Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Bakalářská práce

Vytvoření a implementace administračního rozhraní pro virtualizační platformu KVM

Weiss Pavel

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Pavel Weiss

Systémové inženýrství a informatika
Informatika

Název práce

Vytvoření a implementace administračního rozhraní pro virtualizační platformu KVM

Název anglicky

Creating and implementing an administration interface for KVM virtualization platform

---

Cíle práce

Cílem této bakalářské práce je vytvořit administrační rozhraní, umožňující ovládání a monitoring KVM (Kernel-based virtual machine), registrace, správu virtuálních strojů a správu pravomocí. Rozhraní by mělo být přístupné skrze instalační skript, dále by mělo umožnit připojení na MySQL nebo ORACLE databáze. Rozhraní bude ovládáno pomocí programovacího jazyka PHP, nebo přímo pomocí virsh příkazů. Celé rozhraní by mělo ulehčit a zefektivnit práci na KVM virtuálních strojích.

Metodika

Metodika bakalářské práce bude založena na vytipování relevantních informačních zdrojů z oblasti virtualizace a vytváření aplikací pomocí CodeIgniteru, jenž je open source framework pro programovací jazyk PHP, za účelem získání potřebných poznatků, jenž poslouží jako teoretická východiska pro vlastní práci. K vytváření bude sloužit software určený k vytvoření administračního rozhraní, například jako PhpStorm nebo PSPad. Výsledek vlastní práce bude podroben hodnocení z hlediska uživatelské přívětivosti a funkčnosti a zároveň bude implementován buď na virtuální stroj, běžící v reálném čase, nebo na virtuálním stroji, spuštěném záměrně pro tuto práci.

Doporučený rozsah práce

35-40 stran

Klíčová slova

KVM, Linux, php, virtualizace, administrační rozhraní, CodeIgniter

---

Doporučené zdroje informací

British Columbia Institute of Technology. CodeIgniter User Guide [online]. ©2014. Dostupné z: https://www.codeigniter.com/user_guide/

Cooper Alan. About Face: The Essentials of Interaction Design. 4. USA: John Wiley & Sons, Inc.. ©2014. 722s. I SBN: 978-1-118-76657-6

Ivanov Konstain. KVM Virtualization Cookbook. UK : Packt Publishing. ©2017. 471s. ISBN 978-1-78829-467-6

Nixon Robin. Learning PHP, MySQL, JavaScript, and CSS. 2. USA: O'Reilly Media. ©2012. 582s. ISBN: 978-1-449-31926-7

The kernel development community. The Linux Kernel [online]. ©2016. Dostupné z: https://www.kernel.org/doc/html/v4.10/index.html/

Předběžný termín obhajoby

2019/20 LS – PEF

Vedoucí práce

Ing. Jiří Brožek, Ph.D.

Garantující pracoviště

Katedra informačního inženýrství

Elektronicky schváleno dne 19. 2. 2020

Ing. Martin Pelikán, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 19. 2. 2020

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 15. 03. 2020

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci " Vytvoření a implementace administračního rozhraní pro virtualizační platformu KVM " jsem vypracoval(a) samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 22.3.2020 _____

Poděkování

Rád bych touto cestou poděkoval svému vedoucímu bakalářské práce Ing. Jiřímu Brožkovy, Ph.D. z katedry informačního inženýrství, za rady a tipy, potřebné pro tvorbu bakalářské práce. Též bych rád poděkoval Ing. Jaromíru Salákovi, za spoustu poskytnutých informací z oboru informačních technologií a za morální podporu

Vytvoření a implementace administračního rozhraní pro virtualizační platformu KVM

Abstrakt

Tato bakalářská práce řeší samostatnou tvorbu administračního rozhraní, včetně zásad spojených s jejich tvorbou a možnou implementací. Cílem bylo vytvoření administračního rozhraní, jež by umožňovalo ovládat virtuální stroje běžící na Linuxovém modulu KVM. Práce probíhala na základě prvotního průzkumu stávajících řešení pro virtualizaci, podrobného průzkumu možností pro tvorbu rozhraní, které by bylo dostupné mezi platformami a následně vyhodnocení jejich nedostatků či naopak jejich vhodných funkcí a výkonnosti. Po vybrání klíčových prvků pro vytvoření rozhraní se práce věnuje právě dané tvorbě a možné implementaci. K řešení tvorby byl využit již zmiňovaný modul KVM, virtualizační nástroj Virsh, hypertextový preprocesor (PHP), framework CodeIgniter a především PhpStorm určený pro vznik kódu. Podstatou konečného rozhraní je především jeho jednoduchost, přehlednost a rozšiřitelnost spolu s funkcemi, které poskytují informace z prvotního průzkumu. Výsledkem je tedy administrační rozhraní, které jednoduše zvládne ovládat virtuální stroje, předává informace o nich, obsahuje možnost zadávání příkazů ručně. A zároveň je snáze implementovatelné, především díky informacím z této práce a stylizaci kódu.

Klíčová slova: KVM, Linux, PHP, virtualizace, administrační rozhraní, CodeIgniter

Creating and implementing an administration interface for KVM virtualization platform

Abstract

This bachelor thesis solves creation of administration interface, including principles related to their creation and possible implementation. The aim was to create an administration interface that would allow to control virtual machines running on Linux KVM module. The work was based on the initial survey of existing virtualization solutions, a detailed survey of the possibilities for creating interfaces that would be available between platforms and subsequently evaluate their shortcomings or their appropriate functions and performance. After selecting the key elements for creating the interface, the thesis deals with the given creation and possible implementation. The above mentioned KVM module, Virsh virtualization tool, hypertext preprocessor (PHP), CodeIgniter framework and especially PhpStorm designed for code creation were used. The essence of the final interface is primarily its simplicity, clarity and extensibility along with the functions that provide information from the initial survey. The result is an administration interface that simply manages virtual machines, transmits information about them, and includes the ability to enter commands manually. At the same time, it is easier to implement, mainly thanks to information from this work and code stylization.

Keywords: KVM, Linux, PHP, virtualization, administration interface, Codeigniter

Obsah

Seznam obrázků

Seznam tabulek a grafů

Seznam použitých zkratek

PHP – Hypertext Preprocessor
XML – Extensible Markup Language
HTML – Hypertext Markup Language
VM – Virtuální stroj
KVM – Kernel-based Virtual Machine
CLI – Příkazová řádka

1 Úvod

Využívání administračních rozhraní je součástí práce administrátorů v oboru informačních technologií. Využívají jej především pro spravování a získávání informací o informačních systémech. Správné administrační rozhraní pomáhají a ulehčují práci s různými systémy, skrze mnoho odvětví. Ovládání systémů lze několika způsoby, kdy nejpoužívanější je textové rozhraní (příkazová řádka), která umožňuje pracovat se systémy podrobněji a využívat všechny jejich funkce a dále grafické rozhraní, případně administrační rozhraní, které ale může více či méně ovlivňovat rozsah možností, avšak poskytuje přívětivější přístup k systému. Spolu s grafickým rozhraním je často používáno již zmíněné textové rozhraní. Rozhraní se mohou dělit podle různých kritérií, která budou prozkoumána. Kdy se tedy sleduje především odvětví a dovednosti uživatele.

Virtualizace je značným milníkem v poskytování služeb a informací uživatelům internetu. S virtualizací je spojeno několik postupů a technik, jenž poskytují tvůrcům (programátorům) mnoho možností. Hlavními znaky virtualizace jsou přizpůsobitelnost potřebám uživatelů, možnost oddělení nezávislých služeb a snadnější využívání zdrojů. Pro virtualizaci na platformě KVM (kernel-based machine), jsou využívána již některá rozhraní. Nejčastěji jej využívají korporace, spravující rozsáhlé množství serverů, nebo i menší společnosti, využívající jej například pro správu serverů, nabízených k hostingu.

Hlavním cílem pro vytvoření rozhraní, je splnění minimálních požadavků pro ovládání virtuálních strojů a při nejmenším poskytování možností, jež jsou dostupné v již známých rozhraních a zjištění požadavků uživatele. Pro splnění cíle je potřebný průzkum stávajících nástrojů ať už volně dostupných nebo dostupných pod placenou licencí. Zjištění možností virtualizace, možnosti nadstaveb (rozšíření) virtualizace, a především značné porozumění danému tématu.

2 Cíl práce a metodika

2.1 Cíl práce

Cílem práce je prozkoumání funkcí a eliminace chyb nebo nedostatků stávajících rozhraní pro virtualizační platformu KVM. Poté navrhnout administrační rozhraní, jež bude splňovat minimální požadavky jako jsou ovládání VM, jejich monitoring a správu. Následně pak vytvoření daného rozhraní a otestování, zda dané požadavky splňuje a je funkční. Dané rozhraní bude připojené na databázi a zároveň bude umožňovat rozšiřitelnost a úpravu zkušenějším uživatelům.

2.2 Metodika

Základem metodiky je především studium literatury vztažené k danému tématu a analýze informací z věrohodných zdrojů. Jedná se především o průzkum stávajících řešení z oblasti virtualizace, a to konkrétně grafických rozhraní pro modul KVM. Součástí průzkumu je též vytipování vhodných frameworků pro PHP, jejich porovnání a výběr relativně vhodného kandidáta pro tvorbu rozhraní. Zároveň půjde o vytvoření kódu, které bude dostupné pro nováčky ale i profesionály.

Dále pomocí znalostí získaných při studiu a poznatků z analýzy, vybrat nejvhodnější nástroje pro tvorbu rozhraní. Dané rozhraní musí splňovat minimální požadavky ve formě vhodných funkcí, které jsou zjištěny pomocí analýzy a zároveň odstranění nedostatků stávajících řešení pro virtualizaci. Spojením poznatků z teoretické a praktické části následně dané rozhraní vytvořit.

3 Teoretická východiska

3.1 Linux

Svobodný operační systém založený na unixovém typu. Jedná se o základní softwarové vybavení počítače, který umožňuje jeho užívání a ovládání. Je to prostředník mezi uživatelskými programy a hardwarem. Od programů dostává požadavky a v případě, že to podmínky dovolí, jim přiděluje hardwarové prostředky (čas procesoru, místo v paměti a podobně). Spouští se jako první po spuštění počítače. Linux se stal velmi známým operačním systémem, především pro volnost, kterou poskytuje vývojářům a administrátorům. [1]

Občas se označení Linux používá jako název pro operační systém spolu se softwarem a často je milně chápán pouze jako příkazová řádka, což není pravda. Přes příkazovou řádku sice lze systém ovládat, ale je též k dispozici spolu s grafickým rozhraním. Je vyvíjen pod licencí GNU, proto je jeho zdrojový kód volně k dispozici. Jedná se o operační systém určený pro osobní počítače a servery. Upravený Linux je k dispozici i pro jiné platformy, než jsou osobní počítače. Je složen z několika částí, kde na úplném pozadí je jádro, které komunikuje s hardwarem a poskytuje služby procesům. [1] [2]

Spouštění úloh umožnuje jádro tohoto systému, kdy programy mohou být složeny i z více procesů. Každý proces pak může mít vlastní podprocesy. Jedná se o multi-uživatelskou platformu, tedy že jeden uživatel může spouštět i programy ostatních uživatelů, kdy může jeden nebo více uživatelů spravovat počítač a ostatní jen využívají pro svou práci.

Na rozdíl například od Windowsu, jenž je poskytován pouze společností Microsoft, má Linux mnoho distribucí. Tudíž prvotní kontakt s Linuxem může být lehce zmatečný. Dají se rozdělit na distribuce od komerčních firem jako je Red Hat či Novell a distribuce tvořené komunitou, z nejznámějších Debian a Fedora. Tyto distribuce mohou mít různé využití od domácího využití po tvorbu serverů. Rovněž se za některé platí a některé jsou zcela zdarma a dovolují volné sdílení a upravování.

Nejznámější distribuce

Debian

Nejstarší z distribucí. Striktní open-source distribuce je vyvíjena ze strany komunity a je zavázána zůstat svobodným softwarem na vždy. Na tom si primárně tako distribuce zakládá, poskytuje totiž vysokou míru možností, jak nakládat s touto distribucí. Skládá se ze základních programů a pomůcek. Je k dispozici na jedenácti počítačových architekturách.
K dispozici je jako instalační médium nebo jako „LIVE“ distribuce, kdy pro běh stačí například flash disk, na který se Debian nahraje a není potřeba dalších instalací. [3]

Fedora

Založená na distribuci Red Hat a stejnojmennou společností sponzorovaná. Jedná se však o volnou distribuci, podobnou Debianu a je znatelně zaměřena na osobní počítače. Je k dispozici pro známější platformy. Obecně bývá k dispozici s desktopovým prostředím GNOME, avšak se dá přednastavit i prostředí jiné. Stáhnout se dá jako Fedora Workstation, Fedora Server a Fedora Cloud. [4]

Red Hat

Jedna z prvních distribucí, jejíž název je značně viditelný i na logu této distribuce (červený klobouk). Nejedná se o svobodný software, tudíž jsou verze tohoto Linuxu placené. Jelikož se jedná o komerční distribuci, má podporu ve společnosti Red Hat, která se stará o chod a bezpečnost tohoto systému. [5]

SUSE (Novell)

Tato distribuce se dělí komerční a komunitní verzi (openSUSE). Je dosti přívětivá hlavně pro začátečníky. Komunitní verze je pak vyvíjena a testována její komunitou. [6] Dovoluje využití balíčků od ostatních distribucí. O komerční verzi se společnost stále stará a vydává pro ni aktualizace. Je k dispozici zdarma po dobu 60 dnů, případně lze zakoupit ihned. [7]

Ubuntu

Vychází z Debianu a přímo se specializuje na desktopovou verzi. Je volně

> k dostání, a navíc má jak komunitní, tak i profesionální podporu. Nové verze jsou pravidelně vydávány a mají omezenou podporu v opravách a aktualizacích. Je též vhodný i pro začátečníky. V posledních letech se Ubuntu začalo zaměřovat i na provedení ve verzích pro cloud, server, internet věcí a přenosná zařízení. Projekt Ubuntu se zaměřuje i na školy, kdy pomocí vzdělávajícího softwaru podporuje studenty a učitele. [8]

Podobně jako na ostatních operačních systémech, jsou i zde k dispozici programy ve stylu repositářů. Ty jsou založeny na systémů balíčků, které obsahují balíčky s programy, jenž lze instalovat. Tento způsob umožňuje snadné instalování a odebírání částí systému a aplikací. Součástí repositářů jsou jak aktualizace systému, tak i některé opravy chyb. [2]

