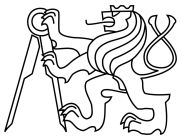
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ



ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Úprava prostředí pro průběžnou integraci projektu Clondike
Student:	Michael Hartman
Vedoucí:	Ing. Josef Gattermayer
Studijní program:	Informatika
Studijní obor:	Informační technologie
Katedra:	Katedra počítačových systémů
Platnost zadání:	Do konce letního semestru 2016/17

Pokyny pro vypracování

- Analyzujte současný automatický skript pro vytvoření překladu pro projekt Clondike.
- Nastudujte cloudové služby pro průběžnou integraci.
- Vyberte vhodnou službu pro překlad projektu Clondike z repozitářů na Githubu.
- Upravte současné sestavovací skripty pro vybranou službu.
- Implementované řešení otestujte měřením dle dohody s vedoucím práce.

Seznam odborné literatury

Dodá vedoucí práce.

L.S.

prof. Ing. Róbert Lórencz, CSc.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 31. ledna 2016

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra počítačových systémů

Bakalářská práce

Úprava prostředí pro průběžnou integraci projektu Clondike

Michael Hartman

Vedoucí práce: Ing. Josef Gattermayer

16. května 2016

Poděkování

Děkuji svému vedoucímu práce Ing. Josefu Gattermayerovi za laskavé vedení a konzultace. Rovněž děkuji svému otci Ing. Michalovi Hartmanovi, MBA, za podporu po celou dobu mého studia, zejména pak v jeho závěru. Celé své rodině a přátelům děkuji za toleranci během studia a povzbuzení či pomoc ve chvílích, kdy jsem to nejvíce potřeboval.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 16. května 2016 .....

Odkaz na tuto práci

Hartman, Michael. *Úprava prostředí pro průběžnou integraci projektu Clondike*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2016.

Abstrakt

Tato práce popisuje současné řešení pro průběžnou integraci projektu Clondike a představuje několik cloudových služeb, které mají potenciál současné řešení nahradit. V práci je dále popsán způsob nasazení a konfigurace jedné z popsaných cloudových služeb a úprava automatických sestavovacích skriptů projektu pro zvolenou cloudovou službu. Výsledkem práce je kompletní cloudové řešení průběžné integrace pro projekt Clondike. Rozšíření zvoleného řešení o testování a další služby je předmětem další práce kolegů, kteří se na projektu podílejí.

Klíčová slova Clondike, průběžná integrace, cloud, klastr

Abstract

This thesis describes current solution for continuous integration of Clondike project and introduces several cloud services potentially capable of replacing current solution. The thesis later describes configuration steps of one of the described cloud services and modification of Clondike project build scripts for chosen cloud service. The result of this thesis is complete cloud solution for continuous integration of project Clondike. Extending the soluton by testing and other services is the subject of further work of colleagues who are participating in project development.

Keywords Clondike, continuous integration, cloud, cluster

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Tato práce se zabývá řešením průběžné integrace projektu Clondike. Clondike je projekt, jehož cílem je distribuce výpočetně náročných úloh mezi více výpočetních uzlů. Tyto výpočetní uzly, na rozdíl od obdobných projektů, nejsou pro tento účel plně vyhrazeny. Projekt je implementován jako záplata do Linuxového jádra.

Průběžná integrace je soubor technik a nástrojů, který urychluje vývoj softwaru. Vyznačuje se častým zanášením změn zdrojových kódů do centrálního repozitáře a automatickým sestavením a testováním projektu při každé takovéto změně.

V první polovině práce je věnován prostor analýze procesu sestavení a integrace projektu Clondike a současného řešení jeho automatizace lokálním nástrojem pro průběžnou integraci BuildBot. Dále je proveden průzkum cloudových služeb pro průběžnou integraci s důrazem na využitelnost pro projekt Clondike.

Motivací pro zajištění průběžné integrace cloudovou službou je vysoká dostupnost služby a také bezpečnost dat ve smyslu zálohování. Projektový tým se rovněž nemusí zabývat správou hardware pro provoz lokálního řešení průběžné integrace a administrací souvisejícího programového vybavení.

Druhá polovina práce je věnována popisu nasazení vybrané cloudové služby a její konfigurace pro zajištění průběžné integrace projektu Clondike. Konfigurací se rozumí i úprava automatických sestavovacích skriptů projektu Clondike pro funkčnost s danou cloudovou službou.

Použití vybrané cloudové služby pro průběžnou integraci zajistí přehledný záznam změn zdrojových kódů a jejich důsledků. Vývojáři budou o těchto důsledcích informováni a budou mít k dispozici obraz virtuálního disku s operačním systém, ve kterém bude integrována vždy aktuální verze projektu Clondike, jehož sestavení zajistí integrační služba.

Funkčnost řešení průběžné integrace cloudovou službou je v rámci práce ověřena testováním a měřením.

Kapitola 1

Cíl práce

Hlavním cílem práce je nahrazení současného řešení pro průběžnou integraci projektu Clondike cloudovou službou. Pro naplnění cíle práce je nejdříve analyzováno současné řešení pro průběžnou integraci projektu Clondike. Analyzován a propsán je především nástroj realizující průběžnou integraci. Zároveň jsou podrobně prostudovány a popsány automatické sestavovací skripty, které realizují integrační proces.

Dále je provedena analýza cloudových služeb pro průběžnou integraci. Práce představuje některé cloudové služby, které splňují nefunkční požadavky projektu Clondike (zejména provoz zdarma). Následně je zhodnocena jejich využitelnost pro projekt Clondike a na základě jejich porovnání zvolena pro projekt Clondike ta nejvhodnější.

V závěrečné části práce je popsán proces získání licence pro projekt Clondike a konfigurace dané služby. Součástí procesu konfigurace je úprava automatických sestavovacích skriptů, které realizují sestavení a integrační proces projektu Clondike.

1.1 Představení projektu Clondike

Název projektu - „Clondike“ - je akronymem vyjadřujícím „CLuster Of Non-Dedicated Interoperating KErnels“ [4], což volně přeloženo znamená klastr nevyhrazených kooperujících jader. Myšlenka projektu je ve sdílení výkonu počítačů s operačním systémem Linux. Pokud některý z počítačů v klastru začne vykonávat výpočetně náročné úlohy a zároveň výkon jednoho nebo více dalších počítačů z klastru není plně využit, je tato úloha migrována na jeden z těchto počítačů, kde je proveden výpočet a výsledek zaslán zpět na původní počítač. Tím je lépe využit výkon celého klastru a zkrácen čas potřebný na dokončení výpočtu.

Klastr Clondike nemá hiearchickou strukturu, to znamená, že všechny počítače zapojené do klastru spolu vzájemně komunikují a je možné je do klastru kdykoliv libovolně přidávat a zase z něj odebírat. Na každém z počítačů tedy

běží zároveň server i klient a každý počítač klastru poskytuje svůj výpočetní výkon a zároveň může využívat výpočetního výkonu celého klastru [35].

1.2 Motivace

Vývoj projektu Clondike probíhá už od roku 2006 [4]. Na vývoji se podílí zejména studenti ČVUT FIT a ČVUT FEL v rámci závěrečných prací. Tento způsob vývoje způsobuje značnou fluktuaci a velké množství vývojářů, kteří do projektu přispívají. Smyslem nasazení jednoduché a přehledné služby průběžné integrace je usnadnění a urychlení procesu vývoje a testování projektu. Použití integrační služby provozované v cloudu zároveň zajišťuje vysokou dostupnost služby a bezpečnost (ve smyslu zálohování) dat a použitých automatických sestavovacích skriptů

1.3 Definice pojmů

Artefakt je v kontextu služby Bamboo soubor vytvořený během integračního procesu [2].

Bash je shell kompatibilní s shellem sh, který začlenil užitečné rozšíření některých jiných shellů [41].

Cloud je množina síťových služeb poskytující škálovatelnou, většinou personalizovanou, levnou výpočetní platformu dostupnou na vyžádání s garantovanou kvalitou služeb, která je dostupná snadnou a snadno šiřitelnou cestou [44].

Distribuce je koherentní sestavení všech komponent systému GNU/Linux, od knihoven a systémových komponent, přes jádro a ovladače až po aplikace, které bylo rozsáhle otestováno [27].

Git je verzovací systém. Je to program, který slouží pro sledování změn v souborech v času a umožňuje spolupráci na projektu více vývojářům [26].

GitHub je webová stránka sloužící ke správě repozitáře Git. Umožňuje jednodušší spolupráci na projektu poskytováním centrálního úložiště dat, webového rozhraní a dalších vlastností a nástrojů, které podporují efektivní spolupráci a vývoj [26].

Interpret je program, který překládá a provádí příkazy jazyka vyšší úrovně jeden po druhém [9].

Jádro je základní komponenta operačního systému. Chová se jako můstek mezi aplikacemi a zpracováním dat prováděným na úrovni hardwaru, k čemuž používá systémová volání a komunikaci mezi procesy [21].

Knihovna je kolekce standardních programů a rutinních postupů, které jsou uložené v souboru a připravené k okamžitémů použití [10].

Kořenový adresář je nejvyšší umístění v hierachickém souborovém systému. V Unixových souborových systémech je označen znakem dopředného lomítka „/“ [42].

Modul je část kódu, která může být do jádra zaváděna a zase z něj odebírána dle aktuálních požadavků. Rozšiřuje funkčnost jádra bez nutnosti restartování operačního systému [40].

Operační systém je nejdůležitější program běžící na počítači. Každý počítač s obecným účelem použití musí mít operační systém pro spouštění dalších programů a aplikací. Operační systém vykonává základní úkony mezi které patří rozpoznávání vstupu z klávesnice, vysílání výstupu na obrazovku a správa souborů a adresářů na disku [13].

Prostor jádra je část paměti, ve které je spuštěno jádro operačního systému a ve které poskytuje své služby [23].

Platforma je systém, na kterém mohou být spouštěny aplikace. Platformu tvoří operační systém a počítač, na kterém běží [12].

Portace je instalace programu vyžadující nezdokumentované změny nezbytné k adaptaci pro nové prostředí [34].

Linux je komunitně vyvíjený operační systém [22].

Repozitář je server, na kterém jsou uloženy všechny sledované soubory včetně historie jejich změn [37].

Scp je program pro zabezpečené kopírování souborů mezi počítači na počítačové síti [36].

Shell je základní rozhraní pro uživatele a aplikace pro interakci s operačním systémem Linux a jádrem [31].

Sh je původní Unixový shell [43].

Skript je program nebo sekvence instrukcí, která je interpretována nebo prováděna jiným programem na místo počítačového procesoru [24].

Socket je jeden z konců obousměrného komunikačního kanálu mezi dvěma programy komunikujícími přes počítačovou síť [20].

Token je řetězec nebo objekt, který může reprezentovat autentikační údaje [15].

Uživatelský prostor je část paměti vyhrazená pro aplikace v počítači, na kterém běží operační systém typu Linux. Takové operační systémy mají vyhrazenou část paměti pro jádro a jinou vyhrazenou část pro uživatelské aplikace [6].

Záplata je malý textový dokument, který obsahuje definici změn mezi dvěmi různými verzemi zdrojového kódu [33].

Kapitola 2

Analýza současného řešení

Projekt Clondike se sestává ze dvou hlavních částí, které je nutno sestavovat zvlášť. První část projektu je implementována v prostoru jádra a při její aktualizaci je nutno sestavit celé nové jádro operačního systému, neboť je implementována jako záplata (v případě implementace jako modul by to nutné nebylo). Druhá část je implementována v uživatelském prostoru a její sestavení není závislé na první části.

2.1 Požadavky na prostředí pro sestavení projektu Clondike

2.1.1 Společné požadavky

Projekt Clondike je implementován pro platformu Debian Linux. Portace na jinou distribuci z rodiny operačních systémů s jádrem Linux je možná, ale není předmětem této práce, proto se práce zabývá pouze sestavením a integrací do této konkrétní platformy. Sestavení projektu je možné provést i na jiné příbuzné distribuci, následující požadavky však předpokládají použití distribuce, ve které jsou k dispozici základní Linuxové nástroje a uživatelský `shell Bash` (Bourne Again Shell) nebo `sh` (Bourne shell). Obě komponenty projektu Clondike zároveň k zajištění správného překladu vyžadují následující nástroje.

- Git - Klient verzovacího systému Git (pouze pokud nemáme k dispozic zdrojové kódy projektu)
- Gcc - Překladač zdrojových kódů jazyka C/C++ (pro sestavení jádra 3.6.11 maximálně verze 4.4)
- Make - Program pro automatizaci překladu zdrojových kódů

Bez těchto nástrojů není možné sestavit žádnou komponentu projektu Clondike.

