České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra telekomunikační techniky

Bakalářská práce

Simulace bezdrátových průmyslových sítí s nízkou propustností

Ondřej Šulc

Studijní program: Komunikace, multimédia a elektronika
Studijní obor: Síťové a informační technologie
Vedoucí práce: Ing. Ondřej Vondrouš

Praha, Květen 2016

České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra telekomunikační techniky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: **Ondřej Šulc**

Studijní program: Komunikace, multimédia a elektronika
Obor: Síťové a informační technologie

Název tématu: **Simulace bezdrátových průmyslových sítí s nízkou propustností**

Pokyny pro vypracování:

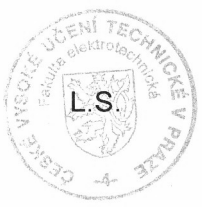
Seznamte se s principem fungování radiových modemů RipEX a specifickými vlastnostmi úzkopásmových bezdrátových sítí. Současně se seznamte s diskrétním simulačním prostředím OMNeT++ a pracovními rámci INET a INETMANET. Na základě simulačního modelu zařízení RipEX navrhněte a implementujte model sítě a definujte komunikační profily vycházející z komunikace v reálných průmyslových sítích. Výsledky získané simulací porovnejte s výsledky měření na reálné síti a závěry vhodně prezentujte.

Seznam odborné literatury:

[1] Pužmanová, R.: *Moderní komunikační sítě od A do Z*, 2.aktualizované vydání. Brno: Computer Press, 2006, 430 s. ISBN 80-251-1278-0.
[2] Banks, J.: *Discrete-Event System Simulation*. 2nd Ed. Upper Saddle River: Prentice-Hall, 1996, 548 s. ISBN 0-13-217449-9.

Vedoucí: Ing. Ondřej Vondrouš

Platnost zadání: do konce letního semestru 2016/2017



prof. Ing. Boris Šimák, CSc.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 21. 12. 2015

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 27.5.2016

Ondřej Šulc

Poděkování

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Ondřeji Vondoušovi za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci. Dále bych rád poděkoval své rodinně a přátelům za veškerou podporu, kterou jsem od nich nejen během psaní této práce a studia dostal.

Abstrakt

Účelem této práce je seznámení se s principem fungování rádiových modemů RipEX. Práce dále shrnuje dostupné simulační nástroje a jejich možnosti. Soustředí se však zejména na diskrétní simulační prostředí OMNeT++ a jeho pracovní rámce INET a MiXiM. Pro tento program je totiž napsán simulační model modemu RipEX, který je v této práci porovnáván s reálným zařízením. V práci je popsán způsob jakým je RipEX a jeho simulační model porovnán a jsou zde prezentovány výsledky tohoto měření.

Klíčová slova

simulace, OMNeT++, úzkopásmové sítě, rádiomodemy, RipEX, FlowTester

Abstract

This thesis serves as an introduction to radio modem RipEX and its functionality. It also describes available simulation frameworks with focus on discrete simulation environment OMNeT++ and its libraries INET and MiXiM. The environment is used for comparison of RipEX and its simulation model. This thesis describes the process of the comparison and presents the conclusion of this measurement.

Key words

simulation, OMNeT++, narrowband networks, radio modems, RipEX, FlowTester

Obsah

Seznam použitých zkratek

BGP Border Gateway Protocol 7

CLI Command Line Interface *(Příkazová řádka)* 11

COM Communication port 12

CPFSK Continuous-phase frequency-shift keying 2

CSV Comma-separated values 10

GMSK Gaussian Minimum Shift Keying 2

IDE Integrated Development Enviorment 6, 7

IoT Internet of things 1

IPv4 Internet Protocol version 4 7, 17

IPv6 Internet Protocol version 6 7, 17

LDP Label Distribution Protocol 8

MAC Media Access Control 8, 24

MANET Mobile ad hoc Network 7

MPLS Multiprotocol Label Switching 7

NED Network Description 6–8

OSPF Open Shortest Path First 7

P2MP Point-to-multipoint 10

P2P Peer-to-peer 10

PPP Point-to-point Protocol 7

PPTP Point-to-Point Tunneling Protocol 14

RSVP-TE Resource Reservation Protocol - Traffic Engineering 8

RTT Round-trip time 10

SCADA Supervisory Control and Data Acquisition 2, 11

SCTP Stream Control Transmission Protocol 17

SDR Software Defined Radio 2

SSH Secure Shell 11

TCP Transmission Control Protocol 7–9, 16, 17, 23, 24

TCP/IP Transmission Control Protocol/Internet Protocol 7, 8, 10

UDG Universální datový generátor 1, 7–9, 15, 18

UDP User Datagram Protocol 7–10, 16–24

USB Universal Serial Bus 11

VPN Virtual Private Network 14

1 Úvod

Známé české přísloví říká, dvakrát měř a jednou řež. Již dávno před vynálezem počítačů a datových sítí tedy platilo, že než bude jakýkoliv projekt realizován, musí se náležitě vyhodnotit náklady, rizika nebo výkonnost. V případě realizace datových sítí to samozřejmě platí také. Pokud navrhujeme malou síť s malým počtem prvků, je poměrně jednoduché odhadnout, jak bude fungovat bez dalších nástrojů nebo otestovat navrhované řešení v laboratoři. Při návrhu rozsáhlejších sítí se však složitost systému rychle zvyšuje a přestává tak být možné síť před realizací otestovat v laboratoři nebo se spolehnout na vlastní odhad. V takovém situaci existuje možnost využití simulace. V simulaci může být systém vyzkoušen bez nutnosti vlastnit skutečná zařízení a může být téměř libovolně škálován. Jediným limitem je výkon počítačů, na kterých simulace běží. K přípravě podobné simulace je však potřeba dostatečně přesný model simulovaných zařízení.

Tím se dostávám k zadání této práce. V té se zabývám modelem rádiového modemu RipEX od společnosti Racom. Důvod pro zkoumání právě tohoto modemu je jeho použití v senzorových sítích a sítích Internet of things (IoT). Ty jsou v dnešní době velmi aktuální, a tak vzniká mnoho nových návrhů sítí, které je třeba otestovat. Nejlepším řešením se tak jeví testování pomocí simulace. Pro získání věrohodných výsledků je však potřeba nejprve otestovat přesnost modelu, se kterým je simulace prováděna. Přesnost tohoto modelu jsem testoval v simulačním prostředí OMNeT++ a porovnával jsem jej s reálnými přístroji umístěnými v laboratoři. Při testování jsem používal mnoho nástrojů. Kromě samotného simulačního prostředí OMNeT++ to byly jeho rozšiřující knihovny INET a MiXiM a Universální datový generátor (UDG). Abych měl s čím výsledky testování porovnat, musel jsem stejné testy jako v simulaci provést také na reálných modemech RipEX. K tomuto účelu jsem využil zařízení FlowTester.

Struktura této práce je následující. V první části se zabývám úzkopásmovou rádiovou komunikací, jejím využitím v průmyslu a efektivním nakládáním s rádiovým spektrem. V další kapitole se věnuji samotnému procesu simulace. Kapitola je o tom, jak simulace vzniká a co nás k jejímu využití vede. Je v ní popsáno, jaké jsou možnosti simulací, jaké jsou jejich nedostatky a kdy je tedy vhodné simulace využít. Poslední částí této kapitoly je klasifikace druhů simulací a přehled využívaných simulačních nástrojů. Třetí kapitola se věnuje konkrétnímu simulačnímu nástroji, který byl v této práci využit. Tím nástrojem je OMNeT++ a v kapitole je popsáno jeho simulační prostředí a způsob vytváření modelu a běhu simulace. V posledních dvou kapitolách teoretické části je popsán router RipEX a zařízení FlowTester použité pro jeho testování. Modem RipEx je popsán z hlediska fyzických parametrů i programového vybavení a podobně je popsán i Flowtester.

Další část práce se zabývá již samotným měřením. Nejdříve je zde popsán způsob měření. Dále jsou popsány komunikační profily použité pro generování testovacích dat a způsob jejich definince v programu OMNeT++ a na zařízeních FlowTester. Dále jsou zde presentovány výsledky jednotlivých měření v podobě grafů a jejich analýza. Poslední kapitola shrnuje získané poznatky.

2 Rádiové spektrum a úzkopásmové bezdrátové sítě

Neustále se zvyšující požadavky na bezdrátové spektrum, způsobené velkým zájmem o bezdrátovou komunikaci, nás nutí k jeho stále efektivnějšímu využívání. Tato snaha je patrná již od samého počátku rádiového přenosu. Již v této době se dbalo na to, aby vysílání nejen využilo svůj přidělený kanál, ale aby zbytečně nenarušovalo ostatní vysílání.

Jedním ze způsobů, jak efektivně využít přenosové spektrum, je uzpůsobit šířku kanálu přenášenému obsahu. Pokud tedy není aplikace náročná na množství přenesených dat, můžeme využít úzkopásmových sítí. Rádiové modemy využívající úzkopásmovou komunikaci se používají například pro aplikace založené na protokolech Supervisory Control and Data Acquisition (SCADA), monitorování transportu ropy a jiných surovin nebo telemetrii. Tyto modemy většinou operují ve frekvenčním pásmu mezi 30 MHz a 1 GHz a šířkou pásma až 25 kHz. Tato frekvenční pásma jim umožňují komunikaci i na velké vzdálenosti.

Takovéto rádiové modemy musí velmi striktně dodržovat pravidla provozu a fungovat za různých podmínek po celém světě. Je potřeba aby měly vysoký dynamický rozsah, velký útlum energie vysílané v sousedních kanálech, vysokou citlivost a selektivitu. Zároveň však musí mít malé rozměry, malou spotřebu a musí být funkční ve velkém rozpětí teplot. Aby mohly modemy být takto flexibilní, je u nich často využíváno konceptu Software Defined Radio (SDR). Zařízení jsou tak schopna pracovat s různou šířkou pásma, používat různé digitální modulace, vysílat s proměnlivou rychlostí a využívat techniky pro boj s negativními vlivy rádiového kanálu. Podobné systémy většinou používají modulace s konstantní obálkou a skokově neměnnou fází jako jsou Gaussian Minimum Shift Keying (GMSK) a Continuous-phase frequency-shift keying (CPFSK). Skokově neměnná fáze je použitá za účelem vysokého utlumení signálu na sousedních kanálech a zvýšení odolnosti vůči nelinearitě kanálu.

