

UNIVERZITA HRADEC KRÁLOVÉ
FAKULTA INFORMATIKY A MANAGEMENTU
KATEDRA INFORMATIKY A KVANTITATIVNÍCH METOD

Orchestrace a management virtuálních síťových
funkcí

DIPLOMOVÁ PRÁCE

Autor: Bc. Ondřej Smola

Studijní obor: Aplikovaná informatika

Vedoucí práce: Ing. Vladimír Soběslav, Ph.D.

Hradec Králové

duben, 2016

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a uvedl jsem všechny použité prameny a literaturu.

V Hradci Králové dne 16. dubna 2016

Ondřej Smola

Poděkování

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean placerat. Duis pulvinar. Maecenas lorem. Mauris tincidunt sem sed arcu. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt.

Anotace

Tato diplomová práce pojednává o aktuálním tématu, kterým je Virtualizace síťových funkcí (Network function virtualization).

Annotation

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean placerat. Duis pulvinar. Maecenas lorem. Mauris tincidunt sem sed arcu. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Phasellus rhoncus. Praesent vitae arcu tempor neque lacinia pretium. Mauris suscipit, ligula sit amet pharetra semper, nibh ante cursus purus, vel sagittis velit mauris vel metus. Etiam posuere lacus quis dolor. Curabitur bibendum justo non orci. Praesent in mauris eu tortor porttitor accumsan. Nullam lectus justo, vulputate eget mollis sed, tempor sed magna. Donec quis nibh at felis congue commodo. Integer tempor. Maecenas libero.

Obsah

1	Úvod	1
2	Základní problematika virtualizace síťových funkcí	3
2.1	Potřeba virtualizaci síťových funkcí (NFV)	3
2.2	Základní princip virtualizace	5
2.3	Cloud Computing	6
2.3.1	Distribuční modely	7
2.3.2	Modely nasazení	8
2.4	Souvislost NFV a SDN	9
2.4.1	Service Chaining	10
2.5	Architektura NFV a VNF	11
2.5.1	Infrastruktura NFV	13
2.5.2	Virtuální síťová funkce	14
2.5.3	Management a orchestrace NFV	15
2.6	Možné technologie pro řešení	17
2.6.1	Hypervisory	17
2.6.2	VNF	18
2.6.3	Cloud platforma	18
2.6.3.1	OpenStack	18
2.6.3.2	VMware vCloud	19
2.6.4	SDN	19
3	Popis navržených řešení	20
3.1	Požadavky na VNF řešení	20
3.2	Architektura navrženého řešení	20
3.3	Load balancer as a Service	22
3.3.1	Neutron LbaaS	22
3.3.2	LbaaS heat template	24
3.3.3	Ukázka použití LbaaS heat templatu	25
3.4	Firewall as a Service	27
3.4.1	Scénář NAT	28
3.4.2	Scénář HA firewall	29
3.4.3	FwaaS template	29
3.5	Ukázka použití FwaaS heat templatu	29
3.5.1	Fortigate VM	30
3.5.2	PFSense	30

4	Shrnutí poznatků	33
5	Závěr	34
	Literatura	35
	Přílohy	I

1 Úvod

V dnešní době dochází v datových centrech k nasazování nových moderních technologií. Jednou z nich je například virtualizace a to především v oblasti výpočetního výkonu a úložišť. Je již běžnou praxí, že v datových centrech vše běží na jedné fyzické infrastruktuře, která je abstrahovaná na jeden souvislý blok výpočetního výkonu a jeden souvislý blok úložiště. Dalším takovýmto funkcionálním blokem v datových centrech jsou počítačové sítě. Avšak v počítačových sítích byl, oproti dvěma zmíněným oblastem, pomalejší vývoj inovací a není zde tolik vyžívána virtualizace. Pro zvýšení efektivity je proto nutné, aby se počítačové sítě staly programovatelnými a mohli být spravovány z jednoho centrálního místa.

Dnes je však zatím síťové funkčnosti soustředěno ve fyzických proprietárních zařízeních jako jsou routery, firewally či load balancery. To znamená, že provozovatelé počítačových sítí se při spouštění nových síťových služeb musí na tyto zařízení spoléhat. Což může vést k zdlouhavému nasazování, zvýšené spotřebě energií a investici do školení pracovníků pro dané proprietární zařízení. Zároveň zde není možnost, aby síť mohla být dynamicky ovládána dle aktuálních požadavků uživatelů sítě. Například vývojář nemůže hned nasadit aplikaci do produkce. Musí nejprve čekat na síťový tým než patřičně nakonfiguruje síťové prvky pro správné a bezpečné fungování celé infrastruktury.

Virtualizace síťových funkcí se zaměřuje na transformaci způsobu, jakým síťový architekti přistupují k oblasti počítačových sítí a to pomocí stávajících a neustále se vyvíjejících virtualizačních technologií. Snaha je tedy přesunout mnoho typů síťového příslušenství z fyzických síťových prvků do standardních průmyslově používaných serverů a úložišť, které mohou být umístěny v datových centrech či přímo u koncových zákazníků. Tímto lze dosáhnout virtuálních síťových funkcí, které mají naprosto stejnou funkcionalitu jako síťové funkce umístěné v síťových prvcích.

Cílem této diplomové práce je analyzovat aktuální stav v oblasti virtualizace síťových funkcí. Dále je cílem navrhnout jednoduchý framework pro virtualizaci síťových funkcí spolu s ukázkou několik příkladů síťových funkcí. Celý framework by měl sloužit k možnosti rychlého a jednoduchého nasazení vybraných síťových funkcí. Současně by k jeho vytvoření měli být použity aktuálně dostupné technologie. Toto řešení musí být univerzální, nezávislé na vendorech a flexibilní.

Celá struktura této práce je rozdělena na několik částí. V druhé kapitole jsou vysvětleny hlavní pojmy a problematika oblasti virtualizace síťových funkcí. Třetí je popisu návrhu řešení frameworku pro virtualizaci síťových funkcí a popisu použitých použitých technologií pro tento návrh. Ve čtvrté kapitole je věnována testování a ukázce, jak navržený framework funguje. Na konci této práce dojde k závěrečnému shrnutí.

Závěrečná práce byla zpracována ve spolupráci s firmou tcp cloud a.s., která poskytuje implementace jednoho z nejlepších cloudových řešení na světě. Firma umožnila využít jejich stávající infrastrukturu v nejmodernějším datovém centru v České republice, které je v budově Technologického centra Písek s.r.o.

2 Základní problematika virtualizace síťových funkcí

Jak již vyplývá z názvu, tak tato kapitola se zabývá základní analýzou a popisem problematiky spojené s oblastí virtuální síťových funkcí. Budou zde vysvětleny hlavní důvody pro virtualizaci síťových funkcí, základní principy virtualizace, cloud computing a referenční architektura frameworku pro virtualizaci síťových funkcí. Poté budou zmíněny příklady některých dostupných technologií a řešení, které k virtualizaci síťových funkcí lze použít.

Pro lepší pochopení a přehlednost celé této práce zde budou rozlišeny následující pojmy, se kterými se lze také setkat v odborné literatuře a které budou dále v této práci používány.

- Síťová funkce (Network function - NF) - Toto je komponenta síťové infrastruktury, která má dobře definované funkční chování, jako například směrování, NAT, Load balancing, Intrusion detection, atd.
- Virtuální síťové funkce (Virtual network function - VNF) - Je stejná jako NF, ale zde je funkčnost implementována pomocí softwaru a je nezávislá na hardwaru, na kterém běží.
- Virtualizace síťových funkcí (Network Functions Virtualization - NFV) - Zde se jedná o označení celého konceptu či frameworku.

2.1 Potřeba virtualizaci síťových funkcí (NFV)

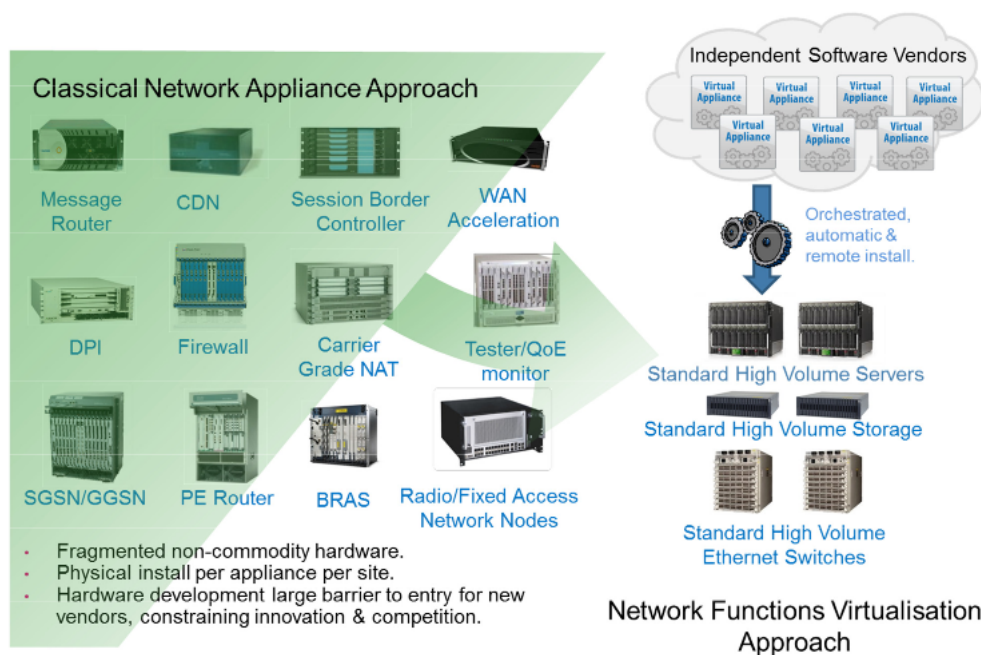
Produkční vývoj v telekomunikačním průmyslu se tradičně řídil přísnými standardy kvůli stabilitě a kvalitě komunikace. Přestože tento model v minulosti fungoval, tak vedl nevyhnutelně k dlouhým produkčním cyklům, pomalému tempu vývoje a spoléhání se na proprietární či specializovaný hardware. S příchodem výrazné konkurence v komunikačních službách, od rychle postupujících organizací operujících ve velkém měřítku na veřejném internetu, podnítil poskytovatele služeb pro hledat nových způsobů, jak změnit dosavadní způsob produkčního vývoje.

Pro vyřešení toho problému bylo navrženo v publikacích [1] a [2] skupinou několika telekomunikačních provozovatelů ETSI řešení ve formě virtualizace síťových funkcí (network functions virtualization). Později vzniklo více projektů zabývajících se touto oblastí jako například OPVNF [3] Hlavní cíle tohoto řešení jsou zlepšit následující aspekty provozu telekomunikačních sítí:

- Smíření investičních nákladů – snížení potřeby nákupu jednoúčelových hardwarových zařízení, možnost platby pouze za využití kapacity a snížení rizik přílišného předimenzování kapacit
- Snížení provozních nákladů – snížení prostoru, napájení a požadavky na chlazení, zjednodušení správy a řízení síťových služeb
- Urychlení Time-to-market – zkrácení doby pro nasazení nových síťových služeb, chopení se nových příležitosti na trhu, vyhovění potřebám zákazníka
- Doručit agilitu a flexibilitu – možnost rychle škálovat (rozšiřovat nebo zmenšovat služby) dle měnících se požadavků od zákazníka. Podpora služeb, které mají být dodány pomocí softwaru na libovolném standardním serverovém hardwaru

Jak je uvedeno v [4] a [5], tak celá myšlenka je založena na tom, že dojde k separování softwarové funkcionality v síťových prvcích od proprietárního hardwaru, na kterém běží. To umožní se síťovými funkcemi zacházet jako s klasickými softwarovými aplikacemi, které mohou běžet na standardním komerčně dostupných serverech jenž organizace v současnosti používají. Tím bude zároveň umožněno flexibilní nasazování těchto síťových funkcí a jejich dynamický provisioning. Díky tomu, že jsou síťové funkce odděleny od hardwaru, tak je také možné jejich vhodnější umístění v topologii. To znamená dle požadavků na umístění mohou být nasazeny v datových centrech, síťových uzlech či přímo v uživatelské koncové bodě. Hlavní koncept virtualizace síťových funkcí znázorňuje obrázek č. 2.1.

