Univerzita Hradec Králové
Fakulta informatiky a managementu
Katedra informačních technologií

Softwarově definované sítě
Vývoj aplikace v prostředí OnePK

Bakalářská práce

Autor: Lukáš Nemecz
Studijní obor: AI3

Vedoucí práce: Ing. Agáta Milanov, Ph.D.

Hradec Králové duben 2016

Prohlášení:

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně a s použitím uvedené literatury.

V Hradci Králové dne 25.4.2016 Lukáš Nemecz

Poděkování:

Děkuji vedoucí bakalářské práce Ing. Agátě Milanov, Ph. D. za metodické vedení a odborné rady při psaní práce.

Anotace

Bakalářská práce se zabývá technologií softwarově definovaných sítí (SDN) a vývojem aplikace v prostředí OnePK od společnosti Cisco. V práci jsou popsány principy SDN, jejich výhody a možné problémy, stručný popis OpenFlow a programovatelnosti SDN. Dále je představeno prostředí Cisco OnePK a jeho funkce, výhody, zabezpečení a některé nedostatky. Praktická část práce se zabývá samotným vývojem aplikace zjednodušující konfiguraci směrovačů a obsahuje funkci implementace vlastní logiky směrování.

Hlavním cílem práce je představení principu SDN, předvedení možností nástroje OnePK a následně vytvoření funkční aplikace za použití tohoto nástroje.

Annotation

Title: Software-Defined Networking

Subtitle: Development of application in OnePK environment

This Bachelor thesis is focused on technology of software-defined networking (SDN) and development of application in OnePK environmet from Cisco company. There are described principles of SDN as well as its benefits and possible challenges with brief description of OpenFlow and SDN programmability. Also Cisco OnePK environment is introduced with its functions, advantages, security and some drawbacks. Last part of this thesis is focused on developing the application which simplyfies configuration of routers and contains function which implements custom logic of routing.

Main goal of this Bachelor thesis is to introduce SDN principle and to show possibilities of OnePK followed by development of functional application using this tool.

Obsah

1 Úvod

Technologie v oblasti výpočetní techniky se neustále vyvíjejí a zdokonalují a jinak tomu není ani v případě počítačových sítí. V současné době se ve spojitosti s počítačovými sítěmi často objevuje slovní spojení „softwarově definované sítě“. Jedná se o relativně nový trend, který se primárně zaměřuje na programovatelnost sítě a má potenciál vyřešit mnoho současných problémů, se kterými se počítačové sítě běžně potýkají, jako je bezpečnost, dostupnost, spolehlivost apod.

Společnost Cisco Systems, Inc [1] (dále jen Cisco) je jednou z největších společností v oblasti počítačových sítí na světě a jedním z předních výrobců síťových zařízení. Cisco existuje od roku 1984 a v současné době má přes 70 tisíc zaměstnanců po celém světě, včetně České Republiky. Společnost investuje své prostředky i na vzdělávání prostřednictvím své internetové akademie (Cisco Networking Academy [2]) s níž spolupracuje i Univerzita Hradec Králové. Cisco se rovněž zaměřuje na vývoj nových technologií a jednou z jeho odpovědí na softwarově definované sítě je nástroj OnePK [3]. [1]

V prostředí nástroje OnePK má uživatel možnost vytvářet vlastní aplikace, které budou schopné řídit operace v počítačové síti. Tato možnost je velkým přínosem, jelikož takové aplikace mohou značně urychlit a zjednodušit konfiguraci síťových zařízení jak pro pokročilé uživatele, tak pro začátečníky. Hlavní výhodou těchto aplikací by však mohla být možnost vytvoření zcela nových funkcí pro síťová zařízení naprogramováním jejich chování v závislosti na situaci.

Hlavním cílem této práce je představení trendu softwarově definovaných sítí a poukázání nejen na výhody tohoto přístupu, ale i na možné problémy. Dále si práce klade za cíl představení prostředí OnePK, popis jeho funkcí a nakonec vytvoření a zdokumentování funkční aplikace sloužící pro zjednodušení konfigurace směrovačů a obsahuje funkci, která implementuje vlastní logiku směrování, za použití tohoto nástroje.

2 Rešerše literatury

Tato bakalářská práce se zabývá relativně mladou oblastí počítačových sítí, pro kterou zatím existuje jen malé množství literatury v tištěné formě. Citováno bylo hlavně z internetových zdrojů, kterých je o této problematice větší množství. Většina publikovaných prací o probírané problematice je v anglickém jazyce, který obsahuje pojmy, které český jazyk neumí vždy správně vyjádřit či nahradit (např. forwarding, controller, apod.), a proto pro takové pojmy v citovaných zdrojích musel být zvolen správný překlad. Tento překlad pak musel být dodržován v rámci celé práce.

V práci bylo citováno z mnoha zdrojů, ze kterých lze dva označit jako hlavní. Zmíněnými hlavními zdroji jsou: Open Networking Foundation [4] (dále jen ONF) a Cisco.

Organizace ONF byla založena v roce 2011 a jejím hlavním cílem je podpora a šíření softwarově definovaných sítí (dále jen SDN) pomocí vývoje veřejně dostupných standardů, například OpenFlow [5]. Z tohoto důvodu se ONF se stala hlavním zdrojem pro tuto práci, zejména pro popis základních principů a architektury SDN a popisu OpenFlow.

Druhým hlavním zdrojem byly publikace od společnosti Cisco, ze kterých byly převážně čerpány informace o nástroji OnePK, v menší míře pak o SDN i OpenFlow.

Ostatní citované zdroje sloužily pro doplnění informací pro lepší pochopení dané problematiky. Těmito zdroji byly například dvě knihy: SDN: software defined networks od Thomase D. Nadeau [11] a Introduction to software defined networking: OpenFlow od Vishala Shukly [14]. Dalšími zdroji byly hlavně internetové publikace z internetových magazínů, publikace a články z různých společností a univerzit.

3 Úvod do technologie SDN

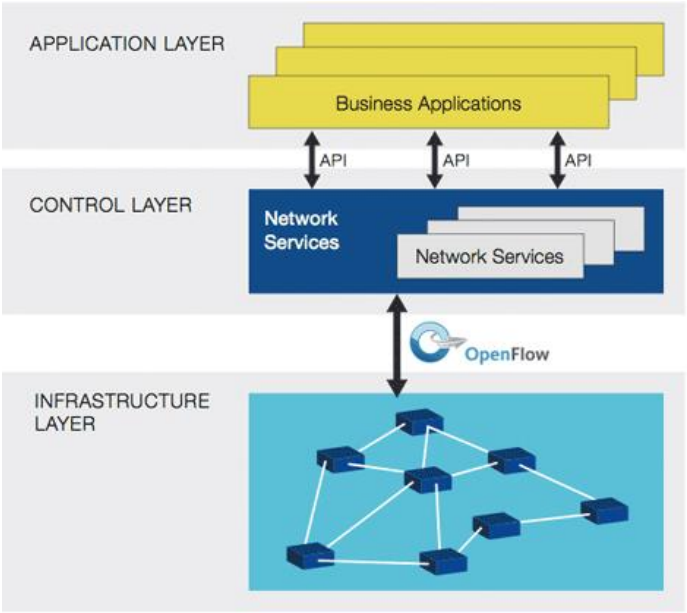
Tato kapitola představuje technologii SDN. Popisuje nejen základní principy, funkce a využití této technologie, ale i některé problémy spojené s jejím nasazením.

3.1 Výhody SDN

SDN má spoustu výhod a řeší mnoho problémů. Mezi výhody patří centralizovaná správa a řízení zařízení nezávisle na výrobci. Další výhodou je rychlá inovace a to díky schopnosti realizovat nové síťové možnosti a služby bez nutnosti konfigurace jednotlivých zařízení. Velkou výhodou je i jednoduchá programovatelnost za použití běžných programovacích prostředí, čímž vznikají nové možnosti vývoje. Zvýšené zabezpečení a spolehlivost sítě jsou výsledkem centralizované a automatizované správy síťových prvků, jednotných pravidel a redukce možných chyb při konfiguraci. SDN poskytuje vysokou rychlost, flexibilitu a také značně snižuje celkové náklady na správu sítě. [6]

3.2 Základní principy

SDN je přímo programovatelná síťová architektura, ve které je řízení sítě odděleno od datového toku. Takto oddělené řízení, které bylo dříve pevně vázáno na jednotlivých síťových zařízeních, do přístupných výpočetních zařízení umožňuje abstrakci základní infrastruktury pro aplikační a síťové služby, které mohou spravovat síť jako logickou nebo virtuální entitu. [6]



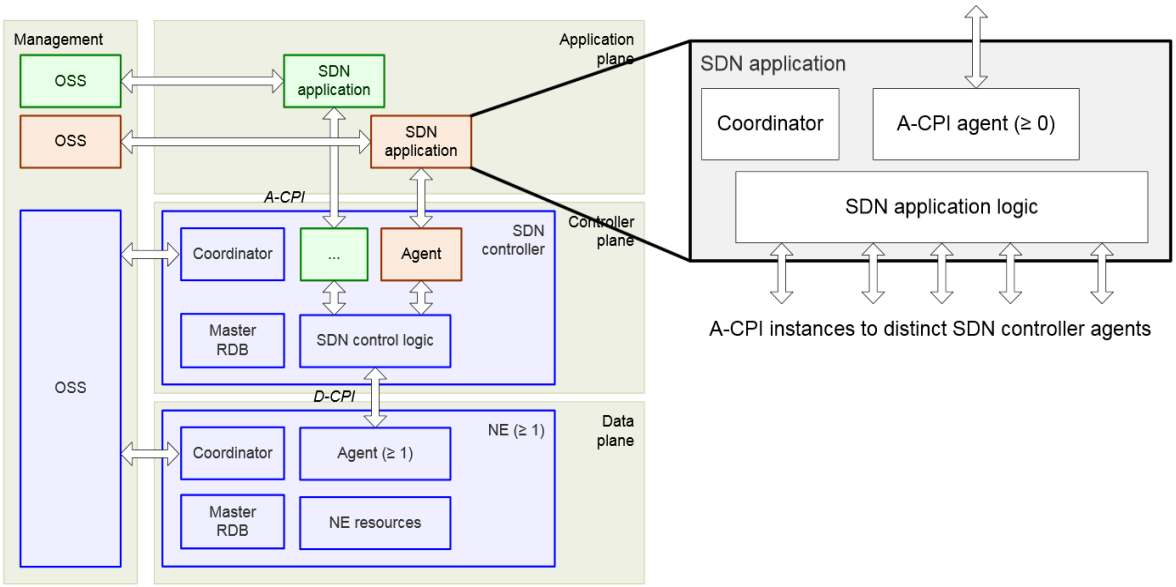
Obr. č. 1 Architektura softwarově definované sítě. [6]

Obrázek č. 1 znázorňuje logický pohled na SDN architekturu. Funkce sítě jsou obsaženy v SDN kontrolerech (angl. controller), které si udržují celkový přehled sítě. Síť se aplikacím jeví jako jeden logický přepínač (angl. switch). Díky SDN získávají podniky kontrolu nad celou sítí z jediného logického bodu nezávisle na výrobci hardwaru, což velmi zjednodušuje síťové operace a návrh. SDN také značně zjednodušuje samotná síťová zařízení, jelikož nepotřebují umět zpracovávat velké množství protokolových standardů, ale pouze přijímat instrukce od SDN kontrolerů. [6]

3.2.1 Aplikační vrstva

Aplikační vrstva obsahuje aplikace využívající základní síť. Aplikace standardně nemohou mít vliv na základní síť, nicméně SDN umožňuje aplikacím požádat o konkrétní chování sítě od kontroleru. Například, kontroler může směrovat provoz v síti, který patří k určité aplikaci, jinak, než zbylý provoz v síti ze stejných hostů. [7]

Označení rozhraní z SDN kontroleru na aplikační vrstvu může mít různá zastoupení. Společnost Open Networking Foundation ho ve své literatuře označuje application-controller plane interface (A-CPI). V jiných zdrojích lze nalézt označení northbound interface (NBI), případně northbound applications programming interface (northbound API). Pro tuto práci bylo vybráno označení A-CPI. Rozhraní slouží pro komunikaci s SDN kontrolerem, jak je znázorněno na obrázku č. 2. [8]



Obr. č. 2 Detail SDN aplikace. [8]

SDN aplikace může vyvolat jiné externí služby a využít jakýkoliv počet SDN kontrolerů pro splnění svých úkolů. **OSS spoj** (Operations support system) a funkce

koordinátoru naznačují, stejně jako jiné hlavní bloky architektury, že i SDN aplikace vyžadují určitou znalost jejich prostředí a rolí.

Koordinátor je funkční komponenta SDN kontroleru, která se chová jako manažer. Ve všech pohledech na datové, kontrolní a aplikační modely vrstev, vyžadují klienti a servery řízení, proto je koordinátor obsažen ve všech třech vrstvách.

Entita aplikační vrstvy může mít několik rolí. V roli klienta operuje na instanci vystavené serverovou entitou, která může představovat SDN kontroler nebo podřízenou aplikaci. V roli serveru vytváří instance pro ostatní aplikace. Tyto aplikace jsou klienti, kteří komunikují se serverovým agentem SDN aplikace. Entita aplikační vrstvy může mít také obě role zároveň. [8]

SDN aplikace jsou programy, které explicitně a programově komunikují s SDN kontrolerem přes A-CPI, aby vyjednali své požadavky a potřebné chování sítě. Kromě toho mohou získat abstraktní pohled na síť pro své vnitřní rozhodovací účely. [9]

3.2.2 Kontrolní vrstva

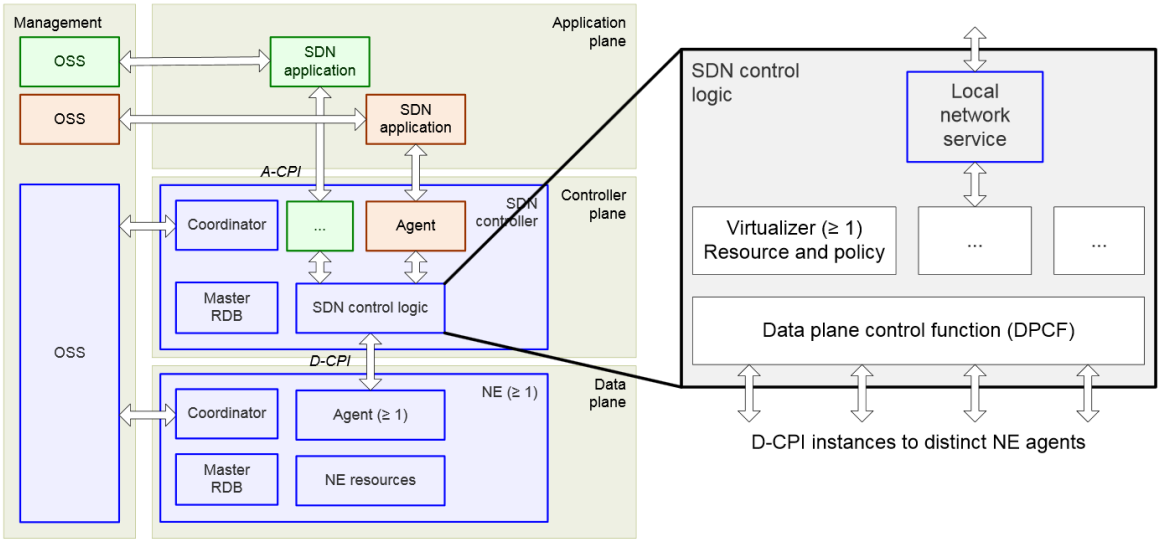
Kontrolní vrstva je logicky centralizována a oddělena od datové vrstvy. SDN kontroler, při komunikaci s SDN aplikacemi, překládá požadavky aplikací a řídí datové cesty SDN. Veškerá rozhodnutí jsou založena na současném pohledu na celou síť a ne v rámci limitované viditelnosti jednotlivého síťového skoku (angl. network hop), jak je tomu u dnešních směrovačů (angl. router). SDN kontroler lze chápat jako operační systém sítě, který buduje a prezentuje logickou mapu sítě službám nebo aplikacím, které jsou implementovány přímo na ni. [10]