Aby byla plně využita šířka pásma, většinou tato zařízení nepoužívají okruhově orientované přepínání, které je neefektivní, protože neustále zabírá maximální šířku pásma. Místo toho používají packetově orientované přepínání, které je však náročnější na synchronizaci.

3 Simulace

3.1 Motivace pro tvoření modelů a simulací

Simulací v tomto významu slova rozumíme imitaci reálného světa. Simulovat můžeme jakýkoliv proces nebo systém. Abychom mohli provést jakoukoliv simulaci, nezáleží na tom, zda ručně nebo na počítači, tak musíme nejprve vytvořit simulační model. Model představuje napodobeninu simulovaného systému a skládá se ze sady předpokladů a jejich vztahů vyjádřených matematicky, logicky nebo symbolicky. [2] Můžeme simulovat již existující systémy, u kterých zjišťujeme, jakých výsledků dosáhneme různými změnami, nebo simulujeme systémy, které teprve vytvořit chceme a potřebujeme předem odhadnout jejich chování. V obou případech nám však simulační nástroje pomáhají ušetřit čas a peníze oproti testování na reálných systémech, pokud je toto testování vůbec možné, například z bezpečnostních důvodů (jaderné reakce).

Simulace jsou tak velmi populární, nemusíme totiž experiment fyzicky provést, a přesto získáme data odpovídající realitě. Můžeme zkoušet nové procesy bez narušení probíhajících procesů, testovat nové hardwarové kombinace, odhadnout budoucí problémy zrychlením simulačního času, zjistit chování systému při maximálním zatížení a podobně. Díky tomu se dnes simulací využívá v podstatě ve všech odvětvích včetně vědy (předpověď počasí, šíření epidemií), techniky (statika, počítačové systémy), ekonomiky (vývoj cen), výuky (pochopení fungování systémů) a zábavy (letecké simulátory, filmové efekty).

Při zvažování zda je simulace vhodným nástrojem je vždy vhodné si ověřit, zda neexistuje jednodušší způsob jak odhadnout dopad změn na systém. Mnohdy je výhodnější provést experiment v reálném světě, ověřit řešení analyticky pomocí matematických vztahů, nebo jen myšlenkovým pochodem za použití selského rozumu. Dále je vždy nutné si uvědomit, zda náklady vynaložené na simulaci nepřevyšují náklady ušetřené danou změnou. Při návrhu také můžeme zjistit, že nedovedeme zajistit dostatečnou přesnost modelu anebo se model může stát příliš složitým pro běh simulace i na nejvýkonnějších počítačích. V takových případech také není vhodné jít cestou simulace.

3.2 Simulační systémy a jejich klasifikace

Abychom mohli modelovat určitý systém, musíme mu nejprve porozumět. Systém se skládá z prvků, pomocí nichž se s snaží dosáhnout určitého cíle. Příkladem může být továrna, kde pracovníci, stroje a díly jsou prvky a dohromady produkují výsledný výrobek. Každý z prvků má své vlastnosti a může vykonávat činnosti, které zaberou určitou dobu. Množina prvků jednoho systému může zároveň být podmnožinou prvků systému komplexnějšího. Stav systému lze definovat jako souhrn všech jeho prvků a jejich vlastností potřebných k jeho popisu. V případě naší továrny to může být počet pracovníků a strojů a jejich schopnost vyrábět výrobek nebo poruchovost. Stav systému se změní nějakou událostí. Ta může být interního charakteru – porucha stroje, nebo charakteru vnějšího – nová objednávka.

Systémy můžeme klasifikovat jako stochastické nebo deterministické. [10] deterministických systémů, známe všechny vstupní veličiny a simulací tedy dostaneme konkrétní výstupní veličiny. I jedna jediná vstupní veličina, která se se nechová deterministicky, nám systém

změní na stochastický. U nedeterministických systémů nedostaneme konkrétní výsledky, protože z náhodných vstupních veličin nemůžeme dostat jiné než náhodné veličiny výstupní. Výsledky simulací stochastických systémů jsou tak jen odhady na základě statistických modelů vstupních veličin.

Dále můžeme systémy klasifikovat jako diskrétní nebo spojité. V praxi je však jen velmi málo systémů striktně spojitých nebo diskrétních. Většinou však lze určit, zda jsou změny v systému převážně diskrétní nebo spojité a podle toho ho charakterizovat. Příklad spojitého systému může být například průběžně se měnící výška hladiny u přehrady. Naopak délka fronty paketů čekajících na odeslání se bude vždy měnit o celý paket jen ve chvíli jeho přijetí nebo odeslání a bude tedy diskrétní. A protože se v této práci budu zabývat simulací digitální komunikace, budu se dále věnovat jen diskrétním systémům a jejich simulacím.

Simulace diskrétních systémů je tedy modelování systémů, jejichž stav se mění jen v konkrétních časových okamžicích. Pro tvorbu modelů takovýchto systémů používáme numerické metody. Neřešíme je tedy analyticky, ale za použití matematických modelů a do kterých dosazujeme konkrétní numerické hodnoty. Nedostaneme se tak ke konkrétnímu výsledku, ale vytváříme a zaznamenáváme modelovanou historii. Tu potom analyzujeme a získáme tak odhad toho jak by se choval systém skutečný.

3.3 Postup pro vytvoření simulace

Při vytváření modelu a následné simulace můžeme rozlišovat několik etap procesu. Nejdříve je potřeba formulovat problém a očekávané přínosy simulace na jeho řešení. V této fázi je třeba zhodnotit, zda se simulace vyplatí a zda je vůbec možná. V další etapě je vytvořen model. Model je abstrakcí reality a vždy je potřeba rozeznat základní vlastnosti systému a zachytit jejich esenci v modelu. Model by měl být dostatečně komplexní, aby vystihl realitu, ale přílišná komplexnost není žádoucí, protože jen zabírá více času při modelování i samotné simulaci. Po vytvoření tohoto abstraktní modelu je ho třeba zapsat formou programu. Vzniká tak simulační model.

Před samotnou simulací je ještě potřeba ověřit zda výsledný model koresponduje s realitou a zda je korektně zapsán v simulačním programu. V mnoha případech není, a pak je nutné ho vylepšit a opakovat tento proces dokud se model nestane validním. Následuje etapa simulace. Dle navržených scénářů, které definují různé alternativy pro běh, je simulace provedena a její výsledky zaznamenány. Dle následné analýzy jsou vytvořeny předběžné závěry a je rozhodnuto, zda je potřeba odsimulovat další alternativy.

Po analýze všech výsledků je vytvořena dokumentace jak samotných výsledků, tak i průběhu simulace a samotného procesu vytváření modelu. Dokumentace výsledků je důležitá pro jejich následné využití v praxi a dokumentace procesu simulace je důležitá v případě nutnosti simulaci zopakovat nebo pro potvrzení její validity.

3.4 Nástroje pro tvorbu modelů a simulace

V dnešní době je naprostá většina simulací dělána na počítačích. Obecně existují tři postupy pro tvorbu simulací. Lze použít buďto obecné programovací jazyky jako je Java nebo C++, nebo můžeme použít nějaký programovací jazyk uzpůsobený pro potřeby simulace, např.

Modelica. Poslední možností je využití specializovaných programů.

Hlavní nevýhodou použití obecného programovacího jazyka je nutnost programovat vše od začátku. Tento způsob zabírá hodně času a hlavně je náchylný na tvorbu chyb. Proto se dnes tento způsob již téměř nevyužívá. Výhodou ale je, že naprogramování alespoň jednoduché simulace touto metodou nám pomůže porozumět konceptům a algoritmům používaným při simulacích. Zároveň jsou tyto jazyky hojně rozšířené, a proto vzniká mnoho knihoven určených pro použití v simulacích, které eliminují potřebu programovat vždy vše znovu od začátku. Při sloučení takovýchto knihoven a přidáním grafického prostředí může vzniknout komplexní simulační nástroj jako je OMNeT++.

Další možností je využití speciálního programovacího jazyka. Takovéto speciální programovací jazyky usnadňují zápis struktury a chování modelů a simulačních experimentů. Přidávají také možnost automatické kontroly správného zápisu modelu. Jejich nevýhodou je malá rozšířenost, která vede k menší podpoře a pomalejšímu vývoji. V případě, že nechceme nebo neumíme programovat, můžeme využít specializovaných programů. Takovéto programy se většinou soustředí na konkrétní oblast simulace a nabízí grafické prostředí. Jejich použití je tak většinou jednodušší, ale daní za tuto jednoduchost bývá jednoúčelovost. Příkladem takovýchto programů může být například Spice pro simulaci obvodů.

4 OMNeT++

OMNeT++ je objektově orientovaný modulární síťový simulační rámec, který pracuje v diskrétním čase.[8] Není to specializovaný program jako již zmíněný Spice, ale obecný simulační nástroj, který poskytuje infrastrukturu pro simulace jakýkoliv diskrétním událostí. Může být požit pro simulaci sítí metalických nebo bezdrátových sítí, protokolů nebo i hardware a software, který nemá se sítěmi nic společného.

