VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

IMPLEMENTACE SÍŤOVÝCH BEZPEČNOSTNÍCH ALGORITMŮ V DOMÁCÍM SMĚROVAČI

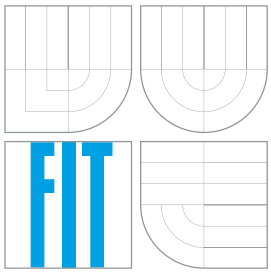
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

TOMÁŠ PROCHÁZKA

BRNO 2011

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

IMPLEMENTACE SÍŤOVÝCH BEZPEČNOSTNÍCH ALGORITMŮ V DOMÁCÍM SMĚROVAČI

IMPLEMENTATION OF NETWORK SECURITY ALGORITHMS IN HOME ROUTER

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — TOMÁŠ PROCHÁZKA
AUTHOR

VEDOUCÍ PRÁCE — Ing. VLASTIMIL KOŠAŘ
SUPERVISOR

BRNO 2011

Abstrakt

Tato práce se zabývá minimalistickou linuxovou distribucí OpenWRT a její instalací v směrovači Linksys WAG160Nv2. Popisuje vybraný směrovač a diskutuje postup při měření propustnosti, která byla zvolena jako kritérium pro hodnocení výkonnosti směrovače. Dále je zkoumána výkonnost směrovače při použití původního systému a se systémem OpenWRT. V dalších kapitolách je popsán návrh a implementace odposlouchávacího algoritmu, jehož hlavním účelem je zachytávání síťového provozu na směrovači. V poslední části jsou dosažené výsledky diskutovány.

Abstract

This bachelor thesis briefly describes minimalist linux distribution OpenWRT and its installation in Linksys WAG160Nv2 router. It describes a selected router and it discusses a measuring procedure of a throughput, wich was chosen as a criterion for an evaluating the performance of the router. Furthermore, it examined the performance of the router using the original system and running OpenWRT. Design and implementation of interception algorithms, whose main purpose is to capture network traffic on the router, are described in the subsequent parts. Results are presented in the final part.

Klíčová slova

směrovač, OpenWRT, bezpečnostní algoritmy, Broadcom, Backfire, Linksys, odposlech, propustnost

Keywords

router, OpenWRT, security algorithms, Broadcom, Backfire, Linksys, intercept, throughput

Citace

Tomáš Procházka: Implementace síťových bezpečnostních
algoritmů v domácím směrovači, bakalářská práce, Brno, FIT VUT v Brně, 2011

Implementace síťových bezpečnostních algoritmů v domácím směrovači

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Vlastimila Košaře

.....
Tomáš Procházka
10. května 2011

Poděkování

Rád bych poděkoval vedoucímu práce Ing. Vlastimilovi Košařovi za to, že mi věnoval svůj čas a poskytl mi odbornou pomoc při řešení této práce.

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

Router, neboli směrovač, se stal nedílnou součástí dnešních moderních domácností. Tento trend je podpořen cenovou dostupností zařízení a faktem, že dostatečně splňují hlavní účel, pro který jsou kupovány, tedy připojení ke globální síti, rozbočení internetového připojení mezi více počítačů v domácnosti, sdílení technických zařízení v domácí síti atd. Z cenového a funkčního hlediska bychom mohli směrovače rozdělit do dvou skupin. Domácí, které jsou na nižší cennové úrovni, a jejich možnosti, funkce a výkonnost odpovídají oblasti využití: domácnosti, malé firmy a instituce. Zařízení s velkou funkční výbavou, vysokým výkonem a také vyšší cenou bychom mohli zařadit do skupiny „profesionálních zařízení".

V této práci se budeme zabývat domácími routery. S příchodem OpenWRT se naskytuje možnost modifikovat firmware těchto zařízení a rozšiřovat tak jejich funkčnost a možnosti při stejné ceně a hardwaru zařízení.

Cílem práce je implementace vybraného bezpečnostního algoritmu do operačního systému OpenWRT a zprovoznění tohoto operačního systému v domácím směrovači. Dále má být porovnána výkonnost směrovače před a po jeho úpravě. V závěru práce budou dosažené výsledky diskutovány.

Práce je členěna do několika kapitol. První tři teoreticky zaměřené kapitoly se věnují popisu OpenWRT, měření propustnosti a popisu vybraného routeru. První kapitola přiblíží čtenáři OpenWRT jeho verzování, modifikaci a popíše využití flash paměti směrovače, na kterém je nainstalováno OpenWRT. Další kapitola obsahuje doporučení pro testování síťových zařízení a seznámí čtenáře s programem Iperf. Následující kapitola uzavírá teoreticky zaměřenou část popisem směrovače Linksys WAG160Nv2 z funkčního a hardwarového hlediska.

Po teoretickém úvodu navazuje praktická část, která obsahuje čtyři kapitoly. V první z nich je popsána instalace OpenWRT do domácího směrovače. Následující kapitola obsahuje postup při měření propustnosti a uvádí naměřené výsledky před a po úpravě systému směrovače. Další kapitola popisuje implementaci síťového bezpečnostního algoritmu, který má monitorovat síťový provoz na směrovači Linksys WAG160Nv2. Praktickou část zakončí rozšiřující kapitola zabívající se algoritmickým testováním výkonnosti procesorů.

Kapitola 2

OpenWRT

OpenWRT je popisováno jako minimalistická GNU/Linuxová distribuce pro vestavěná zařízení, která umožňuje vytvoření firmwaru podporující plně zapisovatelný souborový systém se správou balíčků [12]. Tento systém přináší možnost výběru nejrůznějších utilit, nástrojů a konfigurace systému, narozdíl od firmwarů výrobců, které jsou většinou pevně dané a nelze je dále rozšiřovat či měnit. OpenWRT patří mezi open-source projekty, které jsou licencovány pod GPL.

Začátek tohoto projektu je datován počátkem roku 2004. První verze OpenWRT byla založena na GPL zdrojových kódech pro Linksys WRT54G. Tato verze byla široce rozšířena a dodnes je využívána aplikacemi jako Freifunk-Firmware nebo Sip@Home [12].

Celý systém je ovládán příkazovým řádkem dostupným z ssh, přes telnet nebo přes sériovou konzoli. Systém nenabízí klasické uživatelské rozhraní, tak jako většina dnešních moderních distribucí Linuxu. Pro základní konfiguraci routeru může posloužit webové rozhraní, které lze doinstalovat.

OpenWRT je dostupné pouze pro některé routery a jejich platformy. V současné době jsou nejvíce podporována zařízení značek Linksys a TP-Link. Mezi nejčastěji podporované platformy se řadí Atheros a Broadcom. Při výběru routeru pro OpenWRT je tedy nutné nejdříve zjistit, zda je kompatibilní a s jakou verzí OpenWRT. Přehled podporovaných routerů je dostupný ná stránkách projektu[1].

2.1 Verze

Většina linuxových distribucí používá číslování verzí. OpenWRT je značeno kódovými jmény doplněné o číselné značení. Zajímavostí kódových jmen jsou jejich názvy, pojmenované dle alkoholických míchaných nápojů s návodem k jejich výrobě umístěném přímo v úvodním výpisu systému. První stabilní verze byla pojmenována jako „White Russian“.

Stabilní verze jsou značeny čísly, které jsou složeny z roku a měsíce, kdy byly vytvořeny. Označení může obsahovat přídavné třetí číslo, které udává úpravu vydané verze [12]. V níže uvedené tabulce 2.1 je uveden přehled stabilních verzí OpenWRT.

2.1.1 Backfire 10.03

Backfire 10.03 je kódové jméno poslední stabilní verze OpenWRT, která vyšla v Dubnu roku 2010. Tato verze oproti verzím nižším podporuje širší spektrum platforem a přináší

[1] http://wiki.openwrt.org/toh/start

Verze	Datum vydání
Backfire 10.03	Duben 2010
Kamikaze 8.09.2	Leden 2010
Kamikaze 8.09.1	Červen 2009
Kamikaze 7.09	Září 2008
Kamikaze 7.07	Září 2007
Kamikaze 7.06	Červen 2007

Tabulka 2.1: Přehled stabilních verzí

určitá vylepšení. Přehled některých změn konfigurace Backfire 10.03 od poslední stabilní verze [12]:

- linuxové jádro 2.6.32
- nový formát konfigurace switche pro Broadcom zařízení
- nový web server uhttpd
- podpora rootfs na externích médiích
- knihovna uClibc 0.9.30 pro jazyk C
- podpora platformy pro ADSL modem/router brcm63xx

2.2 Modifikace OpenWRT

Hlavní výhoda OpenWRT spočívá v dostupnosti zdrojových kódů, které si může uživatel upravit, a před jejich překladem nastavit konfiguraci výsledného image pro router. Dále lze jednoduše rozšiřovat funkcionalitu routeru velkou sadou balíčků. Jde tedy o systém, který lze velmi jednoduše upravovat a nastavovat dle potřeb uživatelů a možností routerů.

2.2.1 Balíčkový systém

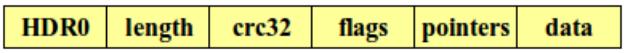
Balíčkový systém je podobný těm, co známe z běžných linuxových distribucí. Jde o odlehčenou verzi nazývanou Opkg nebo Ipkg[2]. Ovládání je obdobné. Balíčky lze instalovat, odebírat atd. Instalace balíčků je možná dvěma způsoby:

- instalace přímo ze systému pomocí příkazů
- vytvoření balíčku v adresáři package a následná kompilace zdrojových kódů

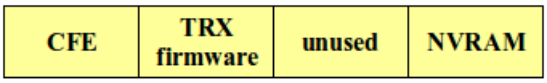
2.2.2 Úprava zdrojových kódů

Výsledný systém lze také upravovat pomocí zdrojových kódů. Z uvedených možností se však jedná o nejkomplikovanější úpravy, při kterých musí uživatel přesně vědět, co a kde může upravit.

[2]Ipkg používají starší verze OpenWRT



Obrázek 2.1: Struktura hlavičky trx firmwaru [12]



Obrázek 2.2: Struktura flash paměti před nabootováním systému [12]

2.3 Popis struktury flash paměti

Ve flash paměti routeru jsou uloženy důležité části pro běh a funkčnost celého systému. Nalezneme zde bootloader, firmware s operačním systémem a oblast s konfiguračními daty. Například u platforem Broadcom flash paměť obsahuje CFE[3] bootloader, firmware se systémem a NVRAM oddíl.

Bootloader se stará o inicializaci paměti a zavádění firmwaru při startu routeru. Firmware bývá často v zařízení aktualizován a měněn. Z tohoto důvodu je od bootloaderu v paměti flash oddělen. Při nahrání špatného firmwaru, nebo při selhání aktualizace, nám proto postačí obnovit pouze firmware. V mnoha případech bootloader poskytuje mechanismus pro obnovu, který umožňuje přepsání flash paměti směrovače. Například bootloader CFE, který je používán v zařízeních Broadcom, využívá pro obnovu TFTP serveru [12].

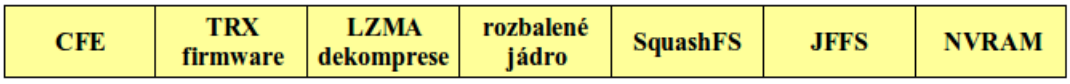
Firmware je programové vybavení, které řídí elektronické zařízení. V našem případě se jedná o systém OpenWRT. Firmware bývá nejčastěji zapouzdřen do trx nebo bin formátu. Oba formáty jsou téměř stejné. Ve formátu bin jsou k začátku připojeny informace o modelu zařízení.

Následující informace byly převzaty z: [12]. Jak bylo výše uvedeno, flash paměť obsahuje oddíl s firmwarem. Tato část obsahuje firmware ve formátu trx, což je pouze zapouzdření, které je znázorněno na obrázku 2.1. HDR0 reprezentuje hodnotu udávající hlavičku trx firmwaru. Za touto hodnotou následuje délka a kontrolní součet dat, příznaky, ukazatele a samotná data.

Když pomineme zapouzdření, celá flash paměť bude mít strukturu, která je uvedena na obrázku 2.2. Firmware bývá velmi malý, proto není využita celá oblast mezi zavaděčem a oblastí NVRAM.

Samý začátek firmwaru obsahuje jádro, které se stará o zavádění systému. Jádro je však velké, a proto je komprimováno. Při startu systému, zavaděč nabootuje do LZMA[4] programu, který jádro dekomprimuje do paměti a začne s jeho vykonáváním.

Za jádrem následují souborové systémy. V OpenWRT se využívá kombinace SquashFS a JFFS2 (Journalling Flash File System). SquashFS je souborový systém pouze pro čtení, který je schopný dosáhnout vysoké komprese dat. Pro zápis do paměti flash je využíván JFFS2. Popis struktury flash paměti po nabootování systému je uveden na obrázku 2.3.



Obrázek 2.3: Struktura flash paměti po nabootováním systému [12]

[3] CFE – Common Firmware Enviroment
[4] LZMA – Lempel Ziv Markov Chain Algorithm

Kapitola 3

Měření propustnosti

Výrobci síťových zařízení často uvádí vyšší výkonnosti jejich zařízení, než skutečně dosahují, aby produkt získal lepší umístění na obchodním trhu. Hodnoty uváděné na obalech produktů nejsou většinou ani teoreticky dosažitelné. Jestliže se snažíme zjistit přesné parametry zařízení, nezbývá nám nic jiného, než si je ověřit.

Propustnost definujeme jako maximální rychlost přenosu, při které nejsou přenášené pakety zařízením zahazovány [13]. Lze ji měřit pomocí softwarových programů, nebo hardwarových zařízení. Hardwarové měřiče dosahují často přesnějších výsledků, zaleží však, pro jaký účel hodláme měření uskutečnit. V této práci bude dále diskutováno pouze softwarové měření propustnosti.

