TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií

Studijní program: B2612 – Elektrotechnika a informatika
Studijní obor: 1234R567 – Název studijního oboru

Srovnání virtualizačních technologií

Comparison of virtualization technologies

Bakalářská práce

Autor: **Josef Pražák**
Vedoucí práce: Mgr. Milan Keršláger
Konzultant:

V Liberci 28. 5. 2009

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií
Ústav nových technologií a aplikované informatiky
Akademický rok: 2008/2009

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: Josef PRAŽÁK

Studijní program: B2612 Elektrotechnika a informatika

Studijní obor: Informatika a logistika

Název tématu: Srovnání virtualizačních technologií

Zásady pro vypracování:

1. Účel a možnosti virtualizačních technologií
2. Srovnání virtualizačních technologií v Linuxu a jejich alternativ v ostatních OS
3. Měření vybraných charakteristik, ověření prezentovaných schopností
4. Závěr, doporučení dle získaných dat

Rozsah grafických prací: dle potřeby dokumentace

Rozsah pracovní zprávy: cca 40 stran

Forma zpracování bakalářské práce: tištěná/elektronická

Seznam odborné literatury:

http://www.xen.org
http://kvm.qumranet.com
http://www.vmware.com
http://en.wikipedia.org/wiki/Comparison_of_virtual_machines

Vedoucí bakalářské práce: **Mgr. Milan Keršláger**
Ústav nových technologií a aplikované informatiky

Datum zadání bakalářské práce: **31. října 2008**

Termín odevzdání bakalářské práce: **29. května 2009**

prof. Ing. Václav Kopecký, CSc.
děkan

L.S.

prof. Dr. Ing. Jiří Maryška, CS
vedoucí ústavu

V Liberci dne 31. října 2008

Prohlášení

Byl(a) jsem seznámen(a) s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 o právu autorském, zejména § 60 (školní dílo).

Beru na vědomí, že TUL má právo na uzavření licenční smlouvy o užití mé bakalářské práce a prohlašuji, že **s o u h l a s í m** s případným užitím mé bakalářské práce (prodej, zapůjčení apod.).

Jsem si vědom(a) toho, že užít své bakalářské práce či poskytnout licenci k jejímu využití mohu jen se souhlasem TUL, která má právo ode mne požadovat přiměřený příspěvek na úhradu nákladů, vynaložených univerzitou na vytvoření díla (až do jejich skutečné výše).

Bakalářskou práci jsem vypracoval(a) samostatně s použitím uvedené literatury a na základě konzultací s vedoucím bakalářské práce a konzultantem.

Datum 29. 5. 2007

Podpis

Poděkování

Především bych chtěl poděkovat své rodině a přítelkyni Pavlíně za podporu v průběhu celého studiu. Také bych chtěl poděkovat panu Mgr. Milanu Keršlágerovi za trpělivost a pomoc s touto prací.

Abstrakt

Cílem práce je seznámení s účelem a možnostmi virtualizačních technologií v prostředí Linuxu a jejich porovnání podle vybraných charakteristik.

V dnešní době je důležité hledat úspory jak v oblasti energií tak v investiční. Jednou z cest je virtualizace. Díky virtualizaci je možné snížit počet fyzických serverů, ale virtualizace přispívá i k zvýšení bezpečnosti a dostupnosti služeb, umožňuje testování aplikací. V práci se zabývám nejběžnějšími technikami virtualizace, testuji jejich chování a možnosti. Je patrné, že výhodnou volbou pro virtualizaci serverů je hypervisor Xen, zejména pro vysoký výkon při provozu více virtuálních strojů a možnost provozovat Linux i Windows platformu hostovaných systémů. Naopak pro virtualizace stolního počítače je vhodné použít jiný nástroj například VMwreWorkstation nebo VirtualBox

Abstract

The objective of my bachelor's work is an introduction to the purpose and possibilities of the virtualization technologies in the Linux environment and their comparison according certain characteristics. Nowadays it is important to search for new savings in costs and energy. In the information technologies it is virtualization, which can help us with that. It is possible to reduce amount of servers, what finally save energy and hardware. But virtualizations improve security, availability of the services. It can help with testing new applications or systems. In my work I compare more common virtualization technologies and implementations. Xen is a very convenient choice for server virtualization, it is obvious. It has high performance in multiple guests' environment and it is also possible run Linux and Windows guests. But for desktop virtualization is better choice different virtualization system, for example VMware Workstation or VirtualBox.

Obsah

Abstrakt ... 5
Seznam zkratek a termínů ... 8
1 Úvod ... 9
1.1 Cíl práce ... 9
2 Principy virtualizace ... 10
2.1 Historie virtualizace ... 10
2.2 Přínosy virtualizace ... 10
2.3 Virtualizace, virtuální systém ... 11
2.3.1 Hypervisor ... 11
Standardní model počítače x86 bez virtualizace ... 12
2.3.2 Virtualizace a hardware ... 12
Módy procesoru ... 12
Hardwarová podpora virtualizace ... 13
Intel VT (IVT) ... 14
AMD virtualization (AMD-V) ... 14
3 Virtualizace platformy ... 15
3.1 Emulace / Simulace hardware ... 15
3.1.1 Princip Emulace a simulace ... 15
3.1.2 Implementace ... 16
QEMU ... 16
DOSBox ... 17
Bochs ... 17
3.2 Nativní virtualizace a plná virtualizace ... 17
3.2.1 Princip nativní virtualizace ... 17
3.2.2 Implementace ... 18
VMware Workstation 6.5 ... 18
VMware Player ... 19
Virtual Box ... 20
KVM ... 20
3.3 Paravirtualizace ... 22
3.3.3 Princip paravirtualizace ... 22
3.3.4 Implementace ... 22
Xen ... 22
Xen plná virtualizace ... 24
3.4 Vritualizace na úrovni operačního systému ... 25
3.4.1 Princip virtualizace na úrovni operačního systému ... 25
3.4.2 Implementace ... 25
Chroot ... 25
OpenVZ ... 26
Správa systémových zdrojů ... 26
3.5 Aplikační virtualizace ... 27
3.5.1 Princip Aplikační virtualizace ... 27
3.5.2 Implementace ... 27
Java Virtual Machine ... 27
3.6 Kompatibilní vrstva ... 28
3.6.1 Implementace ... 28
Wine ... 28
DOSEMU ... 28

4 Porovnání virtualizačních technik 29
4.1 Metodika testování 29
4.1.1 SysBench 29
Účel testu 29
Módy testu 29
SysBench konfigurace 30
Výstupy testů 31
4.1.2 Komprimace bzip2 31
Účel testu 31
Konfigurace testu 31
Výsup testu 31
4.1.3 Testovací nástroj ab 32
Účel testu 32
Ab konfigurace 32
Výstup testu 32
4.1.4 Živá migrace 32
Princip 32
Účel testu 33
4.1.5 Použitý hardware 33
4.1.6 Použitý software 33
4.1.7 Definice testovaného prostředí 34
4.2 Výsledky zátěžových testů 36
4.2.1 SysBench 36
4.2.2 Komprimace bzip2 (pbzip) 41
Test více procesorů 42
4.2.3 Test Apache http serveru 43
4.2.4 Živá migrace 46
Xen 46
OpenVZ 47
4.3 Vyhodnocení výsledků 48
4.3.1 OpenVZ 48
Doporučení 48
4.3.2 Xen 49
Doporučení 49
4.3.3 KVM 50
Doporučení 50
4.3.4 VMware Workstation 51
Doporučení 51
4.3.5 VirtualBox 51
Doporučení 52
4.4 Přínosy a použití virtualizace 52
5 Závěr 54
Příloha A – DOSBox 56
Příloha B – Hardware Qemu 56
Příloha C – Běžící VMware Workstation 57
Příloha D – VirtualBox 58
Příloha E – Běžící systém s nástrojem KVM 59
Příloha F – Nástroj Virtual Machine Manager na běžícím systému 60
Příloha H – Přehled testovaných implemetací 61
Příloha I – Skript použitý pro testovaní nástrojem SysBench 62

Seznam zkratek a termínů

Termín	Popis
Hostitelský systém (Host)	Operační systém, který zprostředkovává spouštění virtuálních strojů
Hostovaný systém (Guest)	Virtualizaovaný systém
Hypervisor (Virtual Machine Monitor)	Virtualizační vrstva (podrobnější popis v článku 2.1.1 Hypervisor)
Nativní systém	Nevirtualizaovaný systém
Tar (tar balík)	V případě souboru se jedná o větší počet souborů spojených do jednoho – tar balíku

1 Úvod

Virtualizace v posledních několika letech pokročila o velký kus vpřed. Nejvíce je to patrné u serverových technologií, kde je již virtuální server běžným pojmem. Důležitým podnětem, který přispěl k rozvoji a rozšíření virtualizace, je masivní rozvoj vícejádrových procesorů, spolu s implementací podpory virtualizačních technologií výrobci procesorů. Ani desktopové projekty však nezahálejí a i pro tento segment trhu se objevila řada zajímavých projektů, které se nadále rozvíjejí. V práci se zaměřím jak na typicky serverové řešení v podobě paravirtualizace (Xen), tak i na desktopově orientovaný software jako je například nativní virtualizace.

1.1 Cíl práce

Cílem práce je seznámit se s možnostmi virtualizačních technologií, porovnat je s informacemi zjištěnými z literatury a na základě výsledků obecných i reálných zatěžovacích testů a praktických zkušeností nastínit možnosti jednotlivých implementací a stanovit doporučení pro jejich využití v praxi. Příkladem může být posouzení, která z technologií je vhodnější pro provozování serverových technologií (například webhosting) a která je v hodnější pro virtualizaci desktopových počítačů.

2 Principy virtualizace

2.1 Historie virtualizace

Termín virtualizace se v prostředí počítačů objevuje od 60. let dvacátého století. Ve Watson Research Center – výzkumném středisku IBM – se v polovině roku 1960 započal projekt M44/44X, který byl zaměřen na výzkum stránkovacího mechanismu virtuálních strojů. Architektura byla založena na virtuálních strojích, kde hostující systém byl IBM 7044 (M44) a každý virtuální stroj byla experimentální image hlavního systému X44. V tomto případě se jednalo o částečnou virtualizaci s použitím hardware i software. Termínem virtualizace je nahrazen původní název pseudostroj pocházející z počátků systému CP-40 který již používal plnou virtualizaci.

2.2 Přínosy virtualizace

V tomto oddíle bych chtěl naznačit důvody, proč vlastně věnovat virtualizaci pozornost, co nám může virtualizace přinést. S nárůstem výkonu počítačů se jednotlivé stroje začaly využívat pro více aplikací (programů) a také je začalo využívat více uživatelů současně. Sice se využijí efektivně výpočetní zdroje, ale toto přináší také určitá rizika. Je evidentní, že může dojít třeba i k nechtěnému zásahu do chování ostatních programů běžících pod jedním systémem. Nemusí jít o poškození dat, ale třeba jeden program vyčerpá systémové prostředky a způsobí pád celého systému nebo ostatních aplikací. Díky virtualizaci lze tedy vyřešit otázku bezpečnosti a hospodárnosti.

Na otázku přínosů virtualizace lze nalézt řadu různých odpovědí v dokumentaci projektů a na stránkách výrobců virtualizačního software. Především v dnešní době je důležité snižování nákladů, ať už za pořízení hardware nebo provozních nákladů. Pokusil jsem se shrnout ty zajímavé a z mého hlediska podstatné přínosy.

Přínosy virtualizace:

- **Agregace serverů**
  - redukce počtu fyzických serverů
  - menší náklady na pořízení a provoz
  - optimalizace serverové infrastruktury
  - zvýšení dostupnosti a usnadnění managementu

- **Izolování aplikací**
  - zvýšení bezpečnosti
  - oddělení aplikací včetně OS
  - vyrovnávání zátěže (migrace)

- **HW abstrakce**
  - sdílení HW mezi několik OS a aplikací
  - podpora nového HW ve starém OS

- **Testovací a vývojové prostředí**
  - úspora HW (na jednom stroji více architektur)
  - virtuální sítě
  - testování různých scénářů před nasazením nových systémů

2.3 Virtualizace, virtuální systém

V prostředí počítačů pojem virtualizace vyjadřuje abstrakci (iluzi) hardware, který ve skutečnosti neexistuje. Virtualizovat lze jednotlivé hardwarové komponenty (virtuální CPU, paměť, disk, síťová karta), nebo celé virtuální systémy složené z virtuálního hardware. Výhodou virtualizovaného prostředí je především možnost snáze přizpůsobit virtualizované prostředí potřebám uživatele. Příklad počítače s virtuálním systémem popisuje obrázek číslo 2. Na hostitelském systému - hardware (nejspodnější vrstvě) a operačním systému - běží virtualizační nástroj (v tomto případě emulátor). Ten odděluje hostovaný systém od hostitelského, emuluje potřebný hardware a poskytuje další funkcionalitu, zejména nezbytný management.

Hostovaný systém běží za pomoci virtualizační vrstvy (hypervisoru) na emulovaném hardware stejně jako na hardware skutečném. Stejně tak i spouštěné procesy pracují s virtuálním hardwarem jako se skutečným. Takto je možné sdílet jeden fyzický stroj více uživateli za pomoci virtuálních strojů. Virtuální stoje se chovají jako nezávislé stroje i při běhu na jenom hardware.

Vitrualizované prostředí by mělo splňovat některé podmínky. V prvé řadě musí být splněny požadavky na bezpečnost. Virtuální stroje od sebe musí být navzájem izolované, dále je požadována snadnost nasazení a správy. Také musí být možné přenášet virtuální stroje mezi fyzickými a výkon virtuálního stroje nesmí být dramaticky nižší než u stroje fyzického.

2.3.1 Hypervisor

Hypervisor (také znám pod názvem Virtual machine monitor – VMM) je virtualizační vrstva, která dovoluje více operačním systémům běžet na hostitelském počítači ve stejném čase.

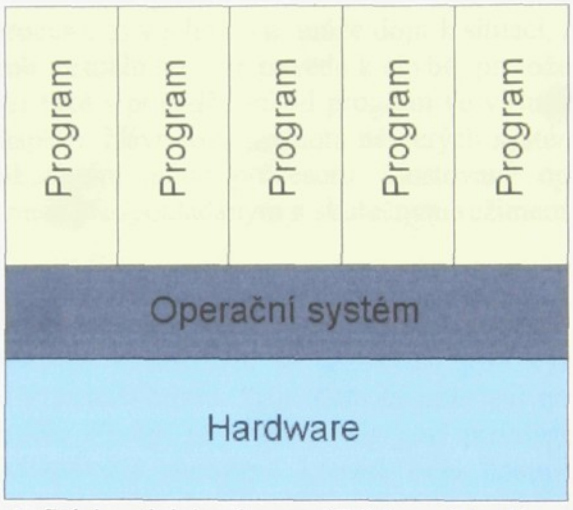
Hypervisory lze rozdělit na dvě základní skupiny:

a) Nativní – v tomto případě se jedná o software, který běží přímo na dané hardwarové platformě. Načítá se tedy ještě před startem vlastního operačního systému. Řídí operační systém a hardware hostovaného systému. Příkladem tohoto typu hypervisoru jsou například projekty XEN, VMware ESX Server, L4 microkernels, TRANGO, IBM LPAR hypervisor, Microsoft Hyper-V a SUN Logical Domains Hypervisor KVM, které změní kompletně Linux kernel na hypervisor.
b) Hostitelský – jedná se o software, který běží uvnitř operačního systému. Hostované systémy) tedy běží ve třetí vrstvě nad hardwarem. Příklad: VMware Server, VMware Workstation, VMware Fusion, open source QEMU, VirtualBox, Parallels Worktation and Parallels Desktop.

Standardní model počítače x86 bez virtualizace

Počítač, tak jak ho známe, vychází z von Neumannovy architektury. Počítač se skládá z funkčních jednotek – procesor, (řídící jednotka + ALU) paměť, vstupní a výstupní zařízení, obecně periferií. Díky hardware je možný běh operačního systému. Obrazně řečeno, operační systém běží ve vrstvě hned „nad" hardwarem a není zde žádná mezivrstva. Zde je důležité podotknout, že i na takovémto systému je možné se s virtuálním zařízením setkat. Například virtuální disk poskytovaný řadičem, virtuální paměť operačního systému, ale my se zaměříme na virtualizaci operačních systémů a celých strojů a toto nebudeme uvažovat.

V této práci se zaměříme na operační systém Linux. Důvodem je především jeho volná dostupnost a široká podpora virtualizační technologií. Proto budeme předpokládat, že na počítači běží právě operační systém Linux - CentOS 5.3. Na obrázku číslo 1. je schématicky zobrazen počítač s operačním systémem. Hardware leží nejníže, vrstva nad ním je operační systém Linux, v němž, nebo nad nímž běží programy (aplikace).



Obr. 1. Schématické zobrazení PC bez virtualizace

2.3.2 Virtualizace a hardware

Módy procesoru

Jak jsem uvedl v úvodu, virtualizovat lze prakticky veškerý hardware. Například běžně používaná virtuální paměť počítače se realizovala díky relativně snadné úpravě hardware (rozdělení paměti na stránky, podpora mapování virtuálních a fyzických stránek, souvislé adresování virtuální oblasti, jejíž stránky mohou být libovolně nesouvisle mapovány na fyzickou paměť, odkládání nepoužívaných stránek na pevný disk a vytvoření iluze mnohem větší paměťové kapacity než je skutečně dostupná). Oproti tomu virtualizace procesoru x86 je obtížnější. Tento procesor označovaný také jako IA-32 nebyl navrhnut pro virtualizaci, ale pro běh jednoho operačního systému, a proto jsou řešení použitá na této architektuře jsou pouze softwarová. Obtížná virtualizace souvisí s takzvanými režimy běhu procesoru založenými na vrstvách nebo prstencích – „ring". Jedná se o bezpečnostní princip, kdy procesor umožňuje několik úrovní běhu 0, 1, 2, 3. Typické jsou dva režimy. Privilegovaný režim – označovaný jako vrstva 0, kdy proces běžící v této vrstvě má přístup přímo k hardware. V tomto módu běží jádro operačního systému, jedná se o takzvaný ‚kernel space'. Může provádět privilegované instrukce, jako je například nastavení mapování paměti, zákaz přerušení, IO operace, řízení CPU. Naopak v neprivilegovaném režimu (mód 3) běží

aplikace uživatelů – je označován jako ‚user space'. Software běžící v nižší privilegovanější vrstvě může kontrolovat běh programu ve vyšší vrstvě. Většina procesorů architektury x86 běží v režimu 0 a 3.

Pro běh hostovaného operačního systému je tedy zapotřebí aby hypervisor také simuloval běh procesoru jak v neprivilegovaném režimu, tak v režimu privilegovaném. Ve skutečnosti ale není možné, aby hostovaný systém běžel plně v privilegovaném režimu, protože by mohlo dojít k modifikaci kódu hypervisoru a dat, nebo například způsobit problémy v běhu hostitelského systému. Ve skutečnosti je za pomoci hypervisoru zaveden další režim běhu procesoru „ring 1" ve kterém běží operační systém hostovaného systému. Tato technika je nazývána *„ring deprivileging"*. V modelu 0/1/3 běží operační systém hostovaného systému v modu 1 (ring 1) a aplikace virtualizovaného systému standardně v modu 3, což umožňuje jednoduše oddělit běžící aplikace a operační systému virtuálního stroje. Protože hostovaný systém nemá přímý přístup k hardware, musí hypervisor zachytávat systémová volání, vykonat je a generovat patřičnou odezvu. Toto je místo, kde virtuální systémy výkonnostně zaostávají oproti systémům konvenčním. Navíc i zde stále existuje nebezpečí vzniku kolizí. Například pokud se hostovaný operační systém dotáže na stav procesoru v domnění, že procesor je v jeho režii, může dojít k situaci, že hypervisor vrátí stav skutečného procesoru, nikoli virtuálního, což povede k chybě, protože se stav procesorů liší. Podobná chyba může nastat také v případě, pokud program ve virtuálním stroji běží v jiném módu, než pro který byl napsán. Návratová hodnota některých systémových instrukcí může obsahovat informace o aktuálním módu procesoru. Hostovaný operační systém je pak schopný rozeznat konflikt mezi předpokládaným a skutečným režimem běhu procesoru.

Hardwarová podpora virtualizace

Z důvodu problematické virtualizace na platformě x86 výrobci procesorů AMD a INTEL přišli s virtualizačním rozšířením. Principem je zavedení nového módu procesoru, kdy oproti původnímu modelu 0/3 byl implementován plně privilegovaný mód, ve kterém běží hypervisor, a neprivilegovaný mód, ve kterém běží hostovaný operační systém. Důležitou vlastností je, že navíc oba dva mody procesoru v sobě obsahují opět všechny úrovně běhu. Toto umožňuje běh hostovaného operačního systému v požadovaném módu s iluzí plné kontroly procesoru, přitom nehrozí kolize, protože operační systém ve skutečnosti neběží v privilegovaném módu.

Hlavním přínosem virtualizačního rozšíření je to, že umožňuje běh neupraveného hostovaného operačního systému v hostitelském systému bez výrazných ztrát výkonu, jak je tomu například u emulátorů. Zjednodušuje návrh hypervisoru, zvyšuje jeho spolehlivost a robustnost. Hypervisor je schopný díky podpoře virtualizačního hardware snáze pracovat s událostmi, výjimkami a zdroji přiřazenými virtuálnímu stroji.

Intel VT (IVT)

IVT je zkratka pro Intel Virtualization Technology, dříve známá jako Intel Vanderpool technology. Firma Intel zařadila podporu virtualizačních technologií pro platformu x86 v roce 2005. V té době se jednalo o některé typy procesorů Pentium4 6x1 a 6x2, Pentium D a serverovou verzi Xeon. Podpora virtualizačních technologií se přenesla do většiny modelů novějších typů Core a Core2, již tedy 64 bit procesorů. Dnes je již součástí většiny procesorů Core2, Core i7 a zcela jistě nebude chybět u nových generací. Jedná se o sadu virtualizačních rozšíření, která přinášejí podporu virtualizace procesoru pro více virtuálních strojů.

