Jihočeská univerzita

Pedagogická fakulta Katedra fyziky

Využití speciálního operačního systému „MikroTik RouterOS“ v sítích lokálních ISP

bakalářská práce

Vedoucí diplomové práce

Ing. Michal Šerý

Autor

Beno Slávik

Anotace

Cílem mé práce je zcela funkční ukázka konfigurace zařízení, které řídí v síti lokálního poskytovatele Internetu veškerý datový provoz. V teoretické části práce se stručně seznámíte s možnostmi zařízení, které je ovládáno operačním systémem Mikrotik RouterOS. Další kapitolou se krátce zaměříme na vlastnosti a schopnosti dnes velmi rozšířeného operačního systému Linux. Praktická část ukazuje funkční mechanismy řízení datových toků, především se jedná o ukázky z nastavení výchozí brány sítě, priorizace jednotlivých služeb a inteligentní filtrace nežádoucích dat. Poslední kapitola pojednává o kontrolních mechanismech, které zlepšují komplexní správu sítě a mohou poskytnout vhodné informace pro správný a včasný zásah při poruše.

Celou práci jsem koncipoval jako možnou příručku pro mírně pokročilé provozovatele malých bezdrátových sítí. Pomocí návodů v ní uvedených je možné dosáhnout uspokojivých výsledků při ostrém provozu sítě poskytující přístup k Internetu.

Annonation

The objective of my diploma work is to show configuration of devices that are used by Internet providers. During theoretical part you acquaint with possibilities which provide devices with Mikrotik RouterOS operating system. The next chapter deals with characteristics and capabilities of Linux operating system. The applied part of diploma work shows mechanisms for taking control of data flows, especially configuration samples of default gateway, services priority changing and intelligent unwanted packet filtration. The last chapter deals with control mechanisms which are making the complex management of the network much easier. These mechanisms are important for correct and well timed failure action.

The whole diploma work is drawn like basic manual for moderately advanced network operator. With help of included instructions it is possible to achieve satisfactory success during real operation of network which is providing Internet.

Prohlašuji, že jsem bakalářskou práci na téma **„Využití speciálního operačního systému „MikroTik RouterOS“ v sítích lokálních ISP“** vypracoval samostatně a použil jsem jen pramenů, které cituji a uvádím v seznamu použité literatury.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské práce, a to v nezkrácené podobě, elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách.

V Českých Budějovicích dne 24. dubna 2007

…….
Beno Slávik

Obsah

1 Úvod

Internet – v dnešní době běžný standard ve světě informačních technologií. Tato globální síť, vytvořená spojením mnoha počítačů po celém světě, poskytuje mnoho služeb jako například WWW, e-mailová pošta, sdílení souborů, televize, rádio atd. O vzniku a vývoji Internetu již bylo napsáno mnoho, informace o současných možnostech jsou též obecně v povědomí většiny lidí. Tato práce si klade za cíl přiblížit běžnému člověku informace o tom, jak náročnou prací je správa a konfigurace tzv. „černých boxů“ – zařízení, které řídí veškerý provoz v sítích lokálních poskytovatelů Internetu.

V době pokročilých možností, kvalitních a inteligentních zařízení není ani tak problémem vytvořit síť pro šíření internetu mezi uživatele, ale spíše najít dostatečně kvalifikované pracovní síly pro správu a údržbu takové sítě. Široká škála zařízení v cenách od několika tisíc až do milionových částek poskytuje slušnou nabídku na způsob realizace takové sítě.

První kapitola o systému MikrotikOS podává stručný výtah z funkcí, kterými zařízení disponují. Pokud bych měl podat komplexní přehled o možnostech tohoto operačního systému, napsal bych další samostatnou práci. Stručně vyzdvižené základní a nejpoužívanější funkce však dokáží nastínit obraz širokých a téměř nevyčerpatelných možností zařízení s MikrotikOS softwarem.

Další částí práce jsem se zaměřil, jak vypadá konfigurace některých služeb v operačním systému Linux. Výčtem několika nejpoužívanějších služeb a ukázkami z jejich konfigurací jsem nastínil, jak může vypadat funkční server v ostrém provozu.

Poslední dvě kapitoly jsou ryze praktické. Nejdříve se věnuji samotnému nastavení routerů s MikrotikOS – konfigurace WIFI zařízení, filtrování protékajících dat, agregace klientů, řízení rychlostí, aj. Dále ukazuji možnosti grafického zobrazení stavu různých částí sítě, přehledné pomůcky pro sledování datových průtoků a vytížení jednotlivých linek.

2 Operační systém MikroTikOS

2.1 Co je systém MikroTikOS

MikroTikOS je jednou z mnoha distribucí systému Linux, která je však plně komerční. Tento speciální operační systém se zaměřuje především na možnost snadného vybudování inteligentních lokálních sítí. Jednoduchá údržba a nikterak obtížná konfigurace dává možnost vybudovat kvalitní sítě i méně pokročilým technikům. Mezi nejpoužívanější funkce se jistě řadí routing, firewalling, omezování šířky pásma, bezdrátové přístupové body a klienti, virtuální privátní sítě či skriptování. Základními funkcemi však potenciál zařízení není rozhodně vyčerpán – k dispozici je mnoho dalších a méně běžných možností. Výhodné je i využití některých přídavných programů jako například komplexní monitorovací systém **The Dude** nebo různé analyzátory a čítače síťového provozu.

2.2 Požadovaný hardware

MikroTikOS je možné provozovat jak na běžných osobních počítačích architektury „IA-32 kompatibilní“, tak na speciálních platformách vyrobených přímo pro podobné systémy (routerboardy, wrapboardy). Při použití v běžných PC je požadována minimálně čtvrtá generace procesorů (minimální takt 100 MHz), dále je podporována pátá generace procesorů (Intel Pentium, AMD K5 nebo Cyrix), případně jakákoliv novější jednoprocesorová stanice. Minimální velikost operační paměti je 32 MB, doporučuje se 64 MB a více. Jako zaváděcí médium slouží standardní IDE/ATA zařízení a případně SATA zařízení. S redukcí do ATA rozhraní je možné použít Flash karty a Microdrive disky – paměťové karty jsou výhodné především díky spolehlivosti a rychlosti.

Pokud router nebude nasazen na složité síťové operace, je možné použít speciální **routerboardy**. Jejich výhoda je především kompaktní velikost a jednoduchost. Nevýhodou je pouze malý výkon a omezený počet rozšiřujících zařízení (miniPCI sloty, ethernet zásuvky, aj.).

Mezi nejpoužívanější zařízení patří v poslední době právě tzv. **routerboardy**. Jako víceméně optimální pro běžné potřeby různých ISP se jeví Routerboard 532. Obsahuje poměrně dostatečně výkonný procesor MIPS 266 MHz, který je však možné v BIOSu zařízení přetaktovat až na 333 MHz. Další parametry jsou: 32 nebo 64 MB DDR operační paměť, 3x ethernet LAN, 2x miniPCI slot, 128 MB NAND paměti pro instalaci operačního systému, možnost napájet po ethernetu (PoE). S předinstalovaným MikroTikOS systémem (licence L4) je možné desku zakoupit v ceně kolem čtyř tisíc včetně DPH. Oba miniPCI sloty mohou být osazeny bezdrátovými kartami a tím značně zhodnotí celé zařízení.

2.3 Standardní funkce

2.3.1 Routování

Systém nabízí dva základní druhy routování paketů – dynamické a statické. Statické routování jsou běžné routy přidané správcem k explicitnímu směrování provozu do vzdálených sítí a pro specifikování výchozích bran v síti. Dynamické routování nabízí automatické přidání routy po přiřazení IP adresy k fyzickému adaptéru a dále je možné využít speciálních protokolů pro dynamické směrování.

Policy-Based Routing – metoda PBR vychází z principu označování jednotlivých paketů „routing-mark“ maskou (příznakový bit) a vybrání definované výchozí brány pro takto označené pakety.

ECMP (Equal Cost Multi-Path) routing – silný mechanismus, který nabízí jednoduše použitelný „load balancing“ – rozvržení zátěže na více zařízení/linek. Pro každý jednotlivý pár „zdroj-cíl“ IP adres se volí jiná brána z definovaného rozsahu. Tyto pravidla jsou buď generována podporovanými protokoly RIP či OSPF, nebo přidáním obyčejné statické routy s obsahem více výchozích bran.

2.3.2 Kontrola toku dat

V široké nabídce možností pro kontrolu toku dat vždy najdeme tu nejvhodnější alternativu pro naše potřeby. Systém podporuje základní sadu front – **PFIFO**, **BFIFO**, **SFQ**, **RED**, **PCQ**, **HTB**.

PFIFO, BFIFO **–** fronty založené na principu First-In, First-Out (P – packet, B – bytes). Jediným možným parametrem je velikost fronty – po naplnění této velikosti je každý další paket zahozen. Tyto fronty však nemají téměř praktické použití pro omezení rychlostí.

SFQ – Stochastic Fairness Queuing – základní funkcí je vnést do rozdělení provozu určitou dávku spravedlivosti. Algoritmus SFQ podle nastavené velikosti zásobníku propouští pakety pro jednotlivé streamy tak, aby každé spojení bylo obslouženo a při velkém zatížení nedocházelo k potlačení některých datových toků. SFQ také nelze využít přímo pro omezení rychlostí, slouží pouze jako pomůcka pro spravedlivé přidělení množství dat.

RED – Random Early Detection – mechanismus, který zabraňuje přetížení sítě pomocí náhodného zahazování paketů. V případě, že se naplnění fronty přiblíží velikosti parametru **red-min-threshold**, systém náhodně vybere a zahodí některé příchozí pakety. V případě úplného naplnění velikosti fronty – parametr **red-max-threshold –** se zahazují veškeré pakety do doby, než se fronta částečně uvolní. Díky vlastnosti **TCP** protokolu a jeho kontrolním mechanismům dojde k automatickému snížení rychlosti odesílání paketů na zdrojové/cílové straně spojení. **UDP** protokol bohužel tyto mechanismy nemá, proto *RED* nemá prakticky žádný vliv na přizpůsobení rychlosti. Samozřejmě je dodržena cílová maximální rychlost, poněvadž pakety jsou zahazovány. Nedochází však k reakci protokolu na „zhoršenou kvalitu linky“ zapříčiněnou náhodným zahazováním paketů.

PCQ – Per Connection Queuing – tento algoritmus dokáže nahradit drobné nedostatky *SFQ* mechanismů. *PCQ* vytváří své další podfronty, které jsou tříděné parametrem **pcq-classifier**. Jednotlivé podfronty klasifikujeme na základě zdrojové a cílové adresy nebo portu. Použití je vhodné zejména díky jednoduchosti a praktičnosti.

HTB – Hierarchical Token Bucket – nejsložitější a zároveň nejmocnější nástroj pro řízení datových toků. Jedná se o shaper s podporou tříd – je možné vytvářet složitou avšak vysoce propracovanou stromovou strukturu přesně podle potřeb zákazníků. Každá třída může mít jednu nebo více podtříd a zároveň musí mít jednu rodičovskou třídu. V operačním systému MikrotikOS se používají pro *HTB* dva parametry pro omezení rychlosti a jeden parametr pro nastavení priority obsloužení. Parametr **limit-at** je rychlostí, která je zaručená – CIR (Committed Information Rate) a **max-limit** je maximální možnou dosažitelnou rychlostí – MIR (Maximum Information Rate). Dále je možno nastavením **priority** upřednostnit vyřízení požadavků v určité třídě. **Burst** je užitečná funkce pro krátkodobé navýšení aktuální rychlosti. Jednoduše nastavitelná třemi parametry: **burst-time** označuje čas pro výpočet průměrné aktuální rychlosti, **burst-threshold** je hodnota průměrné rychlosti, do které se uplatní parametr navýšení na rychlost **burst-limit**. Každou 1/16 nastaveného času dochází k výpočtu průměrné rychlosti za uplynulou periodu. Pokud je tato vypočtená rychlost menší než hodnota **burst-threshold,** pak dojde k navýšení rychlosti třídy na hodnotu uvedenou v **burst-limit**. Jednoduše to lze demonstrovat rozdílem při stahování většího souboru a prohlédnutím běžné WWW stránky. Za předpokladu, že poslední periodu času neprotékala data, je průměrná rychlost toku dat nula, a proto vyhovuje hodnotě v **burst-threshold.** Uplatní se **burst-limit** a krátkodobě mohu stahovat touto rychlostí (logicky nastavenou větší než je hodnota **max-limit** u této třídy). Pokud je však množství dat větší a při výpočtu průměrné hodnoty přeskočíme velikost **burst-threshold**, dojde k omezení na rychlost **max-limit**. Rozdíl v běžném prohlížení stránek může být obrovský – s hodnotou **max-limit** na 512 kbps, **burst-limit** na 2048 kbps a **burst-threshold** na 384 kbps dosáhneme výborné špičkové krátkodobé rychlosti až dva megabity, ovšem při stahování většího objemu dat se rychlost vrátí na maximum definované třídou a neovlivní další zákazníky nedostatkem zdrojové konektivity.

2.3.3 Filtr

Mezi další nezbytné funkce kvalitního routeru patří samozřejmě funkční filtrování a označování paketů. Klasicky je filtr rozdělen do třech „řetězů“ – **input**,

output a **forward**. Dále je možné přidávat vlastní řetězy a vytvářet sofistikované přehledné stromové uspořádání. Mezi základní vlastnosti patří stavový firewall, filtrování protokolů p2p, klasifikace provozu dle mnoha podmínek – MAC adresy, IP adresy (jednotlivé či dle celých sítí), porty, protokoly (dále i dle jednotlivých FLAG ukazatelů), značky toků a spojení, obsahu paketů, počtu paketů za určitý časový okamžik a mnoho dalších.

- *INPUT* – zahrnuje pouze pakety určené pro samotný router
- *FORWARD* – pakety pouze procházející routerem (routované, bridgeované, natované, aj.)
- *OUTPUT* – pakety generované přímo routerem

Bridgeovaný provoz – mód pro rychlé předávání paketů prakticky nezatěžující router. Router pouze kontroluje, na jakou MAC adresu je daný paket směrován, a podle toho se rozhoduje, zda předá paket na další rozhraní nebo ne. Pokud se jedná o provoz IP, tento prochází nejdříve **prerouting** řetězem, vstupuje do **forward** a vystupuje **postrouting** řetězem. Ostatní provoz vstupuje přímo do fronty daného rozhraní bez proskakování jednotlivých filtračních řetězů.

Routovaný provoz – obsahuje tři základní typy vyhodnocení a prostupu paketů.

- Provoz směrovaný pro samotný router vstoupí přes **prerouting** do **input** řetězu a je zpracován vnitřně.
- Provoz generovaný routerem odchází **output** řetězem do **postrouting**, kde může být dále zpracován a přizpůsoben našim požadavkům.
- Provoz pouze prostupující nejprve projde **prerouting** řetězem, kde může být náležitě upraven, dále **forwardem** a vystoupí přes **postrouting**, kde můžeme opět uplatnit různá pravidla pro zpracování.

Connection tracking – samozřejmostí kvalitního zpracování paketů je funkce uchování informací o jednotlivých spojeních - tzv. stavové informace. Základní stavy jsou: **new** – paket začínající nové spojení, **established** – paket již patřící k nějakému známému spojení, **related** – paket navazující další nové spojení, které je však spjato již s nějakým známým předchozím. Příkladem pro **related** budiž například otevření datového kanálu při FTP přenosech – na příkazovém portu se obě strany dohodnou kde, kdo a jak otevře datové spojení. Toto spojení bude mít právě

vlajku **RELATED** ve vztahu k **ESTABLISHED** příkazovému spojení. Posledním typem spojení je **invalid** – paket, který není součástí nějakého známého spojení, může to být například podvržené číslo pořadí (sequence number) a další pokusy o zneužití routeru.

Značkování paketů – další neméně důležitou vlastností je možnost přidávat různé značky paketům uvnitř samotného zařízení. Značení je pouze pro vnitřní potřeby a nepřenáší se dále do sítě. Nabízí široké využití v rámci celého routeru, nejčastěji pro seskupování paketů při kontrole toku dat.

Network address translation – NAT – překlad adres, používá se pro oddělení místních sítí od samotného Internetu. Existují dva základní typy překladu – překlad zdrojových adres a překlad cílových adres.

- **Source NAT** – překlad zdrojových adres – pakety iniciované privátní sítí jsou bránou pozměněny náhradou za veřejnou adresu, která je směrovatelná v rámci celého Internetu.
- **Destination NAT** – překlad cílových adres – změna paketů určených pro cílovou privátní síť. Pomocí **dstnat**u je možné zpřístupnit počítač z privátní sítě do celého Internetu.

Počítače z privátní sítě nemají možnost přímé end to end konektivity, ta je suplována bránou s funkcí NAT. Některé speciální aplikace však nemohou v tomto režimu pracovat (například bezpečné IP tunely používané pro některé druhy VPN spojení - IPsec). Zvláštním případem cílového respektive zdrojového NATu jsou funkce **redirect** respektive **masquerade.**

- **Redirect** v hlavičce paketu pozměňuje pouze cílový port, cílová adresa zůstává stejná, paket tedy zpracovává to rozhraní, na něž paket dorazil.
- **Masquerade** funkce při průchodu paketu nastaví v hlavičce adresu daného rozhraní, jímž paket právě prochází. Router si vede vnitřní databázi těchto překladů, aby zpětně mohl přicházející pakety uvádět do správného stavu, doručitelného do privátní sítě.

Podrobněji aplikaci filtru a kontroly toku dat proberu v praktické části této práce.

2.3.4 Skriptování

Možnost využití integrovaného programovacího jazyka nabízí zjednodušení a aplikaci mnoha funkcí. Pomocí skriptů můžeme provádět různé kontroly routeru, nastavování parametrů front, filtru, aj. Pomocí skriptů se dá například naprogramovat plně funkční FUP (Fair User Policy) s omezením dat a rychlostí. V mém modelovém případě jsem využíval skripty k přidávání zákazníků do routeru – vytváření položek v **mangle** (značkování paketů), **forward** (masquerade, případně transparentní NAT pro veřejné IP adresy) a k vytvoření **simple queue** (garantovaná linka), aj.

