VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Absolvování individuální odborné praxe

Individual Professional Practice in the Company

2014

Jakub Skokan

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Jakub Skokan**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Absolvování individuální odborné praxe
Individual Professional Practice in the Company

Zásady pro vypracování:

1. Student vykoná individuální praxi ve firmě: Relbit, s.r.o.
2. Struktura závěrečné zprávy:
a) Popis odborného zaměření firmy, u které student vykonal odbornou praxi a popis pracovního zařazení studenta
b) Seznam úkolů zadaných studentovi v průběhu odborné praxe s vyjádřením jejich časové náročnosti
c) Zvolený postup řešení zadaných úkolů
d) Teoretické a praktické znalosti a dovednosti získané v průběhu studia uplatněné studentem v průběhu odborné praxe
e) Znalosti či dovednosti scházející studentovi v průběhu odborné praxe
f) Dosažené výsledky v průběhu odborné praxe a její celkové zhodnocení

Seznam doporučené odborné literatury:

Podle pokynů konzultanta, který vedl odbornou praxi studenta.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Marek Běhálek, Ph.D.**

Konzultant bakalářské práce: Pavel Šnajdr

Datum zadání: 01.09.2013

Datum odevzdání: 07.05.2014

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 2. května 2014

.....

Jakub Skokan

Rád bych na tomto místě poděkoval všem, kteří mi s prací pomohli, zejména mému vedoucímu práce a konzultantovi.

Abstrakt

Tento text popisuje vykonávání bakalářské práce formou odborné praxe ve firmě Relbit, s.r.o. Práce pojednává o mé motivaci absolvovat praxi, o firmě, jejím zaměření a projektech, kterými se zabývá. Dále obsahuje seznam zadaných úkolů, které jsem se snažil vyřešit. Detailně je rozepsán postup práce, návrh řešení a implementace, případně testování a provoz v produkčním nasazení. Výstupem práce je celkové zhodnocení praxe, pracovní úspěchy a neúspěchy, získané znalosti a zkušenosti.

Klíčová slova: odborná praxe, firma, Relbit, hosting, cloud, bakalářská práce

Abstract

This text describes bachelor thesis in form of individual professional practice in Relbit, s.r.o. It is about my motivation to participate in professional practive, about the company, its products and specialization. It contains a list of tasks I was trying to solve. It also features report of my work, proposed design, implementation, problem solving, testing and deployment in production environment. The results are overall eveluation of practice, work-related successes and failures, acquired knowledge and experiences.

Keywords: professional practice, company, Relbit, hosting, cloud, bachelor thesis

Obsah

1 Úvod

Tento text popisuje mou praxi a náplň práce během dvou semestrů ve firmě Relbit, s.r.o.

Firmu Relbit, s.r.o. jsem si vybral z důvodu zajímavého zaměření a používání technologií, se kterými bych se rád naučil pracovat, nebo se o nich dozvědět více. V rámci praxe jsem si chtěl vyzkoušet řešení reálných problémů, které člověk může potkat při provozu aplikací v produkčním prostředí.

Hledal jsem takovou práci, ve které bych mohl naplno využít své programovací schopnosti a povědomí o linuxových systémech. Vybíral jsem tak, abych nepracoval jen na vysokoúrovňových aplikacích a uživatelských rozhraních, ale podíval se i do hlubin operačního systému, služeb, jejich nastavení, sítí a síťových protokolů.

Současně mě lákalo podívat se na to, jak se v praxi využívají virtualizačních technologie k rozložení zátěže, dynamickému přidělování prostředků a vytvoření cloudových služeb, aniž by o tom uživatel věděl.

Více informací o firmě, její činnosti a produktech se lze dočíst v kapitole 2.

Seznam zadaných pracovních úkolů, na kterých jsem po dobu dvou semestrů pracoval, je v kapitole 3.

Velká část mé práce se týkala dvou projektů, vpsAdmin a Bitoxy, které je nutné nejdříve představit a uvést do kontextu. Více o těchto projektech v kapitole 3.1.

Samotná práce, návrh, postup, řešení úkolů a problémů s tím souvisejících je popsáno v kapitole 4.

Závěrečné hodnocení praxe, popisující úspěchy a neúspěchy, získané a využité znalosti je v kapitole 5.2.

2 O firmě

Relbit, s.r.o. je firma založená v roce 2010. Firmu tvoří mladý a velice přátelský kolektiv. Zabývá se poskytováním cloudových služeb a hostováním aplikací využívajících PHP [1] a MySQL [2].

V říjnu 2013 došlo ke zveřejnění produktu eviacloud pod svobodnou licencí Apache License 2.0 [3]. Jedná se o platformu pro hostovaní PHP aplikací, která se dokáže škálovat podle zátěže a dát tak uživateli potřebný výkon k vyřízení požadavků. V eviacloudu [4] běží většina PHP aplikací bez jakýchkoliv úprav, což byl častý problém podobných služeb.