Součástí Intel VT je také Virtualization for Directed I/O (VT-d). VT-d umožňuje například DMA remaping, kdy virtuální stroj může přímo přistupovat do paměti s minimální účastí hypervisoru, nebo přemapování přerušení. Další funkcí je Virtual Machine Device Queues (VMDq). Jedná se o hardwarové vylepšení, které s pomocí různých front a třídících pravidel umožní zvýšení síťové propustnosti ve virtuálním prostředí. Podporu Intel VT lze v Linuxu jednoduše ověřit přítomností ***VMX*** (virtual machine extension) flagu v /proc/cpuinfo. V některých implementacích je možné virtualizační rozšíření vypnout v BIOSU.

AMD virtualization (AMD-V)

Firma AMD (Advanced Micro Devices) zavedla virtualizační rozšíření pro 64 bitové procesory architektury x86 s názvem AMD Virtualization, zkráceně (AMD-V), který je také znám pod kódovým názvem projektu jako „Pacifica".

AMD-V je přítomen v některých verzích procesoru AMD Athlon 64 (stepping F a G), Athlon 64 X2, Turion, Opteron, Phenom a všech novějších procesorech AMD. Podporu virutalizačního rozšíření lze v Linuxu ověřit výpisem /proc/cpuinfo, kde je přítomen flag ***SVM***. Mezi technologie, které přináší virtualizační rozšíření AMD, patří například implementace technologie IO Memory Management Unit (IOMMU), která převádí adresy virtuální paměti na paměť fyzikou. Direct Connect Architecture přináší vylepšení správy paměti, přímý přístup CPU do paměti, CPU na I/O zařízení, meziprocesorové komunikace a Rapid Virtualization Indexing (RVI), který zvyšuje výkon systému díky upravenému systému správy paměti při běhu více virtualizovaných aplikací .

3.1.2 Implementace

QEMU

Qemu je emulátor procesoru, který využívá dynamického překladu (Binary translation) pro dosažení vyššího výkonu při emulaci. Spolu s emulací CPU, Qemu také poskytuje potřebná zařízení pro běh nemodifikovaného hostovaného operačního systému. Qemu naprogramoval Fabrice Bellard a jedná se o svobodný software šířený pod licencí GNU LGPL. Qemu byl využit jako základ pro některé další virtualizační projekty. Pro zvýšení výkonu emulátoru byl vyvinut Linuxový kernelový modul KQEMU, také známý jako Qemu Akcelerátor. Běžně se výkon Qemu pohybuje od 10% do 30% systému nominálního. S pomocí tohoto jaderného modulu lze dosáhnout více jak 90% výkonu systému nativního.

Qemu podporuje dva módy:

- Plná emulace systému. V tomto módu je emulován kompletní systém, to znamená nejenom procesor, ale také další potřebná zařízení. Může být tedy použit pro běh odlišného operačního systému bez restartu PC, nebo například k debuggingu systémového kódu.
- Uživatelská emulace. V tomto módu je možné pomocí Qemu pouštět programy původně zkompilované pro jiné procesory na odlišném typu CPU. Například pro spouštění Wine - Windows API emulatoru, Dosemu, usnadnění cross-platform kompilace.

Pro lepší představu o funkcionalitě Qemu, uvádím některé základní funkce:

- emulace procesorů typu x86, x86_64, MIPS R4000, Sun SPARC, ARM PowarPC, Power Macintosh
- SMP podpora
- Podpora virtuálních síťových karet
- Podpora „rostoucích" diskových image
- Podpora snapshotů
- Podpora suspendu
- Akcelerace pomocí KQEMU kernelového modulu
- Konzolové nástroje
- Podpora VNC serveru

Qemu v sobě přináší vcelku komplexní funkcionalitu. Instalace je jednoduchá a pro vlastní spuštění stačí několik příkazů a můžeme instalovat virtuální stroj. Ovládání je snadné za pomoci grafického prostředí lze jednoduše ovládat běh virtuálního stroje.

Vekou nevýhodou je rychlost, kdy oproti nativnímu hardware je patrný propad výkonu. Modul Kqemu přináší sice zlepšení, ale systémová režie je stále vysoká. Qemu nemůže konkurovat výkonem projektům jako je KVM nebo VirtualBox, či VMware Workstation, ale je důležité si uvědomit, že se jedná o emulátor a jeho použití je směrováno pro emulaci platformy. Projektu Qemu byl využit například pro projekt XEN, nebo KVM využívající Qemu jako nástroj pro provozování virtuálních storjů. Hardware emulované emulátorem Qemu je zobrazeno pomocí příkazu „lspci" a „cat /proc/cpuinfo" v příloze B.

3 Virtualizace platformy

Virtualizace platformy se používá pro označení virtualizace celých počítačů nebo operačních systémů. V tomto oddíle si popíšeme různé přístupy k virtualizaci platformy.

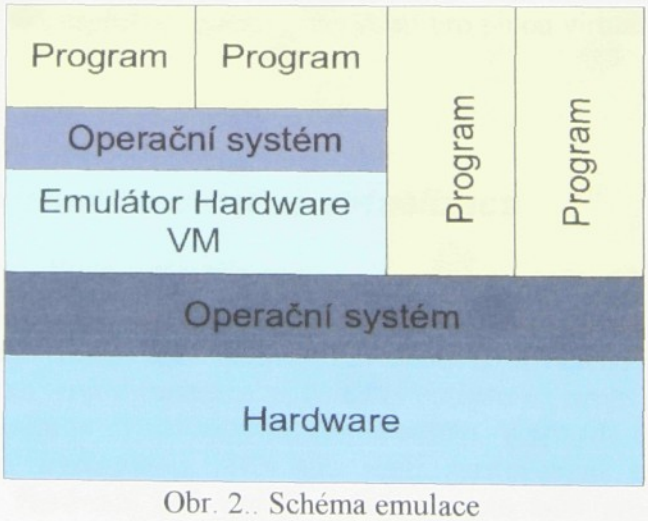
3.1 Emulace / Simulace hardware

3.1.1 Princip Emulace a simulace

Při virtualizaci se emulátory používají k úplné simulaci programu, operačního systému, nebo celého hardware. Hlavním rysem emulátorů je, že umožňují emulaci jiného typu CPU, než na kterém běží. Emulátor například simuluje kompletní počítač Intel x86 (386, 486) včetně instrukčních sad procesoru MMX, SSE, 3DNow!, IO zařízení jako klávesnice, síťové karty, grafické karty. Lze tedy nezávisle na použitém operačním systému nebo procesoru spouštět neupravené hostované aplikace nebo celé operační systémy na jiné platformě, než pro kterou jsou naprogramovány, což je hlavním rysem emulátorů. Bohužel tato výjimečná vlastnost je vykoupena velikým propadem výkonu oproti systému skutečnému.

Schéma emulace je na obrázku číslo 2. Hostovaný systém běží nad vrstvou emulátoru. Emulátor zachytává všechny instrukce hostovaného operačního systému a transformuje je pro daný hardware. Hostovaný systém v žádném okamžiku nepřistupuje ke skutečnému hardware, přístup je vždy „zprostředkován". Často se využívá takzvaného binárního překladu.

Tato forma virtualizace se používá například pro vývoj aplikací, simulace prostředí jiných architektur, herní emulátory, emulace starých systémů.



Obr. 2.. Schéma emulace

DOSBox

Z názvu je patrné, že se jedná o emulátor prostředí MS-DOS. Přesněji DOSBox emuluje počítač IBM PC kompatibilní s běžícím operačním systémem MS-DOS. Program je volně dostupný pod licencí GNU GPL. Používá se ke spuštění MS-DOS programů v prostředí Linuxu, ale i dalších operačních systémů.

DOSBox emuluje kompletní PC Intel x86 obsahující grafickou kartu, zvukovou kartu, myš, joystick, modem a další hardware. V současné době DOSBox podporuje i 64 bitové systémy. Využití najde především pro spuštění aplikací vyžadujících ke svému běhu MS-DOS. Je oblíbenou aplikací pro spouštění starých MS-DOS her, nebo například účetních programů.

DOSBox působí velice jednoduchým ale také praktickým dojmem. Prostředí DOSBoxu, které se zobrazí po spuštění emulátoru je zobrazeno v příloze A.

Bochs

Bochs je program simulující kompletní PC Intel x86, včetně periferií, BISOu, instrukčních sad. Lze nastavit, aby simuloval různé generace procesorů x86, počínaje 386, přes Pentia až po x86-64.

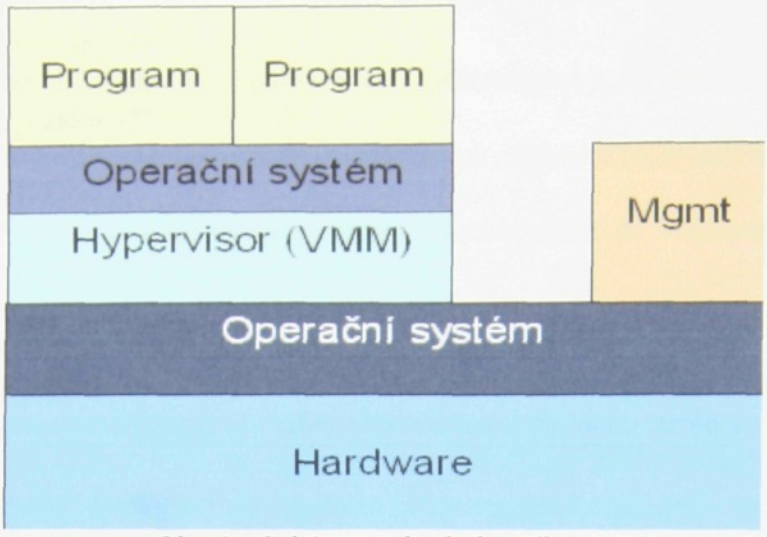
Bochs je volně dostupný, je distribuován pod GNU licencí. Autorem je Kevin Lawton a naprogramován je v C++. Bochs je v současné době možné provozovat na různých platformách vedle x86, také PPC, Sun, MIPS, Alpha. Podporované operační systémy jsou Linux, Windows, Mac OS X. Umožňuje běh řady hostovaných operačních systémů, jako například DOS, Windows, Linux, Amiga.

Bochs je pro ovládání složitější a komplikovanější, pro jeho rozchození si nový uživatel musí vyčlenit něco času a také trpělivosti a ani to nemusí vždy vést k požadovanému výsledku. Z mého pohledu je Bochs určen spíš pro vývojáře a debuging, než pro uživatele, kteří potřebují jen nemodifikovaný operační systém v prostředí emulátoru. BIOS původně vytvořený pro Bochs se s úspěchem používá například pro plnou virtualizaci v projektu XEN nebo KVM.

3.2 Nativní virtualizace a plná virtualizace

3.2.1 Princip nativní virtualizace

Nativní Virtualizace je virtualizační technika, která slouží k simulaci kompletního virtuálního stroje a umožňující běh nemodifikovaného hostovaného operačního systému. Důležitým rysem tohoto typu virtualizace je, že CPU virtuálního stroje je stejného typu, jako procesor stroje, na kterém je virtualizační nástroj spouštěn. V případě nativní virtualizace se využívá hostitelského hypervisoru. Ten leží mezi hostovaným operačním systémem a nativním hardware. Hardware leží v nižší vrstvě, a proto není pro hostovaný operační systém přímo dostupný. Místo toho se stará o sdílení a zpřístupnění hardware hypervisor, který má potřebná oprávnění. Některé instrukce hostovaného OS musí být zachyceny a řízeny virtualizační softwarem. Plná nebo nativní virtualizace přináší o něco vyšší výkon než emulace, ale stále se z důvodu režie hypervisoru pouze blíží nativnímu stroji. V současné době se na platformě x86 (X86_64) masivně začalo využívat podpory hardwarové virtualizace, což vede opět k nárůstu výkonu virtuálních strojů.



Obr. 3.. Schéma nativní virtualizace

3.2.2 Implementace

VMware Workstation 6.5

Přesto, že se jedná se o komerční produkt, zmíním zde i VMwareWorkstation od firmy VMware. Stránky firmy jsou: http://www.vmware.com/ .Tato americká firma je lídrem v oblasti virtualizace a specializuje se právě na virtualizační software. Firma VMware poskytuje velké modulární celky serverové infrastruktury s propracovaným managementem, kdy je v rámci virtuálních technologií možné provozovat ucelené prostředí serverové infrastruktury složené z jednotlivých virtuálních serverů (VMware ESX) a platform (WMware VIrtual SMP, VMware VMFS).

V oblasti klientských počítačů nebo pracovních stanic nabízí na architektuře x86 produkt VMware Workstation. Jedná se o komerční uzavřený software, v současné době je však možné stáhnout jej na 1 měsíc pro otestování. Existuje však verze volně dostupná, ovšem s omezenou funkčností. Jedná se o VMware Player.

VMware Workstation je virtualizační nástroj, který se řadí do nativní virtualizace a virtualizace platformy x86, x86-64. Lze tedy na jenom stroji typu x86 provozovat několik virtuálních strojů stejné platformy a to bez úprav operačního systému hostovaného i hostitelského operačního systému.

Podporované Operační systémy jsou Linux a Windows, od verze 6 jsou podporovány i 64 bitové verze operačních systémů. Podpora hostovaných systémů je široká, Windows, Linux, Solaris x86, Netware, FreeBSD.

Instalace je snadná buď pomocí rpm balíčku, nebo instalačního skriptu. Ovládání je přehledné a intuitivní za pomoci propracovaného grafického prostředí. Nespornou výhodou je komplexnost ovládání, kdy lze jednoduše od tvorby virtuálních strojů, jejich spouštění, tvorbu shapshotů, konfigurace sdílených složek, virtuálních sítí a jiných nastavení jednoduše upravovat nastavení virtuálních strojů. Další zajímavou vlastností je možnost ovládání za pomoci textového rozhraní. Nově od verze 6 dokáže VMware Workstation využít hardwarové podpory virtualizace. Grafické prostředí VMware Workstation 6 je zobrazeno v příloze C.

Z pohledu hostitelského stroje běží v systému proces hypervisoru vmware-vmx spolu s několika dalšími zajišťující management (mware-tray), nebo síťovou komunikaci (vmware-bridge). Proces vmware-vmx alokuje pro virtuální stroj systémové zdroje, jako je paměť a procesor.

Některé dostupné funkce:

- podpora více procesorů
- každý stroj má vlastní konfiguraci svého zařízení (disky, IO zařízeni, CD)
- možnost přidělit až 8 GB RAM
- snadné přepínání mezi virtuálními stroji, jejich zastavení (suspend) a oživení (resume)
- možnost konvertovat operační systém z fyzického stroje do VMware
- podpora více monitorů
- podpora USB 2.0
- možnost tvorby snímků (snapshots) a klonování strojů
- módy síťové karty, NAT, přemostění (Bridged), lokální virtuální (host-only)
- Doplňky hostovaného stroje (sdílené adresáře hostovaného stroje, ovladače grafické karty, myši, USB)
- Zabudovaná podpora VNC
- Konzolové nástroje (vmrun)
- Dokáže využít hardwarové podpory virtualizace
- součástí instalace je VMware player

VMware Player

VMware Player je zdarma dostupný software pro „přehrávání" virtuálních strojů již dříve vytvořených například právě ve VMware Workstation. Jedna z chybějících funkcí je tedy tvorba virtuálních strojů, tvorba snímků a také jsou dost omezeny možnosti konfigurace.

VMware Workstation patří skutečně ke špičce mezi virtualizačními nástroji pro desktopy. Přináší komfortní ovládání a práci s virtuálním strojem v grafickém prostředí. Instalace a konfigurace je snadná, buď pomocí rpm, nebo tar balíčku lze instalovat kompletní virtualizační program. Nevýhodou je, že se jedná o produkt komerční. Sice je zde VMware Player, který je zadarmo, ale ten je opravdu dost omezený. Proto lze domácímu uživateli jen doporučit volně dostupnou alternativu k VMwareWorkstation, která se jmenuje VirtualBox.

Virtual Box

VirtualBox je stejně jako VMwareWorkstation virtualizační nástroj – hostitelský hypervisor, který umožňuje nativní virtualizaci platformy x86. VirtualBox emuluje dostatek zařízení potřebných pro běh nemodifikovaného operačního systému. Při vývoji VirtualBoxu bylo využito zdrojových kódů emulátoru Qemu, což přineslo celkové urychlení vývojové fáze a odstranění chyb.

VirtualBox vyvinula Německá firma Innotek, GmbH, která je nyní součástí Sun Microsystems, která pokračuje ve vývoji a VirtualBox se stal součástí virtualizační platformy Sun xVM . Innotek uvolnil většinu kódu emulátoru pod licencí GNU GPL 2 jako nekomerční verzi produktu. Existuje také komerční verze, která obsahuje některé další funkce jako například podporu USB, RDP server (Remote Desktop Prtocol), iSCSI. Nicméně komerční verze je také k dispozici pro domácí použití na stránkách firmy: http://www.virtualbox.org/.

Instalace je snadná, jsou k dispozici rpm balíčky pro různé distribuce. VirtualBox dokáže využít hardwarovou podporu virtualizace od firmy Intel IVT a AMD-V. Přítomnost hardwarové podpory virtualizace je detekovaná „tray" ikonkou.

Pro síťovou komunikaci je k dispozici síťové rozhraní „most", které umožňuje plnohodnotnou síťovou komunikaci virtuálního stroje, nebo mód NAT, ve kterém je virtuální stroj pro okolní stroje „neviditelný".

Dostupné funkce:

- Snímky (Snapshots), umožňuje uložení aktuálního stavu VM
- Bezešvý mód (Seamless mode), umožňuje zobrazovat okno aplikace běžící ve virtuálním stroji přímo do hostícího systému
- Schránka (Clipboard)
- Sdílené adresáře (Shared folders), jednoduché sdílení souborů mezi systémy
- Speciální ovladače a utility napomáhající přepínání mezi systémy
- Textové nástroje pro ovládání VM (rozšiřující GUI)
- Vzdálená plocha (Remote display)
- Podpora CPU s virtualizační rozšířením
- Experimentální OpenGL ovladače
- Podpora 64 bit. systémů

VirtualBox je poměrně nový, ale z mého pohledu velice povedený program na poli Nativní virtualizace x86 platformy. Obsahuje všechny potřebné funkce pro pohodlí uživatele. Ovládání a konfigurace je díky propracovanému grafickému rozhraní velice příjemné a intuitivní. Směle se vyrovná konkurentům jako je VMware Player, nebo Micorsoft Virtual PC. Jeho předností je rychlost a jednoduchost, což je umocněno tím, že je zadarmo. Mé zkušenosti jsou velice pozitivní. Běžící systém s nástrojem VirtualBoxu je zobrazen v příloze D.

KVM

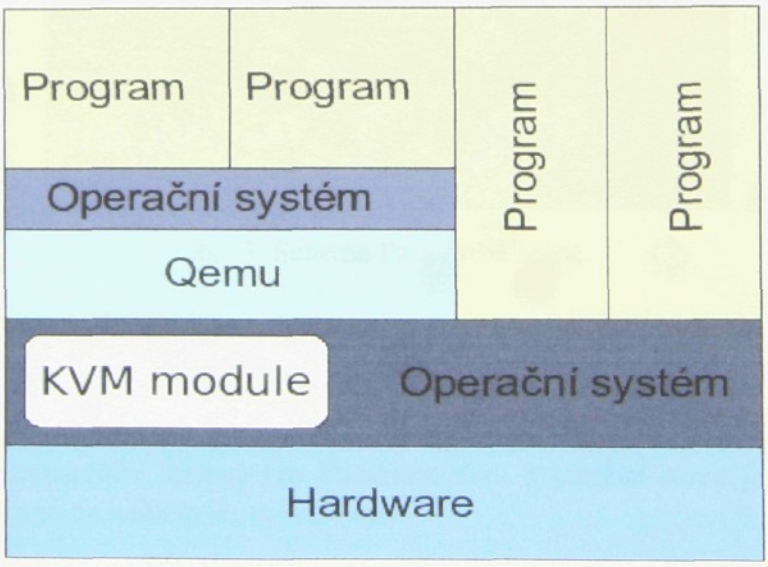
KVM (Kernel Based Virtual machine), je řazen do kategorie plné virtualizace, nově také „hardware enabled", tedy volně přeloženo s podporou hardware. KVM je projekt poměrně nový. Start projekt KVM se datuje do poloviny roku 2006 a do hlavní větve jádra byl zařazen od verze 2.6.20. V současné době je KVM portováno na Linux x86 (x86_64), BSD a pracuje se na PowerPC a IA64 platformách. Podporované hostované systémy jsou Linux, BSD, Solaris, Windows, Haiku, ReactOS.

KVM pro svůj běh potřebuje hardwarovou podporu virtualizace současných mikroprocesorů x86 Intel VT a AMD SVM. Tvůrci KVM přichází se zajímavou myšlenkou: proč neustále investovat zdroje do vývoje hypervisoru v reakci na vývoj hardware, když v podstatě díky Linux kernelu již takovýto hypervisor máme. Schéma KVM je na obrázku číslo 5. S přidáním virtualizační funkce do standardního kernelu můžeme využit propracované funkce plánovače úloh kernelu, kdy proces virtuálního stroje je řízen standardním plánovačem (schedulerem). Není tedy zapotřebí implementovat další mechanizmus a paměť je alokována alokátorem Linux kernelu se znalostí NUMA technologie.

Princip vychází z toho, že proces v Linuxu může běžet standardně ve dvou módech, privilegovaném „kernel mode" a neprivilegovaném „user mode". KVM využívá třetí mód, takzvaný „guest", který má svůj vlastní kernel a user mód. Přínosem je, že guest mód má možnost provádět některé privilegované operace nebo instrukce, které by v user módu nemohl provádět, což zjednodušuje a zrychluje funkci virtuálního stroje.

KVM je v kernelu přítomno v podobě jaderných modulů kvm, kvm-intel a kvm-amd. Při načtení příslušných modulů kvm se vytvoří zařízení /dev/kvm. Pro vlastní spouštění virtuálních strojů se využívá upraveného emulátoru Qemu, který obstará potřebný virtuální hardware a funkcionalitu pro jejich správu. Qemu proces mapuje fyzickou paměť hostovaného operačního systému a umožňuje spouštění privilegovaných operací s pomocí guest modu. Běžící systém s KVM je zobrazuje příloha E.

