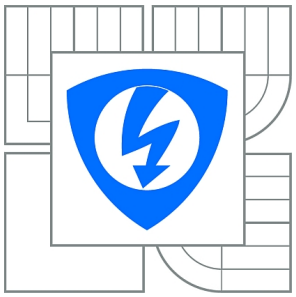
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

ZABEZPEČENÍ OPERAČNÍHO SYSTÉMU LINUX

SECURITY OF LINUX OS

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

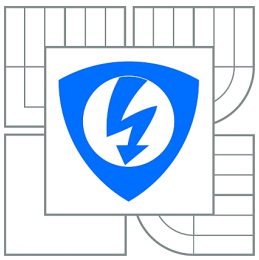
AUTOR PRÁCE — Bc. MILAN POLÁCH

AUTHOR

VEDOUCÍ PRÁCE — Ing. MICHAL VYMAZAL

SUPERVISOR

BRNO 2011



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student: Bc. Milan Polách ***ID:*** 98121
Ročník: 2 ***Akademický rok:*** 2010/2011

NÁZEV TÉMATU:

Zabezpečení operačního systému Linux

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je vytvoření aplikace ve vhodném programovacím jazyku pro nastavení zabezpečení vybrané distribuce Linuxu. Aplikace bude zaměřena na nastavení síťového zabezpečení (analýza provozu, nastavení firewallu, detekce průniku do systému, atd.) a možnosti přidávaní nových profilů na omezení práv aplikací. Aplikace usnadní uživatelům s omezenými znalostmi Linuxu nastavení vhodného zabezpečení systému.

DOPORUČENÁ LITERATURA:

[1] TOXEN, B., Bezpečnost v Linuxu : Prevence a odvrácení napadení systému. [s.l.] : Computer Press, 2003. 876 s. ISBN 80-7226-716-7.
[2] KABELOVÁ, A., DOSTÁLEK, L., Velký průvodce protokoly TCP/IP a systémem DNS . [s.l.] : Computer Press, 2008. 488 s. ISBN 978-80-251-2236

Termín zadání: 7.2.2011 ***Termín odevzdání:*** 26.5.2011

Vedoucí práce: Ing. Michal Vymazal

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

ABSTRAKT

Tato diplomová práce je zaměřena na možnosti lepšího zabezpečení síťového provozu opreračního systému GNU/Linux pomocí vhodného nastavení pravidel Netfilter. Byl vytvořen program pro umožnění snadné konfigurace pravidel pro ip adresy verze 4 i verze 6.Tento program umožňuje nejen nastavit jednotlivá pravidla, ale i zasáhnout do nově požadovaného provozu a rozhodnout, jak s ním bude dále pracováno. V teoretické části práce je nejdříve popsátna sítová komunikace pomocí modelu TCP/IP, dále seznámení s Netfilter a nastínění lokální bezpečnosti. V praktické částí jsou popsány jednotlivé použité technologie a metody pro tvorbu programu.
Výsledkem práce je snadno ovladatelný program umožňující nastavit pravidla firewallu pro i ip adresy verze 6 s možností rozhodování o nově navazovaném síťovém provozu. operačního systému, kteří mají zájem o lepší zabezpečiní svého počítače bez znalosti Netfilter.

KLÍČOVÁ SLOVA

GNU/Linux, Ubuntu, firewall, iptables, Netfilter

ABSTRACT

This thesis is focused on the possibility of better networking security operating system GNU/Linux with an appropriate set of rules Netfilter. There was established a program to allow easy configuration of rules for IP Address versions 4 and 6. This program not only allows to set individual rules, but also interfere with the newly required service and decide, how it will be further worked with. The first is the theoretical part describes the network communication with the model TCP/IP, the following is the introduction of Netfilter and outlining the local security. The practical part describes the various technologies and methods used for programming.
The result of this work is easy to use program to set firewall rules for IP Address versions 6 with the possibility of deciding on the new established network traffic. The program is designed for new users of the operating system, who want to better secure their computer without the knowledge of Netfilter.

KEYWORDS

GNU/Linux, Ubuntu, firewall, iptables, Netfilter

POLÁCH, Milan *Zabezpečení operačního systému Linux*: diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2011. 53 s. Vedoucí práce byl Ing. Michal Vymazal

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Zabezpečení operačního systému Linux“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

Brno ....

(podpis autora)

OBSAH

Úvod **8**

1 Linux **9**

2 Zabezpečení síťového provozu **11**
2.1 Model TPC/IP 11
2.1.1 Vrstva sítového rozhraní 11
2.1.2 Síťová vrstva 11
2.1.3 Transportní vrstva 16
2.1.4 Aplikační vrstva 17
2.2 Firewall 19
2.2.1 Netfilter - Iptables 20
2.2.2 Netfilter-layer7 23
2.2.3 Analýza zabezpečení 23

3 Lokální bezpečnost **25**
3.1 Uživatelé a jejich práva 25
3.1.1 Přístupová práva k souborům 25
3.2 Bezpečná hesla 26
3.2.1 Stínová hesla 27
3.3 Apparmor 27
3.4 Logování událostí 28
3.4.1 Programy pro kontrolu systému 29

4 Návrh zabezpečení síťového provozu **30**
4.1 Současné možnosti 30
4.1.1 Využití Apparmor 30
4.1.2 Využití Selinux 30
4.1.3 Využití NFqueue 31
4.2 Programové řešení firewallu 32
4.2.1 Python 32
4.2.2 Pygtk 32
4.2.3 Python Nfqueue 33
4.3 Návrh firewallu 34
4.3.1 Semag 34
4.3.2 Hlavní okno 35
4.3.3 Pomocná okna 38
4.3.4 Daemon 40

4.3.5 Netfilter pravidla 41
4.3.6 Začlenění do systému 45
4.3.7 Možnost využití na jiných OS než Ubuntu 46
4.4 Testování 46
4.4.1 Omezení síťové komunikace programů 47
4.4.2 Časové zpoždění 47
4.4.3 Testování pravidel 48

Závěr **50**

Literatura **51**

Seznam symbolů, veličin a zkratek **53**

SEZNAM OBRÁZKŮ

1.1 Oblíbenost jednotlivých desktopových distribucí OS Linux za rok 2010 [2] 9
2.1 Hlavička IPv4 12
2.2 Hlavička IPv6 14
2.3 Hlavička TCP 16
2.4 Hlavička UDP 17
4.1 Gksudo a získání práv 34
4.2 Hlavní okno programu Semag 35
4.3 Načítání a ukládání pravidel 36
4.4 Nastavení pravidel Ipv4 a Ipv6 37
4.5 Zobrazení probíhající komunikace 38
4.6 Přednastavené nastavení pravidel 39
4.7 Pokročilé nastavení pravidel 39
4.8 Začlenění programu Semag do menu OS. 45

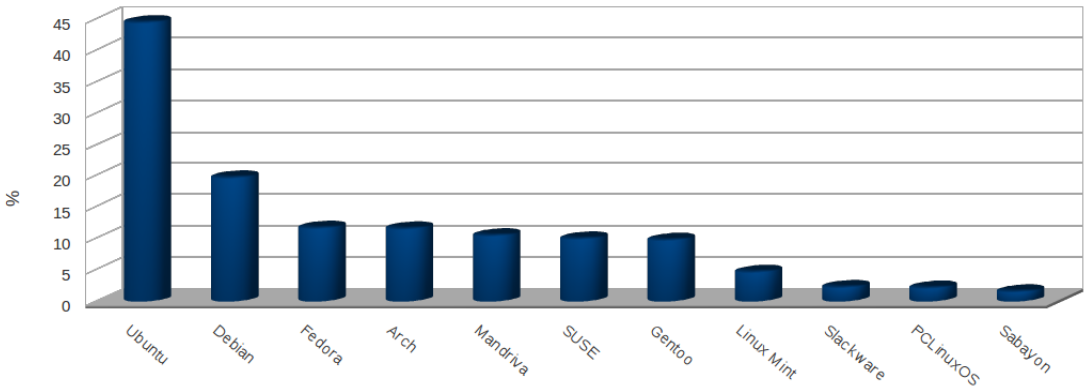
ÚVOD

V současné době neustále roste četnost síťových útoků nejen na firmy, ale také přímo na data jednotlivých uživatelů, které mají uložena na svých počítačích. Pokud se zaměříme zejména na nové uživatele, ve většině případech nevědí, jak zlepšit své zabezpečení nebo se o to dokonce ani nezajímají. U dnes nejpoužívanějších operačních systémů firmy Microsoft existuje mnoho svobodných i komerčních programů na zabezpečení počítače. U méně početné skupiny operačních systémů GNU/Linux je také dostupné velké množství projektů zabývající se nastavením pravidel Netfilter. Ve většině případech se jedná jen o různě propracované generátory pravidel bez možnosti detekce nových stavů v síti. Je také stále více nasazován nový protokol IPv6. Je tedy možné, aby uživatel nastavil vhodně zabezpečení pro IPv4, ale zcela neúmyslně zapomněl také zabezpečit IPv6. Tím mohou vznikat velká bezpečnostní rizika.

Cílem této práce je prostudování a seznámení se s problematikou zabezpečení síťového provozu na vybrané distribuci GNU/Linux. Dále navrhnout a vytvořit program umožňující novým nebo méně zkušeným uživatelům vhodně zabezpečit svůj počítač pro protokoly IPv4 a IPv6 a informovat uživatele o nových stavech na síťovém rozhraní.

1 LINUX

Linux je víceuživatelský systém vycházející ze standartu UNIX. První linuxové jádro navrhl finský student Linus Torvalds v roce 1991. Linus pracoval na operačním systému Minix. Pro tento systém nebylo možné sehnat zdrojové kódy, proto začal pracovat na vlastním systému. V době, kdy vznikal Linux, se už pracovalo na projektu s názvem GNU (GNU's Not Unix) s cílem vytvořit nový operační systém založený na UNIX, který bude využívat svobodné programy. Pro tento účel byla vytvořena licence GNU GPL, kterou využívají projekty GNU, Linux a pod touto licencí mohou být šířeny jejich části. Pro projekt GNU mělo být využito jádro Hurdu, ale z důvodu rychlejšího vývoje se využilo jádro Linux a vznikl projekt GNU/Linux dnes známý spíše pod názvem Linux.



Obr. 1.1: Oblíbenost jednotlivých desktopových distribucí OS Linux za rok 2010 [2]

Projekty vytvářené na základě GNU/Linux se nazývají distribuce. V dnešní době je několik stovek živých distribucí (živá distribuce je taková , kde se jejím vývojem zabývají nejméně 3 vývojáři) a celkový počet je v řádu několika tisíc. Vetšinou se jedná o odnože hlavních distribucí. Mezi nejznámější distribuce patří :

- Debian
- Ubuntu
- Red Hat
- Fedora
- Gentoo
- Madriva

- Centos
- OpenSuse

V teto práci se budeme zabývat zabezpečením distribuce Ubuntu, která vychází z distribuce Debian. Ubuntu je vyvíjeno firmou Canonical Ltd, kterou vlastní Mark Shuttleworthem. Jméno dostala distribuce podle jihoafrického pojmu ubuntu, což znamená lidskost ostatním“ [3] Ubuntu je dnes jedna z nejvíce rozšířených distribucí. Její zaměření je hlavně na osobní počítače a notebooky a lze ji využít i pro serverové řešení. Filozofií distribuce je poskytnutí Ubuntu pod otevřenou licencí s využitím svobodného softwaru. Z tohoto důvodu existuje několik druhů zdrojů pro získání softwaru a to:

- main – zde jsou základní balíčky, které jsou testovány a podporovány vývojáři [3]
- restricted – obsahuje balíčky, které jsou potřebné pro správný chod systému, ale nejsou dostupné zdrojové kódy. Patří sem například ovladače grafických karet. Je zde omezená podpora vývojářů, protože ti nemohou upravit zdrojové kódy. [3]
- universe – zde je software, který může, ale nemusí dodržovat restriktivní licenci a není podporován vývojáři Canonical Ltd. [3]
- multiverse – obsahuje balíčky, které spadají pod nesvobodný software. [3]

Na ubuntu můžeme využít několik grafických prostředí. Mezi nejznámější patří Gnome obsažené v distribuci Ubuntu, Kde v distribuci Kubuntu a Xfce s Xubuntu. Ubuntu je vhodné pro uživatele, kteří začínají s Linuxem. Instalace je jednoduchá a čistý nainstalovaný software je hned připravený k použití. Dále se seznámíme jaké má takto čistě nainstalovaný systém bezpečností mezery a jaké kroky jsou nutné k jejich odstranění.

2 ZABEZPEČENÍ SÍŤOVÉHO PROVOZU

V této části se budeme zabývat jaké jsou nežádoucí možnosti pro proniknutí do systému pomocí síťového rozhraní. Nebudeme zde řešit situaci, kdy už se útočníkovi podařilo dostat do našeho systému a pokusy o získání informací nebo root práv. Tyto problémy budou řešeny v následujících kapitolách.

Na začátku si popíšeme, jak vypadá sítová komunikace, jak vypadají jednotlivé přenášené zprávy a co obsahují. Následně se seznámíme s možností, jak analyzovat probíhající komunikaci a jak nastavit firewall pomocí iptables. Seznámíme se s log soubory a jejich obsahem, ze kterých lze zjistit co se v systému děje a provést potřebné protiopatření. Nakonec zjistíme, jaké služby nám v systému běží a jaká je jejich úroveň zabezpečení.