Kontrolní vrstva vytvoří místní datový soubor, který je použit k vytvoření záznamů v předávací tabulce (angl. forwardnig table). Tyto záznamy jsou použity datovou vrstvou ke směrování provozu v síti mezi vstupními a výstupními porty zařízení. Datový soubor, který se používá na uložení síťové topologie, se nazývá báze směrovacích informací (angl. routing information base (RIB)). RIB je obvykle udržována v konzistentním (bezesmyčkovém) stavu skrze výměnu informací mezi ostatními instancemi kontrolní vrstvy. Záznamy v předávací tabulce jsou běžně nazývány bází předávacích informací (angl. forwarding information base (FIB)) a často jsou zrcadleny mezi kontrolní a datovou vrstvou zařízení. FIB je naprogramován, jakmile se RIB považuje za konzistentní a stabilní. K provedení tohoto úkolu je potřeba, aby si kontrolní entita/program nejprve ručně vytvořili pohled na síťovou topologii, která vy-

hovuje určitým omezením. Tento pohled lze naprogramovat ručně, naučit pozorováním nebo může být složen z částí informací získaných komunikací s ostatními instancemi kontrolní vrstvy. [11]

Nedílnou součásti kontrolní vrstvy je SDN kontroler. Kontroler určuje tok (angl. flow) dat na datové vrstvě. Každý tok v síti musí nejprve získat povolení od kontroleru, který ověří, že komunikace probíhá dle pravidel sítě. Jestliže je tok povolen, je vypočítána cesta, po které data projdou, a je přidán záznam pro tento tok každému přepínači v cestě. Jelikož se o všechny složité funkce stará kontroler, přepínače jen jednoduše spravují tabulky toků (angl. flow tables), do kterých může přidat záznamy pouze kontroler. Komunikace mezi kontrolerem a přepínačem je dosažena za pomocí rozhraní. [12]

Podobně jako u A-CPI i rozhraní SDN kontroleru mezi datovou a kontrolní vrstvou se nazývá různě. V literatuře ONF se značí jako data-controller plane interface (D-CPI), southbound interface (SBI), případně southbound API. Pro tuto práci bylo vybráno označení D-CPI. Rozhraní slouží pro komunikaci datové vrstvy s SDN kontrolerem, který má několik funkčních komponent znázorněných na obrázku č. 3.



Obr. č. 3 Detail kontrolní logiky SDN. [8]

Komponenta **funkce kontroly datové vrstvy** (Data plane control function (dále jen DPCF)) efektivně vlastní zdroje, které má k dispozici a užívá je tak, jak má nařízeno od OSS / koordinátora nebo virtualizérů[1], které ji kontrolují.

[1] V architektuře SDN je virtualizace přidělení abstraktních zdrojů konkrétním klientům či aplikacím. Cílem je abstrahovat funkce sítě od vyhrazeného hardwaru, aby mohly být tyto funkce umístěny například na serverových platformách v cloudových datových centrech. [8]

Virtualizér je další důležitou komponentou. Lze ho považovat za proces, který přijímá specifické požadavky klienta přes A-CPI, ověřuje tyto požadavky na základě zdrojů přidělených klientovi, překládá požadavky do hlediska souvisejících zdrojů a předává výsledky na DPCF a D-CPI. Virtualizér a DPCF, a někdy i další funkce SDN kontroleru, musí spolupracovat, aby poskytovaly funkce jako je interpretace oznámení, sdílení zdrojů, implicitní služby poskytovatele a transakční integritu.

Model kontroler-agent je vhodný pro znázornění vztahu mezi ovládanou a ovládající entitou a vztahuje se rekurzivně na architekturu SDN. Ovládanou entitou je určen **agent**. Jde o funkční komponentu, která představuje zdroje a schopnosti klienta v prostředí serveru.

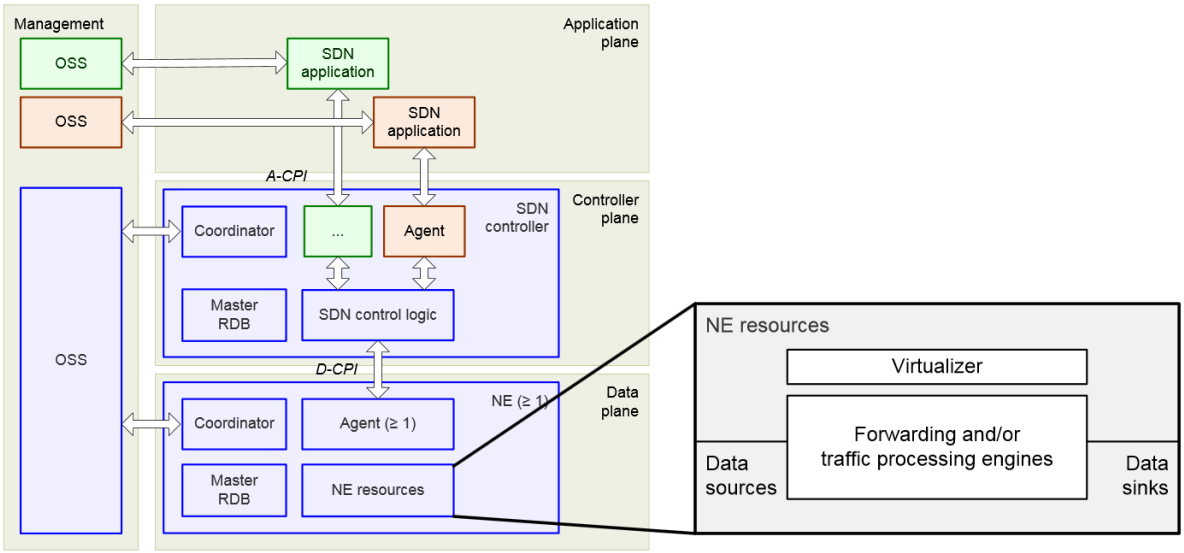
Databáze zdrojů (angl. resources database (RDB)) obsahuje aktuální instanci informačního modelu a potřebné podpůrné funkce. [8]

3.2.3 Datová vrstva

Datová vrstva je někdy označována také jako vrstva infrastruktury (infrastructure layer). Důvodem je skutečnost, že se zde nachází fyzické síťové prvky, které spolu tvoří topologii sítě – infrastrukturu.

Datová vrstva se stará o příchozí datagramy za pomocí série operací na úrovni spoje, které na datagramu provedou základní kontroly. Bezchybný datagram je zpracován kontrolní vrstvou prováděním náhledů do FIB tabulky (nebo tabulek), které byly dříve naprogramovány kontrolní vrstvou. Tento postup je někdy označován jako rychlá cesta pro zpracování paketu, protože nepotřebuje provádět žádné další operace, než jen zjistit cílovou destinaci paketu za pomocí naprogramované FIB. Výjimka nastává v případě, že paket neodpovídá pravidlům, například když je detekována neznámá cílová destinace. V tomto případě jsou pakety poslány na procesor cest, kde je kontrolní vrstva může dále zpracovávat pomocí RIB. [11]

Kromě rozhodování o směrování může datová vrstva také implementovat některé malé služby/funkce běžně nazývané jako směrovací funkce (například ACL a QoS). V některých systémech tyto funkce používají své vlastní diskrétní tabulky, zatímco ostatní představují rozšíření předávacích tabulek. [11]



Obr. č. 4 Detail zdrojů síťových prvků. [8]

Na obrázku č. 4 jsou zobrazeny důležité komponenty datové vrstvy. Blok **zdrojů síťových elementů** (angl. network element (NE) resources) obsahuje datové zdroje, přijímače dat, prostředky pro směrování a/nebo zpracování provozu v síti a **virtualizér**, jehož funkcí je abstrahovat zdroje na SDN kontroleru a prosazovat pravidla sítě. Další komponentou je **databáze zdrojů** (RDB), která na této vrstvě obsahuje všechny informace o zdrojích známých danému síťovému prvku. Funkcí **agenta** datové vrstvy je vykonávat na datové vrstvě instrukce přijaté od SDN kontroleru. **Koordinátor** datové vrstvy je entita, která rozděluje zdroje datové vrstvy různým agentům klienta a stanovuje pravidla pro jejich užití. [8]

3.3 OpenFlow

K převedení konceptu SDN do praktické realizace je potřeba splnit dvě podmínky. Nejprve musí být SDN kontrolerem řízena společná logická architektura ve všech přepínačích, směrovačích a v dalších síťových zařízeních. Tato logická architektura může být implementována různými způsoby na různých síťových zařízeních od různých výrobců, a to dokud ji kontroler vidí jako jednotnou logickou funkci. Poté je potřeba standard, zabezpečený protokol, který bude operovat mezi kontrolerem a síťovým zařízením.

Obě tyto podmínky splňuje OpenFlow, který je jak protokolem mezi kontrolerem a síťovými prvky, tak i specifikací logické struktury síťových funkcí. OpenFlow byl vyvinut na Stanfordské Univerzitě ve verzi 1.0. Verze 1.2 byla první verzí, kterou

vydala společnost Open Networking Foundation, která tento projekt převzala od Stanfordské Univerzity. [12]

V době psaní této práce byla k dispozici verze 1.5. Podrobnější rozpis verzí a změn, které přinesly, lze nalézt v [13].

OpenFlow je otevřený protokol založený na standardech, který definuje, jak může být kontrolní vrstva nakonfigurována a ovládána kontrolerem. Používáním OpenFlow může kontroler řídit tok paketů v síti.

V klasické síti mají přepínače a směrovače uloženy své informace v různých formátech (směrovací tabulka, Mac tabulka, atd.), které jsou vypočítány složitými protokoly nasazenými v celé síti. Na základě těchto tabulek je naprogramována datová vrstva. Protokol OpenFlow standardizuje jediný centralizovaný protokol, který dokáže vytvořit a spravovat tabulky toků a nahrazuje tím všechny ostatní směrovací tabulky. Datová vrstva je poté naprogramována na základě těchto tabulek. [14]

3.4 Problémy spojené s SDN

SDN přináší spousty výhod, avšak existují zde i problémy spojené s implementací a údržbou.

Prvním problémem je **nedostatek standardů** pro plnou kontrolu zařízení. Snahou přijít s novým softwarem pro stávající zařízení vznikly nejasnosti nad mírou obecnosti vytvoření síťových služeb. Otázkou v tomto problému bylo, zda by SDN mělo nabízet nastavení zařízení nebo pouze síťové operace. OpenFlow je částečným řešením tohoto problému, avšak operuje jen na rozhraní mezí kontrolní a datovou vrstvou. Pro rozhraní mezi kontrolní a aplikační vrstvou žádný standard neexistuje.

Další obtíží je **nedostatek softwaru** pro kontrolu služeb, což je technologie, která se stará o sestavování cest a řízení provozu v SDN síti. Každá implementace SDN vyžaduje tento software pro vytvoření virtuálních sítí. Uživatelé SDN musí spoléhat na poskytnutí tohoto softwaru od výrobce síťového vybavení. Správný software je důležitý pro podporu sítě se zařízeními od různých výrobců. [15]

Velký problém představují **útoky vedené na kontroler** v SDN architektuře otevřené neoprávněnému přístupu a využití. Mimo jiné, při nepřítomnosti robustní, zabezpečené platformy kontroleru je možné, aby se útočník vydával za kontroler a prováděl tak škodlivé aktivity. V minulosti byly takové útoky vedeny především na DNS servery. Při podobném útoku na kontroler by však mohly být napáchány pod-

statně větší škody. Bezpečnostní technologie jako je TLS (zabezpečení transportní vrstvy (angl. Transport Layer Security)) společně se vzájemným ověřováním kontrolerů a jejich přepínačů může zmírnit tyto hrozby, avšak v současné době jde jen o volitelnou funkci.

Dalším z možných útoků je DoS (odmítnutí služby (angl. Denial of Service)) útok. SDN obsahuje dvě možnosti zachování k novému toku v případě, že neexistuje žádná shoda v tabulce toků. Buď celý paket, nebo jen část jeho hlavičky je poslána kontroleru k vyřešení. Při velkém provozu v síti znamená poslání celého paketu na kontroler zabrání poměrně velké šířky pásma. Nicméně i kdyby byla poslána jen informace z hlavičky paketu, tak samotný paket musí být uložen v paměti uzlu, dokud se nevrátí příslušná položka z tabulky toků. V tomto případě by bylo pro útočníka jednoduché spustit DoS útok na uzlu vytvořením řady neznámých toků a přetížit tím tak paměť přepínače. [16]

3.5 Programovatelnost SDN

V programovatelné síti se o chování síťových zařízení a kontrolu toku stará software, který operuje nezávisle na hardwaru sítě. [17]

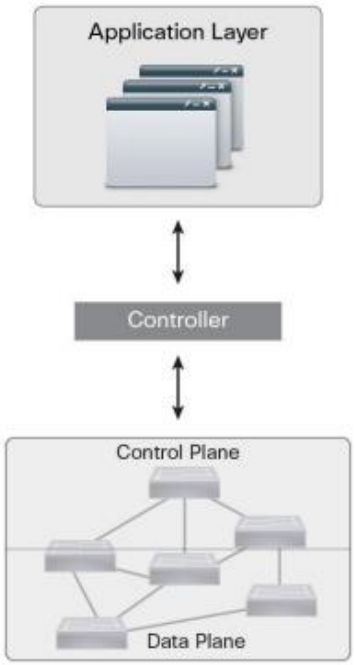
Dnešní síťová zařízení mají předinstalovaný software od výrobce, a tím se zužují možnosti zákazníka k vlastnímu nastavení. SDN architektura rozděluje tato zařízení na datovou a kontrolní vrstvu, čímž dělá kontrolní vrstvu programovatelnou a tím umožňuje abstrahovat síťové zařízení od aplikace a vrstev služeb, ke kterým se chová jako k virtuální entitě. [18]

Jak již bylo vysvětleno v dřívější kapitole, A-CPI slouží pro přístup aplikace na kontroler. Díky tomu je možné vytvořit a nasadit aplikaci, která bude napsaná v nějakém programovacím jazyce (například Java) a umožní uživateli schopnost upravovat a vynucovat si jisté chování od sítě v momentě, kdy tuto aplikaci kontroler načte. Pokud by byla aplikace schopna plně využívat A-CPI byl by potenciál takové aplikace značný. [19]

Kompletní separace kontrolní vrstvy od datové vrstvy je však radikální změnou oproti návrhu a běhu většiny dnešních sítí. V klasickém SDN přístupu k síťové architektuře představuje SDN výzvu, co se týče dostupnosti, výkonu, škálovatelnosti a bezpečnosti. Jelikož je veškerá funkcionalita přenesena na kontroler, tak musí být vysoce dostupný. Jeho výkon musí být dostatečný na to, aby zvládal rychlé požadavky

a extrémní zátěžové podmínky v síti. Kontroler musí být škálovatelný na velmi velkých sítích i s tisíci uzly. V neposlední řadě musí být také vysoce zabezpečený, kdy pouze schválené aplikace budou moci upravovat a měnit danou síť.