Linux lze velmi dobře upravovat pro vlastní požadavky. Například jej lze plně ovládat pomocí příkazové řádky ze vzdáleného zařízení stejně, jako kdyby byl fyzicky k dispozici. Chování tohoto systému je v celku předvídatelné a lze je různě ovládat. Díky dostupnosti, je pro něj vyvíjena řada programů případně i her, které jsou dostupné i pro ostatní operační systémy. Navíc pak umožňuje využít programu Wine, který dovoluje na Linuxu spouštět aplikace dostupné na systému Microsoft Windows. [2] To však není jediná z možností, jak tyto aplikace spouštět. Lze využít funkcí virtualizace, kdy se na daném systému vytvoří virtualizovaný operační systém a ten se chová, jako kdyby běžel odděleně na vlastním fyzickém stroji. To dovoluje též spouštět i ty hry, které pro Linux nejsou dostupné, jelikož většina společností nebere tuto platformu dostatečně výdělečnou v oblasti her. [9]

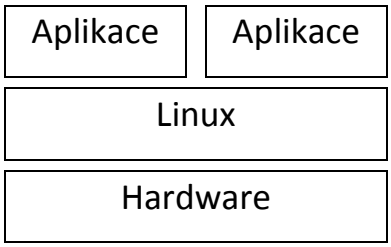
3.2 Virtualizace

Pojmem virtuální se často rozumí kombinace fyzické výbavy počítače a programu. Pojí se s tím tedy i sám internet, kdy samotný server, který poskytuje služby je brán jako součást virtuálního světa. Zde je však myšlena virtualizace ve smyslu vytvoření virtuálního stroje na stroji, s již běžícím operačním systémem. Pokud se tedy připojíme na takovýto stroj, chová se jako skutečný počítač i když běží jen jako proces. [10]

Nynější virtualizace dovoluje běh několika virtuálních strojů s odlišným operačním systémem, prostředím a hardwarovým vybavením. Pro představu, na počítači s operačním systémem Linux, čtyř-jádrovým procesorem a 8 GB operační paměti běží 2 virtuální stroje. Jeden ze strojů disponuje operačním systémem Windows 10, 2 jádry procesoru a 4 GB operační paměti. Na druhém je pak nainstalován Linux, který má jedno jádro procesoru a 2 GB operační paměti. Oba systémy běží nezávisle na sobě i přes to, že jsou spuštěna na jednom počítači. Některé z nástrojů pro virtualizaci disponují nadměrným využíváním zdrojů. Neboli že mají větší počet procesorů nebo větší paměť, oproti původnímu počítači. V celku jde tedy ve své podstatě o iluzi, ve které vytváříme virtuální stroje, které mají vlastní operační systém a vlastní virtuální zdroje (procesor, paměť, disk, síťové karty a podobně). Čím že vlastně vytvoří celý virtuální počítač. Virtualizace je ve své podstatě označení technik, metod a prostředků, které dovolují v počítači přistupovat ke zdrojům jiným než fyzickým způsobem. [9]

Virtualizace má mnoho důvodů. Mezi primární se může řadit spolehlivost a šetření zdrojů. Důvod je však individuální a odvíjí se od dané situace. Šetření zdrojů je bráno ve smyslu nákladů na provoz, včetně energií a prostor, potřebných k uschování serverů. [10] Místo několika serverů se snadno může využít jeden s dostatečnými hardwarovými zdroji. Existuje mnoho výhod, kde všechny vychází z možnosti virtuální stroje kdykoliv vypnout, zapnout, omezit nebo jim přidat zdroje, přesouvat je jednoduše mezi fyzickými servery, například kvůli údržbě. Migrace pak může probíhat i bez výpadku služeb. Záloha a obnova pak má jednoduchý princip ve formě obrazů disku a paměti. V případě výpadku nebo porušení integrity dat, lze tyto obrazy využít a jednoduše pokračovat od posledního uloženého bodu. Jednou z vlastností je i bezpečnost virtualizace. Virtuální stroje mohou běžet samostatně bez potřeby komunikovat s dalšími stroji, tudíž klient se připojí jen na povolený virtuální stroj bez zásahu do ostatních. Navíc tento stroj nemusí mít povolenou komunikaci přímo s hardwarem, ale volá jen funkce systému. Virtualizace se hodí i na testování, když je potřeba odzkoušet aplikaci na několika verzích systému, případně na odlišném operačním systému s jiným vybavením, a to bez ztráty dat či ohrožení hostujícího stroje. Existují i další případy, kdy není potřeba vytvářet kompletní virtuální systém. Například metoda diskového zapojení RAID. Ta umožňuje zapojení více disků, které se tváří jako jeden, případně naopak, z jednoho vytvořit disků víc.

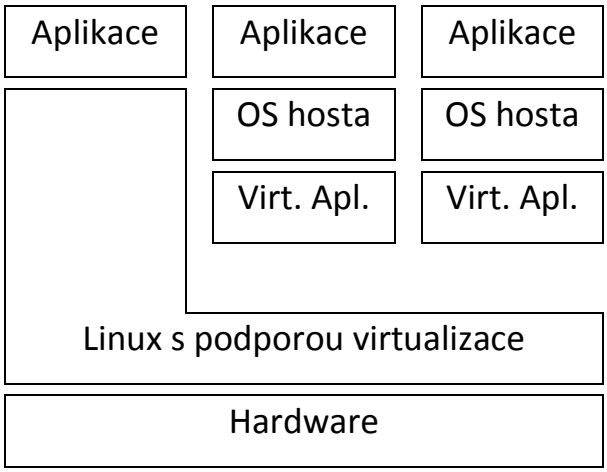
3.2.1 **Virtualizace na úrovni operačního systému**



Obrázek 1 - Schéma virtualizace na úrovni OS [11]

Jedná se o druh virtualizace, kdy se v rámci jednoho OS vytváří oddělená prostředí. Hlavní výhodou je nízká náročnost na zdroje. Nejedná se o oddělená prostředí v plném slova smyslu, oddělení není úplné. Virtualizační vrstva se nachází mezi OS a virtuálními stroji. Umožňuje to například několikanásobný běh jedné služby, bez potřeby vytvářet pro každou jednotku vlastní systém. Podporuje tedy jen jeden OS na hostující server a dané služby. Jednou ze známých služeb je FreeBSD Jail, jenž je vylepšením chrootu. [12]

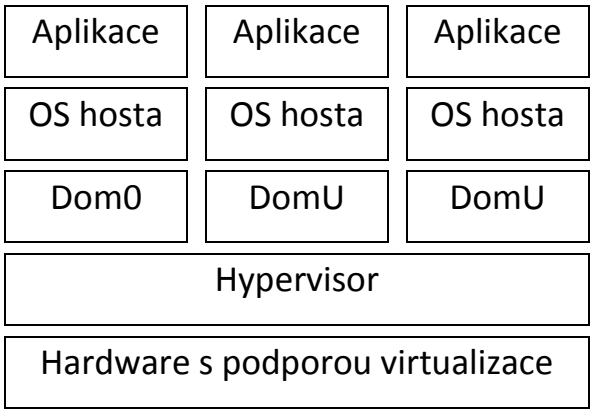
3.2.2 **Paravirtualizace**



Obrázek 2 - Schéma paravirtualizace [11]

Hlavní částí je jádro operačního systému hostujícího stroje, které je speciálně upraven a zároveň disponuje hypervisorem, který dovoluje virtuálním strojům přístup k hardware. Operační systém hostovaného stroje je též upraven, tudíž nemá přímý přístup k hardwaru, ale jejich přístupy jsou vedeny na hypervisora. Paravirtualizace poskytuje větší výkon oproti úplné virtualizaci díky efektivní komunikaci s hypervisorem, na druhou stranu však je hostovaný stroj do značné míry závislý na hlavním stroji. Tento typ virtualizace umožňuje například Xen. [13]

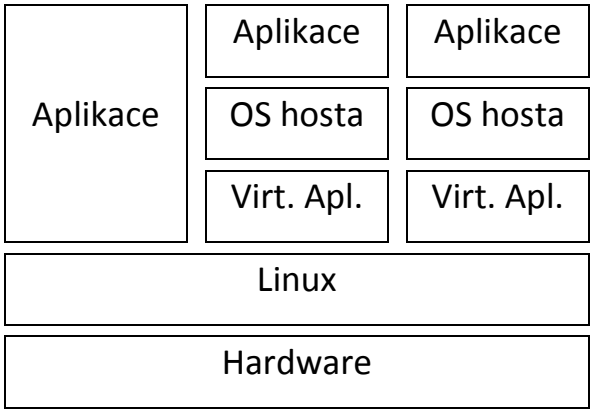
3.2.3 **Úplná virtualizace**



Obrázek 3 - Schéma úplné virtualizace [11]

Veškeré součásti stroje jsou virtualizované. Požadavek na hardware je v tomto případě předán hostitelskému systému, požadavek je zpracován a poslán zpět na hostovaný stroj. Úplná virtualizace vyžaduje, aby hostovaný systém měl stejnou architekturu jako hostující. Dochází zde ke sdílení stejných prostředků a instrukčních sad jako na fyzickém stroji a systém nepozná, že běží v prostředí, kde sdílí prostředky s dalšími stroji. Je zde oddělena fyzická a programová vrstva, čímž dochází ke značnému snížení výkonu. Hypervisor zde představuje vybavení stroje a některé z operací provádí ve vlastním prostředí na místo využití hardwaru. Zde není potřeba operační systém nijak upravovat. Zástupcem je například Hyper-V. [13]

3.2.4 **Emulace**



Obrázek 4 - Schéma emulace [11]

Emulace dovoluje provozovat virtuální systém jiné architektury. Zde je však potřeba všechny operace hostovaného stroje interpretovat. Kvůli tomu dochází ke snížení výkonu. Jedná se o nejméně efektivní způsob, jak virtualizovat. I přes to je často využíván pro jeho

výhody. Možnost použít i jiné architektury je využívána pro tvorbu softwaru pro procesory, ke kterým není fyzický přístup, nebo spouštět aplikace, dostupné pro jiný operační systém. Instrukce hostovaného stroje jsou interpretovány buď staticky nebo dynamicky. Statický překlad nejdříve celý program zpracuje a poté ho provede. Dynamický se provádí za běhu, díky čemuž je rychlejší. Zástupcem emulace na dynamickém překladu je QEMU. [11]

3.3 Virtualizační nástroje

3.3.1 KVM

KVM, je open source, založený na Linuxovým modulu, poskytuje nástrojům třetích stran zajistit virtualizaci. Jelikož se jedná o modul jádra, KVM využívá opakovaně mnoho funkcí linuxového jádra a vytváří infrastrukturu pro virtualizaci. Je možné ho využít jako úplné řešení pro Linux na x86 strojích, s technologií Intel VT nebo AMD-V, tudíž nefunguje na procesorech, bez virtualizační schopnosti. Podporuje též Sony Playstation 3, který má oficiální podporu Linuxu. Umožňuje běh mnoha virtuálních strojů, jako je Linux nebo Windows, při čemž každý stroj má vlastní virtuální hardware (disk, grafická karta, síťová karta a další). [14]

KVM je součástí Linuxu od verze 2.6.20, což se pojí s trendem virtualizace. Přidáním ovladače, umožňující virtualizaci hardwaru. Přidává i uživatelské zařízení (/ dev / kvm), zpřístupňující uživatelské možnosti virtualizace. Ovladač umožňuje spuštění virtuálního počítače v plně virtualizovaném počítači (vlastní virtuální pevné disky, síťové a grafické adaptéry a displej). Ovladač má 3 provedení a to jádro, uživatelský režim a režim hosta. Mezi uživatelské zařízení patří například QEMU. [14]

Pracuje na principu konverze Linuxu do type-1 hypervizora. Každý hypervizor pak potřebuje operační systém, komponenty jako plánovač procesů, zásobník vstupů a výstupů, správce paměti a další, k běhu virtuálních strojů. Tyto komponenty potřebuje, jelikož je částí Linux kernelu. Každý virtuální stroje je implementován jako obyčejný proces, ovládaný standardním plánovačem Linuxu, s dedikovaným virtuálním hardwarem. [15]

Funkce KVM

-Bezpečnost

KVM využívá kombinace zvýšeného zabezpečení Linuxu (SELinux) zabezpečené virtualizace (sVirt) pro větší bezpečnost a izolaci virtuálních strojů. sVirt rozšiřuje zabezpečení kolem virtuálních počítačů (SELinux), čímž umožňuje aplikaci povinné kontroly přístupu (MAC), na vytvořený virtuální stroj a zabraňuje chybám, zadaných ručně. [14]

-Úložný prostor

Schopnost používat jakékoli uložiště, které je podporované systémem Linux, včetně lokálních disků a síťového uložiště (NAS). Automatická konfigurace mapovače zařízení lze pak použít pro vylepšení úložiště a zajištění redundance. Je zde také podpora sdílení systému souborů, umožňující sdílení obrazu VM mezi více hostů. Obrazy disků alokují místo úložiště na požádání, nikoliv však dopředu. [14]

-Podpora Hardwaru

Certifikované hardwarové platformy, podporované Linuxem, jsou k dispozici pro KVM použití. Může za to přispívání výrobců hardwaru k vývoji jádra, tudíž jsou mnohé hardwarové funkce v jádře Linuxu rychle přijímány. [14] [15]

-Správa paměti

KVM využívá správy paměti systému Linux. Paměť VM lze tedy zaměnit, zálohovat velkými svazky pro lepší výkon a sdílet či zálohovat souborem disku. [14]

-Migrace

Podpora migrace neboli schopnosti přesouvat běžící VM mezi hostiteli bez přerušení. VM zůstane zapnutý, stejně jako síťové připojení a běžící aplikace, během přenosu VM. KVM si stav VM ukládá, lze ji tedy uložit a později obnovit. [14] [15]

-Přidání zařízení za běhu

Virtuálním strojům dovoluje libovolně přidávat a odebírat PCI zařízení, paměť nebo procesory během toho, co systém na nich běží, a to bez potřeby

přerušit běh stroje. Tato funkce se nazývá Hotplug a dá se přirovnat funkci Plug-and-play. Například Hyper-V zatím funkci Hotplug nepodporuje i když na ní pracuje. [14]

3.3.2 **VMware vSphere**

Virtualizační software od americké společnosti Wmware. Je založen na principu cloudu. Ten umožňuje spravovat aplikace v běžném prostředí přes cloud napříč platformami. Je nabízen v několika verzích, kdy verze Platinum nabízí mnoho funkcí, jako je zvýšená bezpečnost, která je dále nabízena společností VMware. Jelikož se jedná o cloudovou službu, souvisí s tím poplatky, určené dle vybavenosti stroje a verze. Jednou z výhod vSphere je, že není potřeba investovat do hardwarové infrastruktury, avšak omezuje zásahy do systému a hardware není v případě potřeby fyzicky k dispozici. Po hlavním nastavení je stroj automaticky připraven, tudíž není potřeba vše ručně instalovat a nastavovat. Mezi další funkce patří: [16]

-Podpora výkonu a hybridní cloud

Hardware společnosti, je speciálně stavěn a upravován pro vysoký výkon, čímž nabízí vhodné prostředí. Umožňuje využívat strojové učení a umělou inteligenci, pro zajištění ještě vyššího výkonu. [16]

-Zjednodušení datových infrastruktur

Podporuje tím maximalizování bezpečnosti v případě kritické chyby. Dále umožňuje snadnou správu virtuálních sítí z centralizovaného rozhraní. [16]

-Migrace z Linuxu

Dovoluje přenést celý Linuxový server. Dochází k udržování dostupnosti a aktuálnosti systému, odvrací případné problémy a dává možnost obnovy po havárii. [16]

3.3.3 **Hyper-V**

Software společnosti Microsoft, dříve nazýván Windows Server Virtualization. Systém je rozdělen na 4 části. Hardware, jedna z hlavních částí systému. Hypervisor, technologie umožňující virtualizaci. Kernel Mode, který vytváří virtuální stroje a řídí komunikaci s

hypervisorem. Jako poslední je User Mode. Ten se pak stará o běh aplikací a správu virtuálních strojů. [17] [18]

Hyper-V je stavěný na serverovém systému. Vlastní tedy vlastní operační systém a pomocí virtualizace lze spouštět další virtuální systémy. Díky vlastnímu operačnímu systému je vhodný i pro běžné uživatele.

