Internetový firewall založený na filtrování paketů

Radek Dvořáček

Bakalářská práce
2007

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav aplikované informatiky
akademický rok: 2006/2007

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Radek DVOŘÁČEK**
Studijní program:	**B 3902 Inženýrská informatika**
Studijní obor:	**Informační technologie**
Téma práce:	**Internetový firewall založený na filtrování packetů**

Zásady pro vypracování:

1. **Firewally obecně, popis a funkce, jejich provoz a nastavení v OS LINUX.**
2. **Použití a nastavení iptables.**
3. **Tvarování toku (provozu) v síti – shaping.**
4. **Blokování provozu sítí peer to peer. Překlad síťových adres a portů, maskování – (NAT) pro přístup na internet např. při nedostatku věřejných IP adres.**
5. **GUI aplikace a nádstavby pro nastavení firewallu pod OS Linux.**
6. **Nastavení a provoz iptables pro malou síť v OS Linux.**

Rozsah práce:

Rozsah příloh:

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

Seznam odborné literatury:

1. Matthew Strebe, Charles Perkins: Firewally a proxy-servery Praktický průvodce. Computer Press Brno, 2003.
2.Dilip C. Naik: INTERNET - standardy a protokoly. Computer Press, Brno, 1999.
3.Libor Dostálek: Velký průvodce protokoly TCP/IP - Bezpečnost. ComputerPress, Praha, 2001.
4. World Wide Web:
http://www.netfilter.org
http://www.root.cz
http://www.abclinuxu.cz

Vedoucí bakalářské práce:	**Ing. Martin Sysel, Ph.D.** Ústav aplikované informatiky
Datum zadání bakalářské práce:	**13. února 2007**
Termín odevzdání bakalářské práce:	**24. května 2007**

Ve Zlíně dne 13. února 2007

prof. Ing. Vladimír Vašek, CSc.
děkan

L.S.

doc. Ing. Ivan Zelinka, Ph.D.
ředitel ústavu

ABSTRAKT

Cílem této bakalářské práce je osvětlení a praktická implementace firewallu na bázi filtrování paketů spolu s překladem adres v malé síti (malá a střední firma), která je připojena k Internetu, za pomoci operačního systému Linux a jeho iptables. Dále pak také využití možností jako je blokování provozu sítí peer to peer a tvarování toku v síti. V neposlední řadě ukázka nástrojů pro snadnější nastavování firewallu přes grafická rozhraní v operačním systému Linux.

Klíčová slova: firewall, překlad adres, iptables, netfilter, tvarování toku

ABSTRACT

The main point of this work is to show how firewall based on packet filtering with network address translation works. Furthermore, this firewall works in a small network which is connected to the internet trought IPtables implemented in Linux. In addition, I would like to describe the usage of shaping technique for traffic control especially for peer to peer networks. The last but not least I would like to show the possibilities of graphical user interface used for setting up the firewall under the Linux.

Keywords: firewall, NAT, IPtables, netfilter, shaping

POĎEKOVÁNÍ

Děkuji svému vedoucímu bakalářské práce Ing. Martinu Syslovi, Ph.D. za trpělivost, užitečnou metodickou pomoc a cenné rady při zpracování mé bakalářské práce.

Motto

„Ten kdo se ve jménu bezpečnosti vzdává svobody, nezaslouží si ani svobodu ani bezpečnost.“

Benjamin Franklin

OBSAH

ÚVOD

Celosvětová počítačová síť Internet se v posledním desetiletí rozšířila do všech oborů lidské činnosti. Počítače se staly nezbytnou pracovní pomůckou a připojení k internetové síti je již obvyklé v téměř každé domácnosti. Dle posledních statistických údajů využívá Internet přes 1.1 miliardy populace planety [1].

Internet je založen na komunikačních protokolech TCP/IP, které byly navrženy v osmdesátých letech minulého století. V této době se nepředpokládalo jejich masivní nasazení a proto neobsahují téměř žádné bezpečnostní prvky. Ruku v ruce s rozvojem internetu je tedy nutné rozvíjet technologie zajišťující bezpečnost provozu počítačových sítí. Otevírá se tak nový svět nástrojů zajištujících plynulý chod počítačových sítí, ale i počítačů a zařízení samotných.

Ve skutečnosti se tedy snažíme dosáhnout co nejvyšší úrovně zabezpečení a zamezit tak proniknutí či úniku informací z vnitřní sítě, ale i samotného počítače. Existuje celá řada bezpečnostních technik a programů snažících se dosáhnout tohoto cíle. Vedle šifrované komunikace, programů zabraňujících průniku virů a spyware do počítače, antispamových programů zabraňujícím příjmu nevyžádané pošty, automatického stahování a instalace bezpečnostních záplat operačního systému je právě firewall základním stavebním prvkem zabezpečení počítače. Je obecně známo, že počítač přímo připojený k internetové síti bez aktivního firewallu a antivirového programu je schopen správně fungovat jen několik desítek minut.

Základní firewall, tvořící zavřené dveře mezi vnitřní privátní sítí nebo samotným počítačem a internetem, je dnes obsažen víceméně v každém operačním systému. Vzhledem k tomu, že se jedná o bezpečnostní technologii je nespornou výhodou otevřenost zdrojového kódu. Právě operační systém Linux se svým robustním a výkoným firewallem je toho dobrým příkladem. Využitelnost tohoto operačního systému jako kvalitního firewallu s mnoha možnostmi stoupá s jeho snadnou dostupností a cenou.

Svět firewallů a bezpečnostních technologií a postupů, který je zde částečně popsán bude bezesporu i nadále velmi důležitým a rozvíjejícím se prvkem v oblasti počítačových sítí a komunikace vůbec. Právě bezpečnost a ochrana elektronicky zpracovávaných dat narůstá na významu spolu s prudkým rozvojem informačních a komunikačních technologií. Je třeba si uvědomit, že možnost připojení na Internet obsahuje již téměř každý mobilní telefon. Jak daleko je doba, kdy si právě Vaše automatická pračka bude stahovat z této veřejné sítě program na praní nových typů textilií a Vaše lednice objednávat potraviny?

I. TEORETICKÁ ČÁST

1 FIREWALLY

Internet je výkoným prostředkem komunikace mezi jednotlivými počítači i celými sítěmi. Postupem času se k internetu připojuje celá řada privátních síti, které jsou tak propojeny z veškerým okolním světem a dalšími sítěmi. Nabízí se tak možnost průniku do těchto sítí, zneužití a krádež informací z privátních sítí hackery. Pro zabezpečení privátních sítí i osobních počítačů se používají firewally coby kontrolní body mezi jednotlivými privátními sítěmi a internetem [2].

1.1 Historie

Je třeba si uvědomit, že dříve nebylo elektronických komunikačních prostředků. Nebyla možnost zasílání emailů a rychlé elektronické komunikace. Lidé využívali poštovních a telefonních služeb. Internet vznikal postupně od 70 let jako projekt pod názvem ARPANET, zadaný vládou USA agentuře ministerstva obrany Advanced Research Project Agency (ARPA). Jednalo se o experimentální síť vytvořenou za účelem komunikace vládních organizací bez větších zabezpečovacích prvků. V devadesátých letech švýcarský institut pro jaderný výzkum CERN vytvořil první hypertex a dal tak základ internetu s webovými stránkami tak jak jej dnes chápeme. V roce 1984 bylo k internetu připojeno pouhých 1000 počítačů [3][4].

Významná změna nastala po události z 2.listopadu 1988. Tato událost měla změnit Internet navždy. Peter Yee z výzkumného centra NASA Ames oznámil na mailing listu TCP/IP: „We are currently under attack from an Internet VIRUS!“ (Právě jsme pod útokem internetového viru!). Tento červ vypuštěný z americké univerzity MIT, později nazvaný jako Moris Worm, byl prvním známým červem šířícím se po internetu a tudíž také první známkou nedostatečného zabezpečení internetu. Výsledkem byla především větší snaha o zvýšení bezpečnosti na internetu [3][4].

Slovo firewall je převzato z anglického originálu. V běžném užití se jedná o protipožární ochrannou bariéru užívanou zejména ve stavitelství. Dá se říci, že se jedná o „zeď mezi námi a těmi ostatními“. V tomto pojetí funguje internetový firewall, který odděluje privátní sítě a Internet a kontroluje komunikaci mezi nimi [4].

První firewally sloužily pouze k oddělení jednotlivých privátních sítí LAN. Pravděpodobně prvním publikovaným bezstavovým komerčním firewallem byl firewall od firmy DEC v roce 1988. jednalo se o firewall založený na filtrování paketů. Bezpečnostní politika firewallu byla prostá: Povol všem „uvnitř“ přístup „ven“ a zabraň komukoliv „zvenku“

přístup „dovnitř“. Jednalo se o velmi účinný, ale limitovaný bezpečnostní systém [4][5].

Následovala druhá generace firewallů, založených již na filtrování paketů s kontrolou stavu, vyvíjených společností AT&T. Později se objevují firewally aplikační. Rozvoj techologií firewallů se samozřejmě nezastavil a pokračuje s rozvojem internetu dál. [5].

1.2 Základní pojmy

Pro pochopení firewallů, jejich možností a principů je vhodné znát základní pojmy z oblasti počítačových sítí. Podrobněji lze dohledat v literatuře např. v [6][7].

IP protokol (Internet protocol)

IP protokol společně s protokolem TCP tvoří základ dnešního internetu. Jedná se o protokol sloužící k přenosu dat v síti. Přenos je rozdělen do bloků zvaných datagramy (analogie se slovem telegram). Každý datagram je přenášen samostatně a nezávisle na ostatních. Nejčastěji používaná verze protokolu je IPv4. Nástupcem je verze IPv6. Verze protokolu úzce souvisí s adresací (viz. dále pojem *IP adresa*)[8].

Paket, datagram

Paket je základní jednotkou přenosu v moderních počítačových sítích. Paket se skládá ze tří základních prvků a to, hlavičky, datové oblasti a traileru. Hlavička obsahuje informace potřebné pro doručení paketu do místa určení. Datová oblast slouží k přenosu dat. Trailer označuje konec paketu a obsahuje kontrolní součet. Kontrolní součet slouží k odhalení nežádoucích změn během přenosu paketu v síti[8].

Paket zajišťuje takzvaný spolehlivý přenos dat v síti. Spolehlivým přenosem je míněno zajištění kontroly přenosu. Odesílatel i příjemce se dozví o případné chybě při přenosu. V případě datagramu (IP datagram) se jedná o takzvaný nespolehlivý přenos. Příjemce ani odesílatel není informován o případném odstranění poškozeného datagramu. Spolehlivost se u IP protokolu zajišťuje pomocí protokolu v bezprostředně vyšší vrstvě síťové architektury. Typicky pomocí protokolu TCP[8].

IP adresa

IP adresa slouží k jednoznačné identifikaci zařízení v prostředí internetu. Veškerá data, která jsou zasílána přes počítačovou síť ve formě paketů obsahují IP adresu. Dle použité verze protokolu jsou adresy typu IPv4 nebo IPv6. Nejčastěji používaná je verze IPv4 [9].

Protokol IPv4 používá 32-bitové adresování. Z toho vyplývá možnost použít 2^{32} IP adres, z čehož některé jsou vyhrazeny pro privátní sítě a některé slouží k všesměrovému vysílání. Adresa IPv4 se skládá ze tří základních částí a to, adresy sítě, adresy podsítě a adresy počítače (zařízení). Adresy jsou rozděleny do tříd. Typicky je IP adresa zapisována dekadicky, např. adresa třídy C – 192.168.1.1. V současné době se využívá NAT pro sdílení veřejných IP adres. Tato metoda v důsledku zpožďuje nástup protokolu IPv6 [9].

IP adresa protokolu IPv6 řeší problém nedostatečně velkého adresního prostoru IPv4. Tato adresa je již 128-bitová. Adresa IPv6 se zapisuje jako osm skupin po čtyřech hexadecimálních číslicích, např. 2001:0718:1c01:0016:0214:22ff:fec9:0ca5 [9].

1.2.1 Referenční modely počítačových sítí

Referenční modely jsou používány k popisu komunikace v počítačových sítích a Internetu. K nejpoužívanějším modelům patří TCP/IP a ISO/OSI model. V této bakalářské práci je odkazováno nejčastěji na model ISO/OSI. Není to z důvodu, že by model TCP/IP byl nedostatečný, ale model ISO/OSI je standardizován normou ISO 7498 [10][11].

TCP/IP

Pravý význam zkratky TCP/IP pochází z anglického Transmission Control Protocol/Internet Protocol. Česky značí primární transportní protokol/protokol síťové vrstvy. V tomto pojetí se jedná o rodinu protokolů TCP a IP, které jsou dnes nejčastěji používané k přenosům dat na Internetu. Referenční model TCP/IP je pojmenován podle nejpoužívanějších internetových protokolů, jimiž jsou TCP/IP [10].

Protokolová architektura TCP/IP je definována sadou protokolů pro komunikaci v počítačové síti, zejména Internetu. Do rodiny TCP/IP řadíme internetové protokoly, jako např. TCP, IP, SMTP, POP3, UDP, FTP [10].

Dle referenčního modelu je síťová komunikace rozdělena do několika vrstev. Výměna informací mezi vrstvami je přesně definována. Každá vrstva využívá služeb vrstvy nižší a poskytuje své služby vrstvě vyšší [10].

Referenční model TCP/IP se skládá ze čtyř vrstev a to, vrstvy síťového rozhraní, internetové vrstvy, transportní vrstvy a aplikační vrstvy. Někdy je mezi fyzickou a internetovou vrstvu uváděna vrstva spojová.

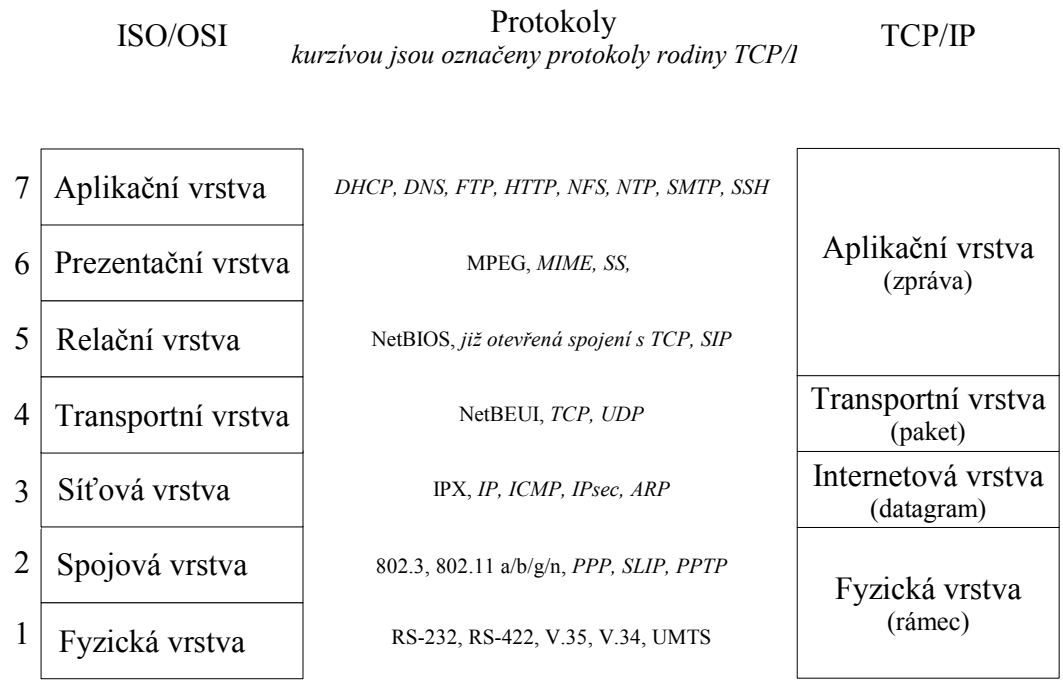
Každá vrstva je specifická svou činností např. síťová vrstva umožňuje přístup k fyzickému médiu. Každé vrstvě přísluší patřičné protokoly. Přehled vrstev modelu je zobrazen na obrázku č. 1 [10].

ISO/OSI

Tento referenční model je vypracován organizací ISO, která jej v roce 1984 přijala jako normu ISO 7498. Na rozdíl od modelu TCP/IP je model ISO/OSI více abstraktní a slouží jako základní norma pro propojování systémů [11].

Referenční model ISO/OSI je založen na sedmivrstvé síťové architektuře. Mezi tyto vrstvy patří vrstva fyzická, spojová, síťová, transportní, relační, prezentační a aplikační. Princip je obdobný jako u modelu TCP/IP [11].

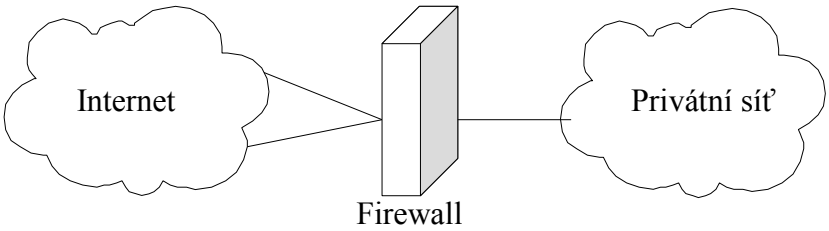
Porovnání obou modelů a popis jednotlivých vrstev je uveden na obrázku č. 1. Jsou zde také uvedeny příklady protokolů v návaznosti na danou vrstvu referenčních modelů [11].



Obr.1. Provnání referenčních modelů ISO/OSI a TCP/IP s uvedením protokolů.

1.3 Funkce firewallu

Firewall je síťové zařízení softwarového nebo hardwarového charakteru, které slouží k zabezpečení a řízení komunikace mezi sítěmi z různou důvěryhodností. Jedná se o kontrolní bod mezi sítěmi, které od sebe může a zpravidla také odděluje. Příkladem může být firewall mezi internetem představujícím zónu s velmi nízkou důvěryhodností a privátní sítí LAN s vysokou důvěryhodností. Názorná ilustrace je na obrázku č.2 [5].



Obr.2. Firewall jako kontrolní bod mezi veřejnou a privátní sítí

Firewally kontrolují a schvalují pokusy o připojení mezi jednotlivými sítěmi. Firewall za pomoci předem vytvořených pravidel pro komunikaci kontroluje tok všech paketů mezi jednotlivými sítěmi. Firewall dle pravidel určí zda pakety propustit, zablokovat nebo zahodit. Robustní firewally mohou chránit síť na všech vrstvách síťového modelu OSI počínaje vrstvou spojovou a konče vrstvou aplikační. Při správném nastavení firewallu se dosahuje vysoké míry zabezpečení sítě a sampozřejmě také bezpečného připojení k internetu [2].

Firewall je zpravidla umístěn na hranici jednotlivých sítí. Představuje tak optimální místo pro služby související se zabezpečením a zároveň tvoří úzké hrdlo mezi sítěmi. Pro internetové poskytovatele a podniky s potřebou obsáhlého síťového provozu jsou vyvíjeny velmi rychlé a robustní firewally. Některé země dokonce za pomocí vysoce výkoných firewallů kontrolují obsah internetu a provádějí cenzuru [2].

Následuje výčet možných funkcí firewallu:

- *Filtrování paketů:* Ve své podstatě firewally provádějí kontrolu na úrovni paketů protokolů TCP/IP. Odmítají pakety od neautorizovaných uživatelů a pokusy o připojení k neautorizovaným službám. Jsou založeny na znalosti z jaké adresy a portu na jakou adresu a port paket prochází. Fungují na třetí a čtvrté vrstvě modelu OSI [2][5].

- *Překládání síťových adres (NAT):* Překládá IP adresy hostů v interní síti a odstiňuje a skrývá je tak před monitorováním zvenčí. Vytváří tak účinnou bariéru mezi jednotlivými sítěmi. Používá se také při nedostatku veřejných IP adres kdy se jedna IP adresa sdílí hostitelům v interní síti [2].

- *Proxy:* Proxy jsou jinak nazývány jako aplikační brány. Zcela oddělují hostitele mezi interními a externími sítěmi a to až na úrovni sedmé aplikační vrstvy modelu OSI [5].

- *Šifrovaná autentizace:* Povoluje vstup do interní sítě na základě autentizace uživatelů externích sítí [2].

- *Propojování VPN:* Ustavuje spojení mezi dvěma privátními sítěmi přes Internet takzvanými zašifrovanými tunely. Fyzicky oddělené sítě tak mohou namísto pronajatých linek využívat internet. Zpravidla se využívá výhody šifrované komunikace mezi privátními sítěmi [2].

- *Rozhraní pro administraci:* Většina dnešních firewallů má implementováno rozhraní pro nastavení pravidel a bezpečnostní politiky. Tato funkce však není nezbytnou nutností. Zejména firewally v operačním systému Linux se nastavují pomocí konfiguračních souborů. Grafické rozhraní je v mnoha případech nadstavbou pro zlepšení komfortu uživatele [12].

- *Antivirové služby:* Některé firewally umožňují jako dodatkovou službu použít antivirových programů k prohledávání a identifikaci virů. Výhodou takového řešení je rychlost a centralizace řešení v interní síti [2].

- *Intrusion detection systém:* Jedná se o systém detekce průniku. Služba zajišťující sledování a popřípadě i oznámení průniku do vnitřní sítě. Součástí služby jsou zpravidla záznamy do logovacích souborů a odesílání informace na email správce.

