naporom upravljanja ali posredovanja ponudnika storitev so deljeni računalniški viri lahko hitro rezervirani in sproščeni. Osnovne lastnosti tega modela so:

- Samostojen dostop na zahtevo do storitev; uporabnik storitev lahko sam rezervira in upravlja računalniške vire. Pri tem ni potrebna človeška interakcija z vsakim ponudnikom storitev.

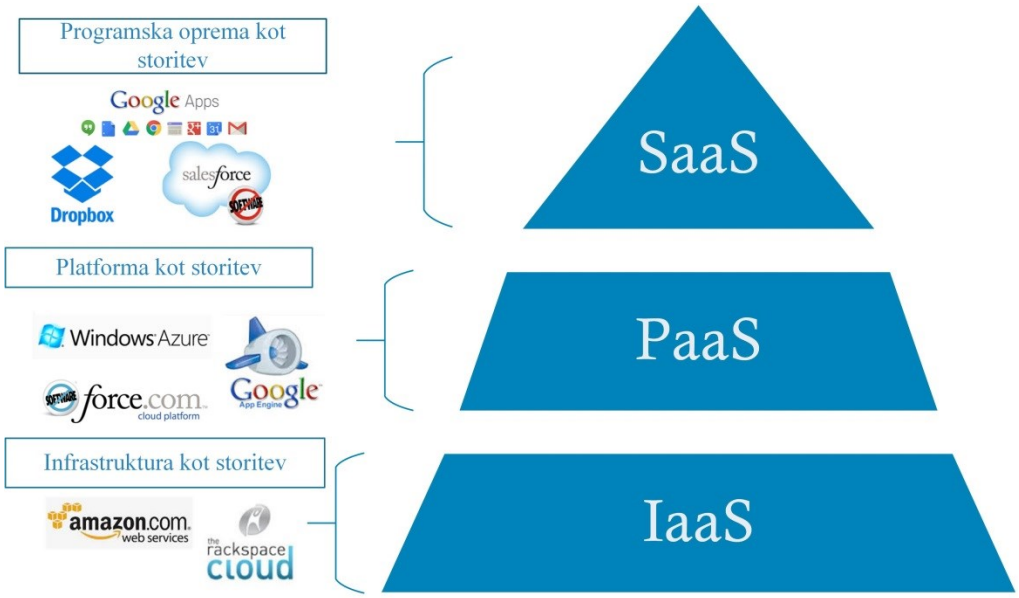
- Širok dostop do omrežja; zmogljivosti so na razpolago preko omrežja, dostopne preko standardnih mehanizmov in lahkih (*ang. thin*) ali debelih (*ang. thick*) odjemalcev platforme (npr. mobilni telefoni, tablice, prenosni računalniki).

- Združevanje virov; računalniški viri ponudnika storitev so združeni, da jih lahko uporablja več uporabnikov, in sicer po več najemniškem modelu (*ang. multi-tenant model*) z različnimi fizičnimi in virtualnimi viri, ki so dinamično dodeljeni ali premeščeni glede na potrebe uporabnikov storitev.

- Hitra elastičnost; zmožnosti so lahko avtomatsko elastično rezervirane in sproščene v skladu s povpraševanjem. Uporabniku storitev se te pri rezervaciji pogosto prikažejo kot neomejene in se jih lahko uporabi kadarkoli.

- Merjene storitve; oblačni sistemi avtomatsko kontrolirajo in optimizirajo uporabo računalniških virov, ki se lahko spremljajo, nadzorujejo ter poročajo in s tem zagotavljajo preglednost tako za ponudnika kot tudi za uporabnika storitev [45].

Računalništvo v oblaku delimo na tri storitvene modele, ki so prikazani na sliki v nadaljevanju.



Slika 1: Različni modeli storitev računalništva v oblaku

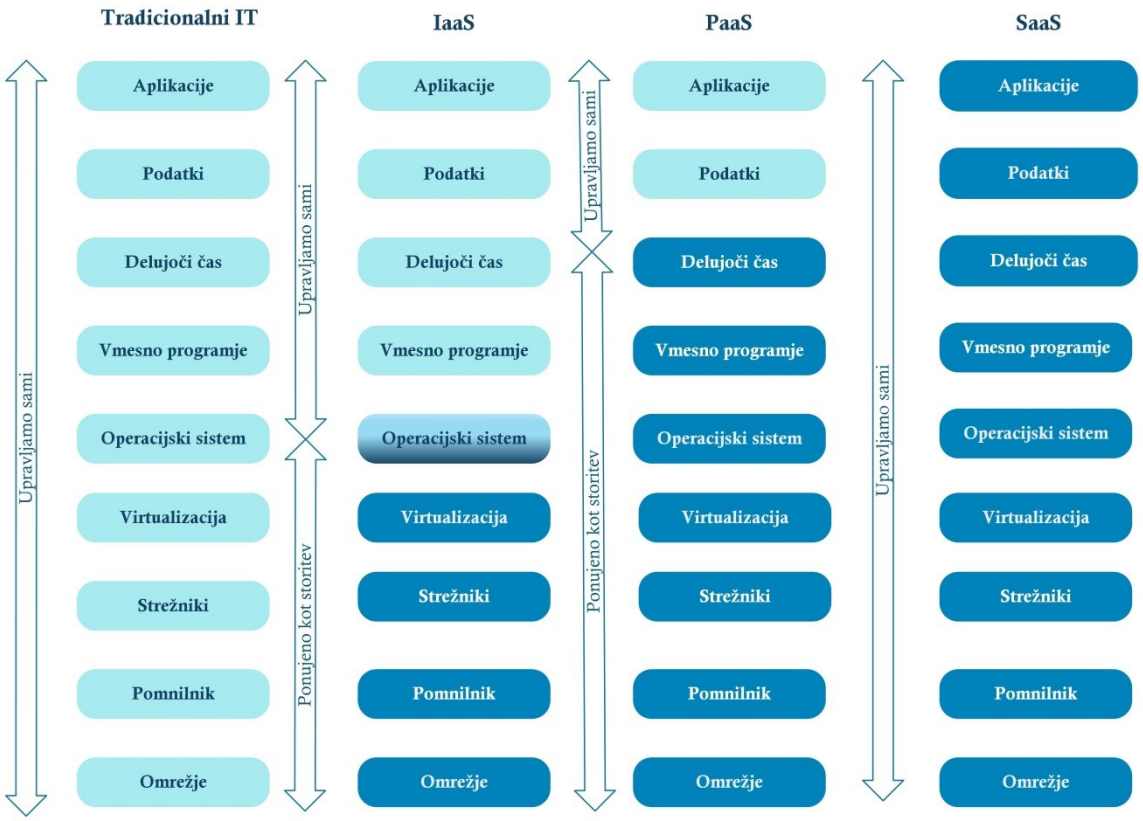
Storitveni modeli računalništva v oblaku so:

- Infrastruktura kot storitev – IaaS (*ang. Infrastructure as a Service*); osnovni nivo storitev v oblaku, kjer so računalniški viri in informacijska infrastruktura v lasti ponudnika storitev in jih ta ponuja kupcem oz. uporabnikom v najem na zahtevo. Uporabniki lahko sami rezervirajo vire z uporabo spletnega grafičnega vmesnika, ki služi kot upravljavska konzola za celotno okolje. Poleg tega imajo uporabniki lahko možnost dostopa do virov skozi API vmesnike. Uporabniki skrbijo za instalacijo in vzdrževanje operacijskega sistema, podatke in aplikacije ter dodeljeno omrežje in omrežne komponente.

- Platforma kot storitev – PaaS (*ang. Platform as a Service*); osrednji nivo storitev v oblaku, ki ponuja široko zbirko aplikacij infrastrukturnih storitev, kot so različne integracije, upravljanje poslovnih procesov ali storitve baze podatkov. Ponudnik zagotavlja razvojno okolje za aplikacije, kar tipično vključuje tudi operacijski sistem, bazo podatkov in spletni strežnik. Uporabniki skrbijo le za aplikacije ter njihove konfiguracije in ne skrbijo za ostale komponente ali oblačno infrastrukturo.

- Programska oprema kot storitev – SaaS (*ang. Software as a Service*); najvišji nivo storitev v oblaku, ki so v lastništvu in upravljanju enega izmed ponudnikov storitev ter oddaljeno dostavljene uporabnikom. Storitve so dostopne skozi aplikacije, do katerih uporabnik dostopa preko različnih naprav in odjemalcev, kot so spletni vmesniki (npr. spletna pošta) ali namenske aplikacije. Pri tem modelu uporabnik lahko le dostopa do storitve ter jo v

omejenem obsegu tudi konfigurira, za vse ostalo pa skrbi ponudnik oblaka [45, 46, 47, 48, 49].



Slika 2: Porazdelitev odgovornosti glede na različne modele storitev računalništva v oblaku

Razlikujemo tudi tri različne načine postavitve oblaka, in sicer:

- Zasebni oblak (*ang. private cloud*); oblačna infrastruktura je namenjena za uporabo le eni organizaciji. Lahko je v lastništvu in upravljanju organizacije, tretje osebe ali kombinacija obojega. V večini primerov je postavitev na lokaciji organizacije (*ang. on-premises*), čeprav to ni pogoj.

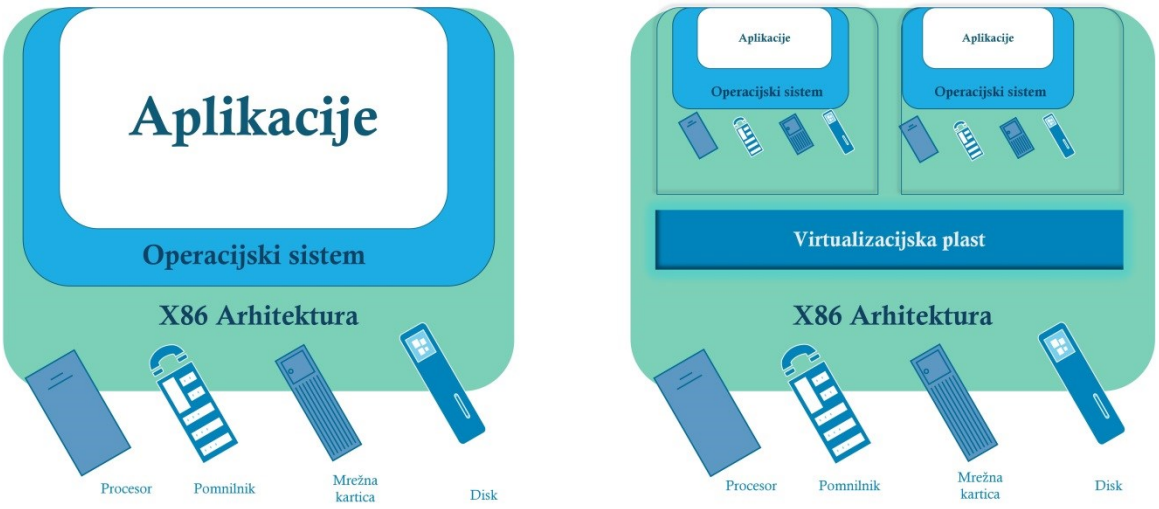
- Javni oblak (*ang. public cloud*); oblačna infrastruktura je na razpolago vsakemu uporabniku in širši javnosti. Tehnično gledano ni bistvene razlike med zasebnim ali javnim oblakom. Ker se nahaja v prostorih ponudnika storitev, dostopajo uporabniki storitev do virov preko javnega omrežja in spleta.

- Hibridni oblak (*ang. hybrid cloud*); oblačna infrastruktura je kombinacija zasebnega in javnega oblaka, ki so še zmeraj različne entitete, ampak povezane skupaj ter ponujajo prednosti večmodelnega pristopa. Na primer: nekatere organizacije občutljive poslovne podatke shranjujejo le v svojih prostorih in znotraj privatnega oblaka, medtem ko si začasne dodatne potrebe po povečanju kapacitet zagotovijo iz javnega oblaka [45, 46, 47, 52].

2.1 Virtualizacija

Virtualizacija je zmožnost simulacije strojne platforme z uporabo programske opreme. Vse funkcionalnosti so ločene od strojne opreme in simulirane kot virtualna instanca s sposobnostjo, da ta deluje enako kot strojna oprema.

Koncept virtualizacije ima svoje korenine v šestdesetih letih prejšnjega stoletja in »mainframe« računalnikih. Takrat je bil IBM glavni pobudnik pri razvoju t. i. »time-sharing« rešitev. Pri tem gre za način deljene uporabe računalniških virov znotraj večje skupine uporabnikov, s ciljem izboljšati učinkovitost tako uporabnikov kot tudi izkoriščenost zelo dragih računalniških virov. Ta model je predstavljal velik preboj na področju informacijskih tehnologij. Podjetja in celo uporabniki so lahko uporabljali računalnike, ne da bi jih dejansko kupili. Podobni razlogi so gonilo virtualizacije tudi danes. Najboljši način, kako izboljšati izkoriščenost virov in hkrati poenostaviti upravljanje, je z virtualizacijo [19].



Slika 3: Primerjava koncepta delovanja fizičnega in virtualiziranih strežnikov

Nekatere prednosti uporabe virtualizacije so:

- manjši stroški porabe energije zaradi konsolidacije strežnikov,
- manjši stroški vzdrževanja strojne opreme,
- hitrejše in lažje vzdrževanje ter upravljanje in optimizacija virov,
- neodvisnost od ponudnika strojne opreme,
- učinkovitejše in lažje poslovanje skozi uporabo virtualizacijskih tehnologij (npr. enkapsulacija OS in aplikacij v eno datoteko, živa migracija, posnetki stanja), ki omogočajo enostavnejše mehanizme visoke razpoložljivosti in izdelave varnostnih kopij.

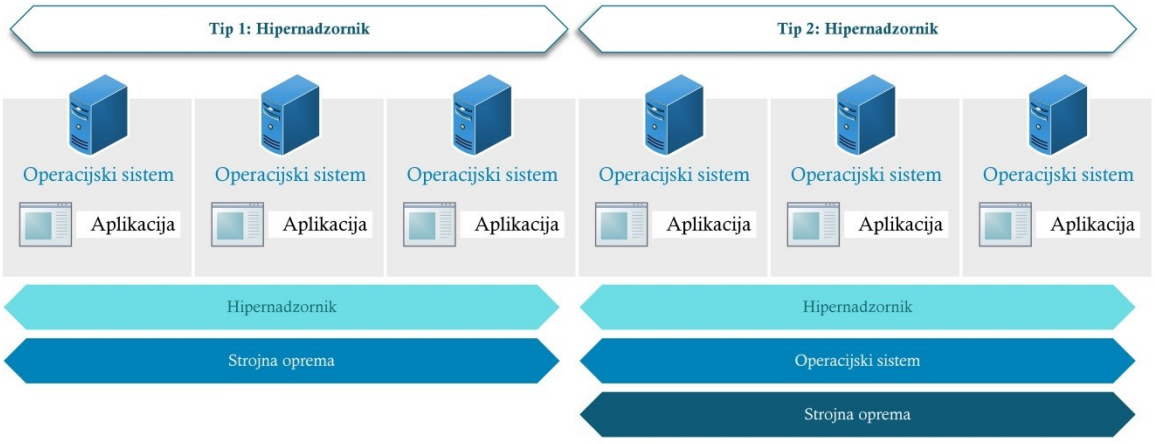
Obstaja več konceptov virtualizacije, nekateri so opisani v nadaljevanju.

2.1.1 Virtualizacija strojne opreme

Programska oprema, ki se izvaja na gostiteljskem računalniku (*ang. host machine*), omogoča ustvarjanje navideznih računalnikov (*ang. virtual machine*). Ti se s stališča uporabnika in operacijskega sistema obnašajo popolnoma enako kot fizični računalniki. Programska oprema, ki kontrolira virtualizacijo strojne opreme, je hipernadzornik (*ang. hypervisor*). Obstajata dva tipa hipernadzornikov:

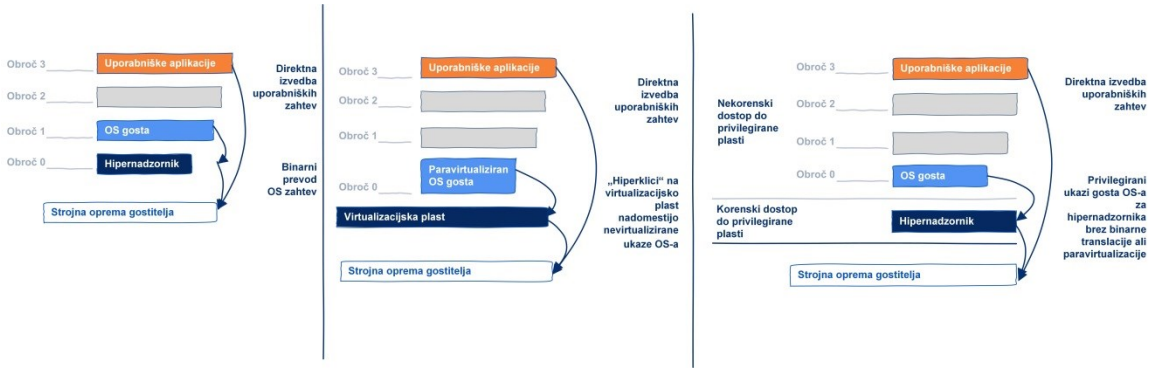
- Tip 1 hipernadzornik (*ang. Type 1 hypervisor*); izvaja se direktno na sistemski strojni opremi. Primer: VMware ESXi, Microsoft Hyper-V, Xen.

- Tip 2 hipernadzornik (*ang. Type 2 hypervisor*); izvaja se na operacijskem sistemu gostitelja. Primer: VMware Workstation, Microsoft Virtual PC, Oracle Virtual Box, KVM, QEMU.



Slika 4: Razlike med hipernadzornikoma tip 1 in tip 2

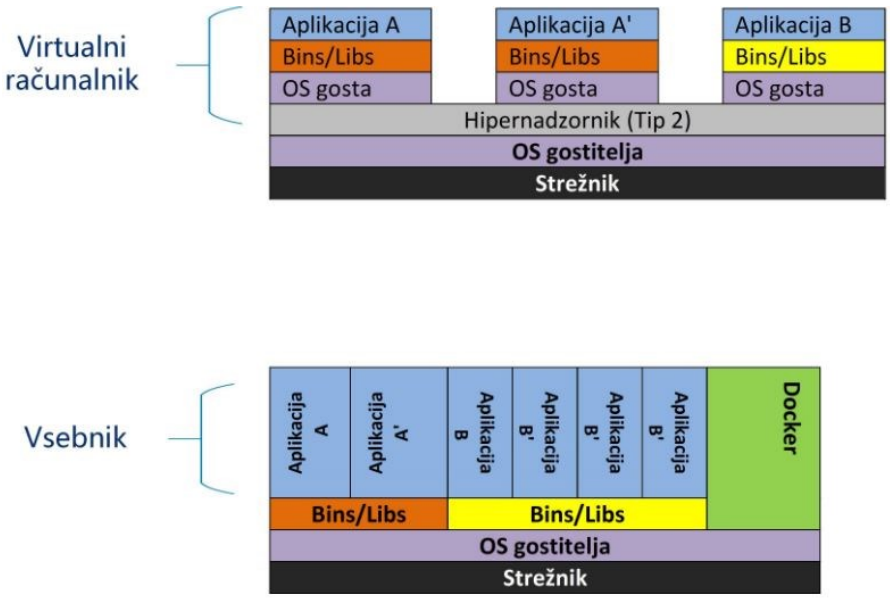
Razlikujemo polno virtualizacijo (*ang. full virtualization*), paravirtualizacijo in strojno podprto virtualizacijo (*ang. hardware assisted virtualization*). Polna virtualizacija omogoča takšno simulacijo strojne opreme, da na njej v izolaciji teče nemodificiran operacijski sistem gosta (*ang. guest*). Operacijski sistem gosta se virtualnega okolja niti ne zaveda. Paravirtualizacija je koncept, pri katerem so namesto simulacije strojne opreme v uporabi posebni programski vmesniki (*ang. application program interface – API*), ki omogočajo izvajanje programov gosta v izolaciji oz. posredovanje njihovih ukazov direktno operacijskemu sistemu gostitelja. Virtualno okolje je v tem primeru za aplikacije transparentno, ampak ne tudi za operacijski sistem gosta, na katerem se izvajajo. Pri strojno podprti virtualizaciji zaradi mikroprocesorske podpore in strojno podprtih instrukcij je možno, da gost preko hipernadzornika direktno posreduje privilegirane ukaze strojni opremi gostitelja brez potrebe po binarni translaciji ali paravirtualizaciji [12].



Slika 5: Prikaz koncepta polne binarne virtualizacije, paravirtualizacije in strojno podprte virtualizacije

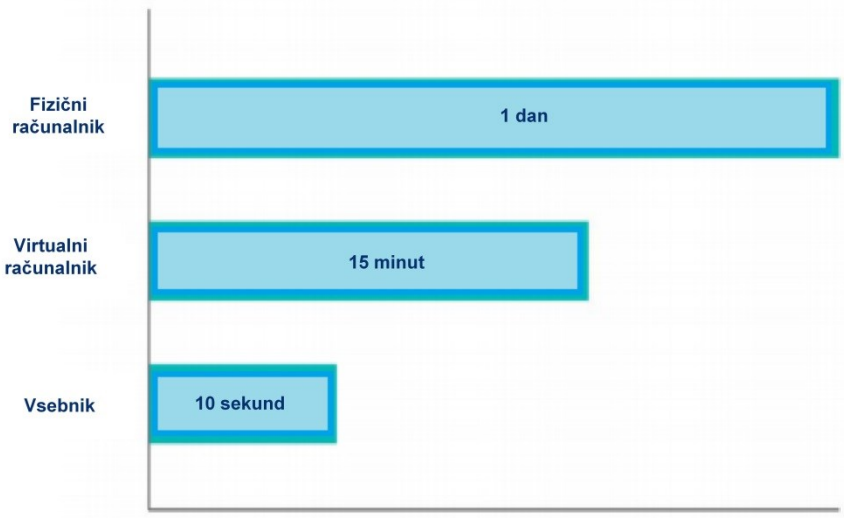
2.1.2 Virtualizacija na nivoju operacijskega sistema

Virtualizacija na nivoju operacijskega sistema (*ang. operating system-level virtualization*) je koncept, pri katerem jedro (*ang. kernel*) operacijskega sistema omogoča izvajanje več izoliranih instanc na enem gostitelju. Večina implementacij je zaradi zgodovinskih tehnoloških razlogov narejenih na Linuxu, saj imajo začetki koncepta korenine v Unixu in uporabo sistemske funkcije »chroot« že od leta 1982. Med najbolj znanimi implementacijami so: LXC (Linux Containers), OpenVZ, Linux-VServer, Solaris Containers, FreeBSD jails [15]. Zelo veliko zanimanje industrije je v zadnjem času povzročil Docker, ki je v svoji osnovi LXC. S svojimi orodji je omogočil, da se aplikacija skupaj z vsemi svojimi komponentami in odvisnostmi *(ang. dependency)* enkapsulira v virtualni vsebnik (*ang. container*) in izvaja na kateremkoli Linux strežniku [16]. Vsebniki se med sabo lahko povežejo skozi lastne omrežne vmesnike z lastnimi IP naslovi. S tega stališča se obnašajo popolnoma enako kot omrežno povezani fizični računalniki [15].



Slika 6: Primerjava virtualnih računalnikov in vsebnikov

Za razliko od tradicionalnih hipernadzornikov, ki za koncept potrebujejo emulacijo kompletnega virtualnega dela strojne podpore, vsebniki delujejo znotraj istega operacijskega sistema in je zaradi tega izkoriščenost strojne opreme veliko večja. Dodatne prednosti uporabe vsebnikov so: manj operacijskih sistemov za upravljanje, večja mobilnost aplikacij, lažja implementacija popravkov operacijskega sistema, lažja implementacija popravkov aplikacij, veliko hitrejše in lažje upravljanje, hitrejši odziv na potrebe po dodatnih virih itd.



Slika 7: Časovna primerjava upravljanja aplikacij v različnih načinih delovanja

2.1.3 Virtualizacija omrežij in omrežnih funkcij – SDN, NV, NFV

Programsko določena omrežja (*ang. software defined networking – SDN*), omrežna virtualizacija (*ang. network virtualization – NV*) in virtualizacija omrežnih funkcij (*ang. network function virtualization – NFV*) so tehnologije, ki se medsebojno dopolnjujejo, niso pa soodvisne. Ponujajo nove načine, kako dizajnirati, uvajati in upravljati omrežja in omrežne storitve. SDN loči kontrolni in posredovalni nivo ter omogoča centraliziran pogled na omrežje, kar omogoča boljše upravljanje in avtomatizacijo omrežnih storitev. NV omogoča kreiranje virtualnih omrežij, ločenih od fizične infrastrukture, ter odpravlja ovire pri dosedanjih tradicionalnih pristopih. NFV ločuje omrežne funkcije, kot so na primer DNS (sistem domenskih imen), DPI (globoko analiziranje paketov in prometa), CDN (omrežje za dostavo vsebin), od specializirane strojne opreme proizvajalcev tako, da se te lahko izvajajo kot del programske opreme na standardnih fizičnih ali virtualnih strežnikih.

Skupni cilji vseh treh tehnologij so [3]:

- premakniti funkcionalnosti v programsko opremo,
- namesto specializirane (*ang. proprietary*) opreme proizvajalcev uporabljati standardne strežnike in stikala,
- uporabljati in razvijati programske vmesnike (*ang. application program interface – API*) in njihove prednosti,
- podpirati bolj učinkovito orkestracijo, virtualizacijo in avtomatizacijo omrežnih storitev.

2.2 Omrežja

Tradicionalne omrežne tehnologije niso bile dizajnirane, da zadostijo potrebam današnjih uporabnikov. Zahteve podjetij, ponudnikov storitev in končnih uporabnikov so zmeraj bolj kompleksne s stališča omrežnih tehnologij. Skrbniki omrežij s tradicionalnimi orodji proizvajalcev in ročnimi konfiguracijami omrežnih elementov jih s težavami rešujejo. Omejitve trenutnih tradicionalnih omrežnih tehnologij so sledeče [2]:

- Kompleksnost; če znotraj današnjega tradicionalnega omrežja želimo dodati ali premakniti katerokoli napravo, to zahteva večje število ročnih operacij. Po navadi je treba spremeniti konfiguracijo več stikal, usmerjevalnikov, požarnih pregrad ter posodobiti sezname za kontrolo dostopa (*ang. access control list – ACL*), virtualna lokalna omrežja (*ang. virtual local area network – VLAN*) itd. Hkrati je treba upoštevati tudi posebnosti posameznih proizvajalcev in načine, kako je to možno realizirati na določeni opremi.

- Nezmožnost skalabilnosti; hkrati z rastjo zahtev znotraj podatkovnih skladišč morajo rasti tudi omrežja. Do zdaj se je širitev ali rast omrežja reševala z začetnimi predimenzioniranimi sistemi in predvidljivimi vzorci prometa. V sodobnih virtualnih podatkovnih skladiščih s tisoči virtualnih strežnikov in medsebojnih povezav tega ni več možno reševati na takšen način, saj so vzorci prometa izredno dinamični in nepredvidljivi. Tako imenovani »Mega ponudniki«, kot so Google, Facebook, se soočajo z izjemnimi izzivi, kako povečevati svoje omrežne, procesorske in podatkovne vire. Takšna podjetja potrebujejo t. i. »hiper-razširljiva« visokozmogljiva omrežja s povezavami med več tisočimi strežniki.

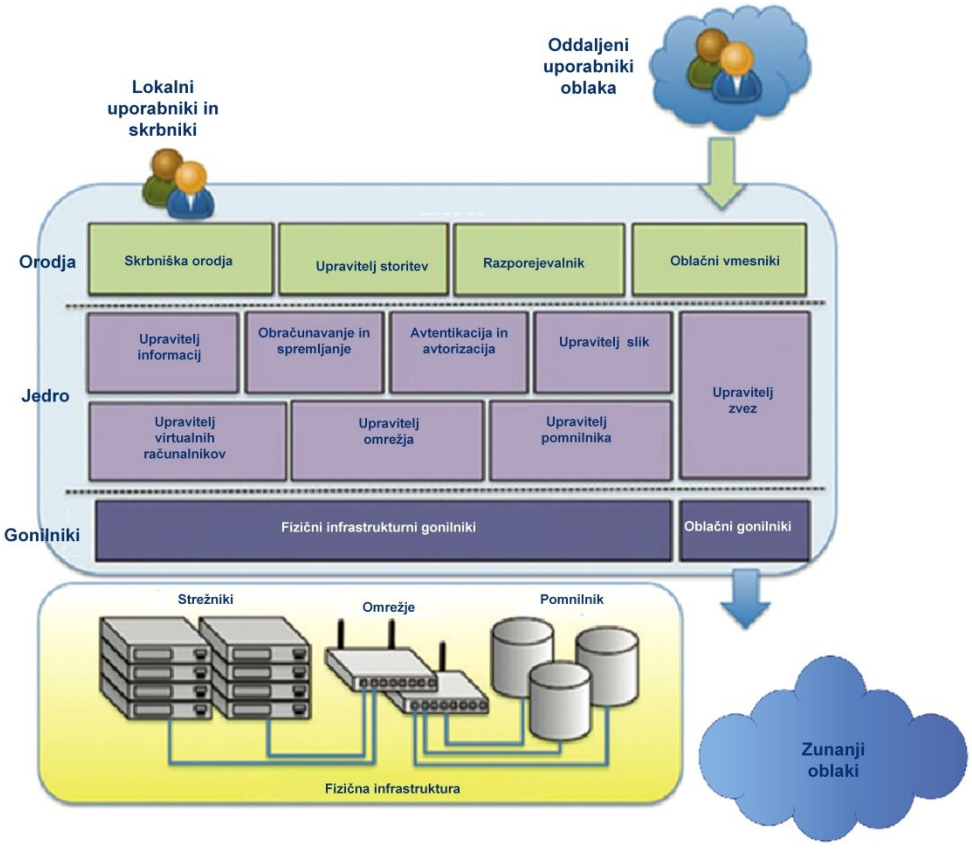
- Odvisnost od proizvajalcev; ponudniki storitev in podjetja iščejo nove načine, kako ponuditi konkurenčne storitve v zmeraj bolj zahtevnem poslovnem okolju. Pri tem so pogosto omejeni z opremo proizvajalcev, ki uporabljajo lastne nestandardizirane protokole. Pomanjkanje odprtih vmesnikov ter večletni razvojni cikli produktov dodatno onemogočajo prilagajanje opreme njihovim individualnim potrebam.

2.3 Oblačne platforme

V tem poglavju so opisane nekatere odprtokodne oblačne platforme, ki se osredotočajo na arhitekturni model oblaka infrastruktura kot storitev – IaaS. Izbrane so bile glede na trenutno razširjenost uporabe ter tudi glede na priljubljenost v odprtokodnih skupnostih. Oblačne platforme, kot so Amazon EC2, Microsoft Azure, IBM SoftLayer in Google Compute Engine, niso upoštevane pri izbiri, ker gre za komercialne rešitve.

2.3.1 OpenNebula

OpenNebula je odprtokodna oblačna platforma, katere razvoj se je začel leta 2005 kot razvojni projekt. Prva javna verzija je bila objavljena leta 2008. Od leta 2010 pa obstaja tudi komercialna različica, za katero skrbi podjetje OpenNebula Systems, ki sta ga ustanovila prvotna avtorja razvojnega projekta. Razen gradnje privatnih oblakov omogoča tudi gradnjo hibridnih oblakov skozi integracijo z javnimi oblačnimi ponudniki, kot so Amazon Web Services, IBM SoftLayer in Microsoft Azure. Podpira različne hipervizorje, in sicer: KVM, Xen ter VMware ESXi. Za razliko od odprtokodne oblačne platforme OpenStack, ki zagotavlja funkcionalnost skozi integracijo več različnih komponent, ponuja OpenNebula to v enem samem paketu. To je hkrati prednost (enostavna in hitra uporaba) ter tudi pomanjkljivost platforme (slabša integracija z drugimi oblačnimi platformami). Virtualne instance oziroma navidezni računalniki tečejo na hipernadzornikih, sama oblačna platforma pa jih kreira ter skrbi za njihov nadzor in upravljanje. Storitve platforme se izvajajo na gostitelju.



Slika 8: Prikaz OpenNebula arhitekture

OpenNebula za upravljanje omrežnega dela oblaka uporablja t. i. Virtual Network Manager, ki omogoča preslikavo virtualnih omrežjih na fizična omrežja. Tako se fizična omrežja lahko razdelijo na več manjših omrežjih. Z uporabo različnih omrežnih gonilnikov (Virtual Network Manager drivers) se omogoči izolacija navideznih računalnikov v različna virtualna omrežja. V nadaljevanju so opisani posamezni gonilniki in njihove lastnosti:

- dummy; privzeti gonilnik, ki ne opravlja nobene omrežne operacije. Varnostna pravila (*ang. Security Groups rules*) so tudi prezrta,

- Security Groups; varnostna pravila se uporabljajo za filtriranje specifičnega omrežnega prometa,

- 802.1q; omejevanje dostopa do omrežja preko VLAN označevanja (*ang. tagging*), ki zahteva tudi strojno podporo na omrežnih stikalih,

- VXLAN; omejevanje dostopa do omrežja z uporabo novejšega VXLAN protokola, ki omogoča veliko večje število virtualnih omrežij (16 milijonov) kot VLAN označevanje (4096), pri tem se uporablja UDP enkapsulacija in IP multicast,

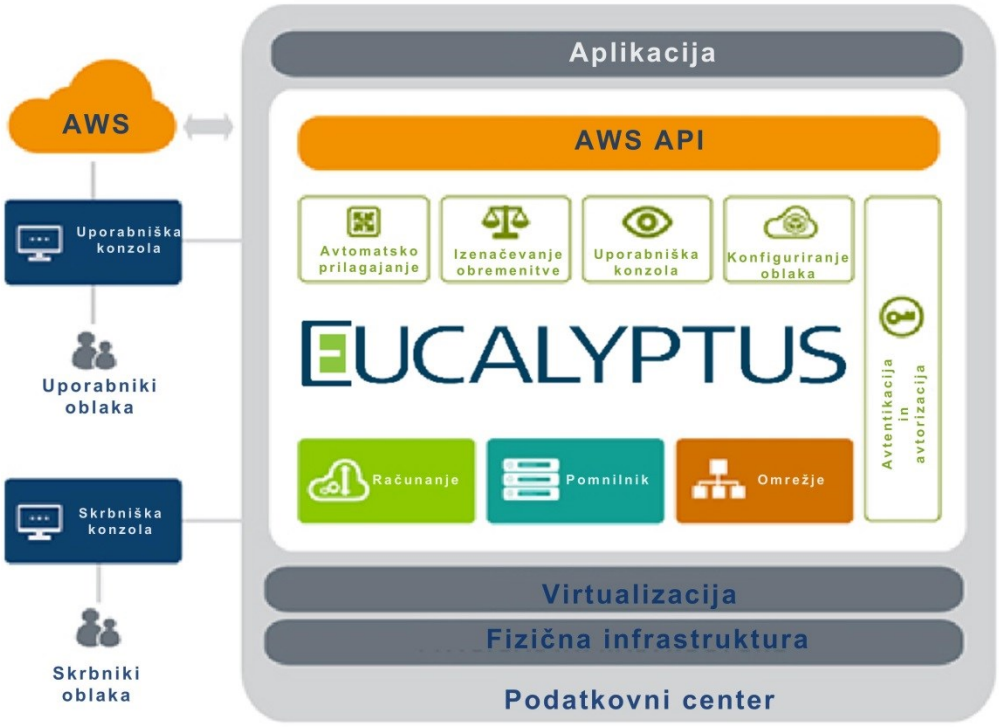
- ebtables; omejevanje dostopa do omrežja s pravili ebtables, ki filtrirajo eternet okvirje,

- ovswitch; omejevanje dostopa do omrežja z uporabo Open vSwitch virtualnega stikala,

- VMware; uporaba VMware omrežne infrastrukture za zagotavljanje izoliranih in 802.1q kompatibilnih omrežij za navidezne računalnike, ki tečejo znotraj VMware hipervizorja [31, 36, 37].