2.1.2 Požadavky na část pracující v prostoru jádra

Pro sestavení jádra se záplatou implementující projekt Clondike jsou v systému vyžadovány tyto další nástroje:

- initramfs-tools - Nástroj pro vytvoření initramfs
- ncurses-dev - Knihovna ncurses

Bez těchto nástrojů není možné sestavit jádro Linuxu s funkčností Clondike.

2.1.3 Požadavky na část pracující v uživatelském prostoru

Část projektu pracující v uživatelském projektu je implementována v jazyce C, zbývající část je implementována v jazyce Ruby. Pro jeho běh je tedy potřeba v systému interpret jazyka Ruby, pro samotné sestavení ale není vyžadován. Pro sestavení jsou kromě společných nástrojů pro obě části vyžadovány tyto další knihovny:

- libnl1 - Knihovna pro aplikace používající sockety `netlink`
- libnl-dev - Vývojářská knihovna a hlavičkové soubory pro knihovnu `libnl`

Bez těchto dodatečných knihoven nemůže být sestavena část projektu Clondike pracující v uživatelském prostoru.

2.2 Postup sestavení projektu Clondike

2.2.1 Prostor jádra

2.2.1.1 Záplata jádra

Pro aplikaci záplaty jádra implementující funkčnost Clondike je třeba mít k dispozici zdrojové kódy jádra Linux a záplatu Clondike pro odpovídající verzi jádra. Zdrojové kódy jádra Linux jsou dostupné z `www.kernel.org`. Záplaty jádra jsou k dispozici v repozitáři GitHub projektu Clondike ve složce patches (`https://github.com/FIT-CVUT/clondike/tree/master/patches`).

Pro aplikaci záplaty do jádra musí být nastaven pracovní adresář na adresář se zdrojovými kódy jádra Linux. Aplikace záplaty se provede příkazem:

```
# patch -p1 < cesta_k_zaplate
```

2.2.1.2 Sestavení jádra

Pro sestavení jádra je třeba k dispozici konfigurační soubor `.config`. Je možné získat a použít soubor `.config` z repozitáře GitHub projektu Clondike (nachází se v adresáři sources a příslušném podadresáři dle verze jádra, například `kernel_3.6.11`), případně vytvořit vlastní pomocí příkazu

```
# make menuconfig
```

Volby potřebné pro úspěšné sestavení jádra s funkčností Clondike jsou popsány v [39].

Do podadresáře clondike v adresáři se zdrojovými kódy jádra Linux je třeba umístit zdrojové kódy Clondike z adresáře `sources/kernel_verze/src` v repozitáři GitHub projektu Clondike.

Před samotným sestavením jádra se příkazem make clean vyčistí adresář od případných starých objektových souborů. Sestavení jádra je provedeno příkazem make s volitelným přepínačem -jx, kde x je číslo specifikující případnou úroveň paralelismu. Více vizte `man make` [28].

2.2.1.3 Sestavení modulů

Sestavení modulů jádra Linux se provede spuštěním příkazu make modules v pracovním adresáři se zdrojovými kódy jádra Linux.

2.2.1.4 Vytvoření initramfs

Soubor `initramfs` je vytvořen příkazem `mkinitramfs`, kterému je jako argument parametru `-o` předáno jméno výstupního souboru a dále mu je jako argument předáno číslo verze jádra, pro kterou je vytvářen soubor initramfs. Více o syntaxi a parametrech příkazu `mkinitramfs` v manuálu `mkinitramfs` [25].

2.2.2 Uživatelský prostor

2.2.2.1 Sestavení Director API

Zdrojové kódy a hlavičkové soubory komponenty Director API pracující v uživatelském prostoru se nacházejí v adresáři `clondike/userspace/director-api`. Tato komponenta je implementována v jazyce C, v konečném důsledky je tedy pro její sestavení použit překladač gcc. Pro usnadnění procesu sestavení je ale používán nástroj Make. V adresáři se zdrojovými kódy je k tomuto účelu poskytnut konfigurační soubor `Makefile`, ve kterém jsou definovány závislosti tříd a postup sestavení komponenty. Sestavení komponenty Director API je provedeno spuštěním příkazu `make` v pracovním adresáři se zdrojovými kódy komponenty.

2.2.2.2 Sestavení Ruby Director API

Zdrojové kódy komponenty Ruby Director API se nacházejí v adresáři `clondike/userspace/ruby-director-api`. Tato komponenta je implementována v jazyce Ruby a pro její sestavení je rovněž využíván nástroj Make. Sestavení je provedeno spuštěním příkazu make v pracovním adresáři se zdrojovými kódy komponenty.

2.2.2.3 Sestavení NPFS Serveru

Zdrojové kódy projektu NPFS Server se nacházejí v adresáři `clondike/root/npfs_install`. V adresáři je rovněž konfigurační soubor `Makefile`. Sestavení projektu je provedeno příkazem `make` v pracovním adresáři se zdrojovými kódy NPFS serveru.

2.3 Integrace projektu Clondike

2.3.1 Požadavky na integraci

2.3.1.1 Soubor obrazu disku s OS pro integraci

Projekt Clondike je implementován pro distribuci Debian s jádrem Linux 3.6.11. Pro jeho integraci do operačního systému nástroj potřebuje soubor s obrazem disku, na kterém je nainstalovaný tento operační systém.

2.3.1.2 Nástroj VMware

Pro práci s obrazem disku vyžaduje nástroj na průběžnou integraci Clondike přítomnost nástroje VMware. Manipulace s virtuálním diskem (jeho připojení, přístup k němu a odpojení) je zajištěno nástrojem VMware, který lze obsluhovat uživatelského shellu.

2.3.1.3 Uživatelská práva

Nástroj pro průběžnou integraci v procesu integrace připojuje a odpojuje virtuální disk, dočasně mění umístění kořenového adresáře a provádí další pokročilé operace. K zajištění práv k provedení těmto operacím musí být zajištěna možnost nástroje v průběhu integrace eskalovat svá práva na úroveň superuživatele `root`.

2.3.2 Proces integrace

2.3.3 Prostor jádra

2.3.3.1 Instalace jádra

Před instalací jádra je pracovní adresář nastaven na adresář se sestaveným jádrem Linux s funkčností Clondike. Jádro je nainstalováno pomocí příkazu

```
# make install
```

provedeného v pracovním adresáři. Tím jsou soubory sestaveného jádra umístěny do adresáře `/boot`.

2.3.3.2 Instalace modulů

Instalace modulů je provedena příkazem

```
# make modules_install
```

v tomtéž pracovním adresáři. Tím jsou soubory sestavených modulů umístěny do složky `/lib/modules/verze`, kde verze je příslušná verze jádra, pro kterou byly moduly sestaveny.

2.3.3.3 Instalace initramfs

Instalace souboru initramfs je provedena umístěním do kořenového adresáře disku s nainstalovaným operačním systémem pro integraci projektu Clondike.

2.3.3.4 Aktualizace zavaděče

V operačním systému do kterého je projekt Clondike integrován je třeba provést aktualizaci zavaděče například příkazem `update-grup` nebo `update-grub2`.

2.3.4 Uživatelský prostor

2.3.4.1 Director API

Sestavenou komponentu Director API je třeba umístit to adresáře `/root/userspace/director-api` operačního systému pro integraci projektu Clondike.

2.3.4.2 Ruby Director API

Sestavenou komponentu Ruby Director API a konfigurační soubor Makefile je třeba umístit do adresáře `/root/userspace/ruby-director-api` operačního

systému pro integraci projektu Clondike. Po umístění všech potřebných souborů je třeba v daném adresáři spustit příkaz

```
# make install
```

Tím je integrace komponenty Ruby Director API dokončena.

2.3.4.3 NPFS Server

Sestavený projekt i zdrojové kódy projektu NPFS server je třeba umístit do adresáře `/root/npfs_install` operačního systému pro integraci projektu Clondike. Do adresáře `/root` operačního systému pro integraci je třeba umístit konfigurační soubor `.migration.conf` z adresáře `clondike/root` v repozitáři GitHub. Do adresáře `/root/npfs` umístíme spouštěcí skript NPFS serveru `npfs-start.sh` z adresáře `clondike/root/npfs`.

2.4 Současné řešení automatizace (Buildbot)

Současné řešení průběžné integrace projektu Clondike využívá nástroj BuildBot. Postup sestavení a integrace Clondike je nastaven v konfiguračním souboru master.cfg řídící části (Build Mater) nástroje BuildBot. Obě části nástroje, tedy jak řídící část BuildMaster, tak část přímo vykonávající sestavování a integraci BuildSlave, běží na virtuálním serveru hostovaném FIT ČVUT.

2.4.1 Stažení zdrojových kódů z repozitáře

Prvním krokem je stažení aktuálních zdrojových kódu projektu Clondike z repozitáře GitHub (`https://github.com/FIT-CVUT/clondike.git`). Nástroj uloží všechny stažené soubory z repozitáře do adresáře `build/clondike`. Pro stahování je použit přírustkový (incemental) mód, což proces zrychluje a šetří datové přenosy, protože se stahují jen soubory u nichž došlo ke změně.

2.4.2 Skript precompile.sh

Po stažení souborů z repozitáře je v pracovním adresáři build/scripts spuštěn skript precompile.sh příkazem

```
# ./precompile.sh 2>../logs/precompileerr.log
```

Tímto způsobem je standardní chybový výstup skriptu přesměrován do souboru `precompileerr.log` v adresáři `build/logs`.

Skript je implementován v jazyce Bash, což je patrné z první řádky skriptu, na které je použitý shell definovaný zápisem

```
#!/bin/bash
```

Na začátku skriptu jsou deklarované a definované funkce pro výpis informativních a chybových hlášení. Po deklaraci a definici funkcí následuje deklarace a definice proměnných. V proměnných je definována cesta k adresáři pro logování, cesta k chybovému logu, verze linuxového jádra a cesta pro připojení virtuálního disku. Následuje několik kroků, kde po vykonání každého z nich skript informuje o provedené akci a jejím výsledku hlášením pomocí výše definovaných funkcí a v případě neúspěchu se ukončí.

V dalším kroku skript zkontroluje existenci adresáře `build/logs` pro logování a v případě jeho neexistence jej vytvoří. Po jeho vytvoření mu nastaví práva pro čtení, zápis i spouštění jeho vlastníkovi, všem ostatním nastaví práva jen čtení a spouštění.

Dále skript zkontroluje existenci souboru `precompileerr.log` v adresáři `build/logs`. V případě jeho neexistence jej vytvoří a nastaví práva pro čtení a zápis vlastníkovi souboru, všem ostatním nastaví práva pro čtení.

Po zajištění prostředí pro logování skript zkontroluje přítomnost adresáře se zdrojovými kódy linuxového jádra. V případě jeho existence adresář smaže.

Dále skript zkontroluje existenci archívu se zdrojovými kódu linuxového jádra. Pokud takový archív neexistuje, tak jej stáhne pomocí programu wget z úložiště dostupného na `www.kernel.org`.

Po zajištění přítomnosti archívu se zdrojovými kódy linuxového jádra skript tento archív rozbalí.

Následně se skript přepne do adresáře s rozbalenými zdrojovými kódy linuxového jádra a na jádro aplikuje odpovídající záplatu z adresáře `build/clondike/patches`.

Po aplikaci záplaty do linuxového jádra skript zkopíruje zdrojové soubory Clondike ze složky `build/clondike/sources/kernel_linuxver/src` do složky `clondike` v adresáři zdrojových kódů linuxového jádra.

Nakonec skript zkopíruje konfigurační soubor jádra ze složky `build/clondike/sources/kernel_linuxver` do složky se zdrojovými kódy linuxového jádra pod názvem `.config`.

Pokud všechny následující kroky proběhnou v pořádku, skript o tom vypíše informativní hlášení a ukončí se.

2.4.3 Připojení virtuálního disku

Po skončení skriptu precompile.sh nástroj BuildBot připojí virtuální disk, do kterého bude následně integrovat projekt Clondike. Připojení virtuálního disku je provedeno pomocí skriptu mount.sh, který je spuštěn příkazem

```
# sudo ./mount.sh 2>../../logs/mounterr.log
```

v pracovním adresáři `build/scripts/sudoscripts`. Tímto způsobem spuštění je zajištěn běh skript s efektivními oprávněními uživatele `root` a přesměrování standardního chybového výstupu do souboru `mounterr.log` v adresáři `build/logs`.

Skript `mount.sh` je stejně jako skript `precompile.sh` implementován v jazyce Bash a na jeho začátku jsou deklarovány a definovány funkce pro výpis informativních a chybových hlášení a proměnné.