2.1 Model TPC/IP

Aby bylo možné definovat síťovou komunikaci, byl vytvořen model TCP/IP (Transmission Control Protocol / Internet Protocol) Je to soubor definic o podobě síťové komunikace. Protokol TCP/IP má čtyři vrstvy, mezi kterými je definováno, jakým způsobem budou mezi sebou komunikovat a přenášet zprávy. Dále by měly jednotlivé vrstvy poskytovat služby vyšší vrstvy a využívat služeb nižších vrstev. V TCP/IP se o provoz starají tyto vrstvy :

- Vrstva sítového rozhraní
- Síťová vrstva
- Transportní vrstva
- Aplikační vrstva

2.1.1 Vrstva sítového rozhraní

Jedná se o rozhraní, které je přímo připojené k fyzickému médiu po kterém jsou přenášena data. Má na starosti ovládání cesty paketu v síti. Tato vrstva se může lišit podle použitého druhu sítě, kde je protokol TCP/IP využíván. [4]

2.1.2 Síťová vrstva

Tato vrstva má na starost přenos dat mezi výchozí a cílovou stanicí. K datum je přidána hlavička s cílovou a zdrojovou adresou. Vrstva se nestará o chybovost dat. O odstranění chyb se starají vyšší vrstvy. Na této vrstvě jsou používány protokoly IP, ARP, ICMP.

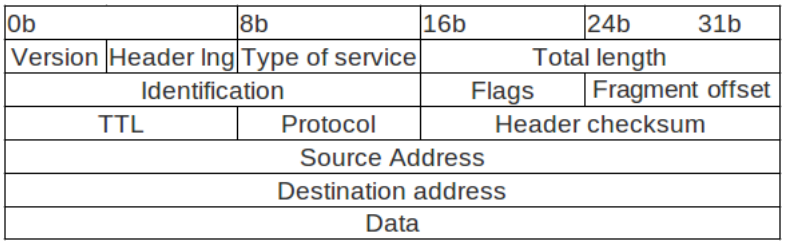
IP(Internet Protocol)

Využívá se pro přenos dat v sítích využívajících paketový přenos. Protokol se nezajímá jak a zda budou data doručena na cílovou stanici, ale jen o jejich vyslání. Spoléhá se na aktivní prvky jako jsou switche nebo routery, jejichž úkolem je zaslat data k cíli tou nejlepší možnou známou cestou. Jedná se o nespolehlivou službu, kde není jisté, zda data dorazí nebo dorazí vícekrát. K zajištění spolehlivosti se může využít například protokol vyšší vrstvy TCP. IP umožňuje spojit lokální sítě do jednoho celku. Je tvořen z několika dílčích částí a to samotným protokolem IP, služebními protokoly ICMP, IGMP, ARP a RARP [4]

IP datagram

Pro přesnost dat potřebujeme znát údaje o odesílateli a příjemci . K tomu se využívají IP adresy. Dnes se využívají dvě verze a to dnes nejvíce rozšířená IPv4, která využívá 4 bajtovou délku a verze IPv6, která má 6 bajtů. V současné době je adresní prostor využívající IPv4 téměř vyčerpán a začíná se stále více využívat IPv6.

IPv4



Obr. 2.1: Hlavička IPv4

Ukázka části hlavičky IPv4 získaná programem Wireshark.

```
Internet Protocol, Src: 147.229.218.35 (147.229.218.35)
Dst:     255.255.255.255 (255.255.255.255)
Version: 4
Header length: 20 bytes
Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
Total Length: 155
Identification: 0x189d (6301)
Flags: 0x00
Fragment offset: 0
```

```
Time to live: 128
Protocol: UDP (0x11)
Header checksum: 0xb3ac [correct]
Source: 147.229.218.35 (147.229.218.35)
Destination: 255.255.255.255 (255.255.255.255)
```

Datagram se skládá z několika hlavních částí a to je záhlaví, kde jsou uvedeny potřebné informace k doručení a části kde jsou data uložena. Nyní si popíšeme jednotlivé části hlavičky datagramu. Pro lepší přehlednost byl zachycen jeden datagram pomocí programu Wireshark

Version – udává nám jestli je použit protokol IPv4 nebo IPv6. **Header length** – sděluje nám informace o velikosti hlavičky.

Type of service – využívá se pro nastavení nejvhodnějších parametrů pro daný. Dnes se využívá pro QoS (Quality of Service).

Total length – udává velikost celého datagramu. Maximální velikost je 65535 bajtů. [4]

Identification – sděluje nám pořadové číslo, které bylo přiděleno odesílatelem.

Time to live – TTL udává přes kolik aktivních prvků může datagram projít. Při průchodu aktivním prvkem se sníží doba života. Pokud datagramu TTL vyprší, bude zahozen. Je to ochrana proti zahlcení sítě bludnými datagramy.

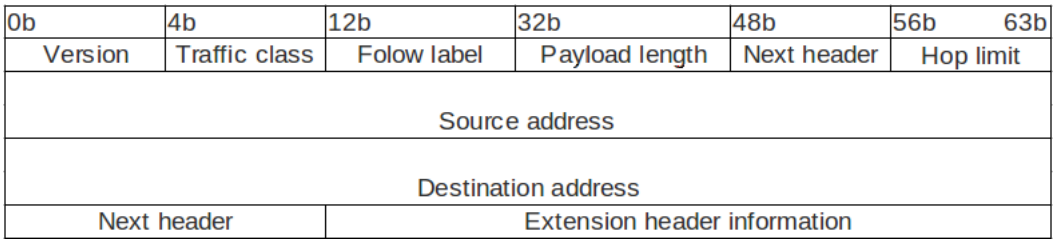
Protocol – udává jaké protokoly vyšších vrstev jsou využity. Můžeme se zde setkat s TCP, UDP nebo s jedním ze služebních protokolů.

Header checksum – udává číslo, které bylo vygenerováno na posledním aktivním prvku. Na následujícím aktivním prvku se provede stejná operace a pokud se čísla shodují, lze předpokládat ,že hlavička neobsahuje chyby . Kontrolní součet hlavičky se mění v průběhu cesty, protože na aktivních prvcích dochází ke změně TTL a tím vzniká nutnost znovu provést kontrolní součet záhlaví.

Source address a Destination address – udává zdrojovou a cílovou adresu ve čtyř bajtovém formátu pro každou stranu. Z příkladu lze vyčíst, že zdrojová adresa je 147.229.218.35 a datagram je určen pro všechny stanice v dané síti.

IPv6

Oproti verzi IPv4 má verze IPv6 hlavičku menší. Byly vynechány informace jako jsou například Fragnemt ofset nebo Flags. Vynechání je způsobeno tím, že u aktivních



Obr. 2.2: Hlavička IPv6

prvků už nedochází k fragmentaci. O tu se starají zdrojová a koncová zařízení. Dále byla odstraněna položka Header length, protože verze IPv6 má pevnou hlavičku o velikosti 40 bytů. Místo toho byla přidána informace o velikosti datové oblasti. Mezi další prvky, které byly odebrány patří Header Checksum. O zabezpečení se starají protokoly vyšší vrstvy. V dnešních sítích už nedochází k chybám tak často a pokud by se nějaká vyskytla v hlavičce IPv6, může v nejhorším případě nastat situace, že data budou doručena na jinou adresu.
Ukázka části hlavičky IPv6 získané programem Wireshark

```
    Internet Protocol Version 6
0110 .... = Version: 6
.... 0000 0000 .... .... .... .... .... = Traffic class: 0x00000000
.... .... .... 0000 0000 0000 0000 0000 = Flowlabel: 0x00000000
Payload length: 460
Next header: UDP (0x11)
Hop limit: 1
Source: fe80095:ad71:3c35:1df9 (fe80095:ad71:3c35:1df9)
Destination: ff02(ff02)
User Datagram Protocol, Src Port: ssdp (1900), Dst Port: ssdp (1900)
Source port: ssdp (1900)
Destination port: ssdp (1900)
Length: 460
Checksum: 0xca12 [validation disabled]
Hypertext Transfer Protocol
```

Popis jednotlivých částí hlavičky IPv6:

- **Version** – udává jaká je použita verze IP.
- **Traffic class** – udává hodnotu, která rozhoduje o tom zda se upřednostňuje

rychlost nebo spolehlivost doručení.

- **Payload length** – udává velikost datové oblasti.
- **Next header** – říká zda je připojena další hlavička. Může udávat doplňující udaje, které nejsou uvedeny v klasické hlavičce jaku je například upřesnění cesty nebo fragmentace.
- **Hop limit(TTL), Source addres, Destination addres** – jsou stejné údaje jako u IPv4 jen u adres je použitá délka 128- bitů.

Protokol ICMP

Jedná se o služební protokol, který je součástí IP-protokolu. Má na starosti signalizaci stavů v síti. Zprávy jsou posílány pomocí IP datagramů.
ICMP může nést zprávy typu :

- **Echo** – jedná se o jednoduchou kontrolu dosažitelnosti cílového bodu. Na cílový stroj je zaslán požadavek echo a cílový stroj nám na něj odpoví. [4]
- **Nedoručitelný IP-datagram** – odešle se pokud není možné datagram doručit. [4]
- **Sniž rychlost odesílání** – v síti může nastat situace, že některý z aktivních prvků není schopen odeslat všechny přijímané datagramy. V tom případě aktivní prvek požádá zdrojový stroj, aby snížil rychlost zasílání datagramu a tím snížil zátěž sítě.[4]
- **Vypršelo TTL** – každý datagram má při vytvoření nastavenou hodnotu TTL. Různé OS mají přednastavené jiné doby TTL, například Linux má přednastaveno TTL 64, MS Windows má 128. Při průchodu aktivním prvkem se TTL sníží o 1. Po vypršení TTL to znamená, že pokud je hodnota TTL 0, zašle se zpráva na zdrojový stroj, že datagram nebylo možné doručit. [4]

Protokoly ARP a RARP

Tyto protokoly můžeme využít například při zjišťování MAC adres počítačů ve stejné síti. Pokud chceme komunikovat v lokální síti, potřebujeme znát obě IP adresy a obě MAC adresy. Je to dáno tím, že pro přenos IP-datagramu potřebujeme 4 bajtovou adresu, ale IP datagram je dále zabalen do linkového rámce. Zde je potřeba 6 bajtová adresa. Zdrojový stroj zná vlastní IP a MAC adresu a IP adresu cílového stroje. Pro zjištění MAC vyšle všesměrový ARP požadavek, kde je uvedena IP adresa cíle a

žádost o zdělení MAC. Pokud tokový požadavek přijme stanice, která tento údaj zná, zašle odpověď žadateli. [4]

Protokol IGMP

Služební protokol IGMP se stará o vysílání dat pro více uživatelů . Stará se správu skupin, které mají zájem o vysílání. Každý člen, který je ve skupině, musí po obdržení IGMP zprávy potvrdit členství ve skupině. Využití má například při vysílání stream televize.

2.1.3 Transportní vrstva

Tato vrstva má umožňuje měnit chování sítě pro potřebu použitých aplikaci a odstranění chyb, které se mohou při přenosu vyskytnout. Mezi hlavní protokoly přenosu patří TCP a UDP

TCP

0b	16b 31b
Source port	Destination port
Sequence number	
ACK number	
Hlen No used Code bits	Window
Checksum	Urgent pointer

Obr. 2.3: Hlavička TCP

Jedná se o spojově orientovaný protokol. To znamená, že před vlastním přenosem segmentů se najde vhodná přenosová cesta, a data jsou dále touto cestou distribuována. Segmenty jsou číslovány a kontrolovány. Pokud dojde k poškození nebo k nedoručení segmentu s očekávaným pořadovým číslem, odešle se chybová zpráva a segment jsou znovu odeslána. Využívá se například u služeb SSH,SNMP,HTTP.

Výhody TCP

- zajištění přijetí segmentů s možností detekcí a opravy chyb.
- zajištění správného pořadí segmentů.
- snadná detekce nedoručených segmentů.

Nevýhody TCP

- hlavička obsahuje více informací a tím se zvyšuje objem přenesených dat.
- je nutné potvrzovat přijetí segmentu a tím se zvyšuje zátěž na síti.

UDP

0b	16b 31b
Source port	Destination port
Length	Checksum

Obr. 2.4: Hlavička UDP

Jedná se o spojově neorientovaný protokol, který se využívá, pokud nepotřebujeme kontrolovat doručení dat. Datagramy jsou vyslány do sítě, ale dále už se nekontroluje, jakým způsobem budou data doručena. Klasický příklad využití u tohoto protokolu je při použití real-time aplikací, kde kontrolování správného pořadí dat a jejich potvrzování by vedlo k většímu zpoždění.

Výhody UDP

- malá hlavička, která obsahuje pouze data o zdrojovém a cílovém portu, délku a kontrolní součet.
- malá zátěž sítě(není potřeba potvrzování přijetí).

Nevýhody UDP

- datagramy se mohou při přenosu ztratit nebo přijít ve špatném pořadí a není možnost jak to detekovat.

2.1.4 Aplikační vrstva

Tato vrstva slouží přímo k přenosu dat služeb, které využívá uživatel. Využívají jednu nebo obě služby transportní vrstvy. Při komunikaci se využívají porty, aby bylo možné určit, jaký aplikační protokol mé s daty dále pracovat.

- **FTP** – tento protokol nám umožňuje přenášet data v síti. Využívá se protokolu TCP. Protokol FTP komunikuje v systémem Klient - Server. To znamená, že Klient si zažádá o přenos dat a Server mu po ověření údajů (uživatelské jméno a heslo) umožní přístup k datům. Komunikace probíhá na dvou portech a to TCP/21, kde server naslouchá a pomocí kterého probíhá řízení přenosu a portu TCP/20, který slouží pro vlastní přenos dat. Tento protokol není šifrován, proto je velice snadné zachytit přihlašovací údaje.