Za zmínění stojí poznámka v [4], kde je řečeno, že obecný koncept oddělení síťové funkce od hardwaru ještě nutně neznamená potřebu využití virtualizace. Protože budou síťové funkce dostupné jako software, tak mohou být nainstalovány a provozovány přímo na fyzickém stroji. Ovšem rozdíl je, že tento stroj již nebude speciální hardware, ale klasický server. Tento scénář může být do jisté míry použit při nasazování síťových funkcí v malém měřítku např. v uživatelských koncových bodech. Avšak pro plné využití všech výše zmíněných výhod, které jsou třeba ve velkých datových centrech, je třeba s použitím virtualizace počítat. To vše umocňuje fakt, že většina datových center v současnosti již využívá cloud computing.



Obrázek 2.1: Koncept virtualizace síťových funkcí (NFV)

2.2 Základní princip virtualizace

Základním problémem v tradičním modelu IT infrastruktury je, že servery podporují pouze jeden operační systém v čase. Na tomto systému obvykle běží pouze jedna aplikace. Přestože by na tomto systému popř. serveru mohlo běžet více aplikací, tak je lepší držet aplikace odděleně na různých systémech z důvodu minimalizace potenciálních bodů selhání. Pokud například nastane s aplikací problém, tak častým řešením je restartování systému. Pokud by na systému bylo více aplikací, znamenalo by to jejich vyřazení z provozu po dobu restartu, který může trvat velice dlouho. [6]

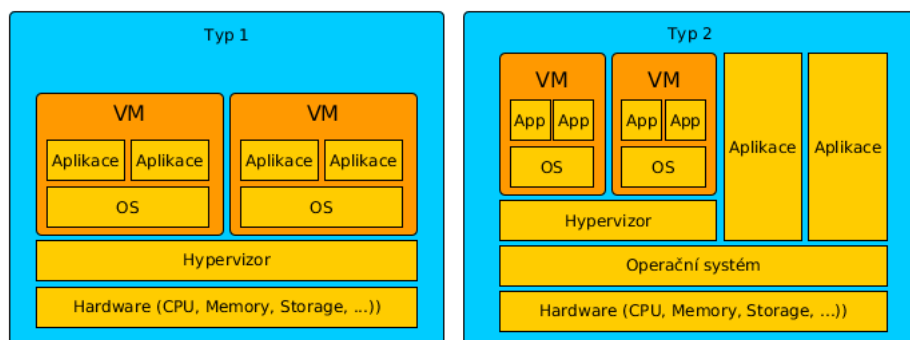
Z výše zmíněných důvodů došlo k velkému rozvoji a nasazování virtualizace. Virtualizací obecně označujeme techniky, které umožňují k dostupným hardwarovým zdrojům přistupovat jiným způsobem, než jakým fyzicky existují. Je tomu díky softwaru, který tento hardware abstrahuje a vytvoří tím virtuální prostředí. Virtualizované prostředí se dá snadněji přizpůsobit potřebám uživatelů, případně skrýt pro uživatele nepodstatné detaily (jako např. rozmístění hardwarových prostředků). Tím je tedy umožněno na jednom fyzickém serveru provozovat více od sebe oddělených virtuálních strojů, které mají každý svůj vlastní operační systém s aplikacemi. Software pro virtualizaci se nazývá hypervisor.[6]

Jak zmiňuje [7], tak existují tyto dva základní typy hypervisorů:

- Typ 1 (Nativní) - Tento hypervisor běží přímo na fyzickém hardwaru. Tím umožň-

ňuje provozovat více operačních systémů na jednom fyzickém stroji. Příkladem takového hypervisoru je VMware ESXi a XEN.

- Typ 2 (Hostovaný) - Na rozdíl od předchozího případu tento typ hypervisoru běží v prostředí operačního systému. Příkladem je například KVM či Microsoft Hyper-V



Obrázek 2.2: Schéma hypervisorů

Obrázek 2.2 zobrazuje schématický popis obou hypervisorů a jejich rozdíl. Problematika virtualizace je velice rozsáhlá a více informací o ní poskytují zdroje [6] a [7].

2.3 Cloud Computing

Jak již bylo zmíněno, tak cloud computing, nebo někdy také označován pouze jako cloud, je oblast, ve které je velký potenciál pro využití virtuálních síťových funkcí. Z tohoto důvodu bude v této části jeho problematika přiblížena.

Cloud je technologie, kterou v poslední době začala provozovat většina větších organizací, jak ukazuje [8]. Cloud Computing má mnoho definic. Dle definice uvedené v [9] ho lze charakterizovat jako poskytování služeb, programů a výpočetních zdrojů servery dostupnými z internetu s tím, že uživatelé k nim mohou přistupovat vzdáleně.

Z technického hlediska tvoří cloud veškeré služby poskytované přes Internet a zároveň i infrastruktura, která tyto zdroje poskytuje. Tuto infrastrukturu tvoří velké množství fyzických serverů, které jsou vzájemně propojeny. Na těchto serverech běží hypervisor, který vytvoří virtuální infrastrukturu. Pro vytváření cloudových služeb zde musí ještě existovat cloudová platforma, která dokáže celou tuto virtuální infrastrukturu spravovat. [9]

Dle [9] existuje 5 základních atributů, kterými se cloud computing vyznačuje. Jsou to tyto:

- Služby dostupné na požádání
- Všudypřítomný síťový přístup
- Sdílení zdrojů
- Vysokou elasticitu
- Měření využitých zdrojů

2.3.1 Distribuční modely

Dle [10] lze cloudové služby lze rozdělit do 3 základních kategorií. V [11] jsou k těmto kategoriím přiřazeny příklady užití z oblasti virtualizace síťových funkcí.

- Infrastructure as a Service (IaaS) - Nejzákladnější model poskytování cloudových služeb. IaaS cloudové platformy nabízejí například výpočetní výkon, virtuální disky, blokové a souborové úložiště či virtuální síť. Poskytovatelé IaaS cloudových platform poskytnou tyto zdroje na vyžádání ze svých datových center. Toto je možné díky skupině hypervisorů v rámci cloudu, které mohou provozovat velké množství virtuálních strojů a mají schopnost škálovat poskytované služby v závislosti na měnících se požadavcích přicházejících od zákazníků. Tento model může tedy sloužit i pro poskytnutí všech potřebných zdrojů celé infrastruktury pro virtualizaci síťových prvků, neboli Network Function Virtualization Infrastructure as a Service. Zde má uživatel pod nejvíce možností, jak navrhnout a spravovat virtuální síťové funkce, protože v zásadě dokáže nasazovat i vlastně navržené síťové funkce a nejen ty, které mu poskytuje provozovatel cloudu.
- Platform as a Service (PaaS) - V modelu Platforma jako služba (PaaS) hostují poskytovatelé cloudových služeb určitou počítačovou platformu, kterou následně poskytují koncovým uživatelům přes Internet. Tato platforma většinou bývá prostředím nějakého operačního systému, prostředím pro běh určitého programovacího jazyka, databáze a webový server. Vývojáři aplikací tím pádem mohou provozovat a případně vyvíjet svá softwarová řešení bez výrazných nákladů a složitého nákupu a konfiguraci potřebného hardwaru a softwaru. Některé PaaS platformy nastavuje výpočetní a úložné prostředky aplikace automaticky tak, aby odpovídala aktuálním požadavkům aplikace bez nutnosti zásahu zákazníka. NVF v tomto modelu může nabízet síťové služby, které se mohou skládat z více virtuálních síťových funkcí, neboli Virtual Network Platform as a Service. Zde je poskytnuta uživateli velká kontrola nad konfigurací a ovládáním celé platformy.

- Software as a Service (SaaS) - V modelu SaaS provozují poskytovatelé cloudových služeb aplikační software v cloudu a uživatelé k tomuto softwaru přistupují pomocí klientského software (např. webové prohlížeče). Uživatelé cloudu tedy nespravují infrastrukturu ani platformu, kde aplikace běží. Není proto třeba zde nic instalovat a spouštět aplikace na vlastních počítačích uživatele, což velmi zjednodušuje údržbu. Cloudové aplikace se liší od ostatních aplikací v možnostech škálování, kterého může být dosaženo díky distribuci úkolů na více virtuálních strojů, a tím reagovat na měnící se poptávku. Tento proces je pro uživatele služby transparentní, uživatel vidí pouze jeden přístupový bod pro danou aplikaci. Do této kategorie služeb může patřit poskytování virtuálních síťových funkcí, která je pouze ve formě softwarové aplikace, neboli VNFaaS. Takovéto aplikace poskytují síťovou funkci pro síťové správce a uživatele nejčastěji v privátním cloudu.

2.3.2 Modely nasazení

Existuje několik základních modelů nasazení cloud computingu resp. cloudových platforem, které uvádí [10]. V [11] lze k nim opět najít určité příklady z oblasti virtualizace síťových funkcí.

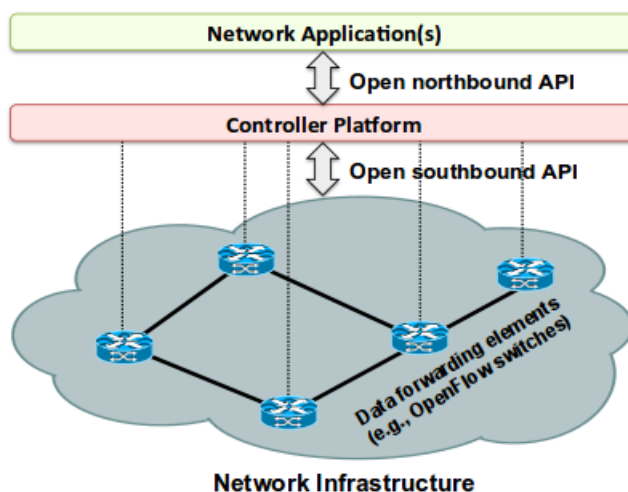
- Privátní cloud - Privátní cloud je infrastruktura provozována výhradně v rámci jedné organizace. Může být spravován interně nebo prostřednictvím třetí strany a hostování může být opět interní nebo externí. Aby mohl podnik využít privátní cloud, musí nejprve navrhnout a uzpůsobit k tomuto účelu svoji stávající infrastrukturu, která musí být virtualizována. Vlastní přechod vyvolává řadu bezpečnostních otázek, které je třeba řešit, aby se zabránilo vážným zranitelnostem celého řešení. Privátní cloud je přesně ten typ modelu, kde lze najít využití pro virtualizaci síťových funkcí.
- Veřejný cloud - Veřejné cloudové jsou cloudové služby, jako jsou aplikace, výpočetní výkon, úložiště a další, které jsou k dispozici široké veřejnosti. Služby jsou poskytovány zdarma nebo podle modelu platby za množství použitých služeb. Je zvykem, že veřejní poskytovatelé cloudových služeb, jako je Amazon AWS, Microsoft nebo Google, vlastní a provozují hardwarovou infrastrukturu a nabízejí k ní přístup pouze přes Internet. V tomto modelu není očekáváno využívání NFV.
- Hybridní cloud - Hybridní cloud je spojení dvou nebo více cloudů (soukromých, komunitních nebo veřejných), které zůstávají samostatné, ale jsou těsně propojeny. Toto složení rozšiřuje možnosti nasazení cloudových služeb a tím umožňuje

IT organizacím využít veřejné cloudové prostředky k uspokojení dočasných potřeb. Tato schopnost umožňuje hybridním cloudům škálovat přes více nezávislých cloudů. V tomto modelu může být využito NFV především na straně soukromého cloudu.

- Komunitní cloud - V rámci komunitního cloudu sdílí infrastrukturu cloudu několik organizací, které mají společné zájmy (bezpečnost, dodržování předpisů, působnost, atd.). Komunitní cloud může být spravován interně nebo prostřednictvím třetí strany. Náklady jsou rozloženy mezi méně uživatelů než na veřejném cloudu. V tomto modelu může být využito NFV, pokud se provozovatelé takového cloudu domluví.

2.4 Souvislost NFV a SDN

Softwarově definované sítě (SDN) je další z nových technologií, která se snaží vylepšit a automatizovat správu stávajících počítačových sítí. Dle [12] jde o koncept, ve které je oddělena řídicí logika (control plane) z jednotlivých routerů a switchů, které přeposílají traffic (data plane). Tím, že dojde k oddělení datové a řídicí vrstvy, se routery a switche stanou pouze přeposílající data a veškerá řídicí logika může být implementována v jednom logicky centrálním místě (SDN Controller). Z tohoto centrálního místa lze do jednotlivých routerů a switchů předávat instrukce pomocí aplikačních programovacích rozhraní (API). Samotný SDN Controller také obsahuje API, které mohou využívat aplikace a tím řídit, resp. programovat celou počítačovou síť.



