Univerzita Komenského v Bratislave
Fakulta Matematiky, Fyziky a Informatiky

Evidenčné číslo 5c447813-2ab0-41f9-9543-1a1fc4b1ae54

Virtuálne Stroje

2011

Adam Saleh

Univerzita Komenského v Bratislave
Fakulta Matematiky, Fyziky a Informatiky

Virtuálne Stroje

Bakalárska práca

Študijný program:	Informatika
Študijný odbor:	9.2.1 Informatika
Školiace pracovisko:	Katedra informatiky FMFI
Školiteľ:	RNDr. Richard Ostertág PhD.

Bratislava 2011

Adam Saleh

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Adam Saleh
Študijný program: informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: 9.2.1. informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský

Názov: Virtuálne stroje

Cieľ:
1. Prehľad existujúcich riešení implementácie špecializovaných virtuálnych strojov.
2. Popis efektívneho spôsobu tvorby virtuálnych strojov optimalizovaných pre špecifickú službu.
3. Praktická implementácia sady takýchto zariadení.

Vedúci: RNDr. Richard Ostertág, PhD.
Katedra: FMFI.KI - Katedra informatiky

Spôsob sprístupnenia elektronickej verzie práce:
bez obmedzenia

Dátum zadania: 22.10.2010

Dátum schválenia: 22.10.2010

doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

.....
študent

.....
Vedúci

Čestne prehlasujem, že na bakalárskej práci som pracoval samostatne na základe vlastných poznatkov, konzultácii a štúdia odbornej literatúry, ktorej úplný prehľad je uvedený v zozname použitej literatúry.

.....

Pod'akovanie

Rád by som pod'akoval RNDr. Ostertágovi, za cenné pripomienky k práci a tiež kolektívu okolo študentského serveru st.dcs.fmph.uniba.sk, za poskytnutie testovacieho prostredia.

Abstrakt

V posledných rokoch sa stala virtualizácia serverových systémov každodennou súčasťou práce administrátora. S rozšírením virtualizácie sa mení aj prístup k vytváraniu serverov a zvyšuje sa popularita používania takzvaných virtuálnych zariadení[1]. Virtuálne zariadenie je nakonfigurovaný virtuálny server, ktorý je optimalizovaný na poskytovanie práve jednej služby. Administrátor si potom môže diskový obraz takéhoto virtuálneho stroja stiahnuť a spustiť na svojej virtualizačnej platforme, bez toho aby musel virtuálny server konfigurovať ručne. Zatiaľ čo v dnešnej dobe už existuje niekoľko firiem ktoré takéto virtuálne stroje „na-kľúč“ poskytujú, nenašiel som zatiaľ ucelenú publikáciu, ktorá by sa zaoberala ich vytváraním. Preto je mojím cieľom popísať zásady vytvárania takýchto virtuálnych strojov na báze Linuxu.

Kľúčové slová: Plná virtualizácia, Zásady bezpečnosti, Virtuálne zariadenia, Open-Source

[1] v angličtine známe pod termínom virtual appliance

Abstract

In recent years, virtualization of server operating systems has became an indispesible part of sysadmins work. With rising utilisation of virtualization there is a groving trend in popularity of so-called virtual appliances. Virtual appliance is a preconfigured instance of virtual server, that is optimized to server one particular purpose. Sysadmin can then download such virtual appliance and run it on his virtualization platform without the need of configuring it from scratch. While today there are many preset virtual appliances, I have yet to find a work that thoroughly describes the creation process. That's why my goal in this thesis is to describe process of creating such virtual appliances, on a basis of Linux Operating System.

Keywords: Full Virtualization, Security, Virtual appliances, Open-Source

Obsah

Zoznam obrázkov

Úvod

Témou tejto práce sú virtuálne zariadenia na báze Linuxu. Preto sa budeme zaoberať prevážne administráciou linuxových serverov a ich optimalizovaním pre poskytovanie rôznych služieb. Možnosť virtualizovať hardvér priniesla administrátorom veľa nových možností, a práve možnosť mať kompletne nastavený server pripravený na nasadenie ako jeden jednoducho prenosný súbor patrí medzi najlákavejšie z nich.

Naším cieľom je popísať zásady tvorby takéhoto prednastaveného virtuálneho stroja.

V prvej kapitole sa budeme venovať prehľadu rôznych riešení tvorby virtuálnych zariadení, a všeobecným zásadám pri ich tvorbe.Pri vytváraní takéhoto prednastaveného serveru sa zaoberáme hlavne jeho znovupoužiteľnosťou, zabezpečením a efektivitou.

- Znovupoužiteľnosť je základnou vlastnosťou virtuálneho zariadenia a hlavným dôvodom prečo vôbec virtuálne zariadenia vytvárať. Pri klasickej inštalácii linuxového serveru sa hýbeme medzi dvoma extrémami. Na začiatku máme systém ktorý vieme použiť v rôznych prostrediach na ľubovoľný účel, ale všetky úpravy je ešte len nutné urobiť. Na konci inštalácie máme funkčný systém optimalizovaný pre žiadanú službu, ktorý však nie je prenosný mimo svojho prostredia. Naším cieľom je aby sme vytvorené virtuálne zariadenie mohli použiť s čo najmenšími úpravami v čo najviac prípadoch.

- Virtuálne zariadenie, ako každý na sieti pripojený server musí spĺňať určité bezpečnostné požiadavky. Práve zabezpečovaniu svojich serverov venujú administrátori veľkú časť svojho času pri nastavovaní sieťových služieb. Preto je naším cieľom uplatniť najčastejšie požadované kritériá priamo na virtuálnych zariadeniach.

- Efektivita serveru je pri jednotlivých inštaláciách často až za stabilitou systému a jeho zabezpečním. Práve virtuálne prostredie však kladie na efektivitu zvýšené požiadavky.Virtuálny stroj bude mať k dispozícii menšie množstvo pamäte, menšie množstvo úložného priestoru a o procesorový čas bude súperiť s ostatnými virtuálnymi strojmi ktoré bežia na rovnakom fyzickom stroji. Virtualizácia zo svojej podstaty prináša ďalšiu réžiu pri prístupe ku skutočnému hardvéru, čím nároky na efektivitu ešte stúpajú.

Druhá kapitola je venovaná konkrétnym službám a ich nastaveniu. V nej popisujeme dva duhy strojov: infraštruktúrne a produkčné.

Infraštruktúrne zariadenia slúžia na vytvorenie prostredia pre produkčné zariadenia, ktŕé uľahčí ich administráciu. Medzi ne bude patriť zariadenie poskytujúce DNS server, centrálnu správu identít, a zariadenie s poskytujúce centrálnu správu logovania a monitorovanie strojov. O týchto zariadeniach predpokladáme, že budú prístupné len z vnútornej siete.

Produkčné zariadenia budú slúžiť na poskytovanie rôznych služieb na internete, alebo inej verejne prístupnej sieti. Medzi ne budú patriť zariadenia poskytujúce prístup k databáze,

e-mailovému serveru a web-serveru. Vďaka infraštruktúrnym zariadeniam sa administrátor spravujúci niekoľko produkčných zariadení nemusí jednotlivo zaoberať zaoberať nastavovaním prístupových hesiel, zálohovania logov a kontrolou ich fungovania.

Jednotliivé programy pre naše zariadenia sme potom vyberali s predpokladom, že pri ich finálnom nasadení bude virtuálnych strojov niekoľko desiatok ktoré bude spravovať niekoľko administrátorov.Napríklad v časti ktorá sa zaoberá centrálnou správou užívateľov sme sa rozhodli použiť na implementáciu adresárový server OpenLDAP v spolupráci s veľmi jednoduchým administrátorským rozhraním prístupným cez príkazový riadok.

Ak by sme plánovali väčšie nasadenie, aké býva bežné v stredne veľkých firmách (niekoľko stoviek užívateľov, z ktorých má každý svoju pracovnú stanicu) tak by sme pravdepodobne použili LDAP server, ktorý bude slúžiť ako databáza pre autentifikáciu cez Kerberos. Kombináciu LDAP a Kerberos napríklad používa riešenie na centrálnu správu identít FreeIPA.Ten má navyše veľmi pekné užívateľské rozhranie, ktoré je pri správe veľkého množstva užívateľov nevyhnutnosťou.

Základné pojmy virtualizácie

Virtualizácia je široká oblasť, ktorá zatiaľ nemá ustálenú slovenskú terminológiu. Preto sme sa rozhodli na začiatok uviesť základné termíny, a význam v akom ich budeme používať.

Virtualizácia je technológia umožňujúca na požiadanie poskytovať virtualizovaný hardvér, na ktorom je možné spustiť zvolený operačný systém. Takto spustený operačný systém by nemal byť sám schopný zistiť že nebeží na fyzickom hardvéri.

Virtuálny stroj je fyzická reprezentácia virtualizovaného hardvéru, najčastejšie vo vo forme súboru s obrazom disku a uloženej konfigurácie ostatných virtuálnych zariadení (počet jadier virtualizovaného procesoru, veľkosť virtualizovanej pamäte).

Virtuálne zariadenie je virtuálny stroj upravený na poskytovanie špecifickej služby. Táto práca sa bude zaoberať ich efektívnym vytváraním.

Hostiteľský systém je fyzický hardvér s operačným systémom na ktorom beží virtualizácia. V angličtine býva označovaný aj ako „host OS“.

Definícia virtualizácie je o niečo užšia než býva zvykom, často sa pod virtualizačné technológie radia aj pokročilé kontajnerové systémy (napr. BSD Jails, openVZ, lcx), alebo systémy ktoré pracujú na princípe para-virtualizácie a vyžadujú na beh v poskytovaných virtuálnych strojoch špeciálne upravený OS. Príkladom môže byť Xen, alebo UserModeLinux. Kontajnerová virtualizácia slúži hlavne ako prostriedok k lepšiemu oddeleniu služieb na počítači.

Keď sa však budeme zaoberať virtualizáciou ako technológiou, ktorá nám dokáže poskytnúť na požiadanie hardvér, aj keď virtuálny, môžeme abstrahovať od konkrétnej implementácie hosťujúceho systému, a zamerať sa na to ako tento hardvér efektívne využiť. Jedným z problémov, ktoré takáto flexibilita prináša je, že administrátor je oveľa častejšie nútený vytvárať a konfigurovať nové servery.

Kapitola 1

Prehľad rôznych riešení

Problém rýchlej konfigurácie nového serveru (a teda aj virtuálneho stroja) má veľa rôznych riešení. Riešenia sa od seba líšia v miere škálovateľnosti, flexibility a aj zložitosti sprevádzkovania.

1.1 Hotové riešenia

V súčastnosti je na trhu niekoľko firiem, ktoré poskytujú virtuálne zariadenia na kľúč. Suse Linux (ktorý je divíziou firmy Novell) zakladá svoju serverovú stratégiu na veľkej knižnici virtuálnych zariadení ktoré si môže ktokoľvek upraviť cez webové rozhranie SuseBuildStudio[2]. Iné firmy poskytujú virtuálne zariadenia ako jeden z distribučných kanálov pre svoj softvér. Týmto smerom sa vydala Zimbra, ktorá týmto spôsobom poskytuje svoj email-server. Ďalej tu máme veľa menších firiem, alebo komunít, ktoré poskytujú svoje vlastné knižnice virtuálnych zariadení, na báze niektorej populárnej distribúcie. Poslednou kategóriou hotových riešení sú potom špecializované distribúcie, ktoré slúžia nejakému pevne vymedzenému účelu. Najčastejšie sa jedná o firewall distribúcie, ale môže ísť aj o distribúcie poskytujúce rýchle sprevádzkovanie NAS, alebo poskytujúce nástroje na stres testy hardvér.

1.2 Vytváranie vlastného zariadenia

Existuje niekoľko dobrých dôvodov, prečo nepoužiť už hotové zariadenie, ale pripraviť si vlastné.

Prvým je otázka zabezpečenia. Zatiaľ čo pri ručnom nastavení servera administrátorovi stačí dôverovať použitej distribúcii a za bezpečnosť nastavenia berie plnú zodpovednosť, pri použití virtuálneho zariadenia musí dôverovať aj samotným tvorcom zariadenia. Overiť či je softvér správne nastavený je ťažšie, než softvér správne nastaviť.

Použitie predpripraveného nastavenia má aj iné než len bezpečnostné úskalia. Často nemusíte súhlasiť s niektorým z rozhodnutí tvorcov virtuálneho zariadenia. Tí môžu mať určité predpoklady o prostredí, v ktorom bude ich zariadenie bežať, ako napríklad prístup k sieti počas inštalácie, alebo rozhranie s verejnou ip-adresou.

Našťastie, vďaka dostupnosti virtualizačných technológií, nie je dnes veľký problém vytvoriť si vlastné virtuálne zariadenie.

1.2.1 Virtuálny stroj ako šablóna

Najjednoduchším spôsobom ako vytvoriť svoje vlastné virtuálne zariadenie je použiť priamo virtuálny stroj.Pretože virtuálny stroj väčšinou pozostáva len z niekoľkých súborov, je jednoduché nastavený stroj skopírovať na novú lokáciu a spustiť.

Jednou z nevýhod tohto prístupu je, že virtuálny stroj býva často naviazaný na konkrétnu virtualizačnú technológiu. Táto nevýhoda sa prejaví len v prípade, ak používate viacero rôznych technológií, alebo plánujete vaše virtuálne zariadenia publikovať. Väčšou nevýhodou je, že jedna takáto šablóna zaberá rádovo jednotky až desiatky GB. Preto sú takéto šablóny použiteľné prevažne vrámci lokálnej siete, s tým že ich uskladnenie si bude vyžadovať značnú časť diskového priestoru.

Problém so zaberaním miesta sa dá elegantne vyriešiť, ak má obraz disku virtuálneho stroja podporu technológie copy-on-write. Takýto disk vie zdieľať niekoľko virtuálnych strojov, a do separátneho súboru sa ukladajú len zmeny oproti pôvodnej šablóne.

1.2.2 Tvorba vlastného inštalačného média

O trochu náročnejšie na vytvorenie je virtuálne zariadenie vo forme obrazu inštalačného CD. Výhodou je, že väčšina virtualizačných technológií podporuje inštaláciu z bootovateľného .iso obrazu, a teda odpadá problém s prenositeľnosťou medzi platformami. Takéto iso potom viete použiť aj pri inštalácii fyzického serveru. Narozdiel od niekoľkogigabajtových virtuálnych strojov má súbor obsahujúci iso obraz len niekoľko desiatok MB. Preto sa oveľa ľahšie prenáša po sieti.

Väčšie distribúcie majú na vytváranie upravených CD vlastné nástroje. Pre Debian je tu preseed a simple-cdd, Fedora má kickstart a livecd-tools. Kickstart a preseed sú technológie umožňujúce zautomatizovať inštaláciu, ďalšie dva programy slúžia na zostavenie inštalačného CD. Kickstart a livecd-tools okrem Fedory používa aj Red Hat Enterprise Linux a Suse Linux, preto je možné ich používať aj v ďalších distribúciách, ktoré sú z nich odvodené. Ide napríklad o Scientific Linux, alebo CentOS. Podobne je možné preseed a simple-cdd použiť s Ubuntu.

Tento spôsob sme nakoniec zvolili pre našu implementáciu virtuálnych zariadení, presnejší popis použitých technológií nájdete v jednej z nasledujúcich kapitol.