2.3.2 Eucalyptus

Eucalyptus je odprtokodna oblačna platforma, ki so jo razvili na Univerzi Santa Barbara v Kaliforniji. Omogoča grajenje zasebnih in hibridnih oblakov. Že leta 2008 je kot prva odprtokodna oblačna platforma omogočila integracijo na nivoju programskih vmesnikov z največjo in najbolj razširjeno komercialno oblačno platformo Amazon Web Services (AWS). To je omogočilo uporabniku, da lahko uporablja enaka orodja za upravljanje zasebnega in javnega oblaka.



Slika 9: Prikaz Eucalyptus arhitekture

Leta 2014 je lastništvo prevzelo podjetje HP in ga začelo ponujati kot del svojih storitev v oblaku HP Helion Cloud. Na razpolago je odprtokodna in komercialna različica. Slednja razen podpore odprtokodnih hipernadzornikov KVM in Xen podpira tudi VMware ESXi. Eucalyptus sestavlja več komponent, in sicer: Cloud Controller, User-Facing Services, Management Console, Object Storage Gateway, Object Storage Provider, Cluster Controller, Storage Controller ter Node Controller. Podprti so štirje različni omrežni načini, ki omogočajo izbiro različnih nivojev varnosti in prilagodljivosti. To so:

- Edge način; zasnovan je tako, da se integrira v že obstoječe omrežne topologije ter ponuja združljivost z večino standardnih omrežnih načinov delovanja Amazonovega oblaka. Omogoča le dva osnovna načina povezovanja, kar lahko v nekaterih okoljih predstavlja omejitev.

- Managed način; v tem načinu Eucalyptus upravlja lokalno omrežje virtualnih instanc in zagotavlja vse omrežne funkcije, ki jih trenutno podpira, vključno z izolacijo omrežja virtualnih instanc, varnostne skupine in pravila, elastične IP-je ter storitev metapodatkov.

- Managed (brez VLAN) način; enak je Managed načinu, le da ne zagotavlja izolacije omrežja virtualnih instanc na 2. plasti komunikacijskega sklada (*ang. Layer 2 – L2*). Ta je

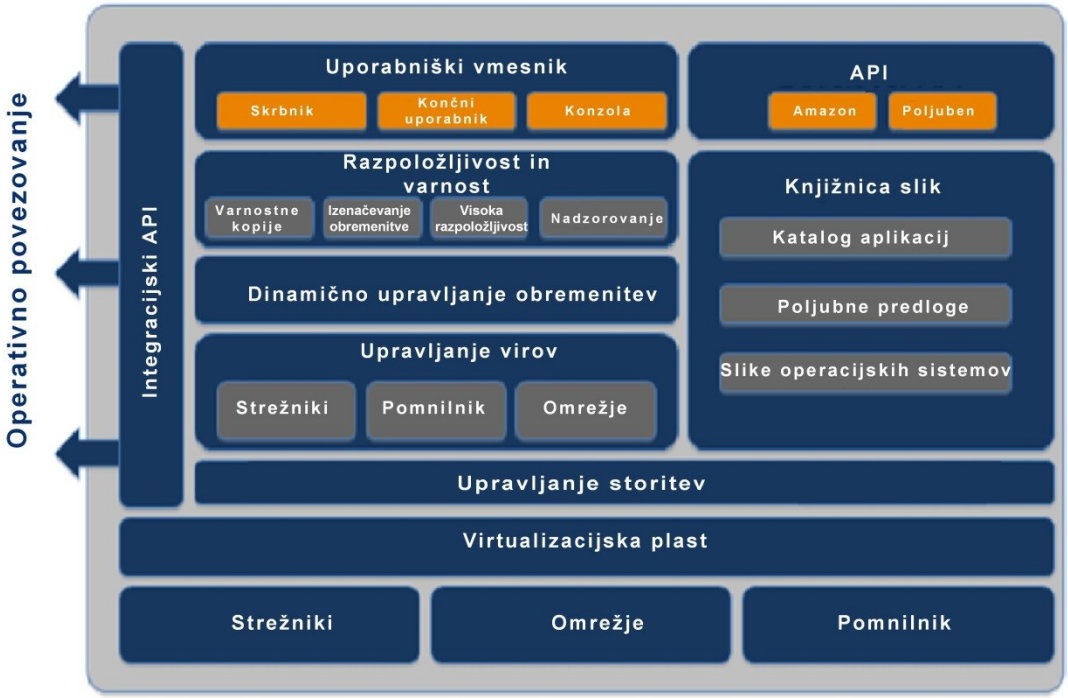
z uporabo različnih varnostnih skupin in pravil na različnih podomrežjih zagotovljena le na 3. plasti komunikacijskega sklada (*ang. Layer 3 – L3*).

- Virtual Private Cloud (VPC) način; z uporabo in integracijo odprtokodne omrežne virtualizacijske platforme MidoNet je omogočena programska abstrakcija omrežja in kreiranje programskih virtualnih omrežij.

Konec leta 2015 se je podjetje HP odločilo, da od januarja leta 2016 ne bodo več ponujali lastnega HP Helion javnega oblaka. Namesto tega se bodo osredotočili na privatne oblake ter njihovo integracijo z drugimi javnimi oblaki, kot sta npr. Amazon Web Services in Microsoft Azure [31, 32, 33, 34, 35, 37].

2.3.3 CloudStack

Razvoj odprtokodne oblačne platforme CloudStack se je začel kot projekt start-up podjetja VMOps leta 2008. Kmalu so se preimenovali v Cloud.com in leta 2010 večino programske kode objavili pod odprto kodo GPLv3. Leta 2011 jih je kupilo podjetje Citrix in nato so tudi preostali del kode objavili pod kodo GPLv3. Naslednje leto je Citrix CloudStack podaril fundaciji Apache Software Foundation (ASF). CloudStack omogoča gradnjo javnih in privatnih oblakov. Zaradi uporabe API vmesnikov se lahko poveže z Amazonovim oblakom in tvori hibridni oblak. Podprto je delovanje več različnih hipernadzornikov, kot so: KVM, Xen, Microsoft Hyper-V, VMware ESXi. Poleg tega so podprti tudi t. i. fizični gostitelji (*ang. Bare Metal hosts*), in sicer novejše verzije Linux operacijskih sistemov: RHEL, CentOS, Fedora ter Ubuntu. Na ta način je znotraj oblaka omogočeno, da se fizični gostitelji kreirajo in upravljajo enako kakor navidezni računalniki. CloudStack podpira tudi Linux Containers (LXC). Upravljanje oblaka je razen grafičnega spletnega vmesnika in ukazne vrstice zagotovljeno tudi skozi RESTful API vmesnike.



Slika 10: Prikaz CloudStack arhitekture

CloudStack omogoča tudi delovanje v visoko razpoložljivem načinu (*ang. high availability*) tako, da se obremenitev razporeja po več različnih vozliščih znotraj oblaka. Podprte so različne omrežne postavitve, ki večinoma spadajo v enega izmed dveh scenarijev:

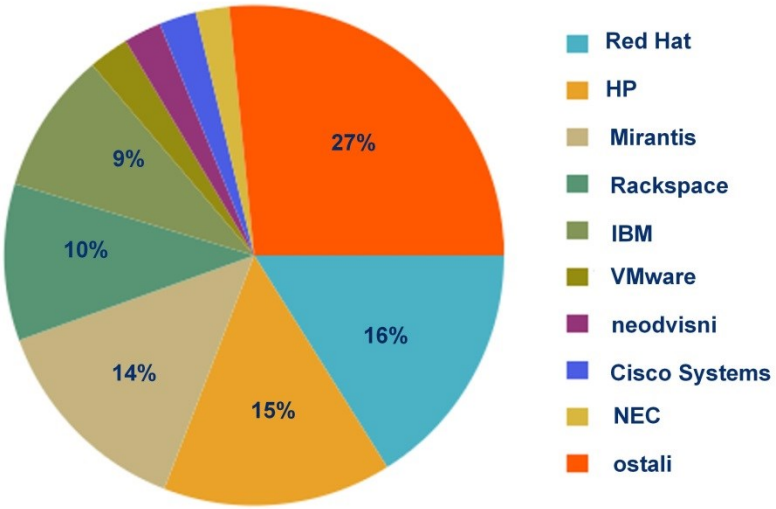
- osnovni scenarij; enako kot pri klasični omrežni postavitvi v Amazonovem oblaku je zagotovljeno enotno L2 omrežje, kjer je izolacija uporabnikov narejena na L3 omrežnem nivoju, ki ga zagotavlja hipernadzornikov omrežni most (*ang. bridge device*).

- napredni scenarij; običajno je zagotovljena omrežna izolacija na L2 nivoju z uporabo VLAN omrežij, čeprav ta kategorija vključuje tudi SDN tehnologije, kot je npr. Nicira NVP, in uporabo programskih vtičnikov, kot sta MidoNet in Open vSwitch [38, 39].

2.3.4 OpenStack

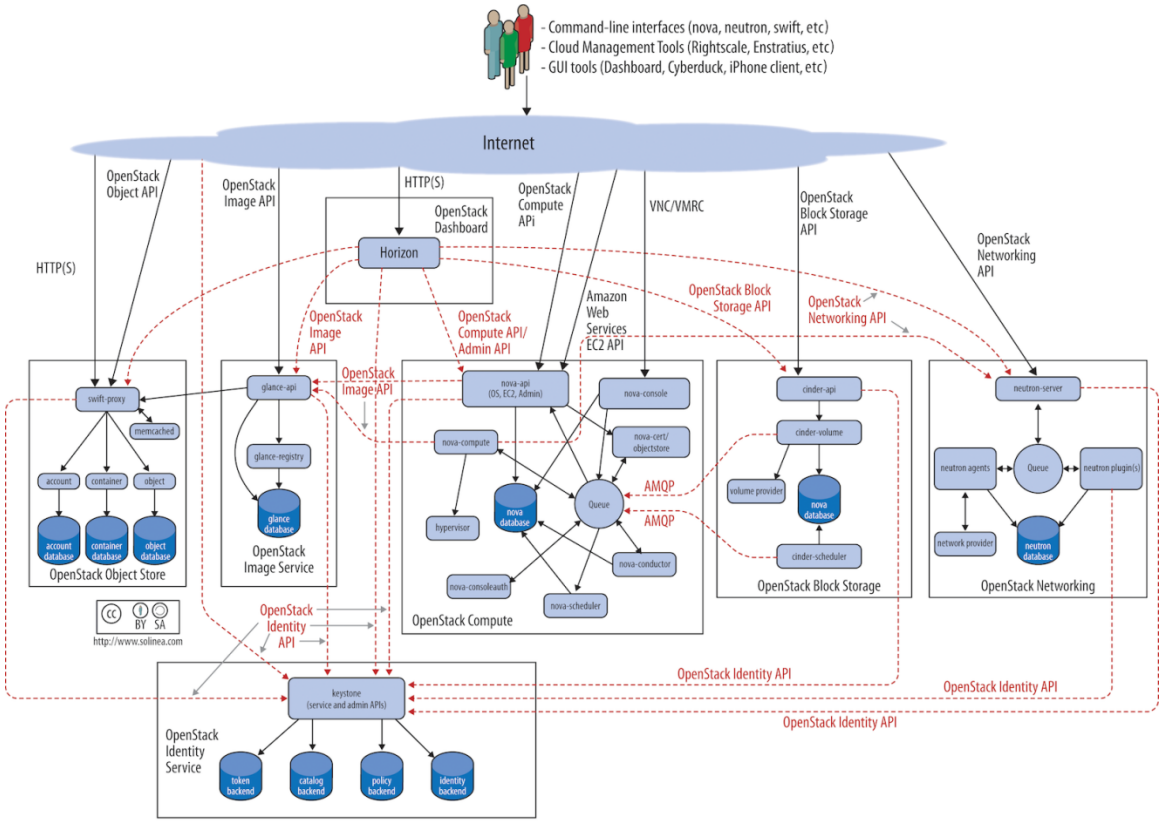
OpenStack je trenutno najbolj razširjena odprtokodna oblačna platforma [41, 42]. Omogoča gradnjo javnih in privatnih oblakov. OpenStack se je začel kot skupni projekt podjetij Rackspace Hosting in NASA-e leta 2010. Istega leta je bila izdana tudi prva verzija. Od leta 2012 je pod upravljanjem neprofitnega združenja OpenStack Foundation. Več kot petsto različnih podjetij, predvsem proizvajalcev in ponudnikov informacijsko komunikacijskih rešitev, se je pridružilo

projektu: AT&T, Canonical, Cisco, Citrix, Dell, Ericsson, Google, HP, Huawei, IBM, Intel, Oracle, Red Hat, VMware itd.



Slika 11: Prikaz statistike razvojnih prispevkov posameznih podjetij pri OpenStack oblačni platformi

Nekateri med njimi ponujajo tudi svoje lastne namenske (*ang. appliance*) verzije OpenStack rešitve, vključno s podporo. Razen tega OpenStack uporabljajo tudi mnoge znane raziskovalne organizacije, kot so npr. CERN, MIT, NASA. OpenStack je sestavljen iz več projektov oziroma komponent. Arhitektura je prikazana na sliki 12.



Slika 12: Prikaz OpenStack arhitekture [40]

Ključne komponente OpenStacka so:

- Nova – Compute; glavna komponenta, ki zagotavlja kreiranje, upravljanje, nadziranje in avtomatizacijo virtualnih instanc. Nova sama nima nobenih virtualizacijskih zmožnosti, ampak namesto tega uporablja libvirt API-je za interakcijo s podprtimi hipernadzorniki. To pa so: KVM, Xen, VMware ESXi/vSphere, Microsoft Hyper-V ter tudi LXC. Za komunikacijo z zunanjimi sistemi uporablja spletne programske vmesnike, ki so kompatibilni z Amazonovim oblakom.

- Glance – Image Service; komponenta, ki zagotavlja skladiščenje, odkrivanje, registracijo in pridobivanje slik navideznih računalnikov. Te lahko uporabimo tudi kot predlogo. Glance uporablja REST programski vmesnik za poizvedovanje o podatkih shranjenih slik.

- Neutron – Networking; komponenta, ki zagotavlja omrežno povezljivost kot storitev za druge OpenStack komponente, kot je npr. Nova. Neutron je zelo modularna komponenta, porazdeljena preko več neodvisnih komponent. Vsaka komponenta, razen Neutron Serverja, se lahko zamenja z drugo komponento, ki ponuja enako storitev. Ta zagotavlja programske vmesnike uporabnikom in drugim OpenStack komponentam, kakor tudi skrbi

za shranjevanje omrežnega stanja in uporabniške konfiguracije v bazi podatkov. Neutron Open vSwitch Agent se izvaja na vsakem Nova in Neutron vozlišču in zagotavlja L2 povezljivost med instancami ter agenti, ki se izvajajo na različnih Nova vozliščih. Skladno z uporabniško konfiguracijo, ki jo pridobi iz Neutron Server komponente, gradi navidezna virtualna prekrivna omrežja. Neutron DHCP Agent se izvaja na Neutron vozlišču in skrbi za dodeljevanje IP konfiguracije instancam. Neutron L3 Agent se tudi izvaja na Neutron vozlišču in skrbi za L3 povezljivost, plavajoče IP naslove (*ang. floating IP adresses*) in varnostna pravila skupin (*ang. security groups*). Pri tem se za IP usmerjanje zanaša na same zmožnosti usmerjanja Neutron vozlišča, za plavajoče IP naslove in varnostna pravila pa se zanaša na *iptables* pravila. L3 in DHCP agent uporabljata tudi Linux *namespaces*, da se lahko različnim uporabnikom zagotovi uporaba enakih podomrežnih naslovov. Neutron Metadata Agent zagotavlja metapodatke o virtualnih instancah. Neutron uporabnikom OpenStack oblaka omogoča gradnjo različnih omrežnih topologij in konfiguriranje naprednih omrežnih politik. Z uporabo različnih vtičnikov (Open vSwitch, Cisco Nexus, MidoNet, OpenContrail, IBM SDN-VE, Linux Bridge itd.) je možno dodatno razširiti omrežne funkcionalnosti s storitvami, kot so VPN, požarne pregrade itd. Trenutno se večinoma kot privzeto uporablja vtičnik Modular Layer 2, ki je nadomestil vtičnika openvswitch in linuxbridge. Podpira različne omrežne scenarije, in sicer Flat, Local, VLAN, GRE, VXLAN.

- Cinder – Block Storage; komponenta, ki zagotavlja virtualni bazen (*ang. pool*) naprav za shranjevanje blokov podatkov. Uporabniki oblaka, ki do teh virov dostopajo in jih rezervirajo skozi RESTfull API, se niti ne zavedajo, kjer so dejansko podatki in na kateri vrsti naprave. Pri tem se uporablja LVM (*ang. Logical Volume Manager*), ki je Linux mehanizem za upravljanje logičnih particij. Na takšen način se lahko enostavno upravlja veliko število različnih vrst diskovnih kapacitet ter enostavno dodaja, spreminja ali zamenja diskovne particije.

- Swift – Object Storage; komponenta, ki zagotavlja porazdeljen in razširljiv sistem hranjenja podatkov. Ti so zapisani na več različnih diskih in strežnikih v podatkovnem središču. V primeru odpovedi diska ali strežnika OpenStack poskrbi za replikacijo podatkov iz aktivnih vozlišč na novo lokacijo. Horizontalna skalabilnost je enostavno zagotovljena z dodajanjem novih strežnikov, saj OpenStack uporablja le programsko logiko za mehanizme replikacije in porazdeljevanja podatkov po različnih napravah, zato se lahko uporablja standardne diske in strežnike.

- Keystone – Identity Service; komponenta, ki zagotavlja avtentikacijo in avtorizacijo za vse OpenStack servise v oblaku. Podpira več oblik preverjanja pristnosti, vključno s standardnim načinom z uporabniškim imenom in geslom ter žetoni (*ang. tokens*). Lahko se integrira tudi z obstoječimi imeniki, kot je npr. LDAP.

- Horizon – Dashboard; komponenta, ki zagotavlja skrbnikom in uporabnikom oblaka spletni grafični portal za dostop do OpenStack storitev, kot so zagon instanc, dodeljevanje IP naslovov in konfiguracija varnostnih pravil skupin. Sama komponenta se ne šteje med osnovne komponente, ampak med dodatne OpenStack komponente. Grafični vmesnik je eden od številnih načinov, kako lahko uporabniki dostopajo do oblačnih virov. Zasnova omogoča prilagajanje (*ang. branding*) potrebam ponudnikov storitev.

OpenStack vključuje številne dodatne komponente. Med bolj izpostavljenimi sta:

- Heat, ki zagotavlja orkestracijski mehanizem in omogoča kreiranje večplastnih uporabniških okolij v oblaku z uporabo predlog. Te temeljijo na tekstovnih datotekah in se jih obravnava kot vhodno kodo v OpenStack oblaku. Tako je možno npr. le z eno predlogo kreirati več instanc, določiti njihove lastnosti, definirati omrežja ter omrežna pravila.

- Ironic, ki zagotavlja kreiranje in upravljanje fizičnih strežnikov na enak način kot za navidezne računalnike [31, 40, 44].

2.3.5 Avtomatizacija in orkestracija oblačne infrastrukture

Oblačne platforme v primerjavi s tradicionalnimi podatkovnimi centri in fizično infrastrukturo že same omogočajo hitrejše in enostavnejše postavljanje informacijskega okolja in infrastrukture. Dodatno zmanjšanje časa in zniževanja stroškov postavitve, vodenja ter prilagajanja oblačne infrastukture dosežemo z uporabo orodij za avtomatsko namestitev in konfiguracijo oblačne infrastrukture. Ta omogočajo samodejno prilagojeno namestitev osnovnega operacijskega sistema ter tudi avtomatizacijo in centralno konfiguracijo vseh storitev, kar zmanjšuje ročno delo in možnosti za človeške napake.

Razen OpenStack orkestracijske komponente Heat, ki je opisana v prejšnjem poglavju, se za avtomatizacijo in orkestracijo oblačne infrastrukture uporabljajo še mnoga druga orodja. Med bolj znanimi odprtokodnimi orodji za avtomatizacijo in orkestracijo oblačne infrastrukture so:

- Puppet je odprtokodno orodje, napisano v programskem jeziku Ruby, ki deluje na številnih Linux, Unix in tudi Microsoft Windows operacijskih sistemih. Uporablja lasten deklarativni jezik za opis konfiguracije sistema ter deluje v načinu odjemalec – strežnik. V starejših verzijah je bil uporabljen protokol XML-RPC, ki ga je v novejših verzijah nadomestil protokol REST. Odjemalec privzeto vsake pol ure na strežniku preveri dosegljivost nove konfiguracije in jo po potrebi prenese ter poskrbi, da je stanje odjemalca posodobljeno glede na trenutno konfiguracijo.

- Chef je odprtokodno orodje, ki je tako kot Puppet napisano v programskem jeziku Ruby. Uporablja se za konfiguriranje in vzdrževanje Linux ali Windows strežnikov ter podpira integracijo z oblačnimi platformami, kot so Amazon EC2, OpenStack, Google Cloud Platform, Microsoft Azure. Takšna integracija omogoča avtomatsko oskrbovanje in konfiguriranje novih virtualnih instanc znotraj oblaka. Razen delovanja v načinu odjemalec – strežnik lahko deluje tudi v samostojni konfiguraciji. Chef strežnik uporablja konfiguracijske datoteke t. i. recepte (*ang. recipies*), ki jih preko lastnega aplikacijskega vmesnika posreduje odjemalcem. V receptih so npr. opisi, kateri programski paketi morajo biti instalirani na gostitelju in v kakšnem vrstnem redu, kateri servisi naj se na njem izvajajo ali katere datoteke je treba nanj zapisati.

- Ansible je brezplačna programska platforma za konfiguriranje in upravljanje računalnikov ter postavitev, ki so sestavljene iz večjega števila vozlišč ter tudi oblačne infrastrukture, kot je Amazon EC2 in OpenStack. Za svoje delovanje ne potrebuje instalacije lastnega odjemalca, saj vse funkcije posreduje in izvaja preko SSH povezave. Za razliko od orodij Puppet in Chef je Ansible napisan v programskem jeziku Python. Linux gostitelji morajo imeti zaradi tega instaliran Python verzije 2.4 ali več ter Windows gostitelji PowerShell 3.0 ali novejšo verzijo. Enostavne ukaze lahko izvajamo direktno iz glavnega orkestracijskega računalnika in ukazne vrstice. Za bolj kompleksne naloge se uporabljajo konfiguracijske datoteke z imenom *Playbooks*, ki so napisane v programskem jeziku YAML.

Razen zgoraj opisanih orodij so med znanimi orodji za avtomatizacijo in orkestracijo oblačne infrastrukture tudi orodja Salt, Fuel, Vagrant, Juju in TripleO [40].

3 Programsko določena omrežja

Programsko določeno omrežje (*ang. Software Defined Networking – SDN*) je nastajajoča omrežna arhitektura, kjer je kontrolni nivo (*ang. control plane*) omrežja ločen od podatkovnega posredovalnega nivoja (*ang. data/forwarding plane*), ki je neposredno programabilen. Migracija nadzora omrežja, ki je bil prej tesno vezan na posamezne omrežne naprave, v dostopne računalniške naprave omogoča abstrakcijo funkcionalnosti omrežja na nižjih nivojih ter se tako omrežje s stališča aplikacij in omrežnih storitev obravnava kot logična ali virtualna entiteta. Takšen koncept je močno poenostavil načrtovanje in upravljanje omrežij [1].

Nekateri od ključnih razlogov, zakaj je treba znova definirati omrežne koncepte, so našteti v nadaljevanju [2]:

- Spreminjanje vzorcev prometa; znotraj podatkovnih centrov se je vzorec prometa bistveno spremenil. Za razliko od preteklosti, ko je komunikacija potekala predvsem na poti med odjemalcem in strežnikom (t. i. »north – south« način), današnje aplikacije dostopajo do različnih podatkovnih baz in strežnikov ter si tudi med sabo posredujejo podatke (t. i. »east – west« način) in šele na koncu vrnejo podatke odjemalcu. Nemalokrat tako izmenjevanje informacij poteka (t. i. »machine to machine« način) med geografsko distribuiranimi podatkovnimi bazami in strežniki ter skozi javne in zasebne oblake, ki zahtevajo prilagodljivo upravljanje prometa in dostop do pasovne širine na zahtevo.

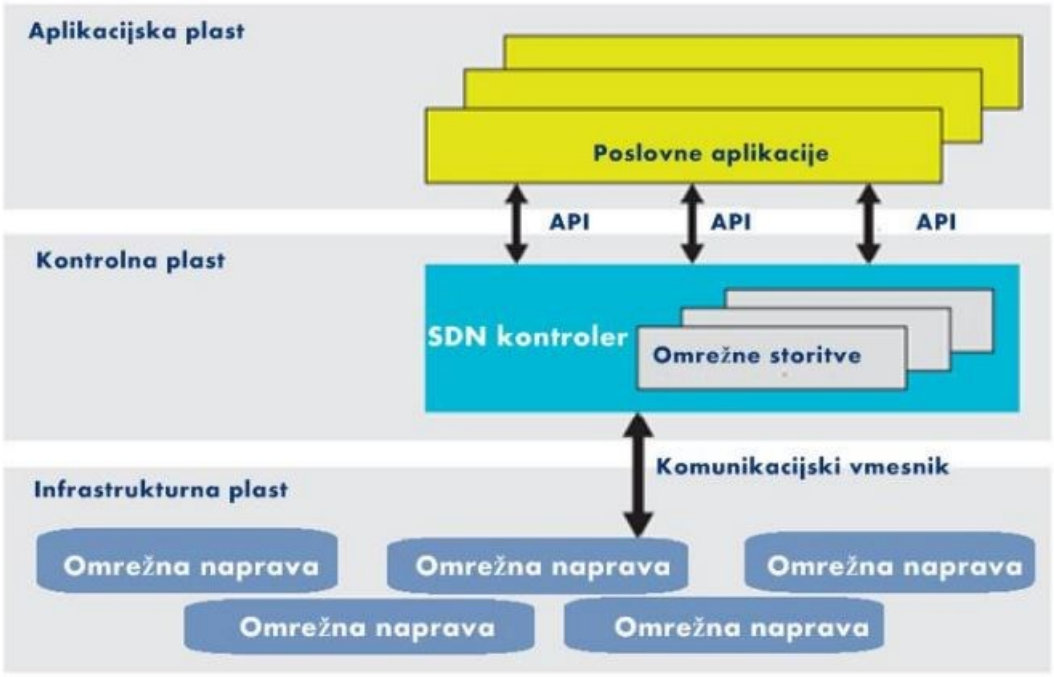
- Novi načini dostopa do poslovnih informacij; uporabniki v službo prinašajo osebne mobilne naprave (*ang. Bring Your Own Device – BYOD*), kot so pametni telefoni, tablice, in želijo tudi z njimi dostopati do poslovnih informacijskih sistemov. Zaradi tega je IT osebje pod pritiskom, kako zagotoviti poslovne informacije tudi na takih napravah in hkrati ohraniti zaupnost poslovnih podatkov. To zahteva prilagodljiva in varna omrežja.

- Rast storitev v oblaku; sodobna poslovna okolja dostopajo do storitev, ki se izvajajo tako v javnih kot zasebnih oblakih. Poslovni uporabniki želijo dostopati do aplikacij in infrastrukture po principu »na zahtevo«. Samopostrežni uporabniški portali (*ang. self-service portals*), iz katerih se lahko na takšen način upravlja in dodeljuje računalniške vire, zahtevajo prilagodljivo skalabilnost procesorskih, podatkovnih in omrežnih virov, po možnosti iz ene skupne točke in z enakimi orodji.

- Masovni podatki (*ang. big data*) zahtevajo večjo prepustnost; upravljanje ogromnih količin podatkov zahteva velikansko procesorsko moč in tisoče med seboj povezanih strežnikov. Zaradi tega je znotraj podatkovnih centrov konstantna potreba po dodatnih

omrežnih virih. Operaterji le-teh so soočeni z dejstvom, da je omrežne vire treba širiti do včasih nepredstavljivih velikosti.

SDN je zelo poenostavil tudi same omrežne naprave, saj ni več treba, da razumejo in procesirajo ogromno število različnih protokolov, ampak le izvedejo ukaze, ki jih sprejmejo od kontrolnega dela. Arhitektura programsko določenih omrežij je prikazana na sliki.



Slika 13: Arhitektura programsko določenih omrežij

Značilnosti programsko določenih omrežij so naslednje:

- Direktna programabilnost; nadzor omrežja je direktno programabilen, ker je ločen od posredovalnih funkcij.

- Agilnost; abstrakcija kontrolnega od posredovalnega dela dopušča skrbnikom dinamično prilagajanje podatkovnih tokov v celotnem omrežju glede na spreminjajoče se potrebe.

- Centralizirano upravljanje; omrežna inteligenca je logično centralizirana v kontrolnem delu, ki zagotavlja globalni pogled na omrežje in le-to se aplikacijam na višjem nivoju predstavlja kot eno samo logično stikalo.

- Programska konfigurabilnost; skrbniki omrežja lahko zelo hitro konfigurirajo, upravljajo in optimizirajo omrežne vire z uporabo dinamičnih avtomatiziranih programov, ki jih lahko sami napišejo, ker programi niso odvisni od lastniške programske opreme posameznih proizvajalcev.

- Odprti standardi, neodvisni od proizvajalcev; uporaba odprtih standardov poenostavlja zasnovo in delovanje omrežja, ker ukazi prihajajo iz kontrolnega dela in ne specifičnih naprav z lastnimi protokoli različnih proizvajalcev [1].

3.1 SDN arhitektura

SDN omogoča, da se aplikacije zavedajo omrežja za razliko od tradicionalnih omrežij, kjer se omrežja zavedajo aplikacij. SDN aplikacije so programi, ki izrecno, neposredno in programsko posredujejo svoje omrežne zahteve in vedenje omrežja SDN kontrolerju. Poleg tega lahko dostopajo do abstraktnega pogleda omrežja in ga uporabijo za svoje interne namene odločanja. Nekaj primerov razlik med tradicionalno omrežno in SDN arhitekturo je naštetih v nadaljevanju [5]:

- Tradicionalne aplikacije lahko le implicitno in posredno sporočajo svoje omrežne zahteve. Zanje je značilno, da vključujejo nekoliko ročnih akcij in več korakov človeške obdelave.

- Tradicionalna omrežja ne morejo podpreti dinamičnih uporabniških zahtev, kot so na primer prepustnost, zakasnitve prenosa ali razpoložljivost. Paketi so sicer lahko označeni z določeno prioriteto, ampak ponudniki storitev tipično ne zaupajo uporabniškim prometnim oznakam. SDN uporabnikom ponuja možnost, da v celoti opredelijo svoje potrebe, ponudniki storitev pa jih temu ustrezno denarno ovrednotijo.

- Tradicionalna omrežja ne sporočajo informacij in omrežnega stanja aplikacijam, ki ga uporabljajo. Znotraj SDN arhitekture SDN aplikacije spremljajo stanje omrežja in se temu ustrezno prilagodijo.

3.1.1 Arhitekturne komponente

V nadaljevanju so opisane SDN arhitekturne komponente in njihove vloge znotraj arhitekture [4].

3.1.1.1 Posredovalni nivo

Posredovalni nivo vsebuje enega ali več omrežnih elementov, od katerih vsak vsebuje vire za posredovanje in procesiranje prometa. Viri so zmeraj abstrakcija fizičnih zmogljivosti ali entitet.

Podatki lahko vstopijo ali zapustijo posredovalni nivo preko fizičnih ali logičnih vrat. Posredovanje in procesiranje prometa upravlja SDN kontroler ali ločeni orkestrirani mehanizmi v povezavi z njim. Posredovalni nivo ne izvaja samostojnih odločitev, ampak samo posredovalne ukaze kontrolerja. Vseeno pa kontrolni nivo lahko konfigurira posredovalni nivo na takšen način, da ta samostojno odgovarja na dogodke, kot so omrežni izpadi, ali podpira funkcije, kot npr. LLDP, STP ali ICMP.

3.1.1.2 Kontrolni nivo

Kontrolni nivo vsebuje niz SDN kontrolerjev, od katerih ima vsak izključne pravice nadzora nad viri, ki jih zagotavlja eden ali več omrežnih elementov posredovalnega nivoja. Sama arhitektura ne določa internega dizajna ali implementacije SDN kontrolerja. Kontroler je logično centraliziran, čeprav ni nujno, da to pomeni obvezno tudi fizično centralizacijo. Zaradi zmogljivosti, razširljivosti ali zanesljivosti je lahko logično centraliziran kontroler distribuiran tako, da več fizičnih instanc sodeluje pri upravljanju omrežja. Več kontrolerjev lahko ima tudi skupen dostop do omrežnih virov pod pogojem, da so kontrolerji med sabo sinhronizirani in da ne pošiljajo neskladnih ali nasprotujočih si ukazov. Informacije, ki si jih kontrolerji izmenjujejo med sabo, naj bi bile sinhronizirane vsaj v t. i. »soft« realnem času *(ang. soft real-time)*. Kontrolne odločitve se sprejemajo glede na trenutni globalni pogled na omrežje in ne glede na izolirane pojave na posameznih omrežnih elementih. SDN kontrolni nivo omogoča popolno kontrolo nad posredovalnim nivojem, kar pomeni, da se v omrežju lahko izvajajo kompleksne in natančne politike z večjim izkoristkom omrežnih virov in garantirano kakovostjo storitev.

3.1.1.3 Aplikacijski nivo

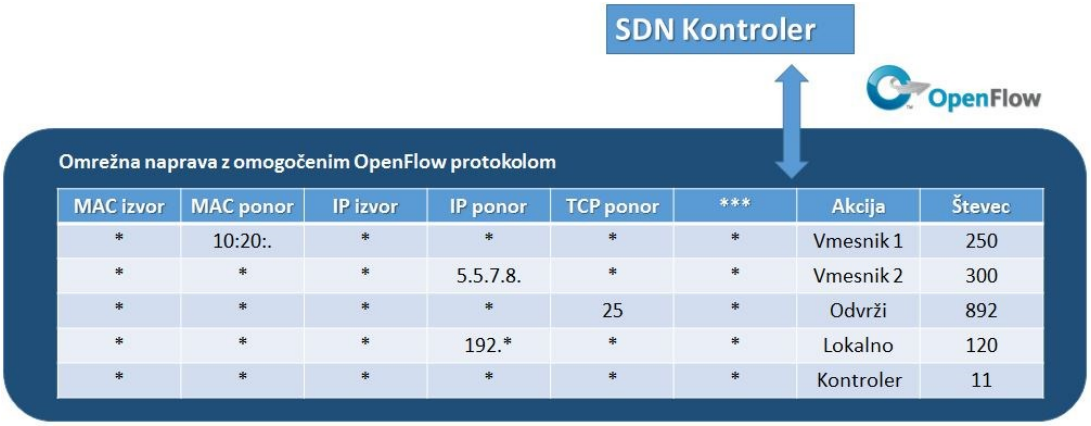
Aplikacijski nivo vsebuje eno ali več aplikacij, od katerih ima vsak izključne pravice nadzora nad viri, ki jih zagotavlja eden ali več SDN kontrolerjev. Tako lahko v skladu s politiko (*ang. policy agreement*) aplikacije izberejo omrežne vire in določijo njihovo obnašanje. SDN aplikacija se lahko sklicuje tudi na druge, zunanje storitve in orkestrira katerokoli število SDN kontrolerjev, da doseže svoje cilje. Na primer Microsoft Lync odjemalci so zmožni sporočiti ali zahtevati takšne karakteristike storitve, ki jih omrežni viri skozi centralizirano koordinacijo tudi zagotovijo.

