Sem vložte zadání Vaší práce.

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra informačních technologií

Bakalářská práce

Rozšíření síťového simulátoru o možnost použití konfiguračních souborů pro konfiguraci síťových prvků

Michal Horáček

Vedoucí práce: Ing. Pavel Kubalík, Ph.D.

13. května 2014

Poděkování

Chtěl bych poděkovat vedoucímu práce Ing. Pavlu Kubalíkovi Ph.D. za námět k bakalářské práci a za rady při její realizaci. Dále děkuji kolegovi Václavu Machovi za spolupráci při tvorbě práce a všem studentům, kteří se zúčastnili testování programu. Chtěl bych také poděkovat své rodině za podporu při psaní této práce.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 13. května 2014 .....

Odkaz na tuto práci

Horáček, Michal. *Rozšíření síťového simulátoru o možnost použití konfiguračních souborů pro konfiguraci síťových prvků*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2014.

Abstrakt

Tato práce se zabývá návrhem, implementací a testováním rozšiřujícího modulu pro simulátor počítačových sítí Psimulator2. Ten je využíván zejména pro výukové účely předmětu BI-PSI na Fakultě informačních technologií. Modul rozšiřuje existující verzi simulátoru o zjednodušené verze protokolu DHCP a DNS. Pro tyto protokoly a pro nastavení síťových rozhraní na virtuálních počítačích na bázi linuxu se práce dále zabývá možností ukládání a načítání konfigurace pomocí virtuálních konfiguračních souboru na jednotlivých zařízeních.

Klíčová slova Síťový simulátor, počítačové sítě, síťové protokoly, linux

Abstract

This bachelors thesis describes the design, implementation and testing of an extending module for computer network simulator Psimulator2. This program is being used mostly for educational purposes of the course BI-PSI at Faculty of Informational Technology. Module adds simplified versions of the DHCP and DNS protocols to the current simulator. For these protocols, and for the settings of network interfaces of virtual linux-based machines, this thesis also deals with the possibility to load and save configurations using configuration files on virtual machines.

Keywords Network simulator, computer networks, network protocols, linux

Obsah

Úvod **1**

1 Specifikace práce **3**
1.1 Cíle práce 3
1.2 Použité nástroje 4

2 Aplikace Psimulator2 **5**
2.1 Frontend 5
2.2 Backend 6

3 Analýza a návrh **9**
3.1 Konfigurace síťových rozhraní 9
3.2 DHCP protokol 11
3.3 DNS protokol 14

4 Realizace **17**
4.1 Konfigurační soubory 17
4.2 Aplikační vrstva 19
4.3 Služba networking 19
4.4 DHCP klient 20
4.5 DHCP server 22
4.6 DNS resolver 23
4.7 DNS server 24

5 Testování **25**
5.1 Jednotkové testy 25
5.2 Uživatelské testování 25

6 Návrhy na vylepšení **29**

Závěr **31**

Literatura **33**

A Seznam použitých zkratek **35**

B Instalační příručka **37**
B.1 Minimální požadavky 37
B.2 Spuštění aplikace 37

C Obsah přiloženého CD **39**

Seznam obrázků

2.1 Ukázka GUI 6

4.1 Architektura konfiguračních souborů 18
4.2 Diagram běhu klientského DHCP vlákna 21

5.1 Schéma sítě použité pro uživatelské testování 26

Úvod

Cílem bakalářského předmětu BI-PSI[1] na FIT ČVUT je seznámit studenty se základními principy fungování počítačových sítí. Na laboratorních cvičeních je zde možné si vlastnoručně vyzkoušet konfiguraci malé sítě propojené přes počítače s OS Linux nebo směrovače Cisco. U mnohých studentů se však jedná o první kontakt s touto problematikou, a proto během cvičení stráví většinu času řešením banálních problémů. Vzhledem k tomuto faktu a omezené hodinové dotaci předmětu nezbývá čas na nic jiného než nejjednodušší možné příklady.

Tímto problémem se ve svých diplomových pracích zabývali v roce 2012 Bc. Tomáš Pitřinec[7], Bc. Stanislav Řehák[8], Bc. Martin Lukáš[6] a Bc. Martin Švihlík[9] při tvorbě aplikace Psimulator2. Jedná se o jednoduchý simulátor počítačových sítí zaměřený zejména na potřeby předmětu Počítačové sítě. Program obsahuje uživatelské rozhraní pro snadnou tvorbu a úpravu schémat počítačových sítí. Backend aplikace umožňuje simulovat počítače s operačním systémem Linux nebo Cisco IOS a síťovou komunikaci mezi těmito počítači na linkové, síťové a transportní vrstvě.

Ve své bakalářské práci se věnuji návrhu a implementaci rozšíření pro linuxová zařízení v Psimulatoru. Jmenovitě se jedná o implementaci protokolů DHCP a DNS. Dále se zabývám využitím virtuálního souborového systému na zařízeních pro možnost ukládání a načítání konfigurace síťových rozhraní a zmíněných protokolů v rámci konfiguračních souborů.

[1] Počítačové sítě, stránky předmětu jsou https://edux.fit.cvut.cz/courses/BI-PSI/

Kapitola 1

Specifikace práce

1.1 Cíle práce

Cílem této práce je rozšíření simulátoru počítačových sítí Psimulator2, který je využíván jako výuková pomůcka v rámci předmětu BI-PSI. Na laboratorních cvičeních tohoto předmětu byly v posledních letech nasazeny systémy Ubuntu a Debian a ani do budoucnosti nic nenasvědčuje výraznější změně. Proto by všechny implementované komponenty měly být co nejvěrnějšími kopiemi jejich ekvivalentů na těchto systémech.

Pro projekt rozšíření dosavadní verze Psimulatoru projevil zájem kromě mě i kolega Václav Mach. Práci na rozšiřujících modulech jsme si rozdělili následujícím způsobem:

1.1.1 Michal Horáček (já):

- Rozšíření o možnost uchování a načítání konfigurace síťových rozhraní pomocí konfiguračních souborů na virtuálních zařízeních
- Rozšíření implementace protokolu DHCP
- Možnost konfigurace DHCP server pomocí konfiguračních souborů
- Implementace DNS protokolu
- Možnost konfigurace DNS protokolu pomocí konfiguračních souborů

1.1.2 Václav Mach

- Návrh a implementace modulu umožňujícího připojení simulátoru do reálné sítě

- Komunikace simulátoru s reálnou sítí na linkové úrovni
- Možnost tohoto propojení s počítači běžících na platformách OS Linux a OS Microsoft Windows
- Přídání možnosti spouštěn simulaci přímo z frontendu simulátoru

1.2 Použité nástroje

Při implementaci rozšíření byl z důvodu integrace s původním programem zvolen jazyk Java. Při vývoji bylo oběma vývojáři používáno vývojové prostředí Netbeans IDE.

Pro distribuci výsledné aplikace bylo zřízeny stránky projektu na adrese https://sourceforge.net/projects/psimulator2/. Na těchto stránkách byl ke správě kódu zřízen repositář Subversion.

Kapitola 2

Aplikace Psimulator2

Z důvodu lepšího pochopení mých rozšíření aplikace Psimulator2[2] se tato kapitola věnuje představení struktury a důležitých prvků původní verze tohoto programu. Důraz je kladen především na části, které přímo souvisí s mými úpravami.

Jak už bylo zmíněno v úvodu této práce, program Psimulator2 vzniknul v roce 2012 v rámci čtyř diplomových prací na FIT CVUT a hlavním cílem tvorby této aplikace byla podpora výuky bakalářského předmětu BI-PSI.