Může též běžet na operačním systému Windows 10, vyžaduje však některé systémové prostředky, v porovnání s ostatními možnostmi virtualizace, je o něco náročnější na hardware. Potřebuje 64bitový procesor (na 32bitových verzí není služba dostupná), minimálně 4GB RAM, což odpovídá přibližně 3 virtuálním strojům v jeden moment. Dále pak samozřejmě prostor na disku a procesor musí podporovat překlad adres druhé úrovně (SLAT). [19] Výhodou serverové verze je primárně využití paměti, výhradně virtuálními stroji. Jako další pak migrace virtuálních strojů mezi stroji. Windows verzi omezuje právě daná paměť. Tu využívají všechny procesy systému a virtuální stroje se tak o paměť dělí dle potřeby. [18]

3.3.4 **Xen**

Open-source projekt, vzniklý v laboratořích univerzity v Cambridge. Umožňuje paravirtualizaci, která je rychlejší než pravá virtualizace, avšak vyžaduje výraznou úpravu hostitelského operačního systému. Dovoluje též klasickou virtualizaci, ale jen ve spojení s procesorem, jenž jí má hardwarově ošetřenou. Pohání některé z největších cloudů, jako je Huawei a Oracle. Také je součástí několika bezpečnostních řešení. [20]

Je k dispozici spolu s distribucí Linuxu BSD. Za využití izolace ovladačů, je možné systém po kritické chybě spustit systém, pomocí ovladačů, uložených ve virtuálním stroji, a to bez poškození systému. Výhodou je i malá paměťová stopa. Používá totiž mikrokernel design, jenž je navíc odolnější než ostatní hypervizoři. [20]

Při porovnání KVM a Xen virtualizace, je patrné, že za normálního provozu je výkonnost srovnatelná. Bohužel však Xen virtualizace musí čekat, dokud nevyjde verze pro novější jádro. KVM je začleněn do hlavní řady jádra, tudíž získá nové vlastnosti rychleji. [20] [15]

3.3.5 **Porovnání virtualizačních nástrojů**

V případě vytvoření vlastní hardwarové infrastruktury vyhrává primárně Hyper-V virtualizace od společnosti Windows. Vede v oblasti RAM tak pevného disku. Dodává se navíc spolu s operačním Windows Server, tudíž podporuje i další možnosti společnosti Windows. KVM na druhou stranu disponuje možností vytvoření vlastního rozhraní, a to navíc za menší cenu oproti Hyper-V. Xen je v porovnání někde u konce. I přes své výhody je zatím méně používaný a nemá podporu ve stabilní firmě. Jelikož u Hyper-V není potřeba řešit operační systém, je tak i vhodnější pro rychlé nasazení oproti KVM, kdy se musí vytvořit nejen hardwarová infrastruktura, ale i softwarová konfigurace. Za využití operačního systému Debian verze 10, lze v nastavení rychle pokročit. Čímž se stává vhodným adeptem pro mnoho projektů, a to hlavně pro komerční účely. Instalace Debianu je připravena tak, aby vyhovovala hardwaru počítače. [15] [16] [18] [20]

VMvare vSphere je vhodný v případě, že uživatel nemá dostatek zdrojů k vytvoření vlastní infrastruktury. Nabízí rychlé a efektivní prostředí, možnost téměř okamžitého navýšení výkonu a bezpečnost garantovanou společností VMvare. Odpadne tedy starost o správu hardwaru a s tím spojené náklady na nákup a udržování infrastruktury. [16]

Parametr	KVM	vSphere	Xen	Hyper-V
Maximum procesorů na VM	240	128	128	64
Maximum RAM na VM	6 TB	6 TB	> 1 TB	1 TB
Maximální velikost VM disku	10 TB	62 TB	2 TB	64 TB
Migrace VM	Ano	Ano	Ano	Ano
Nadměrné využití zdrojů*	Ano	Ano	Ne	Ne
Kopírování VM	Ano	Ano	Ano	Ano
Cena	Zdarma	23160 CZK**	Zdarma	Zdarma***

Tabulka 1 - Porovnání parametrů virtualizačních nástrojů [15] [16] [18] [20]

Popis tabulky:
* Přidělení více procesorů nebo paměti, než je skutečných zdrojů
** vSphere = cena za edici Standard na 1 fyzický procesor
*** Hyper-V = potřeba brát zřetel na cenu Windows 10 / Windows server

3.4 PHP

Rekurzivní zkratkou Hypertext Preprocesor. Je vhodný ze jména pro vývoj dynamických webových stránek, kdy se může vkládat do HTML. Je otevřeným a hojně používaným skriptovacím jazykem. Dovoluje též vývoj konzolových aplikací. Na místo značného počtu pro výstup HTML, PHP stránky obsahují HTML, s kódem, který dělá požadovanou činnost. PHP se uvozuje na začátku a konci znaky <? Php a ?>, kterými se přeskakuje mezi HTML kódem a PHP kódem. [21]

Na rozdíl od skriptů, spouštěných na straně klienta se PHP kód spouští na straně serveru, kde vygeneruje HTML kód jako odpověď, jenž se pošle klientu bez toho, aniž by viděl zdrojový kód tohoto skriptu. Lze využít i možnosti, kdy se na straně serveru nakonfiguruje PHP tak, aby zpracovával veškeré HTML soubory, tudíž klient pak neodhalí způsob, jak je stránka tvořena. [21]

Svojí jednoduchostí láká hlavně nováčky. Existuje mnoho stránek, které sdílejí způsoby, jak psát určité metody, což velmi podporuje komunitu pro PHP. Podporuje nejen jednoduchost, ale umožňuje profesionálním programátorům využívat pokročilých funkcí. Seznam funkcí je sice obsáhlý, avšak již po krátké době může uživatel začít psát jednoduché skripty.

PHP je zaměřeno primárně na práci na straně serveru, to však neomezuje v možnosti využívat propojení externích aplikací se serverem delegováním požadavků na externí aplikaci, jež vrátí výstup. Příkladem toho je například shromažďování formulářových dat, vytváření dynamického obsahu, práce s cookies a mnoho dalšího. [22]

Oblasti využití PHP skriptů:

Server-side skriptování

Nejtradičnější a jedním z hlavních úkolů PHP. Vyžaduje to pouze serverový modul, webový server s PHP instalací a prohlížeč pro přístup k PHP programu. Vše pak může běžet na jednom stroji, což umožňuje experimentování a testování PHP skriptů. [21]

Skripty příkazového řádku

Vytváření PHP skriptu bez vyžadování serveru či prohlížeče. Postačuje pouze PHP analyzátor. Tato možnost je ideální pro skripty, které jsou pravidelně spouštěny například plánovačem úloh. Též umožňují zpracování jednoduchého textu. [21]

Psaní aplikací

I když PHP není nejvhodnějším adeptem pro vytvoření aplikací s grafickým rozhraním, dovoluje použití pokročilých funkcí PHP. Výhodou je též psaní mezi platformových programů. [21]

Mezi využívané se řadí i proto, že jej lze použít na mnoha operačních systémech, jako je Linux, Windows, macOS a pravděpodobně i dalších. Navíc má podporu mnoha webových serverů, podporujících modul FastCGI PHP. Programátor se může rozhodnout mezi procedurálním programováním, objektivně orientovaném programování anebo používat jejich kombinaci. Další funkcí je spojení s databází, která disponuje rozšířením ODBC, komunikace s různými protokoly (IMAP, http, …). Funkcí je mnohem více a všechny jsou k dispozici na oficiální webové stránce PHP. PHP není omezeno pouze na HTML, ale jeho schopností je i zpracování obrázků, PDF souborů, flash filmů, vytváření a použití XML. [21]

3.5 Příklad PHP skriptu

Jedním z nejznámějších prvních příkazů ať už v jazyce PHP nebo ostatních, je hláška „Ahoj světe!“. Bývá prvním příkladem, hlavně proto, že je to nejjednodušší příkaz a lehce se na něm ukáže nejzákladnější princip psaní.

```
<?php echo 'Ahoj světe!';?>
```

Zde je vidět že PHP skript je uvozen znaky <?php a ?>, jak je již uvedeno výše. Metoda echo znamená, že následující řetězec pošle jako výstup do prohlížeče. Každý příkaz je pak zakončen středníkem, díky čemuž se při zpracování pozná, kde jeden příkaz začíná a kde končí.

```
<?php
if(count($data) > 0)
{
    foreach($data as $row) { echo $row->jméno; }
}
else { echo 'V seznamu se nenachází data.';}
?>
```

Další PHP skript vypisuje ze seznamu (pole) jména. Podmínkou je, že v poli se musí nacházet alespoň jeden záznam. Je-li v poli alespoň jeden záznam, metoda foreach postupně

projede každý záznam v poli $data, pro každý záznam zanese položku do pole $row a pak vypíše hodnotu s označením jméno. Pokud tomu tak není, PHP vypíše hlášku „V seznamu se nenachází data.“.

3.6 CodeIgniter

Codeigniter je aplikační framework, což je sada nástrojů pro tvorbu webových aplikací pomocí PHP. Vyžaduje pouze verzi PHP 5.6 nebo novější (funguje i na starší verzi, což však není doporučováno kvůli bezpečnosti a problémy s výkonem). Jako další je pak pro mnoho aplikací potřeba databáze, jako například MySQL verze 5.1 a vyšší, Oracle DB, MS SQL, SQLite, PostrgreSQL a další. Na základě licence MIT, jej lze využít kdekoliv jen uživatel chce, navíc je zdarma. Hlavním záměrem CodeIngiteru je vyvíjení projektů rychleji, oproti psaní kódu úplně od začátku. Základem jsou poskytované sady knihoven pro běžně používané úlohy, jakož je i jednoduché rozhraní, logická struktura, přístup do databází, ověřování formulářů, manipulace s obrázky nebo práce s XML daty. Uživatel se tedy může více zaměřit na kreativitu s minimalizací kódu, který je potřebný pro splnění úlohy. Navíc dochází ke generování URL adres, založených na segmentech místo využití klasického řetězce k přístupu. [23]

Jádro CodeIgniteru potřebuje pouze pár malých knihoven, má tedy výrazně malý dopad na paměť a jednoduše se implementuje. Nejprve načítá základní knihovny, které jsou většinou od sebe odlišné. Ostatní knihovny jsou pak načítány pouze v případě požadavku na ně. Tyto knihovny lze rozšiřovat pomocí vlastních rozšíření, pomocníků nebo rozšířením tříd. Díky tomu je právě základ Codeigniteru tak malý a rychlý. Disponuje též velkou řadou dalších funkcí, jež jsou důkladně popsány v dokumentaci. Mezi ně patří například vysoká kompatibilita pro různé konfigurace a verze PHP, minimální potřebná konfigurace, nevyžaduje používání příkazového řádku, nepotřeba používání omezujících pravidel pro psaní kódu. [23]

Hlavním principem je využití M-V-C (Model-View-Controller) modelu. Ten odděluje prezentační část webu od jeho logické struktury. To je velice dobré hlavně pro projekty, kde uživatel po té využívá pouze šablony, například není potřeba opakovaně psát hlavičku a

zápatí webu, pouze se načte pomocí příkazu $this->load->view(‘jméno_pohledu‘);. Tím tedy minimalizuje vlastní kód. Controller neboli řadič, zde reaguje na události a zajišťuje změny v modelu, čímž aktualizuje pohled. View, tedy pohled, převádí data do podoby, vhodné pro uživatele daného webu. [23]

Codeigniter je navíc důkladně zdokumentován, což velmi často nebývá v této oblasti zvykem, jelikož to není součástí žádného standardu pro tvorbu frameworků. Všechno od prvního spuštění až po práci s pokročilými principy je velmi dobře zdokumentováno. Výhodu k tomu dodává i přátelská komunita uživatelů Codeigniteru, která se podílí na komunitním fóru, kde rozšiřuje řady knihoven, poskytuje návody a pomáhá novým uživatelům s vytvářením aplikací. [23]

Příklad kódu CodeIgniteru

Jako malý příklad poslouží následující příkazy napsané v řadiči. Ty však nepopisují úplnou funkčnost frameworku, jen poukazují na jeho možnosti.

```
<?php
class Server extends CI_Controller {

function virtStroje(){
    $this->load->view('header');
    $this->load->helper('html');
    $this->output->append_output('<p>Virtuální stroje:</p>');
    $provedPrikaz = $this->call_command("$this->cmdListRunVM");
    $vypis = stream_get_contents($provedPrikaz);
    foreach (explode(",", $vypis) as $row){
       $this->output->append_output
            ("<a href='".base_url()."server/load/$row'>$row</a>");
    }
    $this->load->view('footer');
}
?>
```