3.1.1.4 Upravljanje

Upravljanje zajema podporne infrastrukturne naloge, ki naj ne bi bile narejene s strani aplikativnega, kontrolnega ali posredovalnega nivoja. To pomeni tudi izvajanje operacij, ki jih aplikativni, kontrolni ali posredovalni nivo ne smejo izvajati zaradi omrežne politike ali iz drugih razlogov. Primer tega, da SDN komponenti ni dovoljeno izvajanje nekaterih operacij, je SDN kontroler, ki je znotraj strankine domene, medtem ko omrežna politika dovoljuje izvajanje podpornih funkcij le iz domene ponudnika storitev.

3.1.2 OpenFlow

OpenFlow je prvi standardni komunikacijski vmesnik med kontrolnim in usmerjevalnim nivojem znotraj SDN arhitekture. Zunanjim programskim aplikacijam omogoča direkten dostop in manipulacijo usmerjevalnega nivoja tako fizičnih kot virtualnih omrežnih naprav, kot so stikala in usmerjevalniki. Z uporabo koncepta podatkovnih tokov in definiranjem pravil centralni kontroler določi pot prometa skozi omrežne naprave. Pri tem upošteva vzorce uporabe, aplikacije in vire, ki jih ima na razpolago. Zaradi zmožnosti programiranja omrežja za vsak podatkovni tok posebej OpenFlow omogoča zelo natančno upravljanje omrežja v realnem času glede na potrebe aplikacij in uporabnikov. Primer delovanja OpenFlow protokola je prikazan na sliki v nadaljevanju.



MAC izvor	MAC ponor	IP izvor	IP ponor	TCP ponor	***	Akcija	Števec
*	10:20:.	*	*	*	*	Vmesnik 1	250
*	*	*	5.5.7.8.	*	*	Vmesnik 2	300
*	*	*	*	25	*	Odvrži	892
*	*	*	192.*	*	*	Lokalno	120
*	*	*	*	*	*	Kontroler	11

Slika 14: Primer nabora ukazov protokola OpenFlow

OpenFlow se lahko uporablja na obstoječih omrežjih, tako fizičnih kot virtualnih, saj omrežne naprave lahko hkrati podpirajo OpenFlow in tradicionalno usmerjanje. To omogoča dokaj enostavno implementacijo tudi v omrežjih, v katerih je nameščena oprema različnih proizvajalcev [2].

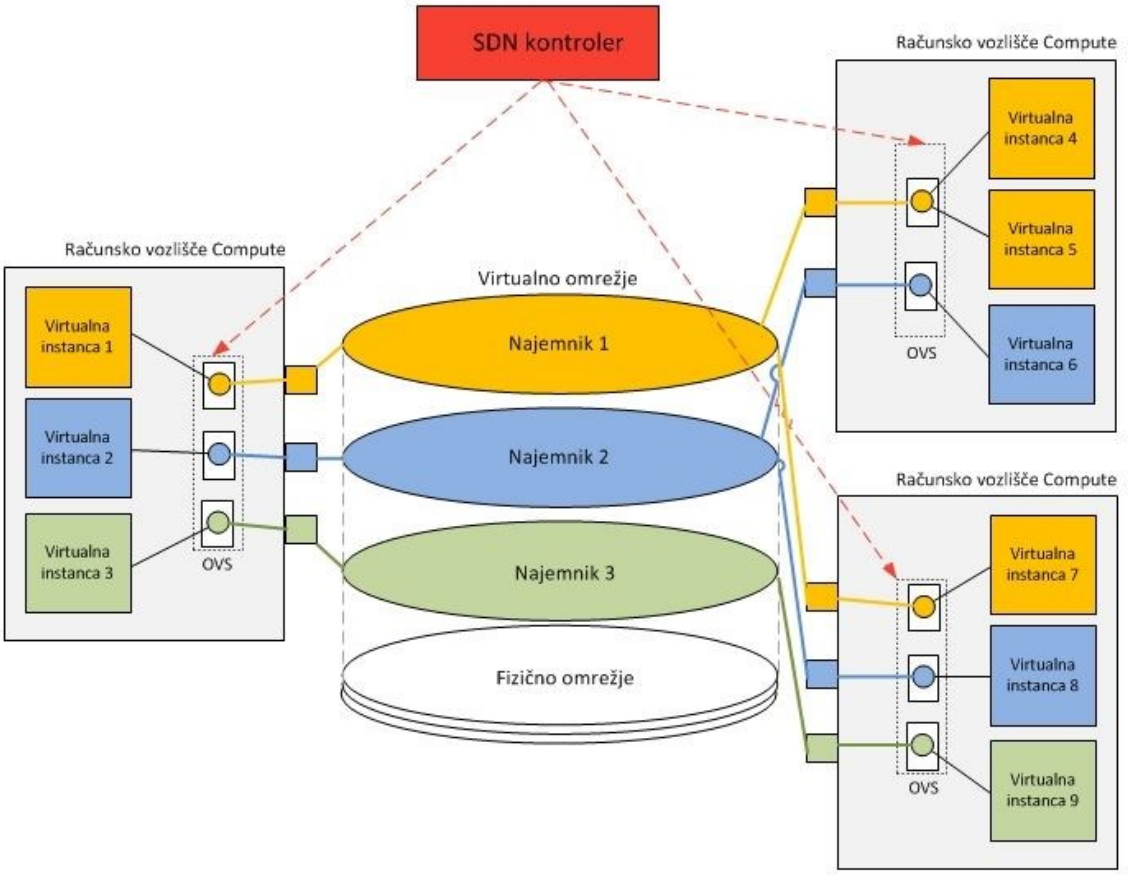
Za OpenFlow standard in razvoj SDN arhitekture skrbi neprofitna organizacija z imenom Open Networking Foundation (ONF). Ustanovljena je bila leta 2011. Sestavljajo jo veliki omrežni operaterji (Deutsche Telekom, NTT, Verizon itd.) ter največja tehnološka podjetja, kot so Google, Facebook, Microsoft, Yahoo! itd.

Novembra leta 2013 je podjetje Cisco predstavilo platformo Application Centric Infrastructure (ACI) kot alternativo SDN konceptu, ki ga zagovarja ONF. Aprila leta 2014 je podjetje Cisco predstavilo še OpFlex protokol kot del ACI platforme in alternativo OpenFlow protokolu. ACI model temelji na t. i. deklarativnem nadzoru ter ne potrebuje centralnega kontrolerja. ACI namreč

le prosi vsak omrežni element, da doseže želeno stanje, in da obljubo, da bo dosegel to stanje, ne da bi elementu natančno povedal, kako to storiti. Cisco to primerja z delovanjem kontrolnega stolpa na letališču. Kontrolorji zračnega prometa dajo samo znak pilotom za vzlet ali pristanek na določenem kraju, dejanska izvedba vzletanja in pristajanja pa je v rokah pilotov. OpFlex je protokol za prenos abstraktne politike v XML ali JavaScript Object Notation (JSON) zapisu od upravljavca politike omrežja do omrežnih elementov [29].

3.2 Omrežna virtualizacija (Network Virtualization – NV)

Omrežna virtualizacija (*ang. Network Virtualization – NV*) omogoča kreiranje logičnih, virtualnih omrežij, ki so ločena od osnovne fizične omrežne opreme z namenom boljše integracije in podpore le-te v virtualnih okoljih. Fizične omrežne naprave v tem primeru skrbijo le za posredovanje paketov, zato so lahko bolj enostavne, cenejše in generične. Zanje se je uveljavil izraz »white box switches«. Virtualna omrežja, ki temeljijo na programski opremi, zagotavljajo abstrakcijo, ki ustvari logičen pogled na strojne in programske omrežne vire in storitve ter njihovo enostavno namestitev in upravljanje. Omrežna virtualizacija znotraj virtualnih okoljih omogoča kreiranje navideznih prekrivnih omrežij (*ang. virtual overlay networks*). S tem zagotavlja popolnoma ločene virtualne omrežne domene in pogoje za delovanje kompleksnih večnajemniških (ang. *multi-tenancy*) okolij [6].

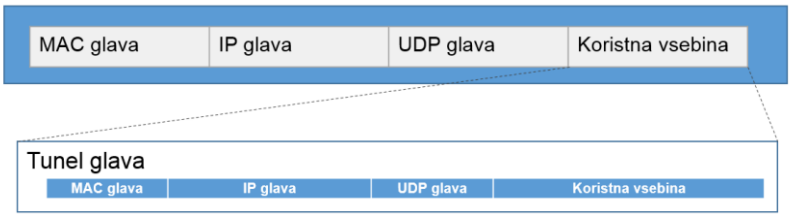


Slika 15: Primer arhitekture omrežne virtualizacije

Dosedanji omrežni mehanizmi in tehnologije, kot so navidezna lokalna omrežja *(ang. virtual local network – VLAN*), navidezno usmerjanje in posredovanje (*ang. virtual routing and forwarding – VRF*), podomrežja, usmerjanje ter varnost, se danes uporabljajo preko mej svojega prvotnega namena in imajo svoje pomanjkljivosti. Na primer VLAN 802.1Q označevanje teoretično podpira le do 4096 navideznih omrežij, navidezna prekrivna omrežja pa 16 milijonov [20]. Problem povezljivosti L2 domen v geografsko razpršenih virtualnih podatkovnih centrih se v navideznih lokalnih omrežjih reši z enkapsulacijo L2 okvirjev (*ang. frame*) v L3 UDP pakete [22]. Tehnologija se v zadnjem desetletju pospešeno uveljavlja v podjetjih, saj ta izkoriščajo prednosti, kot sta učinkovitost in agilnost, ki ju ponuja omrežna virtualizacija.

3.2.1 Navidezna prekrivna omrežja in tunelski protokoli

Navidezna prekrivna omrežja niso možna le znotraj virtualnih okolij, ki se izvajajo v hipernadzorniku, ampak tudi na obstoječi fizični omrežni infrastrukturi. Mehanizem, ki to omogoča, je tuneliranje (*ang. tunneling*), pri tem pa uporablja enkapsulacijo (*ang. encapsulation*). Ko omrežni paket vstopi v robno omrežno napravo navideznega prekrivnega omrežja, omrežna naprava ta paket enkapsulira znotraj novega okvirja *(ang. frame*). Robna omrežna točka, ki zagotavlja prehod z navideznim prekrivnim omrežjem, je virtualna tunelska končna točka (*ang. virtual tunnel endpoint – VTEP*). Tako v primeru virtualnih instanc znotraj oblačne infrastrukture in uporabe virtualnih omrežij takšen enkapsuliran paket potem hipernadzornik posreduje po fizični omrežni infrastrukturi do končne VTEP, kjer se izvede postopek dekapsulacije in potem ustrezno posreduje naprej.



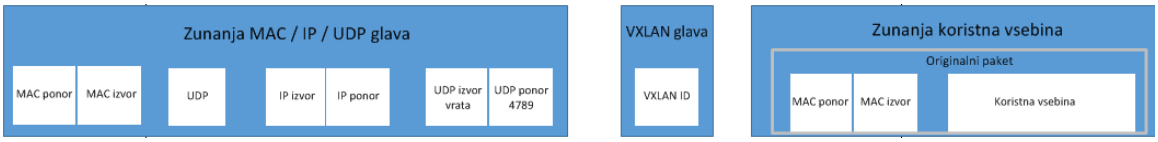
Slika 16: Inkapsulirani okvirji

Končne naprave znotraj navideznega prekrivnega omrežja (npr. virtualne instance, fizični strežniki) se tega procesa niti ne zavedajo in z njihovega stališča gre za standardno L2 povezljivost. VTEP so običajno realizirane na fizični omrežni opremi (npr. namenski prehodi, stikala itd.), od jedra verzije 3.7 pa so podprte tudi na operacijskem sistemu Linux. Pomembno je poudariti, da lahko znotraj enega fizičnega omrežja obstaja več neodvisnih navideznih prekrivnih omrežij hkrati.

Vsi tunelski protokoli temeljijo na mehanizmu, ki se zaradi tega, ker je celoten L2 okvir skupaj z MAC naslovom pošiljatelja enkapsuliran znotraj L3 paketa, imenuje MAC v IP (*ang. MAC in IP*) tuneliranje. V nadaljevanju so našteti nekateri izmed trenutno najbolj razširjenih tunelskih protokolov za tuneliranje omrežnega prometa [56].

3.2.1.1 Navidezna razširljiva lokalna omrežja

Navidezno razširljivo lokalno omrežje (*ang. Virtual Extensible Local Area Network – VXLAN*) je omrežna virtualizacijska tehnologija, ki poskuša odpraviti omejitve tradicionalnih omrežnih tehnologij (npr. skalabilnost, nefleksibilnost znotraj oblačne infrastrukture itd.). VXLAN specifikacijo so primarno razvili v podjetjih VMware, Cisco in Arista Networks. Uporablja mehanizem enkapsulacije L2 okvirjev znotraj L3 UDP paketov (vrata 4789) ter tako omogoča ustvarjanje navideznih L2 omrežij oz. prekrivnih segmentov na vrhu L3 omrežij. Medsebojno lahko komunicirajo samo instance znotraj istega VXLAN segmenta. Vsak VXLAN segment je identificiran s 24 bitnim VNI-jem (*ang. VXLAN Network Identifier*). To omogoča do 16 milijonov logičnih omrežij (VXLAN segmentov) hkrati znotraj iste administrativne domene [30].



Slika 17: Vsebina VXLAN paketa

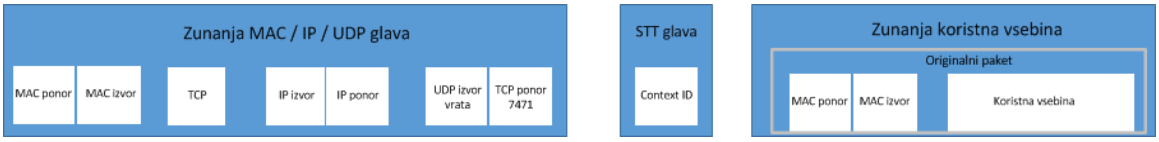
3.2.1.2 Navidezna omrežja z uporabo generične enkapsulacije

Navidezno omrežje z uporabo generične enkapsulacije (*ang. Network Virtualization using Generic Routing Encapsulation – NVGRE*) je omrežna virtualizacijska tehnologija, ki ima podobne lastnosti kot VXLAN tehnologija. Primarno jo je razvilo podjetje Microsoft skupaj s podjetji Intel, HP in Dell. Omogoča ustvarjanje L2 prekrivnih omrežjih in njihovo raztegnitev čez več razpršenih podatkovnih centrov preko L3 omrežij. NVGRE prekrivni segmenti so tudi identificirani s 24 bitnim identifikatorjem (*ang. Virtual Subset ID – VSID*). Uporablja GRE enkapsulacijo (*ang. Generic Routing Encapsulation – GRE*) in GRE glavo (*ang. header*). GRE je ločen in neodvisen IP protokol v istem razredu kot protokola TCP in UDP. Zaradi tega v zunanji glavi NVGRE paketa ni izvornih ali ponornih TCP ali UDP vrat, kar je razvidno iz slike [59].

Slika 18: Vsebina NVGRE paketa

3.2.1.3 Transportno tuneliranje brez stanja

Transportno tuneliranje brez stanja (*ang. Stateless Transport Tunneling – STT*) je tunelski protokol, ki omogoča gradnjo prekrivnih omrežij v navideznih omrežjih. Zasnovalo ga je podjetje Nicira, ki je od leta 2012 del podjetja VMware in njihovih rešitev, namenjenih za virtualna okolja. STT uporablja 64 bitni identifikator (*ang. Context ID*) za razliko od VXLAN in NVGRE, ki uporabljata 24 bitnega, kar mu omogoča veliko večje število prekrivnih segmentov. Čeprav za IP glavo STT uporablja TCP (vrata 7471), je tunelska povezava brez stanja (*ang. stateless*). TCP glava se uporablja iz pragmatičnih razlogov, ker omogoča izkoriščanje funkcij strojnih zmogljivosti obstoječih omrežnih kartic in uporabo mehanizmov LSO (ang. *large segment offload*) in LRO (*ang. large receive offload*). Ta dva mehanizma za povečanje pretoka izhodnega in vhodnega prometa omogočata izvajanje segmentacije paketov na omrežnih karticah namesto na centralni procesni enoti in posledično njeno razbremenitev [60].



Slika 19: Vsebina STT paketa

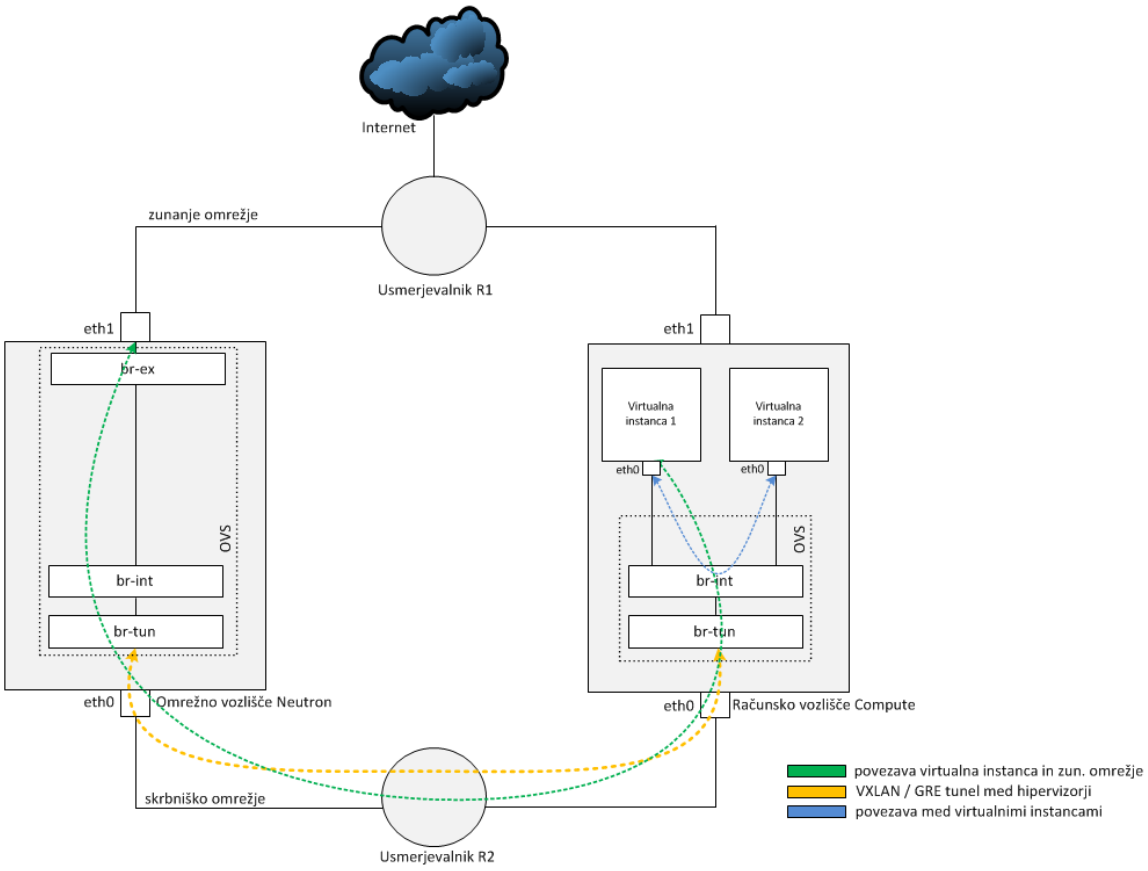
3.2.1.4 Generična navidezna omrežja z enkapsulacijo

Generično navidezno omrežje z enkapsulacijo (*ang. Generic Network Virtualization Encapsulation – GENEVE*) je predlog za novi tunelski protokol, ki združuje funkcionalnosti najbolj popularnih tunelskih protokolov v omrežni virtualizaciji (VXLAN, NVGRE, STT) ter predstavlja nadgradnjo in poenotenje prejšnjih protokolov. Pri predlogu GENEVE protokola sodelujejo podjetja, ki so že bila vključena pri standardizaciji prejšnjih treh najbolj znanih tunelskih protokolov, in sicer: VMware, Intel, Microsoft, Red Hat, Arista Networks itd. GENEVE ima 24 bitni identifikator (*ang. Virtual Network Identifier – VNI*) in za IP glavo uporablja UDP (vrata 6081). Zaradi tega se večini omrežnih naprav predstavlja kot standarden UDP okvir. Tako kot ostali trije tunelski protokoli tudi GENEVE nima nobenih varnostnih mehanizmov in se za komunikacijo preko nezaupnih javnih povezav, kot je internet, priporoča uporaba dodatnih varnostnih protokolov za zaščito komunikacije (npr. IPsec). GENEVE naj bi bil

zasnovan kot prilagodljiv in razširljiv protokol, ki se bo razvijal v skladu s potrebami v prihodnosti [61].

3.2.2 Odprto navidezno stikalo

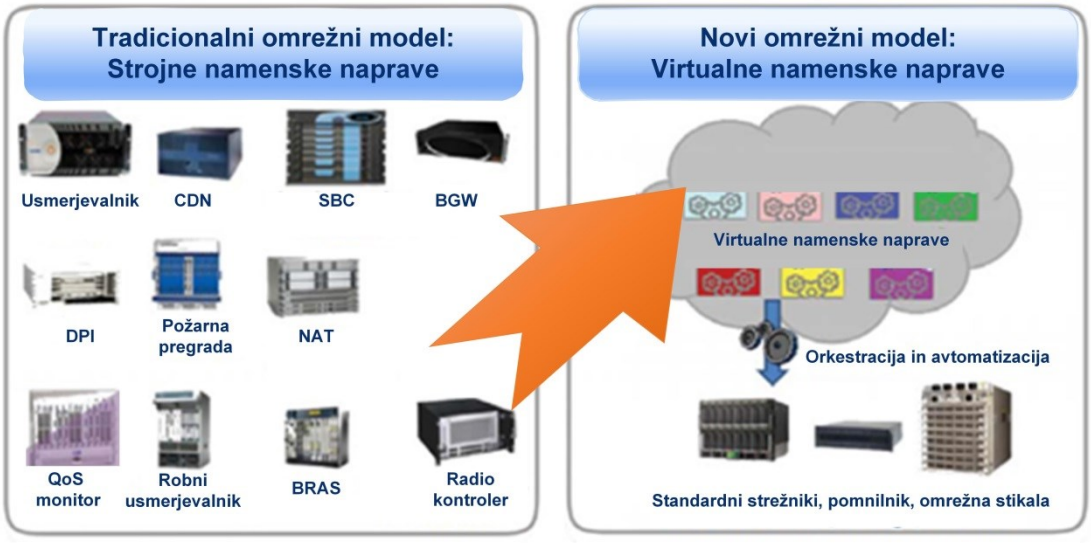
V tradicionalnih omrežjih in podatkovnih centrih se fizični računalniki in strežniki povezujejo z omrežnimi stikali. Z virtualizacijo le-teh je bilo treba omogočiti povezovanje med virtualnimi strežniki ter omogočiti komunikacijo med njimi in fizičnimi strežniki. Povezovanje virtualnih instanc znotraj odprtokodnih hipernadzornikov s fizičnim omrežjem je bilo večinoma zagotovljeno z uporabo vgrajenega Linux L2 mosta (*ang. bridge*). Ker sama zasnova L2 mosta ni ustrezala zahtevam virtualnega večstrežniškega okolja, so v podjetju Nicira v te namene razvili odprto navidezno stikalo (*ang. Open Virtual Switch – OVS*), ki to omogoča. OVS je odprtokodna programska oprema, ki je implementirana v funkciji navideznega večnivojskega omrežnega stikala. Podpira veliko število različnih omrežnih protokolov, kot so: NetFlow, sFlow, LACP, VLAN, LLDP, STP, OpenFlow, GRE, VXLAN, STT, GENEVE, IPsec itd. Poleg tega je podprto delovanje v načinu transparentne distribucije preko več fizičnih strežnikov, tako kot npr. komercialno navidezno stikalo Cisco Nexus 1000V. OVS je integriran v različne oblačne platforme (OpenStack, OpeNebula, CloudStack) ter hipernadzornike (Xen, KVM, VirtalBox, Hyper-V). Od Linux jedra verzije 3.3 so na razpolago uradni paketi za distribucijo Debian, Fedora in Ubuntu. Namesto tradicionalnega načina upravljanja preko protokola SNMP OVS upravljamo in konfiguriramo preko protokola Open vSwitch Database (OVSDB), podatkovne tokove pa preko protokola OpenFlow. Znotraj OpenStack oblačne arhitekture OVS za potrebe omrežnega povezovanja uporablja različne mostove. Za komunikacijo med virtualnimi instancami se most imenuje br-int (*ang. integration bridge*), povezovanje med virtualnimi instancami in hipervizorji na različnih fizičnih strežnikih je zagotovljeno preko mosta z imenom br-tun (*ang. tunnel bridge*), za povezovanje hipernadzornika in fizične omrežne infrastrukture pa preko mosta br-ex (*ang. external bridge*) [68, 69].



Slika 20: Shema OVS povezav v OpenStack oblačni infrastrukturi

3.3 Virtualizacija omrežnih funkcij (Network function virtualization – NFV)

V omrežjih ponudnikov storitev oz. operaterjev je v uporabi veliko število raznolikih lastniških specializiranih strojnih naprav. Začeti ponujati novo omrežno storitev postaja za operaterje zmeraj težje. Soočajo se z vedno večjimi stroški energije, prostora, kompleksnimi integracijami in tudi pomanjkanjem specifičnega znanja, potrebnega za upravljanje lastniških strojnih naprav. Poleg tega naprave, ki temeljijo na strojni opremi, hitro zastarajo. Virtualizacija omrežnih funkcij *(ang. network function virtualization – NFV)* želi rešiti te probleme z uporabo standardnih virtualizacijskih tehnologij in konsolidacijo različnih vrst omrežne opreme v standardizirano industrijsko opremo. Ta oprema so strežniki, stikala, pomnilniške naprave, ki se lahko nahajajo v podatkovnih centrih, omrežnih vozliščih ali v prostorih končnih uporabnikov [24].



Slika 21: Primerjava tradicionalnega in NFV omrežnega pristopa

Nekateri potencialni primeri možnosti uporabe NFV so [24]:

- elementi, ki skrbijo za komutacijo oz. preklapljanje in usmerjanje paketov: usmerjevalniki, širokopasovni prehodi, operaterski prevajalniki omrežnih naslovov,
- funkcije v domačih omrežnih usmerjevalnikih in digitalnih sprejemnikih,
- elementi, ki skrbijo za tuneliranje: IPSec/SSL VPN prehodi,
- analiziranje prometa: DPI in QoS meritve,

- zagotavljanje storitev: spremljanje SLA (*ang. service level agreement*), testiranje in diagnostika,

- NGN (*ang. next generation network*) signalizacija: SBC (*ang. session border controller*), IMS (*ang. IP Multimedia Subsystem*),

- varnostne funkcije: požarne pregrade, antivirusni programi, sistemi za odkrivanje vdorov, SPAM zaščita.

Konkretni primeri uporabe NFV funkcij so podrobno opisani v dokumentu NFV Use Cases [26].

NFV in SDN sta zelo komplementarni, ne pa tudi soodvisni. NFV je lahko implementirana tudi brez SDN, v kombinaciji pa potencialno ponujata večjo dodano vrednost. NFV cilji se lahko dosežejo tudi brez uporabe SDN mehanizmov z uporabo trenutnih tehnologij v podatkovnih centrih. Principi, ki jih ponuja SDN (ločitev kontrolnega in posredovalnega nivoja), lahko izboljšajo učinkovitost, poenostavijo združljivost z obstoječimi postavitvami in olajšajo postopke obratovanja in vzdrževanja. NFV pa je zmožna podpirati SDN tako, da zagotavlja infrastrukturo, na kateri se SDN programska oprema izvaja [25].

Za NFV standard skrbi skupina NFV ISG (Network Function Virtualisation Industry Specification Group), ki je pod okriljem Evropskega inštituta za telekomunikacijske standarde (ETSI). V skupini je trenutno 235 podjetij, vključujoč 37 največjih svetovnih operaterjev. Prva NFV bela knjiga (*ang. white paper*) je bila objavljena oktobra leta 2012. Trenutno je aktualna tretja verzija dokumenta iz oktobra leta 2014 [28].

4 Možnosti uporabe SDN in NFV v omrežjih ponudnikov storitev

Ponudniki storitev in operaterji menijo, da bosta SDN in NFV bistveno spremenila njihovo omrežno arhitekturo in zagotovila koristi v smislu novih storitev in prihodkov, operativno učinkovitost ter prihranke pri naložbah v osnovna sredstva. Kot glavne razloge za uvajanje SDN tehnologij navajajo:

- poenostavitev in avtomatizacijo oskrbovanja (*ang. provisioning*) storitev, kar omogoča agilno prilagajanje storitev in hitro povračilo naložb,

- poenostavitev in avtomatizacijo oskrbovanja omrežnih storitev s ciljem globalnega pogleda na stanje omrežja ne glede na proizvajalca omrežne opreme, omrežni nivo, tehnologijo ali tip opreme (mobilno jedrno omrežje, DSLAM, usmerjevalniki, stikala, itd.),

- avtomatizacijo storitev, ki je ponavadi narejena z uvedbo spletnega portala za stranke (*ang. self care portal*), kar jim omogoča naročanje storitev, njihovo oskrbovanje, obračunavanje, spremljanje stanja ter tudi upravljanje.

Ponudniki storitev in operaterji praviloma začenjajo z manjšimi SDN postavitvami, ki pokrivajo le dele njihovega omrežja. Na takšen način želijo preveriti, ali tehnologija deluje tako, kot je bilo predvideno. Časovni pregled uvajanja SDN in NFV tehnologij pri operaterjih je prikazan v nadaljevanju:

- 2013 je bilo leto preverjanja konceptov (*ang. proof of concept*). Operaterji so v svojih testnih okoljih preverjali različne primere uporabe SDN in NFV tehnologij.

- Leta 2014 je prišlo do prvih dejanskih testiranj na terenu.

- V letu 2015 so se zgodile prve manjše komercialne postavitve, kjer so operaterji v lastnih omrežjih in realnih razmerah preverjali delovanje tehnologij.

- Od leta 2016 do 2020 se pričakuje še več komercialnih postavitev.

Ne glede na pozitiven trend uvajanja SDN in NFV tehnologij v bližnji prihodnosti še ni pričakovati postavitev, ki bi pokrivale večje dele ali kar celotna omrežja. Glede na predvidevanja analitskih hiš bo uvajanje oziroma prehod na SDN trajal še veliko let.

Med ključnimi ovirami hitrejše uvedbe SDN operaterji izpostavljajo nezrelost tehnologije in integracijo z obstoječimi omrežji in procesi. Pred uvedbo novih tehnologij se želijo prepričati, da te ponujajo enake zmogljivosti in zanesljivost, kot npr. obstoječa tradicionalna omrežja, ter razumeti, kaj vse morajo spremeniti pri svoji omrežni arhitekturi in procesih, da lahko izkoristijo prednosti, ki jih prinaša SDN.

Kot glavne razloge za uvajanje NFV tehnologij ponudniki storitev in operaterji navajajo:

- agilnost storitev ter posledično nove vire prihodkov,
- izboljšanje operativne učinkovitosti ter posledično zmanjševanje stroškov poslovanja (*ang. operating expense – OPEX*),
- uporaba standardnih komercialnih strežnikov namesto specialne namenske omrežne opreme ter posledično zmanjševanje investicij (*ang. capital expense – CAPEX*).

Za najbolj primerne primere uporabe NFV tehnologije v svojih omrežjih v naslednjih letih operaterji vidijo:

- upravljanje storitev s pomočjo programske opreme in virtualnih omrežnih funkcij *(ang. virtual network functions – VNF*), kot so npr. požarne pregrade sistemov za preprečevanje vdorov, optimizacijo prostranega omrežja (*ang. wide area network – WAN*) itd.,
- veriženje storitev (*ang. service chaining*), kar omogoča horizontalno usmerjanje omrežnega prometa čez podatkovne centre in zahtevane virtualne omrežne funkcije, ki se izvajajo na navideznih računalnikih namesto na lokalni namenski omrežni opremi,
- omrežno platformo kot storitev, na kateri lahko podjetja namestijo aplikacije, virtualne omrežne funkcije in zgradijo lastna virtualna omrežja,
- virtualne robne usmerjevalnike (*ang. virtual edge routers – vPE*) ali standardne virtualne usmerjevalnike.

Kot ključne ovire uvajanja NFV tehnologij pa ponudniki storitev izpostavljajo:

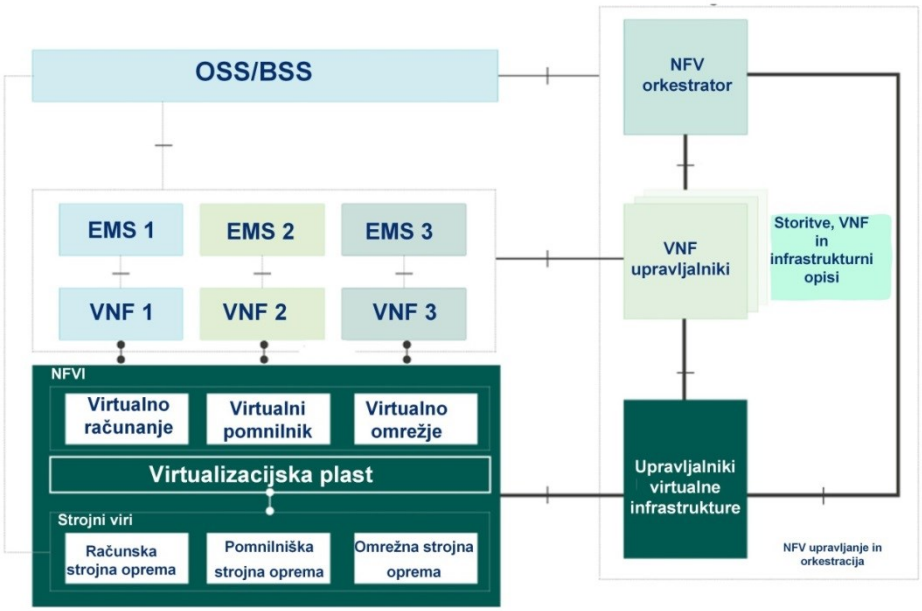
- produkti niso dovolj zreli in zanesljivi (*ang. carrier grade*) za produkcijsko delovanje v omrežjih operaterjev,

- pomanjkanje strokovnjakov na tem področju ter usposabljanja osebja,

- pomanjkanje sistemov za podporo poslovnim procesom in sistemov za operativno podporo procesom (*ang. Business/Operational Support Systems – BSS/OSS*) na področju NFV,

- neznani celotni stroški lastništva (*ang. total cost of ownership – TCO*), kar posledično pomeni nejasnosti pri poslovnih modelih in izračunih upravičenosti uvedbe NFV tehnologij v komercialne namene.

Odprtokodna oblačna platforma OpenStack je identificirana kot ključna komponenta v ETSI NFV arhitekturnem ogrodju. ETSI in Linux Foundation projekt OPNFV sta definirala specifikacije in referenčno platformo za NFV, v kateri je OpenStack izbran za virtualnega infrastrukturnega upravljalnika (*ang. virtualization infrastructure manager – VIM*). Njegova vloga je, da skrbi za upravljanje življenjskega cikla računskih, pomnilniških in omrežnih virov, ki jih zagotavlja NFV infrastruktura. VIM je del funkcije za upravljanje in orkestracijo (*ang. management and orchestration – MANO*), ki krmili dodelitev virtualnih računskih, pomnilniških in omrežnih virov NFV infrastrukture za podporo VNF [26, 53, 54, 55].