1.2.3 Nasadenie vlastného inštalačného serveru

Inštalačný server je o rád komplikovanejšie riešenie, poskytuje však oveľa väčšiu flexibilitu. Pretože funkčný inštalačný server pozostáva z niekoľkých služieb, ktoré môžu byť poskytované rôznym spôsobom, pre lepšiu predstavu nižšie popíšeme prípadovú štúdiu nastavenia jedného takéhoto riešenia [3]. Prezentované riešenie sa síce zaoberalo fyzickými servermi a pracovnými stanicami, základný problém však rieši rovnaký. Inštalačný server sa skladal z týchto komponent:

- PXE boot server, ktorý má uložené inštalačné médiá, z ktorého bootujú stroje na ktorých bude prebiehať inštalácia. Je automaticky nastavený, aby si spúšťaný server stiahol konfiguráciu pre bezobslužnú inštaláciu.
- Web server poskytujúci konfiguráciu pre bezobslužnú inštaláciu, menovite kickstart pre rodinu distribúcií RedHatu, preseed pre rodinu Debianu a jumpstart pre pracovné stanice Solarisu.
- PuppetMaster server, ktorý umožňuje centrálne spravovať konfiguráciu všetkých strojov na ktorých je nainštalovaný Puppet klient.
- SVN repozitár, v ktorom je uložená konfigurácia PuppetMaster serveru, a teda efektívne uchovávajúci konfiguráciu väčšiny strojov.

Pri sprevádzkovaní nového stroja stačí nabootovať z PXE servera, zvyšok inštalácie sa dokončí automaticky. Cez sieť nabootované inštalačné médium začne bezobslužnú inštaláciu, v ktorej

je nakonfigurovaná aj inštalácia a nastavenie Puppet klienta. Po dokončení inštalácie sa stroj reštartuje, Puppet klient stiahne zvyšok konfigurácie serveru z PuppetMaster servera a donastavuje pod'la nej požadované služby.

Čoraz väčšej popularite sa začína tešit' inštalačný server Cobbler, ktorý okrem PXE bootu dokáže priamo spolupracovat' s niektorými virtualizačnými technológiami. V kombinácii s centrálnou správou konfigurácie, akú poskytuje napríklad Puppet, ide o robustné riešenie. Alternatívou k serveru Cobbler môže byt' ešte projekt FAI - Fully Automatic Installation.

Hlavnou nevýhodou takéhoto riešenia je jeho naviazanost' na infraštruktúru. Pri nutnosti spravovat' veľké množstvo strojov ide asi o najlepšie riešenie, je však problém integrovat' ho do už existujúceho produkčného prostredia.

Kapitola 2

Zásady konfigurácie linuxového serveru

V tejto kapitole sa budeme zaoberať základnými zásadami nastavovaní sieťových služieb na linuxovom serveri. Konkrétna implementácia vo forme virtuálnych zariadení sa nachádza v nasledujúcej kapitole.

Správne nastavená služba má v čo najväčšej miere spĺňať vlastnosti informačnej bezpečnosti[4]:

Dôvernosť (confidentiality) Zaručuje, že v systéme sa k informácii dostane len ten, kto má na danú informáciu oprávnenie.

Integrita (integrity) Zaručuje, že získaná informácia je kompletná a pochádza od očakávaného zdroja.

Dostupnosť (avialability) Zaručuje, že požiadavky na získanie informácie, alebo jej zmenu budú spracované vtedy, keď oprávnený užívateľ potrebuje.

Zodpovednosť (accountability) Zaručuje, že každý zásah v systéme je dosledovateľný k jeho pôvodcovi.

Overiteľnosť (auditability) Zaručuje, že je možné zistiť akým sledom udalostí sa systém dostal do súčasného stavu. Zároveň zaručuje že je možné jednoduchým spôsobom zistiť aktuálny stav systému a porovnať ho so žiadaným stavom v ktorom by sa mal nachádzať.

Ak chceme previesť tieto požiadavky do praxe, musíme dodržiavať niektoré základné bezpečnostné zásady. Pri nastavovaní musí administrátor počítať s možným vonkajším útočníkom, vnútorným útočníkom, neobjavenými zraniteľnosťami v použitých programoch, chybami v hardvére a ľudským faktorom vo forme svojich vlastných chýb, alebo chýb svojich kolegov.

Pretože je týchto faktorov veľké množstvo, je nevyhnutné použiť pri konfigurovaní systému dôveryhodný a dôkladný zdroj informácií. V tejto práci budeme na tento účel používať hlavne Guide to the Secure Configuration of Red Hat Enterprise Linux 5[1] a Securing Debian Manual[5]. Tieto dokumenty sprevádzajú administrátora základnými prvkami zabezpečenia týchto dvoch distribúcií.

Z dokumentov môžeme vybrať všeobecné zásady:

- každý server by mal poskytovať len jednu sieťovú službu
- server by mal mať čo najmenej nainštalovaného softvéru a čo najmenej bežiacich procesov
- pri nastavovaní služieb použiť white-listing všade kde je možné.

- administrátor by mal mať najmenšie privilégiá, ktoré mu stačia na administrovanie serveru
- používateľské účty by mali byť pravidelne kontrolované, vrátane kontroly sily hesiel, a vymazávania nepoužívaných účtov.
- všetky dáta prenášané po sieti by mali byť šifrované.
- logovanie by malo byť nastavené tak, aby boli všetky logy zároveň uložené na separátnom log-serveri.
- mali by byť využité bezpečnostné moduly v jadre systému (Netfilter, SELinux)

Pojem white-listing sme v našej práci použili prvý raz, ide o jednoduchý princíp pri konfigurovaní sieťových služieb, ktorý by sa dal charakterizovať v dvoch bodoch:

- V základnom nastavení služby sú všetci klienti zakázaní.
- Povolíme len tých klientov, ktorí k službe musia pristupovať.

Tieto dve pravidlá sa dajú uplatňovať pri konfigurácii firewall-ov, bezpečnostných politík a do veľkej miery aj pri konfigurácii väčšiny sieťových služieb. Princíp white-listingu je v bezpečnostnej praxi všeobecne známy a je bezpečnejší než black-listing, ktorého základným pravidlom je povoliť všetko a neskôr pozakazovať len "služby"ktoré sú nebezpečné.

V niektorých prípadoch však white-listing nemusí byť praktický a je nutné zvoliť inú stratégiu.Napríklad pri používaní black-listov je možné zoznam nežiadúcich klientov postupne automaticky generovať, na základe ich podozrivého správania. Týmto spôsobom napríklad pracuje program fail2ban.Ten automaticky vytvára black-list ip-adries, ktoré majú priveľa neúspešných pokusov o prihlásenie. Týmto vie účinne zastaviť veľké množstvo brute-force útokov.

Praktickému uplatňovaniu týchto zásad pri nastavovaní jednotlivých služieb sa budeme venovať v nasledujúcej kapitole.

2.1 Infraštruktúrne služby

Pri veľkom množstve strojov je prakticky nevyhnutné mať okolo produkčných serverov postavenú určitú infraštrúkuru. Samotná infraštruktúra môže pozostávať z týchto služieb:

Logovací server na ktorom sú zbierané záznamy všetkých strojov. Mať logy operačného systému uložené na viacerých miestach je jednou z dôležitých bezpečnostných zásad. Tie potom môžu slúžiť na zistenie bezpečnostnej diery v prípade úspešného útoku, alebo pri zisťovaní príčiny havarijného stavu, pretože práve v týchto prípadoch sa často stáva, že pôvodné logy nebudú v konzistentom stave.

DNS server ktorý umožňuje preklad medzi doménovými menami a ip-adresami strojov. Nasadzovať vlastný DNS server má svoje pre aj proti. Často je možné na verejné služby použiť už existujúci DNS, a na vašej vnútornej sieti používať na identifikáciu priamo ip-adresy. Populárnym útokom na sieťové služby je práve sfalšovanie prekladu medzi doménovým menom a ip-adresou, preto je používanie pevne definovaných ip-adries bezpečnejšie.

Výhodou DNS systému je väčšia flexibilita, napríklad pri reorganizácii siete stačí meniť záznamy v DNS a nie je treba prepisovať veľké množstvo napevno zadaných ip-adries.Tiež je prívetivý pre administrátorov.Často sa inštaluje v kombinácii s DHCP serverom.

Centrálna správa certifikátov poskytuje služby na vytváranie, uchovávanie a revokovanie elektronických kľúčov a certifikátov. Zabezpečenie veľkého množstva služieb (napr. DNS, LDAP, SSL, a i.) závisí od správne rozdistribuovaných elektronických certifikátov. Navyše je nutné uchovávať kľúče jednotlivých strojov a kľúče užívateľov ktoré môžu používať na svoju autentifikáciu.

Centrálna správa užívateľov Podobne ako je od určitého množstva serverov nepraktické udržiavať na každom stroji separátny hosts súbor, môže byť od určitého množstva užívateľov nepraktické udržiavať separátne passwd súbory. Podobne ako na distribúciu doménových mien uchovávaných v súbore hosts máme DNS protokol, na distribúciu užívateľov na viaceré stroje exituje sa väčšinou používa protokol LDAP. Samotných implementácií však existuje niekoľko. Momentálne najpoužívanejší je Active Directory server od firmy Microsoft, ktorý kombinuje adresárový server prístupný cez LDAP s nástrojom na jednotné prihlasovanie prístupné cez protokol Kerberos. Samotné protokoly majú aj svoje open-source implementácie (OpenLDAP, 389 Directory Server pre LDAP a MIT Kerberos pre kerberos).

Server na sledovanie systémov umožňuje sledovať dostupnosť rôznych služieb. Aj pri, čo do rozsahu, menšom nasadení je vhodné mať k dispozícii službu ktorá vás automaticky upozorní, ak niektorá zo služieb prestane fungovať, alebo sa začne správať neštandardne. Napríklad neočakávaný vysoký prenos dát môže signalizovať útok na niektorý zo strojov.

Inštalačný server ktorý umožňuje rýchle nastavenie nového stroja, alebo reinštaláciu starej služby. V predchádzajúcej kapitole sme ho už opisovali, ako jeden z možných spôsobov na vytváranie virtuálnych zariadení.

Centrálna databáza konfigurácie ktorá uchováva konfigurácie pre rôzne servery. Implementácia môže byť rôzna, niektoré služby (napr. bind DNS server, postfix mail server) dokážu brať konfiguráciu aj z databáz, alebo adresárových serverov (ako napríklad OpenLDAP), dajú sa však použiť aj všeobecnejšie riešenia, ktoré pracujú priamo s textovou konfiguráciou.

Infraštruktúra môže byť nasadenie od nasadenia rôzna, pri malom množstve serverov môže stačiť logovací server.Jednou zo služieb ktorú sme tu nespomenuli, a patrí medzi dôležitú časť infraštruktúry, je sieťové úložisko. Pretože sa zameriavame na virtualizované prostredie, predpokladáme, že prístup k diskovému priestoru za nás vyrieši nami použitá virtualizačná technológia.

2.2 Zálohy dát a obnova po katastrofe

Pri jednotlivých službách je tiež nevyhnutné mať systém záloh dát aj konfigurácie. Tieto zálohy pomáhajú pri audite, ako aj pri obnovách po nepredpokladanom zlyhaní systému. Pretože sa pohybujeme vo virtualizovanom prostredí, je vhodné zistiť, či použitá virtualizačná technológia nemá v sebe priamo integrovaný systém na zálohy. Ďalším spôsobom záloh môže byť skopírovanie celého virtuálneho stroja, toto však nebýva praktické vzhľadom na vysoké nároky na pamäť. Najlepšou možnosťou je podľa našich skúseností zálohovať na aplikačnej vrstve. Takáto záloha je väčšinou niekoľkokrát menšia než kópia celého virtuálneho stroja a tie nie je závislá na konkrétnej virtualizačnej technológii.

2.3 Vysoká dostupnosť

Dostupnosť (aviability) dát patrí medzi požadované vlastnosti informačnej bezpečnosti[4]. Preto je dôležité uvažovať nad tým, ako zaručiť dostupnosť služieb aj v prípade výpadku niektorého zo strojov.

Najčastejším riešením je zostaviť každú zo služieb ako dvojicu strojov, jeden primárny, na ktorom pobeží služba a druhý sekundárny, na ktorý sú zrkadlené dáta primárneho stroja. Sekundárny stroj potom vie pomáhať pri balancovaní záťaže a v prípade poruchy primárneho stroja ho dokáže zastať. Momentálne, aj napriek tomu, že používame virtuálny hardvér, nie je možné týmto spôsobom sprevádzkovať ľubovoľný stroj. Spolupráca medzi primárnym a sekundárnym serverom sa odohráva na aplikačnej vrstve. Preto ak chceme takúto živú zálohu prevádzkovať, je nutné už pri návrhu služby s takýmto nasadením počítať.

Niektoré služby, ako napríklad DNS, Kerberos, alebo LDAP, majú schopnosť pracovať v dvojici master-slave server priamo zabudovanú. Pri master-slave implementácii vysokej dostupnosti obvykle platí, že slave server neposkytuje plnú funkcionalitu. Napríklad u serveru Kerberos je možné meniť databázu len v prípade, že je zapnutý master server, oproti slave serveru beží len čítanie. U LDAP je možné vytvoriť spoluprácu na úrovni master-master, pri ktorej sú oba previazané servery rovnocenné. U iných služieb je to otázkou konkrétnej implementácie, niektoré spoluprácu podporujú, iné nie (toto je prípad logovacieho servera).

Netreba tiež zabúdať, že vysoká dostupnosť a zálohovanie dát sú dve odlišné veci. Technológie vysokej dostupnosti sa väčšinou používajú len na niekoľkých kritických systémov, kde je žiadúce aby služba bežala za každých okolností. To je prípad služieb ako je DNS, alebo služby centrálnej správy užívateľov.

Kapitola 3

Implementácia sady virtuálnych zariadení

3.1 Spôsob implementácie

Na implementáciu našej sady virtuálnych zariadení sme si vybrali linuxovú distribúciu Debian, ktorá má v serverovom odvetví dobrú reputáciu, hlavne pre svoju stabilitu. Výsledné virtuálne zariadenia sú vo forme inštalačných CD, zostavené cez nástroj simple-cdd. Takto vytvorené CD potom automaticky inštaluje minimálny systém, na ktorom potom spustí post-inštalačný skript. Ten dokončí konfiguráciu služieb.

3.1.1 SimpleCDD

Časť konfigurácie prebieha priamo počas inštalácie pomocou súboru default.preseed[12]. Ten obsahuje odpovede na otázky inštalátora ohľadom nastavenia nových partícií, alebo inštalovaných balíkov.

Listing 3.1: default.preseed

```
...
#particie
d-i partman-auto/method string lvm
d-i partman-lvm/device_remove_lvm boolean true
d-i partman-md/device_remove_md boolean true
d-i partman-lvm/confirm boolean true
d-i partman-partitioning/confirm_write_new_label boolean true
d-i partman/choose_partition select finish
d-i partman/confirm boolean true
d-i partman/confirm_nooverwrite boolean true
d-i partman-lvm/confirm_nooverwrite boolean true
...
```

Inštalátor je nastavený tak, aby automaticky premazal prvý harddisk inštalovaného počítača a vytvoril päť partícií pre systémové adresare /var, /usr, /tmp,/home a pre koreŇový adresár /. Toto je výhodné hlavne preto, že na jednotlivé adresáre teraz vieme uplatnit' niektoré nastavenia, ktoré by neboli možné ak by sa nenachádzali na separátnych partíciách[1].