- *Filtrování obsahu:* Filtrování obsahu umožňuje blokování přístupu interních uživatelů k určitému obsahu na internetu. Jedná se o různé skupiny obsahu jako je pornografie, rasismus, informace o virech a hackerskych praktikách. Existují organizace poskytující za úplatu seznamy takovýchto stránek. Firewall může filtrovat na základě vlastních pravidel a klíčových slov nebo také za použití externích poskytovatelů seznamů webových stránek vztahujících se k určité kategorii [2].

1.4 Typy firewallů

Firewally můžeme rozdělit například na *hardwarové* a *softwarové*.

Vysoce výkoné hardwarové firewally používají internetoví poskytovatelé a podniky s vysokou náročností na rychlost průchozích dat. Jsou zpravidla mnohem dražší než softwarově řešené firewally. Každý hardwarový firewall obsahuje software, kterým je možné konfigurovat funkce, pravidla a politiku zabezpečení firewallu. Typickým příkladem levnějších firewallů kombinovaných s hardwarem jsou bezdrátové WiFi routery. Téměř každý takový router obsahuje software umožňující zároveň funkci firewallu na bázi filtrování paketů a některé i NAT popřípadě filtrování obsahu webových stránek.

Softwarové firewally jsou obsaženy dnes již v každém operačním systému z různou úrovní zabezpečení. Na trhu je také spousta dostupných firewallů od mnoha výrobců, které vylepšují doplňují případně úplně nahrazují firewally obsažené v operačním systému. Příkladem vysoce kvalitního firewallu v operačním systému Linux je standartně implementovaný paketový filter založený na programu iptables. Příkladem firewallu pracujícího v OS Linux až na sedmé aplikační vrstvě modelu OSI je program squid [13].

Mnohem důležitější je rozdělení firewallů dle jejich základních principů fungování:

- *Firewally založené na filtrování paketů*
- *Firewally s překladem adres (NAT)*
- *Aplikační firewally (proxy)*
- *Kombinované firewally*

1.4.1 Firewally založené na filtrování paketů (IPtables)

Základním principem fungování firewallů je filtrování paketů. V tomto typu firewallu je vytvořena databáze pravidel a kriterií pro zacházení s pakety takzvané filtry. Filtry porovnávají síťové protokoly (např. protokol IP) a jednotlivé atributy paketů transportních protokolů (např. TCP, UDP) s databází definovaných pravidel a rozhodují jakým způsobem s nimi bude naloženo. Funkce filtrování paketů je klíčová pro každý dnes používaný firewall. Následuje popis jednotlivých typů firewallů založených na filtrování paketů [2].

Bezestavový paketový filtr

Jedná se o bezesporu nejjednodušší implementaci firewallu na bázi filtrování paketů. Tento typ firewallu rohoduje o osudu paketů na základě informací obsažených v paketu. Jedná se o informace jako je typ použitého protokolu, zdrojové a cílové IP adresy, rozsah adres, zdrojového a cílového portu či rozsahu portů. Paket je poté buď propuštěn anebo zamítnut. Zamítnutí může proběhnout dvěma způsoby. Jedním způsobem je zamítnutí paketu a odesílatel je o tom informován, a druhým pak zahození paketu bez informování odesílatele. V nejjednodušším provedení se tedy takový firewall chová jako filtrující směrovač [14].

Výhodou takového firewallu je bezesporu vysoká rychlost zpracování. Nevýhodou je nízká úroveň kontroly procházejícího spojení zejména pro protokoly z vyšších vrstev modelu OSI. Tato vlastnost je dána tím, že bezestavový paketový filtr pracuje zejména na třetí síťové a čtvrté transportní vrstvě síťového modelu OSI. Umožňuje tedy vytvoření a otevření jiných portů a spojení, která mohou být využita jinými protokoly než bylo zamýšleno. V praxi to znamená, že nelze interním uživatelům ale třeba i trojským koňům zabránit ve vytvoření služby, otevření portu na klientské stanici a naslouchání příchozím pokusům o připojení. Stavový filter se zpravidla nastavuje manuálně [2][5].

Stavový paketový filtr

Stavový firewall rozšiřuje a vylepšuje možnosti bezestavového firewallu filtrujícího pakety. Na rozdíl od předchozího typu pracuje již na druhé až páte vrstvě síťového modelu OSI. Tím podstatně zvyšuje možnosti zabezpečení. Stavový paketový filtr si oproti bezestavovým filtrům navíc ukládá, udržuje a využívá informace o již jednou povolených spojeních. Tyto informace si ukládá do tabulky[5][14].

K největším výhodám stavového paketového filtru patří vysoká rychlost. Ta je dána tím, že filter již není nucen rozhodovat o již navázaných spojeních. Dále lze uvádět v pravidlech firewallu směr spojení a firewall je schopen automaticky rozhodnout o povolení portu a spojení pro odpověď pro známé protokoly (například protokol FTP na portu 21). Dynamicky tak otevírá porty na základě známých spojení. Kromě rychlosti se tak zvyšuje i snadnost konfigurace a nastavení firewallu, čímž se snižuje i riziko chyby obsluhy při nastavování firewallu. Další výhodou je možnost vytváření takzvaných virtuálních spojení u bezstavových protokolů jako jsou například UDP a ICMP a tím zvýšení rychlosti funkce filtru [2][5][14].

Obecně se dá říci, že stavové paketové filtry poskytují slušnou úroveň zabezpečení avšak nižší než firewally pracující na aplikační vrstvě síťového modelu OSI. Právě dostupný v OS Linux je typickým představitelem tohoto typu firewallu. Nutno podoknout, že spolu s aplikačním firewallem squid lze dosáhnout vysoké úrovně zabezpečení [5].

Stavové paketové filtry s kontrolou protokolů a IDS

Tento typ filtru pracuje až na sedmé aplikační vrstvě modelu OSI. Umožňuje tedy funkce výše zmíněných filtrů, jako je zjišťování informací o spojení a dynamické přidělování portů pro různá spojení. Navíc je schopen kontrolovat procházející spojení až na úroveň koreknostni procházejících dat známých protokolů i aplikací. Umožňuje tak například zakázat spojení protokolu http ve kterém se objeví indikátory tunelování jiného protokolu. Tunelování vužívají často uživatelé peer to peer sítí jež jsou správcem sítě zablokovány, neboť výrazně snižují propustnost sítě (Napster, Gnutella ale i ICQ). Mechanismus tunelování využívá možnosti namapování služby na jiný port než je povolen, například na běžně povolený port 80 určený pro službu http služící pro zobrazování webových stránek . Stavové paketové filtry taktéž umožňují otevírat pouze kanály pro servery, se kterými bylo iniciováno spojení výhradně ze zabezpečené oblasti tedy například hostitelského počítače nebo vnitřní privátní sítě. [2][5].

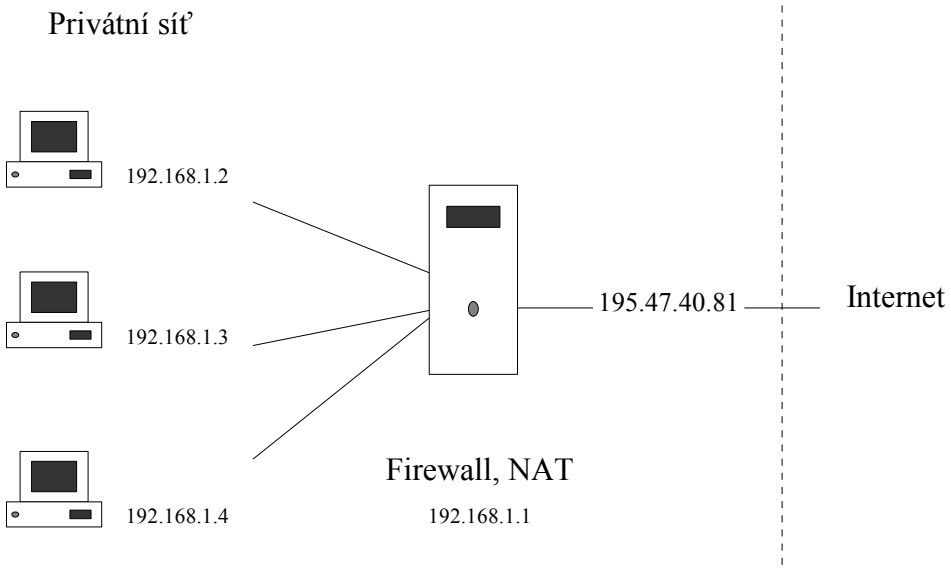
Nově bývá do stavových paketových filtrů integrován systém detekce průniků zkráceně IDS. Systém pracuje podobně jako antivirové systémy. Na základě známých signatur a heuristické analýzy dat je schopen odhalit příznaky a vzorce útoků. Mezi tyto typy útoků patří například skenování povolených portů na hostitelském počítači a jejich rozsahu. Je také schopen odhalit DoS, který ma za cíl zahltit cílový server požadavky a znemožnit tak jeho funkci[5].

Nespornou výhodou těchto systémů je vysoká úroveň zabezpečení daná funkčností až na sedmé aplikační vrstvě síťového modelu OSI. Přitom rychlost zpracování je stále vyšší než u čistě aplikačních firewalů. Paradoxně nevýhodou je vysoká funkcionalita těchto filtrů, která tak otevírá pravděpodobnou možnost vzniku chyby v kódu programu a tím eventuální zneužití a kompromitace celého systému[5].

Podobná funkcionalita je k dispozici ve formě experimentálních modulů pro iptables v linuxovém jádře[5].

1.4.2 Firewally s překladem adres (NAT)

Překlad síťových adres (Network Adress Translation) je jinak nazýván též IP maškaráda (masquerade) nebo také maskování. Ve své podstatě jde o změnu zdrojové nebo cílové IP adresy nebo také portu (NAPT) v samotné hlavičce paketu při jeho průchodu firewallem. Nejčastějším použitím je případ kdy více hostů z privátní sítě (více privátních IP adress) přistupuje na Internet přes jednu veřejnou IP adresu přidělenou internetovým poskytovatelem. V takovém případě se jedná o překlad více interních privátních IP adres na jednu veřejnou IP adresu a opačně – viz. obrázek č.3. Firewall či směrovač si vytváří tabulku spojení, s IP adresami a porty, a dle ní spojení udržuje a předává odpovědi z internetu správnému hostu v interní privátní síti. Spojení tedy inicializuje host z privátní sítě. NAT pracuje na třetí síťové vrstvě modelu OSI [6][15].



Obr.3. Použití dynamického NAT pro sdílení jedné veřejné IP adresy

Největší výhodou NATu je bezesporu možnost snížení počtu přidělených veřejných IP adres a tím i uspoření nákladů. Řeší tedy částečně problém s nedostatkem IP adres při použití protokolu IP verze 4 (Ipv4)[2].

Další nespornou výhodou je vytvoření přirozeného firewallu. Při použití NAT dochází k přirozenému oddělení a skrývání (maskování) interní privátní sítě a externí veřejné sítě. Zvyšuje se tak bezpečnost počítačů v interní síti za NATem. Potenciální útočník nezná cílovou IP adresu. Jelikož inicializace spojení se při použití NAT provádí z daného počítače interní privátní sítě předchází také vytvoření spojení z vnější strany za pomocí trojského koně a podobných škodlivých programů[15].

Použití NAT firewallů ovšem přináší i některá omezení. NAT se implementuje pouze na úrovni TCP/IP. Některé bezestavové protokoly v datové části provozu TCP/IP obsahují IP adresu hostitelského počítače. Tyto protokoly zpravidla vyžadují spojení z bodu do bodu a NAT jim „překáží“ v cestě. Při průchodu paketu přes NAT a přepisu IP adresy v hlavičce paketu, IP adresa uložená v datové části paketu přestává platit (IP adresa v datové části paketu zůstává nezměněna). Následkem toho dojde k přerušení spojení. Řešením může být využití služeb aplikace, která je na vyšší úrovni než TCP/IP. U FTP se využívá takzvaného passive módu [6][15].

NAT nelze použít také pro některé tunelovací protokoly, které obsahují kontrolní mechanismy pro zabezpečení přenosu dat (například IPSec). Změna hlavičky protokolu IP je překážkou pro tyto kontrolní mechanismy[15].

Bráno do důsledků NAT modifikuje paket za provozu, a to neodpovídá standardu pro směrovače. I tak nachází velmi široké uplatnění pro svá nesporná pozitiva[15]

Dle funkce se dají NAT rozdělit do několika skupin.

Jednoduchý NAT

V principu se jedná o nejjednodušší aplikaci překladu IP adres. Každá IP adresa z vnitřní sítě má přidělenu IP adresu z vnější strany sítě a provádí se překlad na tuto adresu a zpět. Převádí adresy 1:1. Nejedná se tedy o sdílení jedné IP adresy pro více hostitelů ve vnitřní síti. Počítač z vnější sítě taktéž může navázat spojení s hostem ve vnitřní síti. Pro aplikaci firewallu je tato varianta nepoužitelná [15].

Jednoduchý NAT s omezením

Omezení jednoduchého NATu spočívá v přidání funkce zablokování možností navázání spojení z vnější sítě. Jedná se tedy již o formu zabezpečení a aplikace firewallu. Spojení z vnější sítě může být navázáno pouze na základě inicializace spojení z vnitřní sítě. Firewall uchovává tabulku spojení[6][15].

Rozšířený NAT

Rozšířený NAT neboli NAPT (Network Adress and Prot Translation) je nejčastěji používán a na rozdíl od předchozích zahrnuje i kontrolu spojení dle čísel portů. V zásadě znemožňuje příchozí spojení na portech, na kterých nebylo inicializováno nebo očekáváno spojení z vnitřní strany síťě[15].

Program iptables jako paketový filter v OS Linux disponuje také možností využití překladu adres na velmi vysoké úrovni.

1.4.3 Aplikační firewally (proxy)

Aplikační firewally nazývané také proxy servery fungují a jsou vázány na protokoly vyšších vrstev síťového modelu OSI, zejména pak na protokoly aplikační vrstvy. Jsou tedy velmi specifické a je třeba mít proxy pro každý typ protokolu. Například http proxy (Web proxy server) pro protokol hhtp. Aplikační firewall je často užíván k úplnému oddělení internetu od vnitřní sítě. Funkce proxy spočívá v tom, že vytváří prostředníka a funguje na bázi klient-server. Klient z vnitřní sítě vytváří požadavek na spojení s internetem. Proxy server tento požadavek převezme a vydává jej za svůj a provede připojení jménem klienta. Opačně pak postupuje i požadavek z veřejné sítě do interní. Pro cílový server je tedy stejně jako u NAT klient z vnitřní sítě nedostupný a obrací se na proxy [2].

Vzhledem k tomu, že se jedná o aplikační firewall, který funguje až na sedmé aplikační vrstvě OSI modelu, je schopen také analyzovat přenesená data až na úrovni aplikací. Dokáží tak například pomocí vyhledávání tagů na html stránkách vyjmout aplety Java jazyka nebo prvky ActiveX komponentů a tím zamezit jejich spuštění na klientské stanici. V kombinaci s NAT jsou schopny zamezit navázání komunikace aplikačních protokolů[2].

Výhodou může také být využití vyrovnávací paměti v případě opakujících se požadavků ze strany klientů. Těmto proxy se říká také Caching proxy. Tím lze také dosáhnout snížení zatížení linky. Proxy server může také podporovat šifrování průchozí komunikace. Existují dva základní typy proxy serverů [16].

Netransparentní proxy

Netransparentní proxy server vyžaduje nastavení na klientském počítači. Například v internetovém prohlížeči se konfiguruje proxy server a přes ten se poté zasílají požadavky pro navázání spojení s webovým serverem[16].

Transparentní proxy

Tento proxy server umožňuje přesměrování běžného provozu na proxy server bez vědomí klienta prostředky operačnéího systému. Na klientském počítači se v tomto případě nemusí nic konfigurovat[16].

Proxy servery vzhledem ke své povaze jsou pomalejší než ostatní typy firewallů. Zpravidla je nesmyslné samostatné použití a jsou kombinovány s paketovými filtry i NAT.

1.4.4 Kombinované firewally

V praxi se setkáváme častěji s kombinací výše uvedených firewallů. Samostatné jednoduché paketové firewally se vyskytují jako základní prvek například v OS Windows. Firewally založené na filtrování paketů jsou zpravidla základním stavebním kamenem dalších typů firewallů například aplikačních, kde samostatné použití zvyšuje riziko průniku. Ideální firewall by dokázal pracovat na všech vrstvách síťového modelu OSI a to velmi rychle. V prostředí Linuxu se používá jako firewall program iptables, který je schopen stavového filtrování paketů, překladu adres a portů a to velmi rychle. Pokud bychom chtěli použít ještě aplikační firewall jako doplněk nabízí se program squid.

2 IPTABLES (NETFILTER)

IPtables je výkoný a mocný nástroj, který slouží ke kontrole a plnému řízení komunikačního toku v síťovém prostředí OS Linux. Iptables samotný je uživatelský program sloužící pro konfiguraci a je součástí komplexnějšího projektu Netfilter. Umožňuje stavbu různých druhů firewallů. Může být použit jako prostý filter nebo stavový firewall s kontrolou obsahu. Stejně tak je vhodný pro překlad síťových adres a portů (NAT) a sdílení internetového připojení, nebo jako základ pro vytvoření transparentního aplikačního proxy serveru. IPtables je součástí každé moderní linuxové distribuce[17].

2.1 Historie IPtables

Prvním firewallem na linuxu byl *IPfw*. Objevil se ve verzi Linux 1.1 a byl převzat z operačního systému BSD. Ve verzi jádra 2.0 přibyl nástroj pro nastavování pravidel *IPfwadmin*. Tento nástroj zároveň přináší základní myšlenku tvorby firewallů na linuxu v podobě řady pravidel (rules), které porovnávají stav paketu s nastavenými pravidly a dle toho rozhodují jak bude s paketem naloženo [18].

Linux 2.2 přináší další vylepšení v podobě *IPchains. IPchains* umožňují sestavení pravidel do řeťezů (chains) a ty následně provázat. Na paket procházející řetězem (chain) jsou aplikována pravidla (rules) a dle nich je pak s paketem naloženo. Až doposud se jedná o bezestavový paketový filter. Kombinací *IPfwadm* a *IPchains* lze dosáhnout filtrování paketů a překladu adres a portů (NAT), nikoliv však sledovat provoz (connection tracking) a obsah paketů. Tyto možnosti přináší až následující vývoj [18].

V roce 1998 založil Rusty Russell (poradce přes IP v Australském národním Linux centru, linux Australia) projekt *Netfilter/IPtables*, založený na jím vytvořeném *Ipchains*. Projekt se postupně rozrůstal a v roce 1999 vzniká Netfilter Core Team, který dále vyvíjí *Netfilter/IPtables* jako svobodný software pod licencí GNU General Public License. První integrovaná verze do linuxového jádra se objevuje v jádře 2.3 v roce 2000. Zajímavostí je soudní proces v německu v roce 2004. Netfilter/IPtables jako takový se objevuje ve směrovačích komerčních společností jako je například Sitecom Germany přičemž porušuje licenci GPL v bodě, který říká, že při použití produktu pod licencí GPL musí být tento také GPL. To znamená zejména zveřejnění zdrojového kódu. Netfilter Core Team tak pod vedením Haralda Welte vyhravá soudní proces, který defakto navrací komerční využívání *Netfilter/IPtables* pod licenci GPL [18].

Vylepšením *IPtables* je rozdělení operací s pakety do tří základních částí a to filtrování paketů, sledování spojení (connection tracking) a překlad síťových adres a portů (NAT). Toto rozdělení ve svém důsledku umožňuje modifikovat, přesměrovávat, sledovat a nakládat s pakety na základě stavu s spojení, tedy nejen na základě znalosti zdrojové a cílové adresy a portu jak bylo zvykem za použití *IPchains. IPtables* umožňuje tedy stavbu stavového firewallu narozdíl od *IPchains,* který není schopen sledovat stav spojení (connection tracking) [18].

V současné době je možno na *IPtables* navázat řadu rozšíření, které tak otevírají další funkcionalitu. Rozšíření (extension) jsou šířena jako záplaty (patches) na zdrojový kód jádra (kernel) systému Linux. Spolu s těmito rozšířeními je k dispozici i nástroj pro aplikaci rozšíření na jádro, který se nazývá *patch-o-matic.* Stejně tak je vyvíjen *IPtables* pro protokol IP verze 6 [18].

2.2 Vlastnosti a možnosti využití IPtables

Mezi základní vlastnosti systému IPtables/Netfilter patří:

- *bezestavový paketový filter (IPv4 a Ipv6)*
- *stavový paketový filter (IPv4 a Ipv6)*
- *překlad adres a portů – NAT/NAPT (IPv4)*
- *možnost roširovaní a vytváření vrstev pro další využití*
- *modulární struktura [13][19]*

Následuje výčet možností využití IPtables:

- *stavba firewallu na bázi bezestavového i stavového filtrováni paketů*
- *využití NAT, sdílení internetu, anebo stavba transparentního proxy serveru*
- *stavba směrovače a nastavení politiky pravidel pro zacházení s průchozím tokem*
- *změna IP hlaviček paketů a manipulace s pakety (mangeling)*
- *zaznamenávání síťového provozu (logging)*
- *řízení toku síťového provozu (za pomocí rozšiřujících modulů)[13][19]*

2.3 Princip IPtables (Netfilter)

Jak víme komunikace po síti je založena na zasílání dat v paketech. Každý procházející paket obsahuje kromě uživatelských dat také hlavičku nesoucí informace o zdrojové a cílové IP adrese, zdrojovém a cílovém portu a další informace které k paketu náleží a popisují komunikaci. IPtables/Netfilter rozhoduje jak naložit s vlastním síťovým paketem [20].