Eviacloud má jednoduché webové rozhraní jak pro administrátory systému, tak pro uživatele. Nevyžaduje odbornost uživatele a zároveň je velice snadný i pro administrátory.

Relbit poskytuje jednak PaaS eviacloudu s podporou dle vybraného plánu, nebo je možné si eviacloud nainstalovat na svou vlastní infrastrukturu, ovšem bez podpory.

Ve firmě jsem pracoval na externích projektech, které Relbit interně využívá, konkrétně vpsAdmin a vpsadmininstall. Dále jsem pracoval na součásti eviacloudu zvané Bitoxy.

3 Úkoly

V průběhu odborné praxe jsem měl zadány následující úkoly:

- Zimní semestr
  - vytvořit instalátor pro vpsAdmin,
  - navrhnout a vytvořit program pro synchronizaci dat,
  - udržovat a rozvíjet stávající kód vpsAdminu.
- Letní semestr
  - udržovat a rozvíjet stávající kód vpsAdminu,
  - navrhnout a započít práci na vpsAdmin 2.0,
  - dopracovat log přístupů do Bitoxy.

O jednotlivých projektech a úkolech píšu podrobněji níže.

3.1 Informace o projektech

O projektech, na kterých jsem pracoval, je zde napsáno pár základních informací. Detailní popis činnosti a práce na nich je v kapitole 4.

3.1.1 vpsAdmin

Velká část mé práce se týkala open-source projektu vpsAdmin. Jedná se o webovou administraci virtuálních serverů. vpsAdmin byl původně vytvořen pro potřeby občanského sdružení vpsFree.cz Pavlem Šnajdrem.

vpsAdmin spravuje kontejnery vytvořené virtualizační technologií OpenVZ. Dokáže se starat o více fyzických serverů a virtuálních serverů na nich.

Pojem kontejner a VPS označuje jednu a tu samou věc.

Projekt se skládá ze tří částí.

- vpsAdmin – webové rozhraní pro uživatele i administrátory
- vpsAdmind – démon běžící na všech fyzických serverech, které jsou součástí instalace vpsAdminu
- vpsadminctl – program pro administrátory nainstalován po boku vpsAdmind a slouží k jeho ovládání

vpsAdmind se připojuje do databáze a s webovým rozhraním přímo nekomunikuje. vpsAdmind vytváří soket, přes který s ním komunikuje vpsadminctl.

Další součástí je samozřejmě vpsadmininstall, jeho využití je ale dobrovolné.

Mezi nejdůležitější funkce patří:

- rozdělení serverů do lokací a jejich nastavení,
- správa VPS, IP adres, konfigurace,
- migrace VPS mezi servery,
- zálohování VPS,
- připojování sdílených úložišť do VPS,
- sledování přenesených dat,
- hromadné operace,
- správa uživatelů, atd...

Webová část je napsána ve skriptovacím jazyku PHP, vpsAdmind a vpsadminctl v Ruby [5]. Jako databáze je použita MySQL, popřípadě MariaDB [10].

vpsAdmin je uvolněn pod licencí GNU/GPL [6].

3.1.2 Bitoxy

Bitoxy je FTP/FTPS [7, 8] proxy server napsaný v C++ s využitím Qt frameworku [11], který slouží jako load-balancer mezi uživateli a interními FTP servery.

Pro rozkládání zátěže používá Bitoxy tzv. „směrovače“. Podle zadaného uživatelského jména je spojení přesměrováno na daný interní FTP server. Směrovač je abstraktní rozhraní a je možno jej libovolně implementovat. Součástí Bitoxy je statický směrovač, který přesměruje všechny uživatele na jeden FTP server, nebo SQL směrovač, který konfigurovatelným dotazem do databáze zjistí, kam uživatele přesměrovat.

Bitoxy čeká, až uživatel poskytne svůj login. Podle něj pozná, kam má spojení přesměrovat. Heslo už validuje interní FTP server. Jestliže uživatel neexistuje, Bitoxy pokus o přihlášení simuluje a po zadání hesla jej vždy odmítne, aby se nedalo poznat, zda uživatel neexistuje, či je špatné heslo.

Bitoxy v rámci svého procesu rozprostírá uživatele mezi více vláken a dokáže tak využít všech procesorů v systému k dosažení maximální propustnosti. V jedné instanci lze vytvořit více FTP/FTPS proxy serverů s různými nastaveními.

Bitoxy je součástí eviacloudu. Je to volně dostupný software uvolněný pod licencí Apache License 2.0.

4 Práce na projektech

4.1 Program vpsadmininstall

Mým prvním úkolem bylo vytvořit instalační program pro vpsAdmin.

4.1.1 Požadavky

Cílem bylo vytvořit instalační program, který bude velice jednoduchý na použití a nebude vyžadovat odbornost uživatele, aby nemusel vědět nic o tom, jak funguje vpsAdmin či OpenVZ [9] kvůli vytvoření několika virtuálních serverů.

Spuštění instalace by se mělo provést jedním příkazem. Vše, co lze, se musí zjistit automaticky a uživatel nemá být zbytečně zatěžován.