Jako přínos u KVM vidím to, že je součástí standardního jádra. Není tedy nutné instalovat hypervisor, vše potřebné obstarají jaderné moduly. Výkon a chování systému je opravdu svižnější oproti emulátoru Qemu. Patrné je to i na hostitelském systému v zatížení procesoru. Ovládání je s emulátorem Qemu totožné, lze použít i virtuální stroje vytvořené v Qemu. Trochu je znát, že projekt je stále ještě v počátcích, protože ne vše funguje, jak by se dalo očekávat a konfigurace je omezena na konzolové nástroje což nemusí každému vyhovovat.

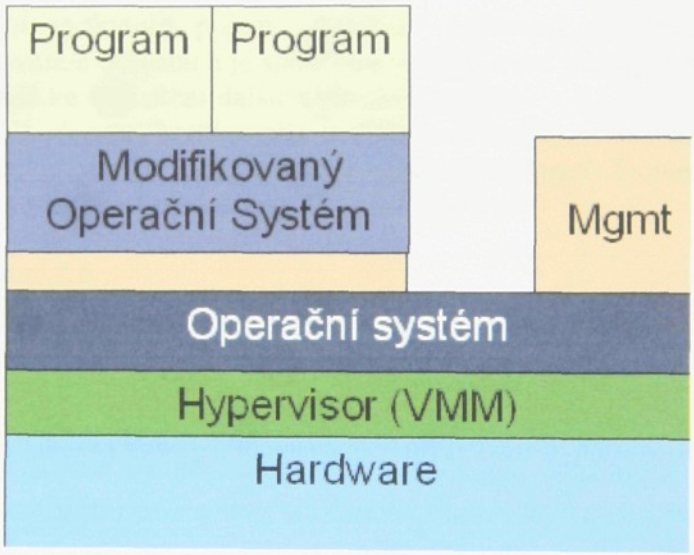


Obr 4. Schéma Linuxu s KVM

3.3 Paravirtualizace

3.3.3 Princip paravirtualizace

Paravirtualizace se řadí mezi populární, především u serverů častěji používané virtualizační techniky. U tohoto typu virtualizace se používá nativní hypervisor, který je spuštěn ještě před vlastním operačním systémem a až následně se spouští vlastní OS. Hypervisor v tomto případě vytváří softwarové rozhraní velice podobné, ne však totožné se skutečným hardwarem. Paravirtualizace pro běh hostovaných systémů potřebuje modifikovat hostovaný operační systém Schéma paravirtualizace je na obrázku číslo 6. Účelem hypervisoru, v tomto případě neprivilegovanější a nejníže ležící vrstvy (nad hardware), je řídit přístup k hardware. Díky tomuto není nutná rekompilace nebo zachytávání systémových volání, protože operační systém se sám podílí na virtualizačním procesu a také se poměrně výrazně zjednodušuje vlastní návrh hypervisoru. Jako hlavní výhodu se udává vysoký výkon, blížící se téměř skutečnému systému. Běh hostovaných systémů je pak řízen za pomocí řídících utilit.



Obr. 5. Schéma Paravirtualizace

3.3.4 Implementace

Xen

Typickým představitelem paravirtualizace je Xen. Xen je open-source paravirtualizační hypervisor, určený pro platformu x86. S pomocí Xenu je možné spouštět více virtuálních strojů na jednom stroji fyzickém.

Xen byl vyvinut na univerzitě v Cambridge pod vedením Iana Pratta, zakladatele XenSource, Inc. V současné době je XenSource vlastněn Citrix Systems a od 22. 10. 2007 se Xen projekt přesunul na http://www.xen.org/. Projekt je podporován firmami Citrix, IBM, Hewlett-Packard, Novell, Red Hat, Sun Microsystems a dalšími a stal se součástí řady komerčních produktů.

Dostupné funkce:

- Vysoký výkon hostovaných systémů, podpora SMP
- „Live migration" – přesun běžícího virtuálního stoje mezi dvěma fyzickými
- podpora až 32 virtuálních procesorů
- podpora 32 bitové x86 a 64 bitové x68_64 platformy
- podpora Intel VT-x pro běh nemodifikovaných operačních systémů
- široká podpora hardware – téměř všechny Linuxové ovladače

Xen lze provozovat na upravených verzích Linuxu, NetBSD, Solarisu. Jako hostované systémy mohou být použity upravené unixové systémy. Díky podpoře virtualizačních technologií v CPU lze jako hostovaný systém v současné době provozovat i neupravené proprietární systémy jako je např. MS Windows. Xen je k dispozici buď jako zdrojový kód, nebo v podobě již zkompilovaných binárních souborů v podobě RPM, nebo tarballů. Některé Linuxové distribuce mají volitelně k instalaci xen kernel. Je to například RedHat, CentOS, Fedora nebo openSUSE.

V Xenu se vyskytuje pojem „doména" (domain). Existuje doména0, které představuje privilegovanou doménu a je spouštěna vzápětí hned po hypervisoru. Tato doména „nula" nám pak slouží ke spouštění dalších virtuálních strojů a k jejich řízení, respektive je k tomu oprávněna. Na ukázce konfigurace z GRUB je vidět, že je místo souboru kernelu (vlinuz) v řádku kernel spouštěn xen a Linuxový kernel až následně. DoménaU (domU) je pak označován neprivilegovaný virtuální stroj běžící nad doménou0.

```
title Xen 3.0 / XenLinux 2.6
    kernel /boot/xen-3.0.gz dom0_mem=262144
    module /boot/vmlinuz-2.6-xen0 root=/dev/sda4 ro console=tty0
```

Obr. 6. Ukázka konfigurace zavaděče při použití hypervisoru Xen

CPU

Jak bylo zmíněno v úvodu věnovanému operacím procesoru, pro umožnění virtualizace se přidává další mód procesoru. Xen ve své implementaci zavádí kromě běžných módů procesoru 0 a 3 také mód 1, ve kterém běží hostovaný operační systém. Jak je obvyklé, aplikace běží v módu 3 a privilegované operace v módu 0.

Paměť

U nemodifikovaných systémů se pracuje se souvislou operační pamětí. Xen při paravirtualizaci vyžaduje úpravu operačního systému (kernelu), aby umožňoval pracovat s nesouvislou pamětí. Operační systém je pak sám odpovědný za alokaci paměti a správu paměťových stránek, nicméně zápis do paměti je zachycován a korigován hypervisorem.

Oproti plné virtualizaci je paravirtualizace v podaní Xenu složitější o fakt, že virtuální stroje neumí použít kernel uvnitř image virtuálního stroje. Je tedy nutné umístit kernel virtuálních strojů někde uvnitř domény0. (například v /boot) Některé distribuce nepoužívají zvláštní kernel pro doménu0 a pro doménuU, ale použijí kernel pouze jeden společný pro oba

stroje. Tento kernel se pak nazývá Xenkernel. Výhodou je široká podpora hardware, tedy univerzálnost. Nevýhodou je delší doba kompilace, množství modulů v kernelu a nutnost použití initrd a častěji je také zdrojem potíží.

Nástroje pro management Xend, xm, Virtual Machine Manager

Z pohledu hostitelského stroje není možné běžnými systémovými nástroji sledovat zatížení virtuálních systémů, a proto se spolu s hypervisorem do systému instalují nástroje pro management, které to umožňují. Podrobněji bych se chtěl zmínit o nástroji, který se jmenuje Virtual Machine Manager. Používá se pro snadné ovládání Xenu za pomoci grafického prostředí. Mezi základní funkce patří vedle zobrazení informací o stavu virtuálního stroje opět tvorba virtuálních strojů, konfigurace hardware, jako je množství paměti, CPU nebo například přidání pevného disku. Systém s běžícím nástrojem Virtal Machina Manager ukazuje příloha F.

Pro řízení Xenu z příkazové řádky lze použít nástroj „xm". Jednoduše lze zobrazit zobrazovat informace o běžících strojích, množství paměti, virtuálních CPU, které domény mají atd. Nástroj xm lze použít za předpokladu, že běží xend – daemon, který zajišťuje řízení virtuálních strojů. Spouští se po startu v prostředí privilegované dom0.

Xen je již součástí většiny linuxových distribucí a určitě na poli serverové virtualizace najde své uplatnění. Díky grafickému nástroji Virtual Machine Manager je vytvoření virtuálního stroje podobné jako u nástrojů plné vizualizace VMware nebo VirtualBox, což zjednodušuje nasazení a ovládání. Poskytuje vysoký výkon blížící se nativním systémům. Díky podpoře plné virtualizace má Xen potenciál zasáhnout i do oblasti desktopů, kde lze jednoduše instalovat například MS Windows do virtuálních strojů.

Xen plná virtualizace

Vývojáři Xenu s příchodem podpory hardwarové virtualizace u procesorů implementovali plnou virtualizaci. Aby tato funkce byla dostupná, je tedy nutné mít tuto podporu u procesoru dostupnou a aktivovanou.

Instalace virtuálního stoje je za pomoci nástroje virt-manager podobná, ovšem vlastní implementace je už poněkud odlišná. Hlavní rozdíl je patrný už při výpisu procesů v dom0, kdy je možné sledovat proces „qemu-dm". Zjednodušeně řečeno, tento proces v sobě zahrnuje celý virtuální systém, respektive jeho požadavky na systémové prostředky hostitelského systému, tak jak je tomu například u plné virtualizace s pomocí nástroje VirtulBox. Z názvu je patrné, že se jedná se o upravenou verzi emulátoru Qemu. Je zřejmé, že plná virtualizace bude výkonnostně zaostávat za paravirtualizací. Na obrázku 6. je uveden výpis procesu qemu-dm.

```
/usr/lib64/xen/bin/qemu-dm -d 1 -m 1024 -boot c -serial pty -vcpus 1 -acpi -usb -k en-us -
domain-name CENTOS-FULL -net nic,vlan=1,macaddr=00:16:3e:61:47:20,model=rtl8139 -
net tap,vlan=1,bridge=xenbr0 -vnc 127.0.0.1:1 –vncunused
```

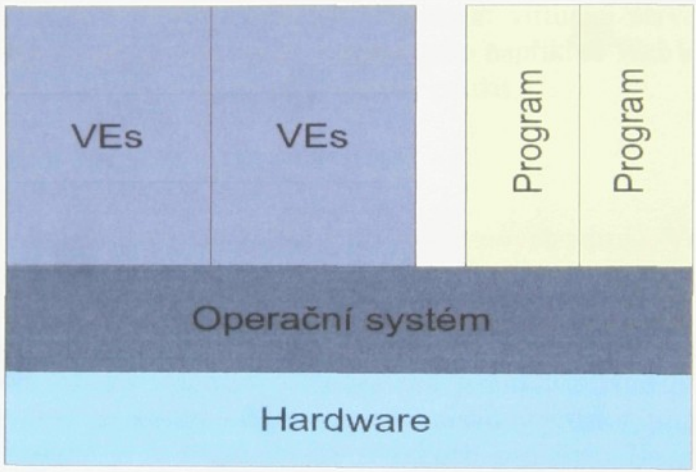
Obr. 7. Výpis procesu qemu-dm

3.4 Vritualizace na úrovni operačního systému

3.4.1 Princip virtualizace na úrovni operačního systému

Virtualizace na úrovni operačního systému umožňuje běh více izolovaných virtuálních prostředí (severů). Tyto servery se z pohledu jejich uživatele chovají jako skutečné servery. Technika virtualizace na úrovni operačního systému se liší od předchozích technik, protože při virtualizaci se nevyužívá hypervisor, ale upravený kernel. Nemusí zde být simulován hardware, který by umožňoval nezávislý běh operačního systému. Virtuální stroje běží přímo nad operačním systémem hostitelského stroje, využívají jeho kernel.

Přesněji řečeno, jako takový ve skutečnosti běží operační systém pouze jeden, v kontejnerech běží pouze procesy spuštěné při startu virtuálního prostředí. Nejedná se tedy o virtuální stroje v pravém slova smyslu, ale spíše o virtuální prostředí. Virtualizační prostředí zajišťuje jak vzájemnou izolaci virtuálních strojů, tak izolaci od prostředí hostitele. Jednotlivé virtuální servery jsou nazývány virtuálním prostředím (VEs), nebo kontejnery. Schéma virtualizace na úrovni operačního systému je na obrázku 7.



Obr. 8. Schéma virtualizace na úrovni OS

3.4.2 Implementace

Chroot

Mezi aplikační vizualizaci lze zařadit i příkaz „chroot". Název je odvozen od anglického „change root". Byl představen v roce 18. 4. 1982, původně určen pro BSD. Jedná se o volný software šířený pod licencí GNU GPL. Příkaz umožní změnu kořenového adresáře pro daný proces a jeho potomky. V prostředí chrootu je proces izolován od původního systému a nemá možnost přistupovat k jeho adresářové struktuře. Této vlastnosti se s oblibou využívá u procesů, které jsou vystaveny útokům z vnější, jako je Apache nebo Bind.

Pro úspěšný chroot je tedy potřeba mít adresářovou strukturu, do které se chceme přepnout, a také mít k dispozici heslo uživatele root. Chroot tedy nevyužívá prostředí žádného hypervisoru, ani nevytváří žádné virtuální zařízení, vše je nutné připojit před vlastním chrootem.

Chroot je poměrně jednoduchý a nevyžaduje úpravu kernelu, proto je užitečným nástrojem například při obnově systému, testování a vývoji, kompatibilitě aplikací, kontrole závislostí.

OpenVZ

OpenVZ je komunitní open source projekt, šířený pod licencí GPL. Je základem pro komerční produkt Parallels Virtuoso. OpenVZ umožňuje běh nezávislých systémů v kontejnerech (označovaných také jako virtuální prostředí – VEs, nebo virtuální privátní server VESs). Navenek se kontejner tváří jako samostatný systém s vlastní IP adresou, vlastními uživateli, knihovnami, běžícími službami, /proc souborovým systémem.

Z pohledu hostitelského stroje jsou však patrné běžící procesy v jednotlivých kontejnerech a je také možné procházet adresáři jednotlivých kontejnerů. OpenVZ tedy vychází z principu chrootu. Na stránkách projektu http://wiki.openvz.org/Main_Page jsou k dispozici patche pro kernel a nástoje pro management.

Pro paralelní běh několika systémů je nutné, aby byl kernel upraven patchem. Patch kernelu zajistí virtualizaci, izolaci, správu zdrojů a také například možnost živé migrace. V kontejnerech nelze provozovat jiný systém než Linux, ale alespoň je možné provozovat různé distribuce Linuxu. Některé distribuce, jako je Fedora nebo SUSE, nabízejí možnost instalace kernelu s patchem OpenVZ, což odstraní náročnější konfiguraci a kompilaci kernelu.

Na internetových stránkách projektu jsou volně dostupné šablony široké řady Linuxových distribucí, a tak není problém otestovat více virtuálních prostředí. S pomocí několika příkazů lze vytvořit a jednoduše nakonfigurovat virtuální server. Po pohodlnější ovládání existují i volně dostupné webové managery, jako například WebVZ, s jehož pomocí není zapotřebí v podstatě konzolové nástroje prakticky použít.

Správa systémových zdrojů

U tohoto typu virtualizace je důležitá správa systémových zdrojů. Ve skutečnosti tomu totiž není tak, že si virtuální stroj alokuje paměť, procesor atd. při spuštění, ale má dostupné prostředky přímo z hostitelského systému, a tak by ve skutečnosti hrozilo, že by mohl jeden kontejner při nadměrné zátěži ovlivnit všechny kontejnery vyčerpáním zdrojů. Proto přináší patch kernelu možnost omezit systémové zdroje pro jednotlivé kontejnery. Například je možné limitovat jednotlivé parametry virtuální paměti, diskový prostor, přidělování procesoru atd. Velice pěkný manuál je k dispozici na stránkách projektu. Na druhou stranu tato implementace přináší výhodu v tom, že systémové prostředky nevyužívané jedním kontejnerem mohou být v daný okamžik přiděleny jinému kontejneru.

Pro management slouží nástroje vzctl, vzpkg, vzdump. Nástroj se používá pro řízení kontejnerů. S pomocí příkazu vzctl lze vytvořit kontejner z existující šablony operačního systému, poté provést potřebné nastavení jako je síťové jméno, nastavení sítě, nastartovaní stroje a potřebných služeb, vše vně kontejneru. Jak je patrné, při použití skriptu je vytvoření nového virtuálního prostředí je otázkou sekund.

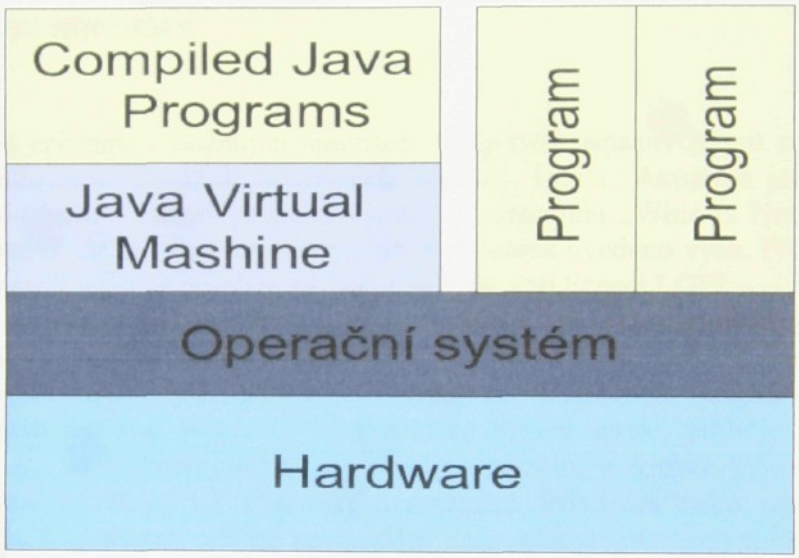
Pro vytváření šablon operačního systému slouží nástroj vzpkg. Zjednodušeně lze šablonou označit za lehce upravený zapakovaný operační systém.

OpenVZ je poměrně snadné instalovat a konfigurovat, alespoň z mého pohledu. Velice příjemná je možnost rychlého vytvoření a konfigurace kontejneru z šablony. Proto je OpenVZ vhodný pro použití v oblasti web hostingu, pronajímání virtuálních serverů, zmenšování počtu fyzických serverů, vývoji software a výuce.

3.5 Aplikační virtualizace

3.5.1 Princip Aplikační virtualizace

Aplikační virtualizace je založena na technice použití softwarové virtualizační vrstvy mezi počítačovým programem, který chceme spustit a operačním systémem na kterém chceme program spustit. Schéma aplikační virtualizace je zobrazeno na obrázku č. 2. Když spustíme program nad touto virtualizační vrstvou, virtualizační vrstva zachycuje všechna volání (například I/O operace) a předává je operačnímu systému. Takto spuštěný program se chová, jako by měl přímý přístup k operačnímu systému. Programy jsou spuštěny na lokálním stroji, ovšem instalace se liší od konvenčních instalací. Instalační zdroje a komponenty, jako jsou soubory a konfigurace, jsou zpravidla uloženy v jednotlivých balíčcích. Balíčky jsou pak s použitím virtualizační vrstvy interpretovány a zpřístupňovány jako by byly instalovány na lokálním systému. Typickým představitelem Aplikační virtualizace je Java Virtual Machine.



Obr. 9. Schéma aplikační virtualizace

3.5.2 Implementace

Java Virtual Machine

Asi nejznámějším představitelem Aplikační virtualizace je Java Virtual Machine (JVM). JVM zde představuje virtualizační vrstvu mezi programem a operačním systémem. Je tedy nutné nejprve implementovat JVM určenou pro platformu (Linux, Windows) a poté lze spouštět programy kompilované pro JVM. Java programy jsou kompilované do takzvaného Java bytecodu, který lze chápat jako strojový kód JVM. Bytecode je poté puštěn pomocí Java Virtual Machine. Pro spuštění programu je potřeba mít v Linuxu instalován Java Runtime Environment.

3.6 Kompatibilní vrstva

Závěrem stručného přehledu typů virtualizace a některých jejich implementací bych rád zmínil takzvanou techniku označovanou jako kompatibilní vrstva (compatibility layer) a to i přesto, že se nejedná o skutečnou virtualizační techniku nebo emulaci. Myslím, že je dobré vysvětlit, v čem se liší od virtualizačních technik. Jednoduše lze říci, že tato vrstva slouží pro spouštění programů určených pro odlišné systémy na daném stroji. Kompatibilní vrstva funguje podobně jako některé emulátory, které využívají binárního překladu (Bochs, Qemu). Zachytávají se systémová volání určená pro cizí systém a za pomoci kompatibilní vrstvy jsou přeložena pro běžící systém. Většinou za pomoci knihoven z cizího systému je poté možné relativně jednoduše spouštět binární soubory kompilované pro jiný systém. Důležitým rozdílem je tedy to, že se neemuluje žádný hardware ani operační systém, jedná se pouze o soustavu API. Typickým příkladem je stále se rozvíjející projekt Wine, nebo DOSEMU.

3.6.1 Implementace

Wine

Wine je aplikace umožňující spouštění programů napsaných pro operační systémy Microsoft Windows v prostředí operačních systémů Linux. Aktuálně jsou podporovány systémy 32 i 64 bitové. Název projektu vychází z akronymu „Wine Is Not Emulátor", což přeloženo znamená „Wine není emulátor", jak je nakonec uvedeno výše. Projekt Wine začal v roce Bob Amstadt a Eric Yougdale a ktuálně je šířen pod licencí LGPL.

Wine implementuje Windows API výhradně v uživatelském prostředí bez potřeby zavedení modulů jádra. Služby poskytované kernelem Windows jsou v případě Wine poskytovány démonem pojmenovaným Wineserver. Wineserver přináší vedle základní funkcionality poskytované Windows také některé funkce navíc, jako je integrace do X Windows Linuxu. Je samozřejmé, že ne všechno chování a funkce Windows kernelu se podařilo implementovat ať už z důvodu uzavřenosti Windows nebo nemožnosti použít nativních ovladačů hardware, což má za následek nefunkčnost některých aplikací.

Wine je stále se rozvíjející projekt a určitě zaslouží pozornost.