2.3.1 Pravidla, řetězce, tabulky

Každý paket prochází skrz IPtables definovanými pravidly (rules). Tato pravidla jsou seskupena a uspořádána do řetězců (Chains) které definují politiku, tedy jak bude s paketem naloženo. Řetězce jsou dále seskupeny do vlastních filtrovacích tabulek (Tables), kde každá tabulka představuje různý druh práce s pakety. Například pro paketový filter se jedná o tabulku FILTER s řetězci INPUT, OUTPUT a FORWARD, které obsahují pravidla pro práci s pakety (viz tab. 1) [18].

Tab. 1. Příklad struktury tabulky v systému IPtables

Tabulka FILTER		
Řetězec FORWARD	Řetězec INPUT	Řetězec OUTPUT
Pravidlo 1	Pravidlo 1	Pravidlo 1
Pravidlo 2	Pravidlo 2	Pravidlo 2
Pravidlo 3	Pravidlo 3	Pravidlo 3
….....	….....	….....
Pravidlo x	Pravidlo x	Pravidlo x

Jiná tabulka existuje také pro překlad síťových adres a portů (NAT), další pro označování paketů a změnu některých údajů hlavičky paketu (mangle)[21].

Každé pravidlo (rule) obsahuje definici podmínky a cíl. U procházejícího paketu se kontroluje zda vyhovuje (match) podmínce pravidla a na základě toho se rozhoduje o cíli (target) neboli o tom, jak se s paketem naloží[18][21].

2.3.2 Podmínky stanovené pravidlem (matches)

Podmínky stanovené v pravidle, se kterými se hledá shoda (match) pro průchozí paket, mohou být definovány jako omezení na určitou zdrojovou nebo cílovou IP adresu,

specifický port daný pro určitou aplikaci, omezení na lokální síť, nebo lze stanovit třeba jen podmínku specifikující, že se má aplikovat na paket pouze cíl, popřípadě skok na jiné pravidlo nebo řetězec[21].

Můžeme tedy například definovat v pravidle podmínky, kdy se hledá shoda pro pakety protokolu TCP, které přicházejí ze síťové karty napojené na lokální síť a přitom z určité zdrojové IP adresy a portu. Pro paket se pak hledá shoda (match) s těmito podmínkami.

Pakliže není splněna jen jedna z pravidlem definovaných podmínek, pak není nalezena shoda pro celé pravidlo a pokračuje se dalším pravidlem v řetězci[18].

Jestliže jsou splněny všechny podmínky definované pravidlem, provede se s paketem operace specifikovaná v cíli (target) pravidla[18].

Detailní popis možných podmínek (matches) jsou popsány v manuálových stránkách (man) každé linuxové distribuce. Jsou zadávány jako parametr uživatelského programu iptables.

2.3.3 Cíle pravidla (targets)

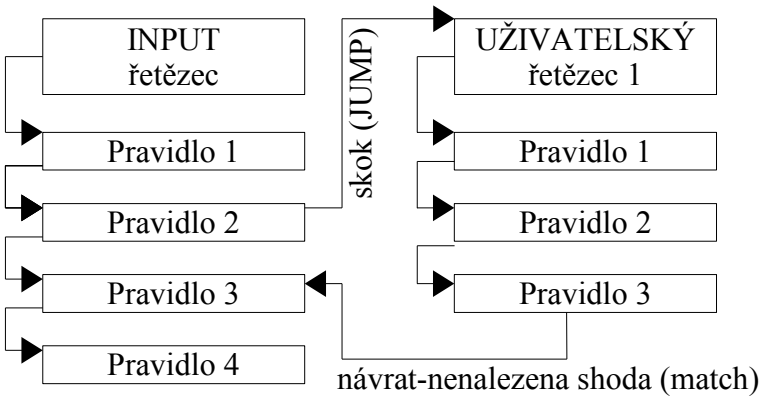
Cíl (target) pravidla říká, jak naložit s paketem pokud byla nalezena shoda (match) se všemi podmínkami definovanými v pravidle. Základními pravidly jsou ACCEPT, DROP, REJECT[21].

ACCEPT jednoduše znamená přijetí paketu. Ten může být dále zpracován aplikací stejně jako odeslán do další tabulky s řetězci pravidel. Přijetí znamená ukončení průchodu paketu aktuálním řetězcem a tabulkou. Neznamená ovšem, že paket nemůže být zahozen nebo s ním jiným způsobem naloženo při průchodu dalšími tabulkami[21].

DROP značí, že paket má být zahozen. Není dále transformován přes žádná pravidla, řetězce potažmo tabulky. Paket je jednoduše zablokován a zahozen. Paket je mrtev a cíl pravidla o tom žádným způsobem již neinformuje[21].

REJECT je obdoba DROP s tím rozdílem, že oznamuje odesilateli chybovým hlášením, že paket byl zablokován. REJECT může být použít a v podstatě také má smysl jej použít jen ve třech řetězcích a to INPUT,OUTPUT a FORWARD. Může být použit také ve vnořených řetězcích do těchto tří nadřazených[21].

Cílem pravidla může být kromě přijetí či zamítnutí paketu také definice skoku (JUMP) na jiné pravidlo. Na obrázku č. 4 je zobrazen příklad skoku[18][21].



Obr.4. Ukázka skoku (JUMP) mezi řetězci pravidel definovaných v IPtables

Pokud například máme vytvořený uživatelsky definovaný řetězec a v něm naše pravidla můžeme nadefinovat skok (JUMP) z předdefinované tabulky (např. z řetězce INPUT tabulky FILTER) na námi definovaný řetězec pravidel. Paket procházející řetězec INPUT je po té přesměrován pravidlem ve kterém jsme nadefinovali skok do námi uživatelsky definovaného řetězce pravidel. Pokud se v tomto řetězci nalezne shoda (match) s podmínkami definovanými v některém z pravidel vykoná se cíl (target). Dále už se neprochází pravidla v uživatelském řetězci ani v nadřazeném řetězci INPUT. Prochází však ostatní řetězce a tabulky v normálním pořadí. Pokud se shoda nenalezne, navrací se zpět do předdefinovaného řetězce INPUT a prochází dál pravidlo po pravidle až do nalezení shody nebo splnění politiky řetězce[21].

Existuje celá řada dalších cílů (target). Namátkou například DNAT, který slouží při překladu síťových adres k přepisu cílové IP adresy nebo SNAT, sloužící k přepisu zdrojové IP adresy. Podrobný seznam všech možných cílů lze nalézt v odkazech na internetových stránkách projektu Netfilter – http://www.netfilter.org [13][21].

2.4 Struktura IPtables

2.4.1 Netfilter a IPtables

Ačkoliv pojmy *IPtables* a *Netfilter* jsou si velmi blízké, jedná se o dva nezávislé prvky.

Netfilter je systém zabudovaný v jádře (kernel) OS Linux, který umí pracovat s moduly (IPtables je jedním z nich) a pracovat se síťovými pakety. Netfilter obsahuje základní předdefinované řetězce pravidel pro práci s pakety[22].

IPtables se skládá ze dvou základních částí. První částí jsou moduly v jádře systému navázané na Netfilter. Mezi tyto moduly patří jak samotný IPtables tak moduly pro zajištění NAT, sledování spojení (connection tracking), záznam (logging) a další. Mohou a nemusí být použity všechny, což je bezesporu výhodou modulárního systému. Tyto moduly obsahují základní předdefinovaná tabulky pro práci s pakety[19][22].

Druhou částí je samotný uživatelský program „iptables“, distribuovaný nezávisle na jádře systému. Slouží k vlastnímu přidávání, ubírání a editaci pravidel pro práci s pakety v modulech jádra systému Linux. Na většině linuxových systémů je program nainstalován v adresáři /sbin/. Pokud hovoříme o iptables máme v naprosté většině případů na mysli právě tento uživatelský program (V textu uvádím uživatelský program malými písmeny “iptables“)[22].

2.4.2 Předdefinované řetězce

V jádře (kernel) operačního systému Linux (od verze 2.4) je definována skupina základních řetězců pro práci se síťovými pakety. Jedná se o řetězce PREROUTING, INPUT, OUTPUT, FORWARD a POSTROUTING. Každý z nich definuje politiku zacházení s pakety a slouží svému účelu na cestě paketu[19][20][21].

PREROUTING rozhoduje zda příchozí paket bude předán k dalšímu zpracování do vstupního řetězce INPUT nebo zda bude směrován jinam přes řetězec FORWARD. Také se využívá při překladu síťových adres (NAT), a to před dalším zpracováním v řetězci INPUT nebo FORWARD. Pomocí něj můžeme měnit cílovou IP adresu v hlavičce paketu (target DNAT). Proto nemůže být PREROUTING ze své podstaty využit k samotnému filtrování paketů[21].

INPUT řetězec dělá v podstatě nejtěžší práci. Jedná s ejiž o vlastní filtrování příchozích paketů. Můžeme například vytvořit vnořený řetězec pro vyfiltrování vadných paketů. Pomocí INPUT můžeme definovat různá omezující pravidla pro příchozí pakety. Pokut paket vyhoví pravidlům je předán aplikaci naslouchající na daném portu[21].

OUTPUT řetězec odbavuje odchozí pakety. Pomocí něj je možné omezovat datový tok na specifických portech a bránit tak uživatelům využívání služeb. (Sítě peer to peer)[21].

FORWARD řetězec slouží k směrování paketů. Implicitní politika tohoto řetězce je zahazovat pakety, tedy zabraňovat směrování[21].

POSTROUTING řetězec vykonává závěrečnou práci po řetězci OUTPUT. Stejně jako PREROUTING neslouží k vlastnímu filtrování paketů, ale využívá se v případě NAT.

Konkrétně mění v hlavičce IP paketu zdrojovou IP adresu (cíle SNAT nebo MASQUERADE). Toho se využívá při sdílení jedné veřejné IP adresy do vnitřní sítě LAN[21].

Všechny výše jmenované řetězce jsou standardní součástí systému Netfilter. Vlastní způsob přenosu paketu mezi jednotlivými řetězci je popsán dále.

2.4.3 Předdefinované tabulky

V jádře systému předdefinované řetězce mohou využívat a zpravidla využívají tabulky, které může měnit uživatel. (userspace tables). Mezi základní tabulky patří FILTER, NAT, MANGLE, RAW a samozřejmě mohou být definovány další. Přehled tabulek, základních použitelných cílů v pravidlech tabulky a tabulkou využívaných řetězců je uveden v tab. č. 2[19][21]

Tab. 2. Přehled tabulek, cílů (targets) a návazných řetězců (chains)

Tabulka	***Cíle (targets)***	***řetězce (chains)***
FILTER	ACCEPT, DROP, REJECT ...	PREROUTING, INPUT, OUTPUT, FORWARD, POSTROUTING
NAT	SNAT, DNAT, MASQUERADE, REDIRECT	PREROUTING, POSTROUTING, OUTPUT
MANGLE	TOS, TTL, MARK, SECMARK, CONNSECMARK	INPUT, OUTPUT, FORWARD, PREROUTING, POSTROUTING
RAW	NOTRACK	PREROUTING, OUTPUT

Tabulka FILTER se využívá pro samotné filtrování paketů a to například na základě zdrojové nebo cílové IP adresy, UDP nebo TCP zdrojového nebo cílového portu, označení paketu pomocí tabulky MANGLE, zdrojové MAC adrese a podobně. Může také defragmentovat pakety a využívat informace ohledně spojení a sledovat jej (connection tracking). Přijímá (ACCEPT) nebo zahazuje (DROP/REJECT) pakety dle shody (match) s pravidly (rules) popřípadě definuje další možné operace. FILTER využívá řetězce INPUT, OUTPUT a FORWARD. Při sledování spojení také PREROUTING

a POSTROUTING řetězce[19][21].

NAT tabulka se používá pro překlad síťových adres NAT. Mění zdrojovou (SNAT) a cílovou (DNAT) adresu procházejících paketů. K tomuto dochází vždy v případě prvního paketu a pro ostatní je použita stejná politika. V případě dynamicky přidělované IP adresy se používá v pravidlech této tabulky také cíl MASQUERADE, který je náročnější než SNAT, nicméně dokáže pracovat za použití dhcp. Tabulka NAT využívá předdefinovaných řetězců PREROUTING a POSTROUTING a někdy také OUTPUT[19][21].

MANGLE tabulka se používá pro značení nebo změnu některých údajů v hlavičce paketu. V žádném případě neslouží k filtrování nebo NAT. Dochází zde k takzvanému „mandlování“ (mangeling) paketů. Pomocí toho dosahujeme změn parametrů v hlavičce paketu jako jsou TOS, TTL, nebo pouze označení paketu za účelem dalšího filtrování v za pomocí jiné tabulky. Tato tabulka může využívat všech předdefinovaných řetězců jádra[19][21].

RAW tabulka je používána pouze k označování paketů, u kterých nelze nebo se nevyžaduje sledování spojení. Má pouze jediný cíl pro pravidla a tím je NOTRACK. Využívá řetězce PREROUTING a OUTPUT[21].

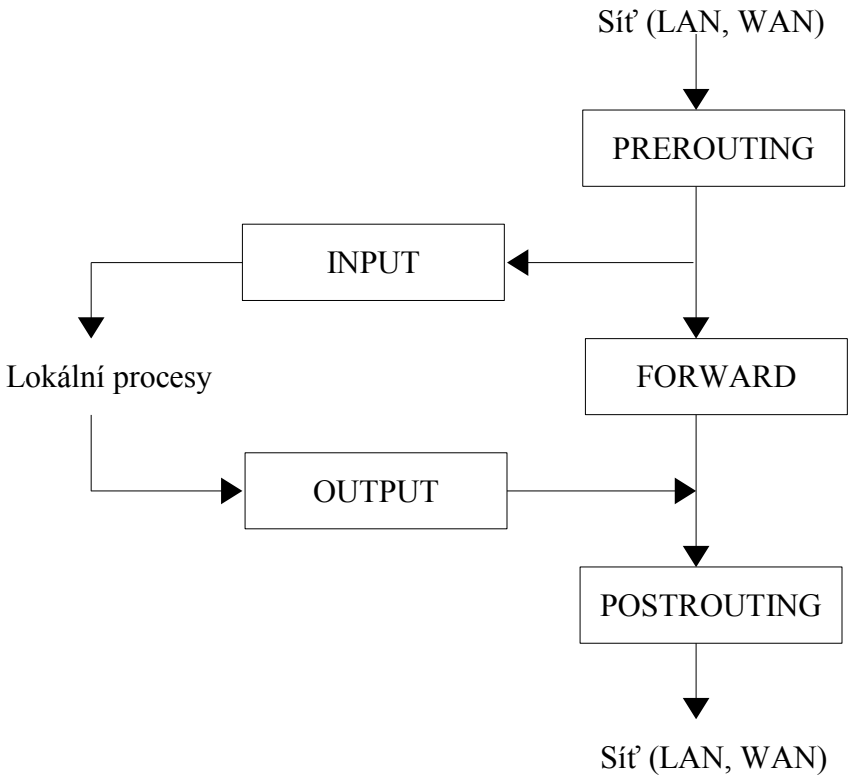
Pokud nedefinujeme žádnou uživatelskou tabulku použijí se předdefinované řetězce v jádru systému [17].

2.5 Průchod paketu

Před příchodem paketů do vlastní aplikace nebo jejich přesměrováním prochází pakety nejdříve firewallem, v tomto případě přes IPtables. S tím souvisí i průchod hardwarem a za pomocí ovladače zařízení se pakety dostávají do jádra systému. Důležité je, že v jádru systému dochází k aktivaci hardwaru až po aktivaci firewallu, čímž se zabraňuje možnému zneužití při startu zařízení.

Každý paket, ať už příchozí či odchozí, postupně prochází minimálně jedním předdefinovaným řetězcem. V každém pravidle takového řetězce se testuje zda vyhovuje (match) podmínce pravidla. Pokud vyhovuje zastaví se průchod řetězcem a s paketem se naloží tak jak je specifikováno v cíli (target) pravidla. Pokud paket projde předdefinovaným řetězcem, v jádře systému, a není nalezena shoda z žádným pravidlem, pak se s paketem naloží tak jak je definováno v takzvané politice řetězce. Politika je definována pouze u předdefinovaných řetězců, nikoliv tedy u uživatelsky definovaných

řetězců. Standardní politikou řetězce může být například ACCEPT – tedy přijmutí paketu. Průchod paketu základní strukturou předdefinovaných řetězců v jádře (kernel) systému Netfilter je zobrazen na obr. č. 5.

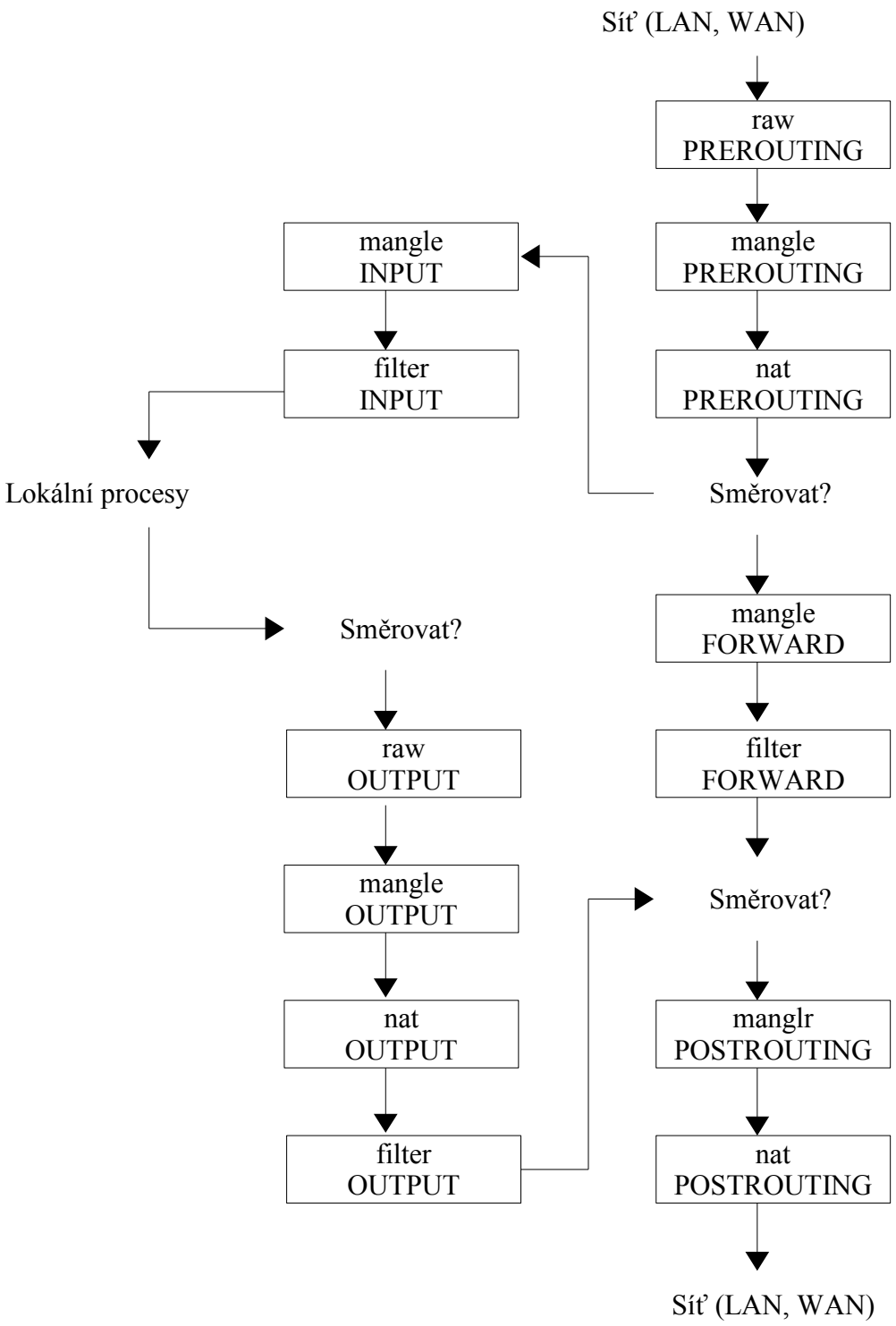


Obr.5. Průchod paketů strukturou předdefinovaných řetězců v jádře systému

Pokud je paket určen pro jiný uzel (host), a je nastavena možnost směrování, pak je přesměrován za pomocí předdefinovaného řetězce FORWARD. Jeli tedy paket přesměrován přes FORWARD neprochází předdefinovanými řetězci INPUT ani OUTPUT.

Pokud paket prochází uživatelsky definovaným řetězcem a není nalezena shoda s žádným pravidlem nebo je řetězec prázdný (bez pravidel definovaných uživatelem) pak je cílem návrat do řetězce, který uživatelsky definovaný řetězec volal. Z toho také vyplývá, že cílem pravidla může být skok na jiný řetězec nebo pravidlo (JUMP).

Na obrázku č. 6 je zobrazeno schéma jak mohou pakety procházet řetězci včetně použitých tabulek (userspace tables). Jsou zde ukázány možnosti IPtables jako takového. Dle tohoto schématu je patrné které řetězce je třeba nastavit například v tabulce NAT pro využití překladu síťových adres. Pro tabulku NAT jsou to konkrétně řetězce PREROUTING (cíl DNAT), OUTPUT a POSTROUTING (cíl SNAT), které jsou využívány.