Slika 22: ETSI NFV arhitektura

5 Opis praktičnega dela

Kot je bilo že predstavljeno v prejšnjih poglavjih, je na razpolago več različic odprtokodnih oblačnih platform. Za potrebe priprav magistrske naloge je postavljeno testno okolje na odprtokodni oblačni platformi OpenStack. Uporabljena je bila različica Liberty, ki je bila aktualna v času izvedbe praktičnega dela magistrske naloge. Osnovna postavitev oblačne platforme OpenStack je bila narejena na operacijskem sistemu Ubuntu 14.04 LTS ter z uporabo instalacijskih skript DevStack. Te omogočajo hitrejšo in bolj optimalno postavitev kompletnega razvojnega OpenStack okolja, ki je primarno namenjeno za potrebe testiranja.

5.1 Opisi omrežnih arhitektur

OpenStack komponenta Neutron omogoča kreiranje in upravljanje omrežnih elementov (omrežja, podomrežja, vrata, usmerjevalniki itd.), ki jih lahko uporabljajo drugi OpenStack servisi. Neutron zagotavlja API, ki omogoča definiranje različnih omrežnih scenarijev in storitev znotraj oblačne infrastrukture. Vsebuje naslednje komponente:

- API strežnik, ki podpira L2 omrežja in upravljanje IP naslovov, kakor tudi razširitev za L3 povezljivost in usmerjanje med L2 omrežji in prehodi do zunanjih omrežij. Podprti so tudi programski vtičniki, ki omogočajo interoperabilnost z različnimi odprtokodnimi in komercialnimi omrežnimi tehnologijami ter omrežnimi elementi.

- Omrežni programski vtičniki in agenti, ki skrbijo za kreiranje vrat, omrežij, podomrežij ter zagotavljajo IP naslavljanje.

- Sporočilna čakalna vrsta, ki sprejema in usmerja klice za oddaljene postopke (*ang. remote procedure call – RPC*) med agenti za izvajanje API operacij.

Med primarnimi funkcijami OpenStack omrežne komponente Neutron je zagotavljanje povezljivosti:

- med virtualnimi instancami v oblačni infrastrukturi,
- do virtualnih instanc v in zunaj oblačne infrastrukture oz. iz zunanjega omrežja in fizičnih omrežnih komponent ter resničnih računalnikov in strežnikov,
- od virtualnih instanc v in zunaj oblačne infrastrukture oz. do zunanjega omrežja in fizičnih omrežnih komponent ter resničnih računalnikov in strežnikov.

OpenStack Neutron omogoča, da ima vsak najemnik več zasebnih omrežij ter da najemniki izberejo svojo lastno IP naslavljanje tudi v primeru, če se IP naslovi prekrivajo s tistimi, ki jih drugi najemniki uporabljajo.

OpenStack podpira sledeče načine izolacije omrežij in prekrivnih tehnologij:

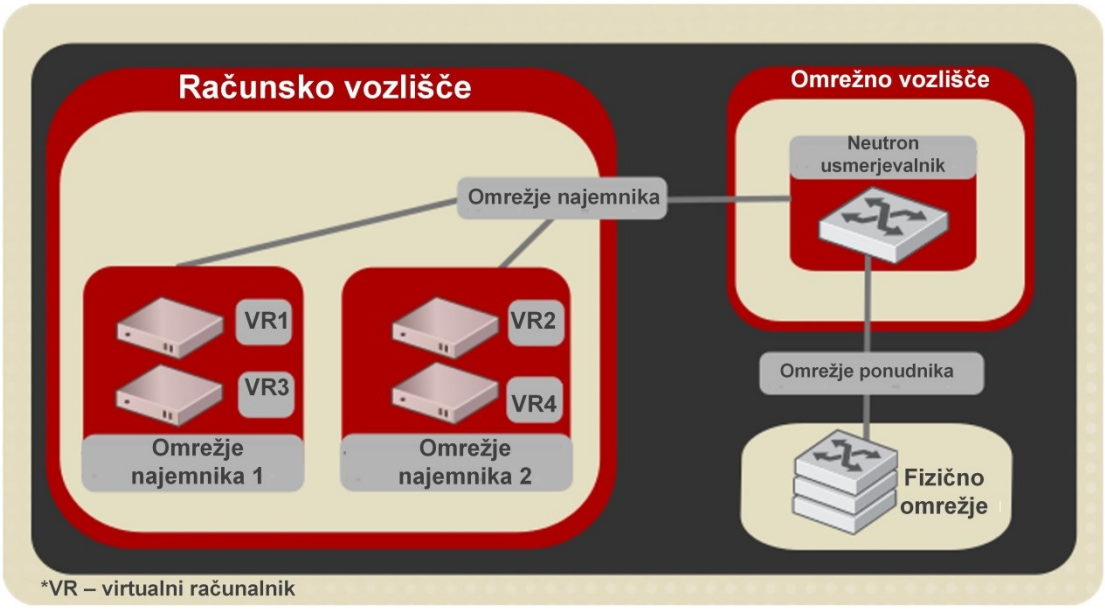
- Osnovno ali enostavno omrežje (*ang. flat network*); vse virtualne instance se nahajajo v istem omrežju, v katerem se ne uporablja označevanje paketov (*ang. vlan tagging*). Omrežje si najemniki med sabo lahko delijo.

- VLAN omrežje; podprto je kreiranje več omrežij (ponudnikov ali najemnikov) z uporabo označevanja paketov, ki ustrezajo VLAN omrežjem, prisotnim v fizičnem omrežju. To omogoča virtualnim instancam medsebojno povezljivost znotraj kompletnega okolja in tudi povezljivost z vsemi drugimi fizičnimi omrežnimi napravami ter resničnim strežnikom in računalnikom znotraj istega L2 VLAN omrežja.

- GRE in VXLAN omrežja; omogočeno je kreiranje omrežij z uporabo tunelskih protokolov in prekrivnih omrežij z namenom kontrole povezljivosti med virtualnimi instancami. Za komunikacijo takšnih omrežij najemnika izven lastnega omrežja ali za komunikacijo z zunanjimi omrežji je potreben Neutron usmerjevalnik. Ta skupaj z uporabo plavajočih IP naslovov zagotavlja tudi povezljivost do virtualnih instanc v oblaku iz zunanjega omrežja ali interneta ter resničnih strežnikov in računalnikov.

Obstajata dve vrsti omrežij, ki jih je mogoče definirati v Neutronu:

- omrežja ponudnika (*ang. provider networks*),
- omrežja najemnika (*ang. tenant networks*).



Slika 23: Prikaz omrežij ponudnika in omrežij najemnika v OpenStack platformi

5.1.1 Omrežja ponudnika

Omrežja ponudnika oziroma zunanja omrežja (*ang. external networks*) so omrežja, ki jih ustvarja OpenStack skrbnik in so neposredno povezana z obstoječim fizičnim omrežjem v podatkovnem centru ter resničnimi strežniki in računalniki. Do teh omrežij je zagotovljeno omrežno usmerjanje znotraj obstoječega fizičnega omrežja v podatkovnem centru. V takih primerih se običajno uporablja L2 ali VLAN omrežja. Možno pa je za omrežja ponudnika uporabiti tudi prekrivna omrežja in tunelske protokole, kot so npr. GRE ali VXLAN, ki se jih potem preslika na fizično omrežje z uporabo programskih ali fizičnih prehodov.

5.1.2 Omrežja najemnika

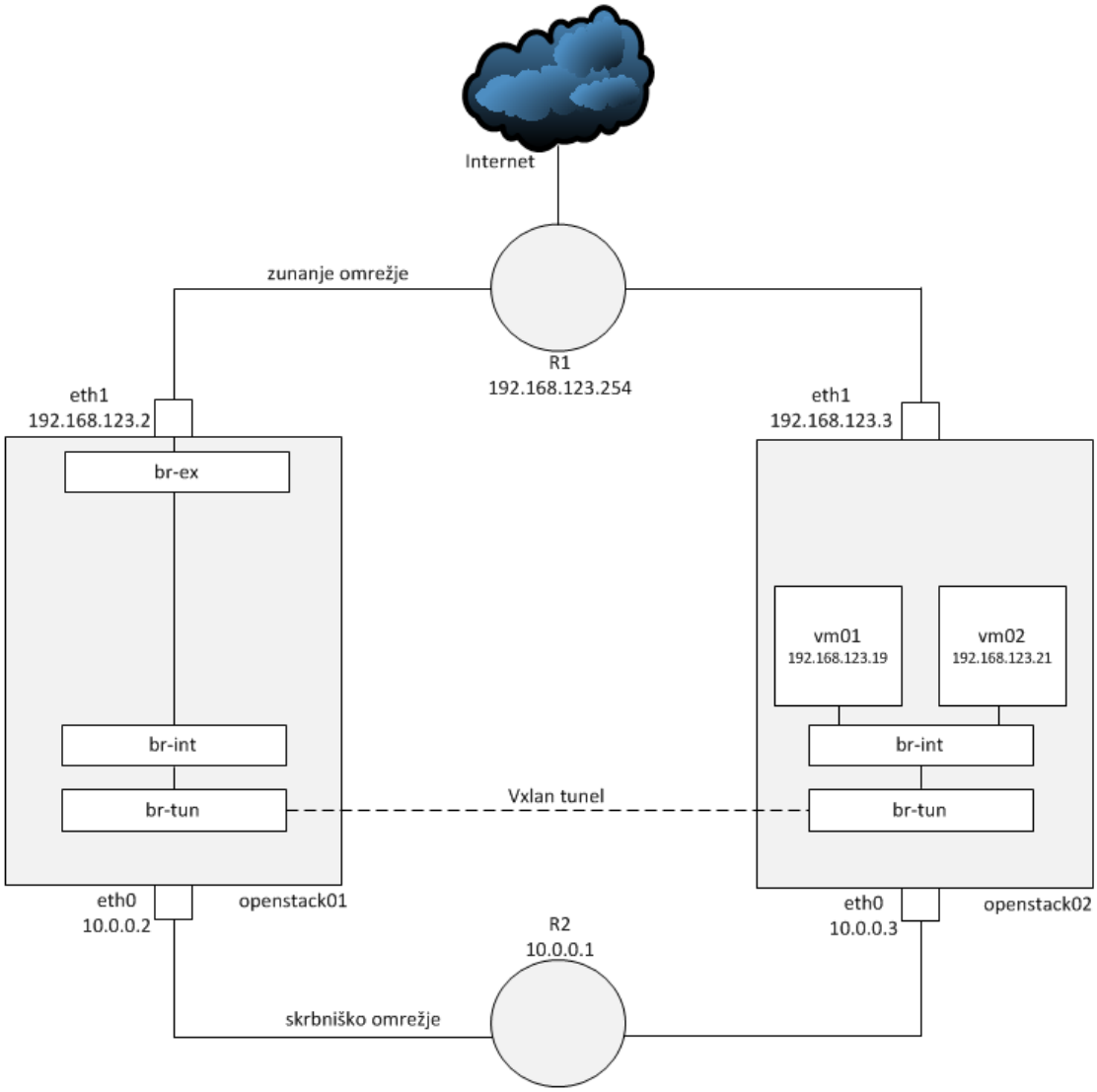
Omrežja najemnika oziroma omrežja projektov (*ang. project networks*) so omrežja, ki jih ustvarjajo najemniki oblačne infrastrukture. Privzeto so med sabo popolnoma izolirana (VLAN, GRE, VXLAN) in se ne delijo z drugimi projekti. Za povezavo do omrežij drugih najemnikov in zunanjih omrežij, v katerih se nahajajo resnični strežniki in računalniki, se zanašajo na Neutron usmerjevalnike. Najemniki se običajno ne zavedajo, kako je postavljeno fizično omrežje.

5.2 Opisi tehničnih možnostih povezovanja navideznih in resničnih računalnikov

Za postavitev oblačne infrastrukture sta v prvih treh primerih bila uporabljena po dva OpenStack vozlišča ter v četrtem in petem primeru zaradi dodatnih tehničnih zahtev po tri OpenStack vozlišča. V šestem primeru je bilo uporabljeno le eno OpenStack vozlišče. Na vsakem OpenStack vozlišču sta bila uporabljena po dva omrežna vmesnika. Prvi omrežni vmesnik (eth0) zagotavlja skrbniški dostop do oblačne infrastrukture ter tudi navidezno prekrivno tunelsko omrežno povezavo med OpenStack vozlišči. Drugi omrežni vmesnik (eth1) zagotavlja povezljivost do fizične omrežne infrastrukture, na kateri so omrežno povezani resnični strežniki in računalniki. Osnovna konfiguracija posamezne postavitve oblačne infrastrukture se izvede z zagonom DevStack skripte, ki prebere OpenStack nastavitve iz konfiguracijske datoteke z imenom »local.conf«. Vsako OpenStack vozlišče ima svojo konfiguracijsko datoteko. Vse konfiguracijske datoteke so priložene v poglavju priloge.

5.2.1 Postavitev oblačne infrastrukture z L2 povezavo v omrežje ponudnika brez označevanja paketov

Postavitev opisuje in prikazuje implementacijo osnovne L2 povezljivosti oblačne infrastrukture in zunanjega omrežja brez označevanja paketov. Pri tem se vse virtualne instance nahajajo v istem L2 omrežju ponudnika. Povezljivost do zunanjega omrežja, na katerem se nahajajo resnični računalniki, je zagotovljena znotraj istega L2 omrežja ter zaradi tega takšna postavitev ne uporablja koncepta plavajočih IP naslovov. Omrežja ponudnika lahko upravlja le skrbnik oblačne infrastrukture, saj njihova konfiguracija zahteva tudi konfiguracijo in poseg v fizično omrežno infrastrukturo. Za postavitev sta uporabljeni dve OpenStack vozlišči. Vozlišče z imenom »openstack01« zagotavlja vse potrebne omrežne storitve ter L2 povezavo virtualnih instanc z zunanjim omrežjem in resničnimi računalniki. Vozlišče z imenom »openstack02« je kontrolno in računsko vozlišče ter zagotavlja vse preostale potrebne OpenStack servise. Opisana postavitev je primerna predvsem za manjša okolja, v katerih ni potrebe po kompleksnih omrežnih topologijah in povezavah. Oblačna infrastruktura v tem primeru s takšnim omrežnim modelom povezovanja le nadaljuje. Usmerjanje omrežnega prometa virtualnih in fizičnih računalnikov se izvaja na fizičnih usmerjevalnikih, ki so del fizičnega omrežja.



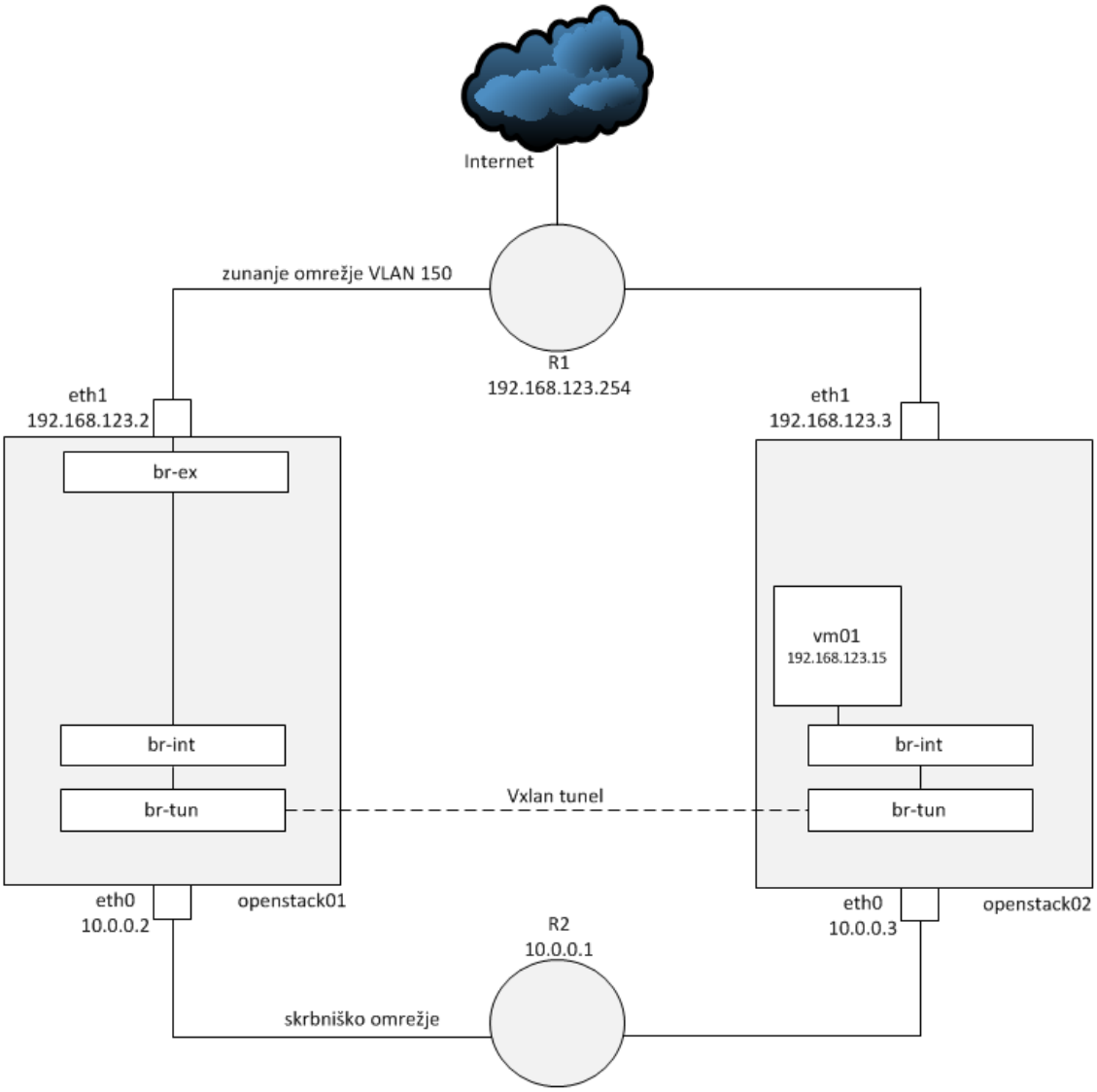
Slika 24: Shema L2 povezave v omrežje ponudnika brez označevanja paketov

Izvajanje DevStack skripte poskrbi za postavitev osnovnega okolja oblačne platforme. Virtualne instance se doda naknadno s serijo konzolnih ukazov, potem ko se preveri uspešnost instalacije in postavitve želene oblačne infrastrukture.

Preverjanje uspešnosti kreiranja infrastrukture oblačne platforme je opisano v poglavju 7.1.5. Kreiranje virtualnih instanc in preverjanje je opisano v poglavju 7.1.6. V zadnjem koraku se povežemo preko SSH povezave na virtualno instanco z imenom »vm01« in s ping ukazom preverimo L2 dosegljivost omrežnih elementov in resničnih računalnikov izven oblačne infrastrukture ter tudi L2 dosegljivost virtualne instance z imenom »vm02«. Ker je privzeto ves dohodni omrežni promet do virtualnih instanc onemogočen, je pred tem še treba dodati varnostna pravila, ki omogočajo ICMP in SSH. Postopek je opisan v poglavju 7.1.7.

5.2.2 Postavitev oblačne infrastrukture z L2 povezavo v omrežje ponudnika z uporabo označevanja paketov VLAN

Postavitev opisuje in prikazuje implementacijo L2 povezljivosti oblačne infrastrukture in zunanjega omrežja z uporabo označevanja paketov VLAN. Pri tem se vse virtualne instance nahajajo v istem L2 omrežju ponudnika. Povezljivost do zunanjega omrežja, na katerega so priključeni resnični računalniki, je zagotovljena znotraj istega L2 (VLAN) omrežja ponudnika ter tako kot v prejšnjem primeru takšna postavitev ne uporablja koncepta plavajočih IP naslovov. Tudi za to postavitev sta uporabljeni dve OpenStack vozlišči. Vozlišče z imenom »openstack01« zagotavlja vse potrebne omrežne storitve ter L2 povezavo virtualnih instanc z zunanjim omrežjem. Vozlišče z imenom »openstack02« je kontrolno in računsko vozlišče ter zagotavlja vse preostale potrebne OpenStack servise. Opisana postavitev je primerna za okolja, v katerih se že uporabljajo L2 mehanizmi izolacije omrežnega prometa z uporabo virtualnih lokalnih omrežij. V tem primeru so vsi morebitni najemniki oblačne infrastrukture del istega lokalnega omrežja. Komunikacija s fizičnim omrežjem in resničnimi računalniki v drugih L2 omrežjih je zagotovljena preko fizičnih usmerjevalnikov v zunanjem omrežju.



Slika 25: Shema L2 povezave v omrežje ponudnika z uporabo označevanja paketov VLAN

Uporabo L2 VLAN povezave do zunanjega omrežja in posledično do resničnih računalnikov, ki so nanj priključeni, se omogoči znotraj DevStack konfiguracijske datoteke:

```
PROVIDER_SUBNET_NAME="provider_net"
PROVIDER_NETWORK_TYPE="vlan"
SEGMENTATION_ID=150
```

Način kreiranja virtualnih instanc je enak postopku v prejšnjem primeru in je opisan v poglavju 7.2.5. Pri kreiranju virtualne instance sama informacija o VLAN ID-ju ni razvidna.

Razen uporabe enega VLAN omrežja kot načina povezave oblačne infrastrukture z zunanjim omrežjem in resničnimi računalniki je mogoče uporabiti tudi več različnih VLAN omrežij hkrati in t. i. povezovalno omrežje (*ang. trunk network*). Takšna postavitev se uporablja predvsem takrat, kadar želimo tudi virtualne instance med seboj izolirati in dodeliti v različna VLAN omrežja na enak način, kot je to že narejeno v fizičnem omrežju z resničnimi računalniki. Virtualne instance in resnični računalniki, ki so v istem VLAN omrežju, se povezujejo direktno brez uporabe usmerjevalnika. Komunikacija med virtualnimi instancami ali resničnimi računalniki v različnih VLAN omrežjih pa poteka preko fizičnega usmerjevalnika v fizičnem omrežju. Postopek kreiranja dodatnega VLAN omrežja ponudnika je opisan v poglavju 7.2.6.

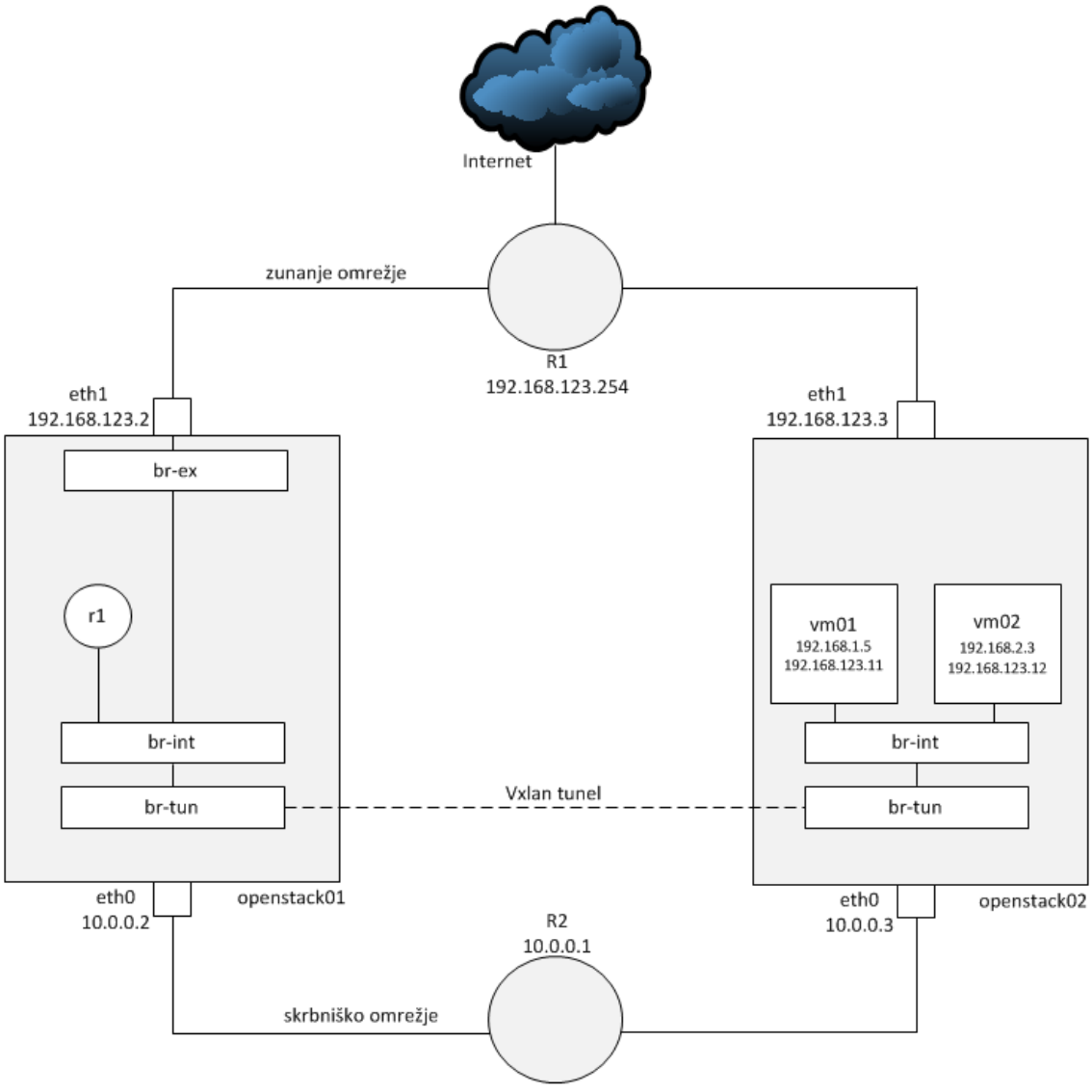
Kreiranje podomrežja in virtualnih instanc se naredi po standardnem postopku kot v primeru uporabe omrežja brez uporabe označevanja paketov. Virtualne instance se ob kreiranju dodeli v različno VLAN omrežje ponudnika tako, da se znotraj ukaza `nova boot` uporabi parameter želenega omrežja ponudnika. V primeru npr. za VLAN 160 je to:

```
--nic net-id=9a931032-c554-7842-8975-964206f3210f
```

5.2.3 Postavitev oblačne infrastrukture z L3 povezavo v omrežje ponudnika in uporabo VXLAN najemniških omrežij

Postavitev opisuje in prikazuje implementacijo L3 povezljivosti oblačne infrastrukture in zunanjega omrežja. Pri takšni postavitvi virtualne instance imajo IP naslove v različnem L2 omrežju od omrežja ponudnika. Povezljivost do zunanjega omrežja in resničnih računalnikov je zagotovljena preko koncepta plavajočih IP naslovov, ki so znotraj bazena IP naslovov L2 omrežja ponudnika. Prikazana sta dva primera. Virtualne instance so v prvem primeru postavitve, ki so v ločenih podomrežjih in VXLAN segmentih povezane s skupnim virtualnim usmerjevalnikom, ki jih povezuje z zunanjim omrežjem in resničnimi računalniki. Takrat je med virtualnimi instancami omogočena povezljivost preko internih IP naslovov. Do zunanjega omrežja in resničnih računalnikov je povezljivost zagotovljena preko plavajočih IP naslovov. Drugi primer pokaže postavitev, ko virtualne instance v ločenih podomrežjih in VXLAN omrežjih ter tudi za povezljivost do zunanjega omrežja in resničnih računalnikov uporabljajo ločene virtualne usmerjevalnike. V tem primeru med virtualnimi instancami ni povezljivosti preko internih IP naslovov in je komunikacija zagotovljena tako, kot do zunanjega omrežja in resničnih računalnikov, in sicer le z uporabo plavajočih IP naslovov znotraj istega L2 omrežja. Za postavitev sta uporabljeni dve OpenStack vozlišči. Vozlišče z imenom »openstack01« zagotavlja vse potrebne omrežne storitve ter L3 povezavo virtualnih instanc z zunanjim omrežjem in resničnimi računalniki. Vozlišče z imenom »openstack02« je kontrolno in računsko vozlišče ter zagotavlja vse preostale potrebne OpenStack servise. Opisana postavitev je primerna za večja okolja, v katerih je treba zagotoviti potrebe po skalabilnosti. Ta pa je v primerih prejšnjih postavitev in uporabe L2 omrežij ponudnika omejena. V primeru uporabe L2 povezave oblačne infrastrukture v omrežje ponudnika brez označevanja paketov je treba zagotoviti ustrezne omrežne mostove, v primeru uporabe označevanja paketov VLAN pa je treba ročno konfigurirati

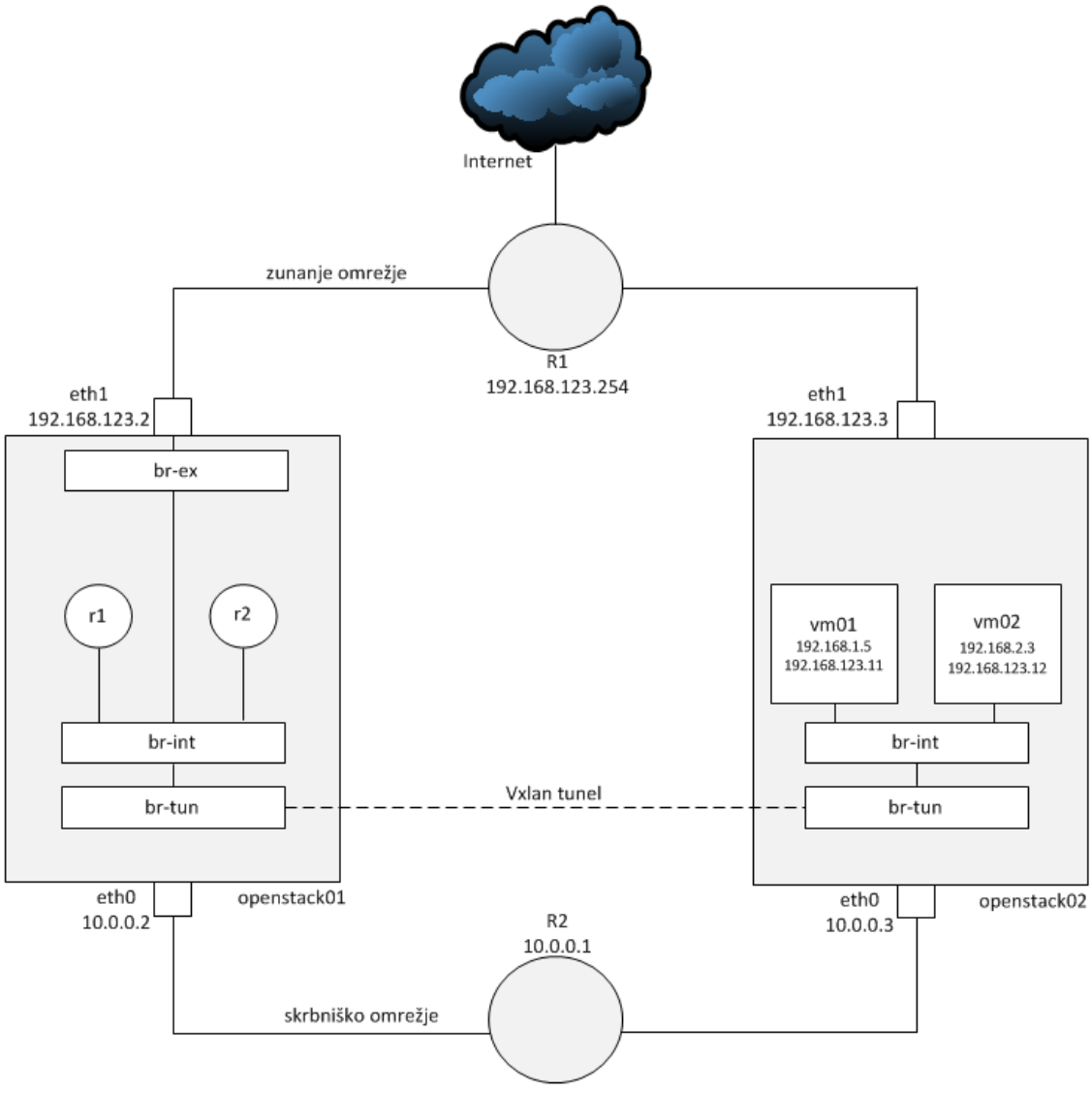
omrežna stikala in prehode. Kot je že napisano v prejšnjem poglavju, se takrat za usmerjanje v zunanja omrežja in internet uporablja zunanje fizične usmerjevalnike in požarne pregrade. Ti takrat izvajajo tudi SNAT/DNAT prevajanje omrežnih naslovov. Ker pa je L3 povezava oblačne infrastrukture in zunanjega omrežja zagotovljena z virtualnimi usmerjevalniki, to pomeni, da skrbniki in najemniki oblačne infrastrukture lahko sami upravljajo, konfigurirajo in glede na potrebe omrežne povezljivosti tudi kreirajo dodatne virtualne usmerjevalnike. To pa omogoča hitrejšo in bolj prilagodljivo infrastrukturo in okolje.



Slika 26: Shema L3 povezave v omrežje ponudnika z enim skupnim navideznim usmerjevalnikom

Preverjanje obstoječe ter kreiranje dodatne infrastrukture je opisano v poglavju 7.3.5. Povezljivost do virtualne instance »vm02« preko internih in zunanjih IP naslovov ter tudi povezljivost do omrežnih elementov in resničnih računalnikov izven oblačne infrastrukture preverimo po postopku, opisanem v poglavju 7.3.6.

V drugem primeru se postavitev razlikuje le v tem, da virtualni instanci »vm01« in »vm02«, ki sta v ločenih podomrežjih in VXLAN segmentih, namesto skupnega virtualnega usmerjevalnika za povezljivost z omrežjem ponudnika in resničnimi računalniki uporabljata ločene zasebne virtualne usmerjevalnike.

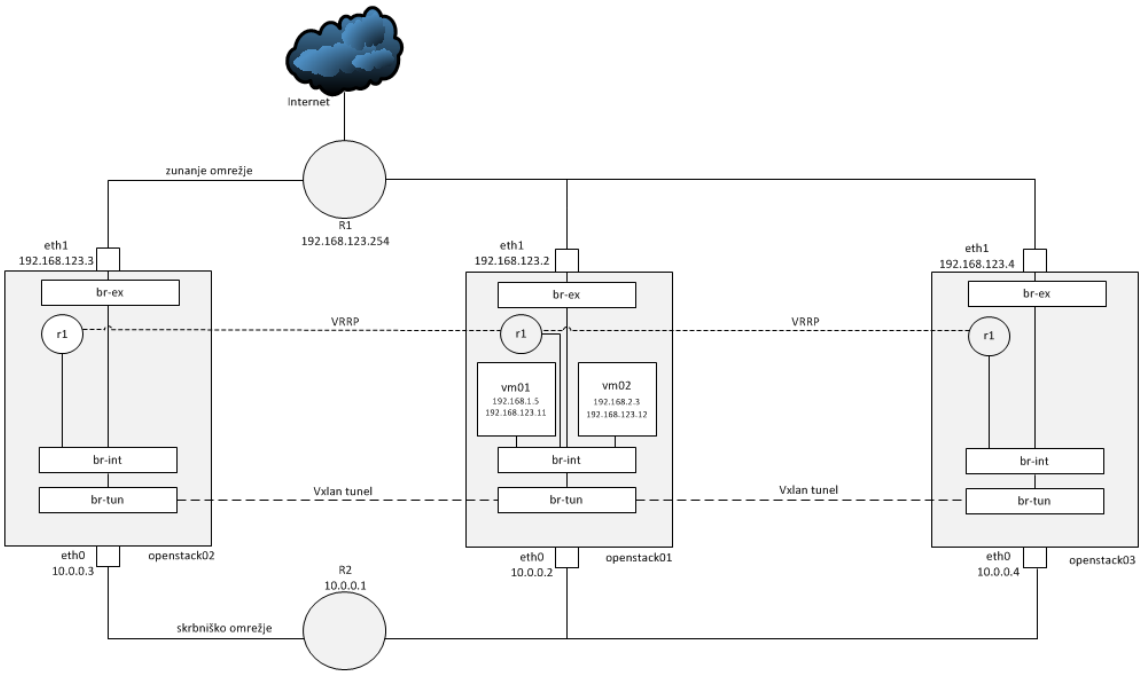


Slika 27: Shema L3 povezave v omrežje ponudnika z dvema ločenima navideznima usmerjevalnikoma

Postopek kreiranja in konfiguracije dodatnega virtualnega usmerjevalnika ter preverjanje povezljivosti med virtualnimi instancami in zunanjimi omrežnimi elementi ter resničnimi računalniki je opisan v poglavju 7.3.7. Med virtualnima instancama »vm01« in »vm02« v tem primeru ni povezljivosti po internih IP naslovih. Povezljivost je zagotovljena le preko plavajočih IP naslovov.

