UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

MikroTik load-balancing pro SOHO využití

Radomír Pleskač

Bakalářská práce
2015

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2014/2015**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Radomír Pleskač**
Osobní číslo:	**I11153**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Mikrotik load-balancing pro SOHO využití**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování:

Cílem práce je praktická aplikace, testování a následné porovnání různých mechanismů load-balancingu (ECMP, Firewall označování, per-user, per-connection, per-packet, per-traffic-type) na platformě Mikrotik, tak aby bylo dosaženo nejlepší propustnosti. Praktická část bude obsahovat topologii zapojení, přesný popis zařízení a jejich konfiguraci. Obsah práce bude uložen na přiloženém CD.

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

DISCHER, Stephen R. W. RouterOS by example: understanding MikroTik RouterOS through real life applications. College Station, Texas: MikroTik, 2011. ISBN 978-061-5547-046.

BIGELOW, Stephen J. Mistrovství v počítačových sítích: správa, konfigurace, diagnostika a řešení problémů. Vyd. 1. Překlad Petr Matějů. Brno: Computer Press, 2004, 990 s. ISBN 80-251-0178-9.

Vedoucí bakalářské práce: **Ing. Soňa Neradová**
Katedra softwarových technologií

Datum zadání bakalářské práce: **20. prosince 2014**
Termín odevzdání bakalářské práce: **11. května 2015**

prof. Ing. Simeon Karamazov, Dr.
děkan

L.S.

Ing. Lukáš Cegan, Ph.D.
vedoucí katedry

V Pardubicích dne 31. března 2015

Prohlášení autora

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 30.4.2015 Radomír Pleskač

Poděkování

Rád bych poděkoval paní Ing. Soně Neradové za cenné rady a informace při vedení mé práce.

Také bych chtěl poděkovat Bc. Miroslavovi Kratochvílovi za vysvětlení některých konceptů a rady týkající se RouterOS.

Anotace

Cílem práce je praktická aplikace, testování a následné porovnání různých mechanismů load-balancingu (ECMP, Firewall označování, per-user, per-connection, per-packet, per-traffic-type) na platformě MikroTik, tak aby bylo dosaženo nejlepší propustnosti. Praktická část bude obsahovat topologii zapojení, přesný popis zařízení a jejich konfiguraci. Obsah práce bude uložen na přiloženém CD.

Klíčová slova

MikroTik, load-balancing, ECMP, connection, firewall, zálohování

Title

MikroTik load-balancing in SOHO enviroment.

Annotation

The goal of this thesis is practical implementation, testing and comparison of different load-balancing methods (ECMP, Firewall marking, per-user, per-packet, per-connection, per-traffic-type) on MikroTik platform, with regards to best possible throughput. Practical part of the thesis will contain network topology, precise description of used networking hardware and its configuration, The content of this thesis will be stored on included CD.

Keywords

MikroTik, load-balancing, ECMP, connection, firewall, fail-over

Obsah

Seznam zkratek

IP	Internet Protocol
NAT	Network Address Translation
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
PPTP	Point-to-Point Tunneling Protocol
ICMP	Internet Control Message Protocol
ARP	Address Resolution Protocol
TTL	Time-to-Live
OSPF	Open Shortest Path First
RIP	Routing Information protocol
ISP	Internet service provider
SOHO	Small Office Home Office
DoS	Denial of Service
OSI	Open Systems Interconnection
PoE	Power over Ethernet
P2P	Peer-to-Peer PoE
TZSP	TaZmen Sniffer Protocol

Seznam obrázků

Seznam tabulek

Úvod

V dnešní době se stává pro naši civilizaci internetová konektivita čím dál tím důležitější, a to především v životě malých firem a z domova pracujících lidí, kde každá chvíle nefungujícího spojení znamená velké potenciální ztráty a nemožnost práce. Z tohoto důvodu se mnoho menších firem rozhodlo nesvěřovat svůj osud do rukou pouhého jednoho poskytovatele (ISP) a pro jistotu si zřídilo připojení dvě. Mnoho z nich se však spokojilo s jednoduchým mnohdy "manuálním" přepínáním v případě výpadku a nikdy nepřemýšlelo o současném využití více připojení najednou pro zvýšení kapacity a rychlosti sítě.

Cílem této bakalářské práce je seznámení se s možnostmi současného využití více internetových připojení, jejich load-balancingu, automatického přepojení na záložní spojení při výpadku a nasazení celého tohoto systému na cenově dostupném zařízení MikroTik v prostředí SOHO.

K pochopení těchto konceptů a jejich nasazení je třeba se seznámit se základy síťového provozu, modelem TCP/IP, směrováním a především protokoly 3. a 4. vrstvy tohoto modelu. Další část se věnuje seznámení se s platformou MikroTik a funkcemi, které budeme využívat a hlavně podrobnějšímu vysvětlení směrování v operačním systému RouterOS a firewallu založeného na modelu linuxového IPTables. Závěrem teoretické části je pak popis a vysvětlení jednotlivých možností implementace load-balancingu a zálohování a jejich porovnání.

Praktická část se zaměřuje na podrobnější vysvětlení a popis konfigurace třech v praxi nejvíce použitelných možností load-balancingu, jejich testování a následný popis a konfiguraci nastavení zálohování.

1 TCP/IP

1.1 Úvod do teorie a definice základních výrazů

Zjednodušeně si počítačovou síť můžeme představit jako běžnou poštovní síť, kde místo dat posíláme balíčky (ty v počítačové síti nazýváme **datagramy** či **pakety**). Balíčky jsou baleny a posílány podle určitých pravidel, aby dorazily na správné místo a abychom je dokázali efektivně rozbalit - tato standardizovaná pravidla, zajišťující efektivní komunikaci se nazývají **protokoly**.

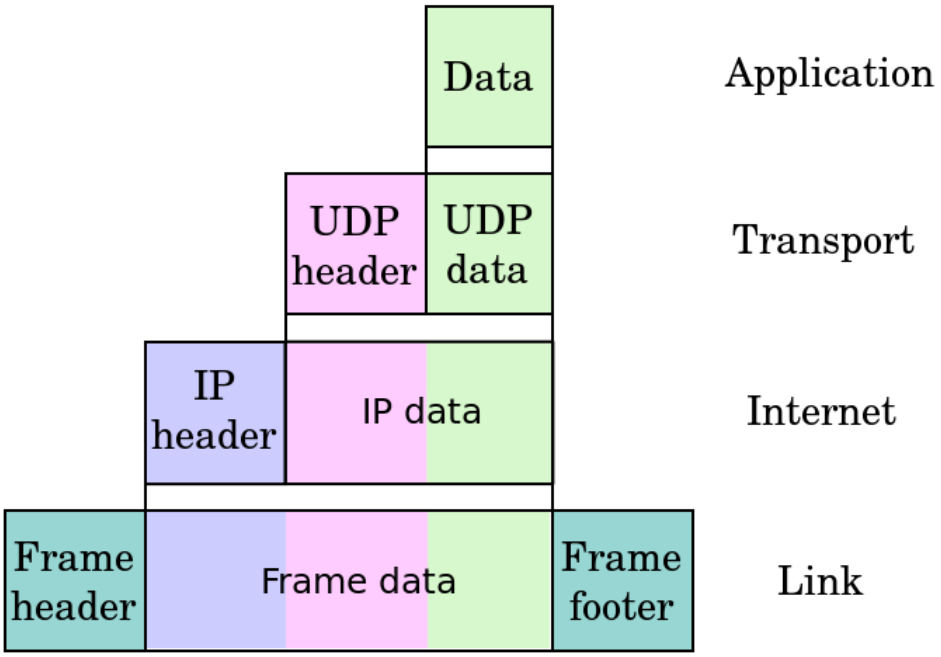
Stejně jako se poštovní balíčky zastavují na překladištích, tak i v počítačových sítích existují zařízení, starající se o manipulaci a co nejefektivnější výběr cesty **(routy)** - těm říkáme **směrovače** (**routery).**

Naším cílem tedy zjednodušeně bude rovnoměrně roztřídit balíčky na dvě skupiny a ty poslat každé jinou cestou. [1,3]

1.2 Zapouzdření a TCP/IP Model

Abychom mohli balíčky třídit a směrovat různými cestami, musíme si nejdříve ukázat, co vlastně takový balíček obsahuje.

Mezi odesláním a přijetím dat existuje několik fází (neboli **vrstev**), ve kterých je náš balíček zpracováván několika protokoly. Protokol vezme data a zabalí je do balíčku (encapsulation), tento balíček je pak předán protokolu další vrstvy, kde je balíček znovu zabalen. Tyto obaly neboli pouzdra přidávají k zprávě **hlavičku** (a patičku), která obsahuje informace daného protokolu (například adresu doručení nebo délku zprávy).



Obrázek 1 – Znázornění zapouzdření a TCP/IP model [3]

Tyto fáze nám pomáhá vysvětlit model TCP/IP. Ten je rozdělený na 4 vrstvy.

Vrstva síťového rozhraní (Link layer) se stará o jednoduchou fyzickou komunikaci mezi hardwarem na lokální síti. Spadají do ní fyzické spojení a technologie jako Ethernet či WiFi a tunelovací protokoly jako PPP (Point-to-Point Protocol).

Síťová vrstva (Internet layer) se stará o doručování, směrování a hledání cest k cíli i pokud není adresa doručení na lokální síti. Využívá k tomu logické adresování. Srdcem síťové vrstvy je **IP** (Internet Protocol) a nás bude zajímat hlavně jeho verze **IPv4**, která je stále nejpoužívanější. Kromě IP na této vrstvě operují ještě další pomocné protokoly jako například ICMP (Internet Control Message Protocol) či směrovací protokoly (RIP, OSPF, BGP a další).

Transportní vrstva (Transport Layer) nám zajišťuje především tzv. End-to-end konektivitu mezi aplikacemi a to pomocí portů. **Port** je 16bitové číslo určující bránu na zařízení, na které naslouchá určitá aplikace (protokol vyšší vrstvy). Klíčovými protokoly této vrstvy jsou **TCP** a **UDP**.