Jakožto jediná podporovaná distribuce byl zvolen Scientific Linux 6, což je volně dostupný klon Red Hat Enterprise Linuxu. Důvodem je, že programátoři OpenVZ už nic jiného než RHEL nepodporují.

Ke spuštění instalátoru uživatel musí mít server s čistou instalací Scientific Linuxu. Instalační program nainstaluje a nastaví vše ostatní.

Instalátor musí umět vytvořit novou instalaci vpsAdminu a také připojení dalšího serveru do stávající instalace.

Výsledkem je plně funkční a nastavená instance vpsAdminu.

4.1.2 Provedení

Projekt je hostován u vpsFree.cz, o.s., jako součást vpsAdminu, i když jej přímo nevyužívá.

Instalátor je kompletně napsán v Bashi a spustí se pomocí jednoho řádku v terminálu.

```
curl -ko- "https://git.vpsfree.cz/?p=vpsadmininstall.git;a=blob_plain;f=install_vpsadmin.sh;hb=HEAD" | bash
```

Výpis 1: Spuštění nové instalace

```
curl -ko- "https://git.vpsfree.cz/?p=vpsadmininstall.git;a=blob_plain;f=install_node.sh;hb=HEAD" | bash
```

Výpis 2: Přidání serveru do existující instalace

Informace o průběhu jsou vypisovány jednak na konzoli a také do souboru */root/.vpsadmin.progress*. Detailní log instalace je uložen v souboru */root/vpsadmin-install.log*, ten je nutný pouze k ladění chyb.

Nastane-li během jakékoliv operace neočekávaná chyba, uživatel je o ní informován a dojde k okamžitému ukončení instalace. Problém lze najít a vyřešit za pomoci dříve zmíněného logu.

Instalátor si nejdříve od uživatele vyžádá nezbytné informace a zadaná data uloží, nainstaluje jádro OpenVZ a utility v uživatelském prostoru a nakonec nastaví server tak, aby po restartu došlo ke spuštění druhého instalačního kroku.

Systém se restartuje do nového OpenVZ jádra.

Ke spuštění instalátoru dojde ještě při načítání systému, uživatel se vůbec nedostane ke konzoli.

Ve druhém kroku dojde k vytvoření kontejneru #101, ve kterém sídlí webové rozhraní s databází a vpsAdmind pro rozesílání e-mailů.

Do tohoto kontejneru je nainstalován webový server Apache2, MySQL databáze a veškeré závislosti vpsAdminu. Instalátor obsahuje konfigurační soubory důležitých služeb.

Po nainstalování webového rozhraní se v databázi vytvoří tabulky a nahrají se do nich předpřipravená data, která obsahují základní nastavení.

Instalátor dále vytvoří soubor */etc/cron.d/vpsadmin*, do kterého vloží periodické spouštění operací, jako opožděné mazání uživatelů a dat, spouštění záloh a rozesílání e-mailů. Většina těchto operací je zakomentována, jelikož vyžaduje pokročilejší znalosti a práci ze strany administrátora.

Nakonec instalátor vytvoří soubor */root/vpsadmin.ready*, který obsahuje informaci o tom, kde může uživatel nalézt webové rozhraní, jaké jsou přihlašovací údaje a podobně. Součástí jsou také přístupové údaje k databázi, které jsou potřebné pro pozdější instalaci dalších serverů do clusteru. Tyto informace jsou taktéž zapsány do souboru */root/vpsadmin-node.txt*. Tento soubor je možné nahrát na server, který chceme přidat do naší instance vpsAdminu, a poté na něm spustit instalátor. Ten si soubor přečte a nebude vyžadovat ruční zadání údajů pro přístup k databázi.

Poté dojde opět k restartu systému a po naběhnutí je vpsAdmin plně funkční.

Práci na tomto projektu jsem se věnoval 14 dní.

4.2 Synchronizace dat pomocí ZFS

Tento projekt jsem měl víceméně ve své režii. Dán byl pouze cíl a to sice synchronizovat data mezi nastaveným počtem počítačů. Aplikace měla být použitelná jak k replikaci dat na serverech, tak k synchronizaci dat např. mezi notebookem a desktopem.

Jako způsob synchronizace byl zvolen systém souborů ZFS a jeho snapshoty, které lze snadno posílat po síti příkazy *zfs send* a *recv*. Alternativou k tomuto by mohlo být periodické spouštění *rsync*, to by ale bylo na větší množství souborů neúnosně pomalé. ZFS naopak neovlivní počty souborů, přenáší se pouze bloky dat.

Architektura systému byla navržena tak, že na každém počítači, který bude součástí synchronizačního procesu, poběží démon, který bude komunikovat s ostatními. Při startu tohoto programu se přečte konfigurační soubor, kde je nastaveno, co a kam se má synchronizovat. U každé synchronizované složky se také dá nastavit minimální interval mezi dvěmi synchronizacemi a jestli sledovat změny ve složce pomocí rozhraní kernelu inotify.

