VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

PC JAKO HRANIČNÍ SMĚROVAČ POSKYTOVATELE SLUŽEB INTERNETU

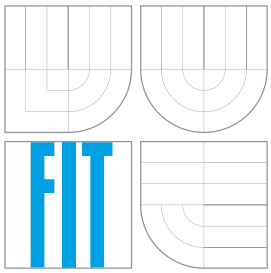
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

MICHAL NOWAK

BRNO 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

PC JAKO HRANIČNÍ SMĚROVAČ POSKYTOVATELE SLUŽEB INTERNETU

PC AS A BORDER ROUTER OF AN INTERNET SERVICE PROVIDER

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — MICHAL NOWAK
AUTHOR

VEDOUCÍ PRÁCE — Mgr. MAREK RYCHLÝ
SUPERVISOR

BRNO 2007

Abstrakt

Práce se snaží zmapovat strukturu sítě středně velkého poskytovatel připojení k internetu v okolí Rousínova a zjistit požadavky poskytovatele na funkčnost směrovače. Práce především diskutuje zvolenou distribuci operačního systému a výběr služeb, především vzhledem k výkonu počítače, na kterém se implementuje softwarový směrovač a stabilitě daných služeb. Pro směrovač je implementováno připojení k internetu skrz protokol IPv6, přesto lze předpokládat, že naprostá většina provozu půjde přes IPv4. Součástí práce je jednoduchá autentizace klientů spojená s blokací těch neznámých, řízení šířky pásma a použití více WAN spojení.

Klíčová slova

Linux, Debian, směrovač, router, ISP, Rousínovsko.net, řízení šířky pásma, transparentní SMTP/POP3/IMAP proxy

Abstract

This Bachelor Thesis tries to investigate structure of a network of medium size Internet Service Provider in Rousínov and surroundings, as well as tries to study demands of ISP that belongs to router's functionality. Thesis discuss mainly choosen distribution of the Linux operating system and used system services with regard to computer and system services performace and stability. Anyway an absolute majority of bandwidth will belong to IPv4 traffic, Router is IPv6 capable throught Tunnel Broker. Another components of thesis are basic authentication, blocking of unknown clients, traffic shaping and basics of theory about WAN interfaces bonding.

Keywords

Linux, Debian, router, ISP, Rousinovsko.net, traffic shaping, transparent SMTP/POP3/IMAP proxy

Citace

Michal Nowak: PC jako hraniční směrovač poskytovatele služeb Internetu, bakalářská práce, Brno, FIT VUT v Brně, 2007

PC jako hraniční směrovač poskytovatele služeb Internetu

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Mgr. Marka Rychlého

.....
Michal Nowak
13. května 2007

Poděkování

Rád bych poděkoval panu Ing. Martinu Molákovi, zástupci ISP Rousínovsko.net, za pomoc při výběru vhodných služeb pro směrovač a za pomoc při nasazování do testovacího provozu.

Obsah

1 Úvod **5**

2 Současný stav sítě, volba hardware a distribuce operačního systému **6**
2.1 Současný stav sítě . . . 6
2.2 Hardware . . . 7
2.2.1 Výběr datového média . . . 7
2.3 Volba souborového systému . . . 8
2.4 Realizace snahy o minimální množství zápisů na médium . . . 8
2.5 Diskuze: specializovaná vs. obecná distribuce . . . 9
2.5.1 Výhody z pohledu administrátora . . . 9
2.5.2 Nevýhody z pohled administrátora . . . 9
2.6 Volba distribuce operačního systému . . . 10
2.6.1 Fáze první: Gentoo Linux . . . 10
2.6.2 Fáze druhá: Debian GNU/Linux . . . 11
2.6.3 Instalace operačního systému . . . 11

3 Výběr a nastavení služeb **13**
3.1 PAT - Port Address Translation . . . 13
3.2 Výběr vhodných služeb . . . 14
3.2.1 SSH server/klient . . . 15
3.2.2 Plánovač úloh Cron . . . 16
3.2.3 Syslog, zaznamenávání chodu systému . . . 17
3.2.4 DNS cache . . . 17
3.2.5 DHCP server . . . 18
3.2.6 Transparentní SMTP/POP3/IMAP proxy . . . 19
3.2.7 Antivirový software . . . 20
3.2.8 Antispamový software . . . 20
3.2.9 Podpora IPv6 v operačním systému a přidělování IPv6 adres . . . 20
3.2.10 Web server pro nenadálé situace . . . 22

4 Autentizace klientů, účtování provozu a řízení šířky pásma **24**
4.1 Autentizace klientů . . . 24
4.2 Skript `gen.sh` . . . 25
4.3 Účtování provozu . . . 25
4.3.1 MRTG . . . 25
4.3.2 Monitorování systémových prostředků . . . 25
4.3.3 Monitorování provozu uživatelů sítě . . . 26
4.4 Řízení šířky pásma . . . 27

4.4.1 Volby ovlivňující šířku pásma . . . 28
4.4.2 Značkování paketů . . . 28
4.4.3 Další metody řízení šířky pásma . . . 29
4.4.4 Řízení šířky pásma na dle použitého aplikačního protokolu . . . 29
4.5 Využití více WAN spojení – prevence výpadku . . . 29
4.5.1 Parametry modulu `bonding.ko` . . . 30
4.5.2 Praktické použití . . . 30

5 Závěr **31**

A Konfigurace `dnscache` **33**

B Nastavení snižující počet zápisů na médium **34**
B.1 Konfigurační soubor `/etc/fstab` . . . 34
B.2 Skript `/etc/init.d/copytoram` . . . 34
B.3 Skript `/etc/init.d/writefromram` . . . 35

C `p3scan` - kontrola emailů **36**

D Popis skriptu `gen.sh` **38**
D.1 Funkce `gen_iptables_rules` . . . 38
D.2 Funkce `gen_tc_rules` . . . 38
D.3 Funkce `gen_dhcp` . . . 39

Zadání

PC jako hraniční směrovač poskytovatele služeb Internetu

PC as a Border Router of an Internet Service Provider

Vedoucí: Rychlý Marek, Mgr., UIFS FIT VUT
Oponent: Očenášek Pavel, Ing., UIFS FIT VUT
Přihlášen: Nowak Michal
Zadání:

1. Seznamte se se strukturou počítačové sítě vybraného ISP a se SW a HW požadavky na hraniční směrovač připojující síť ISP do Internetu.
2. Zvolte vhodnou distribuci GNU/Linux a SW splňující požadované funkce. Zvažte omezení daná HW směrovače a sítí ISP a diskutujte zvolené prostředky.
3. Nastudujte problematiku a navrhněte řešení pro přidělování a směrování privátních i veřejných IPv4 adres a pro přidělování IPv6 adres a připojení do IPv6 Internetu pomocí IPv4.
4. Nastudujte problematiku a navrhněte řešení pro autentizaci klientů, účtování provozu a řízení šířky pásma.
5. Seznamte se s problematikou využití více WAN rozhraní pro rozdělování zátěže a prevenci výpadku spojení. Navrhněte řešení.
6. Realizujte směrovač na PC. Implementujte řešení z bodu 3 a alespoň částečně (např. pro neměnné parametry) řešení z bodu 4.
7. Ověřte řešení v testovacím provozu. Diskutujte výsledky a navrhněte možná vylepšení.

Kategorie: Počítačové sítě
Operační systém: GNU/Linux
Literatura:

- Hunt, Craig. Konfigurace a správa sítí TCP/IP. Vyd. 1. Computer Press, Praha. 1997.
- Bert Hubert. Linux Advanced Routing & Traffic Control HOWTO. [http://lartc.org/howto/]
- Leonardo Balliache. Network Traffic Control Network Modeling. [http://www.opalsoft.net/qos/]

Komentář: Práce vyžaduje kladný vztah k operačnímu systému GNU/Linux a základní znalost počítačových sítí.

Konzultantem práce bude zástupce středně velkého ISP se sídlem v Rousínově (preferuje se elektronická komunikace). Po domluvě bude studentovi zapůjčen HW pro implementaci, včetně vysokorychlostního připojení v místě působnosti ISP (pokud bude potřeba). Možnost odměny v případě kvalitně odvedené práce s praktickým uplatněním.

Licence

Licenční smlouva je uložena v archivu Fakulty informačních technologií Vysokého učení technického v Brně.

Kapitola 1

Úvod

Smyslem této bakalářské práce je vytvořit směrovač pro středně velkého poskytovatele připojení k internetu (ISP) občanské sdružení Rousínovsko.net. Směrovač současný nedosahuje výkonu, který si narůstající počet uživatelů žádá a neobsahuje služby, které by zajistili uživatelům větší bezpečí při spojení s internetem. Návrh směrovače byl konzultován se zástupcem ISP a je vytvářen na základě jeho představ o vylepšení této části síťové infrastruktury s důrazem na dostupnost, minimální potřebu správy a co nejširší uplatnění svobodného a otevřeného softwaru.

V první části práce se zaobírám především výběrem vhodné distribuce operačního systému a hardware, načrtávám strukturu sítě. Diskutuji zvolenou distribuci a porovnávám její vhodnost pro použití na směrovači vzhledem k jiným obecným distribucím a specializovaným distribucím. Jednou ze stěžejních částí celé práce je sekce "Realizace snahy o minimální množství zápisů na médium", ve které se snažím nalézt teoretická i praktická východiska pro eliminaci zápisů na datové médium, které jsem pro směrovač zvolit, a to USB flash disk tvořený z logických obvodů typu NAND.

Ve druhé části popisuji především zvolené služby. Popisuji vybrané implementace a porovnávám je s jejich možnými náhradami. Některá zvolená východiska můžou být kontroverzní a věřím, že mnozí administrátoři produkčních systémů by se mnou nesouhlasili, konkrétně mluvím o sekci "DNS cache", kde rozebírám, mimo jiné, konflikt mezi filosofií djbdns a BIND jakožto nejpoužívanějších DNS a DNS cache serverů.

Třetí část je věnována především autentizaci klientů, řízení šířky pásma a účtování provozu. Autentizace probíhá na úrovni MAC adres a IP adres, pokud se snaží do internetu přistoupit někdo, jehož dvojice MAC:IP adresa není registrována, přístup je mu odepřen. Účtování provozu je realizováno pomocí MRTG a statistiky uživatelů jsou dostupné na web serveru, který je spuštěn na směrovači. Šířka pásma je všem klientům poskytována stejná.

V poslední kapitole se snažím zhodnotit svůj osobní přínos pro realizaci směrovače, zhodnotit další vývoj jak směrovače tak i celé sítě a navrhnout zlepšení a shrnout, co mi práce na bakalářské práci dala.

Kapitola 2

Současný stav sítě, volba hardware a distribuce operačního systému

2.1 Současný stav sítě

Rousínovsko.net je středně velký poskytovatel internetového připojení v Rousínově a přilehlých obcích; expanze postupuje tak, jak dovolují finanční prostředky a jiné proměnné vlivy. V současné době jsou zahrnuty tyto obce [36]

- Rousínov,
- Habrovany,
- Velešovice,
- Vítkovice,
- Koválovice a
- Komořany.

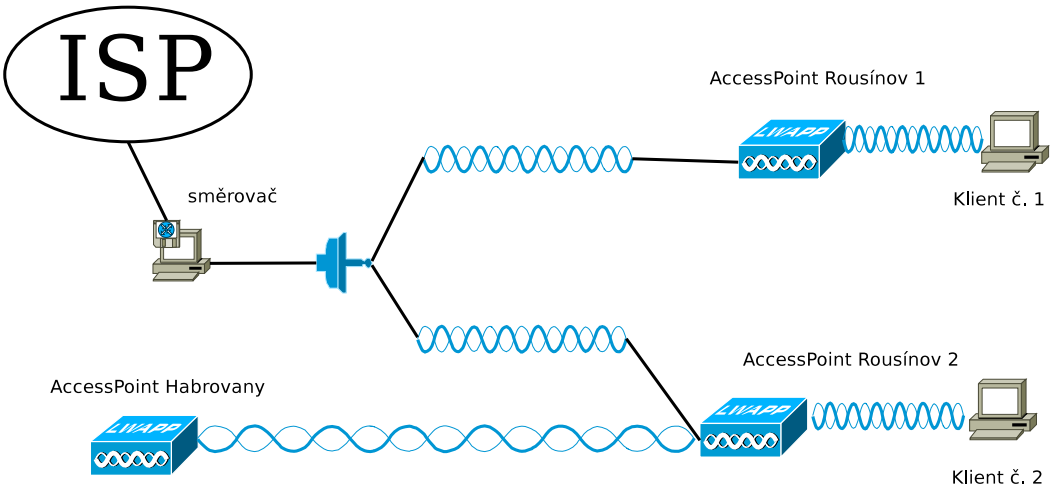
Pokryta jsou především nejhustěji obydlená místa a důležité veřejné prostory, jako například náměstí.

