Univerzita Hradec Králové
Fakulta informatiky a managementu
Katedra informačních technologií

Analýza a optimalizace softwarových firewall na operačních systémech Linux

Diplomová práce

Autor: Miroslav Bartoš

Studijní obor: K-AI2

Vedoucí práce: Mgr. Josef Jan Horálek, Ph.D.

Hradec Králové březen 2016

Prohlášení

Prohlašuji, že jsem diplomovou práci zpracoval samostatně a s použitím uvedené literatury.

V Poličce dne 27.3.2016 Miroslav Bartoš

Anotace

Práce pojednává o problematice tvorby firewallu na počítači s operačním systémem linux. Nejprve jsou popsány typy jednotlivých firewallů, které jsou používané ve valné většině případů. Další část je věnována vlastním principům ochrany počítačové sítě proti možným útokům. Popsány jsou i možné nedostatky firewallů a případy, kdy je firewall neúčinný. Následující kapitola je věnována firewallům v nejužívanějších linuxových distribucích – Ubuntu a Fedora. Jsou popsány konfigurace pomocí nejčastějších konfiguračních nástrojů těchto distribucí. V předposlední části jsou analyzovány typy nejčastějších síťových útoků, proti kterým by měl firewall síť chránit. Nakonec jsou sestaveny konfigurace pro dva typy firewallů – jeden jednoduchý, který může sloužit jako oddělení jakýchkoliv dvou sítí, druhý s demilitarizovanou zónou, který již vyžaduje složitější konfiguraci tzv. na míru, neboť je nutné přesměrování toku dat běžících na určitých portech do konkrétních počítačů. V Závěru práce jsou popsány výsledky základního testování takto nastavených firewallů pomocí v operačním systému Linux běžně dostupných testovacích nástrojů.

Anotation

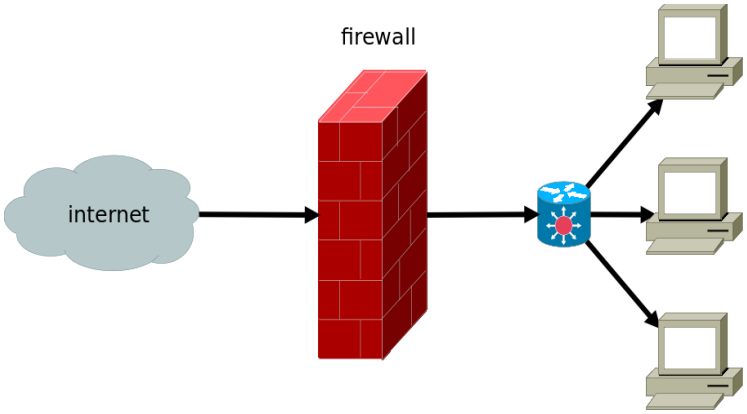
Title: Analysis and optimization software firewall on Linux OS

The Diploma Thesis is about the creation of the firewall on a PC running Linux. First there are described various types of firewalls that are used in the vast majority of cases. The next section is devoted to the principles of self-protection of the network against possible attacks. Described are also possible shortcomings of firewalls and cases where the firewall is ineffective. The following chapter is dedicated to firewalls, the most used Linux distributions - Ubuntu and Fedora. There are described configurations using the most common configuration tools of these distributions. In the penultimate section we analyze the most common types of network attacks, against which a firewall should protect the network. Finally, there are assembled configurations for two types of firewalls - one simple, which can serve as a separation of any two networks, one with a demilitarized zone, which no longer requires a complicated tailor-made configuration as it is necessary to divert the flow of data running on specific ports to specific computers. The final part describes the results of basic testing tools of such set firewalls using commercially available test instruments in a Linux operating systém.

Obsah

1 Úvod

Na konci 80. let minulého století vznikla s rozvojem internetu i potřeba větší ochrany počítačů před útokem přes počítačovou síť. Bylo třeba nějak kontrolovat a rozlišovat především vstupní data přicházející do počítače v chráněné počítačové síti. Mělo se jednat tedy o jakousi hradbu či plot s kontrolou u vstupní brány, která kontrolovala, kdo smí a kdo



Obrázek 1: firewall, zdroj: autor

nesmí dovnitř. Pro systém, který tuto kontrolu umožňoval, se právě ke konci 80. let minulého století začal používat termín firewall.

Obrázek 2: vrstvy OSI modelu, zdroj: autor

Firewall měl tedy zajistit, aby se nikdo nežádoucí nedostal do vnitřní počítačové sítě, kde se nacházejí chráněná data, a aby si někdo cizí nemohl prohlížet provoz v této síti. Firewall tak plní především tyto funkce:

- chrání vnitřní síť před neoprávněným přístupem do ní;
- chrání vnitřní síť před neoprávněným čtením provozu v síti;

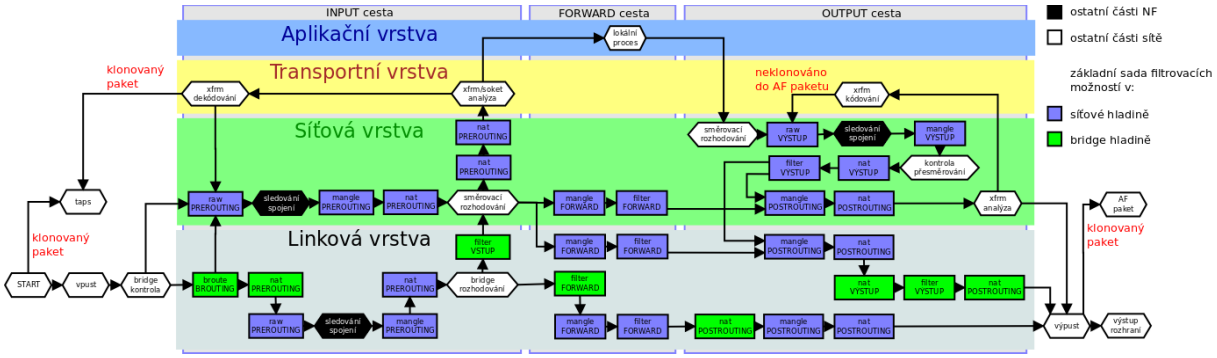
- brání průniku nežádoucího softwaru do vnitřní sítě.

Firewally vznikly jak hardwarové, tak softwarové. První softwarové firewally byly nazývány paketové filtry, v angličtině uváděné jako stateless firewall. Název vycházel z vlastnosti filtru neukládat a nepamatovat si informace o stavu povolených spojení. Tento nedostatek řešila až druhá generace softwarových firewallů – stavový firewall.

U zrodu prvních komerčně úspěšných firewallů stála izraelská firma Check Point[1]. Její produkt FireWall-1 se stal prvním velmi úspěšným stavovým firewallem. Do té doby se používaly především firewally paketové a aplikační, pracující na třetí, resp. sedmé vrstvě OSI modelu. FireWall-1, jako součást balíku VPN-1, se používá dodnes. Na rozdíl od většiny ostatních stavových firewallů, které pracují na čtvrté vrstvě OSI modelu, pracuje tento mezi druhou a třetí vrstvou a provádí tak kontrolu na nejnižší softwarové vrstvě. Je tak tedy schopen chránit nejen vnitřní počítačovou síť, ale i sám sebe.

Firewally můžeme současně rozdělit podle typu takto:

- paketový filtr (firewall);
- stavový filtr (firewall);
- aplikační filtr (firewall).



Obrázek 3: tok paketů přes netfilter, upraveno podle: Jengelh, CC BY-SA

Nicméně většina firewallů kombinuje uvedené filtry. V době, kdy se Linux stal stabilním operačním systémem, začal se používat i pro implementaci firewallů. Při výběru vhodné technologie padla volba na ipfilter z operačního systému OpenBSD. Po začlenění do linuxového jádra se změnil jeho název na netfilter. V linuxovém jádře 2.0 se pro tento paketový filtr používal administrativní nástroj nazvaný Ipfwadm, v jádře 2.2 pak nástroj ipchains. Do jádra 2.6 byl implementován administrátorský nástroj iptables, který je ve velké míře používán dodnes.

Nicméně vývojáři netfilteru nezdokonalovali pouze nástroje na vlastní konfiguraci firewallu, ale do nové verze linuxového jádra 3.13 začlenili zcela přepracovaný paketový filtr/firewall s názvem nftables. Tento framework se skládá podle [2] ze tří hlavních částí:

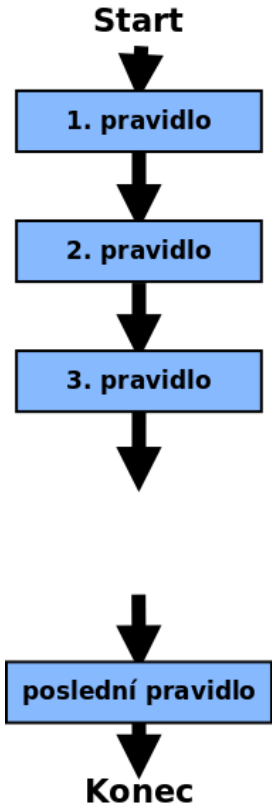
- jádrové implementace;
- libnl – knihovna pro komunikaci netlink;
- nft – uživatelského konfiguračního nástroje.

Nft využívá jinou syntax než původní nástroje. Nicméně kvůli zpětné kompatibilitě s iptables a ip6tables je nftables doplněn ještě o vrstvu starající se o překlad pravidel z uvedených starších konfiguračních nástrojů.

2 Analýza firewall řešení

2.1 ACL – Access Control List (seznam pravidel pro přístup)

Termínem ACL se podle [3] označuje seznam pravidel, podle kterých vyhodnocuje firewall příchozí pakety. Směrovač se chová jako filtr paketů, když propouští nebo naopak nepropouští, pakety podle určitých pravidel filtrování. Jako zařízení 3. vrstvy směrovač pravidla filtrování používá pro určení, zda povolit nebo zakázat paket na základě zdrojové a cílové IP adresy nebo na základě zdrojového a cílového portu. Tato pravidla jsou definována pomocí seznamu pravidel pro přístup přístupu neboli ACL.



Obrázek 4: vykonávání ACL, zdroj: autor

Sady pravidel můžeme chápat jako řetězce (chain) tvořené jednotlivými články – pravidly. Podle [4] ke zpracování jednotlivých pravidel dochází postupně od prvního k poslednímu, dokud některému z nich příchozí paket nevyhoví. Pokud se tak stane, je vykonána příslušná akce. Tyto akce mohou být tři:

- Accepted – paket je akceptován a je mu tudíž umožněn bezproblémový průchod firewallem;

- Droped – paket je zahozen bez jakéhokoliv upozornění;
- Rejected – paket je odmítnut s tím, že odesilatel je o této činnosti informován.

Pokud paket nevyhoví žádnému pravidlu, mohou nastat dvě možnosti:

- jde-li o vestavěný řetězec, vykoná se výchozí politika daného řetězce:
  - ACCEPT – přijmout;
  - DROP – zahodit;
- jde-li o uživatelský řetězec, je paket vrácen tam, odkud byl do uživatelského řetězce odeslán – RETURN.

ACL využívá modul iptables, což je mocný nástroj ke konfiguraci linuxového firewallu. Základní syntaxe jednotlivých pravidel je podle [5] následující:

```
iptables [tabulka] [akce] [řetězec] [ip_část] [rozšíření] [cíl]
  [cíl_rozšíření]
```

Je nutné si při psaní pravidel uvědomit, že je velmi důležité mít je napsána správně, neboť pokud paket vyhoví prvnímu pravidlu, už nepokračuje dál a ostatní pravidla ignoruje.

Rozebereme si podle [5] jednotlivé části:

[tabulka] - všechny řetězce patří do nějaké tabulky. O jakou tabulku se jedná, se pozná podle přepínače -t, --table. Iptables mají podle [6] tyto tři vestavěné tabulky:

- filter - pokud není použit přepínač -t, je tato použita jako výchozí;
- nat;
- mangle.

[akce] – těch je poměrně velké množství, podrobněji si je probereme dále. Jako jediné jsou označeny velkým písmenem. Mezi nejpoužívanější patří:

- -A, --append - přidá nové pravidlo na konec řetězce;
- -D, --delete - smaže pravidlo;
- -N, --new-chain - vytvoří nový, uživatelem definovaný, řetěz zvoleného jména, které musí být jedinečné;
- -L, --list - vypíše všechna pravidla;
- -F, --flush - vymaže všechna pravidla v řetězci;
- -P, --policy - nastaví výchozí politiku řetězce.

[řetězec] – název řetězce, jehož parametry nastavujeme.

[ip_část] – v této části se nastavuje:

- -p, --protocol - ICMP, UDP, TCP nebo ALL;
- -s, --src, --source/-d, --dst, --destination – zdrojová nebo cílová IP adresa;
- -i, --in-interface/-o, --out-interface – vstupní/výstupní zařízení (síťová karta);
- -g, --goto – přesměruje vyhodnocení pravidla do určeného řetězce.

[rozšíření] – rozšiřují pravidla, pokud nestačí základní části. Pokud není v ip_části použit přepínač -p, je nutno je aktivovat pomocí přepínače:

- -m.

[cíl] – cíle se používají k určení akce, která se vykoná, bude-li paket vyhovovat pravidlům, a také ke specifikaci politiky řetězce. Čtyři cíle jsou v iptables vestavěné, ostatní je možné doplnit pomocí rozšiřujících modulů. Vestavěné cíle jsou:

- ACCEPT;
- DROP;
- QUEUE;
- RETURN.

Cíl se určuje pomocí přepínače:

- -j, --jump.

[cíl_rozšíření] – umožňují rozšířit původní cíle o další. Tato rozšíření jsou například:

- LOG;
- SNAT/DNAT;
- REDIRECT;
- MANGLE;
- MASQUERADE.

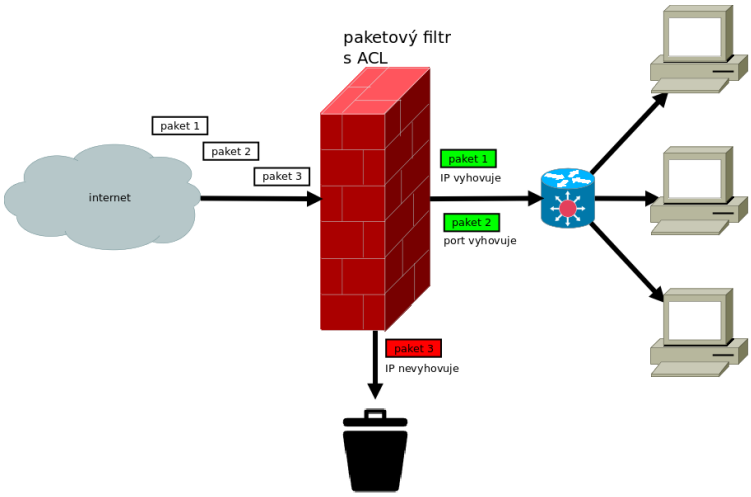
Příklad pravidla, které předá příchozí paket jdoucí na port 80 (HTTP) na vnitřní server 192.168.1.3 na port 8080, by podle [5] vypadal takto:

```
iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 80 -j DNAT --to-
  destination 192.168.1.3:8080
```

2.2 Paketový filtr

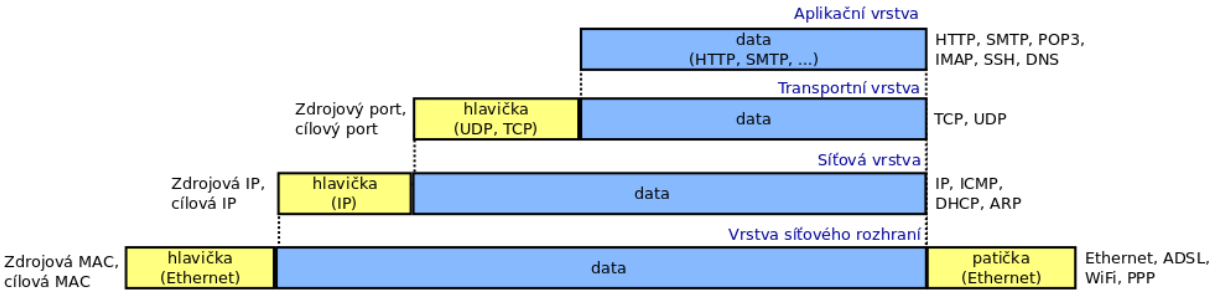
Paketový firewall je nejstarším a nejjednodušším typem firewallu pracujícím na třetí vrstvě ISO/OSI modelu sítě[6]. Na principu tohoto filtru fungují první verze netfilter v OS Linux. Tento typ firewallu zpracovává síťový provoz na základě IP adresy (3. vrstva) nebo na základě čísla portu (4. vrstva). Ty pak pouze vyhodnocuje na základě uložených pravidel – Access Control List. Při kontrole procházejícího paketu mohou pro něho právě podle těchto pravidel nastat výše zmíněné tři možnosti:

- Accepted;
- Droped;
- Rejected.



Obrázek 5: princip paketového filtru, zdroj: autor

Jedná se tak vlastně o směrovač, který má jistou, i když jen jednoduchou rozhodovací pravomoc. Paketový filtr je však ve vyhodnocování průchozích paketů velmi rychlý, neboť neprovádí jejich hlubší analýzu. Musí však zpracovat všechny pakety jeden po druhém. Je tak skoro stejně rychlý jako obyčejný směrovač.