Obr.6. Průchod paketů skrz firewall (IPtables)

2.6 Syntaxe a použití iptables

Pro vlastní nastavení politiky předdefinovaných řetězců a definici pravidel slouží uživatelský program iptables. Použití programu již předpokládá nainstalované patřičné moduly v jádru systému. Minimální verze jádra pro použití iptables je verze 2.4.

Syntaxe iptables je:

```
iptables -t[TABLE]-akce[CHAIN][RULE]-m[match ext.]-j[TARGET][target ext.]
```

Parametr *-t* definuje tabulku (NAT,MANGLE,RAW,FILTER), pokud ji chceme využít. V opačném případě se využívá předdefinovaných řetězců jádra.

Přehled základních možných *akcí* pro práci s řetězci *(CHAIN)* je uveden v tabulce č. 3[17]

Tab. 3. Akce pro práci z řetězci

Akce	***Popis***
-A	Přidání nového pravidla na konec řetězce
-D	Smaže pravidlo (dle čísla pravidla)
-R	Nahradí pravidlo jiným pravidlem
-I	Vloží pravidlo na začátek řetězce
-L	Vypíše pravidla řetězce
-F	Vyprázdní všechna pravidla v řetězci
-N	Vytvoří nový řetězec
-X	Smaže řetězec
-P	Nastaví výchozí politiku řetězce
-E	Přejmenuje vlastní řetězec

CHAIN jsou předdefinované řetězce v jádře, nebo námi vytvářené řetězce

Pro definici pravidla *(RULE)* je možné použít přepínačů. Základní přepínače jsou uvedeny v tabulce č. 4 [23].

Tab. 4. Přepínače pro definici pravidel

přepínač	***Popis***
-p	protokol
-s	zdrojová adresa/maska podsítě
-d	cílová adresa/maska podsítě
-i	vstupní zařízení (síťová karta – eth0, lo)
-o	výstupní zařízení (síťová karta)
-j	skok (JUMP), cíl který se má provést

Match extension slouží k definici dalších vlastností pokud nám nestačí definice v předchozí takzvané základní IP části. V tuto chvíli můžeme využít přídavných modulů k jádru. Pro příklad uvedu rozšíření multiport s jehož použitím lze definovat až 15 portů oddělených čárkou. Bez tohoto rozšíření by bylo nutné napsat pro každý port samostatné pravidlo[17][21].

K dalším rozšířením patří např. *state* umožňující kontrolu spojení (connection tracking) tedy funkci stavového firewallu. Dále pak *limit* pro kontrolu toku dat a další[17].

TARGET jednoduše slouží k definici cíle při nalezení shody s pravidlem. Například ACCEPT, DROP, REJECT, JUMP, GOTO a další.

Target extension se používá podobně jako *match extension.* Využívá tedy rozšiřujících modulů. Využívá se např. pro změnu paketů a logování. Typickým příkladem změny paketu je použití NAT, kdy měníme zdrojovou a cílovou IP adresu. K tomu používáme *target extension -to-source* a *-to-destination*[17].

Příklad použití:

```
iptables -P INPUT  DROP
```

Nastaví zahazování příchozích paketů. Nastaví politiku řetězce INPUT na DROP.

```
iptables -A INPUT -p icmp -j DROP
```

Nastaví nové pravidlo v řetězci INPUT a to takové, že všechny pakety protokolu icmp jsou implicitně zahazovány. V praxi to znamená naříklad, že při pokusu o ping na lokální síťové zařízení nedostaneme odpověď, neboť paket bude zahozen. Stejně tak při pokusu o ping na jiný uzel (host) nedostaneme odpověď a to ne z důvodu, že by byl zablokován odchozí icmp paket, ale odpověď přicházející přes řetězec INPUT bude zahozena [23].

```
iptables -A INPUT -i eth0 -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Tento příklad ukazuje nastavení stavového filtrování se sledováním spojení na zařízení *eth0.* Přijímány jsou pakety u kterých bylo vytvořeno *(ESTABLISHED)* spojení, nebo bylo vytvořeno nové spojení některým z již probíhajících *(RELATED*). Při tomto nastavení využíváme rozšíření *state* [23].

Konkrétní příklad nastavení iptables pro malou síť s NAT je uveden v praktické části bakalářské práce. Taktéž na internetu je možné najít množství příkladu nastavení iptables.

Pro přehlednější práci s iptables je namístě vytvořit si skript ve kterém nastavujeme iptables. V tomto skriptu můžeme volat potřebné moduly nutné pro naše nastavení iptables, po té vyprazdňujeme nadefinovaná pravidla a nastavujeme vlastní. Nastavení také ukládáme. Tento skript je vhodné volat při každém startu počítače.

Nápomocné při vytváření firewallu nám mohou být internetové stránky s automatickou generací skriptů pro iptables. Například na internetových stránkách http://easyfwgen.morizot.net/gen/ .

3 SHAPING A BLOKOVÁNÍ PROVOZU SÍTÍ PEER TO PEER

V současné době jsou sítě peer to peer velmi populární. Jedná se o různé programy (například bittorrent, P2P klienti), které vytěžují velkým objemem přenášení dat internetové spojení a zamezují tak provozu na síti. Obzvláště to můžeme pocítit při využívání aplikací citlivých na kvalitu spojení a vyžadující kontinuální proud dat jako jsou multimediání konference nebo i VoIP telefonie. Při velkém vytížení může dojít i k tomu, že nejsou zobrazovány internetové stránky. Nežádoucí jsou sítě peer to peer zejména v komerční sféře [24][25].

3.1 Pojmy

Quality of Service definuje kvalitu toku dat v síti. S pojmem QoS souvisí následující pojmy , které určují výslednou kvalitu přenosu dat na síti.

Latence, neboli doba zpoždění, nám udává v jaké době projde paket z jednoho bodu do druhého.

Šířka pásma (bandwidth) udává kolik dat jsme schopní přenést v jednom časovém okamžiku. Šířka pásma velmi úzce souvisí s latencí.

Například internetové připojení poskytované ISP může mít šířku pásma 2048/128 (download/upload). Limitující je odchozí (upload) šířka pásma, která je 128kbps. Při použití DSL modemu máme navíc velmi vysokou latenci. Pokud chceme zaslat více dat než je 128kbps hromadí se ve výstupní frontě (čeká se na odeslání) modemu a zvyšuje se latence. Tomu můžeme zabránit právě tvarováním toku ještě před odesláním dat do modemu.

Tvarování toku (shaping) umožňuje kontrolu odchozího síťového toku a to kontrolovat a rozdělovat šířku pásma (bandwidth control) a rychlost odchozího provozu (rate limiting). Příchozí tok dat nejsme schopni do jisté míry ovlivnit. U protokolu TCP lze, v dobrém slova smyslu, využít vlastnosti, že pokud budeme zahazovat odchozí potvrzovací pakety, pak odesílající strana postupně zpomaluje tok dat [26][24].

Tvarování toku je založeno například na algoritmu HTB. S jeho pomocí lze kontrolovat odchozí síťový tok a rozdělovat fyzické spojení na vícero pomalejších virtuálních spojení. Po té jsou směrována data dle typu do patřičných virtuálních spojení, a tím zajištěna prioritizace přenosu dat [25].

3.2 Strategie tvarování toku

3.2.1 Možnosti blokování sítí P2P

Existuje několik možností jak je možné blokovat P2P a tvarovat tok pomocí iptables [25]:

- *dle portu*
- *dle velikosti paketu*
- *dle typu provozu*

Blokování sítí P2P podle příchozího a odchozího portu není příliš smysluplné. Úplné zablokování portu může spíše upozornit uživatele, který začne hledat jinou cestu jak dosáhnout žádoucího provozu. To se může dít změnou portu v klientské aplikaci nebo tunelováním provozu přes jiný port, např. port 80 určeného pro protokol http a běh internetových stránek. Po té nemáme žádnou možnost zjistit jaký port používá P2P aplikace. Je tedy výhodnější provoz neblokovat, ale tvarovat![25]

Upřednostňování síťového provozu dle velikosti paketu může přinášet některé výhody. Předně sítě peer to peer již ze svého principu přenosu většího objemu dat používají vetší pakety. Naproti tomu například VoiP telefonie využívá menších paketů. To nám umožňuje upřednostňovat některé pakety. Nevýhodou může být opět možnost změny velikosti paketu v klientské aplikaci P2P [24][25].

Pro tvarování provozu (shaping) je nejvhodnější využít znalosti typu provozu v síti. Toho dosáhneme analýzou obsahu paketů za pomocí iptables. Analýzou obsahu paketu jsme schopni identifikovat a po té i označit (mark) pakety, které jsou určeny pro P2P provoz a to bez ohledu na to přes jaký port běží či jaká je velikost paketu. Pro zjišťování obsahu paketu jsou k dispozici rozšiřující moduly (match extension) pro iptables. Jedním z těchto modulů je ipp2p. Dalším je například i7-filter [27][25].

3.2.2 Upřednostňování provozu v síti (prioritizace)

Při tvarování toku lze upřednostňovat tok na základě znalosti obsahu paketů. Můžeme tak rozdělit síťový tok do několika základních kategorií[24][25]:

- *Citlivý provoz*
- *Ostatní provoz*

- *Prohlížení internetových stránek*
- *Provoz P2P*

Citlivý provoz je interaktivní provoz například VoIP telefonie, přenášení video konferencí, také třeba hraní online her. Tento provoz vyžaduje minimální latenci a vysokou kvalitu přenosu (QoS). Nízkou latenci vyžadují i protokoly ssh a icmp. Ideální je dát tomuto provozu nejvyšší prioritu, nebo jej vynechat z tvarování vůbec a tím zajistit hladký průběh [24][25].

Do ostatního provozu zařadíme vše co nespadá do první kategorie a podléhá tvarování toku[25].

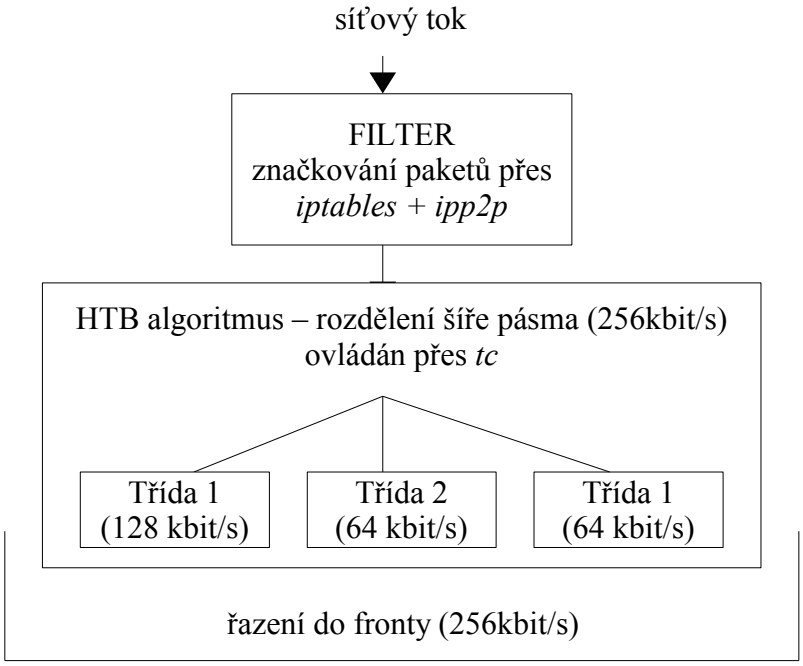
Prohlížení webových stránek a protokol smtp (emailová komunikace) může mít větší latenci než výše uvedené interaktivní aplikace. Je však žádoucí aby byl tento provoz upřednostněn před provozem sítí P2P[25].

Provoz P2P je ze své podstaty původcem zahlcení šířky pásma a tím i zvýšení latence. Tomuto typu provozu nastavujeme pomocí shaping nejnižší prioritu s ohledem na využívání sítě[25].

3.3 Princip tvarování síťového toku

K tvarování a kontrole síťového toku a jeho upřednostňování se využívá několika algoritmů. Příkladem je například algoritmus HTB – Hierarchical Token Bucket, zabudovaný v jádře. Tento algoritmus rozděluje fyzické spojení na menší virtuální spojení, kterých využíváme pro prioritizaci datové komunikace. HTB je schopno zajistit šířku pásma nikoliv však interaktivitu. To je dáno tím, že HTB počítá objem přenesených dat nikoliv pakety[25].

HTB využívá IPtables. Ve filtru IPtables (tabulka MANGLE) si označíme (mark) paket na základě identifikace obsahu paketu. Tím určíme prioritu daného paketu. K posouzení obsahu se použije například rozšíření (match extension) *ipp2p*. Takto označený paket poté prochází přes algoritmus HTB. HTB, pak rozděluje pakety dle priorit a odesílá je do nadefinovaných tříd. Třídy slouží již k vlastnímu omezování síťového toku. Algoritmus HTB můžeme ovládat pomocí programu z balíku iproute2 a tím je *tc (traffic control)*. Schéma je na obrázku č.7 [28][25][21].



Obr.7. Schéma tvarování toku a použitých nástrojů

Algoritmus HTB má hierarchickou strukturu. Každé síťové rozhraní má přiřazeny fronty (queueing discipline, zkráceně qdisc). Paket posílaný jádrem na síťové zařízení jde právě přes tuto frontu (qdisc). Nejjednodušeji si lze qdisc představit jako FIFO frontu. Fronta je schopna uchovávat data po určitý časový okamžik [25].

Fronta (qdisk) může obsahovat třídy (clasess). Třídy mohou dále obsahovat další třídy, mluvíme pak o „rodiči“ a „dětech“. Z této hierarchické struktury vyplývá možnost upřednostnění toku dat z některých tříd oproti jiným. Třídy navíc neuchovávají žádná data a ihned je odesílají do fronty. Pokud fronta (qdisk) není naplněna , doplňuje a odesílá data dle nastavení priorit jednotlivých tříd. Každá třída a fronta je definována dvojicí čísel 1:0. První číslo nám udává číslo zařízení (síťového rozhraní). Nulové druhé číslo udává, že se jedná o kořenovou frontu (root qdisc). Nenulové druhé číslo nám říká, že jde o třídu. Nejpraktičtější je využít sestupného číslování směrem k číslu fronty [29].

Při tvarování toku síťového provozu definujeme pro každou třídu garantovanou propustnost (rate) a maximální propustnost (ceil). Například na obrázku č. 7 je fronta s garantovanou propustností (rate) 256kbit/s. Tato je rozdělena do podřazených tříd na 128 a dvakrát 64 kbit/s. To znamená pro tyto podřazené třídy se jedná o garantované propustnosti (rate), nicméně každá z těchto tříd může mít maximální propustnost (ceil) stejnou jako nadřazená fronta tedy 256 kbit/s. Toho využíváme v případech kdy fronta není

naplněna zcela a je doplňována z jednotlivých tříd dle předdefinovaných priorit[25][28].

Princip algoritmu si tedy lze představit jako vědro s dírou na dně. Do vědra naléváme vodu a ta vytéká konstantní rychlostí, bez ohledu na přítok nebo objem vědra. Tak si lze představit odchozí frontu síťového toku. Snažíme se, aby odchozí síťový tok měl konstantní rychlost a dosahujeme toho tvarováním toku, konkrétně propustnosti, a prioritizací[29].

3.4 Syntaxe a použití tc

K nastavování algoritmu HTB jádra systému slouží program *tc (traffic control).* Ten může předpokládat předchozí označení paketů použitím iptables.

Základní syntaxe příkazu je[29]:

```
tc [volby] OBJEKT { Příkaz
```

Volby *-s (statistika), -d (detaily), -b (skriptovy soubor)* lze provádět s objekty typu fronta *(QDISC),* třída *(CLASS),* filter *(FILTER).*

Příkaz má tedy módy týkající se řazení do front, tříd a filtrů:

```
tc qdisc [add|change|replace|link] dev ZAŘÍZENÍ [parent ČÍSLO RODIČE|
     root] [handle ČÍSLO QDISC] qdisc [parametry]

tc class [add|change|replace] dev ZAŘÍZENÍ parent ČÍSLO QDISC [classid
     ČÍSLO TŘÍDY] qdisc [parametry]

tc filter [add|change|replace] dev ZAŘÍZENÍ [parent ČÍSLO RODIČE|root]
     protocol PROTOKOL prio PRIORITA filtertype[paramtery] flowid ČÍSLO
```

Nejlepší způsob vysvětlení voleb a parametrů je jednoduchý přiklad. Následující příklad ukazuje využití tvarování toku rozlišením aplikace dle čísla portu (ssh 22, P2P klient 6881:6889). Po rozlišení a označení paketů za pomocí *iptables* jsou nastaveny pomocí *tc* priority a omezen provoz[25].

Pomocí iptables si označíme pakety :

```
iptables -t mangle-A FORWARD -p tcp --dport 22 -j MARK --set-mark 1
  iptables -t mangle -A OUTPUT -p tcp --dport 22 -j MARK --set-mark 1

  # označíme číslem 1 všechny pakety z cíloveho portu 22 - ssh

iptables -t mangle -A FORWARD -i eth0 -p tcp --sport 6881:6889
           -j MARK --set-mark 4
iptables -t mangle -A FORWARD -i eth0 -p tcp --dport 6881:6889
           -j MARK –set-mark 4
```

```
# označíme nejnižší prioritou 4 zdrojové i cílové pakety portu,
# který využívá aplikace P2P, konkrétně se zde jedná o bittorrent
```

Specifikujeme hlavní frontu (root qdisc) a podřízené třídy s definicemi propustnosti:

```
tc qdisc add dev eth0 root handle 1:0 htb default 103 r2q 1

  # přidá na zařízení „eth0“ hlavní frontu s ID 1:0 při využití HTB

  tc class add dev eth0 parent 1:0 classid 1:1 htb rate 256kbit

# vytvoří rodičovskou třídu 1:1 navázanou na nadřazený qdisc 1:0 s
  garantovanou propustností 256kbit.

tc class add dev eth0 parent 1:1 classid 1:101 htb
      rate 192kbit ceil 256kbit
tc class add dev eth0 parent 1:1 classid 1:104 htb
      rate 64kbit ceil 128kbit

# vytvoří podřízené třídy 1:101 a 1:104 a nastavuje pro tyto třídy
  garantovanou a maximální propustnost. Třída 1:104 je určena pro P2P.
```

Roztřídíme označené pakety do jednotlivých tříd:

```
tc filter add dev eth0 parent 1:0 protocol ip prio 0 handle 1 fw
      classid 1:101
tc filter add dev eth0 parent 1:0 protocol ip prio 3 handle 4 fw
      classid 1:104