Síť je založena na technologii IEEE 802.11g [35], známé též jako Wi-Fi, neboli bezdrátové připojení. V síti jsou rozmístěny přístupové body a zákazník se připojuje k nejbližšímu z nich, na který má přímou viditelnost. Vzdálenost mezi klientem a přístupovým bodem je mezi 500-800 metrů. Samotné spojení je realizováno anténou spojenou koaxiálním kabelem s PCI kartou v počítači.

Jako přístupové body se používají především: Ovislink WL5460, USRobotics USR5450, Ovislink 1120 a místy i jiné typy. Použité přepínače: Edimax, Ovislink a TP-link. K zabezpečení se používá většinou WEP, čtyři přístupové body podporují WPA, ale jenom softwarové, hardwarové neumí.

Celkový počet uživatelů je 300, ale toto číslo se mění s připojováním dalších obcí do sítě a reprezentuje spíše počet připojených domácností.

Připojení do vnějšího internetu je realizováno spojem na 10 GHz při šířce pásma 10/10 MBit/s. Čas odezvy ICMP paketu (round trip time) přes program **ping** z uživatelské stanice na směrovač poskytovatele připojení je asi 20 ms z toho tři čtvrtiny tvoří latence uvnitř sítě.



Obrázek 2.1: Schéma sítě Rousínovsko.net

2.2 Hardware

Současným směrovačem je PC s procesorem AMD Sempron 2600+ (1,4 GHz, 64-bitová varianta), operační paměť 256 MB DDR RAM, síťové karty firmy RealTek.

Protože není úplně jednoduché implementovat směrovač na směrovači již běžícím, dostal jsem starší osobní počítač, který dříve sloužil jako herní stanice. Jeho současná konfigurace je následující

- AMD Athlon XP 3200+ (2,2 GHz, 32-bitová varianta),
- operační paměť 768 MB RAM,
- síťové karty RealTek a nvidia.

Místo karet RealTek RTL8139 jsem měl možnost zvolit karty 3COM, ale vybral jsem je, protože ovladače pro RealTek jsou v jádře velmi dlouhou dobu, firma sama k nim dodává technickou dokumentaci, případně platí vývoj otevřených ovladačů a v neposlední řadě tyto karty/čipy vyrábí v milionových nákladech, a proto věřím v jejich odladěnost [45], přestože na jejich adresu jsou i negativní kritiky, které říkají, že i když čipová sada 8139 podporuje bus-master DMA, má tak špatně udělané rozhraní, že všechen výkonnostní zisk plynoucí z DMA je tímto ztracen [48].

2.2.1 Výběr datového média

Jako datové médium jsem použil 4 GB USB 2.0 flash disk. Počítač sice disponoval pevným diskem, ale ten jsem zavrhl, protože obsahuje mechanicky se pohybující části a ty nemají tak dlouhou životnost jako například flash disky, compact flash karty a jiná nerotující nevolatilní média.

Za nevýhodou flash disků považuji to, že můžou být snadno zničeny mechanickým zásahem, například neopatrnou manipulací vyrvány z USB portu, případně zničeny elektrickým proudem, který by mohl vzniknout u neznačkového počítače. Relativní nevýhodou je také maximální počet zápisů do jednoho bloku - tato hodnota se u běžně prodávaných modelů pohybuje ve stovkách tisíců průpisů [47]. Většině služeb běžících na pozadí by trvalo

desítky let, než by blok zničila (v této chvíli předpokládám, že je použit souborový systém, který nepoužívá rozprostřený zápis a zápis probíhá do stejných bloků), avšak některým, jako například **syslogu**, by to mohlo trvat pár dní.

2.3 Volba souborového systému

Pravděpodobně nejvhodnějším souborovým systémem na flash médium je JFFS2 [20], které nepoužívá, tak jako starší, souborové systémy na flash disky, překladovou vrstvu na flash disku k emulaci funkce pevného disku, ale umisťuje souborový systém přímo na čip. Na rozdíl od JFFS, které podporovalo pouze NOR flash paměti, JFFS2 podporuje i NAND flash disky, které mají pouze sekvenční V/V rozhraní a nemohou být mapovány do paměti [4]. Souborový systém podporuje kompresi třemi algoritmy: zlib, rubin a rtime a obsahuje Garbage Collector, který nahrazuje kruhový záznam v JFFS.

I když je JFFS2 dobrým kandidátem na využití ve směrovači má některé nevýhody, které rozhodly v jeho neprospěch. Nevýhodou rodiny JFFS souborových systému je nutnost kontroly všech uzlů v době připojení souborového systému, což je by byl u 4GB flash disku velký časový problém [21]. Jako vážný problém při nasazení JFFS2 se ukázala neexistence uživatelské dokumentace, která by popisovala filosofii vytváření a provozu souborového systému v praxi.

Druhou volbou byl systém ReiserFS, který mám na desktopu, ale ten se ukázal nepoužitelný. Z neznámého důvodu na něm zápis trval neúměrně dlouhou dobu. Nakonec byl využit léty prověřený souborový systém **ext3** s vypnutým **noatime** [19], které zaručí, že se nebudou aktualizovat přístupové časy k uzlům typu **i** (inode), což zajistí rychlejší přístup k médiu a také jeho delší životnost.

2.4 Realizace snahy o minimální množství zápisů na médium

Protože jsem se rozhodl použít "klasický" souborový systém a ne takový, který pro zápis používá celý prostor média, musím zajistit, aby se na flash disk zapisovalo co nejméně.

Díky dobré koncepci adresářové struktury v Debianu, jsou soubory, do kterých se zapisuje bez přičinění uživatele, v adresáři `/var`. Kořenový adresář je uložen na svazku, který je připojen pouze pro čtení; adresáři `/rw/var` je připojen jako **tmpfs**, tedy je alokován do paměti (jeho velikost je omezena pouze velikostí virtuální paměti), do tohoto adresáře je při startu operačního systému skriptem `/etc/init.d/copytoram` nakopírován obsah `/var`, ten je tak v paměti a poté je příkazem `mount --bind /rw/var /var` připojen přes "fyzicky" `/var`. Všechny zápisy poté probíhají do paměti. Při ukončování běhu operačního systému se spustí skript `/etc/init.d/writefromram` a ten zajistí zpětné překopírování změněných souborů zpátky na disk a odpojí nepoužívané svazky v paměti.

Tyto jednoduché skripty jsou součástí dodatku B.

Tato operace se ukazuje někdy jako časově náročná, trvá až 15 minut, ale nepředpokládá se, že by restart systému byl příliš častý, proto můžeme tuto nedokonalost považovat za nepříliš závažnou. Druhou vlastností toho systému načítání celých adresářových struktur do operační paměti je to, že při výpadku elektrického napájení budou tato data ztracena, a protože běh systému bez restartu, a tedy i bez uložení změněných dat, lze počítat na měsíce, můžeme po opětovném nastartování počítače dostat částečně neaktuální systém. Pravděpodobně tím nejdůležitějším, co by se mohlo ztratit, jsou emaily nedoručené uživatelům,

nastavení DNS cache a záznamy, které jsou sice v adresáři `/var/log`, ale nedostaly se tam skrz syslog daemona.

2.5 Diskuze: specializovaná vs. obecná distribuce

Na stávajícím směrovači je "disketová" distribuce **floppyFw** verze 2.99.4 obsahující jádro 2.4.29, zkompilované gcc-3.4.3, iptables a HTB v3 k řízení šířky pásma.

2.5.1 Výhody z pohledu administrátora

Za výhodou toho řešení považuji snadnost jeho použití pro danou situaci, tedy vytvoření samostatně stojícího směrovače internetového provozu, minimální nároky na administrátora a snížení nároků na systémové zdroje, protože distribuce obsahuje pouze software nutný k běhu směrovače a nic navíc.

Další zástupci vývojového směru specializovaných distribucí [46] jsou

- **na operačním systému Linux** – Shurdix, ClarkConnect, Coyote Linux, Voyage Linux
- **a na operačním systému FreeBSD** – m0n0wall (FreeBSD 4.11)

2.5.2 Nevýhody z pohled administrátora

Výkon

Podstatnou nevýhodou pro rozrůstající se síť je samotná koncepce této mini distribuce. Minimalismus se objevuje na všech vrstvách. Jádro verze 2.4 je menší než obdobné jádro verze 2.6 [23], systémové nástroje jsou kompilovány oproti oproti uClibc, což jde v duchu distribuce, poskytovat nástroje s minimálními paměťovými nároky, ale za cenu ztráty výkonu [38]. Naproti tomu to, že sestavování probíhalo kompilátorem gcc-3.4.3 bylo už v minulosti považováno za výhodu. Koncem roku 2004 považovali vývojáři Linuxového jádra gcc verze 2.95 lepší co do rychlosti kompilace i produkovaného kódu než gcc verze 3.2 a předverze řady 4.0 [1].

Lze ale namítnout, že modernější specializované distribuce obsahují často stejný software jako distribuce obecné a navíc nabízejí onu zmíněnou minimalističnost. S tímto se vyrovnávají následující dva argumenty o rozšiřitelnosti a bezpečnosti.

Rozšiřitelnost

Takovouto samostatně stojící (standalone) distribuci, je mnohem těžší rozšířit o programové vybavení, které není její integrální součástí, než distribuci obecnou jako například Gentoo, Debian nebo CentOS.

Přidání nového programu, který tvůrci distribuce nepovažovali za potřebný, může být složité, či přímo nemožné v případě distribucí na médiu pouze pro čtení. Podobné problémy nastávají i u uchovávání konfigurace, která se musí ukládat na jiném médiu.

Bezpečnost

Za velice odvážné, avšak z dlouhodobé perspektivy nerozumné považuji zvolení distribuce, o kterou se stará jednotlivec, případně malá skupina vývojářů, ve svém volném čase a

zadarmo. Pokud se nalezne bezpečnostní chyba v jádře nebo službě, je správce systému často odkázán na svého distributora, až uveřejní opravenou verzi. U velkých komunitních distribucí je reakční doba v řádu dnů; pokud administrátor systému nesleduje obdobu bezpečnostní hlášení CERT ve své distribuci (DSA – Debian Security Advisories [5], GLSA – Gentoo Linux Security Advisories [15], ...), dozví se o chybě i její opravě ve chvíli, kdy se pokusí aktualizovat systém. A podle mého názoru není důvod takovéto aktualizace nedělat každý den. Jak by ale administrátor měl vyřešit ten samý problém při použití mini distribuce, když její správce je dočasně nedostupný, případně přestal jevit o své dílo zájem?

Odpověď je nasnadě: bude muset přejít na jinou distribuci, nebo ponechat systém svému osudu a doufat, že je pro útočníka nezajímavý. Je zřejmé, že tento přístup je nezodpovědný a odůvodnitelný snad jen v domácím prostředí.

Naštěstí u některých výše zmíněných specializovaných distribucí lze získat placenou podporu.

2.6 Volba distribuce operačního systému

Z argumentů výše uvedených vyplývá, že inklinuji k použití obecné distribuce a k její následné specializaci.

2.6.1 Fáze první: Gentoo Linux

Protože na desktopu mám nainstalovaný Gentoo Linux, což je meta distribuce založená na kompilování aplikací podle daných předpisů [13], a jsem s jeho správou i výkonem spokojen, byl pro mě intuitivní volbou. Od poskytovatele připojení byl do budoucna přislíbený počítač, který v té chvíli fungoval jako směrovač. Protože jsem počítač neměl fyzicky, moje pracovní stanice je 32-bitová a slíbený počítač 64-bitový, a neměl jsem zkušenosti s instalací a provozem Gentoo na 64-bitové platformně, zkoušel jsem pokusné instalace vytvářet v emulačním softwaru **Qemu** s předpoklady, že

- instalace ve virtuálním stroji bude posléze přenesena na stroj fyzický,
- programy nutné pro správu systému, před dodáním PC, budu *cross-kompilovat* [12]na stanici a následně přenášet do virtuálního stroje,
- v reálném provozu by nemuselo být možné instalovat - tedy kompilovat - programy, proto by k tomu byl použít jiný počítač.