Ve spojení s plánovačem můžeme spouštět skripty buď v přesném časovém okamžiku nebo v různých časových intervalech, případně kombinací obou možností. Prakticky jsem využíval plánovače pro zasílání zálohy konfigurace routeru na e-mailovou adresu. Tento jednoduchý skript:

```
/system backup save name=mail
/tool e-mail send to="zalohy@***.cz" subject="zaloha konfigurace - Dukla" file=mail.backup
```

zařadím do plánovače pomocí příkazu:

```
add interval=1d name="zaloha_mail" on-event=mail
```

a každý den na tuto e-mailovou adresu přijde soubor *mail.backup*, který později můžu do routeru zpětně nahrát a obnovit tak plnou konfiguraci (například dojde-li k havárii HW).

2.3.5 Bezdrátové sítě

Dnes již nezbytnou rozšiřující funkcí je tvorba bezdrátových sítí přímo pomocí speciálního HW, tzv. routerboardů. MikrotikOS podporuje dva nejrozšířenější typy čipsetů bezdrátových adaptérů - Prism a Atheros. Díky velké nabídce rozšiřujících karet je možné provozovat spoje na základě IEEE norem 802.11a, 802.11b a 802.11g. Výbornou vlastností systému je možnost nastavení

vlastní hodnoty **ack-timeout** – časového období, po které se čeká na potvrzení příjmu paketu druhou stranou spoje. Variabilním nastavením je možné dosáhnout spojení na velmi velké vzdálenosti – za ideálních podmínek až 50 kilometrů. Takový spoj však nemusí poskytovat dokonalou spolehlivost. V běžných podmínkách není problém vytvořit spoj 20 km dlouhý při propustnosti až 17 Mbit/s (za použití vysoce směrových antén, kvalitních přijímačů a normy IEEE802.11a). Při použití Atheros čipsetu je dále možné pro zlepšení propustnosti zapnout kompresi dat. Ta má však využití víceméně pouze pro prohlížení webových stránek, kde je častý přesun textu. V případě stahování dat či použití p2p sítí bude komprese spíše na obtíž.

Nstreme protokol – vlastní sada vylepšení přenosů od tvůrců MikrotikOS se týká point to point (multipoint) spojů. **Nstreme2** jde ještě dál a umožňuje použít dvě bezdrátové karty v páru – jedna pro příjem, druhá pro vysílání. Hlavní výhodou je například funkce „client polling“, která snižuje přístupovou dobu k médiu – karta před vysíláním neověřuje, zdali nevysílá některá další. O ověření se právě stará systém volby vysílací jednotky. Protokol prakticky neomezuje možnost tvořit dálkové spoje, při dlouhých spojích nedochází k tak velkým poklesům rychlosti jako u běžných bezdrátových protokolů.

Pro představu, jak jednoduchou konfiguraci MikrotikOS nabízí, se podívejte na následující ukázku. Nastavuji dva routery pro spoj v pásmu 5 GHz. Nastavení probíhá v příkazovém řádku (tučnou kurzívou jsou uvedené příkazy), pro ještě jednodušší postup je možné využít grafického rozhraní **WINBOX**, které si ukážeme prakticky později.

```
[admin@Hnizdo] interface wireless> set 0 mode=ap-bridge
frequency=5805 band=5ghz disabled=no ssid=hnizdo-dukla
name=AP_hnizdo_dukla
[admin@Hnizdo] interface wireless> print
Flags: X - disabled, R - running
0 name="AP_hnizdo_dukla" mtu=1500 mac-address=00:0C:42:05:00:22
arp=enabled
disable-running-check=no interface-type=Atheros AR5413
radio-name="000C42050022" mode=ap-bridge ssid="hnizdo-dukla" area=""
frequency-mode=superchannel country=no_country_set antenna-gain=0
frequency=5805 band=5ghz scan-list=default rate-set=default
supported-rates-b=1Mbps,2Mbps,5.5Mbps,11Mbps
supported-rates-a/g=6Mbps,9Mbps,12Mbps,18Mbps,24Mbps,36Mbps,48Mbps,
54Mbps
basic-rates-b=1Mbps basic-rates-a/g=6Mbps max-station-count=2007
ack-timeout=dynamic tx-power=default tx-power-mode=default
noise-floor-threshold=default periodic-calibration=default
burst-time=disabled fast-frames=no dfs-mode=none antenna-mode=ant-a
```

```
wds-mode=disabled wds-default-bridge=none wds-ignore-ssid=no
update-stats-interval=disabled default-authentication=yes
default-forwarding=yes default-ap-tx-limit=0 default-client-tx-
limit=0
hide-ssid=no security-profile=default disconnect-timeout=3s
on-fail-retry-time=100ms preamble-mode=both
[admin@Hnizdo] interface wireless> /ip address
[admin@Hnizdo] ip address> add address=10.1.0.1/24
interface=AP_hnizdo_dukla
[admin@Hnizdo] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
# ADDRESS NETWORK BROADCAST INTERFACE
0 10.1.0.1/24 10.1.0.0 10.1.0.255 AP_hnizdo_dukla
```

Ukázka nastavení RouterBoardu na mód AP.

- Příkaz ***set*** – číslo za příkazem set udává pořadové číslo rozhraní – tedy v našem případě první karta v systému. ***Mode=ap-bridge*** nastavuje příslušný režim pro vysílání – všechny možnosti **Mode** jsou následující
  - AP-bridge: access point mód – v režimu point to multipoint
  - bridge: access point mód – v režimu point to point
  - nstreme-dual-slave: komunikace protokolem nstreme
  - sniffer: přepnutí karty do promiskuitního módu – tj. odposlech všech rámců na dané frekvenci a ukládání do souboru pro pozdější analýzu
  - station: režim klient
  - station-wds, wds-slave: režim klient s podporou WDS rozšíření (wireless distribution system)
  - alignment-only: pomůcka pro správné nasměrování antény – ukazuje sílu signálu a úroveň šumu
- ***Frequency*** – číselně v MHz, nutno nastavit na hodnotu některého kanálu – nelze volit libovolné číslo.
- ***Band*** – vysílací pásmo
- ***SSID*** – Service Set IDentifier – název sítě, použité pro odlišení bezdrátových sítí.
- ***Name*** – název rozhraní
- Příkazem ***print*** vypíšeme podrobné nastavení a informace o sekci, ve které se právě nacházíme.
- ***/ip address*** – příkazy začínající lomítkem značí změnu aktuální sekce, pokud nevíme, jaké sekce je možné proskakovat, použijeme výše zmíněný příkaz **print**.

Podobně jako jsme nastavili první RouterBoard pro AP režim, nastavíme druhý do režimu klient.

```
[admin@Dukla] interface wireless> set wlan1 name=Dukla mode=station
ssid=hnizdo-dukla band=5ghz disabled=no
[admin@Dukla] interface wireless> print
Flags: X - disabled, R - running
0 R name="Dukla" mtu=1500 mac-address=00:0B:6B:34:5A:91 arp=enabled
disable-running-check=no interface-type=Atheros AR5213
radio-name="000B6B345A91" mode=station ssid="hnizdo-dukla" area=""
frequency-mode=superchannel country=no_country_set antenna-gain=0
frequency=5180 band=5ghz scan-list=default rate-set=default
supported-rates-b=1Mbps,2Mbps,5.5Mbps,11Mbps
supported-rates-a/g=6Mbps,9Mbps,12Mbps,18Mbps,24Mbps,36Mbps,48Mbps,
54Mbps
basic-rates-b=1Mbps basic-rates-a/g=6Mbps max-station-count=2007
ack-timeout=dynamic tx-power=default tx-power-mode=default
noise-floor-threshold=default periodic-calibration=default
burst-time=disabled fast-frames=no dfs-mode=none antenna-mode=ant-a
wds-mode=disabled wds-default-bridge=none wds-ignore-ssid=no
update-stats-interval=disabled default-authentication=yes
default-forwarding=yes default-ap-tx-limit=0 default-client-tx-
limit=0
hide-ssid=no security-profile=default disconnect-timeout=3s
on-fail-retry-time=100ms preamble-mode=both
[admin@Dukla] interface wireless> /ip address
[admin@Dukla] ip address> add address=10.1.0.2/24 interface=Dukla
[admin@Dukla] ip address> print
Flags: X - disabled, I - invalid, D - dynamic
# ADDRESS NETWORK BROADCAST INTERFACE
0 10.1.0.2/24 10.1.0.0 10.1.0.255 Dukla
```

Okamžitě po nastavení obou routerů je možné nově vytvořený spoj plně využít.

2.3.6 Ostatní služby

Jen pouhý výčet všech funkcí systému MikroTikOS by byl velmi obsáhlý. V této kapitole se tedy přehledně zaměříme pouze na nejdůležitější a nejčastěji využívané možnosti a funkce s jejich stručným popisem.

- **FTP server** – jednoduchý zabudovaný FTP server slouží především k přenosům rozšiřujících nebo aktualizačních balíčků. V případě routerboardu je výrazným handicapem nedostatek místa pro tyto soubory (malá kapacita integrované paměti).
- **MAC Level Access** – výhodné pro první nastavení routerboardu, který nemá z výroby nastavenou žádnou IP adresu (není možné použít klasické způsoby konfigurace jako je WINBOX aplikace, telnet nebo SSH konzole).

- **SSH, Telnet** – oba protokoly obsahují jak část serverovou, tak klientskou. Přes protokoly je možné systém plně spravovat a konfigurovat.
- **WINBOX konzole** – aplikace pro správu routeru v plně grafickém prostředí. Veškeré funkce je možné spravovat stejným způsobem jako pomocí běžné textové konzole. Hierarchie příkazů ani parametry se při použití v různých prostředích nemění.
- **IP tunely** – široká podpora protokolů.
    - **IPIP –** tunel vyhovující definici v RFC 2003, existující IP pakety jsou přebaleny do nových IP paketů a následně směrovány a na cílovém hostiteli opět rozbaleny. Nejčastěji se využívá pro zpřístupnění Intranetu v různých vzdálených místech pomocí běžného Internetu.
    - **L2TP** – Layer 2 Tunnel Protocol – podporuje kryptované kanály v IP sítích – používá se pro zabezpečené spojení router-to-router nebo stejně jako IPIP tunely pro spojení lokálních sítí přes Internet.
    - **PPPoE** – Point to Point Protocol Over Ethernet – emulace PPP spojení přes klasické ethernetové sítě. V dnešní době běžně používáno většími poskytovateli pro jednoznačnou identifikaci uživatelů. PPPoE poskytuje mnoho výhod – ve spojení s RADIUS autentifikací můžeme každému uživateli jednoduše sledovat datové toky, zamezit zneužití linky neoprávněnými osobami, atd.
    - **PPTP** – obdoba L2TP protokolu.
- **Grafické výstupy** – od verze 2.9.x podporuje MikroTikOS grafické výstupy různých funkcí. Praktické ukázky budou obsahem kapitoly 5.1.
- **DHCP** – podpora server i klient.
- **DNS** – systém je možné nastavit do režimu DNS cache, běžně odpovídá klientům na UDP a TCP portu 53, pomocí vnitřní cache paměti však dokáže minimalizovat množství požadavků posílaných externím DNS serverům.
- **HTTP proxy** – užitečná funkce – ukládá do vnitřní paměti webové stránky, které při dalším požadavku nemusí opět stahovat z Internetu, ale zašle je ze své paměti. Proxy může fungovat standardně, tj. klienti si jej nastavují

ve svém prohlížeči. Samozřejmě nechybí i podpora pro transparentní proxy. Veškeré požadavky se řídí sofistikovanými access listy, které pracují na mnoha principech – zdrojové adresy, cílové adresy, URL adresy, metody požadavku (connect, get, post, aj.). Nechybí možnost logování všech požadavků pro případné ladění služby.

- **NTP služby** – Klient synchronizuje lokální hodiny dle vzdáleného serveru. Služba server dokáže poskytovat informace pro dotazující se klienty – vhodné jako časový server pro lokální síť.
- **Bandwidth, ICMP bandwidth** – služby pro monitorování prostupnosti jednotlivých linek.
- **TORCH** – Realtime Traffic Monitor – vynikající pomůcka pro sledování procházejícího provozu routerem. Pomocí mnoha definovatelných pravidel je možné přesně zjistit, jaký provoz právě protéká určitým rozhraním. Provoz můžeme klasifikovat dle následujících parametrů: *protokol, zdrojová adresa, cílová adresa, zdrojový port, cílový port*.
- **UPS monitor** – v moderně vybavené síti je nezbytné napájení záložním zdrojem v případě výpadku dodávky elektrické energie. Pomocí této funkce je možné monitorovat stav UPS zařízení a případně reagovat na jeho pokyny.

Pro každého poskytovatele může být důležitá zcela odlišná funkce. Systém MikroTikOS se snaží být univerzální pomůckou, a tak obsahuje i mnoho dalších méně významných a používaných funkcí. Vývoj tohoto softwaru probíhá velice rychle, a proto se nové vlastnosti programují dle požadavků provozovatelů těchto zařízení.

3 Operační systém Linux v síťovém využití

3.1 Linux jako multifunkční server

Když běžný uživatel zadá do internetového vyhledávače pojem „LINUX“, dostane jako odpověď miliony výsledků. Množstvím těchto nalezených výsledků se Linux směle vyrovnává nejrozšířenějšímu operačnímu systému Windows. Mnoho uživatelů osobních počítačů toto slovo zná jako určité „heslo“, ale ve skutečnosti neví, co vše se pod tímto slovem skrývá. Dovolím si citovat z internetového portálu ***www.linux.cz***:

„Linux je moderní operační systém (OS), jehož **ovládání je stejně přívětivé jako u jiných systémů** a který **obsahuje velké množství ovladačů** pro nejrůznější standardizovaný hardware.“

Jako pokročilý uživatel a specialista na určitá počítačová odvětví se však musím pokusit alespoň o částečnou kritiku. *„Linux je moderní systém“*, to rozhodně nelze popřít. *„Linux obsahuje velké množství ovladačů pro nejrůznější standardizovaný hardware“* – s tímto tvrzením lze víceméně též souhlasit, i když se najde stále ještě nemálo výrobců, kteří pro své komponenty nejsou ochotni uvolnit ovladače pro linuxové distribuce. Avšak většina výrobců si je v dnešní době již vědoma toho, že tento systém na trhu ukrajuje nezanedbatelnou část coby operační systém, a tak si nedovolí oficiálně tento systém nepodpořit a možná tak ztratit mnoho, dříve spokojených, zákazníků. *„Ovládání je stejně přívětivé jako u jiných systémů“* – tato formulace může a nemusí být pravdivá. Dle mého názoru závisí na pohledu jednotlivého uživatele.

Všeobecným problémem je zažitý dlouholetý stereotyp, kdy existovaly pouze Windows a vše ostatní bylo „špatné, málo rozšířené, nefunkční, aj.“. Jde o subjektivní hodnocení každého z nás. Největším problémem není větší složitost a náročnost na ovládání systému, problémem většiny z nás je spíše neochota učit se cokoliv jiného, když to, co mám, znám a ovládám, i přes drobné nedostatky vyhovuje. Faktem je, že dnešní podoba grafického rozhraní většiny distribucí Linuxu je prakticky shodná s vizuálním stylem Windows XP, které jsou nejvíce zastoupené na trhu s operačními systémy. Většina běžných činností v systému je na první pohled těžko rozeznatelná od postupů v systémech Windows. Jedno však systému neupřeme

– pod čepicí těchto vykrášlených vlastností, funkcí a programů se pořád skrývá ten „ošklivý zdroják či konfigurák“, který na vás vybafne při prvním menším problému v systému. A v tento moment se z krásného a jednoduše ovladatelného systému stává, pro běžného uživatele, nepochopitelná smršť anglických výrazů a pojmů. Stále se odvažuji tvrdit, že systém Linux je pouze pro pár nadšenců, kteří mají čas a chuť poznávat, učit se a trápit se.

Poslední dobou Linux prožívá obrovský boom a začíná se rozšiřovat i na zcela běžné osobní počítače. Nespornou výhodou systému je jednoduchost nové instalace v případě havárie. Celé nastavení systému se provede pouhým nakopírováním zálohovaných konfiguračních souborů – žádné složité klikání a nastavování všeho úplně od začátku.

Linux je odvozeninou komerčního serverového operačního systému UNIX. Historie sahá až do roku 1983, kdy **Richard Stallman** napsal vlastní odnož UNIXu, jejímž cílem byla existence volně šiřitelného (tj. zdarma) operačního systému – projekt byl nazván **GNU**. Později se začalo pracovat na tzv. jádře, které musí zajistit komunikaci operačního systému s hardwarem – projekt **Hurd**. Známějším jménem je Linus Torvalds, který vytvořil vlastní unixové jádro, jež nakonec oslovilo širší veřejnost. Tento projekt se začal zdárně vyvíjet, zatímco projekt GNU víceméně dodnes stagnuje.

Pro nás však přívětivost Linuxu není stěžejní, my se zaměříme na jeho praktické a víceméně nejčastější využití jako **síťového serveru**.

Využití Linuxu jako síťového serveru může být chápáno jako obrovský komplex služeb pro poskytovatele – routování paketů, QOS služby a omezení rychlostí, poskytování prostoru uživatelům pro WWW prezentace, výkonný firewall, databázový server, kompletní poštovní služby (odesílání emailů, příjem přes zabezpečené i nezabezpečené protokoly pop3 či imap, webové rozhraní pro email), FTP server, DNS server a mnoho dalších služeb. Pro představu se podíváme, co se skrývá pod jednotlivými nejpoužívanějšími službami systému.