Měření propustnosti je diskutováno v RFC dokumentech, například: Benchmarking Methodology for Network Interconnect Devices (RFC 2544), Methodology for IP Multicast Benchmarking (RFC 3918), Benchmarking Methodology for LAN Switching Devices (RFC 2889). Pro měření propustnosti zařízení na síťové vrstvě ISO/OSI modelu je nejvhodnější RFC 2544 – Benchmarking Methodology for Network Interconnect Devices, které bude dále rozebíráno.

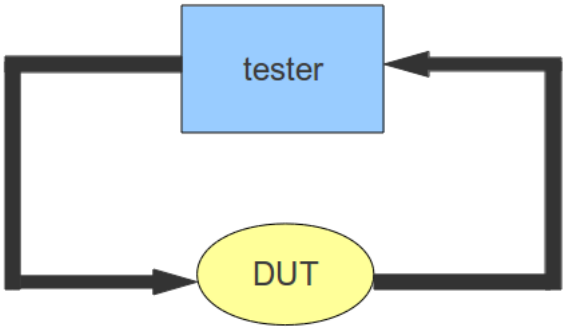
3.1 Způsob měření

Obecně je při testování síťových zařízení nutné dodržovat zásady týkající se zařízení a jeho nastavení, které jsou popsány právě v RFC 2544. Pro měření je doporučeno použít následujících „topologií“. Ideálním způsobem pro testování je použití testovacího zařízení s přijímacími i odesílacími porty viz. obrázek 3.1. Tester vysílá data, která jsou zpracována testovaným zařízením[1] a dále přeposlána opět testovacímu zařízení, které vyhodnotí, zda byla přijata správná data. K testování lze také použít topologii uvedenou na obrázku 3.2, kterou dosáhneme stejných výsledků, jako v předchozím případě.

3.1.1 Nastavení testovaného zařízení

U testovaného zařízení se předpokládá, že všechny podporované protokoly budou nakonfigurovány a povoleny. Dále je nutné zajistit, aby konfigurace nebyla během měření změněna. Prováděný test opakujeme několikrát a následně z naměřených hodnot vypočítáme průměr, aby byl výsledek co nejpřesnější. Je vhodné změřit vliv filtrů[2] na propustnost. To provedeme změřením propustnosti při stejné konfiguraci se zapnutým a vypnutým filtrem [13].

[1] V anglickém jazyce je používán termín DUT – device under test
[2] Např. firewall



Obrázek 3.1: Kruhová topologie pro testování

Obrázek 3.2: Topologie s oděleným přijmačem a vysílačem pro testování

3.1.2 Formát a velikost ethernetových rámců

Při testování by měly být použity standardní ethernetové rámce pro TCP/IP a jejich formát by měl být uveden ve zprávě o měření. Formát ethernetového rámce popisuje obrázek 3.3.

Testy by měly být provedeny s několika různě velkými ethernetovými rámci, zejména s nejmenší a největší možnou velikostí. Pro každý protokol[3] a přenosové technologie[4] musí být provedeny testy zvlášť. Doporučené velikosti rámců pro Ethernet: 64, 128, 256, 512, 1024, 1280, 1518 bytů [13].

3.1.3 Postup při měření propustnosti

1. skrze testované zařízení pošleme známý počet rámců známou rychlostí
2. spočítáme počet rámců přenesených zařízením
3. jestliže je počet přijatých rámců menší než poslaných, test opakujeme
4. v případě, že je počet přijatých a vyslaných rámců stejný, zaznamenáme si výslednou rychlost

[3] Např. RIP, RIPv2, OSPF, IGRP, EIGRP
[4] Např. Ethernet, Token Ring, FDDI

Obrázek 3.3: Formát ethernetového rámce [9]

3.1.4 Interpretace výsledku

Výsledkem měření by měl být graf, který zobrazuje závislost rychlosti přenosu na velikosti rámce. V grafu by také měla být zobrazena teoretická závislost rychlosti přenosu na velikosti rámce. Vyjádření o hodnotě propustnosti musí obsahovat [13]:

- změřenou maximální rychlost přenosu
- velikost použitého rámce při naměření hodnoty propustnosti
- teoretický limit pro testované přenosové médium
- typ protokolu použitého při testování

3.2 Iperf

Softwarové nástroje pro měření výkonu síťových zařízení jako Iperf a Network Weather Service (NWS) patří mezi nejuznávanější a nejpoužívanější programy. Oba používají pro měření jiné metody, každý má své výhody a nevýhody [14]. V dalším textu se budeme zabývat pouze programem Iperf.

Iperf je nástroj umožňující měření propustnosti síťových zařízení. Jedná se o open source software dostupný pro Windows i Unix systémy. Alternativou tohoto programu je Jperf, který obsahuje grafické prostředí napsané v Javě.

Umožňuje testovat propustnost při použití TCP nebo UDP protokolu transportní vrstvy ISO/OSI modelu. Testování je prováděno pomocí klientské a servrové části programu. Serverová část přijímá vysílaná data od klientské části po předem nastavenou dobu. Klientská část se snaží přenést co nejvíce náhodných dat. Podle velikosti přenesených dat a časového úseku přenosu je vypočítána propustnost. Iperf podporuje pomocí parametrů programu nastavení možností přenosu, jako například: velikost ethernetového rámce, TCP nebo UDP přenos, velikost TCP okna atd.

Kapitola 4

Linksys WAG160Nv2

Linksys je dnes[1] divizí velmi známé firmy v oblasti síťových technologií, a to Cisco. Od roku 1995, kdy byl založen, se zabývá především menšími síťovými produkty pro kancelářské a domácí využití [8]. Dalšími produkty firmy Linksys jsou modemy, USB Wi-Fi moduly, síťová úložiště, VOIP telefony a další.

4.1 Popis směrovače

Model Linksys WAG160Nv2 je ADSL2+ Modem-Router s Wi-Fi. Jedná se o směrovač střední cennové kategorie, který poskytuje komplexní služby se zabezpečením proti útokům.

4.1.1 Výčet hlavních funkcí a výbavy Linksys WAG160Nv2

Tento výčet je převzat z [10].

- 4 ethernetové porty podporující MDI/MDIX[2], port 1 lze nastavit jako WAN
- jeden port typu DSL
- standardy pro bezdrátové sítě: 802.11b, 802.11g, 802.11n,
- rozšířená podpora bezpečnosti: filtrování portů, filtrování MAC a IP adres, DMZ, NAT
- šifrování Wi-Fi spojení: WPA, WPA2
- podpora VPN
- podpora statického a dynamického směrování, směrování RIPv1, RIPv2, DHCP server

4.1.2 Hardwerová specifikace Linksys WAG160Nv2

- architektura: MIPS
- procesor: Broadcom BCM6358, 300MHz
- flash paměť: MX29LV320AB, 4096KB

[1] Dříve byl Linksys samostatnou firmou, od roku 2003 je „spojen“ s Cisco
[2] Technologie detekující typ kabelu, umožňující automatickou konfiguraci v závislosti na typu kabelu.

Obrázek 4.1: Linksys WAG160Nv2 [10]

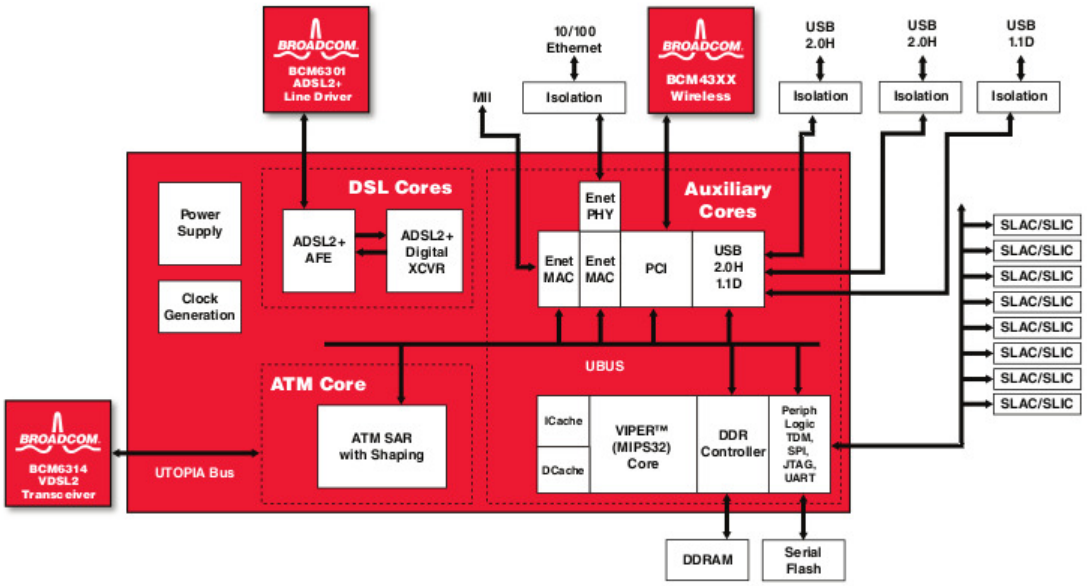
- ram paměť: EtronTech EM6AA160TS-5G 32MB SDRAM
- Wi-Fi adaptér: Atheros AR9223-AC1A
- ethernetový adaptér: Broadcom BCM5325EKQMG
- bootloader: CFE
- JTAG, sériový port

4.2 Popis mikroprocesoru Broadcom BCM6358

Jedná se o ADSL2+ Gateway Chip, který byl představen společností Broadcom roku 2006 v Chicagu. Následující popis a schéma jsou převzata z [1]

- podpora pro: USB 2.0, Bluetooth, vícekanálové VOIP, IEEE 802.11 a/b/g/n, dual 10/100 Ethernet
- ADSL vysílač a příjmač vyhovující standardům: G.992.1, G.992.2, G.992.3, G.992.5, G.993.1, G.993.2, a T1.413
- vysoce výkoný VIPER MIPS32 CPU
- 16/32-bitová paralelní rozšiřující sběrnice podporující CardBus, PCMCIA a mini-PCI
- víceknálová 8/16-bit TDM/PCM sběrnice
- sériové a paralelní Flash rozhraní s podporou DDRAM
- ATM SAR hardware podporující traffic shaping a QoS

Obrázek 4.2 popisuje vnitřní strukturu čipu BCM 6358



Obrázek 4.2: BCM 6358 system diagram [1]

4.3 Architektura MIPS32

Architektura MIPS (Microprocessor without Interlocked Pipeline Stages) byla vyvinuta před více než 20 lety na universitě Stanford. Dnes je vyvíjena společností MIPS Technologies. Jedná se o jednoduchou, efektivní a flexibilní architekturu typu RISC. Tato architektura je nejrozšířenější v embeded zařízeních díky své velké instrukční sadě, rozžířitelnosti z 32bitů na 64btů a širokému spektru vývojových nástrojů. Na dnešních ochodních trzích má zastoupení v SOHO sítích, při automatizaci kancelářských prací,v síťové a telekomunikační infrastruktuře atd. Tyto informace byly převzaty z [11].

Vlastnosti architektury MIPS32

- pevná velikost 32-bitových instrukcí
- velká instrukční sada RISC, s tří operandovými instrukcemi
- 32 bitů virtuálního adresového prostoru a až 36 bitů fyzického adresního prostoru
- jednoduché adresovací režimy
- podpora 8-bitových, 16-bitových, 32-bitových instrukcí
- flexibilní správa softwaru pro zásobníkové operace
- podpora systémů Big-Endian a Little-Endian
- dopředná kompatibilita s MIPS64
- volitelná správa paměťové jednotky (MMU - Memory Management Unit) s:
  - mechanismem překladu adres TLB nebo BAT
  - programovatelnou velikostí stránky

Kapitola 5

Instalace systému OpenWRT

Po výběru vhodného směrovače a otestování jeho výkonnosti byly zahájeny přípravy k instalaci OpenWRT do směrovače. V následujícím textu je mimo jiné uvedeno, jak získat firmware potřebný pro nahrání do směrovače, nebo jak sestrojit sériový TTL převodník pro komunikaci s routerem.

5.1 Firmware

Jak bylo výše naznačeno, k instalaci je potřebný firmware s OpenWRT. Ten lze získat dvěma způsoby. Jednou z možností je stažení firmwaru ze stránek projektu OpenWRT. Každé zařízení, které je podporováno, má na webových stránkách[1] dostupné různé verze firmwarů, co se týče konfigurací a verzí systému. Tato možnost je především vhodná pro uživatele, kteří nechtějí dělat nestandardní změny v systému a vystačí si se zvolenou konfigurací. Druhou možností, jak získat firmware, je vytvoření image ze zdrojových souborů, které jsou dostupné přes SVN.

5.1.1 Kompilace zdrojových kódů

Vytvoříme si složku pro zdrojové kódy:

```
mkdir OpenWRT
cd OpenWRT
```

stáhneme zdrojové kódy stabilní verze:

```
svn co svn://svn.openwrt.org/openwrt/branches/backfire
```

nebo vývojové verze:

```
svn co svn://svn.openwrt.org/openwrt/trunk/
```

Nastavíme vlastnosti pro výsledný image, zejména volbu Target System na BCM63XX, ostatní položky mohou zůstat defaultně nastaveny. Konfiguraci uložíme a můžeme zahájit kopilaci.

```
make menuconfig
make
```

První kompilace je zdlouhavá, protože jsou stahovány toolsety a utility. Vybuildovaný image s OpenWRT je dostupný ve složce ~`/OpenWRT/backfire/bin/brcm63xx/`.

[1] http://downloads.openwrt.org/backfire/10.03/

5.1.2 Úprava zdrojových souborů pro Linksys WAG160Nv2

U směrovačů Linksys WAG160N je nutné rozlišovat jejich verze, protože typ WAG160Nv2 není nativně podporován, a je nutná úprava zdrojových kódů k částečné podpoře.