Obrázek 6: zapouzdření dat v síti TCP/IP, upraveno podle: Mudrák, CC

Zároveň využívá málo paměti. Toho je zase dosaženo tím, že paketový firewall neudržuje žádné informace o průchozích paketech a nemusí tak provádět při jejich průchodu zpětnou kontrolu s informací o již dříve prošlých paketech. Tato pravidla uvádějí, která IP adresa smí komunikovat s kterou IP adresou, případně na jakém čísle portu. Každý paket, který přichází do paketového filtru je považován za nový, neznámý bez ohledu na to, zda již dříve filtrem procházel nebo nikoli.

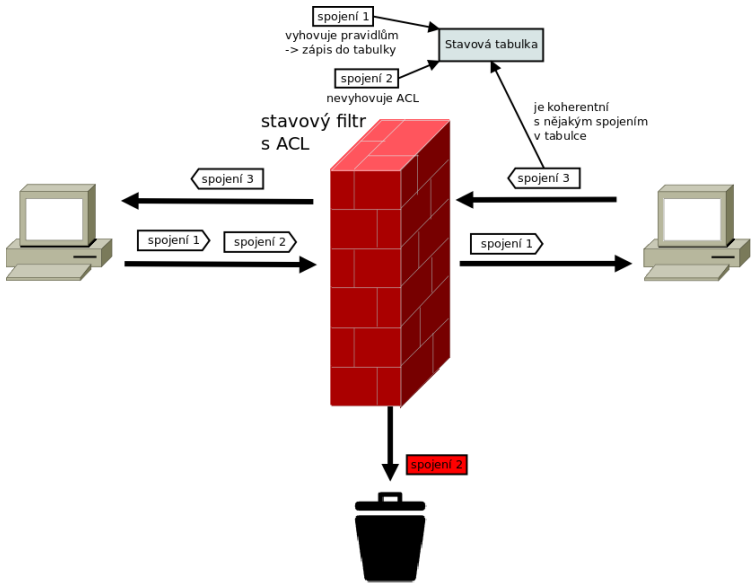
Velkou nevýhodou tohoto filtru je velmi omezená možnost bránit se podvrženým paketům. Tyto pakety v sobě nesou falešné informace o výchozích či cílových adresách nebo portech. Firewall není odolný proti útokům na vyšších vrstvách ISO/OSI modelu a navíc vyžaduje velmi kvalitní sadu pravidel – ACL.

2.3 Stavový filtr

Bezstavové paketové filtry jsou jednodušší na implementaci, ale složitější na nastavení a nakonec mnohem méně bezpečné než paketové filtry, které udržují stav uskutečněných spojení – stavové filtry. Ty pracují na čtvrté vrstvě ISO/OSI modelu jak uvádí [6].

Postup práce stavového filtru je následující:

1. Pro každé nové příchozí spojení kontroluje firewall (stejně jako paketový filtr) ACL pravidla, zda je pro toto spojení povolen průchod.



Obrázek 7: princip stavového filtru, zdroj: autor

2. Pro akceptovaná spojení je učiněn zápis do stavové tabulky, která obsahuje parametry, jako jsou zdrojová/cílová IP adresa a port, relevantní TCP příznaky a SEQ a ACK hodnoty.

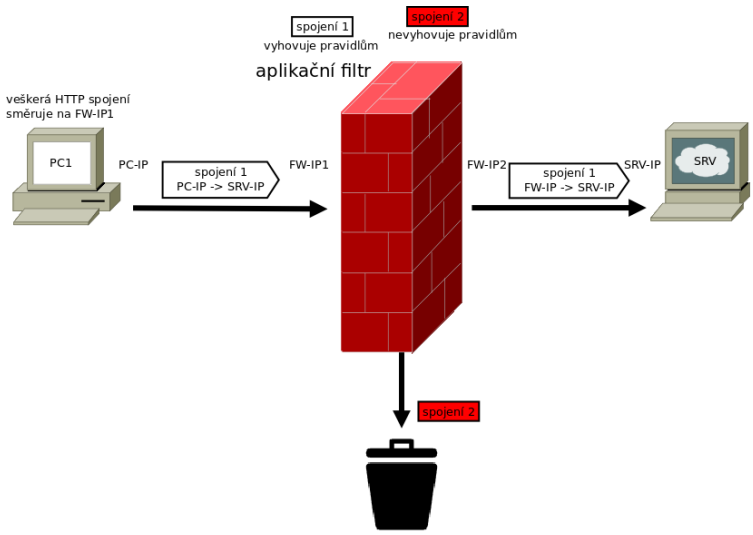
3. Vracející se pakety jsou porovnávány se zápisy ve stavové tabulce a průchod je povolen pouze těm, které jsou koherentní s TCP definicí cílového počítače. Po ukončení spojení je zápis z tabulky odstraněn.

Tímto dochází k výraznému urychlení vyhodnocovacího procesu v porovnání s předchozím typem. Nicméně tím, že si firewall uchovává informace o uskutečněných spojeních, má i vyšší nároky na paměť.

Pomocí firewallu je možné definovat pravidla pro odchozí pakety a zakázat všechny příchozí pakety, kromě vracejících se paketů již navázaných spojení z místní sítě. Toho je využíváno například při FTP spojení, kdy řídící komunikace běží na portu 21 a vlastní datová komunikace pak na portu 20.

2.4 Aplikační filtr

Do skupiny aplikačních filtrů patří i tzv. aplikační proxy firewally. Ty pracují na aplikační sedmé vrstvě ISO/OSI modelu. To však pouze v případech známých protokolů, např. HTTP či FTP. Tyto firewally umějí rovněž blokovat i jednotlivé příkazy FTP protokolu, jako jsou PUT a GET. Jak uvádí [6], je typickým představitelem tohoto filtru aplikace Squid, což je svobodný kešovací proxy server, který primárně pracuje s protokoly HTTP a FTP.



Obrázek 8: princip aplikačního filtru, zdroj: autor

Klasický aplikační firewall pracuje rovněž na sedmé vrstvě ISO/OSI modelu. Tím získává o toku dat detailní informaci. Rozhoduje se na základě MAC nebo IP adresy, portů a protokolů. Pouze tento typ filtru umí spravovat různé P2P protokoly, které používají počítačové hry nebo IM aplikace. Komunikace přes aplikační firewall je realizována pomocí dvou spojení a to takto:

1. klient se připojí na aplikační bránu;

2. filtr aplikační brány spojení zpracuje;
3. dle klientského požadavku otevře spojení na server pod „svým jménem“;
4. získaná data předá opět původnímu klientovi.

Výše uvedeným postupem je dosaženo toho, že server nezná zdrojovou adresu původního klienta, protože jako zdrojová adresa požadavku je uvedena vnější adresa aplikační brány. Údaje o chráněné síti jsou tak skryty.

Zvláště první varianta je využívána k tvorbě detailních logů či ke kešování obsahu. Typickými operacemi, které tyto filtry zabezpečují, jsou např. filtrování nežádoucích URL nebo bezpečnostní omezení protokolu FTP. Tyto filtry se velice často používají ve spojení s paketovým filtrem k zabránění útoku typu Denial of Service (DoS).

Z předchozího textu vyplývá, že udaný firewall je výpočetně nejnáročnější. Mezi jeho další nevýhody patří netransparentnost, neboť každá aplikace musí podporovat připojení pomocí proxy a být správně nastavena.

3 Principy ochrany sítě s využitím firewallu

Jak již bylo řečeno v úvodu, firewally můžeme rozdělit dle následujících kritérií:

- softwarový;
- hardwarový;

přičemž obsahem této práce je pouze softwarový firewall pracující pod operačním systémem Linux.

Firewall je zařízení sloužící k ochraně vnitřní sítě před potenciálním nebezpečím přicházejícím z vnější sítě. Umožňuje blokovat neoprávněný přístup, ale také povolit pouze určené služby. Jedná se tedy o jakýsi kontrolní bod přes který prochází veškerá síťová komunikace a na kterém jsou definována pravidla této komunikace. Na firewallu většinou platí, nebo by mělo platit pravidlo, co není povoleno, je zakázáno [6].

Ochrana počítačové sítě prostřednictvím firewallu tak spočívá především v postavení zábrany do cesty útočníkovi od jeho počítače k naší chráněné síti. Firewall k tomuto účelu používá sady pravidel, pomocí nichž určuje, které pakety budou moci projít a které nikoli. V případě aplikačního filtru je ochrana rozšířena i na jednotlivé aplikace, kterým je zakázáno, či povoleno připojení k internetu.

Pomocí strategie konfigurací těchto pravidel můžeme rozeznávat dva základní typy firewallů:

1. paranoidní - vše je zakázané, povolené jsou jen vybrané funkce (tato metoda je považována za lepší), tvorba firewallu pomocí této metody by měla být preferována;
2. benevolentní – zde je naopak vše povoleno, zakázány jsou pouze vybrané funkce.

V případě paranoidního nastavení by počáteční řetězec v iptables měl mít tvar:

```
iptables -P INPUT DROP

iptables -P OUTPUT DROP

iptables -P FORWARD DROP
```

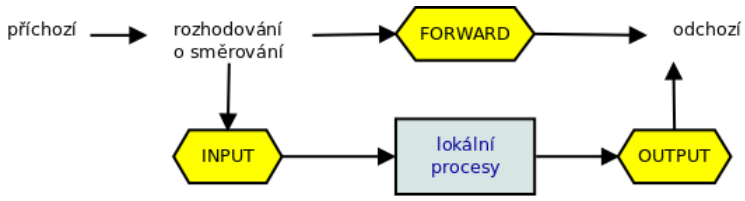
Pravidla používaná firewally ke zpracovávání paketů využívají některé jejich vlastnosti, jako například:

- protokol TCP nebo UDP;
- zdrojovou a cílovou IP adesu;
- zdrojový a cílový port;

- na aplikační vrstvě obsah paketu.

Základ netfilteru [7] tvoří vestavěné řetězce. Jsou to tyto tři:

- INPUT;
- OUTPUT;
- FORWARD.



Obrázek 9: základní řetězce v netfilteru, zdroj: autor

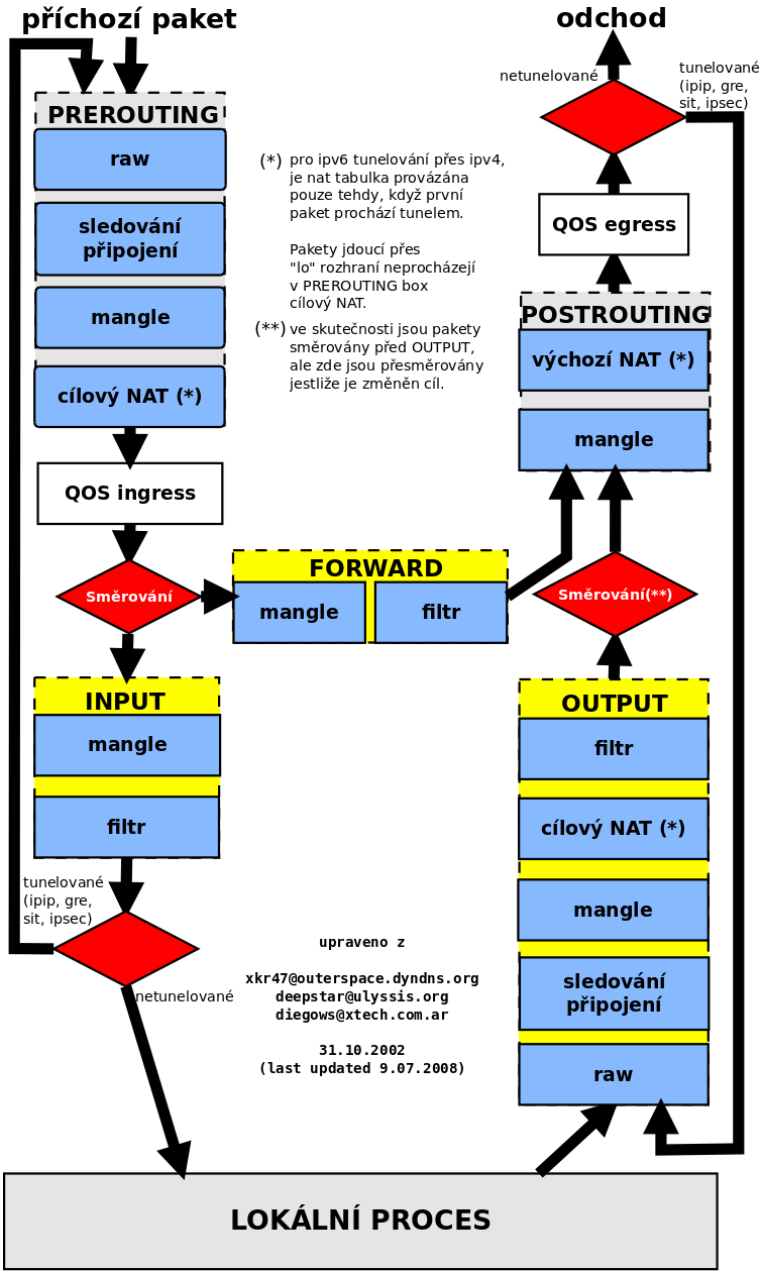
Tři žluté šestiúhelníky na obrázku 9 představují výše uvedené řetězce. Pokud paket dojde k některému z nich, je o jeho osudu rozhodováno na základě pravidel definovaných v tomto řetězci. Pokud pravidlo říká DROP, je tento paket okamžitě zahozen. Pokud ale pravidlo říká ACCEPT, pokračuje paket v další cestě diagramem.

Pojmem řetězec označujeme kontrolní seznam pravidel. Každé pravidlo říká, že „pokud záhlaví paketu vypadá takto, pak tady je to, co dělat s paketem". V případě, že pravidlo neodpovídá záhlaví paketu, pak je paket porovnáván s dalším pravidlem v řetězci. A konečně, pokud nejsou žádná další pravidla k porovnávání, pak se jádro rozhodne podle politiky řetězce, co s paketem dělat. V zabezpečených systémech tato politika obvykle říká, aby jádro zahodilo paket (provedlo DROP).

1. Jakmile paket vstoupí (řekněme skrz síťovou kartu), jádro se nejprve podívá na cílový port. Toto je nazýváno „routing – směrování“.
2. Je-li určen pro tento box, projde paket dolů diagramem do řetězce INPUT. Jestliže vyhoví, přijmou ho všechny čekající procesy.
3. V opačném případě, tedy pokud jádro nemá povoleno předávání nebo neví jak paket předat, je paket zrušen. Pokud je předávání povoleno a paket je určen pro jiné síťové rozhraní (pokud jiné rozhraní máme), pak paket pokračuje na našem diagramu směrem doprava do řetězce FORWARD. Pokud je přijat, je paket odeslán.
4. Nakonec program běžící v boxu může posílat síťové pakety. Tyto pakety procházejí skrz řetězec OUTPUT okamžitě: jestliže je to vyhodnotí jako ACCEPT, pak paket pokračuje ven na rozhraní, pro které je určen.

Firewall tak vlastně provádí správu a řízení síťového připojení počítače. Bez ohledu na typ poskytují firewally podle [8] následující služby:

- Předávání síťového provozu - mnoho firewallů se chová jako směrovače, aby spolu mohly komunikovat odlišné sítě (chování lze dosáhnout např. pomocí *iptables*, programu CLI, který spolupracuje s *netfilter* v jádře systému Linux).
- Vymezení sítě - použití firewallu je způsobem, jak vytvořit hranici mezi sítěmi. Tato hranice pomáhá řídit a organizovat síťový provoz.
- Ochrana před skenováním sítě, útoky DoS a sniffingem - firewall monitoruje odchozí a příchozí provoz a umožňuje vybraný provoz omezovat.
- Filtrování portů a adres IP - jedná se o schopnost přijmout, nebo odmítnout provoz na základě adresy IP a čísla portu.



Obrázek 10: průchod paketu, upraveno

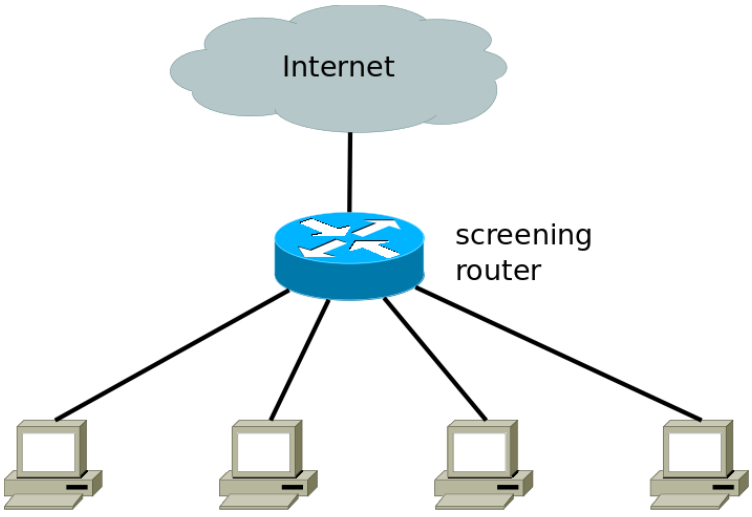
- Filtrování obsahu - server proxy jakožto typ firewallu umožňuje inspekci např. URL. Vhodnou konfigurací lze dosáhnout blokování obsahu, který administrátor považuje za nevhodný.
- Autentizace a šifrování - firewall může mít schopnost ověřit uživatele a šifrovat spojení mezi sebou a firewallem v jiné síti.
- Tvorba logů - firewall umožňuje zpětné zkoumání detailů provozu, který jím prošel.