# třídíme pakety podle značek z iptables do patřičných tříd, tedy pakete
  označené „1“ do třídy 1:101 z vyšší prioritou (0)a propustností a
  pakety označené „4“ do třídy 1:104 z nižší prioritou (3) a menší
  propustností
```

Oficiální popis syntaxe příkazu tc je dostupný v man stránkách každé linuxové distribuce.

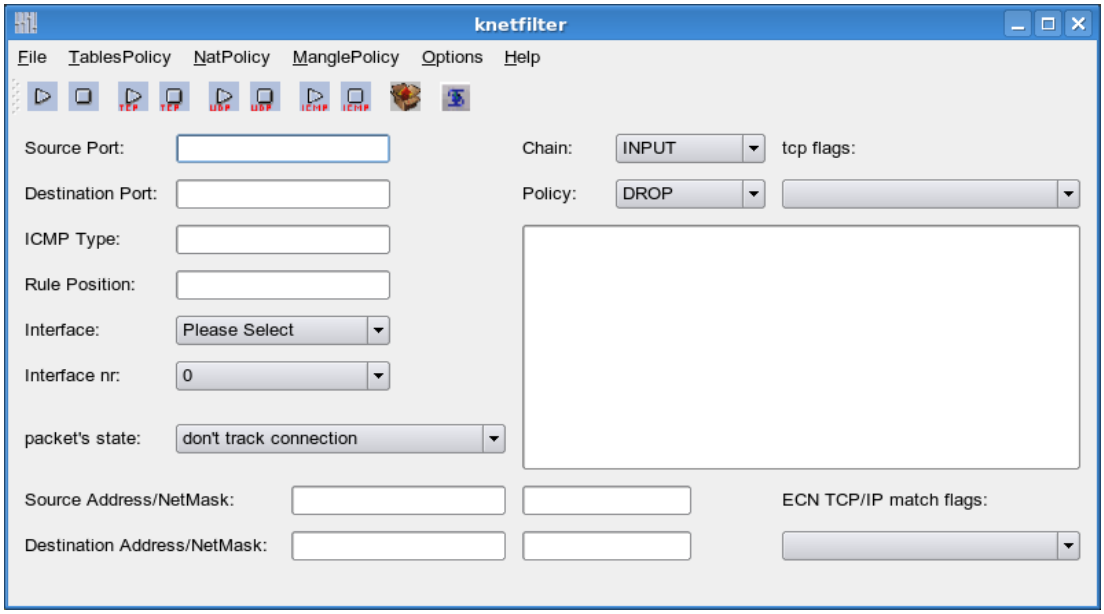
4 GUI APLIKACE A NADSTAVBY FIREWALLU V OS LINUX

Pomoc při nastavení IPtables/Netfilter mohou poskytnout grafická uživatelská rozhraní. Jelikož je však firewall na bázi IPtables velmi komplexní a flexibilní záležitostí není nikterak jednoduché udělat GUI. Vždy se předpokládá znalost IPtables jako takového. Existuje vícero GUI pro IPtables, některá přímo integrovaná do grafických nadstaveb systému jako jsou KDE a GNOME, a některá jsou koncipována jako webová rozhraní [21].

Je pochopitelné, že samotné spuštění je grafické aplikace je nutné udělat pod uživatelem root. Jen administrátor má práva na změny a nastavení v IPtables/Netfilter.

4.1 Knetfilter

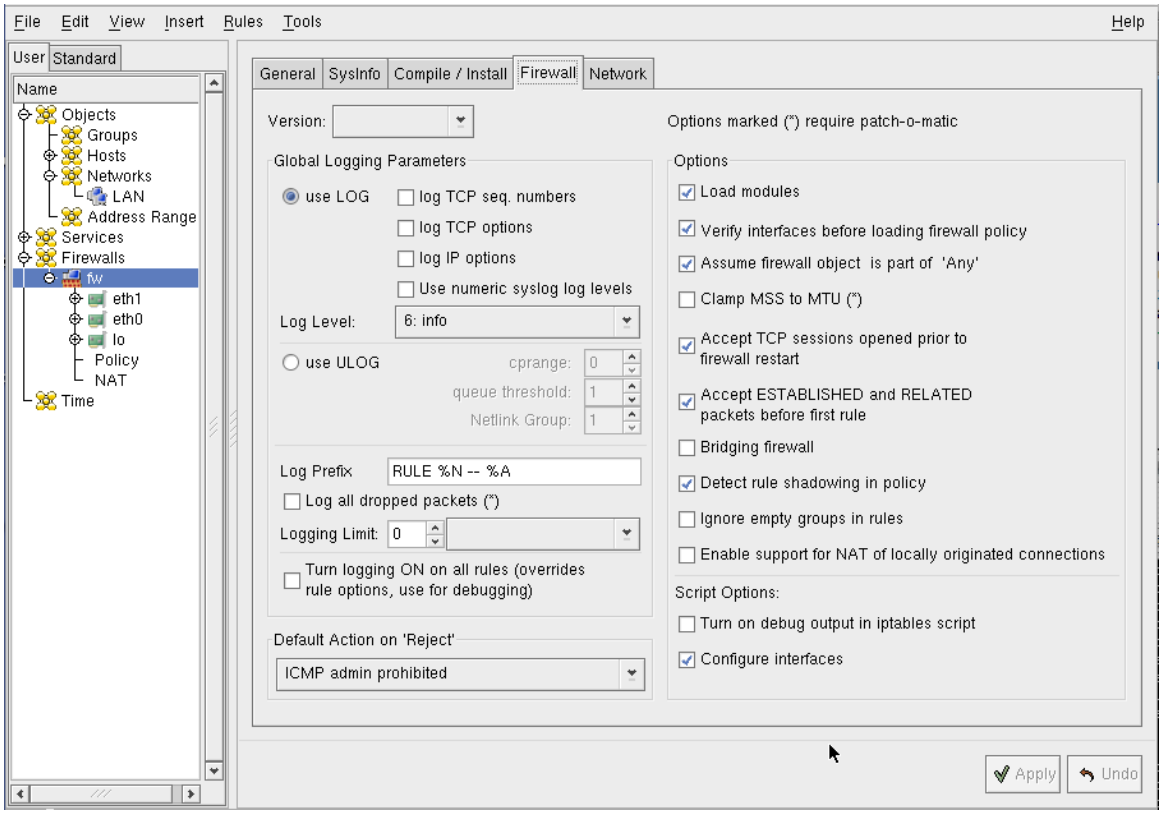
Jako první příklad uvádím program *knetfilter* (obrázek č. 8). Knetfilter je spustitelný pod prostředím KDE. Kromě nastavení a změn proveditelných v IPtables/Netfilter a stavby firewallu umožňuje použití *tcpdump* a *nmap*. Za pomocí tcpdump je možné monitorovat síť. Nmap slouží ke zjišťování portů, které jsou otevřeny na síťovém rozhraní. Jinými slovy, jedná se o port scanner.



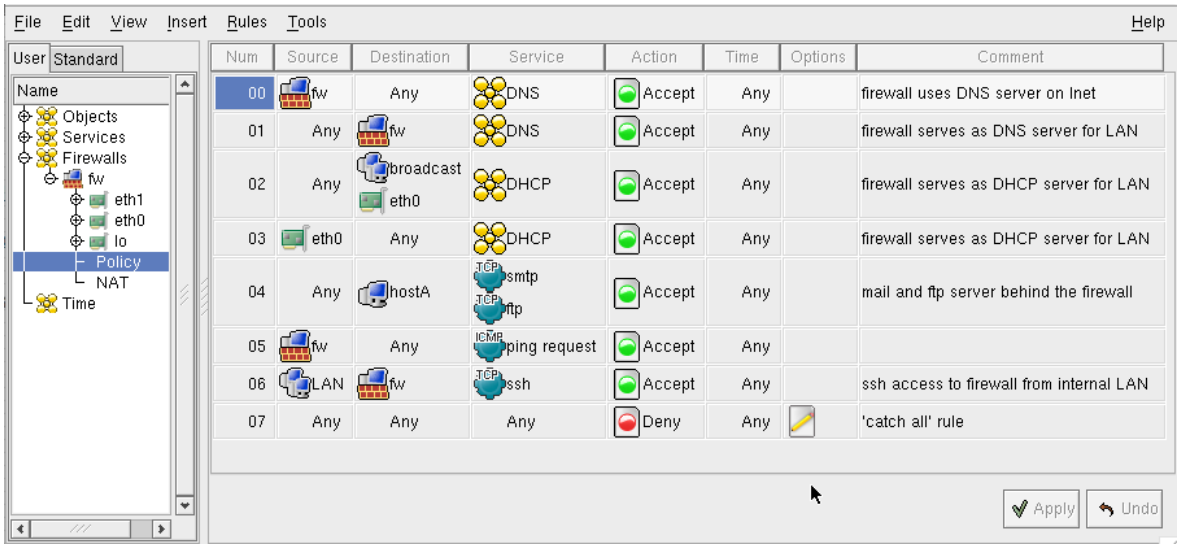
Obr.8. Knetfilter

4.2 Firewall Builder

Velmi komplexním nástrojem pro různé platformy je program fwbuilder (obrázek č.9,10). Jedná se o objektově orientované GUI. Vytváří databázi síťových objektů a politik, se kterými je možné zacházet metodou „drag and drop“. Podporuje různé typy firewallů, jako je IPtables, IPfilter, openbsd, CISCO PIX [30].



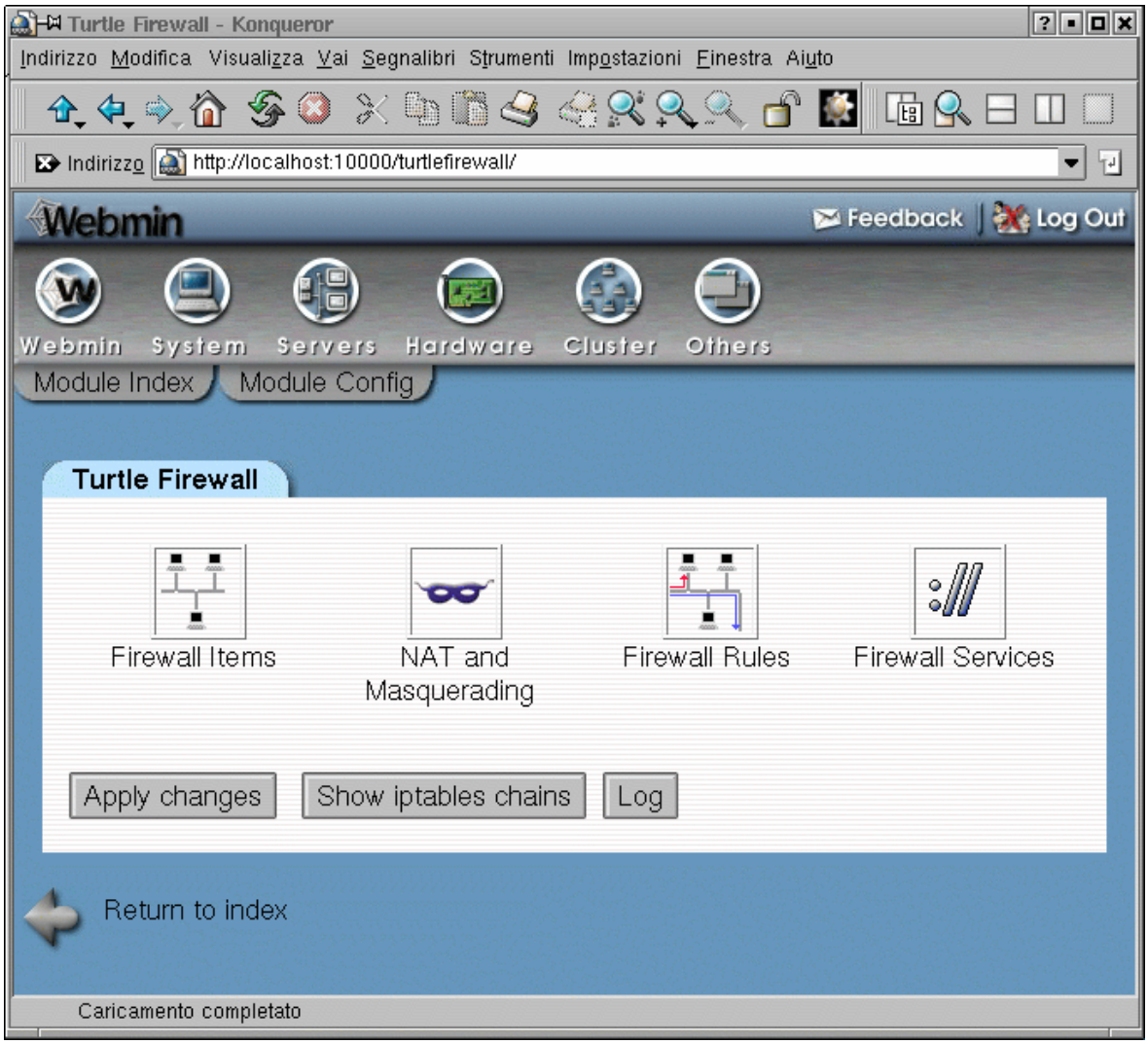
Obr.9. Firewall Builder – možnosti IPtables



Obr.10. Knetfilter – nastavení politiky

4.3 Turtle Firewall

Turtle firewall je ukázkou jak může vypadat nastavování IPtables/Netfilter přes webové rozhraní. Jde o velmi hezké provedení, nicméně je náročnější na instalaci a vyžaduje podporu dalších programů jako je apache apod. Ukázka je zobrazena na obrázku č. 11 [21].



Obr.11. Ukázka webového rozhraní Turtle firewall

Existuje mnoho GUI s různou úspěšností implementace IPtables/Netfilter do GUI. Přesto je nastavení pomocí souboru jako skriptu je stále tou nejúčinnější metodou. Mnoho grafických programů nám může pomoci při generaci takovéhoto souboru jako základu pro konfiguraci vlastního firewallu.

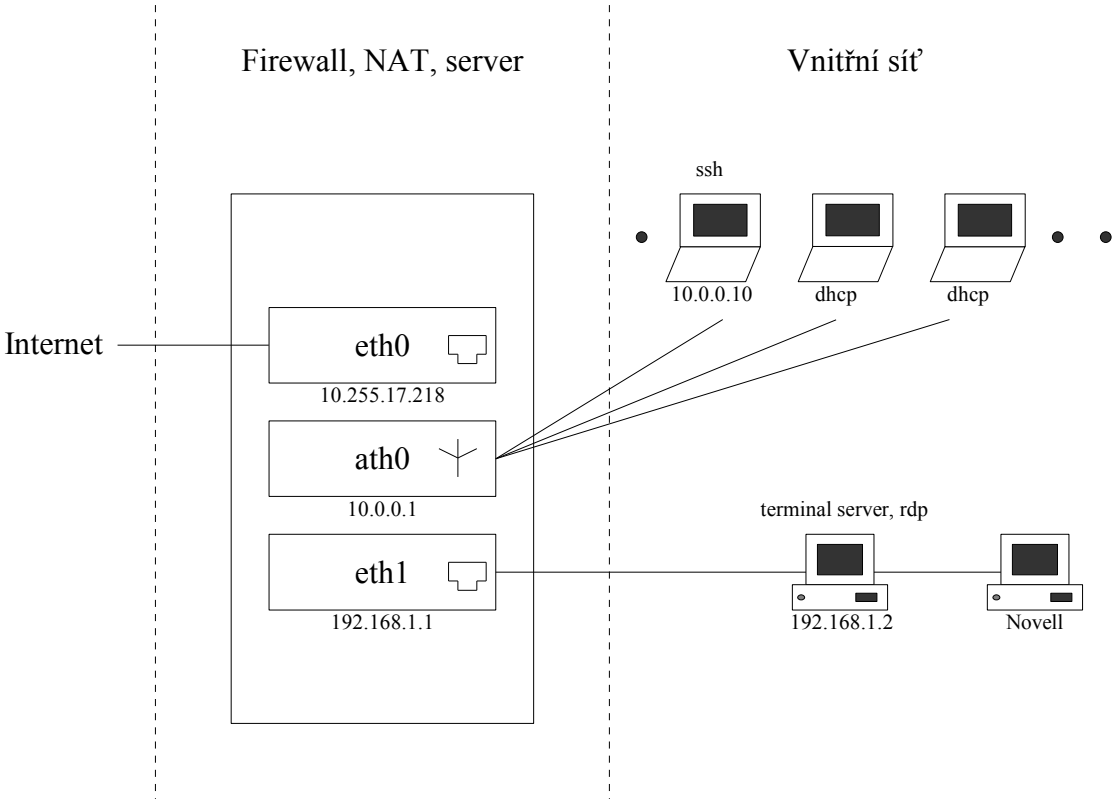
II. PRAKTICKÁ ČÁST

5 NASTAVENÍ FIREWALLU PRO MALOU SÍŤ V OS LINUX

V praktické části bakalářské práce bych rád demonstroval nastavení firewallu založeného na IPtables pod operačním systémem Linux. V rámci nastavení firewallu bude využit i překlad síťových adres a portů – NAT pro sdílení internetového připojení. Tato síť i s nastavením je provozována ve firmě SUNNY Computer Technology Europe, s.r.o. se sídlem v Brně. Nastavení se provádí na linuxové distribuci Gentoo s jádrem 2.6.18-r4.

5.1 Schéma sítě

Pro lepší představu je na obrázku č. 12 ukázána struktura počítačové sítě ve firmě SUNNY Computer Technology Europe, s.r.o.



Obr.12. Struktura počítačové sítě

Počítačová síť je složena z jednoho serveru, na kterém běží Linux s implementaci IPtables jako firewallu. Server obsahuje tři síťová rozhraní a to *eth0, eth1 a ath0*.

Rozhraní *eth0* slouží k připojení na Internet. Má staticky přidělenou IP adresu, kterou poskytuje ISP - 10.255.17.218 s maskou sítě 255.255.255.248. Z důvodu, že poskytovatel

internetového připojení přiděluje pouze jednu veřejnou IP adresu, což je běžný stav, se provádí překlad síťových adres NAT.

Síťové rozhraní *ath0* slouží pro směrování lokální bezdrátové wifi sítě. Na toto rozhraní se připojují klienti z lokální firemní sítě s dynamickým přidělováním IP adres. Na jednom klientovi je nadefinována statická adresa 10.0.0.10, na kterou je směrován provoz ssh pro možnost zprávy z vnější sítě.

Rozhraní *eth1* slouží pro připojení kabelové vnitřní sítě. K tomuto rozhraní je připojen počítač vyžadující rychlejší datový tok, který nemůže být zajištěn přes wifi rozhraní. Jedná se o propojení s počítačem, na kterém běží Novell a účetnictví a vyžaduje přístup zvenčí. Může také sloužit jako servisní kanál v případě poruchy na bezdrátovém rozhraní.

Na serveru kromě firewallu běží také webový server apache, mysql a dhcp pro přidělování adres na rozhraní ath0. Sítě na rozhraních ath0 a eth1 jsou vzájemně propojeny.

5.2 Potřebná nastavení

Pro provoz firewallu a nastavení počítačové sítě je třeba provést následující kroky:

- *nastavení jednotlivých síťových rozhraní*
- *nastavení potřebných voleb jádra systému pro provoz IPtables a NAT*
- *softwarového vybavení*
- *nastavení IPtables coby firewallu s NAT*

5.3 Nastavení síťových rozhraní

Všechna síťová rozhraní mají přiděleny statické IP adresy. Nastavení je v distribuci Gentoo Linux uloženo v souboru */etc/conf.d/net:*

```
# in /etc/conf.d/net
config_eth0=("10.225.17.218 netmask 255.255.255.248")
routes_eth0=("default via 10.225.17.217")

config_eth1=("192.168.1.1/24")
routes_eth1=("10.225.17.218")

config_ath0=("10.0.0.1/24")
routes_ath0=("10.225.17.218")
```

Skripty */etc/init.d/net.*,* které se spouštějí při startu serveru berou nastavení z výše zmíněného souboru. Důležité je spouštět skript pro firewall ještě před nastartováním těchto

rozhraní z důvodů zajištění bezpečnosti.

5.4 Konfigurace kernelu

Pro správnou funkci IPtables potřebujeme mít zapnuté související moduly v jádře systému. V tomto případě je nastavení prováděno na verzi jádra 2.6.18-gentoo-r4. Aktuální stav zda je IPtables/Netfilter podporován naším jádrem lze zjistit příkazem:

```
grep -i netfilter /usr/src/linux/.config        nebo:

zgrep -i netfilter /proc/config.gz (kernel musí podporovat
export nastaveni do proc/config
```

Výpis by měl vypadat následovně:

```
CONFIG_NETFILTER=y
```

Ve výpisu se mohou objevit další volby důležité například pro nastavení NAT. Pokud není podpora netfiltru v jádře zakompilována je nutno ji nastavit a jádro překompilovat. Nastavení provedeme příkazem *make menukonfig* v adresáři /usr/src/linux. Nastavení se ukládá do souboru *.config* v tomtéž adresáři. Je dobré si jej před vlastní konfigurací zazálohovat.

Vlastní nastavení jádra:

```
Networking  --->
 Networking options  --->
  [*] Network packet filtering (replaces ipchains)  --->
   Core Netfilter Configuration  --->
      <*> Netfilter netlink interface
      < >   Netfilter NFQUEUE over NFNETLINK interface
      < >   Netfilter LOG over NFNETLINK interface
      <*> Netfilter Xtables support (required for ip_tables)
      <*>   "CLASSIFY" target support
      <*>   "CONNMARK" target support
      <*>   "MARK" target support
      <*>   "NFQUEUE" target Support
      <*>   "NOTRACK" target support
      <*>   "comment" match support
      <*>   "connbytes" per-connection counter match support
      <*>   "connmark" connection mark match support
      <*>   "conntrack" connection tracking match support
      <*>   "DCCP" protocol match support
      <*>   "ESP" match support
      <*>   "helper" match support
      <*>   "length" match support
      <*>   "limit" match support
      <*>   "mac" address match support
      <*>   "mark" match support
      <*>   IPsec "policy" match support
      <*>   Multiple port match support
      <*>   "pkttype" packet type match support
```

```
        <*>   "quota" match support
        <*>   "realm" match support
        <*>   "sctp" protocol match support (EXPERIMENTAL)
        <*>   "state" match support
        <*>   "statistic" match support
        <*>   "string" match support
        <*>   "tcpmss" match support