Verze 2 tohoto routeru obsahuje kontrolní mechanismus, který se snaží zabránit provozu neproprietárních firmwarů na těchto směrovačích. Systém ochrany spočívá v umístění dvou tzv. pid konstant (pid a pid2) ve flash paměti. Při spuštění směrovače bootloader (CFE) kontroluje jejich výskyt. Jestliže nenajde jednu z pid konstant, zastaví proces bootování a tím znemožní celý běh systému ve směrovači. První pid se nachází přímo za zavaděčem, ovšem pid2 je umístěno ve vrchních dvou megabytech flash paměti. Řetězcová konstanta, kterou bootloader vyhledává je `"seRcOmM"` [12].

Problém s nenalezením konstanty pid2 vznikl při nahrání firmweru OpenWRT s JFFS (verze se SquashFS obsahuje JFFS overlay), které se „roztáhlo“ ve zbývající flash paměti a přepsalo konstantu pid2. Při restartování směrovače a následném spuštění již bootloader pid2 nenašel a zastavil bootování systému, který dokumentuje následující výpis:

```
no pid2
FIRMWARE HAD BEEN DESTORYED!!
```

Řešením, které by umožňovalo provozovat OpenWRT na tomto směrovači, bylo odstranění souborového systému JFFS z firmwaru za cenu, že nebude možné provádět trvalé změny za běhu OpenWRT.

Níže uvedené úpravy zajistí vytvoření firmwaru bez JFFS. Ve všech `firstboot` skriptech ve složkách se zdrojovými kódy zakomentujeme řádek s:

```
boot_run_hook switch2jffs
```

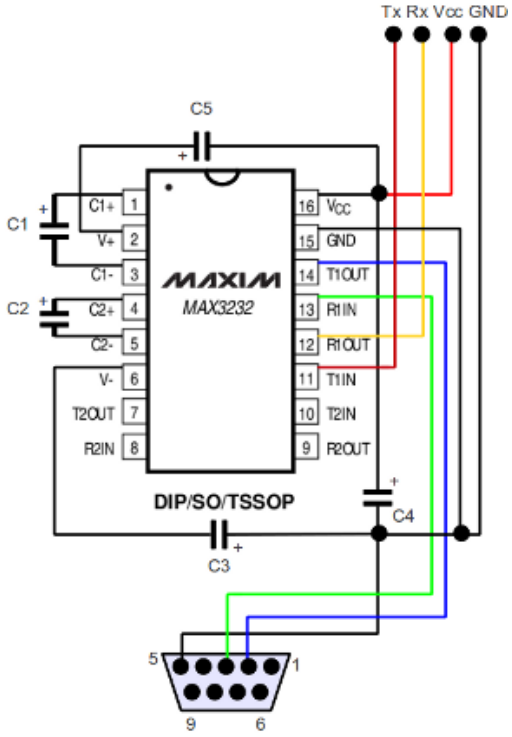
a nahradíme například pouze výpisem.

```
if [ "${0##*/}" = "firstboot" ]; then
    if [ "$1" = "switch2jffs" ]; then
       echo "our hack"
       #boot_run_hook switch2jffs
       ...
fi
```

Tato jednoduchá úprava nám zajistí, že systém OpenWRT nebude obsahovat JFFS, které by přepsalo řetězcovou konstantu pid2.

Dále je třeba nastavit parametry pro desku našeho routeru, aby byl upravený image použitelný pro Linksys WAG160Nv2. Upravíme soubor: `board_bcm963xx.c`, který se nachází v adresáři:
`~/build_dir/linux-brcm63xx/linux-2.6.32.10/arch/mips/bcm63xx/boards/`. Do souboru je třeba doplnit parametry udávající vlastnosti desky zařízení Linksys WAG160Nv2, protože nebylo v okamžik úprav podporované. Soubor je dostupný v přílohách na CD. Poslední úprava je nutná v souboru: `~/target/linux/brcm63xx/image/Makefile` kde přidáme řádek do `define Image/Build`:
`$(call Image/Build/CFE,$(1),96358GW,6358,96358GW-bc310,,-y 5)`, který zajistí vybuildování obrazu s OpenWRT pro tento typ směrovače a jeho desku.

Předchozí úpravy byly provedeny v měsíci lednu, avšak po několika týdnech byly pro zařízení Linksys WAG160Nv2 komunitou OpenWRT vytvořeny opravné soubory, které řeší problém s konstantou pid2. Od tohoto okamžiku je Linksys WAG160Nv2 mezi podporovanými směrovači OpenWRT.



Obrázek 5.1: Schéma zapojení sériového TTL převodníku na 3.3V

5.2 Nahrání firmwaru do směrovače

Přepsat flash paměť směrovače firmwarem lze standardně několika způsoby. Nejdostupnější a nejlehčí variantou je webové rozhraní směrovače, kde se nachází nabídka pro aktualizaci firmwaru. Tato možnost však při práci s OpenWRT není příliš vhodná, protože přepsáním flash paměti vadným firmwarem bychom mohli původní obraz se systémem poškodit a následně bychom nebyli schopni touto metodou firmware obnovit.

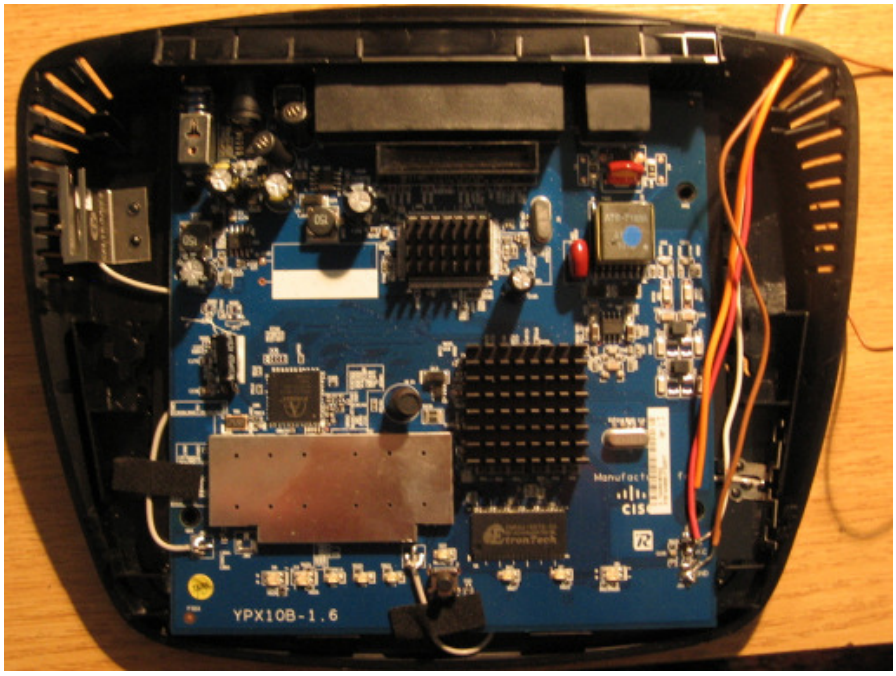
Elegantní, ale náročnější metodou, při které máme směrovač plně pod kontrolou, je aktualizace firmwaru pomocí sériového rozhraní. Téměř všechny směrovače mají toto rozhraní implementované, ale většina směrovačů toto rozhraní nemá vyvedené v podobě sériového portu, známeho z klasického PC. Navíc sériové rozhraní v PC pracuje s napájecím napětím 12V, kdežto zvolený směrovač s 3.3V. Následující problém řeší sériový TTL převodník na 3.3V, který lze zakoupit v elektronických obchodech na internetu, nebo je možné převodník sestrojit za výhodnější cenu, než je „hotové řešení“.

5.2.1 Sériový TTL převodník na 3.3V

K sestrojení převodníku budeme potřebovat následující součástky:

- čip MAX3232
- 5 kondenzátorů 0.1μF (minimálně na 15V)
- konektor do sériového portu PC

Výše uvedené součástky zapojíme dle schématu na obrázku 5.1. Max 3232 umožňuje regulovat pracovní napětí pomocí kondenzátorů v rozmezí 3 – 5.5V. S jakým napětím



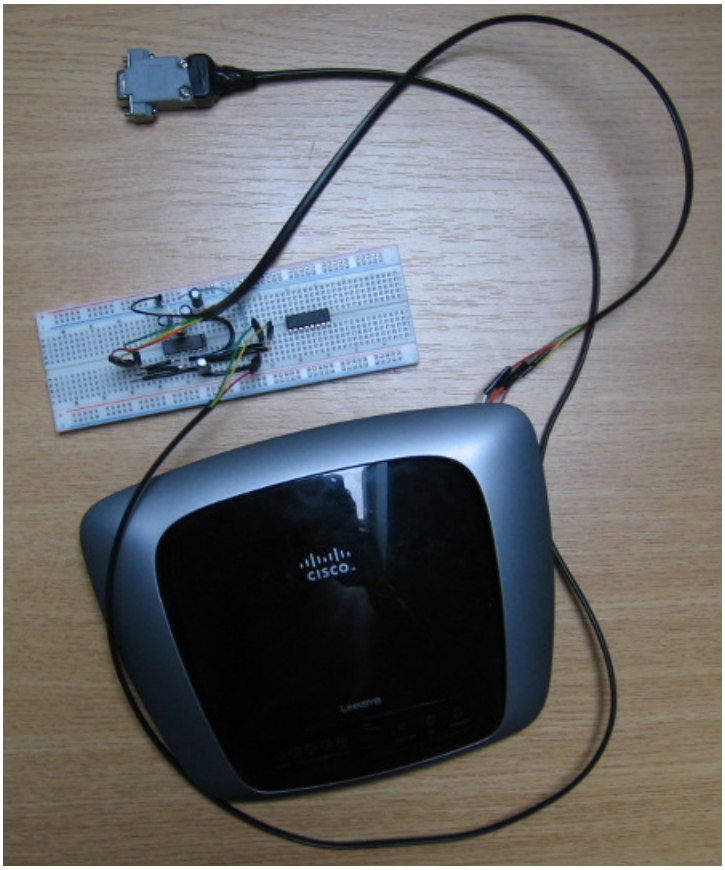
Obrázek 5.2: Vnitřní část routeru s připájenými vodiči převodníku

bude pracovat, záleží na kapacitě kondenzátorů. Tyto hodnoty lze vyčíst z příslušného datasheetu. Na obrázku 5.1 je naznačen konektor sériového portu, který se zapojí do PC, vodiče označené Tx, Rx, Vcc a GND budou připájeny k plošnému spoji směrovače viz. obrázek 5.2.

5.2.2 Přepsání flash paměti směrovače

Jakmile máme vytvořený správný image s firmwarem, hotový sériový TTL převodník, můžeme přistoupit k procesu přepsání flash paměti dle následujících kroků:

1. na PC nainstalujeme TFTP server – pro tyto účely vhodně poslouží freewarové nástroje jako např.: CiscoTFTP nebo SolarWinds
2. IP adresu PC nastavíme na 192.168.1.100 – tuto adresu musí mít i TFTP server
3. image s OpenWRT přejmenujeme na bcm963xx_fs_kernel (bez přípony) a umístíme na příslušné místo v PC (záležní na typu TFTP serveru)
4. spojíme PC se směrovačem pomocí převodníku viz. ilustrační foto 5.3
5. spustíme sériovou konzoli např. Hyper Terminál s nastavením:
   - bity za sekundu: 115200 baudů
   - datové bity: 8 bitů
   - parita: bez parity
   - stop bity: 1 stop bit
   - řízení toku: bez hardwarové a softwarové kontroly



Obrázek 5.3: Router se sériovým TTL převodníkem

6. při zapnutí směrovače podržíme klávesu enter, která přeruší proces bootování a zpřístupní nabídku CFE
7. v prostředí CFE použijeme příkaz `f` pro přepsání flash paměti obrazem s OpenWRT
8. následně proběhne stažení obrazu do směrovače, kam je následně nainstalován

Příkaz mtd pro přepis flash paměti v OpenWRT

Jestliže máme funkční systém OpenWRT ve směrovači, lze pomocí příkazu `mtd` v OpenWRT provést přepsání flash paměti. Soubor s firmwarem musí být uložen ve směrovači.

```
mtd -r write <firmware> linux
```

5.3 Obnova nefunkčního směrovače

Při nahrávání OpenWRT do směrovače může dojít vinou špatného image k nefunkčnosti běhu celého systému. Většinou je však dostupná nabídka bootloaderu CFE, díky které jsme schopni příkazem `f` nahrát například původní firmware výrobce[2]. Bohužel se někdy můžeme dostat i do stavu, že nejsme schopni použít standardní nabídky CFE k nahrání

[2] http://www.linksysbycisco.com/AE/en/support/WAG160N/download

nového obrazu se systémem. Taková situace může nastat například při nenalezení konstanty pid. V tomto případě je proces bootování zastaven dříve, než bootloader vyzve k zmáčknutí klávesy enter, a není se tak možné dostat do nabídky CFE, která mimo jiné slouží právě pro nahrání nového firmwaru.

5.3.1 Postup obnovy

V tomto případě lze k obnovení použít rozhraní JTAG. Veškerý proces obnovy by byl zřejmě zdlouhavý a složitý proti metodě s použitím tzv. Download módu. Do tohoto módu lze směrovač dostat přidržením tlačítka reset při procesu bootování. Následně stačí použít program Sercomm firmware updater. Níže jsou uvedeny body k postupu obnovy tímto způsobem.