DOSEMU

DOSEMU přináší prostředí MS-DOS, je v podstatě alternativou k aplikaci DOSBox. Vychází z kombinace technik virtualizace hardware a emulace pro dosažení téměř skutečné rychlosti procesoru 8086-kompatibilního. V současné době je podporován pouze 32 bitový systém.

4 Porovnání virtualizačních technik

Tato část práce je věnována porovnání výkonnostních parametrů několika implementací virtualizačních technik. Pro objektivní porovnání výkonnostních parametrů jsem vybral testovací nástroj SysBench, který poskytuje požadovanou funkcionalitu pro prověření různých parametrů systému. Jedná se o open source projekt a zdrojové kódy jsou dostupné na adrese: http://sysbench.sourceforge.net.

Jako alternativu k zátěžovým testům nástroje SysBench jsem provedl test komprimace velkého „tar" balíku za pomoci běžně dostupného nástroje bzip2, přičemž doba komprimace byla měřena za pomoci příkazu „time". Zároveň se výsledek použije pro ověření výsledků benchmarku SysBench.

Pro demonstraci reálné zátěže jsem vybral test web serveru Apache. Pro generování zátěže jsem použil nástroj „ab" (Apache http server benchmarking tool). Testoval jsem 4 druhy zátěže: statickou HTML stránku, PHP aplikaci, redakční systém Drupal (PHP+MySQL) a jednoduchý skript napsaný v Perlu.

Pro doplnění jsem provedl test živé migrace u Xenu a OpenVZ

4.1 Metodika testování

4.1.1 SysBench

SysBench je modulární, multiplatformní nástroj pro testování a ověření parametrů operačního systému (benchmark), specielně parametrů důležitých pro provoz zatížených databázových systémů. Cílem tohoto testovacího nástroje je poměrně rychle a snadno získat přehled o výkonnosti systému bez nutnosti konfigurace a instalace komplexních testovacích nástrojů pro databáze.

Při testu SysBench spouští definované množství vláken, která poté současně generují zátěž. Způsob zátěže se liší podle typu zvoleného testu a také lze omezit množství požadavků, nebo dobu, po kterou bude zátěž generována

.

Účel testu

Získání základních výkonnostních charakteristik systému použitelných pro vzájemné porovnání mezi testovanými systémy. Porovnání výkonnosti systému neovlivněného a ovlivněného jinými hostovanými systémy.

Módy testu

CPU

Jedná se o jednoduchý test zaměřený na zjištění výpočetního výkonu procesoru. Testovací úloha obsahuje výpočet prvočísel z čísla zadaného při spuštění testu. Výpočty jsou realizovány pomocí 64 bitových celých čísel. Test je spouštěn paralelně, až do dosažení časového limitu, nebo definovaného počtu dotazů.

Threads

Test je zaměřen na měření rychlosti plánovače (scheduler). SysBench vytvoří určené množství vláken a definované množství mutexů a jednotlivá vlákna o ně soupeří.

Mutex

Cílem tohoto testu je ověřit výkonnost implementace mutexu. V průběhu testu při paralelním běhu více vláken se jednotlivá vlákna snaží uzamknout mutex, ale pouze na krátký časový úsek, čímž je docíleno neustálého uzamykání a odemykání mutexu.

Memory

Test měří sekvenční čtení a zápis do operační paměti. V závislosti na parametrech se dá určit, zda vlákna budou přistupovat do globálních nebo lokálních bloků paměti.

FileIO

Tento test vytváří různé druhy zátěže souborového systému (file I/O). V přípravné fázi je vytvořena struktura souborů a adresářů, která se v průběhu testu používá pro generování zátěže.

OLTP

Tento test je určen pro měření skutečné výkonnosti databázového systému. Před spuštěním vlastního benchmarku je v přípravné fázi vytvořena databáze s tabulkou a následně je do tabulky zapsáno určité množství řádků. Níže je uveden SQL příkaz pro vytvoření testovací tabulky.

```
CREATE TABLE `sbtest` (
`id` int(10) unsigned NOT NULL auto_increment,
`k` int(10) unsigned NOT NULL default '0',
`c` char(120) NOT NULL default '',
`pad` char(60) NOT NULL default '',
PRIMARY KEY (`id`),
KEY `k` (`k`);
```

Obr. 10. Příkaz pro vytvoření testovací tabulky

Pro testování databáze se v tomto testu, pokud je podpora v databázovém systému, používají transakce. Pokud dojde v průběhu testu k vymazání některého z řádků v tabulce, následně je řádek znovu vytvořen. Proto může být test opakován bez obav, že bude výsledek zavádějící. V tomto testu se používají dotazy typu SELECT FROM, SELECT DISTINCT, UPDATE SET

SysBench konfigurace

Jednotlivé módy benchmarku jsou spouštěny tak, aby doba běhu jednotlivých testů byla v řádech minut, příliš krátké testy by nemusely odhalit jemné rozdíly. U jednotlivých módů benchmarku se provedlo 10 spuštění testu a jako výsledek je použit aritmetický průměr z výsledných hodnot. Pro snadné spuštění benchmarku jsem vytvořil jednoduchý skript, který spustí požadované testy a výsledky ukládá do souboru. Skript pro spuštění je v příloze I, kde jsou patrné hodnoty použité v testu. Test síťového subsystému je za pomoci síťového klienta, který generuje zátěž, přičemž parametry pro spuštění benchmarku se neliší od lokálního testu přes unix socket. Konfigurace testu SysBench je v tabulce 1.

Tab. 1: Parametry spuštění nástroje SysBench

Test	Testovaná vlastnost	Parametry
CPU	Početní operace	2 vlákna, počet dotazů 10000, výpočet prvočísla do hodnoty 140000
Threads	Robustnost implementace vláken	64 vláken, počet dotazů 10000, 4 thready zamknuty
Mutex	Robustnost implementace vláken	64 vláken, počet dotazů 10000, počet uzamčení 10000000, počet mutexů 4096
Memory	Rychlost operací s pamětí – zápis	8 vláken, 400GB zápis do paměti
FileIO	Rychlost diskových operací	8 vláken, náhodné čtení a zápis 5 GB, maximální počet dotazů 200000
OLTP	Rychlost databázových operací	8 vláken, při testu použity transakce, počet řádků v tabulce 100000

Výstupy testů

Výstupem jednotlivých testů je doba testu, která je hlavním kritériem pro porovnávání. Čím rychleji se úloha provede, tím je lepší výsledek. Podrobnější informace o testovacím nástroji jsou k dispozici v dokumentaci na stránkách projektu.

4.1.2 Komprimace bzip2

Komprimace je vzhledem k její náročnosti na výkon a dobré škálovatelnosti se vzrůstajícím výkonem často využívaným způsobem porovnávání výkonnosti procesorů. V tomto testu poslouží také jako alternativa k benchmarku SysBench pro ověření získaných výsledků. Běžně používaný nástroj pro komprimaci souborů bzip2 není potřeba představovat. Domovská stránka je http://bzip.org, kde je dostupná dokumentace a zdrojové kódy. Pro přehled o výkonu při více procesorech jsem použil upravenou verzi bzip2 - pbzip2.

Účel testu

Získání praktických výkonnostních charakteristik systému použitelných pro vzájemné porovnání mezi testovanými systémy a ověření obecných testů nástroje SysBench.

Konfigurace testu

Pro test byl použit tar balík obsahující velké množství souborů různých typů. Jedná se o „zatarovanu" strukturu operačního systému Linux. Tento tar balík (testfile.tar) je poté za pomoci nástroje bzip2 komprimován za použití maximální komprese. Velikost souboru před komprimací je 3692600 KB a velikost souboru po komprimaci je 1152600 KB, kompresní poměr je 3,2. Soubor byl zvolen s ohledem na přiměřené generování diskových operací a zátěže procesoru.

Příkaz použitý při testu: *„time bzip2 –k -9 testfile.tar"*

Výsup testu

Výsledkem je výstup nástoje „time" a použit je skutečný čas komprese (real time).

4.1.3 Testovací nástroj ab

Ab je nástroj pro měření výkonnosti Apache HTTP serveru. Je navržen tak, aby dal uživateli představu o aktuální výkonnosti instalace serveru Apache.

Účel testu

Získání praktických výkonnostních charakteristik web serveru Apache použitelných pro vzájemné porovnání mezi testovanými systémy. Porovnání výkonu jediného zatíženého systému a výkonu systému s více virtuálními stroji současně.

Ab konfigurace

Celkem jsem provedl 4 druhy měření. Testy jsou opakovány 3x a použit je nejlepší výsledek. Zároveň je měřeno zatížení systému „system load".

- Statická html. Vytvořil jsem jednoduchou statickou stránku, která je při testu načítána.
- PHP. Při testu jsem použil aplikaci phpAlbum, do které jsem nahrál 250 fotografií. Poté jsem generoval zátěž na konkrétní stránku aplikace.
- PHP + MySQL. Při testu jsem použil známý systém pro správu obsahu – Drupal. Vytvořil jsem několik stránek a poté jsem zatěžoval systém.
- Perl. Vytvořil jsem jednoduchý skript a poté jsem zatěžoval systém dotazy na tento skript.

Jako parametr se testovacímu nástroji předává celkový počet dotazů generovaných na server a množství paralelních (konkurenčních) dotazů které se budou provádět.

Výstup testu

Výsledek udává počet dotazů za vteřinu, které je schopen Apache obsloužit. U každého testu jsem měřil také „system load", který bude pomocným ukazatelem při porovnávání výsledků.

4.1.4 Živá migrace

Živá migrace je ceněnou vlastností v oblasti virtualizace. Přínos živé migrace je zřejmý – minimalizace doby, kdy je služba mimo provoz. Po dobu migrace nedochází k přerušení provozu serveru a služby poskytované hostovaným systémem nejsou přerušeny. Fakticky dochází pouze ke zvýšení odezvy virtuálního stroje. Měla být splněna podmínka, že migrace nesmí trvat příliš dlouho.

Princip

Velice zjednodušeně lze průběh živé migrace popsat tak, že se obsah paměti zdrojového stroje přenese do paměti stroje cílového bez přerušení provozu.

První fáze spočívá v tom, že zdrojový stroj pokračuje v běhu a zároveň jsou přenášeny některé stránky paměti do cílového stroje. Stránky změněné v průběhu migrace se musí znovu přenést.

Další fáze spočívá v zastavení zdrojového stroje, přenesení stránek paměti do cílového stroje a nastartování stroje cílového. V poslední fázi je cílový stroje pleně aktivován, připojí si zařízení a propaguje IP adresu.

Účel testu

Ověřit chování „migrovaného" systému, dobu migrace.

4.1.5 Použitý hardware

Pro test jednotlivých typů virtualizace, stejně jako pro otestování systému bez virtualizace, využijeme vždy stejný hardware. Informace o serveru, na kterém jsou testovány virtualizační nástroje, jsou v tabulce 1. Konfigurace disků do pole RAID0 byla použita z důvodu snahy urychlení diskových operací a omezení jejich vlivu na výsledky testů, i když v reálném provozu se častěji pro systémové disky používá konfigurace disků do pole RAID1.

Pro zpracování výsledků jsem použil tabulkový procesor MS Excel, soubor s naměřenými a zpracovanými hodnotami je na přiloženém CD (vyhodnocení-testy.xls).

Tab. 2: Použitý hardware

Procesor	2x Intel Xeon – X5460, Quad-Core – 3.16 GHz
Paměť	16 GB, 667 MHz
Čipová sada	Intel 5400
Řadič disků	LSI Logic SAS1068
Pevné disky	2x FUJITSU 300 GB, Model: MBA3300RC, konfigurace RAID0
Základní deska	Hewlett-Packard – 0A98h
Síťová karta	Broadcom Corporation NetXtreme BCM5755 Gigabit Ethernet
Virtualilzační podpora	Intel Virtualization, Intel IO Virtualizaton
Switch	Gigabit switch 3Com
Síťový klient	Shodná konfigurace s jako testovací server

4.1.6 Použitý software

Tab. 3: Použitý software

SysBench	Verze 0.4.12
MySQL	Verze 5.0.45, RPM mysql-5.0.45-7.el5
Operační system	CentOS 5.3, x86_64, kernel 2.6.18-128.1.6.el5
Nástroj bzip2	Verze 1.0.3, balíček bzip2-1.0.3-4.el5_2
Nástroj time	Balíček time-1.7-27.2.2
Nástroj ab	Verze 2.0.40, phpAlbum, Drupal

4.1.7 Definice testovaného prostředí

Pro lepší přehlednost jsem umísil popis testovaných prostředí do tabulek 4 až 10.

Tab. 4: Nativní systém (referenční)

Procesor	1x Intel Xeon X540 3,16 GHz
Paměť	1 GB
Operační systém	CentOS release 5.3 (Final), kernel 2.6.18-128.1.10.el5
Typ virtualizace	Žádná
Souborový systém	EXT3, použit samostatný diskový oddíl
Parametr kernelu	Mem=1G, maxcpus=1

Tab. 5: XEN – paravirtualizace

Procesor	1x VCPU
Paměť	1 GB
Operační systém	CentOS release 5.3 (Final), kernel 2.6.18-128.1.10.el5xen
Typ hypervisoru	Nativní hypervisor - paravirtualizace, XEN xen-3.0-x86_64 který je součástí distribuce.
Souborový systém	EXT3, použit samostatný diskový oddíl

Tab. 6: XEN – plná virtualizace

Procesor	1, 2, 4 VCPU
Paměť	1 GB
Operační systém	CentOS release 5.3 (Final), kernel 2.6.18-128.1.10.el5xen
Typ hypervisoru	Nativní hypervisor - paravirtualizace, XEN xen-3.0-x86_64 který je součástí distribuce.
Souborový systém	EXT3, použit samostatný diskový oddíl
Parametr hypervisoru	Parametr dom0_mem=1535M (omezené množství operační paměti eliminuje nežádoucí diskovou chache)

Tab. 7: VirtualBox

Procesor	1x VCPU
Paměť	1 GB
Operační systém	CentOS release 5.3 (Final), kernel 2.6.18-128.1.10.el5xen
Typ hypervisoru	Hostitelský hypervisor – nativní virtualizace, balíček VirtualBox-2.2.2_46594_rhel5-1
Souborový systém	EXT3, virtuální disk v souboru .vdi
Doplňky	Instalovány doplňky hostovaného OS

Tab. 8: VMware Workstation

Procesor	1, 2 VCPU
Paměť	1 GB
Operační systém	CentOS release 5.3 (Final), kernel 2.6.18-128.1.10.el5
Typ virtualizace	Hostitelský hypervisor, plná virtualizace, balíček VMware-Workstation-6.5.2-156735.x86_64.rpm
Souborový systém	EXT3, použit virtuální disk – soubor vmdk
Jiná nastavení	Vypnutá disková cache pro zápis z důvodu zavádějících výsledků (write through)
Doplňky	Instalovány doplňky hostovaného OS

Tab. 9: KVM

Procesor	1, 2, 4 VCPU
Paměť	1 GB
Operační systém	CentOS release 5.3 (Final), kernel 2.6.18-128.1.10.el5
Typ virtualizace	Nativní virtualizace s prvky paravirtualizace (hardware-asisted), balíček kvm-85-1.el5, komod-kvm-85-1.el5
Souborový systém	EXT3, použit soubor typ qcow2
Parametr kernelu	Limit hostitelského mem=1536M, maxcpus=2

Tab. 10: OpenVZ

Procesor	1,2,4 CPU
Paměť	1 GB
Operační systém	CentOS release 5.3 (Final), Kernel ovzkernel-2.6.18-128.1.1.el5.028stab062.3
Typ virtualizace	Aplikační virtualizace, verze nástroje vzctl-3.0.23-1
Souborový systém	EXT3
Kvóty	Výchozí konfigurace je upravena pro 1GB operační paměti, počet současně otevřených souborů nastaven na 512. (PRIVVMPAGES="262144: 278528")
Parametr kernelu	Maxcpus=1, mem=2G

4.2 Výsledky zátěžových testů

4.2.1 SysBench

Zátěžové testy jsem provedl u hostovaného systému bez další zátěže a u systému s dalšími 4 běžícími systémy. Díky opakovanému pokusu se zatíženým systémem je možné zjistit míru, jakou ovlivňuje nezatížený hostovaný systém ostatní hostované systémy. Zkratky použité v grafech a tabulkách jsou uvedeny v tabulce 11.

Pro přehlednější prezentaci získaných hodnot grafy uvádí procentuelní poměr výsledných hodnot jednotlivých implementací ku systému nativnímu.

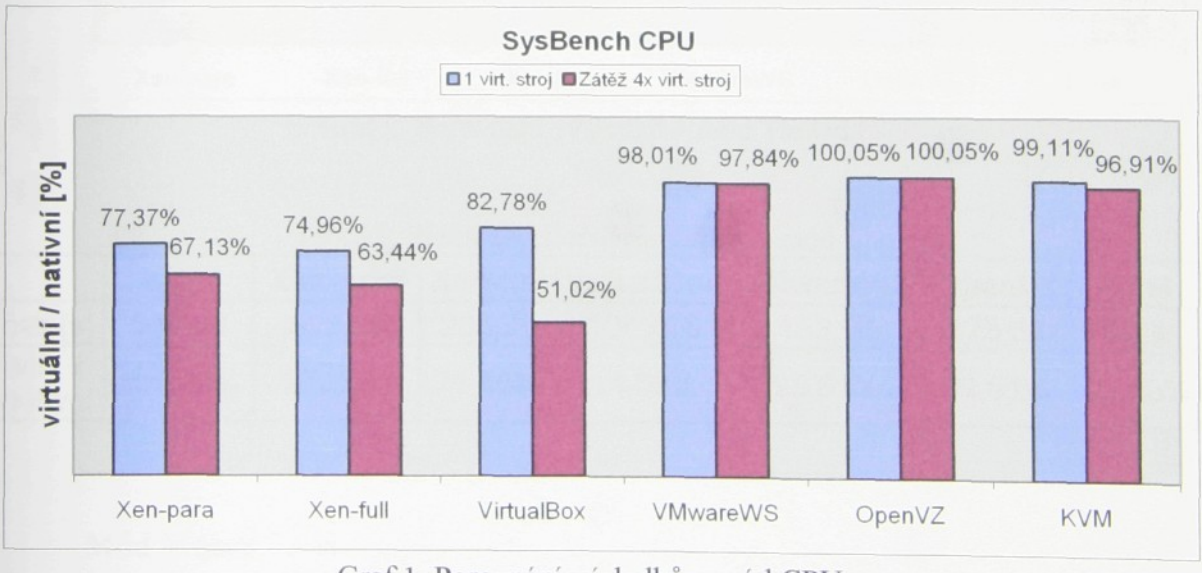
Tab. 11: použité zkratky

Popis	Zkratka
Nevirtualizovaný systém	Nativní
Upravený operační systém spuštěný za pomoci hyperivsoru XEN	XEN-para
Neupravený operační systém spuštěný za pomoci hyperivsoru XEN	XEN-full
Virtualizační nástroj VirtualBox	VirtualBox
Virtualizační nástroj VMware Workstation	VMwareWS
Kontejner operačního systému běžící za pomoci OpenVZ kernelu	OpenVZ

Mód CPU

Graf 1. uvádí výsledky získané nástrojem SysBench v módu CPU. Konkrétní hodnoty jsou v tabulce 12. Trochu za očekáváním zůstal Xen kdy by se dal u paravirtualizace předpokládat lepší výsledek. Naopak překvapením je výsledek KVM a VMware Workstation, které dosáhlo přes 99%, respektive 97% výkonu systému nativního.

Velkým propadem výkonu reagoval na souběh více systémů VirtualBox.



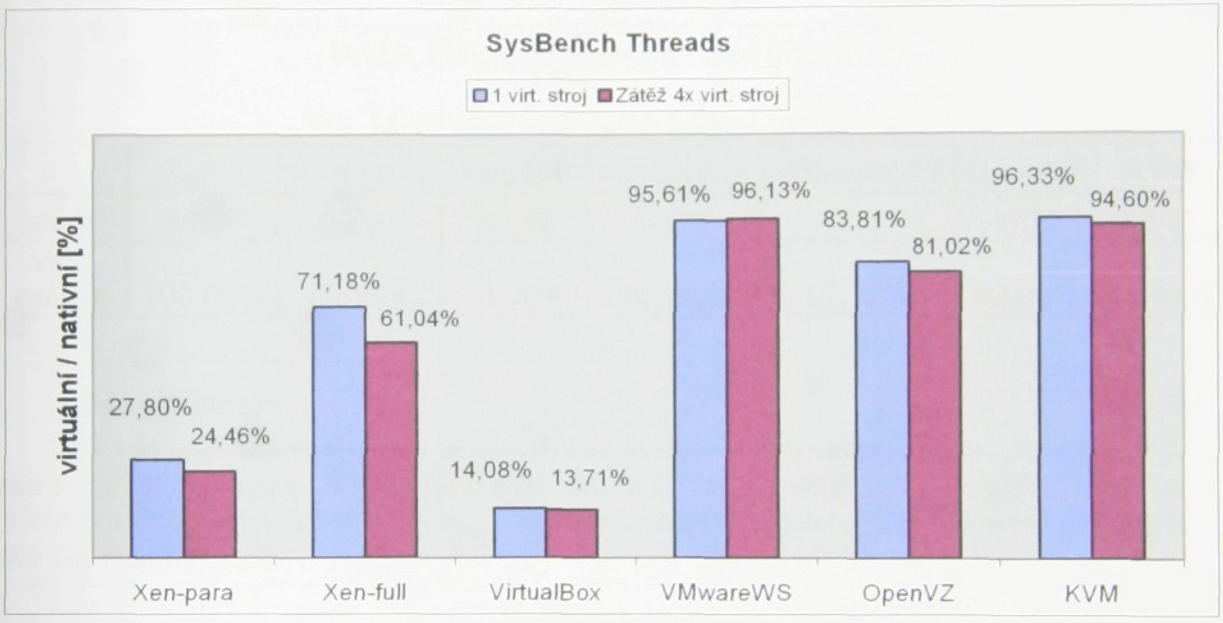
Graf 1. Porovnání výsledků – mód CPU

Tab. 12: Přehled výsledných hodnot – mód CPU

	Nativí	Xen-para	Xen-full	VirtualBox	VMwareWS	OpenVZ	KVM
čas [s]	287,42	371,47	383,42	347,23	293,25	287,27	290,00
virt / nativní	100,00%	77,37%	74,96%	82,78%	98,01%	100,05%	99,11%

Mód Threads

U tohoto testu opět KVM a VMware Workstation dosáhly výborných výsledků. Zajímavý je výsledek OpenVZ, který ztrácí téměř 20% na systém nativní. Příčinu lze hledat v úpravě kernelu, který musí zajišťovat vzájemnou izolaci kontejnerů a rozeznat ve kterém kontejneru proces běží.