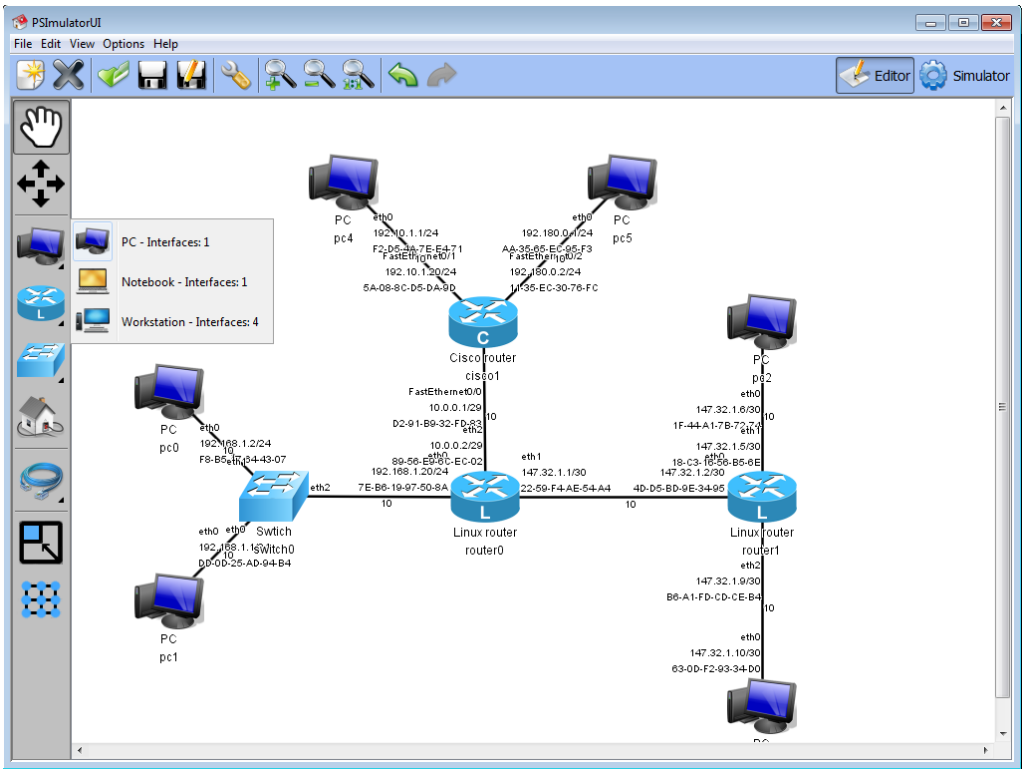
Od programu se od už počátku jeho vývoje také očekávalo, že bude bez obtíží použitelný alespoň na operačních systémech Windows, Linux a MacOS X. Pro zjednodušení přenositelnosti mezi těmito platformami byl celý simulátor vyvíjen v jazyce Java.

Psimulator se skládá ze dvou hlavních komponent. První z nich je frontend, který představuje grafické uživatelské rozhraní pro tvorbu schémat virtuálních počítačových sítí. Druhou komponentou je potom backend, v němž probíhají veškeré simulace.

2.1 Frontend

Podoba uživatelského rozhraní je zobrazena na obrázku 2.1. Tato komponenta slouží především k tvorbě a úpravě topologie požadované sítě. Virtuální prvky sítě je zde možné nejen vytvorit, ale zároveň i na jejich jednotlivých rozhraních nastavit údaje jako jsou jméno tohoto rozhraní, IP a MAC adresa. Možné je také určit, zda je při startu simulace rozhraní aktivní. Výsledné schéma je následně uloženo do XML souboru, který bude použit pro předání informací o podobě virtuální sítě backendu simulátoru.

[2]Stránky projektu https://code.google.com/p/psimulator/



Obrázek 2.1: Ukázka uživatelského rozhraní simulátoru

Součástí uživatelského rozhraní je také telnet klient, s jehož pomocí je po spuštění simulace v backendu možné ovládat jednotlivá virtuální zařízení.

Kolega Václav Mach se ve své práci zabývá mimo jiné i přidáním možnosti zahájit simulaci přímo z grafického rozhraní bez nutnosti spouštění backendu. Více informací o tomto rozšíření je možné nalézt v jeho bakalářské práci[3].

Detailnější popis implementace frontend části programu je obsažen v diplomové práci původního autora Bc. Martina Švihlíka[9].

2.2 Backend

Backend představuje serverovou část programu. Umožňuje simulovat posílání paketů mezi virtuálními zařízeními na bázi linuxu a Cisco IOS. Při startu simulace dojde k načtení topologie a konfigurace sítě. Podle těchto informací jsou jednotlivá zařízení vytvořena a následně spuštěna.

Mezi nejdůležitější komponenty implementované v původní verzi backendu patří:

- protokoly ARP, IPv4, ICMP a UDP
- směrování paketů
- dynamický a statický překlad adres
- příkazy ping a traceroute

2.2.1 Síťové vrstvy

Síťová komunikace je v rámci zařízení rozdělena do vrstev podle modelu ISO/OSI.

Komunikace mezi jednotlivými vrstvami probíhá prostřednictvím dvou bufferů. Z přijímacího bufferu jsou zpracovávána data určena pro dotyčnou vrstvu. Do odesílacího bufferu jsou potom vkládána data, která vrstva předává jinam.

Vrstvy a aplikace na virtuálních počítačích jsou realizovány za pomoci uspávacího vlákna. Jeho princip spočívá v tom, že po dokončení veškeré práce příslušné vrstvy se uspí. K jeho probuzení potom může dojít dvěma způsoby:

1. Vlákno jiné vrstvy nebo aplikace vloží do přijímacího bufferu vrstvy uspaného vlákna data ke zpracování a následně uspané vlákno probudí.
2. Simulátor obsahuje budík, pomocí kterého si může vlákno před uspáním nastavit dobu probuzení. Tato možnost se využívá především při timeoutu komunikace některé z aplikací.

Více o síťové komunikaci je možné nalézt v diplomových pracech Bc. Tomáše Pitříince[7] a Bc. Stanislava Řeháka[8].

2.2.2 Souborový systém

Pro práci s linuxovými zařízeními existuje v simulátoru jednoduchý souborový systém. Ten byl implementován s využitím knihovny TrueZip[3]. Při startu simulace dojde v pracovním adresáři projektu k vytvoření složky,

[3]Stránky knihovny: https://truezip.java.net/

která obsahuje soubory reprezentující souborové systémy jednotlivých zařízení.

Práce se souborovým systémem kopíruje základní funkcionalitu reálného linuxového počítače. K ovládání jsou implementovány zjednodušené verze základních linuxových příkazů jako jsou např. mv nebo cd. Pro tvorbu a úpravu virtuálních souborů byl vytvořen jednoduchý textový editor.

Detailnější popis souborového systému se nachází v diplomové práci jeho autora Bc. Martina Lukáše[6].

Kapitola 3

Analýza a návrh

V této kapitole se zabývám analýzou a návrhem specifikovaných rozšíření programu Psimulator. U komponent, které již v nějaké podobě jsou v programu obsaženy, analyzuji jejich nedostatky a věnuji se možnostem jejich vylepšení.

3.1 Konfigurace síťových rozhraní

V současné verzi simulátoru je na linuxových zařízeních možné upravovat nastavení příslušných síťových rozhraní příkazem *ifconfig*. Takto zadaná změna je ovšem dočasná, protože se neuloží do XML souboru, který předává informace o schématu mezi backendem a backendem simulátoru. Po novém startu simulace bude rozhraní opět nastavené podle informací zadaných prostřednictvím grafického rozhraní.

Toto chování je záměrné a kopíruje reálné chování příkazu *ifconfig* na OS Linux. Přesto by však bylo vhodné přidat možnost, jak trvale upravit konfiguraci síťového rozhraní, aniž by bylo nutné přepínat do frontendu, a následně restartovat spuštěnou simulaci.