5.2.4 Postavitev oblačne infrastrukture z L3 visoko razpoložljivo povezavo v omrežje ponudnika

Postavitev opisuje in prikazuje implementacijo L3 visoko razpoložljive povezljivosti oblačne infrastrukture in zunanjega omrežja. Tako kot v primeru prejšnje postavitve virtualne instance imajo IP naslove v različnem L2 omrežju od omrežja ponudnika, povezljivost do zunanjega omrežja in resničnih računalnikov pa je zagotovljena z uporabo plavajočih IP naslovov, ki so v enakem L2 omrežju ponudnika. Prejšnjo postavitev nadgrajuje tako, da dodatno zagotavlja visoko razpoložljivo L3 povezavo v zunanje omrežje z uporabo virtualnega usmerjevalnega redundantnega protokola (*ang. virtual router redundancy protocol – VRRP*). Ta s pomočjo posebnih statusnih (*ang. keepalived*) paketov omogoča hiter preklop L3 storitev. Te zagotavlja dodatno ali več omrežnih vozlišč, ki delujejo v visoko razpoložljivem načinu. Podobno kot v prejšnjem primeru, tudi v tem primeru ves omrežni promet najemnikov, ki uporabljajo storitve usmerjanja, potuje samo skozi eno glavno (*ang. master*) omrežno vozlišče ne glede na število dodatnih omrežnih vozlišč, ki zagotavljajo HA način usmerjanja. V primeru odpovedi glavnega omrežnega vozlišča njegovo vlogo prevzame eno izmed omrežnih vozlišč v stanju pripravljenosti (*ang. stand-by*). Pri tej postavitvi so uporabljena tri OpenStack vozlišča. Vozlišče z imenom »openstack01« je kontrolno in računsko vozlišče ter dodatno zagotavlja tudi omrežne storitve v stanju pripravljenosti. Vozlišče »openstack02« je glavno omrežno vozlišče ter vozlišče »openstack03« dodatno omrežno vozlišče v stanju pripravljenosti. Postavitev je primerna za večja okolja, v katerih je treba zagotoviti visoko razpoložljivost omrežne povezljivosti med oblačno infrastrukturo ter fizičnim omrežjem in resničnimi strežniki in računalniki. Pri tem je treba opozoriti, da HA način usmerjanja in preklop iz glavnega omrežnega vozlišča na vozlišče v stanju pripravljenosti ohranja stanje omrežnih povezav le za virtualne instance, ki imajo plavajoče IP naslove. Postavitev ne odpravlja problema omejitve pasovne širine in hitrosti omrežnega dostopa virtualnih instanc. Za povečanje omrežne zmogljivosti je bolj primerna postavitev oblačne infrastrukture z uporabo porazdeljenih virtualnih usmerjevalnikov, ki je opisana v poglavju 5.2.5.



Slika 28: Shema L3 visoko razpoložljive povezave v omrežje ponudnika

Preverjanje kreiranja oblačne infrastrukture in statusa omrežnih servisov na OpenStack vozliščih je opisano v poglavju 7.4.7.

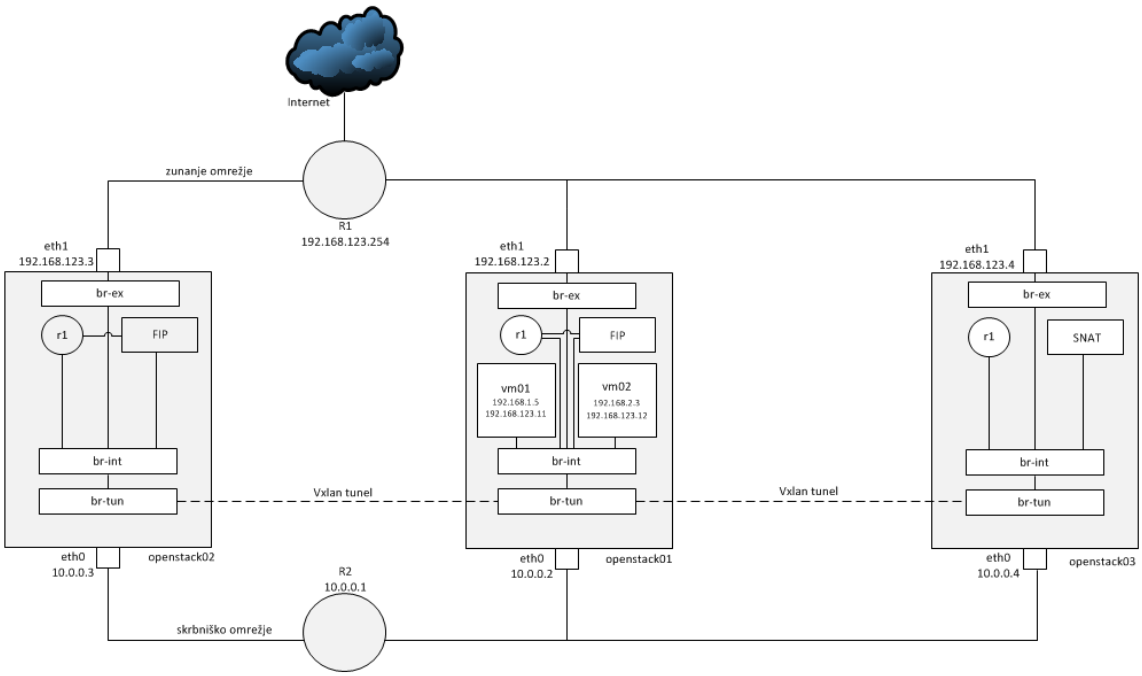
Ker DevStack skripta trenutno nima možnosti ustvariti vseh potrebnih elementov L3 visoko razpoložljive (*ang. high availability – HA*) postavitve, je treba omrežji ponudnika in najemnika, ustvarjeni ob zagonu skripte, izbrisati in ju ponovno ročno dodati z vsemi potrebnimi omrežnimi elementi. Postopek je opisan v poglavju 7.4.8.

Preverjanje uspešnosti oglaševanja VRRP (*ang. virtual router redundancy protocol*) protokola ter omrežne povezljivosti virtualnih usmerjevalnikov je opisano v poglavju 7.4.9.

5.2.5 Postavitev oblačne infrastrukture z L3 povezavo v omrežje ponudnika in uporabo porazdeljenih virtualnih usmerjevalnikov

Postavitev opisuje in prikazuje implementacijo L3 povezljivosti oblačne infrastrukture in zunanjega omrežja z omogočeno funkcionalnostjo porazdeljenega virtualnega usmerjanja. Takšna postavitev nadgrajuje standardno L3 oblačno omrežno arhitekturo na način, da omogoča direktno povezljivost računskih OpenStack vozlišč in zunanjega omrežja ter resničnih računalnikov. Za virtualne instance, ki uporabljajo plavajoče IP naslove, je usmerjanje med najemniškimi in zunanjim omrežjem ter resničnimi računalniki v celoti podprto na računskih vozliščih. Takšna arhitektura odpravlja kritično točko odpovedi (*ang. single point of failure*) standardnega

OpenStack omrežnega vozlišča ter ga tudi razbremeni. Usmerjanje je v celoti podprto tudi v primeru virtualnih instanc s fiksnimi ali plavajočimi IP naslovi (*ang. floating IP – FIP*), ki uporabljajo omrežja ponudnika znotraj istega porazdeljenega virtualnega usmerjevalnika. Virtualne instance, ki uporabljajo fiksne IP naslove, pa še zmeraj za komunikacijo in storitve prevajanja omrežnega izvornega naslova (*ang. source NAT – SNAT*) med omrežji najemnika in zunanjim omrežjem potrebujejo omrežno vozlišče. Pri tej postavitvi so uporabljena tri OpenStack vozlišča. Vozlišče z imenom »openstack01« je kontrolno in računsko vozlišče ter dodatno zagotavlja tudi omrežno storitev porazdeljenega virtualnega usmerjevalnika. Vozlišče »openstack02« je dodatno računsko vozlišče z omogočeno storitvijo porazdeljenega virtualnega usmerjevalnika. Vozlišče »openstack03« je omrežno vozlišče z omogočeno SNAT funkcionalnostjo. Opisana postavitev je primerna za večja okolja, v katerih je treba zagotoviti visoko prepustnost omrežnega prometa med oblačno infrastrukturo in fizičnim omrežjem ter resničnimi strežniki in računalniki. Za okolja, v katerih je bolj kot omrežne zmogljivosti oblačne infrastrukture pomembno zagotoviti visoko razpoložljivost omrežnega dostopa virtualnih instanc in fizične infrastrukture ter resničnih strežnikov in računalnikov, je primernejša postavitev v načinu L3 HA, opisana v prejšnjem poglavju. Vključno z različico Liberty OpenStack ne omogoča združevanje načinov L3 visoke omrežne razpoložljivosti in porazdeljenih virtualnih usmerjevalnikov.



Slika 29: Shema L3 povezave v omrežje ponudnika z uporabo porazdeljenih virtualnih usmerjevalnikov

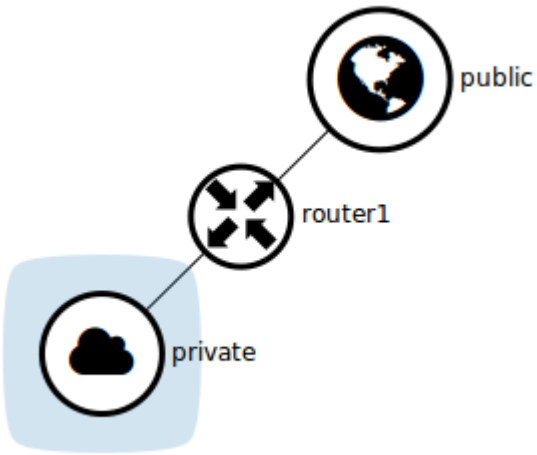
Preverjanje kreiranja oblačne infrastrukture ter statusa omrežnih servisov je opisano v poglavju 7.5.7. Postopek kreiranja zunanjega in najemniškega omrežja ter virtualnih instanc je enak opisu v prejšnjem primeru. Na omrežnem vozlišču uspešnost kreiranja »snat«, »qrouter« in »qdhcp« imenskih prostorov preverimo tako, kot je opisano v poglavju 7.5.8. Uspešnost kreiranja »qrouter« ter »fip« imenskega prostora na računskem vozlišču z virtualnimi instancami preverimo na način, opisan v poglavju 7.5.9.

5.2.6 Postavitev oblačne infrastrukture z uporabo avtomatizacije in orkestracije

Postavitev opisuje in prikazuje implementacijo oblačne infrastrukture z uporabo instalacijske skripte DevStack ter OpenStack Heat konfiguracijske predloge. Ta temelji na tekstovni datoteki in se jo obravnava kot vhodno kodo v OpenStack oblaku. Pri tej postavitvi je uporabljeno eno OpenStack vozlišče »openstack01«, ki je hkrati v vlogi kontrolnega, računskega in omrežnega vozlišča ter ima tudi omogočen Heat orkestracijski mehanizem.

DevStack skripta z vhodnimi podatki, ki jih prebere iz konfiguracijske datoteke »local.conf«, namesti ustrezno OpenStack oblačno infrastrukturo in v tem primeru dodatno ustvari omrežja in podomrežja ponudnika »public« in najemnika z imenom »private« ter virtualni usmerjevalnik »router1«, ki zagotavlja L3 povezljivost do zunanjega omrežja. Postopek instalacije DevStack je opisan v poglavju 7.6.4.

Uspešnost kreiranja oblačne infrastrukture lahko preverimo tudi preko OpenStack spletnega vmesnika ter izbire možnosti »Network Topology«. Takrat se nam prikaže vizualna shema omrežja, kot je prikazano na sliki spodaj.

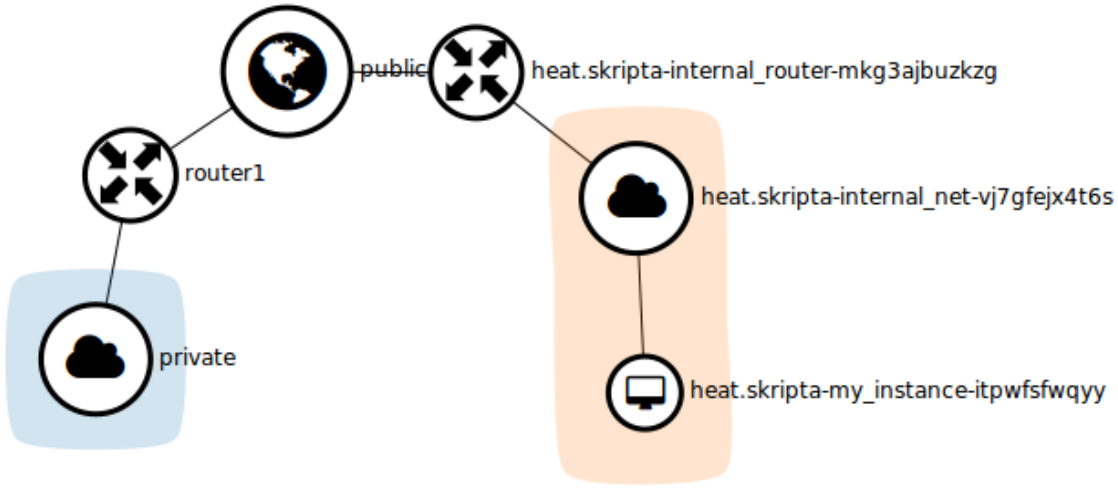


Slika 30: Shema omrežne topologije iz spletnega vmesnika OpenStack nadzorne plošče

Uspešnost kreiranja oblačne infrastrukture preverimo iz ukazne vrstice po postopku, opisanem v poglavju 7.6.5.

Nato izvedemo ukaz za zagon Heat predloge, ki ustvari dodatno omrežje in podomrežje najemnika ter tudi virtualno instanco znotraj le-tega z dostopom do zunanjega omrežja. Postopek je opisan v poglavju 7.6.6.

Uspešnost izvedbe Heat predloge je razvidna iz spletnega vmesnika OpenStack in nadzorne plošče.



Slika 31:Shema omrežne topologije iz spletnega vmesnika OpenStack nadzorne plošče po izvedbi Heat predloge

Uspešnost izvedbe Heat predloge in kreiranja oblačne infrastrukture preverimo iz ukazne vrstice po postopku, opisanem v poglavju 7.6.7.

6 Zaključek

Stanje globalne ekonomske krize v zadnjih nekaj letih ne kaže bistvenih znakov izboljšanja. Podjetja iščejo načine, kako znižati stroške ter ohraniti konkurenčnost. Veliko podjetij je prisiljenih v odpuščanje zaposlenih, hkrati pa se pričakuje večja učinkovitost. Zaradi številnih konkurentov in hitrih sprememb potreb na trgu morajo podjetja tudi sama hitro reagirati. Kupci pričakujejo nižje cene, boljšo kvaliteto in večjo fleksibilnost. Vse to vodi v bolj kompleksne in zahtevne sisteme ter bolj kompleksno poslovno okolje. To zahteva nove pristope pri vzpostavljanju in upravljanju IT infrastrukture in virov. Virtualizacija strežnikov je že uveljavljena tehnologija, na omrežjih pa se spremembe dogajajo šele zdaj.

Programsko določena omrežja, omrežna virtualizacija in virtualizacija omrežnih funkcij so nove tehnologije, ki popolnoma spreminjajo dosedanje omrežne koncepte. Čeprav se je SDN uradno začel šele nedavno (leta 2008), je že do zdaj pritegnil veliko pozornosti ponudnikov storitev, podjetij in proizvajalcev. Novi proizvajalci, ponudniki in »start-up« podjetja navdušeno iščejo inovativne primere uporabe v praksi. Zaradi hitrega razvoja več podobnih pristopov hkrati (SDN, NFV, NV itd.) in tehnoloških prekrivanj je uporabnikom včasih težko določiti mejo med njimi in jih razumeti. Že uveljavljeni veliki proizvajalci želijo obdržati svojo tržno pozicijo in predlagajo nekakšne svoje lastne verzije SDN. To pomeni, da kupci še zmeraj potrebujejo njihovo specifično opremo in včasih tudi specifične lastne protokole. Ravno ti zaprti sistemi so do zdaj bili razlog počasnega razvoja in napredka. SDN s svojimi odprtimi standardi, neodvisnimi od proizvajalcev, naredi omrežje odprto in programabilno. Razvoj novih storitev in funkcionalnosti ni več vezan na strojne namenske naprave, ampak na programsko opremo, ki jo lahko vsak sam napiše.

Odprtokodne oblačne platforme, kot je to trenutno najbolj razširjena platforma OpenStack, združujejo koncepte virtualizacije strežnikov, omrežne virtualizacije, programsko določenih omrežij in virtualizacije omrežnih funkcij. Podjetja in uporabniki so s tem pridobili možnost, da razen komercialnih rešitev uporabljajo tudi odprtokodne in brezplačne rešitve za postavljanje lastne oblačne infrastrukture. Ta za razliko od tradicionalnih rešitev, ki v podatkovnih centrih uporabljajo fizično opremo, omogoča polno programabilnost infrastrukture ter med drugim razvoj aplikacij, ki so narejene za takšno okolje (npr. vsebniki) in se jih lahko brez težav preseli iz fizične opreme v zasebne in javne oblake.

Kot je bilo ugotovljeno v prejšnjih poglavjih, iz različnih razlogov le ni možno virtualizirati vse informacijske infrastrukture. V takih primerih je treba zagotoviti povezljivost med virtualnimi in resničnimi strežniki in računalniki. Način povezovanja med oblačno in fizično infrastrukturo je odvisen od uporabniških in aplikacijskih zahtev. Tudi nekateri omrežni protokoli ne dovoljujejo uporabo prevajanja omrežnih naslovov in za pravilno delovanje v takih primerih potrebujejo namenske prehode. V splošnem obstajata dva načina omrežnega povezovanja, in sicer povezovanje na L2 ali L3 omrežni plasti. Oba načina povezovanja imata svoje prednosti in pomanjkljivosti. Prednosti L2 načina povezovanja so hitrost, ni potrebe po spremembah IP naslovov v primeru selitev virtualnih in resničnih računalnikov ter zaradi tega ni potrebe po

spremembah na nivoju konfiguracije omrežja, uporaba VLAN-ov je enostaven in uveljavljen mehanizem za L2 izolacijo omrežij itd. Prednosti L3 načina povezovanja pa je večja prilagodljivost in skalabilnost ter boljši nadzor in upravljanje prometa z uporabo različnih orodij in mehanizmov, kot je npr. kakovost storitev. Izolacijo omrežnega prometa v primeru L3 povezovanja lahko dosežemo z uporabo virtualnih prekrivnih omrežij. Pri obeh načinih omrežnega povezovanja virtualnih in resničnih računalnikov je treba upoštevati tudi potrebe po zagotavljanju omrežne zmogljivosti in visoke razpoložljivosti. Kot je bilo prikazano na praktičnih primerih, OpenStack trenutno ne omogoča kombinacije obeh načinov delovanja hkrati.

Čeprav z vsako novo izdajo ponuja več možnosti in podpira več funkcionalnosti, ima odprtokodna oblačna platforma OpenStack, razen prej omenjene omejitve omrežne arhitekture, še nekaj omejitev. Ker je sestavljena iz več različnih komponent, je postavitev in konfiguracija okolja preveč kompleksna. Zaradi tega za razumevanje delovanja zahteva strokovnjake s širokim razponom znanja različnih informacijsko-komunikacijskih področij. Nekateri ponudniki ponujajo lastne komercialne OpenStack različice, ki omogočajo lažjo avtomatizacijo in orkestracijo oblačne infrastrukture in storitev. Razen tega, da zanje nudijo tudi tehnično podporo, so v komercialnih različicah tudi odpravljene napake in hrošči iz odprtokodne različice, katerih odpravljanje se v slednjem primeru pogosto prestavi na novejše različice. Glede na potek in rezultate testiranj pri praktičnem primeru naloge se za postavitve OpenStack oblačne platforme v produkcijskem okolju trenutno priporoča uporaba komercialnih različic.

Na koncu pa ne smemo pozabiti, da je največja vrednost takšne platforme, kot je OpenStack, ravno v njeni odprtosti in modularnosti. Tako kot se je zgodilo v primeru operacijskega sistema Linux, ki je bil v devetdesetih letih bistveno bolj kompleksen kot zdaj, se bo zgodilo tudi z OpenStackom. Z zrelostjo tehnologije se zmanjša tudi njena kompleksnost.

7 Priloge

7.1 Konfiguracijske datoteke za postavitev oblačne infrastrukture z L2 povezavo v omrežje ponudnika brez označevanja paketov

7.1.1 Devstack konfiguracija za vozlišče openstack01

```
[[local|localrc]]
HOST_IP=10.0.0.2
SERVICE_HOST=10.0.0.2
MYSQL_HOST=10.0.0.2
RABBIT_HOST=10.0.0.2
GLANCE_HOSTPORT=10.0.0.2:9292
PUBLIC_INTERFACE=eth1

IP_VERSION=4

ADMIN_PASSWORD=password
MYSQL_PASSWORD=password
RABBIT_PASSWORD=password
SERVICE_PASSWORD=password
SERVICE_TOKEN=password

## Neutron moznosti
Q_USE_SECGROUP=True
PHYSICAL_NETWORK=provider_external
OVS_PHYSICAL_BRIDGE=br-ex

Q_USE_PROVIDER_NETWORKING=True
Q_L3_ENABLED=False

#Ne uporabi Nova-Network
disable_service n-net

#Storitve ki jih zagotavlja Neutron vozlisce
ENABLED_SERVICES+=,q-svc,q-dhcp,q-meta,q-agt

#Neutron omrezne moznosti uporabljene za ustvarjanje Neutron podomrezij

FIXED_RANGE="192.168.123.0/24"
NETWORK_GATEWAY=192.168.123.254
PROVIDER_SUBNET_NAME="provider_net"
PROVIDER_NETWORK_TYPE="flat"


#Tempest
disable_service tempest
```

7.1.2 Omrežna konfiguracija za vozlišče openstack01

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback


#vmesnik eth0 staticen IP naslov
auto eth0
 iface eth0 inet static
     address 10.0.0.2
     netmask 255.255.255.0
     gateway 10.0.0.1
     dns-nameservers 8.8.8.8


#vmesnik eth1 v promiscous nacinu
auto eth1
iface eth1 inet manual
    up ifconfig $IFACE 0.0.0.0 up
    up ip link set $IFACE promisc on
    down ip link set $IFACE promisc off
    down ifconfig $IFACE down


#vmesnik br-ex staticen IP naslov
auto br-ex
   iface br-ex inet static
   address 192.168.123.2
   netmask 255.255.255.0
   gateway 192.168.123.254
   dns-nameservers 8.8.8.8
```

7.1.3 Devstack konfiguracija za vozlišče openstack02

```
[[local|localrc]]
HOST_IP=10.0.0.3
SERVICE_HOST=10.0.0.2
MYSQL_HOST=10.0.0.2
RABBIT_HOST=10.0.0.2
GLANCE_HOSTPORT=10.0.0.2:9292
ADMIN_PASSWORD=password
MYSQL_PASSWORD=password
RABBIT_PASSWORD=password
SERVICE_PASSWORD=password
SERVICE_TOKEN=password
IP_VERSION=4

#Storitve ki jih zagotavlja Compute vozlisce
ENABLED_SERVICES=n-cpu,rabbit,q-agt
```

```
#Open vSwitch moznosti omrezja ponudnika
PHYSICAL_NETWORK=default
OVS_PHYSICAL_BRIDGE=br-ex
PUBLIC_INTERFACE=eth1
Q_USE_PROVIDER_NETWORKING=True
Q_L3_ENABLED=False

#Tempest
disable_service tempest
```

7.1.4 Omrežna konfiguracija za vozlišče openstack02

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

#vmesnik eth0 staticen IP naslov
auto eth0
 iface eth0 inet static
     address 10.0.0.3
     netmask 255.255.255.0
     gateway 10.0.0.1
     dns-nameservers 8.8.8.8


#vmesnik eth1 v promiscous nacinu
auto eth1
iface eth1 inet manual
    up ifconfig $IFACE 0.0.0.0 up
    up ip link set $IFACE promisc on
    down ip link set $IFACE promisc off
    down ifconfig $IFACE down


#vmesnik br-ex staticen IP naslov
auto br-ex
    iface br-ex inet static
    address 192.168.123.3
    netmask 255.255.255.0
    gateway 192.168.123.254
    dns-nameservers 8.8.8.8
```

7.1.5 OpenStack ukazi za preverjanje uspešnosti kreiranja oblačne infrastrukture

Preverjanje uspešnosti kreiranja omrežja:

```
amir@openstack01:~/devstack$ source admin.openrc
amir@openstack01:~/devstack$ neutron net-list
+--------------------------------------+------------------+--------------------------
----------------------------+
| id                                   | name             | subnets
|
```

```
+--------------------------------------+------------------+--------------------------
----------------------------+
| d09fd87d-01d3-49b0-8648-2cec81816bc9 | provider_external | f51d987b-028c-492d-8f12-
ed1321b146e9 192.168.123.0/24 |
+--------------------------------------+------------------+--------------------------
----------------------------+
```

Preverjanje statusa izvajanja agentov na posameznih OpenStack vozliščih:

```
amir@openstack01:~/devstack$ neutron agent-list
+--------------------------------------+--------------------+-------------+-------+----
------------+----------------------------+
| id                                   | agent_type         | host        | alive |
admin_state_up | binary                     |
+--------------------------------------+--------------------+-------------+-------+----
------------+----------------------------+
| 0df82886-a167-4562-bc0e-538e10798196 | Open vSwitch agent | openstack01 | :-)   |
True           | neutron-openvswitch-agent |
| 1a83f842-90f7-4f67-a61c-c8333190b969 | Metadata agent     | openstack01 | :-)   |
True           | neutron-metadata-agent    |
| 5021805c-fbdb-4451-9c6d-3fad4150aaea | Open vSwitch agent | openstack02 | :-)   |
True           | neutron-openvswitch-agent |
| ab42b1e0-3c1d-49b5-9696-f13ec98a1b94 | DHCP agent         | openstack01 | :-)   |
True           | neutron-dhcp-agent        |
+--------------------------------------+--------------------+-------------+-------+----
------------+----------------------------+
```

Preverjanje statusa navideznega prekrivnega vxlan omrežja med OpenStack vozliščama:

```
amir@openstack01:~/devstack$ sudo ovs-vsctl show
385e02e0-e387-4393-b720-66a61b320f26
    Bridge br-int
        fail_mode: secure
        Port "tapca76fe72-f5"
            tag: 1
            Interface "tapca76fe72-f5"
                type: internal
        Port patch-tun
            Interface patch-tun
                type: patch
                options: {peer=patch-int}
        Port int-br-ex
            Interface int-br-ex
                type: patch
                options: {peer=phy-br-ex}
        Port br-int
            Interface br-int
                type: internal
    Bridge br-ex
        Port br-ex
            Interface br-ex
                type: internal
        Port phy-br-ex
            Interface phy-br-ex
                type: patch
                options: {peer=int-br-ex}
        Port "eth1"
            Interface "eth1"
    Bridge br-tun
```

```
        fail_mode: secure
        Port "vxlan-0a000002"
            Interface "vxlan-0a000002"
                type: vxlan
                options:    {df_default="true",    in_key=flow,    local_ip="10.0.0.2",
out_key=flow, remote_ip="10.0.0.3"}
        Port br-tun
            Interface br-tun
                type: internal
        Port patch-int
            Interface patch-int
                type: patch
                options: {peer=patch-tun}
    ovs_version: "2.0.2"

amir@openstack02:~/devstack$ sudo ovs-vsctl show
e469a2bd-dbaa-4663-b7b0-a1d0f8a04665
    Bridge br-ex
        Port phy-br-ex
            Interface phy-br-ex
                type: patch
                options: {peer=int-br-ex}
        Port "eth1"
            Interface "eth1"
        Port br-ex
            Interface br-ex
                type: internal
    Bridge br-int
        fail_mode: secure
        Port br-int
            Interface br-int
                type: internal
        Port patch-tun
            Interface patch-tun
                type: patch
                options: {peer=patch-int}
        Port int-br-ex
            Interface int-br-ex
                type: patch
                options: {peer=phy-br-ex}
    Bridge br-tun
        fail_mode: secure
        Port patch-int
            Interface patch-int
                type: patch
                options: {peer=patch-tun}
        Port br-tun
            Interface br-tun
                type: internal
        Port "vxlan-0a000001"
            Interface "vxlan-0a000001"
                type: vxlan
                options:    {df_default="true",    in_key=flow,    local_ip="10.0.0.3",
out_key=flow, remote_ip="10.0.0.2"}
    ovs_version: "2.0.2"
amir@openstack02:~/devstack$
```

Preverjanje vzorcev slik operacijskih sistemov, ki so na razpolago, ter pripravljenih vzorcev virtualnih instanc:

```
amir@openstack01:~/devstack$ source demo.openrc
amir@openstack01:~/devstack$ nova image-list
```

```
+--------------------------------------+---------------------------------+--------+----
----+
| ID                                   | Name                            | Status |
Server |
+--------------------------------------+---------------------------------+--------+----
----+
| 227c7b2d-b272-4f31-8825-ea123d76dfb2 | cirros-0.3.4-x86_64-uec         | ACTIVE |
|
| 126a6eb8-1aa6-43b8-8f56-984ca4ad38bc | cirros-0.3.4-x86_64-uec-kernel  | ACTIVE |
|
| 484db26c-7660-48d1-b6b4-b2688be2a133 | cirros-0.3.4-x86_64-uec-ramdisk | ACTIVE |
|
+--------------------------------------+---------------------------------+--------+----
----+

amir@openstack01:~/devstack$ nova flavor-list
+----+-----------+-----------+------+-----------+------+-------+-------------+---------
--+
| ID | Name      | Memory_MB | Disk | Ephemeral | Swap | VCPUs | RXTX_Factor |
Is_Public |
+----+-----------+-----------+------+-----------+------+-------+-------------+---------
--+
| 1  | m1.tiny   | 512       | 1    | 0         |      | 1     | 1.0         | True
|
| 2  | m1.small  | 2048      | 20   | 0         |      | 1     | 1.0         | True
|
| 3  | m1.medium | 4096      | 40   | 0         |      | 2     | 1.0         | True
|
| 4  | m1.large  | 8192      | 80   | 0         |      | 4     | 1.0         | True
|
| 5  | m1.xlarge | 16384     | 160  | 0         |      | 8     | 1.0         | True
|
+----+-----------+-----------+------+-----------+------+-------+-------------+---------
--+
```

7.1.6 OpenStack ukazi za kreiranje in preverjanje virtualnih instanc

Kreiranje virtualne instance:

```
amir@openstack01:~/devstack$ nova boot --flavor m1.tiny --image cirros-0.3.4-x86_64-uec
--nic net-id=d09fd87d-01d3-49b0-8648-2cec81816bc9 --security-group default --key-name
mykey vm01
+--------------------------------------+--------------------------------------------------
-----------------+
|  Property                                                                    |  Value
|
+--------------------------------------+--------------------------------------------------
-----------------+
|  OS-DCF:diskConfig                                                           |  MANUAL
|
|    OS-EXT-AZ:availability_zone                                                          |
|
|   OS-EXT-STS:power_state                                                     |   0
|
|  OS-EXT-STS:task_state                                                   |  scheduling
|
|  OS-EXT-STS:vm_state                                                     |  building
|
|   OS-SRV-USG:launched_at                                                     |   -
|
```

```
|    OS-SRV-USG:terminated_at                                                 |    -
|
|   accessIPv4                                                                |
|
|   accessIPv6                                                                |
|
|  adminPass                                                          |  LKQ86Vh3aXuX
|
|   config_drive                                                              |
|
|  created                                                        |  2016-05-12T13:48:27Z
|
|  flavor                                                             |  m1.tiny  (1)
|
|   hostId                                                                    |
|
| id                                          | 2170ea98-e593-4f0b-9c80-71835b4cee9e
|
| image                                   | cirros-0.3.4-x86_64-uec (0026eb20-789a-4b8e-
9620-7ca1f30838ff) |
|   key_name                                                                  |  mykey
|
|   metadata                                                                  |  {}
|
|   name                                                                      |  vm01
|
|               os-extended-volumes:volumes_attached               |               []
|
|   progress                                                                  |  0
|
|   security_groups                                                       |  default
|
|   status                                                                    |  BUILD
|
|  tenant_id                                        | 1937da8c8cc9485d877c1229a83a6171
|
|  updated                                                        |  2016-05-12T13:48:27Z
|
|  user_id                                          | d918240a3fc541ff842e53102df87e08
|
+-------------------------------------+--------------------------------------------
-----------------+
amir@openstack01:~/devstack$ nova boot --flavor m1.tiny --image cirros-0.3.4-x86_64-uec
--nic  net-id=d09fd87d-01d3-49b0-8648-2cec81816bc9  --security-group  default  --key-name
mykey vm02
+-------------------------------------+--------------------------------------------
-----------------+
|  Property                                                                   |  Value
|
+-------------------------------------+--------------------------------------------
-----------------+
|   OS-DCF:diskConfig                                                     |  MANUAL
|
|     OS-EXT-AZ:availability_zone                                             |
|
|    OS-EXT-STS:power_state                                                   |    0
|
|   OS-EXT-STS:task_state                                                 |  scheduling
|
|   OS-EXT-STS:vm_state                                                   |  building
|
|    OS-SRV-USG:launched_at                                                   |    -
|
```

```
|    OS-SRV-USG:terminated_at                                                  |    -
|
|   accessIPv4                                                                 |
|
|   accessIPv6                                                                 |
|
|   adminPass                                                          |  37Hh28KEsEoK
|
|   config_drive                                                               |
|
|  created                                                           |  2016-05-12T13:48:43Z
|
|   flavor                                                             |  m1.tiny  (1)
|
|   hostId                                                                     |
|
| id                                              | e91e6478-984d-4c63-a81d-0a63713886d1
|
| image                                    | cirros-0.3.4-x86_64-uec (0026eb20-789a-4b8e-
9620-7ca1f30838ff) |
|   key_name                                                                 |  mykey
|
|   metadata                                                                   |  {}
|
|   name                                                                     |  vm02
|
|               os-extended-volumes:volumes_attached               |               []
|
|   progress                                                                   |  0
|
|   security_groups                                                        |  default
|
|   status                                                                   |  BUILD
|
|  tenant_id                                          | 1937da8c8cc9485d877c1229a83a6171
|
|  updated                                                           |  2016-05-12T13:48:44Z
|
|  user_id                                            | d918240a3fc541ff842e53102df87e08
|
+-------------------------------------+-------------------------------------------
-----------------+
```