Firewally nestojí v síti osamoceny a jakožto pomyslné brány do sítě značně ovlivňují podle [9] podobu síťové topologie. Způsobů, jak firewally v síti nasadit, je přitom více a ten pravý je potřeba vždy volit podle konkrétních požadavků.

3.1 Architektura Single-box

Základní a vlastně i nejjednodušší variantou nasazení firewallu v síti je taková architektura, kdy je firewall nasazen pouze v jednom počítači a tento se pak nachází na okraji chráněné sítě. Toto řešení má jisté nevýhody, jako např. to, že veškerá bezpečnostní opatření jsou soustředěna do jednoho místa. Z toho plyne ale i výhoda, kdy je třeba konfigurovat a tím pádem i spravovat pouze jediný objekt. Toto řešení tak upřednostňuje praktickou stránku věci před bezpečností a je tak vhodné spíše pro menší sítě.

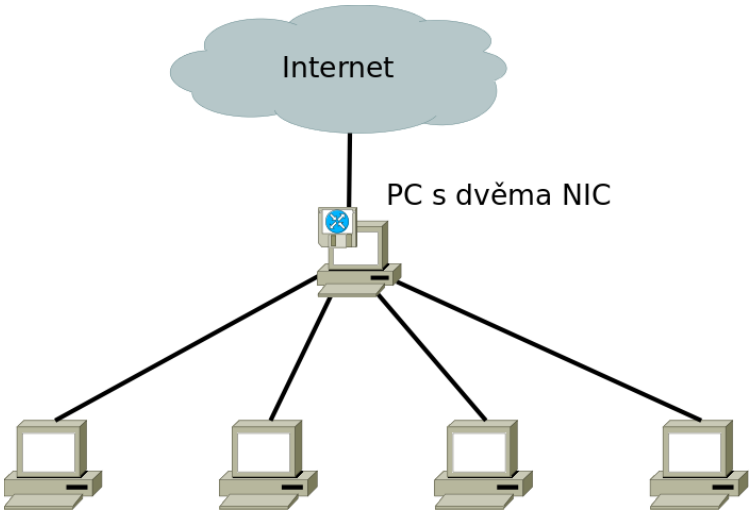
Při implementaci této architektury na celou síť může jako firewall posloužit základní paketový filtr, tzv. *screening router*. Jelikož je router vyžadován již pro připojení do internetu,



Obrázek 11: screening router, upraveno podle [10]

jedná se tak o řešení, které upřednostňující cenové hledisko. Nicméně princip paketového filtru může způsobovat problémy s provozem některých služeb. K eliminaci těchto problémů, lze použít počítač s alespoň dvěma síťovými rozhraními, tzv. *dual-homed hostitele*. Na něm je vypnuta funkce směrování paketů, aby fungoval jako firewall. Zároveň tento hostitel funguje

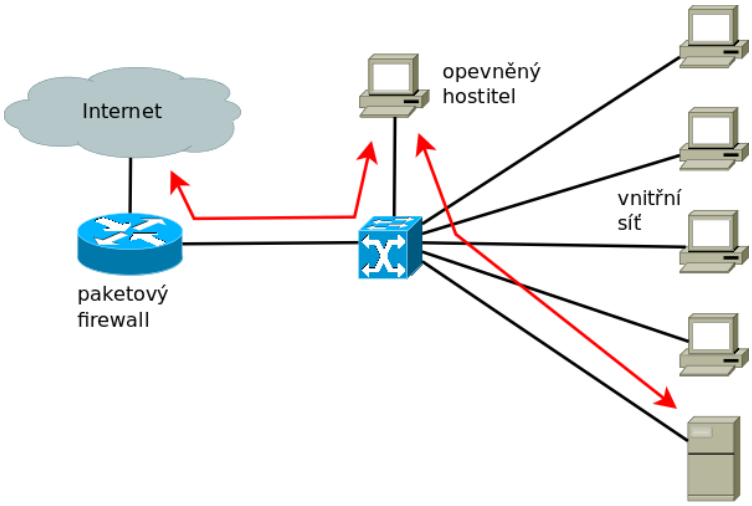
jako prostředník, přes nějž probíhá komunikace systémů z různých sítí (vnitřní/vnější). Toto řešení již odpovídá principu aplikační brány. Za cenu ztráty jistého výkonu tak ale získáváme vysokou úroveň kontroly nad provozem z a do chráněné sítě. Obě tato zařízení lze spojit v tzv. *multi-purpose box*.



Obrázek 12: dual-homed hostitel, upraveno podle [10]

3.2 Architektura Screened Host

Architektura *Screened Host* poskytuje služby z hostitele, který je součástí pouze vnitřní sítě, na rozdíl od použití *dual-homed hostitele*, připojeného k vnitřní a vnější síti. Tomuto



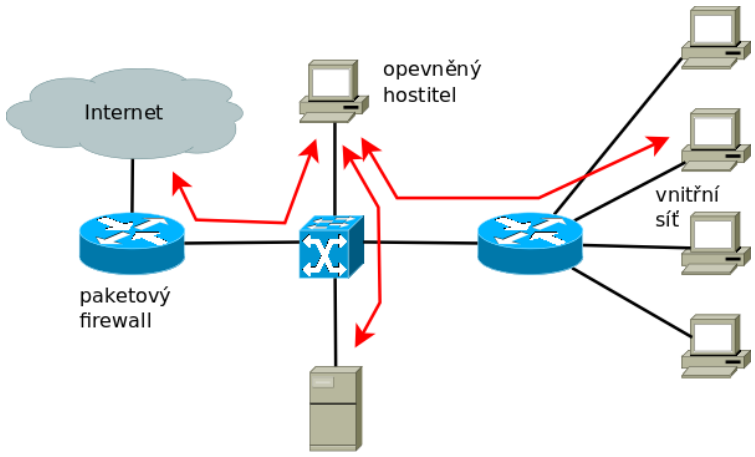
Obrázek 13: screened host, upraveno podle [10]

hostiteli potom říkáme opevněný. Směrovač, stojící na rozhraní vnitřní a vnější sítě, poskytuje základní filtrování paketů, kdy všem vnějším systémům povoluje otevírat spojení pouze k opevněnému hostiteli. Systémy z vnitřní sítě zase mohou komunikovat směrem ven pouze s pomocí opevněného hostitele jakožto prostředníka. Z tohoto důvodu je zvláštní postavení opevněného hostitele třeba brát v úvahu při řešení otázky bezpečnosti a provozovat

na něm jen nezbytné služby. A jelikož se tak stává úzkým hrdlem bezpečnosti, je třeba jej autorizovat pouze k nezbytným úkonům. Pokud se podaří zkompromitovat opevněného hostitele je zkompromitována celá síť. I tak je obecně architektura *Screened Host* považována za vyšší úroveň zabezpečení oproti architektuře *Single-box*.

3.3 Architektura Screened Subnet

Vytvořením podsítě k architektuře *Screened Host* se přidává další bezpečnostní vrstva, jakýsi nárazník, který dále odděluje vnější a vnitřní síť. Této podsíti se říká demilitarizovaná zóna (DMZ) a slouží k umístění počítačů, které mají být přístupné z internetu i z intranetu.



Obrázek 14: screened subnet, upraveno podle [10]

Na firewallu se pak definují pravidla, s kterými počítači umístěnými v DMZ smějí komunikovat počítače ve vnitřní síti a na jakých portech. Další pravidla pak definují, které porty jsou otevřeny z internetu do DMZ apod. Pomocí architektury *Screened Subnet* se tak odstraňuje problém selhání jediného bodu architektury *Screened Host* a to pomocí izolace opevněného hostitele (hostitelů) v podsíti. Aplikační brána, která je umístěna v DMZ, pak kontroluje všechna spojení jdoucí přes tuto zónu. Router na hranici vnější síť/DMZ musí ale veškeré pakety směrovat na opevněného hostitele. Je-li úspěšně kompromitována aplikační brány z vnější sítě není kompromitována vnitřní síť. Pro úspěšné napadení vnitřní sítě by útočník musel úspěšně obejít nejen zařízení na hranici vnější síť/DMZ, ale i další zařízení stojící na hranici DMZ/vnitřní sítí.

3.4 Architektura Split-Screened Subnet

V této architektuře, která vychází z architektury *Screened Subnet*, je demilitarizovaná zóna pro citlivější kontrolu datových toků dále rozdělena *dual-homed hostitelem*. Poskytuje tím lepší kontrolu než jen pouhé filtrování paketů. Lze tak docílit kvalitní vícevrstvé ochrany nebo např. zajistit administrativní přístup (tím, že je administrativní provoz separován, je možné kromě bezpečnosti zlepšit i výkon) k počítačům, které provozují služby dostupné z internetu.

3.5 Architektura Independent Screened Subnets

Z architektury *Screened Subnet* rovněž vychází architektura *Independent Screened Subnet* a to přidáním dalších směrovačů na hranách sítě. Tím je vytvořeno více nezávislých perimetrů a dosaženo redundance. Rovněž je možné oddělit odchozí spojení/služby (umožňují uživatelům připojit se k internetu, jako např. webová proxy) a příchozí spojení/služby (např. z Internetu přístupný webový server) a dosáhnout velmi silného zabezpečení.

Popis jednotlivých architektur vychází z [9] a [10].

3.6 Nedostatky firewallu

Je ovšem třeba také zmínit, proti čemu firewall nebrání. Jsou to podle [11] především:

- viry, malware apod;
- vnitřní ohrožení a útoky (nespokojení pracovníci, špatná bezpečnostní politika …);
- útoky, které neprocházejí firewallem, ale které ho obcházejí (osobní modemy, neautorizované bezdrátové připojení...);
    - Neodpovědný uživatel může buď z neznalosti, nebo záměrně vytvořit alternativní komunikační cestu do vnější sítě. Zpravidla ji realizuje použitím modemového spojení nebo vytvořením tunelu v legální službě a takto vytvořená cesta je zcela mimo možnost kontroly firewallu, tudíž nechráněna.
- útoky přes služby, které jsou na firewallu povoleny (HTTP, SMTP...);
- neznámé hrozby;
    - Proto se musí klást důraz na průběžnou aktualizace instalací programového vybavení.
- odposlech, modifikace nebo zničení dat při přenosu po síti.

3.7 Analýza možných útoků

Až dosud bylo zatím zmiňováno, jak firewall funguje, před čím chrání a před čím nikoliv. V této kapitole bude popsáno, jaké konkrétní typy útoků na firewall jsou možné a jaké jsou schopné napáchat případné škody. Nebudou zde uvedeny všechny možné útoky, ale pouze ty nejčastější a nejzávažnější[12][13]. Tyto útoky mají především:

- zajistit komunikaci s počítači ve firemní síti nebo zajistit přístup ke službám firemní sítě;
- zjistit komunikaci mezi počítači ve firemní síti;
- spustit kód (program) na pracovní stanici nebo serveru v napadené firemní síti.

3.7.1 Sniffing

Jedná se o odposlouchávání síťového provozu a zjišťování, jaká data jsou v síti přenášena[14]. Nejedná se v pravém smyslu slova o útok, ale spíše o shromažďování informací potřebných pro přípravu útoku. Rozhodující pro to, jaké informace budou získány, je umístění snifferu v síti. Zejména v přepínaných sítích, kde je společný segment minimalizován, je použití snifferu problematické, neboť většina informací jde mimo něj.

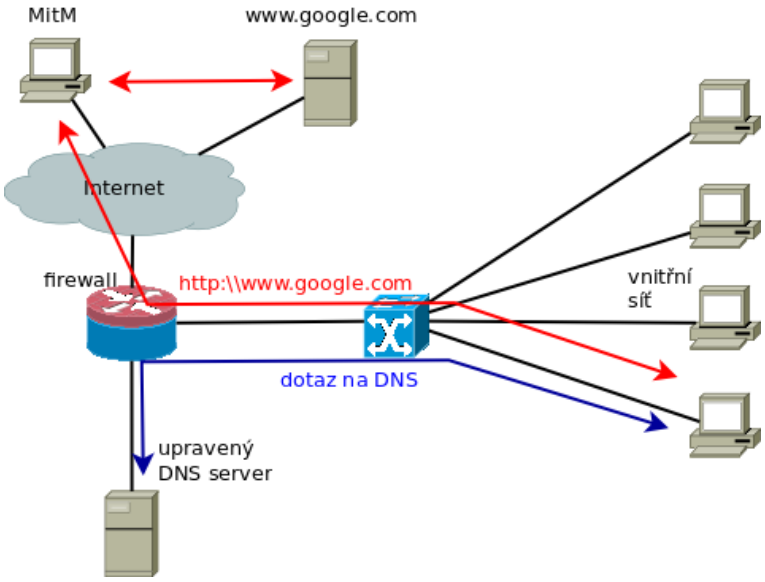
Vlastní práce snifferu je jednoduchá:

- přepne síťové rozhraní do tzv. promiskuitního módu;
- přijímá všechny pakety, které se na síti pohybují.

Tyto pakety jsou zaznamenány a vyhodnocovány, a to především – typ protokolu, IP adresy, MAC adresy, nastavení příznaku. Součástí analýzy je také vydělení datové části, která obsahuje vlastní přenášenou zprávu. Tak je možno odposlechnout komunikaci v síti, zachytit otevřeně přenášená hesla nebo jiné citlivé údaje.

3.7.2 Man-In-the-Middle - MItM

Jedná se o druh sniffingu [14], kdy dochází k přesměrování síťového provozu přes počítač útočníka. Tím je usnadněn odposlech síťové komunikace a navíc je možné procházející data upravovat, aniž by příjemce cokoliv poznal. Rozeznáváme tak útok pasivní, nebo aktivní. V dnešní době navíc není třeba, aby útočník byl fyzicky uvnitř sítě, neboť je možné provoz přes útočníkův počítač vhodně přesměrovat. Častokrát je tento útok realizován pomocí upravené DNS tabulky v DNS serveru tak, aby síťový provoz procházel k původnímu cíli přes útočníkův počítač. Tomuto typu útoku se říká DNS spoofing.



Obrázek 15: útok MItM s pomocí upraveného DNS serveru, upraveno podle [13]

Na podobném principu pracuje i útok DHCP spoofing, který je realizován pomocí DHCP serveru. Jelikož dotaz na zjištění DHCP serveru v síti je broadcastový, obdrží ho všechny počítače v síti. Pak už stačí, aby útočníkův počítač odpověděl jako první, a vnutil tak oběti vlastní nastavení sítě, kde veškerá komunikace prochází přes jeho počítač a odposlech je velice jednoduchý.

Dalším typem MItM útoku jsou útoky ARP cash poisoning a Port stealing, které využívají v přepínaných sítích aktivní prvek, switch. Pomocí úpravy CAM tabulky a podvržené MAC adresy si útočník přesměruje veškerou síťovou komunikaci opět přes svůj počítač.

3.7.3 Denial of Service - DoS

Denial of Service (odepření služby) jsou podle [14] útoky, při nichž nedochází přímo k neoprávněnému přístupu do sítě oběti nebo zachytávání jeho paketů. Jak uvádí [6], jde o útoky znemožňující přístup ostatních uživatelů k různým službám nebo stránkám oběti. Toho dociluje útočník pomocí zahlcení serverů jím vysílaným velkým množstvím dotazů. Často se používají po úspěšných útocích jiného typu jako prostředek pro zahlazení stop, nebo po neúspěšném útoku. To proto, aby oběti vznikla alespoň nějaká škoda. Tyto útoky jsou prováděny buď na třetí síťové vrstvě ISO/OSI modelu, kdy je znemožněno vlastní síťové spojení, nebo na sedmé aplikační vrstvě, kdy je daná aplikace přetížena nebo přestane fungovat zcela.

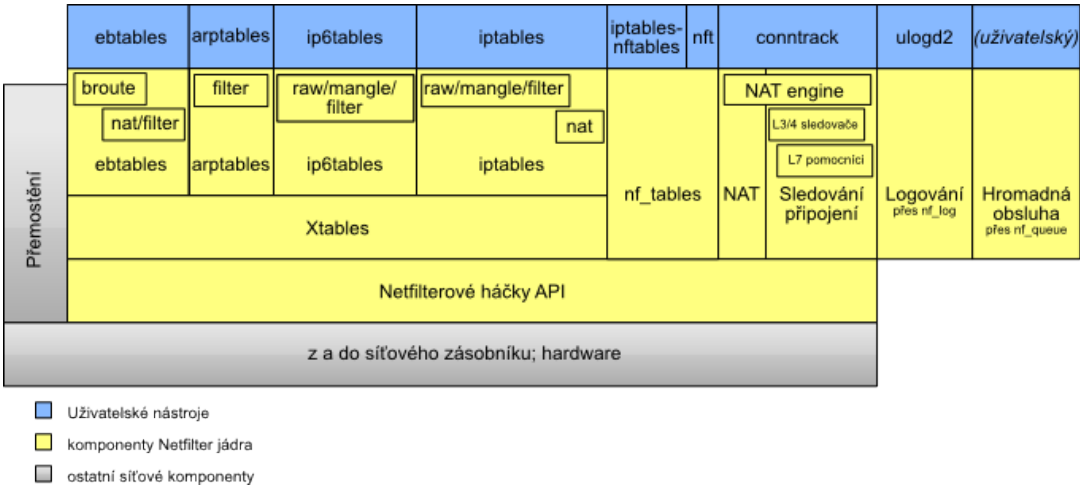
Typů útoku Dos je celá řada a každý z nich je podle [14] něčím specifický:

- ICMP floods – tento typ útoku využívá specifickou vlastnost všesměrového vysílání. Tak je možné zahltit počítač oběti například neustálým odesíláním příkazu ping.
- Peer-to-peer útok – tento útok je realizovatelný v P2P sítích pomocí chyby v jednotlivých klientech, a to odpojením od serveru a pokusem připojit se k serveru novému, v tomto případě počítači oběti.
- Distributed DoS – distribuovaný DoS útok je typický tím, že k útoku na počítač je využito většího množství počítačů, a to buď s vědomím vlastníků, ale většinou bez jejich vědomí. To se děje zpravidla pomocí malwaru, kterým je počítač infikován a který spustí útok přesně v zadaný čas, nebo pomocí trojského koně, který běží na pozadí a čeká na útočníkův povel. Takovým počítačům se říká zombie.
- Nuke – při tomto útoku je cílový počítač zahlcen velkým množstvím ICMP paketů, které mají špatný CRC součet. Oběť je pak opravou těchto paketů tak zaneprázdněna, že přestane reagovat na další požadavky.
- LAND útok – k tomuto útok se využívá zmatení operačního systému. Na počítač oběti je zaslána speciální zpráva, která má stejnou cílovou i zdrojovou adresu, a to adresu oběti. Některé operační systémy si pak začnou odpovídat samy sobě a tím se zahltí.