Existuje však tzv. hybridní model SDN (obrázek č. 5), který neseparuje kontrolní vrstvu od datové kompletně, ale jen částečně. Tento přístup umožňuje využívat mnoho výhod z klasického SDN modelu řízeného centrálním kontrolerem a zároveň dovolí uživatelům využívat stávající možnosti sítě, na které spoléhají. V mnoha případech stačí jen upgrade operačního systému sítě, namísto kompletní modernizace celé síťové infrastruktury. Dále tento přístup umožňuje zvolení si aplikací a funkcí, které se mají přenést na kontroler nebo aplikační server. Hybridní model SDN je schopen minimalizovat některé z potenciálních problémů, kterým čelí klasický SDN přístup k síťové architektuře. Právě tohoto přístupu využívá Cisco OnePK. [20]



Obr. č. 5 Hybridní model SDN. [20]

4 Cisco OnePK

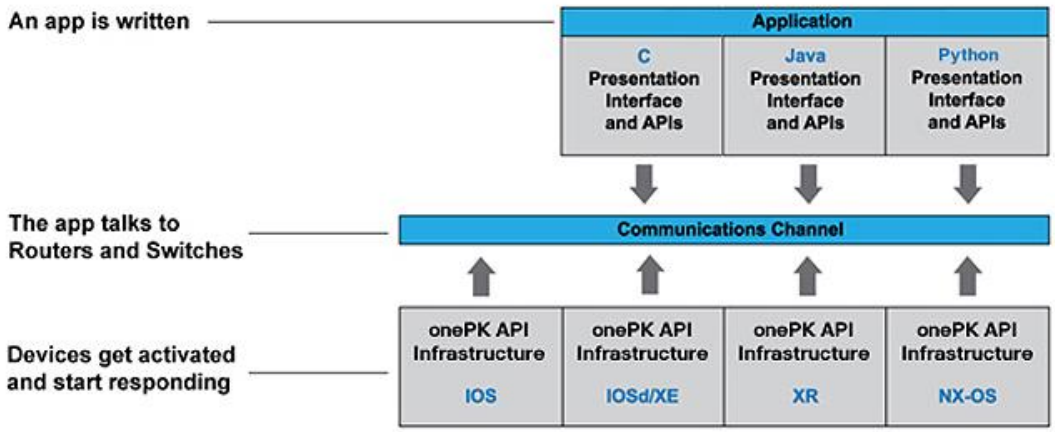
V této kapitole je popsána platforma Cisco OnePK, její funkce, nedostatky a zároveň jsou zde představeni dva nástupci této platformy: Cisco Open SDN Controller a APIC Enterprise Module.

4.1 Přehled OnePK

Cisco Open Network Environment [21] (ONE) je komplexním řešením pomáhající sítím být otevřenější, programovatelnější a lepší v detekci přítomnosti aplikací v síti. Široké možnosti Cisco ONE pomáhají uspokojit potřeby mnoha segmentů trhu včetně nových konceptů jako je SDN. Jedná se o portfolio doplňkových technologií, které zahrnuje multiplatformní API, technologie kontroleru a agenta a technologie překrývání sítí. Multiplatformní API jsou rovněž i součástí Software Development Kit (SDK), která se nazývá ONE Platform Kit (OnePK). [21, 22]

OnePK je komplexní sada multiplatformních API poskytující programový full-duplexní přístup na Cisco zařízení. Jedná se o sadu nástrojů umožňující vývojářům vytvářet vlastní aplikace, které interagují se síťovými prvky společnosti Cisco způsobem, který dříve nebyl možný. [22]

Architektura OnePK (obrázek č. 6) je tvořena třemi hlavními prvky: prezentační vrstvou, OnePK API infrastrukturou a komunikačním kanálem. Tyto tři prvky spolu tvoří konzistentní a adaptabilní architekturu umožňující různé programovací jazyky a modely nasazení pro aplikace fungující v síti. [23]



Obr. č. 6 Architektura OnePK. [22]

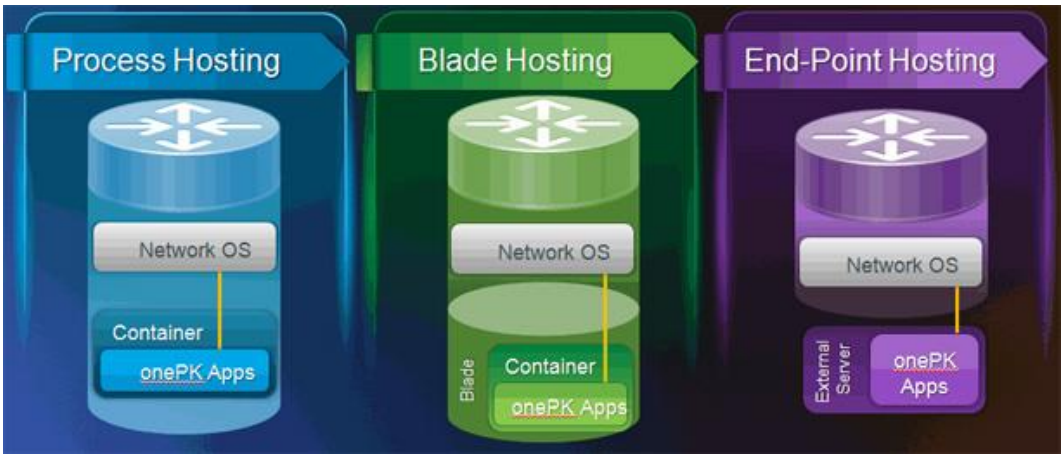
Prezentační vrstva se skládá z API knihoven, které mohou uživatelé využít pro svou aplikaci. Tyto knihovny jsou dostupné v jazycích C, Java a Python a byly na-

vrženy jen s několika závislostmi, takže je lze snadno integrovat stávajícími nástroji a požadavky vývoje.

OnePK API infrastruktura umožňuje přístup k funkcím dostupným na směrovači a přepínači. Jedna z jejich hlavních výhod je abstrahování rozdílů mezi operačními systémy a platformami. Například pokud aplikace uživatele použije funkci pro volání a čtení informací o interface v systému IOS, pak stejná funkce bude fungovat pro všechny zbylé operační systémy Cisco (IOSd/XE, XR a NX-OS).

Model komunikace poskytuje rychlý, bezpečný a rozšiřitelný kanál mezi aplikací a síťovým prvkem (aplikace musí být ověřeny předtím, než je jim dovoleno přistoupit k funkcím vrstvy API infrastruktury). [22]

Na obrázku č. 7 jsou představeny možnosti nasazení OnePK aplikace. Aplikace mohou operovat přímo na směrovači nebo přepínači, na blade serveru (jestliže ho dané zařízení podporuje), nebo na externím serveru nacházející se kdekoliv v síti. Volba modelu nasazení je dána možnostmi cílového zařízení a požadavky aplikace. [22]



Obr. č. 7 Typy hostingů OnePK. [22]

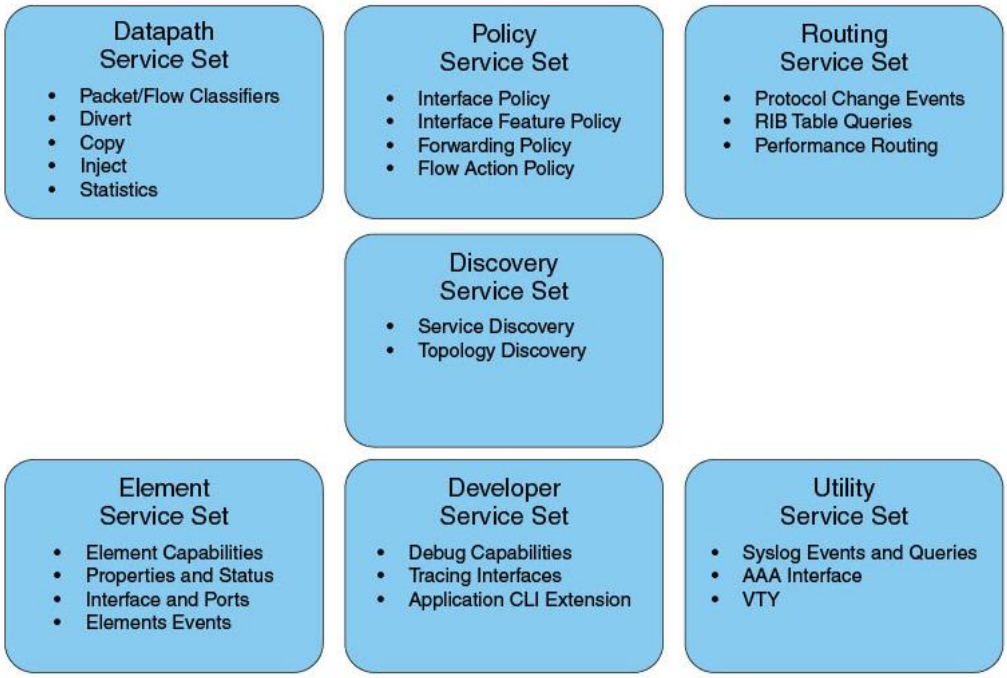
Při **Process Hostingu** běží aplikace přímo na síťovém prvku v Linuxovém prostředí izolovaném od procesů, spuštěných na procesoru daného prvku, pomocí kontejnerů.

Blade Hosting umožňuje aplikacím běžet blízko kontrolním a datovým vrstvám a vlastnit vyhrazené zdroje k provedení úloh.

End-Node Hosting umožňuje aplikacím využívat průmyslové platformy od velkých zařízení náročných na výpočet, jako jsou víceprocesorové/vícejádrové servery se systémem Linux nebo Windows až po mobilní zařízení se systémem Android nebo iOS. [22]

4.2 Funkce OnePK

Součástí OnePK jsou i service sety rozdělující jednotlivá API do skupin podle jejich funkcionality. Základních service setů je v současné době 7, jak ukazuje obrázek č. 8. [22, 23, 24]



Obr. č. 8 Základní service sety. [23]

Element service set se stará o vytvoření a správu spojení k síťovým prvkům. Ověřené spojení vytvoří relaci, která zpřístupní ostatní service sety. Dále obsahuje API starající se o získání a nastavení síťového zařízení, jeho vlastností, stavu a informací o jeho interface.

Datapath service set umožňuje uživatelům implementovat vlastní logiku zpracování paketů/toků. Tento service set je dostupný pouze pro jazyk C.

Policy service set dovoluje aplikacím nastavit několik pravidel pro směrování, včetně filtrování, ACL a QoS.

Routing service set poskytuje možnost čtení RIB a dovolí uživateli bezpečně modifikovat směrovací/přepínací logiku síťového prvku.

Utilities service set poskytuje přístup ke zprávám syslog[2] a k AAA (Authentication, authorization and accounting) funkcím síťového prvku a především umožňuje otevřít virtuální terminál (VTY) na síťovém prvku, ke kterému je aplikace připojena.

[2] Syslog je standard pro záznam programových zpráv. [25]

Discovery service set slouží aplikaci pro nalezení vzdálených nebo místních síťových prvků a síťové topologie.

Developer service set obsahuje několik dodatečných služeb pro vytváření aplikací a odladění těchto aplikací.

Identity service set se neřadí mezi základní service sety, neboť je podporován pouze na platformě IOS (ISR G2). Tento service set má na starosti přidávání, upravování načítání atributů a odstraňování relací.

Další service sety se nazývají rozšířené (angl. Extension service sets), které nejsou podporovány na všech platformách. Jedním z příkladů takového service setu je **Location**. Ten je dostupný jen na platformě IOS-XE (ASR1K) a slouží k získání či nastavení polohy síťového zařízení. Poloha může být znázorněna názvem města (například Hradec Králové), geologickou pozicí (zeměpisná šířka, délka, nadmořská výška) nebo vlastním názvem (například J7).

Dalším příkladem je **Mediatrace** service set dostupný pouze na platformě IOS (ISR G2) a umožňuje nastavit cestu pro kontrolu systémového profilu a zobrazení výsledků. [22, 23, 24]

4.3 Zabezpečení OnePK

OnePK poskytuje přístup k síťovému zařízení, proto je důležité zabezpečení vytvořených aplikací. OnePK přistupuje k směrovacím a přepínacím mechanismům síťového prvku přes mechanismus zabezpečené meziprocesové komunikace (angl. inter-process communication (IPC)). IPC zajišťuje, že aplikace nejsou v konfliktu se stávajícím směrováním a přepínáním, protože nedochází k přímému přístupu do jádra funkcionality síťového prvku.

Vnitřní IPC mechanismus chrání aplikace jak z náhodného, tak i úmyslného přetečení bufferu. Jakýkoli IPC přístup pocházející z koncového zařízení, na kterém je OnePK aplikace, je ověřen a šifrován pomocí kryptografického protokolu TLS (Transport Layer Security).

Mimo tento typ zabezpečení má správce sítě možnost sledovat všechny aplikace běžící na síťových zařízeních a libovolnou z nich vypnout, jestliže se nechová dle předpokladů. [26]

4.4 Nedostatky OnePK

OnePK obsahuje mnoho funkcí, ale mnoho jich není implementováno, alespoň ne v současné době. Například pro dynamické směrování neexistuje žádná funkce. Některé změny v konfiguraci síťového prvku jsou aktivní, pouze pokud je aplikace připojena, po odpojení aplikace se tyto změny ztratí. Jedním z příkladů je nastavení statické cesty na směrovači pomocí OnePK aplikace, která bude ve směrovací tabulce označena písmenem „a“, což znamená, že cesta byla naučena aplikací. Tato informace se ihned po odpojení aplikace ztratí, funkce statického směrování je proto značně omezena. Většinu těchto problémů řeší VTY service set, pomocí kterého lze implementovat libovolnou funkci směrovače nebo přepínače.

V prostředí OnePK neexistuje žádná funkce pro zjištění síťových prvků, ke kterým se může aplikace připojit, aniž by se připojila alespoň na jeden síťový prvek. Pro připojení na síťový prvek je nutné znát IP adresu interface, přes který se aplikace připojí, nebo úplný název daného prvku v síti.

4.5 Dostupnost OnePK

Během psaní této práce došlo k neočekávanému oznámení o ukončení podpory pro OnePK. Celé prohlášení včetně posledních operačních systémů podporujících OnePK lze nalézt na [27]. Cisco však odkazuje na své technologie další generace, jejichž přehled lze nalézt na [28]. Na této webové stránce jsou zmíněny i Cisco Open SDN Controller [29] a APIC Enterprise Module [30], které jsou blíže představeny v následujících kapitolách této práce.

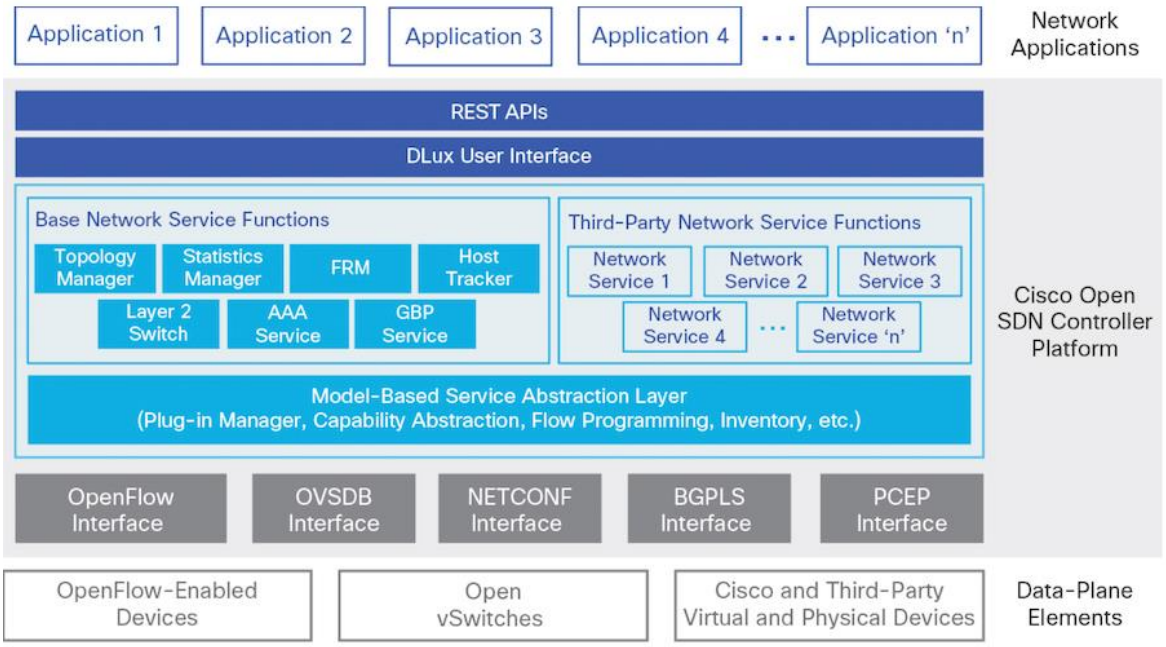
4.6 Cisco Open SDN Controller

Cisco Open SDN Controller (dále jen OSC) je komerční distribucí OpenDaylight[3] umožňující rychlou adaptaci na změny prostřednictvím automatizace síťové infrastruktury založené na standardech. Abstrahuje složitost správy heterogenních síťových prostředí s cílem zlepšit poskytování služeb a snížit provozní náklady.