První akcí skriptu mount.sh je test, zda je virtuální disk připojen. V případě, že ano, tak zavolá skript `umount.sh` (který disk odpojí) vypíše o tom informační hlášku. V případě, že disk připojený není neudělá skript nic.

V dalším kroku skript otestuje existenci souboru virtuálního disku a pokusí se vypsat jeho kořenový adresář.

Nakonec skript připojí virtuální disk do složky definované proměnnou na začátku skriptu.

2.4.4 Příprava adresáře pro sestavení jádra

Po připojení virtuálního disku nástroj BuildBot připraví adresář se zdrojovými kódy linuxového jádra na sestavení jádra zavoláním příkazu

```
# make clean 2>../../logs/makecleanerr.log
```

Důsledkem spuštění tohoto příkazu je zmazání všech starých objektových souborů z případných předchozích pokusů o sestavení jádra. Obsah standardního chybového výstupu je přesměrován do souboru `makecleanerr.log` ve adresáři `build/logs`.

2.4.5 Sestavení jádra

Po přípravě adresáře se zdrojovými kódy linuxového jádra pro kompilaci nástroj BuildBot zahají kompilaci jádra příkazem

```
# make -j4 2>../../logs/makeerr.log
```

Jádro se začne sestavovat ve čtyřech paralelně běžících vláknech a obsah standardního chybového výstupu je přesměrován do souboru `makeerr.log` v adresáři `build/logs`.

2.4.6 Sestavení modulů jádra

Po sestavení linuxového jádra nástroj BuilBot zahájí sestavování modulů jádra příkazem

```
# make modules 2>../../logs/makemoduleserr.log
```

Obsah standardního chybového výstupu je přesměrován do souboru `makemoduleserr.log` v adresáři `build/logs`.

2.4.7 Skript postcompile.sh

Skript `postcompile.sh` je rovněž implementován v jazyce Bash. Na jeho začátku jsou deklarované a definované stejné funkce pro výpis informativních a varovných hlášení jako v případě skriptů `precompile.sh`, `mount.sh` a `umount.sh`. Skript implementuje několik kroků, přičemž po provedení každého z nich zkontroluje návratovou hodnotu a na základě ní oznámí úspěch a pokračuje, případně vypíše chybovou hlášku, zavolá skript `umount.sh` (který odpojí připojený virtuální disk) a ukončí se.

V prvním kroku dojde k zavolání skriptu `mount.sh`, který připojí virtuální disk s operačním systémem pro integraci projektu Clondike do umístění definovaného proměnnou `mountpoint` na začátku skriptu. V tomto kroku ve skutečnosti dojde k odpojení a opětovnému připojení disku, neboť disk již byl připojen v jednom z předchozích kroků. Na funkčnost řešení to však nemá vliv.

Po připojení virtuálního disku skript smaže z jádra operačního systému na tomto virtuálním disku všechny jaderné moduly z umístění `/lib/modules/$linuxver`.

Když jsou všechny staré moduly jádra odstraněny, provede skript instalaci nových modulů příkazem

```
# make INSTALL_MOD_PATH=${mountpoint} modules_install
```

Po instalaci modulů skript do operačního systému na připojeném virtuálním disku nakopíruje soubory nového sestaveného jádra.

Po nakopírování nového jádra do virtuálního disku jsou dále nakopírovány také soubory projektu Clondike (sestavený projekt, řídící skripty a další soubory projektu pracující v uživatelském prostoru).

Dále jsou skriptem připraveny adresáře pro `npfs` a konfigurační skripty projektu Clondike.

V dalším kroku skript provede sestavení Director API sekvencí příkazů `make clean` a `make` v adresáři `clondike/userspace/director-api`, kde je k tomuto účelu připravený `Makefile`.

Po sestavení Director API skript stejným způsobem sestaví Ruby director API. Sekvence příkazů make clean a make je v tomto případě spuštěna v adresáři `clondike/userspace/ruby-director-api`, kde je opět připravený `Makefile`.

Skript pokračuje sestavením NPFS serveru. Jako pracovní adresář nastaví `/root/npfs_install` ve virtálním disku a zde spustí sekvenci příkazů `make` a `make install`.

Po dokončení sestavení NPFS serveru skript zkopíruje spouštěcí skript a konfigurační soubory upravené pro potřeby projektu Clondike do adresáře se sestaveným NPFS serverem.

Posledním krokem souvisejícím s integrací projektu Clondike do operačního systému na připojeném virtuálním disku je vytvoření initramfs. To skript zajistí zavoláním příkazu

```
# chroot ${mountpoint} mkinitramfs \
# -o /boot/initrd.img-${linuxver} ${linuxver}
```

Ten nejprve dočasně přenastaví kořenový adresář našeho systému na kořenový adresář virtuálního disku a následně se `initramfs` postará o vytvoření `initram` disku.

Po dokončení integrace projektu Clondike do operačního systému na virtuálním disku skript pokračuje vytvořením souboru obsahujícího samé logické nuly, který zabere celý zbytek virtuálního disku a následně tento soubor smaže. Tímto způsobem je zajištěna maximální úroveň budoucí komprimace souboru virtuálního disku libovolnou metodou i bez znalosti vnitřní struktury souboru a souborového systému virtuálního disku.

Pokud všechny předchozí kroky skončili úspěšně, tak o tom skript vypíše informační hlášku a ukončí se s návratovým kódem 0 značícím úspěch.

2.4.8 Odpojení virtuálního disku

Odpojení virtuálního disku je provedeno spuštěním skriptu `umount.sh` pomocí příkazu `sudo`, zajišťujícího potřebnou eskalaci práv a standardní chybový výstup je přesměrován do souboru `umounterr.log` v adresáři `build/logs`.

Skript umount.sh zavolá nástroj vmware a pomocí parametru -d mu předá cestu, kam byl disk připojen. Nástroj vmware následně virtuální disk odpojí. V případě neúspěchu o tomto skript vypíše varovné hlášení a ukončí se, v případě úspěchu vypíše informativní hlášení a rovněž se ukončí.

2.4.9 Komprese virtuálního disku

Když je virtuální disk úspěšně odpojen, je nástrojem BuilBot spuštěn skript compress.sh pomocí příkazu sudo zajišťujícího potřebnou eskalaci práv a standardní chybový výstup je přesměrován do souboru `compresserr.log` v adresáři `build/logs`.

Skript `compress.sh` pro kompresi souboru virtuálního disku používá nástroj `gzip`. Nástroji jako parametry jméno souboru virtuálního disku a jméno

výstupního archivu, pomocí parametru `-c 9` nastaví nejvyšší úroveň komprimace.

Po dokončení komprimačního procesu skript nastaví výstupnímu archivu práva pro čtení a zápis vlastníkem, všem ostatním nastaví práva pro čtení. Poté skript vypíše informační hlášení a ukončí se.

Kapitola 3

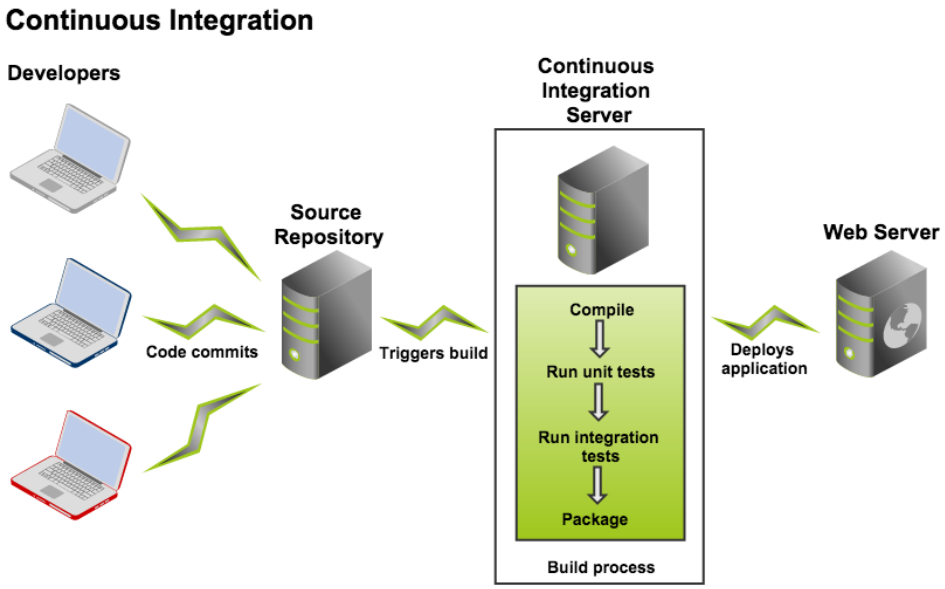
Analýza služeb pro průběžnou integraci

3.1 Cloudové služby pro průběžnou integraci

Podle [29] průběžnou integrací rozumíme sadu postupů vedoucí od změn ve zdrojových kódech projektu, přes jeho sestavení a nasazení až po jeho testování. Výhodou průběžné integrace je průběžné testování kódu, eliminace problémů s nasazením při dokončení projektu a v důsledku zrychlení vývoje. Pro nasazení systému průběžné integrace je zároveň nutné splnit několik základních prerekvizit, aby bylo možné plně využít výhod tohoto postupu. Patří mezi ně:

- Udržování jediného repozitáře zdrojových kódů pro celý projekt
- Časté zanášení změn zdrojových kódů do repozitáře
- Automatizace procesu sestavení projektu
- Automatizace testovacího procesu
- Sestavování projektu ve stejném jako produkčním prostředí
- Rychlé opravy chyb zanesených do projektu změnami zdrojových kódu

Splnění všech výše definovaných požadavků klade nároky na vývojářský tým v průběhu celé práce na projektu. Zřízení repozitáře, automatizace procesu sestavení a testování jsou většinou zátěží zejména na začátku procesu vývoje a v jeho pozdější fázi už jsou jen mírně upravovány (zejména testovací část) [38]. Požadavek na časté zanášení změn zdrojových kódů do repozitáře a rychlé opravy chyb tímto způsobených ale vyžaduje disciplínu týmu po celou dobu procesu vývoje. Obecný přehled procesů průběžné integrace ilustruje obrázek 3.1.



Obrázek 3.1: Obecný přehled procesů průběžné integrace [19]

Nástroje pro průběžnou integraci mohou být provozovány přímo na technickém vybavení subjektu, který se stará o vývoj daného projektu nebo mohou být provozovány nějakou třetí stranou tzv. v cloudu. V případě provozu nástroje třetí stranou nevznikají subjektu stojícímu za vývojem daného projektu žádné další nároky na technické vybavení a nástroj je mu poskytován jako služba [5]. Tato třetí strana pak zpravidla službu průběžné integrace poskytuje za úplatu. Typickým modelem je pronájem s měsíční či roční frekvencí placení za tuto službu. Některé z těchto služeb však nabízejí poskytnutí nástroje pro průběžnou integraci pro projekty s otevřeným zdrojovým kódem zdarma. Tato práce se zabývá společným projektem FEL a FIT ČVUT, jehož zdrojový kód je otevřený. Práce se věnuje pouze cloudovým službám pro průběžnou integraci, které program zdarma pro projekty s otevřeným zdrojovým kódem nabízejí, což byl jeden z požadavků vedoucího práce.

3.2 Travis CI

3.2.1 Prostředí pro sestavení

Každý pokus o sestavení a integraci projektu probíhá v novém prostředí, neovlivněným předchozími procesy. Jako prostředí je vždy použit nový virtuální server s operačním systémem dle konfigurace. Travis CI nabízí čtyři různá prostředí, která se liší operačním systémem, rychlostí spuštění virtuálního stroje,

použitým souborovým systémem a dostupnou pamětí. Pro sestavení projektu Clondike je nejdůležitější právě použitý operační systém. Travis CI nabízí tři různé možnosti:

- Ubuntu 12.04 LTS Server Edition
- OS X Mavericks
- Ubuntu 14.04 LTS Server Edition 64 bit

Vzhledem k požadavkům na sestavení projektu Clondike je možné uvažovat pouze prostředí s operačním systémem Ubuntu, který splňuje vlastnosti operačních systémů z rodiny Linux, obsahuje základní Linuxové nástroje a oba uživatelské shelly Bash a sh. Další parametry prostředí už nejsou klíčové, respektive všechny jsou kompatibilní s procesem průběžné integrace projektu Clondike.

Výhodou použití nového virtuálního stroje pro každou iteraci průběžné integrace je zaručené neovlivnění právě probíhající integrace žádným z předchozích běhů.

Nevýhodou je naopak úprava prostředí při každé iteraci integračního procesu. V případě integrace projektu Clondike je to zejména instalace nástroje VMware a stažení (a na konci procesu opět nahrání) virtuálního disku s operačním systémem pro integraci projektu Clondike. Podrobněji jsou požadavky na prostředí popsány v kapitole 2.1.