- Podvržený distribuovaný DoS útok (DRDoS) – jedná se o podobný útok jako ICMP floods, kdy je síť zahlcena velkým množstvím dotazů, kde je jako jejich původce uveden počítač oběti, který je pak zahlcen velkým množstvím odpovědí na tyto dotazy.
- Slowloris – při tomto útoku není počítač oběti zahlcen, ale naopak je na něm udržován v otevřeném stavu port pomocí pomalého posílání částí paketu. Server tak stále čeká na celý dotaz, na který má odpovědět.
- Neúmyslný útok – tento útok je prováděn převážně na jinak málo navštěvované webové servery, kdy se na ně z nějakého důvodu (např. reklama) připojí větší množství klientů, než na kolik jsou dimenzovány, a tím se zahltí.

4 Implementace firewallů v GNU/Linux

Podle [7] Linux dokázal filtrovat síťové pakety již od verze jádra 1.1 v roce 1994. Tento systém byl založen na *ipfw* původem z operačního systému BSD. Poté byl ve starších jádrech jako základní firewall využíván *ipfwadm,* a to do verze jádra 2.0. Ten byl v jádře 2.1.102 nahrazen *ipchains*. Oba nástroje vycházely rovněž z operačního systému BSD. V listopadu roku 1999 se tzv. Coreteam rozhodl dále nevyvíjet, a nechat tak zemřít nástroj *ipnatctl* (dřívější nástroj na konfiguraci NAT v *netfilteru*) a zároveň položil základ pro moduly *iptable*_nat, *iptable*_filter a *iptable*_mangle. Ty začaly být okamžitě mohutně



Obrázek 16: komponenty Netfilteru, upraveno podle: Jan Engelhardt, 2014

implementovány. Na jaře 2000 jsou *iptables* již představovány na linuxových konferencích jako přídavný modul pro oficiální linuxový strom. Po rozšíření Coreteamu byly započaty práce na novém *netfilteru* pro začlenění do nového jádra 2.4. Tím se *netfilter* stal definitivně nástupcem ipchains.

Uvedený firewall kombinuje paketové filtry a NAT/NAPT (zejména tři konkrétní druhy NAT, tzv. maškarádu, přesměrování portů a přesměrování adres). *Netfilter* je tak sada háčků v linuxovém jádře, která umožňuje modulům jádra registrovat zpětná volání do síťového zásobníku. Registrovaná volání jsou pak vyvolána zpět zevnitř síťového zásobníku pro každý procházející paket. Hlavními vlastnostmi *netfiltru* jsou:

- bezstavové filtrování paketů (IPv4 a Ipv6);
- stavové filtrování paketů (IPv4 a Ipv6);
- všechny druhy překladu síťových adres a portů, např. NAT / NAPT (IPv4 a Ipv6);
- flexibilní a rozšiřitelné infrastruktury;
- vícenásobné vrstvy API pro rozšíření třetích stran.

Pomocí těchto firewallů je možné:

- vytvořit internetový firewall na základě bezstavového i stavového filtrování paketů;
- nasadit snadno dostupné bezstavové a stavové firewallové clustery;
- použít NAT a maškarádu pro sdílení přístupu k internetu, pokud nemáme dostatek veřejných IP adres;
- použít NAT k vytvoření transparentního proxy serveru;
- pomoci TC a iproute2 systémů vytvořit sofistikované QoS a routery s nastavenou politikou směrovacích pravidel, tzv. PBR routery;
- provést další manipulaci s pakety (mandlování), například změnu TOS / DSCP / ECN bitů IP hlavičce.

Netfilter kombinuje tři síťové funkce (forwarding, filtrování, NAT) do jednoho pravidla nastavitelného pomocí *iptables*. Série příkazů, které se používají pro nastavení *iptables,* může být uložena do skriptu shellu pro snadné použití. Existují ovšem i GUI nástavby pro iptables, jako např. Firestarter, Gufw apod.

Iptables jsou podle [15] rozděleny do tří, podle [16] do čtyř nezávislých tabulek:

- filter;
- nat;
- mangle;
- raw.

Tabulky, které jsou v *iptables* konkrétně přítomné, záleží na konfiguraci jádra a na nastavení jádrových modulů. S kterou tabulkou se pracuje, je nutno specifikovat přepínačem *-t, –table jméno-tabulky*. Pokud tabulka není specifikována, použije se výchozí tabulka - filter.

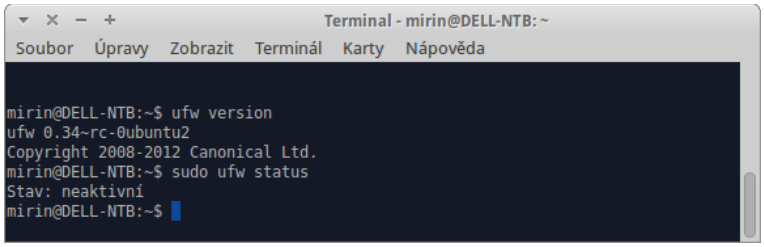
Popis tabulek, jak uvádí [16]:

- filter - výchozí tabulka vhodná pro základní filtrování, logování, počítání… Obsahuje tři vestavěné řetězy pravidel:
  - INPUT - pro pakety přicházejících do systému;
  - OUTPUT - pro pakety odcházející ze systému;
  - FORWARD - pro předávané (routované) pakety;

- nat - používá se pro pro překlad adres (NAT), maškarády, přesměrovávání portů (port forwarding…). Používá se pouze pro první spojení paketu. Má vestavěné také tři řetězy pravidel:
  - PREROUTING - pro alternování příchozích paketů;
  - POSTROUTING - pro alternování odchozích paketu;
  - OUTPUT - pro alternování lokálně generovaných paketů;
- mangle - vhodná pro vychytávky související s alternováním paketů;
- raw - slouží k nastavování výjimek.

4.1 Ubuntu

Výchozí konfigurační nástroj firewallu *netfilter* v současné distribuci Ubuntu je podle [17] UFW – Uncomplicated Firewall. UFW je vyvinut tak, aby zjednodušil konfiguraci firewall



Obrázek 17: výchozí stav ufw, zdroj: autor

přes *iptables*. Poskytuje uživatelsky přívětivý způsob, jak vytvořit IPv4 nebo IPv6 firewall na hostitelském počítači. Ve výchozím nastavení je však UFW zakázáno. Balíky firewallu, které jsou součástí nové instalace operačního systému jsou:

- libnetfilter-conntract3 – uživatelská knihovna poskytující programovací rozhraní;
- libnfnetlink0 – nízkoúrovňová knihovna související s komunikací jádro/uživatelský prostor;
- libxtables10 – uživatelský interface k netfilterovému frameworku *xtables;*
- ufw – front-end pro *iptables*, pomocí kterého lze *iptables* spravovat jednodušeji.

UFW není určen k tomu, aby poskytl úplnou funkčnost firewallu prostřednictvím příkazové řádky, ale místo toho poskytuje snadný způsob, jak přidat nebo odebrat jednoduché pravidlo. Toho se v současné době používá hlavně pro firewally na hostitelských počítačích.

Základní syntax UFW je podle [17]:

- povolení/zakázání UFW:

```
sudo ufw enable/disable
```

- povolení spojení:

```
sudo ufw allow <port>/<optional: protocol>
```

- zakázání spojení:

```
sudo ufw deny <port>/<optional: protocol>
```

- kontrola stavu:

```
sudo ufw status verbose
```

- vymazání existujícího pravidla – uvedení *delete* před pravidlo.

UFW může povolovat a zakazovat služby i podle názvu. K tomu slouží soubor /etc/services, který obsahuje seznam těchto služeb. Příkazy by potom vypadaly takto:

- povolení nebo zakázání služby:

```
sudo ufw allow/deny <service name>
```

Dalším zajímavým příkazem je příkaz:

- zobrazení výsledku pravidla bez jeho provedení:

```
sudo ufw --dry-run allow <port>/<optional: protocol>
```

Aplikace, které ke své činnosti potřebují mít otevřené některé porty, mohou obsahovat UFW profil, který tyto porty popisuje. Profily jsou uloženy v /etc/ufw/applications.d a mohou být editovány v případě, že je potřeba tyto porty změnit.

- Vypsání, která aplikace má instalován UFW profil:

```
sudo ufw app list
```

- Vypsání, které porty a protokoly má daná aplikace povoleny:

```
sudo ufw app info <aplikace>
```

- Povolení provozu na portu pomocí aplikačního profilu:

```
sudo ufw allow <aplikace>
```

- Příklad rozšířené syntaxe by vypadal takto:

```
ufw allow from 192.168.0.0/24 to any app Samba
```

UFW dovoluje také podle [17] použití IP maškarády. Ta dovoluje počítačům uvnitř sítě, které tak mají soukromou IP adresu, spojení s internetem pomocí počítače provádějícího tzv. maškarádu. Síťový provoz z vnitřní sítě směřující do internetu musí být při zpáteční cestě přesměrován zpět na počítač, který tento paket vyslal. To je činnost, při které jsou

nahrazovány zdrojové IP adresy paketů počítačů vnitřní sítě za adresu počítače s maškarádou proto, aby mohli být tyto směrovány zpět, neboť adresy vnitřní sítě nejsou z internetu dostupné. K udržení informace o tom, který vracející se paket patří k jakému počítači, který ho zaslal, slouží v linuxu Connection Tracking (conntrack). Odcházející paket proto může být „převlečen“, jako by odcházel z počítače s maškarádou. Celý tento proces označuje Microsoft Sdílení připojení k internetu.

IP maškarády lze dosáhnout pomocí vlastních pravidel. To je možné proto, že je UFW zpětně kompatibilní s *iptables* pravidly. Ty jsou uloženy v souborech /etc/ufw/*.rules. To je ideální místo pro přidávání starších pravidel *iptables* používaných mimo UFW, jako například pravidel pro síťovou bránu nebo most. Pravidla jsou rozdělena do dvou souborů podle toho, mají-li být vykonána před nebo po vlastních pravidlech UFW. Nastavení maškarády je pak následující:

1. v /etc/default/ufw změnit řádek DEFAULT_FORWARD_POLICY na „ACCEPT“:

   ```
   DEFAULT_FORWARD_POLICY="ACCEPT"
   ```

2. odkomentovat v /etc/ufw/sysctl.conf:

   ```
   net/ipv4/ip_forward=1
   ```

3. přidat do /etc/ufw/before.rules za hlavičku pravidla:

   ```
   # nat Table rules

   *nat

   :POSTROUTING ACCEPT [0:0]


   # Forward traffic from eth1 through eth0.

   -A POSTROUTING -s 192.168.0.0/24 -o eth0 -j MASQUERADE


   # don't delete the 'COMMIT' line or these nat table rules won't be
   processed

   COMMIT
   ```

4. restartovat UFW:

   ```
   sudo ufw disable && sudo ufw enable
   ```

Jak uvádí [17], IP maškarádu lze samozřejmě nastavit i pomocí původních pravidel *iptables*. Zde by byl postup následující:

1. podobně jako v případě UFW je prvním krokem povolení přesměrování IPv4 paketů v souboru /etc/sysctl.conf odkomentováním následujícího řádku:

```
net.ipv4.ip_forward=1
```

2. spustíme příkaz sysctl, aby se projevilo nové nastavení v sysctl.conf:

```
sudo sysctl -p
```

3. k zapnutí IP maškarády zadáme jediné pravidlo:

```
sudo iptables -t nat -A POSTROUTING -s 192.168.0.0/16 -o eth0 -j
MASQUERADE
```

4. jelikož při tvorbě firewallu v režimu brány má každý řetězec v tabulce „filter“ nastavenou politiku na DROP nebo REJECT, je nutné pro provoz IP maškarády nastavit řetězec FORWARD následovně:

```
sudo iptables -A FORWARD -s 192.168.0.0/16 -o eth0 -j ACCEPT

sudo iptables -A FORWARD -d 192.168.0.0/16 -m state –state
ESTABLISHED,RELATED -i eth0 -j ACCEPT
```

5. je-li třeba spustit maškarádu při každém startu počítače, vložíme některý z výše uvedených příkazů do souboru /etc/rc.local, např. takto:

```
iptables -t nat -A POSTROUTING -s 192.168.0.0/16 -o eth0 -j
MASQUERADE
```

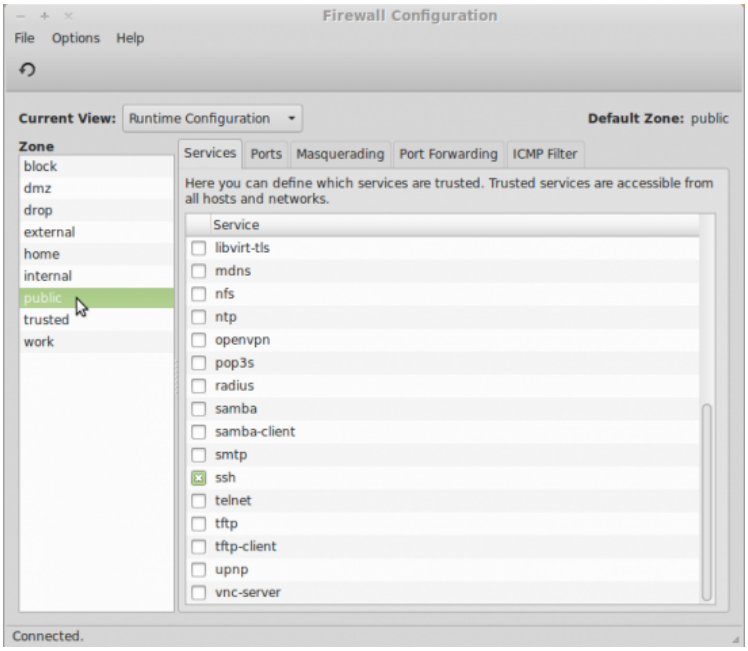
4.2 Fedora

Ke konfiguraci firewallu nepoužívá distribuce Fedora *iptables* ani *ufw*, ale nástroj firewalld, jak uvádí [18]. Ten umožňuje dynamické řízení firewallu s podporou síťových/firewallových zón k definování důvěryhodných síťových spojení nebo rozhraní. Má samozřejmě podporu pro protokol IPv4 a IPv6 pro síťové mosty a má oddělené jednorázové povely od možností trvalých konfigurací. Podporuje i přímé přidávání firewall pravidel pro služby nebo aplikace.

Předchozí model firewallu s *iptables* byl statický a každá změna tak vyžadovala firewall restartovat. To zahrnovalo také odstranění, případně načtení, modulů firewallu *netfilter*, které byly třeba po nové konfiguraci. Odstraňování modulů znamenalo přerušení práce stavového firewallu a opětovné navázání spojení.

Současný firewallový démon však řídí nastavování dynamicky a provádí změny bez nutnosti restartovat celý firewall, proto není nutné znovu načíst všechny firewallové moduly jádra. Nicméně používání firewallového démona vyžaduje, aby veškeré modifikace firewallu byly řešeny pouze přes něho, a byla tak zajištěna synchronizace mezi tímto démonem a firewallem v jádře. Firewallový démon tak nemůže analyzovat pravidla přidaná přes iptables či přes terminál. Démon poskytuje informace o aktuálním aktivním nastavení firewallu přes D-BUS a také změny přijímá prostřednictvím D-BUS pomocí ověřovacích metod PolicyKit.

Co se týče síťových zón, je třeba říci, že síťová zóna definuje míru důvěryhodnosti síťového připojení. Z pohledu vztahů je nutno uvést, že spojení může být pouze částí jedné zóny, ale jedna zóna může být použita pro mnoho spojení.