1. nainstalujeme program Sercomm firmware updater[3] [4]
2. nastavíme IP adresu PC na 192.168.1.100 a připojíme UTP kabelem k směrovači
3. router uvedeme do Download módu, přidržením tlačítka reset během bootování – LED dioda se znakem POWER červeně bliká
4. spustíme Sercomm firmware updater, ten by měl identifikovat náš router
5. v programu vybereme soubor s firmwarem a zvolíme možnost Update

Tato metoda se může zdát vhodnou i při funkčních stavech směrovače pro aktualizaci firmwaru. Opravdu tomu tak může být, ovšem stále platí, že nemáme směrovač zcela „pod kontrolou“. Nevidíme žádnou odezvu od směrovače. Veškeré výpisy při bootování a startu OpenWRT nejsou dostupné tak, jako při použití sériového rozhraní a konzoli. V případě chyby ji tedy nejsme schopni bez textových výstupů identifikovat.

[3] verze pro Windows: http://www.nslu2-linux.org/wiki/Main/SercommFirmwareUpdater
[4] verze pro Linux: https://github.com/jal2/WAG160Nv2

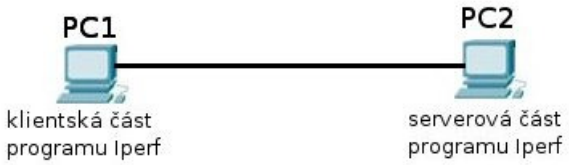
Kapitola 6

Test výkonnosti směrovače

Pro otestování výkonnosti byl zvolen test měření propustnosti směrovače, který byl prováděn programem Iperf. K tomuto účelu byly potřeba dva počítače. Na jednom z počítačů byla spuštěna serverová část a na druhém klientská část programu Iperf. Tento test byl proveden s původním systémem směrovače a po úpravě se systémem OpenWRT.

6.1 Postup měření

Testování probíhalo v několika fázích, které byly podobné pro případy před a po úpravě systému směrovače. Nejdříve byla ověřena propustnost ethernetové linky mezi dvěma počítači bez směrovače, viz. obrázek 6.1.

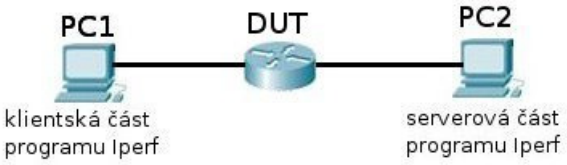


Obrázek 6.1: Zapojení pro testování propustnosti mezi dvěma PC

Následně byla změřena propustnost switche. Rozhraní LAN je propojováno speciálním obvodem zajišťujícím vlastní přepínání. Přenášená data tak vůbec neprojdou do hlavního procesoru, proto by mělo být zpoždění minimální a při použití TCP protokolu by měla být propustnost ovlivněna jen nepatrně (díky velikosti TCP okénka a round-trip zpoždění[1]). Při použití UDP protokolu by propustnost měla být ovlivněna jen round-trip zpožděním. Po úpravě systému směrovače nebyla testována výkonnost při přepínání, jelikož obvod realizující přepínání je hardwarový a softwarová změna by na něj neměla mít vliv.

Dále byla měřena propustnost směrovače při směrování, a to s vypnutým a zapnutým firewallem. Pro měření propustnosti při přepínání a směrování bylo použito zapojení 6.2. Toto zapojení se mírně lišilo u měření propustnosti při přepínání, kdy oba počítače byly připojeny do LAN portů. V případě testování při směrování byl jeden počítač zapojen do LAN portu a druhý do WAN portu směrovače.

[1] Čas, který uplyne od odeslání dat po jejich příjem a zpracování cílovým zařízením.



Obrázek 6.2: Zapojení pro testování propustnosti při přepínání a směrování

6.1.1 Nastavení programu Iperf

Měření propustnosti při protokolu TCP

Klientská část programu:

```
iperf -c [server_IP] -M segment_size
```

Serverová část programu:

```
iperf -s
```

Hodnota parametru `M` značí velikost segmentu (MTU), parametr `s` spouští servrovou a parametr `c` klientskou část programu.

Měření propustnosti při protokolu UDP

Klientská část programu:

```
iperf -c [server_IP] -u -b 100M -l buff_size
```

Serverová část programu:

```
iperf -s -u
```

Parametr `u` spouští program pro UDP měření, hodtona parametru `b` udává cílovou šířku pásma. Pomocí parametru `l` je možné nastavit velikost UDP packetu.

6.1.2 Použitá zařízení pro testování a jejich parametry

Pro testovací účely byly použity:

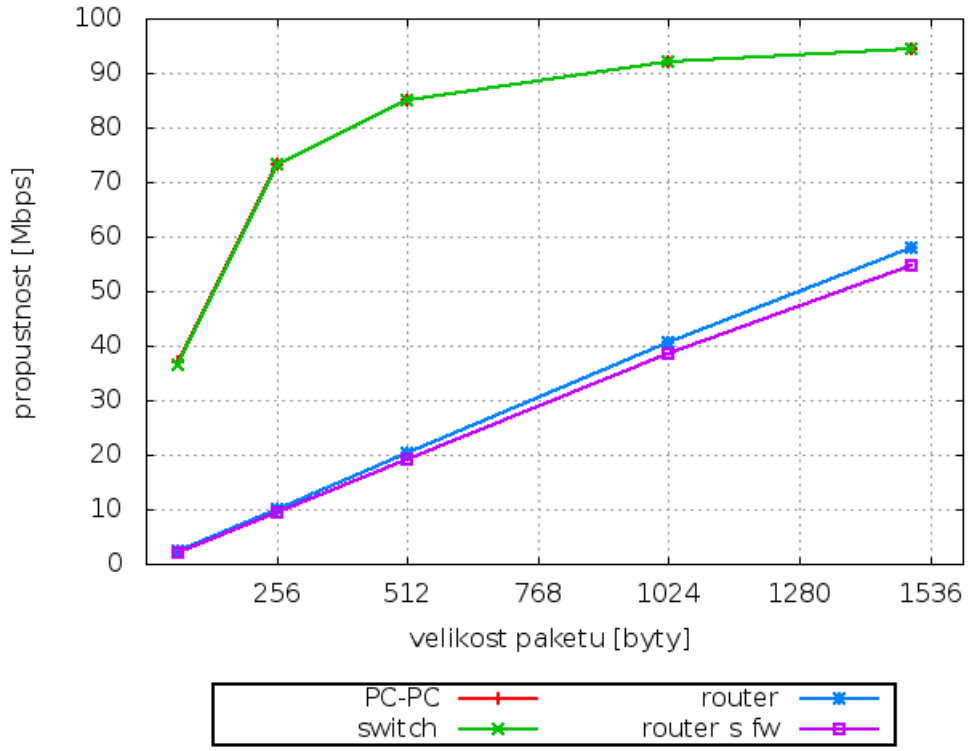
- 2 x PC NetFPGA-Cube
  - procesor: AMD Phenom Quad Core 9650
  - motherboard: Asus M3N78-VM
  - RAM paměť: 4GB – DDR2
  - pevný disk: 500GB SATA
  - síťové rozhraní: Intel Pro/1000 PT Dual Port NIC

6.2 Výsledky testování před úpravou systému

6.2.1 Měření pomocí protokolu TCP

MTU [B]	**PC-PC** [$Mbps$]	**switch** [$Mbps$]	**router** [$Mbps$]	**router s fw** [$Mbps$]
64	37.10	36.60	2.22	2.06
256	73.10	73.10	9.95	9.45
512	84.90	84.90	20.24	19.10
1024	92.10	92.10	40.56	38.42
1500	94.50	94.50	57.80	54.78

Tabulka 6.1: Propustnost při protokolu TCP pro 100Mb



Obrázek 6.3: Graf propustnosti při protokolu TCP pro 100Mb

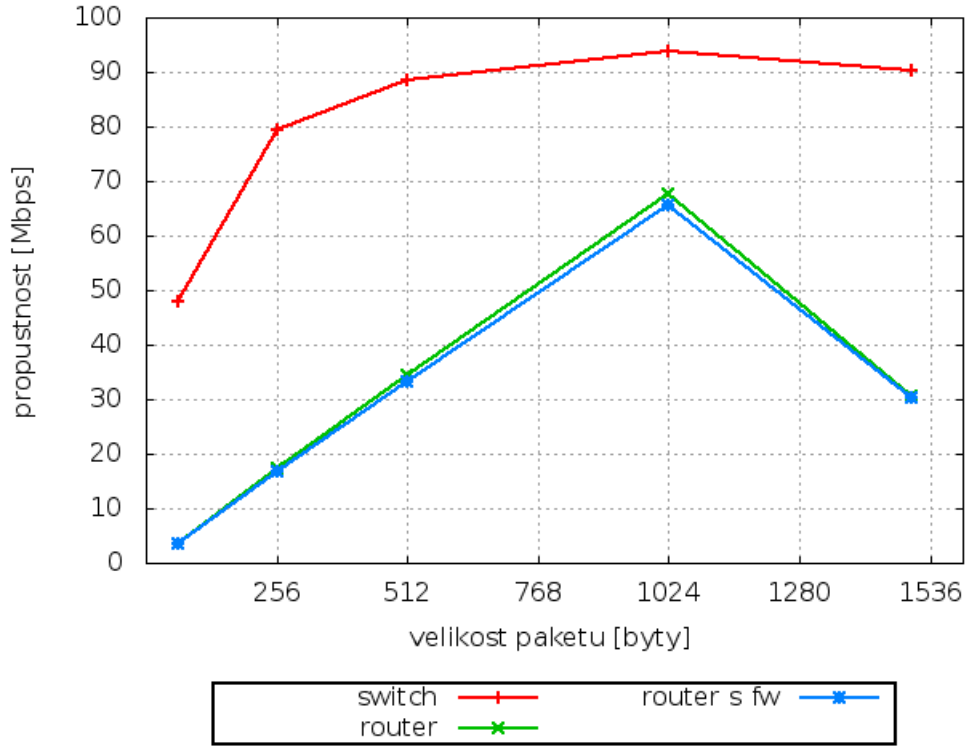
Z grafu propustnosti 6.3 byla potvrzena teorie, že největší propustnost by mělo mít zapojení PC – PC, protože mezi počítači není žádné zařízení manipulující se síťovým tokem. Průběh propustnosti při zapojení PC-PC, překrývá naměřený průběh propustnosti při přepínání. Zde je vidět, že i při přepínání zařízení dosahuje téměř přesně stejného průběhu, jako při zapojení PC-PC. Při směrování již dochází k výraznému omezení propustnosti, která se ještě mírně sníží s použitím firewallu.

Je také zřejmé, že se zmenšující se velikostí paketů klesá i propustnost. Jako optimální velikost paketů se při použití TCP protokolu jeví velké pakety.

6.2.2 Měření pomocí protokolu UDP

MTU [B]	**switch** [$Mbps$]	**router** [$Mbps$]	**router s fw** [$Mbps$]
64	48.00	3.45	3.58
256	79.34	17.38	16.70
512	88.44	34.44	33.14
1024	93.80	67.64	65.54
1500	90.40	30.58	30.22

Tabulka 6.2: Propustnost při protokolu UDP pro 100Mb



Obrázek 6.4: Graf propustnosti při protokolu UDP pro 100Mb

U protokolu UDP zůstalo zachováno pořadí propustnosti jednotlivých zapojení jako u TCP, viz. graf 6.4. Nejvyšší propustnost vykazuje zapojení směrovače při přepínání a následně směrování bez firewallu a s firewallem. Propustnost měřená při použití protokolu TCP je při vzájemném porovnání zapojení nižší, než-li u UDP. Zde je potvrzen výše uvedený fakt, že při použití protokolu TCP je propustnost nepatrně ovlivněna round-trip zpožděním a velikostí TCP okénka, kdežto u protokolu UDP je propustnost ovlivněna pouze round-trip zpožděním.

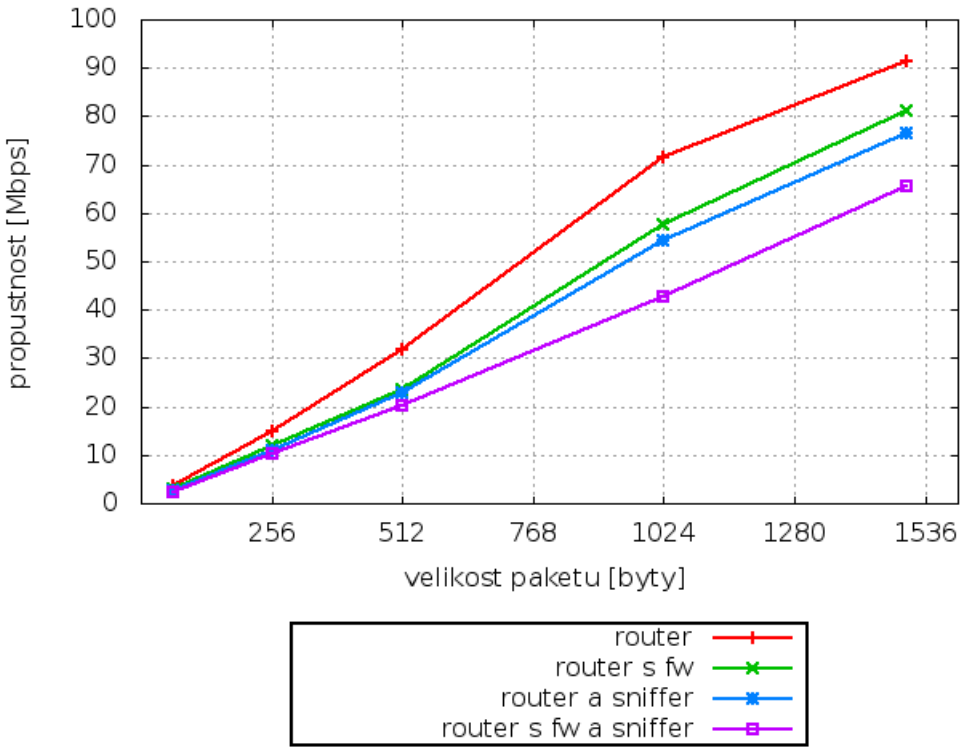
Velice zajímavý je zlom v průběhu naměřené propustnosti routeru a routeru s firewallem. Takovéto chování u protokolu UDP je nestandardní. Zde byl očekáván nárůst propustnosti se zvětšující se velikostí packetů tak, jako u TCP. Podobný průběh lze nalézt i v grafu po úpravě systému 6.6, který je uveden níže.