- **SFTP** – jedná se o protokol FTP, který je tunelován přes protokol SSH, který je šifrován. Tím je zajištěna lepší bezpečnost přenosu.

- **SSH**– jedná se současně o program a protokol pro přístup ke vzdálenému terminálu. Přenos dat pomocí ssh je šifrován a tím je zajištěna bezpečnost přenosu dat sítí. Tento protokol nahrazuje některé starší protokoly jako jsou Telnet ,Rlogin a Rshs. Komunikace probíhá na portu TCP/22 [4]

- **HTTP** – tento protokol slouží pro přenos hypertextových dokumentů. Ke svému provozu využívá port TCP/80. Tento protokol patří mezi nejvíce využívané, protože nám zprostředkovává přístup k informacím. Komunikace probíhá pomocí módu Klient - Server , kdy klient požádá o data a server je poskytne. Tyto data nejsou šifrována.

- **HTTPS** – jedná se o zabezpečený protokol HTTP, kde jsou data šifrována algoritmem SSL nebo TSL.

- **POP3** – protokol se využívá pro stahování emailových zpráv ze serveru. Nevýhodou tohoto protokolu je, že klient musí stáhnout všechny zprávy a teprve poté má možnost je editovat.

- **IMAP** – slouží pro správu emailů podobně jako POP3, ale máme zde možnost zprávy na serveru upravovat.

- **SMPT** – jedná se o protokol, který má na starost přenos zpráv v sítí. Ke spojení se využívá port TCP/25 a data jsou stahována přímo mezi serverem a klientem. Pro přístup ke zprávám se využívá protokolu POP3 nebo IMAP.

- **TELNET** – jedná se o protokol pro přístup ke vzdálenému terminálu podobnému protokolu SSH. Protokol TELNET komunikuje na portu TCP/23. Jeho nevýhodou je že nepoužívá šifrování dat.

- **DNS** – protokol nám zajišťuje překlad IP adres lépe srozumitelných. Například, pokud by jsme chtěli stáhnout http data ze vzdáleného serveru, potřebovali bychom znát jeho IP adresu, která může být 89.77.68.53 . Pokud bude na jmenném serveru uložen záznam, že tato adresa patří doméně www.jaknaserver.cz. Stačí zadat pouze tento název, který nám pomocí DNS vrátí hledanou IP adresu.

- **DHCP** – je protokol, který nám umožňuje přidělovat automaticky údaje jako jsou IP adresa, maska sítě, výchozí brána a DNS servery. Je tím usnadněna práce, protože není potřeba tyto údaje nastavovat na každém zařízení jednotlivě. Klient komunikuje na portu UDP/68 a server naslouchá na UDP/67 .

Po připojení do sítě vyšle broadcast packet DHCPDISCOVER. Na to odpoví server packetem DHCPOFFER . Klient má možnost výběru z několika IP. Následuje žádost DHCPREQUEST, kde požádá server o přidělení vybrané IP a server dopoví packetem DHCPACK. Po obdržení této zprávy může klient vybranou IP používat.

2.2 Firewall

V dnešní době není bezpečné se pohybovat na síti se systémem, který nemá alespoň základní bezpečností nastavení. Můžeme vyžívat například nějaký externí prvek, který nám pomůže ochránit počítače před vnějším útokem. Takový prvek je například HW firewall, který má jen jednu hlavní funkci a to ochránit vnitřní sít před útokem.

Pro lepší zabezpečení je vhodnější nespoléhat se pouze na jednu vrstvu zabezpečení, ale mít alespoň základní zabezpečení na každém prvku. Tím dosáhneme toho, že pokud se podaří útočníkovi překonat jednu překážku, čeká na něj hned další a nemá okamžitý přístup do systému. Takové zabezpečení můžeme nalézt například u domácí sítě připojené pomocí routeru, na kterém je instalován firewall. Pokud se útočníkovi povede překovat tuto ochranu, stále se ještě musí dostat přes lokální firewall, který bude na námi využívané stanici.

Můžeme se setkat s několika druhy firewallů a to :

- **Paketový filtr** – jedná se o nejjednodušší formu firewallu a také nejstarší. V nastavení firewallu se definují pravidla jaké přenosy jsou povoleny a jaké přenosy jsou zakázány. To znamená, že do nastavení zadáme přímo IP adresu na port, na který je přenos povolen. Pokud příchozí nebo odchozí spojení vyhovuje těmto kritériím, spojení se povolí. Výhoda paketového filtrování spočívá v jeho rychlosti. Najdou se zde ale také nevýhody. Filtr musí kontrolovat každý přenesený paket a to může způsobovat některým službám problémy. Další nevýhoda je, že pakety nejsou důkladněji kontrolovány. V Linuxu je dostupný paketový filtr ipchair [6]

- **Stavové filtry** – jedná se o podobný princip jako u paketového filtrování, ale je přidána možnost pamatovat si údaje o probíhajících spojeních. To znamená, že firewall má předem stanovená pravidla, jaký přenos povolit a jaký odmítnout s tou výhodou, že nemusí kontrolovat jednou prověřený provoz. V praxi to znamená, že firewall prověří přenos dat ze vzdáleného serveru. Pokud přenos

vyhoví pravidlům nastaveným na firewallu, je přidán mezi probíhající přenosy a pakety nejsou kontrolovány. Výhoda tohoto principu je, že není potřeba kontrolovat celé spojení, ale jen jeho část. U Linuxu se můžeme s tímto tipem filtrování setkat u iptables. [6]

- **Aplikační brány** – jedná se o odlišný systém, než jsou Paketové a Stavové filtry. Spojení probíhá přes prostředníka, který obstarává funkci firewallu (Proxy firewall). V praxi to znamená, že data jsou kontrolována na Aplikační vrstvě Proxy firewallu a následně předána dál. Při této metodě nezná cílový prvek IP adresu. Zná pouze adresu Proxy serveru, přes který je komunikace řízena. Nevýhodou Aplikační brány je, že nedosahuje rychlostí kontroly jaké jsou na Paketovém a Stavovém filtru. Tato nevýhoda je ale kompenzována lepší kontrolou přenášených dat. [6]

2.2.1 Netfilter - Iptables

Projekt Netfilter nám umožňuje natavit pravidla pro filtrování. Existuje několik metod, jak lze firewall nastavit. Pro Linuxová jádra 2.2 a starší se využívá nástroj ipfwadm. Od jader 2.2.x byl přidán paketový filtrovací nástroj ipchains. Pro novější jádra 2.3.15 byl implementován nástroj iptables, který je zpětně kompatibilní s předchozími verzemi. Iptables nám umožňuje dle nastavení používat Paketové ale i Stavové filtrování. Stále se testuje prvek nazvaný netfilter-layer7, který nám umožňuje filtrovat provoz přímo na Aplikační vrstvě a tím máme lepší možnost se rozhodnout, jaký provoz chceme zakázat nebo povolit.

V iptables jsou tři hlavní směry, kudy může komunikace procházet a to jsou :

- **INPUT** – zde se rozhoduje o veškerém provozu, který je směrován na lokální stanici z vnější sítě. U předchozích verzí byla možnost INPUT brána jako všechen provoz přicházející na danou stanici. Vznikal tím problém, že základní politiku pro příjem(ACCEPT a DROP) nebylo možné rozlišit pro data určená pro lokální stanici a data, která byla pouze průchozí.
- **OUTPUT** – zde se definuje politika pro odchozí data z lokální stanice.
- **FORWARD** – jedná se o data, která nejsou určena pro lokální stanici, ale pouze přes ní procházejí. S tímto provozem se můžeme setkat pokud poskytujme sítové služby dalším stanicím.

Pravidla pro rozhodování zda se má provoz povolit jsou v řetězci. To znamená, že nejprve se zkontroluje jaká politika je pro daný směr nastavena. Můžeme se setkat

se dvěma typy a to :

- **ACCEPT** – kdy se postupně procházejí pravidla a pokud data neodpovídají zadaným pravidlům, provoz je povolen.
- **DROP** – i zde se kontrolují postupně jednotlivá pravidla zda podmínce data vyhoví nebo ne. Pokud se dojde na konec řetězu, data budou zahozena.

Nyní si ukážeme, jak má vypadat definice jednotlivých pravidel a co znamenají jednotlivé části kódu.
Syntaxe příkazu iptables vypadá takto :

iptables *příkaz pravidlo rozšíření* [5]

Příkazy

V táto části syntaxe nastavujeme jaká akce se má po zadání provést. Máme následující možnosti :

- `-A --append třída specifikace` – přidá na konec řetězce zvolené třídy nové pravidlo. [8]
- `-D --delete třída specifikace` – smaže pravidla ve zvolené třídě, která odpovídají specifikaci. [8]
- `-I --insert` *`třída`* – přidá nové pravidlo na začátek řetězce. [8]
- `-R --replace` *`třída specifikace`* – nahradí pravidlo, které ve zvolené třídě odpovídá specifikaci. [8]
- `-L --list` *`třída`* – vypíše pravidla zvolené třídy. Pokud není třída zvolena, vypíší se všechna pravidla. [8]
- `-F --flush` *`třída`* – smaže pravidla ve zvolené třídě. Pokud není zvolena, smažou se všechny pravidla. [8]
- `-N --new-chain` *`třída`* – založí novou uživatelskou třídu. [8]
- `-D --delete-chain` *`třída`* – smaže uživatelem založenou třídu. [8]
- `-E --rename-chain` *`starý-název nový-název`* – přejmenuje uživatelem definovanou třídu. [8]
- `-P --policy` *`třída specifikace`* – nastaví u vybrané třídy politiku. [8]

Pravidla

Pravidla nám určují podmínky, jaké druhy provozu se mají aktuálním pravidlem kontrolovat. U následujících pravidel je možnost přidat další část kódu a to [!] , což znamená, že dané pravidlo platí pro všechen provoz kromě pravidla uvedeného v aktuálním příkazu.

- `-p --protocol` ***protokol*** – udává protokol, pro které se pravidlo vztahuje. Protokoly mohou být například TCP, UDP, ICMP. [8]
- `-s --source` ***adresa[maska]*** – sděluje jaká zdrojová adresa vyhovuje aktuálnímu pravidlu. [8]
- `-d --destination` ***adresa[maska]*** – smaže uživatelem založenou třídu. [8]
- `-j --jump` ***cíl*** – definuje akci, která se provede, pokud data splní podmínku. [8]
- `-g --goto` ***třída*** – pokud data splní podmínku, mají se předat do jiné třídy. [8]
- `-i --in-interface` ***rozhraní*** – definuje, z jakého rozhraní mají data přijít, aby splňovala podmínku. [8]
- `-o --out-interface` ***rozhraní*** – definuje, jaké je výstupní rozhraní, aby byla podmínka splněna. [8]
- `-f --fragment` – udává pravidlo, které platí pro vše, kromě prvního fragmentu fragmentovaných dat. [8]
- `-c --set-counter` – využívá se pro statistické účely. [8]

Rozšíření

Pokud potřebujeme specifikovat, jaký provoz chceme filtrovat, můžeme použít doplňující moduly. Mezi rozšířené možnosti patří :

- `--src-type` ***typ*** – upřesňuje, jaký typ příchozí adresy je použit. [8]
- `--dst-type` ***typ*** – upřesňuje, jaký typ adresy je použit pro odchozí spojení. [8]
- `--connlimit-above` ***hodnota*** – udává, kolik je povolených současných spojení. [8]

- `--source-port` *`port`* – sděluje nám, jaký musí být příchozí port, aby splnil podmínku. [8]
- `--destination-port` *`port`* – udává, jaký musí být odchozí port na splnění podmínky. [8]
- `--icmp-type` *`typ`* – udává typ ICMP zprávy, aby splnil podmínku. [8]
- `--mac-source` *`adresa`* – sděluje jakou očekává MAC adresu pro splnění podmínky. [8]

2.2.2 Netfilter-layer7

Jedná o filtr, kde je možné rozhodovat na základě dat získaných přímo z aplikační vrstvy. Oproti klasickému filtrování, kdy můžeme nastavit hodnoty jako jsou zdrojová adresa nebo port, na kterém má být provoz kontrolován. Dovede Layer7 přímo zakázat například službu Bittorent a není důležité, na kterém portu poběží. Vývoj Layer7 začal v roce 2003 a stále se ještě vyvíjí. [7]

Výhody Layer7:

- Snadné filtrování protokolů, které využívají nepředvídatelné porty.
- snadná implementace do Iptables.

Nevýhody Layer7

- Je potřeba, aby Layer7 byl jako modul jádra. Pro méně zkušené uživatele to může představovat problém.
- Nelze zaručit, že všechny protokoly bude možné filtrovat. Seznam dostupných protokolů s jejich hodnocením kvality je dostupný na `http://l7-filter.sourceforge.net/protocols`

2.2.3 Analýza zabezpečení

Aby bylo možné ověřit zabezpečení systému je potřeba provést testy, které nám odhalí možná rizika. Pro testování můžeme využít některý z online programů, které jsou dostupné na internetu. Další možnost využít lokální programy, mezi které patří například program nmap.

Nmap

Jedná se o aplikaci pro skenování sítě. S její pomocí máme možnost zjistit jaké jsou v lokální sítí připojené počítače, nebo jaké mají otevřené porty. Můžeme ale také testovat jednotlivá nastavení firewallu a zjistit jaký provoz nám firewall propustí a který odmítne.