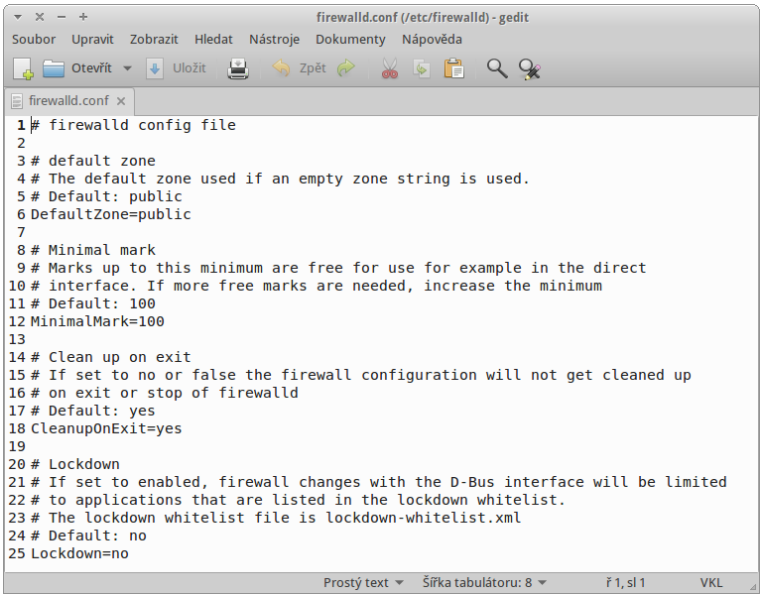


Obrázek 18: firewallD, zdroj [19]

Zde jsou zóny poskytované nástrojem firewalld řazeny podle úrovně výchozí důvěryhodnosti od nedůvěryhodných k důvěryhodným podle [18]:

- drop
  - všechny příchozí pakety jsou bez odpovědi zahozeny, možné jsou pouze odchozí pakety;
- block
  - všechna síťová spojení jsou odmítnuta se zprávou „icmp-host-prohibited“, možná jsou pouze připojení k síti zahájené v rámci tohoto systému;
- public
  - pro použití na veřejných místech, kdy se nedůvěřuje ostatním počítačům v síti a pouze vybraná příchozí připojení jsou akceptována;
- external
  - pro použití na externích sítí se zapnutou maškarádou, kdy se nedůvěřuje ostatním počítačům v síti a pouze vybraná příchozí připojení jsou akceptována, zejména pro routery;

- dmz
  - pro počítače v demilitarizované zóně, které jsou veřejně přístupné s omezeným přístupem k interní síti, pouze vybraná příchozí připojení jsou akceptována;
- work
  - pro počítače ve firemních sítích, kde jsou všechny počítače důvěryhodné a pouze vybraná příchozí spojení jsou akceptována;
- home
  - pro počítače v domácích sítích, kde jsou všechny počítače důvěryhodné a pouze vybraná příchozí spojení jsou akceptována;



Obrázek 19: výchozí nastavení firewallD, zdroj: autor

- internal
  - pro počítače uvnitř vnitřní sítě, kde jsou opět všechny počítače důvěryhodné a pouze vybraná příchozí spojení jsou akceptována;
- trusted
  - všechna síťová spojení jsou akceptována.

Volbu použité zóny volíme podle [18] podle toho, která zóna nejlépe vyhovuje potřebám. K vlastnímu nastavení nebo přidání příslušné zóny se používají některá rozhraní, jež firewallD nabízí. Jsou to grafické rozhraní firewall-config, terminálové rozhraní firewall-cmd nebo D-BUS rozhraní. Rovněž je možné vytvořit nebo nakopírovat soubor s nastavením zóny do příslušného konfiguračního adresáře. Adresář @PREFIX@/lib/firewalld/zones je použit

pro výchozí a záložní konfiguraci a adresář /etc/firewalld/zones je použit pro uživatelskou konfiguraci. Zóna je uložena do ifcfg spojení parametrem ZONE=*možnosti*. Pokud tento parametr chybí nebo je prázdný, je ve firewallu použitá výchozí zóna nastavena v souboru firewalld.conf. Pokud je síťové připojení řízeno pomocí NetworkManageru, můžete také pro změnu zóny použít nm-connection-editor.

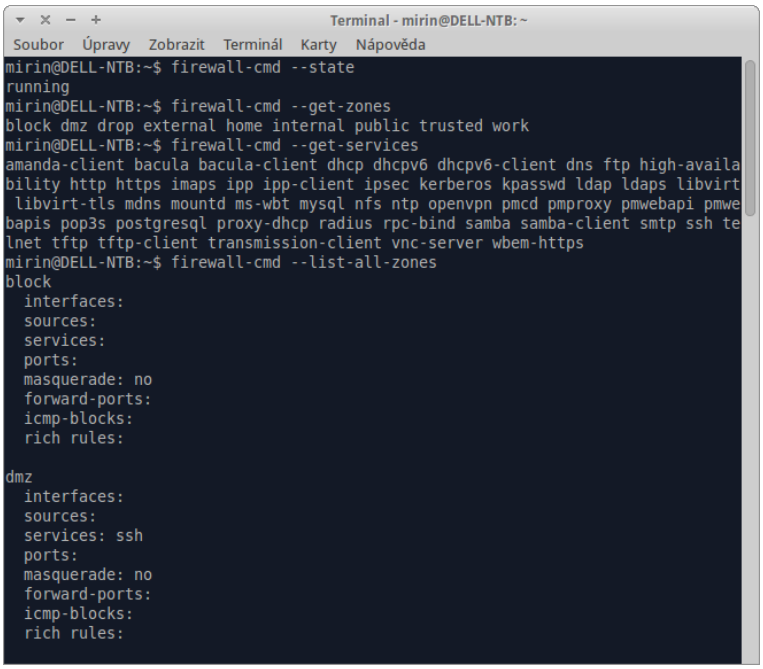
Klient příkazového řádku firewall-cmd podporuje všechny funkce firewallu. Nejčastěji používané všeobecné příkazy podle [18] jsou:

- stav firewallD:

  ```
  firewall-cmd –state
  ```

- seznam všech podporovaných zón:

  ```
  firewall-cmd --get-zones
  ```



Obrázek 20: ukázka výpisu firewall-cmd, zdroj: autor

- seznam všech podporovaných služeb:

  ```
  firewall-cmd --get-services
  ```

- seznam všech podporovaných icmp-typů:

  ```
  firewall-cmd --get-icmptypes
  ```

- seznam všech zón s nastavenými parametry:

  ```
  firewall-cmd --list-all-zones
  ```

- výpis výchozí zóny síťového připojení:

```
firewall-cmd --get-default-zone
```

- nastavení výchozí zóny:

```
firewall-cmd --set-default-zone=<zone>
```

- výpis aktivních zón:

```
firewall-cmd --get-active-zones
```

- výpis zóny vztahující se k danému síťovému rozhraní:

```
firewall-cmd --get-zone-of-interface=<interface>
```

- přidání rozhraní do zóny - přidá rozhraní k zóně, pokud k ní již nebylo přiřazeno dříve. Pokud je možnost *--zone* vynechána, je použita výchozí zóna:

```
firewall-cmd [--zone=<zone>] --add-interface=<interface>
```

- změna zóny patřící do rozhraní - podobné jako *--add-interface*, ale přiřadí rozhraní nové zóně, i když bylo předtím v jiné zóně:

```
firewall-cmd [--zone=<zone>] --change-interface=<interface>
```

- dotaz, zda je rozhraní v nějaké zóně:

```
firewall-cmd [--zone=<zone>] --query-interface=<interface>
```

- zapnutí/vypnutí/dotaz na panic režimu k blokování veškeré síťové komunikace v případě nouze:

```
firewall-cmd --panic-on|--panic-off|--query-panic
```

Pomocí nástroje firewall-cmd je možné podle [18] zadávat dočasné příkazy. Změny, které tyto povely vykonají, trvají pouze do restartu firewallu nebo počítače. Jsou to především příkazy:

- povolení služby v zóně - pokud není nastavena zóna, bude použit výchozí zóna. Pokud je nastaven timeout, bude služba povolena v zóně pouze po dobu <seconds> sekund. Pokud je služba již aktivní, není žádná varovná zpráva:

```
firewall-cmd [--zone=<zone>] --add-service=<service> [--
timeout=<seconds>]
```

- zakázání služby v zóně:

```
firewall-cmd [--zone=<zone>] --remove-service=<service>
```

- povolení portu a protokolu v zóně:

```
firewall-cmd [--zone=<zone>] --add-port=<port>[-<port>]/<protocol>
```

```
[--timeout=<seconds>]
```

- zakázání portu a protokolu v zóně:

```
firewall-cmd [--zone=<zone>] --remove-port=<port>[-<port>]/<protocol>
```

- povolení/zakázání maškarády v zóně:

```
firewall-cmd [--zone=<zone>] --add-masquerade|--remove-masquerade
```

- povolení/zakázání ICMP bloků v zóně:

```
firewall-cmd [--zone=<zone>] --add-icmp-block=<icmptype>|--remove-
icmp-block=<icmptype>
```

- povolení přesměrování nebo mapování portů v zóně - port je buď mapován na stejný port na jiném hostiteli, na jiný port na stejném hostiteli, nebo na jiný port na jiném hostiteli. Port může být buď jeden *<port>,* nebo rozsah portů *<port>*-*<port>*. Protokol je buď TCP, nebo UDP; *toaddr* je adresa IPv4. Přesměrování portů je možné pouze v IPv4 z důvodu omezení jádra:

```
firewall-cmd [--zone=<zone>] --add-forward-port=port=<port>[-
<port>]:proto=<protocol> {:toport=<port>[-<port>]|:toaddr=<address>
|:toport=<port>[-<port>]:toaddr=<address> }
```

- zakázání přesměrování nebo mapování portů v zóně:

```
firewall-cmd [--zone=<zone>] --remove-forward-port=port=<port>[-
<port>]:proto=<protocol> {:toport=<port>[-<port>]|:toaddr=<address>
|:toport=<port>[-<port>]:toaddr=<address> }
```

- příklad přesměrování ssh na hostitele 127.0.0.2 v zóně *home:*

```
firewall-cmd --zone=home –add-forward-port=port=22:proto=tcp:toaddr=
127.0.0.2
```

Tak jako dočasné příkazy, je možné pomocí firewal-cmd zadávat i příkazy trvalé – permanentní. Ty se nevykonají podle [18] okamžitě, ale až po restartu systému nebo firewallu. Je proto nutné používat oba typy příkazů. Všechny trvalé příkazy jsou stejné jako dočasné nebo všeobecné příkazy, pouze musejí mít na prvním místě parametr *--permanent*. Zde jsou opět některé z nich:

- seznam podporovaných permanentních zón/služeb/ICMP typů:

```
firewall-cmd --permanent --get-zones|--get-services|--get-icmptypes
```

- povolení trvalé maškarády v zóně:

```
firewall-cmd --permanent [--zone=<zone>] --add-masquerade
```

Dalším typem příkazů jsou přímé volby. Ty sice umožňují přímý přístup k nastavení firewallu, vyžadují však jistou znalost základních pojmů iptables. Přímé volby by měly být použity jako poslední možnost v případech, kdy požadovanou vlastnost není možné nastavit

například pomocí –add-service=<service> nebo –add-rich-rule=<pravidlo>. Prvním argumentem každého takového příkazu pak musí být *ipv4* (pro IPv4 iptables), *ipv6* (pro IPv6 ip6tables) nebo *eb* (pro ethernetovou bránu ebtables). Syntaxe příkazů pak podle [18] vypadá takto:

- předání příkazu firewallu - <args> mohou být všechny iptables, ip6tables a ebtables argumenty příkazového řádku:

```
firewall-cmd --direct --passthrough { ipv4|ipv6|eb } <args>
```

- přidání nového řetězce do tabulky:

```
firewall-cmd [--permanent] --direct --add-chain { ipv4|ipv6|eb }
<table> <chain>
```

- přidání pravidla s argumentem <arg> do řetězce v tabulce s prioritou:

```
firewall-cmd [--permanent] --direct --add-rule { ipv4|ipv6|eb }
<table> <chain> <priority> <args>
```

- vypsání všech pravidel přidaných do řetězce <chain> v tabulce <table> jako řádkový seznam oddělených argumentů:

```
firewall-cmd [--permanent] --direct --get-rules { ipv4 | ipv6 | eb }
<table> <chain>
```

Současné funkce firawallD jsou podle [18] následující - D-BUS rozhraní, Zóny, Služby, ICMP typy, Přímé rozhraní, Dočasná konfigurace, Trvalá konfigurace, Tray Applet, Grafický konfigurační nástroj, Klient příkazové řádky, Podpora pro ebtables, Výchozí/Chybová konfigurace v /usr/lib/firewalld, Nastavení systémové konfigurace v /etc/firewalld.

5 Analýza nasazení firewall ve firemním prostředí

Jak je uvedeno v [20], je firewall bezpečný a důvěryhodný počítač zapojený mezi privátní a veřejnou síť. Firewallový systém má nastavena určitá pravidla udávající, jaký síťový provoz může propouštět a jaký má být zablokován nebo odmítnut. V některých velkých organizacích se firewally používají i uvnitř sítě na ochranu citlivých oddělení firmy před ostatními zaměstnanci.

Jak již bylo zmíněno výše, je možné firewall konstruovat různými metodami. Nejpokročilejší metoda používá několik samostatných systémů a rozděluje síť na různě zabezpečené úrovně. Dva routery fungují jako filtry umožňující průchod pouze přesně definovaným typům provozu a mezi nimi jsou v demilitarizované zóně umístěny síťové servery, jako například poštovní brána, webový server a podobně. Takováto konfigurace může být velmi bezpečná a umožňuje snadno nastavit, kdo se může připojit z vnější do vnitřní sítě a kdo z vnitřní do vnější. Tento typ ochrany se používá obvykle ve větších společnostech.

V menších firmách je typičtější firewall tvořený pouze jedním počítačem, který zajišťuje vše. Firmy většinou neprovozují vlastní webový ani poštovní server, ale využívají placeného hostingu. Použití pouze jednoho firewallu je méně bezpečné řešení, protože pokud se v samotném firewallu objeví chyba, která umožní neautorizovaný přístup k němu, může být narušena celá bezpečnost sítě. Je nutno zmínit, že u velkého množství malých firem tento počítač s firewallem nahrazuje základní typ ADSL nebo WAN/LAN routeru s nastaveným firewallem a nějak fungující firewall na každé stanici uvnitř sítě.

5.1 Další hlediska

Při pohledu na problém firewallu umístěného v malé firmě je třeba také zmínit i jiné úhly pohledu než jen softwarové nastavení vlastního firewallu. To by sice mělo být primární, ale rozhodně ne jediné (pro někoho ani ne hlavní) hledisko při rozhodování, jaký firewall a zda vůbec do firmy pořídit.

Prvním z těchto hledisek může být hledisko ekonomické, nebo-li takové, které říká, na kolik celý linuxový firewall vlastně přijde v porovnání se základním modelem routeru s firewallem. Základní modely těchto zařízení je možné koupit od 700,- Kč, avšak rozumnější zařízení stojí lehce přes 1000,- Kč (routery s rychlostí rozhraní 1000Mb/s). Cena počítače, na kterém byl nakonfigurován a testován výše uvedený firewall byla nulová, neboť se jednalo o počítač, který sloužil jako záznamové zařízení původního analogového kamerového systému. Dostalo se mu tak nového využití. Jediné náklady, které tak firmě vznikly na nákup HW byla cena dvou PCI síťových karet 1000Mb/s, což představovalo částku dvakrát necelých 300,- Kč. Dvě síťové karty byly použity pro vytvoření demilitarizované zóny, jejíž konstrukce je někdy na levném routeru nemožná. Pokud by nebyl použitelný počítač k mání, cena

repasovaných desktopů se pohybuje od necelých 2000,- Kč. Jelikož je GNU/Linux pro počítač zdarma, náklady na operační systém jsou 0,- Kč. Následující tabulka tyto náklady shrnuje.

	router s firewallem	PC s GNU/Linux
vlastní HW	1 000,00 Kč	2 000,00 Kč
síťové karty	0,00 Kč	600,00 Kč
operační systém	0,00 Kč	0,00 Kč
Celkem	**1 000,00 Kč**	**2 600,00 Kč**

Tabulka 1: cenové porovnání

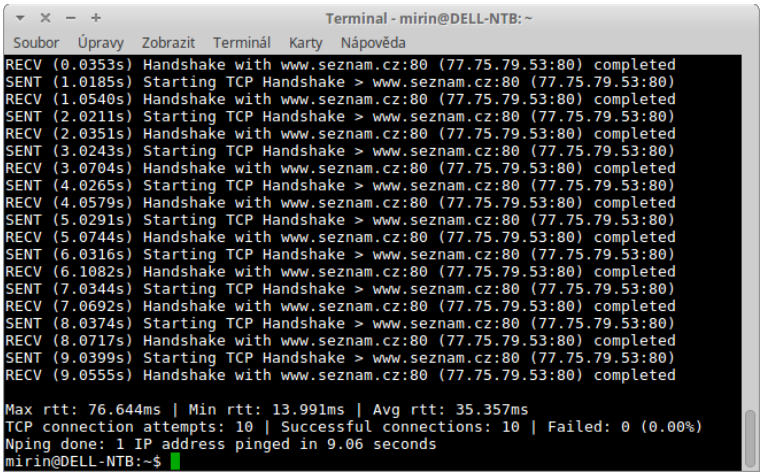
Do celkových nákladů by bylo možné zahrnout ještě cenu práce za nastavení jednotlivých firewallových řešení. Jelikož ale dnes již snad každá firma disponuje nějakým IT specialistou, ať vlastním či externím, a vlastní nastavení linuxového firewallu z předchozího textu není příliš obtížné, mohl by to tento specialista zvládnout v rámci svých možností a povinností při správě výpočetní techniky. Bohužel je třeba ale zmínit i to, že ne všem těmto „specialistům“ se do práce s linuxem chce. Pokud by bylo uvažováno, že nastavení firewallu by prováděla např. externí firma, je velká pravděpodobnost, že nastavení linuxového firewallu by přišlo firmu jistě dráž, neboť před vlastním nastavením je nutné ještě nainstalovat samotný operační systém na počítač.