6.3 Výsledky testování po úpravě systému

6.3.1 Měření pomocí protokolu TCP

MTU [B]	router [Mbps]	router s fw [Mbps]	router a sniffer [Mbps]	router s fw a sniffer [Mbps]
64	3.76	3.13	2.79	2.37
256	14.98	12.04	11.06	10.34
512	31.72	23.52	22.96	20.10
1024	71.64	57.66	54.22	42.62
1500	91.40	81.10	76.62	65.52

Tabulka 6.3: Propustnost při protokolu TCP pro 100Mb po úpravě systému



Obrázek 6.5: Graf propustnosti při protokolu TCP pro 100Mb po úpravě systému

Z naměřených výsledků uvedených v grafu 6.3 vyplývá, že se propustnost při použití OpenWRT celkově zvětšila. Oproti propustnosti při použití firmwaru od výrobce zařízení je téměř dvakrát větší. Podle předpokladů největší propustnosti dosahuje zařízení při směrování s vypnutým firewallem. V porovnání s grafem propustnosti před úpravou systému 6.3 je zde ale patrně větší vliv firewallu na propustnost.

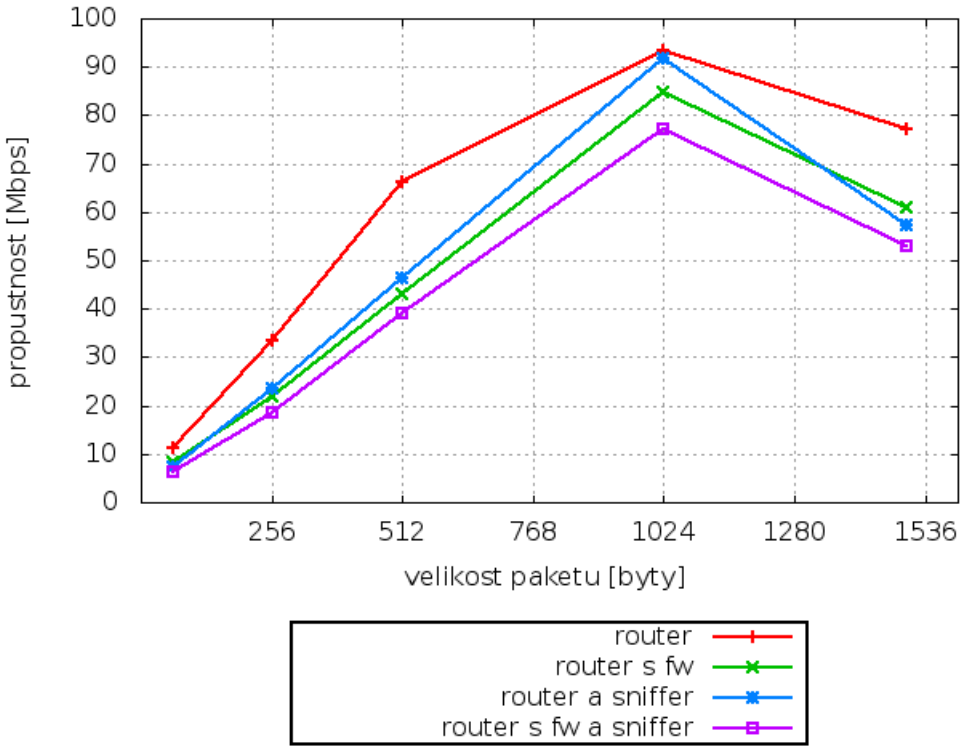
Dále byl také zkoumán vliv odposlouchávacího programu na propustnost zařízení. Podle grafů je patrné, že propustnost při použití tohoto programu klesá. Zde je nutné podotknout, že kvůli malému výkonu procesoru knihovna pcap nestíhá zachytávat mnoho packetů v rych-

lém sledu (tak jak je tomu např. při měření propustnosti pomocí programu Iperf) a ty, které nestihne zachytit, zahodí. V konečném důsledku tento efekt způsobuje patrně vyšší propustnost při použití tohoto progamu, než by měla být.

6.3.2 Měření pomocí protokolu UDP

MTU [B]	**router** [$Mbps$]	**router s fw** [$Mbps$]	**router a sniffer** [$Mbps$]	**router s fw a sniffer** [$Mbps$]
64	11.38	8.32	7.24	6.27
256	33.28	21.76	23.62	18.66
512	66.28	43.06	46.50	39.18
1024	93.50	84.72	91.68	77.02
1500	77.06	60.94	57.12	52.92

Tabulka 6.4: Propustnost při protokolu UDP pro 100Mb po úpravě systému



Obrázek 6.6: Graf propustnosti při protokolu UDP pro 100Mb po úpravě systému

Graf propustnosti po úpravě systému a při použití protokolu UDP 6.6 vykazuje velmi podobný průběh, jako graf před před úpravou systému 6.4. Po úpravě systému, tak jako u propustnosti při použití protokolu TCP, celková propustnost vzrostla téměř dvakrát více. Obdobně jako u TCP, největší propustnosti směrovač dosahuje při směrování s vypnutým firewallem. Se zapnutým firewallem se propustnost při směrování sníží. Pouze orientačně,

z výše uvedených důvodů, byly v grafu uvedeny průběhy propustnosti zařízení Linksys se spuštěným odposlouchávacím programem.

Porovnáním grafů propustnosti před a po úpravě systému 6.4 a 6.6 lze nalézt rozdíl při propadu propustnosti, který nastal u UDP protokolu. Před úpravou systému tento propad činil asi 50%. Po nahrání OpenWRT do směrovače a následném otestování propad činil pouze asi 30%. Zde je vidět, že i přes podivné chování průběhu propustnosti při použití protokolu UDP vykazuje OpenWRT vyšší propustnost, než původní systém.

Kapitola 7

Implementace bezpečnostního algoritmu

7.1 Výběr a popis algoritmu

Vzhledem k tomu, že OpenWRT nám zpřístupňuje samotný směrovač téměř jako počítačovou platformu, lze pro směrovač vytvářet nejrůznější programy. Implementace programů v routeru s OpenWRT je téměř, ne-li úplně, stejná jako pro počítačové platformy. Jak již bylo výše zmíněno, musíme počítat s omezenými prostředky jako je paměť dat nebo operační paměť. Se zásadním rozdílem se setkáváme při překladu. Programy do OpenWRT je třeba překládat překladači pro platformy, na kterých má program fungovat. Portování algoritmu do OpenWRT lze provést dvěma způsoby: kompilace progamu jako balíčku, nebo cross kompilace.

Pro implementaci byl vybrán odposlouchávací algoritmus, který má umožnit sledování síťového toku na směrovači. Využití takového programu bychom našli například v reálném prostředí. V dnešní době je důležité monitorování síťového provozu pro odhalení útoků, anomálií a potenciálních nebezpečí na síti. Profesionální zařízení taková řešení nabízí, avšak u domácích směrovačů často takové možnosti nemáme.

Na program obsahující odposlouchávací algoritmus byly kladeny následující požadavky:

- odposlouchávání pouze určitých IP adres
- ukládání zachyceného síťového provozu do pcap formátu
- zachycený síťový provoz ukládat na počítač (server)
- zachycená data posílat přes šifrované spojení

7.2 Návrh programu

Po výběru algoritmu následovala etapa návrhu výsledného programu. Při návrhu bylo zohledněno, že zachycená data mají být ukládána na počítač – tedy mimo router. Z tohoto hlediska se nabízela možnost vytvořit aplikaci klient – server, kde program v roli serveru bude spuštěn na počítači a ukládat předávaná data do souboru ve formátu pcap.

7.2.1 Klientská část

Pro klientskou část programu bylo navrhnuto následující rozhraní:

- adresa serveru – IP adresa počítače (serveru), na který budou ukládána data
- port serveru – port na, kterém server bude naslouchat
- soubor s IP adresami - pro snazší ovládání programu budou jednotlivé IP adresy umístěny v souboru
- interface – interface, na kterém bude program spuštěn

Klientská část programu bude provádět samotné zachytávání síťového provozu pomocí funkcí knihovny pcap. Aby bylo možné zachytávat pouze vybrané IP adresy, byla pro tento účel navrhnuta stromová struktura Trie v binární podobě. Filtrování vybraných IP adres by bylo možné i v serverové části programu, to by však znamenalo zbytečné generování síťového provozu pro přenos dat na stranu serveru, proto byla zvoleno filtrování na straně klienta. Dále byla navržena funkce pro navázání spojení se serverovou částí programu, která bude využívat BSD socketů. Jedním z dalších požadavků bylo, aby posílaná data byla přenášena přes šifrované spojení. Pro realizaci bylo navrženo použití OpenSSL, které využívá spojení vytvořeného pomocí socketů. Zejména pro přehlednost je v návrhu zahrnuto filtrování komunikace mezi klientem a servrem, která je ukládána do souboru.

7.2.2 Serverová část

U serverové části bylo zvoleno jednoduché rozhraní, které zahrnuje pouze číslo portu, na kterém má být server spuštěn. Pro čtení dat přicházejících od klienta byla navržena funkce, která zaznamená každé nové příchozí spojení od klienta a uloží obsah dat do nového souboru. Zapisování dat ve formátu pcap, zajistí knihovní funkce `pcap_dump()`.

7.3 Implementace

Implementace programu probíhala bez větších změn dle návrhu. V této podkapitole budou popsány důležité části obou programů. Při implementaci byly použity informace z [6] a [7].

7.3.1 Klientská část

Zpracování argumentů příkazové řádky zajišťuje funkce `getParams()`, která ukládá jednotlivé argumenty do struktury `tParams`. Po zpracování parametrů programu je vytvořena stromová struktura Trie, která je implementována v souboru `aux.c`. O naplnění stromové struktury se stará funkce `insert_data_into_trie()`, která zároveň provádí při načítání dat ze souboru jejich validitu. Po inicializační části následuje jádro klientské části, které tvoří zachytávání datového provozu na síti a odesílání těchto dat serverové části.

Zachytávání síťového provozu realizuje funkce `capture_packet_block()`, na jejímž začátku jsou vytvořeny struktury pro šifrované spojení a další potřebné proměnné a struktury. Po inicializaci je vybraný interface nastaven pro zachytávání packetů pomocí knihovní funkce `pcap_open_live()`. Pro spojení klienta se serverem byla vytvořena funkce `connect_to_server()`. Do této funkce je odkazem předána proměnná `my_socket`, která je dále používána pro vytvoření šifrovaného spojení. Po nastavení struktury SSL je proveden

handshake se serverovou částí programu. V případě úspěšného navázání šifrovaného spojení program vstupuje do nekonečné smyčky, která provádí zachytávání síťového toku. Před vstupem do nekonečné smyčky je zjištěna IP adresa a port klientské části pomocí funkce `get_cli_info()`. Tyto údaje jsou důležité pro vyfiltrování komunikace mezi klientem a serverem, která generuje značný provoz a mohla by mást uživatele při prohlížení zachycených dat.

Knihovní funkcí `pcap_next()` jsou v nekonečné smyčce zachytávány jednotlivé packety. Ze zachycených dat jsou zjištěny informace jako např.: IP adresa zdroje, cíle, verze IP protokolu, zdrojový a cílový port. Tyto informace jsou ověřovány proti získaným informacím z funkce `get_cli_info()`. V případě, že zachycený packet obsahuje stejnou IP adresu (zdrojovou nebo cílovou) a zároveň stejný port (zdrojový nebo cílový), není dále packet zpracováván a dochází k dalšímu čtení packetu. V opačném případě dochází k filtrování podle stromové struktury Trie, která obsahuje IP adresy, které mají být sledovány. Jestliže se v této stromové struktuře zachycená IP adresa z packetu nachází, je zachycený packet odeslán serverové části pomocí funkce `SSL_write()`.

Trie

Binární vyhledávací strom Trie je datová struktura pro vyhledávání, při kterém používá jako vodítko bity klíčů uložených v uzlech stromu, které jsou uspořádány [4]. Tato struktura nám umožňuje snadné procházení, vkládání či rušení prvků stromu.

Vlastní implementace vyhledává a vkládá do stromu celé IP adresy, ne IP adresy s prefixy. Uzel stromu je reprezentován strukturou `tTrieNode`, která obsahuje ukazatele na pravý a levý synovský uzel.

Pro vkládání IP adres do Trie byla implementována funkce `TrieInsert()`, která prochází IP adresu v binární podobě bit po bitu a podle hodnoty bitu určuje zda vytvoří pravý nebo levý synovský uzel. Po průchodu IP adresy v binární podobě je vytvořena jedinečná struktura reprezentující danou IP adresu.

Funkce `TrieSearch()` provádí vyhledávání ve vytvořeném vyhledávacím stromu. Při vyhledávání je opět procházena IP adresa v binární podobě bit po bitu. Jestliže narazíme na uzel, který ukazuje na `NULL` a nejsme při procházení binární IP adresy nakonci, není vyhledávaná IP adresa uložena ve stromu a funkce vrací `FALSE`. V případě, že narazíme na uzel, jehož levý a pravý synovský uzel ukazuje na NULL a při procházení binární IP adresy jsme přesně nakonci, nalezli jsme vyhledávanou IP adresu a funkce bude vracet `TRUE`.

7.3.2 Serverová část

Serverovou část programu tvoří jednodušší struktura, než u klientské části. Při zpracovávání parametrů programu je parsováno pouze číslo portu, na kterém bude server spuštěn.