Existuje několik druhů skenování které můžeme s nmap použít. V případě že použijeme parametr -sS (TCP SYN) zašle se na daný stroj požadavek na otevření spojení. Pokud nám zpráva RST znamená to že na daném portu neběží žádná služba, nebo nemáme k dané službě přístup. V opačném případě pokud přijde potvrzení ACK tak lze předpokládat že na daném portu běží některá ze služeb. Mezi další možnosti patří -sU (UDP scan) a -sO(Protocol scan).Princip detekce je podobná na port se pošle požadavek na spojení (v případě scanu portu se zasílá packet s číslem protokolu), a očekává zda nám přijde ICMP zpráva protocol unreachable , která nás informuje o nedostupnosti.[6]

Použití nmap je následující :

```
ml@ml:~$ sudo nmap -O localhost
Starting Nmap 5.21 ( http://nmap.org ) at 2010-12-15 21:32 CET
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000099s latency).
Hostname localhost resolves to 2 IPs. Only scanned 127.0.0.1
rDNS record for 127.0.0.1: localhost.localdomain
Not shown: 997 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
25/tcp  open  smtp
631/tcp open  ipp
Device type: general purpose
Running: Linux 2.6.X
OS details: Linux 2.6.31 - 2.6.32
Network Distance: 0 hops
```

Z ukázky je patrné, že jde velice snadno zjisti jaký druh systému používáme a které služby jsou volně přístupné. Dále by bylo možné pokračovat v podrobnější skenování a zjistit přesně jaké služby a verze jsou použity a zda jsou zabezpečeny firewallem.

3 LOKÁLNÍ BEZPEČNOST

Až doposud jsme se zajímali o možnosti, jak ztížit, případně zabránit útočníkům v přístupu do systému. Nyní si řekneme něco o tom, jak funguje zabezpečení, které je v samotném systému a jaké jsou další možnosti toto zabezpečení dále zdokonalit s použitím dalších programů.

Operační systém Linux není dokonalý. Pokud ho porovnáme s dnes nejvíce rozšířeným operačním systémem Windows, vyjde nám Linux jako bezpečnější, ale při špatném nastavení zabezpečení snadno napadnutelný. Zdání bezpečnosti je způsobeno tím, že systém Windows je dnes nejvíce rozšířený systém a útočníkům se vyplatí zaměřit se na jeho méně zkušené uživatele, než na ty zkušenější, kteří se většinou systémy s Linuxem zabývají. Další výhoda Linuxu je v právech jednotlivých uživatelů. Aby bylo možné na Linuxu provést zásadnější změny, musí útočník získat práva superuživatele. A to není snadná záležitost. Dá se toho docílit různými nástrahami, které můžeme na superuživatele přichystat tzv. Trojské koně, nebo se pokusit prolomit heslo k jeho účtu.

3.1 Uživatelé a jejich práva

V systému Linux existují dva základní druhy uživatelů a to superuživatel, který má právo číst, zapisovat a spouštět všechny soubory v systému. Druhou skupinou jsou uživatelé, u kterých jsou práva omezena. Záleží na superuživateli, jaké možnosti jednotlivým uživatelům přidělí.

Uživatelé mohou patřit do jedné nebo několika skupin a podle toho mají přístup k datům jiného uživatele.

3.1.1 Přístupová práva k souborům

V Linux se dají práva pro práci rozdělit podle toho, kdo s nimi pracuje a jaké k nim má oprávnění.

Oprávnění lze rozdělit na tři druhy a to :

- **Právo čtení** – umožňuje každému uživateli číst daný soubor. Pokud ale nemá nastavená ostatní práva, nemůže soubor nijak dále využívat.
- **Právo zápisu** – umožňuje zapsat data do souboru.
- **Právo spuštění** – pokud není přiděleno toto právo, nelze soubor spustit. Může se vyskytnout situace, že je přiděleno pouze právo a ostatní práva nejsou

udělena. Tím dojde k situaci, kdy uživatel může program používat, ale nemůže jej editovat.

K samotnému souboru má přístup vlastník, dále skupina, ve které vlastník je a ostatní uživatelé. Pro každou z těchto částí lze nastavit možnosti práce se souborem.

Zápis práv

Práva k souborům můžeme zapsat :

```
chmod 751 seznam.txt
```

Příkaz `chmod` nám umožňuje v Linuxu nastavit práva k souborům. Seznam číslic udává jaká práva souboru udělíme a poslední je jméno souboru, u kterého chceme změnit práva.

Práva můžeme zapsat postupně jednotlivými zkratkami `r` pro čtení, `w` pro zápis a `x` pro spuštění. Toto je nutné definovat pro všechny tři části (vlastník, skupina, ostatní).

Druhá možnost využít číselného zápisu kdy `r` odpovídá číslu 4, `w` odpovídá 2 a `x` odpovídá 1. Výsledná práva se určují sečtením jednotlivých hodnot. Například pro právo čtení a spuštění dostaneme potřebnou hodnotu sečtením 4(právo čtení) + 1(právo spuštění) = 5 . Z uvedeného příkladu odpovídají práva 751 tomu, že vlastník má všechna práva, skupina má právo čtení a spuštění. Ostatní uživatelé mají právo jen soubor spustit.

3.2 Bezpečná hesla

Základním prvkem k dobře zabezpečenému systému je vhodně zvolené heslo jak u superuživatele, tak i u jednotlivých uživatelů.

Jak by nemělo vypadat heslo ?

- nemělo by obsahovat pouze číselné znaky od 0-9 : 278164
- nemělo by obsahovat jména nebo názvy míst, datum narození : pepa, heslo, 12.11.1967
- neměly by to být za sebou jdoucí kombinace kláves na klávesnici :qwertz, asdfgh
- heslo by nemělo obsahovat méně než 9 znaků : R2toDf, foetS3

- nepoužívat pro tvorbu jakékoli osobní údaje, které by mohl někdo zjistit : Alan (jméno psa)

Jak by mělo vypadat bezpečné heslo ?

- obsahuje alespoň 12 znaků : ijacuengoens
- obsahuje alespoň jedno velké písmeno a jednu číslici : Admrudbo56f

3.2.1 Stínová hesla

U některých distribucí Linuxu jsou hesla stále ještě uložena v souboru `/etc/passwd` ve formě hash kódu. Kontrola hesla probíhá pomocí nevratné matematická operace. Heslo ve stejném formátu je už uloženo v souboru passwd. Pokud se výsledky rovnají, znamená to, že je uživatel ověřen. Nevýhoda uložení hesel přímo v souboru passwd spočívá v tom, že tento soubor je čitelný pro všechny uživatele systému. Útočník má tedy usnadněnou práci k získání hesel. Stačí mu pouze provádět stejnou matematickou operaci na vlastní vygenerovaná hesla. Pokud je nalezena shoda, znamená to, že heslo některého uživatele bylo odhaleno.

Pro zabezpečení se využívají stínová hesla. To znamená, že hesla jsou uložena v souboru `/etc/shadow` a v souboru `/etc/passwd` jsou nahrazeny hash kódy pouze odkazem. Výhoda použití stínových hesel spočívá v tom, že práva pro práci se souborem `/etc/shadow` má pouze superuživatel. Autorizace pak vždy proběhne s právem superuživatele a tím zabrání, aby bylo možné číst tento soubor nepovolanou osobou.

Případnému útočníkovi se tím ztíží práce, protože mu už nestačí pouze zjistit heslo běžného uživatele a pak zkoušet odhalit heslo ze souboru `/etc/passwd` . Potřebuje heslo pro superuživatele, aby mohl přečíst soubor `/etc/shadow` a dostal se i ostatním heslům.

3.3 Apparmor

Programování není jednoduché a lehko můžeme udělat několik neúmyslných chyb, zvlášť pokud kód programu má několik tisíc řádků. Takovéto chyby nám ale mohou nechat zranitelný systém tím, že útočník chybu využije pro získání práv.

Řešit tento problém jde jednak pravidelným aktualizováním programů, které používáme. Tím se zajistí odstranění chyb, které vývojáři odhalili. Další řešení je využít některý z projektů, který se zaměřuje na nastavení a omezení práv jednotlivých programů. Chyby, které jsou stále v programu tím sice neodstraníme, ale pokud bude vše správně nastaveno, ztížíme útočníkům průnik do systému. Můžeme

si vzít příklad z aplikace Firefox. Pokud nebude systém zabezpečen, může útočník využít chyby v kódu Firefoxu a dostat se do systému. Pokud ale nastavíme hranice co program Firefox má povoleno udělat a co nesmí, ztížíme útočníkům práci. Úročník sice využije stejnou chybu v kódu jako když byl systém nezabezpečen. Tím ale podobnost končí. Útočník se nemůže dostat za vytvořené hranice a má tím omezen rozsah působnosti.

3.4 Logování událostí

V operačních systémech běží neustále mnoho programů. Pro přehlednost a kontrolu co který program dělá je možnost nastavení logování událostí, kdy systém zaznamená akci, kterou program provedl do textového souboru. Tyto soubory můžeme dále upravovat a získat z nich důležité informace co se v systému děje. V distribucích založených na Debianu se záznamy ukládají do souborů v `/var/log` . Možnost nastavení, které zprávy z aplikaci se mají zaznamenat lze upravit v souborech `/etc/rsyslog.d/`
Mezi nejdůležitější log soubory v systému patří:

- **messages** – zde se zaznamenávají události dle nastavení `/etc/rsyslog.d/` . Jsou zde uvedeny informace z PAM kdy se jaký klient autorizoval, zaznamenávají se zde i přihlášení uživatelů.
- **sendmail** – zpřístupňuje informace o vzdálených přístupech na program sendmail.
- **auth** – jedná se o souhrn informací, kde byla vyžadována autorizace. Nalezneme zde jednotlivé požadavky PAM na autorizaci ale i neúspěšné požadavky na přístup k některé ze služebámí. Je zde i zaznamenáno jaké akce byly provedeny s právem superuživatele.
- **syslog** – zde se na rozdíl od messages zaznamenávají pouze chybové zprávy.

Souborů, do kterých se ukládají informace je mnoho a můžeme z nich zjistit mnoho informací kdo zkouší prolomit zabezpečení firewallu, nebo jaké akce nebyly povoleny aplikacím, které byly omezeny programem Apparmor.

```
Dec 12 21:15:29 ml sshd[651]: pam_unix(sshd:auth): authentication failure;
logname= uid=0 euid=0 tty=ssh ruser= rhost=81.222.134.91  user=root

Dec 12 21:15:32 ml sshd[651]: Failed password for root from
81.222.134.91 port 34007 ssh2
```

```
Dec 12 23:16:45 ml sudo:       ml : TTY=pts/4 ; PWD=/etc ; USER=root ;
COMMAND=/usr/bin/vim /etc/hosts.allow
```

V ukázce je možné vidět tři vybrané zprávy z logu `/var/log.auth.log` . První záznam nás upozorňuje, že se někdo z udané ip adresy pokouší přihlásit na uživatele root. Následující záznam říká že bylo zadáno špatné heslo a spojení nebylo povoleno. Poslední záznam udává, že se uživatel root otevřel soubor `/etc/hosts.allow`

3.4.1 Programy pro kontrolu systému

Pro kontrolu stavu systému nám samotné záznamy v log souborech nestačí a proto je potřeba vhodným programem kontrolovat a říct systému jak na vzniklé situace reagovat.

TCP Wrappers

Tento program nám umožňuje definovat jaké spojení TCP chceme povolit a jaké naopak zakázat. Pro konfiguraci se používají soubory `/etc/host.allow` , kde nastavujeme , které aplikace z jaké ip adresy mají do našeho systému přístup. Druhý soubor `/etc/host.deny`, slouží naopak k určení nepovolených spojení.

Ukázka souboru nastavení TCP Wrappers :

```
$ cat /etc/hosts.allow
sshd: 192.168.0.0/255.255.255.0
sendmail: ALL
$ cat /etc/hosts.deny
ALL: ALL
```

Z ukázky je patrné nastavení nastavení polity zakaž vše co není povoleno. Neomezené spojení je na aplikaci sendmail. Na ssh se můžeme připojit pouze z vnitřní sítě. Pokud se při ukázkovém nastavení pokusíme připojit na ssh, přidá se do log souborů zpráva o zamítnutí spojení.

```
sshd[8424]: refused connect from 81.222.134.91 (81.222.134.91)
```

4 NÁVRH ZABEZPEČENÍ SÍŤOVÉHO PROVOZU

4.1 Současné možnosti

V současné době máme možnosti si vybrat z mnoha řešení, které se zabývají jednorázovým generováním pravidel pro Netfilter. Tato řešení jsou výhodná v jednorázovém nastavení pravidel. Při změně, například doinstalování nového programu, je potřeba v generátoru upravit pravidla a následně jednorázově vygenerovat nová pravidla pro Netfilter.

Druhá skupina firewallů nám umožňuje snadnější editaci pravidel spolu s dalšími funkcemi. Mezi nejznámější na platformě Debian patří například Ufw [10] nebo Firestarter [10]. U většiny těchto programů je dostupné grafické rozhraní, které nám usnadňuje snadno provádět jednotlivá nastavení. Oproti klasickým generátorům pravidel Netfilter zde máme možnost jednoduše přidat nová pravidla, případně zjistit, jaký provoz je zahazován. Některé z těchto firewallů nám umožňují pracovat i IPv6. U firewallu Ufw, který je instalován v základní verzi operačního systému Ubuntu, je možné pracovat s IPv6. V grafické nástavbě GUfw, kterou použije většina méně zkušených uživatelů, je možnost pracovat pouze s IPv4. V současné době, kdy je IPv6 stále více nasazována, si tím můžeme neúmyslně nechat nezabezpečený systém.