Do ekonomického hlediska by se dala zakomponovat i energetická náročnost provozu obou typů firewallů, neboť i cena za elektrickou energii tvoří náklady firmy. Zde je třeba zmínit, že konstrukce s linuxovým firewallem je samozřejmě energeticky náročnější. V následující tabulce jsou uvedeny průměrné hodnoty příkonu běžného hodinového provozu, které předchozí tvrzení potvrzují.

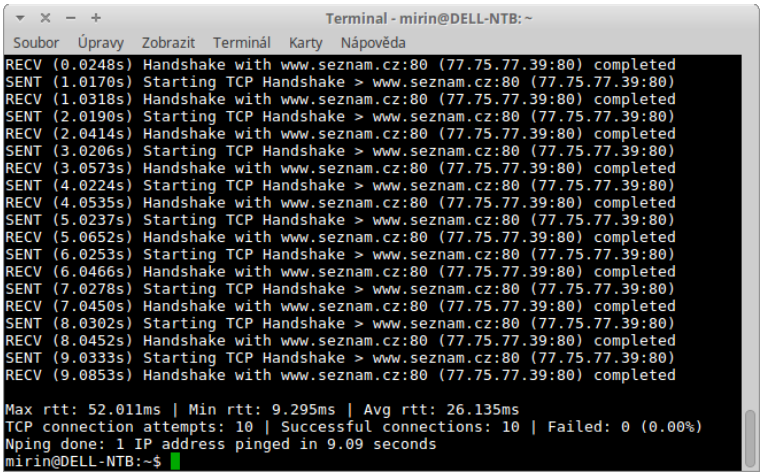
	router s firewallem	PC s GNU/Linux
příkon	4W	101W
%	3,96%	100%

Tabulka 2: příkon firewallů

Dalším možným hlediskem při volbě, jaký firewall zvolit, je hledisko rychlosti průchodu paketů firewallem. Na následujících obrazcích je zobrazen výsledek testu pomocí programu *nping*, který generuje síťové pakety a měří a analyzuje dobu jejich odezvy. Bylo několikrát odesíláno vždy deset náhodných paketů na adresu www.seznam.cz. První obrázek zobrazuje výsledek testu při zapojení původního routeru s firewallem, druhý pak při použití linuxového firewallu.



Obrázek 21: nping s původním routerem, zdroj: autor



Obrázek 22: nping s linux firewallem, zdroj: autor

Výsledky všech testů se vešly zhruba do intervalu ±10ms. Z těchto testů je zřejmé, že rychlost průchodu paketů přes firewall a jejich odezvy linuxový firewall nezpomaluje, spíše naopak. Toto hledisko se tak jeví jako výhodnější pro nové řešení.

Svou roli při výběru možného řešení ochrany vnitřní sítě by mohla hrát i možnost rozšiřování firewallu založeného na iptables o další moduly. Jedním ze zajímavých je modul pro odesílání logů o činnosti firewallu pomocí protokolu Netflow. Tento modul je samozřejmě možné stáhnout a doinstalovat zcela zdarma. Vlastní nastavení je opět velice snadné. To co je třeba logovat se na server odešle pomocí nově přidaného cíle NETFLOW, např.:

```
-j NETFLOW ipadresa:port
```

I když i většina malých routerů s firewallem umožňuje logování, jedná se nejčastěji o tvorbu logů do vlastní paměti. K jejich čtení je pak třeba se na router připojit a číst tyto informace např. pomocí webového prohlížeče, což je jistě náročnější, než číst veškeré informace o dění v síti (tedy nejen o dění na firewallu) na jednom místě.

Klady a zápory jednotlivých řešení jsou shrnuty do následující tabulky:

	router s firewallem	PC s GNU/Linux
cena pořízení a konfigurace	+	-
energetická náročnost	+	-
možnost konfigurace	-	+
bezpečnost	-	+
logování na server	-	+
Souhrn	**2 +**	**3 +**

Tabulka 3: souhrn výhod a nevýhod

Z výše uvedených hledisek jasně vyplývá, že v ekonomických hlediscích jasně dominuje původní řešení s jednoduchým routerem. Pokud se ale přihlédne k hlediskům neekonomickým, je třeba konstatovat, že zde má již jasnou převahu řešení linux+iptables.

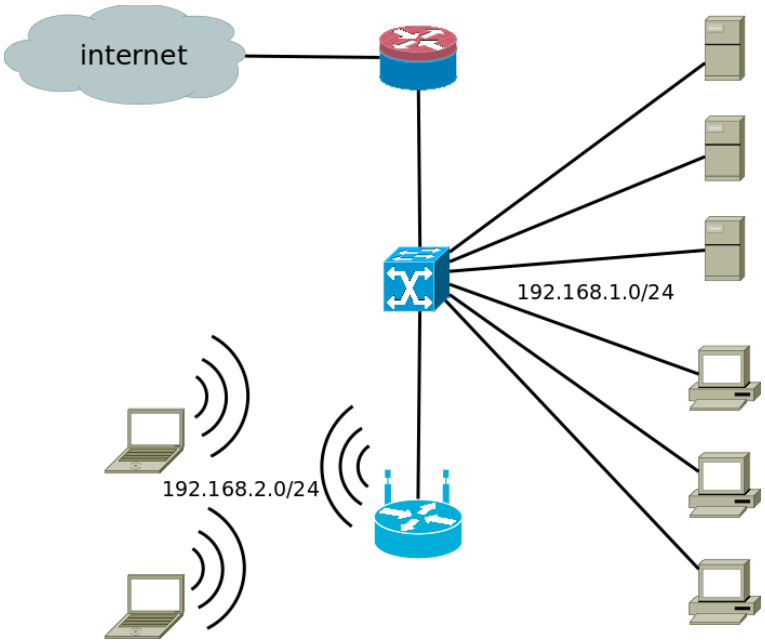
5.2 Aktuální stav ve firmě

Pro analýzu je zvolena menší firma, která má aktuálně tyto vlastnosti a parametry počítačové sítě:

- připojení k internetu pomocí Wi-Fi sítě 5,4 Ghz;jedna veřejná IP adresa;
- vnitřní neveřejná síť 192.168.1.0/24 pro firemní počítače;
- vnitřní neveřejná síť 192.168.2.0/24 pro návštěvníky oddělená od předchozí sítě routerem. Tato síť je dostupná převážně přes Wi-Fi připojení;
- ve vnitřní síti je DVR zařízení pro záznam obrazu z kamer, které je dostupné z internetu;
- jeden počítač slouží jako souborový server (dostupný pouze z vnitřní sítě);
- jeden počítač jako server pro účetní a informační software (dostupný pouze z vnitřní sítě);
- jeden počítač pro obsluhu docházkového terminálu (dostupný pouze z vnitřní sítě).

Celkem jsou tak v současné době zapojeny v síti dva jednoduché firewally, po jednom v každém routeru, a několik dalších v hostitelských počítačích. Zde se jedná o aplikace Windows Firewall. Na routeru mezi internetem a první vnitřní sítí je nastaven forwarding několika portů z internetu do vnitřní sítě. To se týká především portu 22, portu 8080 a portu 37777. Jedná se o porty pro SSH komunikaci, pro vzdálenou správu souborového serveru, pro komunikaci s webovým rozhraním ESET Remote Administrator serveru a portu pro komunikaci se serverem kamerového systému pomocí mobilních telefonů. Forwarding SSH

portu je v tomto případě ne zcela ideální a rovněž komunikace na ESET server nebude v budoucnu nutná.



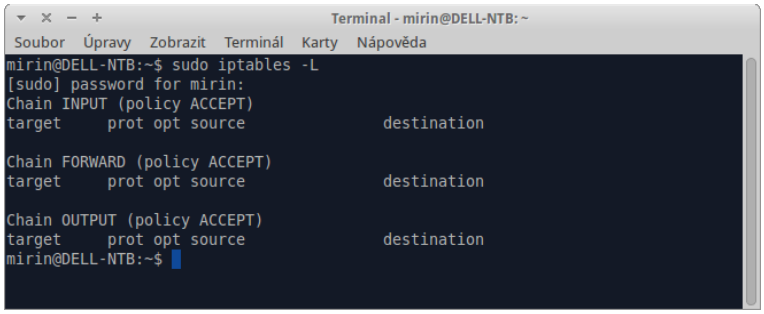
Obrázek 23: současný stav ve firmě, zdroj: autor

Co se týká omezení komunikace směrem z vnitřní sítě do internetu, zde nejsou ve firewallu uplatněna žádná restriktivní opatření, takže pro komunikaci směrem ven jsou povoleny všechny porty. Vzniká tu tedy riziko, že naváže-li nějaká škodlivá aplikace spojení do internetu na libovolném portu, může na tomto portu v opačném směru data i přijímat. Tento stav představuje asi největší bezpečnostní riziko, ale ne jediné. Přinejmenším stejným rizikem je i chybějící restrikce přístupu z internetu, neboť pokud dojde k překonání firewallu je, podle kapitoly 3.1, dostupná celá síť.

Lepším řešením firewallů by jistě bylo využití architektury Screened Subnet, tedy do demilitarizované zóny umístit dva nebo tři počítače sloužící jako docházkový server, server kamerového systému, případně server s ESET Remote Administratorem.

6 Návrh optimalizace nasazení firewall na GNU/Linux

Výchozí nastavení firewallu v jedné z nejrozšířenějších distribucí linuxového operačního systému Ubuntu je znázorněno na obrázku 24. Všechny výchozí politiky u řetězců tabulky filter jsou nastaveny na ACCEPT, což jinými slovy znamená, že systém nijak neomezuje ani nefiltruje síťovou komunikaci. To samozřejmě není z pohledu firemní bezpečnosti ideální. Je

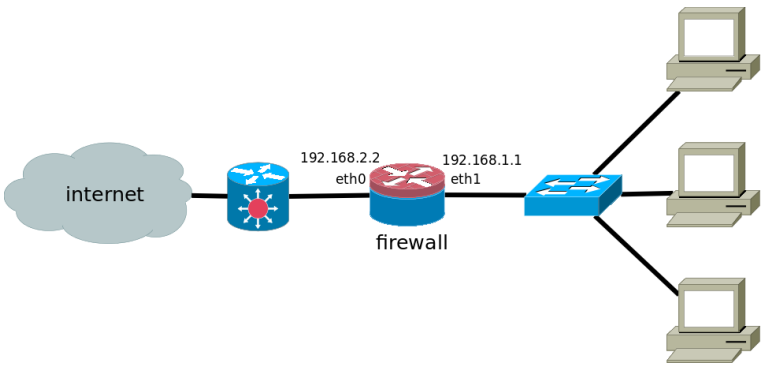


Obrázek 24: výchozí stav netfilteru, zdroj: autor)

proto vhodné změnit nastavení firewallu ze strategie benevolentní na strategii paranoidní a povolit průchod pouze těm paketům, které jsou potřebné pro provoz konkrétní firmy. Tomuto požadavku vyhoví i poměrně jednoduchý firewall (kapitola 6.1). Pokud je ovšem nutné zajistit a zabezpečit přístup k počítačům ve vnitřní síti zajišťujícím i vnější služby, je doporučováno umístit takovéto servery do demilitarizované zóny (kapitola 6.2).

6.1 Firewall nasazený mezi internetem a vnitřní sítí

Výchozí parametry počítače s implementovaným firewallem:



Obrázek 25: firewall mezi vnější a vnitřní sítí, zdroj: autor

- internetová síťová karta eth0:
  - adresa: 192.168.2.2;

    - maska: 255.255.255.0;
    - brána: 192.168.2.1;
- intranetová síťová karta eth1:
    - adresa: 192.168.1.1;
    - maska: 255.255.255.0.

Jako první příkaz nastavení firewallu je vhodné provést smazání všech případných pravidel – vyčistit firewall:

```
iptables -F
iptables -t nat -F
```

Jako druhý krok je doporučeno vytvořit pravidla, která zakáží veškerou komunikaci přes firewall. Jak uvádí např. [21], dosáhneme toho pomocí:

```
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT DROP
```

Těmito dvěma základními kroky je učiněn firewall absolutně neprůchodný. Jakýkoliv paket, který se na něm objeví, bude bez vyjímky zahozen. Některé prameny, jako např. [4] povolují veškerá odchozí spojení směrem ven z firewallu. Nicméně z důvodu vyššího zabezpečení je jistě lepší filtrovat i pakety odchozí. Je třeba tedy povolit ta spojení, která chceme používat, ve směru dovnitř i ven. Jako první povolíme podle [22] loopback pro případnou možnost kontroly:

```
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT
```

Jako další v pořadí je třeba nastavit a povolit veškeré potřebné porty pro průchozí pakety. Jedná se především o porty pro přístup k internetu a e-mailu. Doho docílíme pomocí příkazů:

```
iptables -A FORWARD -i eth1 -o eth0 -m state --state NEW -j ACCEPT
iptables -A FORWARD -i eth1 -o eth0 -p tcp --dport 80 -j ACCEPT
iptables -A FORWARD -i eth1 -o eth0 -p tcp --dport 443 -j ACCEPT
iptables -A FORWARD -i eth1 -o eth0 -p tcp --dport 8080 -j ACCEPT
iptables -A FORWARD -i eth1 -o eth0 -p tcp --dport 8081 -j ACCEPT
iptables -A FORWARD -i eth1 -o eth0 -p tcp --dport 25 -j ACCEPT
iptables -A FORWARD -i eth1 -o eth0 -p tcp --dport 220 -j ACCEPT
```

```
iptables -A FORWARD -i eth1 -o eth0 -p tcp --dport 20 -j ACCEPT

iptables -A FORWARD -i eth1 -o eth0 -p tcp --dport 21 -j ACCEPT
```

Číslo portu	Služba portu
20, 21	FTP
23	Telnet
53	DNS
80	HTTP
123	NTP
143, 220	IMAP
445	SMB
993	IMAPS, SSL
1433	Microsoft SQL
3050	Firebird
5432	PostgreSQL
8080, 8081	Tomcat

Číslo portu	Služba portu
22	SSH
25	SMTP
67, 68	DHCP
110	POP3
137, 138, 139	NetBIOS, SMB
443	HTTPS
465	SMTPS, SSL
995	POP3S, SSL
1194	OpenVPN
3306	MySQL
5800, 5900	VNC

Tabulka 4: výběr nejpoužívanějších portů

Uvedené příkazy by bylo možné sloučit do jednoho pomocí nového přepínače *--match multiport --dports*, takže potom by předchozí příkazy vypadaly takto:

```
iptables -A FORWARD -i eth1 -o eth0 -p tcp --match multiport --dports
  80,443,8080,8081,25,220,20,21 -j ACCEPT
```

Zpět do vnitřní sítě je třeba povolit pouze ta spojení, která byla již navázána směrem ven z vnitřní sítě. K tomu použijeme, jak uvádí [6] i [23], příkaz:

```
iptables -A FORWARD -i eth0 -o eth1 -m state --state ESTABLISHED,RELATED
  -j ACCEPT
```

Nic z toho by ale nefungovalo, pokud by nebyla povolena maškaráda. Firewall by totiž nevěděl, na jaké adresy má vracející se pakety odesílat. Zapneme ji tedy pomocí povelu:

```
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

Dalším příkazem je vhodné povolit ping na firewall z vnitřní sítě a zároveň odfiltrovat pokusy o zahlcení pomocí ICMP:

```
iptables -A INPUT -i eth1 -s 192.168.1.0/24 -d 192.168.1.1 -p ICMP
  --icmp-type echo-request -m limit --limit 1/s --limit-burst 5 -j
  ACCEPT

iptables -A OUTPUT -o eth1 -d 192.168.1.0 -p ICMP --icmp-type echo-reply
  -j ACCEPT
```

Pro vzdálenou administraci firewallu je dobré mít povolenu ještě komunikaci pomocí protokolu SSH. Kvůli vyšší bezpečnosti je to umožněno pouze z vnitřní sítě. V některých případech je doporučován ještě striktnější možnost SSH komunikace, a to pouze z určité adresy, což je však pro tento případ zbytečné. Povolení komunikace z celé vnitřní sítě docílíme pomocí:

```
iptables -A INPUT -i eth1 -s 192.168.1.0/24 -p tcp –-dport 22 -j ACCEPT

iptables -A OUTPUT -o eth1 -d 192.168.1.0/24 -p tcp –-sport 22 -j ACCEPT
```

Jelikož je firewall tvořen klasickým počítačem s běžným operačním systémem Linux, bylo by vhodné tento systém udržovat v aktuálním stavu. K tomu je v systému Ubuntu využíván příkaz apt-get upgrade. Aby tento příkaz fungoval, je třeba povolit odchozí spojení na portu 80 spolu s DNS portem 53:

```
iptables -A OUTPUT -o eth0 -p tcp -m state --state NEW -j ACCEPT

iptables -A OUTPUT -o eth0 -p udp --dport 53 --sport 1024:65535 -j
  ACCEPT

iptables -A OUTPUT -o eth0 -p tcp --dport 80 -j ACCEPT

iptables -A INPUT -i eth0 -p udp --sport 53 --dport 1024:65535 -j ACCEPT

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

Na závěr je nutné zapnout IP přesměrovávání, které je ve výchozím stavu vypnuté. To je možné provést například pomocí velmi jednoduchého skriptu:

```
#!/bin/bash

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

který spustíme samozřejmě pomocí *sudo* práv.

6.2 Firewall použitý k vytvoření DMZ

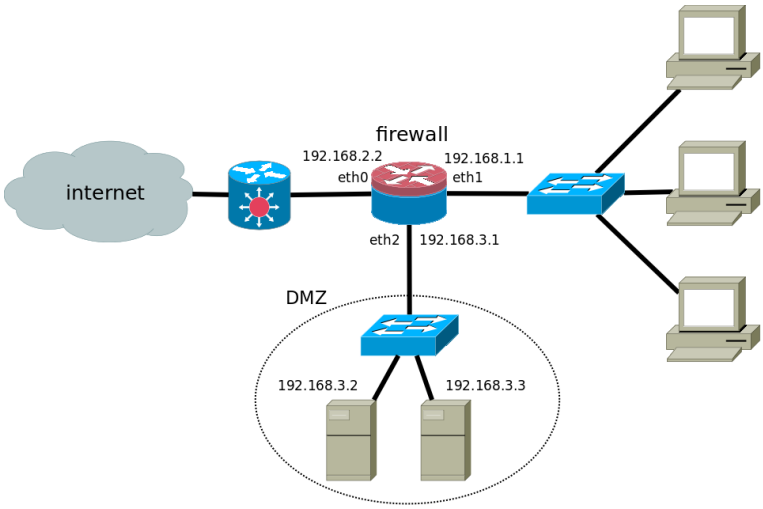
Výchozí parametry počítače s implementovaným firewallem:

- internetová adresa: 192.168.2.2;
- internetové rozhraní: eth0;
- adresa vnitřní sítě: 192.168.1.1;
- rozhraní vnitřní sítě: eth1;
- adresa v DMZ zóně: 192.168.3.1;
- rozhraní DMZ zóny: eth2.

Parametry počítačů v DMZ:

- adresa počítače s docházkovým systémem (HTTP serverem): 192.168.3.2;
- adresa počítače s kamerovým systémem (port 37777): 192.168.3.3.

Jako první bod v pořadí je doporučeno opět vyčistit firewall a poté vytvořit pravidla, která zakáží veškerou komunikaci dovnitř, ven i přes firewall. Jak uvádí [24] a [25], dosáhneme toho opět pomocí:



Obrázek 26: firewall s DMZ, zdroj: autor

```
iptables -F
iptables -t nat -F
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT DROP
```

Stejně jako v předchozím příkladu je samozřejmě třeba i zde povolit ta spojení, která chceme využívat. Jako první tedy povolíme podle [22] loopback pro případnou možnost kontroly:

```
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT
```

Pro vzdálenou administraci firewallu je opět dobré mít povolenu komunikaci pomocí SSH z vnitřní sítě. Toho docílíme pomocí:

```
iptables -A INPUT -i eth1 -s 192.168.1.0/24 -p tcp –-dport 22 -j ACCEPT
iptables -A OUTPUT -o eth1 -d 192.168.1.0/24 -p tcp –-sport 22 -j ACCEPT
```

Následuje povolení všech potřebných portů pro průchozí pakety z vnitřní sítě do internetu. Toho lze docílit stejně jako v předchozí kapitole pomocí příkazů:

```
iptables -A FORWARD -i eth1 -o eth0 -m state --state NEW -j ACCEPT

iptables -A FORWARD -i eth1 -o eth0 -p tcp --dport 80 -j ACCEPT

iptables -A FORWARD -i eth1 -o eth0 -p tcp --dport 443 -j ACCEPT

iptables -A FORWARD -i eth1 -o eth0 -p tcp --dport 8080 -j ACCEPT

iptables -A FORWARD -i eth1 -o eth0 -p tcp --dport 8081 -j ACCEPT

iptables -A FORWARD -i eth1 -o eth0 -p tcp --dport 25 -j ACCEPT

iptables -A FORWARD -i eth1 -o eth0 -p tcp --dport 220 -j ACCEPT

iptables -A FORWARD -i eth1 -o eth0 -p tcp --dport 20 -j ACCEPT

iptables -A FORWARD -i eth1 -o eth0 -p tcp --dport 21 -j ACCEPT
```

nebo zkráceně:

```
iptables -A FORWARD -i eth1 -o eth0 -p tcp --match multiport --dports
  80,443,8080,8081,25,220,20,21 -j ACCEPT
```

Stejně tak již navázaná odchozí spojení je třeba podle [23] povolit i při cestě zpět do vnitřní sítě, a to opět pomocí příkazu:

```
iptables -A FORWARD -i eth0 -o eth1 -m state --state ESTABLISHED,RELATED
  -j ACCEPT
```

Aby průchod firewallem pro vracející se pakety fungoval, povolíme opět maškarádu, a to opět pomocí povelu:

```
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

Je také třeba použít PREROUTING řetězec v NAT tabulce, který zkontroluje a zahodí všechny pokusy o přístup na zjevně falešné IP adresy:

```
iptables -t nat -A PREROUTING -i eth0 -s 10.0.0.0/8 -j DROP

iptables -t nat -A PREROUTING -i eth0 -s 172.16.0.0/12 -j DROP
```

Povolení NAT pro cílové IP dle komunikačních portů uvnitř DMZ zóny je trochu složitější. Nejprve musíme povolit přesměrování pro síťová rozhraní, potom přesměrování paketů tekoucích na určených portech na příslušné IP adresy počítačů v demilitarizované zóně:

```
iptables -A FORWARD -i eth0 -o eth2 -m state --state NEW -j ACCEPT

iptables -A FORWARD -p tcp -i eth0 -o eth2 --dport 80 -j ACCEPT

iptables -A FORWARD -p tcp -i eth0 -o eth2 --dport 37777 -j ACCEPT

iptables -A FORWARD -i eth2 -o eth0 -m state --state ESTABLISHED,RELATED
  -j ACCEPT
```

```
iptables -t nat -A PREROUTING -p tcp -i eth0 -d 192.168.2.2 --dport 80
  -j DNAT –to 192.168.3.2:80

iptables -t nat -A PREROUTING -p tcp -i eth0 -d 192.168.2.2 --dport
  37777 -j DNAT --to 192.168.3.3:37777
```

Jako jsme museli povolit průchod paketům z vnitřní sítě do internetu, musíme stejnou věc udělat i pro pakety jdoucí z vnitřní sítě do DMZ:

```
iptables -A FORWARD -i eth1 -o eth2 -m state --state NEW -j ACCEPT

iptables -A FORWARD -i eth1 -o eth2 -p tcp --dport 80 -j ACCEPT

iptables -A FORWARD -i eth1 -o eth2 -p tcp --dport 37777 -j ACCEPT

iptables -A FORWARD -i eth2 -o eth1 -m state --state ESTABLISHED,RELATED
  -j ACCEPT



iptables -t nat -A POSTROUTING -o eth2 -j MASQUERADE
```

Z důvodu vlastní aktualizace jsou opět povoleny odchozí a příchozí pakety na portech 80 a 53:

```
iptables -A OUTPUT -o eth0 -p tcp -m state --state NEW -j ACCEPT

iptables -A OUTPUT -o eth0 -p udp --dport 53 --sport 1024:65535 -j
  ACCEPT

iptables -A OUTPUT -o eth0 -p tcp --dport 80 -j ACCEPT

iptables -A INPUT -i eth0 -p udp --sport 53 --dport 1024:65535 -j ACCEPT

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

Nakonec je podle [24] nutné zapnout IP přesměrovávání. Provedení je možné pomocí tohoto jednoduchého skriptu::

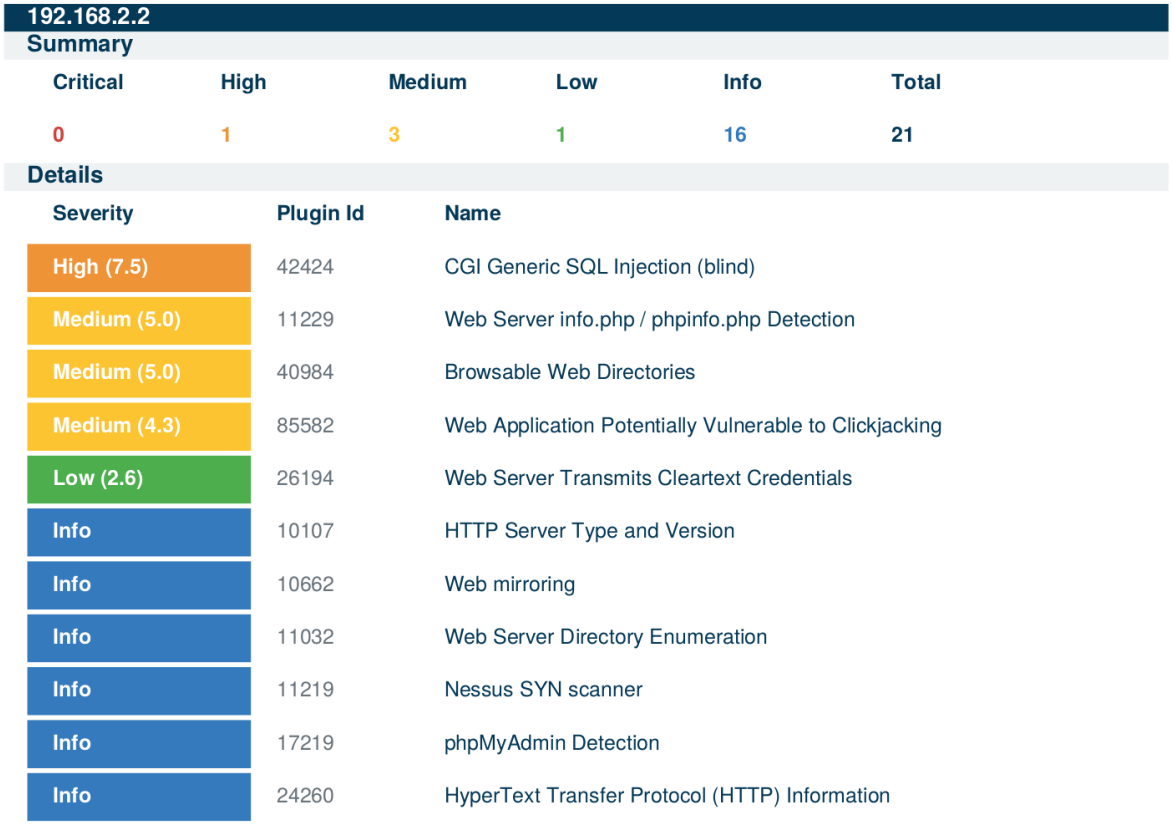
```
#!/bin/bash

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

6.3 Test firewallu

Oba firewally byly po své konečné konfiguraci testovány pomoci aplikace Nessus [26]. Nessus používá více než jeden milion uživatelů na celém světě pro hodnocení zranitelnosti a konfigurace počítačové sítě. Identifikací slabých míst a problémů s konfigurací tak zabraňuje síťovým útokům, které používají hackeři k průniku do PC sítě. Na obrázku 27 je znázorněn výsledek testu nazvaného „WEB test“, který kontroluje zranitelnost webových serverů, a to především:

- možnost procházení disků;

- sken šesti adresářů do šířky;
- test známých zranitelností u běžně používaných webových aplikací;
- vyzkoušení všech metod HTTP protokolu;
- pokus o znečištění HTTP protokolu.



192.168.2.2

Summary

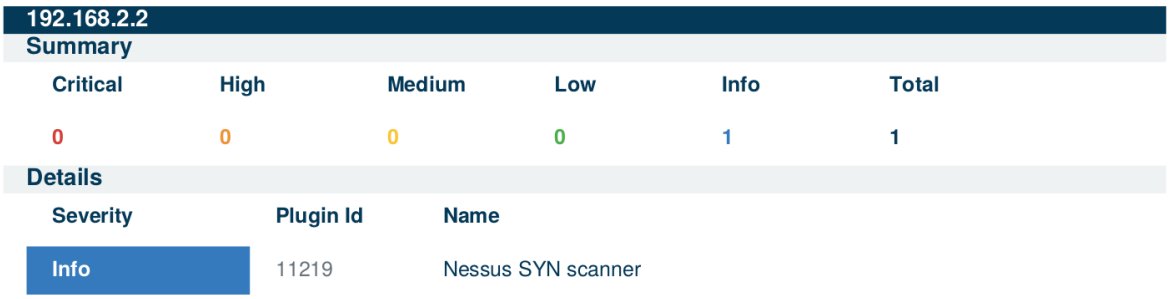
Critical	High	Medium	Low	Info	Total
0	1	3	1	16	21

Details

Severity	Plugin Id	Name
High (7.5)	42424	CGI Generic SQL Injection (blind)
Medium (5.0)	11229	Web Server info.php / phpinfo.php Detection
Medium (5.0)	40984	Browsable Web Directories
Medium (4.3)	85582	Web Application Potentially Vulnerable to Clickjacking
Low (2.6)	26194	Web Server Transmits Cleartext Credentials
Info	10107	HTTP Server Type and Version
Info	10662	Web mirroring
Info	11032	Web Server Directory Enumeration
Info	11219	Nessus SYN scanner
Info	17219	phpMyAdmin Detection
Info	24260	HyperText Transfer Protocol (HTTP) Information

Obrázek 27: WEB test bez spuštěného firewallu, zdroj: autor

Na obrázku 28 je zachycen výsledek po spuštění firewallu. Jejich vzájemným porovnáním je zřejmé, že se zabezpečení výrazně zvýšilo. Především nejkritičtější místo, náchylnost CGI aplikace na podstrčení SQL dotazu, bylo zcela eliminováno.



192.168.2.2

Summary

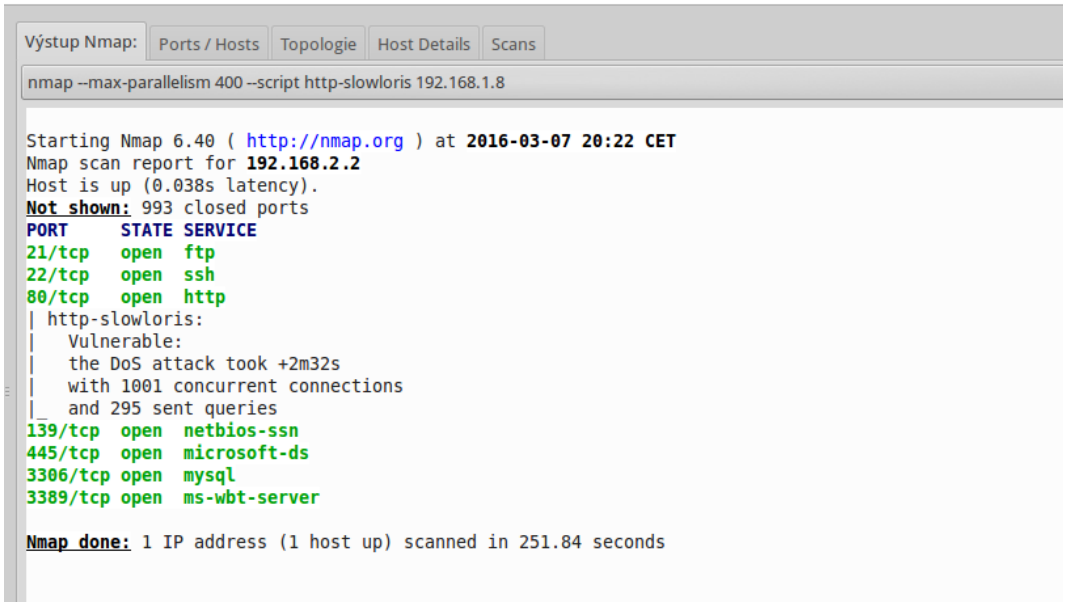
Critical	High	Medium	Low	Info	Total
0	0	0	0	1	1

Details

Severity	Plugin Id	Name
Info	11219	Nessus SYN scanner

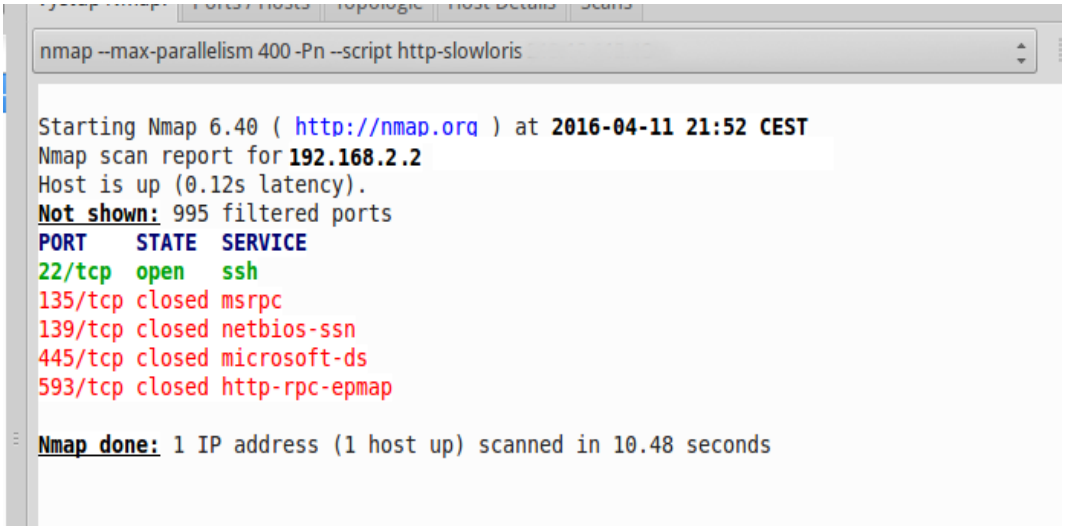
Obrázek 28: WEB test po spuštění firewallu, zdroj: autor

Pro kontrolu funkčnosti firewallu byl použit DDoS útok pomocí obecně známé aplikace nmap, která obsahuje velké množství vložených testovacích skriptů pro kontrolu nastavení a funkčnosti počítačové sítě. Obrázek 29 znázorňuje výsledek testu před zapnutím výše uvedeného firewallu. Z obrázku je zřejmé, že počítač je zranitelný pomocí DDoS útoku a má otevřené některé porty, které ale nebudou pro další provoz využívány. Jsou proto pomocí firewallu zakázány až na port 22 využívaný pro dočasnou vzdálenou komunikaci.