3.2.2 Program pro projekty s otevřeným zdrojovým kódem

Nástroj Travis CI podporuje projekty s otevřeným zdrojovým kódem plánem, který uživatele prakticky neomezuje a je zdarma. Projektům s otevřeným zdrojovým kódem je služba poskytnuta neomezenou dobu s jediným omezením, tzv. fair use. Znamená to, že uživatel služby by se ji měl snažit využívat rozumně a nesnažit se ji záměrně a neefektivně přetěžovat. Tímto způsobem by měl zohlednit fakt, že je mu služba poskytována zdarma, stejně tak jako dalším uživatelům, kteří ji využívají pro průběžnou integraci svých projektů. Programově ale uživatel není omezen počtem paralelních procesů sestavení, počtem repozitářů ani počtem přispěvatelů do projektu.

3.3 Circle CI

3.3.1 Prostředí pro sestavení

Produkt pro průběžnou integraci jako testovací prostředí poskytuje virtuální stroj s následujícími základními parametry [18]:

- Operační systém Ubuntu 12.04

- Architektura 64 bit
- Jádro Linux verze 3.2
- Kompilátory gcc a g++ verze 4.6
- Nástroj Make verze 3.81

Nejdůležitější požadavek na operační systém tedy produkt Circle CI s prostředím poskytujícím prostředí založené na Linuxové distribuci Ubuntu splňuje. Pro splnění dalších požadavků na integrační proces projektu Clondike musí být testovací prostředí produktu Circle CI modifikováno a doplněno o další nástroje.

Nástroj Circle CI poskytuje mnoho předpřipravených nástrojů pro různě programovací jazyky. V případě kompilátorů gcc a g++ jsou to verze 4.6 až 4.9. Pro sestavení a integraci projektu Clondike s jádrem Linux 3.6.11 ale může být použit kompilátor gcc nejvýše verze 4.4 [39]. Úprava testovací prostředí Circle CI a instalace nástrojů používaných pro sestavení a integraci Clondike je možná (například pomocí kompilace požadované verze nástroje gcc ze zdrojových kódů), avšak uživatelsky nepřívětivá.

Produkt Circle CI podobně jako produkt Travis CI každou iteraci procesu integrace projektu používá nový virtuální stroj neovlivněný předchozími iteracemi integračního procesu. Výhody a nevýhody, které z tohoto přístupu plynou jsou popsány v kapitole 3.2.1.

3.3.2 Program pro projekty s otevřeným zdrojovým kódem

Projektům s otevřeným zdrojovým kódem je produkt Circle CI poskytován zdarma v rozsahu čtyř kontejnerů [7]. V každém kontejneru může běžet zvlášť celý integrační proces. Ke kvalifikaci k tomuto programu stačí udržovat veřejně publikované zdrojové kódy projektu.

3.4 Bamboo

3.4.1 Prostředí pro sestavení

Služba Bamboo Cloud od firmy Atlassian nabízí dvě možná řešení prostředí pro sestavení projektů [8].

Prvním z nich je využití výpočetní kapacity firmy Amazon prostřednictvím registrace v Amazon Web Services a využití služby Amazon EC2 Spot Instances. Tato varianta vyžaduje platbu za službu firmě Amazon [1], proto ji práce dále neuvažuje.

Druhým možným řešením je využití vlastního serveru. V případě této možnosti je na klientův server nainstalován klient služby Bamboo Cloud, tzv. Remote Agent, který přes síť internet komunikuje se serverovou instancí služby, ze které je řízen proces průběžné integrace.

Výhodou takového řešení je plná kontrola nad volbou prostředí použitého pro průběžnou integraci. Volba operačního systému, programového vybavení i dalších parametrů je zejména vhodná pro projekty vyžadující nestandardní postup sestavení a integrace. Dále může být pro všechny budoucí iterace integračního procesu zajištěno uspokojení datové závislosti, čímž může být výrazně zkrácena doba integračního procesu. V případě projektu Clondike se to týká zejména souboru virtuálního disku s operačním systémem pro integraci projektu Clondike.

Nevýhodou řešení s využitím vlastního serveru je možnost ovlivnění budoucích iterací sestavení některou z minulých. Tomuto nedostatku je třeba předcházet pečlivou konfigurací integračního procesu.

3.4.2 Program pro projekty s otevřeným zdrojovým kódem

Společnost Atlassian poskytuje licenci zdarma pro produkt Bamboo projektům, které splňují následující kritéria:

- Projekt je distribuován pod jednou z licencí schválenou organizací Open Source Initiative
- Zdrojové kódy jsou k dispozici ke stažení
- Projekt má veřejně dostupnou webovou stránku
- Software firmy Atlassian je veřejně přístupný

Pro získání licence zdarma za splnění výše definovaných podmínek je třeba nejdříve produkt společnosti Atlassian začít používat v režimu zkušební doby (trial) a nakonfigurovat dle požadavků společnosti Atlassian.

Po konfiguraci produktu v režimu zkušební doby je třeba na webových stránkách společnosti Atlassian (`www.atlassian.com`) vyplnit žádost o udělení licence s bližší specifikací projektu [11]. Součástí žádosti je:

- Jméno projektu
- URL projektu
- Popis projektu
- Volba typu produktu dle hostingu (vlastní server nebo cloud společnosti Atlassian)
- Volba produktů společnosti Atlassian, kterých se žádost o licenci týká
- Seznam případných rozšíření (add-ons) o které žadatel o licenci jeví zájem
- URL instance produktu společnosti Atlassian běžící v režimu zkušební doby

- Prohlášení o budoucím záměru produkt (ne)nabízet komerčně
- Použitá Open Source licence
- Webová adresa, na které jsou dostupné zdrojové kódy projektu
- Kontaktní jméno
- Kontaktní email
- Časová zóna, ve které se žadatel o licenci nachází
- Fyzická adresa
- Další volitelné poznámky a dodatky o projektu

Po vyplnění žádosti o licenci v rámci programu pro projekty s otevřeným zdrojovým kódem je žádost postoupena pro zpracování společností Atlassian. V případě produktů hostovaných na vlastích zařízeních bývá žádost zpracována během přibližně 3 pracovních dní. V případě produktů hostovaných v cloudu společnosti Atlassian (což je případ produktu Bamboo Cloud potenciálně použitelného pro naplnění cíle této práce), si společnost vyhrazuje na zpracování žádosti přibližně dva týdny.

Kapitola 4

Návrh

4.1 Návrh cloudové služby pro průběžnou integraci

Na základě analýzy navrhuji řešení založit na produktu Bamboo společnosti Atlassian.

4.1.1 Řídící část integračního procesu

Navrhuji použít produkt Bamboo ve verzi Cloud. Tím bude zajištěno uložení veškerých skriptů a konfigurace integračního procesu v cloudu poskytovatele, v tomto případě firmy Atlassian. Tím bude naplněn jeden z hlavních cílů této práce. Rozhraní k instanci služby zajišťující průběžnou integraci projektu Clondike bude poté dostupné pomocí webového rozhraní dostupného na adrese `http://jmenoprojektu.atlassian.net`.

4.1.2 Integrační server

Použití produktu Bamboo ve verzi Cloud umožňuje proces průběžné integrace realizovat na vlastním serveru nebo v cloudu společnosti Amazon. Výhody a nevýhody těchto možností jsou popsány v analýze v části 3.4.1.

Na základě srovnání těchto možností navrhuji provozovat integrační proces na vlastním virtuálním serveru. Navrhuji použít virtuální server poskytovaný Fakultou informačních technologií ČVUT, na kterém je v současné době provozován nástroj BuildBot. Kromě možnosti okamžitého využití tím bude zároveň zajištěna přítomnost všech nástrojů nutných ke správnému průběhu integračního procesu.

4.2 Návrh úprav sestavovacích skriptů

4.2.1 Změna interpretu z Bash na Sh

Navrhuji současné automatické sestavovací skripty, které jsou realizované v jazyce Bash, upravit tak, aby byli kompatibilní s jazykem Sh. Interpret Shell (`/bin/sh`) je výchozím intepretem, kterým jsou spouštěny všechny integrační skripty spuštěné nástrojem Bamboo. Na začátku skriptu je možné spustit další libovolný podřadný shell (subshell), včetně shellu Bash; do tohoto shellu už ale nejsou automaticky přenášeny definice a deklarace funkcí proměnných a další nastavení shellu. Úpravou skriptů pro běh v prostřední Sh bude zajištěna dostupnost všech proměnných. Jako vedlejší efektu bude zvýšena kompatibilita sestavovacích skriptů díky větší rozšířenosti interpretu Sh a zpětné kompatibilitě interpretu Bash se Sh.

4.2.2 Změna přístupu k užívaní příkazu sudo

V současném řešení průběžné integrace projektu Clondike je většina skriptů (všechny skripty kromě skriptu precompile.sh) spouštěna pomocí příkazu `sudo` a celé jsou vykonávány s právy superuživatele `root`. V rámci úpravy sestavovacích skriptů navrhuji zajistit jejich spouštění pod uživatelem se standardními uživatelskými oprávněními a využití příkazu `sudo` jen ke spouštění kritických částí integračního procesu, které zvýšení uživatelské oprávnění skutečně vyžadují.

4.2.3 Jednotná deklarace a definice funkcí

Navrhuji deklarace a definice funkcí pro výpis hlášení během integračního procesu realizovat v jediném souboru. V současném řešení průběžné integrace projektu Clondike jsou tyto funkce deklarovány a definovány zvlášť na začátku každého automatického sestavovacího skriptu. V případě potřeby změny nějaké funkce je třeba tuto změnu vykonat ve všech skriptech zvlášť, což způsobuje zvýšené riziko nekonzistence funkcí a zanesení chyby. Navrhuji funkce deklarovat a definovat jednotně a tyto deklarace a definice načítat ostatními skripty na začátku jejich zpracování.

4.2.4 Jednotná správa proměnných

Navrhuji ke správě proměnných využít nástroj Bamboo, který umožňuje jejich definici a správu mimo sestavovací skripty [3]. Současné automatické sestavovací skripty definují potřebné proměnné na začátku skriptu, po deklaraci a definici funkcí pro výpis hlášení. Při potřebě změny je nutné tuto provést ve všech skriptech, které jsou změnou ovlivněny. Tímto přístupem vzniká prostor pro nekonzistenci a zanesení chyby. Zároveň neexistuje jednoduchý způsob zobrazení jmen a hodnot všech proměnných používaných během integračního

procesu. Použitím nástroje Bamboo bude zajištěna konzistence dat, jednoduchá možnost změny hodnot proměnných a snadná dostupnost přehledu o jménech a hodnotách používaných proměnných. Takto spravované proměnné jsou nástrojem Bamboo exportované a v sestavovacích skriptech dostupné pomocí direktivy $bamboo_jmeno_promenne [3].

4.2.5 Rozdělení sestavovacích skriptů na dílčí úlohy

Navrhuji sestavovací a integrační proces rozdělit a více dílčích úloh a každou z nich implementovat v samostatném skriptu. V současném řešení průběžné integrace projektu Clondike jsou v některých skriptech realizovány nesouvisející úlohy. Z názvů některých těchto skriptů není jasný účel skriptu a vykonávaná úloha. Rozdělením na více kratších úloh s jasným účelem dojde ke zpřehlednění integračního procesu.

Kapitola 5

Realizace

5.1 Žádost o licenci

Pro zahájení využívání produktu Bamboo Cloud bylo nejprve třeba požádat o licenci společnost Atlassian, která produkt Bamboo vyvíjí. Společnost Atlassian poskytuje projektům s otevřeným zdrojovým kódem licenci zdarma. Podmínky získání licence a postup žádosti je popsán v kapitole 3.4.2.

Před vyplněním žádosti byla zřízena služba v módu dočasné licence (trial) na adrese `http://clondike.atlassian.net`. Společnost Atlassian umožňuje vyzkoušení produktu Bamboo Cloud v módu dočasné licence po dobu 7 dnů. Schvalování licence zdarma pro projekty s otevřeným zdrojovým kódem obvykle trvá asi dva týdny [11].

Po zřízení služby v módu dočasné licence jsem vyplnil všechny potřebné údaje v žádosti o licenci zdarma a žádost odeslal. Jako kontakt na administrátora služby jsem uvedl svou emailovou adresu a jako kontakt pro komunikace ohledně licence jsem uvedl vedoucího práce (který se projektu Clondike věnuje dlouhodobě).

Po uplynutí 7 dnů skončila platnost dočasné licence. Licence zdarma pro projekty s otevřeným zdrojovým kódem stále nebyla udělena. Pro zachování současné konfigurace služby a možnosti dál s produktem pracovat jsem službu předplatil na jeden měsíc. Ani po obvyklé lhůtě dvou týdnů ale nebyla trvalá licence zdarma udělena. V reakci na to jsem zahájil komunikaci s technickou podporou společnosti Atlassian.