Poslední skupina téměř se nevyskytující pro operační systémy založené na Debian, jsou firewally s možností reakce na nové stavy v sítovém provozu. Tyto firewally jsou běžné i v operačních systémech Microsoft Windows. Pokud se nový program pokusí o síťovou komunikaci, je informován uživatel, který muže následně sítový provoz pro tuto aplikaci povolit nebo případně zakázat.

4.1.1 Využití Apparmor

Apparmor poskytuje možnost omezit jednotlivé programy a definovat pro ně pravidla. Můžeme tak kontrolovat i to, zda mají možnost využívat síťovou komunikaci. Mezi hlavní výhody patří možnost velice přesně daný program kontrolovat. Naopak mezi slabiny při využiti Apparmor jako součást firewallu, je nutnost definovat vlastní pravidlo pro každý program. Není možné využít obecné pravidlo pro více programů.

4.1.2 Využití Selinux

Selinux začali vyvíjet NSA (National Security Agency) a Secure Computing System na počátku devadesátých let. Linux využívá diskrétní řízení přístupu (DAC, Discretionary Access Control), kdy má každý souboru jeden ACL (Access Control List)

[11], kde jsou definována přístupová pravidla pro vlastníka, skupinu a ostatní.

Selinux používá povinné řízení přístupu (MAC, Mandatory Access Control). Programy jsou spuštěné v sandboxech definující prostor, ve kterém mohou programy pracovat. Je možné velice přesně definovat možnosti jednotlivých programů a to i v případě, že běží pod právy superuživatele. Pro tvorbu firewallu je toto téměř ideální metoda omezení přístupu k sítovému rozhraní. Je možnost pracovat v Selinuxu i se sockety, což je základní prvek síťové komunikace v linuxu.

Pro využití Selinuxu je potřeba, aby byl přímo zakomponován v jádře. Některé distribuce, například Fedora, mají již Selinux v jádře. Současné distribuce vycházející z Debian, Selinux v jádře nemají a je nutno po každém vydání nového jádra je znovu upravovat.

4.1.3 Využití NFqueue

NFqueue modul pro Netfilter nám umožňuje přidat další metodu, jak filtrovat data. V základu Netfiter máme tři základní možnosti, jak s pakety naložit a to metody ACCEPT pro povolení, DROP pro zahození bez odpovědi a REJECT na zahození dat a informování protistrany.

NFqueue nám umožňuje čtvrtou možnost a to zpracovat data mimo Netfilter. Využívají se fronty, do kterých jsou pakety ukládány. Je možné vytvořit více těchto front a následně s nimi odděleně pracovat. Není vhodné takto filtrovat všechen povoz, vznikalo by tím zbytečně dlouhé zpoždění síťové komunikace a vzrostla by zátěž na daném stroji. Výhodné je toto řešení využít u stavového firewallu. V Netfilter stačí zachytit nově navazovanou komunikaci, tedy tu v případě TCP obsahující příznak SYN a zaslat jí do příslušné fronty nfqueue. Následně v externím programu rozhodnout o dané komunikaci.

Nevýhoda je mírné zpoždění při navázání spojení. V případě jeho povolení se již jedná o spojení navázané a Netfilter s ním tak již dále pracuje. Pokud se zaměříme na výhody tohoto řešení, zjistíme, že není potřeba vytvářet pro všechny programy profily jako v případě Apparmor nebo složitěji upravovat jádro a následně odchytávat sockety jako v případě Selinux. Stačí pouze doplnit Netfilter o modul NFqueue a v externí aplikaci kontrolovat nová spojení.

4.2 Programové řešení firewallu

Vytvořené programové řešení je zaměřeno pro nové, méně zkušené uživatele. Je kladen důraz na snadné nastavení firewallu. V grafického rozhraní je možné se uživatele dotazovat na nově navazovaná spojení. Je tvořeno pro osobní počítače, kde ve většině případech má uživatel právo dočasně převzít práva superuživatele. U serverových řešení je předpoklad zkušených administrátorů, kde se pravidla Netfilter nastaví pouze při počáteční konfiguraci, případně při výraznější změně služeb.

4.2.1 Python

Python je objektově orientovaný programovací jazyk, někdy mylně považován za vhodný pouze pro výuku programování. Samotný python je napsán v programovacím jazyku C a tím je u Pyhonu zaručena rychlost běhu programu.

Existuje několik verzí Pythonu. U programu byla zvolena verze Python 2.6 . V současné době je vyvíjena i nová verze Python 3.0, která není zpětně kompatibilní se staršími verzemi. Tato nová verze nebyla zvolena z důvodu prozatím menšího používání a zatím ne zcela stabilních a otestovaných prvků potřebných pro tuto práci.

Python oproti jiným programovacím jazykům klade důraz na jednoduchost a přehlednost, avšak ne na úkor funkčnosti. Klasický příklad nalezneme při vyhledávání v poli. Není potřeba sepsat celý cyklus na kontrolování postupně všech prvků, ale stačí použít jednoduchou podmínku, zda je prvek v poli obsažen.

4.2.2 Pygtk

Pří výběru vhodného prostředku pro tvorbu grafické části programu bylo přihlédnuto ke skutečnosti, že noví uživatelé zvolí základní instalaci Ubuntu využívající grafického prostředí Gnome. Gnome samotné je postaveno na GTK. Z tohoto důvodu byl zvolen prvek pro grafickou část projekt s názvem PyGtk, tedy knihovně umožňující komunikaci mezi Pythonem a GTK. [9]

Při tvorbě grafického rozhraní je možné využít klasického ručního definování jednotlivých prvků, nebo některé z grafických nástaveb jako například Glade. Po testování bylo zvoleno ruční definování jednotlivých grafických prvků. Při použití grafických nástaveb docházelo k chybám už při zobrazení prázdného základního okna. Chyba byla nejspíše způsobena špatnou kompatibilitou Pythonu s grafickou nástavbou.

4.2.3 Python Nfqueue

Z modulu Nfqueue v Netfilter je možné zachytit část provozu a následně s ní dále pracovat. Do požadované fronty (nfqueue) se nám uloží celý obsah packetu tak, jak ho obdržel firewall. Následně z něho je možné získat důležité informace jako cílovou a zdrojovou ip adresu, porty na jaké provoz je směrován a který protokol je použit. Je možné si prohlédnout i samotná přenášená data, ale ty v tomto případě nejsou důležitá.

S pomocí takto získaných dat je nadále možné zjistit, jaký program požaduje síťovou komunikaci, případně kdo tuto komunikaci očekává. Podle těchto informací máme pak nadále dvě možnosti jak s packetem naložit a to zahodit (Drop), nebo packet navrátit zpět (ACCEPT). V případě povolení provozu se packet navrátí zpět do sítě. Obě komunikující strany zaznamenají jen mírné zpoždění. [8]

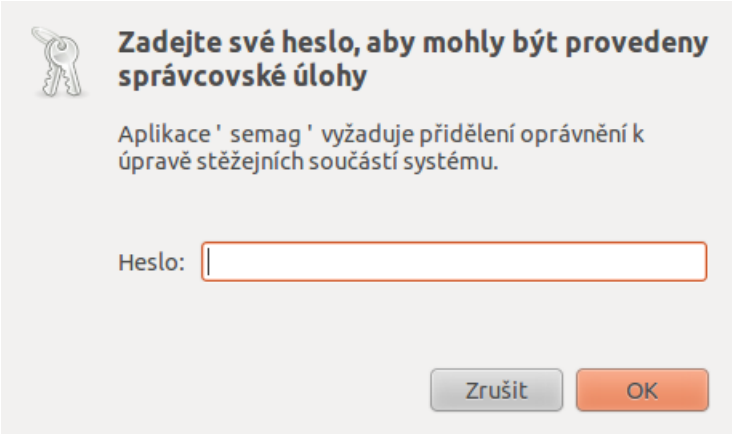
4.3 Návrh firewallu

Jak už bylo řečeno v úvodu kapitoly, program není navrhnut pro serverová řešení ani pro zprávu více uživateli. Následně budou popsány jednotlivé prvky, které zajišťují funkci firewallu. Koncepce je zčásti rozdílná než u klasických konfigurátorů iptables, které řeší firewall jako neměnný celek. Název Programu vznik Semag vznikl spojením anglických slov Security manager.

4.3.1 Semag

Program Semag je možné spustit z menu OS nebo pomocí příkazu semag. V souboru semag jsou definovány základní funkce pro kontrolu spuštění programu a několik parametrů pro rychlé ovládání a možnost automatického spuštění po startu operačního systému.

Hlavní požadavek je kladen na bezpečnost. Je tedy nežádoucí, aby si kterýkoliv uživatel mohl program spustit a pohlížet si pravidla firewallu, případně je upravovat. V Ubuntu je znemožněno přihlásit se přímo jako uživatel root. Je tak učiněno s ohledem na bezpečnost systému. Pro práci pod právy superuživatele se používá příkaz sudo. Při spuštění programu se zkontrolují práva uživatele. Toto zjistíme jednoduchým dotazem na uživatelovo UID (User ID). Pokud dostaneme výsledek 0, znamená to, že uživatel práva v současné době má a je možné pokračovat v běhu programu. Pokud dostaneme jiné číslo, je potřeba tyto práva získat. Práva je možné získat dotázáním v konzoli nebo pomocí příkazu `gksudo`

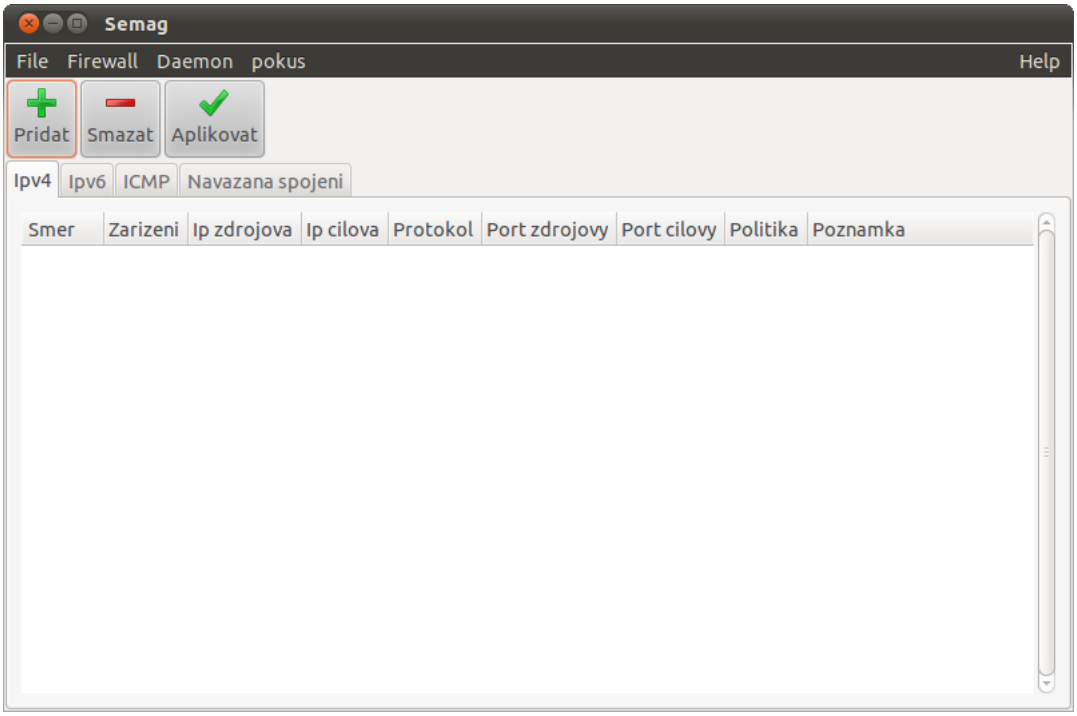


Obr. 4.1: Gksudo a získání práv

Mezi další funkce základního souboru je rozlišení jednotlivých parametrů programu, se kterými může být spuštěn.

- **enable** – využívá se pro zavedení pravidel do Netfilter. Postupně se spustí poslední vygenerovaná pravidla pro `iptables` a `ip6tables`
- **disable** – nastaví pravidla Netfilter do základního nastavení. To znamená, že vymaže všechna pravidla a všechen provoz povolí. Aby se nemusela mazat pravidla postupně, využívají se příkazy `iptables-restore` a `ip6tables-restore` se základním nastavením.
- **start** – parametr pro spuštění daemona, který se stará o vyhodnocení dat z nfqueue. Vytvoří se nový samostatný proces nezávislý na Semag.
- **stop** – provede ukončení daemona. Z tmp souborů zjistí identifikační číslo, které tam bylo při jeho vytvoření uloženo a tento proces ukončí.
- **restart** – jedná se o postupnou kombinaci příkazů stop a start.