Dva z výše uvedených předpokladů se však nepodařilo naplnit. Ne všechny programy bylo možné cross-kompilovat (například **vim** [40]) a do virtuálního stroje se musely přenést jejich 32-bitové varianty, čímž se ztrácel možný výkonnostní zisk vzhledem k 32-bitové platformě. Samotná kompilace ve virtuálním stroji je velice pomalá. Dalším nenaplněným předpokladem byla existence druhého počítače, který by poskytoval výpočetní výkon nutný při instalaci softwaru. Počítač sice existuje, ale je na něm nainstalován operační systém neunixového typu a jeho použití by vše dále ztěžovalo.

Z komplikací, které se při pokusu o umístění Gentoo Linuxu objevily , se zdá použití této distribuce nevhodné pro použití na směrovači, pro který není dostupná *kompilovací farma*, která by odstínila explicitní nároky distribuce založené na kompilování ze zdrojových kódů.

Dalším důvodem pro nenasazení Gentoo Linuxu v prostředí, kde pravděpodobně nebude vždy přítomen administrátor je to, že v této distribuci není dostupná vývojové větev, která

by se dala označit za stabilní. Stabilní v tom smyslu, že pokud se objeví chyba v nainstalovaném programu, tak se ta chyba opraví ve verzi programu, která je nainstalována. V Gentoo to často funguje tak, že pokud chce administrátor odstranit bezpečnostní chybu, tak musí nainstalovat nejnovější verzi programu, což může přinést komplikace, co se týče změny konfiguračních souborů, nutnosti aktualizování i jiných aplikací a podobně. Ale tento důvod jsem si uvědomil až dávno poté, co byl nainstalován Debina GNU/Linux.

2.6.2 Fáze druhá: Debian GNU/Linux

Debian Linux je distribuce složená především ze svobodného a otevřeného softwaru [6] dostupná pro široké spektrum platforem. Vyznačuje se stabilitou systému, minimálními změnami v průběhu životního cyklu a širokou komunitou. Debian se vyskytuje v několika verzích, v době psaní této práce jsou to tyto [41]

- old stable – předchozí stabilní – Debian 3.1 Sarge,
- stable – současná stabilní – Debian 4.0 Etch,
- testing – testovací/budoucí stabilní – Debian Lenny,
- unstable – nestabilní – Debian Sid (nestabilní verze se *vždy* jmenuje Sid).

Verze Debianu jsou pojmenovávány podle postaviček z filmu Toy Story (česky: Příběh hraček) [42]. *Sid* (nestabilní verze) je pojmenován podle emocionálně labilního chlapce, který pravidelně ničil hračky.

Už v době vydání (2005) byl Debianu Sarge považován za zastaralý, co se týče softwarové výbavy, a často se nedal nainstalovat na tehdejší hardware. Dá se říct, že to Debianu dost uškodilo, co se mediálního obrazu týče, a především ti, kteří přešli k Ubuntu, nad ním lámali hůl. Přesto byla tato verze stabilní, a to, podle mého názoru, je důvod, proč si administrátoři Debian vybírají. Debian s kódovým označením Etch, který byl plánován na prosinec 2006, a vyšel se zpožděním 8. dubna 2007, tato stigmata překonává a v něm obsažený software je mnohdy aktuálnější než ten v Gentoo Linuxu, a přesto je stabilní a jde bez problému nainstalovat i na moderní hardware. Debian je dle mého názoru nejvhodnější distribucí k implementaci směrovače na operačním systému Linux.

2.6.3 Instalace operačního systému

O instalaci Debianu koluje zvěst, že se dá provést pomocí pěti úderů do klávesy **Enter**. V mém případě byla o něco složitější.

K instalaci jsem se snažil ze začátku použít tzv. **netinstall** ISO obraz, vedla mě k tomu snaha o co nejaktuálnější systém hned po instalaci, ale protože můj poskytovatel připojení k internetu nepřiděluje IP adresu neznámé MAC adrese a softwarová změna této adresy se mi v příkazovém shellu, který je součástí instalátoru Debianu, **busybox**u, z neznámého důvodu nepodařila, použil jsem k instalaci první CD ze sady instalačních disků. Tento ISO obraz obsahuje vše, co je třeba k základní instalaci systému, kterou jsem zvolil. Po instalaci je systém tvořen jen tím, co je nutné pro elementární funkčnost, zbytek bude nainstalován odděleně.

Po instalaci stále ještě testovací verze je nutné systém co nejdříve aktualizovat; protože jsem pořád neměl připojení k internetu, "změnil" jsem pomocí programu **ip** z balíku programů **iproute** (známého také jako **iproute2**) MAC adresu síťové karty,

```
ip link set eth0 address 00:0a:e4:53:92:61
```

požádal DHCP server o přidělení IP adresy,

```
/etc/init.d/networking restart
```

a pomocí

```
aptitude update && aptitude -VD dist-upgrade
```

povýšil aplikace na opravené verze. Některé programy byly nahrazeny verzemi novějšími, protože když jsem systém instaloval, Debian Etch byl ještě testovací verzí, takové změny jsou v ní přípustné.

Kapitola 3

Výběr a nastavení služeb

3.1 PAT - Port Address Translation

Poskytovatel připojení k internetu Rousínovsko.net nemá přiděleno dostatek veřejných IPv4 adres, proto je k připojení tří stovek klientských počítačů nutné spustit překlad adres a portů (PAT) - oficiálně označovaný jako **symetrický NAT** a v Linuxové komunitě jako "maškaráda" (`MASQUERADE`). Překlad bude prováděn technikou *Full cone NAT* - v textu dále nazývanou jako PAT - definovanou v RFC 3489 [33]. Tento dokument definuje čtyři typy NAT a to: "Full cone NAT", "restricted cone NAT", "port restricted cone NAT" a "symmetric NAT".

Toto rozdělení je ale diskutabilní. Bylo definováno v době, kdy existovalo několik implementaci NAT/PAT, které se v různých ohledech lišily a tato standardizace nezpůsobila, že by se k sobě přiblížily. Druhým důvodem je, že toto vymezení vzniklo, až při přípravě protokolu STUN, který měl umožnit zjištění existence a typu NATu a firewallů mezi aplikací a internetem. Dá se předpokládat, že dodavatelé síťového infrastruktury, kteří nepodporují STUN, ani neparticipují na IETF toto rozdělení NATů nedodržují.

PAT úzce souvisí s NAT. NAT - Network Address Translation - definuje statické nebo dynamické překládání vnitřních adres v síti na adresy vnější, které mají přístup dále do jiné části sítě nebo do internetu. Pro příklad na vnitřním rozhraní je síť 192.168.1.0/24 a na vnějším 10.10.10.0/8 ; pokud paket směřuje z vnitřní sítě ven, změní se zdrojová adresa v hlavičce paketu z 192.168.1.X na 10.10.10.Y . Pokud cílové zařízení zachytí tento paket, odpoví na adresu 10.10.10.Y, paket přijde na směrovač, který provedl překlad, a ten podle NAT tabulky distribuuje paket dál do vnitřní sítě. Nikdo kromě směrovače, neví, kdo je ve skutečnosti odesilatelem dat – je to možné brát jako jistou formu ztížení práce útočníkovi, ale v žádném případě ne jako bezpečnostní (*security*) záležitost, či ochranu před útokem [3].

NAT v našem případě lze použít pouze v případě, že máme dostatek veřejných IP adres, na která je třeba ty neveřejné překládat, což nemáme.

PAT funguje stejně jako NAT a navíc překládá porty. Počítač z lokální sítě odesílá paket, se svou zdrojovou adresou a portem, který přijde na směrovač, ten přiřadí nově vzniklému spojení číslo portu z množiny dostupných portů a vloží tento port do pole *zdrojová adresa* hlavičky paketu. Směrovač poté odešle paket na výstupní rozhraní.

Směrovač si vytvoří záznam v překladové tabulce, který obsahuje

- vnitřní IP adresu,
- vnitřní zdrojový port a

- vnější port.

Následující pakety ve spojení jsou už přiřazovány k tomuto záznamu.

Vzdálený počítač vloží zdrojovou adresu a port do polí v hlavičce pro cílovou adresu a cílový port. Směrovač v lokální síti už zajistí překlad hlavičky na údaje dále identifikující cestu ke klientovi.

PAT: výhody

1. Dovoluje připojování zařízení k internetu, i když celosvětově dochází k nedostatku volných veřejných IPv4 adres [43].
2. Klienti ve vnitřní síti jsou jistým způsobem "chráněni", protože jejich IP adresa je skryta. Existuje menší nebezpečí přímých útoků.

PAT: nevýhody

1. Se zařízením za překladem portů není možné vytvořit přímé spojení, které je nutné například pro FTP, SIP, H.323. Musí se použít tzv. *connection tracking* moduly v jádře [44].
2. Protokol UDP lze protunelovat za PAT pomocí protokolu STUN (aplikace **chownat**), ale to není možné u nejčastěji používaného symetrického NAT/PAT.
3. PAT porušuje pravidlo, na kterém je postaven internet: Každé zařízení má možnost se spojit s jakýmkoliv jiným zařízením.
4. PAT, a NAT obecně, nefunguje s IPSec, protože NAT modifikuje hlavičku, což nejde dohromady s kontrolami prováděnými IPSecem. Jiné tunelovací protokoly se potýkají s podobným problémem.
5. PAT zpomaluje nástup IPv6, protože z pohledu uživatele internetu (tedy především webu) jej připojení skrz PAT neomezuje, a na druhou stranu internet nové generace nenabízí nic, co by v IPv4 internetu nebylo.

Implementace v Netfilteru

```
iptables -A FORWARD -i ${LAN_IFACE} -s 192.168.0.0/255.255.0.0 -j ACCEPT
iptables -A FORWARD -i ${WAN_IFACE} -d 192.168.0.0/255.255.0.0 -j ACCEPT
iptables -t nat -A POSTROUTING -o ${WAN_IFACE} -j MASQUERADE
```

Směrování klientů, kteří nejsou součástí systému PAT, mají veřejnou IP adresu, probíhá nastavením pravidla ve firewallu, kdy se povolí jejích přesměrování a zároveň se přidají informace o klientech do směrovací tabulky.

3.2 Výběr vhodných služeb

Z následujících kritérií lze dovodit, jaké programy lze najít v základní instalaci Debianu a obecně v jeho repozitářích [7]. Programy

- mají dobrou historii, co se týče bezpečnostních chyb,

- jsou široce používané,
- jsou podporované a dále vyvíjené svými autory (tzv. *upstream*) a
- svobodné dle kritérií Debianu.

Proto například **sendmail**, který měl velice špatnou bezpečnostní historii [2], není součástí výchozí instalace, ale je nahrazen programem **Exim**, který je ale méně používaný.

Podobná kritéria, spolu se snadnou a jasnou konfigurovatelností, jsem si dal i já. Snažil jsem se používat balíčky obsažené v samotné distribuci, ale výjimečně jsem vybral i ty, které jsem si musel zkompilovat sám.

3.2.1 SSH server/klient

SSH server je služba umožňující vzdálené připojení a následné spuštění zabezpečeného (šifrovaného) shellu skrz nezabezpečenou síť (například internet nebo LAN). Tato služba nahrazuje nezabezpečenou službu **rshd**. K výběru jsou tři řešení implementující protokol SSH v2

- lsh,
- SSH Tectia (známý též jako "SSH.com"),
- OpenSSH.

lsh je GNU GPL implementující SSH protokol v2, poslední verze - 2.0.3 - vyšla 9. května 2006. Je těžké zjistit, zda ho někdo používá na tak exponovaném místě, jako bezesporu směrovač internetového provozu je, v popisku u Debianího balíčku s touto serverovou i klientskou aplikací je zdůrazněno, že může mít problémy s bezpečností.

Aplikace server/klient **SSH Tectia**, produkovaná společností SSH Communications Security, není svobodným softwarem a ani jsem o ní neuvažoval - uvádím ji pouze pro úplnost.

Jednoznačně nejpoužívanější implementací v1 a v2 protokolu SSH je **OpenSSH** od komunity kolem operačního systému OpenBSD. Tato služba je dostupná pro širokou paletu operačních systému od Linuxu, přes varianty BSD až po ty komerční. V základní konfiguraci není třeba nic měnit, pouze pokud bychom chtěli podporovat X protokol a například dovolovat spouštění X aplikací na serveru s jejich následným zobrazením na straně klienta, povolíme v konfiguračním souboru `/etc/ssh/sshd_config` následující volbu

```
X11Forwarding yes
```

Klientskými aplikacemi jsou v případě OpenSSH programy **ssh**, **scp** a **sftp** a nahrazují **telnet**, **rlogin**, **rcp** a jiné. Použití **ssh** k připojení k vzdálenému serveru je běžnější než alternativou **putty**, známou především na nesvobodných operačních systémech.