Uspešnost izvedbe kreiranja virtualnih instanc:

```
amir@openstack01:~/devstack$ nova list
+--------------------------------------+------+--------+------------+-------------+----
-------------------------------+
| ID                                   | Name | Status | Task State | Power State |
Networks                       |
+--------------------------------------+------+--------+------------+-------------+----
-------------------------------+
| 2170ea98-e593-4f0b-9c80-71835b4cee9e | vm01 | ACTIVE | -          | Running     |
provider_external=192.168.123.19 |
| e91e6478-984d-4c63-a81d-0a63713886d1 | vm02 | ACTIVE | -          | Running     |
provider_external=192.168.123.21 |
+--------------------------------------+------+--------+------------+-------------+----
-------------------------------+
```

7.1.7 OpenStack ukazi za preverjanje L2 omrežne povezljivosti

```
amir@openstack01:~/devstack$ nova secgroup-list
+--------------------------------------+---------+------------------------+
| Id                                   | Name    | Description            |
+--------------------------------------+---------+------------------------+
| 4497c20a-8aa1-492c-bfcf-7ad7e57dfd20 | default | Default security group |
+--------------------------------------+---------+------------------------+
amir@openstack01:~/devstack$ nova secgroup-add-rule default icmp -1 -1 0.0.0.0/24
+-------------+-----------+---------+------------+--------------+
| IP Protocol | From Port | To Port | IP Range   | Source Group |
+-------------+-----------+---------+------------+--------------+
| icmp        | -1        | -1      | 0.0.0.0/24 |              |
+-------------+-----------+---------+------------+--------------+
amir@openstack01:~/devstack$ nova secgroup-add-rule default tcp 22 22 0.0.0.0/24
+-------------+-----------+---------+------------+--------------+
| IP Protocol | From Port | To Port | IP Range   | Source Group |
+-------------+-----------+---------+------------+--------------+
| tcp         | 22        | 22      | 0.0.0.0/24 |              |
+-------------+-----------+---------+------------+--------------+

amir@openstack01:~/devstack$ ssh cirros@192.168.123.19
The authenticity of host '192.168.123.19 (192.168.123.19)' can't be established.
RSA key fingerprint is 5d:86:cc:fa:93:84:cf:10:e5:cb:45:49:f7:bc:97:3c.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.123.19' (RSA) to the list of known hosts.
$ ifconfig
eth0      Link encap:Ethernet  HWaddr FA:16:3E:02:81:DA
          inet addr:192.168.123.19  Bcast:192.168.123.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:fe02:81da/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:372 errors:0 dropped:0 overruns:0 frame:0
          TX packets:63 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:32178 (31.4 KiB)  TX bytes:6934 (6.7 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

$ ping 192.168.123.254
PING 192.168.123.254 (192.168.123.254): 56 data bytes
64 bytes from 192.168.123.254: seq=0 ttl=64 time=16.387 ms
64 bytes from 192.168.123.254: seq=1 ttl=64 time=2.221 ms
64 bytes from 192.168.123.254: seq=2 ttl=64 time=2.389 ms
^C
--- 192.168.123.254 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 2.221/6.999/16.387 ms
$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: seq=0 ttl=59 time=14.293 ms
64 bytes from 8.8.8.8: seq=1 ttl=59 time=11.797 ms
64 bytes from 8.8.8.8: seq=2 ttl=59 time=11.509 ms
64 bytes from 8.8.8.8: seq=3 ttl=59 time=11.539 ms
^C
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 11.509/12.284/14.293 ms
$ ping 192.168.123.21
```

```
PING 192.168.123.21 (192.168.123.21): 56 data bytes
^C
--- 192.168.123.21 ping statistics ---
20 packets transmitted, 0 packets received, 100% packet loss
$ exit
Connection to 192.168.123.19 closed.
amir@openstack01:~/devstack$
```

7.2 Konfiguracijske datoteke za postavitev oblačne infrastrukture z L2 povezavo v omrežje ponudnika z uporabo označevanja paketov VLAN

7.2.1 Devstack konfiguracija za vozlišče openstack01

```
[[local|localrc]]
HOST_IP=10.0.0.2
SERVICE_HOST=10.0.0.2
MYSQL_HOST=10.0.0.2
RABBIT_HOST=10.0.0.2
GLANCE_HOSTPORT=10.0.0.2:9292
PUBLIC_INTERFACE=eth1

IP_VERSION=4

ADMIN_PASSWORD=password
MYSQL_PASSWORD=password
RABBIT_PASSWORD=password
SERVICE_PASSWORD=password
SERVICE_TOKEN=password

#Neutron moznosti
Q_USE_SECGROUP=True
PHYSICAL_NETWORK=default
OVS_PHYSICAL_BRIDGE=br-ex

Q_USE_PROVIDER_NETWORKING=True
Q_L3_ENABLED=False

#Ne uporabi Nova-Network
disable_service n-net

#Storitve ki jih zagotavlja Neutron vozlisce
ENABLED_SERVICES+=,q-svc,q-dhcp,q-meta,q-agt

#Neutron omrezne moznosti uporabljene za ustvarjanje Neutron podomrezij

FIXED_RANGE="192.168.123.0/24"
NETWORK_GATEWAY=192.168.123.254
PROVIDER_SUBNET_NAME="provider_net"
PROVIDER_NETWORK_TYPE="vlan"
SEGMENTATION_ID=150

#Tempest
```

```
disable_service tempest
```

7.2.2 Omrežna konfiguracija za vozlišče openstack01

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback


#vmesnik eth0 staticen IP naslov
auto eth0
 iface eth0 inet static
     address 10.0.0.2
     netmask 255.255.255.0
     gateway 10.0.0.1
     dns-nameservers 8.8.8.8


#vmesnik eth1 v promiscous nacinu
auto eth1
iface eth1 inet manual
    up ifconfig $IFACE 0.0.0.0 up
    up ip link set $IFACE promisc on
    down ip link set $IFACE promisc off
    down ifconfig $IFACE down


#vmesnik br-ex staticen IP naslov
auto br-ex
   iface br-ex inet static
   address 192.168.123.2
   netmask 255.255.255.0
   gateway 192.168.123.254
   dns-nameservers 8.8.8.8
```

7.2.3 Devstack konfiguracija za vozlišče openstack02

```
[[local|localrc]]
HOST_IP=10.0.0.3
SERVICE_HOST=10.0.0.2
MYSQL_HOST=10.0.0.2
RABBIT_HOST=10.0.0.2
GLANCE_HOSTPORT=10.0.0.2:9292
ADMIN_PASSWORD=password
MYSQL_PASSWORD=password
RABBIT_PASSWORD=password
SERVICE_PASSWORD=password
SERVICE_TOKEN=password
IP_VERSION=4

#Storitve ki jih zagotavlja Compute vozlisce
```

```
ENABLED_SERVICES=n-cpu,rabbit,q-agt

#Open vSwitch moznosti omrezja ponudnika
PHYSICAL_NETWORK=default
OVS_PHYSICAL_BRIDGE=br-ex
PUBLIC_INTERFACE=eth1
Q_USE_PROVIDER_NETWORKING=True
Q_L3_ENABLED=False
```

7.2.4 Omrežna konfiguracija za vozlišče openstack02

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

#vmesnik eth0 staticen IP naslov
auto eth0
 iface eth0 inet static
     address 10.0.0.3
     netmask 255.255.255.0
     gateway 10.0.0.1
     dns-nameservers 8.8.8.8


#vmesnik eth1 v promiscous nacinu
auto eth1
iface eth1 inet manual
    up ifconfig $IFACE 0.0.0.0 up
    up ip link set $IFACE promisc on
    down ip link set $IFACE promisc off
    down ifconfig $IFACE down


#vmesnik br-ex staticen IP naslov
auto br-ex
    iface br-ex inet static
    address 192.168.123.3
    netmask 255.255.255.0
    gateway 192.168.123.254
    dns-nameservers 8.8.8.8
```

7.2.5 OpenStack ukazi za kreiranje virtualnih instanc

```
amir@openstack01:~/devstack$ nova boot --flavor m1.tiny --image cirros-0.3.4-x86_64-uec
--nic net-id=31df734c-db00-4f29-9d2f-3fcd06b0be09 --security-group default --key-name
mykey vm01
+--------------------------------------+-----------------------------------------------
-----------------+
|  Property                                                                  |  Value
|
+--------------------------------------+-----------------------------------------------
-----------------+
|   OS-DCF:diskConfig                                                        |   MANUAL
|
```

```
|     OS-EXT-AZ:availability_zone                                                |
|
|    OS-EXT-STS:power_state                                                  |    0
|
|   OS-EXT-STS:task_state                                                 |   scheduling
|
|   OS-EXT-STS:vm_state                                                     |   building
|
|    OS-SRV-USG:launched_at                                                  |    -
|
|    OS-SRV-USG:terminated_at                                                |    -
|
|   accessIPv4                                                                   |
|
|   accessIPv6                                                                   |
|
|  adminPass                                                              |  TxPWm8LtLBmf
|
|   config_drive                                                                 |
|
|  created                                                          |  2016-05-12T08:41:41Z
|
|  flavor                                                                 |  m1.tiny  (1)
|
|   hostId                                                                       |
|
| id                                              | 0ae39159-a402-4391-8dcd-eb86b00ec6ab
|
| image                                  | cirros-0.3.4-x86_64-uec (7438492a-d04e-4b23-
93aa-340df6da25a9) |
|   key_name                                                                  |  mykey
|
|   metadata                                                                    |  {}
|
|   name                                                                       |  vm01
|
|               os-extended-volumes:volumes_attached               |               []
|
|   progress                                                                     |  0
|
|   security_groups                                                         |   default
|
|   status                                                                    |  BUILD
|
|  tenant_id                                        | 408377ef92e4488ba9e3ae0964d4ff3d
|
|  updated                                                        |  2016-05-12T08:41:41Z
|
|  user_id                                          | 1c8c082b3766491f83fca3a006edb8bb
|
+-------------------------------------+-------------------------------------------
-----------------+
amir@openstack01:~/devstack$ nova list
+--------------------------------------+------+--------+------------+-------------+----
-------------------+
| ID                                   | Name | Status | Task State | Power State |
Networks              |
+--------------------------------------+------+--------+------------+-------------+----
-------------------+
| 0ae39159-a402-4391-8dcd-eb86b00ec6ab | vm01 | ACTIVE | -          | Running     |
default=192.168.123.15 |
+--------------------------------------+------+--------+------------+-------------+----
-------------------+
```

7.2.6 OpenStack ukazi za kreiranje dodatnega VLAN omrežja ponudnika

```
amir@openstack01:~/neutron      net-create      provider_external2      --shared      --
provider:physical_network      provider2      --provider:network_type      vlan      --
provider:segmentation_id 160
Created a new network:
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| id                        | 9a931032-c554-7842-8975-964206f3210f |
| name                      | provider_external2                   |
| provider:network_type     | vlan                                 |
| provider:physical_network | provider2                            |
| provider:segmentation_id  | 160                                  |
| router:external           | False                                |
| shared                    | True                                 |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tenant_id                 | a1afrcc4321r8923fd235432651a9062     |
+---------------------------+--------------------------------------+
```

7.3 Konfiguracijske datoteke za postavitev oblačne infrastrukture z L3 povezavo v omrežje ponudnika in uporabo VXLAN najemniških omrežij

7.3.1 Devstack konfiguracija za vozlišče openstack01

```
[[local|localrc]]
HOST_IP=10.0.0.2
SERVICE_HOST=10.0.0.2
MYSQL_HOST=10.0.0.2
RABBIT_HOST=10.0.0.2
GLANCE_HOSTPORT=10.0.0.2:9292
PUBLIC_INTERFACE=eth1

IP_VERSION=4

ADMIN_PASSWORD=password
MYSQL_PASSWORD=password
RABBIT_PASSWORD=password
SERVICE_PASSWORD=password
SERVICE_TOKEN=password

#Ne uporabi Nova-Network
disable_service n-net
#Storitve ki jih zagotavlja Neutron vozlisce
ENABLED_SERVICES+=,q-svc,q-dhcp,q-meta,q-agt,q-l3


#Neutron omrezne moznosti uporabljene za ustvarjanje Neutron podomrezij
```

```
Q_USE_SECGROUP=True
FLOATING_RANGE=192.168.123.0/24
FIXED_RANGE=192.168.1.0/24
Q_FLOATING_ALLOCATION_POOL=start=192.168.123.10,end=192.168.123.30
PUBLIC_NETWORK_GATEWAY=192.168.123.254
NETWORK_GATEWAY=192.168.1.1
Q_L3_ENABLED=True
PUBLIC_INTERFACE=eth1
Q_PLUGIN=ml2
Q_ML2_TENANT_NETWORK_TYPE=vxlan

#Open vSwitch moznosti omrezja ponudnika
Q_USE_PROVIDERNET_FOR_PUBLIC=True
OVS_PHYSICAL_BRIDGE=br-ex
PUBLIC_BRIDGE=br-ex
OVS_BRIDGE_MAPPINGS=public:br-ex

#Tempest
disable_service tempest
```

7.3.2 Omrežna konfiguracija za vozlišče openstack01

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback


#vmesnik eth0 staticen IP naslov
auto eth0
 iface eth0 inet static
     address 10.0.0.2
     netmask 255.255.255.0
     gateway 10.0.0.1
     dns-nameservers 8.8.8.8


#vmesnik eth1 v promiscous nacinu
auto eth1
iface eth1 inet manual
    up ifconfig $IFACE 0.0.0.0 up
    up ip link set $IFACE promisc on
    down ip link set $IFACE promisc off
    down ifconfig $IFACE down


#vmesnik br-ex staticen IP naslov
auto br-ex
   iface br-ex inet static
   address 192.168.123.2
   netmask 255.255.255.0
   gateway 192.168.123.254
```

```
   dns-nameservers 8.8.8.8
```

7.3.3 Devstack konfiguracija za vozlišče openstack02

```
[[local|localrc]]
HOST_IP=10.0.0.3
SERVICE_HOST=10.0.0.2
MYSQL_HOST=10.0.0.2
RABBIT_HOST=10.0.0.2
GLANCE_HOSTPORT=10.0.0.2:9292
ADMIN_PASSWORD=password
MYSQL_PASSWORD=password
RABBIT_PASSWORD=password
SERVICE_PASSWORD=password
SERVICE_TOKEN=password
IP_VERSION=4

#Storitve ki jih zagotavlja Compute vozlisce
ENABLED_SERVICES=n-cpu,rabbit,q-agt

#Open vSwitch moznosti omrezja ponudnika
PUBLIC_INTERFACE=eth1

NOVA_VNC_ENABLED=True
NOVNCPROXY_URL="http://10.0.0.2:6080/vnc_auto.html"
VNCSERVER_LISTEN=$HOST_IP
VNCSERVER_PROXYCLIENT_ADDRESS=$VNCSERVER_LISTEN

# Tempest
disable_service tempest
```

7.3.4 Omrežna konfiguracija za vozlišče openstack02

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

#vmesnik eth0 staticen IP naslov
auto eth0
 iface eth0 inet static
     address 10.0.0.3
     netmask 255.255.255.0
     gateway 10.0.0.1
     dns-nameservers 8.8.8.8


#vmesnik eth1 v promiscous nacinu
auto eth1
iface eth1 inet manual
    up ifconfig $IFACE 0.0.0.0 up
    up ip link set $IFACE promisc on
    down ip link set $IFACE promisc off
```

```
    down ifconfig $IFACE down


#vmesnik br-ex staticen IP naslov
auto br-ex
    iface br-ex inet static
    address 192.168.123.3
    netmask 255.255.255.0
    gateway 192.168.123.254
    dns-nameservers 8.8.8.8
```

7.3.5 OpenStack ukazi za preverjanje obstoječe in kreiranje nove oblačne infrastrukture

Preverjanje uspešnosti izvedbe DevStack skripte in postavitev omrežja ponudnika in omrežja najemnika:

```
amir@openstack01:~/devstack$ neutron net-list
+--------------------------------------+---------+--------------------------------------
----------------+
|  id                                                                  |  name       |  subnets
|
+--------------------------------------+---------+--------------------------------------
----------------+
| 57220c3f-9499-4432-ab5e-75b7658210b2 | private | 9c4cf116-f292-458b-a436-478d6d32e9d4
192.168.1.0/24 |
| 4bf29239-8828-48f0-9db0-46d6a9a1ccbe | public  | 7f987236-6659-4a97-8699-2ad840ea3775
|
+--------------------------------------+---------+----nov-----------------------------
-------------------+
amir@openstack01:~/
```

Kreiranje novega privatnega omrežja najemnika ter podomrežja:

```
amir@openstack01:~/devstack$ neutron net-create private2
Created a new network:
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
| id                    | 37dd8a7b-242e-4263-854b-caf9df1b6074 |
| mtu                   | 1450                                 |
| name                  | private2                             |
| port_security_enabled | True                                 |
| router:external       | False                                |
| shared                | False                                |
| status                | ACTIVE                               |
| subnets               |                                      |
| tenant_id             | 98dad44a2f424dd8ad7824d7f30cb443     |
+-----------------------+--------------------------------------+
amir@openstack01:~/devstack$  neutron  subnet-create  private2  192.168.2.0/24  --name
private2-subnet
Created a new subnet:
+-------------------+------------------------------------------------------+
| Field             | Value                                                |
+-------------------+------------------------------------------------------+
| allocation_pools  | {"start": "192.168.2.2", "end": "192.168.2.254"} |
```

```
| cidr              | 192.168.2.0/24                                  |
| dns_nameservers   |                                                 |
| enable_dhcp       | True                                            |
| gateway_ip        | 192.168.2.1                                     |
| host_routes       |                                                 |
| id                | 7fe38b23-aa68-4664-9573-6c3a8ee9b24f            |
| ip_version        | 4                                               |
| ipv6_address_mode |                                                 |
| ipv6_ra_mode      |                                                 |
| name              | private2-subnet                                 |
| network_id        | 37dd8a7b-242e-4263-854b-caf9df1b6074            |
| subnetpool_id     |                                                 |
| tenant_id         | 98dad44a2f424dd8ad7824d7f30cb443                |
+-------------------+-------------------------------------------------+
amir@openstack01:~/devstack$ neutron subnet-list
+--------------------------------------+-----------------+---------------+------------
--------------------------------------+
| id                                   | name            | cidr          |
allocation_pools                       |
+--------------------------------------+-----------------+---------------+------------
--------------------------------------+
| 7fe38b23-aa68-4664-9573-6c3a8ee9b24f | private2-subnet | 192.168.2.0/24 | {"start":
"192.168.2.2", "end": "192.168.2.254"} |
| 9c4cf116-f292-458b-a436-478d6d32e9d4 | private-subnet  | 192.168.1.0/24 | {"start":
"192.168.1.2", "end": "192.168.1.254"} |
+--------------------------------------+-----------------+---------------+------------
--------------------------------------+
amir@openstack01:~/devstack$
amir@openstack01:~/devstack$ neutron net-show private
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| id                        | 57220c3f-9499-4432-ab5e-75b7658210b2 |
| mtu                       | 1450                                 |
| name                      | private                              |
| port_security_enabled     | True                                 |
| provider:network_type     | vxlan                                |
| provider:physical_network |                                      |
| provider:segmentation_id  | 1068                                 |
| router:external           | False                                |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   | 9c4cf116-f292-458b-a436-478d6d32e9d4 |
| tenant_id                 | 98dad44a2f424dd8ad7824d7f30cb443     |
+---------------------------+--------------------------------------+

amir@openstack01:~/devstack$ neutron net-show private2
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| id                        | 37dd8a7b-242e-4263-854b-caf9df1b6074 |
| mtu                       | 1450                                 |
| name                      | private2                             |
| port_security_enabled     | True                                 |
| provider:network_type     | vxlan                                |
| provider:physical_network |                                      |
| provider:segmentation_id  | 1099                                 |
| router:external           | False                                |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   | 7fe38b23-aa68-4664-9573-6c3a8ee9b24f |
```

```
| tenant_id                | 98dad44a2f424dd8ad7824d7f30cb443     |
+--------------------------+--------------------------------------+

amir@openstack01:~/devstack$ neutron net-list
+--------------------------------------+----------+--------------------------------------
-----------------+
|  id                                                                  |  name         |  subnets
|
+--------------------------------------+----------+--------------------------------------
-----------------+
|   4bf29239-8828-48f0-9db0-46d6a9a1ccbe   |   public        |   7f987236-6659-4a97-8699-
2ad840ea3775                |
|   57220c3f-9499-4432-ab5e-75b7658210b2   |   private       |   9c4cf116-f292-458b-a436-
478d6d32e9d4 192.168.1.0/24 |
|    37dd8a7b-242e-4263-854b-caf9df1b6074   |    private2    |   7fe38b23-aa68-4664-9573-
6c3a8ee9b24f 192.168.2.0/24 |
+--------------------------------------+----------+--------------------------------------
-----------------+
```

Znotraj omrežja najemnika z imenom »private« kreiranje prve virtualne instance z imenom »vm01«:

```
amir@openstack01:~/devstack$ source demo.openrc
amir@openstack01:~/devstack$ nova boot --flavor m1.tiny --image cirros-0.3.4-x86_64-uec
--nic  net-id=57220c3f-9499-4432-ab5e-75b7658210b2  --security-group  default  --key-name
mykey vm01
+--------------------------------------+----------------------------------------------
-----------------+
|  Property                                                                  |  Value
|
+--------------------------------------+----------------------------------------------
-----------------+
|   OS-DCF:diskConfig                                                        |   MANUAL
|
|     OS-EXT-AZ:availability_zone                                                    |
|
|    OS-EXT-STS:power_state                                                      |    0
|
|   OS-EXT-STS:task_state                                                  |   scheduling
|
|   OS-EXT-STS:vm_state                                                    |   building
|
|    OS-SRV-USG:launched_at                                                      |    -
|
|    OS-SRV-USG:terminated_at                                                    |    -
|
|   accessIPv4                                                                       |
|
|   accessIPv6                                                                       |
|
|  adminPass                                                               |  7gLAquFSrzRw
|
|   config_drive                                                                     |
|
|  created                                                                 |  2016-05-13T13:07:35Z
|
|  flavor                                                                  |  m1.tiny  (1)
|
|   hostId                                                                           |
|
```

```
| id                                          | f6e9c464-c6d6-4a23-9707-c0452d4f73e1
|
| image                                  | cirros-0.3.4-x86_64-uec (876fde6e-8a44-4eba-
9901-066d73a48848) |
|   key_name                                                                    |   mykey
|
|   metadata                                                                      |   {}
|
|   name                                                                        |   vm01
|
|               os-extended-volumes:volumes_attached                |                []
|
|   progress                                                                      |   0
|
|   security_groups                                                           |   default
|
|   status                                                                      |   BUILD
|
|  tenant_id                                        |  98dad44a2f424dd8ad7824d7f30cb443
|
|  updated                                                        |  2016-05-13T13:07:36Z
|
|  user_id                                          |  0e9d2570c2724ccd9a1ea790e74efa0a
|
+-------------------------------------+-----------------------------------------------
-----------------+
amir@openstack01:~/devstack$ nova list
+--------------------------------------+------+--------+------------+-------------+----
-----------------+
| ID                                   | Name | Status | Task State | Power State |
Networks            |
+--------------------------------------+------+--------+------------+-------------+----
-----------------+
| f6e9c464-c6d6-4a23-9707-c0452d4f73e1 | vm01 | ACTIVE | -          | Running     |
private=192.168.1.5 |
+--------------------------------------+------+--------+------------+-------------+----
-----------------+
```

Znotraj omrežja najemnika z imenom »private2« kreiranje virtualne instance z imenom »vm02«:

```
amir@openstack01:~/devstack$ source demo.openrc
amir@openstack01:~/devstack$ nova boot --flavor m1.tiny --image cirros-0.3.4-x86_64-uec
--nic net-id=37dd8a7b-242e-4263-854b-caf9df1b6074 --security-group default --key-name
mykey vm02

+-------------------------------------+-----------------------------------------------
-----------------+
|   Property                                                                    |   Value
|
+-------------------------------------+-----------------------------------------------
-----------------+
|   OS-DCF:diskConfig                                                         |   MANUAL
|
|     OS-EXT-AZ:availability_zone                                                 |
|
|    OS-EXT-STS:power_state                                                       |   0
|
|   OS-EXT-STS:task_state                                                   |   scheduling
|
|   OS-EXT-STS:vm_state                                                       |   building
|
```

```
|    OS-SRV-USG:launched_at                                                    |    -
|
|    OS-SRV-USG:terminated_at                                                  |    -
|
|   accessIPv4                                                                 |
|
|   accessIPv6                                                                 |
|
|  adminPass                                                          |  Ai7rnV8uRxQA
|
|   config_drive                                                               |
|
|  created                                                       |  2016-05-15T11:00:21Z
|
|  flavor                                                             |  m1.tiny  (1)
|
|   hostId                                                                     |
|
| id                                          | 174c0caa-3e8a-46ea-aacf-206e2f25c20b
|
| image                                    | cirros-0.3.4-x86_64-uec (876fde6e-8a44-4eba-
9901-066d73a48848) |
|   key_name                                                                 |  mykey
|
|   metadata                                                                   |  {}
|
|   name                                                                     |  vm02
|
|               os-extended-volumes:volumes_attached               |               []
|
|   progress                                                                   |  0
|
|   security_groups                                                        |  default
|
|   status                                                                   |  BUILD
|
|  tenant_id                                        |  98dad44a2f424dd8ad7824d7f30cb443
|
|  updated                                                       |  2016-05-15T11:00:21Z
|
|  user_id                                          |  0e9d2570c2724ccd9a1ea790e74efa0a
|
+--------------------------------------+-------------------------------------------
-----------------+
amir@openstack01:~/devstack$ nova list
+--------------------------------------+------+--------+------------+-------------+----
---------------------------------+
| ID                                   | Name | Status | Task State | Power State |
Networks                         |
+--------------------------------------+------+--------+------------+-------------+----
---------------------------------+
| 6e708de3-0e78-4807-9d34-b063ae7304cb | vm01 | ACTIVE | -          | Running     |
private=192.168.1.5              |
| 174c0caa-3e8a-46ea-aacf-206e2f25c20b | vm02 | ACTIVE | -          | Running     |
private2=192.168.2.3             |
+--------------------------------------+------+--------+------------+-------------+----
---------------------------------+
```

Preverjanje razpoložljivosti plavajočih IP naslovov, kreiranje dodatnega ter dodelitev virtualnim instancam:

```
amir@openstack01:~/devstack$ nova floating-ip-list
+--------------------------------------+----------------+-----------+----------+-------
-+
| Id                                   | IP             | Server Id | Fixed IP | Pool
|
+--------------------------------------+----------------+-----------+----------+-------
-+
| 528ac2d2-7942-4153-acac-2b9692a8a711 | 192.168.123.11 | -         | -        | public
|
+--------------------------------------+----------------+-----------+----------+-------
-+
amir@openstack01:~/devstack$ nova floating-ip-create
+--------------------------------------+----------------+-----------+----------+-------
-+
| Id                                   | IP             | Server Id | Fixed IP | Pool
|
+--------------------------------------+----------------+-----------+----------+-------
-+
| d5f5b586-ab62-4a9d-9316-2572d0d2f09f | 192.168.123.12 | -         | -        | public
|
+--------------------------------------+----------------+-----------+----------+-------
-+

amir@openstack01:~/devstack$ nova floating-ip-associate vm01 192.168.123.11
amir@openstack01:~/devstack$ nova floating-ip-associate vm02 192.168.123.12
amir@openstack01:~/devstack$ nova list
+--------------------------------------+------+--------+------------+-------------+----
-----------------------------------+
| ID                                   | Name | Status | Task State | Power State |
Networks                           |
+--------------------------------------+------+--------+------------+-------------+----
-----------------------------------+
| 6e708de3-0e78-4807-9d34-b063ae7304cb | vm01 | ACTIVE | -          | Running     |
private=192.168.1.5, 192.168.123.11  |
| 174c0caa-3e8a-46ea-aacf-206e2f25c20b | vm02 | ACTIVE | -          | Running     |
private2=192.168.2.3, 192.168.123.12 |
+--------------------------------------+------+--------+------------+-------------+----
-----------------------------------+
amir@openstack01:~/devstack$
```

7.3.6 Postopek preverjanja omrežne povezljivosti

```
amir@ubuntuos:~$ ssh cirros@192.168.123.11
The authenticity of host '192.168.123.11 (192.168.123.11)' can't be established.
RSA key fingerprint is d9:2f:49:59:3d:f9:ca:c0:c1:d1:b4:ef:1d:ce:90:a5.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.123.11' (RSA) to the list of known hosts.
cirros@192.168.123.11's password:
$ ifconfig
eth0      Link encap:Ethernet  HWaddr FA:16:3E:C0:AE:4F
          inet addr:192.168.1.5  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:fec0:ae4f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1450  Metric:1
          RX packets:460 errors:0 dropped:0 overruns:0 frame:0
          TX packets:339 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:49313 (48.1 KiB)  TX bytes:36072 (35.2 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
```

```
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

$ ping -c4 192.168.2.3
PING 192.168.2.3 (192.168.2.3): 56 data bytes
64 bytes from 192.168.2.3: seq=0 ttl=63 time=4.184 ms
64 bytes from 192.168.2.3: seq=1 ttl=63 time=2.203 ms
64 bytes from 192.168.2.3: seq=2 ttl=63 time=2.409 ms
64 bytes from 192.168.2.3: seq=3 ttl=63 time=1.936 ms

--- 192.168.2.3 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 1.936/2.683/4.184 ms
$ ping -c4 192.168.123.12
PING 192.168.123.12 (192.168.123.12): 56 data bytes
64 bytes from 192.168.123.12: seq=0 ttl=63 time=4.520 ms
64 bytes from 192.168.123.12: seq=1 ttl=63 time=2.143 ms
64 bytes from 192.168.123.12: seq=2 ttl=63 time=2.064 ms
64 bytes from 192.168.123.12: seq=3 ttl=63 time=2.076 ms

--- 192.168.123.12 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 2.064/2.700/4.520 ms
$ ping -c4 8.8.8.8
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: seq=0 ttl=58 time=16.714 ms
64 bytes from 8.8.8.8: seq=1 ttl=58 time=11.611 ms
64 bytes from 8.8.8.8: seq=2 ttl=58 time=12.518 ms
64 bytes from 8.8.8.8: seq=3 ttl=58 time=11.577 ms

--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 11.577/13.105/16.714 ms
$ exit
Connection to 192.168.123.11 closed.
amir@ubuntuos:~$
```

7.3.7 OpenStack ukazi za kreiranje dodatne oblačne infrastrukture in testiranje omrežne povezljivosti

Kreiranje dodatnega virtualnega usmerjevalnika »router2« in konfiguracija:

```
amir@openstack01:~/devstack$ neutron router-create router2
Created a new router:
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
| external_gateway_info |                                      |
| id                    | 06637d8c-706e-42be-ae7b-6178ef975c2d |
| name                  | router2                              |
| routes                |                                      |
| status                | ACTIVE                               |
| tenant_id             | 98dad44a2f424dd8ad7824d7f30cb443     |
+-----------------------+--------------------------------------+
amir@openstack01:~/devstack$ neutron router-gateway-set router2 public
Set gateway for router router2
amir@openstack01:~/devstack$ neutron router-interface-add router2 private2-subnet
```

```
Added interface bcf04352-d27f-4da5-b9c7-f4128c688e87 to router router2.
```

Vzpostavitev SSH seje z virtualno instanco »vm01« ter uporaba ukaza ping in preverjanje povezljivosti do virtualne instance »vm02« in zunanjih omrežnih elementov:

```
amir@ubuntuos:~$ ssh cirros@192.168.123.11
cirros@192.168.123.11's password:
$ ifconfig
eth0      Link encap:Ethernet  HWaddr FA:16:3E:C0:AE:4F
          inet addr:192.168.1.5  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:fec0:ae4f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1450  Metric:1
          RX packets:743 errors:0 dropped:0 overruns:0 frame:0
          TX packets:531 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:77128 (75.3 KiB)  TX bytes:59850 (58.4 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)


$ ping -c4 192.168.2.3
PING 192.168.2.3 (192.168.2.3): 56 data bytes

--- 192.168.2.3 ping statistics ---
4 packets transmitted, 0 packets received, 100% packet loss
$ ping -c4 192.168.123.12
PING 192.168.123.12 (192.168.123.12): 56 data bytes
64 bytes from 192.168.123.12: seq=0 ttl=62 time=8.998 ms
64 bytes from 192.168.123.12: seq=1 ttl=62 time=2.225 ms
64 bytes from 192.168.123.12: seq=2 ttl=62 time=2.064 ms
64 bytes from 192.168.123.12: seq=3 ttl=62 time=2.207 ms

--- 192.168.123.12 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 2.064/3.873/8.998 ms
$ ping -c4 8.8.8.8
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: seq=0 ttl=58 time=16.584 ms
64 bytes from 8.8.8.8: seq=1 ttl=58 time=12.006 ms
64 bytes from 8.8.8.8: seq=2 ttl=58 time=11.715 ms
64 bytes from 8.8.8.8: seq=3 ttl=58 time=11.485 ms

--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 11.485/12.947/16.584 ms
$ exit
```

Ponovitev prejšnjega postopka na virtualni instanci »vm02«:

```
amir@ubuntuos:~$ ssh cirros@192.168.123.12
cirros@192.168.123.12's password:
$ ifconfig
eth0      Link encap:Ethernet  HWaddr FA:16:3E:13:69:90
```

```
          inet addr:192.168.2.3  Bcast:192.168.2.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:fe13:6990/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1450  Metric:1
          RX packets:312 errors:0 dropped:0 overruns:0 frame:0
          TX packets:227 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:35878 (35.0 KiB)  TX bytes:26536 (25.9 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:6 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:672 (672.0 B)  TX bytes:672 (672.0 B)

$ ping -c4 192.168.1.5
PING 192.168.1.5 (192.168.1.5): 56 data bytes

--- 192.168.1.5 ping statistics ---
4 packets transmitted, 0 packets received, 100% packet loss
$ ping -c4 192.168.123.11
PING 192.168.123.11 (192.168.123.11): 56 data bytes
64 bytes from 192.168.123.11: seq=0 ttl=62 time=5.802 ms
64 bytes from 192.168.123.11: seq=1 ttl=62 time=2.458 ms
64 bytes from 192.168.123.11: seq=2 ttl=62 time=2.202 ms
64 bytes from 192.168.123.11: seq=3 ttl=62 time=2.031 ms

--- 192.168.123.11 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 2.031/3.123/5.802 ms
$ ping -c4 8.8.8.8
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: seq=0 ttl=58 time=15.122 ms
64 bytes from 8.8.8.8: seq=1 ttl=58 time=12.049 ms
64 bytes from 8.8.8.8: seq=2 ttl=58 time=11.408 ms
64 bytes from 8.8.8.8: seq=3 ttl=58 time=12.155 ms

--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 11.408/12.683/15.122 ms
$ exit
Connection to 192.168.123.12 closed.
amir@ubuntuos:~$
```