Pracovníkům technické podpory společnosti Atlassian se nepodařilo žádost o licenci dohledat a byl jsem požádán o vytvoření nové. To jsem učinil a pro urychlení procesu jsem jako administrátora i jako kontakt pro komunikaci ohledně licence uvedl svou akademickou adresu. Byl jsem tedy okamžitě vybaven jak přístupem ke službě Bamboo Cloud, tak referenčním číslem žádosti o licenci, které bylo zasláno na adresu kontaktu pro komunikaci ohledně licence. Pracovníkům technické podpory jsem zaslal referenční číslo žádosti o licenci.

Reakce pracovníků technické podpory byla promptní a žádost o licenci pro projekt Clondike byla projednána ve zrychleném řízení. Trvalá licence zdarma na produkt Bamboo Cloud za účelem průběžné integrace projektu Clondike byla vystavena za 3 pracovní dny.

5.2 Instalace a konfigurace Bamboo Remote Agent

5.2.1 Vytvoření uživatele Bamboo

Na virtuálním serveru FIT ČVUT byl za účelem provozu komponenty Bamboo Remote Agent vytvořen separátní uživatelský účet `bamboo`. Uživatelskému účtu byl vytvořen domovský adresář `/home/bamboo`.

5.2.2 Stažení Bamboo Remote Agent

Instalační soubor klienta služby Bamboo Cloud (Bamboo Remote Agent) je k dispozici ke stažení z webového rozhraní vlastní instance služby Bamboo Cloud. Po přihlášení do služby (v případě této práce na adrese `https://clondike.atlassian.net`) je třeba v menu nastavení vybrat volbu agents a přesunout se do administrace agentů. Z administrace agentů se volbou (kliknutím na tlačítko) install new agent přesuneme na rozhraní, ze kterého lze stáhnout instalační soubor klienta služby Bamboo Cloud. Vzhledem k absenci webového prohlížeče (a grafického rozhraní) na serveru FIT ČVUT byla tato operace provedena z jiného počítače a instalační soubor byl poté přesunut na server FIT ČVUT pomocí protokolu scp.

5.2.3 Instalace JDK

Instalační soubor klienta služby Bamboo Cloud je distribuován ve formátu `.jar` [16]. K jeho instalaci a spuštění je třeba celé prostředí JDK (nestačí samotné JRE) [17]. Pro současnou verzi klienta (verze 5.10) služby Bamboo Cloud je vyžadováno Oracle JDK verze 1.8 nebo OpenJDK verze 1.8 [17].

Na serveru FIT ČVUT je nainstalován operační systém Debian ve verzi 7 (Wheezy) a není na něm nainstalováno prostředí JDK. V repozitářích distribuce Debian pro verzi 7 není k dispozici Oracle JDK; OpenJDK je k dispozici nejvýše do verze 1.7. Proto byl do operačního systému přidán PPA repozitář obsahující Oracle JDK ve verzi 1.8. Přidání PPA repozitáře bylo provedeno příkazem

```
# add-apt-repository ppa:webupd8team/java
```

Po přidání PPA repozitáře byla provedena instalace Oracle JDK verze 1.8 příkazem

```
# apt-get install oracle-java8-installer
```

Instalace Oracle JDK 1.8 poté proběhla korektně.

5.2.4 Instalace a spuštění Bamboo Remote Agent

Po instalaci JDK je možné spustit klienta služby Bamboo Cloud. Klient byl spuštěn pomocí příkazu vygenerovaného webovým rozhraním služby Bamboo Cloud:

```
# java -jar \
# atlassian-bamboo-agent-installer-5.11.0-D20160419T121932.jar \
# https://clondike.atlassian.net/builds/agentServer/ \
# -t 16a15b1533173a4de94b63864fadd1b697483df6
```

Řetězec za parametrem `-t` je bezpečnostní token, byl proto pro účely této práce změněn na jinou náhodnou hodnotu.

Tím byla zahájena instalace klienta, která probíhala bez chybových hlášení a zastavila se v okamžiku, kdy se pokusila komunikovat s instancí služby Bamboo Cloud běžící na adrese `https://clondike.atlassian.net`. Instalátor klienta vypsal varovné hlášení o tom, že klient nebyl ve službě autentikován a instalace nebude nadále pokračovat. Pokus o spojení s Bamboo Cloud poté probíhal periodicky každých 60 sekund, dokud nebyla provedena autentikace klienta na straně Bamboo Cloud.

Po autentikaci na straně Bamboo Cloud se úspěšně dokončí instalace klienta. Po úspěšné instalaci klienta je tento spuštěn a připraven k použití. Klient sluby Bamboo Cloud byl nainstalován do adresáře `/home/bamboo/bamboo-agent-home`. Změnu domovského adresáře klienta Bamboo Cloud lze vynutit volbou `-D` při instalaci klienta [16].

Po instalaci se k zastavení činnosti klienta či jeho opětovnému spuštění používá řídící skript `bamboo-agent.sh` umístěný v podadresáři `bin`, který je umístěn v domovském adresáři klienta služby Bamboo Cloud.

5.2.5 Autentikace s Bamboo Cloud

Autentikace klienta služby Bamboo Cloud se provádí z webového rozhraní služby. V rozhraní je třeba se přesunout pomocí záložky nastavení do správy agentů. Zde je seznam všech agentů včetně jejich IP adresy, unikátního ID a stavu, ve kterém se agenti nachází. Při instalaci prvního agenta je zde pouze jeden klient ve stavu čekajícím na autentikaci. Autentikace je provedena volbou Approve access. V případě výskytu neznámého klienta případně vyřazení starého klienta se revokace přístupu provede volbou Revoke access.

5.2.6 Dodatečné úpravy v OS

Integrační proces projektu Clondike vyžaduje provádění operací, ke kterým je potřeba zvýšené uživatelské uprávnění (například připojování a odpojování virtuálního disku a operace na něm). Uživateli `bamboo` bylo proto umožněno spouštět některé příkazy pomocí příkazu `sudo`, který zvýšení uživatelského uprávnění zajistí. Toho bylo docíleno přidáním řádků

```
bamboo ALL = (root) NOPASSWD: /opt/vmware-server-distrib/bin/vmware-mount
bamboo ALL = (root) NOPASSWD: /usr/bin/make
bamboo ALL = (root) NOPASSWD: /usr/bin/[
bamboo ALL = (root) NOPASSWD: /usr/sbin/chroot
bamboo ALL = (root) NOPASSWD: /bin/rm
bamboo ALL = (root) NOPASSWD: /bin/cp
bamboo ALL = (root) NOPASSWD: /bin/chmod
bamboo ALL = (root) NOPASSWD: /bin/mkdir
bamboo ALL = (root) NOPASSWD: /bin/dd
```

do souboru `/etc/sudoers`. Volba `NOPASSWD` zajišťuje, že uživatel při použití příkazu `sudo` není po uživateli požadováno zadání hesla [32]. To je nezbytné pro bezobslužnou automatizaci integračního procesu.

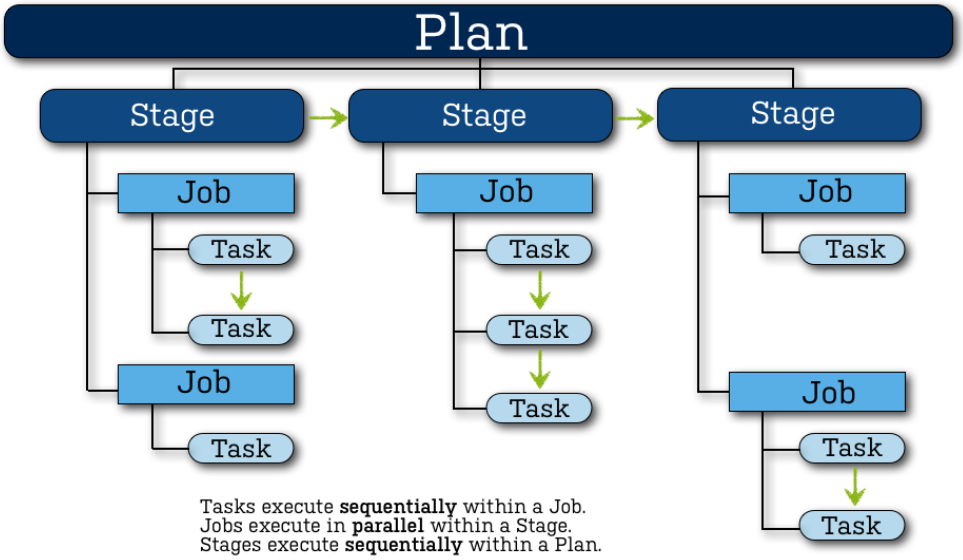
5.3 Konfigurace Bamboo Cloud

Konfigurace služby Bamboo Cloud probíhá přes webové rozhraní. Pro každý projekt může být vytvořen jeden nebo více plánů (definic integračního procesu).

Plán se sestává ze stádií a plánu prací (job plan), přičemž ke každému plánu může existovat více větví (například produkční a vývojová), které mohou mít některá vlastní nastavení (například jiné nastavení hodnot proměnných).

Stádiem se rozumí například část sestavovací či testovací. Plán prací se sestává z jednotlivých úloh, kterými může být spuštění příkazu či skriptu. Úlohy se provádí sekvenčně v definovaném pořadí. Strukturu a způsob provádění integračních operací ilustruje obrázek 5.1.

Po skončení každé úlohy nástroj Bamboo zkontroluje návratovou hodnotu dané úlohy; v případě návratové hodnoty značící úspěch pokračuje další hodnotou; v případě návratové hodnoty značící neúspěch nástroj Bamboo přeskočí všechny další standardní úlohy a provede pouze skupinu úloh, která je nastavena jako „final“.



Obrázek 5.1: Přehled operací a způsobu jejich vykonávání službou Bamboo [19]

5.4 Správa proměnných

Všechny proměnné používané skripty během procesu průběžné integrace byly deklarovány v nastavení integračního plánu pod záložkou „Variables“ (proměnné). Kompletní seznam proměnných a jejich hodnoty deklarovaných tímto způsobem popisuje tabulka 5.1

Jméno proměnné	Hodnota proměnné
clondikeVmdk	/mnt/data/bamboo/production/clondike.vmdk
functions_declaration_file	logging_functions_declaration.sh
github_folder	clondike
linux	linux-3.6.11
linuxver	3.6.11
vmdk_mountpoint	/mnt/data/bamboo/production/mnt
vmware	/opt/vmware-server-distrib/bin/vmware-mount

Tabulka 5.1: Uživatelsky definované proměnné a jejich hodnoty v nástroji Bamboo

Tyto proměnné jsou dostupné ve všech skriptech běžících v prostředí `shellu` pomocí syntaxe `$bamboo_promenna`.

5.5 Způsob spouštění skriptů

5.5.1 Oprávnění skriptů

Všechny skripty, které realizují integrační proces, jsou nástrojem Bamboo spouštěny pod uživatelem `bamboo` se standardním oprávněním. Pokud některá z akcí prováděných skriptem vyžaduje zvýšené uživatelské oprávnění, je tato konkrétní akce vykonána pomocí příkazu `sudo`, který zajistí vykonání s oprávněním uživatele `root`.

5.5.2 Interpret skriptů

Všechny skripty byly upraveny tak, aby byly kompatibilní s jazykem shell a tedy interpretovatelné pomocí interpretru `/bin/sh`. Toto je výchozí interpret, pod kterým jsou nástrojem Bamboo spouštěny všechny uživatelské skripty.

Úpravy skriptů spočívaly v nahrazení některých příkazů, které nebyly kompatibilní s interpretrem `/bin/sh` za příkazy vykonávající stejnou funkčnost, které s interpretrem `/bin/sh` kompatibilní jsou. Například testování podmínek bylo realizováno pomocí následující syntaxe

```
# if [[ podminka ]] ; then
```

Příkaz `[[` je interně zpracovávaný příkaz jazyka Bash, který shell `/bin/sh` nepodporuje. Pro kompatibilitu s interpretrem `/bin/sh` byla použita syntaxe

```
# if [ podminka ] ; then
```

V případě použití této syntaxe je volán externí příkaz `[` (test), který provede vyhodnocení podmínky. Funkčnost příkazu je přitom zachována.

5.5.3 Uložení skriptů

Všechny skripty jsou uloženy na serverové straně v nástroji Bamboo (tzv. inline). Tím je zajištěna bezpečnost uložení a zamezeno ztrátě těchto skriptů. Zároveň je výhodou i snadná editace z webového prohlížeče. V průběhu integračního procesu jsou skripty přeneseny na server, kde proces probíhá, uloženy do dočasných souborů a spuštěny. Po dokončení integračního procesou jsou odstraněny.