Jedná se o open-source software, což znamená neustálé vyvíjení spolu s probíhající inovací a podporou OpenDaylight komunity. [31]

[3] OpenDaylight je nadace s cílem urychlit vývoj technologií dostupných uživatelům, umožnit širší přijímání SDN a vytvořit pevný základ pro NFV (Network Functions Virtualization). [32]

Obrázek č. 9 popisuje, jakým způsobem je OSC vytvořeno a zároveň nastiňuje některé jeho možnosti, jako je například podpora OpenFlow. Další funkce, možnosti, podporované platformy lze nalézt na [31].



Obr. č. 9 Platforma Cisco Open SDN Controller. [31]

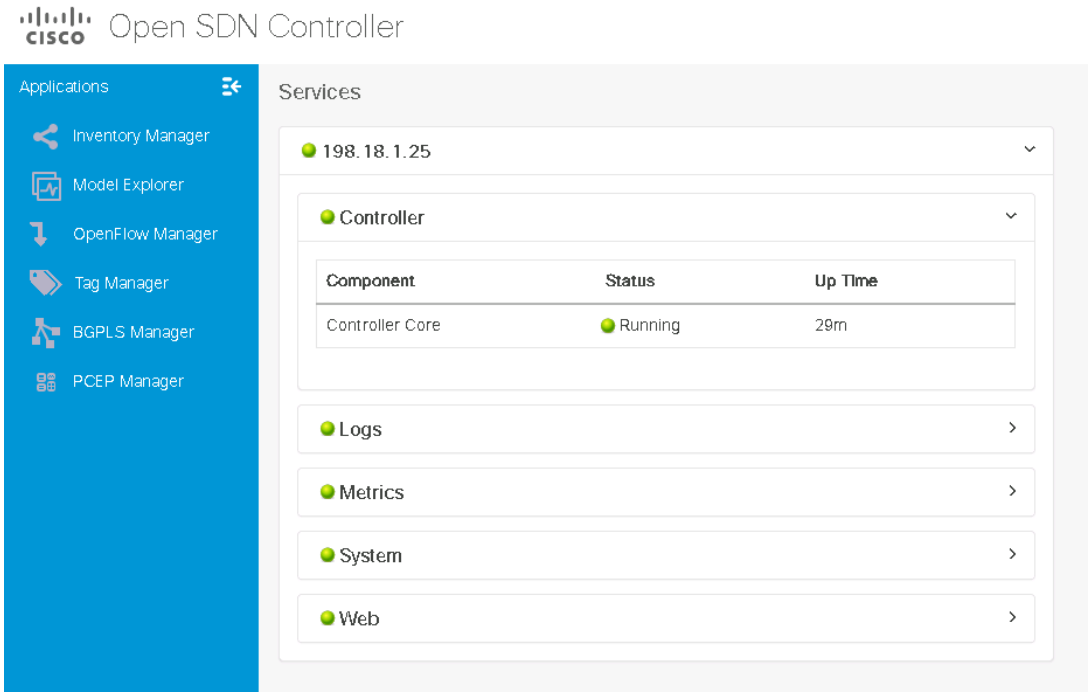
4.6.1 Vyzkoušení Cisco Open SDN Controller

OSC je možné si vyzkoušet přímo na webových stránkách společnosti Cisco. Vytvořením účtu na stránkách Cisco DevNet [33] bude zájemci umožněn přístup do cloudového prostředí dCloud [34] společnosti Cisco. V prostředí je možnost zažádat si o vytvoření relace pro OSC, po zadání konkrétního důvodu (například trénink, učení se apod.) se vytvoří relace, ke které je možné přistoupit ze svého účtu. Prostředí umožňuje výběr z několika relací podle verze OSC, případně typu ukázkového příkladu obsaženého v relaci. Vše probíhá v internetovém prohlížeči, není nutné nic stahovat a nároky na hardware počítače jsou minimální.

Po vytvoření relace je možné k ní přistoupit. K dispozici je základní topologie, dokumentace a další informace o relaci, například uživatelé využívající tuto relaci. Relaci je možné sdílet s dalšími uživateli, a tak testování OSC může probíhat i v týmech.

Přistoupením na vzdálenou plochu počítače v topologii se otevře možnost konfigurace kontroleru, spouštění a zastavování různých služeb a je k dispozici další dokumentace ke konkrétní ukázce ve vybrané relaci, kde jsou detailně popsány funkce a možnosti ovládání dané ukázky.

Na obrázku č. 10 je zobrazen základní pohled na OSC po vytvoření relace a přistoupení k ní.

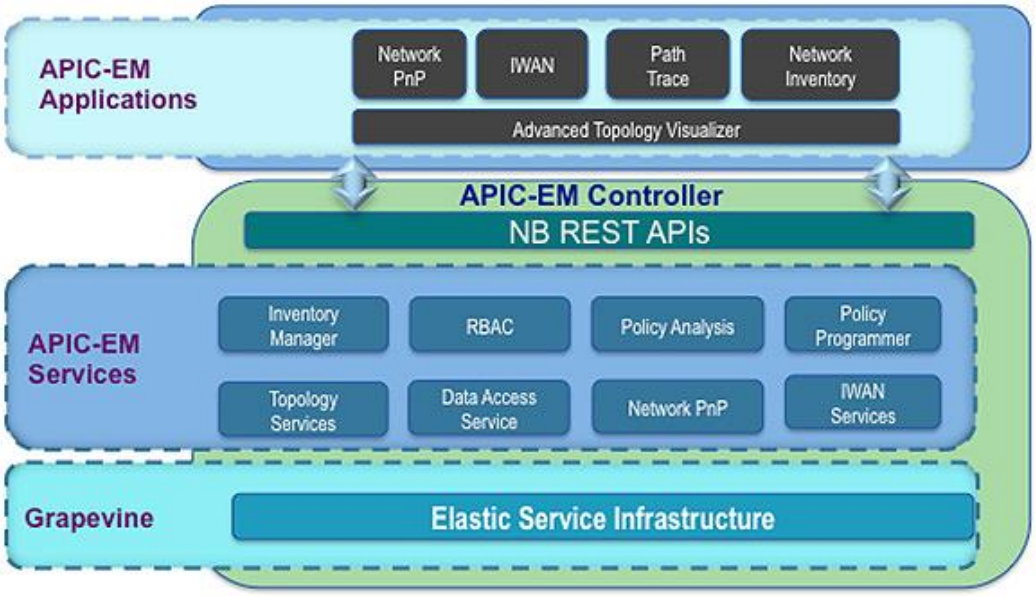


Obr. č. 10 Cisco Open SDN Controller.

OSC lze stáhnout na [35], jestliže je uživatel součástí společnosti, která je partnerem společnosti Cisco.

4.7 APIC Enterprise Module

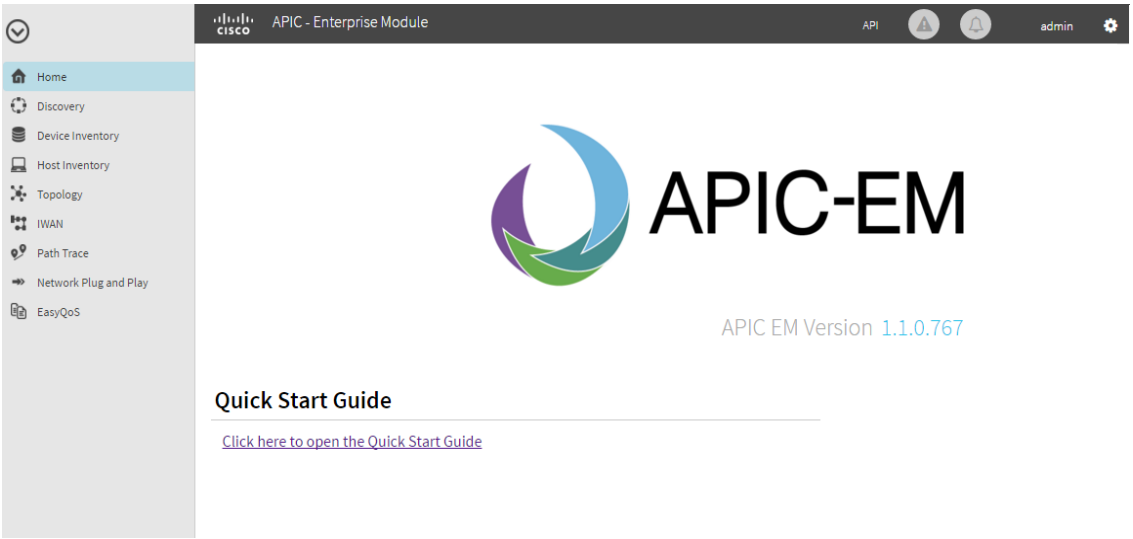
Application Policy Infrastructure Controller Enterprise Module (dále jen APIC-EM) je SDN kontroler společnosti Cisco pro podnikové sítě. Na northbound interface tohoto kontroleru může aplikace přistoupit pomocí REST API. Southbound interface není aplikacím přístupný, ale je využit pro vykonání příkazů zadáné přes REST API. APIC-EM umožňuje uživatelům vytvořit si vlastní kontroler nebo přidat dynamickou SDN funkcionalitu přímo do vlastních aplikací. Na obrázku č. 11 je znázorněna vnitřní struktura APIC-EM. Podrobnou dokumentaci lze nalézt na [37]. [36]



Obr. č. 11 Vnitřní struktura APIC-EM. [36]

4.7.1 Vyzkoušení APIC Enterprise Module

Stejně jako OSC, i APIC-EM lze vyzkoušet přímo na webových stránkách společnosti Cisco. APIC-EM se nenachází na dCloud, ale na tzv. sandboxových laboratořích, které jsou přístupné pouze přihlášeným uživatelům (také DevNet účet). Po přihlášení lze tyto sandboxové laboratoře nalézt na [38]. Jedna z laboratoří s názvem „APIC-EM__GA_DB-Only_Always-On“ je dostupná okamžitě a bez rezervace. Na obrázku č. 12 je ukázána hlavní stránka APIC-EM po přistoupení na výše uvedenou sandboxovou laboratoř a přihlášení se na samotný APIC-EM.



Obr. č. 12 APIC Enterprise Module.

Samotný APIC-EM je volně ke stažení pro přihlášené uživatele na stránkách společnosti Cisco v sekci DevNet.

5 Vývoj aplikace

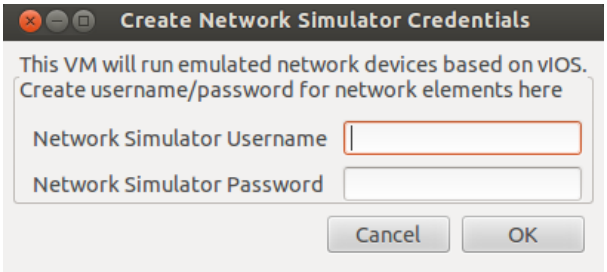
Tato kapitola se zabývá samotným vývojem aplikace. Nejprve je předvedeno vývojové prostředí, ve kterém byla vytvořena vlastní aplikace. Dále jsou popsány funkce aplikace, včetně ukázek kódu, testování a jeho výsledky.

5.1 All-in-one Virtual Machine

Prvním krokem je stažení prostředí OnePK. Cisco mělo na svých webových stránkách k dispozici balíčky v podobě virtuálních strojů s operačním systémem Linux, konkrétně distribuci Ubuntu. K těmto balíčkům bylo možné získat přístup registrací na Cisco DevNet [33], nicméně již bylo zmíněno v kapitole 3.4, že podpora OnePK byla během psaní této práce ukončena a k těmto balíčkům již nelze získat přístup, alespoň dle zkušeností autora. V této práci byla použita verze OnePK 1.2.0-173, z důvodu zálohy této verze při dřívějším testování. Nejnovější verzí je 1.3, nicméně tuto verzi již nebylo možné získat.

Po stažení balíčku s koncovkou „ova“ (Open Virtualization Archive), o velikosti přibližně 2,5GB, je nutné vlastnit program na jeho otevření. Pro tuto práci byl použit VMware Player ve verzi 7.1.2, který je volně ke stažení pro nekomerční účely na stránkách společnosti VMware [39].

Po naimportování virtuálního stroje se spustí Ubuntu a vyzve uživatele k zadání hesla. Heslo je „cisco123“ a po jeho zadání systém uživatele vyzve k jeho změně. Po změně hesla, přečtení a odsouhlasení licence se objeví pracovní plocha. V této fázi systém uživatele vyzve k vyplnění údajů pro přihlášení na virtuální směrovače (tzv. credentials), jak znázorňuje obrázek č. 13. Tyto údaje slouží pro vytvoření uživatele na síťových prvcích ve virtuální topologii, což nahrazuje příkaz „username *username* priviledge 15 password 0 *password*“, který by jinak bylo nutné zadat na síťovém prvku.

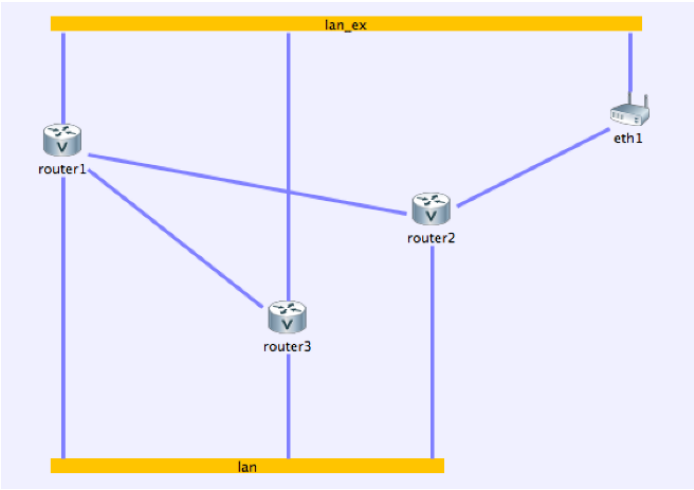


Obr. č. 13 Network Simulator Credentials.

Na obrázku č. 14 je ukázka plochy, která obsahuje vše potřebné k úspěšnému vytvoření aplikace. Plocha obsahuje návody na vytvoření virtuální topologie, dokumentace OnePK, nástroje pro správu předpřipravené virtuální sítě a vývojové prostředí Eclipse obsahující ukázkové aplikace.

Obr. č. 14 Náhled plochy.

Prvním krokem při vývoji aplikace je spuštění virtuální sítě, což lze jednoduše provést kliknutím na ikonu „Start 3node“. Vytvoří se tři virtuální směrovače, ke kterým je možné se připojit a provádět na nich funkce vytvářené aplikace. Zmíněnou topologii znázorňuje obrázek č. 15. Tento obrázek lze nalézt ve formátu pdf v umístění „/usr/share/vmcloud/data/examples/3node/“ na virtuálním stroji. V tomto umístění se nachází i konfigurační soubory jednotlivých směrovačů a soubor pro nastavení virtuální topologie, kde lze přidat, či odebrat síťové prvky. Návod na úpravu virtuální topologie se nachází na ploše ve složce onePK-Documentation, konkrétně soubor „EmulatorUserGuide.pdf“. Pro účely vytvořené aplikace nebyla tato topologie upravována.



Obr. č. 15 Předpřipravená virtuální topologie.

Směrovače připravené k použití vyžadují zadání několika příkazů. Prvním příkazem je *„transport type tls disable-remotecert-validation“*. Bez zadání tohoto příkazu se aplikace nebude moci připojit ke směrovači. Jak již bylo zmíněno v předchozí kapitole, OnePK využívá pro zabezpečení kryptografický protokol TLS, zadáním tohoto příkazu se na směrovači povolí TLS a zároveň se nastaví používání jednosměrné autentifikace.