Apache

Služba poskytuje prostor pro prezentace na síti Internet. Vývoj začal v roce 1996, během deseti let se z Apache http serveru stal fenomén. V roce 2005 byl dle průzkumů Apache nasazen na více než sedmdesáti procentech serverů

poskytujících webový prostor. Tomuto velkému rozšíření jistě pomáhá i vlastnost, že může být bez problémů provozován na počítačích s operačním systémem Windows. V současné době je k dispozici několik plně kompatibilních verzí – aktuální verze 2.2.x, předchozí vývojová řada 2.0.x a stále ještě používaná verze 1.3.x. Poslední zmíněná verze se hodí především na starší a méně výkonné počítače – neklade velké nároky na HW, ale zároveň nedokáže poskytnout takový výkon jako v případě nejnovějších verzí.

Samotný Apache server však v dnešní době nemá prakticky žádný význam – dokáže zpracovávat pouze statické WWW stránky, které jsou v dnešní době rozhodně pasé. Abychom získali plnohodnotný server, je třeba jej doplnit moduly pro rozšířené vlastnosti. Nejrozšířenějšími a nejdůležitějšími jsou PHP a podpora databází (MySQL, PgSQL, ODBC a další ovladače). PHP – *hypertext preprocesor* – univerzální skriptovací jazyk, který můžeme používat přímo uvnitř HTML. Pomocí PHP (ekvivalentem pro Windows prostředí je ASP) se generují složité stránky s dynamickým obsahem, například použití jako internetové on-line obchody, robustní diskusní fóra, výpočtové programy a další. Avšak ani spojení Apache a PHP nedokáže uspokojit všechny požadavky pro moderní tvorbu webových prezentací. Teprve Apache, PHP a podpora databáze činí server zajímavým a široce využitelným.

Databázové servery

Použití databází není samozřejmě omezeno pouze pro webové prezentace. Častým úkolem je poskytovat informace o zákaznících a jejich účtech – pro účely emailové pošty, propojení s FTP servery a dále třeba pro autentizační server Radius. Databázových serverů možných provozovat na Linuxu existuje mnoho. Patří mezi ně jak komerční systémy, tak i volně šiřitelé databáze.

Mezi nejznámější svobodný databázový software patří MySQL a PostgreSQL. MySQL je rychlejší, ale nepodporuje některé užitečné vlastnosti jako jsou například vnořené dotazy. Naproti tomu PgSQL podporuje kompletní jazyk SQL, ale v rychlosti výrazně pokulhává. Z komerčních databází můžeme použít Oracle nebo Sybase.

Poštovní systém

K řízení emailové pošty existuje také mnoho různorodých poštovních systémů. Nejznámějším z nich je jistě Sendmail – robustní MTA (Mail Transfer Agent) nabízí obrovské možnosti v konfiguraci, jeho problémem jsou dlouhodobě známé a přetrvávající bezpečnostní problémy. Jeho největší konkurencí je Postfix, dnes hojně používaný a v mnoha Linuxových distribucích již nastavený jako výchozí. MTA se stará pouze o přenos zpráv z a na server, o doručování do jednotlivých uživatelských složek se stará MDA – Mail Delivery Agent. Těch je na výběr dostatek a jejich použití závisí víceméně na požadavcích jednotlivých ISP. Pro samotné čtení jednotlivých emailů se používají služby POP3 a IMAP. IMAP poskytuje velkou výhodu v tom, že jednotlivé zprávy jsou zpracovávány nejprve na serveru, do poštovního klienta máme možnost stáhnout například pouze hlavičky zpráv, a teprve když se rozhodneme, že tuto zprávu chceme přečíst, stáhneme ji celou. Speciálním případem je ještě tzv. Webmail Service, kdy emailové rozhraní běží přímo na WWW stránkách (Seznam.cz, Centrum.cz a další systémy). Webmail Service je velmi časté využití IMAP protokolu pro rozšířenou práci s poštovními zprávami.

FTP server

V době, kdy se Internet začal dostávat do povědomí i běžných lidí ve společnosti, zažíval protokol FTP (File Transfer Protocol) největšího rozmachu. Na nějakou FTP adresu se dalo narazit prakticky každým krokem Internetem. V dnešní době tento protokol vytlačují různé nové druhy P2P (Peer to Peer) sítí. I přes velkou nelibost nahrávacích společností, filmových producentů, výrobců softwaru a dalších komerčních skupin se prostřednictvím FTP serverů šířilo prakticky vše, co existovalo. Pro tyto zájmové skupiny byl boj proti nelegálním FTP serverům mnohem jednodušší, než je dnešní boj proti P2P sítím. Dnes je FTP protokol využíván víceméně pouze minoritně – nahrávání webových prezentací na servery, distribuce programových aktualizací a samotných programů. Jako prostředek pro sdílení dat v sítích je tento protokol již pasé.

Ve výčtu služeb bychom mohli i nadále podrobně pokračovat, ale to by zabralo místo pro celou další práci. Každopádně pro představu, že operační systém Linux bude vždy tvořit alespoň nedílnou část sítě, to bohatě stačí.

3.2. Doplňkové služby

V šíři poskytovaných služeb je Linux prakticky nevyčerpatelný. Proti Windows je nespornou výhodou, že většina programového vybavení je zdarma – je to základní politika Open Source. Windows však nelze upřít, že se v posledních několika letech objevuje množství freeware programů. Toto vybavení přesto nedokáže vytvořit multifunkční systém, který by byl zcela zdarma. Použití Linuxu nekončí jen u poskytovatelů Internetových služeb. Jako s aplikačním serverem se s Linuxem setkáme prakticky v každé firmě vybavené počítačovou sítí. Ve většině případů slouží tento server třeba jen jako obyčejná výchozí brána do Internetu, poskytující základní služby, jako je routování a masquerading paketů. Především díky nulové ceně si našel oblíbení i jako počítač, kde můžeme nasdílet svá firemní data, zpracovávat komunikaci s odběrateli a dodavateli, apod. Ke sdílení dat v prostředí OS Linux se používá **Samba Server**. Je to programový balík, který implementuje SMB protokol, který je spíše známý ve světě Windows jako NetBios protokol. Dále je možné přes **Samba Server** sdílet tiskárny připojené k samotnému serveru – tím je možné ušetřit náklady na pořízení dražší tiskárny vybavené přímo síťovým rozhraním. Za další užitečnou věc se dá určitě považovat kešovací proxy server – **SQUID**. Mnozí možná namítnou, že doba omezených rychlostí a množství dat tekoucích Internetem je dávno za námi a tudíž tento systém bude plnit zbytečnou funkci. Rozhodně tomu tak není, **SQUID** je dnes multifunkční aplikace, která nabízí mnoho cenných vlastností. Díky dobře nakonfigurovanému systému dokážeme okamžitě získat perfektní přehled o využití Internetu (pouze prohlížení webových stránek). Existují další nadstavbové aplikace, které dokáží přehledně zpracovat logy a nabídnout plné statistiky provozu. Pomocí **SQUID**u je také možné filtrovat přístup na webové stránky, a to pomocí klíčových slov, regulárních výrazů, IP adres, doménových jmen a dalších rozsáhlých parametrů.

3.3 Ukázkové konfigurace

Veškeré pokročilé konfigurace je nutné provádět pouze přes příkazový řádek – tzv. konzoli. To je zásadní rozdíl Linuxu proti MikrotikOS, ve kterém se dá použít příjemné a přehledné grafické prostředí a nastavení si „jednoduše naklikat“.

Na Internetu existuje mnoho více či méně podrobných návodů, jak požadovanou věc správně nakonfigurovat. Málokdy je správce postaven do situace, že něco dělá bez odborné pomoci manuálu či povedeného návodu, který někdo zpracoval po své úporné snaze a později se chtěl podělit s ostatními. Výhodou je, že většina těchto návodů se dá najít i v českém nebo alespoň slovenském jazyce. Většina konfiguračních souborů obsahuje přímo komentáře, co který parametr nastavuje.

Ukázka části konfiguračního souboru *httpd.conf* – http server Apache

```
# Use name-based virtual hosting.
#
#NameVirtualHost *:80
#
# NOTE: NameVirtualHost cannot be used without a port
specifier
# (e.g. :80) if mod_ssl is being used, due to the nature of
the
# SSL protocol.
#
# VirtualHost example:
# Almost any Apache directive may go into a VirtualHost
container.
# The first VirtualHost section is used for requests
without a known
# server name.
#
#<VirtualHost *:80>
#    ServerAdmin webmaster@dummy-host.example.com
#    DocumentRoot /www/docs/dummy-host.example.com
#    ServerName dummy-host.example.com
#    ErrorLog logs/dummy-host.example.com-error_log
#         CustomLog  logs/dummy-host.example.com-access_log
common
#</VirtualHost>

NameVirtualHost *:88
<VirtualHost *:88>
    ServerAdmin benyto@centrum.cz
    DocumentRoot /var/www/html/neplatici
    ServerName Neplatici
    ErrorLog /var/log/httpd/neplatici_error_log
    CustomLog /var/log/httpd/neplatici-access_log common
</VirtualHost>
```

Jak se sami můžete přesvědčit, po krátkém popisu funkce a důležitých upozorněních následuje i praktický příklad, jak danou funkci můžeme využít. Řádky, které začínají znakem # jsou komentáře. Text, který není tímto znakem uvozen, obsahuje konfigurační příkazy.

Další příklad demonstruje složitost tvorby Linuxových pravidel pro firewall. Pro kompletní zvládnutí firewallu v Linuxu je třeba prostudovat manuál,

který má přes 200 stran. Následující ukázka je výňatkem z části inicializačního skriptu.

```
#pravidla pro privatni adresy
/sbin/iptables -A INPUT -i eth2 -s 10.0.0.0/8 -j DROP
/sbin/iptables -A OUTPUT -o eth2 -d 10.0.0.0/8 -j DROP
/sbin/iptables -A INPUT -i eth2 -s 172.16.0.0/12 -j DROP
/sbin/iptables -A OUTPUT -o eth2 -d 172.16.0.0/12 -j DROP
/sbin/iptables -A INPUT -i eth2 -s 192.168.0.0/16 -j DROP
/sbin/iptables -A OUTPUT -o eth2 -d 192.168.0.0/16 -j DROP
/sbin/iptables -A INPUT -i eth2 -s 240.0.0.0/5 -j DROP
/sbin/iptables -A OUTPUT -o eth2 -d 240.0.0.0/5 -j DROP

#sluzby
/sbin/iptables -A INPUT -i eth2 -p tcp -d 81.2.213.2 --dport 21 -j
ACCEPT
/sbin/iptables -A INPUT -i eth2 -p tcp -d 81.2.213.2 --dport 22 -j
ACCEPT
/sbin/iptables -A INPUT -i eth2 -p tcp -d 81.2.213.2 --dport 25 -j
ACCEPT
/sbin/iptables -A INPUT -i eth2 -p tcp -d 81.2.213.2 --dport 53 -j
ACCEPT
/sbin/iptables -A INPUT -i eth2 -p udp -d 81.2.213.2 --dport 53 -j
ACCEPT
/sbin/iptables -A INPUT -i eth2 -p tcp -d 81.2.213.2 --dport 80 -j
ACCEPT
/sbin/iptables -A INPUT -i eth2 -p tcp -d 81.2.213.2 --dport 110 -
j ACCEPT
/sbin/iptables -A INPUT -i eth2 -p tcp -d 81.2.213.2 --dport 143 -
j ACCEPT
/sbin/iptables -A INPUT -i eth2 -p tcp -d 81.2.213.2 --dport 443 -
j ACCEPT
/sbin/iptables -A INPUT -i eth2 -p tcp -d 81.2.213.2 --dport 563 -
j ACCEPT

#vse ostatní zakazat
/sbin/iptables -A INPUT -p tcp --dport 1:1023 --syn -i eth2
-j DROP
```

Následující obrázek ukazuje, jak vypadají tato pravidla zavedená v systému.

```
Chain INPUT (policy ACCEPT 1715K packets, 926M bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       all  --  eth2   *       255.255.255.255      0.0.0.0/0
    0     0 DROP       all  --  eth2   *       127.0.0.0/8          0.0.0.0/0
    0     0 DROP       all  --  eth2   *       240.0.0.0/5          0.0.0.0/0
    0     0 ACCEPT     tcp  --  eth2   *       0.0.0.0/0            81.2.213.2          tcp dpt:21
    0     0 ACCEPT     tcp  --  eth2   *       0.0.0.0/0            81.2.213.2          tcp dpt:22
    0     0 ACCEPT     tcp  --  eth2   *       0.0.0.0/0            81.2.213.2          tcp dpt:25
    0     0 ACCEPT     tcp  --  eth2   *       0.0.0.0/0            81.2.213.2          tcp dpt:53
    0     0 ACCEPT     udp  --  eth2   *       0.0.0.0/0            81.2.213.2          udp dpt:53
    0     0 ACCEPT     tcp  --  eth2   *       0.0.0.0/0            81.2.213.2          tcp dpt:80
    0     0 ACCEPT     tcp  --  eth2   *       0.0.0.0/0            81.2.213.2          tcp dpt:110
    0     0 ACCEPT     tcp  --  eth2   *       0.0.0.0/0            81.2.213.2          tcp dpt:143
    0     0 ACCEPT     tcp  --  eth2   *       0.0.0.0/0            81.2.213.2          tcp dpt:443
    0     0 ACCEPT     tcp  --  eth2   *       0.0.0.0/0            81.2.213.2          tcp dpt:563
    0     0 ACCEPT     tcp  --  lo     *       81.2.213.2           81.2.213.2          tcp dpt:3306
    0     0 DROP       tcp  --  eth2   *       0.0.0.0/0            0.0.0.0/0           tcp dpts:1:1023 flags:0x16/0x02
```

Obr. 1 - Výpis pravidel **IPTABLES**

Poslední příklad bude ze skriptu **HTB.INIT**, s jehož pomocí můžeme priorizovat datové toky, omezovat rychlosti, případně pouze spravedlivě rozdělovat datové pásmo pro více účastníků. V této praktické ukázce je celkové pásmo

rozděleno mezi několik účastníků a každému z nich je zaručená minimální rychlost. Jednotlivé konfigurace klientů jsou uloženy ve zvláštních souborech. Nejprve definujeme kořenovou třídu – soubor *eth1-2.root* obsahuje pouze dva řádky, a to: *RATE=640kbit BURST=15k.*

Podtřída *eth1-2:10* obsahuje:

RATE=320kbit CEIL=640kbit BURST=60k LEAF=sfq.

RATE definuje minimální zaručenou rychlost, *CEIL* definuje maximální dosažitelnou rychlost pro celou třídu, *BURST* říká, kolik dat se může odeslat plnou neomezenou rychlostí, a parametr *LEAF* definuje použití fronty typu SFQ. Nyní již definujeme jednotlivá pravidla pro uživatele - například *eth1-2:10:151.kautman*

`RATE=64kbit`	Minimální zaručená rychlost
`CEIL=640kbit`	Maximální dosažitelná rychlost
`BURST=60k`	Burst počet dat
`LEAF=sfq`	Typ fronty
`PRIO=7`	Priorita s jakou se obsluhují pakety
`RULE=,192.168.0.151/32`	Pravidlo, podle kterého budou
`RULE=,192.168.0.152/32`	klasifikovány pakety

Po spuštění **HTB.INIT** se vykonají příkazy dle konfiguračních souborů a v systému najdeme takto zavedená pravidla pro podtřídu 151 s názvem kautman.

```
filter parent 1: protocol ip pref 100 u32 fh 800::808 order
2056 key ht 800 bkt 0 flowid 1:171
  match c0a800ab/ffffffff at 16
filter parent 1: protocol ip pref 100 u32 fh 800::809 order
2057 key ht 800 bkt 0 flowid 1:171
  match c0a800ac/ffffffff at 16
```

A takto vypadá celá vytvořená hierarchie tříd zavedená v systému:

```
### eth1: traffic classes
class htb 1:101 parent 1:10 prio 3 rate 64000bit ceil
640000bit burst 60Kb cburst 1920b
class htb 1:10 parent 1:2 rate 320000bit ceil 640000bit
burst 60Kb cburst 1920b
class htb 1:111 parent 1:10 prio 3 rate 64000bit ceil
640000bit burst 60Kb cburst 1920b
class htb 1:2 root rate 1000Kbit ceil 1000Kbit burst 15Kb
cburst 2100b
class htb 1:30 parent 1:2 prio 1 rate 1000Kbit ceil
1000Kbit burst 15Kb cburst 2100b
class htb 1:141 parent 1:10 prio 1 rate 128000bit ceil
640000bit burst 60Kb cburst 1920b
class htb 1:151 parent 1:10 leaf 151: prio 7 rate 64000bit
ceil 640000bit burst 60Kb cburst 1920b
class htb 1:5 parent 1:2 leaf 5: prio 0 rate 128000bit ceil
128000bit burst 15Kb cburst 1664b
```

Pokud potřebujeme podrobnější informace, je možné získat informační výpis v této podobě:

```
class htb 1:151 parent 1:10 leaf 151: prio 7 rate 64000bit
ceil 640000bit burst 60Kb cburst 1920b
Sent 42895049 bytes 130441 pkt (dropped 470, overlimits 0
requeues 0)
rate 53208bit 6pps backlog 0b 0p requeues 0
lended: 1093315 borrowed: 211099 giants: 0
tokens: -5042287 ctokens: 6200
```

4 Praktické použití MikroTikOS v sítích drobných ISP

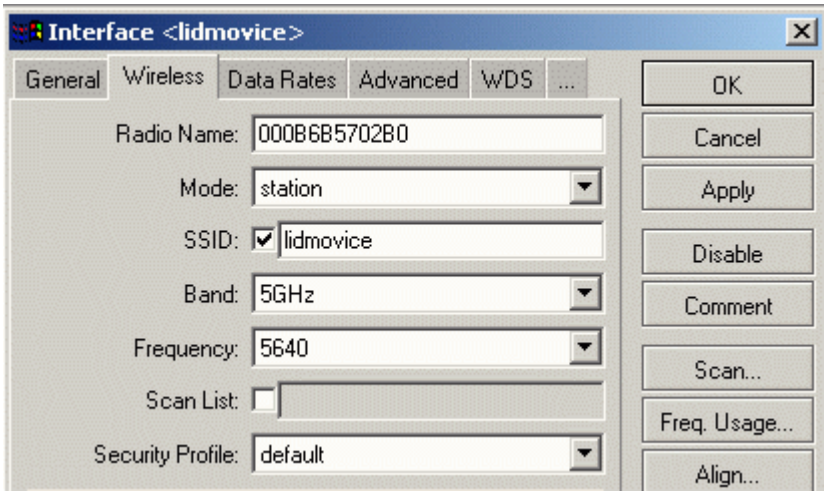
4.1 Nasazení inteligentních přístupových bodů

Inteligentní přístupový bod – v této kapitole se pokusím uvést na pravou míru definici „inteligentní“. V době, kdy neexistovala možnost v Access Pointu manipulovat jakkoliv s datovým tokem či pakety, museli jsme spoléhat na kvalitní hardware a trochu štěstí, že právě onen přístupový bod bude stabilní a bezproblémový. Ovšem s rozvojem mnoha typů p2p sítí se začaly vyskytovat obrovské problémy se stabilitou celých sítí. Princip, na jakém jsou tyto sítě postaveny, dokázal bez problémů zlikvidovat dlouhou dobu zcela stabilní síť.