Zde funkce virtStroje načte pohled „header“, pomocníka „html“ a vypíše text „Virtuální stroje“. Dále provede příkaz $cmdListRunVM přes funkci call_commad, jehož výsledek je vložen do proměnné $provedPrikaz. Funkce stream_get_contents(); převede obsah na srozumitelný text, který je vložen do proměnné $výpis. Výsledkem je pole $výpis. Každý

záznam v poli je příkazem foreach rozdělen a reprezentován proměnnou $row a ihned vypsán jako ULR adresa s položkou $row, jejímž titulkem je též proměnná $row. Jako poslední se načte pohled „footer“, čímž celá funkce končí. Výsledkem pak může být například toto:

Obrázek 5 - Možný výsledek skriptu [Vlastní zpracování]

4 Vlastní práce

4.1 Stávající řešení pro rozhraní KVM

Díky grafickému rozhraní, je ovládání jednodušší hlavně kvůli přehlednosti a nepotřeby zapamatování a zadávání příkazů skrze příkazovou řádku. [24] Některá z rozhraní jsou určena pouze pro společnosti, která práva k němu vlastní a využívají ho pro své účely. Jiná jsou s komerční licencí nebo k dispozici pro použití zdarma. Při výběru distribuce s komerční nebo veřejnou licencí závisí na uživateli, jaké funkce požaduje, jakou platformu preferuje nebo používá. I když základ všech distribucí je velmi podobný, liší se zejména v live podpoře, kdy komerční licence jí často disponují naopak veřejné distribuce jí podporují jen zřídka, ne-li vůbec. [11]

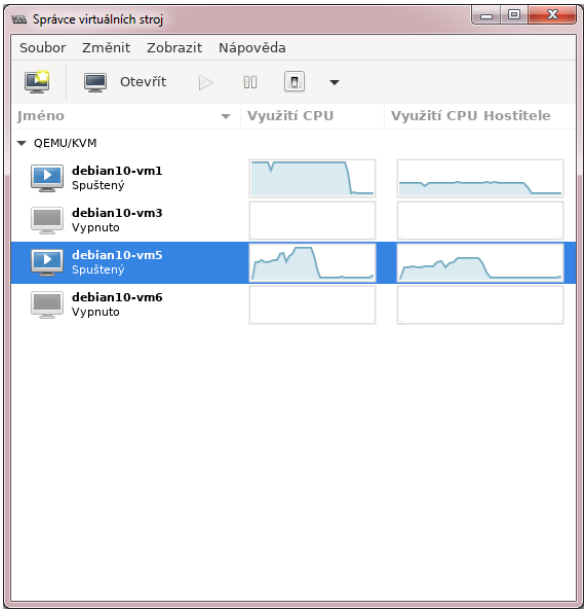
Další stránky popisují některé z nástrojů pro virtualizaci, dostupných pro platformu KVM. Popisují jej z několika základních hledisek. Přesněji název, základní informace, přístup k rozhraní, pro jaké platformy je dostupný, pod jakou licencí je k dispozici a základní průchod aplikací u jedné verze s webovým, desktopovým a textovým rozhraním (hlavní obrazovka, seznam virtuálních strojů, vytvoření virtuálního stroje a připojení k již existujícímu, základní ovládání stroje a přehled základních informací o stroji nebo možnost přidání hardwaru) a zhodnocení rozhraní včetně přívětivosti k uživatelům. V případě že nebudou v průchodu aplikací k dispozici obrázky poskytované autorem programu, tuto část neposkytuje, nebo není pod volně veřejnou licencí, nebude část nebo celou tuto sekci obsahovat. Na konci podkapitoly se nachází tabulka, která obsahuje souhrn požadavků jednotlivých nástrojů, případně jejich cena.

4.1.1 Virtual machine manager (verze 2.0.0)

Desktopové uživatelské rozhraní, využívající zkratku virt-manager. Sloužící pro správu virtuálních strojů, za pomoci libvirt knihovny. Aplikace je stylizována do průvodců, kteří pomáhají uživatele směřovat k požadovanému cíli. K této aplikace přistupuje skrze terminál nebo příkazový řádek s rozšířením pro zobrazení aplikací s grafickým rozhraním, příkazy virt-příkaz, nebo pomocí aplikace na desktopu. Primárně se zaměřuje na stroje s KVM, lze s ním však ovládat i Xen virtualizaci. Je dostupný pod licencí GNU GPL 2, přibližně

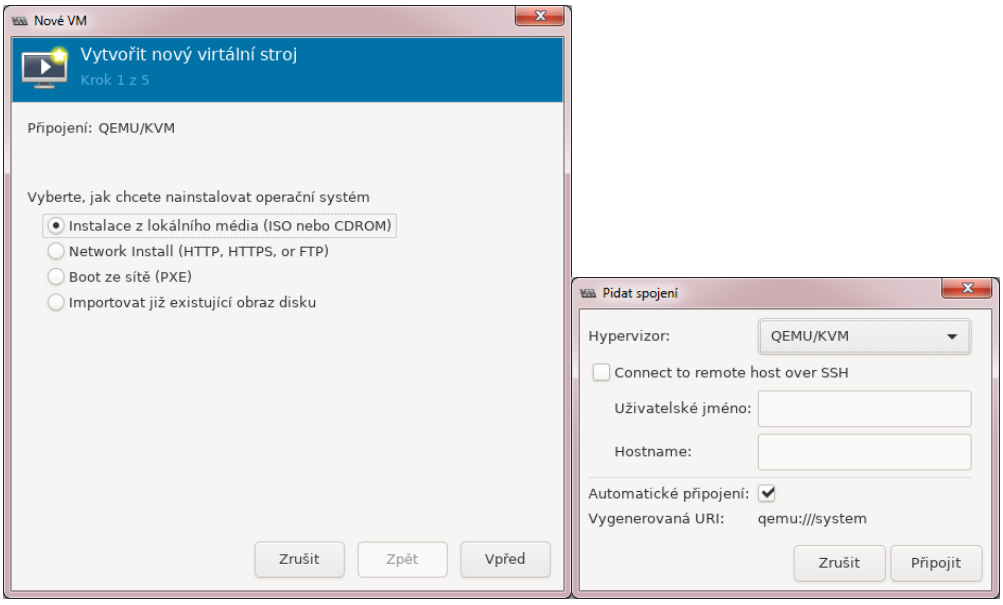
přeloženo jako všeobecná veřejná licence. Ta zaručuje, že jakékoliv upravené verze budou také svobodným softwarem. [25]

Průchod rozhraním



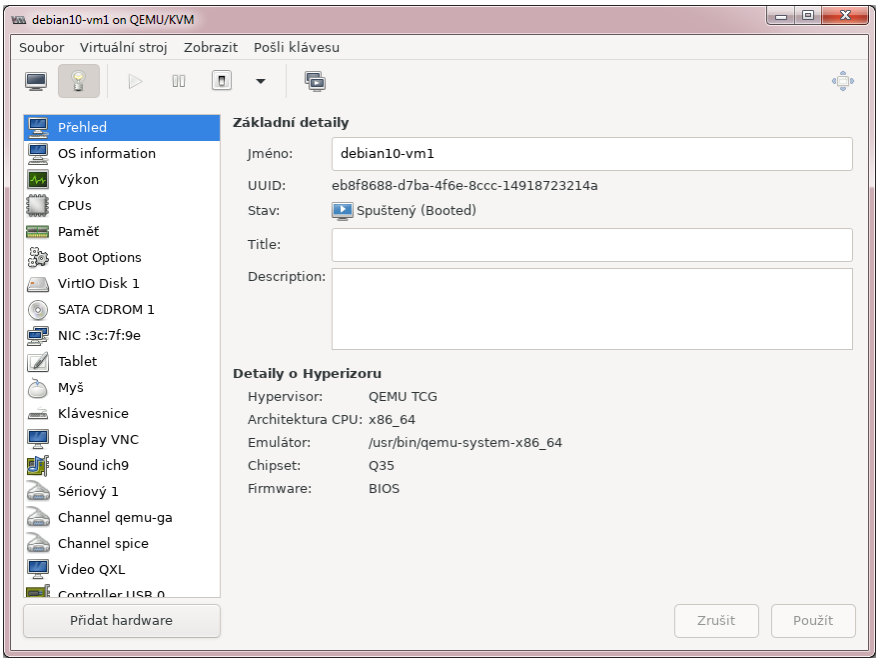
Obrázek 6 - Virt-manager - Seznam VM, úvodní stránka a základní ovládání [25]

Zde je vidět seznam strojů se základními grafy o využití cpu, názvy strojů a jejich stavu. Též zde jsou ovládací prvky na vrchní liště, jako zapnutí, vypnutí, pozastavení, vytvoření či připojení ke stroji a zobrazení terminálu.



Obrázek 7 - Virt-manager - Vytvoření stroje a připojení ke stroji [25]

Na levé straně je vytvoření virtuálního stroje. K dispozici jsou možnosti pro jeho vytvoření, kde nejčastějšími jsou instalace z lokálního média a importování existujícího obrazu disku (primárně kvůli funkcím KVM). Na straně druhé je vidět možnost několikanásobného připojení jak na lokální, tak na vzdálené stroje.



Obrázek 8 - Virt-manager - Přehled informací o VM, přidání hardwaru [25]

Obrázek obsahuje jak základní informace o stroji, tak i informace o různém vybavení počítače, které je možné upravovat stisknutím na danou položku v seznamu. Pod seznamem se nachází přidání hardwaru.

Zhodnocení: Virt-manager se hodí jak pro první kontakt s virtualizačním rozhraním, tak i pro pokročilejší správu. Příjemným bonusem je viditelnost seznamu virtuálních strojů a grafů, spolu s možností je ovládat přímo na úvodní stránce. Další výhodou pro uživatele je minimalizace rozhraní, tudíž není potřeba projíždět několik oken, pro získání požadovaných informací. Zaostává však v konfiguraci stroje, která se mezi synonymy pro konfiguraci špatně hledá. Navíc obsahuje několik konfiguračních oken, které jsou rozlišovány v názvu jen minimálně a každé poskytuje jiné možnosti. Ve verzi pro příkazový řádek, se v několika případech stává, že se rozhraní zasekne nebo neumožňuje provádět některé akce, které

normálně fungují. To však může být způsobeno zobrazovacím rozšířením nebo výkonem používaného stroje.

4.1.2 **Kvm-wrapper**

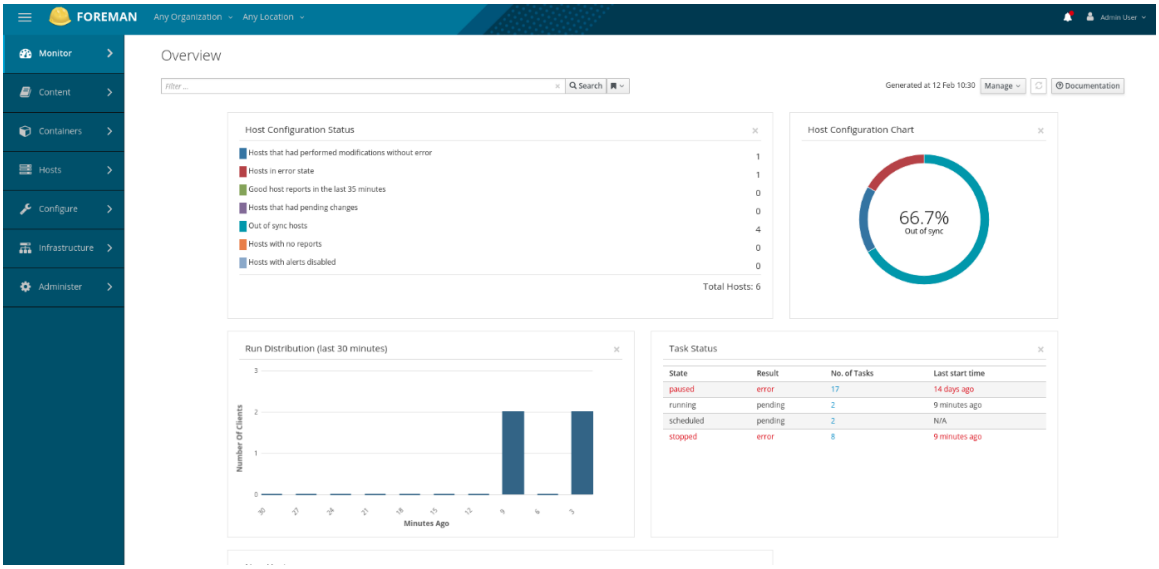
Jednoduché rozhraní s nízkou náročností na paměť. Kvm-wrapper je psán zcela v shellu a navíc se dá velmi snadno upravovat. Není nějak značně upravován aktualizacemi, navíc prozatím není plánován jeho obsáhlejší vývoj nebo častější aktualizace. Jelikož je psán pouze v shellu, je k dispozici pouze pro rozhraní příkazového řádku. Dále je dle názvu patrné, že podporuje pouze platformu KVM. K dispozici je pod licencí WTFPL (v2). Tato licence dovoluje software využívat zcela jakýmkoliv způsobem, což je patrné ze zkratky. Tu však nebudu překládat z důvodu, že bude dokument pročítán na akademické půdě a její podrobný popis je k dispozici ve zdrojích této práce. [26]

Zhodnocení: Příkazová řádka není nejpřívětivějším rozhraním, a to platí i pro toto rozhraní. Sice umožňuje některé pokročilé funkce, avšak se nehodí jako rozhraní pro udržování stálého dohledu nad virtuálními stroji a ani efektivní práci s nimi. Též nelze vytvořit relevantní průchod aplikací.