Dalším příkazem je *„service set vty“*, který povolí aplikaci zasílat příkazy přes VTY. Díky tomuto příkazu je možné naprogramovat funkce, které OnePK zatím neumí, jako je například dynamické směrování.

Všechny příkazy se zadávají v módu *„router(config-onep)#“* do kterého se lze dostat příkazem *„onep“* v konfiguračním módu směrovače.

Po tomto kroku je možné začít programovat aplikaci. K dispozici jsou jazyky C, Java a Python. V této bakalářské práci byl vybrán programovací jazyk Java z důvodu autorovi znalosti tohoto jazyka. Jak již bylo zmíněno, na virtuálním stroji je nainstalováno vývojové prostředí Eclipse, toto prostředí bylo využito při programování aplikace.

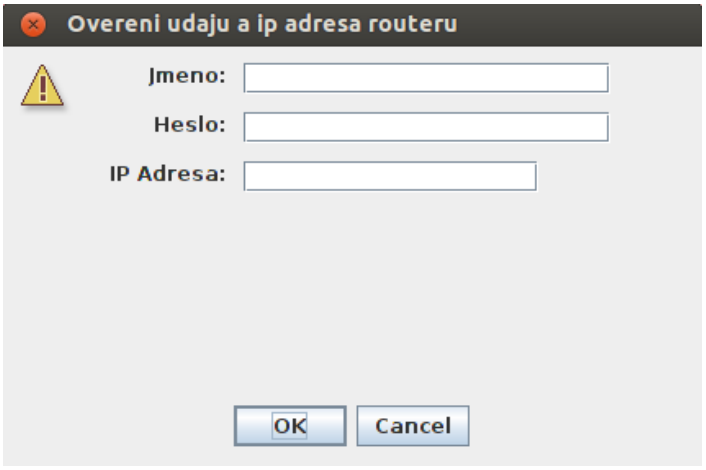
5.2 Funkce vytvořené aplikace

Ihned po spuštění aplikace se zobrazí okno s informacemi o jejích funkcích, požadavcích na příkazy směrovače a tlačítkem pro připojení se k příslušenému směrovači, jak ukazuje obrázek č. 16.

Obr. č. 16 Hlavní okno vytvořené OnePK Aplikace.

5.2.1 Připojení ke směrovači

První funkcí je připojení ke směrovači. Po stisknutí tlačítka „Pripojit k routeru“ se zobrazí okno pro zadání údajů, jak je znázorněno na obrázku č. 17.

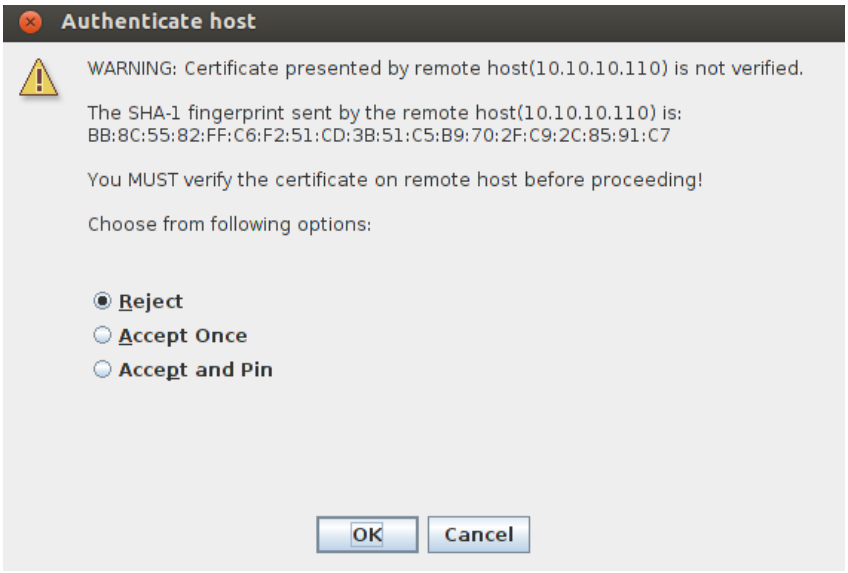


Obr. č. 17 Zadání údajů pro připojení ke směrovači.

Aplikace se nejprve pokusí získat, na základě zadané IP adresy, příslušný síťový prvek a poté se pokusí vytvořit novou relaci připojením na tento prvek. V ukázce jsou pouze metody pro získání prvku a připojení se k němu. Pro funkčnost musí být tyto metody obaleny příkazy try-catch, protože může nastat situace, kdy uživatel zadá například neplatnou IP adresu, nebo se na daný prvek nebude možné připojit, třeba z důvodu nepovoleného TLS protokolu. Toto platí pro většinu použitých metod.

```
private String username;
private String password;
private String ipaddress;
private NetworkElement networkElement;
private SessionHandle sessionHandle;

networkElement = networkApplication.getNetworkElement(InetAddress
                              .getByName(ipaddress));
SessionConfig sessionConf = createSessionConfig();
sessionHandle = networkElement.connect(username, password,
                              sessionConf);
```

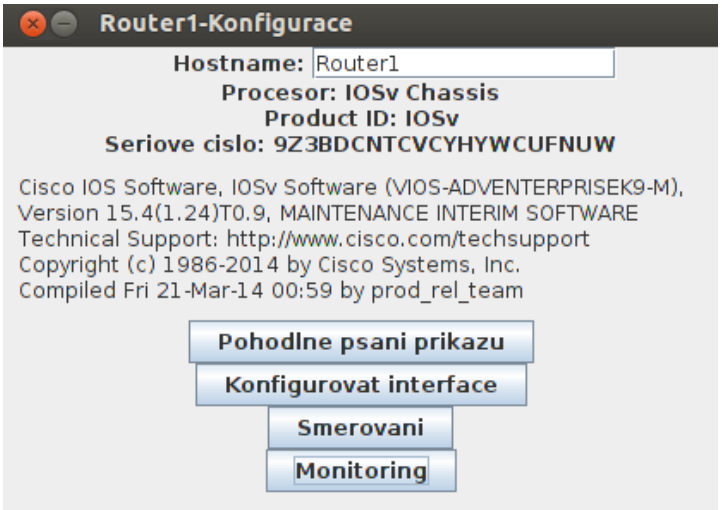
Po zadání správného přihlašovacího jména, hesla a IP adresy interface směrovače, ke kterému se aplikace připojuje, se zobrazí okno pro ověření certifikátu TLS. Ověřovací okno lze vidět na obrázku č. 18.



Obr. č. 18 Ověření certifikátu.

K dispozici jsou tři možnosti: odmítnout (Reject), pro teď přijmout (Accept Once) a přijmout a uložit (Accept and Pin). Při výběru první možnosti se aplikace na směrovač nepřipojí. Při výběru druhé možnosti se aplikace na směrovač připojí, ale při dalším připojení na tento směrovač bude opět nutné ověřit certifikát. Při výběru třetí možnosti se tato volba uloží a při opětovném připojení na daný směrovač již nebude vyžadováno ověřování certifikátu TLS. Tato funkce je přímo převzata z ukázkových příkladů dostupných na virtuálním stroji.

Je-li certifikát přijat, zobrazí se okno s informacemi a možnostmi konfigurace směrovače, které je na obrázku č. 19.



Obr. č. 19 Možnosti konfigurace směrovače.

V okně se nachází první funkce, která využívá VTY, a to změna názvu směrovače. Tuto funkci, včetně otevření a uzavření VTY služby, předvádí následující ukázka.

```
VtyService vtyService = new VtyService(element);
vtyService.open();
vtyService.write("conf t");
//Zmeni hostname
if(cmd == "name"){
      vtyService.write("hostname "+txtName.getText());
      hostname = txtName.getText();
      setTitle(hostname+"-Konfigurace");
      }
int maxResponseLength = 110;
vtyService.setMaxResponse(maxResponseLength);
maxResponseLength = 0;
vtyService.setMaxResponse(maxResponseLength);

vtyService.cancel();
vtyService.close();
vtyService.destroy();
```

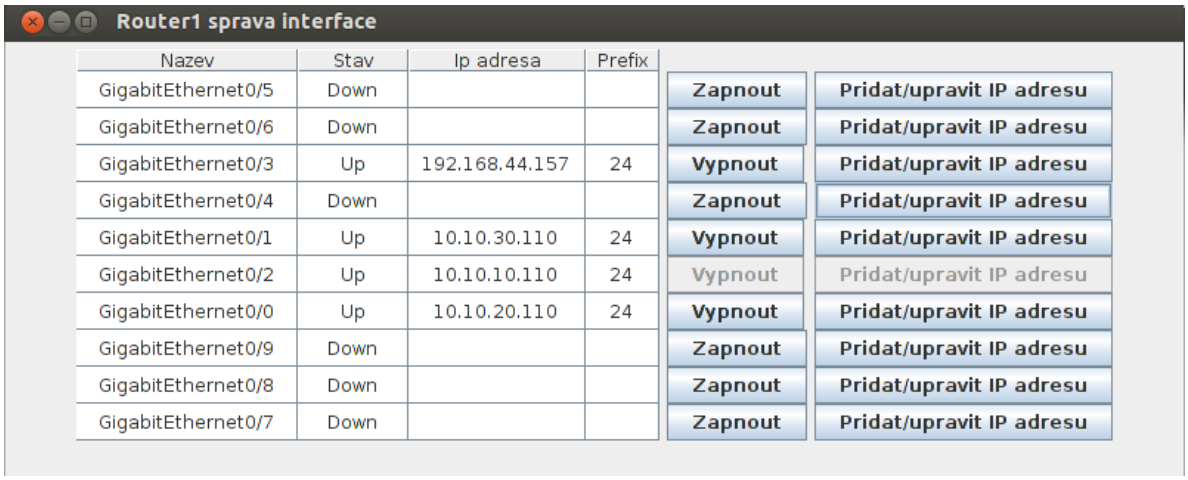
Funkce tlačítka „Pohodlne psani prikazu“ slouží pro zaslání skupiny příkazů, které směrovači zabrání v zasílání zpráv během psaní příkazů, když nastane změna, o které směrovač informuje, například aktivace interface.

```
vtyService.write("no ip domain-lookup");
vtyService.write("line con 0");
vtyService.write("logg syn");
```

Tato funkce původně byla pouze pro ověření funkčnosti VTY. Nakonec však byla ponechána, neboť byla shledána užitečnou.

5.2.2 Správa interface

Po stisknutí tlačítka „konfigurovat interface“ se zobrazí okno s tabulkou interface daného síťového prvku a tlačítky pro jejich ovládání. Uživatel může vypnout či zapnout daný interface a přidat, upravit nebo smazat IP adresu příslušnému interface (obrázek č. 20).



Obr. č. 20 Správa interface.

Pro získání všech interface daného síťového prvku byla použita tato metoda.

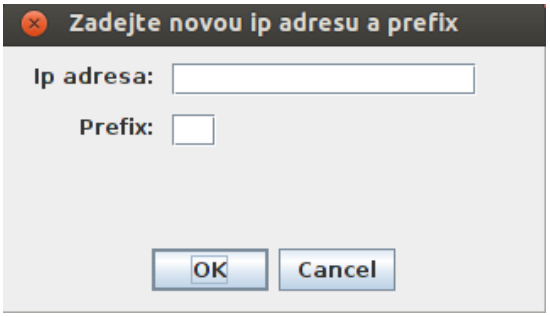
```
interfaceList = netElement.getInterfaceList(new InterfaceFilter());
```

Instanci třídy „InterfaceFilter“ můžeme přidat parametr, například „NetworkInterface.Type.*ONEP_IF_TYPE_ETHERNET*“. Tímto parametrem se nastaví filtrování interface pouze na typ ethernet. Pokud zůstane parametr nevyplněný, metoda získá všechny interface daného směrovače.

Kliknutím na tlačítko „Vypnout“ případně „Zapnout“ se provede následující metoda, která uvede daný interface do stavu „DOWN“ nebo „UP“. Proměnná „state“ nabývá hodnot true nebo false, podle stavu interface.

```
boolean state = false;
if (networkInterfaces.get(i).getStatus().getLinkState()
      .toString() == "ONEP_IF_STATE_OPER_UP") {
      state = true;
}
networkInterfaces.get(i).shutdown(state);
```

Stiskem tlačítka „Pridat/upravit IP adresu“ se zobrazí následující okno (obrázek č. 21).



Obr. č. 21 Přidaní IP adresy pro interface.

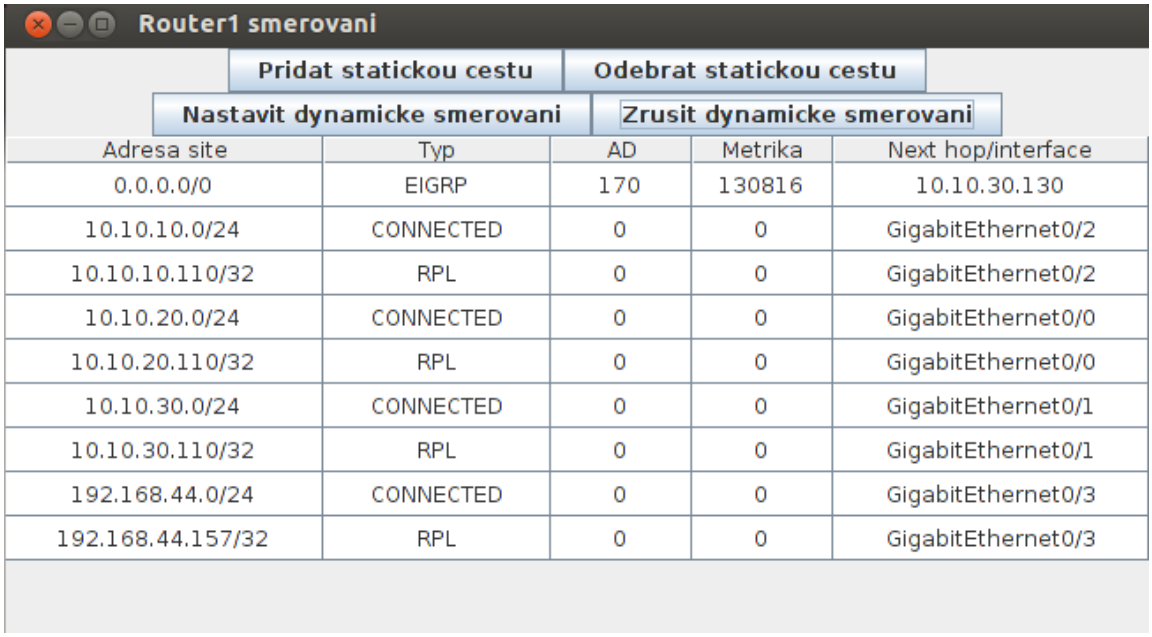
Pokud daný interface dosud nemá přiřazenou IP adresu, textová pole budou prázdná, v opačném případě budou vyplněna příslušnou IP adresou a prefixem. Pro přiřazení IP adresy musí být korektně vyplněna obě textová pole. Pro smazání IP adresy stačí vymazat jedno z textových polí a adresa tohoto interface se odebere. Funkce umí přiřazovat pouze IP adresu verze 4 (IPv4).

Následující ukázka obsahuje metody pro přidání nebo úpravu IP adresy a její smazání na daném interface, kde „ni“ znamená příslušný interface (instance třídy „NetworkInterface“).

```
ni.updateAddress(OnepAddressScopeType.ONEP_ADDRESS_IPv4_PRIMARY,
                                    ipAddress, prefixLen);
ni.deleteAddress(OnepAddressScopeType.ONEP_ADDRESS_IPv4_PRIMARY, ni
                                .getPrefixList().get(0).getAddress(), ni

      .getPrefixList().get(0).getPrefixLength());
```

5.2.3 Směrovací tabulka

Stisknutím tlačítka „Smerovani“ se zobrazí okno ukázané na obrázku č. 22. V okně je zobrazena směrovací tabulka spolu s několika tlačítky pro správu směrování.