7.4 Konfiguracijske datoteke za postavitev oblačne infrastrukture z L3 visoko razpoložljivo povezavo v omrežje ponudnika

7.4.1 Devstack konfiguracija za vozlišče openstack01

```
[[local|localrc]]
HOST_IP=10.0.0.2
SERVICE_HOST=10.0.0.2
MYSQL_HOST=10.0.0.2
RABBIT_HOST=10.0.0.2
```

```
GLANCE_HOSTPORT=10.0.0.2:9292
PUBLIC_INTERFACE=eth1

IP_VERSION=4

ADMIN_PASSWORD=password
MYSQL_PASSWORD=password
RABBIT_PASSWORD=password
SERVICE_PASSWORD=password
SERVICE_TOKEN=password

#Ne uporabi Nova-Network
disable_service n-net
#Storitve ki jih zagotavlja Compute in Neutron vozlisce
ENABLED_SERVICES+=,n-cpu,q-svc,q-meta,q-agt


#Neutron omrezne moznosti uporabljene za ustvarjanje Neutron podomrezij
Q_USE_SECGROUP=True
FLOATING_RANGE=192.168.123.0/24
FIXED_RANGE=192.168.1.0/24
Q_FLOATING_ALLOCATION_POOL=start=192.168.123.10,end=192.168.123.30
PUBLIC_NETWORK_GATEWAY=192.168.123.254
NETWORK_GATEWAY=192.168.1.1
Q_L3_ENABLED=False
Q_PLUGIN=ml2
Q_ML2_TENANT_NETWORK_TYPE=vxlan

#Open vSwitch moznosti omrezja ponudnika
Q_USE_PROVIDERNET_FOR_PUBLIC=True
OVS_PHYSICAL_BRIDGE=br-ex
PUBLIC_BRIDGE=br-ex
OVS_BRIDGE_MAPPINGS=public:br-ex

#Tempest
disable_service tempest

[[post-config|$NEUTRON_CONF]]
[DEFAULT]
core_plugin=ml2
service_plugins=router
allow_overlapping_ips=True
l3_ha=True
dhcp_agents_per_network=2

[[post-config|/$Q_PLUGIN_CONF_FILE]]
[ml2]
type_drivers=flat,vlan,vxlan
tenant_network_types=vxlan
mechanism_drivers=openvswitch,l2population
```

```
[ml2_type_vxlan]
vni_ranges=2000:3000

[ovs]
local_ip=$HOST_IP

[agent]
tunnel_types=vxlan
l2_population=True


[[post-config|$Q_L3_CONF_FILE]]
[DEFAULT]
interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
external_network_bridge =

[[post-config|$Q_DHCP_CONF_FILE]]
[DEFAULT]
interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
enable_isolated_metadata = True
```

7.4.2 Omrežna konfiguracija za vozlišče openstack01

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback


#vmesnik eth0 staticen IP naslov
auto eth0
 iface eth0 inet static
     address 10.0.0.2
     netmask 255.255.255.0
     gateway 10.0.0.1
     dns-nameservers 8.8.8.8


#vmesnik eth1 v promiscous nacinu
auto eth1
iface eth1 inet manual
    up ifconfig $IFACE 0.0.0.0 up
    up ip link set $IFACE promisc on
    down ip link set $IFACE promisc off
    down ifconfig $IFACE down


#vmesnik br-ex staticen IP naslov
auto br-ex
   iface br-ex inet static
   address 192.168.123.2
   netmask 255.255.255.0
```

```
   gateway 192.168.123.254
   dns-nameservers 8.8.8.8
```

7.4.3 Devstack konfiguracija za vozlišče openstack02

```
[[local|localrc]]
HOST_IP=10.0.0.3
SERVICE_HOST=10.0.0.2
MYSQL_HOST=10.0.0.2
RABBIT_HOST=10.0.0.2
GLANCE_HOSTPORT=10.0.0.2:9292
ADMIN_PASSWORD=password
MYSQL_PASSWORD=password
RABBIT_PASSWORD=password
SERVICE_PASSWORD=password
SERVICE_TOKEN=password
IP_VERSION=4

#Storitve ki jih zagotavlja Neutron vozlisce
ENABLED_SERVICES=neutron,rabbit,q-dhcp,q-agt,q-l3,q-meta

#Open vSwitch moznosti omrezja ponudnika
PUBLIC_INTERFACE=eth1

Q_L3_ENABLED=True
Q_PLUGIN=ml2
Q_ML2_TENANT_NETWORK_TYPE=vxlan

NOVA_VNC_ENABLED=True
NOVNCPROXY_URL="http://10.0.0.2:6080/vnc_auto.html"
VNCSERVER_LISTEN=$HOST_IP
VNCSERVER_PROXYCLIENT_ADDRESS=$VNCSERVER_LISTEN

# Tempest
disable_service tempest

[[post-config|$NEUTRON_CONF]]
[DEFAULT]
core_plugin=ml2
service_plugins=router
allow_overlapping_ips=True
l3_ha=True
dhcp_agents_per_network=2

[[post-config|/$Q_PLUGIN_CONF_FILE]]
[ml2]
type_drivers=flat,vlan,vxlan
tenant_network_types=vxlan
mechanism_drivers=openvswitch,l2population

[ml2_type_vxlan]
```

```
vni_ranges=2000:3000

[ovs]
local_ip=$HOST_IP

[agent]
tunnel_types=vxlan
l2_population=True


[[post-config|$Q_L3_CONF_FILE]]
[DEFAULT]
interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
external_network_bridge =

[[post-config|$Q_DHCP_CONF_FILE]]
[DEFAULT]
interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
enable_isolated_metadata = True
```

7.4.4 Omrežna konfiguracija za vozlišče openstack02

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

#vmesnik eth0 staticen IP naslov
auto eth0
 iface eth0 inet static
     address 10.0.0.3
     netmask 255.255.255.0
     gateway 10.0.0.1
     dns-nameservers 8.8.8.8


#vmesnik eth1 v promiscous nacinu
auto eth1
iface eth1 inet manual
    up ifconfig $IFACE 0.0.0.0 up
    up ip link set $IFACE promisc on
    down ip link set $IFACE promisc off
    down ifconfig $IFACE down


#vmesnik br-ex staticen IP naslov
auto br-ex
   iface br-ex inet static
   address 192.168.123.3
   netmask 255.255.255.0
   gateway 192.168.123.254
```

```
   dns-nameservers 8.8.8.8
```

7.4.5 Devstack konfiguracija za vozlišče openstack03

```
[[local|localrc]]
HOST_IP=10.0.0.4
SERVICE_HOST=10.0.0.2
MYSQL_HOST=10.0.0.2
RABBIT_HOST=10.0.0.2
GLANCE_HOSTPORT=10.0.0.2:9292
ADMIN_PASSWORD=password
MYSQL_PASSWORD=password
RABBIT_PASSWORD=password
SERVICE_PASSWORD=password
SERVICE_TOKEN=password
IP_VERSION=4

#Storitve ki jih zagotavlja Neutron vozlisce
ENABLED_SERVICES=neutron,rabbit,q-dhcp,q-agt,q-l3,q-meta

# Open vSwitch moznosti omrezja ponudnika
PUBLIC_INTERFACE=eth1

Q_L3_ENABLED=True
Q_PLUGIN=ml2
Q_ML2_TENANT_NETWORK_TYPE=vxlan

NOVA_VNC_ENABLED=True
NOVNCPROXY_URL="http://10.0.0.2:6080/vnc_auto.html"
VNCSERVER_LISTEN=$HOST_IP
VNCSERVER_PROXYCLIENT_ADDRESS=$VNCSERVER_LISTEN

# Tempest
disable_service tempest

[[post-config|$NEUTRON_CONF]]
[DEFAULT]
core_plugin=ml2
service_plugins=router
allow_overlapping_ips=True
l3_ha=True
dhcp_agents_per_network=2

[[post-config|/$Q_PLUGIN_CONF_FILE]]
[ml2]
type_drivers=flat,vlan,vxlan
tenant_network_types=vxlan
mechanism_drivers=openvswitch,l2population

[ml2_type_vxlan]
vni_ranges=2000:3000
```

```
[ovs]
local_ip=$HOST_IP

[agent]
tunnel_types=vxlan
l2_population=True


[[post-config|$Q_L3_CONF_FILE]]
[DEFAULT]
interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
external_network_bridge =

[[post-config|$Q_DHCP_CONF_FILE]]
[DEFAULT]
interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
enable_isolated_metadata = True
```

7.4.6 Omrežna konfiguracija za vozlišče openstack03

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

#vmesnik eth0 staticen IP naslov
auto eth0
 iface eth0 inet static
     address 10.0.0.4
     netmask 255.255.255.0
     gateway 10.0.0.1
     dns-nameservers 8.8.8.8


#vmesnik eth1 v promiscous nacinu
auto eth1
iface eth1 inet manual
    up ifconfig $IFACE 0.0.0.0 up
    up ip link set $IFACE promisc on
    down ip link set $IFACE promisc off
    down ifconfig $IFACE down


#vmesnik br-ex staticen IP naslov
auto br-ex
   iface br-ex inet static
   address 192.168.123.4
   netmask 255.255.255.0
   gateway 192.168.123.254
   dns-nameservers 8.8.8.8
```

7.4.7 OpenStack ukaz za preverjanje statusa omrežnih servisov

Preverjanje statusa omrežnih servisov na OpenStack vozliščih:

```
amir@openstack01:~/devstack$ neutron agent-list
+--------------------------------------+--------------------+------------+-------+----
------------+---------------------------+
| id                                   | agent_type         | host       | alive |
admin_state_up | binary                    |
+--------------------------------------+--------------------+------------+-------+----
------------+---------------------------+
| 100e9464-c770-4cc7-a821-6c4c32fd1b8c | Metadata agent     | openstack03 | :-)    |
True           | neutron-metadata-agent    |
| 13988274-d930-426a-9ca2-92ff02c298c6 | Open vSwitch agent | openstack02 | :-)    |
True           | neutron-openvswitch-agent |
| 2b2472aa-ec05-465c-8b0c-a9730ca22657 | DHCP agent         | openstack01 | :-)    |
True           | neutron-dhcp-agent        |
| 37a7f8d7-07b9-4f4a-ba20-5c3c61e625f3 | DHCP agent         | openstack03 | :-)    |
True           | neutron-dhcp-agent        |
| 3aeaeadd-815a-4117-956b-f35e1f5ec8a7 | Metadata agent     | openstack01 | :-)    |
True           | neutron-metadata-agent    |
| 50c85c8e-2761-4963-b07b-eb355d1141c7 | L3 agent           | openstack01 | :-)    |
True           | neutron-l3-agent          |
| 7b672e33-6408-458c-b83b-ed3986d7cbac | L3 agent           | openstack03 | :-)    |
True           | neutron-l3-agent          |
| 83fdc33e-74e4-4053-a0fd-c9e7a2203057 | Open vSwitch agent | openstack01 | :-)    |
True           | neutron-openvswitch-agent |
| 91f0de60-0424-499d-b780-f6ed6fec4c85 | DHCP agent         | openstack02 | :-)    |
True           | neutron-dhcp-agent        |
| a01880c5-2176-4072-80f2-98bacea7480f | Metadata agent     | openstack02 | :-)    |
True           | neutron-metadata-agent    |
| e3ca7099-f44f-412b-932e-1541fd025ede | Open vSwitch agent | openstack03 | :-)    |
True           | neutron-openvswitch-agent |
| ff223e6f-796b-41fe-b4e6-143156083b36 | L3 agent           | openstack02 | :-)    |
True           | neutron-l3-agent          |
+--------------------------------------+--------------------+------------+-------+----
------------+---------------------------+
```

7.4.8 OpenStack ukazi za kreiranje L3 visoko razpoložljive oblačne infrastrukture

```
amir@openstack01:~/devstack$ source admin.openrc amir@openstack01:~/devstack$ neutron
net-create public --router:external True --provider:physical_network external --
provider:network_type flat
Created a new network:
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| id                        | ce1d6fd9-3c07-423f-9840-1406dc2fb512 |
| mtu                       | 0                                    |
| name                      | public                               |
| port_security_enabled     | True                                 |
| provider:network_type     | flat                                 |
| provider:physical_network | external                             |
| provider:segmentation_id  |                                      |
| router:external           | True                                 |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
```

```
| tenant_id                 | a2c728f3f8b14c1888e1bba0bdfaa634     |
+---------------------------+--------------------------------------+
amir@openstack01:~/devstack$  neutron  subnet-create  public  192.168.123.0/24  --name
public-subnet --allocation-pool start=192.168.123.10,end=192.168.123.30 --disable-dhcp
--gateway 192.168.123.254
Created a new subnet:
+-------------------+-------------------------------------------------------+
| Field             | Value                                                 |
+-------------------+-------------------------------------------------------+
| allocation_pools  | {"start": "192.168.123.10", "end": "192.168.123.30"} |
| cidr              | 192.168.123.0/24                                      |
| dns_nameservers   |                                                       |
| enable_dhcp       | False                                                 |
| gateway_ip        | 192.168.123.254                                       |
| host_routes       |                                                       |
| id                | acd13de1-34b1-414e-873d-c1203a91836d                  |
| ip_version        | 4                                                     |
| ipv6_address_mode |                                                       |
| ipv6_ra_mode      |                                                       |
| name              | public-subnet                                         |
| network_id        | ce1d6fd9-3c07-423f-9840-1406dc2fb512                  |
| subnetpool_id     |                                                       |
| tenant_id         | a2c728f3f8b14c1888e1bba0bdfaa634                      |
+-------------------+-------------------------------------------------------+
amir@openstack01:~/devstack$ openstack project show demo
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description |                                  |
| enabled     | True                             |
| id          | a2c728f3f8b14c1888e1bba0bdfaa634 |
| name        | demo                             |
+-------------+----------------------------------+


amir@openstack01:~/devstack$      neutron      net-create      private      --tenant-id
a2c728f3f8b14c1888e1bba0bdfaa634 --provider:network_type vxlan
Created a new network:
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| id                        | 1d959fab-9f33-4071-a50e-7dc5c596c44a |
| mtu                       | 1450                                 |
| name                      | private                              |
| port_security_enabled     | True                                 |
| provider:network_type     | vxlan                                |
| provider:physical_network |                                      |
| provider:segmentation_id  | 2031                                 |
| router:external           | False                                |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tenant_id                 | a2c728f3f8b14c1888e1bba0bdfaa634     |
+---------------------------+--------------------------------------+
amir@openstack01:~/devstack$ source demo.openrc
amir@openstack01:~/devstack$  neutron  subnet-create  private  192.168.1.0/24  --name
private-subnet --gateway 192.168.1.1
Created a new subnet:
+-------------------+-------------------------------------------------+
| Field             | Value                                           |
+-------------------+-------------------------------------------------+
| allocation_pools  | {"start": "192.168.1.2", "end": "192.168.1.254"} |
```

```
| cidr              | 192.168.1.0/24                                     |
| dns_nameservers   |                                                    |
| enable_dhcp       | True                                               |
| gateway_ip        | 192.168.1.1                                        |
| host_routes       |                                                    |
| id                | f67fc58c-9cba-4932-a884-f6991cbce4fc               |
| ip_version        | 4                                                  |
| ipv6_address_mode |                                                    |
| ipv6_ra_mode      |                                                    |
| name              | private-subnet                                     |
| network_id        | 1d959fab-9f33-4071-a50e-7dc5c596c44a               |
| subnetpool_id     |                                                    |
| tenant_id         | a2c728f3f8b14c1888e1bba0bdfaa634                   |
+-------------------+----------------------------------------------------+
```

Kreiranje dodatnega virtualnega usmerjevalnika z imenom »private-router«:

```
amir@openstack01:~/devstack$ neutron router-create private-router
Created a new router:
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
| external_gateway_info |                                      |
| id                    | 875da198-1b78-466f-8c35-959c6e47a4aa |
| name                  | private-router                       |
| routes                |                                      |
| status                | ACTIVE                               |
| tenant_id             | a2c728f3f8b14c1888e1bba0bdfaa634     |
+-----------------------+--------------------------------------+
amir@openstack01:~/devstack$ neutron router-interface-add private-router private-subnet
Added interface 777bbcfd-d9fa-48ca-993a-f7d93cf637a3 to router private-router.
amir@openstack01:~/devstack$ neutron router-gateway-set private-router public
Set gateway for router private-router
```

Preverjanje uspešnosti kreiranja HA omrežja:

```
amir@openstack01:~/devstack$ source admin.openrc

amir@openstack01:~/devstack$ neutron net-list
+--------------------------------------+----------------------------------------------
-----+-------------------------------------------------------+
|  id                                                                                  |  name
| subnets                                                |
+--------------------------------------+----------------------------------------------
-----+-------------------------------------------------------+
|              1d959fab-9f33-4071-a50e-7dc5c596c44a              |              private
| f67fc58c-9cba-4932-a884-f6991cbce4fc 192.168.1.0/24   |
|               ce1d6fd9-3c07-423f-9840-1406dc2fb512               |               public
| acd13de1-34b1-414e-873d-c1203a91836d 192.168.123.0/24 |
|       ed40c56d-a522-49d1-95b3-2a845521a8ea       |       HA       network       tenant
a2c728f3f8b14c1888e1bba0bdfaa634         |         fb9344e5-d42f-494d-b462-2348dd69b854
169.254.192.0/18 |
+--------------------------------------+----------------------------------------------
-----+-------------------------------------------------------+
```

Preverjanje uspešnosti kreiranja dodatnega virtualnega usmerjevalnika na omrežnem vozlišču:

```
amir@openstack01:~/devstack$ neutron l3-agent-list-hosting-router private-router
+--------------------------------------+-------------+----------------+-------+--------
--+
| id                                   | host        | admin_state_up | alive |
ha_state |
+--------------------------------------+-------------+----------------+-------+--------
--+
| 50c85c8e-2761-4963-b07b-eb355d1141c7 | openstack01 | True           | :-)   | standby
|
| ff223e6f-796b-41fe-b4e6-143156083b36 | openstack02 | True           | :-)   | active
|
| 7b672e33-6408-458c-b83b-ed3986d7cbac | openstack03 | True           | :-)   | standby
|
+--------------------------------------+-------------+----------------+-------+--------
--+
```

Preverjanje uspešnosti kreiranja HA vmesnikov na virtualnem usmerjevalniku »private-router«:

```
amir@openstack01:~/devstack$ neutron router-port-list private-router
+--------------------------------------+-----------------------------------------------
--+-------------------+-------------------------------------------------------------------
-----------------------+
|  id                                                                                  |  name
|       mac_address                                                 |       fixed_ips
|
+--------------------------------------+-----------------------------------------------
--+-------------------+-------------------------------------------------------------------
-----------------------+
|       02091c6f-b04a-4038-9acb-8d4725fcdf12       |       HA       port       tenant
a2c728f3f8b14c1888e1bba0bdfaa634  |  fa:16:3e:f4:a8:f1  |  {"subnet_id":  "fb9344e5-d42f-
494d-b462-2348dd69b854", "ip_address": "169.254.192.2"}  |
|                          32b5f55a-f78f-487a-9f83-6f2b99e8010a                        |
|    fa:16:3e:73:66:2b    |    {"subnet_id":    "acd13de1-34b1-414e-873d-c1203a91836d",
"ip_address": "192.168.123.10"} |
|                          777bbcfd-d9fa-48ca-993a-f7d93cf637a3                        |
|    fa:16:3e:88:0e:83    |    {"subnet_id":    "f67fc58c-9cba-4932-a884-f6991cbce4fc",
"ip_address": "192.168.1.1"}    |
|       8a9fe864-0b10-41b8-9fba-7ccb2ef8b313       |       HA       port       tenant
a2c728f3f8b14c1888e1bba0bdfaa634  |  fa:16:3e:73:44:6c  |  {"subnet_id":  "fb9344e5-d42f-
494d-b462-2348dd69b854", "ip_address": "169.254.192.1"}  |
|       fb61cca6-5119-4083-8298-366e9179cf4b       |       HA       port       tenant
a2c728f3f8b14c1888e1bba0bdfaa634  |  fa:16:3e:b0:f7:c2  |  {"subnet_id":  "fb9344e5-d42f-
494d-b462-2348dd69b854", "ip_address": "169.254.192.3"}  |
+--------------------------------------+-----------------------------------------------
--+-------------------+-------------------------------------------------------------------
-----------------------+
```

Preverjanje uspešnosti kreiranja imenskega prostora (*ang. namespaces*) »qrouter« na vozliščih:

```
amir@openstack01:~/devstack$ ip netns
qrouter-875da198-1b78-466f-8c35-959c6e47a4aa

amir@openstack02:~$ ip netns
qrouter-875da198-1b78-466f-8c35-959c6e47a4aa

amir@openstack03:~/devstack$ ip netns
qrouter-875da198-1b78-466f-8c35-959c6e47a4aa
```

Opomba: vsi trije imenski prostori morajo imeti enako ime.

Preverjanje HA načina delovanja na omrežnih vozliščih:

```
amir@openstack01:~/devstack$ sudo ip netns exec qrouter-875da198-1b78-466f-8c35-
959c6e47a4aa ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
15: ha-02091c6f-b0: <BROADCAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group
default
    link/ether fa:16:3e:f4:a8:f1 brd ff:ff:ff:ff:ff:ff
    inet 169.254.192.2/18 brd 169.254.255.255 scope global ha-02091c6f-b0
       valid_lft forever preferred_lft forever
    inet6 fe80::f816:3eff:fef4:a8f1/64 scope link
       valid_lft forever preferred_lft forever
16: qr-777bbcfd-d9: <BROADCAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group
default
    link/ether fa:16:3e:88:0e:83 brd ff:ff:ff:ff:ff:ff
17: qg-32b5f55a-f7: <BROADCAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group
default
    link/ether fa:16:3e:73:66:2b brd ff:ff:ff:ff:ff:ff

amir@openstack02:~$ sudo ip netns exec qrouter-875da198-1b78-466f-8c35-959c6e47a4aa ip
addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
17: ha-8a9fe864-0b: <BROADCAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group
default
    link/ether fa:16:3e:73:44:6c brd ff:ff:ff:ff:ff:ff
    inet 169.254.192.1/18 brd 169.254.255.255 scope global ha-8a9fe864-0b
       valid_lft forever preferred_lft forever
    inet 169.254.0.1/24 scope global ha-8a9fe864-0b
       valid_lft forever preferred_lft forever
    inet6 fe80::f816:3eff:fe73:446c/64 scope link
       valid_lft forever preferred_lft forever
18: qr-777bbcfd-d9: <BROADCAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group
default
    link/ether fa:16:3e:88:0e:83 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.1/24 scope global qr-777bbcfd-d9
       valid_lft forever preferred_lft forever
    inet6 fe80::f816:3eff:fe88:e83/64 scope link tentative dadfailed
       valid_lft forever preferred_lft forever
19: qg-32b5f55a-f7: <BROADCAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group
default
    link/ether fa:16:3e:73:66:2b brd ff:ff:ff:ff:ff:ff
    inet 192.168.123.10/24 scope global qg-32b5f55a-f7
       valid_lft forever preferred_lft forever
    inet6 fe80::f816:3eff:fe73:662b/64 scope link
       valid_lft forever preferred_lft forever



amir@openstack03:~/devstack$ ip netns exec qrouter-875da198-1b78-466f-8c35-959c6e47a4aa
ip addr show
seting the network namespace "qrouter-875da198-1b78-466f-8c35-959c6e47a4aa" failed:
Operation not permitted
```

```
amir@openstack03:~/devstack$  sudo  ip  netns  exec  qrouter-875da198-1b78-466f-8c35-
959c6e47a4aa ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
20: ha-fb61cca6-51: <BROADCAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group
default
    link/ether fa:16:3e:b0:f7:c2 brd ff:ff:ff:ff:ff:ff
    inet 169.254.192.3/18 brd 169.254.255.255 scope global ha-fb61cca6-51
       valid_lft forever preferred_lft forever
    inet 169.254.0.1/24 scope global ha-fb61cca6-51
       valid_lft forever preferred_lft forever
    inet6 fe80::f816:3eff:feb0:f7c2/64 scope link
       valid_lft forever preferred_lft forever
21: qr-777bbcfd-d9: <BROADCAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group
default
    link/ether fa:16:3e:88:0e:83 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.1/24 scope global qr-777bbcfd-d9
       valid_lft forever preferred_lft forever
    inet6 fe80::f816:3eff:fe88:e83/64 scope link
       valid_lft forever preferred_lft forever
22: qg-32b5f55a-f7: <BROADCAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group
default
    link/ether fa:16:3e:73:66:2b brd ff:ff:ff:ff:ff:ff
    inet6 fe80::f816:3eff:fe73:662b/64 scope link
       valid_lft forever preferred_lft forever
```

Opomba: Vsi imenski prostori »qrouter« morajo imeti »ha«, »qr« in »qg« vmesnike.

7.4.9 OpenStack ukazi za preverjanje oglaševanja protokola VRRP

```
amir@openstack01:~/devstack$ sudo tcpdump -lnpi eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
IP 169.254.192.1 > 224.0.0.18: VRRPv2, Advertisement, vrid 1, prio 50, authtype none,
intvl 2s, length 20
IP 169.254.192.1 > 224.0.0.18: VRRPv2, Advertisement, vrid 1, prio 50, authtype none,
intvl 2s, length 20


amir@openstack02:~$ sudo tcpdump -lnpi eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
IP 169.254.192.1 > 224.0.0.18: VRRPv2, Advertisement, vrid 1, prio 50, authtype none,
intvl 2s, length 20
IP 169.254.192.1 > 224.0.0.18: VRRPv2, Advertisement, vrid 1, prio 50, authtype none,
intvl 2s, length 20


amir@openstack03:~/devstack$ sudo tcpdump -lnpi eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 65535 bytes
IP 169.254.192.1 > 224.0.0.18: VRRPv2, Advertisement, vrid 1, prio 50, authtype none,
intvl 2s, length 20
IP 169.254.192.1 > 224.0.0.18: VRRPv2, Advertisement, vrid 1, prio 50, authtype none,
intvl 2s, length 20
```

Preverjanje zunanjega IP naslova virtualnega usmerjevalnika najemnika:

```
amir@openstack01:~/devstack$ neutron router-port-list private-router
+--------------------------------------+-------------------------------------------------
--+-------------------+-------------------------------------------------------------------
----------------------+
|  id                                                                                    |  name
|       mac_address                                                    |       fixed_ips
|
+--------------------------------------+-------------------------------------------------
--+-------------------+-------------------------------------------------------------------
----------------------+
|       02091c6f-b04a-4038-9acb-8d4725fcdf12       |       HA       port       tenant
a2c728f3f8b14c1888e1bba0bdfaa634  |  fa:16:3e:f4:a8:f1  |  {"subnet_id":  "fb9344e5-d42f-
494d-b462-2348dd69b854", "ip_address": "169.254.192.2"}  |
|                          3603fcde-9fc6-4353-9290-a4d416f6d4e9                          |
|    fa:16:3e:ed:e7:b6    |    {"subnet_id":    "acd13de1-34b1-414e-873d-c1203a91836d",
"ip_address": "192.168.123.11"} |
|                          777bbcfd-d9fa-48ca-993a-f7d93cf637a3                          |
|    fa:16:3e:88:0e:83    |    {"subnet_id":    "f67fc58c-9cba-4932-a884-f6991cbce4fc",
"ip_address": "192.168.1.1"}    |
|       8a9fe864-0b10-41b8-9fba-7ccb2ef8b313       |       HA       port       tenant
a2c728f3f8b14c1888e1bba0bdfaa634  |  fa:16:3e:73:44:6c  |  {"subnet_id":  "fb9344e5-d42f-
494d-b462-2348dd69b854", "ip_address": "169.254.192.1"}  |
|       fb61cca6-5119-4083-8298-366e9179cf4b       |       HA       port       tenant
a2c728f3f8b14c1888e1bba0bdfaa634  |  fa:16:3e:b0:f7:c2  |  {"subnet_id":  "fb9344e5-d42f-
494d-b462-2348dd69b854", "ip_address": "169.254.192.3"}  |
+--------------------------------------+-------------------------------------------------
--+-------------------+-------------------------------------------------------------------
----------------------+
```

Preverjanje omrežne povezljivosti do zunanjega IP naslova virtualnega usmerjevalnika najemnika z ukazom ping:

```
amir@openstack01:~/devstack$ ping 192.168.123.11
PING 192.168.123.11 (192.168.123.11) 56(84) bytes of data.
64 bytes from 192.168.123.11: icmp_seq=1 ttl=64 time=5.15 ms
64 bytes from 192.168.123.11: icmp_seq=2 ttl=64 time=0.434 ms
64 bytes from 192.168.123.11: icmp_seq=3 ttl=64 time=0.794 ms
64 bytes from 192.168.123.11: icmp_seq=4 ttl=64 time=0.588 ms

--- 192.168.123.11 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2999ms
rtt min/avg/max/mdev = 0.434/1.741/5.150/1.972 ms
amir@openstack01:~/devstack$
```

7.5 Konfiguracijske datoteke za postavitev oblačne infrastrukture z L3 povezavo v omrežje ponudnika in uporabo porazdeljenih virtualnih usmerjevalnikov

7.5.1 Devstack konfiguracija za vozlišče openstack01

```
[[local|localrc]]
HOST_IP=10.0.0.2
```

```
SERVICE_HOST=10.0.0.2
MYSQL_HOST=10.0.0.2
RABBIT_HOST=10.0.0.2
GLANCE_HOSTPORT=10.0.0.2:9292
PUBLIC_INTERFACE=eth1

IP_VERSION=4

ADMIN_PASSWORD=password
MYSQL_PASSWORD=password
RABBIT_PASSWORD=password
SERVICE_PASSWORD=password
SERVICE_TOKEN=password

#Ne uporabi Nova-Network
disable_service n-net
#Storitve ki jih zagotavlja Controle in Compute vozlisce
ENABLED_SERVICES+=,q-svc,q-dhcp,q-meta,q-agt,q-l3


#Neutron omrezne moznosti uporabljene za ustvarjanje Neutron podomrezij
Q_USE_SECGROUP=True
FLOATING_RANGE=192.168.123.0/24
FIXED_RANGE=192.168.1.0/24
Q_FLOATING_ALLOCATION_POOL=start=192.168.123.10,end=192.168.123.30
PUBLIC_NETWORK_GATEWAY=192.168.123.254
NETWORK_GATEWAY=192.168.1.1
Q_L3_ENABLED=True
PUBLIC_INTERFACE=eth1
Q_PLUGIN=ml2
Q_ML2_TENANT_NETWORK_TYPE=vxlan

# Open vSwitch moznosti omrezja ponudnika
Q_USE_PROVIDERNET_FOR_PUBLIC=True
OVS_PHYSICAL_BRIDGE=br-ex
PUBLIC_BRIDGE=br-ex
OVS_BRIDGE_MAPPINGS=public:br-ex

#Tempest
disable_service tempest

[[post-config|$NEUTRON_CONF]]
[DEFAULT]
router_distributed=True

[[post-config|/$Q_PLUGIN_CONF_FILE]]
[ml2]
type_drivers=flat,vlan,vxlan
tenant_network_types=vxlan
mechanism_drivers=openvswitch,l2population
```

```
[ml2_type_vxlan]
vni_ranges=1000:1999

[ovs]
local_ip=$HOST_IP

[agent]
tunnel_types=vxlan
l2_population=True
enable_distributed_routing=True

[[post-config|$Q_L3_CONF_FILE]]
[DEFAULT]
agent_mode=dvr
router_delete_namespaces=True

[[post-config|$Q_DHCP_CONF_FILE]]
[DEFAULT]
dhcp_delete_namespaces=True
```

7.5.2 Omrežna konfiguracija za vozlišče openstack01

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback


#vmesnik eth0 staticen IP naslov
auto eth0
 iface eth0 inet static
     address 10.0.0.2
     netmask 255.255.255.0
     gateway 10.0.0.1
     dns-nameservers 8.8.8.8


#vmesnik eth1 v promiscous nacinu
auto eth1
iface eth1 inet manual
    up ifconfig $IFACE 0.0.0.0 up
    up ip link set $IFACE promisc on
    down ip link set $IFACE promisc off
    down ifconfig $IFACE down


#vmesnik br-ex staticen IP naslov
auto br-ex
   iface br-ex inet static
   address 192.168.123.2
   netmask 255.255.255.0
   gateway 192.168.123.254
```

```
   dns-nameservers 8.8.8.8
```

7.5.3 Devstack konfiguracija za vozlišče openstack02

```
[[local|localrc]]
HOST_IP=10.0.0.3
SERVICE_HOST=10.0.0.2
MYSQL_HOST=10.0.0.2
RABBIT_HOST=10.0.0.2
GLANCE_HOSTPORT=10.0.0.2:9292
ADMIN_PASSWORD=password
MYSQL_PASSWORD=password
RABBIT_PASSWORD=password
SERVICE_PASSWORD=password
SERVICE_TOKEN=password
IP_VERSION=4

#Storitve ki jih zagotavlja Compute vozlisce
ENABLED_SERVICES=n-cpu,neutron,rabbit,q-agt,q-l3,q-meta

# Open vSwitch moznosti omrezja ponudnika
#PUBLIC_INTERFACE=eth1

NOVA_VNC_ENABLED=True
NOVNCPROXY_URL="http://10.0.0.2:6080/vnc_auto.html"
VNCSERVER_LISTEN=$HOST_IP
VNCSERVER_PROXYCLIENT_ADDRESS=$VNCSERVER_LISTEN

# Tempest
disable_service tempest

[[post-config|/$Q_PLUGIN_CONF_FILE]]
[ovs]
local_ip=$HOST_IP

[agent]
tunnel_types=vxlan
l2_population=True
enable_distributed_routing=True

[[post-config|$Q_L3_CONF_FILE]]
[DEFAULT]
agent_mode=dvr
router_delete_namespaces=True
```

7.5.4 Omrežna konfiguracija za vozlišče openstack02

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

#vmesnik eth0 staticen IP naslov
```

```
auto eth0
 iface eth0 inet static
     address 10.0.0.3
     netmask 255.255.255.0
     gateway 10.0.0.1
     dns-nameservers 8.8.8.8


#vmesnik eth1 v promiscous nacinu
auto eth1
iface eth1 inet manual
    up ifconfig $IFACE 0.0.0.0 up
    up ip link set $IFACE promisc on
    down ip link set $IFACE promisc off
    down ifconfig $IFACE down


#vmesnik br-ex staticen IP naslov
auto br-ex
   iface br-ex inet static
   address 192.168.123.3
   netmask 255.255.255.0
   gateway 192.168.123.254
   dns-nameservers 8.8.8.8
```

7.5.5 Devstack konfiguracija za vozlišče openstack03

```
[[local|localrc]]
HOST_IP=10.0.0.4
SERVICE_HOST=10.0.0.2
MYSQL_HOST=10.0.0.2
RABBIT_HOST=10.0.0.2
GLANCE_HOSTPORT=10.0.0.2:9292
ADMIN_PASSWORD=password
MYSQL_PASSWORD=password
RABBIT_PASSWORD=password
SERVICE_PASSWORD=password
SERVICE_TOKEN=password
IP_VERSION=4

#Storitve ki jih zagotavlja Neutron vozlisce
ENABLED_SERVICES=neutron,rabbit,q-agt,q-l3,q-meta

#Open vSwitch moznosti omrezja ponudnika
PUBLIC_INTERFACE=eth1

NOVA_VNC_ENABLED=True
NOVNCPROXY_URL="http://10.0.0.2:6080/vnc_auto.html"
VNCSERVER_LISTEN=$HOST_IP
VNCSERVER_PROXYCLIENT_ADDRESS=$VNCSERVER_LISTEN
```

```
# Tempest
disable_service tempest

[[post-config|/$Q_PLUGIN_CONF_FILE]]
[ovs]
local_ip=$HOST_IP

[agent]
tunnel_types=vxlan
l2_population=True
enable_distributed_routing=True

[[post-config|$Q_L3_CONF_FILE]]
[DEFAULT]
agent_mode=dvr_snat
router_delete_namespaces=True
```

7.5.6 Omrežna konfiguracija za vozlišče openstack03

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

#vmesnik eth0 staticen IP naslov
auto eth0
 iface eth0 inet static
     address 10.0.0.4
     netmask 255.255.255.0
     gateway 10.0.0.1
     dns-nameservers 8.8.8.8


#vmesnik eth1 v promiscous nacinu
auto eth1
iface eth1 inet manual
    up ifconfig $IFACE 0.0.0.0 up
    up ip link set $IFACE promisc on
    down ip link set $IFACE promisc off
    down ifconfig $IFACE down


#vmesnik br-ex staticen IP naslov
auto br-ex
   iface br-ex inet static
   address 192.168.123.4
   netmask 255.255.255.0
   gateway 192.168.123.254
   dns-nameservers 8.8.8.8
```