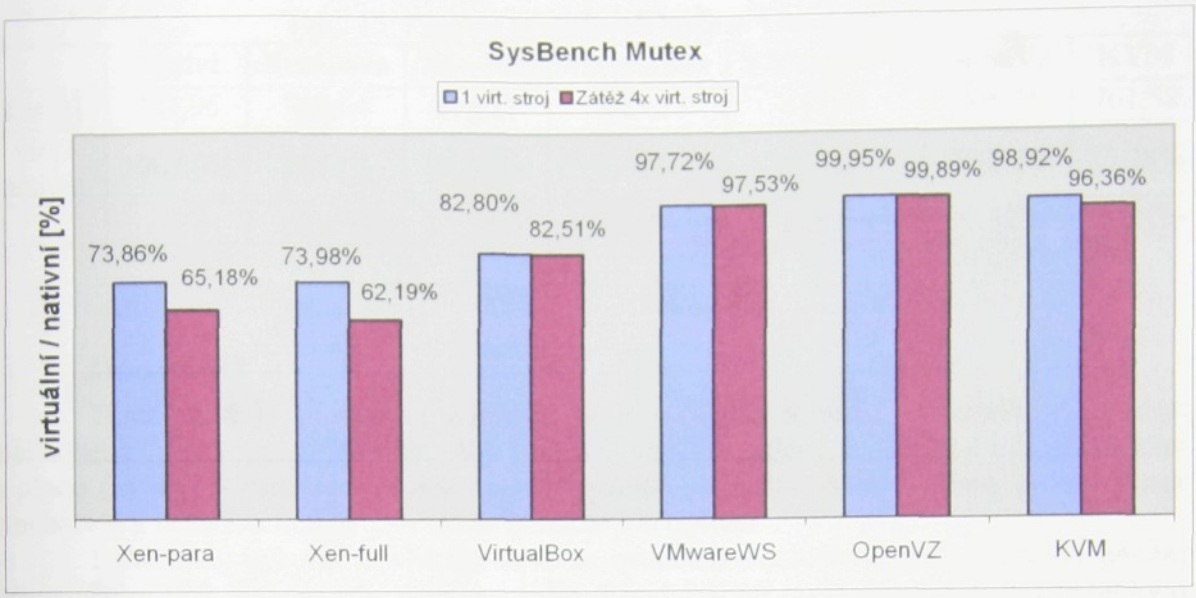
Graf 2. Porovnání výsledků – mód Threads

Tab. 13: Přehled výsledných hodnot – mód Threads

	Nativí	Xen-para	Xen-full	VirtualBox	VMwareWS	OpenVZ	KVM
čas [s]	147,20	529,57	206,79	1045,78	153,96	175,64	152,81
virt. / nativní	100,00%	27,80%	71,18%	14,08%	95,61%	83,81%	96,33%

Mód Mutex

Mód Mutex dopadl vcelku vyrovnaně. Dobrého výsledku tentokrát dosáhl také VirtualBox. Nejhůře dopadl Xen, z výsledku testu Threads a Mutex (graf 2 a 3) je patrné, že Xen je o něco pomalejší při práci s více vlákny. Jen doplním, že test byl nakonfigurován pro běh se 64 vlákny.



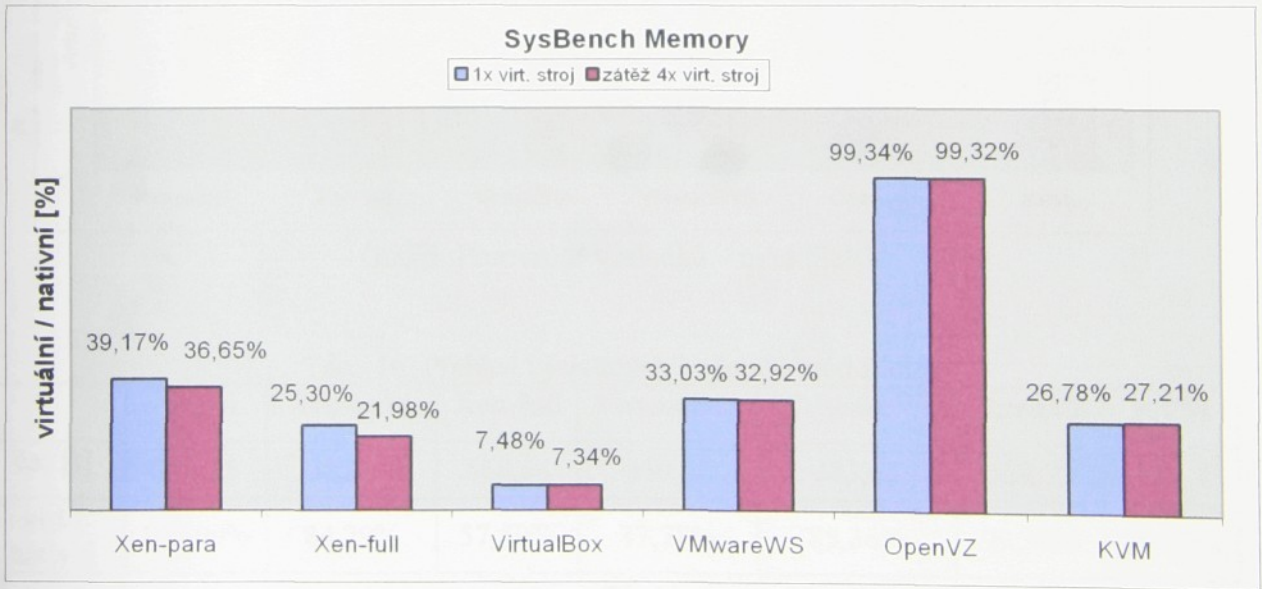
Graf 3. Porovnání výsledků - mód Mutex

Tab. 14: Přehled výsledných hodnot – mód Mutex

	Nativí	Xen-para	Xen-full	VirtualBox	VMwareWS	OpenVZ	KVM
čas [s]	48,77	66,03	65,93	58,91	49,91	48,79	49,31
virt / nativní	100,00%	73,86%	73,98%	82,80%	97,72%	99,95%	98,92%

Mód Memory

Z výsledků módu Memory je patrné, kde je slabé místo virtualizace platformy. Jak je parné z grafu 4, propad i u nejlepšího výsledku je 60 %. Poměrně velký propad výkonu lze přičíst nutnosti asistence hypervisoru při zápisu do paměti. V tomto testu byl simulován zápis 400 GB dat do paměti.



Graf 4. Porovnání výsledků – mód Memory

Tab. 15: Přehled výsledných hodnot – mód Memory

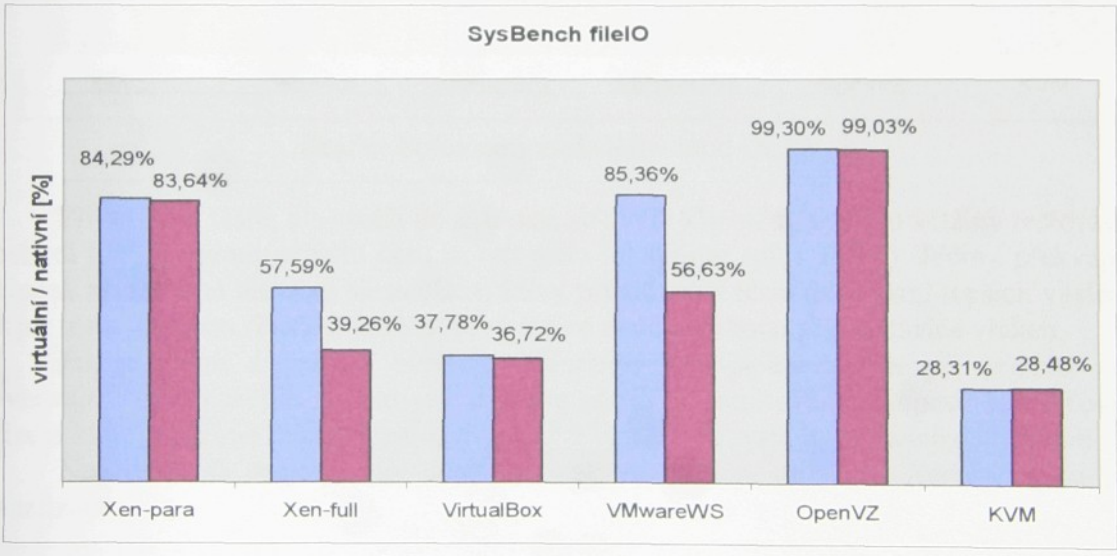
	Nativí	Xen-para	Xen-full	VirtualBox	VMwareWS	OpenVZ	KVM
čas [t]	203,96	520,64	806,27	2727,33	617,44	205,31	761,52
virt / nativní	100,00%	39,17%	25,30%	7,48%	33,03%	99,34%	26,78%
		36,65%	21,98%	7,34%	32,92%	99,32%	27,21%

Mód fileIO

Tento mód je posledním z jednostranně zaměřených tesů. Testovala se rychlost náhodného čtení a zápisu 5GB dat. Aby výsledek nebyl zkreslený, bylo nutné u implementací s plnou (nativní) virtualizací omezit operační paměť na hostitelském systému, protože jinak docházelo k nežádoucímu využití cache u diskových operací.

Jinak vždy výborné KVM zaostává za ostatními implementacemi z důvodu použití pomalého qcow2 souboru jako pevného disku. Vhodnější při optimalizaci pro výkon by bylo využít „raw" blokové zařízení jako u Xenu. Podobně je na tom i VirtualBox, který také využívá virtuální pevný disk v podobě souboru na hostitelském systému. VMware Workstation ukazuje, že pokud je technika použití pevného disku v souboru zvládnuta, je možné dosáhnout i u tohoto typu pevného disku virtuálních strojů výborných výsledků.

VMware WS zaznamenalo výrazný propad výkonu při souběhu více virtuálních strojů.



Graf 5. Porovnání výsledků – mód fileIO

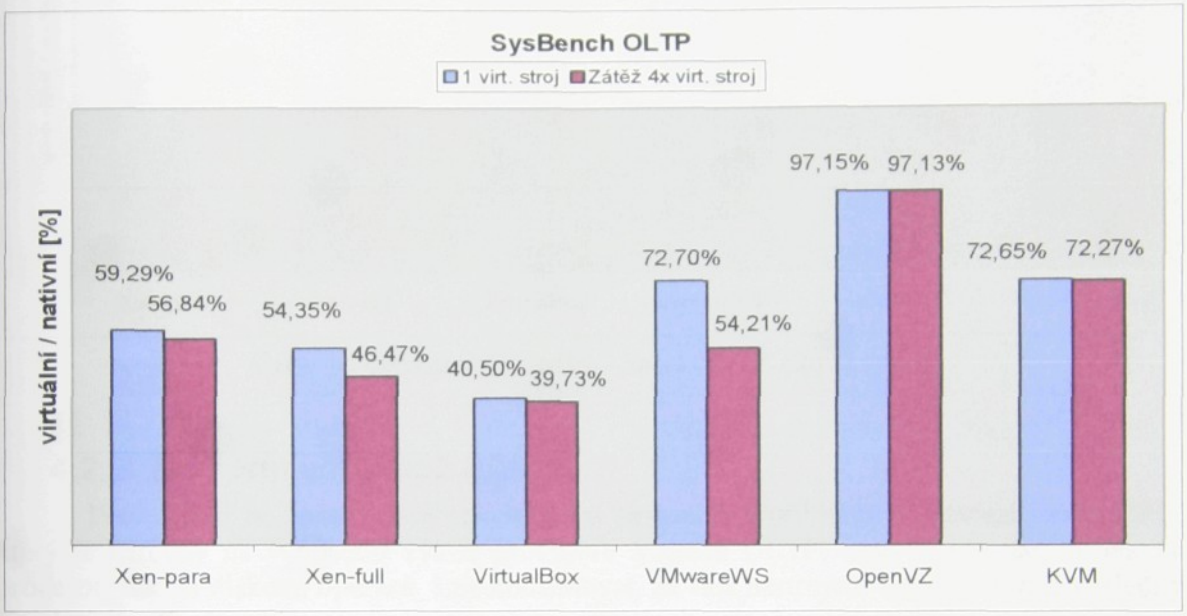
Tab. 16: Přehled výsledných hodnot – mód fileIO

	Nativí	Xen-para	Xen-full	VirtualBox	VMwareWS	OpenVZ	KVM
čas [t]	327,45	388,48	568,64	866,70	383,61	329,75	1156,72
virt / nativní	100,00%	84,29%	57,59%	37,78%	85,36%	99,30%	28,31%

Mód OLTP a OLTP síťový klient

Tento testovací mód simuluje skutečnou zátěž databázového systému, nezaměřuje se na jednu vlastnost, ale testuje praktickou výkonnost celého sytému, která je závislá na více vlastnostech současně. Proto tento test (alespoň podle mého názoru) nejvíce vypovídá o skutečném výkonu systému. I u tohoto testu bylo nutné omezit paměť hostitelského systému, aby se omezila disková cache, kterou hypervisor u nativní virtualizace využívá.

Vysoký propad výkonu při souběhu více virtuálních strojů má VMware Workstation.



Graf 6. Porovnání výsledků – mód OLTP

Při stejném testu, ale zátěži generované síťovým klientem, došlo u většiny testovacích prostředí k podobnému nárůstu času oproti testu lokálnímu (přes 100 s). Jediný překvapivý výsledek přinesl test nástroje VirtualBox, který při síťovém testu dosahoval lepších výsledků než při testu lokálním. Příčinu lze hledat ve velice malém výkonu při práci více vláken.

Jak je patrné z grafu 7, výsledky jednotlivých implementací se při síťovém testu srovnávají, což vypovídá o tom, že vytížení stroje je limitováno propustností síťového zařízení které rozděluje dotazy mezi hostitelský a hostovaný systém, případně datové sítě.

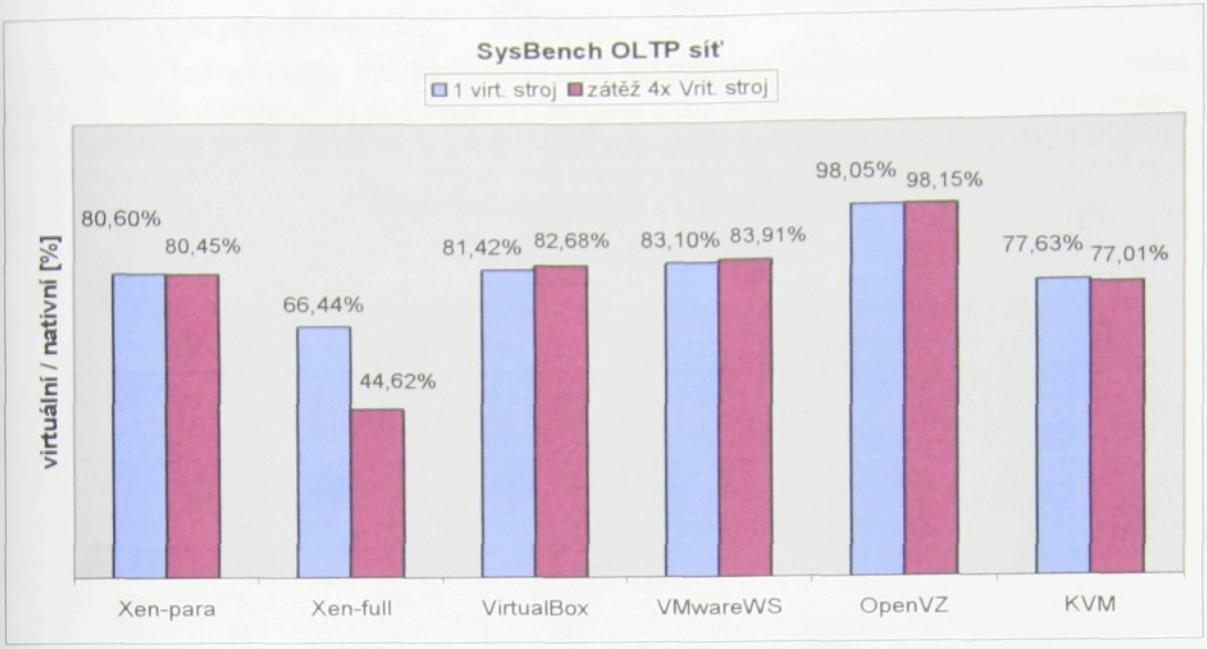
Negativně se projevil vliv souběhu více virtuálních strojů na Xenu v modu plné virtualizace.

Tab. 17: Přehled výsledných hodnot – mód OLTP

	Nativí	Xen-para	Xen-full	VirtualBox	VMwareWS	OpenVZ	KVM
čas [t]	226,90	382,69	417,47	560,17	312,12	233,57	312,34
virt / nativní	100,00%	59,29%	54,35%	40,50%	72,70%	97,15%	72,65%

Tab. 18: Přehled výsledných hodnot – mód OLTP síť

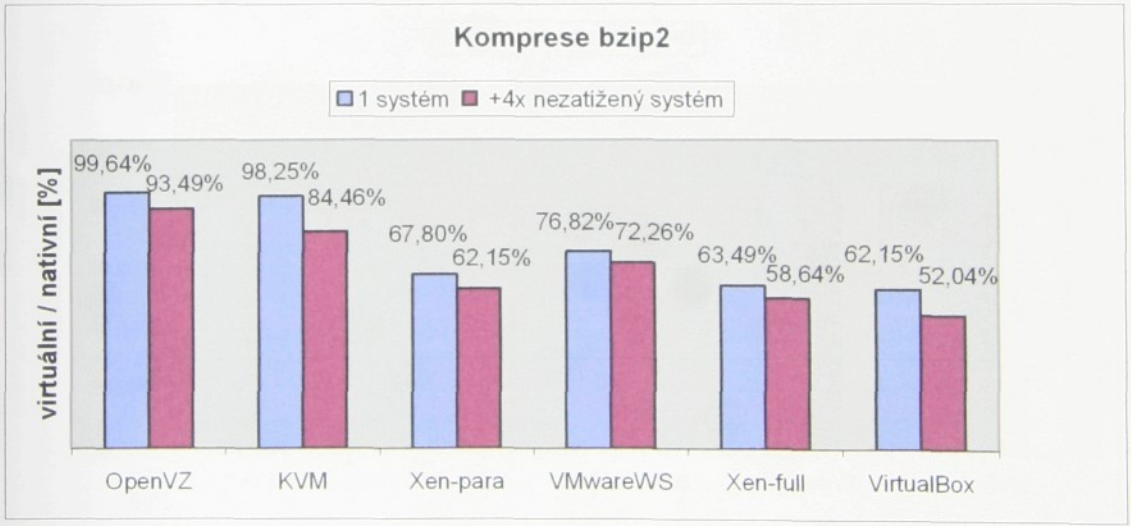
	Nativí	Xen-para	Xen-full	VirtualBox	VMwareWS	OpenVZ	KVM
čas [t]	372,18	461,75	560,17	457,11	447,88	379,59	479,45
virt / nativní	100,00%	80,60%	66,44%	81,42%	83,10%	98,05%	77,63%



Graf 7. Porovnání výsledků – mód OLTP sítová zátěž

4.2.2 Komprimace bzip2 (pbzip)

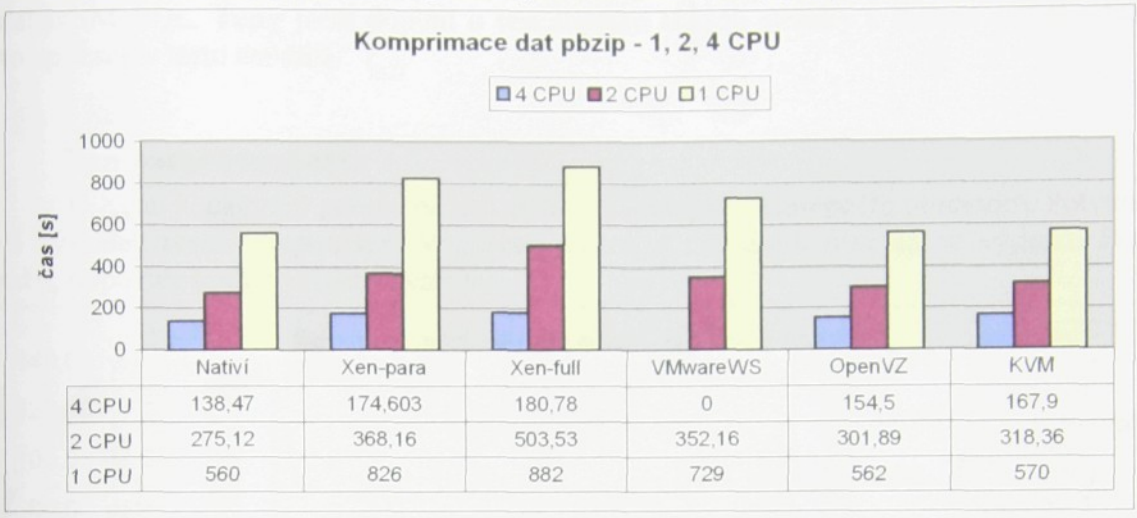
Předpoklad je, že výsledek tohoto testu by měl být přibližně v rozmezí módu CPU, který je náročný na výpočetní výkon procesoru a módu OLTP, který klade nároky jak na procesor, tak na diskové operace. Lze konstatovat, že test nástrojem bzip2 potvrdil výsledky benchmarku SysBench.