Synchronizace probíhá tak, že zaznamená-li démon změny ve složce, udělá snapshot a snaží se data rozeslat na počítače nastavené v konfiguraci. Kontaktuje démony na oněch počítačích a když dostane potvrzení, spustí *zfs send* přes SSH. To samozřejmě vyžaduje mít root uživatele synchronizovaných počítačů propojeny přes klíče tak, aby přihlášení nevyžadovalo heslo.

Tento úkol se mi nepodařilo úspěšně dokončit. Narazil jsem na obtížnosti v implementaci a protože se nejednalo o důležitou práci, tento projekt byl odložen.

Práci na tomto projektu jsem se věnoval 6 dní.

4.3 vpsAdmin

V průběhu zimního i letního semestru jsem se zabýval jednak údržbou stávajícího kódu, opravami chyb a také vývojem nových funkcí, od těch nejmenších úprav po komplexnější funkce.

4.3.1 Přizpůsobení pro vpsadmininstall

Aby mohl instalátor fungovat, bylo nutné udělat pár úprav ve webovém rozhraní i ve vpsAdmind.

Pro správnou instalaci databáze bylo nutné přidat k projektu aktuální a správné schéma databáze, které je při instalaci vytvořeno.

Větší úpravy byly potřeba ve vpsAdmind. Dodělána byla správa SSH klíčů a generování souborů *~/.ssh/authorized_keys* a *~/.ssh/known_hosts*.

vpsAdmin spravuje šablony konfiguračních souborů, které se aplikují na konfiguraci jednotlivých VPS. Přidal jsem tedy do vpsadminctl přepínač, který zajistí, že vpsAdmind šablony z databáze zapíše na disk. vpsadminctl s tímto přepínačem je volán při instalaci.

Zatím nevyřešen je problém aktualizace všech součástí vpsAdminu najednou. Pro distribuci se využívá verzovací systém *git*. Webové rozhraní je nutné aktualizovat ručně zavoláním příkazu *git pull*. Tento příkaz však musí být spuštěn ve VPS #101, kde je rozhraní nainstalováno. vpsAdmind se aktualizuje příkazem *vpsadminctl update*, který musí být spuštěn na všech serverech.

4.3.2 Webové rozhraní

Ve verzovacím systému je za dobu mé praxe několik desítek změn, ty významnější zde popíšu.

V několika sekcích administrace bylo přidáno přesměrování po provedení akcí. Chybějící přesměrování znamenalo, že při aktualizaci se akce vykonala znovu, což může mít nežádoucí důsledky. Tento nedostatek se týkal manipulace VPS, spravování síťových úložišť a záloh.

Důležitou změnou je zvýšení priority všech akcí zadaných ve webovém rozhraní, neboť automaticky vkládané systémové operace, jako například zálohování, zdržovaly uživatele. Příkazy s vyšší prioritou jsou při vykonávání upřednostněny.

Úprav se dočkaly také různé filtrovací formuláře, jejichž možnosti jsem rozšířil. Ku příkladu filtrování příkazu dle úspěšnosti provedení, nebo filtr VPS podle DNS překladačů, které používají.

Pro usnadnění monitorování stavu serverů byl vytvořen skript */cluster_status.php*, jehož výstupem je *OK*, je-li vše v pořádku, případně vypíše servery, které neodpovídají. Výstup tohoto skriptu lze jednoduše pravidelně kontrolovat a automatizovat upozornění správců systému.

Složitější funkcí byla implementace real-time monitoru přenosů dat. vpsAdmind co 10 sekund ukládá do tabulky v paměti v databázi přenesená data pro všechny VPS. Z těchto dat se vypočítá aktuální přenosová rychlost a sestupně seřadí a zobrazí ve webovém

rozhraní. Tuto funkcionalitu bude možno využít k automatické detekci DDoS útoků a blokaci postižených IP adres.

4.3.3 Program vpsAdmind

Vývoj webového rozhraní byl zrcadlen ve vpsAdmind, který slouží i k odesílání e-mailů. Dodělána byla podpora pro vkládání hlaviček typu *Message-ID*. *In-Reply-To* a *References* pro správné zobrazení zpráv v e-mailových klientech ve vláknech.

Vyřešen byl problém s rostoucí velikostí logovacích souborů, které v extrémních situacích dosahovaly až jednotky GB. Nedostatek jsem vyřešil napsáním konfiguračního souboru pro program *logrotate*, který se stará o rotování logů.

Velkým problémem byla manipulace s daty VPS více démony najednou. Na příklad při probíhající záloze uživatel spustil reinstalaci VPS. Záloha tak byla znehodnocena. Doplnil jsem kontrolu globálního zámku VPS, který zajišťuje, aby k datům přistupoval jen jeden proces, na všechna důležitá místa, aby k podobným problémům nedocházelo.

Poslední důležitou úpravou bylo real-time sledování přenosů dat. Přenosy se měří za pomocí *iptables*, tzn. linuxového bezestavového firewallu. vpsAdmind při startů vytvoří pravidla pro všechny IP adresy a co 10 sekund čte počítadla přenesených bajtů a paketů.