4.3.2 Hlavní okno

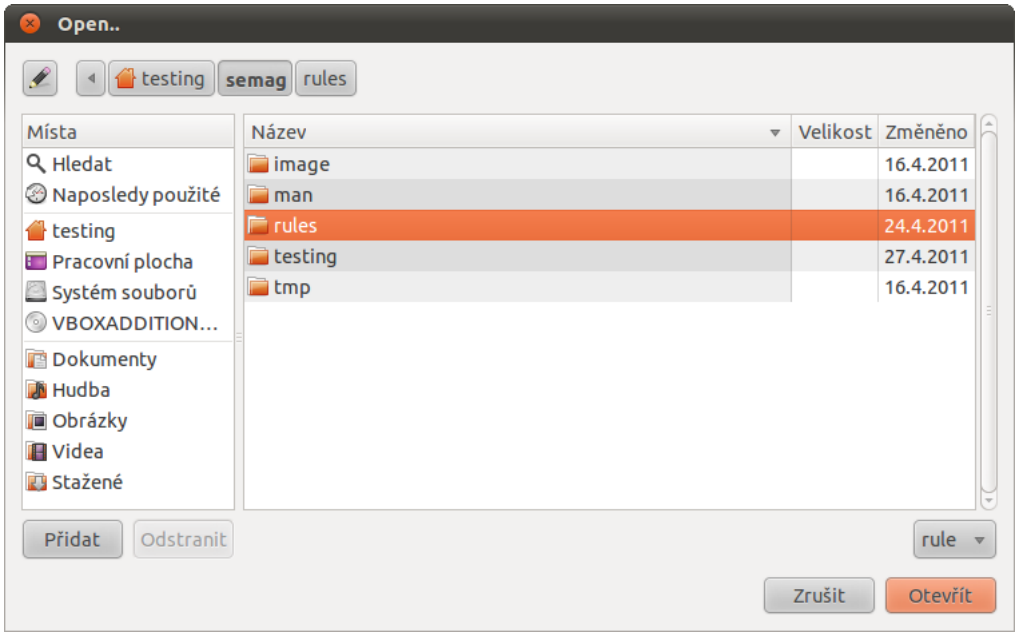


Obr. 4.2: Hlavní okno programu Semag

Hlavní okno programu nám dává přístup k nabídce na stavení firewallu, přidávání jednotlivých funkcí a zprávu pravidel. Samotná pravidla iptables budou probrána později v samostatné kapitole. Okno si můžeme popsat ve třech dílčích částech a to Menu, Ovládání a nejdůležitější část Nastavení.

Jednotlivé prvky jsou pro snadnější orientaci uživatele opatřeny vyskakovací nápovědou. Najdete zde rady o funkci jednotlivých tlačítek, případně jaká hodnota je očekávána pro doplnění.

Menu



Obr. 4.3: Načítání a ukládání pravidel

V základní menu jsou klasické možnosti ukládání a načítání dat. Vše se ukládá v textové podobě, kdy tyto soubory vlastní root. Je nežádoucí, aby měl někdo jiný k těmto souborům přístup. Mohlo by pak dojít k úmyslnému zaměnění dat útočníkem a tím otevření cesty k další nežádoucí komunikaci.

Jednotlivé prvky jsou od sebe odděleny pomlčkami. Uživatel nemusí vyplnit všechna požadovaná pole a ani by to nebylo příliš vhodné. Pokud nejsou některá data vyplněna, následuje další pomlčka. Toto se zpětně zohledňuje při načítání dat.

V následující ukázce je znázorněno několik pravidel pro FTP a SSH s rozdílným používaným zařízením a verzí Ip adresy.

```
ipv4-in-vsechna---TCP--21-ACCEPT-Povoleni FTP-
ipv4-in-eth0-192.168.2.200--tcp--22-ACCEPT-částečné povolení SSH IPv4-
ipv6-in-eth0-192.168.2.200--tcp--22-ACCEPT-částečné povolení SSH IPv4-
ipv4-in-wlan0--192.168.2.100-tcp--22-REJECT-Zakázání příchozího
                                        požadavku SSH po wifi-
```

Následují možnosti pro zapnutí a vypnutí firewallu a taktéž daemona. Poslední položka menu je nápověda s informacemi a několika radami pro uživatele.

Ovládání

Obsahuje prvky pro usnadnění práce s jednotlivými pravidly. Můžeme pravidla přidávat, mazat a také aplikovat.

- **Přidat** – vyvolá nové okno pro nastavení nového pravidla. Pravidlo se následně uloží a pozici označenou kurzorem.
- **Smazat** – smaže vybrané pravidlo firewallu.
- **Aplikovat** – podle uživatelem zadaných dat vygeneruje dva bash scripty a to pro ip adresy verze 4 a 6

Nastavení

Jedná se o základní prvek celého firewallu. Prvky jsou pro lepší přehlednost rozděleny do jednotlivých záložek. Jsou zde dostupné informace o jednotlivých pravidlech ip adres verze 4 i 6. Dále, jelikož se jedná stavový firewall, jsou dostupné informace o právě navázaných spojeních i s názvem programu který komunikuje. Neméně důležitým prvkem je i kontrola přístupu k síti. Nacházejí se zde programy, u kterých chceme omezit síťový provoz.

Smer	Zarizeni	Ip zdrojova	Ip cilova	Protokol	Port zdrojovy	Port cilovy	Politika	Poznamka
out	eth0	192.168.2.100		tcp		22	ACCEPT	Povolení SSH
in	eth0	192.168.2.200	192.168.2.100	tcp		22	ACCEPT	Povolení přichoziho SSH
out	eth0			tcp		80	REJECT	zakázání HTTP

Obr. 4.4: Nastavení pravidel Ipv4 a Ipv6

- Ipv4 a Ipv6 – nastavení pravidel pro oba druhy ip adres. Uživatel má možnost přidat nebo případně smazat pravidla. Program je určen především pro nové uživatele, a proto uživatel z důvodu přehlednosti zadává pouze základní informace. Pro lepší uživatelovu orientaci je možné přidat ke každému pravidlu poznámku, není to však nutnost. Poznámky generování pravidel neovlivňují.
- **navázaná spojení** – každé navázané spojení ve stavovém firewallu je uvedeno v souboru `/proc/net/nf_conntrack`. Není zde však uvedeno, které programy tyto spojení využívají. Je možné programy zpětně najít pomocí programu

Verze IP	Ip zdrojova	Ip cilova	Protokol	Port zdorjovy	Port cilovy	Pravdepodobna sluzba	Komunikujici program
ipv4	205.188.8.158	192.168.2.100	tcp	59209	5190	America-Online	qutim
ipv4	209.85.149.102	192.168.2.100	tcp	51087	80	World Wide Web HTTP	chromium
ipv4	74.125.232.237	192.168.2.100	tcp	47311	80	World Wide Web HTTP	chromium

Obr. 4.5: Zobrazení probíhající komunikace

`lsof`. Ten má přehled o vnitřní komunikaci operační systému. Při vyhledávání jména programu není nutné kontrolovat všechny prvky spojení (ip adresy, porty), stačí nám z `/proc/net/nf_conntrack` získat pouze zdrojový port. Není možné, aby ve stejnou dobu používalo port více aplikaci. Tím lze následně vyhledat s pomocí `lsof` a jeho příslušnými parametry jméno programu.

Aby bylo možné zajistit v této záložce výstupní údaje, je nutné, aby byl stavový firewall aktivní. Pokud není, není ani žádné uložené probíhající spojení a tím nelze dohledat další informace.

- **omezené programy** – pokud je povolený deamon firewallu, filtrují se nová navázaná spojení. Daemon po každém požadavku na nové spojení zkontroluje, zda jméno není v seznamu obsaženo. Pokud ano, dojde k zahození packetu a tím i zakázání komunikace. Pokud daný program není na seznamu, je packet navrácen do sítě, může dojít k vytvoření spojení.

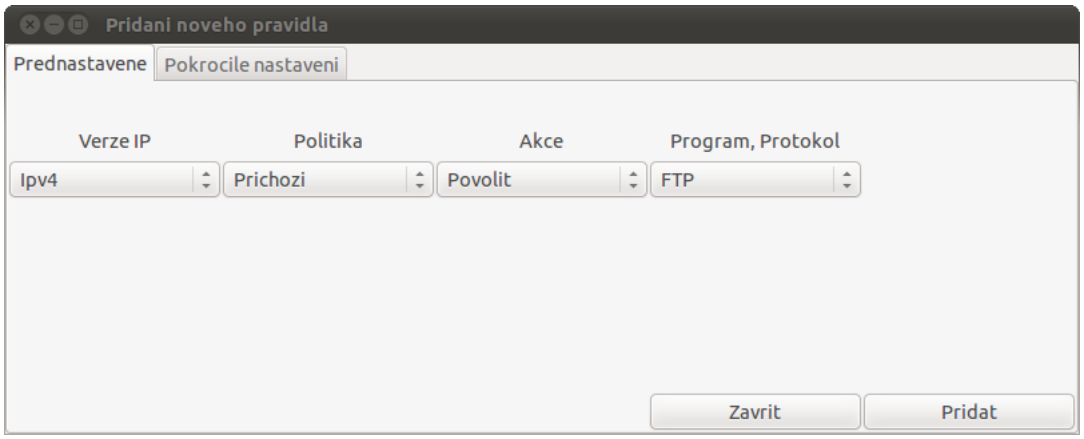
4.3.3 Pomocná okna

Jedná se o pomocné prvky pro usnadnění práce s firewallem. Jsou zde dvě možnosti přidání nových pravidel s různě podrobným nastavením. Další prvek je dotazování uživatele na nově navazovaný provoz.

Přidání pravidel

Program je navržen pro méně zkušené uživatele a s ohledem na to je navrhnuto menu pro přidávání pravidel. Neznamená to ale, že by toto nastavení nějak omezovalo již zkušenější uživatele. Nejsou zde uvedeny složitější možnosti využití nastavení Netfilter jako logování, přesměrování portů atd. Jedná se o obdobu používaných programů na operačním systému Windows.

Menu je rozděleno na dvě položky a to Přednastavené nastavení a Pokročilé nastavení. Rozdíl je především v možnostech nastavení.

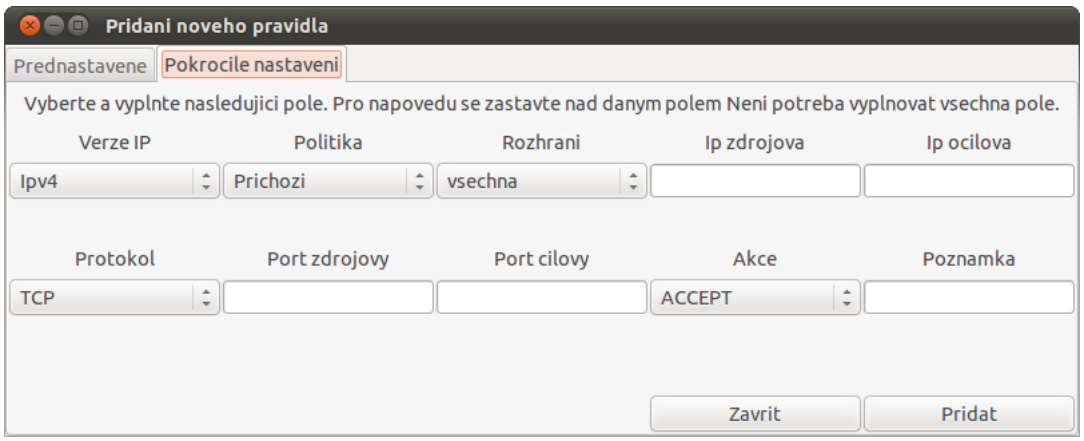


Obr. 4.6: Přednastavené nastavení pravidel

Přednastavené nastavení

Jedná se nejjednodušší formu nastavení při zachování co nejlepší funkčnosti. Menu je rozděleno na čtyři části, kde na počátku vybereme protokol. Následuje nastavení, do jakého směru se má pravidlo vytvořit a zda se tento provoz má zakázat nebo povolit. Poslední nejdůležitější částí je výběr služby, pro kterou se má pravidlo vytvořit. Jsou zde zastoupeny nejznámější používané protokoly a to FTP, SSH, SMTP, POP3, IMAP, SAMBA, HTTP. Při výběru některého z protokolů se automaticky nastaví potřebná konfigurace pro danou službu. V případě služby HTTP jsou to porty 80 pro samotné HTTP a 443 pro HTTPS.

Pokročilé nastavení pravidel



Obr. 4.7: Pokročilé nastavení pravidel

Oproti předchozímu nastavení je zde více uživateli rozšířen prostor pro jednotlivé definovaní pravidel. Zde se jíž předpokládá uživatelova alespoň základní znalost síťového provozu, případně znalost alespoň na jakých portech či rozsahu chce s provozem pracovat. Následně si popíšeme jednotlivé prvky nastavení.

- **Verze IP** – jsou zde možnosti zvolení protokolu IPv4 nebo IPv6.
- **Politiky** – důležité pro uřčení pro jaký směr se mají pravidla vytvořit. Máme zde Příchozí provoz, odpovídající v Netfilter INPUT, Odchozí OUTPUT a průchozí FORwARD.
- **Rozhraní** – je velký rozdíl z jakého síťového zařízení provoz přichází. Na počítači je možné mít více sítových karet.. V případě notebooků to bývá klasická sítová karta a wi-fi karta. Je možné nastavit pro každé síťové zařízení jiná pravidla.

  Protože u každého může být jiný počet síťových karet, spustí se při zapnutí programu jejich detekce pomocí programu `ip` a následně se jednotlivá rozhraní vyfiltrují.
- **Ip adresy** – zde je možnost zadat zdrojovou i cílovou adresu a to ve formátu IPv4 nebo IPv6, podle zvolené verze na počátku konfigurace.
- **Protokol** – jedná se o protokoly vyšších vrstev. V tomto případě jsou pro uživatele dostupné protokoly TCP a UDP.
- **Porty** – rozsah povolených portů je až na hranici 65535. Zde se předpokládá nejvíce využití dolního rozsahu.
- **Akce** – jedná se definování, jak s provozem naložit.
- **Poznámka** – jedná se o doplňkové pole pro uživatele, kde si může zapsat poznámku například k čemu dané pravidlo slouží.