Hlavní část programu vykonává funkce `processing()`. Podobně jako u klientské části programu jsou vytvořeny struktury pro šifrované spojení. Funkce `create_server_socket()` nastaví a vytvoří socket pro spojení s klientem. Následně je funkcí `setup_SSL()` inicializována struktura SSL potřebná pro šifrované spojení. Po této části program vstupuje do nekonečné smyčky. Zde přijímá spojení od klienta, vykoná SSL handshake a vytvoří soubor typu pcap pro zápis s unikátním názvem. Následuje zpracovávání přijímaných dat od klienta. Nejdříve je funkcí `SSL_read()` přijat packet a následně je zapsán funkcí `pcap_dump()` do souboru formátu pcap.

Zpracovávání přijímaných dat probíhá v nekonečné smyčce, která může být přerušena při ukončení klientské části programu. V tomto případě server čeká na další spojení od

klienta. Jestliže dojde k dalšímu spojení s klientem, je vytvořen nový pcap soubor, který je odlišen pořadovým číslem připojení.

7.4 Použití a instalace programu

7.4.1 Kompilace

Před použitím je nutné klientskou a serverovou část programu zkompilovat. Pro kompilaci serverové části využijeme klasického překladače, například gcc. Klientská část musí být přeložena pro cílovou platformu, tedy směrovače Linksys WAG160Nv2. Jestliže jsme stáhli zdrojové kódy OpenWRT a poté je zkompilovali, ve složce:
`/trunk/staging_dir/toolchain_architexture/bin/` najdeme překladač pro naši architekturu[1].

7.4.2 Instalace

Po zkompilování obou částí programu provedeme jejich instalaci. Serverovou část programu postačí umístit na libovolné místo v počítači. Klientskou část spolu se souborem obsahující IP adresy je třeba umístit do směrovače, např. pomocí příkazu `scp`.

7.4.3 Spuštění programu

Nejdříve je nutné spustit serverovou část programu na počítači s parametrem udávající port, na kterém bude naslouchat a očekávat spojení od klientské části. Čísla portů mohou být volena v rozsahu od 0 – 65535. Serverová část programu na výstup zaznamenává příchozí spojení od směrovače a aktuální stav, ve kterém se server nachází. Při chybovém stavu je uživatel informován chybovým hlášením.

Po úspěšném spuštění serverové části může být spuštěna klientská část, která naváže spojení na zadaném portu a IP adrese serverové části. V případě chybně zadaných parametrů programu je uživatel upozorněn chybovým hlášením. Popis jednotlivých parametrů programu je uveden v příloze B.

[1] Viz. přiložené soubory Makefile na CD k bakalářské práci

Kapitola 8

Rozšíření

8.1 Algoritmické testování výkonnosti procesoru

V rámci bakalářské práce jsem se zapojil do projektu výzkumné skupiny ANT@FIT, jehož cílem bylo vytvoření sady nástrojů pro analýzu výkonnosti vestavěných procesorů. Mým úkolem bylo portování naimplementovaných algoritmů do OpenWRT a následné otestování procesoru.

V následujícím textu budou popsány implementované algoritmy a postup při portování algoritmů do OpenWRT. Dále budou uvedeny vybrané výsledky z testování.

8.1.1 Algoritmy

Z algoritmů určených k testování vznikl tzv. toolset. Ten zahrnuje následující algoritmy:

LPM – Longest Prefix Match

TreeBitmap
Vychází z konceptu binární struktury Trie, kterou se snaží zefektivnit tím, že nevytváří Trie po jednom uzlu, ale po tzv. multiuzlech. Každý z těchto multiuzlů reprezentuje tvarově stejný podstrom, konkrétně takový úplný binární podstrom s danou výškou. Tento algoritmus nemusí tedy procházet stromovou stukturu po jednom uzlu, což výrazně snižuje počet přístupů do paměti.

Shape-Shifting Trie
Tento algoritmus také vychází z binární struktury Trie. Je to jakýsi obecný typ TreeBitmap, kde jsou uzly také spojeny do multiuzlů. Tento strom však nemá pevně danou strukturu multiuzlu, respektive ji má uloženou přímo v multiuzlu. Algoritmus vyhledávání prefixu a paměťová náročnoust jsou tedy podobné jako u TreeBitmap. Výhodou proti předchozímu algoritmu je, že při řidších prefixových sadách je vytvořeno méně uzlů a výška tohoto stromu je také nižší co se týče počtu multiuzlů.

Filters

Bloom Filter
Bloomův filter je pravděpodobnostní struktura, která přibližně reprezentuje množinu. Tento základní algoritmus Bloomova filtru umožňuje dvě operace: přidání a dotazování se na prvek. Při přidávání je prvek zpracován několika hashovacími funkcemi, které určí kam má

být prvek vložen – zapíše jedničky na adresy bitového pole. U dotazování dochází opět k hashování, po kterém jsou čteny jedničky z adresy. Jestliže jsou přečteny všechny jedničky, prvek do množiny pravděpodobně patří, jinak určitě ne.

Counting Bloom Filter
U čítacího Bloomova filtru můžeme navíc prvky i odebírat. Odebírání je implementováno pomocí čítačů, které mají omezený rozsah.

Informace o předchozích algoritmech byly čerpány z: [2], [3] a [5].

8.1.2 Portování algoritmů

Algoritmy jsou napsané v jazyce C tak, aby byly přenositelné na různé platformy. Jestliže chceme algoritmy na směrovači spustit, je třeba celý toolset zkompilovat pro danou architekturu směrovače. V tomto případě lze využít balíčkového systému OpenWRT, který provede kompilaci.

V adresáři `/trunk/package` vytvoříme novou složku `toolset`, do níž bude umístěn Makefile pro kompilaci celého balíčku. Tento Makefile má specifickou strukturu, která musí být dodržena, aby bylo možné celý balíček „zakomponovat“ do OpenWRT. Ze souboru Makefile, který je přiložen na CD k bakalářské práci je vidět, že postačí změnit cestu k toolsetu. Žádné jiné úpravy není nutné dělat.

Poté už můžeme balíček zkompilovat a nainstalovat do OpenWRT:

```
make package/toolset/compile
make package/toolset/install
make
```

Aby nebylo nutné kvůli balíčku přepisovat celou paměť flash, je možné zkompilovaný balíček zkopírovat do směrovače a pomocí balíčkového systému opkg nainstalovat:

```
#zkopíruje balíček do směrovače
scp ~/trunk/bin/brcm63xx/packages/toolset.ipk root@IP_ROUTER:/

#instalace ve směrovači
opkg install toolset.ipk
```

Toto řešení předpokládá nastavené heslo pro uživatele root ve směrovači.

8.1.3 Testování a výsledky

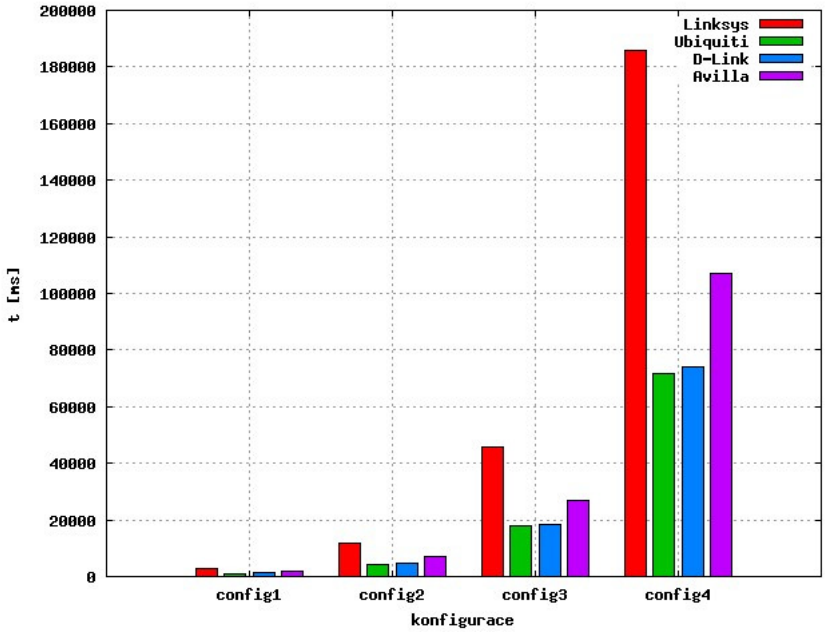
Samotné testování již probíhalo pomocí vytvořených skriptů, které spouštěly algoritmy s různou konfigurací. Faktorem pro otestování výkonnosti procesoru byl čas udávající dobu běhu algoritmu na daném procesoru. Výsledky testů všech procesorů jsou dostupné na Wiki vyzkumné skupiny ANT@FIT. V následující části práce bude uveden seznam testovaných směrovačů (tabulka 8.1) a uvedeny grafy některých výsledků testování.

Bloom filter

Jak je patrné z grafů 8.1 a 8.2 výkonnosti procesorů při použití Bloomova filtru, je čas strávený vykonáváním algoritmu přímo úměrný výkonnosti směrovačů.

Platforma	Procesor	Frekvence CPU	Jádro	Hlavní paměť
Linksys WAG 160N	Broadcom BCM6538	300 MHz	MIPS	32 MB
D-Link DIR-825	Atheros AR7161	680 MHz	MIPS	64 MB
Avila GW2348	Intel IXP425	533 MHz	ARMv5	64 MB
Ubiquiti	Atheros AR7161	680 MHz	MIPS	128 MB

Tabulka 8.1: Přehled testovaných směrovačů



Obrázek 8.1: Graf výkonnosti směrovačů - Bloom filter - operace insert

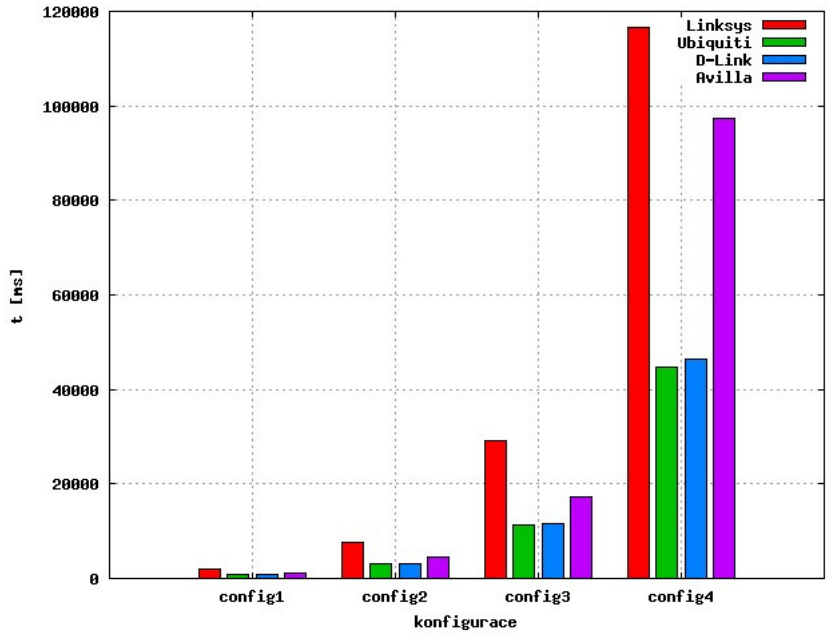
Nejvýraznější výkonnostní rozdíl lze zaznamenat u čtvrté konfigurace, který se projevil u obou výše uvedených grafů. Průběhy u ostatních konfigurací, jsou podobné. Mění se pouze doba vykonávání algoritmu při dané konfiguraci.

Jako nejvýkonější zařízení při operaci Insert u Bloomova filtru (graf 8.1) můžeme označit zařízení D-Link a Ubiquiti. Výsledky u bloomových filtrů jsou ovlivněny výkonností procesoru a propustností paměti. Tudíž i při dvojnásobné hlavní paměti zařízení Ubiquiti je průběh výkonnosti srovnatelný se zařízením D-Link.

Podobný průběh nastává i při operaci Querry u Bloomova filtru graf (8.2). Zajímavý je však průběh u čtvrté konfigurace. Zde si zařízení Avilla výrazně pohoršilo oproti předchozím výsledkům i přes svůj téměř dvakrát větší výkon, než má zařízení Linksys.

Counting Bloom filter

Počítaný Bloomův filtr umožňuje prvky i odebírat. U tohoto algoritmu byla navíc testována výkonnost při odebírání prvku a při dalším dotazu. Průběh všech čtyř grafů odpovídá výkonnosti jednotlivých zařízení, kde v každém grafu je dodržen „tvarový průběh“ pouze s časovým rozdílem operací.



Obrázek 8.2: Graf výkonnosti směrovačů - Bloom filter - operace querry

Při porovnání časových náročností operací se jako nejnáročnější jeví operace insert (graf 8.3) a následně dotaz na vybraný prvek (graf 8.4).

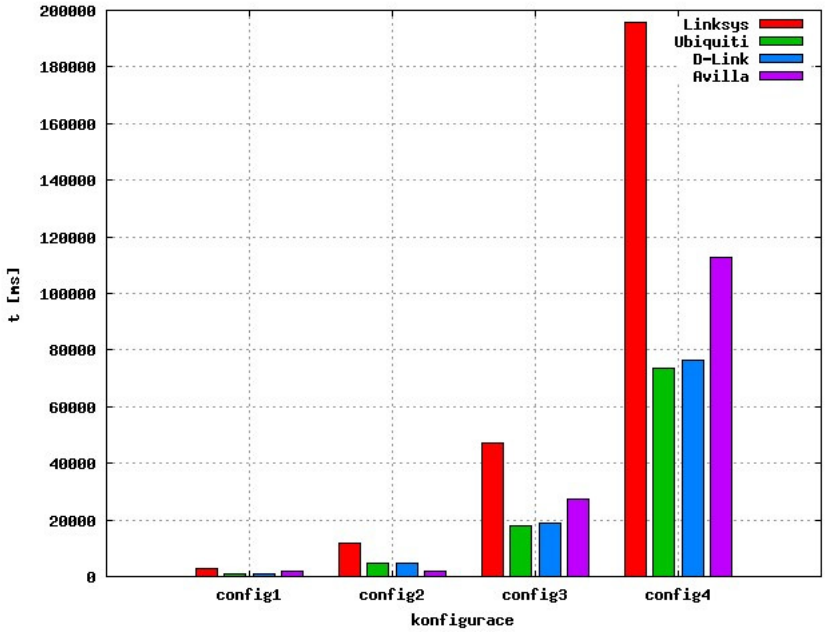
Zajímavé je srovnání počítaného Bloomova filtru a „obyčejného“ Bloomova filtru při operacích insert a querry. Zde je patrné, že počítaný Bloomův filter je nepatrně náročnější, než „obyčejný“ Bloomův filtr.