Adresa site	Typ	AD	Metrika	Next hop/interface
0.0.0.0/0	EIGRP	170	130816	10.10.30.130
10.10.10.0/24	CONNECTED	0	0	GigabitEthernet0/2
10.10.10.110/32	RPL	0	0	GigabitEthernet0/2
10.10.20.0/24	CONNECTED	0	0	GigabitEthernet0/0
10.10.20.110/32	RPL	0	0	GigabitEthernet0/0
10.10.30.0/24	CONNECTED	0	0	GigabitEthernet0/1
10.10.30.110/32	RPL	0	0	GigabitEthernet0/1
192.168.44.0/24	CONNECTED	0	0	GigabitEthernet0/3
192.168.44.157/32	RPL	0	0	GigabitEthernet0/3

Obr. č. 22 Směrovací tabulka.

Následující metody slouží pro získání směrovací tabulky.

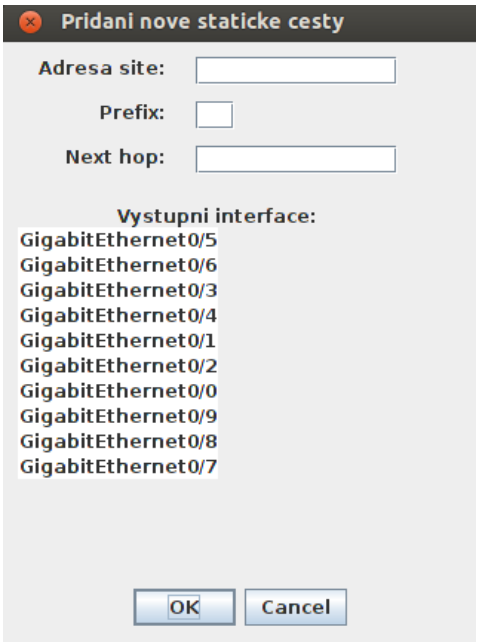
```
private List<Route> routeList;
private NetworkElement netElement;
private Routing routing;
private RIB rib;

routing = Routing.getInstance(netElement);
rib = routing.getRib();
L3UnicastScope aL3UnicastScope = new L3UnicastScope("",
                  AFIType.IPV4, SAFIType.UNICAST, "");
L3UnicastRIBFilter filter = new L3UnicastRIBFilter();
NetworkPrefix networkPrefix = new NetworkPrefix(
                  InetAddress.getByName("0.0.0.0"), 0);
L3UnicastRouteRange range = new L3UnicastRouteRange(networkPrefix,
                  RouteRange.RangeType.EQUAL_OR_LARGER, 0);
routeList = rib.getRouteList(aL3UnicastScope, filter, range);
```

Nejprve se získá RIB, poté se nastaví filtry. V tomto případě jsou parametry instance třídy „L3UnicastScope“ nastaveny na „AFIType.IPV4“ a SAFTType.UNICAST“

což zaručuje zahrnutí pouze Unicast[4] cest s IP adresou verze 4. Instance třídy „L3UnicastRIBFilter“ byla ponechána bez parametrů, aby byly zahrnuty všechny typy záznamů. Této instanci je možné nastavit parametr „OwnerType“, který určuje, o jaký typ záznamu se jedná, například přímo připojená síť („LOCAL“) nebo dynamicky naučená cesta přes protokol OSPF a podobně. Dalším filtrem je instance třídy „NetworkPrefix“. Tento filtr určuje rozsah IP adresy nebo prefixu, případně obojího. Této instanci byly nastaveny parametry IP adresy „0.0.0.0“ a prefixu „0“. Takto nastavené parametry zaručují zahrnutí veškerých IP adres sítí ve směrovací tabulce. Posledním parametrem je instance třídy „L3UniceastRouteRange“ určující počet záznamů. Nastavením posledního parametru této instance na hodnotu „0“ je zajištěno zahrnutí všech záznamů. Pomocí takto nastavených filtrů je možné získat seznam cest, podobně jako při zadání příkazu *„show ip route“* na směrovači.

5.2.4 Statické směrování

Stisknutím tlačítka „Pridat statickou cestu“ se otevře okno pro zadání parametrů statické cesty, které lze vidět na obrázku č. 23.



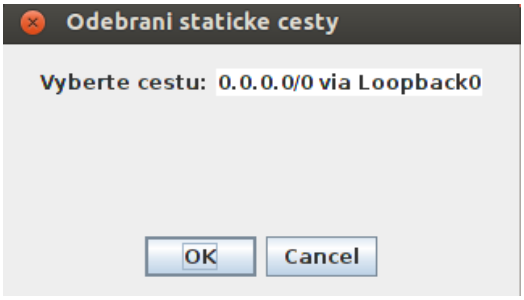
Obr. č. 23 Přidání statické cesty.

Funkce byla implementována pomocí VTY, přestože OnePK obsahuje metody pro přidání statické cesty směrovači. Nicméně cesta přidaná těmito metodami bude ve směrovací tabulce označena jako *„a*“* což znamená, že byla přidána aplikací. In-

[4] Unicast označuje zaslání paketů jedinému cíli. [40]

formace o takové cestě je ihned po odpojení aplikace ztracena, což je nežádoucí. Statické cesty přidané pomocí VTY jsou stejné, jako při zadání příkazů přímo na směrovači. I po odpojení aplikace jsou informace o těchto cestách na směrovači zachovány. Statická cesta vyžaduje tři parametry: IP adresu, prefix a výstupní interface nebo adresa následujícího skoku[5] (angl. next hop). Je-li zadána adresa následujícího skoku a zároveň výstupní interface, bude upřednostněna adresa následujícího skoku.

Pro odebrání statické cesty slouží tlačítko „Odebrat statickou cestu“, po jehož stisknutí se zobrazí okno se seznamem statických cest (obrázek č. 24).

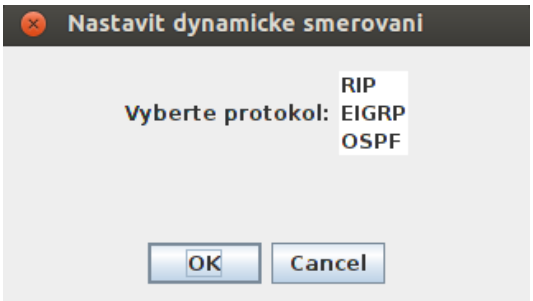


Obr. č. 24 Odebrání statické cesty.

5.2.5 Dynamické směrování

Dynamické směrování není ve OnePK zatím implementováno, proto bylo využito přístupu pomocí VTY. Všechny dynamické protokoly byly implementovány pouze v jejich základní formě a pro IP adresy verze 4.

Po stisknutí tlačítka „Nastavit dynamicke smerovani“ aplikace vyzve uživatele k vybrání dynamického směrovacího protokolu (obrázek č. 25). Implementovány byly protokoly RIP (Routing Information Protocol), EIGRP (Enhanced Interior Gateway Routing Protocol) a OSPF (Open Shortest Path First).



Obr. č. 25 Výběr dynamického protokolu.

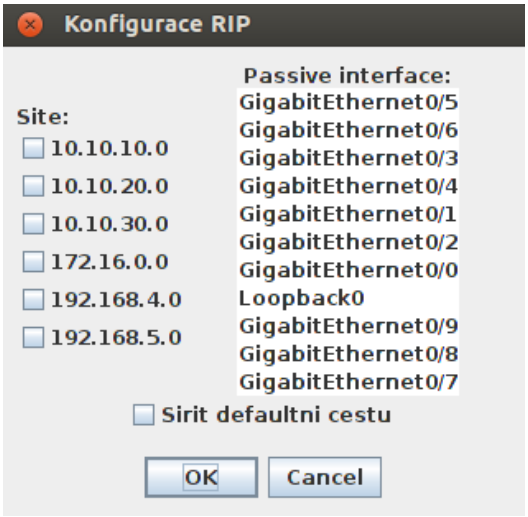
Výběrem a potvrzením protokolu je uživatel aplikací dále vyzván k vyplnění parametrů potřebných pro daný protokol. Stejné okno, jen s odlišným názvem, se

[5] Adresa následujícího skoku označuje IP adresu interface směrovače, ke kterému cesta směřuje. [41]

zobrazí také po stisku tlačítka „Zrusit dynamicke smerovani“, kde se po výběru příslušného protokolu provede příkaz *„no router rip/eigrp/ospf“* na základě vybraného protokolu. Je-li vybrán protokol EIGRP nebo OSPF, uživatel je dále vyzván k zadání čísla autonomního systému, případně ID procesu.

5.2.5.1 Protokol RIP

Pokud si uživatel zvolil protokol RIP, zobrazí se okno ukázané na obrázku č. 26.



Obr. č. 26 Nastavení protokolu RIP.

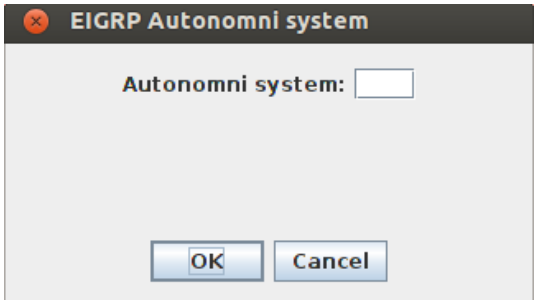
Zaškrtnutím jednoho nebo více políček ze seznamu sítí se po potvrzení konfigurace tlačítkem „OK“ příslušné adresy přidají do seznamu adres, o kterých bude směrovač informovat ostatní směrovače pomocí protokolu RIP. Zaškrtnutím políčka „Sireni defaultni cesty“ se provede příkaz *„default-information originate“*, který nastaví směrovač, aby šířil informaci o defaultní cestě[6]. Posledním parametrem je vybrání pasivního interface[7] (angl. passive interface). Implementace funkce je zobrazena v následující ukázce.

```
vtyService.write("router rip");
for (int i = 0; i < chNets.size(); i++)
      if(chNets.get(i).isSelected())
            vtyService.write("network " + chNets.get(i).getLabel());
if (defChb.isSelected())
      vtyService.write("default-information originate");
for (int i = 0; i < jInterfaces.getSelectedIndices().length; i++)
      vtyService.write("passive-interface " +
            Interfaces.getSelectedValues()[i]);
```

[6] Defaultní cesta je cesta s IP adresou 0.0.0.0 a prefixem 0. Tuto cestu směrovač zvolí, pokud nenalezne žádnou jinou shodu ve směrovací tabulce s cílovou IP adresou paketu. [42]
[7] Pasivní interface je interface, který dynamický protokol nebude využívat k šíření informací. [43]

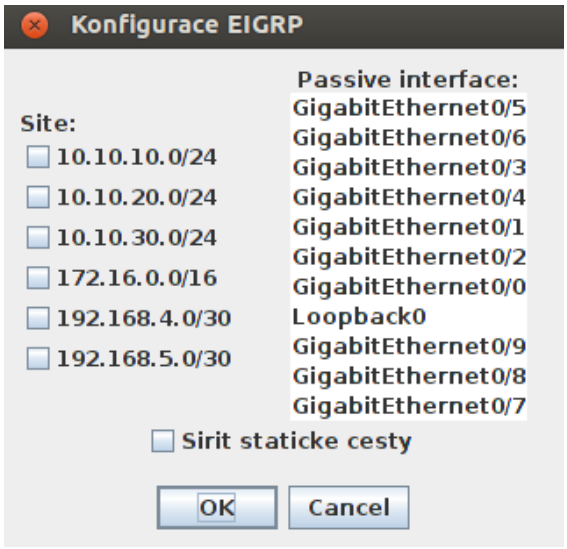
5.2.5.2 Protokol EIGRP

Směrovací protokol EIGRP je v konfiguraci složitější než protokol RIP. Při výběru tohoto protokolu je uživatel nejprve vyzván k zadání čísla autonomního systému, pomocí této funkce je možné síť rozdělit do několika částí, ve kterých může být protokol EIGRP nakonfigurován různými způsoby. Na obrázku č. 27 lze vidět výzvu aplikace pro zadání čísla autonomního systému.



Obr. č. 27 Zadání EIGRP autonomního systému.

Po zadání čísla autonomního systému se objeví podobné okno (obrázek č. 28) jako u protokolu RIP. Nicméně protokol EIGRP vyžaduje kromě adresy sítě také tzv. wildcard masku, která představuje opak ke standartní masce sítě, proto se zde zobrazuje také prefix. Wildcard maska je zadána automaticky na základě prefixu dané sítě.



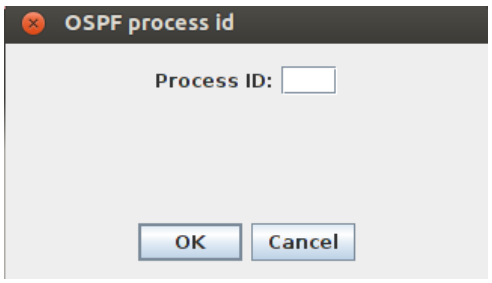
Obr. č. 28 Nastavení protokolu EIGRP.

Oproti protokolu RIP, EIGRP nemá příkaz „default-information originate“, avšak defaultní cestu umí šířit příkazem *„redistribute static“*, který je implementován. Příkazy pro protokol EIGRP jsou předvedeny v následující ukázce.

```
vtyService.write("router eigrp " + autSy);
for (int i = 0; i < chNets.size(); i++){
      if(chNets.get(i).isSelected()){
            String delim = "/";
            String[] cut = (chNets.get(i).getLabel().split(delim));
            String address = cut[0];
            String wildcard = wildcards.get(Integer.parseInt(cut[1]));
            vtyService.write("network " + address + " " + wildcard);
      }
}
if (defChb.isSelected())
      vtyService.write("redistribute static");
for (int i = 0; i < jInterfaces.getSelectedIndices().length; i++)
      vtyService.write("passive-interface "
                  + jInterfaces.getSelectedValues()[i]);
```

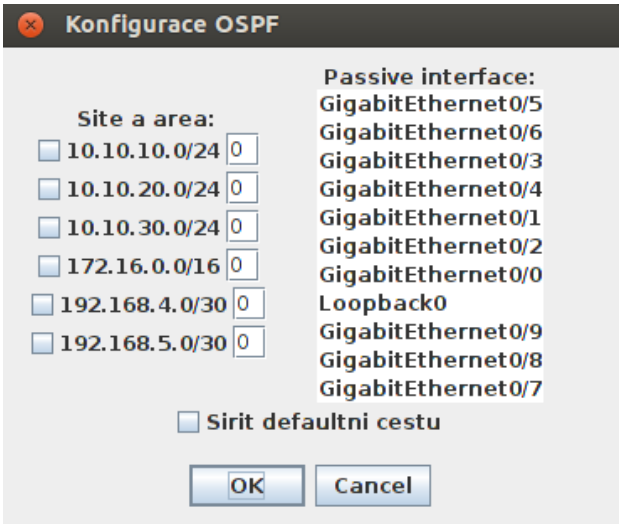
5.2.5.3 Protokol OSPF

Posledním implementovaným dynamickým směrovacím protokolem je protokol OSPF. Jak lze vidět na obrázku č. 29, i po výběru tohoto protokolu je uživatel dále aplikací vyzván k vyplnění, tentokrát ID procesu, což je číslo se stejnou funkcí jako číslo autonomního systému u protokolu EIGRP.



Obr. č. 29 Zadání OSPF Process ID.

I zde je po potvrzení předchozího okna vyžadována dodatečná konfigurace protokolu, jejíž parametry lze vidět na obrázku č. 30.