Obrázek 2.3: Schéma SDN, převzato z [12]

Obrázek č. 2.3 ukazuje jednoduché schéma softwarově definovaných sítí. Celou ar-

chitekturu lze tedy rozdělit do 3 logických vrstev, které spolu komunikují pomocí API.

- Aplikační vrstva - Na této úrovni se nachází samotné síťové aplikace jako jsou například DHCP, ACL, NAT, DNS a další. Jejich vytváření by mělo být poskytováno prostřednictvím nižší vrstvy, nazývané northbound API.
- Northbound APIs - Toto API využívají aplikace pro komunikaci s SDN controllerem.
- Control vrstva - V této vrstvě je centralizována veškerá logika, které dříve byla v síťových prvcích.
- Southbound APIs - Jedná se o skupinu API protokolů, které pracují mezi vrstvou infrastruktury a control vrstvou. Jejím hlavním úkolem je komunikace, která povoluje SDN controlleru instalovat na samotné síťové prvky rozhodnutí definované v aplikační vrstvě.
- Vrstva infrastruktura - Nejnižší vrstvou je samotný hardware pro předávání datagramů na fyzické úrovni. Pro funkčnost celé architektury je nutné, aby zde byla nasazena zařízení, která umí přijímat pokyny od control plane skrze southbound API.

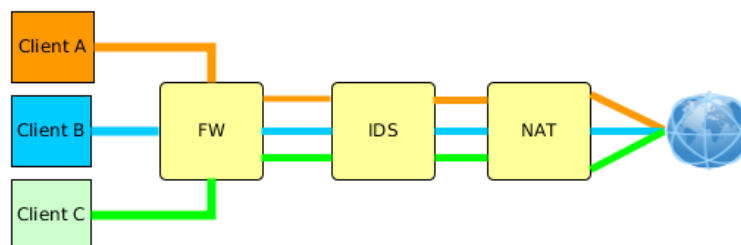
Přestože Softwarově definované sítě a virtualizace síťových funkcí jsou dvě různé technologie a koncepty, tak se navzájem se doplňují. Fakt, že SDN umožňuje programicky ovládat počítačovou síť, lze využít pro poskytnutí programovatelné konektivity mezi jednotlivými virtuálními síťovými funkcemi. Naopak SDN může využít NFV tím, že implementuje potřebné síťové funkce jako software. Může tak virtualizovat SDN Controller, který tak může běžet na co nejvhodnějším místě v datovém centru. Je vidět, že tyto dvě technologie se dobře doplňují, proto jsou často součástí jednoho řešení. [13]

2.4.1 Service Chaining

Jednou z výhod NFV je možnost využít Service Chaining. Service chaining je ve skutečnosti součástí SDN. Jde o princip jakým lze dynamicky pospojovat jednotlivé VNF a ovládat tak toky v síti. [13]

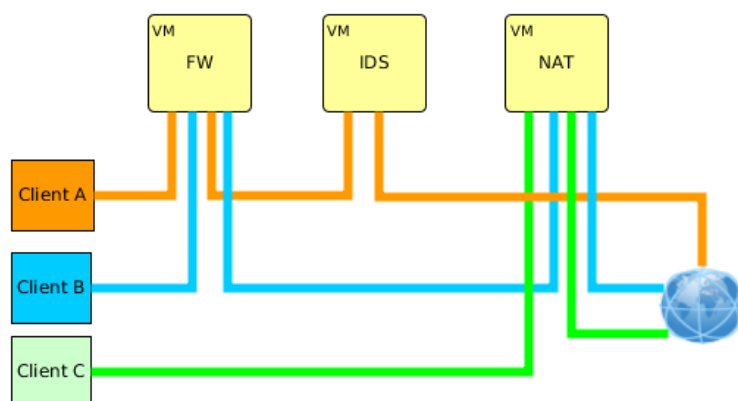
Service chaining není ve skutečnosti nic nového. V klasických počítačových sítích je používán také, ale pomocí fyzických síťových prvků. Jedná se zjednodušeně o způsob zapojení mezi jednotlivými síťovými prvky (či VNF) a způsob, jakým na sebe navazují. Příklad takového zapojení je vidět na obrázku č. 2.4. Zde se provozovatel sítě rozhodl, že odchozí data z klientských stanic musí jít přes firewall, IDS a nakonec přes NAT do Internetu. Příchozí data mají logicky obrácené pořadí. Toto zapojení funguje dobře pro

síť, kde není třeba rozlišovat cestu jakou proudí data jednotlivých uživatelských stanic. Ale není to optimální řešení pro síť s více uživateli, kde každý požaduje jinou síťovou funkci. Potřeba jednotlivých síťových služeb se samozřejmě může v čase měnit. Příklad takové sítě lze nalézt ve většině datových center.



Obrázek 2.4: Ukázka klasického service chainu pomocí fyzických síťových prvků

Zde tedy přichází na řadu VNF spolu s SDN. Protože jednotlivé VNF existují jako virtuální stroje, tak mohou být dynamicky nasazovány dle aktuálních požadavků jednotlivých klientů a pomocí SDN mohou být tyto VM dynamicky pospojovány. Obrázek č. 2.5 ukazuje schéma zapojení, kde každý klient může mít jinou požadovanou cestu do internetu. Je možná i varianta, kde každý klient má své vlastní VNF s jinou konfigurací.

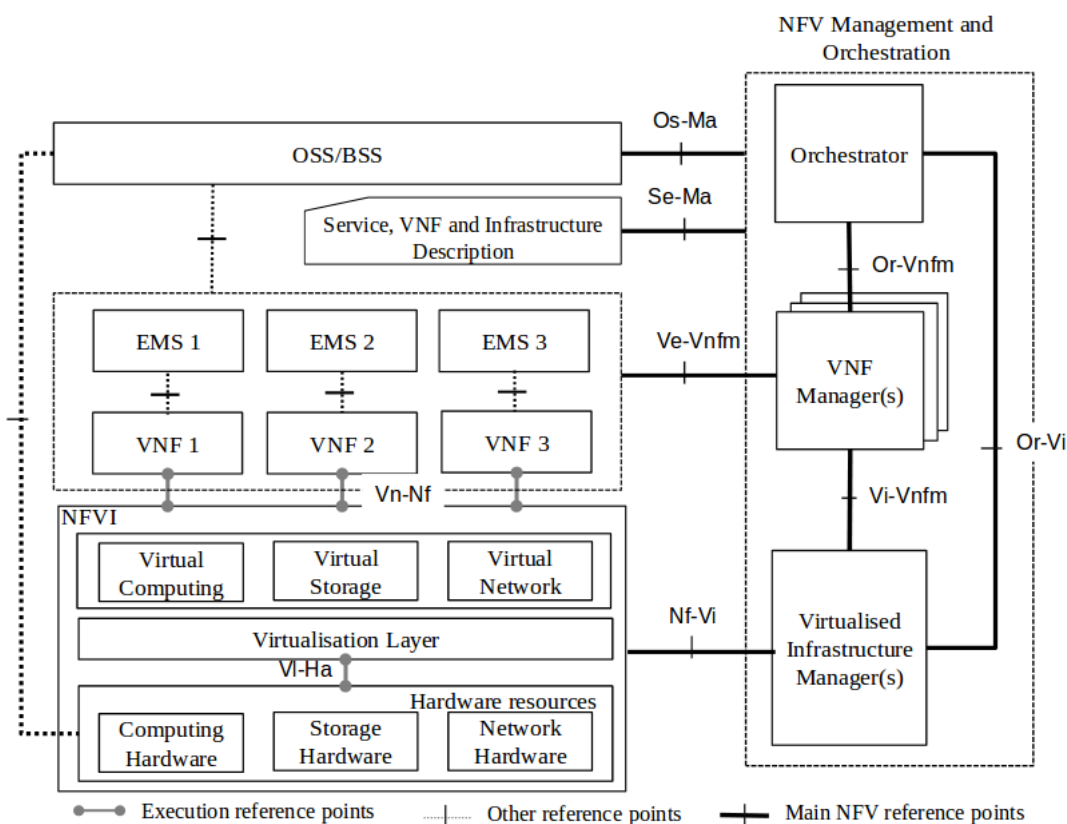


Obrázek 2.5: Ukázka VNF service chainu

2.5 Architektura NFV a VNF

V předchozí sekci byla popsána myšlenka a motivace související s virtualizací síťových funkcí. Protože cílem této práce je navržení jednoduchého NFV frameworku, tak je nejprve nutné se seznámit s jeho obecnou architekturou. V této práci se bude vycházet z referenční architektury NVF [14], která byla navržena organizací ETSI. Jedná se

pouze o funkční návrh bez náznaků konkrétní implementace. Od této skupiny existují i podrobnější návrhy jednotlivých částí celého NFV frameworku, které v této práci budou také popsány v příslušných kapitolách.



Obrázek 2.6: NFV architektura, převzato z [14]

Jak je vidět na obrázku č. 2.6, tak celá architektura se dá rozdělit na tyto 3 hlavní části:

- **Infrastruktura virtualizace síťových funkcí (NFVI)** - Jsou všechny softwarové a hardwarové zdroje potřebné k vytvoření prostředí, ve které mohou být jednotlivé VNF být nasazeny. Tato infrastruktura může být velice rozsáhlá, proto je její součástí i síť poskytující konektivitu mezi vzdálenými lokacemi infrastruktury.[15]
- **Virtualizované síťové funkce (VNFs)** - Jsou softwarové implementace síťových funkcí, jako je např. NAT a routing, které mohou být nasazeny na NFV infrastruktuře.
- **Management a orchestrace NFV (NFV-MANO)** - zde se jedná o řízení softwarových a hardwarových zdrojů v celé infrastruktuře NFV a životního cyklu jednotlivých virtuálních síťových funkcí. Tato část se tedy zaměřuje na řízení a správu

všech úloh související v virtualizaci v NFV frameworku. [15]

Tyto funkční bloky se ještě dále dělí, proto dále v této práci budou tyto jednotlivé části popsány podrobněji a současně k nim budou uvedeny různé možnosti řešení.

2.5.1 Infrastruktura NFV

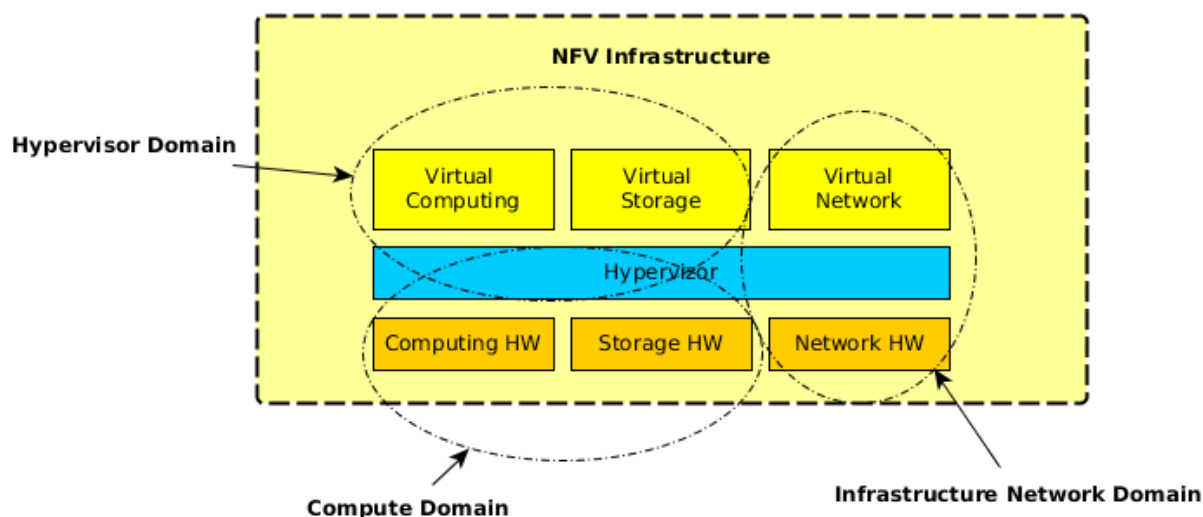
Ve zdroji [16], který detailně popisuje infrastrukturu pro virtualizaci síťových funkcí (NFVI), je uvedeno, že je v ní sdružení všech základních zdrojů potřebných pro běh virtuálních síťových funkcí (VNF). Z tohoto důvodu sem patří veškerý hardware. Do NFVI také patří některé softwarové komponenty, které jsou společné mnoha VNF a poskytují funkcionalitu potřebnou pro podporu nasazení, propojení či managementu VNF. Celou infrastrukturu může tvořit jeden či více strojů, které mají tyto potřebné funkce. Tyto stroje také mohou být umístěny v různých spolu spojených lokacích.