Graf 8. Porovnání výsledků komprese nástrojem bzip2

Test více procesorů

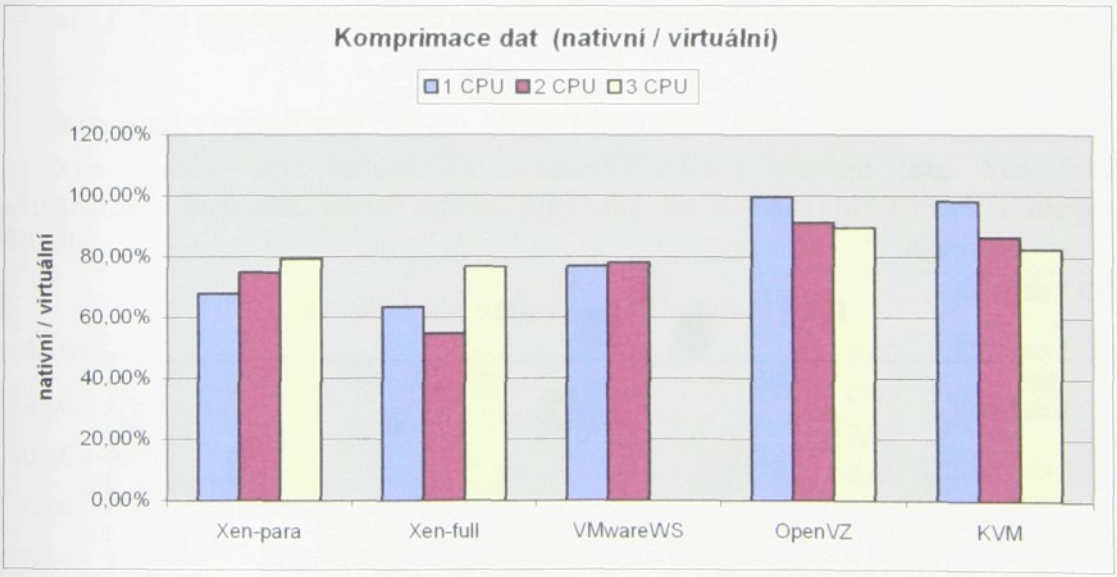
Tento test ukáže na důležitou schopnost jednotlivých technologií využívat pro práci více procesorů. Za ideálních podmínek by se měla doba komprese zkrátit počtem procesorů, které virtuálnímu stroji přidělím. V grafu 9 jsou zobrazeny časy pro jednotlivé počty CPU.



	Nativí	Xen-para	Xen-full	VMwareWS	OpenVZ	KVM
4 CPU	138,47	174,603	180,78	0	154,5	167,9
2 CPU	275,12	368,16	503,53	352,16	301,89	318,36
1 CPU	560	826	882	729	562	570

Graf 9. Porovnání výsledků komprese nástrojem bzip2 na 1, 2, 4 CPU

Graf 10 je ještě zajímavější, protože ukazuje, jak s měnícím se počtem procesorů se mění poměr systému nativního k hostovanému. Z výsledku je patrné, že Xen je při práci s více procesory výborný.



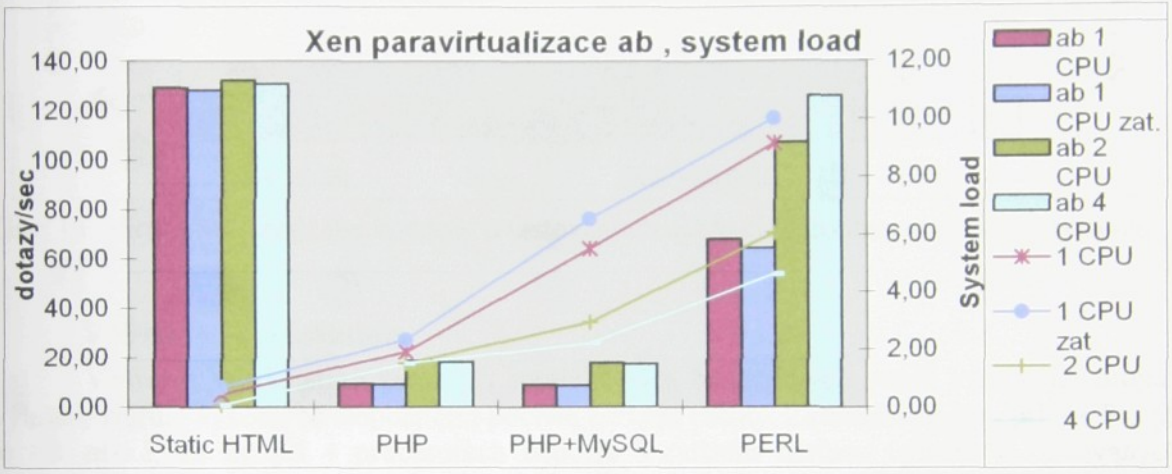
Graf 10. Porovnání poměru nativního ku virtuálnímu systému s měnícím se počtem CPU

4.2.3 Test Apache http serveru

Jako poslední sadu srovnávacích testů jsem připravil testy reálných systémů běžně provozovaných na web serverech. Testoval jsem aplikaci phpAlbum, která reprezentuje aplikace psané v PHP bez SQL databáze a oblíbený redakční systém Drupal, který využívá databázi MySQL. Testy jsem doplnil o test statické HTML stránky a skriptu v Perlu, které tyto aplikace v testu neodliší.

Xen paravirtualizace

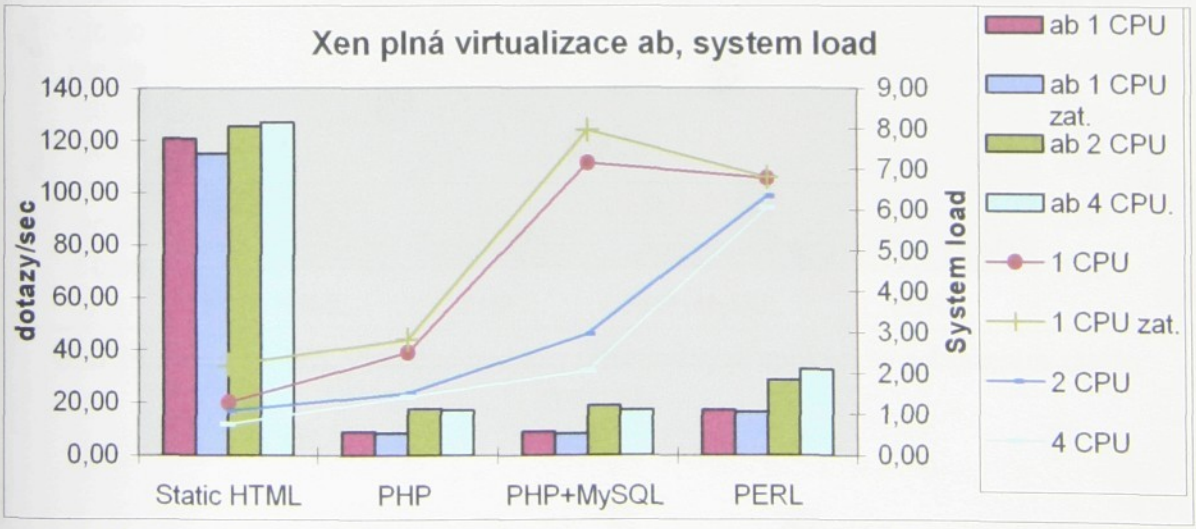
U Xenu je na první pohled patrný nárůst výkonu při větším počtu procesorů. Potvrzuje to i výsledek testu komprimace. Výpočetní výkon 4 procesorů není až na výjimku Perlu využit, odpovídá tomu i vytížení systému.



Graf 11. Xen výsledné hodnoty u testovaných aplikací se zobrazením zátěže systému

Xen plná virtualizace

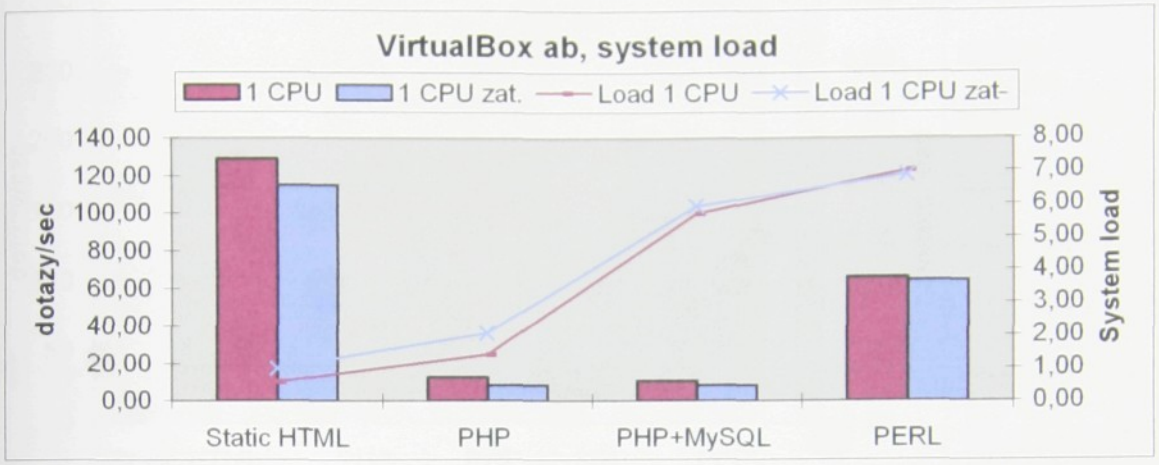
Xen v módu plné virtualizace vykazuje podobné chování jako Xen v módu paravirtualizace. Jinak neměřitelné zatížení systému u statických HTML stránek je zde patrné při 1 CPU.



Graf 12. Xen plná virtualizace - výsledné hodnoty u testovaných aplikací se zobrazením vytížení systému

VirtualBox

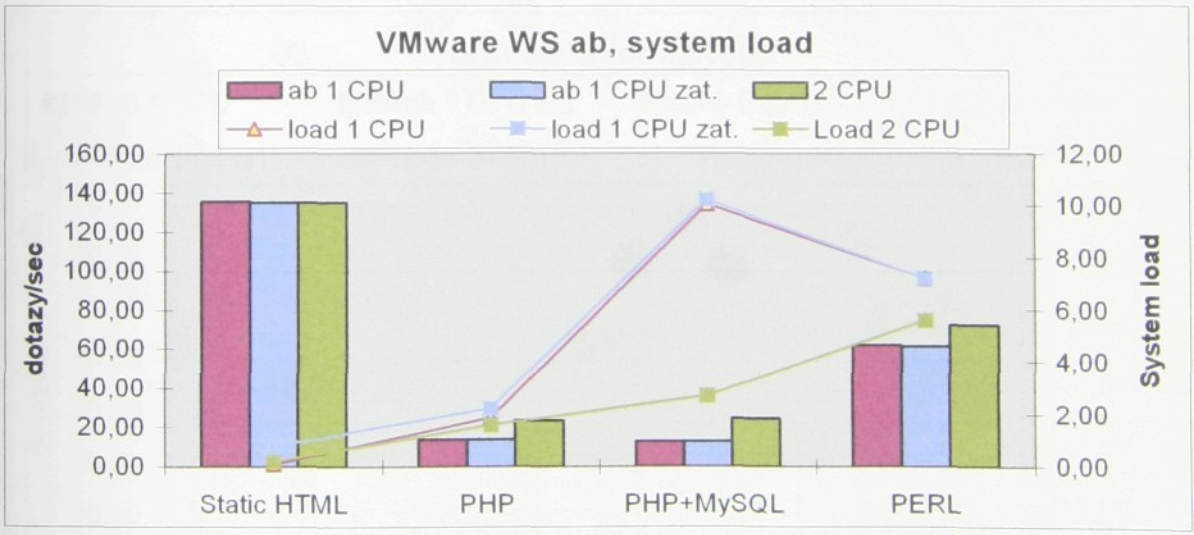
V současné době není u VirtualBox implementována podpora SMP a tak test proběhl jen pro 1 procesor. Hostovaný systém reaguje poměrně nepříznivě na souběh více virtuálních strojů. Systém je sice i s 1 CPU schopen dosáhnout relativně dobrého výsledku, nicméně bez více procesorů je systém při větší zátěži přetížen.



Graf 13. VirtualBox výsledné hodnoty u testovaných aplikací se zobrazením zátěže systému

VMware Workstation

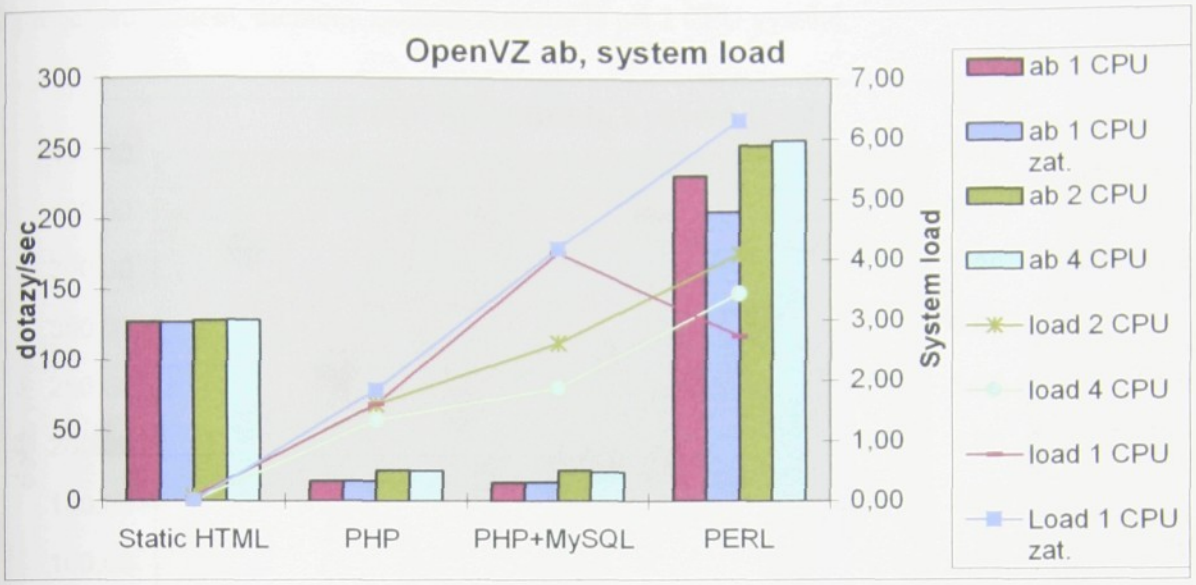
VMware Workstation nemá při souběhu více hostovaných strojů výraznější propad výkonu. Nárůst výkonu se stoupajícím počtem CPU je patrný. Zatížení systému při 2 CPU již je tak nízké, že by při 4 procesorech nedošlo k dalšímu nárůstu výkonu u testovaných aplikací.



Graf 14. VMware WS, výsledné hodnoty u testovaných aplikací se zobrazením zátěže systému

OpenVZ

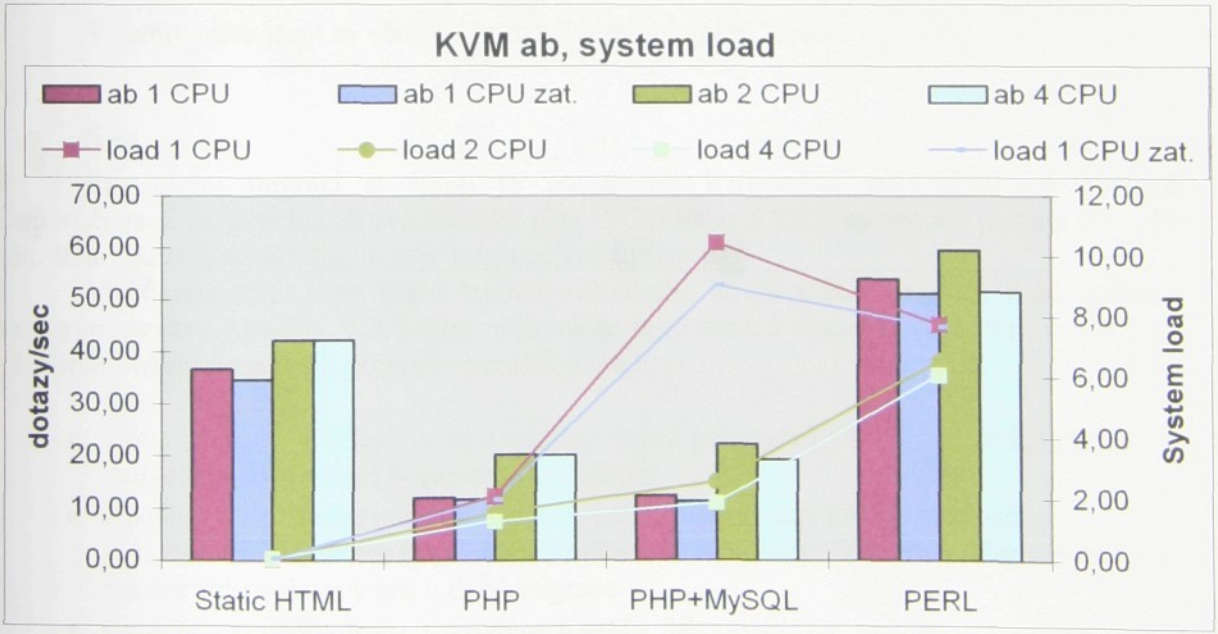
OpenVZ nereaguje na souběh více hostovaných strojů propadem výkonu.



Graf 15. OpenVZ, výsledné hodnoty u testovaných aplikací se zobrazením zátěže systému

KVM

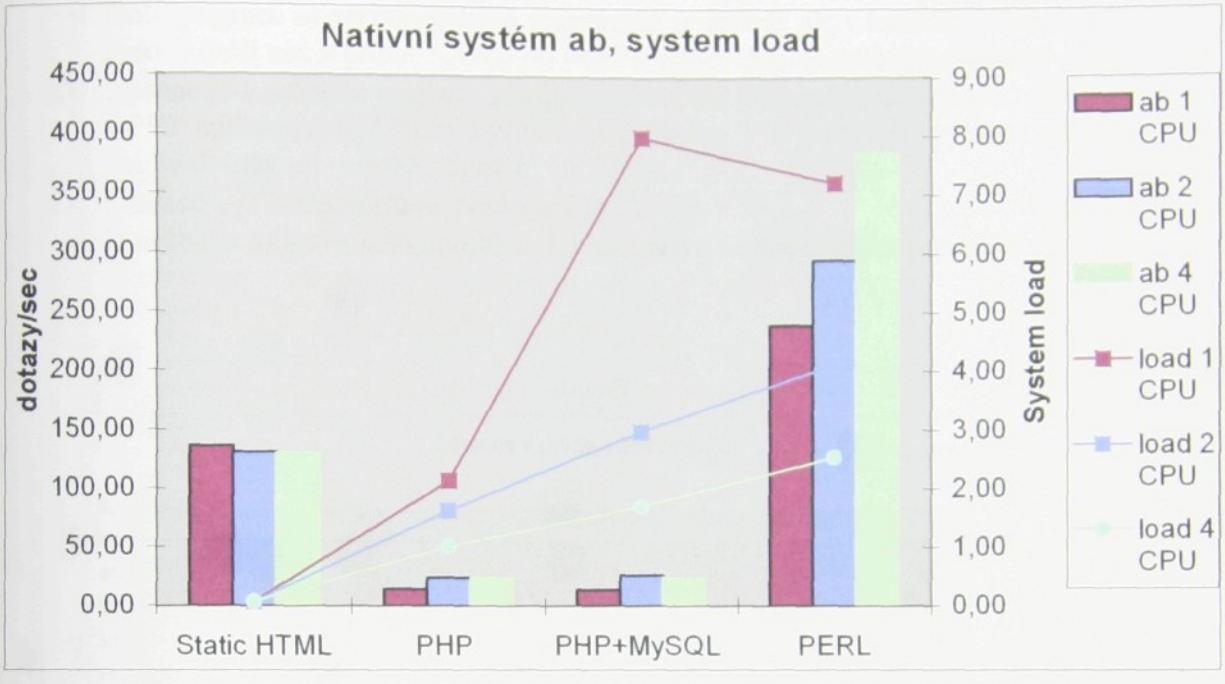
V grafu 16 jsou u KVM patrné 2 vlastnosti. Poměrně negativně reaguje na souběh více hostovaných systémů. Při 2 CPU je patrný poměrně vysoký nárůst výkonnosti spolu poklesem zátěže. 4 procesory v tomto případě nepřináší další zlepšení výkonu.



Graf 16. KVM, výsledné hodnoty u testovaných aplikací se zobrazením zátěže systému

Nativní systém

Pro porovnání s virtuálními stroji udávám hodnoty získané u nativního systému. Zde je patrný vysoký nárůst odbavených dotazů u Perl skriptu. Počet odbavených dotazů se již od 2 procesorů nemění, nicméně zatížení systému je při 2 CPU vysoké.



Graf 17. Výsledné hodnoty testovaných aplikací se zobrazením vytížení systému

4.2.4 Živá migrace

V tomto testu jsem se věnoval pouze 2 implemtacím, Xenu a OpenVZ.

Xen

Pro živou migraci u Xenu je nutné mít k dispozici nějaké síťové úložiště. Doporučovaná jsou rychlá síťová úložiště přes iSCSI nebo SAN. Pro test ale postačí NFS. Po několika zásazích do konfigurace je migrace velice snadná.