Shape-Shifting Trie

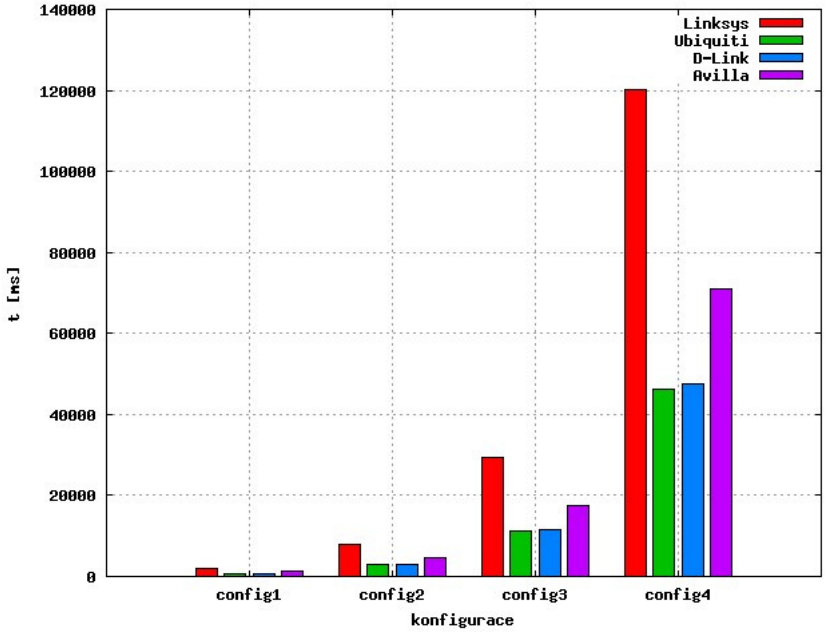
U algoritmu Shape-Shifting Trie dochází k zajímavé konfrontaci mezi zařízeními Ubiquiti a D-Link (graf 8.7). Navzdory předešlým výsledkům výkonnosti testovaných zařízení Ubiquiti vykazuje vyšší výkonnost než D-Link. Jednoznačně nejhůře skončilo zařízení Linksys, kterému vykonávání algoritmu zabralo nejdelší dobu.

TreeBitmap

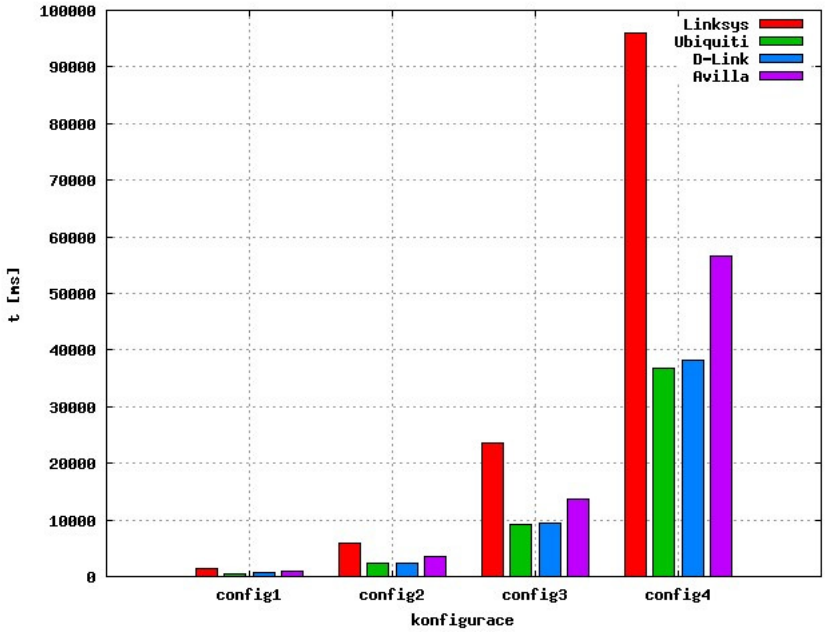
Dalším z testovaných algoritmů LPM byl TreeBitmap (graf 8.8). U totoho typu algoritmu vyniká snad jen velice dlouhá doba vykonávání algoritmu zařízení Linksys. Oba druhy algoritmů – TBM4 a TBM5 mají téměř totožný průběh pouze s malými časovými odchylkami.



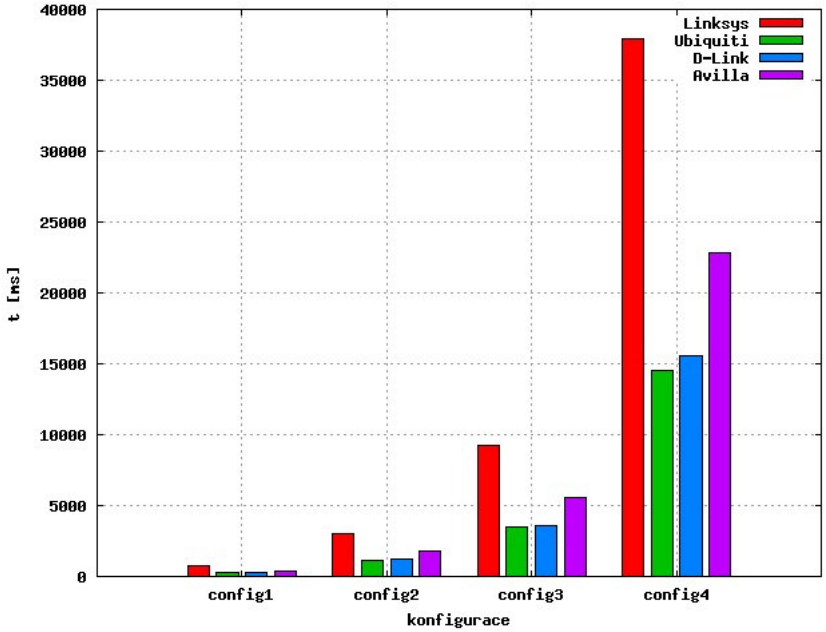
Obrázek 8.3: Graf výkonnosti směrovačů - Counting Bloom filter - operace insert



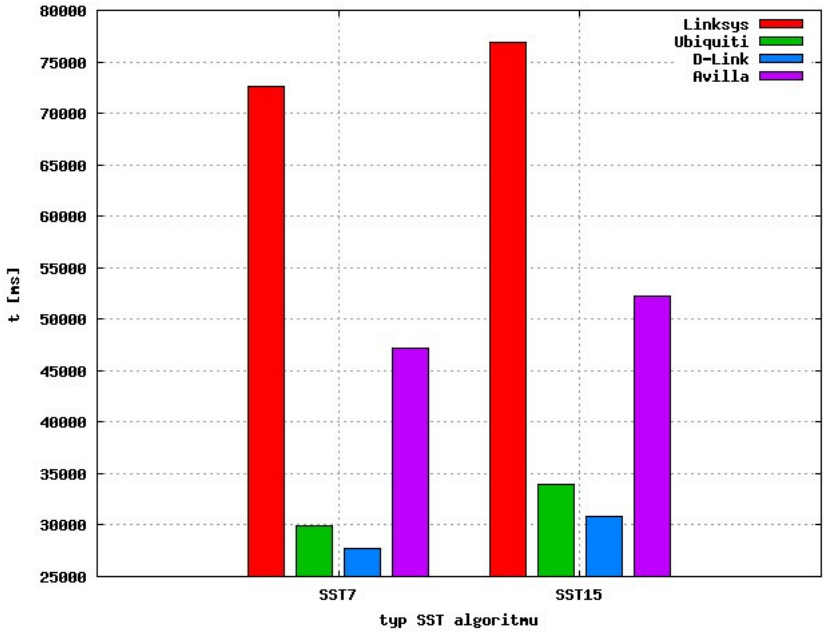
Obrázek 8.4: Graf výkonnosti směrovačů - Counting Bloom filter - operace querry



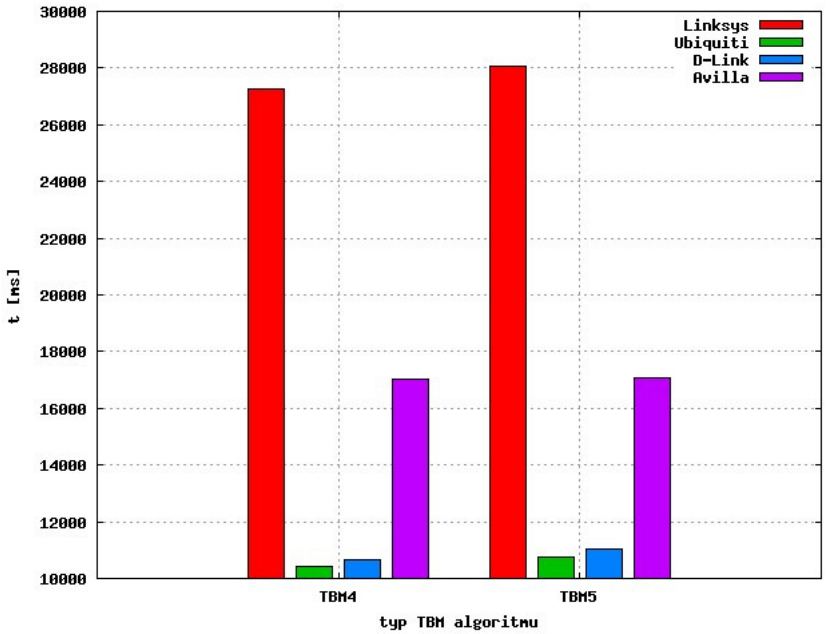
Obrázek 8.5: Graf výkonnosti směrovačů - Counting Bloom filter - operace remove



Obrázek 8.6: Graf výkonnosti směrovačů - Counting Bloom filter - operace querry2



Obrázek 8.7: Graf výkonnosti směrovačů - Shape-Shifting Trie



Obrázek 8.8: Graf výkonnosti směrovačů - TreeBitmap

Kapitola 9

Závěr

Veškerá technologie v dnešní době kráčí kupředu a mnoho z těchto technologií je zpřístupňováno pro širokou veřejnost. Tomuto trendu se nevyhla ani síťová zařízení, která se s příchodem internetu začla pozvolna objevovat na obchodních trzích. Obrovský boom zažívají zejména domácí směrovače, které jsou dnes hojně využívány pro bezdrátové připojení k internetu. Jejich funkčnost a využití lze navíc se systémem OpenWRT rozšířit nad rámec možností systémů, které dodávají jejich výrobci.

V této práci jsem popsal systém OpenWRT, jeho možnosti a využití. Dále jsem se zaměřil na měření propustnosti, které bylo provedeno k porovnání výkonnosti směrovače při použití firmwaru výrobce a při použití OpenWRT. Z výsledků měření jasně vyplynulo, že směrovač dosahuje vyšší propustnosti po úpravě systému, tedy s OpenWRT. Práce také obsahuje popis instalace systému OpenWRT do směrovače Linksys WAG160Nv2. Před instalací bylo nutné vyrobit sériový TTL převodník, který umožňuje neustálou kontrolu činnosti směrovače. Při řešení instalace OpenWRT do směrovače Linksys nebyl tento směrovač přímo podporován, proto byly provedeny úpravy ve zdrojových kódech OpenWRT, aby bylo možné tento systém na směrovači alespoň spouštět. Praktická část byla završena implementací programu s odposlouchávacím algoritmem, který zachytává síťový provoz na směrovači a ukládá jej přes šifrované spojení na vzdálený počítač ve formátu pcap. Odposlouchávací program je schopný zachytávat téměř všechen veškerý síťový provoz, až na rychlé shluky velkých dat[1], kde již nepostačuje výkon procesoru směrovače a knihovna pcap packety zahazuje.

Jako jedno z možných rozšíření bych navrhoval úpravu programu pro podporu IP protokolu verze 6. Například implementace vhodné struktury pro vyhledávání adres k odposlechu, nastavení socketů pro výběr komunikace při protokolu IPv4 nebo IPv6, konfigurace podpory pro IPv6 ve směrovači.

Práce na tomto tématu pro mě byla velmi zajímavá a přínosná jak ze studijního, tak i z osobního hlediska. Seznámil jsem se s novým typem linuxového operačního systému, kterému bych chtěl nadálě věnovat pozornost a využít jeho vlastnosti na maximum, například v naší domácí síti. Jako velké plus hodnotím také možnost zapojení se do projektu algoritmického testování procesorů výzkumné skupiny ANT@FIT, které mě pobídlo k objevování jiných zákoutí OpenWRT a vývoje softwaru, než s jakými bych se setkal při samotné tvorbě této bakalářské práce.

[1] Například: přenos souborů mezi dvěma počítači připojených k routeru

Literatura

[1] Broadcom: BCM 6358 Broadcom Product Brief [online]. 2006-06-05 [cit. 2011-02-21]. URL http://www.datasheetdir.com

[2] Broder, A.; Mitzenmacher, M.: Network Applications of Bloom Filters: A Survey. *Internet Mathematics*, ročník 1, 2004: s. 485–509, ISSN 1542-7951.