[1] napríklad nodev a noexec pre partíciu hostiacu /tmp

Listing 3.2: default.preseed

```
...
d-i preseed/late_command string
 mkdir /target/tmp/postinst;
 cp /cdrom/simple-cdd/* /target/tmp/postinst/ ;
 chmod 755 /target/tmp/postinst/security.sh ;
 in-target /tmp/postinst/security.sh
```

Vd'aka poslednému riadku v default.preseed inštalátor spustí post-inštalačný skript security.sh.

Aj napriek tomu, že výsledkom našej práce sú práve inštalačné cd, po stiahnutí konfigurácie k simple-cdd nieje problém ich d'alej upravovat' a vytvárat' nové zariadenia.

Listing 3.3: saleh-cdd-conf.tar

```
ca-simple-cdd                dns2-simple-cdd
dns-simple-cdd               ldap2-simple-cdd
ldap-simple-cdd              mysql-simple-cdd
postfixadmin-simple-cdd      postfix-simple-cdd
roundcube-simple-cdd         syslog-simple-cdd
wordpress-simple-cdd
```

Inštalačné cd je vytvorené spustením príkazu build-simple-cdd v adresári niektorého zo zariadení. Konfigurácia zariadenia sa nachádza v podadresári profiles/ a výsledný iso-obraz je vytvorený v adresári images/.

3.1.2 Implementované zariadenia

Implementované virtuálne zariadenia sme si rozdelili na dve kategórie, na kategóriu produkčných a infraštruktúrnych serverov.

Produkčné servery budú pokrývat' tri zariadenia: Web Server, Mail Server a SQL Server. Web server je momentálne asi najčastejšie nasadzovaný server na sieti, a pričom HTTP protokol sa stal univerzálnym protokolom, na ktorom stavajú mnohé d'alšie. Približne polovica virtuálnych zariadení ktoré sú k dispozícii na TurnKeyLinux.org sú web-servermi, preto sa týmto virtuálnym zariadeniam budeme venovat' do väčšej hĺbky.

Mail server je zaujímavou službou na spracovanie hlavne preto, že ide o niekoľko na sebe nezávislých ale spolupracujúcich serverov. Tiež je zaujímavé riešit' problém zabezpečenia a ochrany pred nevyžiadanou poštou.

Infraštruktúrne servery budú slúžit' ako podpora pri udržiavaní produkčných serverov v chode. Napriek tomu, že budeme implementovat' len dva druhy produkčných serverov, pri nasadení môžu bežat' až desiatky ich inštancií. Z infraštruktúrnych serverov budeme implementovat' Logserver, DNS, Centrálnu správy identít, Centrálnu správu certifikátov a Sledovanie systémov.

3.2 Základný systém

Najprv predstavíme konfiguráciu základného systému, ktorú budú v praxi zdielat' všetky ostatné zariadenia. V podstate pôjde o určitú šablónu.

3.2.1 Firewall

Na nastavenie firewall-u používame vlastný init.d skript. V základnom nastavení zakážeme všetky spojenia z vonku, všetky prechádzajúce spojenia (je nepravdepodobné, že by malo vir-

tuálne zariadenie slúžit' ako router) a povolíme všetky spojenia zvnútra von.

Listing 3.4: iptables

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

V prípade serveru je možné zvýšit' zabezpečenie tým, že povolíte len špecifické odchádzajúce spojenia (napríklad len spojenie na server poskytujúci bezpečnostné updaty). Rozhodli sme sa, že použit' takýto white-list na odchádzajúce spojenia by bolo nepraktické, vzhl'adom na to, že sa naše virtuálne zariadenia budú pripájat' na rôzne infraštruktúrne služby.

3.2.2 SELinux

Modul SELinux pre Linux priniesol podporu pre Mandatory Access Control, čo by sme mohli preložit' ako „vynútenú kontrolu prístupu". Jedná sa o bezpečnostný mechanizmus, ktorý obmedzuje prístup procesov len k súborom na ktoré majú právo a možnost' vykonávat' len určitú množinu činností. Navyše je tento systém silnejší než tradičný Unixový model práv, ktorý umožňuje vlastníkom objektov menit' ich prístupové práva. Pretože SELinux je nezávislý od Unixového modelu, dokáže systém často ochránit' pred zneužitím bezpečnostných chýb, ktoré by umožnili neoprávnené zvýšenie privilégií[13].

Debian má základnú politiku pre SELinux v jednom z balíčkov, preto sme ho nainštalovali a nastavili do enforcing módu. Táto politika je celkom dobre nastavená pre aplikáciu na server, preto ju nieje nutné upravovat'.

3.2.3 Sudo

Jedným zo základných bezpečnostných nastavení na moderných linuxových distribúciách je zablokovanie root účtu. Administrátor potom k privilegovaným aplikáciám pristupuje cez aplikáciu sudo[14]. Vo väčšine distribúcií je cez sudo skupine užívateľov (najčastejšie administrators, sudoers, alebo wheel) daný skrze sudo prístup ku kompletne celému systému, preto aj skratka sudo znamená superuser do. Je možná aj oveľa jemnejšia konfigurácia. Sudo je možné nastavit' tak, aby mal cez jeho použitie užívateľ možnost' spúšt'at' špecifické príkazy ako iný užívateľ, pričom uchováva informácie o pôvodnom užívateľovi, ktorý príkazy zadal.

V základom systéme je ponechaná skupina sudoers s plným prístupom k root užívateľovi cez sudo, a prvý vytvorený užívateľ je automaticky do tejto skupiny pridaný.

3.2.4 SSH

SSH server je jedinou spustenou službou v základnom systéme. Pretože umožňuje priamy prístup ku konzole virtuálneho zariadenia, je nevyhnutné na túto službu aplikovat' určité základné bezpečnostné opatrenia.

V konfiguračnom súbore sshd_config zakážeme používanie rhost súborov, autentifikáciu na základe identity vzdialeného počítača, prihlasovanie na účet root-a a prihlasovanie prázdnym heslom.

Listing 3.5: /etc/ssh/sshd config

```
IgnoreRhosts yes
HostbasedAuthentication no
PermitRootLogin no
PermitEmptyPasswords no
```

Aj napriek tomu, že je účet root-a zablokovaný je vhodné zakázať ho aj v nastavení SSH servera,pretože práve na root účet je vedené veľké množstvo brute-force útokov.Ak na server pristupuje obmedzené množstvo ľudí, ktoré nebude treba meniť, môže byť vhodné zakázať prístup na heslo úplne, a povoliť len autentifikáciu cez užívateľské digitálne certifikáty[1].

Listing 3.6: /etc/ssh/sshd config

```
RSAAuthentification yes
PubkeyAuthentification yes
PasswordAuthentification no
```

3.2.5 Syslog

Štandardné nastavenie logov je dostatočné, jediné čo chýba je zadefinovať vzdialený log-server. Na logovanie je na strane klienta aj serveru použitý rSyslog[15], ktorý okrem toho, že pracuje cez TCP spojenie, podporuje šifrovaný prenos a logovanie na niekoľko log-serverov zároveň.

Listing 3.7: /etc/rsyslog.conf

```
*.* @@syslog.infrastrructure.local
```

3.2.6 Sledovanie systémov

Okrem logovania je vhodné nastaviť aj procesy sledovacích služieb, ako napríklad SNMP daemon, alebo Munin-node. Nastavovaním týchto služieb sa budeme zaoberať pri nastavovaní zariadenia Sledovania Systémov, základný systém ich však nebude obsahovať vôbec.

Sledovanie systémov prostredníctvom Nagios dá pre väčšinu prípadov rozumne zabezpečiť "zvonku",dotazovaním sa na existujúce služby, ako je napríklad SSH, alebo pomocou požiadavky na ICMP ECHO, napsíklad prostredníctvom programu ping. Nieje preto potrebné inštalovať na sledovaný systém ďalší softvér, alebo dokonca počúvajúci server (ako v prípade SNMP).

3.2.7 Autentifikácia a PAM

Autentifikáciu na Linuxových systémoch má na starosť modul PAM[6]. PAM slúži na centrálnu správu autentifikácie a podporuje ho veľké množstvo linuxových programov. Okrem overovania cez LDAP má k dispozícii aj možnosť autentifikovať sa cez Kerberos, Samba server, RADIUS server, alebo Active Directory.

V prípade, že bude stroj budeme autentifikovať cez LDAP (takto to bude nastavené na produkčných serveroch), sú v konfigurácii PAM-u pridané tieto nastavenia:

Listing 3.8: /etc/pam.d/common-account

```
account     required      pam_unix.so
account     [default=bad success=ok user_unknown=ignore] pam_ldap.so
account     required      pam_permit.so
```

Listing 3.9: /etc/pam.d/common-auth

```
auth    sufficient      pam_unix.so nullok_secure
auth    sufficient      pam_ldap.so use_first_pass
auth    required        pam_deny.so
S
```

S týmto nastavením je možné autentifikovať sa aj bez prístupu k ldap serveru, cez Štandardné súbory passwd a shadow.

Listing 3.10: /etc/pam.d/common-password

```
password    required    pam_cracklib.so retry=3 minlen=8 difok=4
password    sufficient  pam_unix.so md5 obscure nullok try_first_pass
password    sufficient  pam_ldap.so
password    required    pam_deny.so
```

Okrem samotnej autentifikácie je v PAM-e možné nastaviť aj základné kritériá na kontrolu kvality nových hesiel. Napríklad prvý riadok predchádzajúceho konfiguračného súboru zaručuje, že každé nové heslo bude mať aspoň osem znakov a bude sa líšiť sa od predchádzajúceho hesla aspoň v štyroch.

Listing 3.11: /etc/pam.d/common-session

```
session     required    pam_limits.so
session     required    pam_unix.so
session     optional    pam_ldap.so
```

V tejto časti konfigurácie sa nikde nezmieňuje adresa servera, o ktorý sa chceme autentifikovať. Tú nastavíme v ldap.conf, v kde sa nachádza centrálna konfigurácia pre pripájanie rôznych služieb na LDAP server.

Listing 3.12: /etc/ldap.conf

```
uri ldap://ldap.infrastructure.local:389 \
    ldap://ldap2.infrastructure.local:389
base dc=infrastructure,  dc=local
# ...
```

Okrem adresy servera musíme špecifikovať aj koreňový uzol adresárovej databázy. My sme zvolili doménu infrastructure.local a takto sme nastavili aj ldap-server v zariadení Centrálnej Správy Identít. Koreňový uzol celej LDAP databázy preto bude „dc=infrastructure,dc=local“.

Zároveň je vhodné zadať aj sekundárny ldap server.

3.3 Centrálna správa certifikátov

Veľké množstvo protokolov aplikačnej vrstvy sa nezaoberá bezpečnosťou prenášaných dát. Napríklad HTTP prenáša všetky svoje dáta v otvorenej textovej forme, takže nemôže zaručiť dôvernosť, ani autenticitu prenášaných dát. Pretože chceme cez HTTP prenášať citlivé dáta, potrebujeme tieto dáta nejakým spôsobom chrániť. Z tohto dôvodu vznikol protokol HTTPS, ktorý kombinuje HTTP protokol na aplikačnej vrstve so šifrovaním cez SSL protokol.

SSL je protokol, ktorý zabezpečuje autentifikáciu a šifrovanie dát na úrovni transportnej vrstvy. Vytvorenie takéhoto bezpečného autentifikovaného kanálu prebieha takto[16]:

1. Klient pošle serveru požiadavku na vytvorenie bezpečného spojenia.
2. Server pošle klientovi digitálny certifikát so svojím podpísaným verejným kľúčom.
3. Klient skontroluje podpis na certifikáte, a ak vyhodnotí certifikát za dôveryhodný, vygeneruje náhodné číslo, zašifruje ho verejným kľúčom získaným z certifikátu a pošle naspäť serveru. K tomuto číslu má prístup len klient a server.

4. V tomto momente majú klient aj server k dispozícii rovnaké tajné číslo, z ktorého môžu zostrojiť symetrický kľúč na ďalšiu šifrovanú komunikáciu.

Bezpečnosť tohto spojenia závisí od správne nastaveného certifikátu na serveri a schopnosti klienta overiť jeho autentickosť. Tá je väčšinou zaručená podpisom certifikátu dôveryhodnou treťou stranou, ktorá sa nazýva Certifikačná Autorita.

Okrem HTTP, má svoju SSL protokolom zabezpečenú variantu aj protokol IMAP, POP3, SMTP, FTP, LDAP a mnohé ďalšie. Digitálne certifikáty tiež využívajú aj služby mimo SSL. Napríklad protokol SSH dokáže použiť certifikáty na autentifikáciu užívateľa pri prihlasovaní sa na server.

Pre naše účely stačí, ak certifikačná autorita poskytuje tieto služby:

- vydávanie a revokáciu digitálnych certifikátov pre servery
- sprístupnenie certifikátu certifikačnej autority a zoznamu revokovaných certifikátov

3.3.1 OpenSSL

V Debiane prevádzkovanie jednoduchej certifikačnej autority stačí mať nainštalovanú knižnicu openssl. Tá sama o seba poskytuje všetky nástroje potrebné na manažovanie certifikátov. Na uľahčenie práce s certifikátmi má OpenSSL knižnica v Debiane pribalený skript /usr/lib/ssl/misc/CA.pl. Pred samotným použitím je potrebné skript nastaviť[8]:

Listing 3.13: CA.pl

```
...
$CATOP="/var/ssl";
...
```

Listing 3.14: openssl.cnf

```
[ CA_default ]
dir = /var/ssl # Where everything is kept
```

Bez tohto nastavenia by príkazy certifikačnej autority vždy pracovali nad aktuálnym adresám, čo by viedlo k zbytočnému zmätku. Pre urýchlenie práce s certifikátmi je navyše možné zmeniť prednastavené hodnoty v konfiguračnom súbore openssl.cnf.

Listing 3.15: openssl.cnf

```
countryName\_default             = sk
stateOrProvinceName\_default    = Slovensko
localityName\_default           = Bratislava
0.organizationName\_default     = Saleh
organizationalUnitName\_default = Adam
```

Teraz máme vytvorené jednoduché rozhranie k samotnej knižnici openssl. K dispozícii máme tieto základné príkazy.

CA.pl -newca Vytvorí novú certifikačnú autoritu. Pokiaľ už existuje zložka /var/ssl skončí bez zásahu do systému. Pokiaľ zložka doteraz neexistovala vytvorí v nej jednoduchú adresárovú štruktúru na uchovávanie certifikátov. Potom vygeneruje potrebné certifikáty. Samotný privátny kľúč je chránený heslom. Z ostatných súborov v adresári je dôležitý najmä cacert.pem obsahujúci koreňový certifikát.

CA.pl -newreq Vytvorí žiadosť o nový certifikát. Na vyplnenie tejto žiadosti nie je treba mať prístup k privátnemu kľúču certifikačnej autority. Výstupom príkazu sú dva súbory, newkey.pem obsahujúci privátny kľúč a newreq.pem obsahujúci samotnú žiadosť o certifikát. Žiadosť teda môžeme vytvoriť priamo na serveri na ktorom budeme certifikát používať, a na certifikačný server stačí odoslať súbor newreq.pem.

CA.pl -sign Zo žiadosti vytvorí podpísaný certifikát. Na vytvorenie stačí mať v adresári odpovedajúci súbor newreq.pem a poznať heslo ku kľúču certifikačnej autority. Vytvorený certifikát je uložený v súbore newcert.pem

openssl ca -revoke certificate.crt Revokuje certifikát cerificate.crt.

openssl ca -gencrl -out ca.crl Vydá zoznam revokovaných certifikátov a uloží do súboru ca.crl.