5.5.4 Logování

Nástroj Bamboo zajišťuje uložení veškerých informací, které jsou vypsány na standardní výstup nebo standardní chybový výstup, do logovacího souboru. Tento logovací soubor je v průběhu integračního procesu postupně přenášen na serverovou stranu nástroje Bamboo a vypisován do webového rozhraní. Po skončení integračního procesu je logovací soubor k dispozici k prohlížení i ke stažení.

5.6 Kroky průběžné integrace

Integrační proces a automatické sestavovací skripty byly založeny na současném řešení realizovaném pomocí nástroje BuildBot. Integrační proces byl restrukturalizován a ve skriptech byly provedeny úpravy navržené v kapitole 4.2 této práce.

5.6.1 Stažení zdrojových souborů z repozitáře GitHub

V prvním kroku nástroj Bamboo stáhne všechny zdrojové soubory projektu Clondike z repozitáře GitHub. Stažené soubory jsou uloženy do adresáře, který je nastaven pomocí proměnné `${bamboo.github_folder}` (v současné konfiguraci „`clondike`“).

Nástroj je nastaven taky, aby soubory stažené v případné předchozí iteraci integračního procesu zachoval a z repozitáře stahoval pouze změněné soubory. Tím šetří systémové zdroje serveru, na kterém probíhá integrační proces a zkracuje celkové trvání tohoto procesu.

5.6.2 Vytvoření souboru funkcí

V tomto kroku nástroj Bamboo spustí skript, který vytvoří soubor s deklaracemi a definicemi funkcí pro výpis hlášení během integračního procesu. Tyto jsou uloženy do souboru nastaveného pomocí proměnné `functions_declaration_file` (v současné konfiguraci „`logging_functions_declaration.sh`“).

Tento soubor funkcí je v dalších krocích integračního procesu načítán na začátku skriptů realizujících integrační proces pomocí direktivy `.` (tečka). Tím je zajištěna deklarace a definice funkcí v aktuální instanci shellu s platností až do jejího skončení.

5.6.3 Příprava zdrojových souborů jádra OS

V tomto kroku nástroj Bamboo spustí skript, který v pracovním adresáři integračního procesu zajistí přítomnost zdrojových souborů obecného (vanilla) jádra Linux bez úprav.

Skript nejdříve otestuje existenci adresáře se zdrojovými kódy jádra Linux vzniklého při případné předchozí iteraci integračního procesu a v případě jeho existence jej odstraní. Dále skript otestuje existenci archivu (soubor s komprimovanými soubory) se zdrojovými soubory jádra Linux a v případě jeho neexistence jej stáhne z webových stránek `https://www.kernel.org` pomocí nástroje `wget`. Nakonec skript archiv se zdrojovými soubory jádra Linux extrahuje do aktuálního pracovního adresáře.

5.6.4 Záplata jádra OS a vložení zdrojových souborů projektu Clondike

V tomto kroku nástroj Bamboo spustí skript, který nastaví pracovní adresář na adresář se zdrojovými soubory jádra Linux a aplikuje na ně záplatu implementující funkčnost Clondike.

Po aplikaci záplaty skript do adresáře se zdrojovými soubory jádra Linux nakopíruje další soubory nutné pro běh projektu Clondike v prostoru jádra. Nakonec je do adresáře se zdrjovými kódy jádra Linux nakopírován konfigurační soubor `.config` s nastavením procesu sestavení.

5.6.5 Příprava adresáře se zdrojovými kódy jádra OS pro kompilaci

Před zahájením sestavování jádra nástroj Bamboo v adresáři se zdrojovými kódy jádra spustí příkaz

```
# make clean
```

který odstraní případné (již sestavené) objektové soubory a spustitelné soubory, čímž zajistí jejich nové sestavení z aktuálních zdrojových kódů v příštím kroku.

5.6.6 Sestavení jádra OS s podporou projektu Clondike

Nové jádra Linux s funkčností projektu Clondike je sestaveno spuštěním příkazu

```
# make -j4
```

v adresáři se zdrojovými kódy jádra. Argument `-j4` je volitelný a nastavuje počet paralelních úloh pracujících na sestavení jádra (v tomto případě 4). Nastavení počtu paralelních úloh je pro urychlení procesu sestavení jádra volit v souladu s počtem jader stoje, na kterém je úloha vykonávána.

5.6.7 Sestavení modulů pro jádro OS

Moduly pro nové jádro Linux jsou sestaveny spuštěním příkazu

```
# make modules
```

v adresáři se zdrojovými kódy jádra. Sestavením modulů jádra je celé jádro připraveno pro integraci do operačního systému.

5.6.8 Připojení virtuálního disku s OS pro integraci projektu Clondike

K připojení virtuálního disku je použit nástroj `vmware`, konkrétně spustitelný soubor `vmware_mount`. Skript nejdřív otestuje existenci virtuálního disku pro integraci projektu Clondike poté disk připojí. Obě tyto operace provede se

zvýšeným uživatelským oprávněním pomocí příkazu `sudo`, což je vyžadováno nástrojem `vmware` [14].

5.6.9 Instalace modulů jádra OS

Instalace nových modulů pro jádro OS probíhá ve dvou krocích. V prvním kroku skript smaže všechny původní moduly z adresáře `/lib/modules/$bamboo_linuxver` na připojeném virtuálním disku. Ve druhém kroku skript provede instalaci nových modulů pomocí příkazu

```
# make INSTALL_MOD_PATH=${bamboo_vmdk_mountpoint} modules_install
```

provedeného v adresáři se zdrojovými kódy jádra Linux (v této fázi už je v tomto adresáři také sestavené jádra a moduly).

5.6.10 Instalace jádra OS

Instalace nového jádra OS je provedena nahrazením původních souborů jádra v adresáři `/boot`. Skript postupně nahradí soubory `vmlinuz-${bamboo_linuxver}` (spustitelný soubor sestaveného jádra), `System.map-${bamboo_linuxver}` (tabulka symbolů pro jádra) a `.config` (konfigurační soubor jádra) novými soubory, vytvořenými v předchozích krocích integračního procesu.

5.6.11 Kopírování souborů projektu Clondike do OS na virtuálním disku

V tomto kroku je spuštěn skript, který provede kopírování souborů nezbytných pro funkčnost projektu Clondike na připojený virtuální disk.

V první části skriptu je postaráno o řídící skripty projektu Clondike a npfs serveru, které musí být umístěny do složky `/etc/init.d` na připojený virtuální disk. Skript nejdříve nastaví tyto řídící skripty jako spustitelné pomocí příkazu `chmod` a poté je umístí do adresáře `/etc/init.d`.

Ve druhé část skriptu jsou na virtuální disk do adresáře `/root/clondike` zkopírovány všechny soubory projektu Clondike (celý obsah dané větve (branch) repozitáře GitHub).

5.6.12 Sestavení Director API a Ruby Director API

Před sestavením komponent Director API a Ruby Director API skript otestuje existenci adresáře pro konfiguraci komponenty Ruby Director API (`/root/clondike/userspace/simple-ruby-director/conf` a v případě jeho neexistence jej vytvoří.

Sestavení komponenty Director API je provedeno sekvencí příkazů

```
# make clean
# make
```

Tyto příkazy musí být spuštěny v adresáři `/root/clondike/userspace/director-api` na připojeném virtuálním disku. Aktuální shell, ve kterém jsou příkazy spouštěny beží s právy standardního uživatele (`bamboo`). Pro práci na virtuálním disku jsou potřeba práva superuživatele `root`. Z tohoto důvodu se pro spuštění výše uvedené sekvence příkazů použije příkaz `sudo`, který zajistí potřebou eskalaci práv. Aktuálnímu shellu také nemůže být nastaven pracovní adresář na `/root/clondike/userspace/director-api`, proto je pro předání cesty nástroji `make` použit přepínač `-C cesta`, který zajistí provedení v požadovaném adresáři.

Sestavení komponenty Ruby Director API je provedeno sekvencí příkazů

```
# make clean
# make
# make install
```

Tyto příkazy musí být spuštěny v adresáři `/root/clondike/userspace/ruby-director-api` na připojeném virtuálním disku. S tím souvisí stejné problémy, jako při sestavování komponenty Director API, proto je i řešení stejné. Ke spuštění je použit příkaz `sudo`, který zajistí potřebná práva a přepínač `-C cesta`, který zajistí spuštění v příslušném adresáři.

5.6.13 Sestavení a konfigurace NPFS serveru

Před sestavením NPFS serveru je skriptem otestována existence adresáře pro soubory NPFS (`/root/npfs`) na virtuálním disku a v případě neexistence je adresář vytvořen.

Sestavení NPFS severu je provedeno sekvencí příkazů

```
# make clean
# make
```

Tyto příkazy musí být spuštěny v adresáři `/root/clondike/root/npfs_install` na připojeném virtuálním disku. Pro přístup do virtuálního disku je třeba zvýšené uživatelské oprávnění, které je zajištěno spuštěním výše uvedené sekvence příkazů pomocí příkazu `sudo`. Spuštění v požadovaném adresáři je zajištěno použitím parametru `-C cesta` nástroje `make`.

Po sestavení NPFS serveru skript zkopíruje do adresáře `/root` na virtuálním disku konfigurační soubor `.migration.conf` z . Nakonec skript zkopíruje do adresáře `/root/npfs` spustintelný binární soubor `npfs` (sestavený NPFS server) a spouštěcí skript `npfs-start.sh`.

5.6.14 Vytvoření initramdisku

Initramdisk je archiv se základním souborovým systémem, který se do paměti načítá při startu operačního systému. Nástroj Bamboo jej vygeneruje příkazem

```
# sudo chroot ${bamboo_vmdk_mountpoint} mkinitramfs \
# -o /boot/initrd.img-${bamboo_linuxver} ${bamboo_linuxver}
```

Nástroj `sudo` je použit pro eskalaci práv, která skriptu umožní přistupovat na připojený virtuální disk. Nástrojem `chroot` je dočasně nastaven kořenový adresář aktuálního shellu na kořenový adresář připojeného virtuálního disku. Tím je docíleno vygenerování initramdisku pro cílový operační systém (jinak by se nástroj `mkinitramfs` pokušel o vytvoření initramdisku pro operační systém, na kterém je prováděn integrační proces).

5.6.15 Vytvoření a odstranění nulovacího souboru

Cílem integračního procesu je vytvoření archivu s virtuálním diskem, který obsahuje operační systém s integrovanou funkčností projektu Clondike. Použitý archivační nástroj `gzip` však nerozlišuje vnitřní strukturu (souborový systém) souboru virtuálního disku a tak se z jeho pohledu skutečná data a nevyužitý prostor jeví jako souvislá data. Před odpojením virtuálního disku je proto spuštěn skript, který veškeré prázdné místo na virtuálním disku vyplní logickou nulou (čímž umožní maximální možný kompresní poměr). Toto provede příkazem

```
# sudo chroot ${bamboo_vmdk_mountpoint} \
# dd if=/dev/zero of=/nullfile bs=4096k
```

Nástroj `sudo` je použit pro eskalaci práv, která skriptu umožní přistupovat na připojený virtuální disk. Nástrojem `chroot` je dočasně nastaven kořenový adresář aktuálního shellu na kořenový adresář připojeného virtuálního disku. Nástroj `dd` čte ze vstupního souboru nastaveného parametrem `if` (input file) a zapisuje do výstupního souboru nastaveného parametrem `of` (output file). Vstupním souborem je v tomto případě speciální soubor `/dev/zero`, ze kterého lze postupným čtením přečíst nekonečné množství logických nul. Výstupním souborem je soubor `nullfile` v kořenovém adresáři virtuálního disku (ve skutečnosti na jeho jménu nezáleží, pouze na umístění a unikátnosti jména). Parametrem `bs` (block size) je nastaven počet bitů, které jsou přečteny/zapsány při každé iteraci zápisové a čtecí operace. Hodnota 4096k představuje typickou velikost sektoru na dnes používaných pevných discích. Nástroj `dd` zapisuje do souboru `nullfile` logickou nulu tak dlouho, dokud nedojde místo na virtuálním disku, poté se ukončí s chybou informující o došlém místě na zařízení, kde se nachází výstupní soubor. Následně je daný soubor odstraněn (fakticky

je odstraněn jen záznam o jeho existenci z tabulky souborového systému), čímž vznikne na virtuálním disku opět volné místo, které je reálně vyplněno logickými nulami.