[3] Eatherton, W.; Varghese, G.; Dittia, Z.: Tree Bitmap. *SIGCOMM Computer Communication Review*, ročník 34, č. 2, April 2004: s. 97–122, ISSN 0146-4833. URL http://portal.acm.org/citation.cfm?doid=997150.997160

[4] SEDGEWICK, R.; GREE, J.: *Algoritmy v C*. Praha : SoftPress, 2003, ISBN 80-864-9756-9.

[5] Song, H.; Turner, J.; Lockwood, J.: Shape Shifting Tries for Faster IP Route Lookup. In *Proceedings of the 13TH IEEE International Conference on Network Protocols*, Washington, USA: IEEE Computer Society, 2005, ISBN 0-7695-2437-0, s. 358–367. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1544635

[6] STEVENS, R. W.; FENNER, B.; RUDOFF, A. M.: *Unix Network Programming*. Addison-Wesley, 2004, ISBN 0-13-141155-1.

[7] VARGHESE, G.: *Network Algorithmics, An interdisciplinary approach to design fast networked devices*. Amsterdam : Elsevier, 2005, ISBN 0-12-088477-1.

[8] WWW stránky: Cisco Acquires Linksys for $500M. URL http://www.wi-fiplanet.com/news/article.php/2119751

[9] WWW stránky: Cisco Networking Academy - Cisco Systems. URL http://www.cisco.com/web/learning/netacad/index.html

[10] WWW stránky: Linksys by Cisco Wireless-N ADSL2+ Modem Router WAG160N. URL http://www.linksysbycisco.com/ANZ/en/products/WAG160N

[11] WWW stránky: Mips Technologies - Architectures. URL http://www.mips.com/products/architectures/

[12] WWW stránky: OpenWRT wiki - Wireless Freedom. URL http://wiki.openwrt.org

[13] WWW stránky: Request for Comments (RFC). URL http://www.ietf.org/rfc.html

[14] Yildirim, E.; Suslu, I. H.; Kosar, T.: GRID '08 Proceedings of the 2008 9th IEEE/ACM International Conference on Grid Computing. *Grid Computing*, September 2008: s. 266 – 275, ISBN 978-1-4244-2578-5.
URL http://portal.acm.org/citation.cfm?id=1599020.1599064

Příloha A

Tabulky naměřených hodnot

A.1 Před úpravou systému

Tabulka A.1: Propustnost při protokolu TCP, 100Mbit Ethertnet

MTU	Naměřená rychlost pro TCP [Mbps]					průměr
PC – PC (100Mbit)						
64	37.5	36.9	36.7	36.9	37.5	37.10
256	73.3	73.1	73.1	73.1	73.1	73.10
512	84.9	84.9	84.9	84.9	84.9	84.90
1024	92.1	92.1	92.1	92.1	92.1	92.10
1500	94.5	94.5	94.5	94.5	94.5	94.50
Linksys WAG160N jako switch						
64	36.6	36.6	36.6	36.6	36.6	36.60
256	73.1	73.1	73.1	73.1	73.1	73.10
512	84.9	84.9	84.9	84.9	84.9	84.90
1024	92.1	92.1	92.1	92.1	92.1	92.10
1500	94.5	94.5	94.5	94.5	94.5	94.50
Linksys WAG160N jako router bez firewallu						
64	2.21	2.22	2.22	2.23	2.22	2.22
256	9.83	9.84	9.97	10.1	10.0	9.95
512	20.1	20.3	20.3	20.3	20.2	20.24
1024	40.2	40.7	40.7	40.6	40.6	40.56
1500	57.5	57.8	57.9	57.8	58.00	57.80
Linksys WAG160N jako router s firewallem						
64	3.88	3.89	3.93	3.94	3.92	3.91
256	16.1	16.2	16.3	16.2	16.3	16.22
512	31.4	32.0	31.8	31.9	32.0	31.82
1024	59.6	58.2	58.3	58.8	56.9	58.36
1500	80.4	80.1	78.9	80.4	81.2	80.20

Tabulka A.2: Propustnost při protokolu UDP, 100Mbit Ethertnet

MTU	Naměřená rychlost [Mbps]					Šířka pásma [Mbps]					Ztrátovost					průměr	průměr šířky. p.	průměr ztrát.
Linksys WAG160N jakou switch																		
64	48.3	47.9	48.0	47.9	47.9	184.0	198.0	201.0	186.0	201.0	0.74	0.75	0.76	0.74	0.76	48.00	194.00	0.75
256	79.3	79.3	79.3	79.5	79.3	79.4	79.4	79.4	79.5	79.4	0.06	0.07	0.00	0.00	0.00	79.34	79.42	0.03
512	88.4	88.6	88.4	88.4	88.4	88.5	88.6	88.6	88.6	88.5	0.00	0.00	0.22	0.22	0.00	88.44	88.56	0.09
1024	93.8	93.8	93.8	93.8	93.8	93.8	93.8	93.9	93.8	94.0	0.00	0.00	0.00	0.00	0.15	93.80	93.86	0.03
1500	90.4	90.4	90.4	90.5	90.3	92.3	92.3	92.3	92.4	92.2	0.00	0.00	0.00	0.00	0.02	90.40	92.30	0.00
Linksys WAG160N jakou router bez firewallu																		
64	4.3	4.3	2.9	2.9	2.8	213.0	200.0	196.0	197.0	201.0	0.98	0.98	0.96	0.96	0.96	3.45	201.40	0.97
256	17.3	17.4	17.4	17.4	17.4	17.4	17.5	17.4	17.4	17.5	0.18	0.18	0.18	0.18	0.17	17.38	17.44	0.18
512	34.4	34.5	34.4	34.5	34.4	34.5	34.6	34.5	34.6	34.5	0.17	0.16	0.14	0.13	0.17	34.44	34.54	0.15
1024	66.7	68.0	68.2	67.3	68.0	66.8	68.3	68.4	67.4	68.2	0.13	0.18	0.11	0.10	0.15	67.64	67.82	0.13
1500	30.8	30.6	30.6	30.6	30.3	31.5	31.3	31.4	31.3	31.0	0.06	0.05	0.05	0.06	0.05	30.58	31.30	0.05
Linksys WAG160N jakou router s firewallem																		
64	3.9	3.74	3.7	3.8	2.7	206.0	191.0	203.0	206.0	2.71	0.98	0.98	0.96	0.96	0.99	3.58	161.74	0.98
256	16.7	16.7	16.7	16.7	16.7	16.8	16.7	16.7	16.8	16.7	0.00	0.00	0.16	0.17	0.00	16.70	16.74	0.07
512	33.1	33.2	33.1	33.2	33.1	33.2	33.3	33.2	33.3	33.1	0.16	0.17	0.16	0.00	0.17	33.14	33.22	0.13
1024	65.4	65.4	65.5	65.9	65.5	65.6	65.6	65.7	66.0	65.5	0.15	0.16	0.15	0.00	0.11	65.54	65.68	0.11
1500	30.2	30.2	30.3	30.3	30.1	30.9	31.1	31.0	31.0	30.1	0.05	0.05	0.00	0.00	0.03	30.22	30.82	0.03

A.2 Po úpravě systému

Tabulka A.3: Propustnost při protokolu TCP po úpravě systému, 100Mbit Ethertnet

MTU	Naměřená rychlost pro TCP [Mbps]					průměr
Router bez firewallu						
64	3.81	3.11	3.93	3.80	4.17	3.76
256	11.70	16.10	13.80	16.10	17.20	14.98
512	33.30	31.20	30.20	30.60	33.30	31.72
1024	72.30	70.60	72.60	71.40	71.30	71.64
1500	91.40	91.40	91.40	91.40	91.40	91.40
Router s firewallem						
64	3.33	3.09	3.25	3.09	2.91	3.13
256	10.70	12.90	12.30	12.80	11.50	12.04
512	21.90	22.30	23.00	23.10	27.30	23.52
1024	56.90	58.30	58.40	57.00	57.70	57.66
1500	80.10	82.10	81.90	80.40	81.00	81.10
Router bez firewallu a sniffer						
64	2.65	2.90	2.91	2.75	2.73	2.79
256	11.50	10.30	10.30	12.10	11.10	11.06
512	24.10	19.70	23.70	22.80	24.50	22.96
1024	54.30	52.60	53.20	55.70	55.30	54.22
1500	75.00	77.30	76.70	76.50	77.60	76.62
Router s firewallem a sniffer						
64	2.21	2.14	2.14	2.75	2.60	2.37
256	11.00	10.10	10.40	10.20	10.00	10.34
512	20.40	20.70	22.70	19.10	17.60	20.10
1024	45.00	41.70	43.00	40.90	42.50	42.62
1500	66.40	63.90	66.20	65.40	65.70	65.52

Tabulka A.4: Propustnost při protokolu UDP po úpravě systému, 100Mbit Ethertnet

MTU	Naměřená rychlost [Mbps]					Šířka pásma [Mbps]					Ztrátovost					průměr	průměr šířky. p.	průměr ztrát.
Router bez firewallu																		
64	11.7	10.6	11.8	11.2	11.6	102.0	102.0	102.0	75.0	92.2	0.88	0.89	0.88	0.85	0.87	11.38	94.64	0.87
256	33.5	32.7	33.3	33.5	33.4	65.6	77.6	78.5	78.5	78.5	0.49	0.57	0.57	0.56	0.56	33.28	75.74	0.55
512	65.0	67.2	67.0	66.7	65.5	87.9	88.0	88.0	88.0	87.9	0.24	0.24	0.24	0.24	0.24	66.28	87.96	0.24
1024	93.5	93.5	93.5	93.5	93.5	93.6	93.6	93.6	93.6	93.6	0.00	0.00	0.00	0.00	0.15	93.50	93.60	0.00
1500	76.6	77.2	77.5	76.6	77.4	91.9	91.9	92.0	92.0	92.0	0.15	0.14	0.14	0.15	0.14	77.06	91.96	0.14
Router s firewallem																		
64	7.1	7.5	9.1	8.9	9.0	102.0	102.0	102.0	102.0	102.0	0.93	0.93	0.91	0.91	0.91	8.32	102.00	0.91
256	22.0	22.1	21.5	21.6	21.6	78.5	78.5	78.5	78.5	78.5	0.72	0.72	0.72	0.72	0.72	21.76	78.50	0.72
512	43.6	42.8	42.8	43.6	42.5	88.0	88.0	88.0	88.0	87.9	0.50	0.50	0.50	0.50	0.50	43.06	87.98	0.50
1024	85.3	83.1	84.6	85.7	94.9	93.6	93.6	93.6	93.6	93.6	0.09	0.09	0.10	0.08	0.09	84.72	93.60	0.08
1500	61.1	61.7	60.4	59.8	61.7	91.9	92.0	92.0	91.9	91.9	0.32	0.31	0.31	0.31	0.31	60.94	91.94	0.31
Router bez firewallu a sniffer																		
64	6.8	7.4	7.3	7.4	7.3	102.0	102.0	102.0	102.0	102.0	0.93	0.93	0.93	0.93	0.93	7.24	102.00	0.93
256	23.3	23.5	23.5	23.9	23.9	78.5	78.5	78.5	78.5	78.5	0.70	0.69	0.69	0.70	0.69	23.62	78.50	0.68
512	45.8	47.0	47.2	46.8	45.7	88.0	88.0	87.9	88.0	88.0	0.47	0.47	0.46	0.47	0.47	46.50	87.98	0.47
1024	91.6	91.9	91.6	92.0	91.3	93.6	93.6	93.6	93.6	93.6	0.02	0.02	0.02	0.02	0.02	91.68	93.60	0.02
1500	57.1	57.4	57.0	57.2	56.9	92.0	91.9	91.9	91.9	91.9	0.35	0.35	0.35	0.35	0.35	57.12	91.92	0.35
Router s firewallem a sniffer																		
64	5.9	6.3	6.4	6.34	6.5	102.0	102.0	102.0	102.0	102.0	0.94	0.94	0.94	0.94	0.94	6.27	102.00	0.94
256	17.7	19.8	20.2	17.8	17.8	81.0	78.5	78.5	81.0	78.5	0.78	0.74	0.74	0.78	0.78	18.66	79.50	0.76
512	38.9	39.0	38.6	39.7	39.7	88.0	87.9	88.0	88.0	87.9	0.55	0.55	0.55	0.55	0.55	39.18	87.96	0.55
1024	76.8	77.1	76.9	77.3	77.0	93.6	93.7	93.6	93.6	93.7	0.18	0.18	0.18	0.17	0.18	77.02	93.64	0.18
1500	52.3	52.5	53.8	52.3	53.7	92.0	91.9	91.9	91.9	91.9	0.41	0.40	0.40	0.41	0.41	55.92	91.92	0.40

Příloha B

Popis parametrů programu

Klientská část

```
client_sniffer -s <SERVER_IP> -p <SERVER_PORT> -i <INTERFACE> -f <CONF_FILE>

--help       vypíše nápovědu
-s           IP adresa serveru
-p           číslo portu serveru, na kterém naslouchá
-i           interface, na kterém je program spuštěn
-f           název souboru s IP adresami k odposlechu
```

Serverová část

```
client_sniffer -p <SERVER_PORT>

--help       vypíše nápovědu
-p           číslo portu serveru, na kterém má být spuštěn
```

Příloha C

Obsah CD

- písemná zpráva ve formátu pdf
- zdrojový tvar písemné zprávy
- zdrojové kódy programu
- soubor README s návodem k instalaci a spuštění programu
- soubor s aktuální konfigurací směrovače
- soubor `ath_data` se zálohou kalibračních dat pro Wi-Fi
- image s vybuildovaným OpenWRT
- image s originálním firmwarem pro obnovu
- soubor `/etc/network` a `/etc/firewall` k nastavení směrovače
- makefile pro balíčky do OpenWRT
- soubory k úpravě OpenWRT