Tvůrci těchto programů pro sdílení dat v Internetu kladou důraz především na možnost co největšího a nejjednoduššího šíření jakýchkoliv dat mezi co možná největší počet uživatelů. To s sebou právě přináší velké problémy v počtu navazovaných spojení, zahlcování množstvím paketů, aj. Dovolím si tvrdit, že v dnešní době tvoří tyto, většinou víceméně nelegální, datové přenosy alespoň mírně nadpoloviční většinu z objemu dat, který denně proteče Internetem.

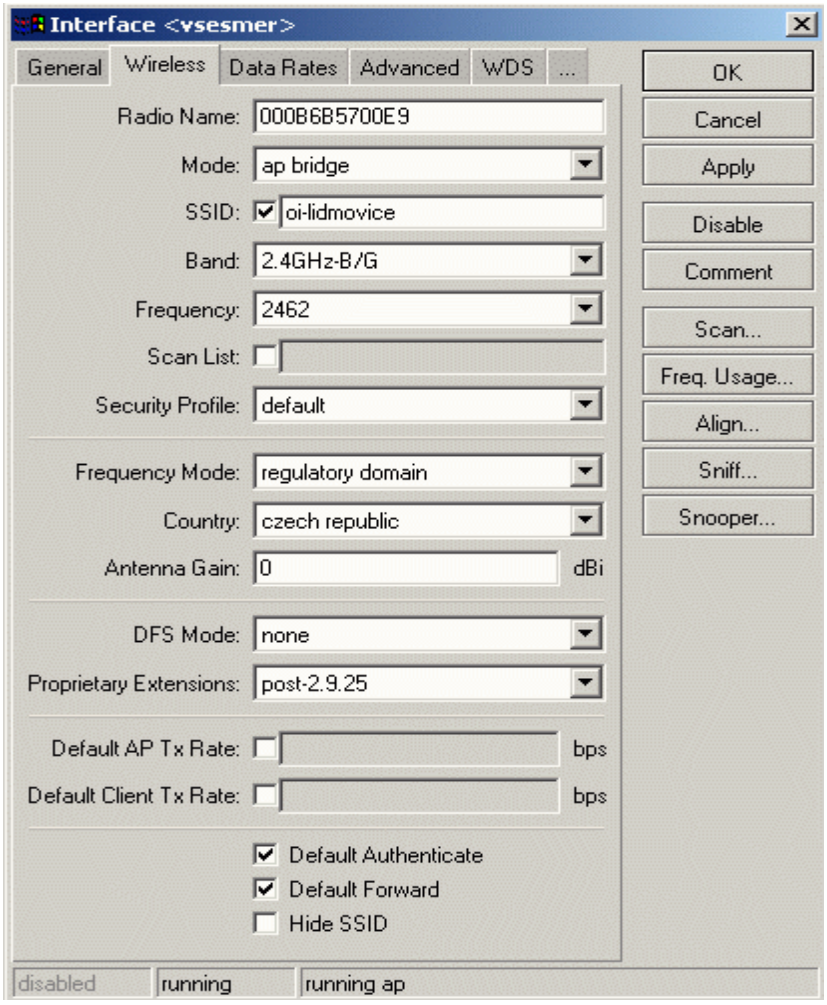
Dalším obecným problémem u bezdrátových sítí je obrovský nárůst množství drobných poskytovatelů a v neposlední řadě také rychle se zvyšující běžně poskytovaná rychlost. Všechny tyto vyjmenované „vlastnosti a možnosti“ můžeme označit za „nepřítele“ většiny poskytovatelů. Podívejme se tedy, jak získat kontrolu nad tokem dat ještě přímo na přístupovém bodě. Drobnou nevýhodou tohoto řešení je mnohem vyšší pořizovací cena než u běžného AP – ta se nám však vrátí během prvních měsíců používání celého systému. Základem AP bude RouterBoard 532 s aktivovaným MikroTikOS systémem (lze použít i jiné alternativy jako například *AP Router (Realtekové bezdrátové čipsety, M0n0wall,* aj.). RouterBoard osadíme požadovaným počtem bezdrátových miniPCI karet (například CM9 v ceně cca 1000,- Kč, CM10 v ceně cca 900,- Kč). V mém ukázkovém případě se jedná o dvě bezdrátové karty typu CM9. Celý router, sloužící nyní zároveň jako přijímací i vysílací část, přijde na zhruba 6 tisíc korun. To je ale pouze mírně vyšší cena než dvě kvalitní profesionální AP. MiniPCI karta CM9 je postavena na čipsetu Atheros 5213, ten podporuje všechny tři nejrozšířenější normy bezdrátových

přenosů – normu **802.11a** pro **5 GHz**, **802.11b** a **802.11g** pro frekvenci **2.4 GHz**. Jednu kartu používám pro příjem dat od poskytovatele – zpravidla budované dnes již páteřními spoji na 5 GHz.



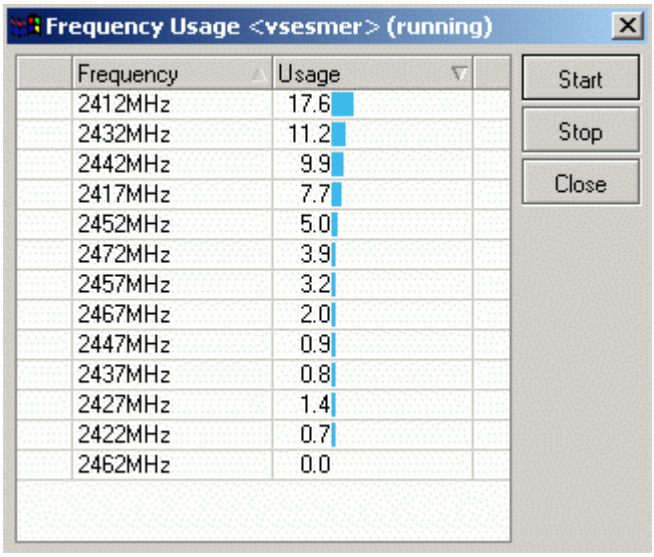
Obr. 2 – Nastavení miniPCI karty do módu stanice v pásmu 5 GHz

Druhou kartu používám jako přístupový bod pro zákazníky. Konfigurace je obdobná.



Obr. 3 – Konfigurace přístupového bodu pro zákazníky

Velice užitečnou funkcí je při konfiguraci přístupového bodu možnost zjistit, jakým způsobem jsou obsazené jednotlivé kanály. K tomu slouží tlačítko **Freq. Usage …** v pravé části konfiguračního okna (obrázek 3).

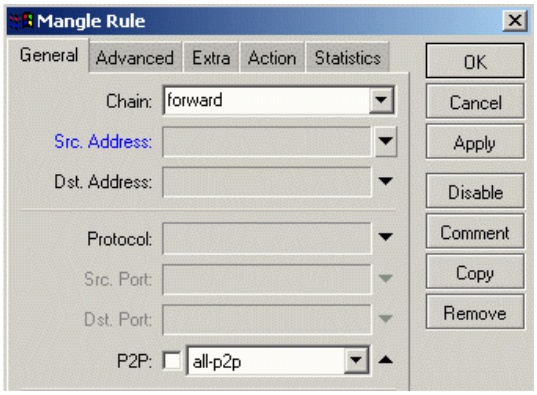


Obr. 4 – Obsazenost jednotlivých kanálů v pásmu 2.4 GHz

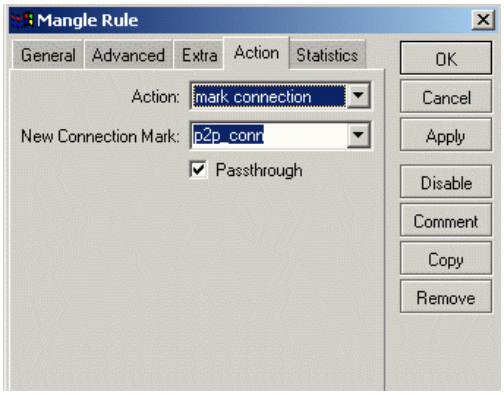
Touto základní konfigurací jsem udělal z RouterBoardu úplně běžné zařízení pro poskytování Internetu zákazníkům v dosahu naší antény. Nyní provedu některá další nastavení pro pokročilé řízení toku naším routerem. Jak jsem zmiňoval v úvodu kapitoly, naším zřejmě největším nepřítelem budou P2P sítě. MikroTikOS disponuje stejně jako Linux detekčními schopnostmi typu datových přenosů v sedmé vrstvě OSI modelu TCP/IP protokolu. Přidám pravidlo do **MANGLE** a na kartě **GENERAL** vyberu místo, kde budu provádět označování – **chain forward** a dále vyberu označování všech p2p sítí. Na kartě **ACTION** pak zvolím akci **Mark connection** a jako značku uvedu třeba **p2p_conn**. Vlastnost **Passthrough** znamená, že paket po průchodu tímto pravidlem pokračuje i nadále ve zpracování ostatními pravidly firewallu – běžná politika při zpracování paketů je totiž odlišná. Každý paket prochází pravidly od prvního k poslednímu. Pokud nějakému pravidlu vyhoví, provede se požadovaná akce a zpracování se ukončí.

Rozpoznání paketů příslušejících různým P2P službám není a z principu ani nemůže být dokonalé – tvůrci vždy budou bojovat proti jakémukoliv omezování jejich programů. V novějších verzích například do jednotlivých paketů přidávají jiné hlavičky, aby tyto filtry obalamutily. Boj *P2P sítě versus ISP* je tak nikdy nekončící

těžká válka. Výsledkem je, že filtry nedokáží korektně ohodnotit všechny pakety patřící těmto službám, a ty proto uniknou pravidlům.

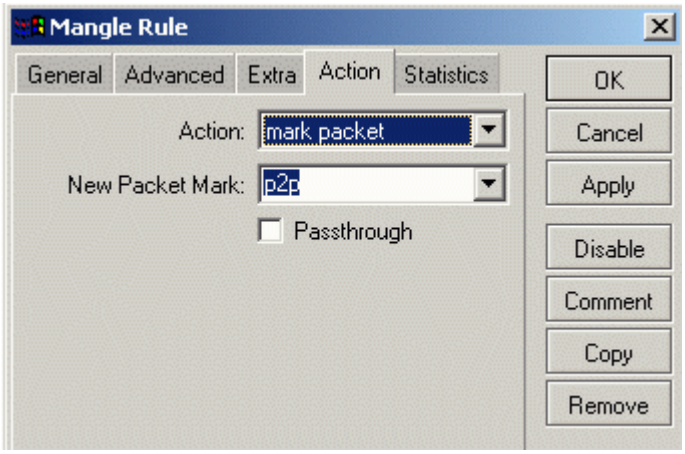


Obr. 5 – vytváření pravidla p2p_conn



Obr. 6 – vytváření pravidla p2p_conn

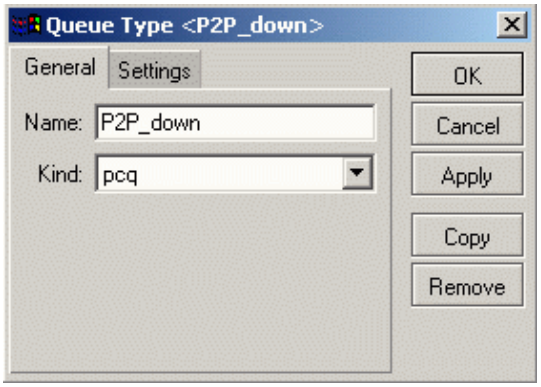
Dále jsem vytvořil pravidlo, které bude vkládat značku pro každý paket. Založení proběhne obdobným způsobem jako u značky pro spojení.



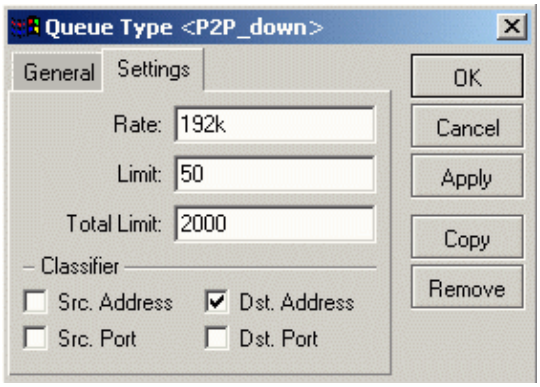
Obr. 7 – značka pro jednotlivé pakety

První část pravidla zůstává shodná (viz obrázek číslo 5), v sekci **ACTION** zvolím **Mark packet** a značku nazvu **p2p**. Funkci **Passthrough** již můžu vypnout, protože s pakety sítí P2P již nebudu provádět žádné označovací práce. Nyní těmito dvěma pravidly kontroluji veškerý tekoucí provoz routerem a v případě, že se objeví známý formát paketu patřící sítím P2P, dostane tento speciální značku. Samozřejmě takové procházení každého tekoucího bajtu klade nemalé požadavky na výkon. Jestliže chci obsluhovat mnoho lidí v rámci jednoho AP či chci na jeden RouterBoard použít více přístupových bodů, musím zajistit adekvátní výpočetní výkon použitého hardware. Použitím RB532 mám dostatečnou rezervu výkonu i při obsluhování více než 100 zákazníků. Samotné označení paketů však nic neřeší.

Dalším krokem bude vytvoření požadovaných filtračních či kontrolních pravidel v rámci mého přístupového bodu. Pro získání kontroly nad tokem dat v P2P sítích se jako nejúčinnější jeví kombinace omezení rychlosti a omezení počtu spojení pro jednoho uživatele. Pro omezení rychlosti použiji specialitu systému MikroTikOS – fronty typu **PCQ**. V sekci **QUEUE List** přidám novou frontu podle následující ukázky.

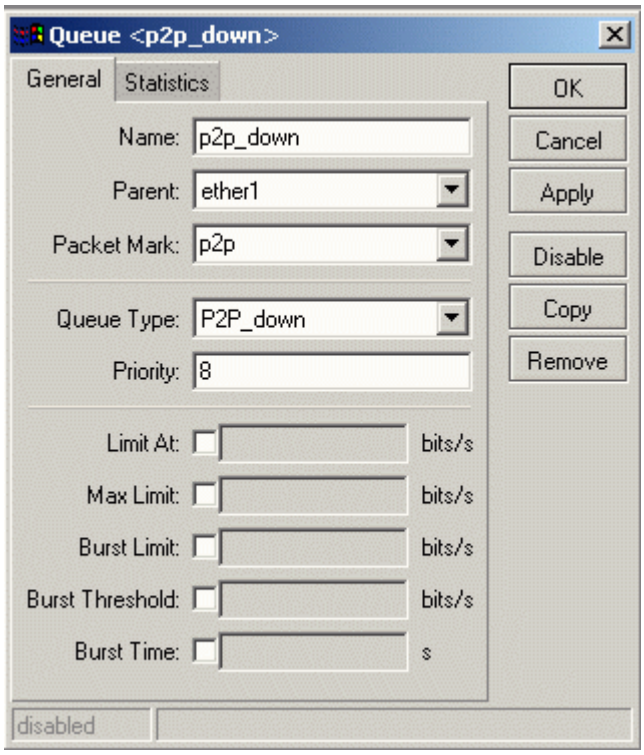


Obr. 8 – definice PCQ – general



Obr. 9 – definice PCQ - settings

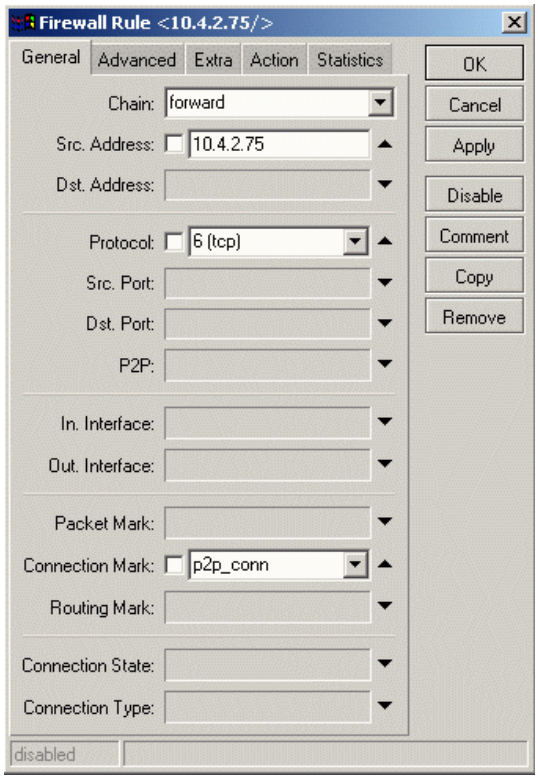
Obrázek číslo osm ukazuje konfiguraci názvu fronty a typu fronty, obrázek číslo devět pak nastavené vlastnosti **PCQ** fronty – nastavil jsem rychlost 192 kbps a klasifikační pravidlo cílovou adresu. Protože na jednotlivých rozhraních vždy lze omezovat pouze odchozí provoz, definuji pro stahování z P2P sítí jako parametr cílovou adresu. To znamená, že všechna data jdoucí na jednu IP adresu jsou seskupována do jedné skupiny. Pro upload do P2P sítí vytvořím obdobnou frontu, budu však definovat klasifikátor jako zdrojovou adresu, protože budu chtít omezovat provoz na výstupním rozhraní a tedy určovat všechen provoz podle jednotlivých zákazníků, tj. dle zdrojových IP adres. Pak už pouze vytvořím reálnou obsluhující frontu v **QUEUE TREE**.



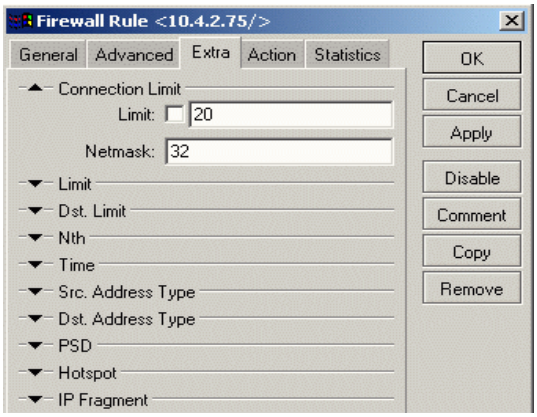
Obr. 10 – vytvoření nové reálné fronty v **QUEUE TREE**

Položka **parent** vyžaduje správné nastavení rozhraní, na kterém bude prováděno omezení rychlosti. Jak jsem již zmiňoval, pro omezení downloadu se bude jednat o rozhraní „vnitřní“ – to, na kterém jsou přímo připojeni klienti. Pro omezení uploadu použiji rozhraní „odchozí“ – tam, kde je připojen zdroj Internetu. **Packet mark** je značka paketu. **Queue type** je typ fronty – vyberu to, co jsem si vytvořil, případně použiji některou z předdefinovaných front. **Priority** definuje s jakou důležitostí se bude tato fronta obsluhovat – stupnice od jedné do osmi, kde jednička je priorita nejvyšší. V mém případě zadaného osmého stupně priority se této frontě přidělí určitý datový tok jen v případě, že nebude vybrán sousedními typy front s vyšší prioritou. Celou stromovou strukturu, jak může například vypadat, ukáži později na obrázku 16.

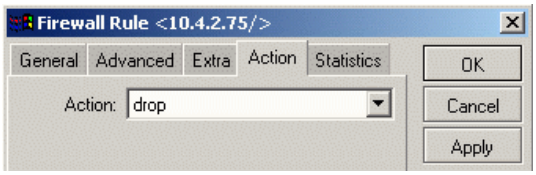
Dalším krokem bude omezení počtu spojení pro P2P datové přenosy. To provedu v sekci **FILTER**. Vytvořím nové pravidlo dle následujícího příkladu.



Obr. 11 – nové pravidlo filtru – general



Obr. 12 – nové pravidlo filtru - extra

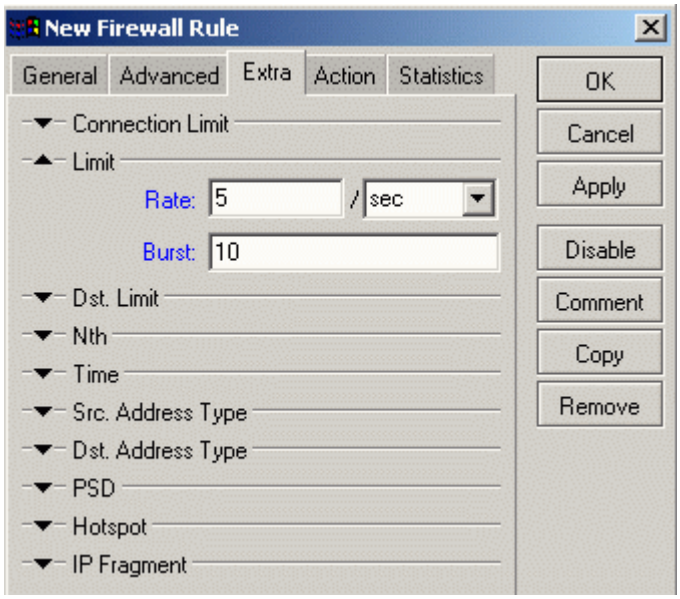


Obr. 13 – nové pravidlo filtru - action

Na kartě **general** vyplním následující položky. **Chain** – místo, ve kterém budu pakety filtrovat – volím řetěz forward, protože v routovaném provozu jdou veškeré pakety od zákazníka do Internetu a opačně právě přes tuto sekci. V **src. address** vyplním IP adresu zákazníka, kterého chci omezit. Typ **protokolu** vyberu TCP, protože pouze u něj lze filtrovat počet „spojení“. Pravidlo použiji pouze na pakety označené *p2p_conn* – **Connection Mark**. Na kartě **extra** si otevřu položku **Connection Limit** a vyplním požadovaný počet spojení, který chci danému zákazníkovi povolit. Nakonec v **action** zvolím **DROP**. Výsledkem je pravidlo, které v případě, že daný klient naváže více spojení než má v **Connection Limit**, vyhodnotí veškeré další pokusy o navázání jako zakázané a příslušné pakety zahodí.

Někteří tvůrci programů pro sdílení dat v P2P sítích šli dál a uvědomili si snadnost omezení počtu spojení v TCP protokolu a rozhodli se zkusit své štěstí s protokolem UDP. To je nespojový protokol, kde nemáte garantovanou kontrolu nad přijetím jednotlivých paketů. Na spolehlivých linkách je protokol UDP stejně výkonný jako TCP. V případě poskytovatele je UDP stejně nebezpečná zbraň jako mnoho otevřených TCP spojení. Jediná možnost, jak trochu zamezit zneužití tohoto protokolu, je propouštění pouze určitého počtu paketů za sekundu. Konstrukce

pravidla zůstane prakticky stejná, pouze v **general** sekci se nahradí TCP protokol hodnotou **UDP**. Karta **extra** se musí samostatně upravit:



Obr. 14 – pravidlo filtru pro omezení toku paketů

Ve výsledku pak bude **FILTER** vypadat přibližně takto:

;;; Drahota										
	drop	forward	10.4.2.50					6 (tcp)	9.5 MiB	37 609
	drop	forward	10.4.2.50				!53	17 (udp)	0 B	0
;;; Drahota torrent port 80 zahodit rovnou										
	drop	forward	10.4.2.50				80	6 (tcp)	127.4 KiB	495
;;; Jedlickova										
	drop	forward	10.4.2.22					6 (tcp)	0 B	0
	drop	forward	10.4.2.22				!53	17 (udp)	7.5 KiB	62
;;; Vavra										
	drop	forward	10.4.2.75					6 (tcp)	17.2 MiB	92 173
	drop	forward	10.4.2.75				!53	17 (udp)	2457.5 KiB	19 461
;;; Novak										
	drop	forward	10.5.3.18					6 (tcp)	39.1 MiB	424 365
	drop	forward	10.5.3.18				!53	17 (udp)	2453.3 KiB	25 013

Obr. 15 – výpis pravidel v sekci FILTER

Pro úplnost uvedu, že při prvotní tvorbě pravidla pro kontrolu toku UDP paketů jsem opomněl, že důležitá součást síťového provozu se odehrává také pomocí tohoto protokolu – jedná se o systém DNS. Proto jsem později do každého z pravidel přidal výjimku, že pakety směřující na port 53 jsou z této filtrace plně vyjmuty. Dalším problémem může být, že většina multiplayerových her používá pro přenosy právě protokol UDP a tímto omezením prakticky můžu hraní znemožnit. Pak už však nezbývá jiná možnost, než se s tímto zákazníkem domluvit na řešení problému.

V rámci přístupového bodu můžu provádět jednoduchým způsobem QOS, a tak „hned u zdroje“ zvyšovat či snižovat prioritu vyřizování jednotlivých požadavků klientů na služby. Většina klientů klade důraz především na rychlost prohlížení stránek, a proto definuji pro celý router pouze několik tříd provozu:

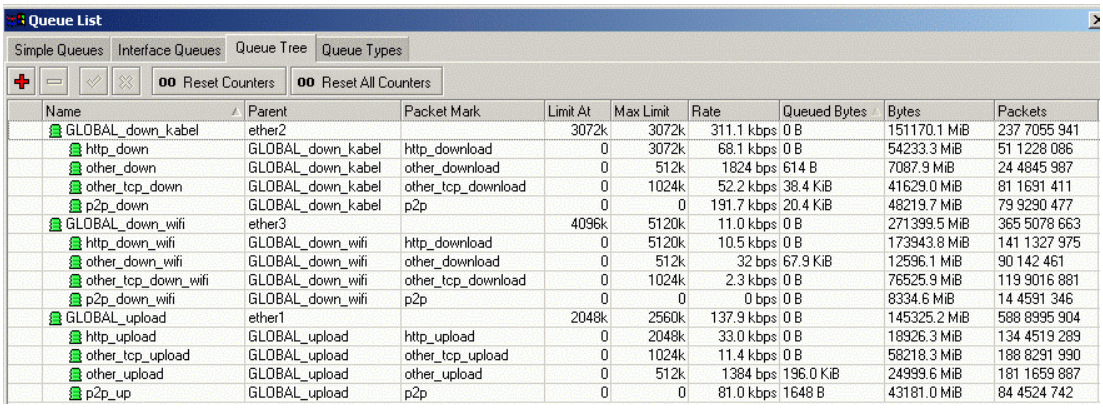
http provoz, *ostatní tcp provoz* a *ostatní provoz*. Pomocí označkování paketů a následnému řazení do jednotlivých front s různou prioritou dochází k čerpání datového pásma způsobem, který vyhovuje většině zákazníků.

accept	forward	10.4.0.0/16						1 (icmp)			58
accept	forward	10.5.0.0/16						1 (icmp)			0
accept	forward	10.20.0.0/16						1 (icmp)			0
accept	forward				10.4.0.0/16			1 (icmp)			10.8 K
accept	forward				10.5.0.0/16			1 (icmp)			803.8 K
accept	forward				10.20.0.0/16			1 (icmp)			295
accept	forward	10.4.0.0/16			0.0.0.0/0	53		17 (udp)			0
accept	forward	10.5.0.0/16			0.0.0.0/0	53		17 (udp)			0
accept	forward	10.20.0.0/16			0.0.0.0/0	53		17 (udp)			0
mark connect...	forward	0.0.0.0/0			0.0.0.0/0					p2p_conn	15319.6 M
mark packet	forward	0.0.0.0/0			0.0.0.0/0				p2p		15319.6 M
mark packet	forward	10.4.0.0/16			0.0.0.0/0	80		6 (tcp)	http_upload		803.1 M
mark packet	forward	10.5.0.0/16			0.0.0.0/0	80		6 (tcp)	http_upload		1211.6 M
mark packet	forward	10.20.0.0/16			0.0.0.0/0	80		6 (tcp)	http_upload		225.8 M
mark packet	forward	0.0.0.0/0	80		10.4.0.0/16			6 (tcp)	http_download		21574.1 M
mark packet	forward	0.0.0.0/0	80		10.5.0.0/16			6 (tcp)	http_download		13628.4 M
mark packet	forward	0.0.0.0/0	80		10.20.0.0/16			6 (tcp)	http_download		2797.2 M
mark packet	forward	0.0.0.0/0			10.4.0.0/16			6 (tcp)	other_tcp_download		1388.6 M
mark packet	forward	0.0.0.0/0			10.5.0.0/16			6 (tcp)	other_tcp_download		7039.4 M
mark packet	forward	0.0.0.0/0			10.20.0.0/16			6 (tcp)	other_tcp_download		314.7 M
mark packet	forward	10.4.0.0/16			0.0.0.0/0			6 (tcp)	other_tcp_upload		325.8 M
mark packet	forward	10.5.0.0/16			0.0.0.0/0			6 (tcp)	other_tcp_upload		1439.1 M
mark packet	forward	10.20.0.0/16			0.0.0.0/0			6 (tcp)	other_tcp_upload		92.1 M
mark packet	forward	0.0.0.0/0			10.4.0.0/16				other_download		504.4 M
mark packet	forward	0.0.0.0/0			10.5.0.0/16				other_download		886.7 M
mark packet	forward	0.0.0.0/0			10.20.0.0/16				other_download		366.5 M
mark packet	forward	10.4.0.0/16			0.0.0.0/0				other_upload		511.6 M
mark packet	forward	10.5.0.0/16			0.0.0.0/0				other_upload		867.8 M
mark packet	forward	10.20.0.0/16			0.0.0.0/0				other_upload		315.4 M

Obr. 15 – celá mangle tabulka inteligentního přístupového bodu

Pro dokonalé pochopení obrázku vysvětlím, co znamenají jednotlivé řádky. Prvních devět řádků priorizuje prostupnost protokolu ICMP a paketů služby DNS. Definicí TOS (Type Of Service) vlajky na hodnotu *16 – min delay* se routeru přikáže, aby tyto pakety bezodkladně odesílal s nejvyšší prioritou a nejmenším zpožděním. Další dva řádky jsou známé – tvořil jsem je při klasifikaci paketů P2P služeb. Všechny ostatní záznamy jsou pro značkování jednotlivých typů paketů. Vždy po trojicích – **upload http**, **download http**, **download ostatní tcp** (vše TCP mimo http provozu na portu 80), **upload ostatní tcp**, **download ostatní** (všechny další protokoly – př. UDP), **upload ostatní**.

Dalším demonstračním obrázkem je ukázka uspořádání stromové struktury jednotlivě vytvořených front.



Name	Parent	Packet Mark	Limit At	Max Limit	Rate	Queued Bytes	Bytes	Packets
GLOBAL_down_kabel	ether2		3072k	3072k	311.1 kbps	0 B	151170.1 MiB	237 7055 941
http_down	GLOBAL_down_kabel	http_download	0	3072k	68.1 kbps	0 B	54233.3 MiB	51 1228 086
other_down	GLOBAL_down_kabel	other_download	0	512k	1824 bps	614 B	7087.9 MiB	24 4845 987
other_tcp_down	GLOBAL_down_kabel	other_tcp_download	0	1024k	52.2 kbps	38.4 KiB	41629.0 MiB	81 1691 411
p2p_down	GLOBAL_down_kabel	p2p	0	0	191.7 kbps	20.4 KiB	48219.7 MiB	79 9290 477
GLOBAL_down_wifi	ether3		4096k	5120k	11.0 kbps	0 B	271399.5 MiB	365 5078 663
http_down_wifi	GLOBAL_down_wifi	http_download	0	5120k	10.5 kbps	0 B	173943.8 MiB	141 1327 975
other_down_wifi	GLOBAL_down_wifi	other_download	0	512k	32 bps	67.9 KiB	12596.1 MiB	90 142 461
other_tcp_down_wifi	GLOBAL_down_wifi	other_tcp_download	0	1024k	2.3 kbps	0 B	76525.9 MiB	119 9016 881
p2p_down_wifi	GLOBAL_down_wifi	p2p	0	0	0 bps	0 B	8334.6 MiB	14 4591 346
GLOBAL_upload	ether1		2048k	2560k	137.9 kbps	0 B	145325.2 MiB	588 8995 904
http_upload	GLOBAL_upload	http_upload	0	2048k	33.0 kbps	0 B	18926.3 MiB	134 4519 289
other_tcp_upload	GLOBAL_upload	other_tcp_upload	0	1024k	11.4 kbps	0 B	58218.3 MiB	188 8291 990
other_upload	GLOBAL_upload	other_upload	0	512k	1384 bps	196.0 KiB	24999.6 MiB	181 1659 887
p2p_up	GLOBAL_upload	p2p	0	0	81.0 kbps	1648 B	43181.0 MiB	84 4524 742

Obr. 16 – ukázka stromového uspořádání datových front

V těchto praktických ukázkách jsem použil RouterBoard, který byl umístěn v panelovém domě. Obsluhoval klienty jak přes bezdrátové rozhraní, tak přes klasický metalický rozvod, proto ve stromové architektuře existují globální fronty jak pro **WIFI** část, tak pro **KABEL** část.

Tato struktura má účel pouze priorizační a speciálně pro P2P datové toky účel limitování rychlosti. Podíváme-li se na strukturu dat, která protečou, zjistíme, že pouhá třetina dat u zákazníků připojených kabelem připadá na prohlížení WWW stránek a s tím spojené přenosy souborů. Druhá třetina dat jsou identifikované P2P toky a poslední třetina připadá na ostatní tcp downloady. Tyto ostatní tcp přenosy jsou však ve velké míře právě nerozpoznané P2P protokoly. U zákazníků připojených bezdrátově je skladba výrazně odlišná, je to způsobeno zřejmě tím, že službu v tomto místě využívají víceméně běžní uživatelé Internetu. Upload však odpovídá spíše skupině hojně využívající P2P sítí.

I tato drobná ukázka z malé části sítě částečně podporuje mojí odvážnou tezi, že alespoň nadpoloviční většina datových toků dnešního Internetu je způsobena využíváním povětšinou nelegálních P2P sítí.

4.2 MikroTikOS jako centrální výchozí brána sítě

MikroTikOS, jako systém mnoha možností, může být bez problémů využit i jako centrální výchozí brána celé sítě. Precizním výběrem síťových funkcí tvůrci dokázali vytvořit příjemnou alternativu jak OpenSource Linuxu – tedy softwaru poskytovaného zcela zdarma, tak také zařízení, jejichž pořízení je otázkou i několika set tisíc. Největší slabinou systému je nižší výkon. Na MikroTiku rozhodně nelze postavit výchozí bránu velké rozlehlé sítě s mnoha tisíci zákazníky. I když postavíme na dnešní dobu hodně nadprůměrně silný počítač, začneme mít při vyšších datových tocích značné problémy. Kombinace NATu, značkování paketů, klasifikace paketů podle značek, řazení do jednotlivých front a jiné funkce klade obrovské požadavky na výpočetní výkon, který začne brzy chybět. V síti, kde bylo zhruba 600 klientů, byla nasazena brána s procesorem P4 2.4 GHz. V odpoledních a večerních špičkách již začínal být znát nedostatek výpočetního výkonu, který se projevoval nepravidelným obsluhováním těch nejvíce priorizovaných služeb a datových toků. Pro menší poskytovatele je řešení pomocí MikroTikOS celkem ideální, neklade tak velké požadavky na znalosti složitých komplexních systémů, jako je Linux či například specializovaných zařízení CISCO a další.

Účelem výchozí brány je zajistit správnou distribuci datových spojení v celé síti, potřebuji tedy přehledné a hlavně dobře fungující routování, dále potřebuji přehledně a pokud možno co nejjednodušeji klasifikovat pakety podle jejich značek, nesmí chybět funkční omezování rychlostí nebo dělení šířky pásma mezi jednotlivé uživatele. V předchozí kapitole jsem podrobně ukázal, jak se dají označovat jednotlivá spojení a pakety, částečně jsem popsal, jak vytvářet fronty. Abychom získali podrobnější pohled do této problematiky, rozeberu v následujících kapitolách princip a funkci QOS priorizace služeb. V poslední kapitole této praktické části se budu zabývat trochu podrobněji komplexním systémem omezování rychlostí na výchozí bráně celé sítě.

4.3 QOS – priorizace služeb

Priorizaci služeb jsem již začal v kapitole o inteligentních přístupových bodech. Základním principem je především označkování jednotlivých služeb pomocí **packet** nebo **connection mark**. Vedlejší možností je použití příznaku **TOS**, který však nezaručuje výsledky, jež bych předpokládal v případě, že dojde k nadměrnému využití kapacity Internetové linky. Pomocí **QOS** můžu předejít mnoha problémům, jako je například úplné zahlcení linky daty z P2P sítí. Na výchozí bráně budu také klasifikovat pakety P2P – prakticky dělám to stejné, co jsem použil již na inteligentních přístupových bodech. Avšak ne všude mám tyto lepší AP a hlavně ne všude na nich přímo omezuji rychlost, někde značení P2P použiji pouze pro účely udělení priority v **QUEUE TREE**. Z datových limitů dále vyjímám veškeré ICMP pakety – pro ně využívám klasifikaci **TOS 16** (min delay). Centrálně pro všechny klienty pak značkuji pouze http provoz, tj. vše, co přichází ze vzdálených serverů z portu 80 TCP.

Dále podle požadavků můžu značkovat datová spojení, která klient potřebuje v nejvyšší kvalitě. Filosofie priorizování různých datových toků pro jednotlivé klienty je z principu „špatná“ a hlavně v určitých případech zcela nemožná. „Špatná“ je z důvodu toho, že každý zákazník může mít jinou představu. Nemožná je z důvodu, že většina prodaných „linek“ jsou sdílené a agregované, na kterých by se pravidlo upřednostňující jeden druh spojení neprojevilo vůbec a nebo bych musel zcela předělat systém „sdílení“ linky, a tak ho několikanásobně zkomplikoval.

;;; Sechovec Libor - 21.9.2005								
passthrough	10.5.2.22/32	all	0.0.0.0/0	all		Sechovec Libor_conn	14745606	212587
;;; Sechovec Libor								
passthrough	10.5.2.21/32	all	0.0.0.0/0	all		Sechovec Libor_conn	355501	2703
accept	0.0.0.0/0	all	0.0.0.0/0	all	schneider		421289007	517463
;;; Vrabcová Iveta - 17.9.2005								
passthrough	10.7.5.4/32	all	0.0.0.0/0	all		Vrabec Peter_conn	35789237	678579
accept	0.0.0.0/0	all	0.0.0.0/0	all	jih_firmy		1702893412	1813990
;;; Baštýř Ladislav - 1.8.2005								
passthrough	10.5.2.24/32	all	0.0.0.0/0	all		Bastyr Ladislav_conn	2859735	8070
;;; Baštýř Ladislav								
passthrough	10.5.2.25/32	all	0.0.0.0/0	all		Bastyr Ladislav_conn	0	0
accept	0.0.0.0/0	all	0.0.0.0/0	all	schneider		10572282	17401
;;; Taraba Jan - 1.9.2005								
passthrough	10.1.1.213/32	all	0.0.0.0/0	all		Taraba Jan_conn	0	0
accept	0.0.0.0/0:80	all	0.0.0.0/0	tcp	group2_http		0	0
accept	0.0.0.0/0	all	0.0.0.0/0	all	group2		0	0
;;; Sperl Martin								
passthrough	10.10.30.2/32	all	0.0.0.0/0	all		Sperl Martin_conn	9621215	56089
accept	0.0.0.0/0:80	all	0.0.0.0/0	tcp	group2_http		0	0
accept	0.0.0.0/0	all	0.0.0.0/0	all	group2		65249287	120159

Obr. 17 – ukázka z **mangle** tabulky na výchozí bráně

Protože se jedná o bránu, na které se provádí NAT, specifikace MikroTikOS říká, že pro označení všech paketů se musí nejprve provést označení spojení a teprve na základě značky pro spojení se označí všechny pakety. U routerů, na kterých se neprovádí překlad adres, je možné označovat veškeré pakety rovnou pomocí jediného pravidla. V **mangle** na výchozí bráně označuji datové toky na základě zdrojových IP adres a provádím klasifikaci do dvou zcela odlišných druhů. Prvním jsou klasifikace paketů, u kterých se provádí omezení rychlosti již někde uvnitř sítě (například sběrná místa na páteři), a druhým způsobem je označování paketů do skupin nazvaných *groupX_http* a *groupX*, tento způsob označení se pak použije pro omezování rychlostí. Na následujícím obrázku je možné vidět, jak se uložené značky spojení ukazují u jednotlivých spojení.

Firewall

Filter Rules | NAT | Mangle | Service Ports | Connections | Address Lists

Tracking

	Src. Address	Dst. Address	Proto...	Connecti...	Connection Mark	P2P	TCP State	Timeout
A	10.13.11.4:2603	212.80.76.201:80	6 (tcp)		(none)		time wait	00:00:01
A	10.13.11.4:2604	194.228.32.21:80	6 (tcp)		(none)		time wait	00:00:02
A	62.4.97.118:4207	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	established	00:29:55
A	62.4.97.118:4212	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	established	00:29:57
A	62.44.18.42:35361	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	established	00:29:57
A	62.44.18.42:35470	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	established	00:29:57
A	62.77.85.143:4197	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	established	00:29:56
A	62.77.119.62:1976	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	established	00:29:56
A	62.168.45.90:1748	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	established	00:29:56
A	62.209.207.252:4049	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	established	00:29:55
A	62.240.165.134:35904	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	close wait	00:00:07
A	62.240.172.45:4848	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	time wait	00:00:01
A	62.240.182.35:3173	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	time wait	00:00:06
A	62.240.183.254:1346	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	established	00:29:57
A	62.240.190.65:1310	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	established	00:29:57
A	62.241.244.93:3301	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	time wait	00:00:02
A	68.161.183.17:50871	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	time wait	00:00:01
	69.158.106.138:1453	10.13.1.4:25050	6 (tcp)		(none)		close	00:20:04
A	69.158.106.138:3414	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	established	00:29:40
	69.158.106.138:4666	10.13.1.4:25050	6 (tcp)		(none)		close	00:02:22
A	80.55.222.34:3943	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	established	00:29:53
A	80.78.154.156:3527	10.13.1.4:25050	6 (tcp)		p2p_conn	direct conn...	established	00:29:56
	80.87.183.24:1695	10.13.1.4:25050	6 (tcp)		(none)		syn recei...	00:00:01
	80.95.245.86:4381	10.13.1.4:25050	6 (tcp)		(none)		time wait	00:00:01
A	81.2.213.2:1117	10.1.1.132:8291	6 (tcp)		(none)		established	00:29:57

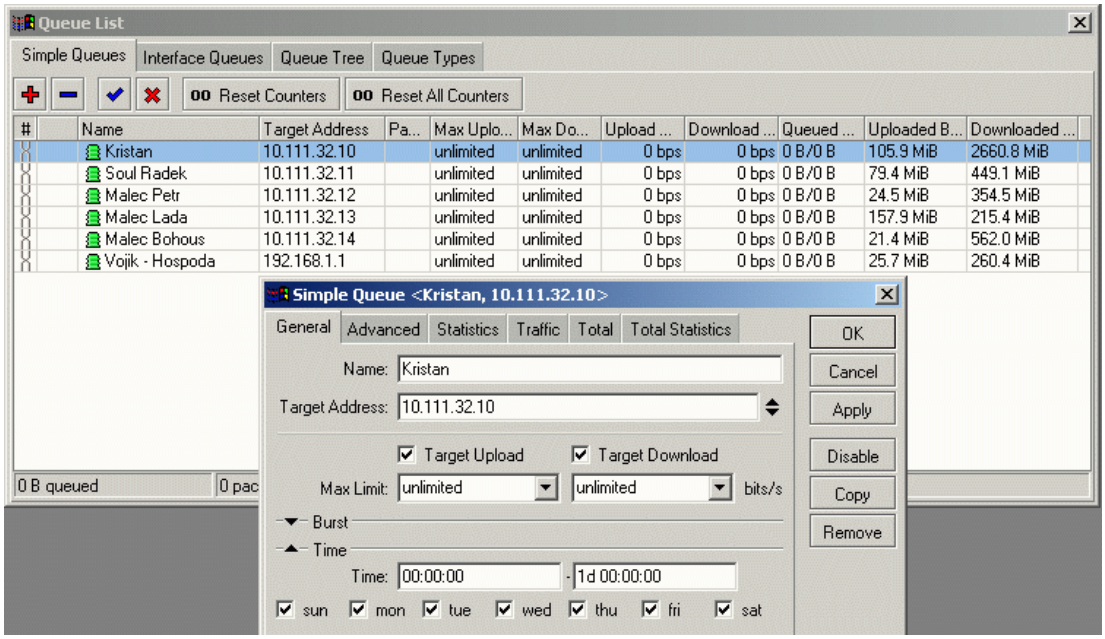
Obr. 18 – Firewall – Connections – ukázka přehledu spojení včetně značek

V **Connection Mark** je vidět označení spojení – u spojení, které není ve stavu *established se* značka nezobrazuje. Hodnota **timeout** značí, jak dlouho bude ještě router držet v paměti parametry tohoto spojení při jeho neaktivitě (zdrojovou, cílovou adresu, port, protokol, aj.), tuto hodnotu je možné změnit v parametrech **Tracking** (tlačítko nahoře vlevo). Výchozí hodnota timeoutu pro established spojení je jeden den, to je zbytečně veliká hodnota, doporučuji ji snížit.

4.4 Reálný management datových toků

V poslední kapitole této části se podíváme, jakým způsobem můžeme omezovat rychlosti jednotlivým uživatelům, případně skupinám uživatelů.

Pro garantované rychlosti existuje v MikroTikOS možnost použít **simple queues**, kde v definici jednoduché fronty pouze vyplním cílovou IP adresu (případně několik IP adres), na kterou se bude vztahovat omezení, a celkovou poskytovanou rychlost. Tato rychlost bude vždy garantovaná v případě, že bude dostatečná kapacita zdrojové linky.



Obr. 19 – Simple Queues

Zde nemají klienti nastavené žádné omezení rychlosti. Definici v **simple queues** v tomto případě využívám pouze pro automatické generování grafů provozu, které MikroTikOS podporuje v jednoduchých frontách. Místo *unlimited* hodnot je možné nastavit požadované rychlosti pro stahování a odesílání dat. Pro zajímavost jsem rozbalil i konfigurační sekci časového nastavení omezení. Přehledně a jednoduše se dá nadefinovat například noční provoz s neomezeným tokem dat. Zjednodušeným způsobem by se pomocí **simple queues** dali definovat i sdílené linky (přidáním více cílových IP adres), ale tuto možnost jsem reálně nikdy nezkoušel a nevím, jakým způsobem by docházelo k dělení datového toku mezi jednotlivé uživatele jednoduché fronty.

Garantovaných linek se prodává minimum, nejvíce se prodá sdílených připojení, které se definují pomocí **Queue Tree**. Pro definici agregovaných linek jsem používal v zásadě dva principy. Buď se určitému agregovanému počtu lidí přiřadí stejný **packet mark** a vytvoří se pro ně jedna fronta v **queue tree** a nebo se využije vlastností PCQ fronty v kombinaci se stejným **packet mark**. Obě možnosti proberu podrobně.

Na obrázku 17 jsem v **mangle** dal každému paketu určitou značku. Zaměřím se na značky *groupX_http* a *group*. V případě jednoduchého systému sdílení linky mezi určitý počet lidí nastavím celé jedné skupině zákazníků jednu značku. Pak v **Queue Tree** definuji reálné fronty pro jednotlivé značky paketů. Na následujícím obrázku si můžeme danou situaci prohlédnout.

Queue List

Interface Queues | Queue Tree | Simple Queues | Queue Types

00 Reset Counters

Name	Parent	Flow	Limit At	Max Limit	Bps	Bytes	Packets
GLOBAL	ether8		15360000	15728640	8.9 Mbps	179027751	724516368
PCQ_BUDOVATELSKA	GLOBAL	budovatelska	0	6144000	942.6 kbps	1412409088	53240136
PCQ_BUDOVATELSKA_FIRMY	GLOBAL	budovatelska_firmy	0	6144000	286.4 kbps	1213869141	15439477
PCQ_DUKLA	GLOBAL	dukla	0	6144000	1616.3 kbps	413177695	46591777
PCQ_DUKLA_1024	GLOBAL	dukla_mego	0	6144000	96.1 kbps	2513993113	3069144
PCQ_DUKLA_2048	GLOBAL	dukla_2mega	0	2048000	0 bps	3985157007	10043097
PCQ_SCHNEIDER	GLOBAL	schneider	0	6144000	2.1 Mbps	2781934992	89096720
PCQ_schneider_firmy	GLOBAL	schneider_firmy	0	6144000	66.3 kbps	2943444503	21596510
VIP	GLOBAL	vip	15360000	15728640	4.7 kbps	2311865796	8732773
firmy_1	GLOBAL		512000	1024000	686 bps	1924522989	2082788
firmy_1_http	firmy_1	firmy_1_http	768000	1024000	0 bps	0	0
firmy_1_other	firmy_1	firmy_1	256000	1024000	686 bps	1924493273	2082788
firmy_10	GLOBAL		512000	1024000	0 bps	2211104890	2933878
firmy_10_http	firmy_10	firmy_10_http	768000	1024000	0 bps	0	0
firmy_10_other	firmy_10	firmy_10	256000	1024000	0 bps	2211111721	2933877
firmy_2	GLOBAL		512000	1024000	8.8 kbps	702204738	8919480
firmy_2_http	firmy_2	firmy_2_http	768000	1024000	0 bps	0	0
firmy_2_other	firmy_2	firmy_2	256000	1024000	8.8 kbps	702204729	8919480

Obr. 20 – **Queue Tree** – definice reálných front

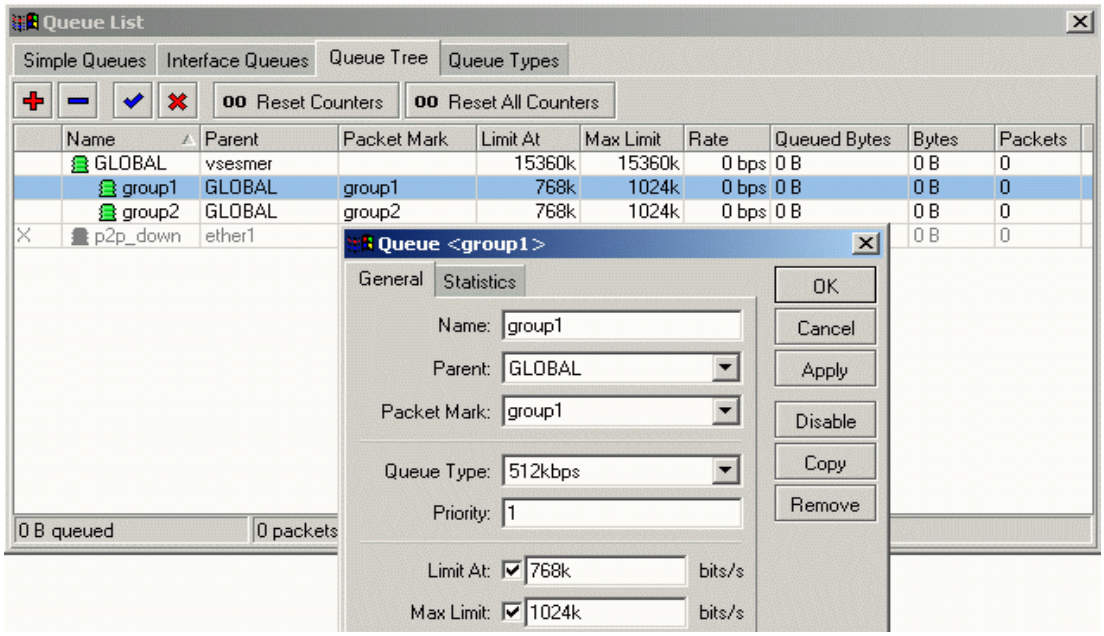
Na obrázku 20 je zobrazena část stromu pro download. První fronta *GLOBAL* je pověšená přímo na rozhraní klientů, a tak může omezovat pouze odchozí tok dat z tohoto rozhraní, což znamená download klientů. Podřízené fronty začínající textem *PCQ_* jsou fronty, které omezují klienty řízené routery na sběrných místech páteře. Principielně je datový tok těchto klientů řízen PCQ frontou s určitou velikostí pro každého zákazníka. K agregaci klientů zde prakticky nedochází. Jedinou výjimkou je, když souhrn toku takto zařazených uživatelů je vyšší než stanovený **Max Limit**. Poté dochází k úměrnému snížení maximální dosažené rychlosti pro všechny právě aktivní uživatele na této větvi. Při půlmegabitových linkách je však dost malá pravděpodobnost, že právě více než 12 lidí bude naplno využívat svou kapacitu. V případě, že by pak začalo vyhrazenou kapacitu 6 Mbit čerpat třeba

15 lidí, dojde k férovému dělení mezi jednotlivé uživatele, každý bude mít místo 512 kbps k dispozici jen 410 kbps.