Networking  --->
 Networking options  --->
  [*] Network packet filtering (replaces ipchains)  --->
   IP: Netfilter Configuration  --->
       <*> Connection tracking (required for masq/NAT)
       [*]   Connection tracking flow accounting
       [*]   Connection mark tracking support
       [ ]   Connection tracking events (EXPERIMENTAL)
       < >   Connection tracking netlink interface (EXPERIMENTAL)
       < >   SCTP protocol connection tracking support
       (EXPERIMENTAL)
       <*>   FTP protocol support
       < >   IRC protocol support
       < >   NetBIOS name service protocol support (EXPERIMENTAL)
       < >   TFTP protocol support
       < >   Amanda backup protocol support
       < >   PPTP protocol support
       <*>   H.323 protocol support (EXPERIMENTAL)
       <*>   SIP protocol support (EXPERIMENTAL)
       <*> IP Userspace queueing via NETLINK (OBSOLETE)
       <*> IP tables support (required for filtering/masq/NAT)
       <*>   IP range match support
       <*>   TOS match support
       <*>   recent match support
       <*>   ECN match support
       <*>   DSCP match support
       <*>   AH match support
       <*>   TTL match support
       <*>   Owner match support
       <*>   address type match support
       <*>   hashlimit match support
       <*>   Packet filtering
       <*>   REJECT target support
       <*>   LOG target support
       <*>   ULOG target support
       <*>   TCPMSS target support
       <*>   Full NAT
       <*>     MASQUERADE target support
       <*>     REDIRECT target support
       <*>     NETMAP target support
       <*>     SAME target support
       <*>     Basic SNMP-ALG support (EXPERIMENTAL)
       <*>   Packet mangling
       <*>     TOS target support
       <*>     ECN target support
       <*>     DSCP target support
       <*>     TTL target support
       < >     CLUSTERIP target support (EXPERIMENTAL)
       <*>   raw table support (required for NOTRACK/TRACE)
       < > ARP tables support
```

Vše je zakompilováno přímo v jádře. Je možné vytvářet pouze moduly, ty je však nutné nahrát pomocí příkazu *modprobe*. Výhodou modulů může být odstranění z paměti, a naopak nevýhodou nutnost zavádění při startu systému.

Po nastavení kernelu je nutno jádro kompilovat. To provedeme příkazem *make.* Pokud zvolíme v nastavení moduly, je třeba moduly nainstalovat příkazem *modules_install.* V závěru je nutno nakonfigurovat používaný zavaděč systému (LILO, GRUB).

5.5 Softwarové vybavení

Pro nastavení IPtables/Netfilter je nutné mít program iptables. V distribuci Gentoo Linux lze nainstalovat příkazem *emerge iptables.* V ostatních distribucích lze využít balíčkovacích systémů, některé mají iptables standardně nainstalován.

Pro rozhraní *ath0* je nakonfigurována a spuštěna služba *dhcp*. Službu opět lze nainstalovat přes *emerge dhcp.* Tato služba dynamicky přiděluje IP adresy připojeným klientům. Službu lze spouštět přes */etc/init.d/dhcpd*. Konfigurační soubor je v adresáři */etc/dhcp/dhcpd.conf* a */etc/conf.d/dhcpd*, kde definujeme na kterém rozhraní služba běží.

Překlad názvů na IP adresy z doménového jména a zpět zajišťuje služba *named.* Služba také zajišťuje vyrovnávací paměť pro překlady doménových jmen a tím šetří čas při dotazech z vnitřní sítě.

Dalším programem je aplikační proxy server *squid.* Primárně slouží jako webový proxy server s vyrovnávací pamětí pro urychlení přístupu na internetové stránky s následnou možnou filtrací na viry a zkoumání obsahu v návaznosti na protokol http. Program se nastavuje v */etc/squid/squid.conf.* Program má velmi široké konfigurační možnosti. Veškerý odchozí provoz na portu 80 (http) je přesměrován pomocí iptables na *squid,* který naslouchá na portu 3128. Tím je vytvořena transparentní proxy. Squid může provádět antivirovou kontrolu.

5.6 Nastavení iptables - skript

Veškerá manipulace s IPtables se provádí pomocí skriptu */etc/init.d/fw.* Skript je napsaný jako služba systému Gentoo Linux a automaticky se spouští při startu systému. Skript akceptuje parametry *start, stop, restart, status, edit.* Skript zjednodušuje a zpřehledňuje práci s iptables. Celý skript je uveden v příloze P1. Pro blokaci nebo povolení určitých portů je vhodné využít seznam portů uvedených v příloze P2 [31].

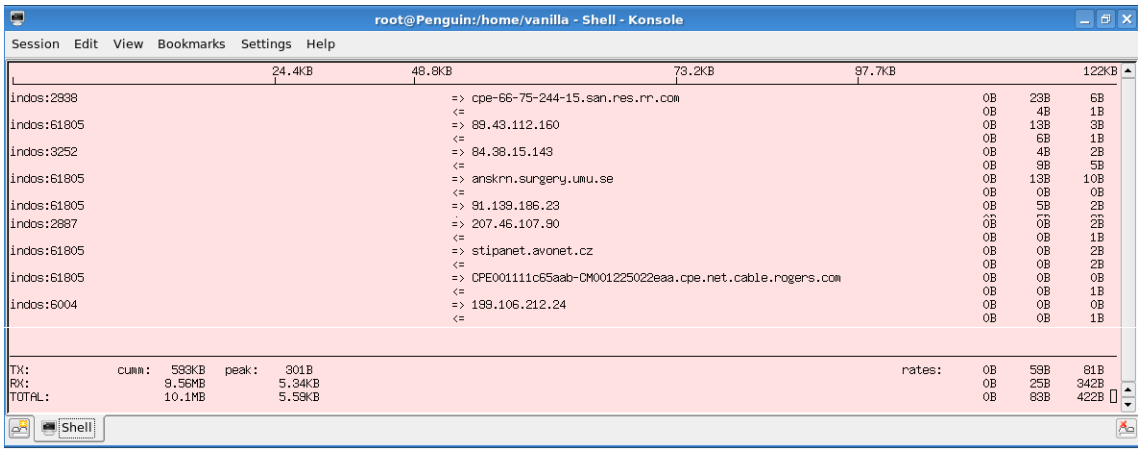
Jednotlivá nastavení použitá ve skriptu (příloha P1):

- *pro zjednodušení a lepší orientaci ve skriptu se nastavují proměnné pro síťová rozhraní, IP adresy a cestu k iptables*
- *nastavení směrování paketů mezi síťovými rozhraními*
- *nastavení implicitní politiky řetězců jádra (INPUT, OUTPUT, FORWARD)*
- *nastavení jednotlivých tabulek PREROUTING, FORWARD, INPUT, OUTPUT, POSTROUTING a specifická nastavení v těchto tabulkách včetně NAT*
- *nastavení používaná při uvolňování síťového rozhraní (restart, vypnutí serveru)*
- *závěrečná nastavení skriptu pro akceptování parametrů např. z příkazové řádky (start, stop, restart, edit, status)*

Takto nastavený skript (příloha P1) je uložen v adresáři */etc/init.d.* Voláme jej při každém zapnutí, vypnutí a restartu serveru. Jednoduše jej můžeme zařadit jako službu při spouštění systému. Na Gentoo Linux toho lze dosáhnout příkazem rc-update, konkrétně *rc-update add fw default.*

5.7 Monitoring sítě

Monitoring stavu sítě pro jednotlivá síťová rozhraní můžeme provést například programem *iftop – viz obrázek č. 13*



Obr.13. Struktura počítačové sítě

ZÁVĚR

Cílem bakalářské práce bylo osvětlit princip fungování firewallů se zaměřením na operační systém Linux a jeho IPtables. Je vysvětlen pojem firewall, jeho funkce a princip.

V návaznosti na to je vysvětlen princip fungování IPtables pod operačním systémem Linux. Dále popis funkce IPtables jako velmi silného nástroje a firewallu s možností překladu síťových adres (NAT), kontrolou obsahu procházejících paketů (stavový firewall) a možnosti použití pro kontrolu síťového toku (shaping), blokování sítí peer to peer a jako základu pro transparentní proxy server.

Pro zjednodušenou konfiguraci IPtables jsou ukázány možnosti použití grafických uživatelských rozhraní, coby nadstaveb na IPtables/Netfilter.

V praktické části bakalářské práce je ukázána implementace firewallu v malé firemní síti. Zde slouží jako stavový firewall s překladem síťových adres (NAT) pro sdílení veřejné IP adresy přidělené ISP. Je zde ukázán postup implementace firewallu, včetně potřebného nastavení síťových rozhraní, konfigurace jádra systému a vytvoření nastavovacího skriptu pro firewall. Tento firewall je provozován ve firmě SUNNY Computer Technology, s.r.o. se sídlem v Brně.

Výhledově je možné se zaměřit na využití tvarování toku sítě (traffic shaping).

CONCLUSION

The main point of this work was to show the function of firewall in Linux. The definition of firewall, its functions and principals were explained.

In additon, the function of IPtables as a very strong and powerfull tool under Linux was demonstrated. IPtables can be used as a state firewall, moreover, it is possible to use NAT for sharing IP address provided by ISP. IPtables is a base tool for implementing traffic shaping and blocking peer to peer networks. Furthermore IPtabels can be used for building transparent proxy server.

There where presented GUI interfaces for IPtables that are useful for better and easier IPtables configuration.

In the last part of this work, there is demonstrated a practical implementation of firewall in a small company network. Firewall in this company is also used for network address translation with masquerade. For implementing firewall is needed to configure network devices, compile kernel with appropriate settings and make script file for setting up the firewall. This firewall is used in SUNNY Computer Technology Europe, s.r.o.

In the future it is possible to focus on implementing traffic shaping.

SEZNAM POUŽITÉ LITERATURY

[1] Miniwatts Marketing Group. *World Internet Usage Statistics News and Population Stats* [online]. 2007, poslední revize 19.03.2007 [cit. 2007-05-01]. Dostupný z: http://www.internetworldstats.com/stats.htm

[2] Matthew Strebe, Charles Perkins. Firewally a proxy-servery: *Praktický průvodce* 1. vyd. Brno: 2003, 450s. 80-7226-983-6.

[3] Pavel Kodýtek. *Historie Internetu* [online]. 2006, poslední revize 31. 01. 2006 [cit. 2007-05-01]. Dostupný z: http://www.webdesign.paysoft.cz/clanky/2006/historie-internetu/

[4] Frederick M. Avolio, *Firewalls and Internet Security-The Internet Protocol Journal* [online].2007, poslední revize 05.02.2007 [cit. 2007-05-01] Dostupný z: http://www.cisco.com/web/about/ac123/ac147/ac174/ac200/about_cisco_ipj_archive_article09186a00800c85ae.html

[5] Wikipedia. *Firewall* [online]. 2007, poslední revize 30.03.2007 [cit. 2007-05-01]. Dostupný z: http://en.wikipedia.org/wiki/Firewall_(networking)

[6] Libor Dostálek a kolektiv. Velký průvodce protokoly TCP/IP: *Bezpečnost* 1. vyd. Brno: 2001, 565s. 80-7226-513-X.

[7] Dilip C. Nail. INTERNET: - *standardy a protokoly* . vyd. Brno: 1999

[8] Wikipedia. *Internet Protocol* [online]. 2007, poslední revize 21.05.2007 [cit. 2007-05-21]. Dostupný z: http://en.wikipedia.org/wiki/Internet_Protocol

[9] Wikipedia. *IP address* [online]. 2007, poslední revize 21.05.2007 [cit. 2007-05-19]. Dostupný z: http://en.wikipedia.org/wiki/IP_address

[10] Wikipedia. *TCP/IP model* [online]. 2007, poslední revize 21.05.2007 [cit. 2007-05-21]. Dostupný z: http://en.wikipedia.org/wiki/TCP/IP_model

[11] Wikipedia. *OSI model* [online]. 2007, poslední revize 21.05.2007 [cit. 2007-05-19]. Dostupný z: http://en.wikipedia.org/wiki/OSI_model

[12] Seminárky. *Firewall data za ohnivou zdí* [online]. 2007, poslední revize 02.05.2007 [cit. 2007-05-01]. Dostupný z: http://seminarky.cz/Firewall-data-za-ohnivou-zdi-6846

[13] Netfilter core team. *The netfilter.org "iptables" project* [online]. 2007, poslední revize 16.04.2007 [cit. 2007-05-01]. Dostupný z: http://www.netfilter.org/

[14] Asknet. *Firewall* [online]. 2007, poslední revize 02.05.2007 [cit. 2007-05-01]. Dostupný z: http://www.asknet.cz/cz_tech.php

[15] Wikipedia. *Network address translation* [online]. 2007, poslední revize 01.05.2007 [cit. 2007-05-01]. Dostupný z: http://en.wikipedia.org/wiki/Network_Address_Translation

[16] Wikipedia. *Proxy server* [online]. 2007, poslední revize 03.05.2007 [cit.]. Dostupný z: http://en.wikipedia.org/wiki/Proxy_server

[17] Csaba Botoš. *Vše o iptables* [online]. 2006, poslední revize 10.01.2006 [cit. 2007-05-04]. Dostupný z: http://www.root.cz/clanky/vse-o-iptables-uvod/

[18] Wikipedia. *Netfilter/iptables* [online]. 2007, poslední revize 04.05.2007 [cit. 2007-05-04]. Dostupný z: http://en.wikipedia.org/wiki/Netfilter/iptables

[19] James Stephens. *Iptables* [online]. 2007, poslední revize 11.05.2007 [cit. 2007-05-13]. Dostupný z: http://www.sns.ias.edu/~jns/wp/iptables/

[20] Miroslav Petříček. *Stavíme firewall* [online]. 2001, poslední revize 12.05.2007 [cit. 2007-05-13]. Dostupný z: http://www.root.cz/clanky/stavime-firewall-1/

[21] Oskar Andreasson. *Iptables Tutorial 1.2.2* [online]. 2006, poslední revize 11.05.2007 [cit. 2007-11-20]. Dostupný z: http://iptables-tutorial.frozentux.net/iptables-tutorial.html

[22] David Coulson. *Mastering IPTables* [online]. 2001, poslední revize 12.05.2007 [cit. 2001-04-04]. Dostupný z: http://davidcoulson.net/writing/lxf/14/iptables.pdf

[23] Kim Nielsen. *Firewalls* [online]. 2007, poslední revize 14.05.2007 [cit. 2007-02-20]. Dostupný z: http://www.gentoo.org/doc/en/security/security-handbook.xml?part=1&chap=12

[24] Wikipedia. *Traffic shaping* [online]. 2007, poslední revize 15.05.2007 [cit. 2007-05-15]. Dostupný z: http://en.wikipedia.org/wiki/Traffic_shaping

[25] Gentoo Linux Wiki. *HowTo Packe shaping* [online]. 2007, poslední revize 16.05.2007 [cit. 2007-02-19]. Dostupný z: http://gentoo-wiki.com/HOWTO_Packet_Shaping

[26] Marek Stopka. *Jak na sdílení pásma aka (traffic) shaping* [online]. , poslední revize 16.05.2007 [cit. 2007-04-24]. Dostupný z: http://www.abclinuxu.cz/faq/site/jak-na-sdileni-pasma-aka-traffic-shaping

[27] E.Friedrich. *About IPP2P* [online]. 2007, poslední revize 16.05.2007 [cit. 2006-06-16]. Dostupný z: http://www.ipp2p.org/

[28] Radek Podgorný. *HTB-Jemný úvod* [online]. 2003, poslední revize 16.05.2007 [cit. 2003-08-28]. Dostupný z: http://www.root.cz/clanky/htb-jemny-uvod/

[29] Jakub Mácha. *Kontrola síťového provozu*. Brno 2000, 44s. Diplomová práce. Masarykova Univerzita, fakulta informatiky. Vedoucí diplomové práce Mgr. Jan Kasprzak.

[30] NetCitadel,LLC. *Firewall Builder* [online]. , poslední revize 15.05.2007 [cit. 2007-03-29]. Dostupný z: http://www.fwbuilder.org/

[31] Wikipedia. *List of TCP and UDP port numbers* [online]. 2007, poslední revize 16.05.2007 [cit. 2007-03]. Dostupný z: http://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

CBQ	Class Based Queuing – Algoritmus používaný pro tvarováni toku v síti
CERN	Organisation européenne pour la recherche nucléaire - Evropská organizace pro jaderný výzkum
DEC	Digital Equipment Corporation – Americká společnost
dhcp	Dynamic host configuration protocol – Protokol pro automatické přidělování a konfiguraci IP adres
DNAT	Destination NAT – Překlad cílové síťové adresy
DSL	Digital subscriver line – Digitální zákaznická přípojka, umožňuje využití telefonní nebo kabelové televize pro vysokorychlostní přenos dat
GPL	General Public Licence - „Všeobecná veřejná licence“, licence pro svobodný software
GUI	Graphical User Interface – Grafické uživatelské rozhraní
HTB	Hierarchical Token Bucket – Algoritmus pro tvarování toku v síti
http	Hypertex transfer protocol – Internetový protokol pro výměnu hypertextu
IDS	Intrusion Detection System – Systém detekce průniku
IP	Internet Protocol – Internetový protokol
ISO	International Organization for Standardization – Mezinárodní organizace pro normalizaci
ISP	Internet Service Provider – Poskytovatel internetového připojení
LAN	Local Afrea Network – Místní, lokální síť
MIT	Massachusetts Institute of Technology – Americká univerzita
NAPT	Network adress and port translation – Překlad síťových adres a portů
NASA	National Aeronautics and Space Administration – Národní úřad pro letectví a kosmonautiku
NAT	Network Adress Translation – Překlad síťových adres
OS	Operační systém
OSI	Open Systems Interconnection Basic Reference Model – Referenční model určený pro standardizaci komunikace v počítačových sítích
QoS	Quality of Service – Protokol pro řízení datových toků v síti
SNAT	Source NAT – Překlad zdrojové IP adresy
TCP/IP	Transmission Control Protocol/Internet – Sada protokolů pro komunikaci přes Internet (Primární transportní protokol / Protokol síťové vrstvy IP)
TOS	Type of Service - Parametr v hlavičce paketu protokolu IP
TTL	Time To Live – Parametr v hlavičce paketu protokolu IP
VPN	Virtual Privat Network – Virtuální privátní síť

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P 1: SKRIPT PRO NASTAVENÍ FIREWALLU

Nejdříve si pro zjednodušení nastavíme proměnné:

```
# IP adresa a vnější rozhraní

INET_IP="10.225.17.218"
INET_IFACE="eth0"

# IP a broadcast adresa rozhraní vnitřní sítě - kabel
LAN1_IP="192.168.1.1/32"
LAN1_BCAST="192.168.1.255/32"
LAN1_IFACE="eth1"

# IP a broudcast adresa rozhraní vnitřní sítě - WiFi
WLAN_IP="10.0.0.1/32"
WLAN_BCAST="10.0.0.255/32"
WLAN_IFACE="ath0"

# Lokální loopback rozhraní
LO_IFACE="lo"
LO_IP="127.0.0.1/32"

# Cesta k programu iptables
IPTABLES="/sbin/iptables"
```

Začínáme s definicí pravidel iptables:

```
# Funkce dále umožňuje volat skript(start,stop,restart,edit)
function iptables_create()
{
```

Zapneme směrování paketů mezi síťovými rozhraními:

```
echo "1" > /proc/sys/net/ipv4/ip_forward
```

Nastavíme implicitní politiku pravidel v řetězcích jádra na zahazování paketů:

```
# Implicitni politikou je zahazovat nepovolené pakety
$IPTABLES -P INPUT DROP
$IPTABLES -P OUTPUT DROP
$IPTABLES -P FORWARD DROP
```

PREROUTING - Nastavíme přesměrování určitých portů na počítače ve vnitřní síti. Konkrétně všechna připojení na port 80 (http) jsou směrována na port 3128 (transparentní proxy - squid). Port 3389 (rdp) pro terminálová připojení je směrován na počítač s IP 192.168.1.2 a port 22 (ssh) na počítač s IP 10.0.0.10 :

```
# Řetězec PREROUTING v NAT tabulce

# Odchozí HTTP požadavky (na port 80 s výjimkou lokálního serveru)
# budou přesměrovány na lokálního squida (na portu 3128) ve funkci
transparentní proxy cache.
$IPTABLES -t nat -A PREROUTING -p tcp -i ! $INET_IFACE --dport 80 -j
```

```
REDIRECT --to-port 3128

$IPTABLES -A PREROUTING -t nat -p tcp -d $INET_IP --dport 3389 -j DNAT
--to 192.168.1.2 #rdp na příjem w2k3

$IPTABLES -A PREROUTING -t nat -p tcp -d $INET_IP --dport 2022 -j DNAT
--to 10.0.0.10:22 #ssh na notebook administrátora
```

Nastavení řetězce FORWARD pro směrování:

```
# Paket navazuje spojení, ale nemá nastavený příznak SYN, pryč s ním
# všechna rozhraní
$IPTABLES -A FORWARD -p tcp ! --syn -m state --state NEW -j DROP

# internetové rozhraní chráníme před SYN flood

# prvně loguji
$IPTABLES -A FORWARD -p tcp -i $INET_IFACE --tcp-flags SYN,FIN SYN,FIN
-j LOG -m limit --limit 10/m --log-prefix="bogus packet: "

# pak pakety zahazuji
$IPTABLES -A FORWARD -p tcp -i $INET_IFACE --tcp-flags SYN,FIN SYN,FIN
-j DROP


# Umožnit přesměrování portu na stanici ve vnitřní síti

# rdp
$IPTABLES -A FORWARD -i $INET_IFACE -o $LAN1_IFACE -p tcp -d
192.168.1.2 --dport 3389 -j ACCEPT

# ssh
$IPTABLES -A FORWARD -i $INET_IFACE -o $WLAN_IFACE -p tcp -d 10.0.0.10
--dport 22 -j ACCEPT

# Směrování z vnitřní sítě ven neomezujeme
$IPTABLES -A FORWARD -i $LAN1_IFACE -j ACCEPT
$IPTABLES -A FORWARD -i $WLAN_IFACE -j ACCEPT

# Směrování mezi sítěmi taktéž neomezujeme
$IPTABLES -A FORWARD -i $WLAN_IFACE -o $LAN1_IFACE -j ACCEPT
$IPTABLES -A FORWARD -i $LAN1_IFACE -o $WLAN_IFACE -j ACCEPT


# Směrování z internetu do vnitřní sítě pouze pro navázaná spojení
# (stavový firewall)

# z internetového rozhraní eth0 na vnitřní rozhraní eth1 (kabel)
$IPTABLES -A FORWARD -i $INET_IFACE -o $LAN1_IFACE -m state --state
ESTABLISHED,RELATED -j ACCEPT

# z internetového rozhraní eth0 na vnitřní rozhraní ath0 (wiFi)
$IPTABLES -A FORWARD -i $INET_IFACE -o $WLAN_IFACE -m state --state
ESTABLISHED,RELATED -j ACCEPT


# pokud paket nevyhoví žádné výše uvedené podmínce prvně bude logován
# a po té spadne do výchozího pravidla DROP. Logování je omezeno na
# (12 x 5 pkt/hod) aby nedošlo k nadměrnému zvětšování log souboru
```

```
$IPTABLES -A FORWARD -m limit --limit 12/h -j LOG --log-prefix
"forward drop: "
```

Nastavení řetězce INPUT:

```
# Paket navazuje spojení, ale nemá nastavený příznak SYN, pryč s ním
$IPTABLES -A INPUT -p tcp ! --syn -m state --state NEW -j DROP

# Zabránění portscan s nastaveným SYN,FIN
$IPTABLES -A INPUT -p tcp -i $INET_IFACE --tcp-flags SYN,FIN SYN,FIN
-j LOG -m limit --limit 10/m --log-prefix="bogus packet: "
$IPTABLES -A INPUT -p tcp -i $INET_IFACE --tcp-flags SYN,FIN SYN,FIN
-j DROP

# Pravidla pro povolené služby
$IPTABLES -A INPUT -i $INET_IFACE -p TCP --dport ftp -j ACCEPT#FTP
$IPTABLES -A INPUT -i $INET_IFACE -p TCP --dport 22 -j ACCEPT #SSH
$IPTABLES -A INPUT -i $INET_IFACE -p TCP --dport 25 -j ACCEPT #SMTP
$IPTABLES -A INPUT -i $INET_IFACE -p UDP --dport 53 -j ACCEPT #DNS UDP
$IPTABLES -A INPUT -i $INET_IFACE -p TCP --dport 53 -j ACCEPT #DNS TCP
$IPTABLES -A INPUT -i $INET_IFACE -p TCP --dport 80 -j ACCEPT #WWW
$IPTABLES -A INPUT -i $INET_IFACE -p TCP --dport 81 -j ACCEPT #WWW
$IPTABLES -A INPUT -i $INET_IFACE -p TCP --dport 5060 -j ACCEPT # SIP
$IPTABLES -A INPUT -i $INET_IFACE -p UDP --dport 5060 -j ACCEPT # SIP
$IPTABLES -A INPUT -i $INET_IFACE -p TCP --dport 993 -j ACCEPT#IMAPSSL
$IPTABLES -A INPUT -i $INET_IFACE -p TCP --dport 995 -j ACCEPT#POP3SSL
$IPTABLES -A INPUT -i $INET_IFACE -p TCP --dport 443 -j ACCEPT #HTTPS
$IPTABLES -A INPUT -i $INET_IFACE -p TCP --dport 5000 -j ACCEPT#FREENX
$IPTABLES -A INPUT -i $INET_IFACE -p TCP  --dport 873 -j ACCEPT #rsync

# Službu AUTH není dobré filtrovat pomoci DROP, protože to může
# vést k prodlevám při navazování některých spojení. Proto jej
# sice zamítneme, ale tak, aby nedošlo k nežádoucím prodlevám.