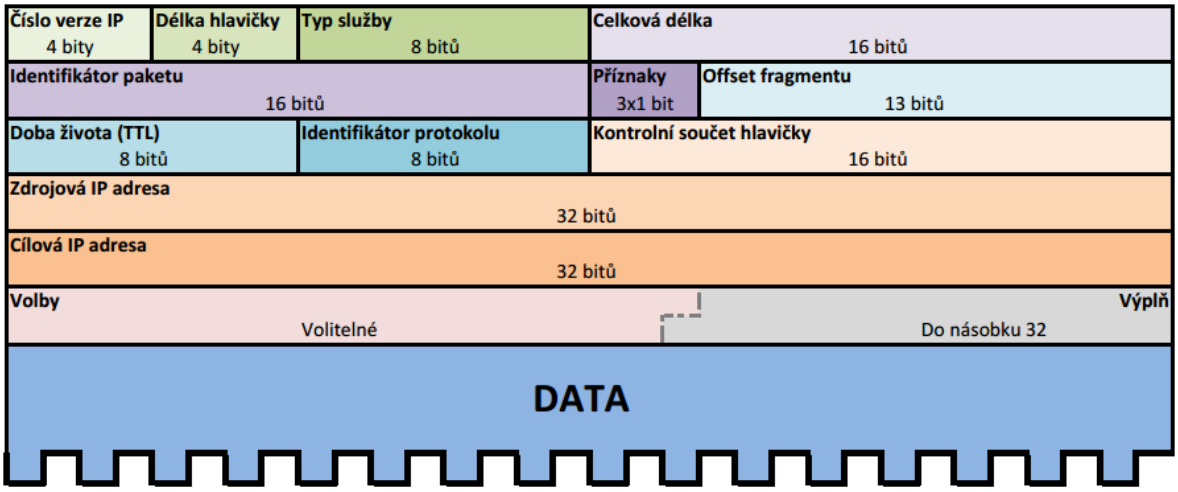
Aplikační vrstva (Application layer) se skládá už přímo z aplikací a protokolů, které využívají služby transportní vrstvy. Například protokol HTTP pro prohlížení webových stránek či FTP na přenos souborů. [1,3,11]

1.3 IPv4

Hlavní funkce protokolu IPv4 je

- adresace koncových stanic v internetu pomocí logického adresování,
- vytváření paketů z protokolů vyšší vrstvy,
- směrování paketů a fragmentace (v případě, že se pakety přenášejí přes síť, která na vrstvě síťového rozhraní nedokáže tak velký paket přenést, IP protokol je schopný ho rozdělit na několik menších a poté zase složit).

Protože pakety/datagramy protokolů vyšších vrstev jsou zabaleny v datové části IP paketu, můžeme říci, že IP paket je základní jednotkou pro přenos v síti.



Obrázek 2 – diagram IPv4 Paketu [3, 11]

Hlavička IPv4 paketu se skládá z těchto polí :

- **Číslo verze IP.** Předává informaci o tom, zda se jedná o IPv4 nebo IPv6.
- **Délka hlavičky.** Určuje celkovou délku hlavičky v násobcích 32 bitů.
- **Typ služby.** Tato část stanovuje parametry přenosu tohoto paketu - prioritu, propustnost, spolehlivost a zpoždění. Dnes se používá jen v mechanismech QoS.
- **Celkové délka.** Určuje celkovou délku paketu v Bytech (v dílech po 8 bit).
- **Identifikátor paketu.** Jednoznačné ID, které rozlišuje jednotlivé pakety. Pokud byl paket fragmentován, dokáže díky tomuto poli zjistit, které části patří k sobě a spojit je.
- **Příznaky (Flags).** Řídí fragmentaci paketu. Určují, zda paket může být fragmentován, případně nesou informaci o tom, zda fragmentován byl a zdali není posledním fragmentem.
- **Offset fragmentu.** Udává, na jaké pozici v původním datagramu začíná tento fragment.

- **Doba života (TTL).** Aby se pakety nezacyklily (neběhaly po síti do nekonečna), je doba životnosti každého paketu omezena. Může být nastavena v rozmezí 0-255 a s každým skokem na cestě se snižuje o 1. Když dojde na nulu, paket je zničen.
- **Identifikátor protokolu.** Určuje, jaký protokol následuje za hlavičkou IP (např. TCP / UDP / ICMP a další).
- **Kontrolní součet hlavičky.** Slouží ke kontrole chyb. Počítá se z hlavičky, a pokud na místě doručení nesouhlasí, je paket zahozen.
- **Zdrojová IP adresa.**
- **Cílová IP adresa.**
- **Volby (Options).** Nepovinné rozšiřující informace nebo požadavky. Obvykle se nepoužívají.
- **Prázdný prostor (výplň).** Zde se zapisují nulové bity, aby měla hlavička velikost násobku 32 bitů.

1.4 IP Adresování a routing

Původně mělo každé zařízení (respektive každé síťové rozhraní připojené k internetu na daném zařízení) přidělenou vlastní jedinečnou 32bitovou adresu. Z důvodu nedostatku adres byly ale později vyčleněny segmenty tzv. privátních adres, což jsou adresy, které nejsou směrovatelné v internetu a nemusí být proto jedinečné. Stanice s těmito adresami mají přístup k internetu pomocí funkce **NAT** (Network Address Translation).

IP protokol poskytuje možnost logického členění sítě na podsítě pomocí tzv. masky podsítě. Tou je IP adresa rozdělena na dvě různě velké části (v závislosti na velikosti masky) :

- **Network ID.** Část adresy, která identifikuje, v jaké podsíti se dá zařízení nalézt. (Podobně jako např. telefonní čísla mají předvolbu státu).
- **Host ID**. Zbytek adresy, podle které se identifikuje zařízení v podsíti.

Pokud host[1] vidí, že je cílová adresa paketu, který odesílá na stejné síti, pošle zprávu rovnou pomocí protokolu linkové vrstvy (tzv. **direct delivery**). Převod mezi logickými adresami (IP) a adresami na linkové vrstvě (hardwarová adresa MAC) zajišťuje ARP (Address Resolution Protocol).

Pokud adresa není v stejné síti, host pošle paket na tzv. **Default Gateway** - což je adresa přilehlého[2] směrovače.

Důležitým konceptem směrování je tzv. Next-hop routing. To znamená, že směrování paketů probíhá krok po kroku, tedy router, který paket přeposílá, nezná a ani nepotřebuje znát celou cestu k cíli, stačí mu znát pouze adresu dalšího přilehlého routru, který je blíže na cestě. Tento proces pokračuje, až dokud nedorazí paket do cíle nebo nevyprší jeho životnost (TTL).

[1] Myšleno zařízení, které je členem dané sítě.
[2] Myšleno ležící, ve stejné sítí - tedy komunikující spolu na linkové vrstvě.

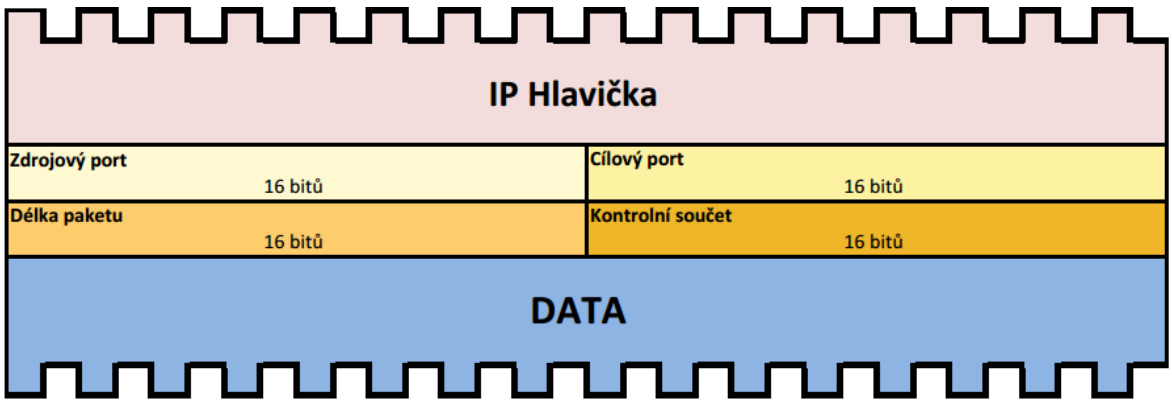
O tom, kam poslat paket, rozhoduje router podle tzv. Routing Table - to je tabulka obsahující záznamy s Network ID a adresou přilehlého routeru, který do této podsítě zná cestu.

Tato tabulka může být naplněna **dynamicky** pomocí směrovacích protokolů, což jsou protokoly, jejichž pomocí si přilehlé routery posílají informace o tom, do jakých sítí jsou schopny poslat paket (např. OSPF, RIP). A nebo **staticky** - ručně. [1, 3, 11]

1.5 UDP

UDP (User Datagram[3] Protocol) je prvním ze dvou protokolů transportní vrstvy sady TCP/IP určených k přenosu dat aplikační vrstvy. Je to jednoduchý protokol, který poskytuje jen a pouze výše zmíněnou funkcionalitu portů. Jeho výhodou je malá hlavička (více místa zbude na data).

Používá se hlavně v případech, kdy preferujeme, aby se data dostala do cíle co nejdříve a to i za cenu chyb - např. VoIP telefonie nebo streaming videa.



Obrázek 3 - diagram UDP datagramu [3, 10]

Hlavička protokolu UDP se skládá z těchto polí :

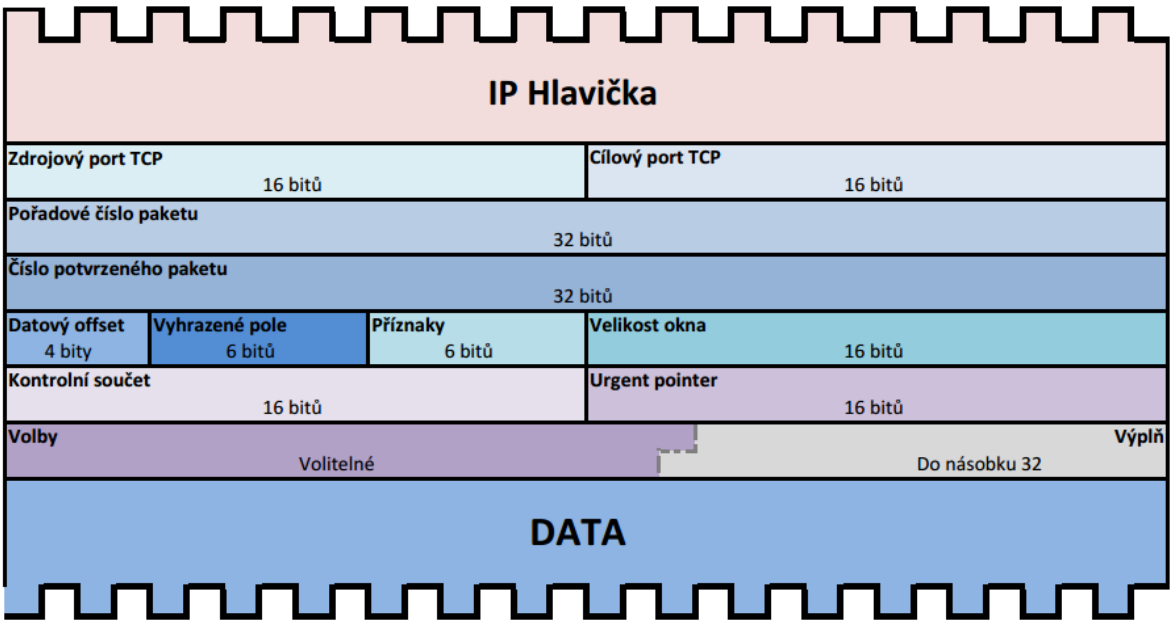
- **Zdrojový port.**
- **Cílový port.**
- **Délka paketu.** Určuje délku celého UDP datagramu (hlavička + data).
- **Kontrolní součet** Nepovinný kontrolní součet celého UDP datagramu. [1, 3, 11]

[3] Pojmy datagram a paket jsou velmi podobné a často zaměňované. Pojem paket se často používá obecně ve smyslu jakékoli síťové zprávy. Správně by se ale pojem paket měl používat jen pro ty zprávy síťové vrstvy, které požadují potvrzení příjmu od cílového příjemce. Pokud tedy posíláme zprávu pomocí protokolu UDP, nejedná se o paket, ale o datagram.