4.1.3 **Foreman (verze 1.24.2)**

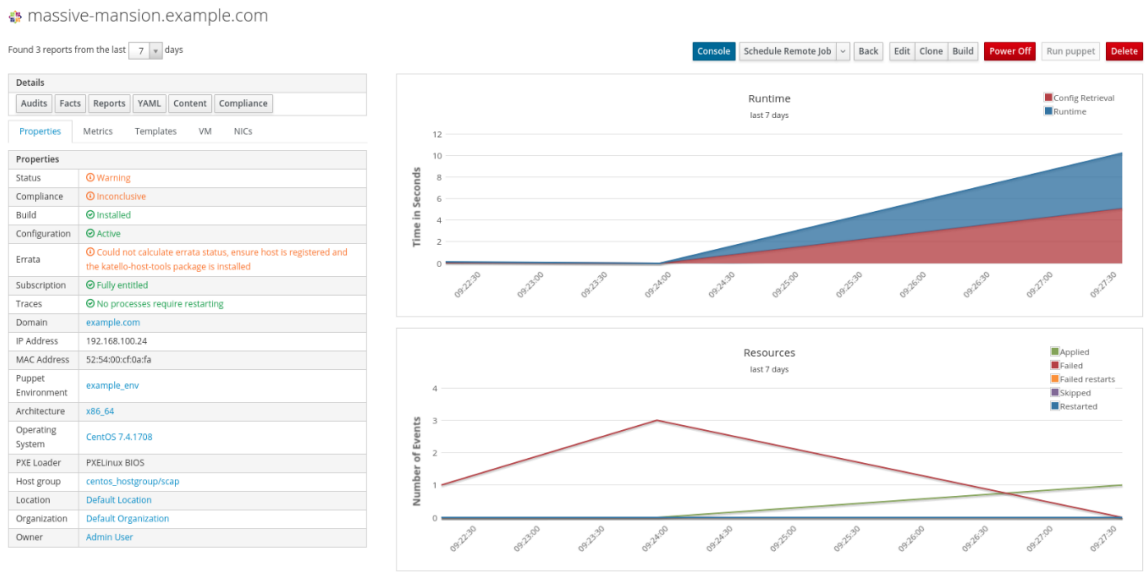
Open-source projekt, pro správu virtuálních strojů od vzniku po jeho zánik. Jedná se o rozhraní s podporou od vývojářů a ovládá se intuitivně. Lze s jeho pomocí automatizovat často spouštěné úlohy, rychle nasazovat aplikace a spravovat jednoduše daný stroj nebo například cloud. Rozhraní je ovládáno přes webové rozhraní nebo omezeně přes příkazovou řádku a je dostupný pro platformy KVM, VMware, oVirt a další. K dispozici je pod licencí GPL 3, jenž je rozšířením verze GPL 2. [27]

Průchod rozhraním



Obrázek 9 - Foreman - Hlavní strana [27]

Na levé straně se nachází základní ovládací panel rozhraní. Jako další například základní informace o celé virtualizaci a 2 grafy. Neobvyklou věcí je vyhledávání, tu většina rozhraní totiž vůbec nepodporuje.



Obrázek 10 - Foreman - Informace o VM [27]

Na levé straně jsou čitelné základní informace o stroji, což je pro uživatele přívětivé, například název stroje je umístěn na místo, kde vhodně poukazuje na místo, kde se uživatel

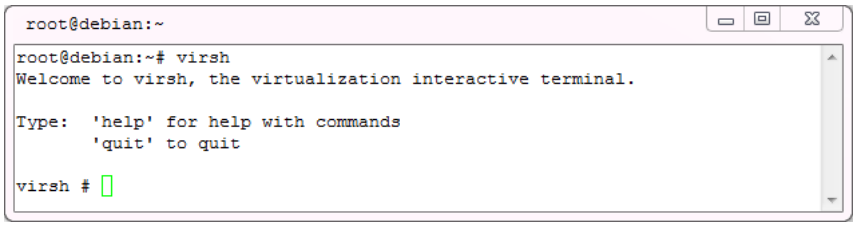
nachází. Pravá strana se věnuje základním možnostem jako úprava, spuštění, smazání stroje a dalšího. Dále jsou tu grafy, které jsou čitelné.

Zhodnocení: Toto rozhraní je vhodným adeptem pro použití mezi profesionály hlavně díky možnosti využít mnoha platforem pro virtualizaci a není potřeba instalovat jakoukoliv aplikaci. Je značně uživatelsky přívětivé a dodržuje některé z nepsaných standardů jako například jediné menu, které je orientované na svislé pozici vlevo či horizontální nahoře. Informování uživatele, kde se právě nachází ať už zvýrazněním v menu nebo značně viditelným názvem v obsahu stránky. Důležité informace se nacházejí v levé horní částí, případně pokračují po levé straně dolu a méně důležité informace jsou na pravé straně nebo vespod a jsou rozděleny no viditelně oddělených bloků. Jediné, co se nejspíše dá vytknout je umístění ovládacích tlačítek v horní části napravo.

4.1.4 Virsh

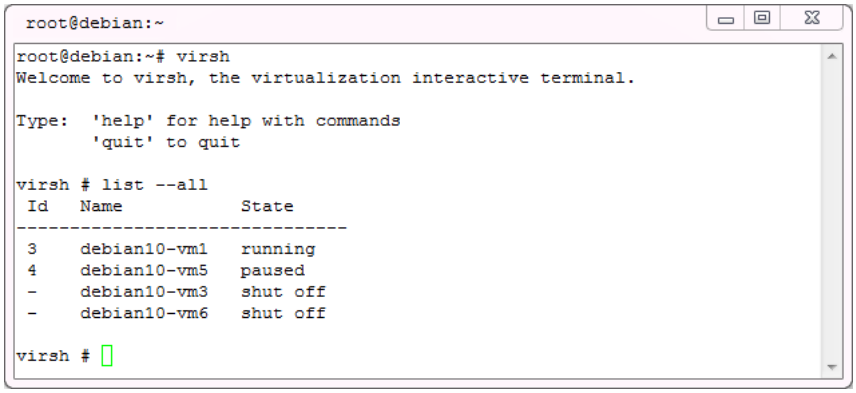
Jeden z rozhraní pro ovládání virtuálních strojů, založený na shellu. Slouží k základnímu ovládání domén, je k dispozici pouze pro rozhraní příkazového řádku a slouží primárně pro ovládání KVM a QEMU strojů. I když se jedná o příkazový řádek, obsahuje podrobnou dokumentaci a snaží se vytvářet prostředí pro uživatele přívětivé. Má licenci LGPL v2+, jenž je zjednodušenou verzí GPL. Nezaručuje záruku pro tvorbu ani na nic dalšího. [28]

Průchod rozhraním



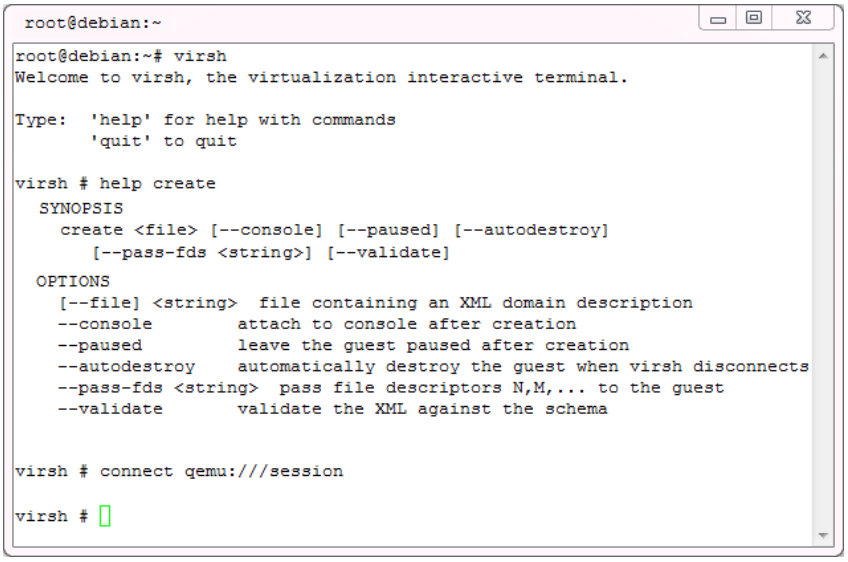
Obrázek 11 - Virsh - Úvodní obrazovka [28]

Uživatele uvítá hláška a dva první příkazy. Již nyní lze Virsh rozhraní plně využívat a zadávat pokročilé příkazy. Příkazem help <příkaz>, lze vypsat informace o příkazu, včetně syntaxe a možných nastavení pro tento příkaz. Příkazem quit se pak z rozhraní odchází.



Obrázek 12 - Virsh - Seznam VM [28]

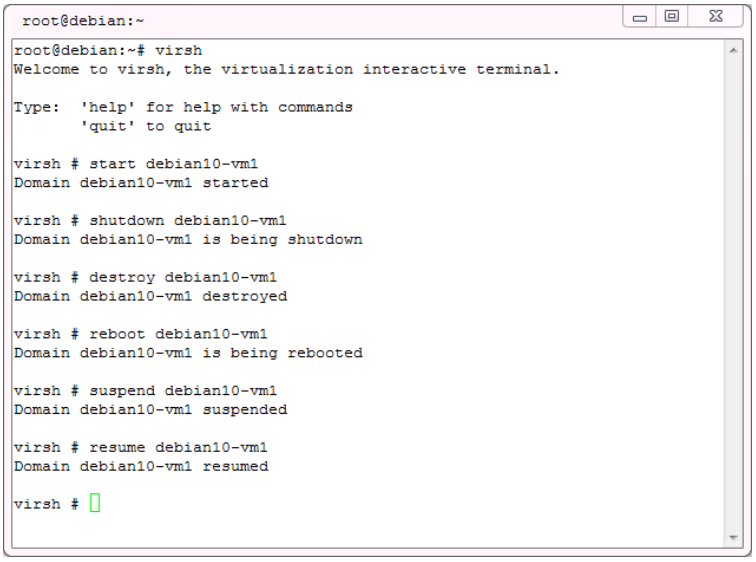
Jedná se o jednoduchý výpis všech virtuálních strojů, který obsahuje identifikační číslo, název a stav. Z ukázky je patrné že stroje debian10-vm3 a 6, jsou vypnuté. Stroj s Id 4 je pozastavený a stroj s Id 3 běží. Pokud je za příkazem list napsáno --all, dojde k výpisu všech strojů. I tento příkaz má další možné nastavení jako například --name, pouze pro výpis jmen domén.



Obrázek 13 - Virsh - Vytvoření VM a připojení k existujícímu stroji [28]

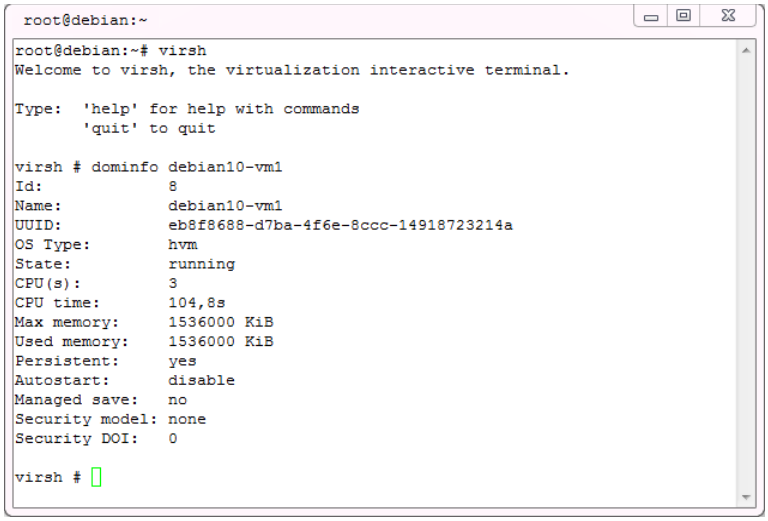
Vytvoření stroje je pomocí Virsh rozhraní značně rozsáhlé a umožňují nastavovat širokou škálu možností. K tomuto však slouží dokumentace, která je dostupná na stránkách projektu a je též zařazena do zdrojů práce. Pro zkrácený popis možností je využito příkazu help

ukazující synopsi a nastavení příkazu create. V předposledním řádku je patrné připojení na lokální stroje. Často se však používá automatické připojování.



Obrázek 14 - Virsh - Základní ovládání stroje [28]

Příkazů pro základní správu je mnoho a zde jsou zmíněny ty nejčastěji používané spolu s ukázkou, jak na ně rozhraní reaguje. Například při zadání příkazu suspend debian10-vm1, dojde k pozastavení domény a Virsh o tom uživatele informuje. Příkazy zde nejsou dány logicky po sobě (což je patrné z příkazu reboot hned po příkazu, který doménu vynuceně vypne) a slouží především jako ukázka.



Obrázek 15 - Virsh - Přehled informací o VM [28]

Pro výpis základních informací neexistuje pouze příkaz dominfo, ale je mnoho dalších, které se liší dle rozsáhlosti, případně zaměření na oblast stroje nebo jeho formálního zpracování. Zde vidíme například stroj s identifikačním číslem 8, disponující třemi procesory a 1,5 GB RAM.

Zhodnocení: Příkazů virsh lze často používat i ve spojení s programy, které jsou dostupné na platformě Linux (například grep). Pro práci se stroji není potřeba rozhraní přímo spouštět a zadávat do něj příkazy, stačí před daný příkaz napsat virsh a prostředí si rozhraní samo otevře a zavře. Nejedná se o grafickou verzi, přes to je však vidět určitá snaha veškeré výstupy stylizovat, právě pro čitelnější výstup jak pro uživatele, tak i pro stroj. Možnost zadávat i pokročilé příkazy je rozhodně výhodou, to však nic nemění na tom, že se vše musí psát ručně, a i přes značnou dokumentaci, může uživatel často narazit na problém a nevědět si rady.

4.1.5 **Red Hat Virtualization (RHV)**

Virtualizační platforma umožňující fungovat na Linuxu a Windows. Je postaven na platformě od stejné společnosti, a to na Red Hat Enterprise Linux. Dovoluje používat nástroje pro správu zdrojů, procesů a poskytuje základ pro budoucí využití cloudu. Lze jej používat přes webové rozhraní s možností příkazové řádky. Podporuje spoustu platforem pro virtualizaci, ale zaměřuje se primárně na QEMU a KVM. Poskytuje vlastní dokumentaci, spolu s podporou tvůrce. Má vlastní komerční licenci. [5]

Zhodnocení: Především díky online podpoře a stabilitě společnosti, se jedná o kandidáta, který zaručuje některé funkce, což některý uživatel může vyžadovat. Je jednoduchý na ovládání a díky velmi obsáhlé dokumentaci od autorské společnosti, je jednoduché případné problémy či nedostatky snadno vyřešit. Výhodou jsou podporované platformy, tudíž je možné jej integrovat kamkoliv. Rozhraní je velmi podobné jako je Foreman. Obsahuje spoustu volitelných grafů, statistik, konfigurací a podobně. Nevýhodou je daná licence, ta se musí ročně platit a spolu s online podporou 24x7x365, vyjde na nemalé peníze, které mohou znamenat pro uživatele nežádoucí náklad.