Fronta *firmy_1* je ukázkou prvního způsobu agregace, obsahuje podfronty *firmy_1_http* a *firmy_1_other*. Firemním klientům je poskytována negarantovaná linka s maximální dosažitelnou rychlostí 1024 kbps. Podfronta *firmy_1_http* řídí tok z web serverů na portu TCP 80 a podfronta *firmy_1_other* řídí veškerý ostatní provoz klientů se značkou paketů *firmy_1*. Hodnota parametru **Limit At** udává minimální garantovanou rychlost. To znamená, při dvou souběžných downloadech, jeden z portu 80 (webové stránky) a druhý např. z FTP serveru, že stahování poběží z web serveru rychlostí 768 kbps, zatímco z FTP serveru pouze 256 kbps. Dále je typ fronty nastaven na SFQ (Stochastic Fairness Queuing), což zaručuje alespoň určitou míru férového dělení datového toku mezi jednotlivé klienty, ovšem za cenu většího zatížení procesoru.

Tento způsob tvorby sdílených linek je výhodnější pro poskytovatele internetu, pevně dodržuje stanovenou agregaci počtem klientů v jedné skupině. Mám-li však dostatek konektivity a volného výpočetního výkonu, můžu upravit agregaci na výhodnější pro klienty.

Druhá metoda neřeší priorizaci webového provozu, ale pracuje s daty jako s celkem (již ochuzeným o datový tok p2p sítí, který se omezuje jednou globální PCQ frontou).



Obr. 21 – agregace klientů pomocí PCQ front

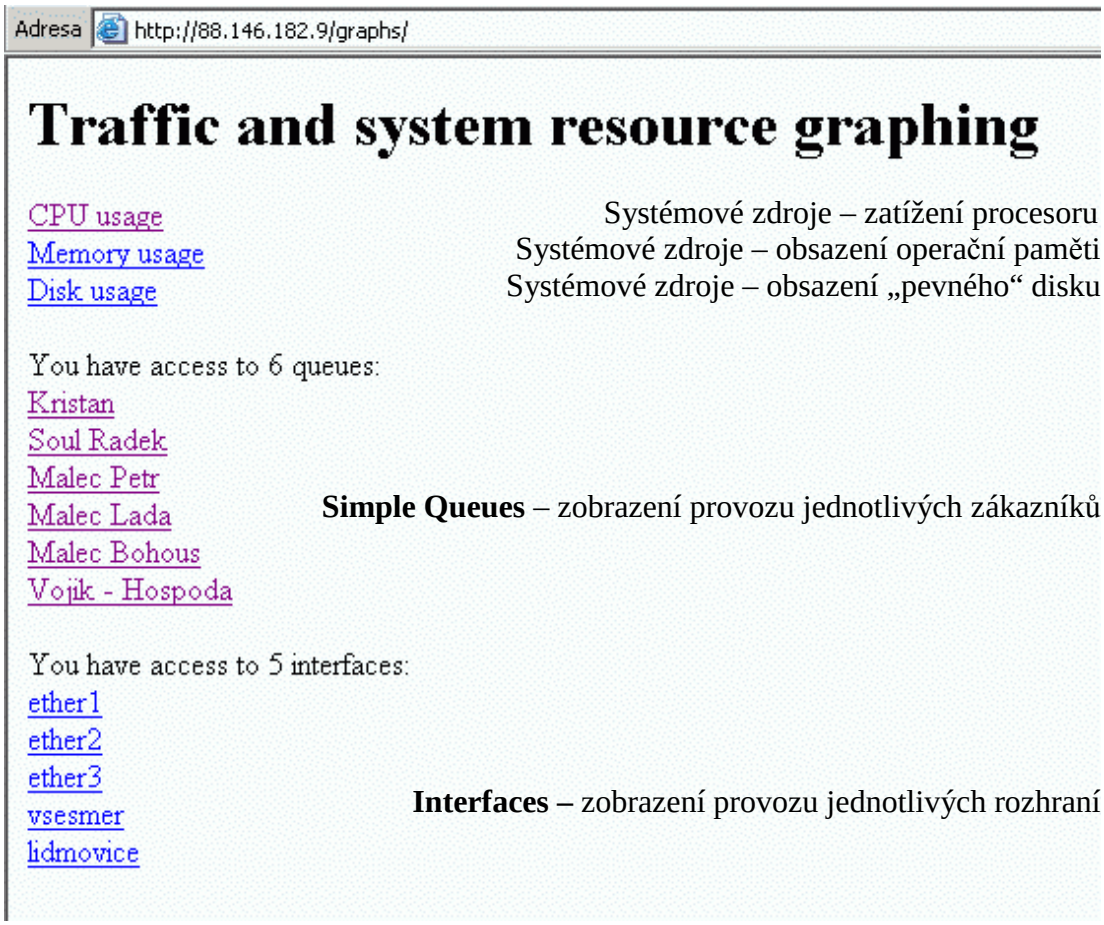
Základem je opět definice globální fronty pověšené přímo na rozhraní, na které se připojují klienti. V případě, že již mám v **mangle** dané označování paketů a nechci na něm nic měnit, přistoupím přímo k nastavení jednotlivých front. Vytvořím podřízenou frontu *group1* – typ fronty vyberu mnou nadefinovanou PCQ frontu – *512kbps*. **Limit-at** nastavím například na 768kbps a **Max Limit** na 1024kbps. Výsledkem bude tato situace: první klient začne plnou rychlostí stahovat data. Pokud se přidá druhý klient ze stejné skupiny, v prvním případě by se již rychlost oběma krátila cca na polovinu. V tomto případě, je-li dostatek konektivity (to si hlídá *GLOBAL* fronta), dostane i druhý klient plnou rychlost stahování. Skupina *group1* je nyní plně vytížená, ale stahují již dva klienti. V případě, že fronta *GLOBAL* nemá již volnou konektivitu k dosažení **Max Limit**, poskytne nám pouze zaručenou rychlost v parametru **Limit At**. Přidá-li se klient třetí, bude se router snažit férově podělit tyto tři klienty v rámci PCQ skupin, tj. každý z nich by měl dostat přiděleno zhruba 340kbps. Férovost tohoto dělení bude také značně záležet na tom, kolik který klient má otevřeno spojení. Pokud mají první dva klienti každý pouze jedno spojení a třetí klient použije nějaký stahovací program, který navazuje pro jeden download více spojení, je větší pravděpodobnost, že klient s vícero spojeními získá i větší šířku datového pásma. Rozdíl by však neměl být nějak závratný.

Způsobů jak omezovat rychlosti klientů nebo jak řešit spravedlivé dělení pásma existuje velké množství. Tyto tři popsané jsou jen zlomkem možností, které MikroTikOS nabízí. Je víceméně na fantazii každého z poskytovatelů, jak bude tuto problematiku řešit. Pro sofistikované způsoby FUP (Fair User Policy) se nabízí například i použití skriptů. Tyto další možnosti jsou nad rámec mé práce.

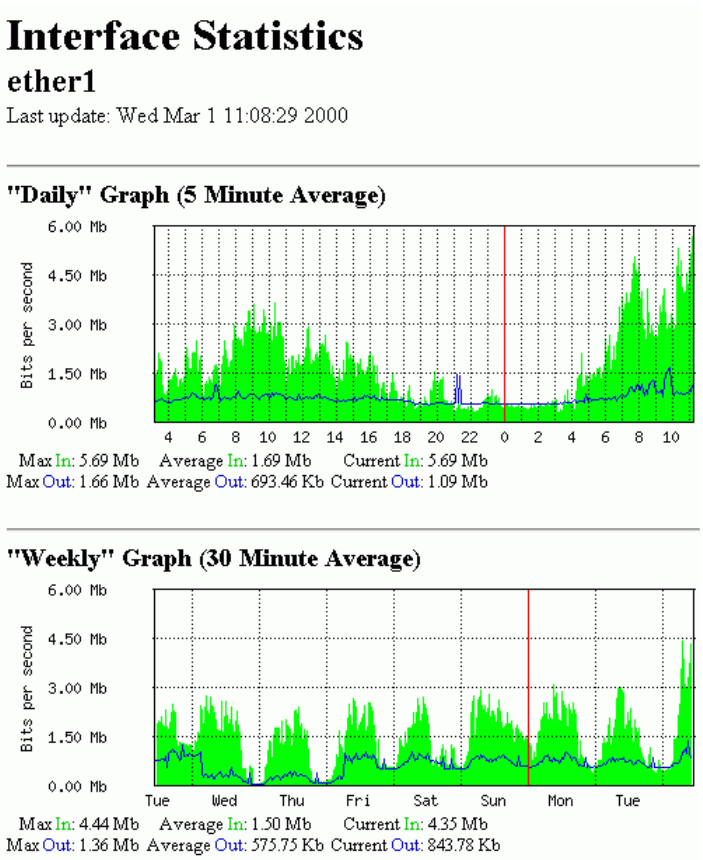
5 Kontrolní mechanismy – grafické zobrazení provozu a funkčnosti sítě

5.1 MRTG – grafické výstupy reálného provozu sítě

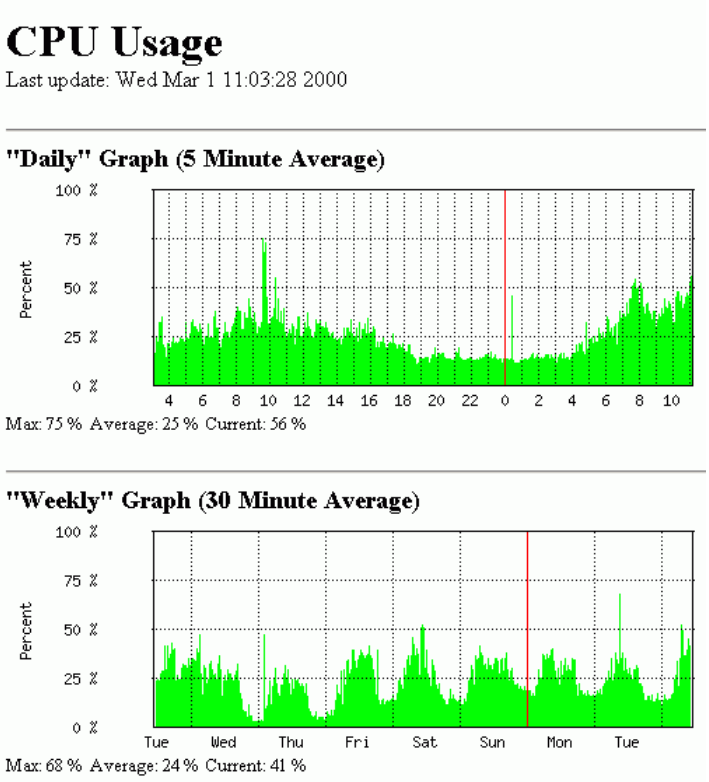
Pro kontrolu chování jednotlivých částí sítě je třeba mít přehled o datových tocích, které danou větví prochází. Pro účely zobrazování datových toků existuje software napsaný v programovacím jazyce Perl, který pracuje jak v Linuxu/Unixu, tak i v prostředí Windows či Netware. Veškeré informace potřebné pro kreslení grafů získává program ze SNMP serveru. Ve verzi 2.9.x již umí MikroTikOS grafy tvořit sám – jedná se o **simple queues, interfaces** a **system resources**.



Obr. 22 – Přehled **Graphing** sekce



Obr. 23 – **Graphing** – denní a týdenní graf provozu rozhraní ETHER1



Obr. 24 – **Graphing** – denní a týdenní graf zatížení procesoru

Stejné grafy mám k dispozici pro každého klienta, který má založenou **Simple Queue**. V případě, že potřebuji zobrazovat provoz stromové struktury, musím použít **MRTG**. Informace o SNMP třídě získám z textového výpisu, nejdříve zjistím ID jednotlivých front a poté SNMP třídy požadovaných informací.

```
[admin@MikroTik] > queue
[admin@MikroTik] queue> tree
[admin@MikroTik] queue tree> print
Flags: X - disabled, I - invalid
 0   name="p2p_down" parent=ether1 packet-mark=p2p limit-at=0 queue=P2P_down priority=8 max-limit=0 burst-limit=0
     burst-threshold=0 burst-time=0s

 1   name="GLOBAL" parent=vsesmer packet-mark="" limit-at=0 queue=default priority=1 max-limit=0 burst-limit=0
     burst-threshold=0 burst-time=0s

 2   name="group9" parent=GLOBAL packet-mark="" limit-at=0 queue=default priority=8 max-limit=0 burst-limit=0
     burst-threshold=0 burst-time=0s

 3   name="group2" parent=GLOBAL packet-mark="" limit-at=0 queue=default priority=8 max-limit=0 burst-limit=0
     burst-threshold=0 burst-time=0s

 4   name="group3" parent=GLOBAL packet-mark="" limit-at=0 queue=default priority=8 max-limit=0 burst-limit=0
     burst-threshold=0 burst-time=0s

 5   name="group4" parent=GLOBAL packet-mark="" limit-at=0 queue=default priority=8 max-limit=0 burst-limit=0
     burst-threshold=0 burst-time=0s
```

Obr. 25 – Zjištění ID jednotlivých front v textovém terminálu

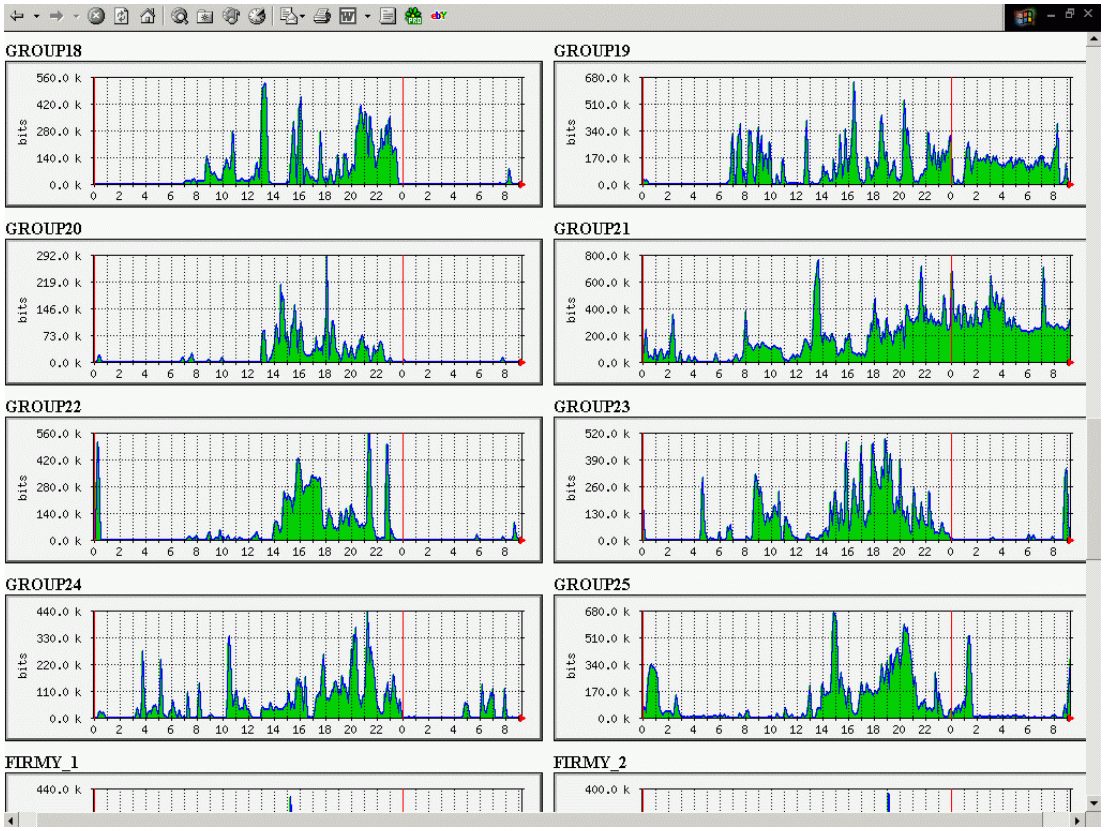
```
[admin@MikroTik] queue tree> print oid
Flags: X - disabled, I - invalid
 0   name=.1.3.6.1.4.1.14988.1.1.2.2.1.2.16777234 packet-mark=.1.3.6.1.4.1.14988.1.1.2.2.1.3.16777234
     bytes=.1.3.6.1.4.1.14988.1.1.2.2.1.5.16777234 packets=.1.3.6.1.4.1.14988.1.1.2.2.1.6.16777234

 1   name=.1.3.6.1.4.1.14988.1.1.2.2.1.2.16777238 packet-mark=.1.3.6.1.4.1.14988.1.1.2.2.1.3.16777238
     bytes=.1.3.6.1.4.1.14988.1.1.2.2.1.5.16777238 packets=.1.3.6.1.4.1.14988.1.1.2.2.1.6.16777238

 2   name=.1.3.6.1.4.1.14988.1.1.2.2.1.2.16777239 packet-mark=.1.3.6.1.4.1.14988.1.1.2.2.1.3.16777239
     bytes=.1.3.6.1.4.1.14988.1.1.2.2.1.5.16777239 packets=.1.3.6.1.4.1.14988.1.1.2.2.1.6.16777239

 3   name=.1.3.6.1.4.1.14988.1.1.2.2.1.2.16777240 packet-mark=.1.3.6.1.4.1.14988.1.1.2.2.1.3.16777240
     bytes=.1.3.6.1.4.1.14988.1.1.2.2.1.5.16777240 packets=.1.3.6.1.4.1.14988.1.1.2.2.1.6.16777240

 4   name=.1.3.6.1.4.1.14988.1.1.2.2.1.2.16777241 packet-mark=.1.3.6.1.4.1.14988.1.1.2.2.1.3.16777241
     bytes=.1.3.6.1.4.1.14988.1.1.2.2.1.5.16777241 packets=.1.3.6.1.4.1.14988.1.1.2.2.1.6.16777241

 5   name=.1.3.6.1.4.1.14988.1.1.2.2.1.2.16777242 packet-mark=.1.3.6.1.4.1.14988.1.1.2.2.1.3.16777242
     bytes=.1.3.6.1.4.1.14988.1.1.2.2.1.5.16777242 packets=.1.3.6.1.4.1.14988.1.1.2.2.1.6.16777242
```