1.6 TCP

TCP (Transmission Control Protocol) je druhou možností, kterou si protokol aplikační vrstvy může vybrat k posílání zpráv. Oproti UDP kromě portů zajišťuje další důležité funkce:

- Protokol TCP neposílá jednotlivé pakety, ale vždy nejdříve vytvoří **spojení (connection)**. Stará se také o udržování tohoto spojení a při skončení přenosů ho ukončí.
- **Spolehlivost přenosů** - pakety vyžadují potvrzení příjmu, a pokud potvrzení nepřijde, je paket přeposlán.
- Při přenosu na síti se může stát, že pakety dorazí v jiném pořadí než byly odeslány. TCP zajistí opětovné **seřazení** těchto paketů do správného pořadí.



Obrázek 4 – diagram TCP Paketu [3, 11]

Hlavička protokolu TCP se skládá z těchto polí :

- **Zdrojový port.**
- **Cílový port.**
- **Pořadové číslo paketu.** Pomocí tohoto čísla může dojít k rekonstrukci dat do původní podoby, v jaké byly odeslány.
- **Číslo potvrzeného paketu.** Vyjadřuje pořadové číslo paketu, který je příjemce připraven přijmout.[4]
- **Datový offset.** Obsahuje velikost hlavičky TCP.
- **Vyhrazené pole.** Je vždy rovno 0 a je rezervované pro budoucí využití.
- **Příznaky.** Řídící funkce URG, ACK, SYN, PSH, RST, FIN).
- **Velikost okna.** Určuje, kolik Byte je posíláno na jedno potvrzení.

[4]tzn. potvrzuje, že mu došly všechny pakety do pořadového čísla (N-1)

- **Kontrolní součet.** Slouží ke kontrole chyb.
- **Urgent pointer.** Ukazuje, která část odeslaných dat je urgentní. Pole je relevantní jen v případě, že je použit příznak URG.
- **Volby.** Nepovinné rozšiřující informace nebo požadavky.
- **Prázdný prostor (výplň).** Zde se zapisují nulové bity, aby měla hlavička velikost násobku 32 bitů.

Už z větší velikosti hlavičky je zřetelné, že protokol TCP má oproti UDP větší režii. A to nejen z pohledu množství odeslaných dat, ale kvůli kontrolám a řazení také v latenci a v procesorovém času. Proto se využívá u přenosů, kde přesnost a bezchybnost je prioritou (například E-mail, HTTP). [3, 11]

2 MikroTik RouterOS

MikroTik je lotyšská firma, založená v roce 1995 za účelem výroby routerů a bezdrátových řešení pro ISP. V dnešní době poskytuje hardware a software ve většině zemí světa.

Začínala jako pouhý poskytoval software, když v roce 1997 vytvořila první verzi jejich operačního systému RouterOS na platformu x86 (PC). Dnes je však dostupný i pro procesory PowerPC a MIPS.

V roce 2002 začala s produkcí vlastního hardwaru pod značkou RouterBoard, operujícího na systému RouterOS. Produkuje vše od malých domácích a SOHO WiFi zařízení přes ISP bezdrátové zařízení až po rackové routery.

RouterOS je operační systém založený na Linuxovém kernelu v2.6. Cílem RouterOS je poskytnout velké množství stabilní funkcionality pro využití v síťovém provozu v uživatelsky přívětivém prostředí.

RouterOS může být nainstalován na PC. Minimální požadavky jsou 5.generace x86 CPU (Intel Pentium / AMD K5), 32MB RAM, 64MB úložiště. Podporuje širokou škálu síťových rozhraní od běžných Ethernetových rozhraní a 802.11a/b/g/n po 10Gbit Ethernet, SFP moduly či 3G modemy.

Ke konfiguraci slouží aplikace Winbox, která dokáže komunikovat jak přes IP, tak i pouze na linkové vrstvě, alternativně můžeme použít Telnet, SSH, sériovou konzoli, webové rozhraní či proprietární funkci mac-telnet.

RouterOS je dodáván v různých licencích, které se liší dostupnou funkcionalitou. Produkty MikroTik RouterBoard mají licenci v ceně. [5, 6, 7]

Tabulka 1 - Druhy licencí RouterOS [6]

Level number	1 (Demo)	3 (WISP CPE)	4 (WISP)	5 (WISP)	6 (Controller)
Price	registration required	volume only	$45	$95	$250
Wireless AP	-	-	yes	yes	yes
Wireless Client and Bridge	-	yes	yes	yes	yes
RIP, OSPF, BGP	-	yes	yes	yes	yes
EoIP tunnels	1	unlimited	unlimited	unlimited	unlimited
PPPoE tunnels	1	200	200	500	unlimited
PPTP tunnels	1	200	200	500	unlimited
L2TP tunnels	1	200	200	500	unlimited
OVPN tunnels	1	200	200	unlimited	unlimited
VLAN interfaces	1	unlimited	unlimited	unlimited	unlimited
HotSpot active users	1	1	200	500	unlimited
RADIUS client	-	yes	yes	yes	yes
Queues	1	unlimited	unlimited	unlimited	unlimited
Web proxy	-	yes	yes	yes	yes
User manager active sessions	1	10	20	50	Unlimited
Number of KVM guests	1	Unlimited	Unlimited	Unlimited	Unlimited

2.1 Routing na platformě MikroTik

Router má několik oddělených míst, kde uchovává směrovací informace - **FIB**, **RIB** a interní **tabulky dynamických routovacích protokolů**[5]**.**

2.1.1 RIB (Routing Information Base)

Obsahuje několik routovacích tabulek rozdělených podle routing-mark. Pokud routa nemá žádný routing-mark, je uchovávána v hlavní tabulce.

2.1.2 FIB (Forwarding Information Base)

Obsahuje kopii nezbytných informací z RIB (např. obsahuje jen nejlepší z více možných cest) a ke každému záznamu přidává navíc informaci o síťovém rozhraní, na kterém je brána dané routy dostupná. Používá se k samotnému rozhodování o směrování.

[5] Kromě protokolu BGP, ten je uchováván v RIB.

Tabulka 2 - Parametry jednotlivých položek směrovací tabulky (tzv. routing-entry neboli rout) [4]

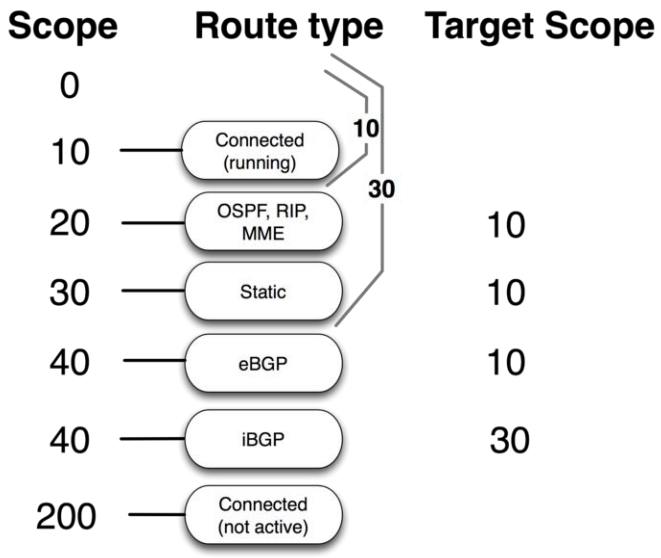
dst-adress	Cílová adresa. Může být buď IP adresa nebo Network ID ve formátu 0.0.0.0/0, kde část za lomítkem specifikuje síťovou masku
gateway	Brána. IP adresa (nebo adresy) [6] přilehlého routru, který zná cestu do cíle.
check-gateway	Nastavení (metody) kontroly brány. Pokud brána přestane na 10 vteřin reagovat, stane se nedostupnou.
distance	Vzdálenost - funguje jako priorita - cesty s kratší vzdáleností do stejného místa mají přednost.
type	Možné volby jsou **unicast, blackhole, prohibit a unreachable**. V případě jiné volby než unicast nebude router provádět směrování, ale bude pakety zahazovat, a to buď potichu (v případě blackhole), anebo odešle zprávu o nedostupnosti.
Route-tag	Specifikuje tag pro potřeby směrovacích protokolů RIP a OSPF.
Pref-src	Slouží k specifikaci IP adresy pro pakety vzniklé na routru putující touto cestou (pokud adresa není lokální, tak je routa neaktivní.)
Routing-mark	...
scope	...
Target-scope	...

Za speciální zmínku stojí parametry **routing-mark, scope** a **target-scope**, které budeme později využívat.

Routing-mark je značka, kterou můžeme označit paket, aby byl směrován pomocí jiné než standardní routovací tabulky a tedy podle jiných pravidel. Toto označujeme jako "**Policy Base Routing**".