Na distribúciu koreňového certifikátu a revokačného zoznamu sme použili web-server lighttpd. Ide o server, ktorý má oproti štandardne používanému Apache oveľa menšie systémové nároky. Na prenos samotný používame https, pri nastavení https je však lepšie nepoužiť priamo certifikát podpísaný našou certifikačnou autoritou. Pri prvom pripojení by musel si klient našu certifikačnú autoritu nemá ako overiť, a musel by ju akceptovať naslepo. Preto je lepšie investovať do certifikátu od verejnej certifikačnej autority, ktorej koreňový certifikát je súčasťou väčšiny webových prehliadačov.

Ďalšou možnosťou je distribuovať koreňový certifikát privátnej certifikačnej autority nejakým iným bezpečným spôsobom, alebo poskytnúť možnosť si autenticitu certifikátu overiť.

3.3.2 Po inštalácii

Po nainštalovaní inštalačného cd sa zariadenie správa ako certifikačná autorita pre firmu Adam Saleh. Toto je možné zmeniť v konfiguračnom súbore /etc/ssl/openssl.cnf a spustením príkazu „/usr/lib/ssl/misc/CA.pl -newca“.

Na serveri je spustený minimálny web server ktorý zobrazuje obsah adresára /var/www. Koreňový certifikát aj revokačný zoznam je doň nutné skopírovať ručne.

3.4 DNS server

Druhým dôležitým infraštruktúrnym zariadením je DNS server. DNS zabezpečuje preklad medzi doménovými menami a IP-adresami jednotlivých strojov. V našom DNS zariadení sa budeme zaoberať špeciálnou doménou infrastructure.local, ktorá v sebe bude uchovávať všetky ip adresy našich infraštruktúrnych strojov. Týmto spôsobom môžeme lepšie zautomatizovať konfiguráciu produkčných strojov.

Produkčný stroj (napríklad web-server) môže takto predpokladať že syslog server sa bude nachádzať na adrese log.infrastructure.local, primárny LDAP server na adrese ldap.infrastructure.local, sekundárny LDAP server na adrese ldap2.infrastructure.local a certifikačná autorita na adrese ca.infrastrucutre.local. Pretože od funkčného DNS bude závisieť správne fungovanie väčšiny služieb, budeme potrebovať aj sekundárny DNS server, pre prípad, že by bol hlavný DNS server mimo prevádzky. Ako DNS server sme použili Bind9.

3.4.1 Chroot Bind9

Na zvýšenie bezpečnosti nášho serveru sme sa rozhodli umiestiť ho do chroot-ovaného prostredia[1]. Proces v chroot-ovanom prostredí má k dispozícii len obmedzený pohľad na filesystém, čo sťa-

žuje prácu prípadnému útočníkovi. Bind9 dokáže sám seba spustiť v chroot-e, predtým ale musíme pripraviť adresár, ktorý sa stane novým koreňovým adresárom pre náš DNS server.

Listing 3.16: chroot-bind.sh

```
mkdir -p /var/lib/named/etc
mkdir /var/lib/named/dev
mkdir -p /var/lib/named/var/cache/bind
mkdir -p /var/lib/named/var/run/bind/run
mv /etc/bind /var/lib/named/etc
ln -s /var/lib/named/etc/bind /etc/bind
mknod /var/lib/named/dev/null c 1 3
mknod /var/lib/named/dev/random c 1 8
chmod 666 /var/lib/named/dev/*
chown -R bind:bind /var/lib/named/var/*
chown -R bind:bind /var/lib/named/etc/bind
```

Náš chroot pre Bind9 sa bude nachádzať v /var/lib/named, ktorý sa stal koreňom nášho minimálneho adresárového stromu. Jednou z nevýhod tohoto chrootu je, že v ňom musíme vytvoriť zariadenia /dev/null a /dev/random, a teda partícia na ktorej sa náš chroot nachádza nesmie byť pripojená s prepínačom nodev. Inak nás chroot v podstate neobmedzuje, DNS server je stále manažovaný z /etc/init.d/bind9 a vďaka symbolickému odkazu môžeme nastavenia editovať v známom /etc/bind. Pred spustením v chroote sme ešte upravili nastavenie rsyslogu, aby s ním dokázal DNS server komunikovať aj po uskutočnení chrootu.

Listing 3.17: chroot-bind.sh

```
echo "$AddUnixListenSocket /var/lib/named/dev/log" >
  /etc/rsyslog.d/bind-chroot.conf
```

Nakoniec sme DNS serveru nastavili adresu chroot adresára.

Listing 3.18: etc/default/bind9

```
OPTIONS="-u bind -t /var/lib/named"
```

Teraz sa po príkaze "/etc/init.d/bind9 start"spustí DNS s prístupom len k minimu súborov.

3.4.2 Ďalšie zabezpečenie

DNS server vieme ďalej obmedziť na pôsobenie len na lokálnej sieti. Toto sa dá ako obmedzením na úrovni firewallu, tak aj na úrovni samotného DNS serveru.

Listing 3.19: /etc/init.d/firewall

```
iptables -A INPUT -p udp -s 192.168.201.0/24
    -d 0/0 --destination-port 53 -j ACCEPT
```

Listing 3.20: /etc/bind/named.conf

```
acl internal{
    192.168.201.0/24;
    localhost;
};
```

```
options{
        allow-query { internal; };
};
```

Týmto nastavením vieme obmedziť odpovedanie DNS serveru len na dotazy prichádzajúce z lokálnej siete (v príklade 192.168.201.0/24).

Ak je na sieti nutné prevádzkovať nielen infraštruktúrny, ale aj verejný DNS server, správne by mali bežať na dvoch odlišných, navzájom nesúvisiacich strojoch[1]. Preto sa tu nebudeme zaoberať nastavovaním views, ktoré umožňujú z jedného servera poskytovať rôzne odpovede pre dotazy z rôznych sietí.

Transferu dátz primárneho na sekundárny server sa budeme zaoberať v jednej z následujúcich sekcií.

3.4.3 Infraštruktúrna Zóna

V základnej inštalácii už máme nastavené dáta našej domény infrastructure.local. Administrátor, ktorý sa rozhodne použiť toto naše zariadenie bude musieť v doprednej aj reverznej zóne zmeniť ip-adresy jednotlivých strojov.

Listing 3.21: /etc/bind/zones/infrastructure.local

```
@    IN      SOA     ns.infrastructure.local.\
                         hostmaster.infrastructue.local. (
                     199802151       ; serial, todays date + todays serial #
                     8H              ; refresh, seconds
                     2H              ; retry, seconds
                     4W              ; expire, seconds
                     1D )            ; minimum, seconds
;            NS      ns              ; Inet Address of name server
;
localhost    A~127.0.0.1
ns           A~192.168.201.104
ns2          A~192.168.201.105
ldap         A~192.168.201.107
ldap2        A~192.168.201.108
syslog       A~192.168.201.112
ca           A~192.168.201.113
```

Okrem doprednej zóny je dôležité mať správne zadefinovanú aj reverznú zónu, ktorá zabezpečuje preklad ip-adries naspäť na doménové meno.

Listing 3.22: /etc/bind/zones/rev.infrastructure.local

```
$TTL 3D
@    IN      SOA     ns.infrastructure.local.\
                         hostmaster.infrastructure.local. (
                     199802151 ; Serial, todays date + todays serial
                     8H      ; Refresh
                     2H      ; Retry
                     4W      ; Expire
                     1D)     ; Minimum TTL
                 NS      ns.infrastructure.local.
```

```
104             PTR     ns.infrastructure.local.
105             PTR     ns2.infrastructure.local.
107             PTR     ldap.infrastructure.local.
108             PTR     ldap2.infrastructure.local.
112             PTR     syslog.infrastructure.local.
113             PTR     ca.infrastructure.local.
```

Po inštalácii zariadenia z CD má DNS server v konfigurácii práve túto zónu, ktorá počíta s tým, že servery sa nachádzajú na podsieti 192.168.201.0/24. Tá pravdepodobne nebude vyhovovať, preto je nutné upraviť ko oba zónové súbory, tak aj konfiguračný súbor „/etc/bind/named.conf“. Po úprave je dôležité zvýšiť sériové číslo v oboch zónach (v príklade 199802151) a prikázať serveru znovu načítať svoje dáta príkazom „/etc/init.d/bind9 reload“.

3.4.4 Sekundárny Server

Sekundárny DNS server sme nastavili tak, aby si vždy keď sa na primárnom serveri objavia, stiahol nové dáta zone-transfer protokolom. To, že sú dáta nové, spozná sekundárny DNS server podľa sériového čísla na zóne primárneho serveru.

Listing 3.23: /etc/bind/named.conf

```
zone "ns.infrastructure.local" {
        type slave;
        file "/etc/bind/zones/infrastructure.local";
        masters { 192.168.201.104; };
        allow-transfer { 192.168.201.104; };
};
zone "168.192.IN-ADDR.ARPA" {
        type slave;
        file "/etc/bind/zones/rev.infrastructure.local";
        masters { 192.168.201.104; };
        allow-transfer { 192.168.201.104; };
};
```

Pri komunikácii medzi primárnym a sekundárnym DNS serverom sú prenášané dáta podpísané, čím vieme overiť, či prijímané dáta skutočne pochádzajú od primárneho serveru. V menšej miere je možné dosiahnuť podobný efekt kontrolou ip-adresy, tú je však možné sfalšovať.

Preto sme sa rozhodli podpísať zónu na primárnom DNS[1]. To je možné vďaka technológii TSIG, kde primárny aj sekundárny server poznajú rovnaký tajný kľúč a pomocou neho vie sekundárny server skontrolovať či prijímané dáta neprišli z nejakého iného zdroja. Existuje aj varianta protokolu kožívajúca súkromný a verejný kľúč, vzhľadom na to, že oba servery budú pracovať len na jednej podsieti, stačí nám jednoduchšie riešenie s len jedným kľúčom. Kľúč môžeme vygenerovať pomocou programu dnssec-keygen.

Listing 3.24: generate-dnskey.sh

```
cd /tmp
dnssec-keygen -a HMAC-MD5 -b 128 -n HOST ns.infrastructure.local
cat Kdns.infrastructure.local*.key
```

Výstupom potom bude ret'azec
„ns.infrastructure.local IN KEY 512 3 157 sbcERX3rgM6N4d7Mfcr1GQ==“,

kde sbcERX3rgM6N4d7Mfcr1GQ== je 64bit kľúč. Konfigurácia zabezpečujúca podpisovanie na strane primárneho servera potom bude vyzerať takto:

Listing 3.25: /etc/bind/named.conf

```
key zone-transfer-key {
  algorithm hmac-md5;
  secret "sbcERX3rgM6N4d7Mfcr1GQ==";
};
zone "infrastructure.local" IN {
  type master;
  allow-transfer { key zone-transfer-key; };
  ...
};
```

Podobne bude vyzerať konfigurácia na sekundárnom serveri:

Listing 3.26: /etc/bind/named.conf

```
key zone-transfer-key {
  algorithm hmac-md5;
  secret "sbcERX3rgM6N4d7Mfcr1GQ==";
};
server IP-OF-MASTER {
  keys { zone-transfer-key; };
};
```

Vďaka tomuto nastaveniu budú spolu komunikovať len také DNS servery, ktoré budú mať v konfigurácii uložený rovnaký tajný kľúč.

3.4.5 Po inštalácii

Práve po nainštalovaní DNS serverov je potrebné na ich sprevádzkovanie vykonať asi najviac dodatočných úkonov v porovnaní s ostatnými virtuálnymi zariadeniami. Nainštalovaný DNS server síce má nastavený chroot, aj ukážku nastavených zón a ich transferu. Konfiguráciu je teda samozrejme nutné upraviť na parametre siete, v ktorej sa servery nachádzajú. Na primárnom serveri ide o tieto úpravy:

1. v súbore /etc/bind/named.conf.local, kde je nutné v direktíve pre reverznú zónu zmeniť sieť (201.168.192.in-addr.arpa).

2. v súboroch /etc/bind/zones/*, kde sú uložené prekladové tabuľky medzi ip-adresami a doménovými menami.

3. ak nebude nainštalovaný sekundárny server, tak v /etc/bind/named.conf.local je zbytočná konfigurácia prenosu zón. Ide o direktívy allow-transfer, allow-notify,a celá konfigurácia pred definíciou zón.

4. ak bude nainštalovaný sekundárny server, tak v /etc/bind/named.conf.local je nutné v direktíve acl, server a allow-notify ip-adresu sekundárneho serveru. Tiež sa tu nachádza tajný kľúč. Pretože kľúč v konfigurácii nieje generovaný pri inštalácii, je spolu so serverom nainštalovaný aj balík dnssec-tools, aby sa dal vygenerovať nový kľúč pomocou dnssec-keygen.

Podobne je nutné vykonať niekoľko úprav aj na sekundárnom serveri.

1. v súbore /etc/bind/named.conf.local, kde je nutné v direktíve pre reverznú zónu zmeniť sieť (201.168.192.in-addr.arpa).

2. ak bude nainštalovaný sekundárny server, tak v /etc/bind/named.conf.local je nutné v direktíve acl, server a allow-notify ip-adresu primárneho serveru. Tiež jesa tu nachádza tajný kľúč. Ten sa musí zhodovať s kľúčom na primárnom serveri, inak transfer nebude fungovať.

3.5 Syslog server a sledovanie systémov

Mať prístup k logom servera je dôležité hlavne v prípade zlyhania niektorej zo služieb. Logy by administrátor mal kontrolovať pravidelne, nie len v prípade, že niečo nefunguje. Tým vie často krát problémom predísť, prípadne zistiť či sa v systéme nedeje niečo nekalé.

3.5.1 rSyslog

Ako logovací daemon sme sa rozhodli použiť rsyslog[15]. Narozdiel od pôvodného syslog-u používa na prenášanie spojení po sieti TCP, je menej náročný ako syslog-ng a navyše je prítomný v základnej inštalácii Debianu. V konfigurácii sme nastavili aby počúval na porte 514.

Listing 3.27: /etc/rsyslog.conf

```
# provides TCP syslog reception
$ModLoad imtcp
$InputTCPServerRun 514
```

Rsyslog tiež podporuje bezpečné pripojenie cez TLS. To však zabezpečuje len dôvernosť prenášaných správ, s ktorou nemáme na vnútornej sieti problém. Akonáhle by bolo potrebné prenášať správy na syslog bežiaci mimo vnútornej siete, má takéto opatrenie zmysel. Pretože sa v našom prípade medzi logovaným strojom a syslog serverom nenachádza žiadne aktívne sieťové zariadenie, na ktorom by sa dali správy odpočúvať nebudeme používať šifrovaný prenos.

3.5.2 LogWatch

Logwatch je program ktorý pomáha adminstrátorom s pravidelnou kontrolou logov[17]. Podľa nastavených pravidiel dokáže zvoleným administrátorom odoslať každý deň e-mail so sumárom záznamov, spolu s niektorými ďalšími štatistikami o systéme. Sumarizácia nespočíva len v aplikovaní filtrov, ktoré vynechajú menej zaujímavé záznamy, ale aj v ich zoskupovaní, takže môžu rýchlo získať prehľad koľko emailov server za deň odoslal, alebo koľko bolo neúspešných pripojení na SSH.