7.5.7 OpenStack ukazi za preverjanje kreiranja oblačne insfrastrukture

Preverjanje statusa omrežnih servisov na OpenStack vozliščih:

```
amir@openstack01:~/devstack$ source admin.openrc
amir@openstack01:~/devstack$ neutron agent-list
+--------------------------------------+--------------------+-------------+-------+----
------------+---------------------------+
| id                                   | agent_type         | host        | alive |
admin_state_up | binary                    |
+--------------------------------------+--------------------+-------------+-------+----
------------+---------------------------+
| 24512556-c8c3-428e-a5c2-7999adcfc89f | L3 agent           | openstack02 | :-)    |
True           | neutron-l3-agent          |
| 2fda8cec-817d-49a0-8efd-360e5fbc6d37 | DHCP agent         | openstack01 | :-)    |
True           | neutron-dhcp-agent        |
| 3b63931c-3f1a-46a6-8dd0-e5019f690a79 | L3 agent           | openstack01 | :-)    |
True           | neutron-l3-agent          |
| 3d9dc5f8-666e-45ee-af19-e358312f17d4 | Open vSwitch agent | openstack01 | :-)    |
True           | neutron-openvswitch-agent |
| 802f6eba-cf75-42d1-b831-1d649b62aaed | Open vSwitch agent | openstack02 | :-)    |
True           | neutron-openvswitch-agent |
| bc203571-c3b6-4755-a352-fc6601965120 | Open vSwitch agent | openstack03 | :-)    |
True           | neutron-openvswitch-agent |
| bd82ce2b-4b5e-4b97-9e30-eb47e41e57f9 | Metadata agent     | openstack03 | :-)    |
True           | neutron-metadata-agent    |
| be2b9289-b397-423f-8c2b-fc167a7da916 | L3 agent           | openstack03 | :-)    |
True           | neutron-l3-agent          |
| c47aa5ca-1f5d-4bbf-b4b0-d433c65401e3 | Metadata agent     | openstack01 | :-)    |
True           | neutron-metadata-agent    |
| cbe839a8-dbeb-4280-9285-6871aa5cdcfc | Metadata agent     | openstack02 | :-)    |
True           | neutron-metadata-agent    |
+--------------------------------------+--------------------+-------------+-------+----
------------+---------------------------+
```

7.5.8 OpenStack ukaz za preverjanje imenskih prostorov na omrežnem vozlišču

```
amir@openstack03:~/devstack$ ip netns
snat-875da198-1b78-466f-8c35-959c6e47a4aa
qrouter-875da198-1b78-466f-8c35-959c6e47a4aa
qdhcp-1d959fab-9f33-4071-a50e-7dc5c596c44a
```

7.5.9 OpenStack ukaz za preverjanje imenskih prostorov na računskem vozlišču

```
amir@openstack01:~/devstack$ ip netns
qrouter-875da198-1b78-466f-8c35-959c6e47a4aa
fip-3b9fg9f6-4ba8-37aa-g631-726gf5689a32
```

7.6 Konfiguracijske datoteke za postavitev oblačne arhitekture z uporabo avtomatizacije in orkestracije

7.6.1 Devstack konfiguracija za vozlišče openstack01

```
[[local|localrc]]
HOST_IP=10.0.0.2
SERVICE_HOST=10.0.0.2
MYSQL_HOST=10.0.0.2
RABBIT_HOST=10.0.0.2
```

```
GLANCE_HOSTPORT=10.0.0.2:9292
PUBLIC_INTERFACE=eth1

IP_VERSION=4

ADMIN_PASSWORD=password
MYSQL_PASSWORD=password
RABBIT_PASSWORD=password
SERVICE_PASSWORD=password
SERVICE_TOKEN=password

#Ne uporabi Nova-Network
disable_service n-net
#Omogoci Neutron omrezne storitve
ENABLED_SERVICES+=,q-svc,q-dhcp,q-meta,q-agt,q-l3


#Neutron moznosti
Q_USE_SECGROUP=True
FLOATING_RANGE=192.168.123.0/24
FIXED_RANGE=192.168.1.0/24
Q_FLOATING_ALLOCATION_POOL=start=192.168.123.10,end=192.168.123.30
PUBLIC_NETWORK_GATEWAY=192.168.123.254
NETWORK_GATEWAY=192.168.1.1
Q_L3_ENABLED=True
PUBLIC_INTERFACE=eth1
Q_PLUGIN=ml2
Q_ML2_TENANT_NETWORK_TYPE=vxlan

#Open vSwitch konfiguracija omrezja Q_USE_PROVIDERNET_FOR_PUBLIC=True
OVS_PHYSICAL_BRIDGE=br-ex
PUBLIC_BRIDGE=br-ex
OVS_BRIDGE_MAPPINGS=public:br-ex


#Omogoci Heat
enable_service heat
enable_service h-eng
enable_service h-api
enable_service h-api-cfn
enable_service h-api-cw


#Onemogoci Tempest
disable_service tempest
```

7.6.2 Omrežna konfiguracija za vozlišče openstack01

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback
```

```
#vmesnik eth0 staticen IP naslov
auto eth0
 iface eth0 inet static
     address 10.0.0.2
     netmask 255.255.255.0
     gateway 10.0.0.1
     dns-nameservers 8.8.8.8


#vmesnik eth1 v promiscous nacinu
auto eth1
iface eth1 inet manual
    up ifconfig $IFACE 0.0.0.0 up
    up ip link set $IFACE promisc on
    down ip link set $IFACE promisc off
    down ifconfig $IFACE down


#vmesnik br-ex staticen IP naslov
auto br-ex
   iface br-ex inet static
   address 192.168.123.2
   netmask 255.255.255.0
   gateway 192.168.123.254
   dns-nameservers 8.8.8.8
```

7.6.3 Heat konfiguracijska predloga

```
heat_template_version: 2013-05-23

description: heat.skripta.yaml predloga za postavitev ene virtualne
instance z novim omrezjem in podomrezjem najemnika

parameters:
   net:
     description: ime omrezja ki se uporabi za kreiranje virtualne
instance.
     type: string
     default: private

resources:

 my_key:
     type: OS::Nova::KeyPair
     properties:
       save_private_key: true
       name: my_key

 internal_net:
```

```
    type: OS::Neutron::Net

  internal_subnet:
    type: OS::Neutron::Subnet
    properties:
      network_id: { get_resource: internal_net }
      cidr: "192.168.20.0/24"
      dns_nameservers: [ "8.8.8.8", "8.8.4.4" ]
      ip_version: 4

  internal_router:
    type: OS::Neutron::Router
    properties:
      external_gateway_info: { network: public }

  internal_interface:
    type: OS::Neutron::RouterInterface
    properties:
      router_id: { get_resource: internal_router }
      subnet: { get_resource: internal_subnet }

  instance_port:
    type: OS::Neutron::Port
    properties:
      network: { get_resource: internal_net }
      fixed_ips:
        - subnet_id: { get_resource: internal_subnet }

  floating_ip:
    type: OS::Neutron::FloatingIP
    properties:
      floating_network: public

  association:
    type: OS::Neutron::FloatingIPAssociation
    properties:
      floatingip_id: { get_resource: floating_ip }
      port_id: { get_resource: instance_port }

  my_instance:
    type: OS::Nova::Server
    properties:
      flavor: m1.tiny
      image: cirros-0.3.4-x86_64-uec
      networks:
       - port: { get_resource: instance_port }
      key_name: {get_resource: my_key}

outputs:
   private_key:
     description: Private key
```

```
      value: { get_attr: [ my_key, private_key ] }
```

7.6.4 Postopek instalacije DevStack

```
sudo apt-get update
sudo apt-get install git
git clone -b stable/liberty  https://github.com/openstack-dev/devstack.git
```

Konfiguracijsko datoteko »local.conf« kopiramo v novo mapo z imenom »devstack«. DevStack skripto zaženemo iz ukazne vrstice z ukazom:

```
./stack.sh
```

V primeru uspešne izvedbe skripte se na koncu instalacije izpiše obvestilo:

```
Horizon is now available at http://10.0.0.2/
Keystone is serving at http://10.0.0.2:5000/v2.0/
Examples on using novaclient command line is in exercise.sh
The default users are: admin and demo
The password: password
This is your host ip: 10.0.0.2
stack.sh completed in 239 seconds.
```

7.6.5 OpenStack ukazi za preverjanje kreiranja oblačne infrastrukture

```
amir@openstack01:~/devstack$ source admin.openrc
amir@openstack01:~/devstack$ neutron agent-list
+--------------------------------------+--------------------+-------------+-------+----
------------+---------------------------+
| id                                   | agent_type         | host        | alive |
admin_state_up | binary                    |
+--------------------------------------+--------------------+-------------+-------+----
------------+---------------------------+
| 0ad4a665-beda-43ef-a6a8-f631f14497aa | Open vSwitch agent | openstack01 | :-)    |
True           | neutron-openvswitch-agent |
| 3b7fc724-049a-4689-bf4c-a7a77ce0c33a | L3 agent           | openstack01 | :-)    |
True           | neutron-l3-agent          |
| 637d3cc1-5ef2-408c-acfd-df709010585a | Metadata agent     | openstack01 | :-)    |
True           | neutron-metadata-agent    |
| c4ba8180-dd26-4737-8c15-4451dcbd6705 | DHCP agent         | openstack01 | :-)    |
True           | neutron-dhcp-agent        |
+--------------------------------------+--------------------+-------------+-------+----
------------+---------------------------+
amir@openstack01:~/devstack$ neutron net-list
+--------------------------------------+---------+--------------------------------------
------------------+
|  id                                  | name    | subnets
|
+--------------------------------------+---------+--------------------------------------
------------------+
| 0c260a4a-01f6-4536-b39e-9fe15a82f205 | public  | 0f5ab1a5-cf21-48fd-8099-92551606001a
192.168.123.0/24 |
| e854b1ac-4d84-4b5f-9f9b-53b35b956f86 | private | 960c1749-da98-4964-bbbf-c0d1e938ad42
192.168.1.0/24   |
+--------------------------------------+---------+--------------------------------------
------------------+
amir@openstack01:~/devstack$ neutron subnet-list
```

```
+--------------------------------------+---------------+-----------------+-----------
-----------------------------------------+
| id                                   | name             | cidr               |
allocation_pools                                |
+--------------------------------------+---------------+-----------------+-----------
-----------------------------------------+
| 0f5ab1a5-cf21-48fd-8099-92551606001a | public-subnet  | 192.168.123.0/24 | {"start":
"192.168.123.10", "end": "192.168.123.30"} |
| 960c1749-da98-4964-bbbf-c0d1e938ad42 | private-subnet | 192.168.1.0/24   | {"start":
"192.168.1.2", "end": "192.168.1.254"}     |
+--------------------------------------+---------------+-----------------+-----------
-----------------------------------------+
amir@openstack01:~/devstack$ neutron router-list
+--------------------------------------+---------+-------------------------------------
-------------------------------------------------------------------------------------
-------------------------------------------------------------------+-------------+-------+
| id                                   | name     | external_gateway_info
| distributed | ha    |
+--------------------------------------+---------+-------------------------------------
-------------------------------------------------------------------------------------
-------------------------------------------------------------------+-------------+-------+
| 50410172-7772-4652-a208-a78cc1da9ecb | router1 | {"network_id": "0c260a4a-01f6-4536-
b39e-9fe15a82f205",   "enable_snat":   true,   "external_fixed_ips":   [{"subnet_id":
"0f5ab1a5-cf21-48fd-8099-92551606001a",  "ip_address":  "192.168.123.10"}]}  |  False
| False |
+--------------------------------------+---------+-------------------------------------
-------------------------------------------------------------------------------------
-------------------------------------------------------------------+-------------+-------+
amir@openstack01:~/devstack$ neutron router-show router1
+-----------------------+-------------------------------------------------------------------
-------------------------------------------------------------------------------------
----------------------------------------+
|    Field                                                                    |    Value
|
+-----------------------+-------------------------------------------------------------------
-------------------------------------------------------------------------------------
----------------------------------------+
|       admin_state_up                                                        |       True
|
|     distributed                                                             |     False
|
|   external_gateway_info   |   {"network_id":   "0c260a4a-01f6-4536-b39e-9fe15a82f205",
"enable_snat":  true,  "external_fixed_ips":  [{"subnet_id":  "0f5ab1a5-cf21-48fd-8099-
92551606001a", "ip_address": "192.168.123.10"}]} |
|    ha                                                                       |    False
|
|  id                                          |  50410172-7772-4652-a208-a78cc1da9ecb
|
|    name                                                                     |    router1
|
|     routes                                                                          |
|
|    status                                                                   |    ACTIVE
|
|   tenant_id                                  |   597f781f33ea4a57938699a3f9ed4244
|
+-----------------------+-------------------------------------------------------------------
-------------------------------------------------------------------------------------
----------------------------------------+
amir@openstack01:~/devstack$ neutron router-port-list router1
+--------------------------------------+------+-------------------+--------------------
-------------------------------------------------------------------+
```

```
| id                                           | name | mac_address         | fixed_ips
|
+----------------------------------------+------+-------------------+--------------------
-------------------------------------------------------------------+
| 2c253504-f3e6-4d45-a211-50526ada6c7f |         | fa:16:3e:92:69:f3 | {"subnet_id":
"960c1749-da98-4964-bbbf-c0d1e938ad42", "ip_address": "192.168.1.1"}    |
| 92cf511d-486f-4417-bd26-14654242f4ce |         | fa:16:3e:30:14:4a | {"subnet_id":
"0f5ab1a5-cf21-48fd-8099-92551606001a", "ip_address": "192.168.123.10"} |
+----------------------------------------+------+-------------------+--------------------
-------------------------------------------------------------------+
```

Preverjanje dosegljivosti zunanjega IP naslova virtualnega usmerjevalnika »router1« z ukazom ping:

```
amir@openstack01:~/devstack$ ping -c 4 192.168.123.10
PING 192.168.123.10 (192.168.123.10) 56(84) bytes of data.
64 bytes from 192.168.123.10: icmp_seq=1 ttl=64 time=0.558 ms
64 bytes from 192.168.123.10: icmp_seq=2 ttl=64 time=0.178 ms
64 bytes from 192.168.123.10: icmp_seq=3 ttl=64 time=0.035 ms
64 bytes from 192.168.123.10: icmp_seq=4 ttl=64 time=0.112 ms

--- 192.168.123.10 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2999ms
rtt min/avg/max/mdev = 0.035/0.220/0.558/0.202 ms
```

7.6.6 OpenStack ukazi za zagon Heat predloge

```
amir@openstack01:~/devstack$ heat stack-create -f heat.skripta.yaml heat.skripta
+--------------------------------------+--------------+--------------------+-----------
----------+--------------+
| id                                   | stack_name   | stack_status       |
creation_time       | updated_time |
+--------------------------------------+--------------+--------------------+-----------
----------+--------------+
| 0a2d9ef4-546b-4b49-a392-ce3d351a3936 | heat.skripta | CREATE_IN_PROGRESS | 2016-05-
29T10:12:32 | None         |
+--------------------------------------+--------------+--------------------+-----------
----------+--------------+
amir@openstack01:~/devstack$ heat stack-list
+--------------------------------------+--------------+-----------------+--------------
-------+--------------+
| id                                   | stack_name   | stack_status    | creation_time
| updated_time |
+--------------------------------------+--------------+-----------------+--------------
-------+--------------+
| 0a2d9ef4-546b-4b49-a392-ce3d351a3936 | heat.skripta | CREATE_COMPLETE | 2016-05-
29T10:12:32 | None         |
+--------------------------------------+--------------+-----------------+--------------
-------+--------------+
```

7.6.7 OpenStack ukazi za preverjanje kreiranja oblačne infrastrukture s Heat predlogo

```
amir@openstack01:~/devstack$ source admin.openrc
amir@openstack01:~/devstack$ neutron net-list
+--------------------------------------+----------------------------------------+------
-----------------------------------------------+
| id                                   | name                                   |
subnets                                          |
```

```
+--------------------------------------+--------------------------------------+------
-----------------------------------------------+
| e854b1ac-4d84-4b5f-9f9b-53b35b956f86 | private                                      |
960c1749-da98-4964-bbbf-c0d1e938ad42 192.168.1.0/24   |
| 0c260a4a-01f6-4536-b39e-9fe15a82f205 | public                                       |
0f5ab1a5-cf21-48fd-8099-92551606001a 192.168.123.0/24 |
|  63e92130-9f76-475b-9424-67c1cca58424  |  heat.skripta-internal_net-vj7gfejx4t6s  |
a6142c8c-a97c-40aa-b4a6-fcd36cd1af97 192.168.20.0/24  |
+--------------------------------------+--------------------------------------+------
-----------------------------------------------+

amir@openstack01:~/devstack$ source admin.openrc
amir@openstack01:~/devstack$ neutron subnet-list
+--------------------------------------+----------------------------------------------+---
--------------+------------------------------------------------------+
| id                                   | name                                         |
cidr             | allocation_pools                                     |
+--------------------------------------+----------------------------------------------+---
--------------+------------------------------------------------------+
| 0f5ab1a5-cf21-48fd-8099-92551606001a | public-subnet                                |
192.168.123.0/24 | {"start": "192.168.123.10", "end": "192.168.123.30"} |
| 960c1749-da98-4964-bbbf-c0d1e938ad42 | private-subnet                               |
192.168.1.0/24   | {"start": "192.168.1.2", "end": "192.168.1.254"}     |
|  a6142c8c-a97c-40aa-b4a6-fcd36cd1af97  |  heat.skripta-internal_subnet-mw7pu5gqcrxm  |
192.168.20.0/24  | {"start": "192.168.20.2", "end": "192.168.20.254"}   |
+--------------------------------------+----------------------------------------------+---
--------------+------------------------------------------------------+
amir@openstack01:~/devstack$

amir@openstack01:~/devstack$ neutron router-list
+--------------------------------------+--------------------------------------+---
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
----------+-------------+-------+
| id                                   | name                                 |
external_gateway_info
| distributed | ha    |
+--------------------------------------+--------------------------------------+---
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
----------+-------------+-------+
| 50410172-7772-4652-a208-a78cc1da9ecb | router1                              |
{"network_id":     "0c260a4a-01f6-4536-b39e-9fe15a82f205",     "enable_snat":     true,
"external_fixed_ips":       [{"subnet_id":       "0f5ab1a5-cf21-48fd-8099-92551606001a",
"ip_address": "192.168.123.10"}]} | False       | False |
|  da096b1b-e76e-44b6-a1e5-1e25f53f905c  |  heat.skripta-internal_router-mkg3ajbuzkzg  |
{"network_id":     "0c260a4a-01f6-4536-b39e-9fe15a82f205",     "enable_snat":     true,
"external_fixed_ips":       [{"subnet_id":       "0f5ab1a5-cf21-48fd-8099-92551606001a",
"ip_address": "192.168.123.11"}]} | False       | False |
+--------------------------------------+--------------------------------------+---
-----------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
----------+-------------+-------+
amir@openstack01:~/devstack$    neutron    router-show    heat.skripta-internal_router-
mkg3ajbuzkzg
+-----------------------+-----------------------------------------------------------------
-----------------------------------------------------------------------------------------
--------------------------------------+
|    Field                                                                    |    Value
|
+-----------------------+-----------------------------------------------------------------
-----------------------------------------------------------------------------------------
--------------------------------------+
```

```
|       admin_state_up                                                                 |       True
|
|     distributed                                                                      |     False
|
|   external_gateway_info   |   {"network_id":   "0c260a4a-01f6-4536-b39e-9fe15a82f205",
"enable_snat":  true,  "external_fixed_ips":  [{"subnet_id":  "0f5ab1a5-cf21-48fd-8099-
92551606001a", "ip_address": "192.168.123.11"}]} |
|    ha                                                                                |    False
|
|  id                                          |  da096b1b-e76e-44b6-a1e5-1e25f53f905c
|
|  name                                      |  heat.skripta-internal_router-mkg3ajbuzkzg
|
|     routes                                                                           |
|
|    status                                                                            |    ACTIVE
|
|   tenant_id                                    |   597f781f33ea4a57938699a3f9ed4244
|
+----------------------+-------------------------------------------------------------
-------------------------------------------------------------------------------------
--------------------------------------+
```

Preverjanje dosegljivosti zunanjega IP naslova novega virtualnega usmerjevalnika z ukazom ping:

```
amir@openstack01:~/devstack$ ping -c 4 192.168.123.11
PING 192.168.123.11 (192.168.123.11) 56(84) bytes of data.
64 bytes from 192.168.123.11: icmp_seq=1 ttl=64 time=0.579 ms
64 bytes from 192.168.123.11: icmp_seq=2 ttl=64 time=0.205 ms
64 bytes from 192.168.123.11: icmp_seq=3 ttl=64 time=0.074 ms
64 bytes from 192.168.123.11: icmp_seq=4 ttl=64 time=0.068 ms

--- 192.168.123.11 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3001ms
rtt min/avg/max/mdev = 0.068/0.231/0.579/0.208 ms
```

Preverjanje uspešnosti kreiranja virtualne instance ter njene omrežne dosegljivosti po plavajočem IP naslovu:

```
amir@openstack01:~/devstack$ nova list
+--------------------------------------+--------------------------------------+--------
+------------+-------------+-------------------------------------------------------------
----------+
| ID                                   | Name                                 | Status
|        Task         State         |         Power         State         |        Networks
|
+--------------------------------------+--------------------------------------+--------
+------------+-------------+-------------------------------------------------------------
----------+
| 17a425e1-6a53-4141-8ff0-5b5d33827f52 | heat.skripta-my_instance-itpwfsfwqyy | ACTIVE
| -              | Running       | heat.skripta-internal_net-vj7gfejx4t6s=192.168.20.5,
192.168.123.14 |
+--------------------------------------+--------------------------------------+--------
+------------+-------------+-------------------------------------------------------------
----------+
amir@openstack01:~/devstack$ ping -c 4 192.168.123.14
PING 192.168.123.14 (192.168.123.14) 56(84) bytes of data.
64 bytes from 192.168.123.14: icmp_seq=1 ttl=63 time=1.86 ms
```

```
64 bytes from 192.168.123.14: icmp_seq=2 ttl=63 time=0.527 ms
64 bytes from 192.168.123.14: icmp_seq=3 ttl=63 time=0.575 ms
64 bytes from 192.168.123.14: icmp_seq=4 ttl=63 time=0.578 ms

--- 192.168.123.14 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3015ms
rtt min/avg/max/mdev = 0.527/0.887/1.868/0.566 ms
```

Povezovanje preko SSH seje na virtualno instanco in preverjanje omrežne povezljivosti do zunanjih omrežnih elementov z ukazom ping:

```
amir@openstack01:~/devstack$ ssh cirros@192.168.123.14
The authenticity of host '192.168.123.14 (192.168.123.14)' can't be established.
RSA key fingerprint is 20:2c:17:7e:ae:c2:05:14:ce:a2:83:79:4a:4c:1e:9f.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.123.14' (RSA) to the list of known hosts.
cirros@192.168.123.14's password:
$ ifconfig
eth0      Link encap:Ethernet  HWaddr FA:16:3E:FD:37:E4
          inet addr:192.168.20.5  Bcast:192.168.20.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:fefd:37e4/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1450  Metric:1
          RX packets:87 errors:0 dropped:0 overruns:0 frame:0
          TX packets:59 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:13702 (13.3 KiB)  TX bytes:6521 (6.3 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

$ ping -c 4 192.168.123.254
PING 192.168.123.254 (192.168.123.254): 56 data bytes
64 bytes from 192.168.123.254: seq=0 ttl=63 time=11.119 ms
64 bytes from 192.168.123.254: seq=1 ttl=63 time=2.063 ms
64 bytes from 192.168.123.254: seq=2 ttl=63 time=2.076 ms
64 bytes from 192.168.123.254: seq=3 ttl=63 time=1.981 ms

--- 192.168.123.254 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 1.981/4.309/11.119 ms
$ ping -c 4 google.com
PING google.com (216.58.209.206): 56 data bytes
64 bytes from 216.58.209.206: seq=0 ttl=57 time=19.383 ms
64 bytes from 216.58.209.206: seq=1 ttl=57 time=12.598 ms
64 bytes from 216.58.209.206: seq=2 ttl=57 time=11.992 ms
64 bytes from 216.58.209.206: seq=3 ttl=57 time=12.058 ms

--- google.com ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 11.992/14.007/19.383 ms
$ exit
Connection to 192.168.123.14 closed.
amir@openstack01:~/devstack$
```

8 Literatura in viri

[1] (2014) ONF. Software-Defined Networking (SDN) Definition. Dostopno na: https://www.opennetworking.org/sdn-resources/sdn-definition

[2] (2012) ONF. Software-Defined Networking: The New Norm for Networks. ONF White paper. Dostopno na: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

[3] (2014) Which is Better – SDN or NFV?. Dostopno na https://www.sdncentral.com/resources/nfv/which-is-better-sdn-or-nfv/

[4] (2014) ONF. SDN Architecture 1.0. Dostopno na: https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/TR_SDN_ARCH_1.0_06062014.pdf

[5] (2013) ONF. SDN Architecture Overview. Dostopno na: https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/SDN-architecture-overview-1.0.pdf

[6] (2014) What is Network Virtualization?. Dostopno na: https://www.sdncentral.com/resources/network-virtualization/whats-network-virtualization/

[7] (2013) On Network Virtualization and SDN. Dostopno na: http://blog.scottlowe.org/2013/04/30/on-network-virtualization-and-sdn/

[8] (2014) OpenFlow. Dostopno na: https://www.opennetworking.org/sdn-resources/openflow

[9] (2014) ONF. OpenFlow Switch Specification 1.3.4. Dostopno na: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.3.4.pdf

[10] (2014) The Evolution of SDN and OpenFlow: A Standards Perspective. Dostopno na: https://www.opennetworking.org/images/stories/downloads/sdn-resources/IEEE-papers/evolution-of-sdn-and-of.pdf

[11] (2011) Hypervisors, virtualization and the cloud. Dostopno na: http://www.ibm.com/developerworks/cloud/library/cl-hypervisorcompare/

[12] (2014) Understanding Full Virtualization, Paravirtualization, and Hardware Assist. Dostopno na: http://www.VMware.com/files/pdf/VMware_paravirtualization.pdf

[13] (2014) Virtualization Overview. Dostopno na: http://www.VMware.com/pdf/virtualization.pdf

[14] (2004) A. Singh. An introduction to Virtualization. Dostopno na: http://www.kernelthread.com/publications/virtualization/

[15] (2013) Containers & Docker: How secure are they? Dostopno na: http://blog.docker.com/2013/08/containers-docker-how-secure-are-they/

[16] (2014) Linux Containers: Why They're in Your Future and What Has to Happen First. Dostopno na: http://www.cisco.com/c/dam/en/us/solutions/collateral/data-center-virtualization/dc-partner-red-hat/linux-containers-white-paper-cisco-red-hat.pdf

[17] (2013) Containers – Not Virtual Machines – Are the Future Cloud. Dostopno na: http://www.linuxjournal.com/content/containers%E2%80%94not-virtual-machines%E2%80%94are-future-cloud?page=0,1

[18] (2104) What is Docker? Dostopno na: https://www.docker.com/whatisdocker/

[19] (2012) Brief History of Virtualization. Dostopno na: https://docs.oracle.com/cd/E26996_01/E18549/html/VMUSG1010.html

[20] (2012) Why We Need Network Abstraction. Dostopno na: http://www.networkcomputing.com/networking/why-we-need-network-abstraction/d/d-id/1233988?

[21] (2014) High Level Architecture for Network Virtualization. Dostopno na: http://gl.access-company.com/121003-2/

[22] (2011) Examining VXLAN. Dostopno na: http://blog.scottlowe.org/2011/12/02/examining-vxlan/

[23] (2014) M. Dillon, T. Winters. Network Function Virtualization in Home Networks. Dostopno na: https://www.opennetworking.org/images/stories/downloads/sdn-resources/IEEE-papers/network-func-virt-in-home-networks.pdf

[24] (2012) ETSI. Network Function Virtualisation – An Introduction, Benefits, Enablers, Challenges & Call for Action. Dostopno na: https://portal.etsi.org/NFV/NFV_White_Paper.pdf

[25] (2014) ETSI. Network Function Virtualisation (NFV) – Network Operator Perspectives on Industry Progress. Dostopno na: https://portal.etsi.org/Portals/0/TBpages/NFV/Docs/NFV_White_Paper3.pdf

[26] (2013) ETSI. Network Function Virtualisation (NFV) Use Cases. Dostopno na: http://www.etsi.org/deliver/etsi_gs/NFV/001_099/001/01.01.01_60/gs_NFV001v010101p.pdf

[27] (2014) ONF. OpenFlow-enabled SDN and Network Function Virtualization. Dostopno na: https://www.opennetworking.org/images/stories/downloads/sdn-resources/solution-briefs/sb-sdn-nvf-solution.pdf

[28] (2014) ETSI. Network Function Virtualization. Dostopno na: http://www.etsi.org/images/files/ETSITechnologyLeaflets/NetworkFunctionsVirtualization.pdf

[29] (2014) OpFlex: An Open Policy Protocol. Dostopno na: http://www.cisco.com/c/en/us/solutions/collateral/data-center-virtualization/application-centric-infrastructure/white-paper-c11-731302.html

[30] (2014) IETF. RFC 7348 Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized layer 2 Networks over Layer 3 Networks. Dostopno na: https://tools.ietf.org/html/rfc7348

[31] Sonali Yadav, »Comparative Study on Open Source Software for Cloud Computing Platform: Eucalyptus, OpenStack and OpenNebula«, *International Journal Of Engineering And Science*, zv. 3, št. 10, 2013.

[32] (2015) Official Documentation for Eucalyptus Cloud. Dostopno na: http://docs.hpcloud.com/eucalyptus/4.2.1/#shared/index.html

[33] (2015) General Purpose Reference Architercure: HPE Helion Eucalyptus. Dostopno na: http://h20195.www2.hp.com/V2/GetDocument.aspx?docname=4AA6-2547ENW

[34] (2015) Simple – and that's the point. Dostopno na: https://www.hpe.com/h30683/us/en/hpe-technology-now/fast-start-your-installation-with-hp-helion-eucalyptus-.html

[35] (2015) A new model to deliver public cloud. Dostopno na: http://community.hpe.com/t5/Grounded-in-the-Cloud/A-new-model-to-deliver-public-cloud/ba-p/6804409#.Vn_DzfkrLDf

[36] (2015) Official OpenNebula documentation. Dostopno na: http://opennebula.org/documentation/

[37] P. Sempolinski, D. Thain, »A Comparison and Critique of Eucalyptus, OpenNebula and Nimbus«, v zborniku *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*, Indianapolis, ZDA, nov. 2010, str. 417–426.

[38] (2015) Official CloudStack documentation: Dostopno na: http://docs.cloudstack.apache.org/en/master/

[39] R. Kumar, K. Jain, H. Maharwal, N. Jain, A. Dadhich, »Apache CloudStack: Open Source Infrastructure as a Service Cloud Computing Platform«, *International Journal Of Engineering And Science*, št. 2, zv. 1, str. 111-116, 2014.

[40] (2015) Official OpenStack documentation: Dostopno na: http://docs.openstack.org/

[41] (2015) Discover, Track and Compare OpenSource. Dostopno na: https://www.openhub.net/

[42] (2015) Cloud Computing Trends: 2015 State of the Cloud Survey. Dostopno na: http://www.rightscale.com/blog/cloud-industry-insights/cloud-computing-trends-2015-state-cloud-survey

[43] (2015) Stackalytics project. Dostopno na: http://stackalytics.com/

[44] (2014) OpenStack Beginner's Guide – For Ubuntu Trusty v 4.2. Dostopno na: https://fosskb.wordpress.com/2014/10/17/install-openstack-on-ubuntu-14-04-lts-trusty-tahr/

[45] P. Mell, T. Grance, »The NIST Definition of Cloud Computing«, NIST Special Publication 800-145, 2011.

[46] (2016) Spletna stran terminološkega slovarja informatike ISlovar. Dostopno na: http://islovar.org/iskanje_enostavno.asp

[47] (2016) IT Glossary. Dostopno na: http://www.gartner.com/it-glossary/

[48] (2011) Cloud computing service models. Dostopno na: http://www.ibm.com/developerworks/cloud/library/cl-cloudservicemodels/index.html

[49] (2013) Understanding the Cloud Computing Stack: SaaS, PaaS, IaaS. Dostopno na: https://www.rackspace.com/knowledge_center/whitepaper/understanding-the-cloud-computing-stack-saas-paas-iaas

[50] (2015) Classification of cloud service using two pics. Dostopno na: https://sites.google.com/site/yudongpage/blog/classificationofcloudserviceusingtwopics

[51] (2016) Microsoft Azure. Dostopno na: https://azure.microsoft.com/en-us/?b=16.01

[52] (2016) Types of Cloud Computing. Dostopno na: https://aws.amazon.com/types-of-cloud-computing/

[53] (2015) SDN Strategies: Global Service Provider Survey. Dostopno na: http://www.infonetics.com/

[54] (2015) NFV Strategies: Global Service Provider Survey. Dostopno na: http://www.infonetics.com/

[55] (2016) OpenStack Foundation Report: Accelerating NFV Delivery with OpenStack. Dostopno na: https://www.openstack.org/telecoms-and-nfv/

[56] P. Goransson, C. Black, *Software Defined Networks: A Comprehensive Approach*, USA: Morgan Kaufmann, 2014, pogl. 4, 6, 7.

[57] J. F. Kurose, K. W. Ross, *Computer Networking: A Top-Down Approach*, Boston: Pearson, 2010, pogl. 9.

[58] T. Vidmar, *Informacijsko – komunikacijski sistem*, Ljubljana: Pasadena, 2002.

[59] IETF. RFC 7637 NVGRE: Network Virtualization Using Generic Routing Encapsulation. Dostopno na: https://tools.ietf.org/html/rfc7637

[60] IETF. A Stateless Transport Tunneling Protocol for Network Virtualization (STT). Dostopno na: https://tools.ietf.org/html/draft-davie-stt-07

[61] IETF. Geneve: Generic Network Virtualization Encapsulation. Dostopno na: https://tools.ietf.org/html/draft-davie-stt-07

[62] K. Adams, O. Agesen , »A comparison of software and hardware techniques for x86 virtualization«, ACM SIGOPS Operating Systems Review, str. 2–13, 2006.

[63] L. van Doorn, »Hardware virtualization trends,« v zborniku *Proceedings of the 2nd international conference on Virtual execution environments*, New York, USA, 2006, str. 45–45.

[64] J. Robin, C. Irvine, »Analysis of the Intel pentium's ability to support a secure virtual machine monitor«, v zborniku *9th USENIX Security Symposium Paper*, Denver, USA, avg. 2000, str. 14–17.

[65] M. Rosenblum, T. Garfinkel, »Virtual machine monitors: current technology and future trends«, *Computer*, št. 38, zv. 5, str. 39–47, 2005.

[66] L. Yunfa, L. Wanqing, J. Congfeng, »A Survey of Virtual Machine System: Current Technology and Future Trends«, v *zborniku Electronic Commerce and Security (ISECS), 2010 Third International Symposium on*, Guangzhou, China, jul. 2010, str. 332–336.

[67] L. Wang, G. Laszevski, A. Younge, X. He, M. Kunze, J. Tao, C. Fu »Cloud computing: A Perspective study«, *New Generation Computing*, št. 28, zv. 2, str. 137–146, 2010.

[68] (2016) Open Virtual Switch. Dostopno na: http://openvswitch.org/

[69] S. Azodolmolky, »*Software Defined Networking with OpenFlow*«, Birmingham: Packt Publishing, 2013, pogl. 8.