4.3.4 Program vpsadminctl

Zabýval jsem se usnadněním použití tohoto programu.

vpsadminctl má přepínač *--help*, který vypíše nápovědu pro použití. Popisuje podporované příkazy s krátkým vysvětlením jejich činnosti. Tento text je však nedostatečný, jelikož nepopisuje všechny implikace, které jednotlivé příkazy mohou mít.

Z tohoto důvodu byla vytvořena manuálová stránka, která detailně popisuje použití programu a všechny situace. U každého příkazu je napsáno, co přesně se stane, jak se systém zachová a jaké mohou být následky. Rozdíl v chování příkazů je ku příkladu v blokování vpsadminctl a čekání na dokončení operace. Některé příkazy čekají na provedení a jiné naopak běží na pozadí.

Po nainstalování lze manuálovou stránku otevřít příkazem *man vpsadminctl*.

Práci na tomto projektu jsem se věnoval 3 dny v zímním semestru a 7 dnů v letním semestru.

4.4 Projekt vpsAdmin2

Jako vývojář vpsAdminu jsem si vědom nedostatků a špatných základů stávající verze, kvůli kterým je obtížné implementovat nové funkce. Zároveň u ní dochází, a čím dál tím častěji bude docházet, k duplicitnímu kódu v eviacloudu a vpsAdminu. Dalším problémem je udržování zpětné kompatibility se starými konfiguracemi, která také komplikuje vývoj.

Z těchto důvodů jsem dostal za úkol navrhnout novou verzi úplně od začátku.

4.4.1 Požadavky

- Prvním programátorským požadavkem bylo vybrat jiný jazyk než PHP. Administrace by měla být mnohem obecnější a ne ušitá na míru jednomu případu. vpsAdmin2 by měl být plně open-source projekt se svou stránkou, dostupnými zdrojovými kódy a kvalitní dokumentací.

  Vyřešit duplicitní kód ve vpsAdminu a eviacloudu.

- Nová verze by měla plně využívat výhod ZFS. To znamená umožnit uživatelům vytvářet datasety, migrace a zálohy VPS dělat přes zfs send a recv, replikace datasetů/dat VPS. Doposud se k většině operací používá rsync, takže tato změna by měla mít za následek výrazné urychlení operací s daty, za předpokladu, že nenarazíme na chyby v implementaci ZFS na Linuxu.

  Veškeré operace budou řešeny na úrovni datasetů, jejich snapshotů a přesouvání. Systém nebude vědět, že dělá zálohu, bude pouze vytvářet a přesouvat snapshot.

- Lokace a servery budou spadat do tzv. „prostředí“. Každé prostředí bude mít svou konfiguraci. Pro představu, prostředí může být produkce, vývoj, testování, apod.

- Uživatel dostane přiděleny určité prostředky (procesor, operační paměť, disk) a bude si je moci rozdělit mezi své VPS. Musí být definována minimální a maximální konfigurace dle prostředí, aby si uživatel nemohl udělat příliš mnoho VPS, nebo jednu moc výkonnou.

- Stanovit, kolik a jak často musí uživatel za služby platit. Sledovat příchozí platby a automaticky je přiřazovat k uživatelům. Na požádání či periodicky vystavovat faktury a dát je uživateli k dispozici v PDF.

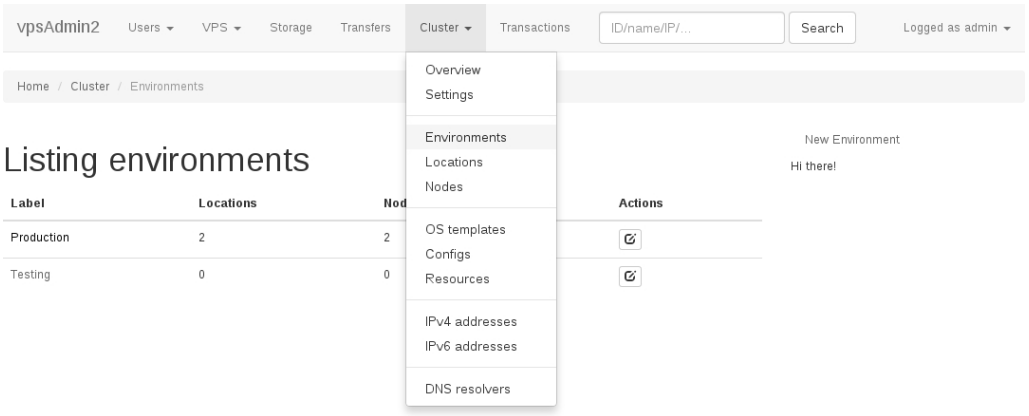
- Systém bude integrován s poštovním serverem za účelem správy uživatelské podpory. Uživatel pošle email na adresu podpory, poštovní server zprávu předá vpsAdminu, ten si ji uloží a rozešle požadavek technikům, kteří mohou odpovědět buď přímo emailem, nebo přes webové rozhraní vpsAdminu. Tento systém usnadní a zpřehlední komunikaci mezi uživateli a techniky.