Nastavenia tohto programu sú v štandardnom adresári /etc/logwatch/. Po inštalácii z balíčkov je tento adresár prázdny, preto sme doň skopírovali obsah /usr/share/logwatch/default.conf/, kde sa nachádza príklad základnej konfigurácie. Táto konfigurácia poskytuje nastavenia pre sumarizáciu logov väčšiny používaných programov (apache, postfix, amavis, a i.).

3.5.3 Sledovanie systémov

Okrem logov jednotlivých aplikácií je dobré mať aj všeobecný prehľad o stave jednotlivých strojov. Väčšina virtualizačných technológií má možnosť poskytovať štatistiky o jednotlivých strojoch, najčastejšie vyťaženie virtuálneho procesora, virtuálnych diskov a sieťových zariadení,

veľkosť využitej operačnej pamäti, a niekedy aj množstvo využitého diskového priestoru. Tieto dáta dávajú veľmi hrubý obraz o bežiacich systémoch a navyše sa technológiu od technológie líšia. Preto sme sa rozhodli zostaviť virtuálne zariadenia, ktoré bude zbierať štatistiky priamo od jednotlivých strojov.

Debian má v repozitároch niekoľko balíkov, ktoré umožňujú sledovať zariadenia, alebo služby na sieti.

3.5.4 Nagios

Nagios umožňuje automatickú kontrolu služieb na sieti. Umožňuje administrátorovi získať rýchly prehľad o tom, ktoré služby momentálne bežia. Konfigurácia sledovania jednotlivých služieb nie je úplne triviálna, preto je dobré využívať templaty. Výhodou Nagiosu je, že na sledovaných strojoch nepotrebuje mať žiadny špeciálny softvér a na sledovanie využíva väčšinou klientské knižnice sledovaných služieb, alebo systémové nástroje (ako napríklad ping).

Nagios pri správnom nastavení vie kontaktovať administrátora pri výpadku niektorej zo služieb, a viesť jednoduché štatistiky o ich up-time.

Konfigurácia Nagiosu je uložená v štandardnom adresári /etc/nagios3. Dôležitý je hlavne adresár conf.d, v ktorom sa nachádza konfigurácia sledovania jednotlivých strjov. V základnej inštalácii je nastavené sledovanie všetkých infraštruktúrnych strojov (ca,ns,ns2,ldap,ldap2). Konfigurácia stroja vyzerá takto:

Listing 3.28: /etc/nagios3/conf.d/ns1.cfg

```
define host{
        use                     generic-host
        host_name               ns.infrastructure.local
        alias                   ns
#       address                 192.168.201.104
        }
```

Položka z adresou je zakomentovaná, aby ju bolo možné čo najjednoduchším spôsobom zmeniť. Samotná kontrola niektorého zo servero je nastavená v súbore hostgroups_nagios2.cfg a services_nagios2.cfg. V konfiguračnom súbore services sú nastavenia skriptov slúžiacich na sledovanie jednotlivých služieb. Ku konfigurácii každého sledovania služby potom prislúcha kupina serverov, ktoré poskytujú sledovanú službu. Ktoré servery prislúchajú ku ktorej skupine je nastavené v konfiguračnom súbore hostgroups. Napríklad nastavenie sledovania SSH potom vyzerá takto:

Listing 3.29: /etc/nagios3/conf.d/services nagios2.cfg

```
check that ssh services are running
define service {
        hostgroup_name                  ssh-servers
        service_description             SSH
        check_command                   check_ssh
        use                             generic-service
        notification_interval           0 ;
}
```

Táto konfigurácia zabezpečí, aby všetky počítače v skupine ssh-servers boli kontrolované pomocou príkazu check_ssh, či je na nich funkčné ssh. Príkaz check_ssh sa nachádza spolu s ostatnými kontrolnými príkazmi v /usr/lib/nagios/plugins/. Nastavenie týchto príkazov sa však nenachádza medzi konfiguráciou Nagiosu ale v separátnom adresári /etc/nagios-plugins. Nastavenie ssh_check vyzerá takto:

Listing 3.30: /etc/nagios3/conf.d/services nagios2.cfg

```
# 'check_ssh' command definition
define command{
        command_name    check_ssh
        command_line    /usr/lib/nagios/plugins/check_ssh '$HOSTADDRESS$'
}
```

Definícia skupiny kontrolovaných serverov potom vyzerá takto:

Listing 3.31: /etc/nagios3/conf.d/hostgroups nagios2.cfg

```
# A~list of your ssh-accessible servers
define hostgroup {
        hostgroup_name  ssh-servers
                alias           SSH servers
                members         *
        }
```

Takto sme pridali do skupiny ssh-servers všetky zadefinované servery. Je ich možné zadefinovať aj vypísaním jednotlivých server-aliasov, ako sú zadefinované v host konfiguráciách. Teraz nagios automaticky pre všetky servery v tejto skupine kontroluje, či sú prístupné cez protokol ssh.

Rovnakým spôsobom je možné zadefinovať sledovanie ostatných služieb, ako sú HTTP servery, pontrola pomocou programu ping, a iné.

3.5.5 Munin

Na sledovanie komplexnejších štatistík a trendov použijeme Munin.Ten pracuje na klient-server princípe, kde každý sledovaný stroj má okrem svojich vlastných služieb nakonfigurovaný aj munin-node server. Munin sa potom v určených intervaloch pýta jednotlivých serverov na ich stav, a z odpovedí kreslí grafy.

Práve to, že Munin každému sledovanému serveru pridá ďalší načúvajúci proces vytvára bezpečnostné riziko. To je ale podla nás dostatočne vyvážené poskytovaním cenných informácií o systéme. Práve na dlhodobejšej štatistike je možné odhaliť inak ťažko pozorovateľné správanie stroja, ktoré môže signalizovať či chybu v konfigurácii, problém s hardvérom, alebo prebiehajúci útok.

Zobrazovanie dát sa v Munine deje prostradníctvom jendoduchého cron-skrpitu, ktorý je spúšťaný každých päť minút a zbiera dáta zo serverov, ktoré majú nainštalovaný munin-node. V základnej inštalácii je nastavený len jediný server a to localhost. Zoznam serverov, ktoré bude munin kontrolovať je zadefinovaný v súbore /etc/munin/munin.conf. Definícia nového serveru potom môže vyzerať takto:

Listing 3.32: /etc/munin/munin.conf

```
# Then our other host...
[db.infrastructure.local]
        address 192.168.201.202
```

Okrem toho sú v tomto súbore ďalšie ukážky konfigurácie, pomocou ktorých sa dajú vytvárať štatistiky pre celú doménu, nielen jednotlivé stroje. Nastavením serveru munin-node, ktorý pre munin vytvára štatistiky sa budeme zaoberať v sekcii ktorá je venovaná MySQL serveru.

3.5.6 Mail server

Časť konfigurácie tohto zariadenia je nutné venovať správnemu nastaveniu posielania e-mailu. V základnej inštalácii Debianu sa nachádza smtp server Exim, ktorý sa dá použiť práve na tento účel. Tento mailserver samozrejme nebude priamo pripojený k sieti, bude slúžiť len ako proxy medzi programami na našom virtuálnom zariadení a skutočným produkčným mailserverom. Pre tento účel má Debian možnosť nastavit' Exim do režimu satelit.V tomto režime smtp server

- nebude prijímať žiadnu poštu
- požiadavky na odoslanie emailu môžu prísť len z tohto virtuálneho zariadenia
- email bude odoslaný cez centrálny smtp server na ktorom má toto zariadenie svoj emailový účet

Týmito obmedzeniami sme pre procesy spustené na virtuálnom zariadení vytvorili jednoduchý spôsob na odosielanie emailu, ktorý ale poskytuje dostatočne málo možností na zneužitie. Na nastavenie potrebujeme vytvoriť v exime nový „router“, zadefinovať mu „transport“ cez ktorý sa bude môcť pripájať na vzdialený server a nakoniec nastaviť prihlasovacie údaje ktorými sa bude náš server o vzdialený server autentifikovať. Po inštalácii nášho zariadenia je ako vzdialený server prednastavený gmail.com .

Listing 3.33: /etc/exim4/conf.d/router/201 exim satelite

```
.ifdef DCconfig_smarthost DCconfig_satellite
# configtype=smarthost or configtype=satellite
send_via_smarthost:
        driver = manualroute
        domains = ! +local_domains
        transport = gmail_smtp
        route_list = * smtp.gmail.com
.endif
```

Listing 3.34: /etc/exim4/conf.d/transport/30 exim4-config remote smtp smarthost

```
...
smarthost_smtp:
  driver = smtp
  port = 587
  hosts_require_auth = $host_address
  hosts_require_tls= $host_address
...
```

Pri zmene vzdialeného serveru bude pravdepodobne nutné zmenit' vzdialený port.

Listing 3.35: /etc/exim4/conf.d/auth/01 smarthost smtp

```
smarthost_login:
     driver = plaintext
     public_name = LOGIN
     client_send = : mojemail@gmail.com : password
```

Po nastavení správneho prihlasovacieho mena a hesla, update konfigurácie a reštarte eximu budú všetky e-maily poslané cez sendmail odchádzať z tejto adresy. Niekoľkými úpravami je možné zmenit' odosielanie cez gmail na ľubovoľný iný smtp server, ktorý má k nastavený smtp-auth.

3.5.7 Zabezpečenie Web Serveru

Služby na monitorovanie zobrazujú svoje dáta cez web-server. Nagios aj Munin sme nainštalovali priamo z binárnych balíkov, preto s nimi dostaneme aj základnú konfiguráciu web-serveru Apache.

Narozdiel od webových aplikácií, ktorých nastavovaním sa budeme zaoberat' o niekoľko podkapitol neskôr, by dáta z týchto služieb nemali byt' verejne prístupné. Preto sme sa rozhodli nastavit' pre server šifrované spojenie a kontrolu prístupu.

Apache má v základnej inštalácii nastavené dva virtuálne servery, default a default-ssl. Na sprevádzkovanie šifrovaného spojenia nám preto stačilo vygenerovat' pre server cerifikát, nastavit' ho v konfigurácii pre default-ssl a spustit' tento virtual-host.

Listing 3.36: /etc/apache2/sites-available/default-ssl

```
SSLCertificateFile    /etc/ssl/certs/nagios-cert.pem
SSLCertificateKeyFile /etc/ssl/private/nagios-key.pem
```

Na ochranu pred nepovoleným prístupom sme nastavili jednoduchú autentifikáciu pomocou .htpasswd súboru. V základnom nastavení je používateľ i„admin“ s heslom „login“

Listing 3.37: /etc/apache2/httpd.conf

```
<Location "/">
        AuthName "Access"
        AuthType Basic
        AuthUserFile /etc/apache2/server.htpasswd
        Require valid-user
</Location>
```

Alternatívne je možné serveru prikázat' odpovedat' výlučne na požiadavky od localhostu, s tým že pripojenie je potom možné cez SSH tunel. Toto riešenie je jednoduchšie na nastavenie, a poskytuje kontrolu prístupu, ako aj šifrované spojenie. Je však oveľa menej užívateľsky prívetivé.

3.5.8 Po inštalácii

Po inštalácii z inštalačného cd bude pravdepodobne najväčším problémom slabé heslo chrániace web-stránky a sebou podpísaný certifikát. Nové heslo je možné vygenerovat' príkazom „sudo htpasswd -c /etc/apache2/server.htpasswd admin“, certifikát je možné vygenerovat' a podpísat' pomocou certifikačnej autority, ktorú sme nastavovali v jednej z predchádzajúcich sekcií.

3.6 Centrálna správa identít

Posledným infraštruktúrnym zariadením,ktoré potrebujeme pred tým než popíšeme konfiguráciu produkčných strojov, je server na centrálnu správu užívateľov.

Štandardne je na Linuxe (a d'alších systémoch Unix-ového typu) databáza užívateľov uložená v súboroch /etc/passwd s prislúchajúcimi hashmi ich hesiel v /etc/shadow. Okrem toho sú pre popis užívateľov a ich schopností dôležité aj súbory /etc/group a /etc/gshadow, kde je špecifikovaná príslušnost' užívateľov k rôznym skupinám.

Najjednoduchší spôsob, ako na serveroch vytvorit' jednotnú množinu používateľov, je synchronizácia týchto súborov. Stačí jeden zo serverov určit' ako master-server, a na ostatných nastavit' prístupové práva tak, aby master-server vedel menit' súbory passwd, group a ich shadow

verzie. Synchronizácia potom môže prebiehať pomocou jednoduchého shell skriptu. Takéto riešenie je postačujúce, ak by sme potrebovali na všetkých zariadeniach udržiavať rovnaký zoznam užívateľov.

My predpokladáme, že na naše produkčné servery bude administrovať niekoľko administrátorov, v niektorých prípadoch nezávisle na sebe. Preto potrebujeme riešenie, ktoré umožní jednoducho pridať nového administrátora, nastaviť mu ním spravované servery, a v prípade potreby mu dostatočne rýchlo vedieť odobrať prístup.

Na implementáciu sme sa rozhodli použiť OpenLDAP server. Na strane produkčných zariadení vieme nastaviť, aby k užívateľom ktorý sú uložení v /etc/passwd pridali aj tých ktorí sú v LDAP databáze. To znamená, že v databáze nám stačí mať uložených len tých užívateľov, ktorí sa potenciálne môžu chcieť pripájať na viaceré stroje. Linux na oddelenie jednotlivých služieb často používa systémových uživateľov[2] ktorých je zbytočné replikovať na všetkých strojoch. Z bezpečnostných dôvodov nechceme mať v databáze uložený ani root-účet.

Možnosť mať uložené užívateľské účty zároveň na jednotlivých strojoch aj v databáze tiež zaručuje, že sa na stroj bude môcť niektorý z administrátorov nalogovať, aj v prípade nedostupnosti centráneho serveru. OpenLDAP navyše umožňuje niekoľko spôsobov replikácie na sekundárny server, čím vieme aj bežných užívateľov produkčných strojov chrániť pred jeho výpadkom.

3.6.1 Primárny LDAP server

OpenLDAP je adresárová databáza, čo znamená že má narozdiel od tradičných relačných databáz[3] udržiava svoje dáta v stromovej štruktúre. Koreňovým uzlom databázy býva názov domény, ku ktorej server prislúcha. Je síce možné prevádzkovať LDAP server aj bez funkčnej domény, my však pri našom virtuálnom zariadení budeme predpokladať, že na sieti na ktorej toto zariadenie pobeží je funkčný DNS server, ktorý stroju s LDAP serverom pridelil doménové meno ldap.infrastructure.local.

Pretože väčšina konfigurácie OpenLDAP-u sa nachádza priamo v databáze[18], namiesto príkladov konfiguračných súborov budeme v tejto sekcii používať exportované časti databázy v LDIF[4] formáte. LDIF je formát ktorý slúži na textový export a úpravu dát v LDAP databázach[9].

Koreňový uzol našej databázy v LDIF formáte vyzerá takto:

Listing 3.38: dc.ldiff

```
dn: dc=infrastructure,dc=local
objectClass: top
objectClass: dcObject
objectClass: organization
o: infrastructure.local
dc: infrastructure
```

V základnej inštalácii je tiež automaticky nastavený administrátor databázy:

Listing 3.39: dc.ldiff

```
dn: cn=admin,dc=infrastructure,dc=local
objectClass: simpleSecurityObject
objectClass: organizationalRole
```

[2] napríklad celosystémový daemon, alebo systémoví užívatelia ktorí sú priamo priradení k niektorej službe, ako www-data,mysqld alebo ftp

[3] PostgreSQL,MySQL, a.i

[4] LDAP Data Interchange Format

```
cn: admin
description: LDAP administrator
```

Jednotlivé položky v databáze sú potom prístupné prostredníctvom ich DN[5] záznamu. DN záznam má podobnú štruktúru ako doménové meno ktoré začína názvom listového uzlu (cn=admin) a končí koreňovým uzlom (dc=infrastructure,dc=local), ktorý je priamo odvodený z domény. Položka cn[6] určuje meno listu.