4.3.4 Daemon

Doposud jsme se zabývali samotným nastavováním chodu programu Semag a definování jednotlivých pravidel. Firewall je nyní možné bez omezení používat pro konfiguraci pravidel. Pokud chceme stávající firewall rozšířit, je možné zakomponovat do systému NFqueue a s jeho pomocí následně odchytávat provoz. Samotný program zatím nepotřeboval doinstalovat externí knihovny. Stačil nám pouze Python 2.6 a pygtk, které jsou už v základní instalaci obsaženy. Pro funkci front je potřeba doinstalovat knihovny `nfqueue-bindings-python python-dpkt`.

Pro funkci daemona je potřeba do firewallu přidat pravidlo přeposílající vybraný provoz do fronty. Samotná implementace je jednoduchá, stačí v pravidlech Netfilter definovat výstup na NFgueue.

```
$IPTABLES -A OUTPUT -m state --state NEW -j NFQUEUE --queue-num 1
```

V ukázce je znázorněna část stavového firewallu s filtrovaným výstupem. Všechen nově navazovaný provoz je směrován na NFqueue do fronty 1. Pokud nezajistíme, aby některý program s frontou pracoval, nebo na packet zaslaný do fronty nebudeme reagovat, docílí se stejného efektu jako by se daný provoz přímo zahazoval. Aby se této situaci, předešlo je potřeba zavést daemona do systému už při jeho startu jako samostatný program. Toho lze docílit požádáním systému o přidělení UID. Tento identifikátor je vhodné otestovat, zda nevznikla kolize s init. Daemona lze ovládat třemi příkazy a to:

- **semag start** – dojde k inicializaci daemona. Po startu se načtou z externího souboru data s informacemi, které programy se mají omezovat. Systém nám přidělí PID, pod kterým bude spuštěn daemon celou dobu svého běhu. Toto číslo zároveň uložíme do tmp, aby bylo možné později daemon vypnout.

  Program neustále hlídá frontu, zda nejsou nové požadavky a pokud ano, provede se jejich zpracování. Podle uživatelem definovaných dat následně dojde k zahození nebo navrácení parketu do sítě.

- **semag stop** – provede se ukončení běhu daemona. Ze souboru uloženého v tmp se získá PID číslo a tento proces se následně ukončí.

- **semag restart** – jedná se o kombinaci příikazů stop a start s kontrlolou předchozího běhu procesu.

Detekce pokusu o průnik je hlídána daemonem. Pokud hlídaný provoz přesáhne parametry, je dána uživateli možnost na tento stav reagovat. Klasický příklad situace je při útoku s příznaky SYN, FIN.

4.3.5 Netfilter pravidla

V dosavadní části byly probráno vytvoření jednotlivých pravidel a zabezpečení jejich běh.Nyní je třeba definovat samotná pravidla programu Semag. Uživatel nemá možnost spravovat všechna pravidla, která Semag vytváří. Je nutno ponechat některá pravidla mimo dosah uživatele pro samotný běh programu Semag. Jedná se zejména o pravidla, která nemá možnost uživatel sám definovat. Jsou to možnosti změn základních politik, nastavení potřebných pravidel pro funkci stavového firewalu a v

neposlední řadě přesměrování nových požadavků do nfqueue. Bez těchto základních pravidel by firewall fungoval, ale nejspíše by neplnil svůj hlavní cíl a to ochranu uživatele.

Program Semag vytváří po každém generování pravidel dva soubory. Pro každou verzi IP protokolu je vlastní konfigurační soubor. Tyto soubory jsou použity při startu počítače a jsou v nich definována jednotlivá pravidla. Protokoly jsou rozděleny z důvodu jiného konfiguračního programu pro IPv6. Protokol IPv4 využívá `iptables` a IPv6 `ip6tables`. Další důležitý prvek je nutnost požít nepatrně jiná pravidla pro novější protokol.

Samotná pravidla se dají rozdělit do dvou skupin. Do předem pevně definovaných pravidel a prostoru pro pravidla definována uživatelem. Nyní si ukážeme a popíšeme jednu z možných konfigurací firewallu pro IPv4. Toto je pouze ukázkové nastavení. Od pravidel nastavených v programu Semag se může lišit.

```
#!/bin/bash
# Pravidla pro nastaveni nastaveni netfilter
#smazani predchoziho nastaveni iptables
/sbin/iptables-restore < rules/default

#zakladni politika##########################################
IPTABLES='/sbin/iptables'

#vsechen prichozo provoz bude defaultne zahazovan
$IPTABLES -P INPUT DROP

#odchozi provoz bude defaultne zahazovan
$IPTABLES -P OUTPUT ACCEPT

#preposilani provoz bude defaultne zahazovano
$IPTABLES -P FORWARD DROP

#priprava retezcu pro uzivatelska data####################

#retezec pro prichozi provoz
$IPTABLES -N in

#retezec pro odchozi provoz
$IPTABLES -N out
```

```
#retezec pro presmerovani
$IPTABLES -N forw

#retezec INPUT ##########################################
#povoleni loopback
$IPTABLES -A INPUT -i lo -j ACCEPT
$IPTABLES -A INPUT -d 127.0.0.1 -j ACCEPT

#povoleni protokolu icpm
$IPTABLES -A INPUT -p icmp -j ACCEPT

#povoleni DNS
$IPTABLES -A INPUT -p UDP --dport 53 -j ACCEPT
$IPTABLES -A INPUT -p TCP --dport 53 -j ACCEPT

#pridani pravidel uzovatele
$IPTABLES -A INPUT -j in

#povoleni jiz navazaneho spojeni
$IPTABLES -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

#retezec OUTPUT #########################################
$IPTABLES -A OUTPUT -p ICMP -j ACCEPT

#pridani pravidel uzovatele
$IPTABLES -A OUTPUT -j out

#vsechny npve navazovana spojeni jsou koutrolovana daemonem
$IPTABLES -A OUTPUT -m state --state NEW -j NFQUEUE --queue-num 3

#retezec FORWARD ########################################
$IPTABLES -A FORWARD -j forw
```

Jedná se klasický bash skript kde, se postupně vykonají jednotlivé příkazy. Na začátku zajistíme, aby pokaždé došlo ke kompletnímu smazání předešlých pravidel. Provádí se to nahráním čisté konfigurace přímo do Netfilter. Nejedná se o stejný stejný formát jako v případě zadávání iptables. Toto jsou už upravená pravidla pro vnitřní běh Netfilter.

Pravidla pro základní politiku jsou zvolena s ohledem na bezpečnost, ale i funkčnost. Všechen příchozí provoz bude zahazován, pokud nebude některým pravidlem povolen. Odchozí provoz je povolen, pokud není některým pravidlem omezen.

Uživatelem definovaná pravidla se ukládají do oddělených řetězců mimo hlavní větev pravidel. Je tím tak zajištěna soudržnost a posloupnost pravidel. Uživatel nemá možnost nějak výrazně zasáhnout do základního běhu. Jsou vytvořeny tři nové řetězce in, out a forw, do kterých je uživatelův provoz směrován.

Provoz na lokální smyčce a lokálním rozhraní není omezen. Je používán některými programy a jeho omezení by mohlo způsobit jejich nefunkčnost. Tento provoz se nedostává do sítě a tedy ani neohrožuje bezpečnost. V tomto případě je povolen celý provoz icmp bez omezení. Záleží na uživateli, jak hodně bude chtít provoz omezit. Není možné ale tento protokol zakázat celý. Je možně pouze omezit počet povolených požadavků za určitou dobu.

Nejdůležitější částí jsou tato pravidla:

```
$IPTABLES -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

$IPTABLES -A OUTPUT -m state --state NEW -j NFQUEUE --queue-num 3
```

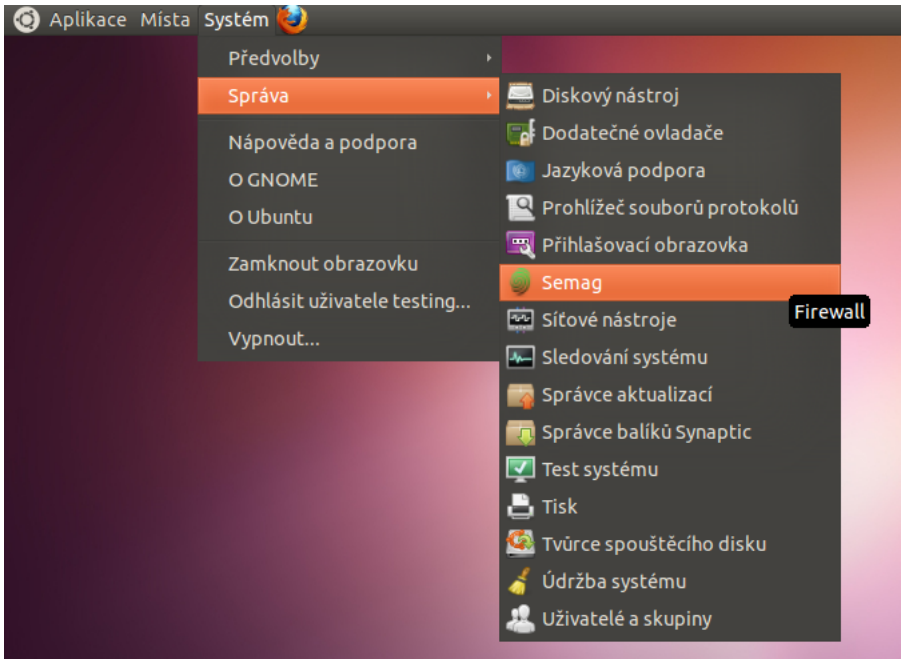
První pravidlo nám povoluje všechen příchozí navázaný provoz. V případě nového příchozího provozu je potřeba v tomto případě přidat pravidla ručně. Jedná se o služby nastavené uživatelem kupříkladu ssh server (sshd). Pro tento případ má uživatel možnost snadno vytvořit pravidla pomocí přednastavených pravidel. Druhé pravidlo zasílá všechen provoz do daemona na frontu 3. V případě Netfilter se jedná o všechen nenavázaný provoz a to i za jistých podmínek i UDP a icmp.

Pro nastavení IPv6 je soubor téměř stejný až na několik vyjímek. Používá se konfigurační nástroj `ip6tables` a je potřeba přidat několik pravidel.

```
$IPTABLES -A INPUT -p ipv6-icmp -j ACCEPT
$IPTABLES -A INPUT -p TCP --dport 53 -j ACCEPT
$IPTABLES -A INPUT -p UDP --dport 53 -j ACCEPT
$IPTABLES -A INPUT -p TCP --dport 546 -j ACCEPT
$IPTABLES -A INPUT -p UDP --dport 546 -j ACCEPT
```

Přidána jsou pravidla pro povolení icmp a povolen je i DNS provoz. Důležitou částí je povolení DHCP. IP adresa verze 6 se skládá z několika částí a jedna tato část se načítá z externího DHCP serveru. Pokud by bylo dhcp zakázáno, nebylo by síťové rozhraní s IPv6 funkčí.

4.3.6 Začlenění do systému



Obr. 4.8: Začlenění programu Semag do menu OS.

Pro správnou funkčnost programu Semag je nutné ho vhodně začlenit do operačního systému. Je sice možné spustit program ručně z domácího adresáře, ale tato možnost není většinou pro uživatele příliš vhodná. Lepším řešením je dát uživateli možnost spustit program přímo z hlavní nabídky operačního systému. Je proto potřeba vytvořit příslušný soubor semag.desktop a umístit ho do souboru /usr/share/applications/

Pro nastaven se používají tyto parametry:

```
[Desktop Entry]
Version=1.0
Name=Semag
Name[cs]=Semag
Comment=Firewall
Comment[cs]=Firewall
GenericName=Semag
GenericName[cs]=Semag
Exec=semag
Terminal=false
X-MultipleArgs=false
```

```
Type=Application
Icon=semag
Categories=GNOME;GTK;Firewall;Settings;System;
```

Jednotlivé parametry udávají vlastnosti programu Semag, jeho umístění a jaké soubory se mají použít. V našem případě je program Semag umístěn do menu Systém a podmenu Správa. Pro uživatele je tak tento program snadno dosažitelný. Zbylé hodnoty definují popis programu pro různé jazykové verze a název ikony.

Instalace programu Semag do operačního systému není složitá. Uživatel si musí přečíst soubor README.txt s uvedeným postupem instalace. Instalace je popsána po jednotlivých krocích. Nejprve je nutné doinstalovat potřebné knihovny. Pro uživatele jsou uvedeny přesné a velice jednoduché příkazy. Stačí tyto příkazy pouze překopírovat do příkazové řádky a ostatní už zajistí systém. Jedná se o příkaz sudo apt-get install a dále potřebné balíčky pro instalaci.

Dalším krokem je samotná instalace programu Semag. Jediné co potřebuje uživatel udělat, je spustit instalační skript rovněž příkazem uvedeným v souboru. Skript se následně postará o začlenění jednotlivých částí do systému a na patřičná místa.

Posledním krokem uživatele je konečné spuštění programu a seznámení se s jeho ovládáním.

Celá instalace není tedy složitá a je koncipovaná pro méně zkušené uživatele.