5.6.16 Odpojení virtuálního disku s OS pro integraci projektu Clondike

V tomto kroku je virtuální disk ze systému odpojen nástrojem `vmware` pomocí příkazu

```
# sudo "$bamboo_vmware" -d "$bamboo_vmdk_mountpoint"
```

Tento krok je nastaven jako „final task“, což způsobí jeho provedení i v případě, že některý z předchozích kroků selže. Tím je zajištěn výchozí stav integračního procesu pro jeho další iteraci.

5.6.17 Komprese virtuálního disku s OS pro integraci projektu Clondike

Po odpojení virtuálního disku je soubor komprimován nástrojem `gzip` s nejvyšší možnou kompresí (nejmenší možnou velikostí výsledného souboru). Komprimovaný archiv je uložen do pracovního adresáře vzdáleného agenta služby Bamboo. Tento krok je rovněž nastaven jako „final task“ a provede se i v případě selhání některého z předchozích kroků.

5.7 Procesy po provedení integrace

5.7.1 Nahrání artefaktů

Po dokončení integračního procesu nástroj Bamboo zkontroluje existenci definovaných artefaktů. V nástroji Bamboo byl definován artefakt s názvem „`clondike.vmdk.gz`“. V případě zjištění existence souboru artefaktu je tento nahrán do cloudu a zpřístupněn ke stažení z webového rozhraní pod záložkou „Artifacts“ konkrétní iterace integračního procesu.

5.7.2 Záznamy o integračním procesu

Všechny výstupy vypsané na standardní výstup a standardní chybový výstup během integračního procesu byly nástrojem Bamboo uloženy a jsou dostupné k prohlížení i ke stažení z webového rozhraní. Součástí záznamů jsou i informace o nastaveném prostředí, ve kterém byl spuštěn integrační proces, hodnotách proměnných a změnách zdrojových kódů projektu Clondike, ke kterým došlo od poslední iterace integračního procesu.

5.7.3 Hlášení o ukončení integračního procesu

Po skončení integračního procesu může nástroj Bamboo volitelně (dle konfigurace) o tomto informovat uživatele nebo celé skupiny služby Bamboo. Například pokud integrační proces selže, může služba Bamboo o tomto zaslat informaci na email uživateli, který provedl poslední změnu (commit) do zdrojových kódů projektu Clondike.

Kapitola 6

Testování implementovaného řešení měřením

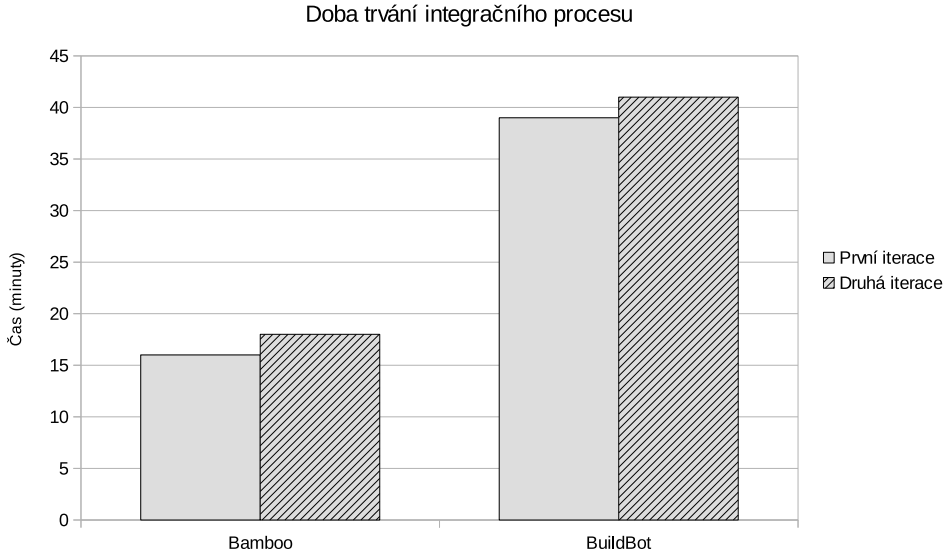
6.1 Měření doby integračního procesu

S vedoucím práce bylo dohodnuto měření času potřebného pro sestavení obrazu virtuálního disku s integrovanou funkčností projektu Clondike. Cílem měření bylo porovnat čas nutný pro sestavení pomocí nástroje BuildBot a čas nutný pro sestavení pomocí nástroje Bamboo. Porovnáván byl čas sestavení produkční verze projektu Clondike ze dne 7. Prosince 2015.

Integrační proces byl v obou případech proveden na virtuálním serveru poskytnutém Fakultou informačních technologií ČVUT. Měření bylo provedeno ve dvou iteracích. Časy měření byly pro účely porovnání rychlosti integračního procesu zaokrouhleny na minuty. Výsledky měření časů sestavení znázorňuje graf 6.1.

Z grafu je patrné, že nástroj Bamboo byl v obou iteracích zhruba o polovinu rychlejší, než nástroj BuildBot. Integrační proces provedený nástrojem Bamboo trval při prvním měření 16 minut a při druhém měření 18 minut. Integrační proces provedený nástrojem BuildBot trval při prvním měření 39 minut a při druhém měření 41 minut.

Možnou příčinu rozdílů v časech jednotlivých iterací provedených stejným nástrojem je možné spatřovat ve dvou měnících se atributech. Prvním je rozdílné aktuální vytížení virtuálního serveru, na němž probíhal integrační proces. Druhým je vytížení fyzického serveru, na kterém je provozován hostitelský operační systém a virtualizační řešení. O dalších virtuálních serverech ani dalších procesech běžících na fyzickém serveru, na kterém je spuštěn virtuální server použitý pro měření, neměl autor práce informace a proto je nemohl adekvátně vyhodnotit.



Obrázek 6.1: Porovnání časů integračního procesu nástroji Bamboo a BuildBot

6.2 Testování funkčnosti

V rámci práce byla otestována funkčnost operačního systému a projektu Clondike na virtuálním disku sestaveném pomocí nástroje Bamboo. Jako testovací prostředí byl použit notebook Dell Latitude E6510 s procesorem Intel Core i7 720QM. Jako hostitelský operační systém byla použita Linuxová distribuce Fedora 23. Jako virtualizační nástroj byl použit VMware Workstation 12.

V nástroji VMware Workstation byl vytvořen virtuální stroj s jedním procesorovým jádrem a 512MB operační paměti. K tomuto virtuálnímu stroji byl připojen disk sestavený nástrojem Bamboo. Po spuštění virtuálního stroje byl operační systém zaveden bez chyb. Bylo provedeno pokusné přihlášení uživatele `root` s heslem `tvrdik`, které proběhlo bez problémů a uživateli byl spuštěn jeho nastavený uživatelský shell (bash). Byla ověřena základní funkčnost operačního systému a virtuální stroj byl vypnut.

Dále byl vytvořen klon tohoto virtuálního stroje za účelem ověření funkčnosti projektu Clondike. Klonu původního virtuálního stroje byla vygenerována nová fyzická adresa síťového rozhraní (MAC adresa) a byl spuštěn. V operačním systému byla staticky nastavena nová IP adresa a vygenerována nová konfigurace a identita služby Clondike. Po této změně konfigurace byl spuštěn původní virtuální stroj. Kontrolou Director logu Clondike (`/tmp/director.log` bylo zjištěno, že na obou strojích byla úspěšně spuštěna služba Clondike, která naslouchá na lokální IP adrese a dále byla navázána komunikace s druhým strojem.

```
8.0.100:54321..
I, [2016-05-11 21:39:55#2397]  INFO -- : Starting director on node with id
 -----BEGIN PUBLIC KEY-----
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCkMxjRSg54eTrqsupITflOrmw3
ruTy2VmGd9l+NCS/tJgPRgDUl99rZLkDfyyj98JOQnfCYilym1RnXVsvDXHbJARE
beW/q5CuuSdz1BDw+YW0g9taYYXkeQ7UQ/gFPQVrcoVcUIkV6WQDyxunER39vBRQ
800ADTrAE2/OD8oojQIDAQAB
-----END PUBLIC KEY-----

[Wed May 11 21:39:55 2016] CliServer 127.0.0.1:4223 start
/root/clondike/userspace/simple-ruby-director/Director.rb:132: warning: instance
 variable @cql3Driver not initialized
Initializing director
Initialization netlink family OK
Initialize netlink family success
Registering pid
Register PID success
Generic netlink channel for director number is: 22
Comm channel initialized
I, [2016-05-11 21:39:59#2397]  INFO -- : Server challenge received
I, [2016-05-11 21:39:59#2397]  INFO -- : Connection attempt to 192.168.0.101:543
21 with proof 16237916982697530526 succeeded.
I, [2016-05-11 21:39:59#2397]  INFO -- : MembershipManager.rb: connectToNode: no
 slotIndex find for node:
192.168.0.100 1/1 ~ # cat /tmp/director.log _
```

Obrázek 6.2: Ukázka výpisu Director logu po propojení dvou strojů

Po propojení obou stojí a potvrzení úspěšného navázání komunikace komponenty Clondike byl proveden test migrace procesu. Nejdříve byl proveden pokusný překlad programu typu „Hello world!“ překladačem `gcc`, který proběhl úspěšně. Kontrolou Director logu ale bylo zjištěno, že plánovač (Load Balancer) projektu Clondike se úlohu rozhodl nemigrovat na vzdálený uzel (kvůli malému vytížení lokálního procesoru). Následně byla nastavena exportována proměnná `EMIG` s hodnotou `1`, která migraci vynutí. Následné spuštění pokusné úlohy bylo s tímto nastavením úspěšně migrováno na vzdálený proces a překlad proběhl bez chyb.

Jako další testovací úloha bylo zvoleno sestavení jádra Linux s cílem zopakovat měření z článku [30]. Sestavení jádra však opakovaně havarovalo a končilo neúspěchem. Během překladu se vyskytovaly různé, zdánlivé náhodné chyby, jejichž příčinu se autorovi nepodařilo jednoznačně identifikovat. Konzultací s ostatními členy týmu podílejícími se na vývoji projektu Clondike bylo zjištěno, že současná verze projektu Clondike není pro podobné měření použitelná a obsahuje chybu, která byla odhalena ve spolupráci s vývojáři Linuxového jádra, ale její oprava ještě nebyla zanesena do repozitáře projektu Clondike.

Závěr

Na začátku práce byla provedena analýza současného řešení pro průběžnou integraci projektu Clondike. Nejdříve byl popsán současný nástroj používaný pro zajištění průběžné integrace (BuildBot) a poté analyzovány a popsány automatické sestavovací skripty, které sestavení a integrační proces realizují.

Následně byl proveden průzkum cloudových služeb pro průběžnou integraci. Na základě hlavních požadavků pro integraci projektu Clondike (zejména provozu služby zdarma) a konzultace s vedoucím práce byly vybrány tři služby, které byly analyzovány podrobněji. U těchto tří služeb bylo popsáno prostředí pro průběžnou integraci, podmínky provozu a jejich vhodnost využití pro projekt Clondike.

Z analyzovaných cloudových služeb pro průběžnou integraci byla vybrána služba Bamboo Cloud, která se na základě analýzy ukázala být pro projekt Clondike nejvhodnější. Po vybrání dané služby byla navržena série úprav automatických sestavovacích skriptů. Tyto navržené úpravy zajistily použitelnost skriptů v kombinaci s vybranou službou pro průběžnou integraci a zpřehlednění integračního procesu.

V závěrečné části byla služba Bamboo Cloud nakonfigurována a zprovozněna. V automatických sestavovacích skriptech byly provedeny nezbytné úpravy pro funkčnost se službou Bamboo Cloud a integrační proces byl rozdělen na více dílčích částí, čímž došlo ke zpřehlednění procesu.

Vzdálený klient služby Bamboo Cloud byl nakonfigurován na virtuálním serveru FIT ČVUT, kde je realizován integrační proces řízený z cloudu službou Bamboo. Rozhraní služby Bamboo Cloud pro konfiguraci a prohlížení záznamů o předešlých iteracích integračního procesu je dostupné na adrese `https://clondike.atlassian.net`.

Bylo provedeno měření a testování nástroje Bamboo a sestaveného virtuálního disku se závěrem, že projekt Clondike byl úspěšně integrován a výsledný systém je funkční.

Tím bylo zadání práce naplněno a služba připravena na použití v praxi při dalším vývoji projektu Clondike.

Budoucí práce

Testování sestaveného virtuálního disku

Pro využití dalších výhod vývoje projektu Clondike s využitém průběžné integraci navrhuji integrační službu Bamboo doplnit o testy sestaveného virtuálního disku. V současném stavu nástroj Bamboo sestaví virtuální disk s operačním systémem s integrovanou funkčností Clondike, nahraje jej do cloudu společnosti Atlassian, ale neprovede na něm žádné testy funkčnosti. Navrhuji ve službě Bamboo vytvořit další fázi (stage), která bude obsahovat automatizované testy funkčnosti sestaveného virtuálního disku a projektu Clondike.

Nasazením automatizovaných testů bude docíleno odhalení případných chyb zanesených do projektu Clondike okamžitě po jejich nahrání do repozitáře GitHub. To umožní jednoznačně určit kdo nese odpovědnost za zanesení chyby a její opravu. Nástroj Bamboo na zanesenou chybu může zaslat odpovědném vývojáři okamžitě po odhalení chyby upozornění, což danému vývojáři umožní okamžitě na zanesení chyby reagovat a může začít pracovat na opravě.

Úprava konfigurace integračního serveru

Integrační proces řízený službou Bamboo je vykonáván na virtuálním serveru poskytnutém Fakultou informačních technologií ČVUT. Na systémovém disku serveru není dostatek volného místa, proto jsou obrazy virtuálních disků uloženy na dalším připojeném disku a práce s připojeným virtuálním diskem vyžaduje zvýšené uživatelské oprávnění.

Navrhuji klienta služby Bamboo (Bamboo remote agent) provozovat na serveru s dostatečně velkým systémovým diskem, aby veškerá potřebná data mohla být uložena v pracovním adresáři klienta služby Bamboo. Pro práci s připojeným virtuálním diskem navrhuji použít balík `Fuse` (file system in user space), který umožní práci s diskem bez zvýšených uživatelských oprávnění [14].

Systémového uživatele `bamboo` by v takovém případě bylo možné provozovat v uzavřeném prostředí pomocí příkazu `chroot`, ze kterého by neměl přístup do kořenové adresářové struktury operačního systému, na kterém běží klient služby Bamboo vykonávající integrační proces. Seznam příkazů, které má uživatel `bamboo` povolen spouštět se zvýšeným uživatelským oprávněním pomocí příkazu `sudo` by poté bylo možné zredukovat na připojování a odpojování virtuálního disku. Tím by byla zajištěna bezpečnost operačního systému i v případě kompromitace uživatelského účtu `bamboo` nebo klienta služby Bamboo.

Vývoj projektu Clondike

Při dalším vývoji projektu Clondike je třeba se nejdříve zaměřit na opravu chyb, které byly během vývoje do projektu zaneseny a způsobují havárii při migraci některých úloh. Možnou příčinou je chyba, kterou se podařilo odhalit ve spolupráci s vývojáři Linuxového jádra, ale její oprava ještě nebyla zanesena do repozitáře projektu Clondike.

Opravou chyb způsobujících nahodilé havárie migrovaných procesů bude umožněno provádět měření výkonnosti klastru Clondike. Zajištěním možnosti měření výkonnosti bude umožněn vývoj výkonnějšího plánovače úloh [30], prezentace projektu na konferencích případě využití projektu v praxi.

Až se podaří odhalit a opravit chyby způsobující havárie migrovaných procesů, bylo by vhodné provést portaci projektu Clondike na některou z aktuálních verzí jádra Linux.

Rovněž by bylo v hodné do projektu zapracovat kontrolní mechanizmy zabraňující zneužití výkonu klastru Clondike, čímž se zabývá diplomová práce [35].

Literatura

[1] Amazon EC2 Spot Instances Pricing. *Amazon Web Services* [online] [cit. 2016-25-02]. Dostupné z: https://aws.amazon.com/ec2/spot/pricing/

[2] Artifact. *Atlassian Documentation: Bamboo Cloud Glossary* [online] [cit. 2016-08-05]. Dostupné z: https://confluence.atlassian.com/bamboocloud/artifact-737184512.html

[3] Bamboo variables. *Atlassian Documentation* [online] [cit. 2016-12-04]. Dostupné z: https://confluence.atlassian.com/bamboocloud/bamboo-variables-737184363.html

[4] Clondike. *Parallel and Distributed Computing Group* [online] [cit. 2016-21-02]. Dostupné z: http://pcg.fit.cvut.cz/structure/clondike

[5] Continuous Integration Essentials. *CODESHIP* [online] [cit. 2016-25-02]. Dostupné z: https://codeship.com/continuous-integration-essentials

[6] Definition of: user space. *PC Magazine* [online] [cit. 2016-11-03]. Dostupné z: http://www.pcmag.com/encyclopedia/term/58234/user-space

[7] Frequently Asked Questions. *CircleCI* [online] [cit. 2016-25-02]. Dostupné z: https://circleci.com/pricing/#faq-section-linux

[8] Getting started with Bamboo Cloud. *Atlassian Documentation* [online] [cit. 2016-16-04]. Dostupné z: https://confluence.atlassian.com/bamboocloud/getting-started-with-bamboo-cloud-737183848.html

[9] Interpreter Vs Compiler : Difference Between Interpreter and Compiler. *Programiz* [online] [cit. 2016-24-03]. Dostupné z: http://www.programiz.com/article/difference-compiler-interpreter

[10] Library (computing). *The Free Dictionary* [online] [cit. 2016-21-04]. Dostupné z: http://www.thefreedictionary.com/Library+(computing)

[11] Open Source Project License Request. *Atlassian* [online] [cit. 2016-14-03]. Dostupné z: https://www.atlassian.com/software/views/open-source-license-request

[12] Operating system / Platform. *Opentracker* [online] [cit. 2016-23-02]. Dostupné z: http://www.opentracker.net/resources/glossary/operating-system-platform

[13] OS - operating system. *Webopedia* [online] [cit. 2016-24-02]. Dostupné z: http://www.webopedia.com/TERM/O/operating_system.html

[14] Running VMware Disk Mount on a Linux Host. *vSphere Documentation Center* [online] [cit. 2016-18-04]. Dostupné z: https://pubs.vmware.com/vsphere-50/index.jsp?topic=%2Fcom.vmware.vddk.utils.doc_50%2Fdiskutils_mount.4.4.html

[15] SecurityToken Class. *Microsoft Developer Network* [online] [cit. 2016-17-04]. Dostupné z: https://msdn.microsoft.com/en-us/library/system.identitymodel.tokens.securitytoken.aspx

[16] Setting up your first remote agent. *Atlassian Documentation* [online] [cit. 2016-24-02]. Dostupné z: https://confluence.atlassian.com/bamboocloud/setting-up-your-first-remote-agent-792305742.html

[17] Supported platforms. *Atlassian Documentation* [online] [cit. 2016-25-02]. Dostupné z: https://confluence.atlassian.com/bamboo/supported-platforms-289276764.html

[18] Test Environment. *CircleCI Documentation* [online] [cit. 2016-23-02]. Dostupné z: https://circleci.com/docs/environment/

[19] Understanding the Bamboo CI Server. *Atlassian Documentation* [online] [cit. 2016-09-05]. Dostupné z: https://confluence.atlassian.com/bamboocloud/understanding-the-bamboo-ci-server-737183798.html

[20] What Is a Socket? *The Java Tutorials* [online]. Oracle [cit. 2016-19-04]. Dostupné z: https://docs.oracle.com/javase/tutorial/networking/sockets/definition.html

[21] What is Kernel? *Techopedia* [online] [cit. 2016-25-02]. Dostupné z: https://www.techopedia.com/definition/3277/kernel

[22] What is Linux. *The Linux Foundation* [online] [cit. 2016-07-03]. Dostupné z: http://www.linuxfoundation.org/what-is-linux

[23] Kernel Space Definition. *The Linux Information Project* [online]. Únor 2005 [cit. 2016-25-02]. Dostupné z: http://www.linfo.org/kernel_space.html

[24] What is script? *Search Enterprise Linux* [online]. Srpen 2005 [cit. 2016-02-04]. Dostupné z: http://searchenterpriselinux.techtarget.com/definition/script

[25] Attems, M.; Bailey, J.; aj.: mkinitramfs. *Linux Man Pages Online* [online] [cit. 2016-27-03]. Dostupné z: http://man.he.net/man8/mkinitramfs

[26] Bell, P.; Beer, B.: *Introducing GitHub: A Non-technical Guide*. O'Reilly Media, Inc., 2014.

[27] Déchelle, D. P. F.; Geiger, G.; Phillips, D.: Demudi: The debian multimedia distribution. *ICMC2001*, 2001. Dostupné z: http://www.demudi.org/doc/paper.pdf

[28] Dennis, M.; Roland, M.; Mike, F.: make(1) - Linux man page. *Linux Documentation* [online] [cit. 2016-03-04]. Dostupné z: http://linux.die.net/man/1/make

[29] Fowler, M.; Foemmel, M.: Continuous integration. *Thought-Works*, 2006: str. 122. Dostupné z: http://www.thoughtworks.com/ContinuousIntegration.pdf

[30] Gattermayer, J.; Tvrdik, P.: Different Approaches to Distributed Compilation. In *Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW), 2012 IEEE 26th International*, May 2012, s. 1128–1134, doi:10.1109/IPDPSW.2012.137.

[31] Guan, X.: *Shell Scripting*. Louisiana State University, Únor 2016. Dostupné z: http://www.hpc.lsu.edu/training/weekly-materials/2016-Spring/Shell-Scripting-17Feb2016.pdf

[32] Horalek, J.; Cimler, R.; Sobeslav, V.: Virtualization solutions for higher education purposes. In *Radioelektronika (RADIOELEKTRONIKA), 2015 25th International Conference*, IEEE, 2015, s. 383–388.

[33] Juhl, J.: Applying Patches To The Linux Kernel. *The Linux Kernel Archives* [online]. Srpen 2005, 2006-01-05 [cit. 2016-01-03]. Dostupné z: https://www.kernel.org/doc/Documentation/applying-patches.txt

[34] Lehey, G.: *Porting UNIX software: from download to debug*. O'Reilly & Associates, Inc., Duben 1995. Dostupné z: http://www.lemis.com/grog/Documentation/PUS/porting_unix_software-complete.pdf

[35] Nový, Z.: *Implementace škodné do projektu Clondike*. Diplomová práce, České vysoké učení technické v Praze, Fakulta informačních technologií, Praha, 2016.

[36] Nyers, L.; Čović, Z.: Distance control of virtual machines in education of a computer laboratory. In *Intelligent Systems and Informatics, 2008. SISY 2008. 6th International Symposium on*, IEEE, 2008, s. 1–3.

[37] Otte, S.: Version Control Systems. *Computer Systems and Telematics, Institute of Computer Science, Freie Universität, Berlin, Germany*, 2009.

[38] Ries, E.: Continuous deployment in 5 easy steps. *Radar O'Reilly* [online]. O'Reilly Media, Inc., 2009 [cit. 2016-25-02]. Dostupné z: `http://radar.oreilly.com/2009/03/continuous-deployment-5-eas.html`

[39] Rákosník, J.; Nový, Z.: Clondike install manual. *Projekt Clondike studentu FIT a FEL CVUT v Praze* [online]. GitHub [cit. 2016-25-02]. Dostupné z: `https://github.com/FIT-CVUT/clondike/blob/master/INSTALL`

[40] Salzman, P. J.; Burian, M.; Pomerantz, O.: What Is A Kernel Module? *The Linux Kernel Module Programming Guide* [online] [cit. 2016-01-03]. Dostupné z: `http://linux.die.net/lkmpg/x40.html`

[41] Schmidt, A.-D.; Schmidt, H.-G.; Clausen, J.; aj.: Enhancing security of linux-based android devices. In *Proceedings of 15th International Linux Kongress. Lehmann*, 2008.

[42] Schneider, R.: Method, system for using file name to access application program where a logical file system processes pathname to determine whether the request is a file on storage device or operation for application program. Červenec 15 2003, US Patent 6,594,675.

[43] Shinpaugh, K.; VanDyke, J.: Introduction to the UNIX Operating System Joe VanDyke. Červenec 1998. Dostupné z: `http://www.vascan.org/webdocs/services/introunix.pdf`

[44] Wang, L.; Tao, J.; Kunze, M.; aj.: Scientific Cloud Computing: Early Definition and Experience. In *HPCC*, ročník 8, 2008, s. 825–830.

Příloha **A**

Seznam použitých zkratek

Clondike Cluster Of Non-Dedicated Interoperating Kernels [4]

SCP Secure Copy - Zabezpečené kopírování

JDK Java Development Kit - Vývojářský balík Java

OS Operating System - Operační systém

API Application Program Interface - Rozhraní pro programování aplikací

CI Continuous Integration - Průběžná integrace

JRE Java Runtime Environment - Běhové prostředí Java

NPFS Named pipe file system - Meziprocesový komunikační kanál

Příloha **B**

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `Bamboo-Clondike.lnk` ..... odkaz na implementované řešení
- `src`
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `BP_Hartman_Michael_2016.pdf` ..... text práce ve formátu PDF