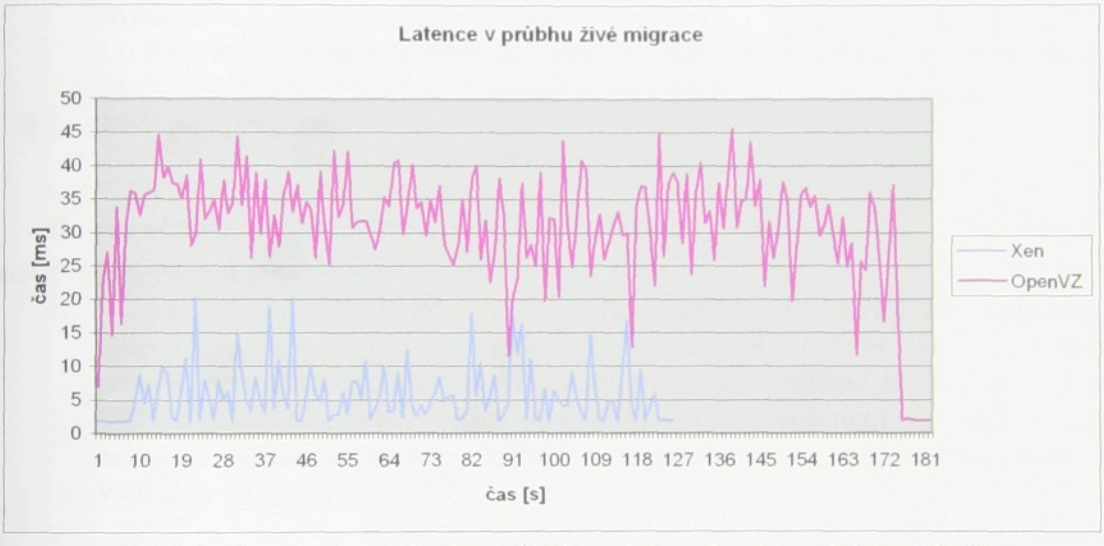
V průběhu testu jsem měřil latenci virtuálního stroje příkazem ping, dobu migrace a odezvu serveru Apache. V Průběhu migrace je také možné sledovat výpisem „xm li", jak cílovému virtuálnímu stroji přibývá operační paměť.

- Doba migrace se pohybovala v rozmezí velmi pěkných 1,75 minuty do 2,5 minut s ohledem na vytížení hostovaného systému
- Apache je v průběhu migrace schopný odbavovat dotazy klientů, propad výkonuu PHP aplikace (phpAlbum) byl na 40 % výkonu. Vliv na pokles výkonu ale má ale veliký datový tok v době migrace
- Latence v průběhu testu je uvedena v grafu 18
- V průběhu migrace nebyl problém být přihlášen v migrovaném prostředí

OpenVZ

V případě OpenVZ je migrace jednodušší v tom, že není potřeba žádné síťové úložiště, protože migrace probíhá za pomoci protokolu ssh. Praktické je v tomto případě použití ssh klíčů pro automatické přihlášení.

- Doba migrace závisí na velikosti virtuálního prostředí, ale v každém případě je řádově delší než v případě Xenu při použití sdíleného úložiště (12,5 minut)
- Apache je v průběhu migrace schopný odbavovat dotazy klientů, propad výkonu u PHP aplikace (phpAlbum) byl na 30 % výkonu. Vliv na pokles výkonu má ale veliký datový tok v době migrace
- Latence v průběhu testu je uvedena v grafu 18
- V průběhu migrace nebyl problém být přihlášen v migrovaném prostředí



Graf 18. Latence naměřená v průběhu migrace hostovaných systémů.

4.3 Vyhodnocení výsledků

V tomto oddíle se pokusím shrnout výsledky získané z testů v předcházejícím oddíle a formulovat doporučení pro vhodné použití jednotlivých technologií.

4.3.1 OpenVZ

OpenVZ se prezentovala výbornými výsledky ve všech testech, důvodem je princip na kterém pracuje, nejedná se o virtuální systém, ale o virtualizaované prostředí.

Při testech se prokázalo:

+ Vysoký výkon prakticky ve všech módech testu blížící se systému nativnímu
+ Nízká ztráta výkonu při běhu více kontejnerů
+ V testech reálných systémů vykazoval výborné výsledky
+ Snadné nasazení
+ Snadná živá migrace s využitím protokolu ssh
- Se vzrůstajícím počtem CPU dochází lehce k poklesu výkonnosti oproti systému nativnímu
- Málo propracovaný management zařízení, například přiřazení blokového zařízení do virtuálního prostředí vyžaduje pracnější nastavování
- Pomalá živá migrace

Doporučení

OpenVZ lze doporučit pro:

- Virtualizaci serverů – Pro vysoký výkon spolu s podporou živé migrace lze OpenVZ doporučit pro prostředí, kde je potřeba větší množství Linuxových serverů zprostředkujících potřebné služby jako je například HTTP, SQL, CUPS, SVN a podobně. Další použití je například při pronajímání virtuálních serverů nebo ve webhostingových službách. Doporučení pro testované aplikace jsou v tabulce 19.

Při konfiguraci virtuálních prostředí je potřeba věnovat větší pozornost konfiguraci využití systémových zdrojů – kvót. Pro začínající uživatele je vhodné použít k ovládání některý z webových nástrojů pro management – například WebVZ je poměrně pěkný a jednoduchý nástroj.

OpenVZ na mě působí stabilním dojmem, při testování jsem se nesetkal s žádnými problémy.

Tab. 19: Doporoučení pro testované aplikace

Aplikace	Doporučení
Statický HTML	Postačí stroj 1 CPU, 1 RAM pro více webů
PHP Aplikace (phpAlbum)	Velmi vytížené servery 4 CPU, 2 GB RAM Jinak postačí 2 CPU 1 GB RAM
PHP + MySQL (Drupal)	Velmi vytížené servery 4 CPU, 2 GB RAM Jinak postačí 2 CPU 1 GB RAM

4.3.2 Xen

Paravirtualizace

Xen v testech SysBench vykazoval nevyrovnané výsledky. Při testu komprimace a při testech reálných systémů Xen prokázal vzrůstající výkonnost.

Při testech se prokázalo:

+ Výborně reaguje na zvyšující se počet CPU. Při 4 CPU, zaostává proti OpenVZ jen o několik odbavených dotazů za vteřinu
+ Nízký propad výkonu při souběhu více hostovaných systémů
+ Propracovaný management zařízení
+ Snadné nasazení hostovaného systému
+ Rychlá a snadná živá migrace (článek 4.2.4)
- Nízký výkon pokud systém pracuje s 1 CPU

Doporučení

XEN lze doporučit pro:

- Virtualizaci serverů - Xen je typickým nástrojem pro virtualizaci serverů (HTTP, SQL, NFS). Vedle možnosti živé migrace také například použitím většího množství virtuálních počítačů nedochází k výraznému zpomalení výkonu. Jeho hlavní výhodou je stabilita, cena a integrace v linuxových distribucích v podobě xenkernelu. Pro pohodlné vytváření virtuálních počítačů a jednoduchý management je vhodné použít nástroj Virtual Machin Monitor.
- Pokud je to možné, vždy použít mód paravirtualizace, je výkonnější

Tab. 20: Doporoučení pro testované aplikace Xen paravirtualizace

Aplikace	Doporučení
Statický HTML	Postačí stroj 1 CPU, 1 RAM pro více webů
PHP Aplikace (phpAlbum)	Velmi vytížené servery 4 CPU, 2 GB RAM Jinak postačí 2 CPU 1 GB RAM
PHP + MySQL (Drupal)	Minimálně lze doporučit 4 CPU, 2 GB RAM

Plná virtualizace

Mód plné virtualizace proti paravirtualizaci výkonnostně zaostává. Zajímavá je vlastnost, kdy se zvyšujícím počtem CPU se zvyšuje příznivě výkonnostní poměr ku výkonu referenčnímu.

Při testech se prokázalo:

+ Mód plné virtualizace lze doporučit pouze pro systémy proprietární, které neumožňují paravirtualizaci. (MS Windows).
- Ani při více procesorech nedosáhne výkon webového serveru úrovně paravirtualizace.

Tab. 21: Doporučení pro testované aplikace Xen plná virtualizace

Aplikace	Doporučení
Statický HTML	Postačí stroj 1 CPU, 1 RAM pro více webů
PHP Aplikace (phpAlbum)	Vytížené servery 4 CPU, 2 GB RAM
PHP + MySQL (Drupal)	Minimálně 4 CPU, 2 GB RAM

4.3.3 KVM

Neměl jsem představu o výkonnosti KVM, a tak jsou pro mě výsledky trochu překvapivé. Použití kernelu jako hypervisoru se ukazuje jako velice efektivní i ve spojení s nativní virtualizací. Především výkon procesoru je prakticky roven nativnímu systému.

Při testech se prokázalo:

+ Vysoký výkon při práci s 1 CPU
+ Malý propad výkonu při běhu více hostovaných systémů
+ Vysoký výkon při testovaní reálných aplikací
- Se vzrůstajícím počtem CPU dochází lehce k propadu výkonu
- Nižší rychlost při diskových operacích
- Problémy se stabilitou

Doporučení

KVM lze doporučit pro:

- Virtualizaci serverů
  Předpokladem jsou výborné výsledky v testech jak při práci s 1 CPU tak více CPU. Již je implementována ceněná podpora živé migrace a hostované stroje lze spouštět například pouze s podporou VNC

- Virtualizaci pracovních stanic
  Výsledky v testu ukazují na vysoký výkon při práci s 1 CPU, existují rozšíření pro podporu virtuálního OpenGL zařízení. Je splněna podmínka, že Desktop by se měl chovat „svižně".

Pro nízký výkon při použití qcow2 virtuálního disku určitě doporučuji použít skutečné blokové zařízení.

Tab. 22: Doporučení pro testované aplikace – KVM

Aplikace	Doporučení
Statický HTML	Postačí stroj 1 CPU, 1 RAM pro více webů
PHP Aplikace (phpAlbum)	Lze doporučit 2 CPU, nedochází k nárůstu výkonu, pouze klesá zátěž systému
PHP + MySQL (Drupal)	Lze doporučit 2 CPU, nedochází k nárůstu výkonu, pouze klesá zátěž systému

4.3.4 VMware Workstation

VMware Workstation ve všech provedených testech dosahoval pěkných výsledků. Tato aplikace může zastoupit VMware Server, volně šířenou aplikaci pro nekomerční a profesionální použití.

Při testech se prokázalo:

+ Vysoký výkon při práci s 1 i 2 CPU
+ Vysoký výkon u testovaných aplikací
+ Podpora Direc3D
- Propad výkonu při běhu více hostovaných systémů

Doporučení

VMware Workstation lze doporučit pro:

- Virtualizace Desktopu – Jak je patrné z názvu, VMware Workstation je produkt určený pro virtualizaci Desktopu. Produkt není zaměřen na intenzivní poskytování web služeb a aplikací, které jsem testoval.

Tab. 23: Doporoučení pro testované aplikace – KVM

Aplikace	Doporučení
Statický HTML	Postačí stroj 1 CPU, 1 RAM pro více webů
PHP Aplikace (phpAlbum)	Lze doporučit 2 CPU
PHP + MySQL (Drupal)	Nelze doporučit

VMware Workstation nabízí kromě výborného výkonu a propracovaného grafického ovládání také spoustu dalších funkcí, jako je například tvorba nahrávek, podpora Direct3D, bohaté konzolové nástroje, nástroje umožňující migraci mezi skutečným a virtuálním strojem, tvorba virtuálních sítí, atd. Celkově VMware Workstation působí velice pěkným dojmem a za cenu 189$ je zajímavou volbou pro firemní prostředí.

4.3.5 VirtualBox

Nejhůře si v testu vedl VirtualBox, ale rozdíl při práci s 1 CPU proti Xenu v módu plné virtualizace je pouze v řádu procent. VirtualBox má již implementovánu podporu virtualizačních rozšíření moderních procesorů, proto je horší výsledek v testu trochu překvapení. Příčinou je velice pomalá práce s pamětí a malý výkon při prácí více vláken. V tomto případě se řešení hledá těžko, doporučit lze otestovat jinou verzi VirtualBoxu nebo použít jiný hardware.

Při testech se prokázalo:

- Celkové nízký výkon v testech
- Propad výkonu při běhu více hostovaných systémů
- Není podpora SMP

Doporučení

VirtualBox lze doporučit pro:

- Virtualizace Desktopu – VirtualBox je typickým nástrojem pro domácího uživatele, který potřebuje využit virtuální počítač například pro testování nebo provozování aplikací, které na jeho standardním systému nechodí. Instalace je jednoduchá a grafické ovládání poskytuje komfortní funkcionalitu. Uživatel se nemusí zabývat problémy, jako je vytváření síťového rozhraní nebo konfigurace kernelu. VirtualBox se choval po dobu testování stabilně a mohu ho jen doporučit.

Tab. 24: Doporoučení pro testované aplikace – VirtualBox

Aplikace	Doporučení
Statický HTML	Postačí stroj 1 CPU, 1 RAM pro více webů
PHP Aplikace (phpAlbum)	Nelze doporučit
PHP + MySQL (Drupal)	Nelze doporučit

KVM, VirtualBox, VMwareWorkstation pro svůj běh užívají moduly jádra. Pokud je modul některé implementace instalován, jiná implementace se nespustí, je nejprve nutné jaderné moduly deaktivovat.

4.4 Přínosy a použití virtualizace

Aplikační vizualizace

Výhody Aplikační virtualizace

- Za největší přínos aplikační virtualizace lze označit multiplatformnost, portabilitu. Program vytvořený ve Windows lze spustit pod Linuxem a naopak.
- Vyšší ochrana operačního systému před špatným kódem, izoluje aplikaci od operačního systému, umožňuje běh programů, které potřebují vysoká oprávnění.
- Umožňuje běh nekompatibilních programů (simulace souborů, registru).
- Snazší migrace mezi operačními systémy.

Nevýhody Aplikační virtualizace

- Nevýhodou je nutnost přítomnosti virtualizační vrstvy
- Pomalejší běh programu nad touto vrstvou než nativních aplikací
- Vyšší nároky na systémové prostředky

Emulace / Simulace hardware

Výhody Emulace

- Největší výhodou je možnost provozovat virtuální stroj jiného typu procesoru, než na kterém je provozován – platformě nezávislé
- Emulace starého hardware, nebo hardware, který je teprve ve vývoji
- Možnost provozovat neupravené operační systémy

Nevýhody Emulace

- Emulátory jsou obecně pomalé

Nativní virtualizace a plná virtualizace

Výhody nativní a plné virtualzace

- Rychlejší než emulace
- Propracovanější funkcionalita
- Využití technologií pro podpor vizualizace
- Možnost provozovat neupravené operační systémy

Nevýhody nativní a plné virtualizace

- Možnost virtualizace jen určitého typu CPU – na kterém je provozován
- Nižší výkon

Paravirtualizace

Výhody nativní a plné virtualizace

- Vysoký výkon
- Využití v oblasti serverových technologií
- Živá migrace

Nevýhody nativní a plné vizualizace

- Nutná úprava hostovaného operačního systému

Virtualizace na úrovni operačního systému

Výhody virtualizace na úrovni os

- Jednoduchost
- Rychlé a snadné nasazení
- Vysoký výkon
- Živá migrace
- Využití v oblasti serverových technologií

Nevýhody virtualizace na úrovni operačního systému

- Nutná úprava kernelu
- Lze provozovat pouze Linuxové systémy

5 Závěr

Teoretické principy virtualizace a virtualizačních technik jsem nastudoval v literatuře, seznámil jsem se i s běžně používanými implementacemi. Získané poznatky jsem shrnul v první teoreticky zaměřené části práce. Každou z 12 zmíněných implementací jsem si i prakticky vyzkoušel.

V praktické části jsem se zaměřil na běžněji používané virtualizační nástroje Xen, VMware Workstation, VirtalBox, KVM, OpenVZ, které jsem podrobil sérii 24 zátěžových testů. Pro každou z implementací jsem připravil totožné prostředí, aby získané hodnoty byly porovnatelné. Výsledky jsou shrnuty v kapitole 4.

První série „syntetických" testů nástrojem SysBench ukázala na silné a slabé stránky jednotlivých implementací virtuálních systémů. Podle grafu 1 – 5 si nejlépe vedl v testech systém OpenVZ, KVM a VMware Workstation.

Další zátěžové testy jsem adresoval do oblasti reálného světa vzhledem k jejich vyšší vypovídací hodnotě. Nejprve jsem otestoval, jak jednotlivé systémy dokáží zvládat zatížení často používaného databázového systému MySQL a jak si poradí s komprimací velkého souboru. Výsledné hodnoty ukazuje graf 6 - 10.

V poslední části testů jsem testoval Apache http server, který byl použit pro poskytování reálných aplikací (Drupal, phpAlbum) a pro testy které jsem sám připravil (statická HTML stránka, Perl skript). U hostovaných systémů jsem měřil množství dotazů na aplikaci, které je server schopný obsloužit, spolu se zatížením systému. Získané hodnoty u jednotlivých aplikací jsou zobrazeny a popsány v článku 4.2.3. Pro ověření chování funkce živé migrace jsem provedl test u Xenu a OpenVZ s měřením vybraných hodnot.

Na závěr jsem v podkapitolách 4.3 a 4.4 shrnul poznatky k jednotlivým implementacím a stanovil doporučení pro použití jak obecně, tak pro testované aplikace.

V testech se prokázal předpoklad, že hardwarová podpora virtualizace u moderních procesorů přinesla výrazné zvýšení výkonu v oblasti nativní a plné virtualizace, především projekt KVM a VMware Workstation dosahují výborných výsledků. Na druhou stranu se prokázalo, že paravirtualizace s pomocí hypervisoru XEN je efektivní při práci s více procesory a souběhu více hostovaných systémů. Podobně OpenVZ pracující na principu virtualizace na úrovni operačního systému je podle předpokladu velice efektivní.

Tato práce umožní potenciálnímu uživateli lépe se zorientovat v prostředí virtuálních strojů, a v případě, že chce sám vyzkoušet nebo používat některý z virtualizačních nástrojů, mu poskytne užitečné doporučení pro výběr vhodného prostředí. Výsledky získané při zatěžovacích testech hostovaných systémů jsou prakticky použitelné pro potřeby nasazení virtuálních systémů v prostředí web hostingu, nebo například v prostředí podnikových sítí, ale i pro běžného uživatele. Ukazují na cílové skupiny pro použití jednotlivých implementací i technologií, jejich možnosti a napovídají, jak mají být systémy dimenzovány pro daný typ aplikace.

Virtualizace je trendem dnešních informačních technologií a zabývat se jí v této práci bylo velice zajímavé.

Seznam použité literatury:

[1] AMD. Industry Leading Virtualization Platform Efficiency
http://www.amd.com/us-en/Processors/ProductInformation/0,,30_118_8796_14287,00.html

[2] Amit Singh. An Introduction to Virtualization.
http://www.kernelthread.com/publications/virtualization/

[3] Fabrice Bellard. QEMU Emulator User Documentation.
http://bellard.org/qemu/qemu-doc.html

[4] GNU dokument. Wikipedia The Free Encyclopedia, Virtualization.
http://en.wikipedia.org/wiki/Virtualization

[5] Christopher Clark. Xen User's Manual
http://www.cl.cam.ac.uk/research/srg/netos/xen/readmes/user/user.html

[6] Inel, Intel® Virtualization Technology
http://www.intel.com/technology/virtualization/

[7] Kurill Kolyskin. Virtualization in Linux
http://download.openvz.org/doc/openvz-intro.pdf

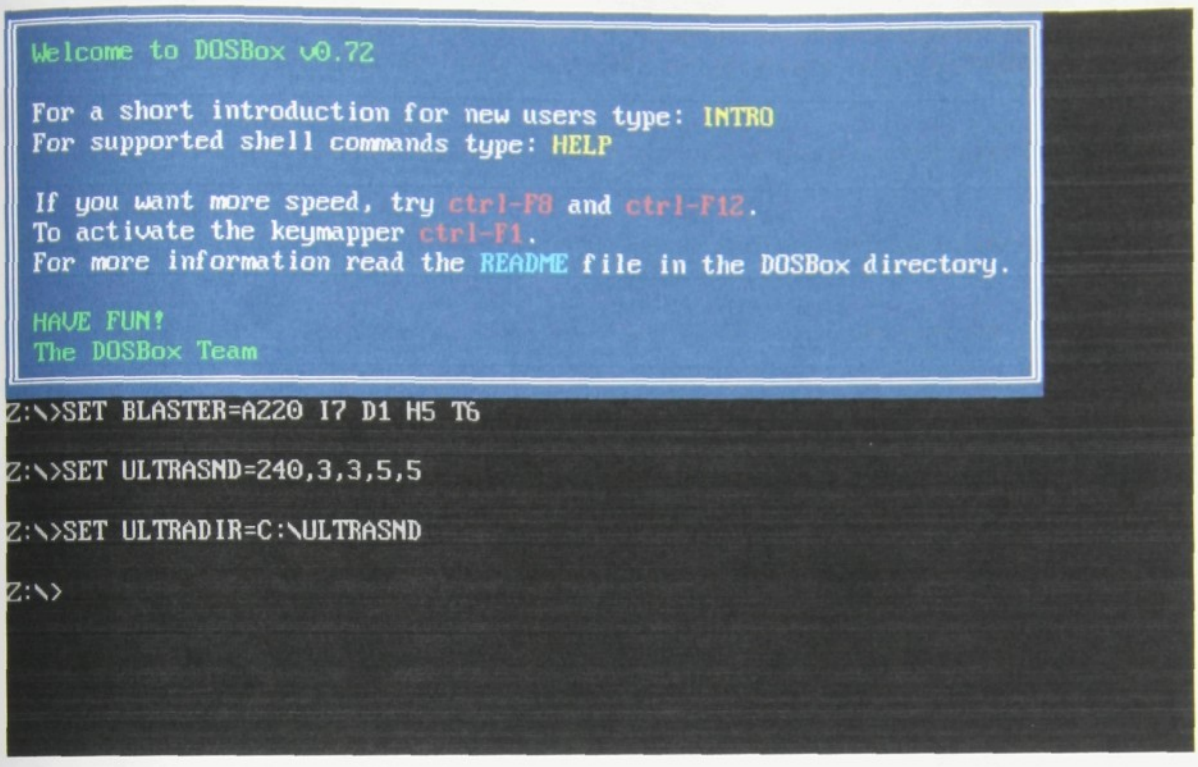
[8] Qumranet. KVM Whitepaper.
http://kvm.qumranet.com/kvmwiki/Documents

[9] Tim Jones. Virtual Linux.
http://www-128.ibm.com/developerworks/library/l-linuxvirt/index.html