4.3.7 Možnost využití na jiných OS než Ubuntu

Program Semag není omezen pouze na operační systém Ubuntu, i když je pro něj navržen. Pro svou správnou funkci potřebuje jen několik externích knihoven, které nejsou běžně v základní instalaci obsaženy. Preferované prostředí je Gnome. V případě jiného prostředí je potřeba doinstalovat potřebné GTK knihovny pro zajištění chodu. Uživatelé operačních systémů založených na Debian mají instalaci jednodušší. Všechny potřebné knihovny se dají jednoduše nainstalovat pomocí balíčkového systému `apt-get`. Například u operačního systému Fedora založeného na Ret Hat nejsou všechny balíčky dostupné a museli by se ručně doinstalovat.

Testování programu proběhlo na operačních systémech Debian wheezy a Ubuntu 11.04 Natty Narwhal s rozhraním Gnome

4.4 Testování

Firewall byl testován z několika hledisek. Jako první byla testována část omezení síťové komunikace programů. Postup testování byl následující.

4.4.1 Omezení síťové komunikace programů

Do povolené databáze byly přidány programy Amarok,Chromium a qutim. Do druhé databáze byl přidán Skype. Tento program je velice vhodný pro toto testování. Při jeho aktivaci se pokouší navazovat mnoho spojení pomocí TCP i UDP. Bylo zjištěno, že pokud nenajde klasické spojení se servery, načte si databázi uživatele pomocí HTTPS (provoz byl povolen pouze na portech 80 a 443). Výsledky dopadly dle očekávání. U povolených programů se nastavení nijak neprojevilo, zatímco programu Skype se nepovedlo přihlásit. Poslední test v této části byla komunikace programu neobsaženého v ani jedné z databází. Po spuštění Firefoxu byl uživatel dotázán, jak má být s provozem naloženo. Program Skype byl následně povolen a otestován. Program se v pořádku přihlásil a byl úspěšně proveden zkušební hovor.

4.4.2 Časové zpoždění

Při užívání tohoto firewallu bude docházet k mírnému zpoždění při navazování nové komunikace. Aby bylo možné zjistit, jak velké zpoždění vznikne, použijeme programy `time` a `wget`. První měření proběhlo při vypnutém firewallu. Testována byla doba, za kterou bude stažena stejná webová stránka. Použitý příkaz byl `time wget www.vutbr.cz`. Výsledek jsme dostaly tento:

```
real    0m0.243s
user    0m0.000s
sys 0m0.008s
```

Druhé měření bylo provedeno při zapnutém firewallu a programu wget uveden v databázi povolených programů.

```
real    0m0.280s
user    0m0.000s
sys 0m0.008s
```

Z výsledku je znát nepatrné zpoždění, ale to může být způsobeno i zpožděním dat v síti. Měření bylo prováděno opakovaně a byly vybrány průměrné hodnoty.

V úvahu přicházejí ještě dvě měření. V jednom případě bude program wget uveden v nepovolených spojeních a v takovém případě nedojde ke stažení dat. Druhá situace nastane, pokud není wget uveden ani v jedné z databází. U tohoto případu odpovídá čas době, za kterou je uživatel schopný rozhodnout o daném spojení a času na samotný přenos.

4.4.3 Testování pravidel

Pro testování pravidel byl použit program `nmap`. Jeden z provedených testů se pokoušel detekovat operační systém a otevřené porty. Ve firewallu je povoleno DNS na portu 53 a server sshd s portem 4455.
Při testování s vypnutým firewallem jsme obdrželi tento výsledek:

```
testing@testing-VirtualBox:~$ sudo nmap -A -T4 -p- 192.168.2.3

Starting Nmap 5.21 ( http://nmap.org )
Nmap scan report for 192.168.2.3
Host is up (0.00036s latency).
Not shown: 65530 closed ports
PORT      STATE SERVICE VERSION
25/tcp    open  smtp    Postfix smtpd
|_smtp-commands: EHLO ml, PIP\begin{list}{NING, SIZE 10240000, VRFY,
                    N, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN
    111/tcp   open  rpcbind
    | rpcinfo:
    | 100000  2    111/udp  rpcbind
    | 100024  1  35516/udp  status
    | 100000  2    111/tcp  rpcbind
    |_100024  1  43555/tcp  status
    4455/tcp  open  ssh     OpenSSH 5.5p1 Debian 6 (protocol 2.0)
    | ssh-hostkey: 1024 fe:12:36:29:b5:c5:e1:44:8c:fc:0a:b1:2c:c7:ba:b2 (DSA)
    |_2048 2c:97:1b:ce:70:1e:8e:c3:60:dd:96:5c:63:85:20:01 (RSA)
    6000/tcp  open  X11     (access denied)
    43555/tcp open  rpcbind
    MAC Address: xx:xx:xx:xx:xx:xx (Compal Information (kunshan) CO.)
    Device type: general purpose
    Running: Linux 2.6.X
    OS details: Linux 2.6.17 - 2.6.31
    Network Distance: 1 hop
    Service Info: Host:  ml; OSs: Linux, Unix

    HOP RTT     ADDRESS
    1   0.35 ms 192.168.2.3]
```

Z výsledku je znát, že pouhým skenováním nezabezpečeného systému lze zjistit mnoho užitečných informací. Od verze používaného systému po jednotlivé používané

služby.
Druhý test byl proveden se zapnutým firewallem.

```
testing@testing-VirtualBox:~$ sudo nmap -A -T4 -p- 192.168.2.3

Starting Nmap 5.21 ( http://nmap.org ) at 2011-05-22 22:21 CEST
Nmap scan report for 192.168.2.3
Host is up (0.00026s latency).
Not shown: 65533 filtered ports
PORT     STATE  SERVICE VERSION
53/tcp   closed domain
4455/tcp open   ssh     OpenSSH 5.5p1 Debian 6 (protocol 2.0)
| ssh-hostkey: 1024 fe:12:36:29:b5:c5:e1:44:8c:fc:0a:b1:2c:c7:ba:b2 (DSA)
|_2048 2c:97:1b:ce:70:1e:8e:c3:60:dd:96:5c:63:85:20:01 (RSA)
MAC Address: xx:xx:xx:xx:xx:xx (Compal Information (kunshan) CO.)
Device type: general purpose|firewall|specialized|proxy server|webcam|WAP
Running (JUST GUESSING) : Linux 2.6.X|2.4.X (96%),
Check Point Linux 2.4.X (91%), Crestron 2-Series (91%),
SonicWALL embedded (89%), AXIS Linux 2.6.X (88%),
Gemtek embedded (88%), Siemens embedded (88%), Chumby embedded (88%)
Aggressive OS guesses: Linux 2.6.17 - 2.6.31 (96%), Linux 2.6.22 (93%),
Linux 2.6.19 - 2.6.31 (93%), Linux 2.6.24 - 2.6.31 (93%),
Linux 2.6.13 - 2.6.28 (93%), Linux 2.6.18 (93%),
Linux 2.6.18 - 2.6.27 (92%), Linux 2.4.20 (Red Hat 7.2) (92%),
Linux 2.6.9 - 2.6.18 (92%), Linux 2.6.22 - 2.6.23 (92%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
Service Info: OS: Linux

HOP RTT     ADDRESS
1   0.26 ms 192.168.2.3
```

Z druhého testu jsem získali už méně informací. Dostupné porty jsou pouze námi povolené. Nebylo zjištěno jméno počítače a operační systém byl pouze odhadován. Pokud by byl nmap spuštěn v rozsahu portu 0-1024, nedokáže OS ani odhadnout.

Testů pravidel bylo provedeno několik a nebyly nalezeny významnější chyby. Drobné chyby byly následně opraveny.

ZÁVĚR

Cílem této práce bylo prostudovat možnosti zabezpečení vybrané distribuce GNU/Linux a na tomto základě navrhnou vhodné síťové zabezpečení. Pro tvorbu zabezpečení byla vybrána distribuce Ubuntu. Pří výběru byl brán ohled na počet uživatelů používající tento systém. Ubuntu je také vhodné jako prostředí pro uživatele začínající s GNU/Linux.

Osobní informace jsou dnes stále více žádaným zbožím. Jednou z možných cest jak získat informace, je napadení uživatele pomocí sítě. Pro operační systémy firmy Microsoft existuje mnoho softwarových produktů na řešení problémů se zabezpečením. Systémy GNU/Linux mají výhodu v jiném řešení správy uživatelů a tím lze získat lepší systémové zabezpečení, ale nevyloučí to možnost napadení uživatele ze sítě. GNU/Linux má v jádře zakomponovaný filtr Netfilter. Samotný filtr nenastavuje jednotlivá pravidla. Využívají se nástavby jako iptables. Existuje mnoho jednorázových generátorů pro Netfilter. Programů s možností lépe definovat jednotlivá pravidla a pracovat s protokolem IPv6 je už méně.

Výsledkem této práce je program pro nastavení pravidel Netfilter vhodný i pro nezkušené uživatele. Cílem bylo vytvořit prostředí pro méně zkušené uživatele, ale ne na úkor funkčnosti. Výsledný program dává uživateli možnost snadno nastavit pravidla pro IPv4 a IPv6. Je důležité nespoléhat pouze na nastavení protokolu IPv4. Jak ukázalo testování, program openssh-server neočekává provoz pouze na IPv4 ale i IPv6. Tím by mohla v systému vzniknout bezpečností mezera. Vytvořený program by měl tomuto zamezit a dát možnost uživateli nastavit vše dle jeho představy. Další důležitá funkce se týká omezení síťové komunikace programů. Tento systém je většinou znám ze systémů Microsoft Windows. Ve firewallu je definován seznam programů s informacemi, zda mohou používat síťovou komunikaci. U GNU/Linux se tyto systémy vyskytují pouze ojediněle. Dříve se systémy GNU/Linux používaly převážně jako serverová řešení, kde se nemusela pravidla často měnit. V současné době se stále často využívají pro osobní počítače. Program kontroluje nově navazovaná spojení a rozhoduje, jak s nimi bude nakládáno. Pokud pro daný proces není dosud vytvořeno pravidlo, je uživatel dotázán jaká akce se má provést. Při testování bylo zjištěno, že zpoždění takto navazovaného nového provozu je zanedbatelné.

Výsledný program dává uživateli k dispozici odlišné řešení než je na GNU/Linux používáno. Přináší uživateli možnost snadno nakonfigurovat stavový firewall s možností rozhodnout, kterým programům bude povolena síťová komunikace.

LITERATURA

[1] KRČMÁŘ, Petr.

Historie operačního systému GNU/Linux. Http://www.root.cz/ [online].

2006, poslední aktualizace 21. 03. 2006 [cit. 11. 10. 2010].

Dostupné z URL:

<http://www.root.cz/texty/historie-operacniho-systemu-gnulinux/>.

[2] *Výsledky ankety o nejoblíbenější distribuci 2010. http://www.abclinuxu.cz* [online].

2010, poslední aktualizace 15. 06. 2010 [cit. 20. 10. 2010].

Dostupné z URL:

<http://www.abclinuxu.cz/clanky/vysledky-ankety-o-nejoblibenejsi-distribuci-2010>.

[3] *Ubuntu* [online].

2010, poslední aktualizace 14. 12. 2010 [cit. 03. 11. 2010].

Dostupné z URL:

<http://www.ubuntu.cz/>.

[4] DOSTÁLEK, Libor; KABELOVÁ, Alena.

Velký průvodce protokoly TCP-IP a systémem DNS

Computer Press, 2000. 426 s. ISBN 80-7226-323-4.

[5] kolektiv autorů

Linux Dokumentační projekt 2. aktualizované vydání

Computer Press, 2001. 1017 s. ISBN 80-7226-503-2.

[6] TOXEN, Bob

Bezpečnost v Linuxu Prevence a odvrácení napadení systému

Computer Press, 2003. 858 s. ISBN 80-7226-716-7.

[7] *Application Layer Packet Classifier for Linux* [online].

2009, poslední aktualizace 07. 01. 2009 [cit. 20. 11. 2010].

Dostupné z URL:

<http://l7-filter.sourceforge.net/>.

[8] Eychenne EYCHENNE, Herve, et al.

iptables(8) - Linux man page [online].

[cit. 21. 11. 2010].

Dostupné z URL:

<linux.die.net/man/8/iptables>.

[9] The GNOME Project and PyGTK Team

PyGTK 2.0 Reference Manual [online].

[cit. 15. 04. 2011].

Dostupné z URL:

<http://www.pygtk.org/docs/pygtk/>.

[10] *Dostupné konfigurátory pro Linux www.linuxsoft.cz* [online].

[cit. 15. 04. 2011].

Dostupné z URL:

<http://www.linuxsoft.cz/sw$_l$*ist.php*?*search*$_w$*orld* = *firewalltlacitko* = *Hledat*>.

[11] HORÁK, Jan.

Jak správně na SELinux. Http://www.root.cz/ [online].

2007, poslední aktualizace 17. 12. 2007 [cit. 11. 5. 2011].

Dostupné z URL:

<http://www.root.cz/serialy/jak-spravne-na-selinux/>.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

GNU	GNU's Not Unix
TCP	Transmission Control Protocol
IP	Internet Protocol
ARP	Address Resolution Protocol
RARP	Reverse Address Resolution Protocol
ICMP	Internet Control Message Protocol
IGMP	Internet Group Management Protocol
QoS	Quality of Service
SSH	Security shell
FTP	File Tranfer Protocol
SFTP	SSH File Transfer Protocol
HTTP	Hyper Text Transfer Protocol
POP3	Post Office Protocol version 3
IMAP	Internet Message Access Protocol
SMTP	Simple Mail Transfer Protocol
TELNET	Telecommunication Network
DNS	Domain Name System
DHCP	Dynamic Host Configuration Protoco