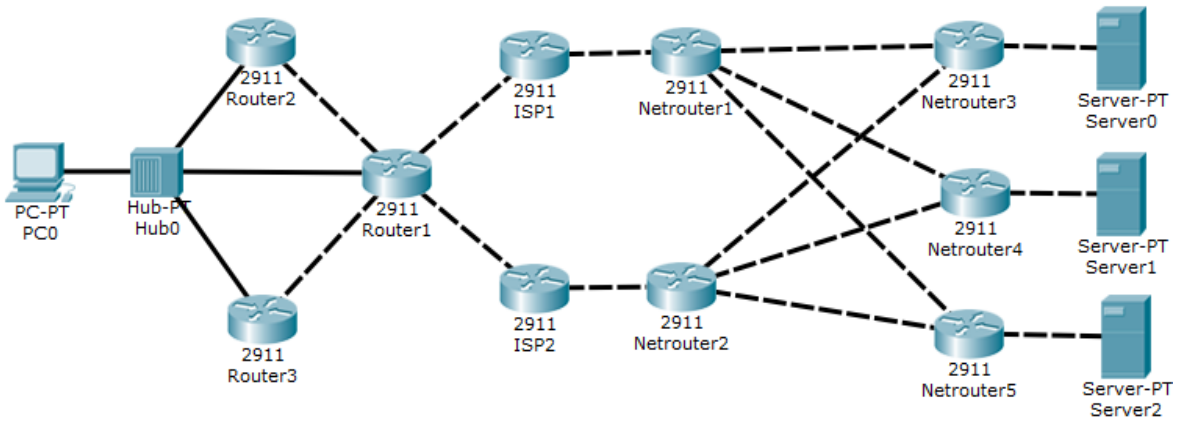
Obr. č. 30 Nastavení protokolu OSPF.

Oproti protokolu EIGRP, OSPF obsahuje navíc tzv. areu, která dokáže, podobně jako ID procesu, rozdělit síť na dílčí části, bez nutnosti opětovné konfigurace protokolu OSPF s jiným ID procesu. Protokol OSPF obsahuje příkaz *„default-information originate“* pro šíření statických cest, stejně jako protokol RIP, a je implementován i stejným způsobem. Následující ukázka obsahuje příkazy pro konfiguraci protokolu OSPF ve vytvořené aplikaci.

```
vtyService.write("router ospf " + id);
for (int i = 0; i < chNets.size(); i++){
      if(chNets.get(i).isSelected()){
            String delim = "/";
            String[] cut = (chNets.get(i).getLabel().split(delim));
            String address = cut[0];
            String wildcard = wildcards.get(Integer.parseInt(cut[1]));
            vtyService.write("network " + address + " " + wildcard +
                        " area " + txtAreas.get(i).getText());
      }
}
if (defChb.isSelected())
      vtyService.write("default-information originate");
for (int i = 0; i < jInterfaces.getSelectedIndices().length; i++)
      vtyService.write("passive-interface "
                  + jInterfaces.getSelectedValues()[i]);
```

5.2.6 Sledování provozu v síti

Poslední funkcí vytvořené aplikace je sledování provozu v síti s přepínáním statických cest do cíle na základě výsledků tohoto sledování. Tato funkce slouží jako ukázka, jak lze OnePK využít v praxi. Jedná se o statickou funkci, využitelnou pouze v dané topologii a pouze pomocí daných interface. Speciálně pro tuto funkci byla vytvořena nová virtuální topologie, simulující reálný provoz, kterou lze vidět na obrázku č. 31.



Obr. č. 31 Virtuální topologie vytvořená pro funkci sledování provozu v síti.

V topologii byly nakonfigurovány dynamické směrovací protokoly OSPF pro horní větev (ISP1 -> Netrouter1 -> Netrouter3, Netrouter4. Netrouter5) a EIGRP pro spodní větev (ISP2 -> Netrouter2 -> Netrouter3, Netrouter4, Netrouter5). Směrovač s názvem „Router1“, na kterém poběží aplikace je propojen se směrovačem „ISP1“ přes svůj interface „GigabitEthernet0/3“ a se směrovačem „ISP2“ přes „GigabitEthernet0/4“. Právě u těchto interface sleduje aplikace jejich zátěž v procentech (Load), chyby (CRC Errors, Frame Errors), rychlost přijmu v bitech za sekundu (Recieve Rate) a počet přijatých multicast paketů. Následující ukázka obsahuje kód sloužící pro sledování statistik interface „GigabitEthernet0/3“.

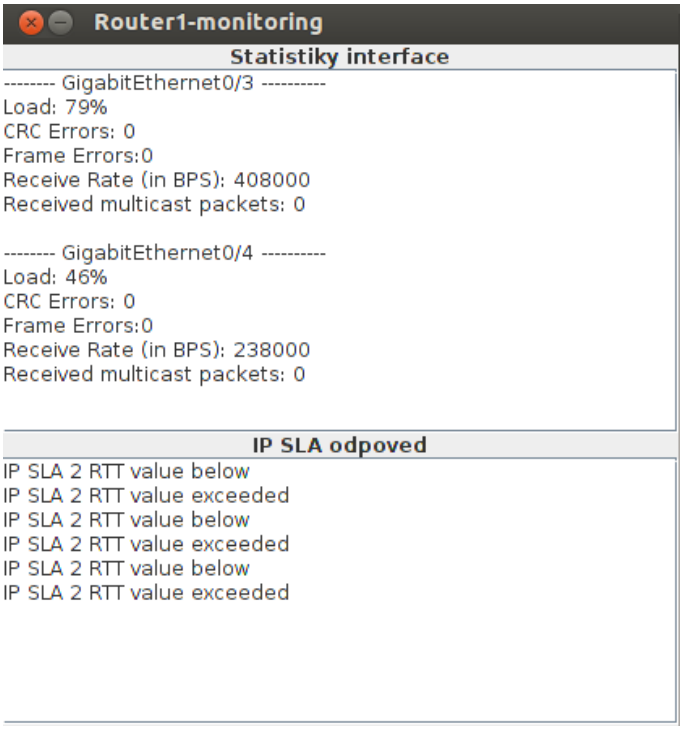
```
networkInterfaces.add(netElement
      .getInterfaceByName("GigabitEthernet0/3"));
areaPoll.append("-------- " + networkInterfaces.get(0).getName()
                + " ---------- \n"
                + ("Load: " + (int)(((float) statistics
                .getReceiveLoad() / 255) * 100) + "% \n")
                + ("CRC Errors: " + statistics.getInErrorCRC() + "\n")
                + ("Frame Errors:"
                + statistics.getInErrorFrame() + "\n")
                + ("Receive Rate (in BPS): "
                + statistics.getReceiveRateBPS() + "\n")
                + ("Received multicast packets: "
                + statistics.getReceiveMulticast() + "\n \n"));
```

Na daném směrovači byly nastaveny IP SLA sondy, které jsou vysílány do cílových stanic (Server0, Server1, Server2) a přináší informace o době trvání cesty do cíle a zpět, tzv. Round-Trip Time (RTT). Sondy jsou označeny čísly 1, 11 a 111 pro horní větev a čísly 2, 22 a 222 pro větev spodní. Maximální hodnoty RTT byly nastaveny na 130 (ms), je-li tato hodnota překročena, zaznamená se přesažení hodnoty RTT dané sondy pomocí syslogu. Minimální hodnoty RTT byly nastaveny na 100 (ms) a označují hranici, pod kterou se musí hodnoty RTT dostat, přesáhnou-li maximální hodnotu a i tuto událost zaznamená syslog. Následuje ukázka konfigurace sondy číslo 1.

```
Router1(config)#ip sla logging traps
Router1(config)#ip sla 1
Router1(config-ip-sla)#icmp-echo 209.14.1.1 source-interface g0/3
Router1(config-ip-sla-echo)#frequency 7
Router1(config)#ip sla schedule 1 life forever start-time now
Router1(config)#ip sla reaction-configuration 1 react rtt
                  threshold-value 130 100 threshold-type
                  immediate action-type traponly
```

První zadaný příkaz slouží pro aktivaci zaznamenávání událostí typu IP SLA pomocí syslogu. Druhý příkaz vytvoří IP SLA sondu s číselným označením 1. Třetí příkaz nastaví sondu tak, aby byla vyslána do cíle s IP adresou „209.14.1.1“, která ozna-

čuje cíl „Server0“, přes interface „GigabitEthernet0/3“. Čtvrtý zadaný příkaz definuje dobu v sekundách, po které se bude sonda opakovaně vysílat do cíle. Pátým příkazem se sonda spustí a její životnost je nastavena na nekonečno. Poslední zadaný příkaz nastaví sondu tak, aby při překročení maximální hodnoty RTT, nebo vrácení se pod minimální hodnotu RTT po přesažení maximální, tyto skutečnosti zaznamenala pomocí syslogu. Okno pro sledování hodnot a odpovědí od nastavených sond se zobrazí po stisku tlačítka „Monitoring“ a lze ho vidět na obrázku č. 32.



Obr. č. 32 Monitorování sítě.

Po nastavení sledování následuje implementace vlastní logiky směrování. Prvním krokem je zajištění, že po vypnutí příslušného interface, nebo při obdržení informace od IP SLA sondy s hodnotou „timeout“ (sonda nedorazila do cíle), se přepnou statické cesty vycházející z tohoto interface. Následující část kódu představuje jednu z podmínek, konkrétně pro případ, že sonda s číselným označením 2 nedorazila do cíle a zároveň také pro případ, kdy sonda s číselným označením 1, která dříve do cíle nedorazila, nyní dorazí, což znamená, že je vždy upřednostňována horní větev, pokud je dostupná.

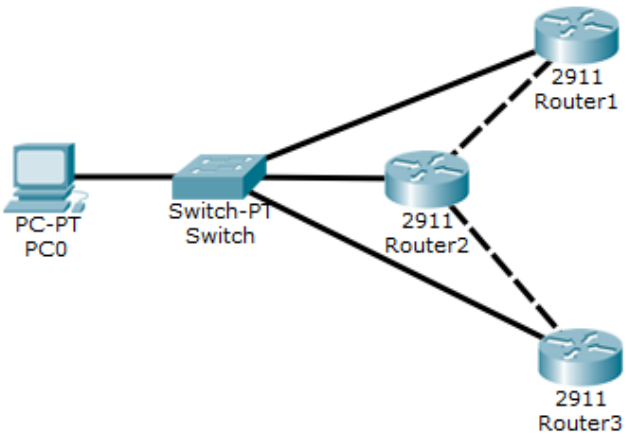
```
if ((timeout.equalsIgnoreCase("occurred") && IPSla2
      .equalsIgnoreCase("2")) || (timeout.equalsIgnoreCase("cleared")
      && IPSla2.equalsIgnoreCase("1"))) {
            vtyService.write("ip route 209.14.1.0 255.255.255.0
            192.168.4.2");
            vtyService.write("no ip route 209.14.1.0 255.255.255.0
            192.168.5.2");
            g41 = false;
            g31 = true;
}
```

Dalším krokem je přepínání statických cest podle zátěže daného interface a RTT. Jeli zátěž na daném interface větší než 80 % a zároveň na druhém interface menší jak 65 %, pak budou sledovány odpovědi sond s informacemi o RTT. Pokud je splněna předchozí podmínka a daná sonda ponese informaci o překročení hodnoty RTT, cesta se přepne. Následující kód je implementací této logiky, konkrétně pro případ, kdy je zátěž na interface směřujícímu ke spodní větvi větší jak 80 %, na interface směřujícímu k horní větvi je zátěž menší jak 65 % a hodnota RTT sondy s číselným označením 1 je pod nastavenou minimální hodnotou nebo hodnota RTT sondy s číselným označením 2 přesáhla nastavenou maximální hodnotu.

```
if ((loadG3 < 166 && loadG4 > 204)) {
      if (((IPSla.equalsIgnoreCase("1") && exOrBelow
            .equalsIgnoreCase("below")) || (IPSla.equalsIgnoreCase("2")
            && exOrBelow.equalsIgnoreCase("exceeded"))) && g31) {
                  vtyService.write("ip route 209.14.1.0 255.255.255.0
                        192.168.4.2");
                  vtyService.write("no ip route 209.14.1.0
            255.255.255.0
                        192.168.5.2");
```

5.3 Testování na reálných síťových prvcích

Vytvořená aplikace byla otestována i na reálných síťových prvcích v síťové laboratoři univerzity. Využity byly tři směrovače Cisco 2911 s operačním systémem IOS ve verzi 15.5 a přepínač Cisco Catalyst 2960. Obrázek č. 33 zobrazuje testovanou síťovou topologii. Přepínač zde slouží pouze jako rozbočovač, aby bylo možné se připojit ke všem třem směrovačům z jednoho počítače, na kterém běží aplikace. Na směrovačích byl vytvořen uživatel příkazem *„username cisco priviledge 15 password 0 cisco“*, bylo povoleno zabezpečení pomocí protokolu TLS a také byl povolen VTY service set. Dále byly nakonfigurovány IP adresy interface směřující k počítači s aplikací a nastavena IP adresa na síťové kartě tohoto počítače. Aplikace se připojila ke všem třem směrovačům a všechny funkce byly dostupné.



Obr. č. 33 Testovaná reálná topologie.

Všechny tři směrovače byly aplikací nakonfigurovány tak, aby bylo jasné, že všechny funkce pracují, jak je vyžadováno. Nejprve byly změněny názvy směrovačů z „Router“ na příslušné názvy dle výše uvedeného obrázku. Poté byly nastaveny IP adresy interface spojující směrovače a IP adresy loopback[8] interface simulující další sítě, případně internet. Následně byla nastavena defaultní cesta na směrovači s názvem „Router2“ s výstupním interface „loopback0“, který simuloval internet. Protokol RIP byl nastaven, aby šířil informace o síti na interface „loopback0“ směrovače „Router3“, protokol EIGRP, aby šířil informace o síti na interface „loopback0“ směrovače s názvem „Router1“ a protokol OSPF byl nastaven, aby šířil informace o síti na interface „loopback1“ směrovače „Router1“. Všem dynamickým protokolům bylo také nastaveno, aby šířili defaultní cestu a informace o sítích spojující směrovače. Tím bylo dosaženo obsažení všech tří typů informací dynamických protokolů a informace o statické cestě ve směrovací tabulce směrovače „Router2“.

Aplikace se chovala stejně jako na virtuálních směrovačích, což znamená úspěšný průběh testování na reálných prvcích.

5.4 Shrnutí

Vytváření aplikací pomocí OnePK je jednoduché a efektivní. Dokumentace poskytnutá na virtuálním stroji je přehledná a jednoznačná, stejně tak je tomu i v případě ukázkových aplikací, jejichž funkce a metody jsou řádně okomentovány. Pokud si uživatel není z komentáře jistý, k čemu daná metoda slouží, nic mu nebrání v tom, aby si příslušnou ukázkovou aplikaci spustil a vyzkoušel.

[8] Loopback interface je virtuální interface vytvořený na směrovači. [44]

Využitím kombinace GUI vytvořeného například v jazyku Java a VTY service setu, lze implementovat libovolnou funkci, která je i běžně dostupná na směrovači s tím, že konfigurace těchto funkcí pomocí aplikace bude mnohem rychlejší a přehlednější. Na směrovačích lze pomocí aplikace vytvořit i nové funkce, například využitím service setu pro zprávy typu syslog. Pokud se směrovač nastaví, aby zaznamenával určité události pomocí syslogu, lze tyto události aplikací zachytit a na základně nich se určitým způsobem zachovat, což bylo ve vytvořené aplikaci demonstrováno funkcí sledování provozu v síti.

Kdyby nebyla ukončena podpora Cisco OnePK, dokázal by být mocným nástrojem i v dnešní době. Přestože se OSC, případně APIC-EM, jeví jako mnohem lepší a jednodušší nástroje, jistě by si i OnePK našel v této konkurenci své místo.

6 Diskuze

SDN má potenciál vyřešit mnoho současných problémů v počítačových sítích, nicméně se jedná pouze o prostředek k řešení těchto problémů a ne o samotné řešení. Jakým způsobem se tohoto přístupu využije je v rukou vývojářů a samotných uživatelů počítačových sítí, kteří se musí sami rozhodnout, jestli chtějí investovat a možná i riskovat. Nicméně s příchodem stále jednodušších a hlavně bezrizikových možností nasazení tohoto přístupu by se v budoucnu mohlo podařit dosáhnout potřebných výsledků. Bezrizikovými možnostmi se v tomto případě myslí možnosti nasazení kontroleru do provozu bez výraznějšího ovlivnění celé sítě a je jen na správci sítě, zda bude tento kontroler využit pro SDN nebo jako obyčejný přepínač.

Velkým problémem stále zůstává zabezpečení sítě a v SDN přístupu, jehož velkou předností je programovatelnost sítě, vzniká možný problém. Pokud by někdo napsal aplikaci, která by byla schopna obejít zabezpečovací protokoly, mohla by v síti napáchat značné škody. V případně OnePK je správce sítě schopen aplikaci vypnout, nicméně záleží, jak rychle bude schopen reagovat. O vypnutí aplikace by se měl starat i IPC, ale otázkou je jeho stoprocentní spolehlivost. Naopak je ale možné, že se podaří vyvinout aplikaci, která bude například zajišťovat kompletní ochranu pro síť a bude odrážet všemožné útoky zaměřené proti ní. Toto však zůstává otázkou budoucnosti.

Dalším tématem je ukončení podpory pro OnePK. Společnost Cisco mohla spíše místo ukončení podpory a zamezení přístupu uvolnit tento nástroj pro veřejnost. Bylo by možné ho využít i pro výukové účely, kde by se mohlo spojit vyučování programování a počítačových sítí zároveň a jelikož síťové prvky vyžadují zadání některých příkazů, aby aplikaci vytvořenou pomocí OnePK přijaly, neměly by být ani obavy z toho, že by tento nástroj někdo využil k páchání škod. Společnost Cisco se jistě mnohému přiučila při vytváření OnePK, takže se investice neztratí, ale i tak se pravděpodobně v budoucnu OnePK nejspíše ztratí a zapomene se na něj.

7 Závěr

V této bakalářské práci byla představena technologie SDN, základní principy SDN přístupu k počítačovým sítím, výhody i možné problémy spojené s nasazením této technologie. Dále byla stručně popsána specifikace OpenFlow a programovatelnost SDN.

Téma SDN je stále více aktuální a stále více se dostává do popředí, což otevírá nové možnosti v počítačových sítích. Hlavním prvkem SDN je programovatelnost sítě a právě tento prvek přináší pro uživatele více nezávislosti na výrobci síťových zařízení.

Jedním z příkladů využití této programovatelnosti sítě je nástroj OnePK od společnosti Cisco. Byl zde popsán základní přehled, funkce a zabezpečení, které tento nástroj využívá a rovněž některé jeho nedostatky, které byly objeveny během vývoje aplikace v tomto prostředí.

Vývoj zmíněné aplikace byl detailně zdokumentován, popsáním všech funkcí, které byly implementovány, včetně názorných obrázků, využitých metod a ukázek kódu. Vytvořená aplikace je využitelná v reálném světě, neboť značně usnadňuje a urychluje konfiguraci směrovačů. Podobným způsobem, jakým jsou v aplikaci vytvořeny jednotlivé funkce, lze vytvořit i spoustu dalších funkcí. Doplněním dalších funkcí by mohla vzniknout aplikace umožňující kompletní konfiguraci směrovačů, bez nutnosti detailní znalosti všech příkazů operačního systému IOS.

V této práci je, aniž by to byl záměr, předvedena i rychlost, s jakou se dnešní technologie vyvíjejí. Během psaní této práce byla ukončena podpora pro nástroj OnePK, který byl relativně novou technologií, a byly představeny dva příklady jeho nástupců.

SDN přístup k počítačovým sítím je jistě uplatnitelný a má v budoucnosti své místo, s dalším vývojem standardů, protokolů, vývojových prostředí a dalších prvků, které budou využívat tohoto přístupu, by se mohly vyřešit některé problémy, kterým počítačové sítě v dnešní době čelí.

8 Seznam použité literatury

[1] Cisco Systems Inc [online]. [cit. 2016-04-21]. Dostupné z: http://www.cisco.com/

[2] Cisco Networking Academy, Cisco Systems, Inc [online]. [cit. 2016-04-25]. Dostupné z: https://www.netacad.com/

[3] OnePK, Cisco Systems, Inc [online]. [cit. 2016-04-21]. Dostupné z: https://developer.cisco.com/site/onepk/

[4] Open Networking Foundation [online]. [cit. 2016-04-21]. Dostupné z: https://www.opennetworking.org/index.php

[5] OpenFlow, Open Networking Foundation [online]. [cit. 2016-04-21]. Dostupné z: https://www.opennetworking.org/sdn-resources/openflow

[6] Open Networking Foundation. Software-Defined Networking: The New Norm for Networks, ONF White Paper [online], 2012. [cit. 2015-04-02]. Dostupné z: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

[7] Ahokas K., Software-defined networking, Aalto University School of Science [online], 2014. [cit. 2015-04-03]. Dostupné z: http://kimia.fi/papers/sdn.pdf

[8] Open Networking Foundation. SDN architecture [online], Issue 1, 2014. ONF TR-502. [cit. 2015-04-03]. Dostupné z: https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/TR_SDN_ARCH_1.0_06062014.pdf

[9] Open Networking Foundation. SDN Architecture Overview [online], Version 1.0, 2013. [cit. 2015-04-03]. Dostupné z: https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/SDN-architecture-overview-1.0.pdf

[10] Meru Networks. An Introduction to SDN: Demystifying Software-Defined Networking for Enterprise Networks [online], 2013. [cit. 2015-04-04]. Do-

stupné z: http://www.merunetworks.com/collateral/solution-briefs/an-introduction-to-sdn-sb.pdf

[11] NADEAU, Thomas D. SDN: software defined networks. Cambridge: O´Reilly, c2013, xxvii, 352 s. ISBN 978-1-449-34230-2.

[12] STALLINGS, William. Software-Defined Networks and OpenFlow, The Internet Protocol Journal [online], Volume 16, No. 1. [cit. 2015-04-07]. Dostupné z: http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_16-1/161_sdn.html

[13] Open Networking Foundation. OpenFlow Switch Specification, Verson 1.5.0 [online], 2014. ONF TS-020. [cit. 2015-04-07]. Dostupné z: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.5.0.noipr.pdf

[14] SHUKLA, Vishal. Introduction to software defined networking: OpenFlow. North Charleston: CreateSpace, c2013, ix, 103 s. ISBN 978-1-48267-813-0.

[15] Kepes B., SDN meets the real-world: implementation benefits and challenges, Nuage Networks [online], 2014. [cit. 2015-04-09]. Dostupné z: http://www.nuagenetworks.net/wp-content/uploads/2014/11/Gigaom-Research-SDN-Meets-the-Real-World-Final.pdf

[16] Sezer S.; Scott-Hayward S.; Chouhan P.K.; Fraser B.; Lake D.; Finnegan J.; Viljoen N.; Miller M.; Rao N., Are we ready for SDN? - Implementation challenges for software-defined networks, Communications Magazine [online], IEEE , vol.51, no.7, July 2013, doi: 10.1109/MCOM.2013.6553676. [cit. 2015-04-09]. Dostupné z: https://netronome.com/wp-content/uploads/2014/07/IEEEComms_Are-we-ready-for-SDN-Whitepaper.pdf

[17] Rouse M., Programmable network (PN) [online], May 2013. [cit. 2016-03-21]. Dostupné z: http://searchsdn.techtarget.com/definition/programmable-network-PN

[18] Ranjan P.; Pandle P.; Oswal R.; Qurani Z.; Bedi R., A Survey of Past, Present and Future of Software Defined Networking, International Journal of Advance Research in Computer Science and Management Studies, Volume 2, Issue 4, April 2014, ISSN: 2321-7782 [online]. [cit. 2016-03-21]. Dostupné z: http://www.ijarcsms.com/docs/paper/volume2/issue4/V2I4-0042.pdf

[19] Hong S.; Baykov R., Xu L.; Nadimpalli S., Gu. G, Towards SDN-Defined Programmable BYOD (Bring Your Own Device) Security [online], 2016. [cit. 2016-03-21]. Dostupné z: https://www.internetsociety.org/sites/default/files/blogs-media/towards-sdn-defined-programmable-bring-your-own-device-security.pdf

[20] Mitchiner M., Prasad R., Software-Defined Networking and Network Programmability: Use Cases for Defense and Intelligence Communities, Cisco Systems, Inc [online], 2014. [cit. 2016-04-01]. Dostupné z: http://www.cisco.com/c/dam/en_us/solutions/industries/docs/gov/software_defined_networking.pdf

[21] Cisco Open Network Environment, Cisco Systems, Inc [online]. [cit. 2016-04-03]. Dostupné z: http://www.cisco.com/web/solutions/trends/open_network_environment/index.html

[22] OnePK Technical Overview, Cisco Systems, Inc [online]. [cit. 2016-04-03]. Dostupné z: https://developer.cisco.com/site/onepk/discover/overview/

[23] OnePK Design Guidelines, Cisco Systems, Inc [online]. [cit. 2016-04-04]. Dostupné z: https://developer.cisco.com/media/onepk_design_guide/GUID-1C3A8615-9BFB-4C2A-B227-CF27DA9AE93B.html

[24] OnePK Java Tutorials, Cisco Systems, Inc [online]. [cit. 2016-04-04]. Dostupné z: https://developer.cisco.com/site/onepk/learn/tutorials/java/

[25] Kent K., Souppaya M., Special Publication 800-92: Guide to Computer Security Log Management, National Institute of Standards and Technology [online],

Gaithersburg 2006. [cit. 2016-04-21]. Dostupné z: http://csrc.nist.gov/publications/nistpubs/800-92/SP800-92.pdf

[26] Cisco Application Developer Security Guide, OnePK Developer Community, Cisco Systems, Inc [online]. [cit. 2016-04-08]. Dostupné z: https://developer.cisco.com/site/onepk/documents/security/security-guide/

[27] Product End-of-Life Notice, Cisco's One Platform Kit (onePK), Cisco Systems, Inc [online], February 2016. [cit. 2016-04-08]. Dostupné z: http://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/bulletin-c25-736612.pdf

[28] Open Device Programmability, Cisco Systems, Inc [online]. [cit. 2016-04-08]. Dostupné z: https://developer.cisco.com/site/odp/

[29] Cisco Open SDN Controller, Cisco Systems, Inc [online]. [cit. 2016-04-21]. Dostupné z: https://developer.cisco.com/site/openSDN/

[30] APIC Enterprice Module, Cisco Systems, Inc [online]. [cit. 2016-04-21]. Dostupné z: https://developer.cisco.com/site/apic-em/

[31] Cisco Open SDN Controller 1.2 Data Sheet, Cisco Systems, Inc [online], October 2015. [cit. 2016-04-08]. Dostupné z: http://www.cisco.com/c/en/us/products/collateral/cloud-systems-management/open-sdn-controller/datasheet-c78-733458.html

[32] OpenDaylight [online]. [cit. 2016-04-08]. Dostupné z: https://www.opendaylight.org/

[33] CiscoDevNet [online]. [cit. 2016-04-08]. Dostupné z: https://developer.cisco.com/site/devnet/home/index.gsp

[34] Cisco dCloud 1.5, Cisco Systems, Inc [online]. [cit. 2016-04-21]Dostupné z: https://dcloud-lon-web-1.cisco.com/dCloud/

[35] Open SDN Controller Software-1.2.1, Cisco [online]. [cit. 2016-04-08]. Dostupné z: https://software.cisco.com/download/release.html?mdfid=286289357&flowid=76402&softwareid=286285525&release=1.2.1&relind=AVAILABLE&rellifecycle=&reltype=latest

[36] APIC Enterprise Module API Overview, Cisco [online]. [cit. 2016-04-08]. Dostupné z: https://developer.cisco.com/site/apic-em/discover/overview/

[37] Getting Started, APIC Enterprise Module (APIC-EM), Cisco [online]. [cit. 2016-04-08]. Dostupné z: https://developer.cisco.com/site/apic-em/documents/getting-started/

[38] Sandbox Labs, Cisco [online]. [cit. 2016-04-08]. Dostupné z: https://devnetsandbox.cisco.com/RM/Topology

[39] VMware Player 7, VMware Inc [online]. [cit. 2016-04-23]. Dostupné z: https://my.vmware.com/web/vmware/free#desktop_end_user_computing/vmware_player/7_0

[40] TechNet, What Is Unicast IPv4 Routing?, Microsoft [online]. March 2003. [cit. 2016-04-21]. Dostupné z: https://technet.microsoft.com/de-de/library/cc736574(WS.10).aspx

[41] Next Hop, Technopedia [online]. [cit. 2016-04-21]. Dostupné z: https://www.techopedia.com/definition/2447/next-hop

[42] Davis D., Cisco administration 101: What you need to know about default routes, TechRepublic [online], April 2007. [cit. 2016-04-22]. Dostupné z: http://www.techrepublic.com/article/cisco-administration-101-what-you-need-to-know-about-default-routes/

[43] Passive-interface, Cisco Tips & Tricks [online], May 2006. [cit. 2016-04-22]. Dostupné z: https://ciscotips.wordpress.com/2006/05/11/passive-interface/

[44] TechLibrary, Understanding the Loopback Interface, Juniper Networks [online], May 2014. [cit. 2016-04-22]. Dostupné z: http://www.juniper.net/techpubs/en_US/junos15.1/topics/concept/interface-security-loopback-understanding.html

9 Seznam použitých zkratek

AAA	Authentication, authorization and accounting
ACL	Access list
A-CPI	Application-Controller Plane Interface
API	Application Programming Interface
APIC-EM	Application Policy Infrastructure Controller Enterprise Module
D-CPI	Data-Controller Plane Interface
DoS	Denial of Service
DPCF	Data plane control function
EIGRP	Enhanced Interior Gateway Routing Protocol
FIB	Forwarding information base
GUI	Graphical User Interface
IP SLA	Internet Protocol Service Level Agreement
IP	Internet Protocol
IPC	Inter-process communication
ONE	Open Network Environment
OnePK	One Platform Kit
ONF	Open Networking Foundation
OSC	Open SDN Controller
OSPF	Open Shortest Path First
OSS	Operations support systém
QoS	Quality of Service
RDB	Resources database
RIB	Routing information base
RIP	Routing Infromation Protocol
RTT	Round-Trip Time
SDK	Software Development Kit
SDN	Software-defined networking
TLS	Transport Layer Security
VTY	Virtual Teletype

10 Seznam obrázků

11 Přílohy

Příloha 1) Obsah CD

- **classes** – ve složce se nachází třídy ve formátu „.java“, ze kterých lze vyčíst zdrojový kód
- **javadoc** – ve složce se nachází vygenerovaná dokumentace vytvořené aplikace
- **vmcloud-topology** – ve složce se nachází soubory základní virtuální topologie
- **vmcloud-test-topology** – ve složce se nachází soubory virtuální topologie pro funkci „Sledování provozu v síti“
- **OnePKApplication.jar** – vytvořená spustitelná aplikace
- **readme.txt** – podrobný rozpis příkazů zadávaných na směrovači nutných k úspěšnému spuštění a běhu aplikace

FIM UHK

UNIVERZITA HRADEC KRÁLOVÉ
Fakulta informatiky a managementu
Rokitanského 62, 500 03 Hradec Králové, tel: 493 331 111, fax: 493 332 235

Zadání k závěrečné práci

Jméno a příjmení studenta: **Lukáš Nemecz**

Obor studia: Aplikovaná informatika
Jméno a příjmení vedoucího práce: **Agáta Bodnárová**

Název práce:
Softwarově definované sítě

Název práce v AJ:
Software-Defined Networking

Podtitul práce:
Vývoj aplikace v prostředí OnePK

Podtitul práce v AJ:
Development of application in OnePK environment

Cíl práce: Vytvoření funkční aplikace, napsané v programovacím jazyce Java, v prostředí OnePK od společnosti Cisco.

Osnova práce:
1. Úvod
2. Teoretický úvod do technologie
3. Prostředí OnePK
4. Vývoj aplikace
5. Diskuze
6. Závěr
7. Literatura

Projednáno dne:
19.10.2014

Podpis studenta

Podpis vedoucího práce