4.1.6 **Porovnání zmíněných rozhraní**

Parametr	Virt-manager	kvm-wrapper	Foreman	Virsh	RHV
Typ rozhraní	Desktop	CLI	Web/CLI	CLI	Web/CLI
Zálohy strojů	Ano	Ano	Ano	Ano	Ano
Podporované platformy	KVM, Xen, Wmware, …	KVM	KVM, Wmware, oVirt, …	KVM, Xen, Wmware, …	KVM, Xen, Wmware, …
Podpora	Ano (komunita)	Ne	Ano (komunita)	Ano (obojí)	Ano (tvůrce)*
Licence	GPL v2	WTFPL	GPL v3	LGPL	Komerční
Cena	Zdarma	Zdarma	Zdarma	Zdarma	23 000 CZK**

Tabulka 2 - porovnání rozhraní Zpracováno dle: [5] [25] [26] [27] [28]

**Tvůrce poskytuje i Live podporu*
***Cena se určuje dle počtu procesorů a edice*

Z tabulky a zhodnocení jednotlivých rozhraní je patrné, že RHV neboli Red Hat Virtualization, je nejlepší volbou především pro uživatele, kteří potřebují rychlé, funkční a multi platformní řešení s výrazně rychlou podporou. Bez ohledu na uživatelskou přívětivost je rozhraní Virsh a Foreman na podobné úrovni, hlavně kvůli podpoře několika platforem a možností zadávat i pokročilé příkazy. Dalším je Virt-manager, ten je vhodný především pro začátečníky a dodává základní informace včetně grafu, bohužel však nepodporuje možnost používat pokročilé příkazy. Na poslední příčku se řadí kvm-wrapper a to hlavně proto, že podporuje pouze KVM platformu, neposkytuje podporu pro uživatele a je dostupný jen jako příkazová řádka.

4.2 Porovnání výkonnosti frameworků pro PHP

Test vypisuje pouze text „ahoj světe“. K frameworkům není připojena databáze, není dostupná šablona a debugging. Taktéž je nastaven na minimální nastavení. Je testováno na PHP verze 7.1.0 s Apache 2.4.18. Test slouží pro výběr nejvhodnějšího kandidátů na použití při tvorbě webových aplikací.

Framework	Požadavků za sekundu	Paměť (MB)
Bez frameworku	7 094	0,34
Codeigniter 3	2 245	0,38
Lumen 5.3	1 543	0,63
Zend 2.5	291	1,34
Laravel 5.3	331	1,53

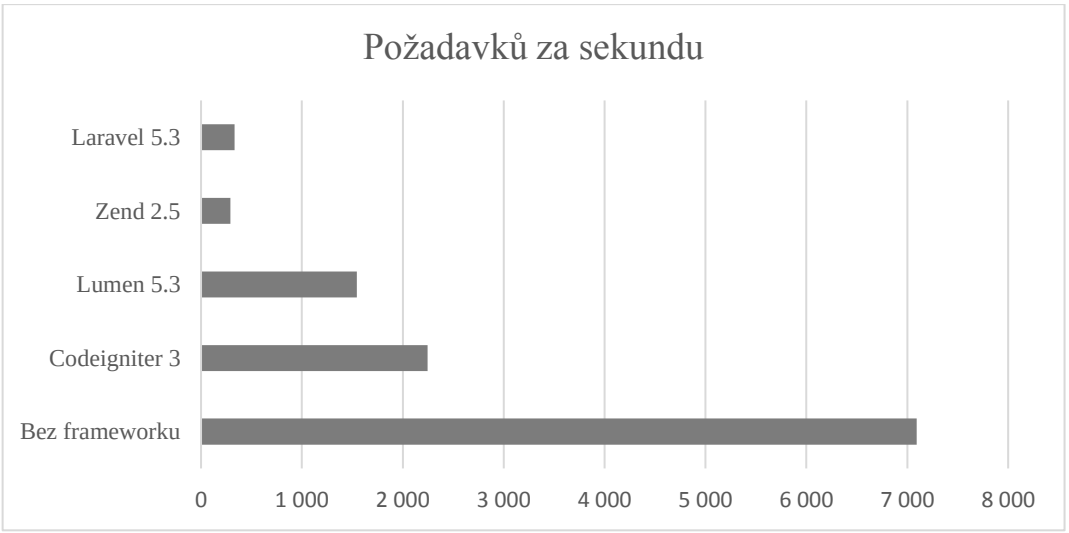
Tabulka 3 - Porovnání PHP frameworků Zpracováno dle: [29]

Popis tabulky:

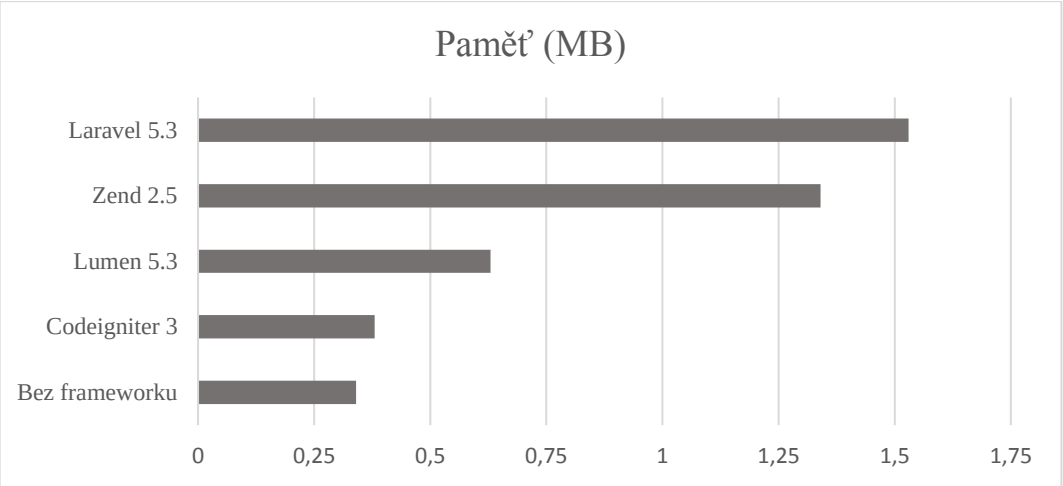
Požadavků za sekundu – kolik požadavků za sekundu dokáže framework zvládnout

Paměť – Kolik paměti daný framework vyžaduje během testu

Grafy:



Graf 1 - Požadavky za sekundu [29]



Graf 2 - Užívaná paměť [29]

Zhodnocení: Z porovnání grafů a hodnot v tabulce, lze jednoznačně vyvodit, že Codeigniter a Lumen jsou vhodnými kandidáty pro tvorbu aplikací, vzhledem k jejich rychlosti. Značným rozdílem pak mezi Codeigniterem a Lumenem je jejich dokumentace a podpora komunity, kde Codeigniter velmi značně vede. Je tedy často vybírán jako základní framework pro tvorbu projektů, a to i díky jednoduchému nastavení, otevřenosti a snadnosti se mu učit. [30]

4.3 Požadavky na rozhraní

Pro tvorbu slouží dané požadavky, jenž určují, jak rozhraní má vypadat a jaké funkce bude poskytovat. Jako zdroj těchto informací slouží získané informace z průzkumu stávajících rozhraní a znalosti získané od pedagogů. [24]

-Základní informace včetně požadavků na přístupném místě

-Možnost konfigurace rozhraní a vládání příkazů ručně

-Zobrazení seznamu a statistik hlavního stroje a VM včetně grafu vytíženosti VM

-Vytvoření, smazání, spuštění, pozastavení a vypnutí VM

-Jednoduché a čisté rozvržení prvků (není příliš komplexní)

-Poskytuje přímý a rychlý přístup k funkcím

-Rozhraní je snadno ovladatelné i při prvním kontaktu

4.4 Vytvoření administračního rozhraní

Vytvoření rozhraní má svá specifika. Obvykle se skládá z požadavků na rozhraní od uživatele, naplánování vytvoření, samostatné vytvoření a otestování. Žádostí je vytvoření administračního rozhraní, jenž bude mít grafické rozhraní, poběží na PHP a bude moci umožňovat základní ovládání strojů případně ovládání pomocí příkazů. [24]

Dle předchozích podkapitol, došlo k vybrání relevantních programů a knihoven. Bude se tedy jednat o webovou stránku běžící na PHP, proto bude na jeho úpravu použit PHPStorm verze 2019.2.3 od společnosti JetBrains. Ten umožnuje jednoduše upravovat webové stránky, procházet celkový projekt a zefektivňuje psaní kódu. Frameworkem pro běh webové stránky bude Codeigniter, především kvůli jeho rychlosti. Jako základ pro virtualizaci poslouží modul KVM. Ten je vybrán proto, že poskytuje nejvíce možností k ovládání stroje a je zdarma. K jeho ovládání bude využito funkcí programu Virsh, jenž dovoluje ovládat VM skrze CLI a díky jeho funkcím a licenci je výhodné ho využít.

Některé z využitých programů, mají pro svůj běh na stroji požadavky. Bez nich může dojít k selhání rozhraní nebo jeho částečné nefunkčnosti. Doporučené hardwarové a softwarové požadavky tedy jsou: [23] [30]

-6 GB místa na disku + další pro VM
-2 GB RAM + další pro VM
-Jedno jádro nebo vlákno procesoru s podporou virtualizace + další pro VM
-MySQL verze 5.1 a vyšší
-Apache verze 2.4.33 a vyšší
-PHP verze 7.0.29 a vyšší
-KVM (balíčky qemu-kvm, libvirt-daemon-system, bridge-utils, libquestfs-tools, genisoimage, virtinst a libsoinfo-bin)
-Správně nakonfigurovaná síť pro přístup k internetu
-Přístup k právům superuživatele na systému Linux

4.5 Průběh vytvoření rozhraní

V prvotní části bylo vybrání vhodných programů a platforem pro tvorbu rozhraní pro KVM modul. Nejdříve jsem prošel několik různých platforem a programů, abych určil, co bude vhodné vybrat. Pro běh rozhraní je to tedy Linux, KVM, Virsh, PHP, Apache, Codeigniter a MySQL. Pro úpravu a testování pak pomohl program PHPStorm, pro drobné úpravy PSPad a na virtualizaci stroj, s Debian Linuxem verze 10.

Po výběru přišel čas na především samostudium a prozkoumání možností vybraných programů. K tomu sloužila dokumentace od autorů, ta je k nahlédnutí v sekci Zdroje. Z části došlo i průzkumu častých chyb a problémů v komunitní sekci, která se však nevěnovala pouze virtualizaci.

V další části došlo na výpis funkcí a požadavků na dané rozhraní spolu se základním návrhem rozložení jednotlivých prvků v rozhraní a celkového vzhledu, pro zvýšení uživatelské přívětivosti, jako například barvy, velikost textů a fonty, vzhled tlačítek a další. Pro tuto část výrazně pomohli znalosti, nabyté z vlastní praxe a z části zkušenosti, získané od pedagogů z předmětů, vztahujících se k oboru IT. Jakožto smysluplný program, musí tento program pomáhat v ulehčení funkcí, jež jsou uživatelsky náročné, případně škodí v průběhu práce. Uživatel komunikuje s počítačem pomocí grafických prvků a není potřeba určovat, jak má software pracovat. Celé rozhraní zároveň nesmí uživatele ovlivňovat na emocionální úrovni. Musí být jednoduché, prvky musí skutečně dělat to co jednoduše popisují a uživatel by měl hned na první pohled pochopit, co prvek dělá. [24]

Horizontální pozice menu byla vybrána především kvůli kognitivním funkcím uživatele, tedy že jakoukoliv aplikaci čte jako knihu z leva doprava, od shora dolů. Software je tvořený pomocí grafického rozhraní. Všechny další prvky se nachází hned pod menu, kde jsou strukturovány do formy knihy. Tedy že se důležité informace nachází z leva doprava, případně na levé straně jsou zásadní informace a na pravé informace doplňující. Pro celé rozhraní bohatě postačují 3 barvy. Menu a zápatí mají šedozelenou barvu (kombinace barvy povzbuzující a klidné). Pozadí zastupuje barva bílá, právě pro její čistotu. Font má pevnou a

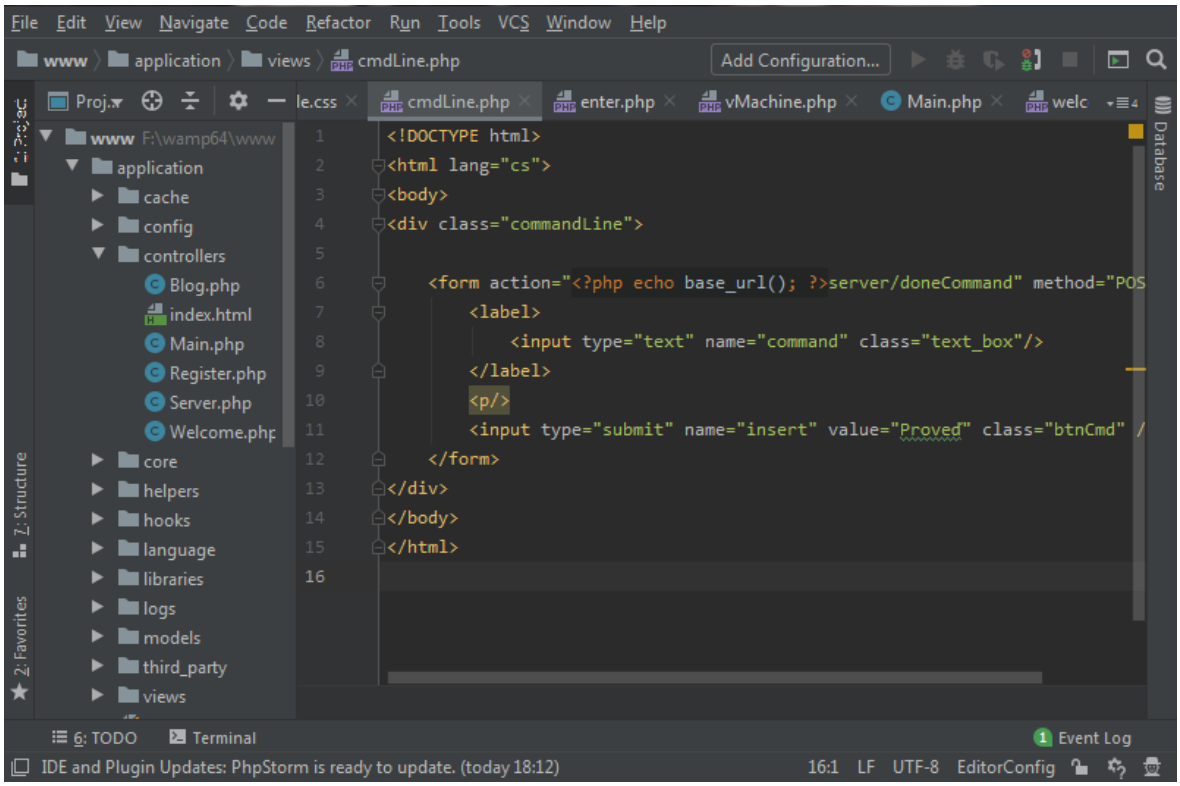
vážnou barvu, přesněji černou. Ta se především hodí pro veškerý text, jelikož je na pozadí dostatečně kontrastní, a proto se dobře čte. [24]

Dále následovalo vytvoření daného rozhraní pomocí PHP a jeho frameworku Codeigniter, což díky předchozímu samostudiu šlo snadno. Nejdříve došlo k vytvoření základních funkcí jako je přihlašování, zobrazování záhlaví a patičky. Následovalo vytvoření pohledů pro zobrazení daných požadavků a s tím spjaté vytvoření funkcí. Nesmím opomenout, že došlo ke zjednodušení správy pravomocí, jelikož se nejednalo o vytvoření rozhraní, které dovoluje vytvářet uživatele a jim dovolovat vytvářet VM. Jedná se o vytvoření rozhraní pro jeden či více strojů, ze kterého lze vytvářet VM a ty dále případně nabízet uživatelům. Správa uživatelů, kteří mohou přistupovat k VM, je tedy spravován pomocí Linuxu. Vytvořený projekt je však zabezpečen uživatelským jménem a heslem, spolu s možností vytvořit uživatele pomocí databáze, který do něj může vstupovat.