3.1.1 Soubor interfaces

Problém přidání možnosti pro trvalou změnu nastavení rozhraní přímo z backendu bude řešen prostřednictvím virtuálního konfiguračního souboru na jednotlivých zařízeních. Pro návrh tohoto souboru jsem se inspiroval konfiguračním souborem `interfaces` na systému Debian. Z velkého množství parametrů, které je možné pomocí tohoto souboru definovat bude pro potřeby simulátoru stačit několik základních. V rámci souboru bude síťové

rozhraní možné nastavit jako statické nebo dynamické. Pro statické rozhraní budou v konfiguračním souboru vypsány údaje o IP adrese a masce na tomto rozhraní. Dále bude možnost přidat informace o výchozích branách pro tato rozhraní. Pro dynamická rozhraní nebudou uvedeny žádné další údaje.

3.1.2 Služba networking

Práci se souborem interfaces bude zajišťovat služba networking, inspirovaná stejnojmennou službou obsaženou v systému Debian. Jediným požadavkem na ní bude, že při každém svém spuštění nebo restartu projde zmíněný konfigurační soubor a podle informací, které z něj získá, upraví nastavení síťových rozhraní. Pro dynamicky nastavená rozhraní služba networking zavolá aplikaci DHCP klienta. Ta zahájí proces půjčky dynamické adresy. Po tomto nastavení následně provede uložení změn do XML souboru se schématem sítě.

3.1.3 Požadavky na komponentu

Služba networking bude před vlastním nastavováním a ukládáním informací o rozhraních zajišťovat, že údaje načtené ze souboru interfaces jsou validní. Nezbytné je kontrolovat zejména tyto údaje:

- existenci povinných údajů (číslo IP adresy a masky sítě) pro statické síťové rozhraní
- v případě výskytu nepovinných parametrů (číslo sítě, broadcast adresa) je třeba kontrolovat, zda se některé ze zadaných údajů navzájem nevylučují
- parametry musí obsahovat validní IP adresy

V případě problému při kontrole bude pro příslušné síťové rozhraní proces nastavení údajů ukončen a na terminál daného zařízení se vypíše odpovídající chybová hláška.

Důležitým požadavkem je, aby konfigurace v souborech virtuálních zařízení obsahovala vždy stejné informace jako hlavní konfigurační soubor. Toho bude zajištěno tak, že při každém startu simulace bude do těchto souborů vyčteno nastavení z XML souboru schématu. Pokud tedy dojde k úpravě informací prostřednictvím grafického rozhraní, budou soubory na virtuálních zařízeních stále obsahovat aktuální informace.

3.2 DHCP protokol

DHCP (Dynamic Host Configuration Protocol) je síťový protokol, který se používá v rámci IP protokolu pro dynamickou distribuci konfiguračních parametrů. Při komunikaci v rámci tohoto protokolu se rozlišují dva účastníci - server a klient. Prací serveru je alokace a správa síťových parametrů jako jsou IP adresy. Dalším úkolem serveru je možnost tyto alokované parametry následně poskytovat žádajícím klientům. Jako klient je potom označováno zařízení, které požaduje a přijímá síťové nastavení od serveru.

Základní mechanismus komunikace mezi DHCP klientem a serverem pro přidělení nové síťové konfigurace se potom odehrává v pěti hlavních fázích:

1. Komunikaci zahajuje klient posláním zprávy DHCPDISCOVER na broadcast lokální sítě.

2. Každý ze serveru, který tuto zprávu obdrží, rozhodne o nastavení, které by bylo pro žádající klienta platné a následně muže odpovědět zprávou DHCPOFFER. Tato zpráva bude obsahovat dostupnou síťovou konfiguraci. Pro vyšší efektivitu protokolu muže server pro příslušného klienta tuto konfiguraci rovnou zarezervovat.

3. Klient si vybere jednu z příchozích DHCPOFFER zpráv a na broadcast odešle odpověď DHCPREQUEST. Tato zpráva musí obsahovat identifikátor serveru, který byl zvolen. Pokud klient do určitého času od odeslání DHCPDISCOVER zprávy neobdrží žádnou DHCPOFFER zprávu, zůstává ve fázi 1 a znovu posílá požadavek DHCPDISCOVER.

4. Server při přijetí zprávy DHCPREQUEST zjistí, zda je určena pro něj. Pokud ne, muže opět uvolnit případné zarezervované konfigurace. Server, jemuž je zpráva určena, si uloží údaje o přidělení konfigurace a toto přidělení klientovi potvrdí zprávou DHCPACK. Tato zpráva by měla mít stejné parametry jako zpráva DHCPOFFER posílaná dříve.

5. Klient přijme zprávu DHCPACK a přijatou konfiguraci si uloží. Zároveň si poznamená po jakou dobu jsou přijaté údaje platné. Pokud do určitého casu od odeslání DHCPREQUEST zprávy neobdrží klient potvrzení, snaží se DHCPREQUEST poslat znovu.

3.2.1 Implementace v původní verzi

Původní verze simulátoru obsahovala pouze základní kostru tohoto protokolu. Server neposkytoval žádné jiné konfigurační údaje než IP adresu.

Seznam takto přidělitelných adres byl dán přímo jejich výpisem ve zdrojovém kódu Java třídy reprezentující server. Při odeslání DHCP paketu byla takto zadefinovaná adresa ze seznamu navždy smazána. Server si navíc neudržoval žádné informace o tom, které adresy a komu poskytoval. Dále na případných klientských počítačích chyběl jednoduchý způsob, jak o přidělení adresy požádat. Celá tato implementace protokolu tedy sloužila především jako šablona pro tvorbu případných dalších aplikací a služeb než jako fungující verze protokolu.

3.2.2 Požadavky na novou verzi protokolu

Mým cílem je implementovat novou verzi tohoto protokolu tak, aby vyhovoval alespoň potřebám výuky na předmětu BI-PSI Mezi hlavní požadavky patří:

- Server si bude uchovávat různé konfigurace pro různé podsítě, do kterých patří klientské počítače
- Server bude umožňovat poskytovat klientům alespoň IP adresu, informace o jmenných serverech a informace o branách na dané podsíti
- Každému pronajmutí konfigurace bude poskytnuta doba platnosti
- Server si bude uchovávat informace o poskytnutých údajích včetně toho, komu je poskytl
- Klientská část protokolu se automaticky vzdá přidělené konfigurace po vypršení doby její platnosti
- Pro stále platný záznam bude dynamicky konfigurované rozhraní obsahovat informace o přidělené konfiguraci i po restartování simulátoru

3.2.3 Návrh implementace protokolu

Pro možnosti konfigurace a spravování DHCP záznamů na straně jak klienta, tak serveru, budu vycházet ze standardní implementace ISC DHCP[4], která je vyučována na laboratorních cvičeních předmětu BI-PSI.