4.1 Struktura simulací

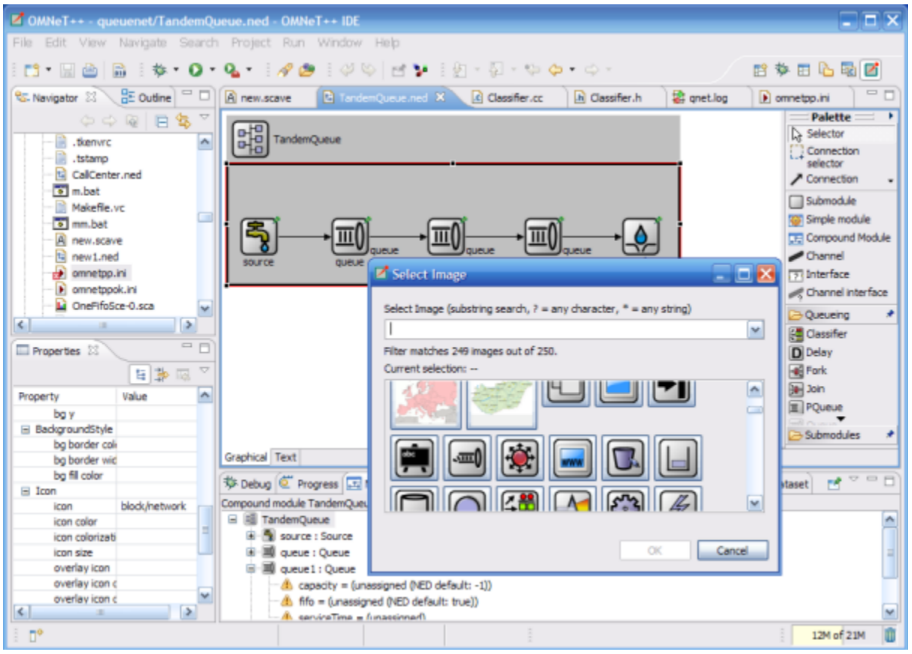
Aby mohl by být OMNeT++ takto univerzální pracuje s moduly. Moduly jsou komponenty, ze kterých lze složit modely pro potřeby simulace podobně, jako lze z jednotlivých dílu sestavit plastikový model letadla. Jednotlivé moduly mohou být velmi obecné a využitelné opakovaně v mnoha různých simulacích nebo specializované pro konkrétní účel. [7]

Moduly jsou psány v jazyce C++ (to je také důvod pro ++ v názvu). Rozlišujeme je na jednoduché a složené. Složené moduly vznikají jak z modulů jednoduchých, tak z již složených modulů a mohou tak být neomezeně vnořovány do složitějších struktur. Již kompletní model je v OMNeTu nazýván sítí a ve skutečnosti je to stále jen složený modul. Každý modul je instancí modulového typu. Modulové typy mohou být uloženy odděleně a vytvářet komplexní knihovny, ze kterých mohou uživatelé čerpat.

Jednotlivé moduly mezi sebou komunikují pomocí bran. Brány tak slouží jako rozhraní pro spojení modulů a přenos zpráv mezi nimi. Tyto zprávy mohou obsahovat jak data, tak další atributy. Spojení lze realizovat pouze mezi moduly na stejné hierarchické úrovni. To znamená, že není možné z vnějšku komunikovat přímo s modulem vnořeným do modulu jiného. Toto omezení umožňuje opakované použití modulů. Spojení mohou být použita jak jako propojení funkčních celků, tak jako model fyzického propojení v datové síti. Každému proto můžeme přiřadit určité parametry a lze mu tak nastavit např. přenosovou rychlost, zpoždění nebo chybovost. Tyto a další parametry, kterými můžeme upravit chování jednotlivých spojení, se nastavují v souborech typu Network Description (NED), kde navrhujeme strukturu modelu a jeho propojení. Dalším souborem pro konfiguraci simulace je omnet.ini. Ten slouží pro nastavení parametrů konkrétního běhu simulace.

4.2 Simulační prostředí

Novější verze OMNeTu++ obsahují vývojové prostředí založené na prostředí Eclipse. Úcel tohoto Integrated Development Enviorment (IDE) je poskytnout uživateli prostředí, které je vhodné pro úpravu modelů psaných v jazyce C++, popisu sítě v jazyce NED i dalších konfiguračních souborů jako omnet.ini na jednom místě. Kromě standartních funkcí IDE jako je zvýrazňování syntaxe, můžeme využít i možnost grafické úpravy souborů NED místo psaní zdrojového kódu. [6]



Obrázek 1: Omnet++ IDE, editace souboru NED [6]

Dále lze přímo v simulačním prostředí vizualizovat a následně vyhodnocovat samotnou simulaci. Při běhu simulace máme na výběr mezi plnou vizualizací, kde je každá událost graficky názorně zobrazena na diagramu simulované sítě. Nebo můžeme sledovat jen log těchto událostí nebo necháme simulaci proběhnout celou bez zobrazování průběžných informací. Poslední možnost je samozřejmě z hlediska výpočetního výkonu nejméně náročná, a proto při složitějších a hlavně opakovaných simulacích preferovaná.
Poslední činnost, se kterou nám může simulační prostředí pomoci, je analýza výsledků. Přímo v prostředí tak můžeme výsledky zapsané v textových souborech (.vec pro vektorové veličiny a .sca pro skalární) zpracovat a získat tak grafy a tabulky výsledných hodnot. Tyto jsou pak uloženy v souboru s koncovkou .anf. Jedna z výhod OMNeTu je opakovatelnost pokusů. Při generování náhodných proměnných totiž používá pseudonáhodný generátor a hodnoty si pamatuje až do ukončení programu. Pokud tedy simulační program nevypneme dostaneme při stejné simulaci vždy stejné výsledky.

4.3 Knihovny rozšiřující Omnet++

4.3.1 Inet

INET je knihovna modelů rozšiřující možnosti OMNeT++. Přidává modely pro protokoly, síťové prvky a další modely, které se hodí pro simulaci komunikačních sítí. Používá se především při testování, návrhu a validaci nových protokolů a síťových konfigurací. [4]
INET přidává modely pro rodinu protokolů Transmission Control Protocol/Internet Protocol (TCP/IP) (TCP, UDP, Internet Protocol version 4 (IPv4), Internet Protocol version 6 (IPv6), Open Shortest Path First (OSPF), Border Gateway Protocol (BGP) a podobně), linkové protokoly pro metalické i bezdrátové sítě (Ethernet, Point-to-point Protocol (PPP), IEEE 802.11 a podobně), protokoly Mobile ad hoc Network (MANET), DiffServ, Multipro-

tocol Label Switching (MPLS) s Label Distribution Protocol (LDP) a Resource Reservation Protocol - Traffic Engineering (RSVP-TE) signalizaci.

Všechny tyto protokoly jsou reprezentovány jednoduchými modely.[9] Stejně jako u jiných modulů OMNeT++ je jejich vnější rozhraní popsáno v souborech NED a jejich implementace v jazyce C++. Z těchto modulů se dají poskládat síťová zařízení bez znalosti jazyka C++ a nutnosti kompilace, čistě pomocí jazyka NED. Lze také využít množství již sestavených zařízení jako jsou koncová zařízení, směrovače nebo přepínače.

Síťová rozhraní jsou většinou reprezentována složenými moduly a obsahují alespoň frontu a Media Access Control (MAC). Dalším typem modulů jsou takové, které uchovávají data (IPv4RoutingTable), konfigurují síť (FlatNetworkConfigurator), pohybují mobilními zařízeními (ConstSpeedMobility) nebo se starají a rádiové kanály (IRadioMedium)

4.3.2 MiXiM

MiXiM je další rámec rozšiřující možnosti OMNeTu. Přidává modely pro simulaci mobilních a fixních bezdrátových sítí.[3] Může být použit pro simulaci bezdrátových senzorových sítí, nositelné elektroniky, sítí složených z vozidel a podobně. Nabízí proto detailní modely šíření vln, jejich interferencí, energetické spotřeby vysílačů a bezdrátové MAC protokoly jako je Zigbee. Jméno tohoto rámce je zkratka z mixed simulator a odkazuje tak na to, že byl vytvořen spojením několika předchůdců.

Nejdůležitějšími znich jsou:

- ChSim (Universitaet Paderborn)
- Mac Simulator (Technische Universiteit Delft)
- Mobility Framework (Technische Universitaet Berlin, Telecommunication Networks Group)
- Positif Framework (Technische Universiteit Delft)

4.3.3 Univerzální datovvý generátor (UDG)

Univerzální datový generátor je určen pro generování libovolného datového toku pro účely simulace. Dovede pracovat s protokoly TCP a UDP a skládá se ze dvou částí. V první části je z požadované datové posloupnosti na transportní vrstvě TCP/IP generován profil datového toku ve formátu XML, který tento datový tok popisuje. Požadovaná datová posloupnost může být uměle definovaná nebo vytvořena na základě analýzy reálného provozu v síti. Při definici datové posloupnosti musíme dodržovat strukturu popsanou v obrázku 2

- Element <**profiles**> - zastřešující element celého definičního souboru
  - Atribut **ver** – verze souboru s datovým profilem (string)
- Element <**profile**> - element jednoho definovaného profilu
  - Atribut **id** – jednoznačný identifikátor profilu (integer : <0 – nekon.>)
- Element <**params**> - element definující sekci obecných parametrů datového profilu
  - Element <**type**> - typ generovaného datového profilu. (string : možnosti jsou „TCP/IP“, „UDP/IP“)
  - Element <**src_ip**> - zdrojová IP adresa (string : IP adresa v dekadickém formátu)
  - Element <**src_port**> - zdrojový port (integer : <1 – 65535>)
  - Element <**dst_ip**> - cílová IP adresa (string : IP adresa v dekadickém formátu)
  - Element <**dst_port**> - cílový port (integer : <1 – 65535>)
- Element <**data**> - element definující blok s předpisem generovaného datového profilu
  - Element <**pkt**> - element definující vlastnosti generovaného paketu
    - Atribut **dir** – atribut určující směr komunikace. (string : možnými parametry jsou „sd“ – source --> destination a „ds“ - destination --> source)
    - Atribut **id** – jednoznačný identifikátor daného paketu (integer : <0 – nekon.>)
    - Atribut **prev** – atribut definující spolu s **pprev** návaznost daného paketu na předchozí (shodné id profilu a id paketu). Typické využití je při generování odpovědi na dotaz nebo ACK došlého paketu. (integer : <0 – nekon.>)
    - Atribut **pprev** - atribut definující návaznost daného paketu na jiný profil. V případě absence se doplní hodnotou -1, která značí aktuální profil. (integer : <-1 – nekon.>)
    - Element <**time**> - V případě nově generovaného paketu (v elementu <***pkt***> se nenachází atribut ***prev)*** jde o dobu poslání paketu od začátku interpretace profilu. V případě odpovědi na paket (atribut ***prev*** v elementu <***pkt***> odkazuje na kauzálního předchůdce) jde o dobu od doručení předchozího paketu do poslání paketu.
    - Element <**payload**> - definice množství a zdroje/typu dat na 4. vrstvě RM ISO/OSI.
      - Atribut **size** – definice množství dat v bajtech (integer : <0 – nekon.>)
      - Atribut **type** – definice zdroje/typu přenášených dat. Pokud není uveden, je použit náhodný zdroj dat. (string : možné hodnoty „rnd“ - náhodná data)

Obrázek 2: Popis jednotlivých použitých elementů a jejich atributů

Druhá část je modul pro OMNeT++, který daný profil interpretuje a vytváří data použitá při simulaci. Současná verze UDG podporuje OMNeT++ verze 4.5 a INET 2.4 a snaží se minimalizovat nezbytnou konfiguraci v omnet.ini. Generovat může datové toky TCP i UDP relace s čistým potvrzováním ACK i s potvrzováním ACK s neseným datovým obsahem. Může také generovat přenos několika současných datových relací, které na sobě mohou být závislé.