Obrázek 29: výsledek DDoS útoku před spuštěním firewallu, zdroj: autor

Další obrázek znázorňuje výsledek stejného testu po zapnutí firewallu. Na něm je patrné, že počítač již není zranitelný pomocí DDoS útoku, a celý test navíc proběhl během několika vteřin.



Obrázek 30: výsledek DDoS útoku po zapnutí firewallu, zdroj: autor

Z výše provedených testů je zřejmé, že pro zajištění větší bezpečnosti v počítačové síti, potažmo na internetu, je třeba používat zabezpečení pomocí firewallů. Již základní nastavení tohoto bezpečnostního nástroje dokáže výrazným způsobem zamezit následkům, které by mohly být způsobeny případnými útoky.

Nasazení linuxového firewallu neodhalilo sice žádný počítač, který by byl napaden nějakým škodlivým softwarem odesílajícím data do internetu na nestandardním portu. Pokud by však takový existoval, byl tento přenos ukončen. Rovněž byly uzavřeny všechny nepoužívané porty, které na jednoduchých firewallech jsou většinou otevřeny a tím zmenšeno riziko případného budoucího problému.

7 Závěr

Z celého předchozího textu vyplývá, že použití firewallu na operačním systému Linux není za pomoci dostupných konfiguračních nástrojů příliš obtížné. Pokud jsou shrnuty pravidla a vlastnosti, které by měl budovaný firewall mít, není jeho konstrukce přespříliš složitá.

Co se týká typu použitého firewallu, je doporučováno, pokud je to jen trochu možné, použít firewall s demiliterizovanou zónou. Jak je popsáno v předchozích kapitolách, vlastní prolomení firewallu pak ještě neznamená ohrožení vnitřní sítě. To byl také hlavní a rozhodující argument pro majitele firmy, proč tento firewall do firmy nasadit.

Konfiguraci firewallu je třeba ale vyřešit skriptem, který je načten vždy při startu systému, neboť předchozí příkazy nastavují iptables pouze dočasně a to do vypnutí nebo restartu systému. Pokud není startovací skript použit, jsou po startu počítače všechny řetězce iptables opět prázdné. Příklad skriptu pro firewall s demilitarizovanou zónou by tedy podle této práce a [24] mohl vypadal tak, jak je uvedeno v příloze. Uvedený skript je doporučeno spouštět okamžitě po nahození síťových rozhraní. Toho lze docílit pomocí vložení řádku na konec souboru */etc/network/intarfaces* odkazujícího na skript uložený v */etc/network/if-up.d*. Je-li tedy název skriptu *fwscript*, bude vložený řádek vypadat takto:

```
post-up /etc/network/if-up.d/fwscript
```

Jelikož firewall popsaný v této práci je jednoúčelový počítač, je možné ho vytvořit i pomocí hardwaru staršího data výroby. Z toho důvodu je více než vhodné použít takovou distribuci Linuxu, která nemá velké systémové nároky a nepotřebuje ani grafické uživatelské rozhraní, proto byla vybrána distribuce Ubuntu server 14.04LTS, jež tento předpoklad zcela splňuje. I proto je firewall vytvořen pomocí klasických iptables příkazů namísto novějších nástrojů ufw nebo firewallD. Toto řešení by naopak bylo výhodnější použít k nastavení firewallu, který by běžel pouze na lokálním počítači, ale nevykonával by pouze funkci firewallu, nýbrž sloužil by zároveň jako plnohodnotná stanice s grafickým uživatelským operačním systémem.

Použitá literatura

[1] FIKKAR, Jaroslav. CheckPoint FireWall-1 3.0: Bezpečnostní řešení pro platformy UNIX a Windows NT. *PC World* [online]. 1998 [cit. 2014-10-28]. Dostupné z: http://pcworld.cz/archiv/checkpoint-firewall-1-3-0-19247

[2] Nftables. *Archlinux* [online]. 2014, 6.11.2014 [cit. 2014-11-10]. Dostupné z: https://wiki.archlinux.org/index.php/nftables

[3] Access control list. *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 16.11.2014 [cit. 2015-07-18]. Dostupné z: https://cs.wikipedia.org/wiki/Access_control_list#Po.C4.8D.C3.ADta.C4.8Dov.C3.A9_s.C3.ADt.C4.9B

[4] DOČEKAL, Michal. Linuxový firewall, základy iptables. *Správa linuxového serveru* [online]. 2010 [cit. 2014-11-03]. Dostupné z: http://www.linuxexpres.cz/praxe/sprava-linuxoveho-serveru-linuxovy-firewall-zaklady-iptables

[5] BOTOŠ, Csaba. Vše o iptables. *ROOT.CZ* [online]. 2006 [cit. 2014-11-04]. Dostupné z: http://www.root.cz/clanky/vse-o-iptables-uvod/

[6] JAMRICH, Marián. *Zabezpečenie siete pomocou firewallu na operačnom systéme Linux.* Bratislava, 2010. Absolventská práce. SPŠE Zochova 9, Bratislava. Vedoucí práce Ing. Zora Hledíková.

[7] *Netfilter* [online]. 1999-2014 [cit. 2014-11-04]. Dostupné z: http://netfilter.org/index.html

[8] Shinder, T. W. Best damn firewall book period. Oxford: Elsevier Science [distributor], 2nd edition, 2007. ISBN 15-974-9218-3

[9] Zwicky, E. *Building Internet firewalls.* Cambridge: O'Reilly, 2nd edition, 2000. ISBN 15-659-2871-7

[10] TŘETINA, Michal. *Analýza principů a funkcí Firewall na IPv6*. Pardubice, 2015. Diplomová práce. Univerzita Pardubice. Vedoucí práce Mgr. Josef Horálek, Ph.D.

[11] PROTIVA, Jan. Bezpečnost počítačových sítí. In: *Bezpečnost počítačových sítí* [online]. 2005 [cit. 2015-10-13]. Dostupné z: http://skola.vydrar.net/DatoveSite/Bezpecnost_pocitacovych_siti.pdf

[12] Methods of Attack. *TLDP: Linux Network Administrators Guide* [online]. 2005 [cit. 2015-10-13]. Dostupné z: http://www.tldp.org/LDP/nag2/x-082-2-firewall.attacks.html

[13] GOWDIAK, Adam. *Techniques used for bypassing firewall systems* [gowdiak-bypassing-firewalls.pdf]. Poznan, 2003 [cit. 2015-10-13].

[14] VESELÝ, Jakub. 2011. *Etický hacking - učební pomůcka pro předmět: Bezpečnost informačních systémů.* Zlín. Diplomová práce. Univerzita Tomáše Bati ve Zlíně. Vedoucí práce Doc. Mgr. Roman Jašek, Ph.D.

[15] PURDY, Gregor N. *Linux iptables: pocket reference*. Sebastopol, CA: O'Reilly, c2004, iii, 91 p. ISBN 05-960-0569-5.

[16] Bezpečnost:firewall:iptables [Ubuntu Česko]: iptables. *Ubuntu Česko* [online]. 2015, 23.6.2015 [cit. 2015-09-30]. Dostupné z: http://wiki.ubuntu.cz/bezpečnost/firewall/iptables

[17] Firewall. 2014. *Ubuntu documentation* [online]. [cit. 2015-10-19]. Dostupné z: https://help.ubuntu.com/14.04/serverguide/firewall.html

[18] FirewallD: FedoraProject. *Fedora* [online]. 2015, 21.7.2015 [cit. 2015-10-22]. Dostupné z: https://fedoraproject.org/wiki/FirewallD#Dynamic_firewall_with_FirewallD

[19] How to replace Ufw with Firewalld in Linux Mint 15. *LinuxBSDos.com* [online]. 2013, 26.6.2013 [cit. 2015-10-26]. Dostupné z: http://linuxbsdos.com/2013/06/26/how-to-replace-ufw-with-firewalld-in-linux-mint-15/

[20] FIREWALL - IPTABLES. *Katedra informatiky VŠB Ostrava: STUDIUM INFORMATIKY* [online]. Ostrava [cit. 2015-10-22]. Dostupné z: http://www.cs.vsb.cz/grygarek/TPS-0304/projekty0304/ipchains/3/Firewall.htm

[21] NICOLA, Carlo U. *Firewalls* [online]. Muttenz. 2012 [cit. 2015-11-06]. Dostupné z: http://web.fhnw.ch/plattformen/ns/vorlesungsunterlagen-1/firewalls/firewalls-and-iptables

[22] Iptables - NAT/ DMZ example. *Pacific Simplicity* [online]. Ron Brash, 2011, 2015-03-15 [cit. 2015-11-11]. Dostupné z: https://www.pacificsimplicity.ca/blog/iptables-nat-dmz-example

[23] iptables Firewall Example. *The Department of Computer Science* [online]. Bozeman, USA: Montana State University, 2004, 2004-01-26 [cit. 2015-11-11]. Dostupné z: http://www.cs.montana.edu/courses/309/topics/nat/firewall.html

[24] iptables DMZ Example. *The Department of Computer Science* [online]. Bozeman, USA: Montana State University, 2004, 2004-01-26 [cit. 2015-11-11]. Dostupné z: http://www.cs.montana.edu/courses/309/topics/nat/dmz.html

[25] Setting DMZ with iptables. *LINUX MADE EASY* [online]. Punjab, Indie: Vipin Gupta, 2008, 2008-01-09 [cit. 2015-11-13]. Dostupné z: http://linuxforall.blogspot.cz/2008/01/setting-dmz-with-iptables.html

[26] *Nessus Vulnerability Scanner: Tenable Network Security* [online]. Columbia, USA, 2016 [cit. 2016-02-26]. Dostupné z: http://www.tenable.com/products/nessus-vulnerability-scanner#

[27] Wikipedie. *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2015-10-14]. Dostupné z: https://cs.wikipedia.org/wiki/Hlavní_strana

Seznam obrázků

Seznam tabulek

Příloha

Startovací skript firewallu s DMZ zónou:

```
#!/bin/bash

###########################################

# Konfigurace firewallu s DMZ zonou

###########################################

# Zadani promennych

LAN_IP="192.168.1.1"

LAN_NET="192.168.1.0/24"

LAN_IFACE="eth1"


INET_IP="192.168.2.2"

INET_IFACE="eth0"


DMZ_HTTP_IP="192.168.3.2:80"

DMZ_DVR_IP="192.168.3.3:37777"

DMZ_IP="192.168.3.1"

DMZ_IFACE="eth2"


LO_IP="127.0.0.1"

LO_IFACE="lo"


IPTABLES="/usr/local/sbin/iptables"


###########################################

# Nahrani všech potrebnych IPTables modulu, pokud jiz nejsou v jadre

/sbin/depmod -a


# Pridani nekterych iptables cilu, jako napr. LOG, REJECT a MASQUERADE.

/sbin/modprobe ipt_LOG
```

```
/sbin/modprobe ipt_MASQUERADE


# Podpora pro sledovani spojeni FTP a IRC
#/sbin/modprobe ip_conntrack_ftp
#/sbin/modprobe ip_conntrack_irc


#DULEZITE:  Povoleni IP forwardingu
echo "1" > /proc/sys/net/ipv4/ip_forward


##############################################
# Nastaveni paranoidniho rezimu
$IPTABLES -F
$IPTABLES -t nat -F
$IPTABLES -P INPUT DROP
$IPTABLES -P OUTPUT DROP
$IPTABLES -P FORWARD DROP


##############################################
# povoleni retezcu pro localhost spojeni
$IPTABLES -A INPUT -i $LO_IFACE -j ACCEPT
$IPTABLES -A OUTPUT -o $LO_IFACE -j ACCEPT


##############################################
# povoleni SSH z LAN site
$IPTABLES -A INPUT -i $LAN_IFACE -s $LAN_NET -p tcp –-dport 22 -j ACCEPT
$IPTABLES -A OUTPUT -o $LAN_IFACE -d $LAN_NET -p tcp –-sport 22 -j ACCEPT


##############################################
# povoleni pruchodu vybranych paketu z LAN site smerem do internetu a zpet
$IPTABLES -A FORWARD -i $LAN_IFACE -o $INET_IFACE -m state --state NEW -j
  ACCEPT
```

```
$IPTABLES -A FORWARD -i $LAN_IFACE -o $INET_IFACE -p tcp --match multiport
   --dports 80,443,8080,8081,25,220,20,21 -j ACCEPT

$IPTABLES -A FORWARD -i $INET_IFACE -o $LAN_IFACE -m state --state
   ESTABLISHED,RELATED -j ACCEPT

iptables -t nat -A POSTROUTING -o $INET_IFACE -j MASQUERADE


##########################################

# kontrola zjevne falesnych zdrojovych adres pristupujicích z internetu

$IPTABLES -t nat -A PREROUTING -i $INET_IFACE -s 10.0.0.0/8 -j DROP

$IPTABLES -t nat -A PREROUTING -i $INET_IFACE -s 172.16.0.0/12 -j DROP

$IPTABLES -t nat -A PREROUTING -i $INET_IFACE -s $INET_IP -j DROP


##########################################

# povoleni vlastni aktualizace operacniho systemu

$IPTABLES -A OUTPUT -o $INET_IFACE -p tcp -m state --state NEW -j ACCEPT

$IPTABLES -A OUTPUT -o $INET_IFACE -p udp --dport 53 --sport 1024:65535 -j
   ACCEPT

$IPTABLES -A OUTPUT -o $INET_IFACE -p tcp --dport 80 -j ACCEPT

$IPTABLES -A INPUT -i $INET_IFACE -p udp --sport 53 --dport 1024:65535 -j
   ACCEPT

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


##########################################

# DMZ cast

# povoleni pruchodu paketu z LAN site do DMZ a zpet

$IPTABLES -A FORWARD -i $LAN_IFACE -o $DMZ_IFACE -m state --state NEW -j
   ACCEPT

$IPTABLES -A FORWARD -i $LAN_IFACE -o $DMZ_IFACE -p tcp --dport 80 -j ACCEPT

$IPTABLES -A FORWARD -i $LAN_IFACE -o $DMZ_IFACE -p tcp --dport 37777 -j
   ACCEPT


$IPTABLES -A FORWARD -i $DMZ_IFACE -o $LAN_IFACE -m state --state
   ESTABLISHED,RELATED -j ACCEPT
```

```
$IPTABLES -t nat -A POSTROUTING -o $DMZ_IFACE -j MASQUERADE


##############################################

# Povoleni IP presmerovani pro DMZ zonu

iptables -A INPUT -i $INET_IFACE -p tcp --dport 80 -j ACCEPT

iptables -A INPUT -i $INET_IFACE -p tcp --dport 37777 -j ACCEPT

iptables -t nat -A PREROUTING -p tcp -i $INET_IFACE -d $INET_IP --dport 80 -j
  DNAT –to $DMZ_HTTP_IP

iptables -t nat -A PREROUTING -p tcp -i $INET_IFACE -d $INET_IP --dport 37777
  -j DNAT --to DMZ_DVR_IP

iptables -A FORWARD -i $INET_IFACE -o $DMZ_IFACE -m state --state NEW -j
  ACCEPT

iptables -A FORWARD -p tcp -i $INET_IFACE -o $DMZ_IFACE --dport 80 -j ACCEPT

iptables -A FORWARD -p tcp -i $INET_IFACE -o $DMZ_IFACE --dport 37777 -j
  ACCEPT


iptables -A FORWARD -i $DMZ_IFACE -o $INET_IFACE -m state --state
  ESTABLISHED,RELATED -j ACCEPT
```

Univerzita Hradec Králové
Fakulta informatiky a managementu
Akademický rok: 2016/2017

Studijní program: Aplikovaná informatika
Forma: Kombinovaná
Obor/komb.: Aplikovaná informatika (ai2-k)

Podklad pro zadání DIPLOMOVÉ práce studenta

PŘEDKLÁDÁ:	ADRESA	OSOBNÍ ČÍSLO
Bartoš Miroslav	E. Beneše 390, Polička - Horní Předměstí	I1300160

TÉMA ČESKY:

Analýza a optimalizace softwarových firewall na operačních systémech Linux

TÉMA ANGLICKY:

Analysis and optimization software firewall on Linux OS

VEDOUCÍ PRÁCE:

Mgr. Josef Horálek, Ph.D. - KIT

ZÁSADY PRO VYPRACOVÁNÍ:

Cílem práce je podrobně zmapovat principy a možnosti firewall na Linux OS a navrhnout možnosti optimalizace využití a nasazení firewall na Linux OS.
Osnova práce:
Úvod
Analýza firewall řešení
Principy ochrany sítě s využitím firewallu
Implementace firewallů na GNU/Linux
Analýza nasazení firewall ve firemním prostředí
Návrh optimalizace nasazení firewall na GNU/Linux
Závěr

SEZNAM DOPORUČENÉ LITERATURY:

E. Zwicky, Building Internet firewalls
G. N. PURDY, Linux iptables: pocket reference
M. DOČEKAL, Linuxový firewall, základy iptables

Podpis studenta: ..... **Datum:** .....

Podpis vedoucího práce: ..... **Datum:** .....