Změnou voleb

```
ForwardX11 yes
ForwardX11Trusted yes
```

v konfiguračním souboru `/etc/ssh/ssh_config` docílíme výše zmíněného spouštění X aplikací na klientské ploše.

3.2.2 Plánovač úloh Cron

Cron je daemon, který spouští naplánované úlohy, které získává pomocí programu **crontab**, v daném čase, například: každou minutu, hodinu, den, týden, měsíc nebo v určitý čas. Dosahuje toho tak, že se spustí každou minutu, provede naplánované úlohy a ukončí se [14]. Na Debianu jsou k úkolování cronu používány následující soubory a adresáře

- uživatelské úlohy (soubory) – /var/spool/cron/crontabs/{user1, root}
- systémové úlohy (adresáře)
  - periodické – /etc/cron.{daily, weekly, monthly}/
  - definované/tématické – /etc/cron.d/

Dillon's Cron

Nejjednodušší z cronů; snaží se být bezpečnou implementací cronu, bez málo používaných funkcí. Například nepodporuje definici proměnných v těle souboru **crontab**. Všechny úlohy jsou spuštěny ze shellu `/bin/sh`.

Fcron

Nejpokročilejší ze všech cronů, obsahuje nejvíce funkcí. Je vhodný pro systémy, které neběží soustavně, protože serializuje nesplněné úlohy a v nejbližším možném okamžiku je spustí. Podporuje nastavování proměnných, každý uživatel může mít svůj vlastní crontab. Přístup ke cronu je řízen skrz soubory `/etc/cron.allow` a `/etc/cron.deny`. **Fcron** se nadále vyvíjí.

Anacron

Anacron je vhodný pro systémy, které neběží kontinuálně. Závisí na jednom z cronů, který by měl **Anacron** jednou denně spustit, a ten pak provede "přeskočené" úlohy (když byl systém vypnutý). De facto se nejedná o cron.

Vixie cron

Vixie cron je implementací SysV cronu. Každý uživatel má svůj crontab, definice proměnných je povolena. Na rozdíl od ostatních cronů podporuje *SELinux* a *PAM*. Je to "střední cesta" mezi **Dcron**em a **Fcron**em. Vybral jsem si ho, protože je na Debianu v základní instalaci, mám s ním zkušenosti z desktopu a většina aplikací, které instalují své vlastní úlohy do crontabu, využívá jeho vlastností.

Práce s cronem

Plánování uživatelských úloh provedeme příkazem `crontab -e` v shellu daného uživatele. Pokud by chtěl administrátor implantovat uživateli *newman* úlohu, zadá v administrátorském shellu: `contab -e -u newman`, čímž se mu otevře editor definovaný proměnnou EDITOR, nebo VISUAL, pokud takové proměnné definované nejsou, použije se program `/usr/bin/editor` [17].

Následující syntaxi lze použít jak v souboru crontab, tak i souborech v adresáři `/etc/cron.d/`, kde jsou "tématické crontaby". Například v souboru `mrtg` budou pravidla pro spuštění tvorby grafu přes program MRTG. Následují příklady syntaxe:

```
#Minuty   Hodiny  Dny    Měsíce   Den v týdnu Příkaz
0         3       1      1        *           /bin/false
30        16      *      1,2      *           /bin/true
*         *       *      1-12/2   *           /bin/who am i
```

3.2.3 Syslog, zaznamenávání chodu systému

Syslog je program, který přeposílá zprávy v IP síti. K tomu obsadí port 514/UDP, čte zprávy, které mu přicházejí, třídí je podle pravidel v konfiguračním souboru a ukládá do adresáře `/var/log/`a také odesílá na logovací server pro případ, že by směrovač byl obsazen útočníkem, který by určitě jako jednu z prvních věcí v našem systému vypnul logovacího daemona.

Jako **syslog** je často označován jednak server sbírající zprávy a pak také vlastní protokol, který je k tomuto posílán [18].

Nejčastěji používaní syslog daemoni

- **metalog** – široce konfigurovatelný logovací daemon, sám provádí rotaci logů (podobně jako samostatný program **logrotate**, který používají ostatní daemoni). Jeho nevýhodou je, že neumí posílat záznamy aktivity na jiné stroje, ani je neumí přijímat.
- **syslog-ng** – asi nejčastěji používaný logovací daemon, neprovádí rotaci záznamů. Je dobře konfigurovatelný a umí odesílat záznamy na jiné stroje, ale tato vlastnost se mi nepovedla nakonfigurovat, a proto jsem použil program následující.
- **sysklogd** – nejstarší, přesto stále vyvíjený, z dnes používaných logovacích daemonů, neprovádí rotaci. Umožňuje odesílání záznamů na jiné stroje. Tento syslog daemon jsem si vybral.

3.2.4 DNS cache

Na směrovači jsem považoval za nutné spustit DNS cache, která by sbírala dvojice rekurzivní DNS požadavek (doménové jméno) — IP adresa. Od klientů v místní síti jako například od webového prohlížeče nebo od emailového klienta získá dotaz na doménové jméno a od vzdáleného DNS serveru získá IP adresu jakožto odpověď.

Samotná cache urychlí příští překlad doménového jména na IP adresu tím, že se neprovede dotaz na vzdálený DNS server, ale použijí se data z cache.

Pro samotnou DNS cache není nutné použít tak rozsáhlý program jako je **bind**, stačí aplikace, která dělá pouze onu cache nebo DNS software, který je modulární. Jako nejvhodnější řešení považuji **dnscache** z balíku **djbdns** Daniela Bernsteina. **Djbdns** vznikl z frustrace způsobené bezpečnostními problémy a nerespektováním standardů ze strany balíku **bind**, především verze 4 a 8. **Djbdns** je naproti tomu, stejně jako všechny programy profesora D. J. Bernsteina, vytvářen s důrazem na bezpečnost (běh v chroot u, oddělení cache a DNS serveru, běh pod nerootovským uživatelem, ...) a tradiční UNIXovou modularitu (každý úkon, který je nutné v rámci DNS systému udělat, řeší samostatný program, který je snadno nahraditelný programem jiným) [10]. Nevýhodou jeho programů je jejich licence, tedy spíše to, že licenci nemají (používá se pro ně označení *license-free software*) [8].

Paměťovou náročnost lze ovlivnit proměnnou `CACHESIZE`, asi 5 % z cache zabírá tabulka s rozptýlenými položkami. Pokud dojde v systému paměť, zahodí se nejstarší záznam a odpověď na aktuální požadavek se zaznamená. Výchozí nastavení po instalaci je `CACHESIZE = 1 MB` [9].

Příklad experimentálního nastavení na velikost 10 MB,

```
# velikost cache
echo 10000000 > /service/dnscache/env/CACHESIZE
# maximální datový limit
echo 10485760 > /service/dnscache/env/DATALIMIT
# restart služby 'dnscache'
svc -t /service/dnscache
```

které zbytečně neomezuje délku platnosti záznamů.
Postup konfigurace `dnscache` viz dodatek A.

3.2.5 DHCP server

Dynamic Host Configuration Protocol je protokol používaný síťovými zařízeními k získání IP adresy a obvykle i jiných údajů pro plnohodnotnou práci v síti. Protokol je typu server/klient. Server je obvykle jeden na segmentu sítě a rozděluje IP adresy ze zadaného rozsahu.

Komunikace serveru a kliente vypadá takto

1. **DHCP DISCOVERY** — **klient** pošle na broadcastu UDP paket na adresu 255.255.255.255; tím požádá všechny dostupné DHCP servery o poskytnutí adresy
2. **DHCP OFFER** — **server** odpoví na unicastu přímo klientovi a nabídne mu IP adresu ze svého rozsahu
3. **DHCP REQUEST** — **klient** přijme adresu a odpoví serveru na broadcastu, aby ostatní DHCP servery věděly, že jejich nabídka nebyla klientem přijata
4. **DHCP ACKNOWLEDGEMENT** — **server** na unicastu klientovi odpoví a dodá mu další požadované informace (výchozí brána, síťová maska, DNS servery, ...)

Implementací DHCP serveru je v Debianu více, některé z nich krátce představím.

udhcp

Je DHCP server/klient vhodný především pro vestavěné systémy, a pokud je kompilován proti uClibc, serverový i klientský binární soubor může mít dohromady jenom 36 kB.

Tento balík není vhodný pro nasazení v síti poskytovatele připojení, protože mu chybí některé vlastnosti a především už není několik let vyvíjený [39].

dnsmasq

Jednoduchý DHCP a DNS server vhodný výhradně pro domácí použití.

ISC DHCP server/klient

ISC DHCP server/klient jsem si zvolil, protože, je aktivně vyvíjený a má široké možnosti konfigurace.

Příklad konfigurace DHCP serveru - přidělování IP adres

```
# definice podsítě
subnet 192.168.0.0 netmask 255.255.0.0 {

        pool {
                # rozsah přidělovaných adres
                range 192.168.1.0 192.168.255.254;
                # nepřidělí adresu neznámému klientovi
                deny unknown-clients;
                # definice klientského zařízení
                host user1 {
                        # adresa HW rozhraní (MAC adresa)
                        hardware ethernet 00:11:22:33:44:55;
                        # přidělí vždy fixní IP adresu
                        fixed-address 192.168.1.1;
                } # host
        } # pool
} # subnet
```

Alternativa: Zeroconf

Alternativou k systému DHCP je Zeroconf, což je souhrnný název pro sadu technik, která automaticky vytvoří použitelnou síť na technologii IP bez toho, aby byla nutná konfigurace zařízení či specializované servery. Zeroconf může usnadnit netechnickým uživatelům nastavení menší sítě, která může obsahovat klientské počítače, tiskárny a jiná zařízení od kterých se dá čekat automatické nakonfigurování. Zeroconf může zastoupit funkci DHCP a DNS serverů [49] [29].

Zeroconf automaticky řeší tyto problémy: výběr síťových adres pro zařízení, jména konkrétních zařízení, zjišťování dostupných služeb v síti.

Hlavní implementace: Apple Bonjour, Avahi a implementace Zeroconf ve Windows CE 5.0 .

Rozšíření Zeroconfu je pomalý a nikdy jsem se s ním nesetkal, jedním z důvodů může být autokonfigurace IP adresy v protokolech IPv4 a IPv6 tzv. *link-local*. Je pravděpodobné, že pro tak velkou síť jako je Rousínovsko.net by se ani nehodil.

3.2.6 Transparentní SMTP/POP3/IMAP proxy

Na směrovači jsem nasadil transparentní SMTP/POP3/IMAP proxy **p3scan**. Idea je taková, že porty 25 (SMTP), 110 (POP3) a 143 (IMAP) jsou přesměrovány pomocí **iptables** na port 8110, na kterém poslouchá **p3scan** a kontroluje obsah emailů na viry a spam.

Pokud je při přijímání emailu jeden z těchto nežádoucích elementů nalezen, je uložen do "karantény" a adresát je uvědoměn v tom smyslu, že zpráva pro něj byla závadná, a pokud má pochybnosti o správnosti jejího přesunu do karantény, a tedy nedoručení, může kontaktovat administrátora, sdělit mu kód, pod jakým byla uložena, a ten zprávu zkontroluje ručně.

Je-li posílán spam nebo zavirovaný email z vnitřní sítě, je ze strany **p3scan**u zrušeno spojení se vzdáleným SMTP serverem a zpráva nedoručena.

p3scan umožňuje definovat program nebo skript, kterému bude při "průchodu" emailu směrovačem předložen soubor s emailem. Vytvořil jsem skript, který otestuje zprávu antivirovým programem na přítomnost virů a antispamovým softwarem na přítomnost spamu,

a pokud oba dva testy skončí negativním výsledkem, je email zhodnocen jako přípustný a pouze v tomto případě je odeslán či přijat. Skript je součástí dodatku C.

S programem **p3scan** byla malá potíž. V repozitáři Debianu byla pouze stará verze, která nepodporovala protokol SMTP, proto jsem si stáhl vývojovou verzi a tu zkompiloval. Přestože je to verze vývojová, nenarazil jsem při jejím používání na žádné problémy.

3.2.7 Antivirový software

Jako antivirový software byl zvolen svobodný Clam AntiVirus, který je periodicky aktualizován službou **clamav-freshclam**. ClamAV jsem nastavil jako server/klient architekturu, její použití je doporučováno jako výkonnější než samostatně stojící řešení. Služba **clamd** poslouchá na portu 3310 a čeká na příchozí spojení. Program, který takovéto spojení obvykle iniciuje, je **clamdscan**.

Ten předá serveru soubor a čeká na rozhodnutí, zda je email zavirován či ne.

```
daily.inc is up to date (version: 3225, sigs: 10920, f-level: 15, builder: sven)
-------------------------------------
Received signal: wake up
ClamAV update process started at Thu May 10 14:34:58 2007
main.inc is up to date (version: 43, sigs: 104500, f-level: 14, builder: sven)
daily.inc is up to date (version: 3225, sigs: 10920, f-level: 15, builder: sven)
-------------------------------------
Received signal: wake up
ClamAV update process started at Thu May 10 15:34:58 2007
main.inc is up to date (version: 43, sigs: 104500, f-level: 14, builder: sven)
daily.inc is up to date (version: 3225, sigs: 10920, f-level: 15, builder: sven)
```

Obrázek 3.1: "Obarvený" záznam činnosti služby aktualizující virové definice

3.2.8 Antispamový software

Za nejvhodnější řešení na zjišťování spamu jsem zvolil **SpamAssassin**. Stejně jako u antivirového řešení jsem dal přednost architektuře server/klient. Server **spamd** naslouchá na portu 783, klientskou aplikací je **spamc**, který je na rozdíl od programu **spamassassin** [Perl] naprogramovaný v jazyce C.

Aby **SpamAssassin** s co největší pravděpodobností rozeznal spam (nevyžádaná pošta) a ham (vyžádaná pošta), je třeba ho doslova "naučit", co je spam a co ham. Z webového archivu jsem si stáhl několik tisíc emailů, které jsou považovány za spam, a nechal jsem je aplikací **sa-learn** naučit. To samé jsem udělal asi se stovkami emailů, které mám ve své osobní schránce, a označil je za ham.

3.2.9 Podpora IPv6 v operačním systému a přidělování IPv6 adres

Aby se klientská stanice mohla připojit do IPv6 internetu, musí pro tento internet nové generace obsahovat podporu. Podpora v rámci operačního systému se dělí na dvě části

- **jádro operačního systému** – podporu pro IPv6 poskytuje jaderný modul **ipv6.ko**. Nejlepší podpora IPv6 je v jádře řady 2.6, v řadě 2.4 už nejsou přidávány nové funkce, které by pokrývaly nejnovější RFC dokumenty.
- **uživatelské aplikace** – Sendmail, Exim, Qmail, BIND, VLC/VLS, Quake, SSH, Apache, Mozilla Firefox, lynx, elinks, Squid, wget, mplayer a další.

Ve všech moderních distribucích je dnes IPv6 podpora implicitně zapnuta jak v jádře, tak v uživatelských aplikacích. Záleží na dodatečné konfiguraci jednotlivých služeb a programů, zda budou připojení skrz internet nové generace preferovat a protokol IP verze čtvrté použijí pouze jako záložní možnost.

Pokud náš poskytovatel připojení k internetu neposkytuje IPv6 konektivitu, musíme si ji zařídit svépomocí skrz tak zvaného "Tunnel Brokera". Tunnel Broker je organizace, která získala komerční adresní rozsah a poskytuje ho svým zákazníkům. Obvyklý rozsah, který je poskytnut zákazníkovi, je /64 (takzvaný prefix a ten nám dovolí připojit $2^{(128-64)}$ zařízení, což je 4 294 967 296 dnešních internetů.

Přidělování IPv6 adres

Jsou dostupné dvě metody automatického přidělování IPv6 adres, bezstavové ([30] a [31]) a stavové ([32]).

1. Bezstavové přidělování IPv6 adres, směrovačem, klientům je řešeno skrz Router Advertisement Daemona **radvd**, který je definován v RFC 2461. Tento daemon očekává *router solicitations* (žádost směrovače) a odpovídá pomocí *router advertisement* (hlášení směrovače), které, v závislosti na nastavení v souboru `/etc/radvd.conf` , obsahuje prefix sítě, MTU linky a další. Těchto několik málo informací stačí klientovi, aby svou IPv6 adresu tzv. *autokonfiguroval*.

   Z MAC adresy klienta `00:11:22:33:44:55` vytvoříme adresu ve formátu **EUI-64** tak, že doprostřed vložíme sekvenci `FFFE` a znegujeme sedmý bit zleva. Výsledkem budiž
   `02:11:22:FF:FE:33:44:55`. Spojením prefixu získaného od Tunnel Brokera **Hurricane Electric**, `2001:470:1F00:3740::/64` a EUI-64 adresy, získáme veřejnou IPv6 adresu `2001:470:1f00:3740:211:22ff:fe33:4455/64` , se kterou můžeme přistupovat do IPv6 internetu.

2. Stavové přidělování adres je řešeno přes DHCPv6. Ačkoliv protokol IPv6 byl navržen tak, aby použití DHCP serveru nebylo nutné, přesto bylo později zahrnuto do tří dokumentů IETF: RFC číslo 3315, 3633 a 3646. Skrz DHCPv6 mohou být šířeny informace, které by se nedaly zjistit z jiných zdrojů – například DNS server. Používají se dvě implementace DHCPv6

   - **wide-dhcpv6** – Implementace navazující na projekty KAME a WIDE, poslední verze vyšla 16. října 2006. Projekt se zdá být více či méně mrtvý.
   - **dibbler** – Aktivně vyvíjený projekt, který byl před vydáním verze 0.6 otestován, zda spolupracuje se sedmi různými, otevřenými i uzavřenými, platformami (získávání a poskytování adres zařízením, relay služby, . . .). Je implementován na GNU/Linuxu a předpokládá se jeho portace na FreeBSD.

Praktické spojení s Tunnel Brokerem

Spojení s Tunnel Brokerem a přístup na internet získáme takto

1. načteme jaderný modul **ipv6.ko**

   ```
   modprobe ipv6
   ```

2. vytvoříme pojmenovaný tunel *sixbone* mezi veřejnou IPv4 adresou směrovače (147.229.194.163) a IPv4 adresou Tunnel Brokera (64.71.128.82)

   ```
   ip tunnel add sixbone mode sit remote 64.71.128.82 local 147.229.194.163
   ttl 255
   ```

3. aktivujeme tunel

   ```
   ip link set sixbone up
   ```

4. asociujeme IPv6 adresu získanou od Hurricane Electric s rozhraním **sixbone**

   ```
   ip addr add 2001:470:1F00:FFFF::1C71/127 dev sixbone
   ```

5. všechen IPv6 provoz směrujeme skrz rozhraní **sixbone**

   ```
   ip route add ::/0 dev sixbone
   ```

6. všechen provoz z adresního rozsahu `2001:470:1F00:3740::/64` pošleme na rozhraní **eth1**, které směřuje do vnitřní sítě

   ```
   ip route add 2001:470:1F00:3740::/64 dev eth1
   ```

7. příkazem `ip -f inet6 addr` si můžeme zkontrolovat nastavení rozhraní

   ```
   1: lo: <LOOPBACK,UP,10000> mtu 16436
     inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
   3: eth0: <BROADCAST,MULTICAST,UP,10000> mtu 1500 qlen 1000
     inet6 fe80::20a:e4ff:fe53:9261/64 scope link
       valid_lft forever preferred_lft forever
   4: eth1: <BROADCAST,MULTICAST,UP,10000> mtu 1500 qlen 1000
     inet6 fe80::219:e0ff:fe0b:6930/64 scope link
       valid_lft forever preferred_lft forever
   7: sixbone@NONE: <POINTOPOINT,NOARP,UP,10000> mtu 1480
     inet6 2001:470:1f00:ffff::1c71/127 scope global
       valid_lft forever preferred_lft forever
     inet6 fe80::c0a8:1/64 scope link
       valid_lft forever preferred_lft forever
     inet6 fe80::93e5:c2a3/64 scope link
       valid_lft forever preferred_lft forever
   ```

3.2.10 Web server pro nenadálé situace

Pro případ, že by směrovač ztratil spojení s vnějším internetem, je nanejvýš vhodné, aby poskytovatel připojení své zákazníky o tomto stavu uvědomil. Dá se předpokládat, že uživatelé v době výpadku používají především službu WWW, a pokud ne, třeba si do té doby vyměňovali zprávy přes XMPP/Jabber, pokusí se v případě problémů na tuto službu připojit. Proto stačí přesměrovat všechen odchozí provoz směřující na port 80 na webový server poskytovatele připojení, kde bude zpráva o nastalé chybě a předpokládané době do vyřešení.

Takovýmto serverem může být i tento směrovač.

Za WWW/HTTP server jsem zvolil **Lighttpd**, protože je snadno konfigurovatelný a zabírá minimum systémových prostředků. Protože tento server hostí také účtování provozu

jednotlivých uživatelů, je dostupný pouze s vnitřní sítě. Kvůli vyšší bezpečnosti je spuštěn v chrootu a pod nerootovským účtem.

Tento příkaz zajistí výše zmiňované přesměrování provozu na portu 80

```
iptable -t nat -A PREROUTING -p tcp -i eth1 --destination ! 192.168.0.0/16
  --dport 80 -j DNAT --to 192.168.0.1:8080
```

Web server může být použit potenciálně i pro jiné účely, třeba když uživatel překročí datový limit při stahování, může mu být na webu zobrazeno, že je odpojen od vnějšího internetu. Případně by provozovatel sítě mohl při prvním přístupu webu od přiděleni IP adresy přesměrovat zákazníka na stránku s aktuálními informacemi o síti nebo s reklamami. Předpokládám, že tuto "vlastnost" by uživatelé nepřijali s nadšením, proto jsem se jí nezabýval jinak, než v úvahou v tomto odstavci.

Kapitola 4

Autentizace klientů, účtování provozu a řízení šířky pásma

4.1 Autentizace klientů

Ověřování klientů je prováděna na dvou úrovních

1. pomocí jaderného frameworku **netfilter**, běžně asociovaného s programem **iptables** a
2. pomocí nastavení DHCP serveru.

Pomocí **iptables** můžeme "spárovat" IP adresu s MAC adresou. Pokud si někdo (staticky) nastaví IP adresu na jinou, než je registrována na směrovači, a bude chtít přistoupit do internetu, bude jím poslaný paket zahozen bez upozornění.

Abych toto nemusel dělat ručně vytvořil jsem si k tomu shellový skript `gen.sh`. Ukázky viz dodatek D, který pomocí přednastavených proměnných zajistí vygenerování pravidel pro firewall, řízení šířky pásma a DHCP server, vzhledem k počtu uživatelů.

```
GRP + HOST  - určují IP adresu (jsou závislé na počtu klientů v síti)
MAC         - MAC adresa klienta

iptables -A FORWARD -s 192.168.${GRP}.${HOST}
  -m mac --mac-source ! ${MAC} -j DROP
```

Logika příkazu je takováto: "Pokud se paket snaží jít skrz směrovač do internetu a jeho IP adresa je **192.168.1.1** a MAC adresa není **00:11:22:33:44:55**, zahoď paket bez upozornění."

Druhou "linií obrany" je přidělování IP adres DHCP serverem jenom těm klientům, kteří jsou definováni v konfiguračním souboru `/etc/dhcp/dhcpd.conf`

```
range 192.168.1.0 192.168.255.254;
deny unknown-clients;
host user1 {
        hardware ethernet 00:11:22:33:44:55;
        fixed-address 192.168.1.1;
}
```

Ostatním klientům nebude IP adresa propůjčena díky volbě `deny unknown-clients`.

4.2 Skript `gen.sh`

Nastavením proměnných `GARANCE1`, `MAXIMUM`, `BURST`, `BAND_DOWN`, `BAND_UP`, `WAN_IFACE` a `LAN_IFACE` dodáme skriptu "představu" o naší síti především co se týče rozhraní směrem do internetu a do vnitřní sítě a také o tom, jakou propustnost budou mít jednotliví uživatelé k dispozici.

Počet uživatelů obsahuje proměnná `USERS`, která je naplněna příkazem `wc -l mac.list | awk ' print $1 '`. V souboru `mac.list` jsou uloženy MAC adresy uživatelů sítě, jedna na řádek.

Skript obsahuje tři funkce. Popis v dodatku D není úplný, pro ucelené informace viz skript samotný v adresáři `/root/ibp/scripts/` na odevzdaném datovém médiu.

4.3 Účtování provozu

Aby bylo možné zjistit, kolik dat uživatelé stahují z internetu, a tím případně porušují pravidla sítě, je na směrovači zavedeno účtování provozu jednotlivým uživatelům. Ke každé klientské IP adrese je je přiřazeno speciální pravidlo ve firewallu netfilter

```
iptables -I FORWARD --destination 192.168.1.1 -j ACCEPT
```

které zajistí, že každý paket, který přijde klientovi s IP adresou 192.168.1.1, bude započítán. Poté si můžeme příkazem `iptables -L FORWARD -v -n -Z` zjistit, kolik bajtů již ke klientovi přišlo. Tento údaj je potřebný pro nástroj MRTG, který si ho každých pět minut přečte a podle něj doplní údaje do grafu.

4.3.1 MRTG

MRTG je software pro monitorování zatížení síťových linek pomocí vygenerovaných grafů. MRTG je napsaný v Perlu a také díky tomu je multiplatformní, funguje na Linuxu, variantách BSD, Novel Netware a jiných; šířený je pod GNU GPL 2 [27]. Původně byl vytvořen k monitorování směrovačů, ale časem se vyvinul k obecnému monitorování systému skrz SNMP.

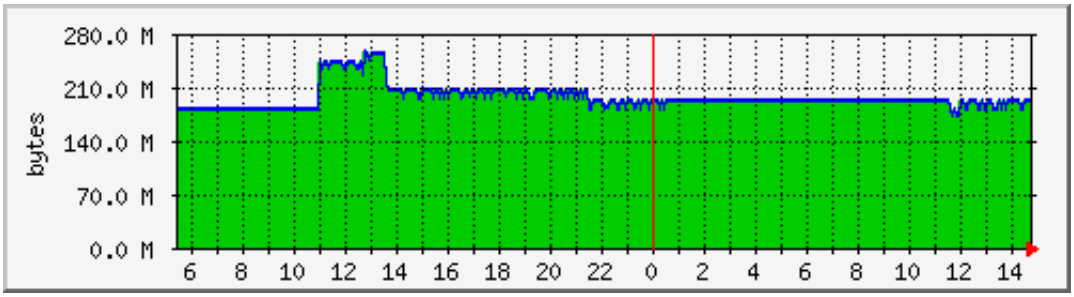
MRTG posílá zařízení pomocí SNMP požadavek s dvěma identifikátory objektů (OID). Zařízení, které musí podporovat SNMP, obsahuje *management information base* (MIB), ve kterém dohledá specifikované identifikátory (OID) a vrátí data. MRTG si poté zaznamená data do logu (záznamu) na klientovi a společně s už dříve sesbíranými údaji vytvoří HTML soubor a vygeneruje grafy vztahující se k danému zařízení.

Protože směrovač de facto poskytuje informace sám sobě, je základní podmínkou, aby na směrovači běžel SNMP daemon, který bude poskytovat informace a data klientovi - v našem případě MRTG.

4.3.2 Monitorování systémových prostředků

Pomocí MRTG se dá nejen sledovat množství přenesených dat, ale také jakékoliv jiné údaje dosažitelné přes SNMP v MIB. Následuje komentovaný příklad

- `WorkDir: /var/www/mrtg`
  Nastaví pracovní adresář, do něj bude generovat výsledky (databáze, grafy)



Obrázek 4.1: Využití operační paměti

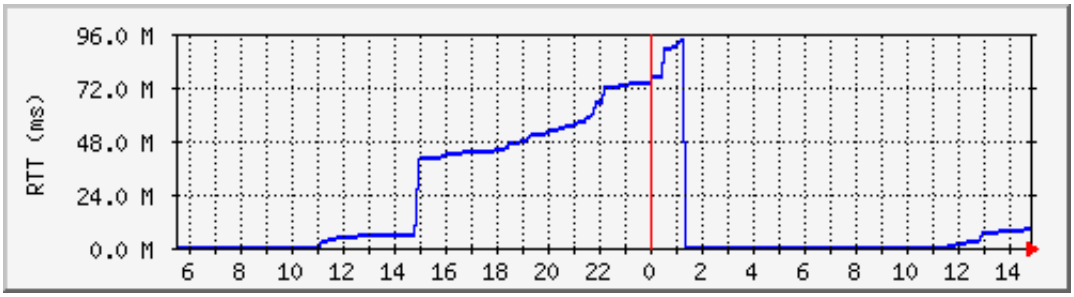
- `LoadMIBs: /usr/share/snmp/mibs/UCD-SNMP-MIB.txt`
  Načtení MIB databáze
- `Target[router01.cpu]:ssCpuRawUser.0&ssCpuRawUser.0:publicrouter01+ ssCpuRawSystem.0&ssCpuRawSystem.0:publicrouter01+ ssCpuRawNice.0&ssCpuRawNice.0:publicrouter01`
  Řetězec určující, která data budou z databáze vybírána.
- `RouterUptime[router01.cpu]: public@router01`
  Na směrovači *router01* zjistí, kolik času uplynulo od posledního spuštění.
- `MaxBytes[router01.cpu]: 100`
  Maximální hodnota na stupnici, vše, co je větší, je zahozeno jako chybná data.

Další položky v konfiguračním souboru jsou více méně nepovinné a zlepšují čitelnost a orientaci ve výsledné stránce i grafech.

4.3.3 Monitorování provozu uživatelů sítě

Vytvoření nového kontrolovaného uživatele

Skript `mkusertraffic.sh <IP adresa>` přidá uživatele se zadanou IP adresou do seznamu kontrolovaných uživatelů tak, že vytvoří soubor s konfigurací pro MRTG, ne nepodobný té výše uvedené.



Obrázek 4.2: "Využití" šířky pásma klientem

Získání údajů o přenesených datech

Pomocí skriptu `getusertraffic.sh <IP adresa>` vrátí počet přenesených dat směrem k uživateli s danou IP v Bytech.

Vygenerování grafů uživateli

Skript `runusertraffic.sh` zjistí IP adresy všech kontrolovaných uživatelů a spustí na jejich konfiguračních souborech MRTG. Po skončení tohoto skriptu jsou všechny grafy pro uživatele vygenerovány a řetězec FORWARD, ze kterého jsou brána data, vynulován.

Automatizace cronem

Pomocí cron daemonu můžeme periodicky naplánovat úlohy, která zajistí automatické aktualizování grafů.

Pravidlem

```
*/5 * * * * root if [ -x /usr/bin/mrtg ] && [ -r /etc/mrtg/cpu.cfg ]; then env
LANG=C /usr/bin/mrtg /etc/mrtg/cpu.cfg >> /var/log/mrtg/mrtg.log 2>&1; fi
```

automatizujeme sbírání dat o vytížení procesoru a pravidlem

```
*/5 * * * * root /etc/mrtg/runusertraffic.sh
```

zajistíme generování grafů pro všechny uživatele.

Publikace grafů

MRTG generuje HTML stránky a obrazové grafy do adresáře /var/www/ , což je také rootovský adresář pro server. Přístup k HTTP serveru je omezen na vnitřní síť z důvodu privátnosti dat.

4.4 Řízení šířky pásma

Žádná síť nemá neomezené prostředky, co se týče propustnosti, chce-li uživatelům nabídnout maximum ze své konektivity, musí klientům garantovat minimální propustnost, která bude vždy dostupná, a maximální propustnost nastavit blízko rychlosti spojení do internetu. Pokud je o šířku pásma větší zájem, než jsme schopni zajistit, přichází na řadu omezování uživatelů.

K omezování potřeb uživatelů nám pomůže *classfull qdisc* ([qdisc = **q**ueueing **disc**ipline] volně přeloženo "řadící disciplína obsahující třídy") a konkrétně jeho HTB v3 (Hierarchical Token Bucket) implementace v jádře novějším než 2.4.20 [16].

Především je důležité si uvědomit, že nemůžeme přímo určovat, kolik dat k nám přichází - můžeme to ale částečně ovlivnit. V obvyklém případě, pokud nám vzdálený server posílá data rychleji, než jsme schopni přijímat, data jsou po cestě k nám zahozena, nebo nedoručena, a pokud server není informován o doručení, sníží rychlost (konkrétní algoritmus změny posílání dat je závislý na implementaci TCP/IP zásobníku). Šířku pásma můžeme ovlivňovat pouze směrem ven z naší sítě a tuto kontrolu je nejlepší dělat a rozhraní směrem do naší sítě. Pokud bychom šířku pásma řídili na vnějším rozhraní, kde je například 256 MBit

ADSL router, tak by k němu mohlo přicházet více dat, než je schopen odesílat, a v tom případě by šířku pásma řídil on. Stav se pokusíme "uměle" navodit pomocí nástrojů k řízení šířky pásma [22].

Dále je nutné mít na paměti, že šířku pásma můžeme řídit pouze u protokolu TCP, ne tak už u UDP, které nemá žádný potvrzovací mechanismus, a nemůže proto "zrychlovat" ani "zpomalovat".

Pravidla jsou jádru předávána programem **tc** (Traffic Control) z balíku **iproute**.

Struktura pravidel je hierarchická, má jeden pojmenovaný kořenový uzel a prochází se rekurzivně. Při zadávaní nových pravidel řízení šířky pásma je dobré všechny předchozí řadící disciplíny vymazat a poté vložit novou – typu HTB. Na tento kořenový uzel, obvykle značený **1:0**, "pověsíme" další uzel, který bude mít šířku pásma stejnou nebo o něco menší, než je konektivita do internetu. Pojmenujeme ho **1:1** a pověsíme na něho další uzly, které už budou reprezentovat šířku pásma rezervovanou pro klienty.

```
tc class add dev eth1 parent 1:0 classid 1:1 htb rate 256kbit
tc class add dev eth1 parent 1:1 classid 1:11 htb rate 64kbit ceil 256kbit
```

Řádek první vytvoří zmiňovaný kořenový uzel. Řádek druhý připojí uzel **1:11** na uzel **1:1** a definuje pro něj, že

1. garantovaná propustnost bude 64 kbit/s a
2. maximální propustnost bude 256 kbit/s (aniž by byli omezeni ostatní uživatelé).

4.4.1 Volby ovlivňující šířku pásma

- **rate** – maximální propustnost, kterou má daná třída a její potomci garantovanou
- **ceil** – maximální propustnost, které může třída dosáhnout, pokud třída předchůdce něco uspořila. Pokud není tato volba explicitně zadána, použije se místo ní implicitně hodnota **rate**. Což znamená že, žádné "výpůjčky" u nadřazené třídy se provádět nebudou.
- **burst** – množství, o které může může být hodnota **ceil** překročena při nevyužití propustnosti z **rate**. U následníka by měla být alespoň tak velká, jako nejvyšší z těchto hodnot u potomků.

4.4.2 Značkování paketů

Aby bylo možné pakety propouštět, či zahazovat, je nutné si je poznačit:

```
iptables -t mangle -A POSTROUTING -d 192.168.1.1 -j MARK --set-mark 1
iptables -t mangle -A POSTROUTING -d 192.168.1.2 -j MARK --set-mark 2
iptables -t mangle -A POSTROUTING -d 192.168.1.3 -j MARK --set-mark 3
```

Pakety jsou označeny čísly (1,3), každý z nich směřuje na jinou cílovou stanici [34]. Následujícími filtrovacími pravidly jsou přiřazeny do správné "škatulky"

```
tc filter add dev eth1 parent 1:0 protocol ip handle 1 fw flowid 1:11
tc filter add dev eth1 parent 1:0 protocol ip handle 2 fw flowid 1:12
tc filter add dev eth1 parent 1:0 protocol ip handle 3 fw flowid 1:13
```

a můžou být řízeny, co se týče šířky pásma.

4.4.3 Další metody řízení šířky pásma

Použitá implementace řízení šířky pásma není přirozeně jedinou, která se dá použít. Možnou alternativou jsou:

- **CBQ** - Je stejně výkonné jako HTB, ale je považováno za obtížněji nastavitelné.
- **IMQ** - Dává pravděpodobně nejsilnější možnosti k řízení šířky pásma, ale nikdy nebyl, a nikdy nebude, zařazen do hlavní větve linuxového jádra, protože, podle některých vývojářů jádra, není korektně naprogramován, a to, co dělá, je teoreticky špatně [11]. Jeho použití na produkčním stroji je diskutabilní a distribucích, ve kterých k pro něho není přímá podpora, jako například v Debianu, velmi složité.
- **IFB** - Je následník IMQ, až donedávna k němu nebyla žádná dokumentace, mělo by nabízet širší možnosti než HTB, ale dosud není pravděpodobně tak stabilní [25].

4.4.4 Řízení šířky pásma na dle použitého aplikačního protokolu

Všechny doposud zmíněné způsoby řízení šířky pásma neanalyzovaly, jaká data přes směrovač proudí. Přitom pro provozovatele může být výhodné omezit uživatele, kteří stahují audiovizuální materiály přes BitTorrent nebo DC++, a dát ušetřenou šířku pásma k dispozici uživatelům webu. Jedno z možných řešení je **l7-filter**, který se skládá ze tří částí: jaderných modulů pro inspekci procházejících dat, z definic rozpoznávajících použitý protokol v datech a z programů v uživatelském prostoru, které spolupracují s netfilterem (iptables). Druhou možností filtrovat data na aplikační úrovni je **IPP2P**, ale ta neprochází aktivní vývojem a nemá takové možnosti jako sada **l7-filter**.

Filtry na aplikační úrovni jsem nepoužil protože poskytovatel připojení takovou službu nemá zájem využívat, filtry můžou být potenciálně náročné na výkon a od směrovače se předpokládá především rychlé a spolehlivé doručování a odesílání paketů. Pokud by ovšem někdy použity byly, pak na spíše na dedikovaném stroji, přes který by procházela všechen průchozí tok dat.

4.5 Využití více WAN spojení – prevence výpadku

Pokud vypadne linka spojující nás s internetem, naše síť ztratí spojení s vnějším světem a budeme muset počkat, až náš poskytovatel připojení chybu odstraní. Tomuto stavu se můžeme vyhnout použitím záložní linky vedené u jiného poskytovatele. Záložní linku můžeme buď používat společně s linkou hlavní, a tím zvýšit šířku pásma anebo ji používat jen při výpadku [24].

Pokud chceme použít jednu linku jako hlavní a druhou jako záložní, nabízí se dva způsoby řešení.

Ten první je nakonfigurovat fyzická zařízení pro spojení s internetem a pak nastavit jednotlivým cestám (route), asociovaným s fyzickým zařízením, vhodné metriky, a tím určit, která z nich bude použita přednostně při vybírání cesty protokolem RIP nebo OSPF, ale v této variantě by musela být podpora směrovacího protokolu na obou stranách spoje [26].

Druhou možností je tzv. bonding, kdy vytvoříme logické zařízení `bond0` a na něj připojíme *závislá* fyzická zařízení - v našem případě by to byly ethernetové síťové karty `ethX`. Výhodou tohoto spojení je rozmanitost využití tohoto spojení v závislosti na použitém módu.

Obecně:

1. můžeme získat buď vyšší rychlost (rychlosti zařízení se sčítají), nebo
2. získat systém *hlavní linka/záložní linka* - takto vzniklý systém by mohl být *vysoce dostupný* (High Availability), protože lze nastavit krátké intervaly testování dostupnosti linek a případně mezi nimi automaticky přepnout.

Bonding zařízení v Linuxu poskytuje modul `bonding.ko`.

4.5.1 Parametry modulu `bonding.ko`

- **mode** – Specifikuje metodu bondování. Výchozí je balance-rr (round robin).
  - **balance-rr nebo 0** – Přenos paketů je v sekvenčním pořadí od prvního závislého zařízení k poslednímu. Tento mód umožňuje rozdělování zátěže a prevenci proti výpadku spojení.
  - **active-backup nebo 1** – Jenom jedno závislé zařízení je aktivní, ostatní závislá zařízení jsou aktivována pouze při výpadku aktivního.
  - **balance-xor nebo 2** – Přenos je závislý na zvolené hashovací funkci. Jako výchozí je zvolena tato: $(zdroj + cil)\%n_{zavislych}$. Možno změnit pomocí nastavení parametru **xmit_hash_policy**.
  - **broadcast nebo 3** – Vysílá vše na všechna závislá zařízení. Mód poskytuje ochranu před výpadkem spojení.
  - **802.3ad nebo 4** – Definováno standardem IEEE 802.3ad. Vytváří agregační skupiny, které sdílejí stejnou rychlost a duplexitu linky - využívá všechna závislá zařízení ve skupině najednou. Hashovací funkce, která vybírá agregační skupinu, je závislá na obsahu volby **xmit_hash_policy** - ne všechny existující funkce jsou 802.3ad - kompatibilní.
  - **balance-tlb nebo 5** – Odchozí množství dat je distribuováno relativně k rychlosti daných zařízení. Příchozí data přicházejí na aktuální závislé zařízení, pokud to selže, jiné závislé zařízení si přivlastní jeho MAC adresu a nahradí ho.
  - **balance-alb nebo 6** – Podobné jako **balance-tlb**, ale obsahuje navíc vyvažování zátěže pro IPv4 provoz.
- **miimon** – Specifikuje frekvenci v milisekundách, která určuje, jak často je linka kontrolována na selhání. Nula znamená "neprováděj kontrolu", hodnota 100 je obecně považována za dobrou hodnotu pro většinu linek.
- **downdelay** – Udává čas, za jak dlouho, v milisekundách, po zjištění chyby bude linka vypnuta.

4.5.2 Praktické použití

```
modprobe bonding mode=balance-alb miimon=100
modprobe e100
ip addr add 192.168.44.1/24 dev bond0
ip link set dev bond0 up
ifenslave bond0 eth0
ifenslave bond0 eth1
```

Kapitola 5

Závěr

Prací na bakalářské práci jsem si měl možnost zjistit požadavky, které mají poskytovatelé připojení na směrovač, získal jsem náhled do struktury sítě a dovolila mi spojit můj vlastní zájem o operační systém Linux se znalostmi z oboru sítí získanými v povinných, ale především ve volitelných předmětech na Fakultě informačních technologií. Za zvláště zajímavé považuji to, že jsem jednak měl možnost vyzkoušet si a sestavit konkrétní směrovač, a především to, že směrovač bude v reálu použit a práce na něm odvedená nebude samoúčelná.

Důležitá je pro mě zpětná vazba, kterou jsem měl po dobu tvorby směrovače od zástupce ISP Rousínovsko.net a také předpokládaná zpětná vazba při nasazování do ostrého provozu od uživatelů sítě.

Z obecného hlediska lze za nejpřínosnější části odvedené práce považovat, odstranění pevného disku a tím snížení pravděpodobnosti odstávky systému, implementaci transparentní SMTP/POP3/IMAP proxy, která spolupracuje s antivirem a nastaveným antispamovým filtrem, jenž pravděpodobně napomůže poskytovateli připojení k odstranění své domény ze seznamu "šiřitelů spamu", na kterou se dostal kvůli často napadeným klientským počítačům, a tím i k plnému využití jeho MX záznamu. Za, především do budoucna, užitečnou část považuji implementaci protokolu IPv6, jako protokolu, kterým se lze připojit do IPv6 internetu. Další důležité části z hlediska ISP jsou bezpochyby řízení šířky pásma a prevence výpadku spojení použitím více WAN spojení.

V průběhu práce jsem se musel vypořádat s problémy především při instalaci základního systému, popsáno v kapitole 2.6.3, neexistencí aktuální verze transparentní proxy p3scan v Debianu, volbou vhodné distribuce Linuxu, kapitola 2.6.1 a 2.6.2 a místy i s problémy drobnějšího rázu. Protože jsem zvolil svobodnou distribuci Linuxu s komunitním vývojem, považoval jsem za samozřejmé, že služby, které bude systém zajišťovat, budou realizovány pomocí softwaru, který je buď

1. svobodný (free),
2. otevřený (open) nebo
3. bez licence, ale s dostupnými zdrojovými kódy (license-free).

Směrovač jsem měl možnost nasadit do testovacího provozu jenom jednou a to v době, kdy v měla síťová infrastruktura Rousínovsko.net problémy s kolizními rámci, kdy se klienti na směrovači objevovali pod stejnou, téměř učebnicovou, MAC adresou 00:11:22:33:44:55. V obě testu se podařilo odzkoušet základní konektivitu do internetu a vnitřní sítě, NAT a účtování provozu přes MRTG. Při dalších testování je nutné prověřit především řízení

šířky pásma, transparentní SMTP/POP3/IMAP proxy a blokování uživatelů, kteří nejsou členy sítě. Do budoucna chci odladit směrovač na stejnou úroveň spolehlivosti jako směrovač stávající a nastavit ho tak, aby splňoval požadavky poskytovatele připojení a tím naplnil jeden z cílů bakalářské práce. Dá se předpokládat, že do doby obhajoby by směrovač mohl být plně funkční.

Dále lze přemýšlet nad celkovým rozvojem sítě jako celku. Síť Rousínovsko.net má plochou strukturu bez směrovačů, všichni uživatelé jsou v jedné LAN síti. Síť je podle informací od poskytovatele připojení zaplavena ARP dotazy. Situaci by vyřešilo přidání směrovačů a přepínačů s podporou VLAN (IEEE 802.1Q), což by si ale pravděpodobně vyžádalo investice do administrace a nového hardware. Správce sítě by uvítal webové administrační nástroje ke správě sítě, ty řeší paralelní bakalářská práce *Správa hraničního směrovače poskytovatele služeb Internetu*.

Použité přepínače a přístupové body jsou ty z levnějších typů, které často obsahují nekvalitně naprogramovaný firmware a přehřívají se; vylepšení sítě by mělo zasáhnout i tuto oblast.

Celkově si dovolím konstatovat, že navržené řešení bude obecně vyhovovat poskytovatelům připojení středně velkého rozsahu o řádově stovkách klientů. U větších poskytovatelů by bylo radno vyměnit některé hardwarové, především síťové karty, a softwarové, p3scan není testován na větší zatížení než desítky paralelních spojení, komponenty.

Dodatek A

Konfigurace `dnscache`

1. Zjistíme, zda jsme připojeni k internetu
   `dnsq a www.aol.com 192.203.230.10`.
2. Vytvoříme uživatele **`dnscache`**, pod kterým bude běžet cache a **`dnslog`**, pod kterým bude běžet (samostatná) logovací služba.
3. Spustíme příkaz
   `dnscache-conf dnscache dnslog /var/lib/dnscache 192.168.0.1`
   poslední argument je adresa, ze které bude náše cache dostupná.
4. Službě **svscan**, která kontroluje běžící procesy v rámci programů **djbdns**, sdělíme, že přibyla nová služba
   `ln -s /var/lib/dnscache /service && sleep 5 && svstat /service/dnscache && svstat /service/dnscache/log`.
5. Vytvoříme soubor, který určí adresy autorizované k přístupu ke cachei
   `touch /var/lib/dnscache/root/ip/192.168`.
6. Do konfiguračního souboru DHCP serveru nebo do souboru `/etc/resolv.conf`, který musí být rezistentní vůči vlivu změny konfigurace z DHCP serveru
   `nameserver 192.168.0.1`.
7. Pomocí příkazu
   `dnsip www.fsf.org`
   zjistíme, jestli cache funguje.

Dodatek B

Nastavení snižující počet zápisů na médium

B.1 Konfigurační soubor /etc/fstab

```
# /etc/fstab: static file system information.
#
# <file system> <mount point>   <type>  <options>                  <dump> <pass>

# pseudo souborovy system, uzivatelsky dosazitelny
proc            /proc           proc    defaults                  0      0

# zavadeci oddil na externim flash mediu pripojenem na IDE kanal
/dev/hdc1       /boot           ext3    defaults,noatime,ro       1      2
# korenovy oddil na USB flash mediu
/dev/sda1       /               ext3    defaults,noatime,ro       0      1

# odkladaci prostor v RAM
tmpfs           /tmp            tmpfs   defaults,noatime,mode=1777 0      0
# drzi obsah adresare s casto menenymi soubory operacni pameti
tmpfs           /rw/var         tmpfs   defaults,noatime          0      0

# CD mechanika 1
/dev/hda        /media/cdrom0   udf,iso9660 user,noauto           0      0
# CD mechanika 2
/dev/hdb        /media/cdrom1   udf,iso9660 user,noauto           0      0
```

B.2 Skript /etc/init.d/copytoram

Inspiraci pro tyto skripty jsem nasel zde [37].

```
#!/bin/sh

        case ‘‘$1’’ in

  start)
```

```
    echo -n 'Kopiruju obsah adresaru urcenych pro zapis do RAM... '
    cp --preserve=all -P -r /var /rw
    echo -n '/var'
    echo 'hotovo'
    echo 'pripojuji tmpfs na /var'
    mount --bind /rw/var /var
  ;;

  stop)
  ;;

esac
```

B.3 Skript /etc/init.d/writefromram

```
#!/bin/sh

case ``$1'' in

  start)
  ;;

  stop)
    echo 'Ukoncuji programy drzici si soubory ve /var...'
    kill `lsof | grep /var | awk '{print $2}' | uniq`
    sleep 1
    kill -9 `lsof | grep /var | awk '{print $2}' | uniq`
    echo 'hotovo'

    sleep 2

    echo -n 'Odpojuji adresare typu tmpfs... '
    umount /var
    mount -o remount,rw /
    echo 'hotovo'

    echo -n 'Kopiruji aktualizovane soubory zpatky na flash disk...'
    cp -r -u --preserve=all /rw/var /
    mount -o remount,ro /

    sync

    echo 'hotovo'
    ;;
esac
```

Dodatek C

p3scan - kontrola emailů

```
OUTPUT=‘‘/tmp/p3scan.test’’

THISPGM=$0
FILENAME=$1
MAILFROM=$2
MAILTO=$3
USERNAME=$4
SUBJECT=$5
MAILDATE=$6
SERVERIP=$7
SERVERPORT=$8
CLIENTIP=$9
CLIENTPORT=${10}
PROTOCOL=${11}
# P3Scan
PROGNAME=${12}
# P3Scan version
VERSION=${13}
# Virus info?
VDINFO=${14}
# hlavicka antiviroveho programu
HEADER=${15}

echo 1 $THISPGM     > $OUTPUT
echo 2 $FILENAME   >> $OUTPUT
echo 3 $MAILFROM   >> $OUTPUT
echo 4 $MAILTO     >> $OUTPUT
echo 5 $USERNAME   >> $OUTPUT
echo 6 $SUBJECT    >> $OUTPUT
echo 7 $MAILDATE   >> $OUTPUT
echo 8 $SERVERIP   >> $OUTPUT
echo 9 $SERVERPORT >> $OUTPUT
echo 10 $CLIENTIP   >> $OUTPUT
echo 11 $CLIENTPORT >> $OUTPUT
echo 12 $PROTOCOL   >> $OUTPUT
```

```
echo 13 $PROGNAME   >> $OUTPUT
echo 14 $VERSION    >> $OUTPUT
echo 15 $VDINFO     >> $OUTPUT
echo 16 $HEADER     >> $OUTPUT

# volame antivirovy program
/usr/bin/clamdscan ${FILENAME}
CLAMSCAN=$?
echo 'ClamAV respoded: '${CLAMSCAN} >> $OUTPUT

if [[ ${CLAMSCAN} == '0' ]]; then
    /usr/bin/spamc -c < ${FILENAME} >> $OUTPUT
    SPAMSCAN=$?
    echo 'spamc respoded: '${SPAMSCAN} >> $OUTPUT
fi

if [[ ${CLAMSCAN} == '0' && ${SPAMSCAN} == '0' ]]; then
    P3SCAN=0;
else
    P3SCAN=1;
fi

echo 'P3SCAN respoded: '${P3SCAN} >> $OUTPUT

exit ${P3SCAN}
```

Dodatek D

Popis skriptu gen.sh

D.1 Funkce gen_iptables_rules

Obsahuje vytvoření NATu, ochranu proti spoofingu a zahazování paketů na portu používaném pro sdílení ve Windows.

```
iptables -F
iptables -t nat -F
iptables -I INPUT 1 -i ${WAN_IFACE} -j ACCEPT
iptables -I INPUT 1 -i ${LO} -j ACCEPT

# NAT/PAT
iptables -A FORWARD -i ${LAN_IFACE} -s 192.168.0.0/255.255.0.0 -j ACCEPT
iptables -A FORWARD -i ${WAN_IFACE} -d 192.168.0.0/255.255.0.0 -j ACCEPT
iptables -t nat -A POSTROUTING -o ${WAN_IFACE} -j MASQUERADE

# povoleni preposilni paketu skrz smerovac
echo 1 > /proc/sys/net/ipv4/ip_forward

# zapne ochranu proti spoofingu
for f in /proc/sys/net/ipv4/conf/*/rp_filter ; do
    echo 1 > $f
done

# zahazuje vschny pakety, ktere prochazeji skrz smerovac a smeruji
# do nebo z portu, ktery je pouzit pro sdileni souboru ve Windows
iptables -A FORWARD --dport 445 -j DROP
iptables -A FORWARD --sport 445 -j DROP
```

D.2 Funkce gen_tc_rules

Vygeneruje základní strukturu pro řízení šířky pásma.

```
# smaze qdisk
tc qdisc del dev $LAN_IFACE root >/dev/null
# vytvori qdisk typu HTB
```

```
tc qdisc add dev $LAN_IFACE root handle 1:0 htb default 1
# na qdisk ‘‘povesi’’ korenovy uzel urcujici maximalni propustnost linky
tc class add dev $LAN_IFACE parent 1:0 classid 1:1 htb
  rate ${BAND_DOWN}Kbit burst $BURST
```

D.3 Funkce gen_dhcp

Funkce pracuje v cyklu, kdy pro každý záznam v souboru `mac.list` vygeneruje pravidlo pro

- **DHCP server** do soubory `/etc/dhcp3/dhcpd.conf`, které zajistí, že klient s MAC adresou `$MAC` dostane vždy stejnou IP adresu:

```
host user${CNT} {
   hardware ethernet ${MAC}
    fixed-address 192.168.${GRP}.${HOST}
}
```

- **řízení šířky pásma**

```
# třída s garantovanou propustnosti ${RATE}
tc class add dev ${LAN_IFACE} parent 1:1 classid 1:1${CNT} htb
  rate ${RATE}Kbit ceil ${MAXIMUM} burst ${BURST}
# prida na konec vetve SFQ qdisc s prepocitavanim kazdych 5 minut
tc qdisc add dev ${LAN_IFACE} parent 1:1${CNT} handle 1${CNT}:0
  sfq perturb ${PERTURB}
# znackovaci pravidlo do firewallu
iptables -t mangle -A POSTROUTING -o ${LAN_IFACE}
  -d 192.168.${GRP}.${HOST} -j MARK --set-mark ${CNT}
# filtrovani paketu, pokud ma paket znacku ${CNT} (defacto
# poradove cislo), tak je poslan na konecnou vetev 1:1${CNT}
tc filter add dev ${LAN_IFACE} parent 1:0 protocol ip handle ${CNT}
  fw flowid 1:1${CNT}
```

  SFQ patří do rodiny řadících disciplín, které jsou založené na *fair-queue* algoritmu více informací viz [28].

- **svázání IP adresy s klientskou MAC adresou**

```
iptables -A FORWARD -s 192.168.${GRP}.${HOST}
  -m mac --mac-source ! ${MAC} -j DROP
```

Literatura

[1] WWW stránky. Abclinuxu: Jaderné noviny 285.
http://www.abclinuxu.cz/clanky/jaderne-noviny/jaderne-noviny-285.
ISSN 1214-1267.

[2] WWW stránky. Cert: Sendmail vulnerability notes.
http://search.cert.org/query.html?col=vulnotes&qt=sendmail&charset=iso-8859-1.

[3] WWW stránky. Cisco: How nat works.
http://www.cisco.com/warp/public/556/nat-cisco.shtml.

[4] WWW stránky. Commsdesign: An introduction to nand flash.
http://www.commsdesign.com/showArticle.jhtml?articleID=183700957.

[5] WWW stránky. Debian security advisories.
http://www.debian.org/security/#DSAS.

[6] WWW stránky. Debian: Společenská smlouva.
http://www.debian.org/social_contract.

[7] WWW stránky. Debian: What do you mean by free software?
http://www.debian.org/intro/free.

[8] WWW stránky. The djb way: djb and copyright.
http://www.thedjbway.org/license_free.html.

[9] WWW stránky. D. j. bernstein: djbdns - how to adjust the cache size.
http://cr.yp.to/djbdns/cachesize.html.

[10] WWW stránky. D. j. bernstein: djbdns - security.
http://cr.yp.to/djbdns/blurb/security.html.

[11] WWW stránky. Frequently asked questions about imq.
http://wiki.nix.hu/cgi-bin/twiki/view/IMQ/ImqFaq.

[12] WWW stránky. Gentoo cross development guide.
http://www.gentoo.org/proj/en/base/embedded/cross-development.xml.

[13] WWW stránky. Gentoo linux: About.
http://www.gentoo.org/main/en/about.xml.

[14] WWW stránky. Gentoo linux cron guide.
http://www.gentoo.org/doc/en/cron-guide.xml.

[15] WWW stránky. Gentoo linux security advisories.
http://www.gentoo.org/security/en/glsa/.

[16] WWW stránky. Htb linux queuing discipline manual - user guide.
http://luxik.cdi.cz/ devik/qos/htb/manual/userg.htm.

[17] WWW stránky. Indiana university: Unix systems support group - automating tasks with cron services. http://www.uwsg.iu.edu/usail/automation/cron.html.

[18] WWW stránky. Ingate firewall: Definitions of terms.
http://www.ingate.com/files/422/fwmanual-en/xa11944.html.

[19] WWW stránky. Internet faq archives: The atime and noatime attribute.
http://www.faqs.org/docs/securing/chap6sec73.html.

[20] WWW stránky. Jffs : The journalling flash file system.
http://sources.redhat.com/jffs2/jffs2-html/node3.html.

[21] WWW stránky. Jffs : The journalling flash file system#mounting.
http://sources.redhat.com/jffs2/jffs2-html/node3.html#SECTION00035000000000000000.

[22] WWW stránky. *Linux Advanced Routing & Traffic Control HOWTO*.

[23] WWW stránky. Linux-kernel archive: Kernel 2.6 size increase.
http://www.uwsg.iu.edu/hypermail/linux/kernel/0307.2/2279.html.

[24] WWW stránky. Linuxnet bonding.
http://linux-net.osdl.org/index.php/Bonding.

[25] WWW stránky. Linuxnet ifb. http://linux-net.osdl.org/index.php/IFB.

[26] WWW stránky. man 8 route.
http://www.die.net/doc/linux/man/man8/route.8.html.

[27] WWW stránky. Mrtg: About. http://www.mrtg.cz/doc/mrtg.en.html.

[28] WWW stránky. *Network Traffic Control Network Modeling*.

[29] WWW stránky. O'reilly: Understanding zeroconf and multicast dns.
http://www.oreillynet.com/pub/a/wireless/2002/12/20/zeroconf.html.

[30] WWW stránky. Rfc 2461: Ipv6 stateless address autoconfiguration.
http://www.ietf.org/rfc/rfc2461.txt.

[31] WWW stránky. Rfc 2462: Ipv6 stateless address autoconfiguration.
http://www.ietf.org/rfc/rfc2462.txt.

[32] WWW stránky. Rfc 3315: Dynamic host configuration protocol for ipv6 (dhcpv6).
http://tools.ietf.org/html/rfc3315.

[33] WWW stránky. Rfc 3489: Stun - simple traversal of user datagram protocol (udp) through network address translators (nats).
http://www.ietf.org/rfc/rfc3489.txt.

[34] WWW stránky. Root: Htb - jemný úvod.
http://www.root.cz/clanky/htb-jemny-uvod/.

[35] WWW stránky. Rousínovsko: Faq. http://www.rousinov.com/new/faq.php.

[36] WWW stránky. Rousínovsko: Mapa pokrytí.
http://www.rousinov.com/new/mapa.php.

[37] WWW stránky. Signal lost: Installing debian unstable on a wrap board with a ro cf disk. https://signal-lost.homeip.net/projects/wrap/.

[38] WWW stránky. uclibc: A c library for embedded linux.
http://www.uclibc.org/about.html.

[39] WWW stránky. udhcp server/client package. http://udhcp.busybox.net/.

[40] WWW stránky. Vim cross compile patch.
http://linuxfromscratch.org/pipermail/patches/2006-June/003055.html.

[41] WWW stránky. Wikipedia: Debian#debian releases.
http://en.wikipedia.org/wiki/Debian#Debian_releases.

[42] WWW stránky. Wikipedia: List of toy story characters#etch.
http://en.wikipedia.org/wiki/List_of_Toy_Story_characters#Etch.

[43] WWW stránky. Wikipedia: Nat#benefits.
http://en.wikipedia.org/wiki/Network_address_translation#Benefits.

[44] WWW stránky. Wikipedia: Nat#drawbacks.
http://en.wikipedia.org/wiki/Network_address_translation#Drawbacks.

[45] WWW stránky. Wikipedia: Realtek#criticism.
http://en.wikipedia.org/wiki/Realtek#Criticism.

[46] WWW stránky. Wikipedia: Softwarové směrovače (informativní seznam).
http://en.wikipedia.org/wiki/Router#.22Software.22_routers.

[47] WWW stránky. Wikipedia: Usb_flash_drive#strengths_and_weaknesses.
http://en.wikipedia.org/wiki/USB_flash_drive#Strengths_and_weaknesses.

[48] WWW stránky. Zdrojové kódy freebsd ovladače k čipům řady 8129/8139.
http://fxr.watson.org/fxr/source/pci/if_rl.c.

[49] WWW stránky. Zero configuration networking (zeroconf).
http://www.zeroconf.org/.