Posledním krokem bylo otestování funkčnosti rozhraní. Odzkoušení, zda se uživatel nemůže dostat kam nemá. Úprava a zjednodušení kódu pro zvýšení výkonu a smazání duplicit. Následně pak kompletace, nahrání potřebných souborů na internetovou službu github a to pro umožnění jednoduché instalace.

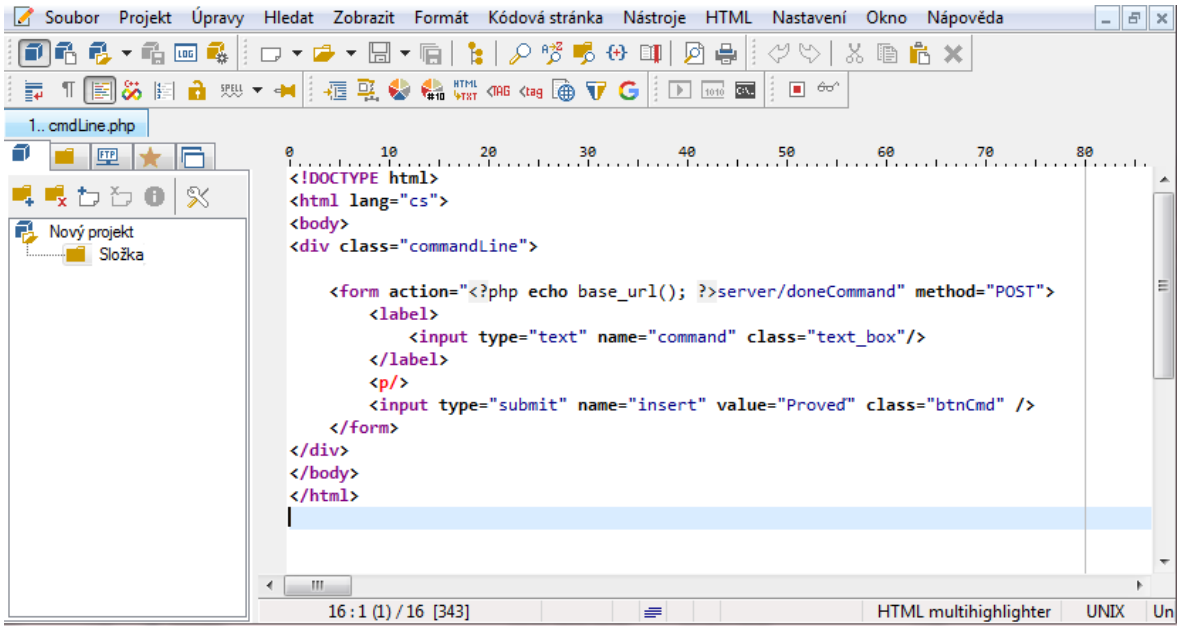
4.6 PhpStorm a PSPad

Jedná se o programy určené pro úpravu kódu. PhpStorm byl nejvíce používán právě pro tvorbu kódu. Je vhodným nástrojem pro programátory, především kvůli nápovědám, kontrole kódu proti chybám a automatickému doplňování psaného kódu. Je často používán mezi programátory. Je určen pro úpravu PHP, HTML a JavaScriptu. Lze též využívat některé z modulů, které jsou dostupné pro PhpStorm, rozšiřující jeho funkčnost. K dispozici je zde i průzkumník projektu, jenž velmi pomáhá v přehlednosti a průchodu v projektu. Pro jeho používání je potřeba zakoupit licence, případně je k dispozici zdarma pro studenty, případně 30denní verze na vyzkoušení. [31]



Obrázek 16 - Ukázka PhpStormu [31]

PSPad je freewarový editor nejen kódu, ale i textu. Vyvíjen je českým programátorem Janem Fialou. Nejedná se o nejprofesionálnější nástroj, ale i přes to je hojně využíván pro úpravu kódu. Pro tuto práci sloužil jen na drobné úpravy případně oddělení kódu pro přehlednost. Sice podporuje spoustu jazyků, není však tak vhodný jako PhpStorm. [32]



Obrázek 17 - Ukázka PSPadu [32]

5 Výsledek

Po celkové práci vyšlo administrační rozhraní, vhodné pro reálné použití.

5.1 Kód

Stylizování kódu vyšlo především z architektury M-V-C, pravidel pro přehlednost kódu a okomentování veškerých funkcí. Dále byl kód rozdělen na dvě části: main a server. Část main se věnuje základním funkcím spojených pouze se zobrazováním základních informací, přihlašování a zjištění statistik. Část server slouží primárně funkcím, které jsou spojené se správou serveru.

Ukázka kódu

```
function login_validation()
{
    $this->load->library('form_validation');
    $this->form_validation->set_rules('username', 'Username', 'required');
    $this->form_validation->set_rules('password', 'Password', 'required');
    if($this->form_validation->run())
    {
        $username = $this->input->post('username');
        $password = $this->input->post('password');
        $this->load->model('main_model');
        if($this->main_model->can_login($username, $password))
        {
            $session_data = array(
                'username' => $username
                );
            $this->session->set_userdata($session_data);
            redirect(base_url() . 'main/enter');
            $this->load->view("header");
            $this->load->view("footer");
        }
        else
        {
            $this->session->set_flashdata
    ('error', 'Špatné jméno nebo heslo!');
            redirect(base_url() . 'main/login');
        }
    }
    else
    {
        $this->login();
    }
}
```

Obrázek 18 - Vlastní kód - Přihlášení

Tato část kódu patří přihlašování. Pokud je tato funkce volána a jsou splněny podmínky, tedy že uživatel zadá jméno a heslo, funkce pokračuje voláním modelu, zda tato data jsou obsaženy v databázi v tabulce users. Když v tabulce obsaženy jsou, dojde k načtení úvodní stránky. Přesněji přesměrování na úvodní stránku a načtení pohledu „header“ a „footer“. V případě že ne, objeví se hláška „Špatné jméno nebo heslo!“.

```
public function volani($vstup)
{   //Provedení volaného příkazu
    $connection = ssh2_connect($this->ipServeru, $this->portServeru);
    ssh2_auth_password($connection, $this->userName,$this->userPasswd);
    $vystup = ssh2_exec($connection, $vstup);
    stream_set_blocking($vystup, true);
    return $vystup;
}
```

Obrázek 19 - Vlastní kód - Volání příkazu

Úkolem této funkce je volání příkazu virsh skrze šifrované připojení na Linuxový stroj. Příkaz je vložen pomocí proměnné $vstup. Nejdříve dojde k připojení k serveru, poté se pomocí příkazu ssh2_exec() příkaz provede a výsledek příkazu se vloží do proměnné $vystup. Tato proměnná pak vyjde jako výstup funkce. Tato funkce je volána například tímto způsobem:

```
$this->cmdTotalRAM = "free mem | grep 'Mem:' | awk '{print $2}'";

$provedPrikaz=$this->volani("$this->cmdTotalRAM");
$this->output->append_output(stream_get_contents($provedPrikaz));
```

Obrázek 20 - Vlastní kód - Volání funkce

Do proměnné cmdTotalRAM je vložen příkaz, vypisující celkovou paměť RAM hlavního stroje. Tento příkaz je pak zpracován právě onou funkcí volani(). Výsledek této funkce je pak vypsán na obrazovku. Podobně to funguje u všech použitých příkazů.

```php
public function showMachine($machine)
{
    $this->vmName = $machine;
    $this->load->view('header');
    $this->load->helper('html');
    link_tag(base_url() . 'css/style.css');
    $this->load->view('vMachine');
    $this->loadMachineInfo($machine);
    $this->load->view('footer');
}
```

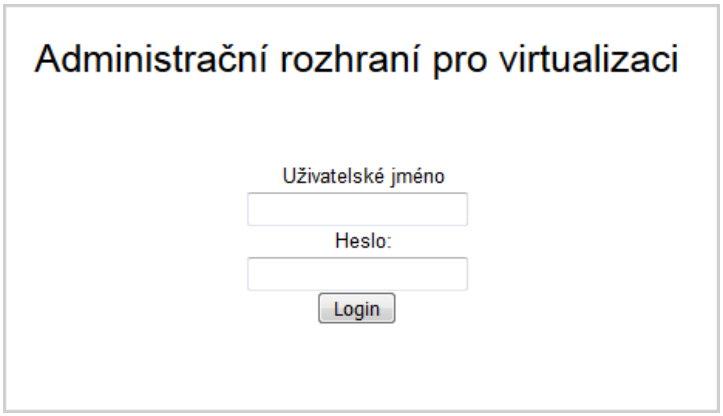
Obrázek 21 - Vlastní kód - Načtení informací o VM

Zde se jedná o načtení základních informací o virtuálním stroji, vygenerování ovládacích prvků a grafu o vytíženosti stroje. Celkově tedy internetové stránky o nich. Přesněji tato funkce převezme proměnnou machine (název načteného stroje). Načte se pohled „header“, pomocník „html“ a ke stránce se připojí soubor s kaskádovými styly pro určení vzhledu stránky. Následuje načtení pohledu „vMachine“, jenž je určen právě pro virtuální stroje. Následuje volání funkce, která zajistí informace o tomto stroji. Nakonec je načten pohled „footer“, jenž je poslední část pro kompletaci stránky.

5.2 Rozhraní

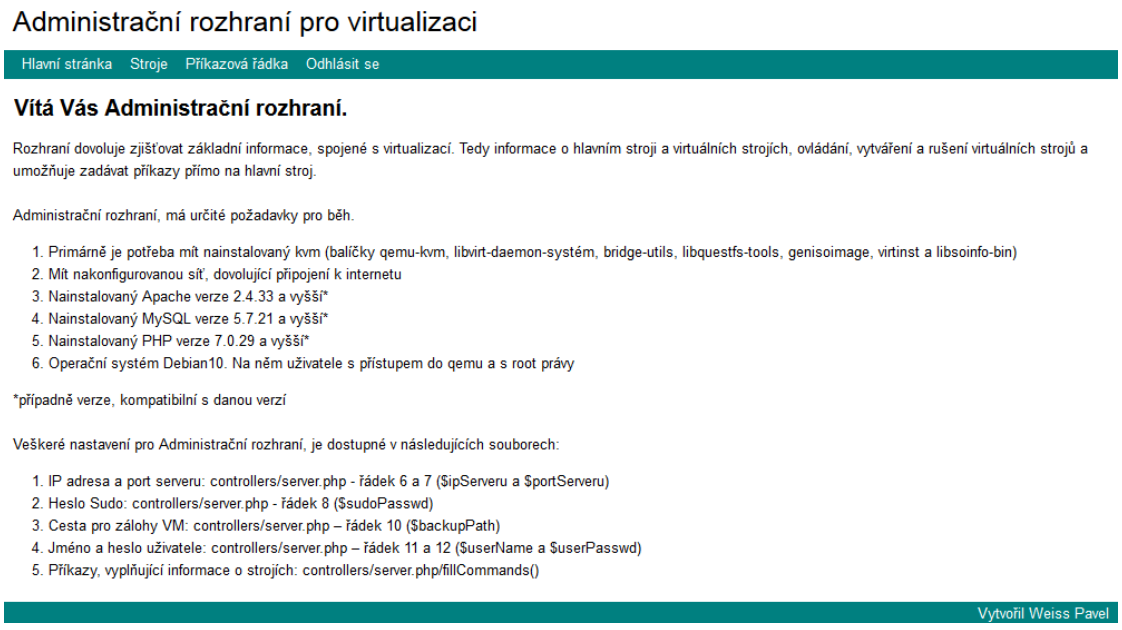
Při vytvoření uživatelského rozhraní bylo využito především vlastních zkušeností v této oblasti. Je zde brán zřetel na jednoduchost, optimalizaci, přístupnost a přehlednost. Při vzniku a během vytváření docházelo několikrát ke kontrole, zda je daný prvek funkční a zda se lze řídit intuicí. I přes zjevnou jednoduchost, poskytuje podobné funkce, jako rozhraní z průzkumu. Navíc podporuje i příkazovou řádku, a to především pro využití pokročilých funkcí.