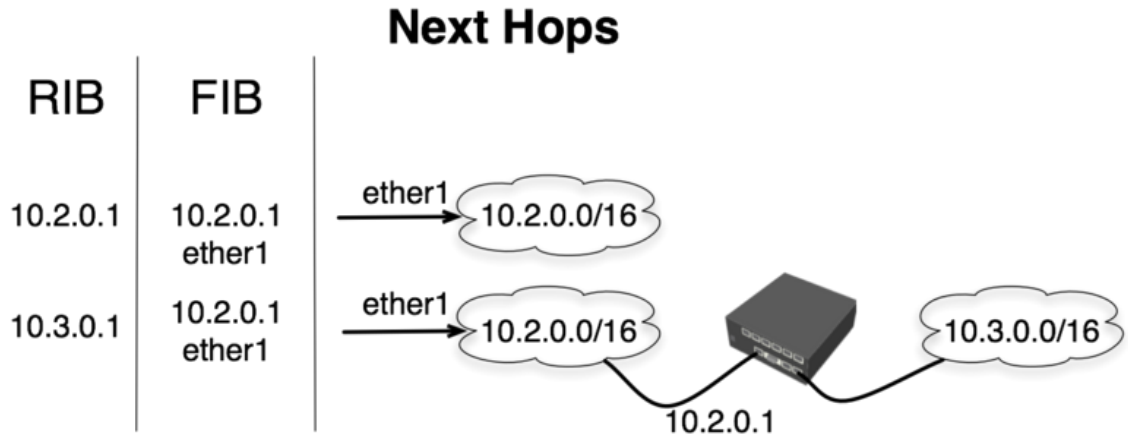
Scope a **target-scope** jsou atributy routy, které specifikují "dosah". Slouží jako ochrana proti tvoření smyček. Například pokud router neví, kde se nachází brána statické routy, tak v standardním nastavení nemůže použít jinou statickou routu, aby zjistil další skok, ale může použít jen a pouze routu, která je přímo dostupná (jejíž brána je dostupná na linkové vrstvě).

[6] Alternativně síťové rozhrání.



Obrázek 5 – Standardní nastavení scope [4]

Tyto atributy můžeme použít k vytvoření tzv. **rekurzivních rout**. Rekurzivní routa je taková routa, jejíž brána není přímo dostupná. Router však pomocí jiné routy zjistil, že je dostupná někde za bránou druhé routy. A tudíž paket pošle na bránu druhé routy.[7] [4]



Obrázek 6 - Reverzní routa [4]

[7] Už z principu next-hop routing je jasné, že ve skutečnosti vůbec nemusí paket přes původní bránu cestovat, ale další router ho může k cíli poslat úplně jinudy.

2.2 MikroTik Firewall / IPTABLES

V RouterOS je směrování portů a všeobecná firewall funkcionalita řešena velmi podobně jako v Linuxu, tedy pravidla víceméně odpovídají linuxovým IPTABLES.

IPTables definuje 5 zachytávacích bodů ("hooks") pro procházející pakety:

Tabulka 3 - IPTables hooks [2, 4]

PREROUTING	Pakety, které právě dorazily na router
INPUT	Pakety, jejichž cílem je sám router, ve fázi předtím než budou odkázány lokálnímu procesu (např. DNS serveru běžícímu na routeru)
FORWARD	Pakety, které prochází routerem (přijdou jedním rozhraním a vychází jiným rozhraním)
OUTPUT	Pakety v okamžiku, co byly vygenerovány lokálním procesem běžícím na routru (např. DNS response)
POSTROUTING	Pakety chvíli před tím než opouští síťové rozhraní routru

Pro každý tento zachytávací bod můžeme nastavit sekvenci pravidel. Zachycený paket prochází seznamem, dokud nenalezne pravidlo, které se k němu vztahuje a vykoná odpovídající akci.

Součástí iptables jsou také tři tabulky, které reprezentují, jak budeme pakety ovlivňovat

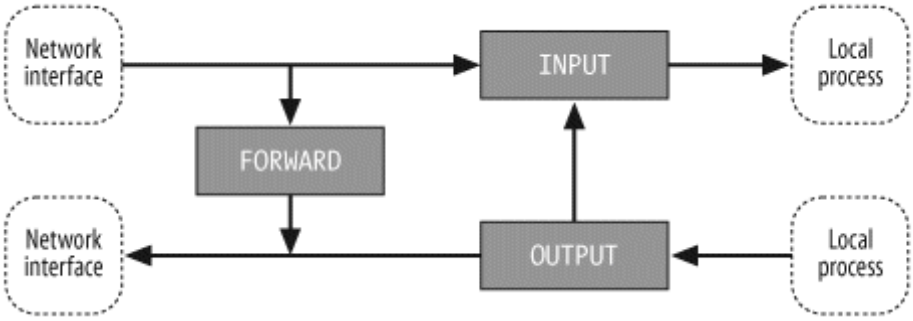
- Filter,
- NAT,
- Mangle.

Tabulky mají **přístup** (tzv. chain) k těmto zachytávacím bodům. [8]

[8] Můžeme definovat i vlastní "chainy", do kterých můžeme posílat pakety k dalším operacím (pomocí akce "jump").

2.2.1 Filter

Soubor pravidel upravujících, jaké pakety mohou vstupovat a odcházet z routru. Jedná se o nejzákladnější formu zpracování paketů.



Obrázek 7 – Packet flow při filtrování [2]

Možné akce pro zpracování paketu tabulkou Filter v RouterOS:

Tabulka 4 - akce tabulky Filter [4]

accept	Paket je přijat a už není zpracováván žádným dalším pravidlem firewallu.
drop	Zahození paketu ("potichu")
reject	Odmítnutí (zahodí a odešle zprávu reject (ICMP))
add-dst-to-address-list / ***add-src-to-address-list***	Přidání IP adresy do "address listu" (seznamu IP adres, který můžeme používat v dalších pravidlech).
jump	Paket skočí do uživatelem vytvořeného nového "chainu".
return	Vrátí paket zpět do "chainu", ze kterého vyskočil.
log	Vloží informace o paketu do logu.
passthrough	Paket ignoruje toto pravidlo a je puštěn k dalšímu (využití např. při sbírání statistik). **Nezaměnit s parametrem passthrough dostupným u některých akcí.** (Ten umožňuje vykonání více pravidel na jednom paketu. Paket vykoná pravidlo a postupuje dál v seznamu.)
tarpit	Zachytí a udržuje TCP spojení. (Tzv. "tekutý písek". Užitečné při ochraně proti DoS útokům či proti automatickému skenování portů.)

2.2.2 NAT (Network address translation)

Druh zpracování paketů, ve kterém měníme zdrojovou/adresátovu IP adresu a/nebo port.

Source NAT(S-NAT / SNAT / srcnat)

Změna zdrojové adresy/portu.

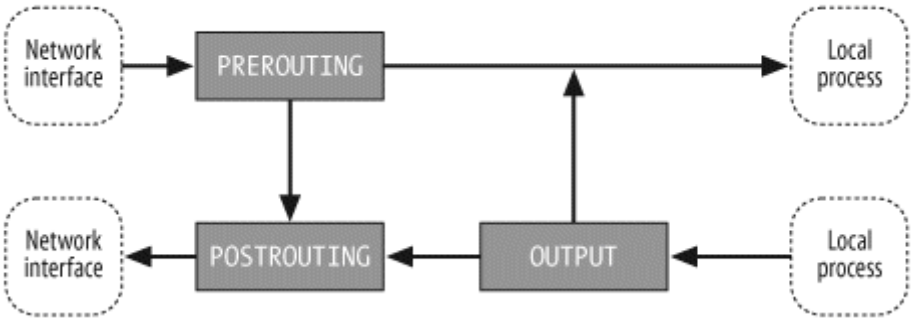
Masquerade je speciální příklad SNATu, využívaný v síti používající DHCP. Oproti standardnímu SNATu je schopný se vypořádat se změnami IP adres při vypršení DHCP lease a adresa je změněna na adresu, kterou zvolí router automaticky.

Nejčastějším využitím SNAT/Masquerade je sdílení jedné IP adresy mezi větším množstvím zařízení. Zařízení za SNATem mohou pouze vytvářet spojení a přijímat odpovědi - není možné se na ně přímo připojit (peer-to-peer).

Destination NAT(D-NAT / DNAT / dstnat)

Změna cílové adresy/portu.

Port Forwarding je příklad DNATu, kdy se router chová jako "proxy" pro služby poskytované ostatními zařízeními. Z pohledu vnější sítě se zdá, jako kdyby všechny tyto služby (např. http server) poskytoval router, ale ve skutečnosti jsou přesměrovávány na jeden či více zařízení. Umožňuje to více zařízením poskytovat služby, aniž by měly každý veřejnou IP adresu.



Obrázek 8 – Packet flow při NATování [2]

Možné akce pro zpracování paketu tabulkou NAT v RouterOS:

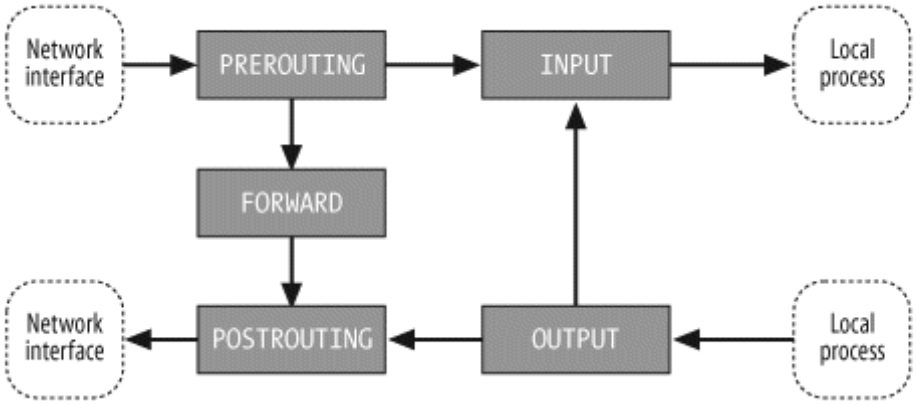
Tabulka 5 - akce tabulky NAT [4]

accept	Paket je přijat a už není zpracováván dalším pravidlem tabulky NAT.
dst-nat	Funkce DNAT.

src-nat	Funkce SNAT.
masquerade	Funkce masquerade.
netmap	1:1 NAT (namapování celé vnější IP na vnitřní 1:1).
redirect	Přesměruje paket na jiný port.
same	Forma SNATu, při které jeden klient vždy dostává stejnou IP adresu (pokud SNATujeme na rozsah adres).
add-dst-to-address-list / ***add-src-to-address-list***	Přidání IP adresy do "address listu" - seznamu IP adres, který můžeme používat v dalších pravidlech.
jump	Paket skočí do uživatelem vytvořeného nového "chainu".
return	Vrátí paket zpět do chainu, ze kterého vyskočil.
log	Vloží informace o paketu do logu.
passthrough	Paket ignoruje toto pravidlo a je puštěn k dalšímu (využití např. při sbírání statistik). **Nezaměnit s parametrem passthrough dostupným u některých akcí.** (Ten umožňuje vykonání více pravidel na jednom paketu. Paket vykoná pravidlo a postupuje dál v seznamu.)