5 FlowTester

Zařízení FlowTester bylo vyvinuto na katedře telekomunikační techniky Fakulty elektrotechnické Českého vysokého učení technického v Praze týmem Dr. Zbyňka Kocura. Slouží pro testování datových sítí a již bylo ověřeno i v komerčních zakázkách. I přesto se neustále vyvíjí a zejména díky dostupnosti klíčových programových aplikací se na vývoji může podílet celá vývojářská komunita. [12]

Pomocí tohoto zařízení lze testovat výkon i spolehlivost široké škály datových sítí založených na rodině protokolů TCP/IP od malého experimentu, jako v této práci, po velké průmyslové sítě určené pro sběr dat a řízení. [5] Testovat lze jak celkové parametry sítě, tak konkrétní služby. Zařízení má takto vysokou flexibilitu díky tomu, že pro měření můžeme definovat libovolný profil generovaných dat. Můžeme tak emulovat reálné aplikace jako http nebo IPTV, testovat protokoly používané v průmyslu nebo testovat reakci sítě na DoS a DDoS útoky. Pro vytváření datového toku UDP je použita aplikace FlowPing. Širokým možnostem testování dále napomáhá to, že FlowTester může pracovat nejen v režimu Peer-to-peer (P2P), kdy síť měříme jen mezi dvěma body, ale také v režimu point-to-multipoint Point-to-multipoint (P2MP). Můžeme tak testovat různé scénáře, ve kterých budou jednotky generovat více datových toků určených různým cílům.

Technické parametry:

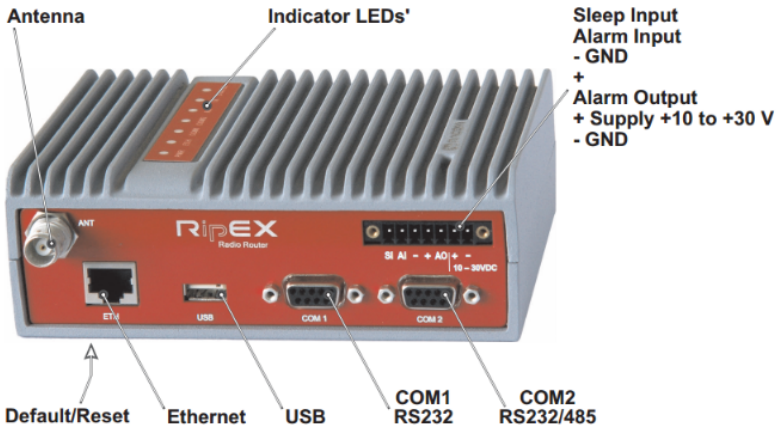
- Datové rozhraní: 2 x RJ-45 Ethernet 10/100/1000BASE-T v režimu síťový most
- Dohledové rozhraní: 1 x RJ-45 Ethernet 10/100/1000BASE-T 1 x DB9 - RS232
- Úložný prostor: SSD 256 GB
- Napájení: 12V DC, maximální příkon 12 W
- Rozsah pracovní teploty: 0 - 85 °C
- Ovládání a dohled: WEB rozhraní, terminál
- Stupeň krytí: IP40

Pro účely této práce byl FlowTester využit jen pro krátkodobá měření, ale lze jej použít i na delší měření testující dlouhodobou stabilitu sítě a její dostupnost. Z obou typů měření zaznamenává FlowTester naměřená data a ty pak prezentuje v podobě textového souboru formátu Comma-separated values (CSV) nebo rovnou vytvoří přehledné grafy. Mezi měřené parametry sítě patří přenosová rychlost, Round-trip time (RTT) a ztrátovost paketů.

6 RipEX

Ripex je vysoce konfigurovatelný rádiový modem, nebo přesněji rádiový IP router. Tento router byl představen firmou RACOM v roce 2011 jako zástupce spolehlivých industriálních rádiových modemů. [1] Jedná se o softwarově definovaný rádiový modem používající operační systém Linux. Jeho hlavní určení je pro profesionální použití a díky zaručené době odezvy a podpoře protokolů SCADA může být použit např. pro dohled při výrobě a distribuci elektřiny, vody, ropy a plynu, anebo pro loterijní transakční sítě a platební terminály.

Pro takovéto použití musí být modem spolehlivý, a proto je každá jednotka testována jak v reálném provozu, tak v klimatické komoře (provozní teplota je mezi -40 a 70 °C). Při výrobě jsou použity heavy duty a insutrial komponenty a modem má certifikaci pro prostředí s nebezpečím výbuchu.



Obrázek 3: Jendotka RipEX a její rozhraní [11]

Další doménou jednotek RipEX je snadná konfigurace. K základnímu nastavení jsou potřeba jen základní znalosti IP. Provádí se pomocí webového rozhraní, které poskytuje různé průvodce nastavení a zobrazí všechny konfigurační parametry na jedné stránce. Další možnosti konfigurace poskytuje Command Line Interface *(Příkazová řádka)* (CLI) přes Secure Shell (SSH) nebo servisní přístup jak přes Universal Serial Bus (USB), tak přes rozhraní ethernet. Update firmware nebo přidávání nově zakoupených funkcí je řešeno automaticky nahráním z USB flashdisku.

Pro použití v odlehlých oblastech např. na trase ropovodu je také důležitou vlastností malá spořeba. Rádiomodemy RipEX disponují funkcí Sleep mode, ve které má modem příkon jen 0,1 W a je ovládán digitálním vstupem a funkcí Save mode, kde modem může být probuzen z úsporného režimu i přijetím paketu na rádiovém kanálu. V tomto módu má modem spotřebu 2W.[11]

6.1 Technické informace

Modem komunikuje v licencovaném pásmu 160, 300, 400 nebo 900 MHz dle zvoleného modelu (model použitý v rámci této práce používá pásmo 300 MHz) a dokáže komunikovat na vzdálenost i 50 km přičemž nevyžaduje přímou viditelnost. Šířka kanálu může být určena mezi 6,25 kHz a 50 kHz a od toho se také odvíjí maximální přenosová rychlost 21 kbps pro nejužší kanál a až 166 kbps při použití plné šířky. Pro dosažení takovýchto rychlostí na velkou vzdálenost je důležitá výjimečná citlivost těchto modemů. Výrobce garantuje následující hodnoty:

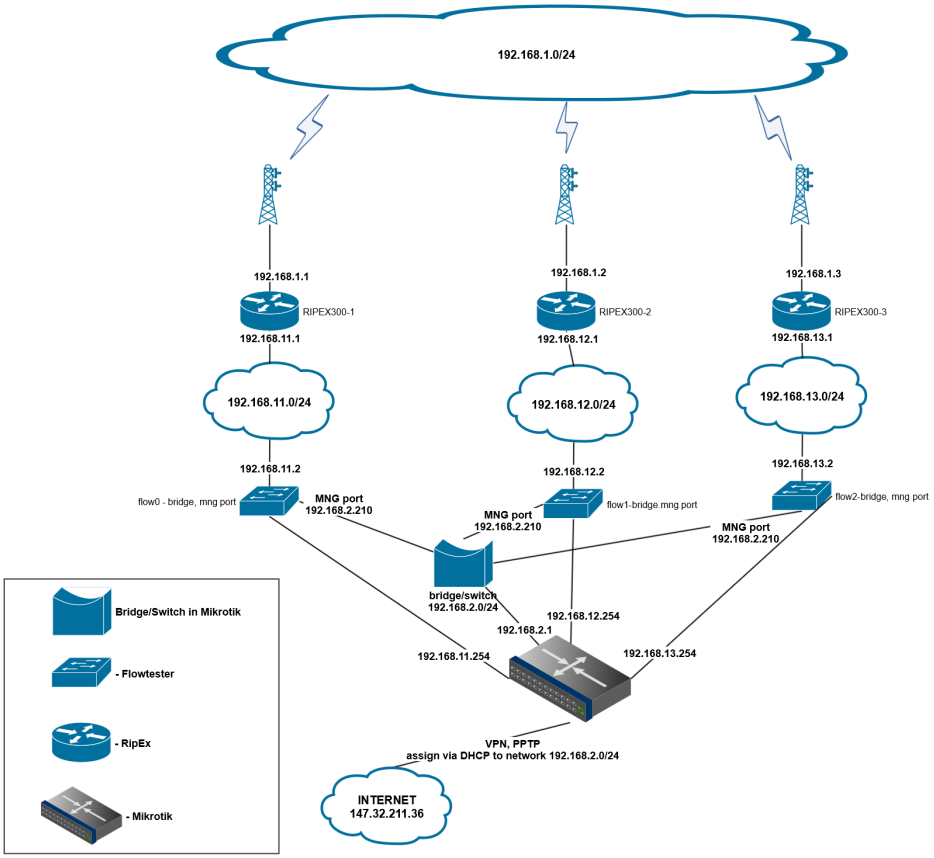
- -99 dBm / 83 kbps / 25 kHz / BER 10e-6
- -115 dBm / 10 kbps / 25 kHz / BER 10e-6

Dle vzdálenosti na kterou chceme komunikovat lze u routeru nastavit vysílací výkon od 0,1 až po 10 W. Při potřebě komunikovat na větší vzdálenosti než je maximální dosah jednoho skoku mezi modemy, může každý modem pracovat i jako repeater nebo můžeme využít hybridní sítě, kdy můžeme RipEX propojit s libovolnou IP síť jako je WLAN nebo internet. Tyto hybridní sítě mohou být také použity jako záložní cesty mezi dvěma modemy RipEX. V případě že primární cesta selže, dojde k automatickému přepnutí na záložní gateway, která nemusí být jen na rádiovém rozhraní, ale právě i na rozhraní ethernet a využívat například přenos pomocí mobilních dat. Takovýchto alternativních cest může být neomezený počet a jsou průběžně kontrolovány a prioritizovány.