- Důležité je poskytovat API, které mohou uživatelé využívat pro správu VPS. Přes toto API by v konečném stavu mělo jít udělat vše, co ve webovém rozhraní. Zpočátku má prioritu tvorba VPS, start, restart, apod.

- Logovat veškeré akce uživatelů, ACL pro přístup ke všem objektům, přehledné uživatelské rozhraní a rozhraní přizpůsobené pro administrátory.

4.4.2 Návrh

Velká část vpsAdminu a celý eviacloud je napsán v Ruby, jehož volba byla jasná. Pro tvorbu webového rozhraní je využit Ruby on Rails. Jako databáze byla ponechána MySQL.



Obrázek 1: vpsAdmin2 - správa prostředí

Duplicitní kódy se budou nejspíše řešit tak, že se vytvoří jednotný obraz operačního systému, který bude poskytovat API. Tento systém poběží na všech serverech. vpsAdmin a eviacloud bude stavět na jeho API. Tento obraz opět bude open-source projekt s dokumentací. API bude zapouzdřovat práci s kontejnery a datasety, tím odstraníme veškeré duplicity v kódu.

Relační model databáze implementující nejdůležitější části systému je v příloze A. Model neobsahuje systém pro podporu ani tabulky týkající se knihoven použitých v kapitole 4.4.3.

4.4.3 Vývoj

S programováním v Ruby jsem už zkušenosti měl, avšak s Ruby on Rails jsem se musel nejdříve seznámit. Zezačátku pro mě bylo obtížné pracovat s Rails, protože všechno je postaveno na principu „konvence má přednost před konfigurací“. Je tedy nutné zapamatovat si nastavená pravidla.

Začal jsem vývojem webového rozhraní vytvořením v projektu Rails. Poté jsem hledal dostupné knihovny, které by se daly použít. Použity jsou knihovny na tvorbu uživatelského rozhraní, autentizaci, autorizaci, logování akcí a generování formulářů.

Bylo vytvořeno jednoduché rozhraní pro uživatele a administrátory, pouze se základními funkcemi jako seznam uživatelů, jejich přidání/odebrání/úprava a podobně pro prostředí na obrázku 1, lokace, servery a VPS, včetně využití historie a ACL.

Práci na tomto projektu jsem se věnoval 12 dnů.

4.5 Bitoxy

Práce na Bitoxy se časově prolínala s vývojem vpsAdmin2, jelikož se přišlo na to, že pokročilé logování aplikace neumí a je to potřeba.

Obrázek 2: Bitoxy - ukázka logu přístupů

Za úkol jsem dostal přidat do programu logování přístupu uživatelů, podobně jako je to např. ve webovém serveru Apache2 či nginx. Logy se mají posílat do syslogu, protože pak lze využít přeposílání logů po síti a nemusíme to implementovat sami.

Konfigurační soubor Bitoxy je ve formátu INI. Lze vytvořit více proxy serverů s různým nastavením. Pro nastavení logů jsem vytvořil novou sekci v konfiguraci. Každý log má svůj název a tím jej přiřadíme k proxy serveru. Můžeme tedy vytvořit několik logů a přiřazovat je k různým proxy serverům.

Každý log má definován formát, který může obsahovat proměnné, za které Bitoxy při logování dosadí konkrétní hodnoty.

S využitím tohoto formátování lze logovat buď do čistého textu, který je zobrazen na obrázku 2, nebo generovat ku příkladu JSON a ten pak jednoduše číst za pomoci knihovny.

```
[Log:syslog:MyLogger]
format = " $client_ip : $client_port  $worker/$conn_id/$cmd_id \"$cmd_str\" -> $bytes_sent \"
    $status_code $status_str\" <- $server_ip:$server_port [$encryption,$proxy,$user]"
```

Výpis 3: Konfigurace logování do čistého textu v Bitoxy

```
[Log:syslog:MyLogger]
format = "{ \"client\": {\"addr\": \" $client_ip \", \"port\": $client_port , \"connection_id\":
    $conn_id, \"command_id\": $cmd_id, \"command\": \"$cmd_str\"}, \"reply\": {\"status\": \"
    $status_code\", \"msg\": \"$status_str\", \"data\": $bytes_sent}, \"server:\" {\"addr\": \"
    $server_ip\", \"port\": $server_port}, \"encrypted\": \"$encryption\", \"proxied\": \"$proxy
    \", \"user\": \"$user\" }"
```

Výpis 4: Konfigurace logování do formátu JSON v Bitoxy

Jakékoliv úpravy mohou znamenat zavlečení nových chyb, které mohou vyústit v pád programu, což v produkčním prostředí znamená problémy. Vytvořil jsem proto jednoduché skripty v Bashi, které simulují zátěž. Na proxy se najednou připojuje několik desítek až stovek klientů a vypisují adresář. Tímto testem jsem eliminoval základní chyby, které vedly např. ke špatnému přístupu do paměti.