Jako dokumentace ISC DHCP serveru jsou odkazovány manuálové stránky dhcpd(8)[2]. Podle těchto stránek patří mezi základní principy fungování aplikace serveru tyto:

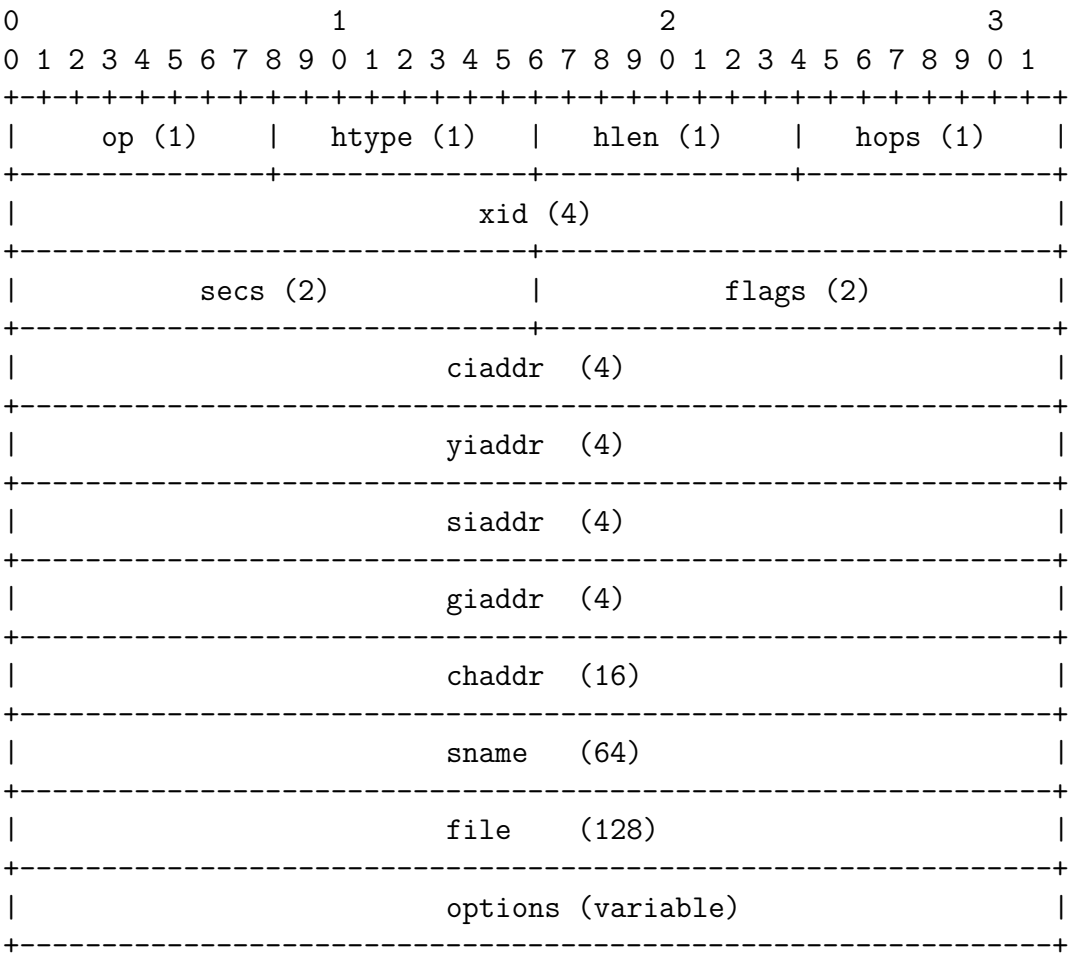
1. při spuštění získá server konfiguraci ze souboru dhcpd.conf

[4]Oficiální stránky http://www.isc.org/downloads/dhcp/

2. z tohoto souboru následně načte seznam přidělitelných adres
3. ze souboru dhcpd.leases získá server seznam adres, které již byly přiděleny a ty ze seznamu přidělitelných adres vymaže
4. před přidělením adresy klientovi nejprve uloží informace o poskytované adrese do souboru dhcpd.leases

3.2.4 DHCP paket

Struktura DHCP paketu je definována podle RFC2131[5] takto:



Format of a DHCP message

Pro potřeby simulátoru budou nutná alespoň tato pole:

- yiaddr - IP adresa, kterou server přiděluje klientovi.
- chaddr - Hardware adresa klienta. Bude použita na straně serveru k identifikaci klienta.
- xid - Identifikační číslo transakce náhodně vygenerované klientem.
- options - Seznam nepovinných parametrů pro konfiguraci jako jsou adresa jmenného serveru nebo brány pro dané rozhraní.

3.3 DNS protokol

DNS (Domain Name System) je hierarchický systém sloužící k překladu mezi číselnými IP adresami a doménovými jmény. Aplikace, které potřebují informace o tomto překladu posílají své požadavky na proces nazývaný *resolver*. Jeho úkolem je potom najít odpověď v lokální databázi, nebo se s dotazem obrátit na některý z DNS serverů. DNS servery si uchovávají data pro prostory doménových jmen nazývaných zóny.

3.3.1 Návrh implementace

Za účelem uchování mapování doménových jmen budou na DNS serveru k dispozici dva typy souborů. Prvním z nich je katalog, ve kterém budou uvedeny všechny zóny dostupné pro tento soubor. Pro každou zónu uvedenou v katalogovém souboru bude dále existovat zónový soubor, který bude obsahovat záznamy o této zóně. Struktura obou těchto souborů se bude snažit kopírovat rozšířenou open-source implementaci Bind[5].

Linuxová zařízení budou obsahovat soubory *hosts*, obsahující lokální databázi s mapováním jmenných na číselné adresy, a *resolv.conf*, ve kterém jsou obsaženy adresy jmenných serverů. Ze strany klienta bude také nutné rozšířit příkazy ping a traceroute tak, aby se jejích pomocí bylo možné dotazovat na vzdálené počítače kromě jejich číselných IP adres i prostřednictvím doménových jmen.

3.3.2 DNS zpráva

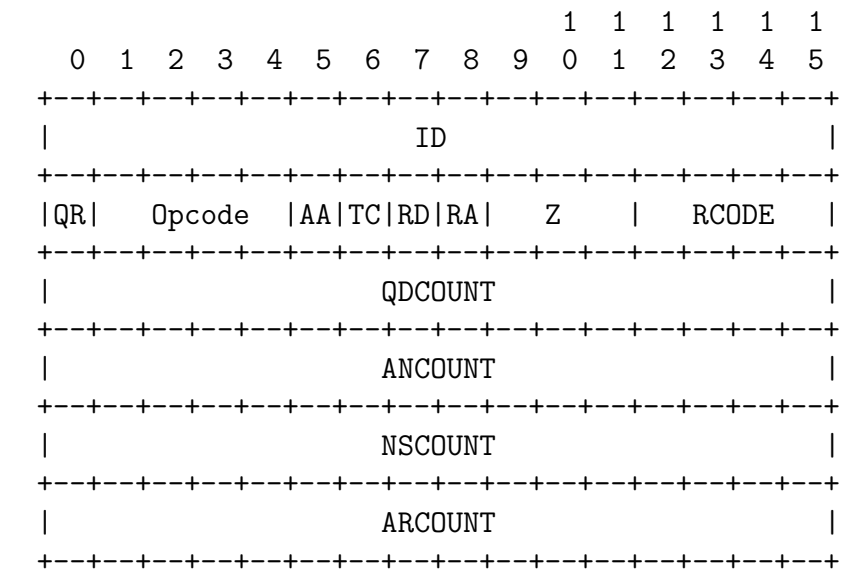
DNS zpráva se podle RFC1035[4] skládá z pěti sekcí:

- Header

[5]Stránky projektu: https://www.isc.org/downloads/bind/

- Question - dotaz kladený jmennému serveru
- Answer - odpověď na dotaz
- Authority - záznamy ukazující na autoritativní jmenné servery
- Additional - záznamy související s dotazem

Header sekce je definována následovně:



Pro implementaci v rámci simulátoru budou potřeba alespoň položky QR, RCODE a pole udávající počet záznamů v jednotlivých sekcích zprávy. Položka QR udává, zda se jedná o požadavek či o odpověď. V položce RCODE je potom zadáno, zda dotazování proběhlo bez problémů, případně specifikuje k jakému problému došlo.