Do databázy sme pridali uzol pod ktorým budeme ukladať unixových uživateľov a uzol pre unixové skupiny. Týmto uzlom začína ich meno položkou ou[7], ktorá určuje organizačnú jednotku vrámci databázy.

Listing 3.40: dc.ldiff

```
dn: ou=people,dc=infrastructure,dc=local
ou: people
objectClass: organizationalUnit

dn: ou=groups,dc=infrastructure,dc=local
ou: groups
objectClass: organizationalUnit
```

Uživateľský účet je v databáze uložený ako objekt triedy posixAccount, a musí okrem názvu účtu obsahovať uid, gid a cestu k domovskému adresáru. Okrem toho môže obsahovať ďalšie nastavenia linuxového účtu.[10] V základnej inštalácii sme nastavili jeden Uživateľský účet, s príhodným názvom „administrator“.

Tento účet bude mať z pohľadu našej produkčných strojoch podobnú funkciu, ako má účet root.Pomocou neho bude možné editovať konfiguračné súbory jednotlivých služieb a ich spúšťanie a vypínanie. Týmto spôsobom vo väčšine prípadov nebude nutné dať administrátorom jednotlivých produkčných strojov prístup priamo k root-účtu a bude stačiť nastaviť sudo na účet administrator.

Listing 3.41: dc.ldiff

```
dn: uid=administrator,ou=people,dc=infrastructure,dc=local
uid: administrator
uidNumber: 2001
gidNumber: 2001
cn: administrator
objectClass: top
objectClass: posixAccount
objectClass: shadowAccount
loginShell: /bin/sh
homeDirectory: /home/administrator
userPassword:{crypt}!
```

LDAP umožňuje aby jeden objekt mal viacero tried. Objekty triedy shadowAccount umožňujú zadefinovať ďalšie atribúty účtu, ako napríklad dátum expirácie hesla,alebo dátum jeho poslednej zmeny. Náš všeobecný administrátorský účet má v základnom nastavení zablokované heslo, pretože sa naň nebude dať prihlasovať priamo,ale bude k nemu pristupované cez sudo.

[5] Distinguished Name
[6] canonical name
[7] organizational unit

Listing 3.42: dc.ldiff

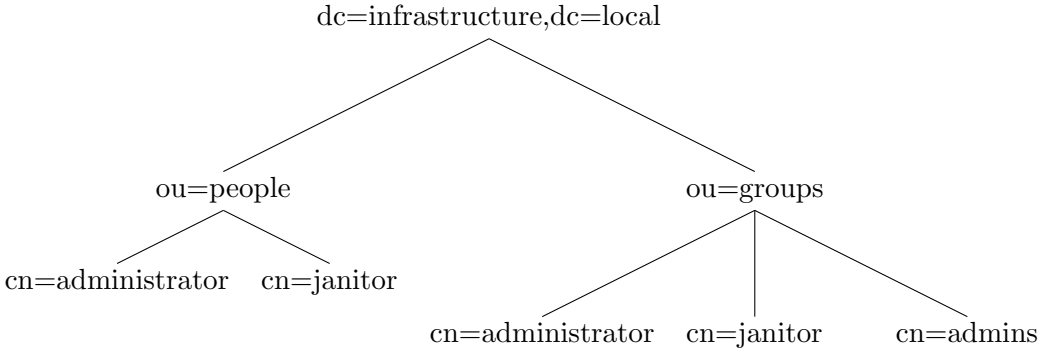
```
dn: cn=administrator,ou=groups,dc=infrastructure,dc=local
cn: administrator
gidNumber: 2001
objectClass: top
objectClass: posixGroup

dn: cn=admins,ou=groups,dc=infrastructure,dc=local
cn: admins
gidNumber: 2000
objectClass: top
objectClass: posixGroup
```

Po nainštalovaní je v databáze vytvorená tiež posix-ová skupina pre náš administrátorský účet a skupina v ktorej budeme udržiavať všetkých administrátorov. Prostredníctvom tejto skupiny budeme na produkčných strojoch nastavovať administrátorom privilegované práva. Tiež sme nastavili testovacieho užívateľa s prihlasovazím menom janitor a heslom „login“, ktorý síce na produkčných strojoch nemá administrátorské práva, ale bude sa s ním dať na ne prihlásiť.

3.6.2 Manažovanie Databázy

Takto vyzerá časť databázového stromu ktorej ukladáme našich uživateľov.



Export celej databázy v ldif formáte je možné dostať príkazom „sudo slapcat“. Ďalšie užitočné prikazy na manažovanie databázy sú[19]:

ldapadd slúži na pridávanie záznamov do databázy. Najčastejšie sme ho používali vo formáte

„ldapadd -cxWD cn=admin,dc=infrastructure,dc=local -f /subor.ldif“,

kde subor.ldif obsahuje dáta ktoré chceme nahrať do databázy,prepínače -cxWD zabezpečujú prihlasovanie cez databázový účet s heslom a cn=admin,dc=infrastructure,dc=local je zvolený databázový účet. Tento databázový účet nemá nič spoločné s posixovými účtami a je uložený v odlišnej časti stromu.

ldapdelete slúži na mazanie záznamov z databázy. Používa sa v podobnej forme ako ldapadd,

„ldapdelete -D cn=admin,dc=infrastructure,dc=local
cn=example,ou=example,dc=infrastructure,dc=local -W“.

Prepínač -D slúži na zadefinovanie prihlasovacieho mena, prepínač -W vynúti zadávanie hesla a bez prepínača je zadané DN zmazávaného objeku

ldapsearch slúži na zadávanie dotazov databáze. Ako jeden z mála príkazov funguje aj s anonymným užívateľom, príkaz „ldapsearch cn=jhonny“ nájde všetky listy databázy kde je cn rovné menu jhonny.

ldappasswd slúži na zmenu hesla databázového účtu. Napríklad príkaz „ldappasswd -AD cn=admin,dc=infrastructure,dc=local -W“ zmení heslo databázovému administrátorovi.

3.6.3 Manažovanie Užívateľov

Manažovať uživateľov je možné priamo cez nástroje OpenLDAP-u. Stačí si vytvoriť LDIF súbor s žiadanými zmenami a potom ho aplikovať cez príkazy ldapadd, ldapmodify, alebo ldapdelete. Týmto spôsobom sme našu databázu nainicializovali. Aby nebolo nutné každú úpravu v databáze prevádzať ručne, sú na serveri nastavený program cpu[8]. Tento balík poskytuje k úprave LDAP schém podobné rozhranie, aké majú Štandardné linuxové nástroje useradd a groupadd[20]. Príklad použitia:

cpu useradd /cpu groupadd Slúži na pridanie užívateľa/skupiny podľa LDAP schémy. Používanie je identické s príkazmi useradd/groupadd, s zopár rozdielmi ktoré sa zaoberajú typom ldap-schémy.

cpu userdel /cpu groupdel Zmaže posixový účet/skupinu. So zapnutým prepínačom -r zmaže aj súbory v domovskom adresári.

cpu usermod /cpu groupmod

3.6.4 Sekundárny LDAP server

V tejto sekcii sa budeme zaoberať replikáciou OpenLDAP databázy na sekundárny server. Dokumentácia OpenLDAP používa termíny poskytovateľ a príjemca. Servery môžu byť v tomto ponímaní producentmi aj konzumentmi zároveň a vytvárať medzi sebou distribuovanú databázu, s rôznymi vzťahmi medzi servermi:

master-slave slave server sa automaticky synchronizuje s master serverom pomocou syncrepl protokolu. Slave server slúži len na čítanie.

master-master replication oba servery sú synchronizované navzájom. Každý zo serverov príjma požiadavky ako na čítanie,tak aj na zápis a úspešnú zmenu spropaguje na všetky ostatné servery. Nezaručuje však konzistenciu dát medzi jednotlivými inštanciami.

mirroring je v princípe vylepšenou verziou slave-master zapojenia, s tým rozdielom, že roly sú serverom prideľované dynamicky. Zatiaľ čo pri master-master systéme môžu klienti zapisovať na ľubovoľný server, pri správne nastavenom mirorovaní je prístupný na zápis vždy len jeden. Pri výpadku serveru, na ktorý bolo možné zapisovať, sú zápisy presmerované na druhý server. Pri obnovení služby sa pôvodný zapisovací server zosynchronizuje so záložným, a ich roly sa vymenia. Týmto spôsobom je pri mirroringu zachovaná konzistencia dát ako pri master-slave, ale poskytuje lepšiu ochranu proti výpadku.

Na replikáciu sme sa rozhodli použiť master-slave prístup s použitím štandartného syncrepl[18]. Na primárnom serveri sme nastavili pre sekundárny práva na čítanie. Replikáciu by bolo možné sprevádzkovať aj bez toho,ale na kopírovanie by sme museli použiť administrátorský účet. Sekundárny server preto bude používať špeciálny účet, ktorý nebude mať právo na čítanie.

[8] Change Password Utility

Listing 3.43: ldaps2.ldiff

```
dn: cn=ldaps2,dc=infrastructure,dc=local
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: ldaps2
description: LDAP server2 replicator
userPassword: bilineatus
```

Na sekundárnom stroji nastavíme databázu na príjem dát.

Listing 3.44: dc.ldiff

```
add: olcSyncrepl
olcSyncrepl: rid=123
  provider="ldap://ldap.infrastructure.local:389/"
  type=refreshAndPersist
  retry="60 30 300 +"
  searchbase="dc=infrastructure,dc=local"
  bindmethod=simple
  binddn="cn=ldaps2,dc=infrastructure,dc=local"
  credentials=bilineatus
```

Takto budeme mať na sekundárnom LDAP serveri vždy k dispozícii živú zálohu všetkých dát.

3.6.5 Po inštalácii

Zariadenia LDAP serverov nebolo možné nastaviť priamo počas inštalácie, pretože pred reštartom systému nebol slapd v konzistentnom stave. Preto na dokončenie inštalácie je potrebné ručne spustiť skript /opt/ldap-postinst/installscheme.sh. Tým dojde k nainštalovaniu databázy.

Po nainštalovaní je pravdepodobne najdôležitejšie zmeniť heslo administrátorskému účtu „cn=admin,dc=infrastructure,dc=local“, pretože heslo „login“ je prislabé na produkčné nasadenie. Nové heslo je potom nutné nastaviť v konfigurácii programu na editovanie užívateľských účtov v databáze, cpu. Jeho konfigurácia sa nachádza v adresári /etc/cpu/cpu.conf. Replikácia na sekundárny server však funguje bez zásahu aj po zmene hesla.

3.7 Databázový server

Prvé z produkčných virtuálnych zariadení bude poskytovať prístup k SQL databáze. Ako databázu sme zvolili MySQL. Pretože sme databázový server nainštalovali z balíčkov, väčšina práce so zabezpečením je už urobená za nás. Proces mysqld beží pod neprivilegovaným užívateľským účtom, ktorý má vypnutý príkazový riadok a zakázaný prístup cez heslo. Dáta sú uložené v adresári /var/lib/mysql ktorý sa nachádza mimo systémovej partície, prístup k nemu má len užívateľ mysql.[11]

Server je nastavený aby čakal na spojenie z lokálnej siete. Firewall má povolené pripojenie cez port 3306. Toto je možné v nastavení firewallu sprísniť a povoliť prístup na tento port len špecificky z tých serverov, ktoré prístup k tejto databáze potrebujú.

Listing 3.45: /etc/init.d/firewall

```
iptables -A INPUT -p tcp -s 192.168.201.201 \
  -d 0/0 --destination-port 3306 -j ACCEPT
```

Z pohľadu operačného systému je naša MySQL inštalácia zabezpečená dostatočne. Ďalšie zabezpečenie už závisí od aplikácií ktoré databázu budú využívať.

Pri nastavovaní databáz pre aplikácie sa budeme držať týchto zásad[11]:

- Každý užívateľ v databázi slúži len na jednu funkciu, s minimálnymi potrebnými právami.
- Žiadny z uživateľov nemá prázdne heslo.
- Neexistuje anonymný užívateľ.

Ďalšou dobrou zásadou je aby cez každého databázového užívateľa pristupoval k databáze práve jeden skutočný užívateľ, či ide o administrátorov databáz, alebo jednotlivé aplikácie ktoré k nej pristupujú. Pretože ale predpokladáme, že nastavenie databázy vykoná práve jeden administrátor, nechali sme v základnej inštalácii databázový root účet. V prípade, že bude databázu spravovať viacero administrátorov, je lepšie vytvoriť každému vlastný administrátorský účet a pôvodného root užívateľa vymazať.

3.7.1 Administrácia cez Sudo

Účet pre užívateľa administrator je prístupný aj na virtuálnom zariadení. Tento účet má plné právo na zapisovanie aj čítanie v adresári /etc/mysql.

Listing 3.46: ls -l /etc/mysql/

```
drwxr-xr-x 2 administrator administrator 4096 May 16 13:22 conf.d
-rw------- 1 administrator administrator  333 May  8 13:55 debian.cnf
-rwxr-xr-x 1 administrator administrator 1198 Nov 30 10:35 debian-start
-rw-r--r-- 1 administrator administrator 3505 Nov 30 10:35 my.cnf
```

Skupine admins sme pridelili práva na prístup k užívateľovi administrator cez sudo,spolu so schopnosťou reštartovať server.

Listing 3.47: /etc/sudoers

```
Runas_Alias     ADMIN = administrator
...
%admins ALL=(ADMIN) ALL, (root) /etc/init.d/mysql
```

Týmto spôsobom, keď sa na stroj prihlási niekto, kto patrí do skupiny admins, má prostredníctvom príkazu sudo k dispozícii všetky prostriedky ktoré potrebuje na administrovanie daabázy.

3.7.2 Munin

Na databázovom servery má zmysel nastaviť aj jeden zo sledovacích uzlov munin-u. Dlhodobé štatistiky slúžia hlavne na odhaľovanie neefektívnych častí v systéme. Práve u databáz často stačí malá zmena v konfigurácii na veľké zlepšenie výkonu.

Po inštalácii má munin-node už v základnom nastavení zapnuté veľké množstvo pluginov. Na sprevádzkovanie štatistík teda stačí len nastaviť server na ktorom beží munin medzi povolené servery.

Listing 3.48: /etc/munin/munin-node.conf

```
...
allow ^127\.0\.0\.1$
...
```

Na serveri na ktorom beží munin (medzi našimi zariadeniami ide o zariadenie poskytujúce syslog server) je potom stačí nastavit' skript generujúci grafy na odber dát z tohot servera.

3.8 Mail server

Ak by sme mali k dispozícii len jeden server, pravdepodobne by naše zariadenia obsahovalo databázu e-mailových schránok, administračné rozhranie k databáze a webové rozhranie k emailom, server na odosielanie e-mailov a server na preberanie e-mailov.

Pretože databázový server už máme v našich zariadeniach pokrytý a aj mail-server aj administračné webové rozhranie sú v podstate klientmi rovnakej databázy, môžeme sa venovat' každému zvlášt'. Na záver sa budeme venovat' samotnému web-mailu.