Každá jednotka RipEX může fungovat ve dvou základních módech. Prvním módem je Bridge mode ve kterém jednotky tvoří transparentní síť. Každý paket přijatý na jednom ze sériových rozhraní Communication port (COM) a následně přes rádio odeslaný další jednotkám je jimi dál odeslán na všechna jejich příslušná COM rozhraní. Může tak být přijat více než jednou koncovou jednotkou. Vzhledem k tomu, že v tomto módu nejsou kolize řešeny na rádiové úrovni, je vhodné použít master-slave dotazování. Druhým módem je Router mode. V tomto módu se RipEX chová jako standartní IP router, který má čtyři rozhraní (Ethernet, Rádio a dva porty COM). Kolize jsou zde již řešeny anti-kolizním protokolem přímo na rádiovém kanálu. A můžeme využít i následující funkce: (možnost prodloužit popsánínm funkcí)

7 Popis laboratorní topologie

Parametry přenosu a celkový výkon modemů RipEX nebylo možné otestovat za podmínek podobných předpokládanému využití. Modemy totiž pracují na frekvencích, které jsou v licencovaném pásmu a bez povolení tak nemohou být používány. Jejich použití by mohlo narušit funkčnost jiných zařízení a případně by z něj mohla plynout pokuta. Z tohoto důvodu byl zvolen minimalistický přitup, kdy jsou modemy umístěny ve vzdálenosti jen několika centimetrů od sebe a jsou nastaveny na nejnižší vysílací výkon, aby se jejich signál nedostal mimo budovu.



Obrázek 4: Schéma zapojení laboratoře

Jak je patrné z obrázku 4 k dispozici byly tři jednotky RipEX. Rádiová rozhraní modemů byla adresována v subnetu 192.168.1.0/24 a všechny jednotky byly nastaveny do režimu router. Modemy se tak navzájem viděly a mohly spolu libovolně komunikovat. Aby bylo možné vytvářet předem definovaný provoz porovnatelný s provozem generovaným v simulaci, byla dále na rozhraní Ethernet připojena zařízení Flowtester. Každé z jejich propojení bylo adresováno na vlastním subnetu. Flowtestery byly dále propojeny pomocí bridge/switche mezi sebou kvůli vzájemné komunikaci během testů. Celá tato sestava byla napojena na router

Mikrotik, který laboratoř zpřístupnil přes Virtual Private Network (VPN) používajíci Point-to-Point Tunneling Protocol (PPTP) z internetu. Všechny testy tak bylo možné provádět vzdáleně bez fyzického přístupu k zařízením.

7.1 Routovací tabulky

7.1.1 RipEX 1

IP adresa	interface	via
192.168.2.0/24	192.168.11.254	ethernet
192.168.13.0/24	192.168.1.3	radio
192.168.12.0/24	192.168.1.2	radio

Tabulka 1: Routovací tabulka RipEX 1

7.1.2 RipEX 2

IP adresa	interface	via
192.168.2.0/24	192.168.12.254	ethernet
192.168.13.0/24	192.168.1.3	radio
192.168.11.0/24	192.168.1.1	radio

Tabulka 2: Routovací tabulka RipEX 2

7.1.3 RipEX 3

IP adresa	interface	via
192.168.2.0/24	192.168.13.254	ethernet
192.168.12.0/24	192.168.1.2	radio
192.168.11.0/24	192.168.1.1	radio

Tabulka 3: Routovací tabulka RipEX 3

7.1.4 Mikrotik

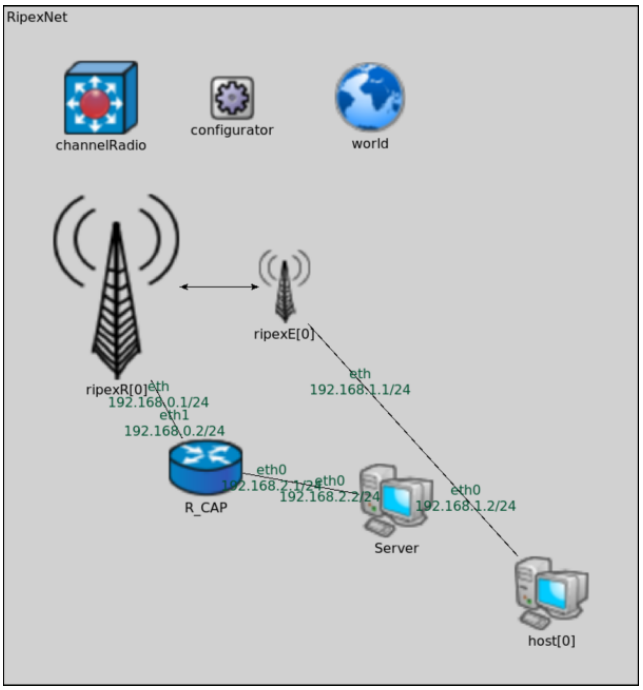
IP adresa	interface	via
147.32.211.0/24	eth9	ethernet
192.168.2.0/24	bridge/switch (mng ports)	ethernet
192.168.11.0/24	eth2	ethernet
192.168.12.0/24	eth3	ethernet
192.168.13.0/24	eth4	ethernet

Tabulka 4: Routovací tabulka Mikrotik

8 Metodika měření

8.1 Nastavení sítě

Abych co nejlépe otestoval shodu mezi modelem RipEXu a reálným zařízením musel jsem nejdříve nakonfigurovat jednoduchou síť modemů RipEX v prostředí simulačního nástroje OMNeT++5. Nakonfiguroval jsem jí podobně jako laboratorní konfiguraci reálných routerů, viz 4. Místo FlowTesterů jsem však použil server a klienta, kteří umí generovat traffic pomocí UDG. Kvůli zachytávání informací jsem mezi jeden Ripex a server vložil ještě router sledující všechna data přes něj poslaná. Z tohoto routeru pochází data použitá pro generování grafů a následné porovnání s reálnými RipExy.



Obrázek 5: Schéma zapojení sítě v simulačním nástroji OMNeT++)

8.2 Testovací profily

Díky tomu že jsem měl nakonfigurované ekvivalentní sítě pro simulaci i reálné měření, stačilo mi dále jen navrhnout stejné testovací profily na obou platformách. V případě testování v simulačním nástroji OMNeT++ byly profily definovány v souborech XML. Tyto soubory jsem generoval pomocí skriptu v jazyce Python, který je součásti UDG a já si ho jen upravil pro vlastní účely. Jak je vidět ze struktury profilu UDG na obrázku 2, u každého jednotlivého paketu bylo třeba definovat jeho velikost, čas odeslání, pořadové číslo a návaznost. V případě definování datového toku pro aplikaci FlowPing, jsem měl práci mnohem jednodušší. Stačilo definovat rychlost a velikost paketů v konkrétních časech a aplikace sama dopočítala potřebné množství paketů a intervaly mezi nimi.

Pro co nejlepší srovnání obou routerů jsem navrhl několik profilů datových toků. Protože

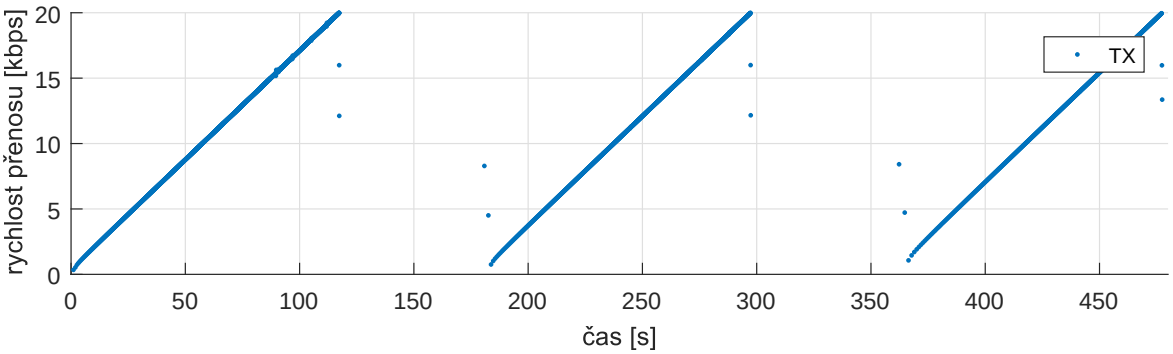
aplikace FlowPing na zařízení Flowtester, které jsem použil k testování reálných zařízení, podporuje pouze protokol UDP, používal jsem ho na většinu měření. Oba routery musely být vždy nastaveny stejně pro měření v simulaci i pro měření na reálných přístrojích.

Nastavení:

- Frekvence: 340 MHz
- Vysílací výkon: 0,1 W
- Šířka kanálu: 25kHz
- Modulace: 4CPFSK
- FEC: FEC 3/4
- Šifrování: vypnuté
- Režim: router
- ACK: zapnuté
- MTU: 1500B

Abych minimalizoval vliv druhu a podoby konkretních posílaných dat vypnul jsem také optimalizaci a kompresi.

Nejdříve jsem testoval profil nazvaný pila, viz obrázek 6, kde jsem vždy pakety o dané velikosti posílal se stále kratšími intervaly a nutil tak modemy k přenosu vyšší a vyšší rychlostí. Pro každý zub takovéto pily jsem pak použil jinou velikost paketu. Vzhledem k malé propustnosti modemů jsem netestoval rychlosti vyšší než 20 kbps což je výrobcem udaná maximální přenosová rychlost při mnou použité šířce kanálu 25kHz.