Po dokončení implementace byl vytvořen RPM balíček pro instalaci na produkční servery. Specifikace tohoto balíčku je součástí zdrojových kódů Bitoxy.

V produkci jsme narazili pouze na jeden problém, kdy po několika týdnech až měsíci provozu dojde k pádu aplikace. Tato chyba už je tam, zdá se, delší dobu a ještě se jí nepodařilo odchytit debuggerem.

Na obrázku 2 je zobrazen úryvek z logu přístupů. Program zachytil nové připojení a pokus o přihlášení, který skončil neúspěšně z důvodu špatných přihlašovacích údajů.

Práci na tomto projektu jsem se věnoval 8 dnů.

5 Výsledky

5.1 Chybějící znalosti

Během praxe jsem musel oprášit znalosti Ruby a s Ruby on Rails jsem začínal od začátku. Krom tohoto jsem měl o všem aspoň jakýsi základní přehled a detaily si rychle dostudoval.

Program pro synchronizaci dat nebyl z časových důvodů a nízké priority dokončen.

5.2 Získané zkušenosti a dovednosti

V praxi jsem nepřetržitě pracoval s linuxovými systémy, virtualizačními technologiemi a balíčkovacímy systémy. Navrhoval, spravoval a udržoval jsem MySQL databáze. Programoval jsem ve skriptovacích jazycích PHP, Ruby a Python. Naučil se základní koncepty a pravidla pro práci v Ruby on Rails. Pracoval jsem také v C++ spolu s rozsáhlým Qt frameworkem. Zabýval jsem se komunikací po počítačových sítích a aplikačními protokoly. Vyzkoušel jsem si práci s moderním systémem souborů ZFS. Naučil se pracovat s debuggerem v konzoli, generovat a číst ladící informace. V neposlední řadě jsem se naučil psát manuálové stránky pro UNIXové systémy.

Celou dobu jsem také využíval a zdokonalil své znalosti pasivní angličtiny. Veškerý vývoj, psaní dokumentace i samotné učení se nových věcí probíhá v angličtině, jelikož je v tomto jazyce dostupné největší a nejkvalitnější množství informací, alespoň co se týká programování a administrování.

5.3 Zhodnocení

Vykonával jsem práci, která mě opravdu baví a chtěl bych v ní pokračovat.

Také jsem byl překvapen, jak může mladý tým několika lidí zvládnout vytvořit tak komplexní produkt, jakým je eviacloud. Je za tím spousta dnů težké práce a trpělivosti.

Jakub Skokan

6 Reference

[1] THE PHP GROUP. *PHP: Hypertext Preprocessor* [online]. 2001 [cit. 2014-04-29]. Dostupné z: http://www.php.net/

[2] ORACLE CORPORATION. *MySQL* [online]. 2014 [cit. 2014-04-29]. Dostupné z: https://www.mysql.com/

[3] Apache License, Version 2.0. *The Apache Software Foundation* [online]. 2004 [cit. 2014-04-29]. Dostupné z: https://www.apache.org/licenses/LICENSE-2.0.html

[4] RELBIT, s.r.o. *PHP and MySQL Cloud Hosting at its best* [online]. 2014 [cit. 2014-04-29]. Dostupné z: https://eviacloud.com/

[5] THE RUBY COMMUNITY. *Ruby Programming Language* [online]. 1999 [cit. 2014-04-29]. Dostupné z: https://www.ruby-lang.org/

[6] GNU General Public License. FREE SOFTWARE FOUNDATION, Inc. *GNU General Public License* [online]. 2007 [cit. 2014-04-29]. Dostupné z: https://gnu.org/licenses/gpl.html

[7] RFC 959. *FILE TRANSFER PROTOCOL (FTP)*. ISI: IETF, 1985. Dostupné z: https://tools.ietf.org/html/rfc959

[8] RFC 4217. *SECURING FTP WITH TLS*. IBM UK Ltd: IETF, 2005. Dostupné z: https://tools.ietf.org/html/rfc4217

[9] PARALLELS. *OpenVZ Linux Containers Wiki* [online]. 2006 [cit. 2014-04-29]. Dostupné z: http://openvz.org/

[10] MARIADB FOUNDATION. *MariaDB, An enhanced, drop-in replacement for MySQL* [online]. 2014 [cit. 2014-04-29]. Dostupné z: https://mariadb.org/

[11] QT PROJECT HOSTING. *Qt Project* [online]. [cit. 2014-04-29]. Dostupné z: https://qt-project.org/

A Návrh relačního modelu pro vpsAdmin2

Relační model je rozdělen do tří kategorií.

A.1 Cluster

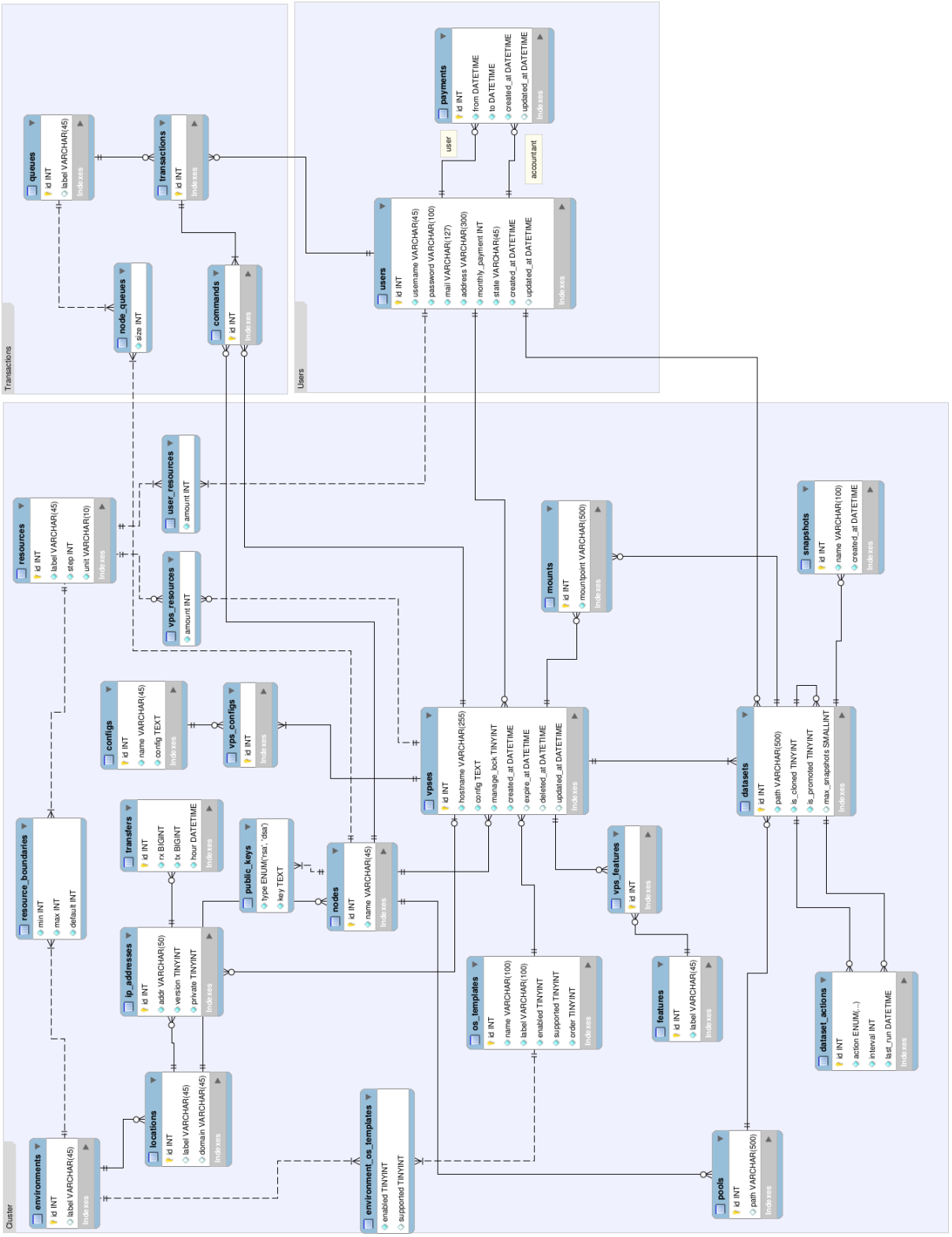
Obsahuje popis infrastruktury systému. Definuje prostředí, lokace, servery a VPS na nich. Dále seznam IP adres a počty přenesených dat, dynamické přidělování prostředků, konfigurace VPS, šablony VPS, datasety, jejich snapshoty, akce datasetů (přesun, replikace) a jejich připojení do VPS, speciální vlastnosti VPS (PPP, NFS, TUN/TAP, iptables).

A.2 Transakce

Reprezentuje příkazy ve webovém rozhraní, které se mají vykonat na serverech. Příkazy mohou být rozděleny do několika front, které se vykonávají paralelně. Každá fronta má omezený počet příkazů, které se uvnitř provádí paralelně.

A.3 Uživatelé

Seznam uživatelů a jejich plateb. Neřeší generování faktur.



Obrázek 3: vpsAdmin2

B Příloha na CD

Přiložené CD obsahuje adresář `Priloha`, v němž se nachází:

- `README` – seznam změněných souborů v jednotlivých projektech
- `bitoxy` – zdrojové kódy FTP/FTPS proxy Bitoxy
- `havesync` – zdrojové kódy synchronizačního nástroje
- `vpsadmin` – webové rozhraní vpsAdmin
- `vpsadminctl` – program pro ovládání vpsAdmind
- `vpsadmind` – serverová část vpsAdminu
- `vpsadmininstall` – instalátor vpsAdminu
- `vpsadmin2` – nekompletní implementace nové verze vpsAdminu