$IPTABLES -A INPUT -i $INET_IFACE -p TCP --dport 113 -j REJECT --
reject-with tcp-reset #AUTH server

# Propouštíme pouze ICMP ping
$IPTABLES -A INPUT -i $INET_IFACE -p ICMP --icmp-type echo-request -j
ACCEPT

# Loopback není radno omezovat
$IPTABLES -A INPUT -i $LO_IFACE -j ACCEPT

# Stejně jako pakety z lokálních sítí, jsou-li určeny pro nás
$IPTABLES -A INPUT -i $LAN1_IFACE -d $LAN1_IP -j ACCEPT
$IPTABLES -A INPUT -i $LAN1_IFACE -d $INET_IP -j ACCEPT
$IPTABLES -A INPUT -i $WLAN_IFACE -d $WLAN_IP -j ACCEPT
$IPTABLES -A INPUT -i $WLAN_IFACE -d $INET_IP -j ACCEPT

# Broadcasty na lokálním rozhraní jsou také naše (hromadné rozesilání,
# samba)

$IPTABLES -A INPUT -i $LAN1_IFACE -d $LAN1_BCAST -j ACCEPT
$IPTABLES -A INPUT -i $WLAN_IFACE -d $LAN1_BCAST -j ACCEPT

# Na portu UDP 67 naslouchá DHCP daemon
$IPTABLES -A INPUT -i $LAN1_IFACE -p udp --dport 67 -j ACCEPT
$IPTABLES -A INPUT -i $LAN2_IFACE -p udp --dport 67 -j ACCEPT
$IPTABLES -A INPUT -i $WLAN_IFACE -p udp --dport 67 -j ACCEPT
```

```
# Pakety od navázaných spojení propouštíme
$IPTABLES -A INPUT -d $INET_IP -m state --state ESTABLISHED,RELATED -j
ACCEPT

# pokud paket nevyhoví žádné výše uvedené podmínce prvně bude logován
# a po té spadne do výchozího pravidla DROP. Logování je omezeno
# na (12 x 5 pkt/hod) aby nedošlo k nadměrnému zvětšování log souboru

$IPTABLES -A INPUT -m limit --limit 12/h -j LOG --log-prefix "INPUT
drop: "
```

Nastavení řetězce OUTPUT

```
# TOS příznaky slouží k optimalizaci datových cest. Pro ssh, ftp a
# telnet požadujeme minimální zpoždění. Pro ftp-data zase maximální
# propostnost

$IPTABLES -t mangle -A OUTPUT -o $INET_IFACE -p tcp --sport ssh -j TOS
--set-tos Minimize-Delay
$IPTABLES -t mangle -A OUTPUT -o $INET_IFACE -p tcp --dport ssh -j TOS
--set-tos Minimize-Delay
$IPTABLES -t mangle -A OUTPUT -o $INET_IFACE -p tcp --sport ftp -j TOS
--set-tos Minimize-Delay
$IPTABLES -t mangle -A OUTPUT -o $INET_IFACE -p tcp --dport ftp -j TOS
--set-tos Minimize-Delay
$IPTABLES -t mangle -A OUTPUT -o $INET_IFACE -p tcp --dport 5060 -j
TOS --set-tos Minimize-Delay
$IPTABLES -t mangle -A OUTPUT -o $INET_IFACE -p udp --dport 5060 -j
TOS --set-tos Minimize-Delay
$IPTABLES -t mangle -A OUTPUT -o $INET_IFACE -p tcp --dport telnet -j
TOS --set-tos Minimize-Delay
$IPTABLES -t mangle -A OUTPUT -o $INET_IFACE -p tcp --sport ftp-data
-j TOS --set-tos Maximize-Throughput

# Povolíme vše co chceme aby odešlo z ven. Implicitně všechna rozhraní
$IPTABLES -A OUTPUT -s $LO_IP -j ACCEPT
$IPTABLES -A OUTPUT -s $LAN1_IP -j ACCEPT
$IPTABLES -A OUTPUT -s $WLAN_IP -j ACCEPT
$IPTABLES -A OUTPUT -s $INET_IP -j ACCEPT

# Povolíme DHCP broadcasty na rozhraních vnitřní sítě
$IPTABLES -A OUTPUT -o $LAN1_IFACE -p UDP --dport 68 --sport 67 -j
ACCEPT
$IPTABLES -A OUTPUT -o $WLAN_IFACE -p UDP --dport 68 --sport 67 -j
ACCEPT

# Ostatní pakety logujeme (neměly by být žádné takové)
$IPTABLES -A OUTPUT -j LOG --log-prefix "OUTPUT drop: "
```

Nastavení řetězce POSTROUTING

```
# IP maškaráda - SNAT
# NATujeme

$IPTABLES -t nat -A POSTROUTING -o $INET_IFACE -j SNAT --to $INET_IP

# TOS příznaky slouží k optimalizaci datových cest. Pro ssh, ftp
# a telnet požadujeme minimální zpoždění. Pro ftp-data zase maximální
```

```
# propostnost
$IPTABLES -t mangle -A PREROUTING -p tcp --sport ssh -j TOS --set-tos
Minimize-Delay
$IPTABLES -t mangle -A PREROUTING -p tcp --sport ftp -j TOS --set-tos
Minimize-Delay
$IPTABLES -t mangle -A PREROUTING -p tcp --dport telnet -j TOS --set-
tos Minimize-Delay
$IPTABLES -t mangle -A PREROUTING -p tcp --sport ftp-data -j TOS --
set-tos Maximize-Throughput
```

Závěrečná nastavení skriptu. Tato funkce se volá při uvolňování rozhraní a maže veškerá nastavení iptables:

```
}
function delete_chain() {
        echo -n "$1/$2: ";
        while [ -z "`$IPTABLES -t $1 -D $2 1 2>&1 `" ]; do
                echo -n "#"
        done
        echo " OK";
}
function iptables_delete()
{
    $IPTABLES -t filter -P INPUT ACCEPT
    $IPTABLES -t filter -P OUTPUT ACCEPT
    $IPTABLES -t filter -P FORWARD ACCEPT

    delete_chain filter INPUT;
    delete_chain filter OUTPUT;
    delete_chain filter FORWARD;
    delete_chain filter IN_FW;
    delete_chain filter logdrop;
    delete_chain filter syn-flood;

    $IPTABLES -X IN_FW 2> /dev/null;
    $IPTABLES -X spoofing 2> /dev/null;
    $IPTABLES -X syn-flood 2> /dev/null;

    $IPTABLES -t nat -P PREROUTING ACCEPT
    $IPTABLES -t nat -P OUTPUT ACCEPT
    $IPTABLES -t nat -P POSTROUTING ACCEPT

    delete_chain nat PREROUTING;
    delete_chain nat OUTPUT;
    delete_chain nat POSTROUTING;

    $IPTABLES -t mangle -P PREROUTING ACCEPT
    $IPTABLES -t mangle -P OUTPUT ACCEPT

    delete_chain mangle PREROUTING;
    delete_chain mangle OUTPUT;
}
```

Další část zpracovává parametry start, stop, restart, edit a stat při volání skriptu:

```
start()
```

```
    {
        ebegin "Starting firewall..."
    #    echo "Starting firewall..."
        iptables_create
        eend $?
    }

    stop() {
    #vypinam shaping
    #    /sbin/tc qdisc del dev eth0 root
        iptables_delete
    }