Obrázek 6: Vizualizace testovacího profilu. Můžeme vidět vždy tři po sobě jdoucí zuby pily, každý z nich je složen z paketů jiné velikosti (64, 128 a 256B)

Po testování zařízení profilem pila jsem se zaměřil na kritické body při tomto testování zjištěné. V těchto bodech jsem provedl měření při konstantní rychlosti a opět jsem vyzkoušel různé velikosti paketů. Tyto body zahrnovaly oblast počínající saturace datového spoje, prvního výskytu chyb nebo oblast s velkým nárůstem ztracených paketů. Každý takový bod jsem vždy testoval profilem o konstantní rychlosti po dobu 120s pro každou velikost paketu, abych dosáhl ustáleného stavu.

Jako poslední jsem se snažil změřit jak se u modelu a reálného zařízení liší práce s protokolem TCP. Kvůli tomu, že nešla k tomuto účelu na zřízeních FlowTester použít aplikace

FlowPing, musel jsem použít jednodušší nástroj iPerf. Tento nástroj slouží primárně k měření maximální propustnosti IP sítí a podporuje tyto protokoly:TCP, UDP, Stream Control Transmission Protocol (SCTP) s IPv4 a IPv6.

8.3 Zpracování měření

Vzhledem k tomu, že pro měření v simulačním nástroji a na reálné síti byly použity jiné nástroje, bylo vhodné získaná data upravit, aby bylo možné výsledné grafy porovnat. Prvním předpokladem bylo vybrat správný výstup z obou programů a zjistit zda byla data již samotným nástrojem průměrována. V případě, že data již průměrována byla, což byl případ například pro počet zahozených paketů, musel jsem odhadnout jakým způsobem a zda se tak stalo v obou paralelních měřeních. V případě rozdílu jsem musel implementovat vlastní průměrovaní, abych mohl vytvořit porovnatelné grafy. Pro tyto účely jsem používal průměrování klouzavým průměrem.

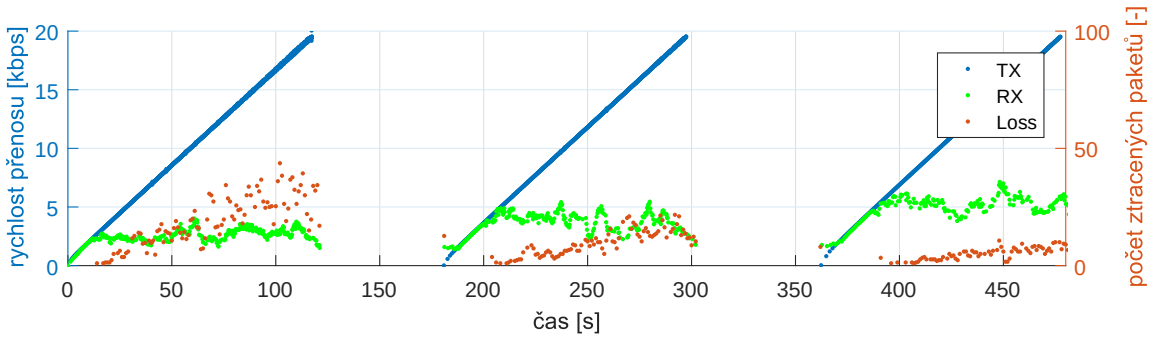
Tímto způsobem jsem průměroval i data, ze kterých by jinak nebyl dostatečně patrný trend. Vyhlazením křivky rychlosti jsem například získal z nic neříkajícího grafu, vypadajícího jako náhodná posloupnost, graf kde jsou jasně patrné limity použitého rádiového kanálu. Samozřejmě bylo nutné vždy data z obou paralelních měření vždy upravovat stejným způsobem, jinak by měření ztratilo smysl, jelikož by data již nebylo možné porovnávat.

9 Výsledky simluací

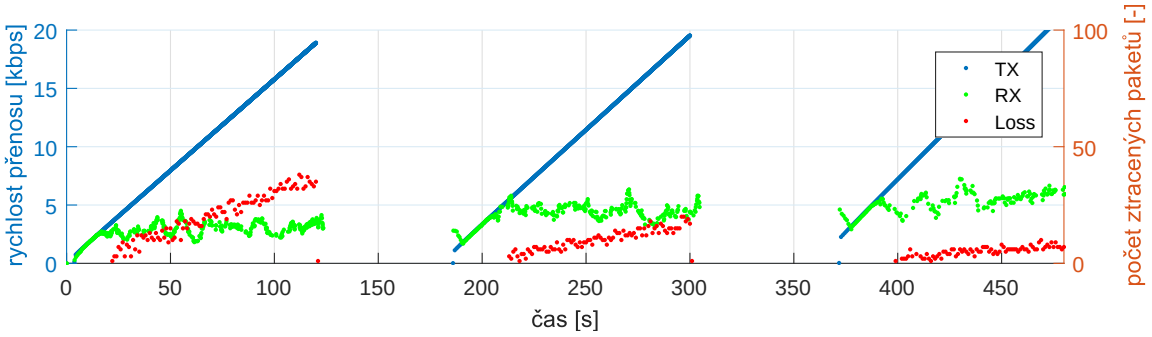
V této sekci bych chtěl shrnout výsledky všech měření. Každé měření jsem prováděl s pokud možno totožným nastavením na reálných routerech i na routerech v simulaci. Ve všech měřeních jsem vypnul kompresi, mohla by totiž být různě účinná pro datový tok generovaný pomocí UDG a pro datový tok z aplikace FlowPing.

9.1 Datový profil pila - UDP

Jako první profil jsem použil profil pila, ze kterého je vidět základní srovnání modelu a reálného zařízení. V tomto profilu posílám data vrůstající rychlostí od serveru na klienta, který data, která přijal, zase odešle zpět. Z toho vyplývá že jde o symetrický režim a přijatá data jsou obrazem dat odeslaných po dvou průchodech mezi modemy RipEX.



Obrázek 7: Graf výsledků z měření protokolu UDP pomocí profilu pila na reálném zařízení RipEX (symetrický režim)

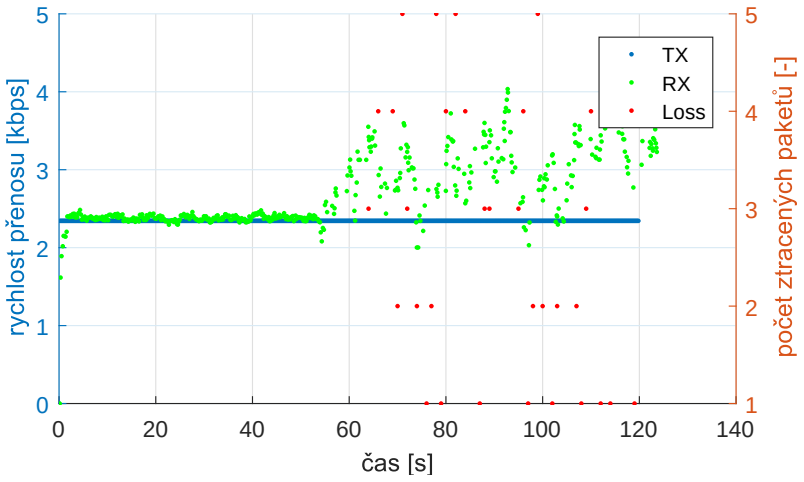


Obrázek 8: Graf výsledků z měření protokolu UDP pomocí profilu pila v simulaci (symetrický režim)

Z výsledných grafů je vidět, že alespoň základaní shoda mezi zařízením RipEX a jeho modelem panuje. V obou měřeních je patrné, že spoj pracuje bez chyb až do saturace datového spoje. Tato chvíle navíc nastává při podobných rychlostech pro odpovídající velikost paketů. U paketů velikosti 64B k saturaci dochází při rychlosti okolo 2,5 kbps, u 128B paketů pod hranincí 5 kbps a u největšího paketu velikosti 256B, ještě o něco později, při rychlosti přes 5 kbps. Do této doby se tak oba spoje chovají stejně. Po saturaci datového spoje již ke zvyšování přenosové rychlosti nedochází a navíc postupně se zvyšujícím se zahlcením kanálů stále více vysílanými daty stoupá ztrátovost paketů. Děje se tak v případě měření na reálném

zařízení, i v simulaci a proto pokud bychom soudili jen podle tohoto grafu, můžeme říct, že shoda modelu se svým vzorem je velmi přesná.