2.2.3 Mangle

Používáme pro specializované změny paketů (například změna TTL, DSCP a podobně) a pro označování (marking) paketů a spojení a přiřazování routing-mark. [2, 4]



Obrázek 9 – Packet flow při Mangle [2]

Možné akce pro zpracování paketu tabulkou Mangle v RouterOS:

Tabulka 6 - akce tabulky Mangle [4]

accept	Paket je přijat a už není zpracováván dalším firewall pravidlem.
change-dscp	Změní DSCP (Differentiated Services Code Point) paketu.
change-mss	Změní Maximum Segment Size v TCP hlavičce paketu.

change-ttl	Změní dobu života (TTL) v IP hlavičce paketu.
clear-df	Zruší "Do Not Fragment" příznak paketu.
mark-connection	Označí connection.
mark-packet	Označí paket.
mark-routing	Označí paket routing markem.
Set-priority	Nastavení priority v IP hlavičce.
Sniff-pc	Využití s TZSP kompatibilními systémy jako např. Wireshark.
Sniff-tzsp	Využití s TZSP kompatibilními systémy jako např. Wireshark.
Strip-ipv4-options	Odstraní parametr options (volby) hlavičky IP paketu.
add-dst-to-address-list / ***add-src-to-address-list***	Přidání IP adresy do "address listu" - seznamu IP adres, který můžeme používat v dalších pravidlech.
jump	Packet skočí do uživatelem vytvořeného nového "chainu".
return	Vrátí packet zpět do "chainu", ze kterého vyskočil.
log	Vloží informace o paketu do logu.
passthrough	Paket ignoruje toto pravidlo a je puštěn k dalšímu (využití např. při sbírání statistik). **Nezaměnit s parametrem passthrough dostupným u některých akcí.** (Ten umožňuje vykonání více pravidel na jednom paketu. Paket vykoná pravidlo a postupuje dál v seznamu.)

2.3 Další důležité funkce RouterOS

2.3.1 Netwatch

Netwatch je funkce, pomocí které můžeme monitorovat zařízení na síti a reagovat na změnu jejich stavu.

Je implementována pomocí protokolu ICMP, a to tak, že periodicky posílá ping na danou adresu, pokud adresa po určitou dobu přestane komunikovat (odesílat reply), tak se její stav nastaví jako "down". Poté, co opět začne odpovídat, je stav nastaven na "up".

Tabulka 7 - Netwatch parametry [4]

host	IP adresa, kterou kontrolujeme.
interval	Interval mezi jednotlivými pingy
timeout	Doba bez odpovědi, po jejímž uplynutí je zařízení považováno za neaktivní ("down").
Down-script	Akce, která se má vykonat, když host změní stav na "down". Může být buď sekvence standardních MikroTik příkazů a nebo skript v jazyce lua.
Up-script	Akce, která se má vykonat, když host opět začne reagovat.

2.3.2 Layer7 Protocols

Layer 7 protocols[9] je funkcionalita MikroTik Firewallu, která nám umožňuje nahlédnout do datové části TCP/UDP paketu a pokusit se identifikovat, o jaký protokol aplikační vrstvy se jedná.

Vždy se kontroluje prvních 10 paketů daného connection (nebo první 2 KB, podle toho, co přijde dřív). Pokud zkontrolovaná část odpovídá uživatelsky definované šabloně, tak je connection označeno podle této šablony. Jinak je označeno jako neznámé a puštěno dál.

Problémem je CPU a paměťová náročnost, zvláště pokud routerem proudí hodně dat. Proto se pravidla obsahující tuto kontrolu většinou dávají na konec Firewall tabulky, aby se tak kontrolovalo jen minimum paketů.

Šablony nejznámějších protokolů se dají nalézt na webových stránkách projektu L7-filter od nadace ClearFoundation.

2.3.3 Connection Tracking

Connection tracking je důležitá funkce routru, která nám umožňuje shlukovat pakety do skupin. Používá k tomu informace z paketů. Díky těmto informacím a díky komunikačním šablonám dokáže klasifikovat i UDP komunikaci jako spojení, i když UDP sám o sobě je protokol, který spojení nevytváří.

Například:
FTP (File Transfer Protocol) využívá dvě oddělené connections, jednu pro přenos souborů a druhou pro správu spojení. Connection tracking využije svou znalost FTP protokolu a z paketů získá dostatečné množství informací, aby dokázal tato spojení identifikovat.

V RouterOS můžeme seznam všech spojení nalézt v sekci *IP* > *Firewall* > *Connections*. [4]

[9]Layer7 - tak je označována aplikační vrstva v modelu OSI (alternativa modelu TCP/IP)

3 Metody Load-balancingu

V této kapitole se pokusím zodpovědět otázku, podle čeho se dají rozdělovat pakety/datagramy na naší síti a jaké výhody a nevýhody dané přístupy mají.

3.1 Podle uživatelů

První, co každého asi napadne, je rozdělovat pakety podle toho, jaký uživatel je odeslal. Tento přístup má výhodu v jednoduchosti pochopení a vyvarování se prakticky všech možných problémů, které mohou nastat v jiných případech.

Uživatele naší sítě jednoduše rozdělíme do dvou skupin podle IP adres. Následně označíme pakety podle toho, z jaké skupiny pocházejí.

Prvním a tím nejvíce do očí bijícím problémem tohoto přístupu je, že ačkoliv může fungovat docela dobře ve větších sítích, kde se rozdíly ve využití mezi jednotlivými uživateli zprůměrují, tak v SOHO sítí (1-10 uživatelů) může být rozložení značně nerovnoměrné a v těch nejmenších SOHO sítích o 1 uživateli je tento přístup úplně bezcenný.

3.2 Podle paketů

Tím druhým řešením, které hodně lidí může napadnout, je rozdělovat přímo pakety jeden po druhém. V praxi je toto řešení však pro naše účely nepoužitelné (až na několik málo výjimek) a to proto, že kvůli různě dlouhým cestám do internetu a různě velkým latencím by pakety dorazily do cíle mimo pořadí, což by způsobovalo problémy s TCP spojeními.

3.3 Podle typu trafficu[10]

Dalším řešením je rozdělovat pakety podle toho, co obsahují. Co paket obsahuje, můžeme poznat třemi způsoby :

- **Podle protokolu síťové nebo transportní vrstvy.** MikroTik Firewall dokáže rozeznat pakety podle toho, jakému protokolu náleží. Jsme tak schopni například rozdělit pakety do dvou skupin, kdy jedna skupina bude obsahovat jen TCP pakety a druhá zbytek (UDP, ICMP, routovací protokoly a podobně).
- **Podle portu transportní vrstvy.** Prvních 1024 TCP/UDP portů patří do skupiny známých portů a podle tabulky[11] jsme tak teoreticky schopni zjistit, co za službu by měl paket směřující na tento port poskytovat. Samozřejmě se ale může stát, že některé služby toto nedodržují a operují na portu, který jim "nepatří". I přesto je toto velmi užitečná věc a dokážeme tak rozdělit například webový traffic (port 80 + 443) od FTP (port 20+21).

[10] Traffic - síťový provoz
[11] Tabulka známých protokolů specifikovaných v RFC 1700

- **Podle protokolu aplikační vrstvy.** K tomuto účelu použijeme dříve zmíněnou Layer7 funkcionalitu. Může sloužit například na oddělení YouTube trafficu či protokolů které fungují na mnoha různých portech (P2P, BitTorrent a podobně).

Tímto přístupem v SOHO prostředí také bohužel většinou nedosáhneme rovnoměrného rozdělení, ale na druhou stranu máme daleko větší kontrolu nad tím, jaký traffic míří přes jaké spojení. A to může být výhodou v případech, kdy například jedno z našich spojení má výrazně lepší latenci, ale nižší kapacitu a my chceme, aby určitý tok dat šel přes něj, zatímco jiný traffic, který není závislý na latenci[12], ne.

Tato metoda ale také vyžaduje větší "údržbu", například se totiž může stát, že L7 filtry přestanou fungovat, protože daná aplikace v nové verzi komunikuje trošku jinak.

3.4 Podle Connection

Na závěr si ukážeme asi nejvíce použitelnou metodu. Vrátíme se k funkci Connection Tracking popsané výše a ukážeme si, jak může být užitečná.

Díky connection trackingu dokážeme shlukovat traffic do skupin paketů, které k sobě patří, a ty posílat stejnou cestou. Tak se vyhneme (alespoň z velké části) problému přerušení spojení.
Hlavní výhodou tohoto přístupu je teoreticky velmi vyrovnané rozložení i na malých sítích, a to i v případě sítě s jen jedním uživatelem.

Na platformě MikroTik tohoto můžeme dosáhnout několik způsoby :

3.4.1 ECMP (Equal Cost Multipath)

je routovací strategie, kdy existuje k jednomu cíli více cest se stejnou vzdáleností (parametr distance). MikroTik implementuje cache[13] paměť a pakety se stejnými adresami, zdrojovým interfacem a parametrem typu služby (v IP hlavičce) posílá stejnou cestou. ECMP je tedy na MikroTiku de facto děleno na connectiony. Problém a hlavní důvod, proč se nedoporučuje ECMP na platformě MikroTik používat, je však tzv. "routing table flush", což znamená smazání této paměti cache. V takovém případě se může stát, že další pakety budou proudit jinou bránou a TCP spojení budou narušena. Tento flush se stává pokaždé, když nastane změna routovací tabulky a zároveň každých 10minut z bezpečnostních důvodů (obrana před DoS útoky).

3.4.2 Nth (neboli "n-tý")

je jednoduchá funkce, která podle rozpočítadla dokáže rozdělit pakety/connectiony na skupiny. Jednoduše každý n-tý prvek jde do n-té skupiny. Problémem tohoto přístupu je, že nemáme žádnou kontrolu nad tím, jaká spojení jsou shlukována dohromady (respektive jaká nejsou v stejné skupině). Kvůli tomu může dojít k problémům s aplikacemi, které najednou využívají víc spojení.

[12] Jako například stahování či video streaming.
[13] Vyrovnávací paměť zařazená mezi dva subsystémy s různou rychlostí, zde použitá k zrychlení routingu.