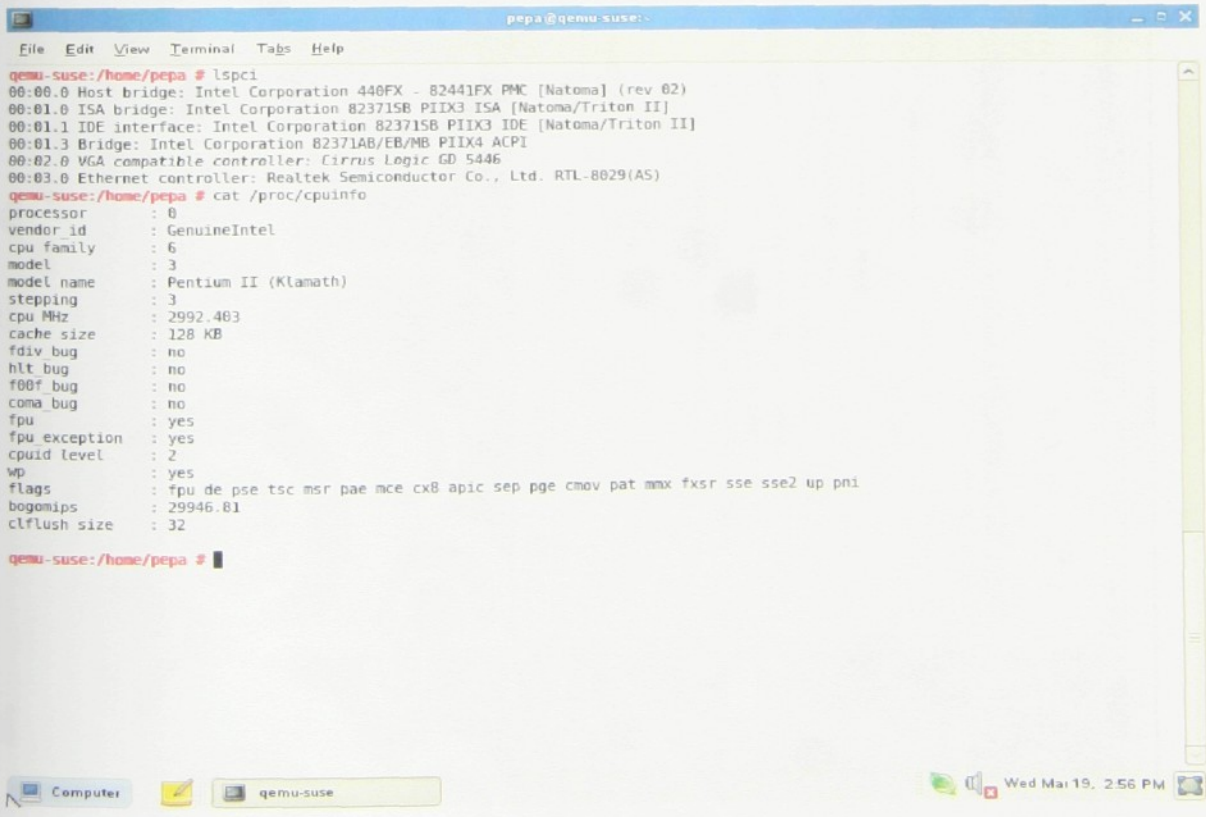
[10] VMware Inc. Virtualization Basics
http://www.vmware.com/virtualization/

[11] Xen User Guide
http://www.xen.org

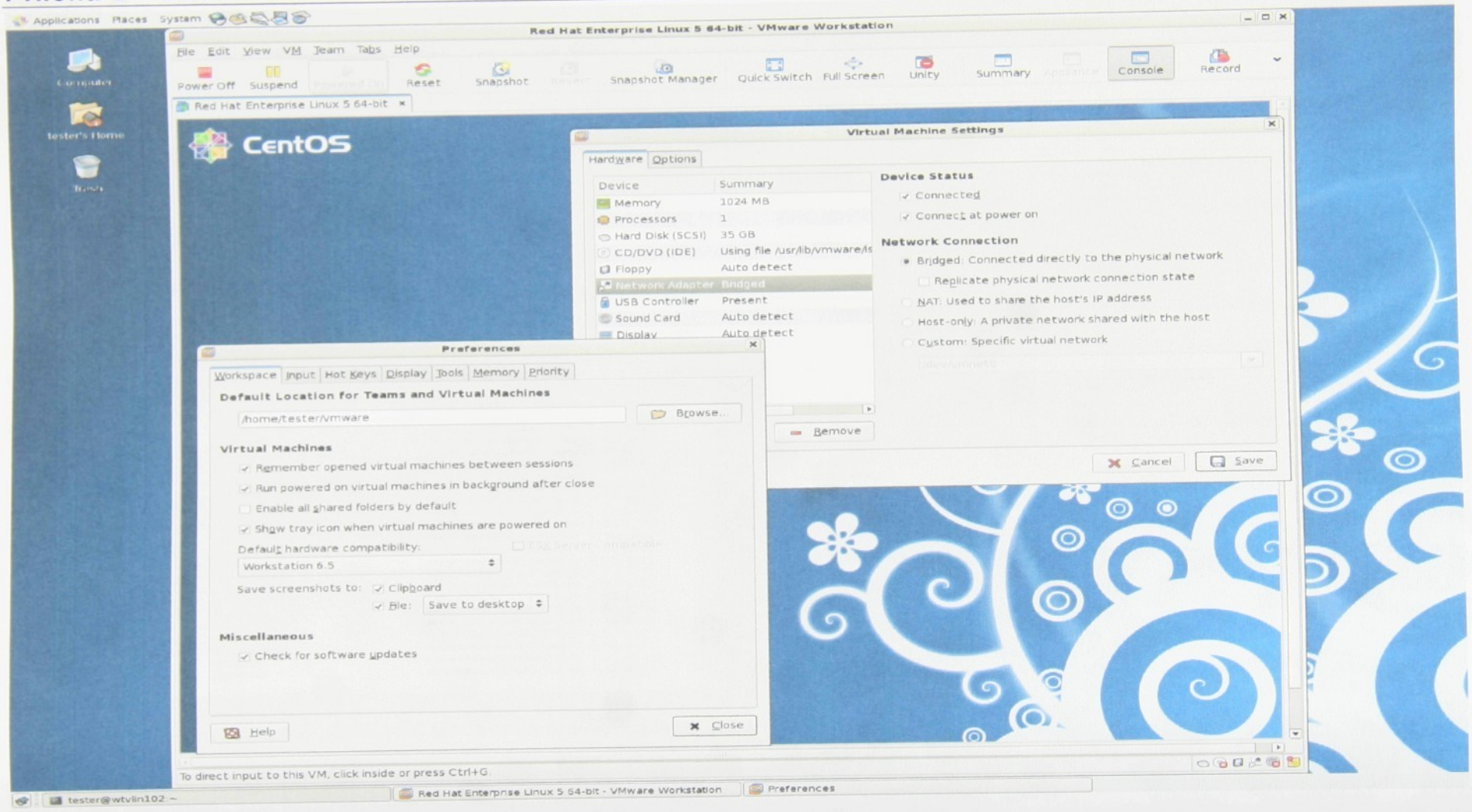
Příloha A – DOSBox



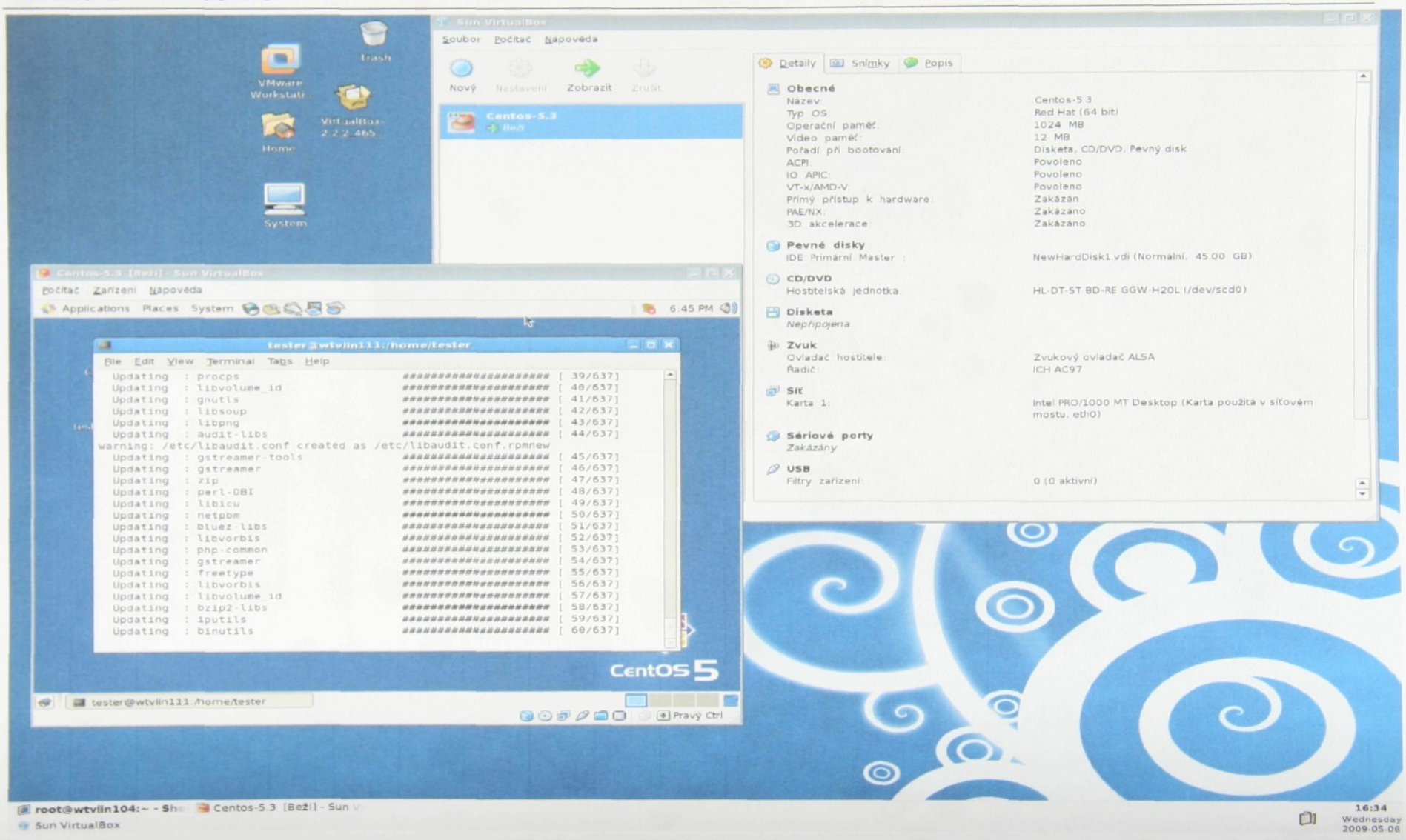
Příloha B – Hardware Qemu



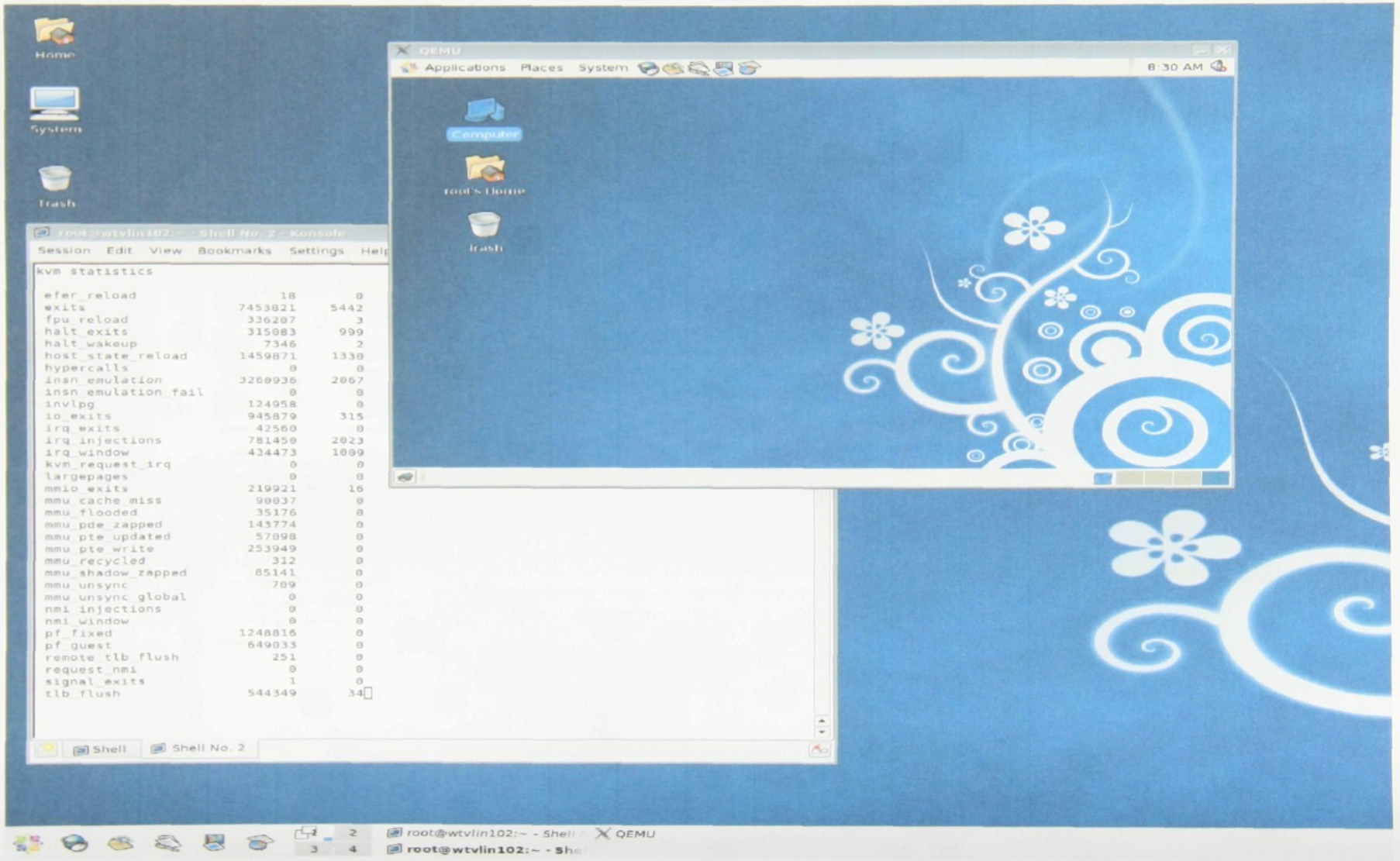
Příloha C – Běžící VMware Workstation



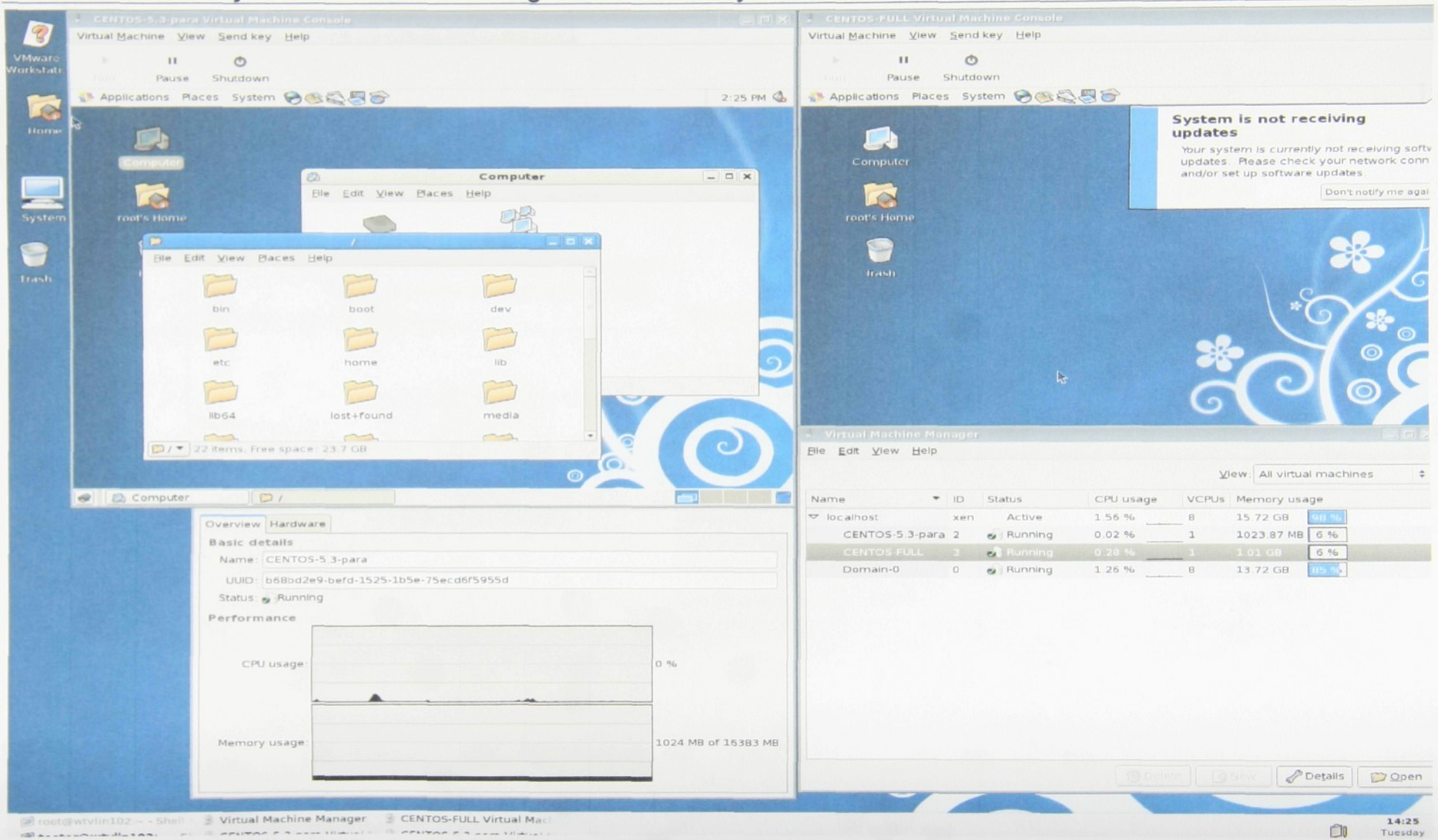
Příloha D – VirtualBox



Příloha E – Běžící systém s nástrojem KVM



Příloha F – Nástroj Virtual Machine Manager na běžícím systému



Příloha H – Přehled testovaných implemetací

Tab. 25: Přehled testovaných implementací

Jméno	Autor	Typické použití	Poznámka	Licence	Alternativa
VMware Workstation 6.5	VMware	Vývojáři a testeři, virtualizace pracovní stanice, výuka	Nativní virtualizace	Proprietární	VirtualBox, KVM, Xen
VirtualBox	Sun Microsystem	Vývojáři, koníčky, virtualizace pracovní stanice, virtualizace serverů	Nativní virtualizace	GPL2, plná verze je s proprietární, pro domácí použití zdarma	KVM, VMware Workstation, Qemu, Xen
KVM	Qumranet, Inc	Vývojáři, koníčky, virtualizace pracovní stanice	Tzv. hardware enabled virtualizace. Poměrně nový projekt, zatím je stále v živém vývoji	GPL2	Qemu, VirtualBox,Xen
Xen	University of Cambridge, Intel, AMD	Virtualizace serverů	Paravitrualizace, nutná úprava hostovaného systému	GPL	VMware ESX, VirtulaBox, OpenVZ
OpenVZ	Komunitní projekt podporovaný firmou Swsoft	Virtualizace a izolace serverů	Virtualizace na úrovni operačního systému. Hostovaný systém stejný jako virtualizační	GPL	VMware ESX, VirtulaBox, Xen

Příloha I – Skript použitý pro spuštění testů SysBench

```
#!/bin/bash
#Formátovaní je upraveno z důvodu umístění v práci
SBIN="/opt/sysbench/0.4.12/bin/sysbench"
DOUTPUT="$HOME/sysbench"
if [ ! -d "$DOUTPUT" ];then
 mkdir -p "$DOUTPUT"
fi
#test CPU
for i in `seq 1 10`
 do
    echo "start cpu_test iteration $i"
    $SBIN     --test=cpu --cpu-max-prime=140000 \
              --num-threads=2 run >> $DOUTPUT/sysbench_CPU.txt
done
#test threads
for i in `seq 1 10`
 do
    echo "start threads_test iteration $i"
    $SBIN     --test=threads --num-threads=64 --thread-yields=10000 \
              --thread-locks=4 run >> $DOUTPUT/sysbench_thread.txt
done
#test mutex
for i in `seq 1 10`
 do
    echo "start mutex_test iteration $i"
    $SBIN     --test=mutex --num-threads=64 --mutex-num=2048 \
              --mutex-locks=10000000 \
              --mutex-loops=10000 run >> $DOUTPUT/sysbench_mutex.txt
done
#test memory
for i in `seq 1 10`
 do
    echo "start memory_test iteration $i"
    $SBIN     --test=memory --num-threads=8 --memory-total-size=400G \
              --memory-block-size=4K \
              --memory-oper=write run >> $DOUTPUT/sysbench_memory.txt
done
#test fileIO
for i in `seq 1 10`
 do
  $SBIN --num-threads=8 --test=fileio --file-total-size=5G --file-test-mode=rndrw \
        --max-requests=200000 cleanup
  $SBIN --num-threads=8 --test=fileio --file-total-size=5G --file-test-mode=rndrw \
              --max-requests=200000 prepare
    echo "start fileio_test iteration $i"
  $SBIN --num-threads=8 --test=fileio --file-total-size=5G --file-test-mode=rndrw \
        --max-requests=200000 run  >> $DOUTPUT/sysbench_fileIO.txt
done
    $SBIN --num-threads=8 --test=fileio --file-total-size=5G \
          --file-test-mode=rndrw --max-requests=200000 cleanup
#test OLTP
$SBIN --test=oltp --mysql-user=user --db-driver=mysql cleanup
$SBIN --test=oltp --mysql-user=user --db-driver=mysql --oltp-table-size=100000 \
prepare
for i in `seq 1 10`
 do
    echo "start OLTP_test iteration $i"
    $SBIN     --test=oltp --num-threads=8  --mysql-user=user \
              --db-driver=mysql --max-requests=100000 \
              --oltp-table-size=100000 run >> $DOUTPUT/sysbench_OPLTP.txt
done
$SBIN --test=oltp --mysql-user=user --db-driver=mysql cleanup
```