Pro zjednodušení lze celou NFV infrastrukturu rozdělit do 3 následujících domén:

- Compute Domain - Do této domény patří veškeré hardwarové zdroje jako jsou servery, úložiště a komponenty, které tyto zdroje obsahují, např. procesory, pevné disky, síťové karty, atd. Zároveň je zde řešen návrh fyzické topologie. [17]
- Hypervisor Domain - Toto je doména, které představuje softwarové prostředí abstrahující hardware v compute doméně a poskytuje je jako virtuální zdroje. Tyto zdroje následně mohou využívat virtuální síťové funkce. [18]
- Infrastructure Network Domain - V této doméně je řešeno veškeré propojení výše zmíněných domén. Tedy fyzické i virtuální infrastruktury.[19]

Funkci obsaženou v jednotlivých doménách znázorňuje obrázek č. 2.7. Více informací na tuto problematiku lze nalézt v [16] a ve zdrojích uvedených u každé domény.

Dá se říci, že referenční návrh infrastruktury pro NFV je podobný jako pro návrh infrastruktury pro cloud computing platformu. Měl by se tedy skládat z generických a komerčně vysoce dostupných serverů, které by měli být zapojeny do switchu a tím by měla být zajištěna konektivita. Na tyto servery je následně nasazen jeden z dostupných hypervisorů. Výběr správného hypervisoru, které jsou v současné době dostupné na trhu, je hlavní podmínka správného a funkčního návrhu této části NFV frameworku. Přehled hypervisorů je uveden v kapitole 2.6.1. V produkčním prostředí by součástí řešení bylo samozřejmě řešení síťového návrhu. Tato práce však má sloužit pouze jako ukázka a z tohoto důvodu zde nebude síťový návrh zmíněn.



Obrázek 2.7: Schéma NFV infrastruktury

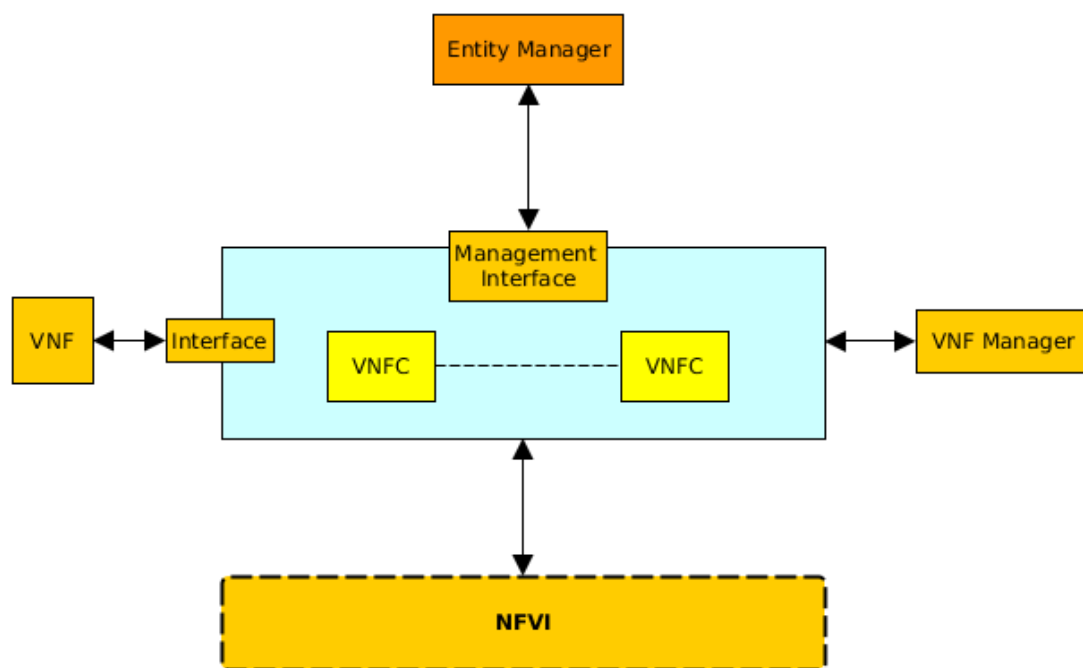
2.5.2 Virtuální síťová funkce

Virtuální síťová funkce (VNF) je dle [20] určitá síťová funkce, která běží na NFV infrastruktuře a je zároveň NFV frameworkem řízena a spravována. Zároveň musí mít dobře definované rozhraní k ostatním síťovým funkcím, k VNF Managerovi a měla by obsahovat management rozhraní či port. Jedna VNF může být obsažena v jednom virtuálním stroji nebo může být roztažena přes více virtuálních strojů.

Na obrázku č. 2.8 je vidět jednoduché schéma virtuální síťové funkce dle referenčního návrhu [20]. Celý životní cyklus VNF, což je vytvoření, spuštění, zastavení, smazání a škálování, řídí VNF Manager, který je součástí NFV managementu a orchestrace. Současně je možné dynamicky změnit aktuální konfiguraci pomocí Entity manageru (EM) přes management interface. EM může spravovat více VNF nebo právě jednu. Vnitřní struktura celé instance může být tvořena více komponentami (VNFC), které spolu mohou být navzájem provázány. Toto provázání však nemusí být viditelné zvenčí.

Pohledem na současný trh zjistíme, že VNF je prakticky poskytována ve 3 základních podobách.

- Softwarová aplikace - V tomto případě je poskytována VNF jako aplikace, která může být nainstalována na běžný operační systém jako je například GNU/Linux.
- Ucelený operační systém - Zde je poskytován přímo celý operační systém, který může být nainstalován do virtuálního stroje nebo i na fyzický server.
- Kompletní VM - Poskytovatel VNF může dát k dispozici rovnou přetvořený ob-



Obrázek 2.8: Schéma virtuální síťové funkce

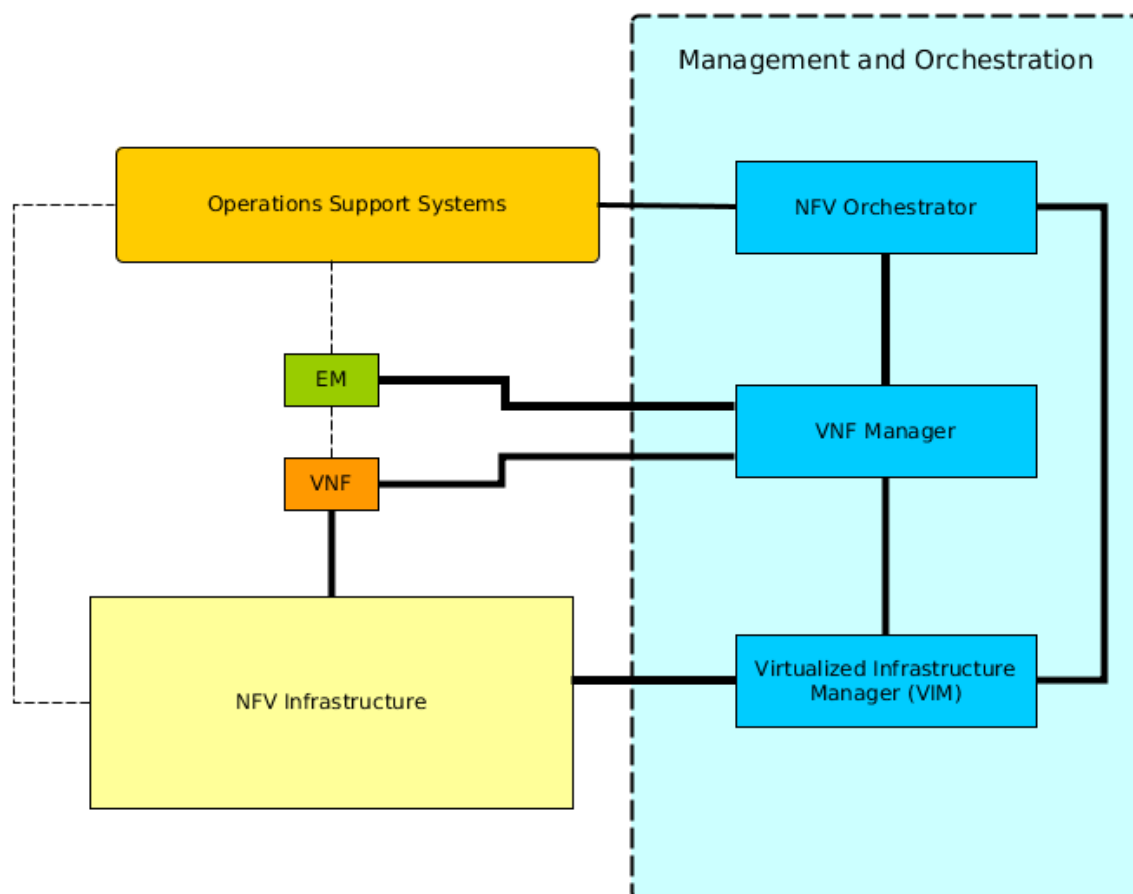
raz virtuálního stroje (image), který může obsahovat operační systém se síťovými funkcemi. Tento systém však nemusí být klasicky dostupný operační systém jako je GNU/Linux či FreeBSD, ale může se jednat o speciálně vytvořený systém od výrobce. Tento způsob budou využívat poskytovatelé, kteří mají proprietární řešení pro síťová řešení jako je například Cisco či Juniper.

2.5.3 Management a orchestrace NFV

Management a orchestrace virtualizace síťových funkcí (NFV MANO) je nejdůležitější část celého NFV frameworku. Je tomu tak, protože MANO zajišťuje správné fungování NFV infrastruktury i jednotlivých virtuálních síťových funkcí. MANO také poskytuje funkce nutné pro provisioning VNF a související operace, jako je jejich konfigurace jednotlivých VNF a infrastruktury, na které běží. Zároveň spravuje a řídí životní cyklus fyzických a virtuálních zdrojů, které slouží pro podporu VNF.

Jak vyplývá z obrázku č. 2.9, tak referenční návrh MANO dle [21] se skládá ze hlavních 3 částí, které se zabývají správnou jednotlivých vrstev NFV frameworku.

- Virtualized infrastructure manager (VIM) - Řídí a spravuje fyzické a virtuální zdroje v jedné doméně infrastruktury. Celková infrastruktura se může skládat z více domén a každá musí mít svůj VIM. Jeho typickými úlohami jsou vytváření,



Obrázek 2.9: Schéma NFV MANO

udržování a uvolňování VM na dostupných zdrojích v doméně. Zároveň musí mít přehled o všech těchto a stavu hardwarových zdrojů.

- VNF manager - Dohlíží na lifecycle management jednotlivých VNF instancí. To znamená, že vytváří, udržuje a ukončuje VNF instance, které běží na jednotlivých VM (ty však spravuje VIM). Opět může existovat více VNF managerů, kteří mohou spravovat jednu či více VNF.
- NFV orchestrator - Zjednodušeně slouží jako řízení a správu všech VIM a všech VNF managerů. Pomocí komunikace s VIM dokáže spravovat dostupné zdroje a pomocí komunikace s VNF managery dokáže řídit síťové služby. Jeho další funkcí je i přehled všech dostupných VNF, neboli katalog VNF, a registrace nových VNF do tohoto katalogu. Ten je pak dostupný uživatelům.

Celý systém je navržen tak, že by měl pracovat společně se stávajícími aplikacemi a

systémy, které potencionální uživatelé používají pro provoz své infrastruktury a podnikových procesů (Operation support system).

V oblasti NFV MANO probíhá v současnosti rozsáhlý vývoj a existuje několik projektů, které se tím zabývají. V článku [22] je nabídnut zajímavý přehled.

2.6 Možné technologie pro řešení

V předchozí části byla popsána referenční architektura, kterou navrhla ETSI. V té jsou specifikovány funkční požadavky a nastíněny potřebná rozhraní. Přesto lze tento návrh považovat za poněkud omezený v rozsahu. Není v něm například definována řízení a správa starších zařízení, což může velice zkomplikovat provoz síťové infrastruktury, která se skládá z VNF i těchto starších zařízení. Kromě toho standardy a referenční implementace VNF, infrastruktury, a MANO prozatím nejsou k dispozici.

Z tohoto důvodu následuje návrh možnosti pro každou z oblastí architektury, které v současnosti mohou sloužit jako její řešení.

2.6.1 Hypervisory

hypervisor je základná součástí cloudové platformy a frameworku pro virtualizaci síťových funkcí. Na trhu již existuje celá řada různých hypervisorů. Zde je uveden stručný přehled těch nejpoužívanějších.