3.4.3 PCC (Per-connection-classifier)

řeší problémy předchozích dvou přístupů. Je to proprietární funkce RouterOS, která dokáže rozdělit connectiony do skupin a zároveň bere v potaz podle jakých parametrů. Connectiony, které mají společné x, jsou vždy ve stejné skupině.

PCC můžeme nastavit, aby dělilo podle:

- **src-address** (zdrojová adresa - tzn. všechna spojení, která mají stejnou zdrojovou adresu, musí být v stejné skupině),
- **src port** (zdrojový port),
- **src-address and port** (zdrojová adresa a zároveň zdrojový port),
- **dst-address** (cílová adresa),
- **dst port** (cílový port),
- **dst-address and port** (cílová adresa a zároveň cílový port),
- **both adresses** (obě adresy),
- **both ports** (oba porty),
- **both adresses and ports** (obě adresy a zároveň oba porty),

Můžeme si tak vybrat od nejvíce konzervativního **src-adress**, čímž docílíme de facto load-balancingu podle uživatele až po nejvíce volné rozdělení **both adresses and ports**, kterým můžeme dosáhnout nejlepšího rozložení za cenu možných problémů s přerušením spojení (viz Nth). [4]

4 Failover, aneb co stane, když jedno připojení přestane fungovat

Dosud jsme se zabývali pouze tím, jak rovnoměrně rozdělit traffic, aniž bychom zmínili ten hlavní důvod, proč většina lidí potřebuje 2 ISP. Tím je, aby v případě výpadku měli záložní spoj a mohli tak dále fungovat.

Určitě jsme si všimli, že díky parametru check-gateway se v případě, že je brána routy nedostupná, routa sama zakáže. Bylo by však chybou se domnívat, že takové řešení je dostatečné. A to jednoduše z důvodu, že pokud nastane chyba kdekoli mezi cílem routy a koncovým bodem v síti ISP, tak se cesta nezakáže. Z principu next-hop routingu totiž vyplývá, že další skok je dostupný, a tak na něj router pošle pakety, aniž by ho zajímalo, jak budou cestovat dál.

Řešením by pochopitelně bylo připojit náš router do dynamických routovacích protokolů obou našich ISP, aby o nedostupnosti přišla zpráva na náš router a ten se tak nesnažil pakety zbytečně posílat. To bychom však spoléhali na benevolenci ISP a ve většině případů nám to z bezpečnostních důvodů nebude povoleno.

Musíme tedy nalézt jiné řešení, jak zkoumat funkčnost připojení. Nejschůdnější cestou je periodicky kontrolovat (pomocí ICMP ping), zda je dostupný server v síti Internet, o kterém víme, že má prakticky 100% uptime[14]. Například DNS servery společnosti Google.com s lehce zapamatovatelnou IP adresou 8.8.8.8.

Toho můžeme dosáhnout buď pomocí funkce **Netwatch** a skriptů nebo alternativně pomocí **rekurzivní routy**.

Případ s funkcí Netwatch je jednoduchý na pochopení - založíme nový netwatch, co bude periodicky kontrolovat dostupnost serveru a pak napíšeme 2 skripty. Jeden, který zakáže danou routu a druhý, který ji povolí. Realizace může být však právě díky skriptování složitější a musíme si dát dobrý pozor a skripty odladit, aby opravdu vykonávaly to, co potřebujeme, a nemohl se stát nějaký problém. Například při vytažení zařízení ze zásuvky v půlce vykonávání skriptu. Alternativně je také třeba dát si pozor na to, aby skripty s novou verzí RouterOS nepřestaly fungovat.

Druhý možný přístup, tedy využití rekurzivní routy se spíše doporučuje do praxe. Princip spočívá v tom, že jako bránu defaultní routy pro veškerý traffic nastavíme přímo adresu, kterou chceme kontrolovat (tedy například 8.8.8.8), a o její vypnutí se postará parametr **check-gateway**. Samozřejmě pokud bychom udělali jenom toto, tak router zahlásí, že neví kde se 8.8.8.8 nachází, takže je routa nedostupná. Proto musíme vytvořit ještě druhou routu, pomocí které routeru řekneme, kde může 8.8.8.8 nalézt (tzn. bránu této routy nastavíme na adresu, která odpovídá původní Default Gateway). A aby tuto routu vůbec bral router v potaz, tak její parametr **scope** musíme nastavit na 10 nebo méně (tzn. jako kdyby brána této routy bylo přímo dostupná na linkové vrstvě).

[14] Doba bez výpadku.

Router pak pakety bude posílat na naši skutečnou gateway s tím, že se bude domnívat, že poputují přes 8.8.8.8 (což se díky next-hop routingu naštěstí nestane) a pakety se dostanou normálně do cíle. [8]

5 Popis použitého hardware

5.1 MikroTik SXT Lite2

SXT Lite2 je venkovní jednotka, vytvořena pro ISP.

RouterBoard má dvě rozhraní - bezdrátový modul pásma 2.4Ghz podporující technologii 2x2 MIMO 802.11 N a 1 100Mbit/s Ethernet port, který podporuje PoE.

Zařízení obsahuje procesor Atheros AR9344 600MHz CPU a 64MB DDR2 RAM.

K zařízení je dodávána Level 3 licence RouterOS. [9]

Testování bylo prováděno na RouterOS verze 6.20.

Obrázek 10 – MikroTik SXT [9]

5.2 MikroTik RouterBoard 750 GL

RouterBoard RB750GL je kompaktním routerboardem s pěti Gigabit Ethernet porty. Je to ideální zařízení pro malé podnikové či domácí sítě. Mimo jiné zařízení podporuje funkcionalitu hardwarového switche (master-port) a PoE.

Mozkem RouterBoardu je procesor Atheros AR7242 o frekvenci 400MHz, vybavený 64 MB DDR paměti. K zařízení je dodávána RouterOS licence Level 4. [10]

Testování bylo prováděno na RouterOS verze 6.20.

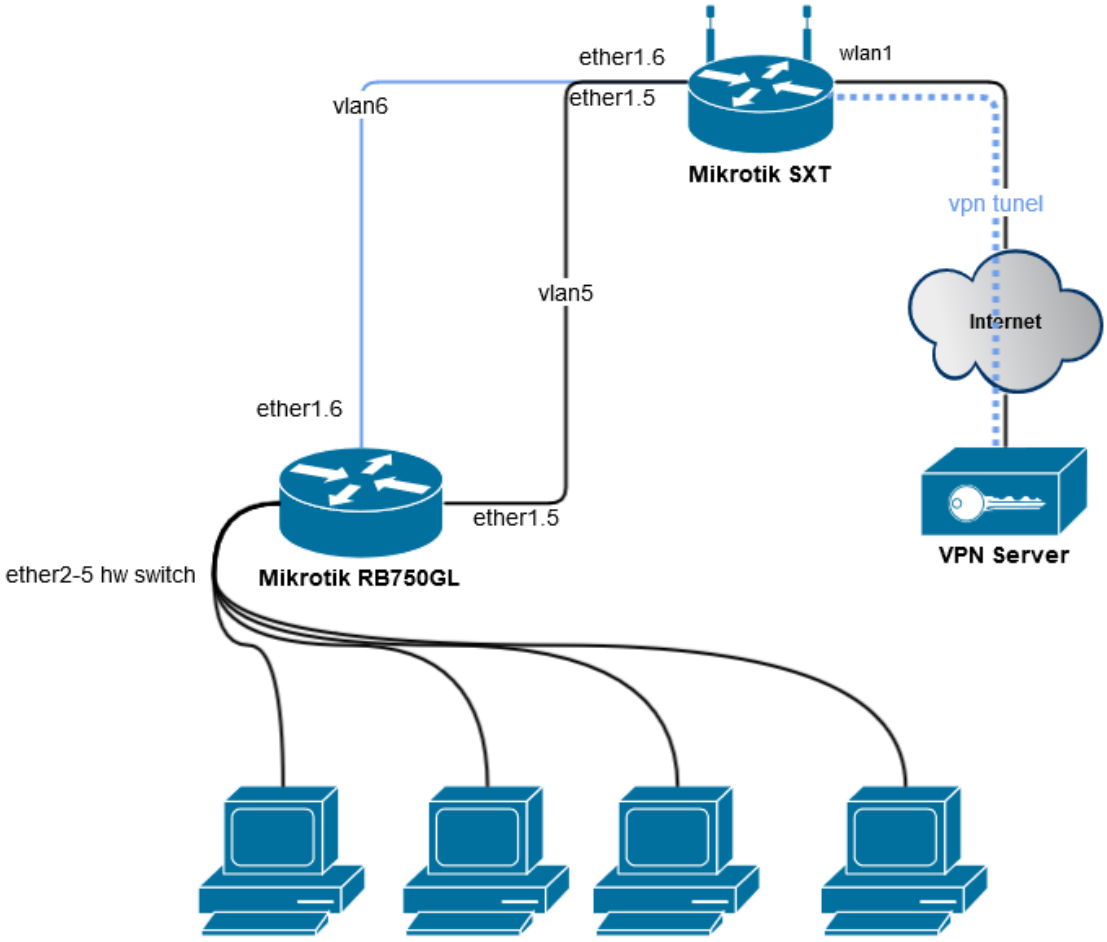
Obrázek 11 – MikroTik RB750GL [10]

5.3 PC

Testování bylo prováděno na dvou PC opatřených Gigabitovou síťovou kartou a operačním systémem Windows 8.1.

6 Popis testovací soustavy

Tato soustava slouží k simulaci situace, kdy máme router s dvěma rozhraními připojenými do dvou sítí různých ISP a třetí rozhraní sloužící jako síť LAN.



Obrázek 12 – Diagram testovací soustavy (Zdroj: vlastní)

Připojení do sítě ISP je skrz bezdrátové rozhraní wlan1 na MikroTik SXT. Na tomto zařízení jsou zároveň vytvořené 2 vnitřní IP rozsahy :

- 192.168.5.0/24 pro sub-interface ether1.5
- 192.168.6.0/24 pro sub-interface ether1.6

Sub-interface a VLANy jsou použity, aby bylo možné přenést více nezávislých sítí přes jeden Ethernet port (více jich bohužel MikroTik SXT nemá).

První z těchto rozsahů je překládán funkcí NAT přímo na adresu přidělenou od ISP a následně routován na Default Gateway přidělenou od ISP. Pomocí funkce Queue je omezena kapacita rozhraní na symetrických 5 Mb/s.