Obr. 26 – Zjištění class jednotlivých front v textovém terminálu

Na obrázku 25 zcela vlevo vidíme ID definovaných front – například fronta *group2* má číslo *3*. Zadám-li v konzoli příkaz *print oid*, vypíší se mi dostupné třídy (class) pro každé známé ID. Pokud chci pomocí **mrtg** kreslit graf pro datový tok *group2*, zadám do konfiguračního souboru celý řetězec za **bytes=** bez tečky na začátku.

```
Target[10.7.0.3_2]:1.3.6.1.4.1.14988.1.1.2.2.1.5.16777240&1.3.6.1.4.
1.14988.1.1.2.2.1.5.16777240:public@10.7.0.3
MaxBytes[10.7.0.3_2]:   64000
Title[10.7.0.3_2]:      group2
PageTop[10.7.0.3_2]:    <H1>group2</H1>
<TABLE><TR><TD>System:</TD><TD>Mikrotik</TD></TR><TR><TD>Description
:</TD><TD>group2</TD></TR></TABLE>
```

Takto bude vypadat část konfiguračního souboru **mrtg.cfg**. Výsledkem celého zobrazení **Queue Tree** pak může být třeba takovýto přehled:



Obr. 27 – **MRTG** grafy jednotlivých skupin

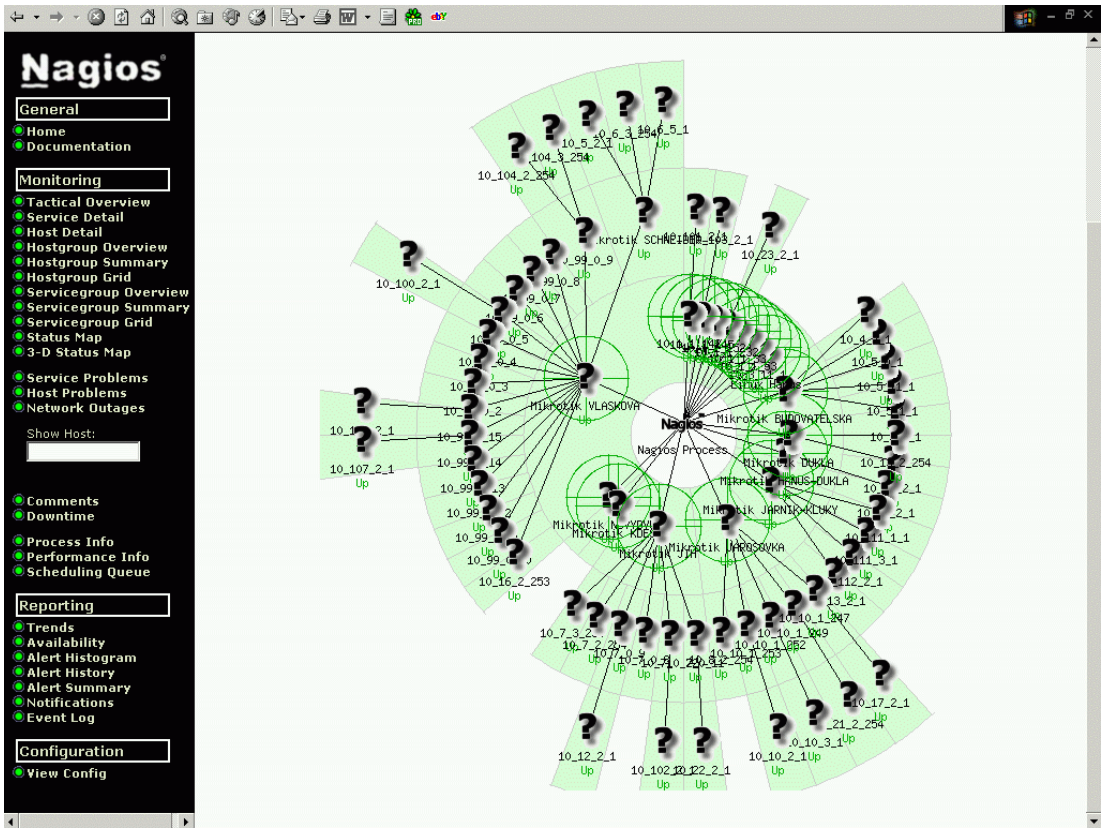
5.2 NAGIOS – komplexní kontrolní systém

Pro automatizovanou kontrolu síťových prvků a dalších zařízení existuje mnoho softwarových pomůcek. Tvůrci MikroTikOS mají například vlastní utilitu, která nabízí bohaté možnosti kontroly celé sítě. Nevýhodou této aplikace je schopnost běhu pouze na operačním systému Windows. Ne každý poskytovatel má v síti Windows server s nepřetržitým během. Ale snad každý má v síti Linux. Pro monitoring tedy použijeme nějaký z rozšířených kontrolních utilit. Mezi nejznámější a nejpoužívanější patří zřejmě **Cacti, Nagios, Netsaint** a **Webalizer**.

V mém případě jsem pro správu sítě nakonfiguroval systém **Nagios**. Konfigurace celého systému není nějak závratně složitá, vyžaduje však určitou orientaci v OS Linux. Základním kamenem celého monitoringu je zkouška dostupnosti síťového prvku pomocí nástroje **ping**. Hlídá se nejen sama odezva prvku, ale také ztrátovost paketů a čas návratu odpovědi. Výsledkem testu jsou tři předem definované stavy: *OK* – systém neposílá žádné upozornění, *WARNING* – ztrátovost nebo čas odpovědi překročí konfiguračním souborem stanovené limity, *CRITICAL* – síťový prvek je nedostupný. Při stavech *WARNING* a *CRITICAL* systém odesílá na předem definované adresy upozornění.

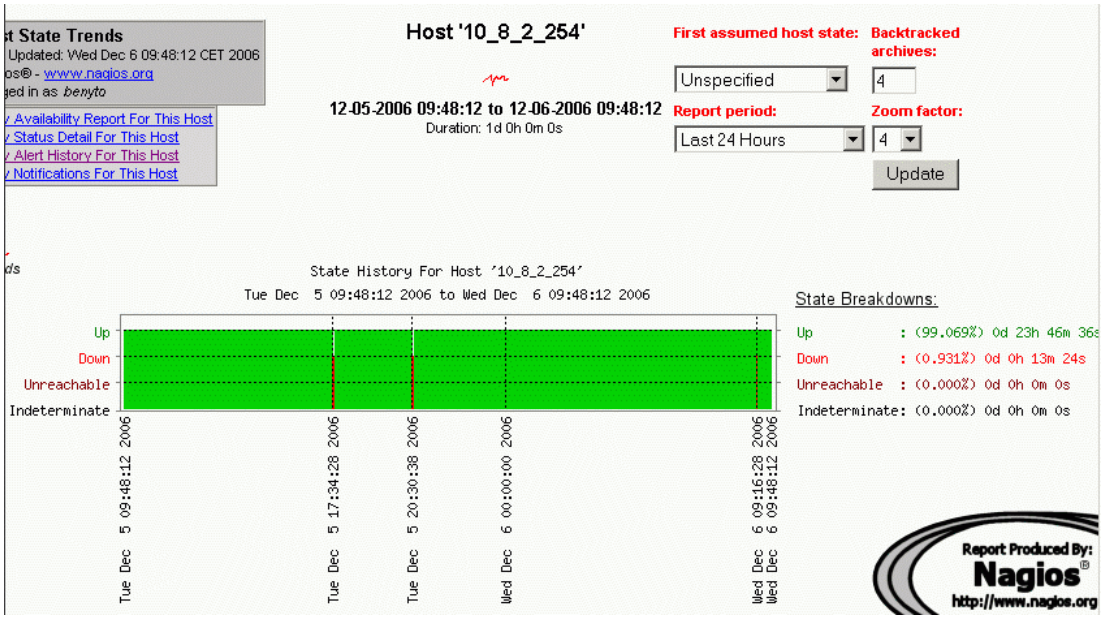
Každý prvek může mít svého administrátora, můžu definovat skupiny prvků, na základě těchto skupin poté provádět různé testy. Samozřejmě je podporována stromová struktura sítě, proto když vypadne router, za kterým jsou větvené další routery, přijde upozornění pouze na nedostupnost „nejvyššího zařízení v hierarchii“, abychom nebyli zahlceni zprávami o nedostupnosti podřízených prvků.

Služba běží jako démon na pozadí. Na webovém rozhraní je možné prohlížet stavy jednotlivých prvků, statistiky výpadků a jiné užitečné informace.



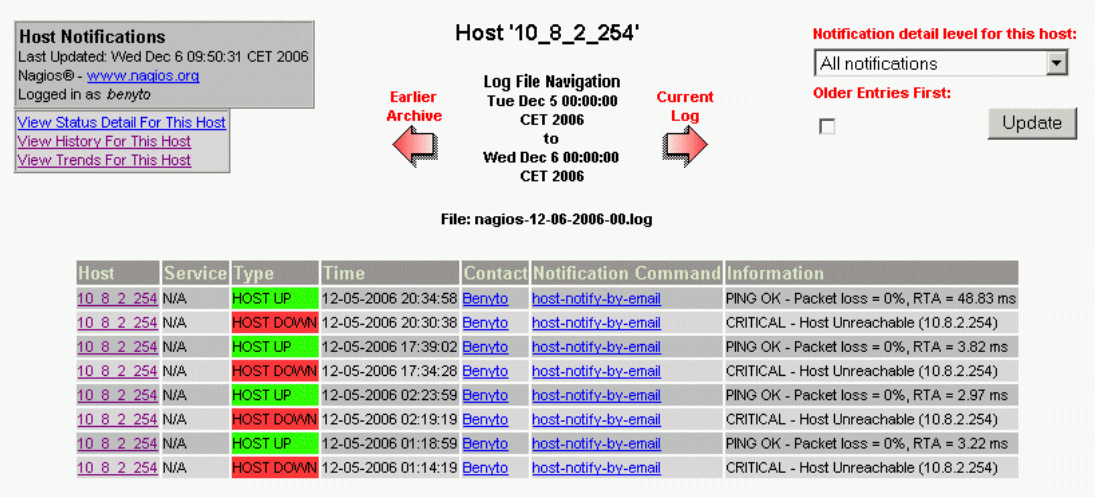
Obr. 28 – **Status Map** sítě zobrazená programem **Nagios**

Při velkém množství prvků se stává mapa mírně nepřehledná, každopádně její využití je spíše pro kontrolu, zdali máme v pořádku hierarchii sítě. Pokud je nějaký prvek nedostupný, změní se zelená podkladová barva výřezu na výstražnou červenou.



Obr. 29 – **Host state trends** – grafické zobrazení dostupnosti prvku

Na obrázku 29 vidíme graf dostupnosti zvoleného prvku. Zelený pruh znamená bezproblémovou dostupnost. Červená barva značí **down** stav prvku – nepřišla odpověď na testovací pakety. Pro každý takový výpadek mám k dispozici jeho přesný čas. Fialová je **unreachable** – to může nastat pouze v případě, že nadřízený prvek sledovaného prvku se dostal do stavu **down** a neodpovídá na kontroly. Takový problém se v mém ukázkovém případě nevyskytl. V pravé části stránky je číselně vypsána statistika – prvek byl dostupný v 99,069 % sledovaného času a nedostupný v 0,931 % času.



Host	Service	Type	Time	Contact	Notification Command	Information
10_8_2_254	N/A	HOST UP	12-05-2006 20:34:58	Benyto	host-notify-by-email	PING OK - Packet loss = 0%, RTA = 48.83 ms
10_8_2_254	N/A	HOST DOWN	12-05-2006 20:30:38	Benyto	host-notify-by-email	CRITICAL - Host Unreachable (10.8.2.254)
10_8_2_254	N/A	HOST UP	12-05-2006 17:39:02	Benyto	host-notify-by-email	PING OK - Packet loss = 0%, RTA = 3.82 ms
10_8_2_254	N/A	HOST DOWN	12-05-2006 17:34:28	Benyto	host-notify-by-email	CRITICAL - Host Unreachable (10.8.2.254)
10_8_2_254	N/A	HOST UP	12-05-2006 02:23:59	Benyto	host-notify-by-email	PING OK - Packet loss = 0%, RTA = 2.97 ms
10_8_2_254	N/A	HOST DOWN	12-05-2006 02:19:19	Benyto	host-notify-by-email	CRITICAL - Host Unreachable (10.8.2.254)
10_8_2_254	N/A	HOST UP	12-05-2006 01:18:59	Benyto	host-notify-by-email	PING OK - Packet loss = 0%, RTA = 3.22 ms
10_8_2_254	N/A	HOST DOWN	12-05-2006 01:14:19	Benyto	host-notify-by-email	CRITICAL - Host Unreachable (10.8.2.254)

Obr. 30 – **Host notifications** – seznam upozornění zaslaných pro zobrazený prvek

V **Host Notifications** můžeme přehledně vidět kolik upozornění bylo zasláno, jakým způsobem, kdy a komu.

6 Závěr

Ve své práci jsem se snažil poskytnout základní náhled do funkce zařízení, která řídí datové toky v sítích lokálních poskytovatelů internetu. Nebylo jednoduché vybrat z nepřeberného množství funkcí, ať už se jednalo o systém MikrotikOS nebo OS Linux. Osobně se na tuto práci dívám jako na možnou příručku pro začínající nadšence v oblasti počítačových sítí s účelem poskytování internetu dalším osobám (komerčně i nekomerčně).

V mém modelovém případě se jednalo o síť tvořenou převážně bezdrátovými zařízeními. Reálné konfigurační ukázky a provozní statistiky jsem pořídil z metropolitní sítě, která v době psaní práce zahrnovala zhruba 600 účastníků. Použití systému MikrotikOS se ukazuje jako výhodné do chvíle, kdy provozuji malou, jednoduchou síť, bez mnoha různých aplikovaných FUP pravidel. I přes možnost využití poměrně silných kvalitních nástrojů – skriptování a plánovače – dochází při velkém počtu takto aplikovaných pravidel k větší míře nepřehlednosti a síť se stává nepohyblivým a těžkopádným „molochem“.

I když se použije poměrně moderní silný hardware (v mém případě P4 2.4 GHz, 128 MB RAM – to celé v rackovém provedení 1U), dochází v případě mnoha filtrovacích a kontrolních pravidel pro více klientů (více než 500) k patrnému selhávání celkové funkce zařízení. Ve špičkách největšího vytížení (běžně kolem čtvrté odpolední a dále po osmé hodině večerní) se mnohokráte stalo, že i když graf vytížení routeru nepřekročil hranici 80ti procent, tak přesto nefungovalo férové dělení konektivity či omezování stahování z p2p sítí. Vyskytly se situace, kdy některým klientům fungoval internet tak jako jindy, ale některým se rychlost rapidně snížila a zvýšily se časové odezvy odpovědí (výrazný nárůst z běžně dosahovaných hodnot 20 – 30 ms na nepoužitelné hodnoty 500 ms a více).

Velkým poučením v tomto případě je, že levná a jednoduchá zařízení v této branži mnohdy nedokáží simulovat činnost, kterou proklamují jejich tvůrci. Pokud se chystá poskytovatel na větší expanzi a nabídku širokého spektra služeb, je nezbytné investovat do kvalitních profesionálních zařízení alespoň do centrálních částí sítě. Jako inteligentní přístupové body s místní úpravou QOS poslouží MikrotikOS naprosto ideálně v porovnání výkonu a ceny!

Veškeré praktické ukázky mé práce jsou převzaty z ostrého provozu metropolitní sítě v Písku. Využití systému MikrotikOS je dnes velice rozšířené, avšak vzhledem k rapidnímu vývoji v oblasti sítí se nedá očekávat, že tento systém přežije delší dobu. Přesto v současné době vidím v jeho využití velký potenciál, který je daný především nízkou cenou a v podstatě celkem jednoduchou správou **základních** služeb.

7 Literatura

DOSTÁLEK, L. : Velký průvodce protokoly TCP/IP: bezpečnost, Praha. Computer Press, 2003.

KABELOVÁ, A, DOSTÁLEK, L.: Velký průvodce protokoly TCP/IP a systémem DNS, Praha, Computer Press, 2002

KOLEKTIV AUTORŮ: Linux Dokumentační projekt. Praha, Computer Press, 2003.

NEMETH, E., GARTH, S., TRENT, H.: LINUX – Kompletní příručka administrátora. Praha, Computer Press, 2004

Internetové a speciální zdroje:

www.mikrotik.com

routery s MikroTikOS – v ostrém provozu

Linux server – Fedora Core 5 – v ostrém provozu