Virtuálne zariadenie mail-serveru musí v zásade poskytovat' tri služby. Príjem e-mailov, odosielanie e-mailov a prístup ku mailovej schránke. Zatiaľ čo príjem a odosielanie elektronickej pošty má na starost' protokol smtp, na prístup k schránke sa používa protokol pop3, alebo imap4. Preto budeme na našom stroji potrebovat' dva servery, jeden pre smtp protokol, druhý pre imap4. Pop3 naše zariadenie nebude podporovat', pretože podporovat' dva rôzne protokoly sa nám zdá zbytočné. Protokol imap4 pracuje s e-mailami priamo na serveri, čo znamená, že klient môže e-maily st'ahovat' nezávisle na sebe na rôzne stroje. Navyše imap4 dobre spolupracuje s webovým klientom RoundCube, ktorý bude jedným z našich virtuálnych zariadení.

3.8.1 Postfix

Ako smtp server sme zvolili Postfix. Pretože plánujeme manažovat' naše mailové schránky cez mysql-databázu, bude pred inštaláciou samotného zariadenia Mail serveru nutné mat' pripravený správne nastavený databázový server. Alternatívne je možné na vytvorenie jednotlivých poštových schránok použit' databázu unixových uživateľov, alebo ldap-server.

Na nastavenie databázy sme použili databázovú schému webovej aplikácie postfix-admin. Týmto spôsobom bude neskôr možné nad rovnakou databázou nastavit' administračné rozhranie.Na vytvorenie schémy sme použili priamo inštalačný skript postfix-adminu[21]. Samozrejme pri inštalácii zariadenia nepoužívame na sprevádzkovanie databázy php-skript, ale sql-dump nastavenej databázy.

Listing 3.49: postfixdb

```
mysql> show tables;
+-------------------------+
| Tables_in_postfixdb     |
+-------------------------+
| admin                   || alias                |
| alias_domain            || config               |
| domain                  || domain_admins        |
| fetchmail               || log                  |
| mailbox                 || quota                |
| quota2                  || vacation             |
| vacation_notification   |
+-------------------------+
15 rows in set (0.00 sec)
```

Samotná postfix-databáza má aj svoje vlastné inštalačné cd.

V nastaveniach postfixu sme zmenili konfiguráciu pre získavanie virtuálnych schránok, aliasov a domén tak, aby používali mysql backend. Po inštalácii je ešte nutné nastaviť správneho datbázového užívateľa, jeho heslo a stroj na ktorom sa databáza nachádza.

Listing 3.50: /etc/postfix/sql/*.cf

```
user = postfix
password = password
hosts = 192.168.201.205
dbname = postfixdb
...
```

3.8.2 Ochrana proti Spamu

Nevyžiadaná pošta, nazývaná tiež Spam[9], je už niekoľko rokov jedným z hlavných problémov, ktoré musia prevádzkovatelia mailových serverov riešiť. Naším nastavením chceme dosiahnuť dve veci:

- náš server nebude možné zneužiť na rozosielanie nevyžiadanej hromadnej pošty.
- vlastníkom e-mailových schránok bude prichádzať minimálne množstvo spamu.

Prvý bod najjednoduchšie splníme tým, že budeme na našom serveri vyžadovať autentifikáciu od každého, kto bude chcieť poslať e-mail. Autentifikácia sa bude diať cez smtp-auth a pretože chceme, aby bral údaje našej MySQL databázy, obsahuje naše zariadenie sasl[10] knižnicu a pam modul na autentifikáciu o mysql.

Nastavili sme novú pam konfiguráciu, ktorá umožňuje autentifikovať sa o náš MySQL server. V databáze sú uložené hash-e hesiel vyrobené md5 verziou crypt funkcie,ktorú poskytuje knižnica crypt.

Listing 3.51: /etc/pam.d/smtp

```
auth       required     pam_mysql.so host=HOST\
           user=postfix passwd=PASSWORD db=postfixdb table=mailbox\
           usercolumn=username passwdcolumn=password crypt=1 md5=true
```

Sasl knižnicu sme nastavili na autentifikáciu o pam:

Listing 3.52: /etc/postfix/sasl/smtpd.conf

```
pwcheck_method: pam
```

Teraz stačí, aby sme nastavili permint_sasl_auth k ostatným metódam kontroly prístupu. Smtp server teraz doručuje e-maily len v prípade, že sa príjemca nachádza na vnútornej sieti, alebo sa odosielateľ autentifikoval.

Listing 3.53: /etc/postfix/main.cf

```
smtpd_sasl_auth_enable = yes
```

[9] v pôvodnom význame ide o nie veľmi chutné mäso v konzerve, ktoré vo svojich scénkach spopularizovala humoristická skupina Monthy Python. Najznámejšia scénka so Spam-om sa odohráva v reštaurácii v ktorej vám čašník k ľubovoľnému jedlu donesie aj konzervu spamu. Nakoniec zistíte, že sa na vašom stole nachádza niekoľkokrát väčšie množstvo Spamu, než skutočne objednaného jedla. Nevyžiadaná pošta má v pomere k žiadanej pošte v tomto ohľade podobné vlastnosti.

[10] Simple Authentication and Security Layer

```
smtpd_sasl_local_domain = myserver
broken_sasl_auth_clients = yes
smtpd_recipient_restrictions = permit_mynetworks,
        permit_sasl_authenticated,
        reject
```

Týmto spôsobom máme relatívne dobrú ochranu pred odchádzajúcim spamom, samozrejme za predpokladu, že naši užívatelia sa nepodieľajú na jeho distribúcii. Ochrana pred príchodzou nevyžiadanou poštou väčšinou spočíva v jej odfiltrovaní. Filtre môžu pracovať na rôznych princípoch:

1. Filter odstraňuje emaily na základe zlého formátu správy, alebo jej hlavičky
2. Filter odstraňuje emaily na základe domény/ip-adresy z ktorej správa prišla. Zoznamy spam-rozosielajúcich domén je možné získať z dôveryhodného zdroja,akým je napríklad spamhaus.org.
3. Filter môže použiť technológiu gray-listingu. Ide o jednoduchý test pomocou ktorého si server vyrába vlastný white-list. Pri doručení nového e-mailu od neznámeho príjemcu server najprv odošle naspäť chybové hlásenie o nemožnosti doručenia e-mailu. Ak bol odosielateľ legitímny, po prijatí hlásenia o chybe odošle znovu rovnakú správu, a server ho pridá do zoznamu overených odosielateľov. Ak odosielateľ chce len rozposlať čo najviac nevyžiadanej pošty, pravdepodobne si chýb o doručení nebude všímať.
4. Filter môže priamo analyzovať obsah správy. Toto je pravdepodobne najzožitejší spôsob, a preto sa väčšinou aplikuje len na e-maily ktoré prešli všetkými ostatnými testami. Filter väčšinou hodnotí správu podľa rôznych kritérií (výskyt určitých slov, používanie veľkých písmen, a i.) a ak hodnotenie správy presiahne určitú hranicu, vyhlási ju za spam. Niektoré filtre majú možnosť sami sa postupne učiť rozlišovať nevyžiadanú poštu, toto učenie však vyžaduje zásah užívateľa.

Základný filter, ktorý bude kontrolovať formát hlavičiek sa dá nastaviť priamo v konfigurácii postfixu[**?**]:

Listing 3.54: /etc/postfix/main.cf

```
smtpd_sender_restrictions =
    reject_unauth_pipelining,
    reject_non_fqdn_recipient,
    reject_unknown_recipient_domain,
    permit_mynetworks,
    permit_sasl_authenticated,
    reject_unauth_destination,
...
```

Tiež sme zadefinovali black-list.

Listing 3.55: /etc/postfix/main.cf

```
...
    reject_rbl_client zen.spamhaus.org,
...
```

Pretože na rozosielanie nevyžiadanej pošty sú využívané stovky infikovaných počítačov tvoriace rozsiahle bot-nety, je blokovanie konkrétnych smtp serverov čoraz menej efektívna obrana proti spamu. Z týchto dôvodov prestal fungovať populárny blacklist dsbl.org. Spamhaus však stále poskytuje dostatočne roziahly zoznam.

Na sprevádzkovanie graylisting-u sme doinštalovali program postgrey. Ten komunikuje s postfixom na lokálnom porte 6000:

Listing 3.56: /etc/postfix/main.cf

```
...
    check_policy_service inet:127.0.0.1:10023
    permit
```

Pretože chceme, aby na našom zariadení bežal aj spamfilter s určitou mierou inteligencie, rozhodli sme sa nastaviť na kontrolu príchodzích správ spamassasin. Ten spolupracuje s postfixom prostredníctvom programu amavis. Amavis slúži práve na pripájanie externých kontrolných programov do mailových systémov. Okrem spamassasinu takto napríklad môžeme pripojiť aj antivírus clam-av, tým sa ale v našom zariadení nebudeme zaoberať.

V main.cf sme nastavili filtrovanie správ cez amavis, ktorý beží na porte 1024:

Listing 3.57: /etc/postfix/main.cf

```
...
content_filter = smtp-amavis:[127.0.0.1]:10024
...
```

Na umožnenie komunikácie s amavisom sme upravili my.cnf v postfixe:

Listing 3.58: /etc/postfix/my.cf

```
smtp-amavis     unix    -       -       n       -       5
smtp
    -o smtp_data_done_timeout=1200
    -o disable_dns_lookups=yes

127.0.0.1:10025 inet    n       -       -       -       -
smtpd
    -o content_filter=
...
```

Vytvárame tu dve nové spojenia, smtp-amavis vytvorí ďalší proces,ktorým bude postfix posielať amavisu poštu na kontrolu, a ten bude posielať naspäť overené správy na port 10025. Konfigurácia príjmania správ na porte 10025 obsahuje ešte ďalších 17 riadkov, ktoré sme pre lepšiu čitateľnosť vynechali. Práve premazanie premennej content_filter je dôležité, ak by ostala v pôvodnom stave, prijatý mail by bol znovu odoslaný na kontrolu do amavis-u.

Samotný spamassasin síce má svoj vlastný konfiguračný súbor, všetky potrebné nastavenia sme ale urobili v konfigurácii amavisd:

Listing 3.59: /etc/postfix/my.cf

```
$sa_spam_subject_tag = "[SPAM] ";
$sa_tag2_level_deflt = 5.0;
$sa_kill_level_deflt = 8.0;
$sa_quarantine_cutoff_level = 20.0;
```

```
@bypass_spam_checks_maps = [0];
```

Posledným riadkom sme vypli vynechávanie spamassasinu, čím sme vynútili protispamovú kontrolu. Predchádzajúce premenná upresňujú zaobchádzanie s kontrolovanými e-mailmi. Spamassasin každému mailu po kontrole pridelí číslo, ktoré je tým väčšie, čím viac sa správa podobá na spam. Pri prekročení prvej hranice sa do predmetu správy pridá reťazec „[SPAM]“, po prekročení druhej už nebude doručený. Všetky prijaté správy potom ďalej putujú do karantény, ktorá je umiestnená v zložke /var/lib/amavis/virusmails. Pri prekročení poslednej hranice je však správa zničená okamžite.

3.8.3 Dovecot

Na preberanie elektronickej pošty sme sa rozhodli použiť imap4 server Dovecot. Po nainštalovaní stačí, podobne ako pri nastavovaní postfixu, zadefinovať v konfiguračnom súbore hostname postfixovej databázy a prislúchajúce heslo.

Listing 3.60: /etc/dovecot/dovecot-mysql.conf

```
connect = host=192.168.201.205 dbname=postfixdb user=postfix password=login
driver = mysql

password_query = SELECT username AS user,password FROM mailbox \
        WHERE username = '%u' AND active='1'

user_query = SELECT maildir, 1001 AS uid, 1001 AS gid FROM mailbox \
        WHERE username = '%u' AND active='1'
```

V nastaveniech je nastavená podpora len pre protokol imap4, tiež sme nastavili, aby sa ako autentifikačná metóda používala práve mysql databáza nastavená v súbore dovecot-mysql.conf.

Listing 3.61: /etc/dovecot/dovecot.conf

```
protocols = imap

auth default {
  mechanisms plain
  userdb sql {
    args = /etc/dovecot/dovecot-mysql.conf
  }
  passdb sql {
    args = /etc/dovecot/dovecot-mysql.conf
  }
}
```

Týmto spôsobom je dovecot nastavený, aby sa cezeň mohli e-mailový klienti dostať k svojim mailovým schránkam.

3.8.4 Inštalácia

Mail server je asi najkomplexnejšie produkčné zariadenie. Na svoje správne fungovanie potrebuje správne nakonfigurovanú mysql-databázu, ktorá je separátnym zariadením, ktorej administrátorské rozhranie, postfixadmin, je ďalším separátnym virtuálnym zariadením. Preto je asi

najlepším postupom najprv nakonfigurovať dvojicu zariadení databáza - postfixadmin, a až potom nainštalovať e-mail server a naviazať ho na už funkčnú sql databázu.

3.9 Web server

Najčastejším dôvodom prevádzkovania serveru na internete je práve prevádzkovanie svojej vlastnej web-aplikácie. Webové servery (a ich klienti, webové prehliadače) sa za posledných niekoľko rokov stali takmer univerzálnym rozhraním medzi komplexnými serverovými aplikáciami a ich užívateľmi.

Posledná sekcia tejto kapitoly je venovaná práve nastaveniu webového serveru. Narozdiel od predchádzajúcich častí, popíšeme nie jedno, ale tri virtuálne zariadenia, ktoré sa v svojich nastaveniach budú mierne líšiť.

3.9.1 Základné bezpečnostné nastavenia

Ako web server sme vo väčšine prípadov použili Apache 2. Hlavným dôvodom je, že ide o jeden z najväčších open-source projektov, a preto má dobre udržiavaný kód a opravné patche sa rýchlo dostávajú do distribučných repozitárov.

Pretože je web-server Apache rozsiahly projekt, väčšina funkcionality je pridávaná prostredníctvom modulov. Vďaka tomu je možné minimalizovať inštaláciu nielen v rámci operačného systému, ale aj vrámci samotného web-serveru.

Na zlepšenie zabezpečenia inštalácie web-serveru Apache sme urobili tieto zmeny v konfigurácii[1]:

Listing 3.62: /etc/apache2/apache2.conf

```
ServerTokens Prod
ServerSignature Off
```

Web-server s týmto nastavením nebude svojim klientom pri štandardných chybových hláseniach posielať informácie o svojej verzii. Bežne web-browsery nepotrebujú vedieť akým web-serverom im bola stránka poskytnutá, pre prípadného útočníka je to však veľmi cenná informácia.

Konfigurácia jednotlivých modulov prebieha v podadresároch mods-enabled a mods-aviable. Samotné konfiguračné súbory sú uložené v adresári mods-aviable, a modul sa zapína pridaním symlinku na odpovedajúci konfiguračný súbor do mods-enabled. Na manažovanie modulov potom slúžia príkazy a2enmod a a2dismod. V základnej konfigurácii sme nechali zapnuté len tieto moduly:

Listing 3.63: /etc/apache2/mods-enabled/*.load

```
alias.load              authz_host.load   negotiation.load
auth_basic.load         authz_user.load   reqtimeout.load
authn_file.load         autoindex.load    setenvif.load
authz_default.load      dir.load
authz_groupfile.load    mime.load
```

S týmito modulmi je apache2 nastavený na poskytovanie statických stránok. Cesta k web-serverom poskytovanému adresáru sa v Debiane nastaví pomocou virtual-hostov, ktoré sa administrujú podobným spôsobom ako moduly. V tomto prípade ide o adresáre sites-enabled,sites-aviable a príkazy a2ensite a a2dissite. Zvolený virtual-host potom príjma požiadavky na základe nastaveného portu, ip-adresy, alebo domény, alebo sub-domény. Týmto spôsobom dokáže jeden web-server poskytovať rôzne stránky pre rôzne domény.