Všechny pakety z druhého rozsahu jsou funkcí Mangle označkovány routing-markem a routovány skrz VPN tunel. VPN tunel je v tomto příkladu použit, abychom docílili

různých cest paketů a především různých zdrojových adres tak, jak by se stalo v reálném provozu. VPN tunelování je docíleno pro jednoduchost konfigurace přes dnes již nepříliš bezpečný PPTP (Point-to-Point-Tunneling-Protocol). Stejně jako u prvního je i na tomto rozhraní omezena kapacita na symetrických 5 Mbit/s.

Na obou rozhraních také běží DHCP server, který automaticky přidělí adresy na RB750GL.

Na RB750GL pochopitelně také běží sub-interface, VLANY a DHCP klient na obou sub-rozhraních.

Síť LAN tvoří zbývající Ethernet porty (2-5), které jsou spojeny pomocí funkcionality hardwarové switche (ether2 se stává tzv. "master portem").

Na tomto rozhraní je vytvořen další rozsah **192.168.7.0/24** s NATem a odpovídajícím DHCP serverem. Na zařízení také běží DNS server, který přeposílá dotazy na servery přidělené od ISP.

Nastavení RouterBoard 750GL tak odpovídá nastavení, které bychom měli v reálném případě dvou různých ISP. [15]

[15] Až na malý rozdíl v absenci VLAN a sub-rozhraní. V reálném případě by jiné sítě mohly být pravděpodobně dostupné skrz dvě různá rozhraní Ethernet.

7 Konfigurace testovací soustavy

7.1 MikroTik SXT Lite2

Pozn.: Nastavení bezdrátové části a věci ponechané v továrním nastavení byly vynechány.

Nastavení IP adres. Vyznačená část obsahovala adresy přidělené od ISP.

```
/ip address
add address=XXX.XXX.XXX.XXX/XX interface=wlan1-gateway
network=XXX.XXX.XXX.XXX

add address=192.168.5.1/24 interface=testVLAN1-5
network=192.168.5.0

add address=192.168.6.1/24 interface=testVLAN2-6
network=192.168.6.0
```

Nastavení DHCP serveru. DNS servery od ISP byly vynechány.

```
/ip dhcp-server network
add address=192.168.5.0/24 dns-server= XXX.XXX.XXX.XXX,
XXX.XXX.XXX.XXX gateway=192.168.5.1

add address=192.168.6.0/24 dns-server= XXX.XXX.XXX.XXX,
XXX.XXX.XXX.XXX gateway=192.168.6.1

/ip pool
add name=dhcp_pool5 ranges=192.168.5.2-192.168.5.254
add name=dhcp_pool6 ranges=192.168.6.2-192.168.6.254

/ip dhcp-server
add address-pool=dhcp_pool5 disabled=no interface=testVLAN1-
5 name=dhcp5
add address-pool=dhcp_pool6 disabled=no interface=testVLAN2-
6 name=dhcp6
```

Konfigurace virtuálních interface (VLAN + PPTP klient). Vyznačená část obsahuje IP adresu VPN serveru a přihlašovací údaje.

```
/interface vlan
add interface=ether1-local l2mtu=1594 name=testVLAN1-5 vlan-
id=5
add interface=ether1-local l2mtu=1594 name=testVLAN2-6 vlan-
id=6

/interface pptp-client
add add-default-route=no allow=pap,chap,mschap1,mschap2
connect-to=XXX.XXX.XXX.XXX dial-on-demand=no disabled=no
keepalive-timeout=60 max-mru=1450 max-mtu=1450 mrru=1600
name=pptp-out1 password=XXXXXXXX profile=default
user=XXXXXXXXXX
```

Nastavení omezování kapacity jednotlivých rozhraní.

```
/queue simple
add max-limit=5M/5M name=queue1 target=testVLAN1-5
add max-limit=5M/5M name=queue2 target=testVLAN2-6
```

Každý paket na chainu prerouting, který přišel z interface `testVLAN2-6`, bude označen routing-markem `VPN`.

```
/ip firewall mangle
add action=mark-routing chain=prerouting in-
interface=testVLAN2-6 new-routing-mark=VPN
```

Nastavení S-NATU.

```
/ip firewall nat
add action=masquerade chain=srcnat comment="vlan5" src-
address=192.168.5.0/24

add action=masquerade chain=srcnat comment="vlan6" src-
address=192.168.6.0/24
```

Přidání dvou default-gateway. Jedné pro pakety označené routing-markem `VPN` a druhé pro zbytek.

```
/ip route
add distance=1 gateway=YYY.YYY.YYY.YYY routing-mark=VPN
add distance=1 gateway=XXX.XXX.XXX.XXX
```

7.2 MikroTik RouterBoard 750GL

Základní nastavení RB750GL. Firewallová pravidla (Mangle) a routy se budou lišit v závislosti na příkladech load-balancingu.

První odstavec je nastavení hardwarové switche, následuje VLAN rozhraní, IP adresy, DHCP klienti na VLAN rozhraních, nastavení DHCP serveru, DNS (jednotlivé servery, které sdružujeme, jsou přidány dynamicky z DHCP) a nakonec nastavení SNATu.

```
/interface ethernet
set [ find default-name=ether1 ] comment=wan
set [ find default-name=ether2 ] comment="eth2-5 hw switch"
set [ find default-name=ether3 ] master-port=ether2
set [ find default-name=ether4 ] master-port=ether2
set [ find default-name=ether5 ] master-port=ether2

/interface vlan
add interface=ether1 l2mtu=1594 name=vlan5 vlan-id=5
add interface=ether1 l2mtu=1594 name=vlan6 vlan-id=6

/ip address
add address=192.168.7.1/24 interface=ether2
network=192.168.7.0

/ip dhcp-client
add add-default-route=no dhcp-options=clientid,hostname
disabled=no interface=vlan5

add add-default-route=no dhcp-options=hostname,clientid
disabled=no interface=vlan6

/ip pool
add name=dhcp_pool3 ranges=192.168.7.2-192.168.7.254

/ip dhcp-server
add address-pool=dhcp_pool3 disabled=no interface=ether2
name=dhcp1

/ip dhcp-server network
add address=192.168.7.0/24 dns-server=192.168.7.1
gateway=192.168.7.1

/ip dns
set allow-remote-requests=yes

/ip firewall nat
add action=masquerade chain=srcnat src-
address=192.168.7.0/24
```

8 Implementace load-balancingu a testování

K testování byly vybrány tři v praxi nejpoužitelnější příklady.

8.1 Příklad 1 : Load-balancing podle uživatele

Přidáme dvě nové default gateway, každou s jiným routing-markem. Funkcí mangle označíme jednotlivé pakety routing-markem na základě toho, z jaké IP adresy pochází. Pro jednoduchost byly IP adresy obou hostů napevno nastaveny, ale klidně bychom mohli použít celé rozsahy, nebo funkci firewallu Address List a pak odpovídajícím způsobem nastavit DHCP server.

```
/ip route
add distance=1 gateway=192.168.5.1 routing-mark=isp1
add distance=1 gateway=192.168.6.1 routing-mark=isp2

/ip firewall mangle
add action=mark-routing chain=prerouting new-routing-
mark=isp1 passthrough=no src-address=192.168.7.254

add action=mark-routing chain=prerouting new-routing-
mark=isp2 passthrough=no src-address=192.168.7.253
```

8.2 Příklad 2: Load-balancing podle typu trafficu

V tomto příkladu budeme dělit traffic na HTTP (port 80) a zbytek. Docílíme toho jednoduchým Mangle pravidlem, pomocí kterého přidáváme routing-mark isp1 všem paketům, jejichž TCP/UDP hlavička obsahuje port 80. V druhém pravidle nastavujeme routing-mark isp2 všem ostatním paketům. Díky parametru passthrough se nemůže stát, že by paket, jenž už dostal routing-mark isp1, mohl dostat routing-mark isp2.

Důležitou věcí je, že označkováváme jen pakety, které přišly z interface ether2 (síť LAN), jinak bychom značkovali i všechny příchozí pakety.To by bylo zbytečné a vyžadovalo by to přidání zpětných rout do routovacích tabulek.

```
/ip route
add distance=1 gateway=192.168.5.1 routing-mark=isp1
add distance=1 gateway=192.168.6.1 routing-mark=isp2

/ip firewall mangle
add action=mark-routing chain=prerouting in-interface=ether2
new-routing-mark=isp1 passthrough=no port=80 protocol=tcp

add action=mark-routing chain=prerouting in-interface=ether2
new-routing-mark=isp2 passthrough=no
```

8.3 Příklad 3: Load-balancing pomocí PCC

Na závěr si ukážeme nejzajímavější případ, a to využití funkce Per-connection classifier. Zde už máme pravidel trošku více, tak si je popíšeme podrobněji.

Toto je srdcem naší konfigurace. První pravidlo vezme všechny pakety procházející chainem prerouting, které přišly z rozhraní ether2 (síť LAN) a **ne**směřují do lokální sítě. Pomocí funkce PCC rozdělíme connectiony, jimž tyto pakety náleží, na **dvě** poloviny. První polovinu (tedy jazykem informatika tu na pozici **0**) označkujeme. (V případě, že bychom měli nestejně rychlá spojení, mohli bychom použít dělení na menší díly.)

Pomocí druhého pravidla pak vezmeme označkované connectiony a všechny pakety v nich označíme routing-markem.

Následující dvě pravidla jsou jen kopií těch prvních a starají se o druhou polovinu connectionů.

```
/ip firewall mangle
add action=mark-connection chain=prerouting dst-address-
type=!local in-interface=ether2 new-connection-mark=isp1
per-connection-classifier=both-addresses-and-ports:2/0

add action=mark-routing chain=prerouting connection-
mark=isp1 new-routing-mark=isp1

add action=mark-connection chain=prerouting dst-address-
type=!local in-interface=ether2 new-connection-mark=isp2
per-connection-classifier=both-addresses-and-ports:2/1

add action=mark-routing chain=prerouting connection-
mark=isp2 new-routing-mark=isp2
```

Toto samo o sobě by však bylo nedostatečné, protože bychom označkovávali i všechny pakety, které sice náleží connectionu, ale míří na náš router. Proto musíme přidat další dvě

pravidla, která musíme dát ještě PŘED ta minulá. Tato pravidla vezmou všechny pakety, jejichž cílem je adresa routru (raději celý rozsah, vzhledem k tomu, že adresu dostáváme z DHCP) a akceptují je (tedy povolí a dál se s nimi firewall nezabývá).

```
add chain=prerouting dst-address=192.168.6.0/24
add chain=prerouting dst-address=192.168.5.0/24
```

Na závěr ještě zmíníme čtyři pravidla, která nejsou naprosto nutná, ale jsou užitečná. Bez nich by nám nefungoval vzdálený přístup na lokální procesy routru (např. webové rozhraní či SSH).

První dvě zachytí pakety, které směřují na lokální proces routru (chain input) a označí jejich spojení. Další dvě se starají o to, aby až na ně budou lokální procesy odpovídat (chain output), byly tyto pakety routovány stejnou cestou (přidají routing-mark). (Pozn.: Vzhledem k tomu, že chain input následuje až po chainu prerouting, je třeba první dvě pravidla vložit úplně na začátek, jinak by tyto pakety zachytil předchozí accept.)

```
add action=mark-connection chain=input in-interface=vlan5
new-connection-mark=isp1

add action=mark-connection chain=input in-interface=vlan6
new-connection-mark=isp2

add action=mark-routing chain=output connection-mark=isp1
new-routing-mark=isp1

add action=mark-routing chain=output connection-mark=isp2
new-routing-mark=isp2
```

Samotná část s routami už není nijak zajímavá, jen je třeba přidat i třetí routu pro pakety bez routing-marku, tedy pro ty, které nebyly nijak označeny[16].

```
/ip route
add distance=1 gateway=192.168.5.1 routing-mark=isp1
add distance=1 gateway=192.168.6.1 routing-mark=isp2
add distance=1 gateway=192.168.5.1
```

8.4 Testování

Už z pohledu na rozličnou složitost jednotlivých konfigurací a počtu Firewall pravidel vyvstává otázka, jak se s tímto load-balancingem dokáže vypořádat relativně levný RouterBoard.

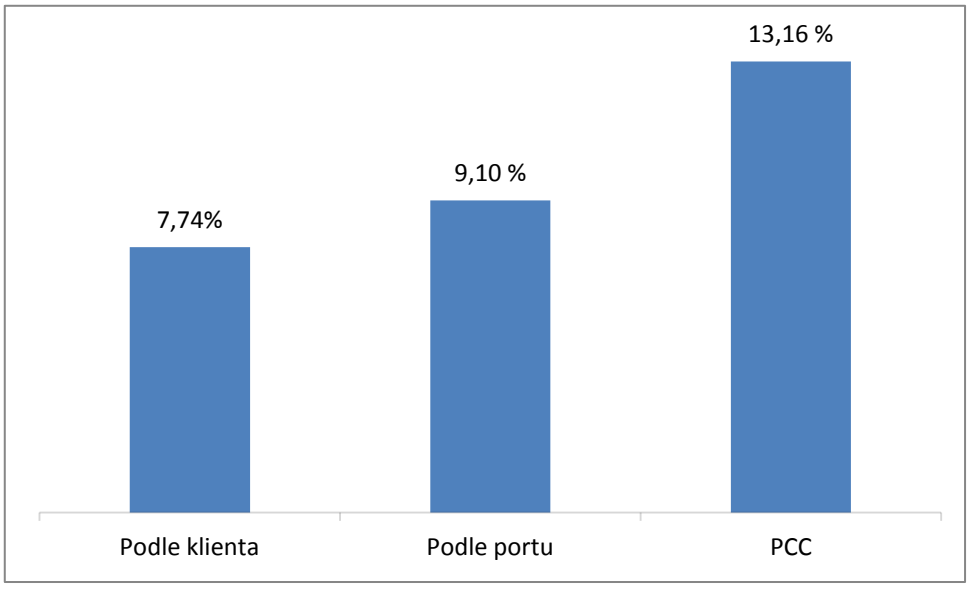
[16] Například pakety vyslané přímo routerem.

Tím nejdůležitějším parametrem, který nás bude zajímat, je využití CPU, díky kterému bychom velmi jednoduše identifikovali úzké hrdlo celého systému.

Měření probíhalo následujícím způsobem : Na dvou PC byl zároveň zapnut síťový provoz schopný naplnit více connections. Na prvním budeme stahovat soubor pomocí BitTorrent protokolu, na druhém pomocí protokolu HTTP. Po několika sekundách, kdy se procesy na plno rozběhly a byly schopny vytěžit ze systému sto procent povolené konektivity, byl zapnut plánovač. Ten každých 10 sekund zaznamenával současné využití CPU do souboru pomocí jednoduchého skriptu.

```
system resource monitor file=soubor.txt append
```

Po deseti minutách byl test ukončen a z hodnot vypočtena průměrná zátěž CPU. V grafu můžete vidět výsledné hodnoty.



Obrázek 13 – Využití CPU (Zdroj: vlastní)

Dále proběhlo testování latence ve vnitřní síti, a to pomocí příkazu ping směrem na MikroTik SXT. Zde však nebyl žádný měřitelný rozdíl. Stejně tak nebyl měřitelný rozdíl v rychlosti - v době našeho měření bylo dosaženo ideálního rozložení trafficu ve všech příkladech.[17]

Z výsledků měření vyplývá, že naše zařízení si dokáže s takovouto zátěží hravě poradit a rozdíly mezi jednotlivými implementacemi jsou minimální. Není tedy důvod neimplementovat složitější metodu PCC, která vede k nejideálnějším výsledkům při menším množství uživatelů.

[17] Přesně tak byl nastaven náš test. Platí pochopitelně omezení vysvětlené v teoretické části. (Pokud bychom tak například testovali stahování na obou PC stejným protokolem, dosáhli bychom v příkladu 2 pouhých ~50%)

9 Implementace fail-over

Na závěr do našeho routru ještě přidáme ochranu proti pádu jednoho ze spojení. K implementaci použijeme v předchozích kapitolách zmíněné reverzní routy. Nastavení jednotlivých routerů se bude lišit minimálně.

Servery, pomocí kterých budeme testovat funkčnost internetu, jsou :

- 8.8.8.8 (DNS server www.google.com)
- 221.132.112.8 (DNS server ISP)

Nejdříve vytvoříme 2 routy, kterými specifikujeme, jakým směrem leží tyto servery. Důležité je nastavit parametr scope na 10 (nebo níže), aby je router bral v potaz.

```
/ip route
add distance=1 dst-address=8.8.8.8/32 gateway=192.168.6.1
scope=10
add dst-address=221.132.112.8/32 gateway=192.168.5.1
scope=10
```

Pak už stačí jen nahradit staré brány (gateway) rout těmito novými, v každé routovací tabulce přidat i záložní routu s nižší prioritou (vyšším parametrem distance) a nastavit metodu parametru **check-gateway** na ping.

```
/ip route
add check-gateway=ping distance=1 gateway=221.132.112.8
routing-mark=isp1
add check-gateway=ping distance=2 gateway=8.8.8.8 routing-
mark=isp1

add check-gateway=ping distance=1 gateway=8.8.8.8 routing-
mark=isp2
add check-gateway=ping distance=2 gateway=221.132.112.8
routing-mark=isp2
```

Závěr

V práci byl nastíněn základní pohled na protokol TCP/IP a především na funkcionalitu a možnosti platformy MikroTik, která je díky své ceně a stabilitě ideálním zařízením do SOHO prostředí.

Popis protokolů nezabíhá do detailů a celá práce je psaná tak, aby jí dokázal porozumět i laik. Díky tomu může práce sloužit k pochopení základů síťového provozu a konceptů load-balancingu, ale zároveň může sloužit i jako příručka k pochopení základních funkcí firewallu a routování v operačním systému RouterOS pro pokročilejší uživatele.

Takový uživatel by po přečtení této práce měl být bez problémů schopen zvolit a následně implementovat vhodný druh load-balancingu pro danou síť.

Testy provedené v práci navíc ukázaly, že i levný RouterBoard je schopný si bez problémů poradit se složitějšími formami load-balancingu a díky tomu cena implementace může zůstat velmi nízká v porovnání s přidanou hodnotou či případnými ztrátami způsobenými výpadky.

Při pozorování současných trendů můžeme vidět, že více a více malých firem se stává na internetu doslova závislými a nemohou si dovolit žádné výpadky. Tato práce je řešením tohoto problému a s load-balancingem a zálohovaným připojením se tak v příštích letech pravděpodobně budeme setkávat ve více a více malých kancelářích.

Literatura

[1] ODOM, Wendell, Rus HEALY a Naren MEHTA. *Směrování a přepínání sítí: autorizovaný výukový průvodce*. Vyd. 1. Brno: Computer Press, 2009, 879 s. Samostudium. ISBN 978-80-251-2520-5.

[2] PURDY, Gregor N. *Linux iptables: pocket reference*. Sebastopol, CA: O'Reilly, c2004, iii, 91 p. ISBN 0596005695.

[3] KOZIEROK, Charles M. *The TCP/IP guide: a comprehensive, illustrated Internet protocols reference*. San Francisco: No Starch Press, c2005, lxxiv, 1539 p. ISBN 159327047x.

[4] Manual, 2013. MikroTik. [online]. [cit. 2015-04-30]. Dostupné z http://wiki.mikrotik.com/wiki/Category:Manual

[5] MikroTik, 2014 [online].[cit. 2015-04-30]. Dostupné z: http://www.mikrotik.com/

[6] MikroTik RouterOS Feature catalog. Q1-Q2 2010 RouterOS. MIKROTIK. *MikroTik* [online]. 2010 [cit. 2015-04-30]. Dostupné z: http://www.mikrotik.com/pdf/what_is_routeros.pdf

[7] MikroTik RouterOS. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2015-04-30]. Dostupné z: http://cs.wikipedia.org/wiki/MikroTik_RouterOS

[8] MikroTik Multi WAN Fail Over Scenarios. In: JAHANZAIB, Syed. *Syed Jahanzaib Personnel Blog to Share Knowledge !* [online]. 2013 [cit. 2015-04-30]. Dostupné z: https://aacable.wordpress.com/2013/04/12/mikrotik-multiple-wan-fail-over-scripts/

[9] SXT 2. MIKROTIK. *Routerboard.com* [online]. 2014 [cit. 2015-05-04]. Dostupné z: http://routerboard.com/RBSXTG2HnD

[10] RB 750GL. MIKROTIK. *Routerboard.com* [online]. 2014 [cit. 2015-05-04]. Dostupné z: http://routerboard.com/RB750GL

[11] Internet protocol suite. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2015-04-30]. Dostupné z: http://en.wikipedia.org/wiki/Internet_protocol_suite

[12] IP OPTION NUMBERS. *Internet Assigned Numbers Authority* [online]. 2013 [cit. 2015-04-30]. Dostupné z: http://www.iana.org/assignments/ip-parameters/ip-parameters.xhtml

Obsah CD

Obsah přiloženého CD:

- soubor `PleskacR_MikroTikLoad-balancing_SN_2015.pdf` – elektronická verze práce