Kapitola 4

Realizace

V této kapitole se zabývám realizací navržených rozšíření. Nejprve se zaměřím na architekturu konfiguračních souborů. V druhé části této kapitoly potom přejdu na popis implementace samotných aplikačních protokolů a na jejich integraci do programu Psimulator2.

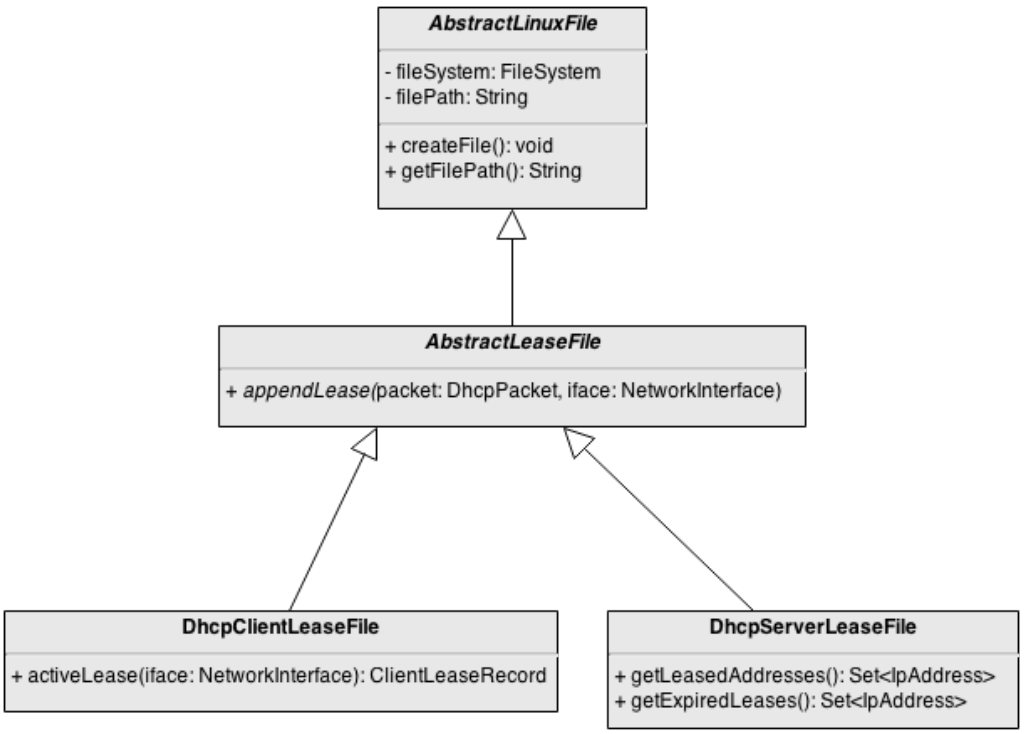
4.1 Konfigurační soubory

Architektura linuxových souborů je zkráceně popsána třídním diagramem na obrázku 4.1. Pro zjednodušení nejsou v rámci tohoto diagramu zmíněny jednotlivé soubory pro konfiguraci aplikací, ani soubory pro ukládání informací o překladu jmenných adres. Všechny třídy reprezentující tyto soubory jsou přímým potomkem třídy `AbstractLinuxFile`.

Třída `AbstractLinuxFile` slouží jako základní prvek celé této architektury a představuje jakýkoli soubor v souborovém systému linuxového počítače. Odkaz na příslušný souborový systém je zadán v proměnné filesystem. Tato třída umožňuje v rámci metody `createFile()` vytvořit na zařízení prázdný soubor. Přesné umístění tohoto souboru je pak dáno proměnnou `filePath`, která je specifikována v konstruktorech jednotlivých souborových tříd. Pokud na tomto umístění soubor už existuje, bude přepsán.

Její potomek `AbstractLeaseFile` potom představuje soubory, které v rámci DHCP protokolu slouží pro uložení záznamů o poskytnutých adresách. Skrze metodu `appendLease()` poskytují potomci této třídy aplikacím DHCP serveru a klienta rozhraní pro přidání těchto záznamů.

V původní verzi simulátoru nebylo možné se soubory manipulovat v módu append. Ten určuje, že při zápisu bude nový text vkládán na konec souboru namísto na začátek. To znamená, že pro přidání záznamu do virtuálního souboru bylo nutné načíst původní obsah tohoto souboru, připojit k němu



Obrázek 4.1: Zjednodušená architektura konfiguračních souborů

nový záznam a celý takto spojený obsah znovu zapsat. V nové verzi už je možné v rámci metody FileSystem.runOutputFileJob při práci se souborem určit, v jakém módu se má otevřít.

4.1.1 DHCP záznamy

Pro uchování informací o úspěšně proběhnutých transakcích si DHCP server i klient ukládají data do vlastních lease souborů.

Jako rozhraní ke klientskému souboru *dhclient.leases*, který obsahuje záznamy přidělených konfigurací slouží třída `DhcpClientLeaseFile`. Kromě možnosti přidávat nové záznamy je pro tuto třídu důležitá metoda getActiveLease(), která má za úkol nalézt pro dané rozhraní platný záznam o přidělení dynamické adresy. Tato metoda je aplikací DHCP klienta volána při spuštění simulátoru. Pokud v rámci frontendu nebyla nastavena pro rozhraní statická IP adresa, je zjišťováno, zda z doby posledního běhu simulace není stále v platnosti záznam, ze kterého by mohla být získána příslušná konfigurace.

Pro obdobu této funkcionality na straně serveru je implementována třída `DhcpServerLeaseFile`. Tato třída nabízí rozhraní pro získání množiny poskytnutých adres, jejichž doba platnosti ještě nevypršela. S pomocí tohoto seznamu se potom server rozhoduje, které adresy může nabízet. O tomto rozhodování je možné dočíst se více v sekci 4.5.

4.2 Aplikační vrstva

Aplikační vrstva slouží ke komunikaci mezi jednotlivými aplikacemi v rámci jednoho zařízení. V simulátoru je reprezentována třídou `ApplicationLayer`.

Aplikace, které přímo nespouští uživatel, ale od kterých je požadováno, aby běžely hned po startu simulace, jsou spouštěny právě v rámci této vrstvy. Příkladem takové služby je klientský DHCP proces, který při svém startu obstarává načtení případné platné dynamické konfigurace.

Při spuštění programu tato vrstva dále obstarává vytvoření virtuálních souborů, u kterých se při běhu programu očekává jejich existence. Pro aplikace na příslušném zařízení potom obsahuje na tyto soubory odkaz.

O samotné předávání požadavků přijatých z nižších vrstev jednotlivým aplikacím se stará již implementovaná transportní vrstva. Tento proces funguje tak, že je zjištěno, na jaký port je příchozí paket určen. Pokud na daném portu nějaká aplikace naslouchá, je prostřednictvím jejího příchozího bufferu požadavek předán a následně je probuzeno vlákno této aplikace.

4.3 Služba networking

Služba networking zajišťuje načítání konfigurací jednotlivých síťových rozhraní virtuálního zařízení z konfiguračního souboru. Její ovládání probíhá prostřednictvím příkazu *service*. Při každém spuštění této služby dojde k načtení příslušných informací ze souboru *interfaces*. Pokud neexistuje, je vytvořen a doplněn údaji, které odpovídají současnému stavu síťových rozhraní. Při načtení validních informací o staticky nastaveném rozhraní provede služba požadované změny. V případě dynamického nastavení je zaslán požadavek o přiřazení adresy na aplikaci DHCP klienta. Po dokončení konfigurace je z důvodu synchronizace s frontendem simulátoru proveden zápis současného stavu sítě do XML souboru obsahujícím schéma dané sítě.