- XEN - Je to hypervisor prvního typu, který pracuje na nejnižší vrstvě. Tato vrstva podporuje jeden nebo více hostovaných operačních systémů. První hostovaný systém se nazývá doménou 0 a slouží k přímému přístupu k hardwaru a jeho management. Do tohoto systému následně je možné přidávat další uživatelské domény, které mohou být linuxové systémy či Microsoft Windows.
- KVM - Jedná se o virtualizaci založenou na linuxovém jádru. Každá virtuální instance má svůj vlastní virtualizovaný hardware včetně síťové karty, disku a grafické karty. Tento typ hypervisoru vyžaduje pro správnou funkci procesor s rozšířením pro virtualizaci hardwaru.
- Microsoft Hyper-V - Zde se jedná o hypervisor od společnosti Microsoft, který lze nalézt v Windows Serverech od verze 2008. Hyper-V je hypervisorově stavěný serverový systém. To znamená, že má svůj hlavní operační systém a pomocí virtualizace se skrze něj mohou spustit další operační systémy.
- VMware ESXi - Varianta ESXi je odlehčená verze ESX klienta, která dovoluje běžet hostitelský systém na výměnném zařízení. Jde o tenký klient s vlastním linuxovým jádrem, který běží přímo nad hardwarovou vrstvou. Kernel ESXi funguje

jako hostitelský operační systém pro vrstvení dalších služeb. Kernel na sebe váže modul vmkernel s dalšími obslužnými funkcemi, který tvoří základní stavební kámen celého řešení. Výhodou tohoto řešení je možnost alokace co největšího množství hardwarových prostředků pro hostované systémy. Tenký klient totiž zbytečně nevyužívá systémové prostředky hostujícího serveru.

2.6.2 VNF

Pokud se podíváme na trh s VNF u některých vendorů, tak zjistíme, že mnozí poskytují virtuální instance, které se dají použít pro účely VNF v této práci. Tato práce je zaměřena především na funkce firewallu a proto zde jsou uvedeny příklady pouze pro ně. Uvedeny jsou hlavně produkty největších a nejpoužívanějších poskytovatelů síťových prvků a také open-source firewall.

- Juniper vSRX - Jde o firewall od společnosti Juniper, který je obdobou jejich fyzického zařízení Juniper SRX. Jde virtuální instanci poskytující funkce pro firewall, routing a pokročilé bezpečnostní funkce pro poskytovatele telekomunikačních služeb a větší společnosti. Toto VM je určené pro privátní, public i hybrid cloud.
- Fortigate-VM - Fortigate Virtual Appliances je řešení pro cloudové prostředí od společnosti Fortinet. Nabízí stejné funkce pro firewall jako jsou obsaženy ve Fortigate fyzických zařízeních.
- Cisco ASA v - Společnost Cisco nabízí Adaptive Security Virtual Appliance (ASA v), která obsahuje stejný software jako fyzické ASA zařízení a většinu funkcí pro firewall, routing a VPN.
- PFSense - PFSense je open-source projekt, který má za cíl poskytnout firewall postavený na operačním systému FreeBSD, který může běžet na klasické architektuře jednodeskových počítačů. Toto řešení poskytuje všechny důležité vlastnosti komerčních firewallů, má jednoduché ovládání a je to otevřené řešení.

2.6.3 Cloud platforma

Pro účely vytvoření infrastruktury, a vůbec možnost využití NFV v datovém centru, je nutná cloudová platforma. Existují několik řešení, které lze pro tyto účely použít. Dvě z nejčastějších jsou OpenStack a VMware vCloud.

2.6.3.1 OpenStack

OpenStack je open-source platformou umožňující postavit IaaS cloud, který může být nainstalován i na běžném hardwaru. Toto řešení má za cíl vytvořit dostupnou clou-

dovou platformu, která bude splňovat všechny potřeby privátních a veřejných cloudů nezávisle na velikosti řešení.

Celá stavba systému OpenStack se skládá z několika na sobě nezávislých projektů, které řeší různé oblasti cloudové platformy. Tyto projekty mezi sebou komunikují pomocí otevřených API a mohou být spravovány pomocí dashboardu. Celé administrace OpenStacku může být prováděna přes webové rozhraní, příkazovou řádku či přímo pomocí příkazů zaslaných do API. Celé toto řešení se vyznačuje jednoduchostí implementace, škálovatelností a rychlým vývojem nových vylepšení. Toto

2.6.3.2 VMware vCloud

Společnost VMware poskytuje pro privátní cloudové systémy své řešení, které označuje jako VMware vCloud.

2.6.4 SDN

Součástí řešení pro datové centrum je dnes i SDN. I zde existuje několik možností. Dvě z nejpoužívanější řešení jsou:

- OpenContrail
- VMware NSX

3 Popis navržených řešení

V předešlé kapitole byla vysvětlena základní problematika, která souvisí s virtualizací síťových funkcí, cloud computingem a softwarově definovanými sítěmi. Zároveň byla popsána referenční architektura frameworku pro virtualizaci síťových funkcí. Tato kapitola bude již věnována konkrétnímu příkladu využití virtuálních síťových funkcí v cloudovém prostředí. Nejprve zde popsána navržená architektura pro privátní cloudovou platformu využívající virtualizaci síťových funkcí, kterou mohou využívat všichni její uživatelé. Pro tuto cloudovou platformu a pro její uživatele byli navrženy dva příklady virtuálních síťových funkcí. U obou příkladů jsou uvedeny scénáře a způsob jakým jsou navrženy.

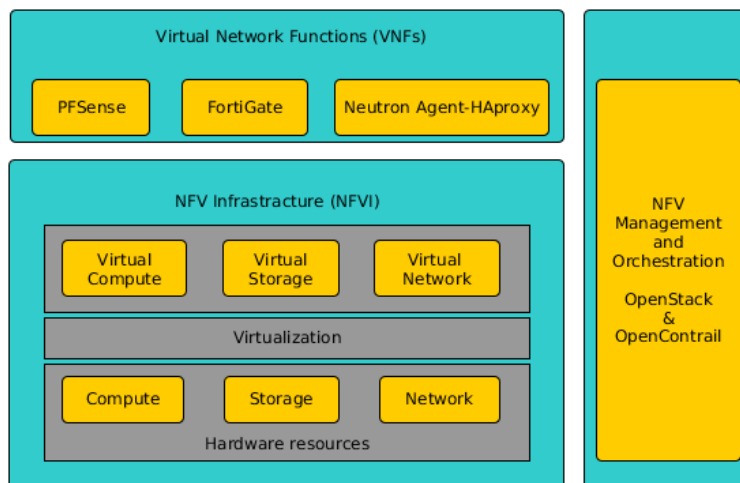
3.1 Požadavky na VNF řešení

- Univerzálnost - Celé řešení musí být postavené tak, aby ho mohli využívat všichni uživatelé dané cloudové platformy.
- Jednoduchost -
- Otevřenost a Flexibilita -
- Kompatibilita se stávající síťovými prvky -

3.2 Architektura navrženého řešení

Architektura navrženého řešení byla implementována pomocí cloudové platformy OpenStack a SDN řešení OpenContrail. Obrázku č. 3.1 znázorňuje tyto technologie v souvislosti s referenční architekturou popsanou v kapitole 2.5. Je nutné říci, že obě technologie nezapadají přímo do jedné z částí referenční architektury. Naopak v některých případech se překrývají nebo se v ní doplňují.

OpenStack byl zvolen, protože se jedná o největší open-source cloudovou platformu na světě. OpenStack tvoří část správy infrastruktury. Hardwarová vrstva infrastruktury se může skládat z libovolných serverů, na kterých je nainstalován KVM hypervizor. Tento hypervizor tvoří virtuální vrstvu a byl vybrán, protože je nejčastěji používán



Obrázek 3.1: Architektura NFV řešení

společně s OpenStackem. Avšak v případě potřeby by zde mohl být použit i jiný hypervizor, pokud bude zachována kompatibilita vůči OpenStacku.

OpenStack spravuje převážně zdroje týkající se výpočetního výkonu (Compute) a uložště (Storage). Tyto zdroje následně přiděluje dle potřeby virtuálním instancím nebo v našem případě instancím, které slouží jako VNF. Bylo však nutné zvolit řešení, které se bude starat o síťování.

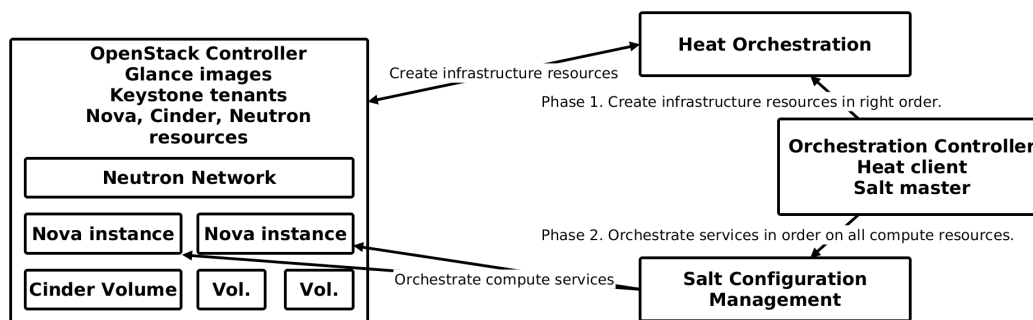
Speciálně pro vyřešení síťování v této infrastruktuře je součástí řešení OpenContrail. Díky tomu je možné vytvářet overlay sítě pomocí VXLAN či MPLSoverGRE, kterými jsou dynamicky propojovány jednotlivé VM a VNF.

Jednotlivá VNF mohou být v OpenContrailu vytvořena pomocí tzv. Servisních Instance a Servisní Templatů. Ty budou v této práci použity pro vytvoření VNF sloužící jako Firewall a budou podrobně popsány v kapitole věnující se vytváření této služby.

Další součástí, která musela být v architektuře navržena, je způsob řízení a správy jednotlivých VNF. Zde se muselo jednat o řešení, jakým automaticky vytvořit a popřípadě i smazat všechny potřebné části potřebné pro VNF. Pro tuto část byl zvolen Heat. Heat je část OpenStacku, která slouží pro automatickou orchestraci. Ten bude v tomto návrhu zastávat roli VFN managera, pomocí kterého budou jednotlivé VNF spravovány. Avšak dalo by se říci, že do této role spadá i OpenContrail, protože právě on umožňuje také spravovat jednotlivá VNF za běhu.

Heat je hlavní projekt v OpenStacku pro orchestraci. Umožňuje uživatelům popsat nasazení komplexních cloudových aplikací v jednom textovém souboru, který se nazývá Heat template. Tyto soubory se dají předat heat enginu, který podle nich dokáže automaticky vytvořit požadované zdroje v OpenStacku i v OpenContrailu.

Z toho návrhu je patrné, že zde není implementovaný NFV orchestrator. Je to zdůvodu toho, že pro účely řešení virtuálních síťových funkcí na cloudové platformě OpenStack s OpenContrailem, která je navržena v této práci, není tato část potřeba.