Ukázka rozhraní



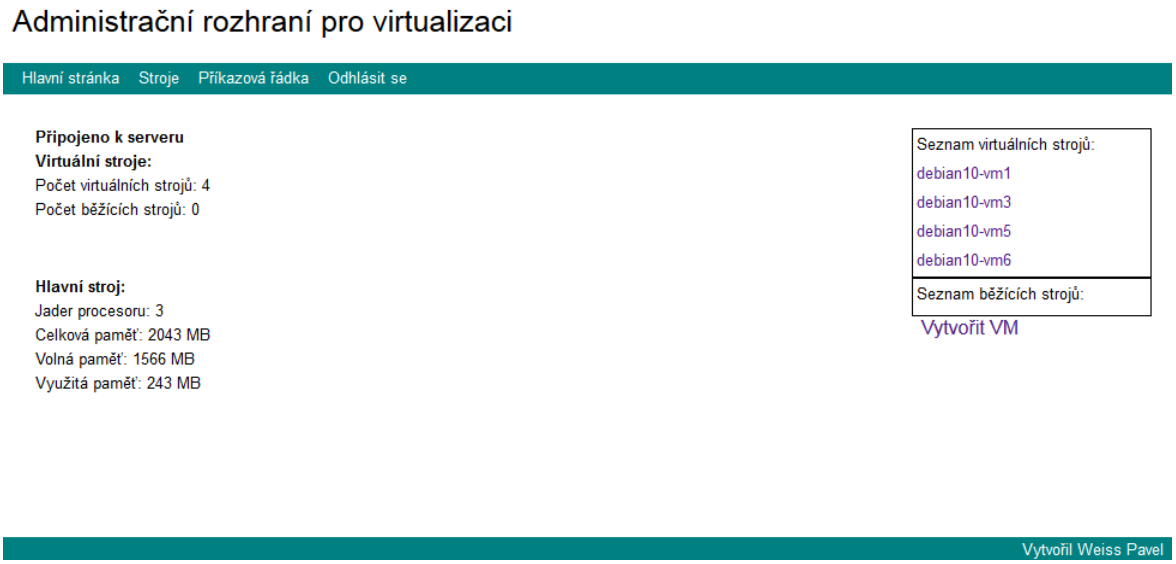
Obrázek 22 - Výsledné rozhraní - Přihlášení

Zde je vidět jednoduché přihlášení do rozhraní, jenž je viditelné při prvním spuštění stránky. V této části nebylo potřeba doplňovat jakékoliv další informace nebo grafické prvky. To hlavně proto, že mohou značně ovlivňovat uživatele nepodstatnými informacemi.



Obrázek 23 - Výsledné rozhraní - Úvodní strana

Úvodní stránka při přihlášení poskytuje část informací, které jsou součástí přílohy v souboru čtiMě.txt. Ve spodní části se pak vyskytují informace o možnostech konfigurace a základního nastavení serveru. Pro zvýšení přehlednosti byl vybrán jednotný font a minimum barevných prvků. Uživatel ví, kde se nachází. V horní části je vidět menu, kde je odkaz na hlavní/úvodní stránku, stroje, příkazovou řádku a možnost odhlášení.



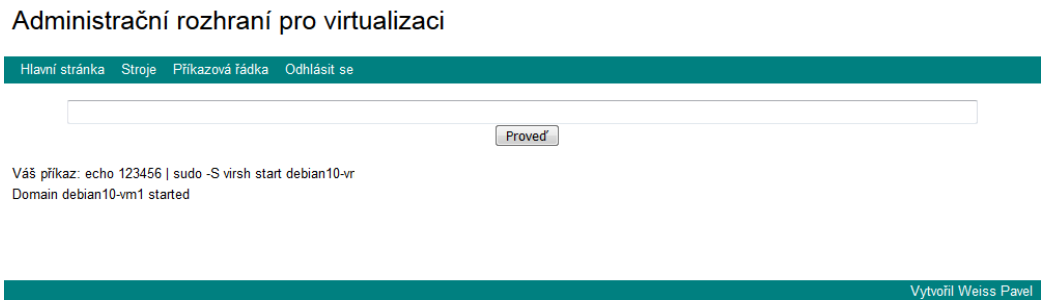
Obrázek 24 - Výsledné rozhraní - Seznam strojů a informace o hlavním stroji

Pod odkazem Stroje, lze nalézt základní informace o hlavním stroji a počet všech a běžících virtuálních strojů. V pravé části se nachází tabulka se seznamem všech VM a všech VM, které jsou aktivní. Hned pod seznamem se nachází možnost vytvoření nového virtuálního stroje.

Obrázek 25 - Výsledné rozhraní - Informace o VM

Na této stránce se nachází informace o VM. V tomto případě se jedná o debian10-vm1. Pod menu je viditelné základní ovládání VM. Levá strana je určena pro informace o VM.

Na pravé straně se nachází graf, jenž ukazuje grafickou informaci o vytíženosti procesoru v procentech. Hodnota může nabývat i hodnot větších, než je 100 %, což signalizuje, že do vytíženosti procesoru na 100 % dochází k zasekávání informací na disku.



Obrázek 26 - Výsledné rozhraní - Příkazová řádka

Jedná se o jednoduchý příkazový řádek, jenž vypíše odpověď Linuxu spolu s příkazem, který byl zadán. Slouží především pro ovládání virtuálních strojů, lze přes ní však používat i ostatní příkazy.

5.3 Zhodnocení rozhraní a implementace

Požadavky byly splněny, rozhraní umí základní ovládání VM. Umožňuje zadávání příkazů ručně, disponuje informacemi ohledně hlavního stroje a VM spolu s grafem pro VM. Výhodou je především možnost kód upravovat pro vlastní potřeby, spolu s přidáváním prvků. Nevyžaduje náročné požadavky na hardware a programy které vyžaduje, bývají velmi často součástí serveru. Rozhraní je především určeno pro osoby, které jsou zdatné v oboru informačních technologií a nejlépe se zaměřením na virtualizaci, Linux, popřípadě PHP, pro úpravy rozhraní. Vhodný je však i pro začátečníky.

Implementace je v celku jednoduchá. Stačí pouze ke splnění požadavků, následované stažení rozhraní dokončení konfigurace, jenž je dostupná přes cestu controllers/server.php.

IP adresa a port serveru: řádek 6 a 7 ($ipServeru, $portServeru)
Heslo Sudo: řádek 8 ($sudoPasswd)
Cesta pro zálohy VM: řádek 10($backupPath)
Jméno a heslo uživatele: řádek 11 a 12 ($userName, $userPasswd)
Příkazy pro vyplnění informací o strojích: funkce fillCommands()

Ostatní konfigurace rozhraní jsou dostupné dle konfigurací codeigniteru, jenž si každý uživatel nastavuje individuálně, vzhledem k použitým technologiím. Po konfiguraci stačí vytvořit v databázi tabulku users, kde se budou nacházet 2 sloupce. Jedním je sloupec „username“ a druhým „password“. Následovně stačí vytvořit uživatele a přihlásit se.

6 Závěr

Primárním cílem této bakalářské práce bylo vytvořit administrační rozhraní a jeho vhodná implementace, které předcházel průzkum technologií pro dané řešení.

Dle průzkumu operačních systémů pro virtualizaci, nástrojů pro virtualizaci a možností tvorby webového rozhraní, došlo k zaměření na operační systém Linux s podporou modulu KVM. Pro tvorbu webového rozhraní následovně posloužil framework CodeIgniter, jenž dle výsledků testu byl nejvýkonnější. Nejvhodnějším kandidátem pro virtualizační rozhraní je RHV. Jeho nevýhodou je však nepřívětivost v oblasti vlastního rozšíření a cena. V případě, že vývojáři značně zapracují na vlastních rozhraních, bude vhodné uvažovat nad rozšířením vytvořeného rozhraní. Výsledkem práce je však rozhraní, které podporuje možnosti vlastního rozšíření jakýmkoliv uživatelem, jenž je v této oblasti dostatečně vzdělán. A to i pro obsáhnutí i ostatních virtualizačních technologií.

Veškeré použité technologie disponují vhodnou dokumentací pro tvorbu. Ta sloužila především ke studiu a získání informací v oblasti virtualizace a tvorbě aplikací. Zajisté si každá ze zmíněných technologií a programů zaslouží vlastní pozornost. Uživatel si především vybírá vhodné možnosti právě pro své požadavky.

Toto téma mě v mnohých oblastech poučilo. Dokumentace tvořená buď autory nebo komunitou, je vhodná pro začátečníky a pokročilé programátory. Dokumentace však nebyla jediným zdrojem. K potřebám tvorby jsem musel mnoho informací nalézt na internetu a v online publikacích. Tím jsem se naučil vyhledávat relevantní informace především v anglickém jazyce, jelikož v českém jazyce k tomuto tématu není dostatek zdrojů. Výrazně jsem si touto prací zlepšil orientaci v terminologii daného oboru. Při vyhledávání však bylo důležité vybrat věrohodné informace.

7 Seznam použitých zdrojů

1. **Red Hat, Inc.** What is Linux? [Online] ©2019. Dostupné z: https://opensource.com/resources/linux.

2. **Togaware Pty Ltd.** GNU/Linux Desktop Survival Guide. [Online] ©2020. Dostupné z: https://www.togaware.com/linux/survivor/.

3. **Software in the Public Interest, Inc.** O Debianu. [Online] ©2019. Dostupné z: https://www.debian.org/intro/about.

4. **Red Hat, Inc. and others.** Fedora's Mission and Foundations. [Online] ©2020. Dostupné z: https://docs.fedoraproject.org/en-US/project/.

5. **Red Hat, Inc.** Red Hat Enterprise Linux. [Online] ©2020. Dostupné z: https://www.redhat.com/en/technologies/linux-platforms/enterprise-linux.

6. **SUSE LLC.** openSUSE Wiki. [Online] ©2020. Dostupné z: https://en.opensuse.org/Main_Page.

7. **SUSE.** SUSE Linux Enterprise Server. [Online] ©2020. Dostupné z: https://www.suse.com/products/server/.

8. **Canonical Ltd.** The story of Ubuntu. [Online] ©2020. Dostupné z: https://ubuntu.com/about.

9. **Citrix Systems, Inc.** What is virtualization? [Online] ©2020. Dostupné z: https://www.citrix.cz/glossary/what-is-virtualization.html.

10. **MATYSKA, Luděk.** Virtualizace výpočetního prostředí. *Zpravodaj ÚVT MU.* 2006, roč. 17, č. 2, s. 9-11. DOI: ISSN 1212-0901. Dostupné z: http://webserver.ics.muni.cz/bulletin/articles/540.html.

11. **YANOVSKYY, Vadym.** Virtualizace. [Online] ©2020. Dostupné z: https://www.fi.muni.cz/~kas/pv090/referaty/2016-podzim/virt.html.

12. **TOMEČEK, Jaroslav.** Virtualizace na úrovni jádra operačního systému. In: abclinuxu.cz. [Online] 31. 7. 2007 [cit. 28.12.2020]. Dostupné z: https://www.abclinuxu.cz/clanky/system/virtualizace-na-urovni-jadra-operacniho-systemu.

13. **MATYSKA, Luděk.** Techniky virtualizace počítačů (2). *Zpravodaj ÚVT MU.* 2007, roč. 17, č. 3, s. 9-12. stránky 9-12. DOI: ISSN 1212-0901. Dostupné z: http://webserver.ics.muni.cz/bulletin/articles/545.html.

14. **Red Hat, Inc.** What is KVM? [Online] ©2020. Dostupné z: https://www.redhat.com/en/topics/virtualization/what-is-KVM.
15. **Public Interest.** KVM. [Online] ©2020. Dostupné z: https://wiki.debian.org/KVM.
16. **WMware, Inc.** vSphere. [Online] ©2020. Dostupné z: https://www.vmware.com/products/vsphere.html.
17. **Microsoft.** Virtualization. [Online] ©2020. Dostupné z: https://docs.microsoft.com/cs-cz/windows-server/virtualization/virtualization.
18. **Microsoft**. Hyper-V on Windows Server. [Online] ©2020. Dostupné z: https://docs.microsoft.com/en-us/windows-server/virtualization/hyper-v/hyper-v-on-windows-server.
19. **Microsoft**. Hyper-V on Windows 10. [Online] ©2020. Dostupné z: https://docs.microsoft.com/en-us/virtualization/hyper-v-on-windows/.
20. **The Linux Foundation®.** Documentation. [Online] ©2020. Dostupné z: https://xenproject.org/help/documentation/.
21. **PHP Documentation Group.** PHP Manual. [Online] ©2020. Dostupné z: https://www.php.net/manual/en/index.php.
22. **NIXON, Robin.** *Learning PHP, MySQL, JavaScript, and CSS.* USA : O'Reilly Media, ©2012. str. 582. ISBN: 978-1-449-31926-7.
23. **British Columbia Institute of Technology.** CodeIgniter User Guide. [Online] ©2014. Dostupné z: https://www.codeigniter.com/user_guide/.
24. **PAVLÍČEK, Josef & SOLANSKÁ, Karolína a kol.** *Učebnice Interakce člověk počítač.*
25. **BERRANGÉ, Daniel P.** Virtual machine manager documentation. [Online] ©2017. Dostupné z: https://virt-manager.org/documentation/.
26. **MARTINET, Dominique & COHEN, Benjamin.** kvm-wrapper. [Online] ©2014. Dostupné z: http://codewreck.org/kvm-wrapper/.
27. Foreman 2.0 Manual. [Online] ©2020. Dostupné z: https://theforeman.org/manuals/2.0/.
28. Red Hat, Inc. Virsh-Linux documentation. [Online] ©2005. Dostupné z: https://linux.die.net/man/1/virsh.

29. BOWMAN, Will. PHP MVC Framework Showdown. In: Medium.com. [Online] 9.6.2017 [cit. 11.2.2020]. Dostupné z: https://medium.com/@asked_io/php-mvc-framework-showdown-7-1-performance-2da52ac9fcba.

30. Slant.co. Codeigniter vs Lumen. [Online] ©2020. Dostupné z: https://www.slant.co/versus/3756/9176/~codeigniter_vs_lumen.

31. Ivanov, KONSTANTIN. *KVM Virtualization Cookbook.* UK : Packt Publishing, ©2017. str. 471. ISBN 978-1-78829-467-6.

32. JetBrains s.r.o. PhpStorm. [Online] ©2020. Dostupné z: https://www.jetbrains.com/phpstorm/.

33. FIALA, Jan. Textový editor PSPad. [Online] ©2020. Dostupné z: http://www.pspad.com/cz/.

34. The kernel development community. The Linux Kernel Documentation. [Online] ©2020. Dostupné z: https://www.kernel.org/doc/html/latest/.

35. Red Hat, Inc. Understanding virtualization. [Online] ©2020. Dostupné z: https://www.redhat.com/en/topics/virtualization.

36. Red Hat, Inc. Product Documentation for Red Hat Virtualization 4.3. [Online] ©2020. Dostupné z: https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.3/.

37. StackShare, Inc. Codeigniter vs Lumen. [Online] ©2020. Dostupné z: https://stackshare.io/stackups/codeigniter-vs-lumen.

Přílohy

Administrační rozhraní.rar (18,8kB)