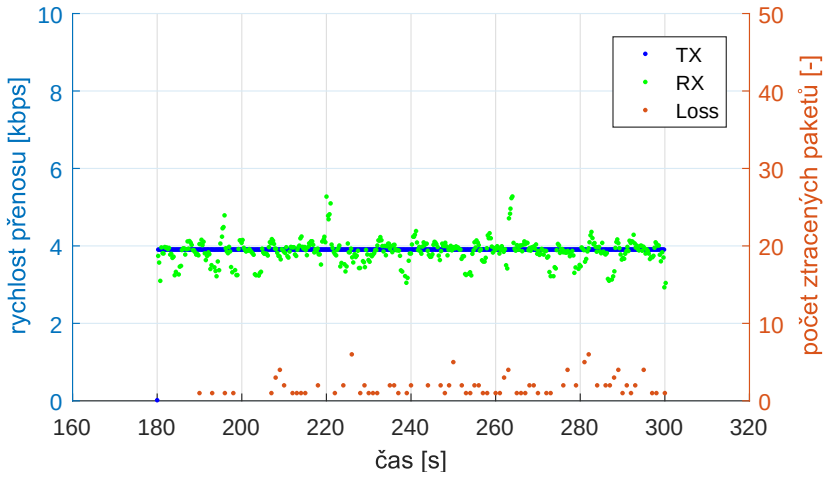
9.2 Konstantní datový profil UDP 2.5 kbps



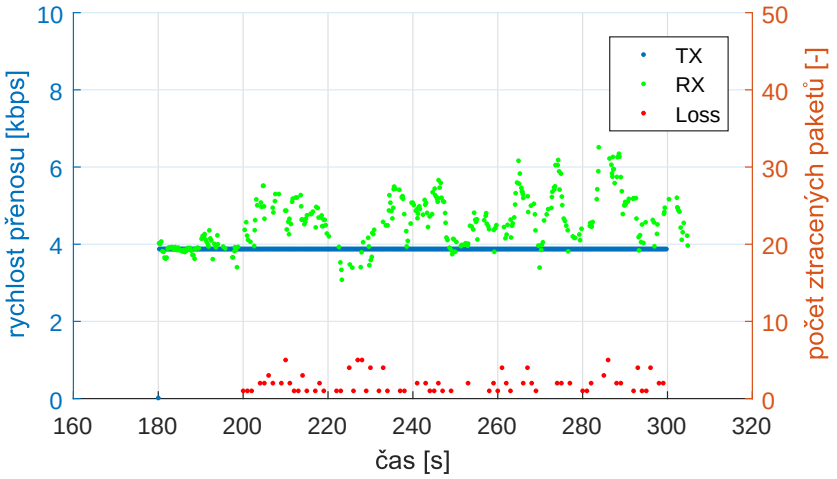
Obrázek 9: Graf výsledků z měření protokolu UDP pomocí konstantního profilu 2.5kbps v simulaci (symetrický režim)

V tomto profilu měřím symetrický konstantní datový tok 2,5 kbps. Na grafu 9 vidíme, že už při této rychlosti začíná mít model RipEXu při velikosti paketu 64B problémy, ale přibližně 50s jí modemy zvládají udržovat bez zahazování paketů. Je to tak velmi zajímavá hodnota, protože je přesně na hranici možností routeru za těchto podmínek. Po 50s se nejspíše naplní fronta, router začne zahazovat pakety a chybovost spoje stoupá. Výsledky reálného měření pro 2,5 kbps jsem nezískal, protože aplikace Flowping očekává, že hodnota nastavené rychlosti bude celočíselná a zadaná v kilobitech za vteřinu. Mnou nastavenou rychlost 2,5 kbps tedy zaokrouhlí na 2 kbps. Vzhledem k tomu, že při rychlosti 2 kbps nedocházelo k žádným chybám a při rychlosti 3 kbps již docházelo k významným ztrátám, můžeme předpokládat, že se skutečný RipEX bude chovat podobně.

9.3 Konstantní datový profil UDP 4 kbps pro pakety velikosti 128B



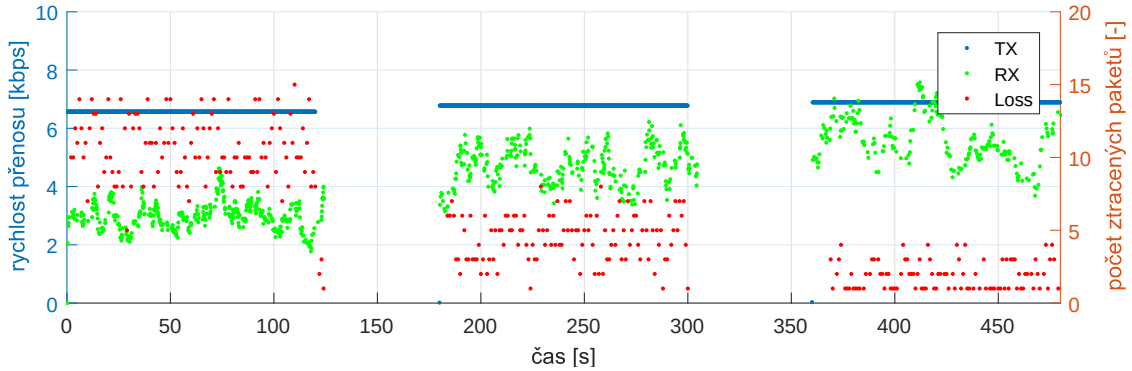
Obrázek 10: Graf výsledků z měření protokolu UDP pomocí konstantního profilu 4 kbps na reálném zařízení RipEX (symetrický režim)



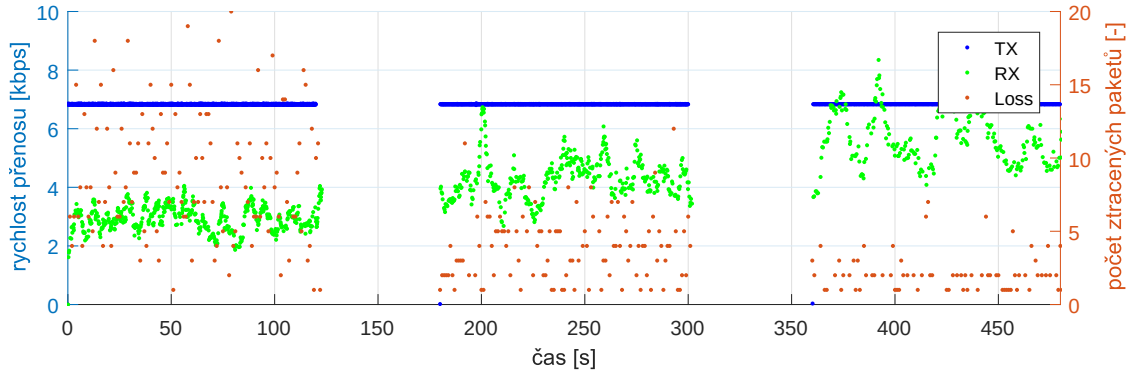
Obrázek 11: Graf výsledků z měření protokolu UDP pomocí konstantního profilu 4 kbps v simulaci (symetrický režim)

Abych měl přímé srovnání reality a simulace v hraniční situaci popsané v minulé podkapitole, otestoval jsem, zda pro jinou velikost paketů nenastane tento jev na celočíselné hodnotě rychlosti v kbps. Podobnou situaci, i když ne tak zřetelnou, jsem našel v symetrickém režimu pro pakety velikosti 128B. Při rychlosti 4 kbps v simulaci vidíme podobné chování jako v předešlém případě, akorát se zde fronta zaplní již po 20s a po jejím zaplnění vzrůstá chybovost spoje. Tentokrát lze stejnou přenosovou rychlost nastavit i v aplikaci FlowPing. Na reálném zařízení je vidět, že se chová při této rychlosti jinak než model. Pakety sice začne ztrácet o něco dříve, ale daří se mu v průměru lépe udržovat přenosovou rychlost 4 kbps. To značí, že rozdíl mezi skutečností a simulací by mohl být způsoben odlišným chováním front na zařízeních. I když tento rozdíl nebude nejspíše mít při normálním používání velký vliv.

9.4 Konstantní datový profil UDP 7 kbps pro pakety velikosti 128B



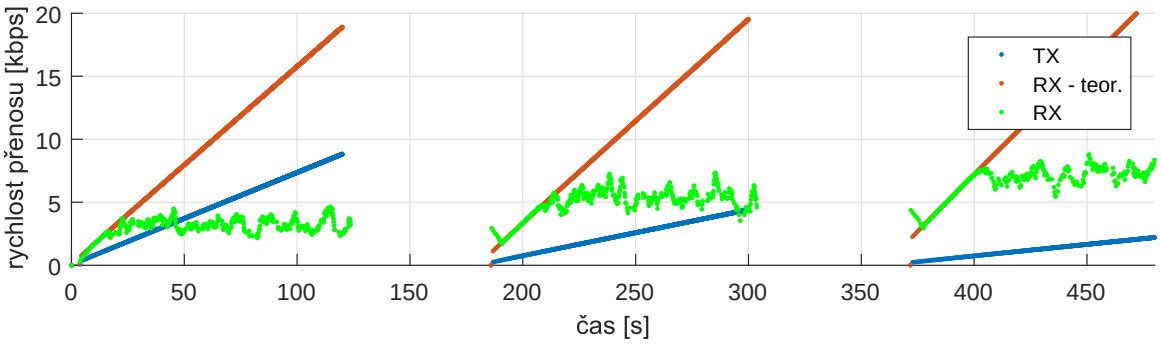
Obrázek 12: Graf výsledků z měření protokolu UDP pomocí konstantního profilu 7 kbps v simulaci (symetrický režim)



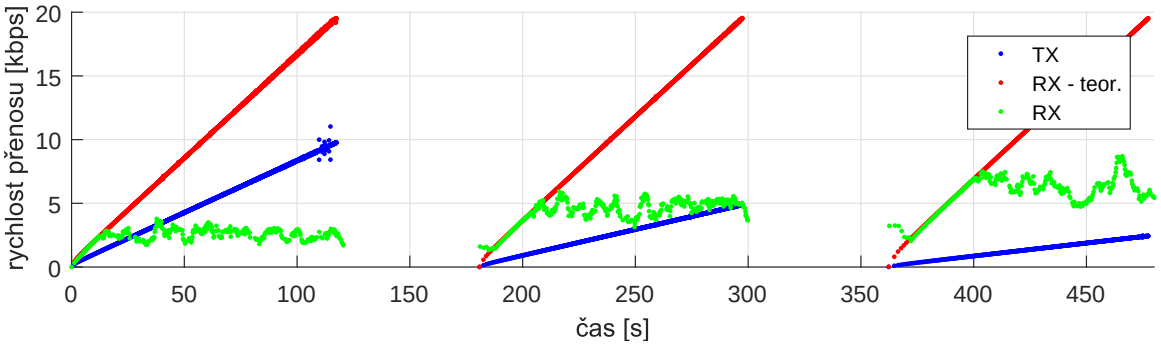
Obrázek 13: Graf výsledků z měření protokolu UDP pomocí konstantního profilu 7 kbps na reálném zařízení RipEX (symetrický režim)

Abych potvrdil, že rozdíl mezi modelem a zařízením zjištěný při konstantním toku 4 kbps je patrný jen v krajních případech, provedl jsem ještě měření při rychlosti 7 kbps. Zde je opět vidět vysoká míra shody v obou měřeních.