Obrázek 3.2: Popis heat orchestrace

3.3 Load balancer as a Service

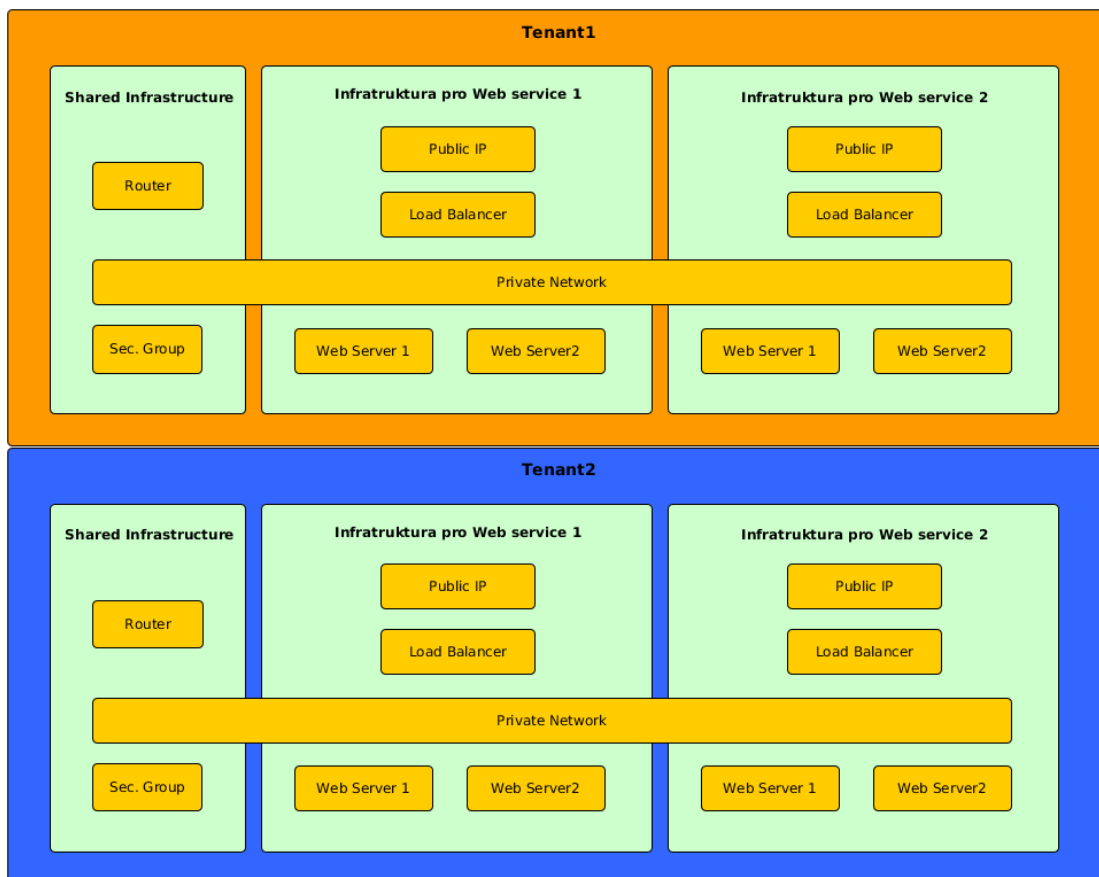
Jednou z nejčastějších síťových funkcí, která je v dnešních počítačových sítích a datových centrech vyžadována je load balancing. Load balancer spravuje příchozí komunikaci a distribuje ji do několika serverů či jiných síťových zdrojů. Tím je zajištěna rozloha zátěže. V cloudovém prostředí je tedy možné velice rychle, dle požadavků uživatele či automaticky dle nastavených parametrů, škálovat (přidávat či odstraňovat) webové servery. Load balancer zároveň monitoruje stav jednotlivých sleduje stav jednotlivých instancí a posílá komunika pouze správně fungujícím instancím.

Na obrázku č. 3.3 je vidět celý koncept load balanceru poskytovaného jako službu v privátním cloudu. Každý uživatel má možnost si dle potřeby vytvořit load balancer pro své webové servery. Pokud provozuje několik webových služeb v jednom projektu (tenantu), může pro každou tuto službu vytvořit vlastní load balancer, který bude nakonfigurován dle požadavků. Tento load balancer je dostupný pro všechny uživatele cloudu, tedy ve všech tenantech.

3.3.1 Neutron LbaaS

Při výběru řešení pro load balancing je nutné zvážit především výkon load balanceru. Na trhu již existuje celá řada fyzických i virtuálních produktů. Fyzická řešení nabízí větší výkon a obvykle i více funkcí. Virtuální pak větší flexibilitu v nasazení a jednodušší konfiguraci.

Avšak existuje jednotná množina funkcí, kterou uživatelé požadují a kterou většina load balancerů poskytuje. OpenStack Neutron navrhl LBaaS jako pokročilou službu



Obrázek 3.3: Load Balancer as a Service

Neutronu, která umožňuje použít jeden soubor API k využití load balancing funkce poskytnuté celou řadou poskytovatelů.

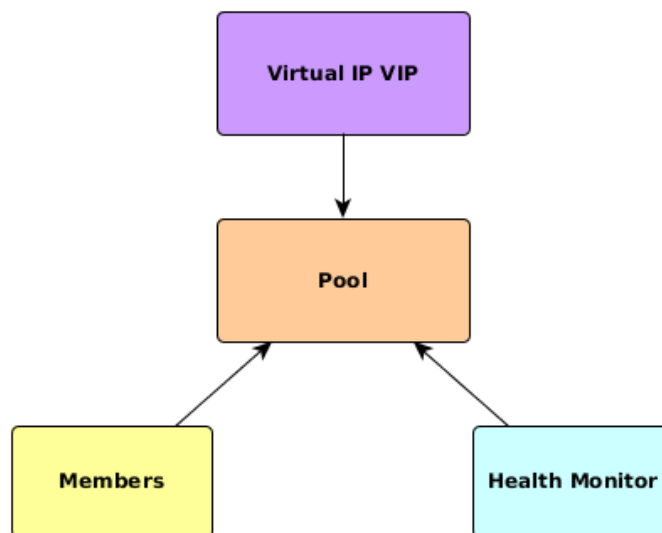
Zjednodušeně toto umožňuje uživatelům OpenStacku přes jedno společné rozhraní ovládat několik různých řešení pro load balancing. Zároveň díky tomu odpadá nutnost seznamování se s implementací a konfigurací těchto různých řešení, která mohou být velmi specifická a odlišná.

V této práci je ukázán příklad využití implementace load balanceru v OpenContrailu, který může být přes toto api ovládán. Avšak zde ukázané řešení může být použito s jakoukoli implementací load balanceru, ať už virtuálního (softwarového) či fyzického, pokud dokáže komunikovat s OpenStack Neutron LbaaS rozhraním. Dle dokumentace OpenContrailu [27] je v něm kontrétní implementace load balanceru řešena pomocí HAProxy. HAProxy je zdarma dostupný open source software pro unix operační systémy [33].

Load balancer se v Neutron LbaaS skládá ze 4 objektů.

- Pool - Označuje síťový rozsah pro webové servery
- Virtální IP (VIP) - ip adresa, na kterou přichází komunikace
- Member - Označuje konkrétní instanci, která je členem poolu.
- Monitor - Monitoruje stav aplikace. Pomocí HTTP, TCP či PING.

Obrázek č. 3.4 zachycuje jednotlivé závislosti mezi těmito objekty. Celý proces probíhá tak, že každý virtuální server, který je asociovaný s daným poolem z něj obdrží IP adresu. Když přijde na VIP nějaký dotaz na danou webovou aplikaci, tak je tento dotaz předán dál na jednu z těchto přiřazeným IP adres. Pokud nastane s aplikací či serverem nějaký problém, který zachytí monitor, tak load balancer ip adresu tohoto serveru přestane posílat komunikaci, dokud není vše zase v pořádku.



Obrázek 3.4: Neutron LbaaS

3.3.2 LbaaS heat template

Aby nemusel uživatel ručně vytvářet load balancer ručně, tak byl celý proces vytváření load balanceru zautomatizován pomocí heat šablony. Tím byla vytvořena VNF. Navržený heat template pro LbaaS v sobě obsahuje následující prostředky, které se po spuštění pokusí vytvořit.

- pool
- members

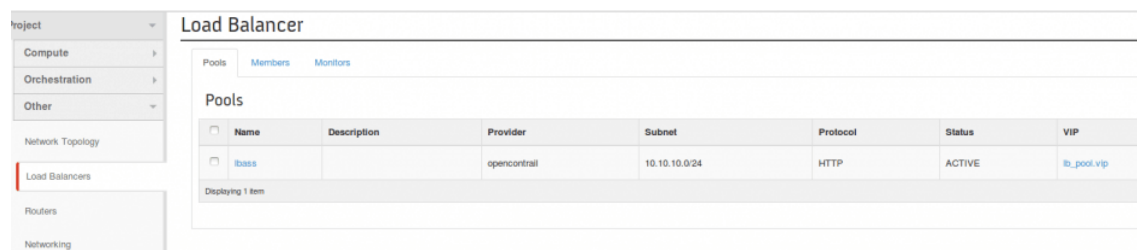
- health monitoring
- 2 web instance
- privatní síť
- public síť

3.3.3 Ukázka použití LbaaS heat templatu

Pro vytvoření heat stacku s Load balancerem je nutné daný template vytvořit pomocí příkazu:

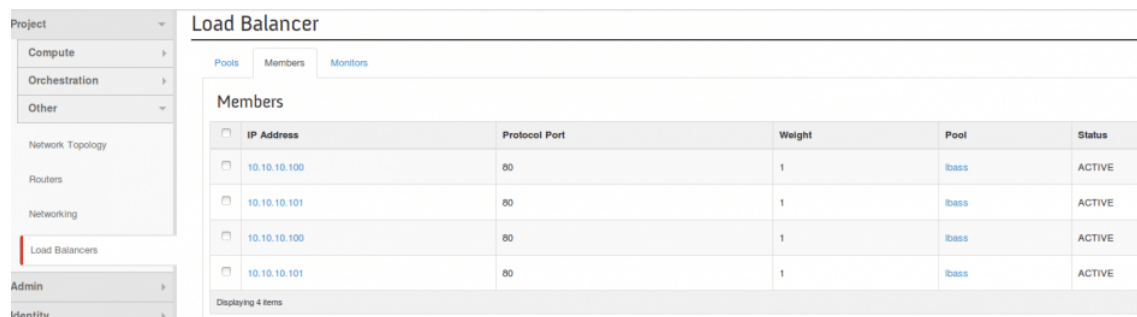
```
heat stack-create -f heat/templates/lbaas_template.hot -e heat/env/lbaas_env.env lbaas
```

Tento příkaz vytvoří všechny již uvedené prostředky pro load balancing. Konkrétní load balancer má nakonfigurovanou virtual ip adresu (VIP) a k ní přiřazenou floating adresu, která je přístupná z externích sítí. Zároveň má tento load balancer přiřazený pool, ke kterému je přiřazena privátní síť 10.10.10.0/24. Na obrázku č. X znázorňuje tento pool a obrázek č. X+1 jsou vidět členové (members) toho poolu.



Name	Description	Provider	Subnet	Protocol	Status	VIP
lbass		opencontrail	10.10.10.0/24	HTTP	ACTIVE	lb_pool.vip

Obrázek 3.5: Vytvořený pool

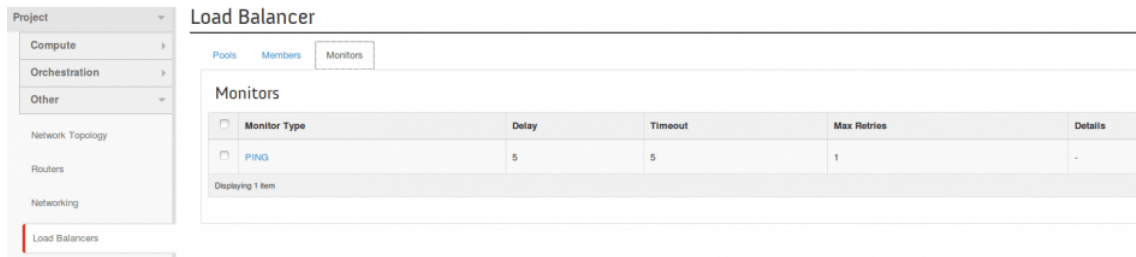


IP Address	Protocol Port	Weight	Pool	Status
10.10.10.100	80	1	lbass	ACTIVE
10.10.10.101	80	1	lbass	ACTIVE
10.10.10.100	80	1	lbass	ACTIVE
10.10.10.101	80	1	lbass	ACTIVE

Obrázek 3.6: Vytvoření members

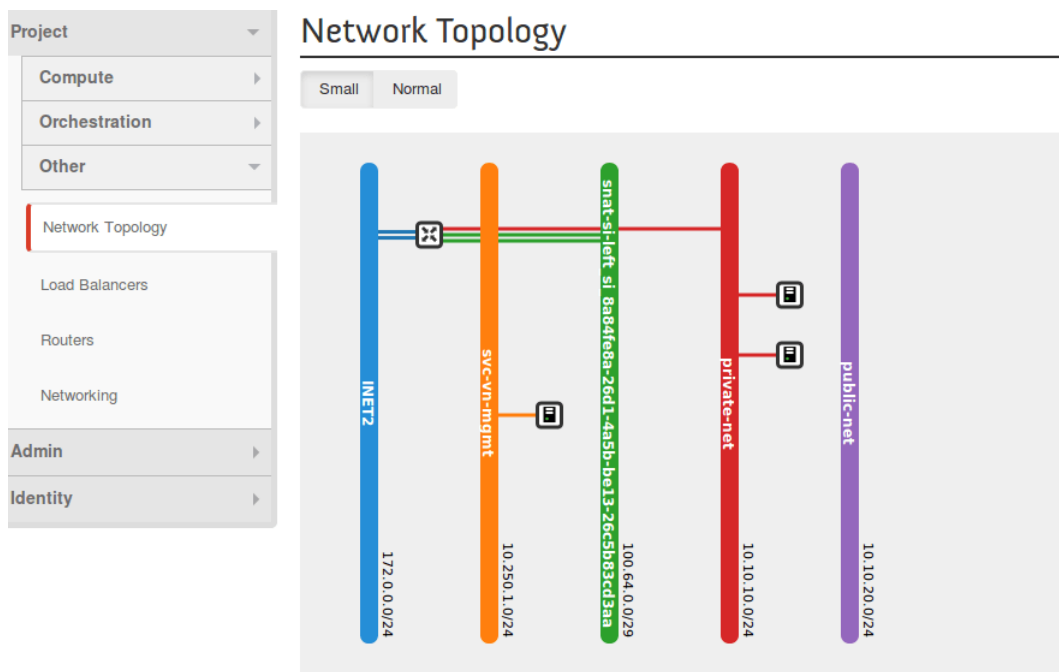
Dalším zdrojem, který byl vytvořen je health monitor, který lze vidět na obrázku č. X+2. Díky němu má load balancer přehled o aktuálním stavu webových instancí. Pokud

by náhodou některá z nich přestala odpovídat, v tomto případě na ping, tak by load balancer na tuto instanci přestal zasílat traffic.