    restart() {
      stop
      sleep 2
      start
    }

    status() {
      $IPTABLES --list | less
    }

    edit() {
      mcedit /etc/init.d/fw
    }

# Konec skriptu fw.
```

PŘÍLOHA P 2: SEZNAM PORTŮ A PŘIRAZENÝCH SLUŽEB.

Známé porty v rozsahu 0–1023 [31]:

Port	*Popis (anglicky)*	*status*
0/TCP,UDP	Reserved; do not use (but is a permissible source port value if the sending process does not expect messages in response)	Oficiální
1/TCP,UDP	TCPMUX (TCP port service multiplexer)	Oficiální
5/TCP,UDP	RJE (Remote Job Entry)	Oficiální
7/TCP,UDP	ECHO protocol	Oficiální
9/TCP,UDP	DISCARD protocol	Oficiální
11/TCP,UDP	SYSTAT protocol	
13/TCP,UDP	DAYTIME protocol	Oficiální
17/TCP,UDP	QOTD (Quote of the Day) protocol	Oficiální
18/TCP,UDP	Message Send Protocol	Oficiální
19/TCP,UDP	CHARGEN (Character Generator) protocol	Oficiální
20/TCP	FTP - data port	Oficiální
21/TCP	FTP - control (command) port	Oficiální
22/TCP,UDP	SSH (Secure Shell) - used for secure logins, file transfers (scp, sftp) and port forwarding	Oficiální
23/TCP,UDP	Telnet protocol - unencrypted text communications	Oficiální
25/TCP,UDP	SMTP - used for e-mail routing between mailservers E-mails	Oficiální
26/TCP,UDP	RSFTP - A simple FTP-like protocol	Neoficiální
35/TCP,UDP	QMS Magicolor 2 printer	Neoficiální
37/TCP,UDP	TIME protocol	Oficiální
38/TCP,UDP	Route Access Protocol	Oficiální
39/TCP,UDP	Resource Location Protocol	Oficiální
41/TCP,UDP	Graphics	Oficiální
42/TCP,UDP	Host Name Server	Oficiální
43/TCP	WHOIS protocol	Oficiální
49/TCP,UDP	TACACS Login Host protocol	Oficiální
53/TCP,UDP	DNS (Domain Name System)	Oficiální
57/TCP	MTP, Mail Transfer Protocol	
67/UDP	BOOTP (BootStrap Protocol) server; also used by DHCP (Dynamic Host Configuration Protocol)	Oficiální
68/UDP	BOOTP client; also used by DHCP	Oficiální
69/UDP	TFTP (Trivial File Transfer Protocol)	Oficiální
70/TCP	Gopher protocol	Oficiální
79/TCP	Finger protocol	Oficiální
80/TCP	HTTP (HyperText Transfer Protocol) - used for transferring web pages	Oficiální
81/TCP	Torpark - Onion routing ORport	Neoficiální
82/UDP	Torpark - Control Port	Neoficiální
88/TCP	Kerberos - authenticating agent	Oficiální
101/TCP	HOSTNAME	
102/TCP	ISO-TSAP protocol	
107/TCP	Remote Telnet Service	
109/TCP	POP, Post Office Protocol, version 2	
110/TCP	POP3 (Post Office Protocol version 3) - used for retrieving E-mails	Oficiální
111/TCP,UDP	SUNRPC protocol	

Port	*Popis (anglicky)*	*status*
113/TCP	ident - old server identification system, still used by IRC servers to identify its users	Oficiální
115/TCP	SFTP, Simple File Transfer Protocol	
117/TCP	UUCP-PATH	
118/TCP,UDP	SQL Services	Oficiální
119/TCP	NNTP (Network News Transfer Protocol) - used for retrieving newsgroups messages	Oficiální
123/UDP	NTP (Network Time Protocol) - used for time synchronization	Oficiální
135/TCP,UDP	EPMAP / Microsoft RPC Locator Service	Oficiální
137/TCP,UDP	NetBIOS NetBIOS Name Service	Oficiální
138/TCP,UDP	NetBIOS NetBIOS Datagram Service	Oficiální
139/TCP,UDP	NetBIOS NetBIOS Session Service	Oficiální
143/TCP,UDP	IMAP4 (Internet Message Access Protocol 4) - used for retrieving E-mails	Oficiální
152/TCP,UDP	BFTP, Background File Transfer Program	
153/TCP,UDP	SGMP, Simple Gateway Monitoring Protocol	
156/TCP,UDP	SQL Service	Oficiální
158/TCP,UDP	DMSP, Distributed Mail Service Protocol	
161/TCP,UDP	SNMP (Simple Network Management Protocol)	Oficiální
162/TCP,UDP	SNMPTRAP	Oficiální
170/TCP	Print-srv	
179/TCP	BGP (Border Gateway Protocol)	Oficiální
194/TCP	IRC (Internet Relay Chat)	Oficiální
201/TCP,UDP	AppleTalk Routing Maintenance	
209/TCP,UDP	The Quick Mail Transfer Protocol	
213/TCP,UDP	IPX	Oficiální
218/TCP,UDP	MPP, Message Posting Protocol	
220/TCP,UDP	IMAP, Interactive Mail Access Protocol, version 3	
259/TCP,UDP	ESRO, Efficient Short Remote Operations	
264/TCP,UDP	BGMP, Border Gateway Multicast Protocol	
311/TCP	Apple Server-Admin-Tool, Workgroup-Manager-Tool	
318/TCP,UDP	TSP, Time Stamp Protocol	
323/TCP,UDP	IMMP, Internet Message Mapping Protocol	
366/TCP,UDP	SMTP, Simple Mail Transfer Protocol. ODMR, On-Demand Mail Relay	
369/TCP,UDP	Rpc2portmap	Oficiální
371/TCP,UDP	ClearCase albd	Oficiální
384/TCP,UDP	A Remote Network Server System	
387/TCP,UDP	AURP, AppleTalk Update-based Routing Protocol	
389/TCP,UDP	LDAP (Lightweight Directory Access Protocol)	Oficiální
401/TCP,UDP	UPS Uninterruptible Power Supply	Oficiální
411/TCP	Direct Connect Hub port	Neoficiální
427/TCP,UDP	SLP (Service Location Protocol)	Oficiální
443/TCP	HTTPS - HTTP Protocol over TLS/SSL (encrypted transmission)	Oficiální
444/TCP,UDP	SNPP, Simple Network Paging Protocol	
445/TCP	Microsoft-DS (Active Directory, Windows shares, Sasser worm, Agobot, Zobotworm)	Oficiální
445/UDP	Microsoft-DS SMB file sharing	Oficiální
464/TCP,UDP	Kerberos Change/Set password	Oficiální
465/TCP	SMTP over SSL - CONFLICT with registered Cisco protocol	Conflict

Port	*Popis (anglicky)*	*status*
500/TCP,UDP	ISAKMP, IKE-Internet Key Exchange	Oficiální
512/TCP	exec, Remote Process Execution	
512/UDP	comsat, together with biff: notifies users of new c.q. yet unread e-mail	
513/TCP	Login	
513/UDP	Who	
514/TCP	rsh protocol - used to execute non-interactive commandline commands on a remote system and see the screen return	
514/UDP	syslog protocol - used for system logging	Oficiální
515/TCP	Line Printer Daemon protocol - used in LPD printer servers	
517/UDP	Talk	
518/UDP	NTalk	
520/TCP	efs	
520/UDP	Routing - RIP	Oficiální
513/UDP	Router	
524/TCP,UDP	NCP (NetWare Core Protocol) is used for a variety things such as access to primary NetWare server resources, Time Synchronization, etc.	Oficiální
525/UDP	Timed, Timeserver	
530/TCP,UDP	RPC	Oficiální
531/TCP,UDP	AOL Instant Messenger, IRC	Neoficiální
532/TCP	netnews	
533/UDP	netwall, For Emergency Broadcasts	
540/TCP	UUCP (Unix-to-Unix Copy Protocol)	Oficiální
542/TCP,UDP	commerce (Commerce Applications)	Oficiální
543/TCP	klogin, Kerberos login	
544/TCP	kshell, Kerberos Remote shell	
546/TCP,UDP	DHCPv6 client	
547/TCP,UDP	DHCPv6 server	
548/TCP	AFP (Apple Filing Protocol)	
550/UDP	new-rwho, new-who	
554/TCP,UDP	RTSP (Real Time Streaming Protocol)	Oficiální
556/TCP	Remotefs, rfs, rfs_server	
560/UDP	rmonitor, Remote Monitor	
561/UDP	monitor	
563/TCP,UDP	NNTP protocol over TLS/SSL (NNTPS)	Oficiální
587/TCP	email message submission (SMTP) (RFC 2476)	Oficiální
591/TCP	FileMaker 6.0 Web Sharing *(HTTP Alternate, see port 80)*	Oficiální
593/TCP,UDP	HTTP RPC Ep Map	Oficiální
604/TCP	TUNNEL	
631/TCP,UDP	IPP, Internet Printing Protocol	
636/TCP,UDP	LDAP over SSL (encrypted transmission)	Oficiální
639/TCP,UDP	MSDP, Multicast Source Discovery Protocol	
646/TCP	LDP, Label Distribution Protocol	
647/TCP	DHCP Failover Protocol	
648/TCP	RRP, Registry Registrar Protocol	
652/TCP	DTCP, Dynamic Tunnel Configuration Protocol	
654/TCP	AODV, Ad hoc On-Demand Distance Vector	
665/TCP	sun-dr, Remote Dynamic Reconfiguration	Neoficiální
666/UDP	Doom, First online FPS	
674/TCP	ACAP, Application Configuration Access Protocol	

Port	*Popis (anglicky)*	*status*
691/TCP	MS Exchange Routing	Oficiální
692/TCP	Hyperwave-ISP	
695/TCP	IEEE-MMS-SSL	
698/TCP	OLSR, Optimized Link State Routing	
699/TCP	Access Network	
700/TCP	EPP, Extensible Provisioning Protocol	
701/TCP	LMP, Link Management Protocol.	
702/TCP	IRIS over BEEP	
706/TCP	SILC, Secure Internet Live Conferencing	
711/TCP	TDP, Tag Distribution Protocol	
712/TCP	TBRPF, Topology Broadcast based on Reverse-Path Forwarding	
720/TCP	SMQP, Simple Message Queue Protocol	
749/TCP, UDP	kerberos-adm, Kerberos administration	
750/UDP	Kerberos version IV	
782/TCP	Conserver (Console management server)	
829/TCP	CMP (Certificate Management Protocol)	
860/TCP	iSCSI	
873/TCP	rsync File synchronisation protocol	Oficiální
901/TCP	Samba Web Administration Tool (SWAT)	Neoficiální
902	VMware Server[2]	Neoficiální
911/TCP	Network Console on Acid (NCA) - local tty redirection over OpenSSH	
981/TCP	SofaWare Technologies Remote HTTPS management for firewall devices running embedded Checkpoint Firewall-1 software	Neoficiální
989/TCP,UDP	FTP Protocol (data) over TLS/SSL	Oficiální
990/TCP,UDP	FTP Protocol (control) over TLS/SSL	Oficiální
991/TCP,UDP	NAS (Netnews Admin System)	
992/TCP,UDP	Telnet protocol over TLS/SSL	Oficiální
993/TCP	IMAP4 over SSL (encrypted transmission)	Oficiální
995/TCP	POP3 over SSL (encrypted transmission)	Oficiální

Registrované porty v rozsahu 1024–49151 [31]:

Port	*Popis (anglicky)*	*Status*
1080/TCP	SOCKS proxy	Oficiální
1099/TCP	RMI Registry	Oficiální
1099/UDP	RMI Registry	Oficiální
1109/TCP	Kerberos POP	
1167/UDP	phone, conference calling	
1176/TCP	Perceptive Automation Indigo home control server	Oficiální
1194/UDP	OpenVPN	Oficiální
1198/TCP,UDP	The cajo project Free dynamic transparent distributed computing in Java	Oficiální
1211/UDP	Microsoft Office Groove	
1214/TCP	Kazaa	Oficiální
1223/TCP,UDP	TGP: "TrulyGlobal Protocol" aka "The Gur Protocol"	Oficiální
1241/TCP,UDP	Nessus Security Scanner	Oficiální
1248/TCP	NSClient/NSClient++/NC_Net (Nagios)	Neoficiální
1313/TCP	Xbiim (Canvii server) Port	Neoficiální

Port	*Popis (anglicky)*	*Status*
	menandmice.com DNS (not to be confused with standard DNS port).	
1337/TCP	Often used on compromised/infected computers. "1337" a "Leet speak" version of "Elite".	Oficiální
1337/TCP	WASTE Encrypted File Sharing Program	Neoficiální
1352/TCP	IBM Lotus Notes/Domino RPC	Oficiální
1387/TCP	Computer Aided Design Software Inc LM (cadsi-lm)	Oficiální
1387/UDP	Computer Aided Design Software Inc LM (cadsi-lm)	Oficiální
1414/TCP	IBM MQSeries	Oficiální
1431/TCP	RGTP	Oficiální
1433/TCP,UDP	Microsoft SQL database system	Oficiální
1434/TCP,UDP	Microsoft SQL Monitor	Oficiální
1494/TCP	Citrix MetaFrame ICA Client	Oficiální
1512/TCP,UDP	WINS	
1521/TCP	nCube License Manager	Oficiální
1521/TCP	Oracle database default listener	Neoficiální
1524/TCP	ingresslock, ingress	
1526/TCP	Oracle database common alternative for listener	Neoficiální
1533/TCP	IBM Sametime IM - Virtual Places Chat	Oficiální
1547/TCP	Laplink	Oficiální
1547/UDP	Laplink	Oficiální
1550	Gadu-Gadu	
1581/UDP	MIL STD 2045-47001 VMF	Oficiální
1627	iSketch	Neoficiální
1677/TCP	Novell GroupWise clients in client/server access mode	
1701/UDP	l2tp, Layer 2 Tunnelling protocol	
1716/TCP	America's Army MMORPG Default Game Port	Oficiální
1723/TCP	Microsoft PPTP VPN	Oficiální
1723/UDP	Microsoft PPTP VPN	Oficiální
1725/UDP	Valve Steam Client	Neoficiální
1755/TCP	Microsoft Media Services (MMS, ms-streaming)	Oficiální
1755/UDP	Microsoft Media Services (MMS, ms-streaming)	Oficiální
1761/TCP,UDP	cft-0	Oficiální
1761/TCP	Novell Zenworks Remote Control utility	Neoficiální
1762-1768/TCP,UDP	cft-1 to cft-7	Oficiální
1812/UDP	radius, RADIUS authentication protocol	
1813/UDP	radacct, RADIUS accounting protocol	
1863/TCP	MSN Messenger	Oficiální
1900/UDP	Microsoft SSDP Enables discovery of UPnP devices	Oficiální
1935/TCP	Macromedia Flash Communications Server MX	Oficiální
1972/TCP,UDP	InterSystems Caché	Oficiální
1975-77/UDP	Cisco TCO (Documentation)	Oficiální
1984/TCP	Big Brother - network monitoring tool	Oficiální
1985/UDP	Cisco HSRP	Oficiální
2000/UDP	Cisco SCCP (Skinny)	Oficiální
2000/TCP	Cisco SCCP (Skinny)	Oficiální
2002/TCP	Cisco Secure Access Control Server (ACS) for Windows	Neoficiální
2030	Oracle Services for Microsoft Transaction Server	Neoficiální
2031/TCP	mobrien-chat - Mike O'Brien <mike@mobrien.com> November 2004	Oficiální
2031/UDP	mobrien-chat - Mike O'Brien <mike@mobrien.com> November 2004	Oficiální

Port	*Popis (anglicky)*	*Status*
2049/UDP	nfs, NFS Server	Oficiální
2049/UDP	shilp	Oficiální
2053/TCP	knetd, Kerberos de-multiplexor	
2074/TCP	Vertel VMF SA (i.e. App.. SpeakFreely)	Oficiální
2074/UDP	Vertel VMF SA (i.e. App.. SpeakFreely)	Oficiální
2082/TCP	Infowave Mobility Server	Oficiální
2082/TCP	CPanel, default port	Neoficiální
2083/TCP	Secure Radius Service (radsec)	Oficiální
2083/TCP	CPanel default SSL port	Neoficiální
2086/TCP	GNUnet	Oficiální
2086/TCP	WebHost Manager default port	Neoficiální
2087/TCP	WebHost Manager default SSL port	Neoficiální
2095/TCP	CPanel default webmail port	Neoficiální
2096/TCP	CPanel default SSL webmail port	Neoficiální
2181/TCP	EForward-document transport system	Oficiální
2181/UDP	EForward-document transport system	Oficiální
2219/TCP	NetIQ NCAP Protocol	Oficiální
2219/UDP	NetIQ NCAP Protocol	Oficiální
2220/TCP	NetIQ End2End	Oficiální
2220/UDP	NetIQ End2End	Oficiální
2222/TCP	DirectAdmin's default port	Neoficiální
2301/TCP	HP System Management Redirect to port 2381	Neoficiální
2302/UDP	ArmA multiplayer (default for game)	Neoficiální
2302/UDP	Halo: Combat Evolved multiplayer	Neoficiální
2303/UDP	ArmA multiplayer (default for server reporting) *(default port for game +1)*	Neoficiální
2305/UDP	ArmA multiplayer (default for VoN) *(default port for game +3)*	Neoficiální
2381/TCP	HP Insight Manager default port for webserver	Neoficiální
2404/TCP	IEC 60870-5-104	Oficiální
2427/UDP	Cisco MGCP	Oficiální
2447/TCP	ovwdb - OpenView Network Node Manager (NNM) daemon	Oficiální
2447/UDP	ovwdb - OpenView Network Node Manager (NNM) daemon	Oficiální
2593/TCP,UDP	RunUO - Ultima Online Server (http://www.runuo.com)	Neoficiální
2598/TCP	new ICA - when Session Reliability is enabled, TCP port 2598 replaces port 1494	Neoficiální
2612/TCP,UDP	QPasa from MQSoftware (http://www.mqsoftware.com)	Oficiální
2710/TCP	XBT Bittorrent Tracker	Neoficiální
2710/UDP	XBT Bittorrent Tracker experimental UDP tracker extension	Neoficiální
2735/TCP	NetIQ Monitor Console	Oficiální
2735/UDP	NetIQ Monitor Console	Oficiální
2809/TCP	corbaloc:iiop URL, per the CORBA 3.0.3 specification. Also used by IBM WebSphere Application Server Node Agent	Oficiální
2809/UDP	corbaloc:iiop URL, per the CORBA 3.0.3 specification.	
2944/UDP	Megaco Text H.248	Neoficiální
2945/UDP	Megaco Binary (ASN.1) H.248	Neoficiální
2948/TCP	WAP-push Multimedia Messaging Service (MMS)	Oficiální
2948/UDP	WAP-push Multimedia Messaging Service (MMS)	Oficiální
2949/TCP	WAP-pushsecure Multimedia Messaging Service (MMS)	Oficiální

Port	*Popis (anglicky)*	*Status*
2949/UDP	WAP-pushsecure Multimedia Messaging Service (MMS)	Oficiální
2967/TCP	Symantec AntiVirus Corporate Edition	Neoficiální
3000/TCP	Miralix License server	Neoficiální
3000/UDP	Distributed Interactive Simulation (DIS), modifiable default port	Neoficiální
3001/TCP	Miralix Phone Monitor	Neoficiální
3002/TCP	Miralix CSTA	Neoficiální
3003/TCP	Miralix GreenBox API	Neoficiální
3004/TCP	Miralix InfoLink	Neoficiální
3006/TCP	Miralix SMS Client Connector	Neoficiální
3007/TCP	Miralix OM Server	Neoficiální
3050/TCP,UDP	gds_db (Interbase/Firebird)	Oficiální
3074/TCP,UDP	Xbox Live	Oficiální
3128/TCP	HTTP used by web caches and the default port for the Squid cache	Oficiální
3260/TCP	iSCSI target	Oficiální
3305/TCP,UDP	ODETTE-FTP	Oficiální
3306/TCP,UDP	MySQL Database system	Oficiální
3333/TCP	Network Caller ID server	Neoficiální
3389/TCP	Microsoft Terminal Server (RDP) Oficiálníly registered as Windows Based Terminal (WBT)	Oficiální
3396/TCP	Novell NDPS Printer Agent	Oficiální
3689/TCP	DAAP Digital Audio Access Protocol used by Apple's iTunes	Oficiální
3690/TCP	Subversion version control system	Oficiální
3724/TCP	World of Warcraft Online gaming MMORPG	Oficiální
3784/TCP	Ventrilo VoIP program used by Ventrilo	Oficiální
3785/UDP	Ventrilo VoIP program used by Ventrilo	Oficiální
3872/TCP	Oracle Management Remote Agent	Neoficiální
3900/TCP	Unidata UDT OS udt_os	Oficiální
4007/TCP	PrintBuzzer printer monitoring socket server	Neoficiální
4089/UDP	OpenCORE Remote Control Service	Oficiální
4089/TCP	OpenCORE Remote Control Service	Oficiální
4100	WatchGuard Authentication Applet - default port	Neoficiální
4111/TCP,UDP	Xgrid	Oficiální
4111/TCP	Microsoft Office SharePoint Portal Server - default administration port	Neoficiální
4226/TCP,UDP	Aleph One (computer game)	Neoficiální
4569/UDP	Inter-Asterisk eXchange	Neoficiální
4662/TCP,UDP	eMule - port often used	Neoficiální
4664/TCP	Google Desktop Search	Neoficiální
4894/TCP	LysKOM Protocol A	Oficiální
4899/TCP	Radmin remote administration tool (program sometimes used as a Trojan horse)	Oficiální
5000/TCP	commplex-main	Oficiální
5000/TCP	UPnP - Windows network device interoperability	Neoficiální
5001/TCP	Slingbox and Slingplayer	Neoficiální
5003/TCP	FileMaker Filemaker Pro	Oficiální
5004/UDP	RTP Real-time Transport Protocol	Oficiální
5005/UDP	RTP Real-time Transport Protocol	Oficiální
5050/TCP	Yahoo! Messenger Yahoo! Messenger	Oficiální
5051/TCP	ita-agent Symantec Intruder Alert	Oficiální
5060/TCP	Session Initiation Protocol (SIP)	Oficiální

Port	*Popis (anglicky)*	*Status*
5060/UDP	Session Initiation Protocol (SIP)	Oficiální
5061/TCP	Session Initiation Protocol (SIP) over Transport Layer Security (TLS)	Oficiální
5104/TCP	IBM NetCOOL / IMPACT HTTP Service	Neoficiální
5121	Neverwinter Nights and its mods, such as *Dungeon Eternal X*	Neoficiální
5190/TCP	ICQ and AOL Instant Messenger	Oficiální
5222/TCP	XMPP/Jabber - client connection	Oficiální
5223/TCP	XMPP/Jabber - default port for SSL Client Connection	Neoficiální
5269/TCP	XMPP/Jabber - server connection	Oficiální
5351/TCP,UDP	NAT Port Mapping Protocol - client-requested configuration for inbound connections through network address translators	Oficiální
5353/UDP	mDNS - multicastDNS	
5402/TCP,UDP	StarBurst AutoCast MFTP	Oficiální
5432/TCP	PostgreSQL database system	Oficiální
5495/TCP	Applix TM1 Admin server	Neoficiální
5500/TCP	VNC remote desktop protocol - for incoming listening viewer	Neoficiální
5517/TCP	Setiqueue Proxy server client for SETI@Home project	Neoficiální
5555/TCP	Freeciv multiplay port, Hewlett Packard Data Protector, SAP	Neoficiální
5631/TCP	Symantec pcAnywhere	Oficiální
5666/TCP	NRPE (Nagios)	Neoficiální
5667/TCP	NSCA (Nagios)	Neoficiální
5800/TCP	VNC remote desktop protocol - for use over HTTP	Neoficiální
5814/TCP,UDP	Hewlett-Packard Support Automation (HP OpenView Self-Healing Services)	Oficiální
5900/TCP	ARD/VNC remote desktop protocol - regular port	Neoficiální
6000/TCP	X11 - used between an X client and server over the network	Oficiální
6001/UDP	X11 - used between an X client and server over the network	Oficiální
6050/TCP	Brightstor Arcserve Backup Exec	Neoficiální
6051/TCP	Brightstor Arcserve Backup Exec	Neoficiální
6112/TCP	"dtspcd" - a network daemon that accepts requests from clients to execute commands and launch applications remotely	Oficiální
6112/TCP	Blizzard's Battle.net gaming service, ArenaNet gaming service	Neoficiální
6346/TCP,UDP	gnutella-svc (FrostWire, Limewire, Bearshare, etc.)	Oficiální
6347/TCP,UDP	gnutella-rtr	Oficiální
6502/TCP,UDP	Danware Data NetOp Remote Control	Neoficiální
6522/TCP	Gobby (and other libobby-based software)	Neoficiální
6543/UDP	Jetnet - default port that the Paradigm Research & Development Jetnet protocol communicates on	Neoficiální
6566/TCP	SANE (Scanner Access Now Easy) - SANE network scanner daemon	Neoficiální
6619/TCP,UDP	ODETTE-FTP over TLS/SSL	Oficiální
6665-6669/TCP	ircu (Internet Relay Chat server)	Oficiální
6679/TCP	IRC SSL (Secure Internet Relay Chat) - port often used	Neoficiální
6881-6999/TCP,UDP	BitTorrent full range of ports used most often	Neoficiální
6891-6900/TCP,UDP	MSN Messenger (File transfer)	Oficiální
6901/TCP,UDP	MSN Messenger (Voice)	Oficiální
6969/TCP	acmsoda	Oficiální
6969/TCP	BitTorrent tracker port	Neoficiální
7000/TCP	Default port for Azureus's built in HTTPS Bittorrent Tracker	Neoficiální

Port	*Popis (anglicky)*	*Status*
7001/TCP	Default port for BEA WebLogic Server's HTTP server - though often changed during installation	Neoficiální
7002/TCP	Default port for BEA WebLogic Server's HTTPS server - though often changed during installation	Neoficiální
7171	Tibia	
7312/UDP	Sibelius License Server port	Neoficiální
7777/TCP	Default port used by Windows backdoor program tini.exe	Neoficiální
8000/TCP	iRDMI - often mistakenly used instead of port 8080 (The Internet Assigned Numbers Authority (iana.org) Oficiálníly lists this port for iRDMI protocol)	Oficiální
8000/TCP	Common port used for internet radio streams such as those using SHOUTcast	Neoficiální
8002/TCP	Cisco Systems Unified Call Manager Intercluster Port	
8008/TCP	HTTP Alternate	Oficiální
8008/TCP	IBM HTTP Server default administration port	Neoficiální
8010/TCP	XMPP/Jabber File transfers	Neoficiální
8080/TCP	Another HTTP Alternate - commonly used for web proxy and caching server, or for running a web server as a non-root user	Oficiální
8080/TCP	Jakarta Tomcat	Neoficiální
8086/TCP	HELM Web Host Automation Windows Control Panel	Neoficiální
8087/TCP	Hosting Accelerator Control Panel	Neoficiální
8118/TCP	Privoxy web proxy - advertisements-filtering web proxy	Oficiální
8087/TCP	SW Soft Plesk Control Panel	Neoficiální
8200/TCP	GoToMyPC	Neoficiální
8220/TCP	Bloomberg	Neoficiální
8222	VMware Server Management User Interface (insecure web interface)[3]. See also, port 8333	Neoficiální
8291/TCP	Winbox - Default port on a MikroTik RouterOS for a Windows application used to administer MikroTik RouterOS	Neoficiální
8294/TCP	Bloomberg	Neoficiální
8333	VMware Server Management User Interface (secure web interface)[4]. See also, port 8222	Neoficiální
8400	Commvault Unified Data Management [5].	Oficiální
8443/TCP	SW Soft Plesk Control Panel	Neoficiální
8767	TeamSpeak - Default UDP Port	Neoficiální
8880	WebSphere Application Server SOAP Connector port	
8888/TCP,UDP	NewsEDGE server	Oficiální
8888/TCP	Sun Answerbook dwhttpd server (deprecated by docs.sun.com)	Neoficiální
8888/TCP	GNUmp3d HTTP music streaming and web interface port	Neoficiální
9001	cisco-xremote router configuration	Neoficiální
9001	Tor network default port	Neoficiální
9043/TCP	WebSphere Application Server Administration Console secure port	
9060/TCP	WebSphere Application Server Administration Console	
9100/TCP	Jetdirect HP Print Services	Oficiální
9101	Bacula Director	Oficiální
9102	Bacula File Daemon	Oficiální
9103	Bacula Storage Daemon	Oficiální

Port	*Popis (anglicky)*	*Status*
9535/TCP	man, Remote Man Server	
9535	mngsuite - Management Suite Remote Control	Oficiální
9800/TCP,UDP	WebDav Source Port	Oficiální
9800	WebCT e-learning portal	Neoficiální
9999	Hydranode - edonkey2000 telnet control port	Neoficiální
9999	Urchin Web Analytics	Neoficiální
10000	Webmin - web based Linux admin tool	Neoficiální
10000	BackupExec	Neoficiální
10008	Octopus Multiplexer - CROMP protocol primary port, hoople.org	Oficiální
10113/TCP	NetIQ Endpoint	Oficiální
10113/UDP	NetIQ Endpoint	Oficiální
10114/TCP	NetIQ Qcheck	Oficiální
10114/UDP	NetIQ Qcheck	Oficiální
10115/TCP	NetIQ Endpoint	Oficiální
10115/UDP	NetIQ Endpoint	Oficiální
10116/TCP	NetIQ VoIP Assessor	Oficiální
10116/UDP	NetIQ VoIP Assessor	Oficiální
11235	Savage:Battle for Newerth Server Hosting	Neoficiální
11371	OpenPGP HTTP Keyserver	Oficiální
11576	IPStor Server management communication	Neoficiální
12345	NetBus - remote administration tool (often Trojan horse). Also used by NetBuster	Neoficiální
13720/TCP	Symantec NetBackup - bprd (formerly VERITAS)	
13721/TCP	Symantec NetBackup - bpdbm (formerly VERITAS)	
13724/TCP	Symantec Network Utility - vnet (formerly VERITAS)	
13782/TCP	Symantec NetBackup - bpcd (formerly VERITAS)	
13783/TCP	Symantec VOPIED protocol (formerly VERITAS)	
14567/UDP	Battlefield 1942 and mods	Neoficiální
15000/TCP	Wesnoth	
15567/UDP	Battlefield Vietnam and mods	Neoficiální
15345/UDP	XPilot	Oficiální
16384/UDP	Iron Mountain Digital - online backup	Neoficiální
16567/UDP	Battlefield 2 and mods	Neoficiální
19226/TCP	Panda Software AdminSecure Communication Agent	Neoficiální
19813/TCP	4D database Client Server Communication	Neoficiální
20000	Usermin - web based user tool	Oficiální
20720/TCP	Symantec i3 Web GUI server	Neoficiální
24800	Synergy: keyboard/mouse sharing software	Neoficiální
24842	StepMania: Online: *Dance Dance Revolution* Simulator	Neoficiální
26000/TCP,UDP	id Software's *Quake* server,	Oficiální
26000/TCP	CCP's EVE Online Online gaming MMORPG,	Neoficiální
27000/UDP	(through 27006) id Software's *QuakeWorld* master server	Neoficiální
27010	Half-Life and its mods, such as *Counter-Strike*	Neoficiální
27015	Half-Life and its mods, such as *Counter-Strike*	Neoficiální
27374	Sub7's default port. Most script kiddies do not change the default port.	Neoficiální
27500/UDP	(through 27900) id Software's *QuakeWorld*	Neoficiální
27888/UDP	Kaillera server	Neoficiální
27900	(through 27901) Nintendo Wi-Fi Connection	Neoficiální
27901/UDP	(through 27910) id Software's *Quake II* master server	Neoficiální

Port	*Popis (anglicky)*	*Status*
27960/UDP	(through 27969) Activision's *Enemy Territory* and id Software's *Quake III Arena* and *Quake III* and ioquake3 (e.g. *Tremulous*) derived games	Neoficiální
28910	Nintendo Wi-Fi Connection	Neoficiální
28960	Call of Duty 2 Common Call of Duty 2 port - (PC Version)	Neoficiální
29900	(through 29901) Nintendo Wi-Fi Connection	Neoficiální
29920	Nintendo Wi-Fi Connection	Neoficiální
30564/TCP	Multiplicity: keyboard/mouse/clipboard sharing software	Neoficiální
31337/TCP	Back Orifice - remote administration tool (often Trojan horse)	Neoficiální
31337/TCP	xc0r3 - xc0r3 security antivir port	Neoficiální
31415	ThoughtSignal - Server Communication Service (often Informational)	Neoficiální
31456-31458/TCP	TetriNET ports (in order: IRC, game, and spectating)	Neoficiální
32245/TCP	MMTSG-mutualed over MMT (encrypted transmission)	Neoficiální
33434	traceroute	Oficiální
37777/TCP	Digital Video Recorder hardware	Neoficiální
36963	Counter Strike 2D multiplayer port (2D clone of popular CounterStrike computer game)	Neoficiální
43594-43595/TCP	RuneScape	Neoficiální

Dynamické a privátní Ports 49152 to 65535:

Ostatní uživatelské porty [31].