4.4 DHCP klient

Klientská část protokolu se skládá ze dvou komponent. První z nich jsou pracovní vlákna vytvořená pro každé síťové rozhraní na zařízení. Druhou je pak správcovský proces, který těmto vláknům předává příchozí požadavky a také zprostředkovává přístup k souboru *dhclient.leases*.

Toto rozdělení představuje jednoduchý způsob, jak spravovat nově přiřazené dynamické konfigurace na jednotlivých rozhraních. Po vypršení doby platnosti záznamu je vlákno obsluhující příslušné rozhraní probuzeno budíkem aplikace. Takto probuzené vlákno je následně zodpovědné za upravení nastavení na rozhraní.

4.4.1 Manažer vláken

Manažerský proces klienta je reprezentovaný třídou `DhcpClient`. K jeho spuštění dochází při startu simulátoru. Při svém vytvoření najde všechna aktivní rozhraní nacházející se na daném virtuálním zařízení a pro každé toto rozhraní vytvoří uspávací vlákno. V momentě, kdy vlákna existují, začíná manažer naslouchat standardně na portu 68 a veškeré příchozí požadavky předává příslušným vláknům.

Pokud od aplikace na zařízení přijde žádost o přiřazení dynamické adresy, vybere vlákno rozhraní, pro které je tento požadavek zamýšlen a toto vlákno probudí.

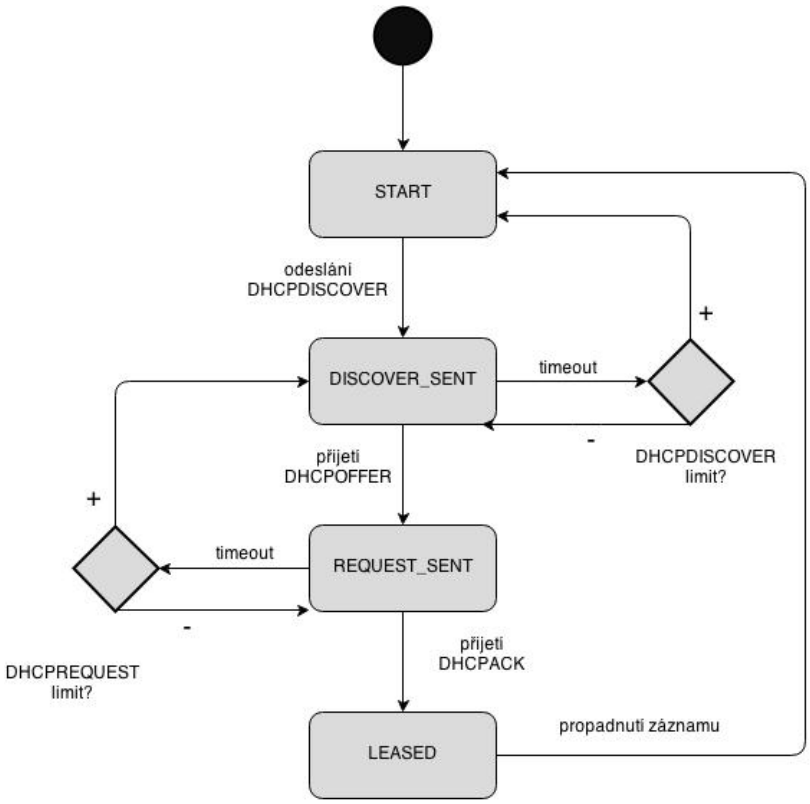
Pro přijatý DHCP paket typu OFFER nebo ACK vyhledá takové vlákno, které tuto odpověď očekává a paket mu předá. Z nižších vrstev se k samotné aplikaci dostane jen paket, který je určen pro zařízení, na němž se tato aplikace nachází. Pokud tedy příchozí paket není očekáván žádným rozhraním, je zahozen.

4.4.2 Klientské vlákno

Vlákno pro síťové rozhraní, pro které mu bylo přiděleno, obstarává celý proces dynamického přiřazení konfigurace. Tento proces je popsán na obrázku 4.2 a sestává ze 4 fází: počáteční fáze, hledání serveru, požadování adresy a přijímání adresy.

4.4.2.1 Počáteční stav

Tento stav je v simulátoru označený jako `START` a vlákno se v něm ocitá hned po svém vytvoření. Vlákno se nejprve ze souboru *dhclient.leases* pokusí



Obrázek 4.2: Diagram běhu klientského DHCP vlákna

načíst platnou konfiguraci pro své rozhraní. Pokud žádnou nenajde, považuje rozhraní za staticky nastavené a uspí se. V opačném případě vlákno provede případné změny v nastavení rozhraní i směrovací tabulky a přepne do stavu `LEASED`. Před svým uspáním ještě zkontroluje dobu vypršení platnosti záznamu a podle ní nastaví systémový budík na své probuzení.

4.4.2.2 Hledání serveru

Prostřednictvím aplikační služby networking nebo přes příkaz dhclient může být vyslán požadavek o přidělení dynamické konfigurace. V případě existence platného záznamu v souboru *dhclient.leases* dojde při konfiguraci k jeho použití. Jinak odešle vlákno skrze své přidělené rozhraní broadcastovou zprávu DHCPDISCOVER a přepne se do stavu `DISCOVER-SENT`. Nastavení tohoto stavu signalizuje manažerskému procesu, že vlákno nyní přijímá zprávy typu DHCPOFFER. Dále pak nastaví systémový budík na své probuzení za dobu uplynutí timeout intervalu. Pokud do této doby dojde k probuzení vlákna příchozí zprávou, přejde do fáze požadování adresy.

V případě, že do uplynutí timeout intervalu nepřijde odpověď od žádného DHCP serveru, snaží se klient požadavek opakovat. Na počet opakovaných žádostí je nastaven limit, při jehož dosažení se klient vzdá snahy o přidělení konfigurace a přechází opět do stavu `START`.

Příkaz dhclient může spolu s požadavkem předat vláknu i odkaz na shell, ze kterého byl spuštěn. V tom případě bude pracovní vlákno do tohoto shellu zapisovat průběh celé komunikace. Pokud byl proces DHCP klienta zavolán v rámci služby networking, dochází k tomuto zápisu vždy.

4.4.2.3 Požadování adresy

Pokud klient úspěšně obdržel ze strany serveru nabídku k přidělení konfigurace, uloží přijaté informace a odpovídá opět na broadcast zprávou DHCPREQUEST. Do této zprávy zahrne i identifikátor serveru, který nabídku odeslal. Zároveň si nastaví stav na `REQUEST-SENT`. Tímto oznamuje manažerskému procesu, že přijímá zprávy potvrzující přidělení konfigurace. Podobně jako u předchozí fáze je potom nastaven timeout interval, po jehož uplynutí klient opakovaně vysílá zprávu DHCPREQUEST. Pokud dojde k dosažení limitu těchto odeslaných zpráv, klient se vrací znovu do fáze hledání serveru.

4.4.2.4 Přijetí konfigurace

Tato fáze nastává v případě, když klient ve stavu `REQUEST-SENT` obdržel zprávu DHCPACK. Nejprve je kontrolováno, že údaje v této zprávě odpovídají údajům obsaženým v původní nabídce ze strany serveru a následně je přikročeno k samotnému nastavení adresy rozhraní, v případě výskytu informací o routerech na síti i k úpravě směrovací tabulky. Pokud mezi přijatými parametry jsou i adresy jmenných serverů pro danou síť, je jejich seznam uložen do souboru *hosts*.

Klientské vlákno nastaví svůj stav na `LEASED`. Tím oznámí manažerskému procesu, že síťové rozhraní je úspěšně nastaveno, a vlákno nepotřebuje přijímat žádné další nabídky.

Před svým uspáním ještě nastaví systémový budík pro probuzení za dobu vypršení platnosti přijaté konfigurace.

4.5 DHCP server

Celá aplikace DHCP serveru běží v rámci jediného vlákna. Při jeho spuštění si server nahraje dostupné konfigurace do paměti. Pro příchozí požadavky je potom velmi rychle schopen určit, jestli bude moci požadavek obsloužit,

a jaké parametry bude případně poskytovat. Ani při zpracování většího množství požadavků potom nedochází k výraznější prodlevě.

Načtená konfigurace na straně serveru má podobu hashmapy. Klíčem je zde adresa podsítě, pro kterou jsou konfigurace definovány. Hodnotou je potom seznam všech adres, které je pro tuto podsíť možné přídělit. Při přijetí paketu typu DHCPDISCOVER server zjistí, pro jakou podsíť je požadavek zamýšlen. Pokud by při přijetí požadavku byl pro požadovanou podsíť tento seznam adres prázdný, pokouší se server nejprve tento seznam aktualizovat. To probíhá porovnáním definovaných adres ze souboru *dhcpd.conf* a přidělených adres ze souboru *dhcpd.leases*, jejichž doba platnosti ještě nevypršela. Pokud by ani po aktualizaci neexistovala pro požadavek volná adresa, je paket požadavku zahozen.

Pokud je možné požadavek typu DHCPDISCOVER obsloužit, odešle se klientovi příslušná nabídka. Zároveň si server uloží MAC adresu klienta, kterou použije k rezervaci nabídnutého záznamu. Pro přijatý DHCPREQUEST paket se server přesvědčí, zda se MAC adresa žádajícího klienta objevuje mezi rezervacemi. Pokud ano, je odeslána zpráva DHCPACK, která obsahuje původní rezervovanou konfiguraci. Pokud se MAC adresa klienta posílajícího DHCPREQUEST neobjevuje v seznamu rezervací, je paket zahozen.a

4.6 DNS resolver

Klientská část protokolu DNS je implementována ve třídě `DnsResolver`. Tato třída od ostatních aplikacích na zařízení přijímá požadavky na překlad jmenných adres na číselné. Pro jeho nalezení nejprve dojde k prohledání souboru *hosts*.

Pokud tento soubor neexistuje, nebo v něm pro dané jméno nebylo nalezeno žádné mapování, dojde ke kontrole, že text argumentu je platné doménové jméno. V případě úspěchu bude požadavek na překlad poslán na jmenný server běžící na zařízení, případně na server, jehož adresa je uvedena v souboru *resolv.conf*. Pokud by dotaz neobsahoval validní doménové jméno, nebo pokud není dostupná adresa žádného jmenného serveru, je aplikaci ohlášen neúspěch.

Při poslání požadavku na vzdálený server je nastaven timeout čekání na odpověď. Po jeho dosažení se resolver pokouší dotazovat na další ze serverů ze souboru *resolv.conf*. Také je možné, že DNS server sám nezná odpověď, ale může poskytnout adresu jiného serveru, který obsahuje databázi o doméně obsažené v původním dotazu.

V momentě, kdy je překlad adresy nalezen, nebo pokud už není komu zaslat dotaz, je aplikaci ohlášen výsledek hledání.

4.6.1 Rozšíření linuxových příkazů ping a traceroute

V minulé verzi simulátoru byla jako platný argument pro příkazy ping a traceroute brána pouze validní IP adresa. Po implementaci DNS protokolu byly tyto příkazy rozšířeny pro práci se jmennými adresami. V případě zadání číselné IP adresy zůstává průběh vykonání příkazu stejný jako v původní implementaci. Pokud však argument nebude rozpoznán jako validní číselná IP adresa, bude průběh příkazu přerušen a na DNS resolver bude zaslán požadavek o překlad.

4.7 DNS server

Zatímco DHCP server je implementován v rámci jednoho vlákna, DNS server je realizován jako vícevláknová aplikace. Narozdíl od DHCP serveru totiž nejsou záznamy při startu načítány do paměti, ale jsou uchovány v rámci zónových souborů. Pokud by tedy server byl implementován jednovláknově, vznikala by mezi zpracováním více požadavků delší prodleva.

Vlákno serveru, které požadavek zpracovává, se nejprve ujistí, že přijatý paket se dotazuje na validní doménové jméno. Pokud by dotaz kontrolou neprošel, posílá server tento paket zpátky. Před samotným posláním změní typ paketu na odpověď a nastaví mu příznak *format error*. V opačném případě server prohledá svůj katalog zónových souborů. Při nalezení souboru pro doménu odpovídající požadavku dojde prostřednictvím třídy `DnsZoneFile` k prohledání tohoto souboru. Nyní mohou nastat tři případy:

1. V souboru bude přímo jedna nebo více odpovědí. Ty budou vloženy do přijatého paketu do sekce ANSWER. Dále mohou být vloženy další informace o doméně.

2. V souboru nebude přimo uvedena odpověď na dotaz, ale bude se zde vyskytovat odkaz na jiný jmenný server, který by tuto odpověď mohl znát. Do přijatého paketu se tedy vloží tento odkaz.

3. Soubor neobsahuje ani odpověď ani odkaz na odkaz na jiný jmenný server. Do přijatého paketu se nevkládá nic.

Následně bude typ paketu nastaven jako odpověď a poslán zpět klientovi.

Kapitola 5

Testování

5.1 Jednotkové testy

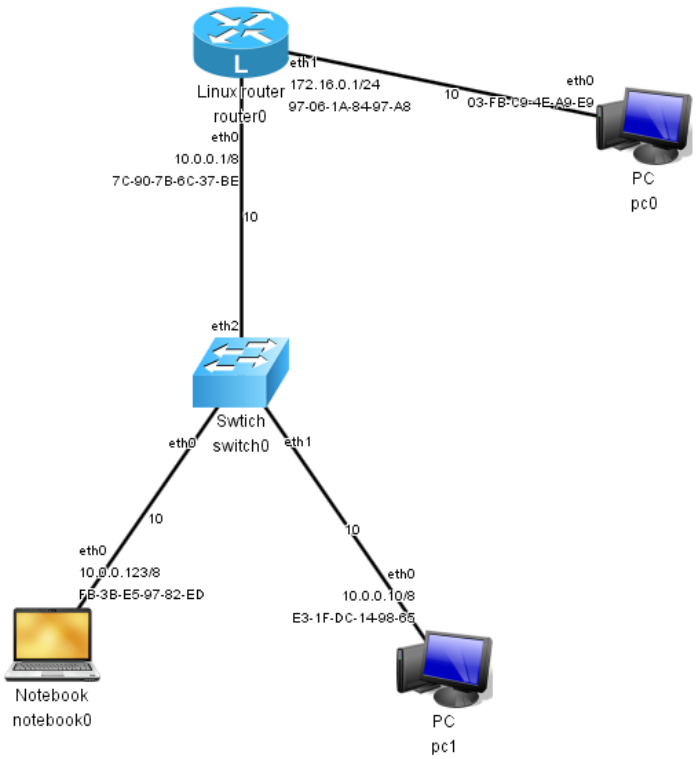
Během implementace byly s použitím knihovny JUnit vytvářeny jednotkové testy pro kontrolu některých tříd a metod. Testovány byly především třídy poskytující rozhraní přístupu k virtuálním souborům. Byly vytvořeny testovací virtuální soubory a následně se kontrolovalo, zda metody vyčtou ze souborů správné záznamy, případně že se zachovají správně při práci s prázdnými nebo neexistujícími soubory.

Aplikace v simulátoru obsahují velké množství komponent, které jsou na sebe při běhu navzájem závislé. Proto na jejich jednotkové testování nebyl kladen takový důraz.

5.2 Uživatelské testování

Uživatelské testování proběhlo dne 6.5.2014 v rámci dvou laboratorních cvičení předmětu BI-PSI a zúčastnilo se ho 21 studentů tohoto předmětu. K testování byly použity notebooky se systémy Windows 7 a 8, MacOS a různými linuxovými distribucemi.

Na začátku testování byl studentům zpřístupněn projekt s předvytvořenou sítí a dotazník obsahující instrukce. Do těchto dotazníků pak mohli zaznamenávat chyby a připomínky k programu. Během celého testování jsme také já a kolega Václav Mach byli dostupní pro řešení problémů přímo na místě.



Obrázek 5.1: Schéma sítě použité pro uživatelské testování

5.2.1 Zadané úlohy

Testovací síť je zobrazena na obrázku 5.1. Zadané úlohy pro moji část implementace byly:

1. Na pc1 změňte pro rozhraní eth0 adresu s pomocí souboru /etc/network/interfaces
2. Na router0 přidejte pro podsíť 10.0.0.0 informaci o nameserveru běžícím na pc1. Přidejte rozsah přidělitelný adres pro podsíť 172.16.0.0
3. Vyzkoušejte přiřazení dynamické adresy příkazem dhclient i skrze soubor /etc/network/interfaces
4. Na pc1 přidejte pro zónu example.com mapování pro www.example.com. Funkčnost ověřte příkazem ping.

Úlohy týkající se propojení s reálnou sítí:

1. Na router0 přidejte další rozhraní a připojte k němu komponentu reálného počítače. Nastavte statickou cestu tak, aby výstupní rozhraní bylo směrem k reálnému počítači.
2. Zajistěte svázání portu router0 a rozhraní hostitelského počítače pomocí příkazu rnetconn.
3. Vyzkoušejte spojení reálné a virtuální sítě příkazem ping.

5.2.2 Vyhodnocení

V této části se věnuji problémům týkající se mých rozšíření simulátoru.

1. DNS server nezjistí informace o mapování adres z vlastní lokální databáze

   Chování bylo opraveno.

2. Nápověda ke konfiguračním souborům by mělo obsahovat výpis možných parametrů nastavení

   Při testování bylo seznam implementovaných parametrů vypsán v přiloženém manuálu. V nové verzi budou přidány okomentované vzory pro konfigurační soubory.

3. Práce s textovým editorem je pomalá a nepohodlná

   Editor byl vyvíjen v původní aplikace. Pro editaci delších souborů není nejvhodnější. Možný námět pro rozšíření simulátoru.

4. Příkaz dhclient během komunikace nevypisuje na terminál žádné informace.

   Jedná se o výchozí chování příkazu. Je možné zapnout verbose mode spouštěním s přepínačem -v.

5. Použitelný je pouze příkaz service, neexistují odpovídající init skripty.

   Pro přidání init skriptu by bylo nutné přidat lepší podporu skriptování do simulátoru. Možný námět pro rozšíření do budoucnosti.

Kapitola 6

Návrhy na vylepšení

Aplikaci Psimulator2 je i nadále možné rozšířovat. Mezi možná vylepšení patří:

- Rozšíření implementovaného DNS protokolu o možnost zpracování reverzního dotazu.
- Existující vestavěný textový editor zahrnuje jenom nejzákladnější funkce a při práci s delšími soubory působí neohrabaně. Bylo by vhodné ho rozšířit např. o více možností pohybu v editovaném textu nebo přídání fuknce vyhledávání v textu.
- DHCP protokol by bylo možné implementovat i pro Cisco zařízení.
- Přidání aplikací a příkazů blíže pracujících s DNS protokolem, např. dig.
- Přidání implementace jednoduchého HTTP serveru, včetně možnosti jeho konfigurace v rámci konfiguračních souborů.

Závěr

V rámci této bakalářské práce se mi podařilo navrhnout a realizovat rozšíření pro existující simulátor počítačových sítí, která splňují zadané požadavky. Na virtuálních linuxových zařízeních přibylo nastavení síťových rozhraní prostřednictvím konfiguračních souborů. Dále byly pro tato zařízení implementovány protokoly DHCP a DNS, včetně možnosti použít k jejich konfiguraci virtuální soubory.

Kolegovi Václavu Machovi se rovněž povedlo splnit požadavky své práce. Výsledný program i s našimi rozšířeními byl úspěšně otestován uživateli, což prokázalo, že je možné jej nasadit v rámci výuky předmětu BI=PSI na Fakultě informačních technologií.

Největším osobním přínosem pro mne bylo získání zkušenosti při práci na větším, již existujícím projektu. Dalším přínosem bylo pochopení důležitosti systematického postupu při tvorbě rozsáhlejších prací. Během implementace jsem také probloubil svou znalost jazyka Java a z blízka se seznámil s pokročilejšími funkcemi vývojového prostředí NetBeans.

Literatura

[1] dhcpd.conf(5), Linux man page: dhcpd configuration file [cit. 2014-04-08]. Dostupné z WWW: <http://linux.die.net/man/5/dhcpd.conf>

[2] dhcpd(8), Linux man page: Dynamic Host Configuration Protocol Server [cit. 2014-04-08]. Dostupné z WWW: <http://linux.die.net/man/8/dhcpd>

[3] Mach, V.: Vizualizace virtuální počítačové sítě: Bakalářská práce. České vysoké učení technické v Praze, 2014.

[4] Network Working Group: RFC 1035: Domain Implementation and Specification. [cit. 2014-04-10]. Dostupné z WWW: <http://tools.ietf.org/html/rfc1035#page-25>

[5] Network Working Group: RFC 2131: Dynamic Host Configuration Protocol. [cit. 2014-04-10]. Dostupné z WWW: <http://tools.ietf.org/html/rfc2131#page-9>

[6] Lukáš, M.: Podpurné komponenty simulátoru pocítacové síte: Diplomová práce. České vysoké učení technické v Praze, 2012.

[7] Pitřinec, T., Síťový simulátor pro výukové účely na bázi prvků OS Linux: Diplomová práce. České vysoké učení technické v Praze, 2012.

[8] Řehák S., Síťový simulátor pro výukové účely na bázi směrovačů CISCO: Diplomová práce. České vysoké učení technické v Praze, 2012.

[9] Švihlík, M.: Vizualizace virtuální počítačové sítě: Diplomová práce. České vysoké učení technické v Praze, 2012.

Příloha A

Seznam použitých zkratek

BIND Berkeley Internet Name Domain

DHCP Dynamic Host Configuration Protocol

DNS Domain Name System

IDE Integrated Development Environment

IOS Internetwork Operating System

IP Internet Protocol

ISC Internet Systems Consortium

MAC Media Access Control

OS Operating System

PSI Počítačové sítě

RFC Request For Comments

XML Extensible markup language

Příloha B

Instalační příručka

B.1 Minimální požadavky

- Java Runtime Environment version 7+

  http://www.oracle.com/technetwork/java/javase/downloads

- Pro propojování virtuální a reálné sítě je nutná knihovna libpcap (winpcap na OS Windows). Instalační balíčky jsou součástí projektového archívu

B.2 Spuštění aplikace

K získání aplikace může kromě přiloženého CD sloužit i stránka projektu https://sourceforge.net/projects/psimulator2/.

Po rozbalení archívu projektu slouží ke spuštění frontendu aplikace soubor **psimulator2_frontend.jar**.

Pro zjištění podrobnějších informací o ovládání programu slouží manuál přibalený v archívu.

Příloha C

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `dist` ..... adresář se spustitelnou formou implementace
- `src` ..... zdrojové kódy implementace
- `text` ..... text práce
  - `src` ..... zdrojová forma práce ve formátu LaTeX, obrázky
  - `BP_Horacek_Michal_2014.pdf` ..... text práce ve formátu PDF