Obrázek 3.7: Vytvořený health monitor

Finální síťovou topologií znázorňuje obrázek č. X+3.



Obrázek 3.8: Vytvořená síťová topologie

Otestování webových serverů lze provést příkazem curl, kterému dáme jako parameter ip VIP nebo floating ip load balanceru. Po několika takovýchto zadání tohoto příkazu je vidět, že oba web servery odpovídají a je probíhá mezi nimi load balancing metodou round robin. Celý tento test je vidět na obr. č. X+4


```
File Edit View Search Terminal Help
root@Management:~# curl 172.0.0.6
Instance 01
root@Management:~# curl 172.0.0.6
Instance 01
root@Management:~# curl 172.0.0.6
Instance 02
root@Management:~# curl 172.0.0.6
Instance 02
root@Management:~# curl 172.0.0.6
Instance 01
root@Management:~# curl 172.0.0.6
Instance 01
root@Management:~# curl 172.0.0.6
Instance 02
root@Management:~# curl 172.0.0.6
Instance 02
root@Management:~# curl 172.0.0.6
Instance 01
root@Management:~# curl 172.0.0.6
Instance 01
root@Management:~# curl 172.0.0.6
Instance 02
root@Management:~# curl 172.0.0.6
Instance 02
root@Management:~# curl 172.0.0.6
Instance 01
root@Management:~#
```

Obrázek 3.9: Test konektivity a load balancingu

3.4 Firewall as a Service

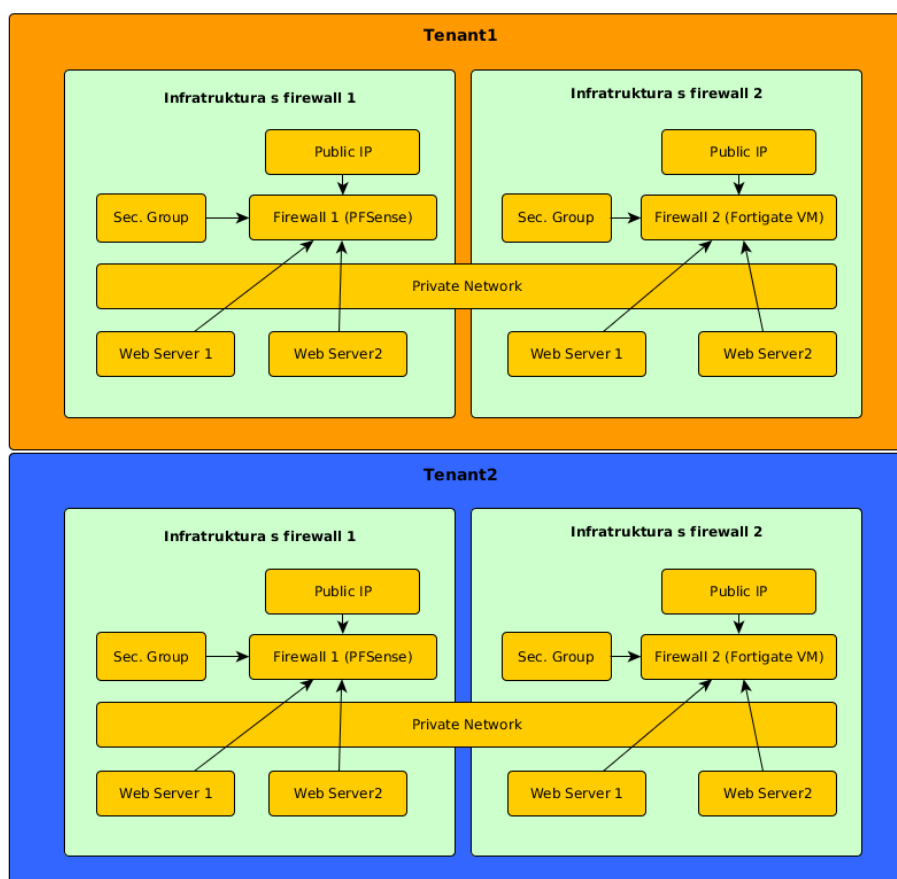
Servisní Template obsahuje obecný předpis pro danou VNF. Mezi tyto informace patří:

- **Název** - Název je označení daného Servisního Templatu. Pomocí něho lze následně identifikovat daný template a spustit dle jeho parametrů Servisní instanci.
- **Image** - Image je image, který má být použit pro vytvoření dané servisní instance. V našem případě se bude jednat o image, který obsahuje požadované síťové funkce. Tento image musí před tím než může být použit nahrán do OpenStacku přes glance.
- **Service Type** - V OpenContrailu, prozatím existují dva typy. Jsou to Traffic Analyzer a Firewall. Pro účely této práce bude používán pouze typ Firewall.
- **Service Mode** - Zde se určuje v jakém modu daný template bude nastaven. Jsou zde možnosti In-Network, In-Network-NAT.
- **Typy síťových portů** - Zde se určuje kolik portů bude daná instance, vytvořená pomocí tohoto templatu mít a jaká bude jejich role. Jsou zde možnosti Left, Right a Management.

Po úspěšném vytvoření Servisního templatu je možné z něj vytvořit libovolný počet Servis Instancí. Ty běží jako klasické instance v OpenStacku, avšak OpenContrail s nimi zachází jiným způsobem.

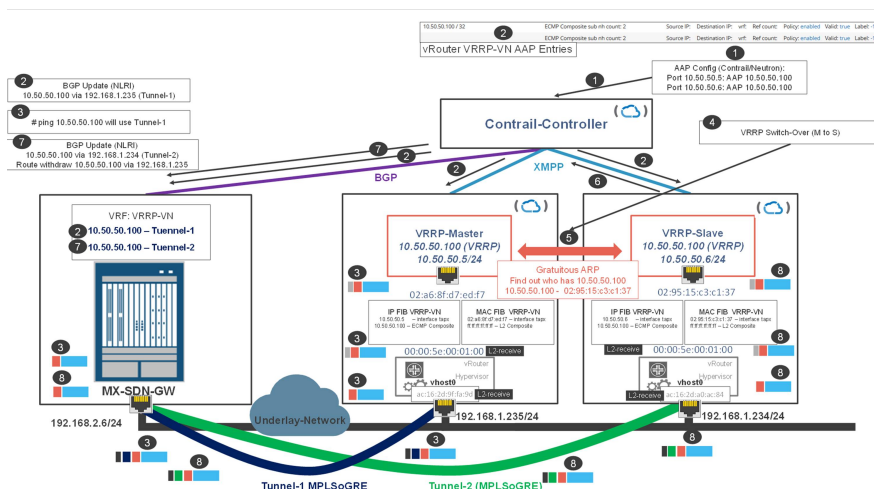
- 1 firewall instanci
- 1 testovací instanci
- 1 management instanci
- management síť
- privátní síť
- contrail policy

3.4.1 Scénář NAT



Obrázek 3.10: Firewall as a Service

3.4.2 Scénář HA firewall



Obrázek 3.11: High Availability Firewall

3.4.3 Fwaas template

Pro Fwaas je narhnut heat template, který obsahuje:

3.5 Ukázka použití Fwaas heat templatu

Pro vytvoření heat stacku s PFSense z templatu lze použít příkaz:

```
heat stack-create -f heat/templates/fwaas_mnmg_template.hot -e he-
at/env/fwaas_pfsense_env.env pfsense
```

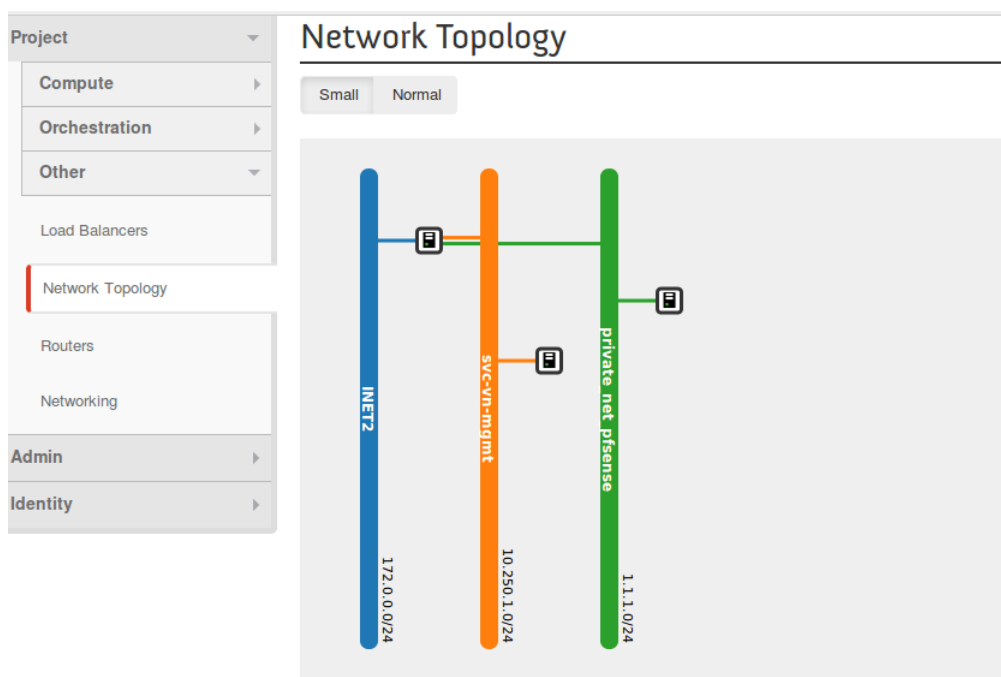
a pro vytvoření heat stacku s Fortigate VM jde vytvořit pomocí příkazu:

```
heat stack-create -f heat/templates/fwaas_mnmg_template.hot -e he-
at/env/fwaas_fortios_contrail.env fortios
```

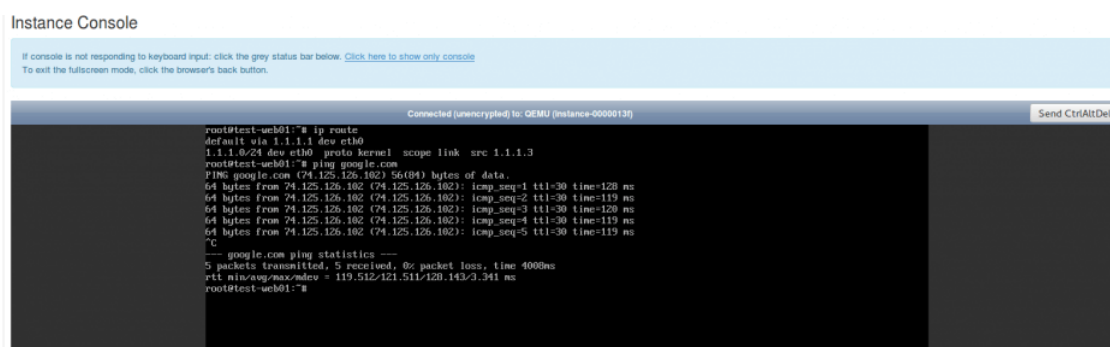
By default, pfsense firewall is configured to NAT after the heat stack is started. As a result, there is no need to make any configuration for this function. Pfsense image was preconfigured with DHCP services on every interface and there is outbound policy for NAT.

After we start the heat with pfsense there is already functional service chaining. Testing instance has default gateway to contrail and contrail redirects it to pfsense.