Toto prináša z bezpečnostného hľadiska problémy. Jedným z nich je nastavenie https spojenia pre rôzne web-aplikácie. SSL spojenie je vytvorené vzhľadom na ip-adresu servera, nie na doménové meno, preto by pri štandardnom nastavení museli všetky virtual-hosty zdielať rovnaký ssl certifikát. Existuje verzia https protokolu, ktorá umožňuje kontrolu certifikátu vzhľadom na doménové meno, tú ale podporuje len málo web browserov.

Z tohoto dôvodu sme na našich produkčných zariadeniach nastavovali vždy len dva virtual-hosty, jeden pre http a druhý pre https. Po inštalácii používa https virtual-host sebou-podpísané certifikáty z balíku ssl-cert. Nastavenie virtual-hostu je v konfiguračnom súbore sites-aviable/default-ssl:

Listing 3.64: /etc/apache2/sites-aviable/default-ssl

```
SSLEngine on
#   A~self-signed (snakeoil) certificate can be created by installing
#   the ssl-cert package.
SSLCertificateFile    /etc/ssl/certs/ssl-cert-snakeoil.pem
SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key
```

Tieto je samozrejme po inštalácii vhodneé nahradiť vlastnými certifikátmi podpísanými certifikačnou autoritou.

3.9.2 PHP

Skriptovací jazyk PHP patrí medzi najpopulárnejšie „serverové“ jazyky používané na písanie web-aplikácií. PHP používajú tri z našich zariadení, preto si pred ich opisom zadefinujeme základnú bezpečnostnú konfiguráciu. Pretože na prepojenie so serverom používame modul mod-php, nachádza sa konfigurácia v adresári /etc/php5/apache2/php.ini:

Listing 3.65: /etc/apache2/sites-aviable/default-ssl

```
# Do not expose PHP error messages to external users
display_errors = Off
# Restrict PHP information leakage
expose_php = Off
# Log all errors
log_errors = On
# Do not register globals for input data
register_globals = Off
# Minimize allowable PHP post size
post_max_size = 1K
# Ensure PHP redirects appropriately
cgi.force_redirect = 0
# Disallow uploading unless necessary
file_uploads = Off
# Disallow treatment of file requests as fopen calls
allow_url_fopen = Off
```

Pôjde o produkčný server,preto vypíname zobrazovanie chybových hlásení, ďalej sme zakázali nahrávanie súborov. V prípade že na našom zariadení budeme očakávať prijímanie súborov, okrem zapnutie file_uploads musíme tiež zväčšiť hodnotu premenných post_max_size a upload_max_filesize.

3.9.3 Web-Aplikácia inštalovaná z distribučných repozitárov

Prvé zariadenie je tvorené štandardným balíčkom, a má slúžiť hlavne ako ukážka toho, čo je možné jednoducho s Debianom vytvoriť. Na ukážku sme vybrali populárnu blogovaciu platformu WordPress, ale s drobnými úpravami je možné podobným spôsobom vytvoriť zariadenie s MediaWiki, alebo Drupalom.

Po inštalácii sme automaticky nastavili aby oba prednastavené virtual-hosty mali za svoj adresár /usr/share/wordpress, do ktorého sa balík wordpress nainštaloval. Pred samotným používaním je nutné nastaviť prístup k databáze v súbore /usr/share/wordpress/wp-config.php a potom spustiť v prehliadači stránku /wp-admin/install.php. Tento skript nainicializuje databázu a nastaví heslo administrátorovi wordpressu.

Listing 3.66: /usr/share/wordpress/wp-config.php

```
/** The name of the database for WordPress */
define('DB_NAME', '');
/** MySQL database username */
define('DB_USER', '');
/** MySQL database password */
define('DB_PASSWORD', '');
/** MySQL hostname */
define('DB_HOST', '');
```

3.9.4 RoundCube

Roundcube je webový imap klient. Na umožnenie prístupu k e-mailovej schránke preto nepotrebuje byť nainštalovaný priamo na emailovom serveri. Táto vlastnosť z neho robí ideálne virtuálne zariadenie. Nieje ho totiž treba nastavovať, a po nainštalovaní sa užívateľ môže cez tento sever prihlásiť priamo ku svojej schránke. RoundCube však stále niektoré nastavenia poskytuje, asi najpodstatnejším je možnosť nastaviť napevno e-mailový server na ktorý sa bude prihlasovať.

RoundCube má v stabilnej distribúcii debianu svoje balíčky, preto je inštalácia prakticky identická s WordPresom. Jediný rozdiel je, že sme sa rozhodli ukladať databázu pre roundcube do SQLite miesto obvyklejšej MySQL databázy.

3.9.5 Web-Aplikácia inštalovaná z iných zdrojov

V niektorých prípadoch je však webová aplikácia v repozitároch zastaralá, alebo nejakým spôsobom zaostáva za originálnou verziou. Tiež sa u webových aplikácií oveľa častejšie stáva, že žiadaná aplikácia sa v distribučných repozitároch nenachádza.

Po inštalácii takejto aplikácie priamo „od zdroja“ si na seba administrátor berie zodpovednosť udržiavať v zaplátanom stave. O pravidelné bezpečnostné opravy sa v našich virtuálnych zariadeniach stará priamo balíčkovací systém, pre aplikácie mimo tohto systému však väčšinou neexistuje jednoduché riešenie ako je udržiavať zaplátané. Najčastejšie neostáva nič iné, len sledovať vydania nových verzií a potom aplikáciu ručne preinštalovať. Našťastie si tento problém vývojári aplikácií začínajú uvedomovať a čoraz častejšie ponúkajú aj flexibilnejší spôsob získania svojej aplikácie než štandardný tarbal. Okrem distribučného balíčku, alebo distribučného repozitára existujú ďalšie dve možnosti, ktoré vývojári používajú čoraz častejšie.

Prvou možnosťou je zverejniť aplikáciu prostredníctvom verzionovacieho systému. Týmto spôsobom je možné updatovanie do veľkej miery zautomatizovať. Tento systém používa napríklad PHP framework Symphony.

Druhou možnost'ou je použit' distribučný kanál špecifický pre konkrétny skriptovací jazyk. Asi najznámejším je CPAN, ktorý poskytuje vývojárom používajúcim perl obrovské množstvo modulov. U CPANu nejde úplne o repozitár aplikácií, väčšina modulov sú skôr knižnice. Vlastný repozitár aplikácií má napríklad PHP, Ruby, Python, alebo Haskel:

Ruby Gems poskytuje viac ako 20 tisíc balíčkov. Manažovanie nainštalovaných balíčkov sa deje prostredníctvom príkazov gem install, gem uninstall a gem update. Systém dokáže updatovat' aj samého seba pomocou príkazu gem update –system. Okrem toho poskytuje možnost' vývojárom publikovat' na platforme svoje aplikácie.

Python PyPI obsahuje zoznam veľkého množstva dostupných balíkov pre Python.Inštalovat' je ich možné bud' pomocou príkazu easy_install je súčast'ou balíka setup-tools, alebo cez program pip.

PHP Pear neobsahuje priamo php aplikácie, jedná sa však o najjednoduchší spôsob, ako na systém dostat' najnovšie verzie php-knižníc. Ovládanie je Štandardné, pomocou príkazov pear install, pear update a pear uninstall.

3.9.6 PostFixAdmin

Virtuálne zariadenie, ktoré obsahuje postfixadmin má po inštalácii slúžit' na administráciu databázy mailového serveru. Ide o jednoduchý web-server s php. Konfigurácia je v /etc/postfixadmin/config.inc.php, kde sa dá nastavit' adresa databázy a odpovedajúce prihlasovacie údaje.

Listing 3.67: /etc/postfixadmin/config.inc.php

```
$CONF['postfix_admin_url'] = '';
$CONF['database_type'] = 'mysql';
$CONF['database_host'] = '192.168.201.205';
$CONF['database_user'] = 'postfix';
$CONF['database_password'] = 'PASSWORD';
$CONF['database_name'] = 'postfixdb';
```

Táto aplikácia už nieje inštalvaná z repozitárov, vývojári však postfixadmin distribujú aj formou .deb balíčku. Inštaláciou samostatného .deb balíčku síce prídeme o možnost' automatických updatov, stále však ide o lepšie riešenie ako inštalovat' aplikáciu priamo zo zdrojov. Balíček nainštaluje aplikáciu podľa debianovských konvencií, takže konfigurácia ostáva v /etc, kde môžeme nájst' aj predpripravený apache.conf na sprevádzkovanie pod web-serverom. Po vydaní novej verzie a stiahnutí novšieho balíčku funguje aj updatovanie relatívne automaticky, stačí preinštalovat' novší balíček cez starý.

Záver

Cieľom našej práce bolo implementovať sadu virtuálnych strojov, a popísať spôsob ich efektívneho vytvárania. Veríme že naše rozhodnutie použiť inštalačné CD, ako metódu vytváranie virtuálnych strojov sa ukáže pre našich koncových uživateľov ako dostatočne flexibilné, a teda bude mať táto práca aj reálne uplatnenie.

Momentálne je v produkčnom nasadení základný template virtuálneho zariadenia, ktorý používame na študentskom serveri st.dcs.fmph.uniba.sk na rýche sprevádzkovanie nových virtuálnych strojov. Aj keď tento template nieje sám o sebe optimalizovaný na poskytovanie niektorej zo služieb, už automatická aplikácia základných bezpečnostných nastavení šetrí veľké množstvo času.

Pri testovaní sprevádzkovania infraštruktúry a jednotlivých produkčných zariadení bolo najväčšie urýchlenie práce badateľná práve pri nastavovaní databázových zariadení a zariadení s rôznymi web-aplikáciami, kde sa nám podarila takmer celá konfigurácia zautomatizovať. Naopak pri inštalácii mail-serveru rozdiel medzi ručnou konfiguráciou a automatizovanou inštaláciou nebol až tak markantný.

V budúcnosti sa môžeme v tejto oblasti uberať ďalej dvoma rôznymi smermi:

- Môžeme pracovať ďalej na infraštruktúre, a preniesť väčšinu nastavení z produkčných strojov do infraštuktúrnych. Týmto spôsobom by sme mohli získať veľkú flexibilitu pokiaľ ide o vytváranie a rušenie produkčných strojov podľa našej potreby, na druhú stranu už ich nieje možné používať mimo nastavenej infraštruktúry.

- Druhá možnosť je zamerať sa na také virtuálne zariadenia, ktoré ku svojmu fungovaniu potrebujú minimum konfigurácie. Príkladom môže byť naše posledné zariadenie obsahujúce RoundCube, ktoré nepotrebuje na svoju funkciu takmer žiadny zásah zvonka.

Medzi ďalšie stroje na strane infraštruktúry by sme v budúcnosti radi zaradili napríklad inštalačný a konfiguračný server, o ktorých sme sa zmienili už v úvode.

Na strane produkčných zariadení by mohlo byť zaujímavé vyskúšať zostrojiť aplikácie na báze bootovateľných cd. Takéto virtuálne zariadenia, ktoré by poskytovali svoju službu už po nabootovaní „inštalačného“ média by potom znížili čas na spustenie novej služby prakticky na nulu.

Literatúra

[1] Operating Systems Division Unix Team of the Systems and Network Analysis Center, *Guide to the Secure Configuration of Red Hat Enterprise Linux 5* rev.4, September 14, 2010
Dostupné na internete: `http://www.nsa.gov/ia/_files/os/redhat/rhel5-guide-i731.pdf`

[2] *SUSE Studio from Novell Wins 2010 American Business Award for Best New Product or Service* Jún 24, 2010
Dostupné na internete: `http://www.novell.com/news/press/2010/6/suse-studio-from-novell-wins-2010-american-business-award-for-best-new-product-or-service-computer-hardware-or-services.html`

[3] Mike Renfro, Tennesee Technological University *Unix Infrastructure Management From Scratch* 2008
Dostupné na internete: `http://blogs.cae.tntech.edu/mwr/infrastructure-management/`

[4] *An Introduction To Information, Network and Internet Security*
Dostupné na internete: `http://security.practitioner.com/introduction/infosec_2.htm`

[5] Alexander Reelsen, Javier Fernández-Sanguino Peña, *Securing Debian Manual* 2000-2011
Dostupné na internete: `http://www.debian.org/doc/manuals/securing-debian-howto/`

[6] , *pam.d(8) - Linux man page* DCE-RFC 86.0, October 1995. Contains additional features, but remains backwardly compatible with this RFC.
Dostupné na internete: `http://linux.die.net/man/8/pam.d`

[7] *snmpd.conf(5) - Linux man page*
Dostupné na internete: `http://linux.die.net/man/5/snmpd.conf`

[8] Franck Martin, *SSL Certificates HOWTO* 2002-10-20
Dostupné na internete: `http://tldp.org/HOWTO/SSL-Certificates-HOWTO/`

[9] G. Good, iPlanet e-commerce Solutions *The LDAP Data Interchange Format (LDIF) - Technical Specification* June 2000
Dostupné na internete: `http://tools.ietf.org/html/rfc2849`

[10] L. Howard, *An Approach for Using LDAP as a Network Information Service* March 1998
Dostupné na internete: `http://www.faqs.org/rfcs/rfc2307.html`

[11] The Center for Internet Security, *Security Configuration Benchmark For MySQL 4.1, 5.0, 5.1 Community Editions* April 2009
Dostupné na internete: `http://benchmarks.cisecurity.org/tools2/mysql/CIS_MySQL_Benchmark_v1.0`

[12] the Debian Installer team, *Appendix B. Automating the installation using preseeding*
Dostupné na internete: http://www.debian.org/releases/stable/i386/apb.html.en

[13] Red Hat Engineering Content Services, *Security-Enhanced Linux*
Dostupné na internete: http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/6/html/Security-Enhanced_Linux/chap-Security-Enhanced_Linux-Introduction.html

[14] Gratisoft, *Sample /etc/sudoers file*
Dostupné na internete: http://www.gratisoft.us/sudo/sample.sudoers

[15] , *RSyslog - Documentation*
Dostupné na internete: http://www.rsyslog.com/doc/manual.html

[16] Kipp E.B. Hickman, *The SSL Protocol* Nov. 29th, 1994
Dostupné na internete: http://www.webstart.com/jed/papers/HRM/references/ssl.html

[17] Michal Dočekal, *Sledování logů pomocí nástroje logwatch*
Dostupné na internete: http://www.linuxexpres.cz/praxe/sprava-linuxoveho-serveru-sledovani-logu-pomoci-logwatch

[18] RJ Systems, *OpenLDAP provider on Debian squeeze* 2011-01-05
Dostupné na internete: http://www.rjsystems.nl/en/2100-d6-openldap-provider.php

[19] *ldapsearch(1) - Linux man page*
Dostupné na internete: http://linux.die.net/man/1/ldapsearch

[20] Blake Matheny, *CPU - Change Password Utility*
Dostupné na internete: http://cpu.sourceforge.net/

[21] Kolekív autorov, *Postifx Admin*
Dostupné na internete: http://sourceforge.net/projects/postfixadmin/files/