9.5 Profil pila UDP v asymetrickém režimu



Obrázek 14: Graf výsledků z měření protokolu UDP pomocí profilu pila v simulaci (asymetrický režim)

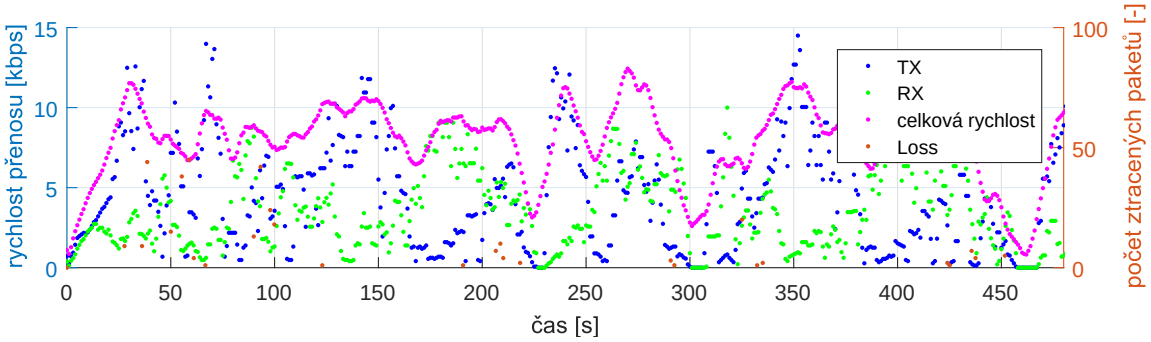


Obrázek 15: Graf výsledků z měření protokolu UDP pomocí profilu pila na reálném zařízení RipEX (asymetrický režim)

V dosud popsaných měřeních byla vždy snaha posílat stejná data v obou směrech. Měření tak fungovalo v symetrickém režimu. Abych získal data z více možných situací a tím pádem mohl přesněji porovnat model modemu RipEX s jeho předlohou, změřil jsem propustnost i v asymetrické režimu. V tomto režimu posílám pakety se stejnou frekvencí oběma směry, ale pakety vysílané nenesou žádnou uživatelskou informaci. Při stejném množství přijímaných dat je tak celková přenosová rychlost menší. Výsledné grafy navíc ukazují jen provoz jedním směrem. TX zde tedy značí data na jedné straně odeslaná a RX na druhé straně přijatá. Můžeme tak porovnat jak daný profil ovlivní jeden průchod přes rádiové spojení místo průchodů dvou jako v předchozích měřeních. I přes tento trochu jiný způsob měření dostáváme velmi podobné výsledky jako v předchozích případech. Simulace a reálné měření probíhají stejně a až do té doby, než nastane saturace datového toku, přijímáme všechna vyslaná data dle teoretického předpokladu. Stejně jako v předchozích měřeních vidíme, že při použití větších paketů saturace nastává později.

9.6 Měření protkolu TCP

I přesto že aplikace iPref, kterou jsem použil pro měření s protokolem TCP tento protokol podporuje, tak nedovede vysílat datový tok o konkrétní rychlosti. Tuto funkci má jen při použití protokolu UDP. Proto jsem jej použil alespoň pro změření maximální dosažitelné propustnosti. Ta mi v opakovaných testech vyšla okolo 10 kbps. To přibližně odpovídá i výsledku měření v OMNeTu. Tam jsem použil stejný profil datového toku jako při měření s protokolem UDP, ale vzhledem k rozdílné funkčnosti těchto protokolů je výsledek úplně jiný. Jakmile totiž dojde k saturaci spoje, přestává TCP posílat pakety v definovaný čas a nejdříve počká na potvrzení přijetí ACK paketů předchozích. Dále se u TCP nedozvíme nic o vlivu velikosti paketů jelikož payload libovolně rozděluje do paketů vlastní velikosti. Do grafu jsem přidal vyhlazenou křivku celkové rychlosti, abych mohl porovnat maximální propustnost s měřením z iPerfu. Může se zdát, že model dosahuje podobné propustnosti jako reálná zařízení, průměr je však 8,1 kbps.



Obrázek 16: Graf výsledků z měření protokolu TCP pomocí profilu pila v simulaci

10 Závěr

Během vypracování této práce jsem se seznámil s mnoha nástroji a technologiemi. Chtěl jsem si v praxi vyzkoušet jak se pracuje s průmyslovými úzkopásmovými modemy a hlavně se seznámit s diskrétním simulačním prostředím OMNeT++. Lákaly mě jeho široké možnosti využití. Proniknout do tohoto simulátoru byla asi nejtěžší část této bakalářské práce. Druhá strana mince jeho flexibility je totiž jeho velká složitost. Zvládl jsem to jen díky množství tutoriálů, kvalitní dokumentaci programu a pomoci svého vedoucího. Poté, co jsem pochopil, jak funguje program samotný a naučil se v něm pracovat, přišel čas se seznámit s modelem simulovaného rádiomodemu RipEX. Je to velmi komplexní model, proto mi i přes velké množství ukázkových příkladů trvalo velmi dlouho připravit takovou konfiguraci sítě, aby odpovídala nastavení reálných routerů. Nestačilo jen nastavit podobné nastavení jako je dostupné z webového rozhraní reálných routerů, ale musel jsem řešit i nastavení vrstvy MAC a podobně. Vždy, když jsem si myslel, že mám vše již správně nakonfigurované a začal jsem s měřením, ukázalo se, že tomu tak není a já musím začít znovu. Nakonec jsem však dospěl ke konfiguraci simulované sítě, která má všechny parametry stejné jako použitá verze skutečných rádiových modemů RipEX a mohl jsem začít měřit.

V rámci měření se největší zkouškou ukázal být protokol TCP. To, jak s ním routery pracují, se mi nepodařilo otestovat zdaleka tak komplexně jak jsem si představoval. Bylo to dáno nejen menším počtem nástrojů, které s ním umí pracovat, ale i jeho přirozenými vlastnostmi. Ty zapříčiňují, že jeho práci nelze ovládat do nejmenších detailů jako je tomu v případě UDP. Měření za použití protokolu TCP by tak mohlo být dále rozvinuto v pokračování této práce.

Dalším oříškem k rozlousknutí se ukázalo být samotné generování grafů. Data pocházela z několika různých programů a v každém zdroji byla trochu jinak prezentována. Ve velkém množství dat bylo náročné vybrat taková, která jsou relevantní a porovnatelná se svým protějškem. Proto jsem musel pro každý měřící nástroj vymyslet nový skript v progrmau Matlab. Myslím, že se mi je poměrně dobře podařilo prezentovat ve formě vhodné pro porovnání mezi sebou.

11 Shrnutí výsledků

Z provedených měření a jejich výsledků vyplývá, že i přesto, že model neodpovídá reálnému zařízení do posledního detailu, panuje tu dostatečná shoda pro běžné využití. Saturace kanálu nastává u modelu i u reálného zařízení při stejných rychlostech pro odpovídající velikosti paketů a přenos konstantní rychlostí před dosažením saturace probíhá u obou zařízení bez chyb. Po zahlcení kanálu se v přenosu začnou objevovat chyby, objevují se však v podobné míře v simulaci i při měření na zařízení RipEX. Pokud se tedy při přenosu netrefíme do situace, kdy hraje velkou roli sebemenší rozdíl v chování front, budeme v simulaci dostávat stejné výsledky jako na skutečných zařízeních. Model je tedy dostatečně přesný jako pomůcka při návrhu sítí postavených na platformě rádiomodemů RipEX.

Reference

[1] automation.com. Racom releases ripex radio modem router. http://www.automation.com/product-showcase/racom-releases-ripex-radio-modem-router, 2016. [Online, 24.5.2016].

[2] Jerry Banks. *Discrete-Event System Simulation*. Prentice Hall, December 2004.

[3] MiXiM Developers. Mixim project — introduction. http://mixim.sourceforge.net/, 2016. [Online, 24.5.2016].

[4] INET. What is inet framework? https://inet.omnetpp.org/Introduction.html. [Online, 24.5.2016].

[5] ČVUT v Praze Katedra telekomunikační techniky, FEL. Flowtester - testování datových sítí efektivně, spolehlivě a cíleně. https://flowtester.fel.cvut.cz/, 2016. [Online, 24.5.2016].

[6] OpenSim Ltd. A quick overview of the omnet++ ide. https://omnetpp.org/doc/omnetpp/IDE-Overview.pdf, 2016. [Online, 24.5.2016].

[7] OpenSim Ltd. User manual - omnet++ version 4. https://omnetpp.org/doc/omnetpp4/manual/usman.html, 2016. [Online, 24.5.2016].

[8] OpenSim Ltd. What is omnet++? https://omnetpp.org/intro, 2016. [Online, 24.5.2016].

[9] INET Manual. Architecture of the inet framework. https://omnetpp.org/doc/inet/api-current/neddoc/index.html?p=inet-architecture.html. [Online, 24.5.2016].

[10] Petr Peringer. Modelovaní a simulace. http://www.fit.vutbr.cz/study/courses/IMS/public/prednasky/IMS.pdf, 2016. [Online, 24.5.2016].

[11] RACOM. Ripex. http://www.racom.eu/cz/products/radio-modem-ripex.html, 2016. [Online, 24.5.2016].

[12] Andrea Vondráková. UnikÁtnÍ flowtester z fakulty elektrotechnickÉ Čvut umoŽŇuje lepŠÍ diagnostiku a monitorovÁnÍ komunikaČnÍch sÍtÍ. https://aktualne.cvut.cz/tiskove-zpravy/20160330-unikatni-flowtester-z-fakulty-elektrotechnicke-cvut-umoznuje-lepsi, 2016. [Online, 24.5.2016].

Seznam příloh

DVD s konfiguračními soubory a výsledky měření

- omnet - složka obsahující soubory z měření v simulaci
  - konfiguracni soubory - konfigurační soubory nutné pro běh simulace
  - testovací profily - datové profily použité při simulaci
  - vysledky - data použitá pro generování grafů
- Flowtester - složka obsahující soubory z měření na reálných zařízeních
  - testovací profily - datové profily použité při měření
  - vysledky - data použitá pro generování grafů