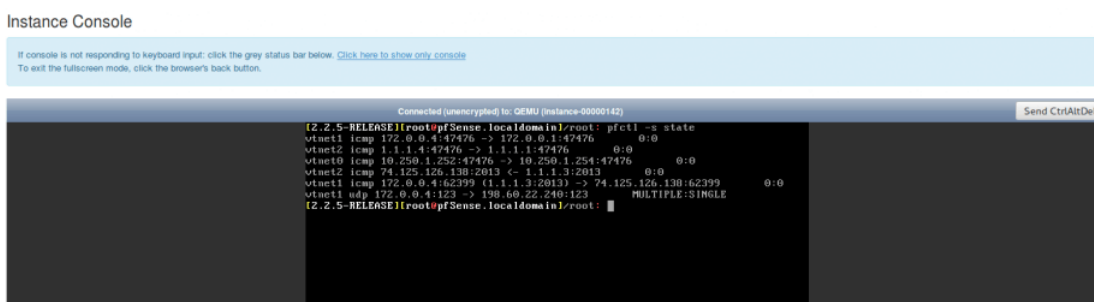
There is also NAT session in pfsense. In shell run command:



Obrázek 3.12: Síťová topologie



Obrázek 3.13: Test konektivity PFSense



Obrázek 3.14: Ukázka NAT session

```
root@mnmg01:~# python fortios_intf.py
This is the diff of the conigs:

This is how to reach the desired state:
config system interface
  edit port1
    set allowaccess ssh ping http https
  next
  edit port2
    set defaultgw enable
  next
  edit port4
    set mode static
  next
  edit port5
    set mode static
  next
  edit port6
    set mode static
  next
  edit port7
    set mode static
  next
  edit ssl.root
    set mode static
  next
end
root@mnmg01:~#
```

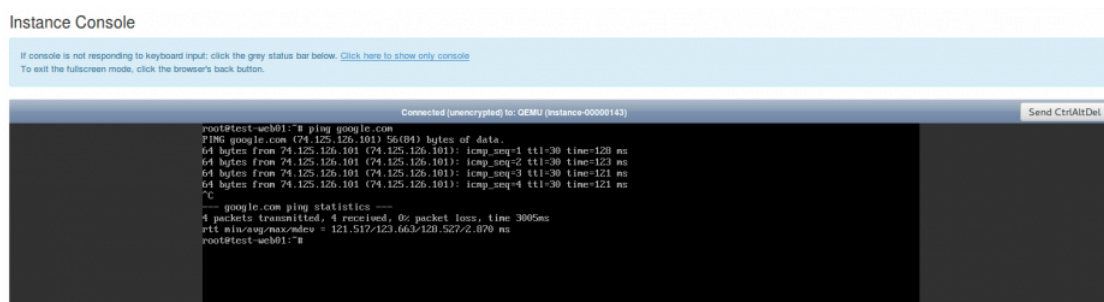
Obrázek 3.15: Fortigate VM intergace konfigurace

```
ubuntu@Management:~$ ssh root@172.0.0.5
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.19.0-26-generic x86_64)

 * Documentation:  https://help.ubuntu.com/
Last login: Tue Jan 12 10:03:49 2016 from mgmtserver14041vag
root@mnmg01:~# ls
fabfile.py  fortigate-formula  fortios_intf.txt  fortios_nat.py  param.py  update.sh
fabfile.pyc  fortios_intf.py  fortios_nat.conf  fortios_nat.txt  text.py
root@mnmg01:~# python fortios_nat.py
This is the diff of the conigs:

This is how to reach the desired state:
config firewall policy
  edit 1
    set nat enable
    set service ALL
    set schedule always
    set srcaddr all
    set dstintf port2
    set srcintf port3
    set action accept
    set dstaddr all
    set logtraffic all
  next
end
root@mnmg01:~#
```

Obrázek 3.16: Fortigate VM NAT konfigurace



Obrázek 3.17: Test konektivity

4 Shrnutí poznatků

K čemu to je dobrý, na co jsem narazil, atd.

5 Závěr

Je v paráda.

Literatura

- [1] R. Guerzoni, "Network Functions Virtualisation: An Introduction, Benefits, Enablers, Challenges and Call for Action. Introductory white paper," in SDN and OpenFlow World Congress, June 2012. [online]. [cit. 2016-04-07]. Dostupné také z: https://portal.etsi.org/NFV/NFV_White_Paper.pdf
- [2]
- [3] Open platform for nfv. <https://www.opnfv.org/>. Accessed September 28, 2014.
- [4] MIJUMBI, Rashid, Joan SERRAT, Juan-Luis GORRICHIO, Niels BOUTEN, Filip DE TURCK a Raouf BOUTABA. *Network Function Virtualization: State-of-the-Art and Research Challenges*. IEEE Communications Surveys. 2016, 18(1), 236-262. DOI: 10.1109/COMST.2015.2477041. ISSN 1553-877x. Dostupné také z: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7243304>
- [5] HAN, Bo, Vijay GOPALAKRISHNAN, Lusheng JI a Seungjoon LEE. *Network function virtualization: Challenges and opportunities for innovations*. IEEE Communications Magazine. 2015, 53(2), 90-97. DOI: 10.1109/MCOM.2015.7045396. ISSN 0163-6804. Dostupné také z: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7045396>
- [6] KUSNETZKY, Dan. *Virtualization: a manager's guide*. Sebastopol, CA: O'Reilly, c2011. ISBN 1449306454.
- [7] J. Smith and R. Nair, "The architecture of virtual machines," Computer, vol. 38, no. 5, pp. 32–38, May 2005. doi: 10.1109/MC.2005.173
- [8] <http://www.rightscale.com/blog/cloud-industry-insights/cloud-computing-trends-2015-state-cloud-survey>
- [9] K. Chandrasekaran. *Essentials of CLOUD COMPUTING*. Boca Raton: CRC Press, 2015. ISBN 978-1-4822-0544-2.

- [10] JENNINGS, Brendan a Rolf STADLER. *Resource Management in Clouds: Survey and Research Challenges*. Journal of Network and Systems Management. 2015, 23(3), 567-619. DOI: 10.1007/s10922-014-9307-7. ISSN 1064-7570. Dostupné také z: <http://link.springer.com/10.1007/s10922-014-9307-7>
- [11] ETSI, “Network Function Virtualization: Use Cases”, http://www.etsi.org/deliver/etsi_gs/NFV/001_099/001/01.01.01_60/gs_NFV001v010101p.pdf, 2013
- [12] KREUTZ, Diego, Fernando M. V. RAMOS, Paulo ESTEVES VERISSIMO, Christian ESTEVE ROTHENBERG, Siamak AZODOLMOLKY a Steve UHLIG. *Software-Defined Networking: A Comprehensive Survey*. Proceedings of the IEEE [online]. 2015, 103(1), 14-76 [cit. 2016-04-09]. DOI: 10.1109/JPROC.2014.2371999. ISSN 0018-9219. Dostupné z: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6994333>
- [13] DOHERTY, Jimmy. *SDN and NFV simplified: a visual guide to understanding software defined networks and network function virtualization*. 1st edition. Indianapolis, IN: Addison-Wesley Professional, 2016. ISBN 9780134306407.
- [14] ETSI Industry Specification Group (ISG) NFV, “ETSI GS NFV 002 V1.2.1: Network Functions Virtualisation (NFV); Architectural Framework,” December 2014. [online]. [cit. 2016-04-07]. Dostupné také z: <http://www.etsi.org/deliver/etsisigs/NFV/001099/002/01.02.0160/gsNFV002v010201p.pdf>
- [15] ETSI Industry Specification Group (ISG) NFV, “ETSI GS NFV 003 V1.2.1: Network Functions Virtualisation (NFV); Terminology for Main Concepts in NFV,” December 2014. [online]. [cit. 2016-04-07]. <http://www.etsi.org/deliver/etsisigs/NFV/001099/003/01.02.0160/gsNFV003v010201p.pdf>
- [16] http://www.etsi.org/deliver/etsi_gs/NFV-INF/001_099/001/01.01.01_60/gs_nfv-inf001v010101p.pdf
- [17] http://www.etsi.org/deliver/etsi_gs/NFV-INF/001_099/003/01.01.01_60/gs_NFV-INF003v010101p.pdf
- [18] http://www.etsi.org/deliver/etsi_gs/NFV-INF/001_099/004/01.01.01_60/gs_nfv-inf004v010101p.pdf
- [19] http://www.etsi.org/deliver/etsi_gs/NFV-INF/001_099/005/01.01.01_60/gs_NFV-INF005v010101p.pdf

-
- [20] http://www.etsi.org/deliver/etsi_gs/NFV-SWA/001_099/001/01.01.01_60/gs_nfv-swa001v010101p.pdf
- [21] http://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_nfv-man001v010101p.pdf
- [22] MIJUMBI, Rashid, Joan SERRAT, Juan-luis GORRICHIO, Steven LATRE, Marinós CHARALAMBIDES a Diego LOPEZ. *Management and orchestration challenges in network functions virtualization*. IEEE Communications Magazine. 2016, 54(1), 98-105. DOI: 10.1109/MCOM.2016.7378433. ISSN 0163-6804. Dostupné také z: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7378433>
<http://network-functions-virtualization.com/mano.html>
- [23]
- [24]
- [25]
- [26]
- [27]
- [28]
- [29]
- [30]
- [31]
- [32]
- [33]

Přílohy

Seznam obrázků

2.1	Koncept virtualizace síťových funkcí (NFV)	5
2.2	Schéma hypervisorů	6
2.3	Schéma SDN, převzato z [12]	9
2.4	Ukázka klasického service chainigu pomocí fyzických síťových prvků . .	11
2.5	Ukázka VNF service chainigu	11
2.6	NFV architektura, převzato z [14]	12
2.7	Schéma NFV infrastruktury	14
2.8	Schéma virtuální síťové funkce	15
2.9	Schéma NFV MANO	16
3.1	Architektura NFV řešení	21
3.2	Popis heat orchestrace	22
3.3	Load Balancer as a Service	23
3.4	Neutron LbaaS	24
3.5	Vytvořený pool	25
3.6	Vytvoření members	25
3.7	Vytvořený health monitor	26
3.8	Vytvořená síťová topologie	26
3.9	Test konektivity a load balancingu	27
3.10	Firewall as a Service	28
3.11	High Availability Firewall	29
3.12	Síťová topologie	30
3.13	Test konektivity PFSense	31
3.14	Ukázka NAT session	31
3.15	Fortigate VM intergace konfigurace	31
3.16	Fortigate VM NAT konfigurace	32
3.17	Test konektivity	32

Seznam tabulek

Seznam ukázek kódu

Podklad pro zadání DIPLOMOVÉ práce studenta

PŘEDKLÁDÁ:	ADRESA	OSOBNÍ ČÍSLO
Smola Ondřej	Polizy 16, Osice - Polizy	11475

TÉMA ČESKY:

Orchestrace a management virtuálních síťových funkcí

TÉMA ANGLICKY:

Orchestration and management of virtual network functions

VEDOUcí PRÁCE:

Ing. Vladimír Soběslav, Ph.D. - KIT

ZÁSADY PRO VYPRACOVÁNÍ:

Cílem této práce je analyzovat možnosti vytváření a nasazení virtuálních sítí v cloud computingu s důrazem na technologie VNF nad NFV a jejich srovnání. V rámci závěrečné práce budou analyzovány metody a nástroje pro vývoj a automatizaci služeb virtuálních sítí. V závěrečné části provede autor implementaci VNF řešení v prostředí cloud computingové platformy OpenStack.

Osnova:

1. Úvod
2. Problematika virtualizace síťových funkcí
3. Testovací prostředí
4. Příklad virtualizace síťových funkcí
5. Shrnutí
6. Závěr

SEZNAM DOPORUČENÉ LITERATURY:

DOSTÁLEK, Libor.; KABELOVÁ, Alena. Velký průvodce protokoly TCP/IP a systémem DNS. 5. aktualizované vydání, Brno: Computer Press, a.s., 2008. 488 s. ISBN 978-80-251-2236-5.

HICKS, Michael. Optimizing Applications on Cisco Networks. 1. vydání. Indianapolis: Cisco Press, 2004. 384 s. ISBN: 978-1-58705-153-1.

HUCABY, David. CCNP SWITCH 642-813 Official Certification Guide. 1. vydání. Indianapolis: Cisco Press, 2011, 533 s. ISBN 978-1-58720-243-8.

Podpis studenta:

Datum:

Podpis vedoucího práce:

Datum: