Wirtualizacja

Paweł Mantur Artur Mączka Tomasz Niedabylski

Plan prezentacji

- Wprowadzenie teoretyczne
- Problemy w architekturze x86
  - Wprowadzone przez AMD i Intela wsparcie sprzętowe
- Prezentacja wybranych programów
  - Dla stacji roboczych
    - VMware
    - Virtual Box
  - Dla serwerów
    - Microsoft Virtual Server
    - Xen
- Podsumowanie

Na czym polega wirtualizacja? (ogólnie)

Wirtualizacja to technika ukrywania charakterystyki zasobów sprzętowych, pozwalająca na bardziej swobodne korzystanie z tych zasobów.

Przykłady:

- pamięć wirtualna
- partycje dysku twardego
- RAID
- maszyny wirtualne języków programowania
- wirtualna infrastruktura

Na czym się skoncentrujemy?

Wirtualizacja w kontekście systemów operacyjnych.

- Tradycyjnie: jeden komputer – jeden OS
  - Taka jest natura sprzętu
  - Tak są projektowane systemy operacyjne
- Potrzeby:
  - Jednoczesna praca z wieloma systemami operacyjnymi
  - Wykorzystywanie różnorodnych aplikacji
  - Efektywne wykorzystanie sprzętu
  - Bezpieczeństwo
  - Przenośność

Różne podejścia

- **Rebootowanie** – rozwiązuje tylko niewielką część problemów, często wystarczy, ale jest niewygodne.
- **Emulacja pełna** – modelujemy programistycznie architekturę, którą chcemy symulować. Strasznie niewydajne, gdyż każda instrukcja procesora jest zamieniana na kod języka programowania, a ten dopiero na instrukcje fizycznego procesora.
- **Emulacja API** danego systemu operacyjnego. Pozwala uruchamiać aplikacje napisane dla innego OS. Dosyć wydajne, jednak bywa problematyczne. Na przykład Wine – emuluje WinAPI na systemie Linux.
- **Wirtualizacja**

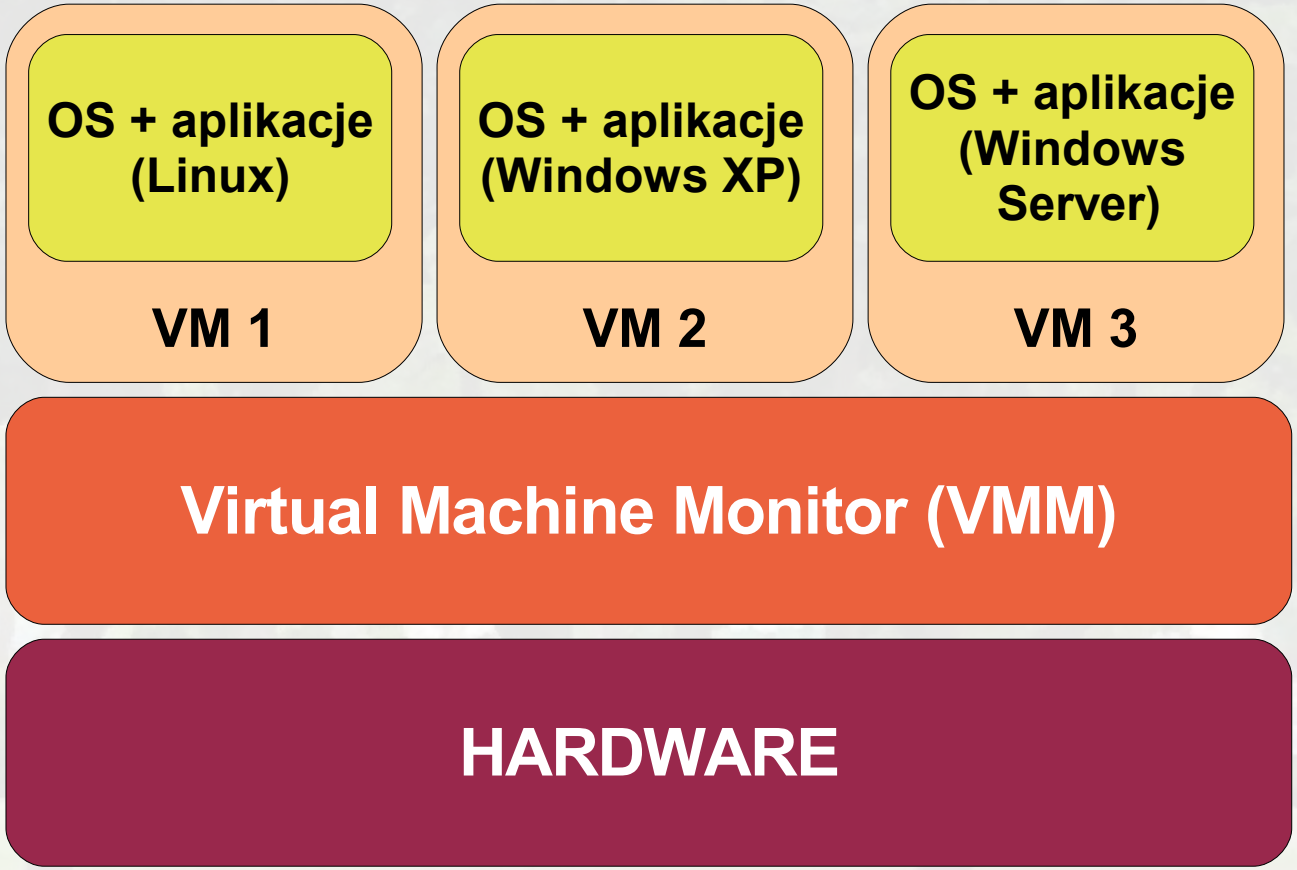
Wirtualizacja

Stosując wirtualizację staramy się aby jak najwięcej instrukcji wykonywało się bezpośrednio na fizycznym sprzęcie. Tam, gdzie nie jest to możliwe, wprowadzamy dodatkową warstwę oprogramowania - **Virtual Machine Monitor (VMM)**, inaczej **Hypervisor**. Warstwa ta pełni rolę arbitra dostępu do zasobów takich jak urządzenia I/O, CPU, RAM.

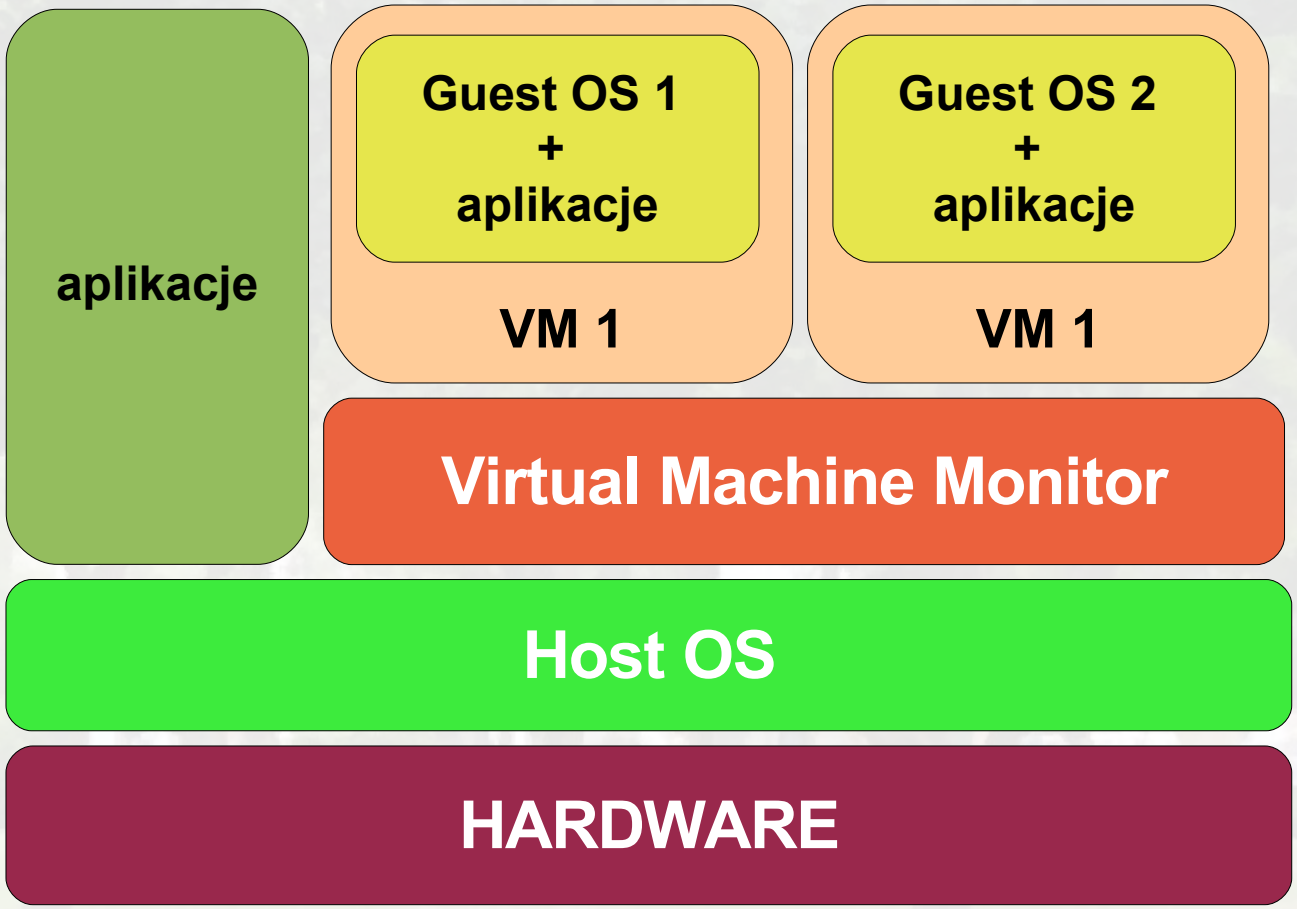
VMM może działać dwojako:

- na czystym sprzęcie (Type 1 – native) VMM musi bardziej przypominać OS, ale VM są bliżej sprzętu
- jako aplikacja systemu operacyjnego (Type 2 – hosted) VMM jest prostszy, ale emulacja jest kosztowniejsza

Type 1 – native VMM



Type 2 – hosted VMM



Wymagania wobec maszyny wirtualnej

W 1974 roku Gerald J. Popek i Robert P. Goldberg przedstawili kryteria właściwego funkcjonowania maszyny wirtualnej. Według nich VM powinna spełniać 3 warunki:

- **odpowiedniość** – program działający na maszynie wirtualnej musi zachowywać się w dokładnie taki sam sposób, jak na rzeczywistym sprzęcie
- **kontrola zasobów** – wirtualna maszyna musi w pełni kontrolować wszystkie zasoby, które są wirtualizowane
- **wydajność** – większa część instrukcji musi być wykonywana bez udziału maszyny wirtualnej

Kiedy osiągamy skuteczność?

Według kryteriów Popka-Goldberga w każdej architekturze można wyodrębnić następujące grupy instrukcji:

- **uprzywilejowane** - ich efektem jest przerwanie lub wywołanie systemowe w trybie użytkownika lub ich brak w trybie jądra
- **wrażliwe** - w trakcie wykonywania mogą zmienić konfigurację zasobów systemowych lub ich działanie jest zależne od konfiguracji systemu

Twierdzenie Popka-Goldberga mówi, że jeśli zbiór instrukcji wrażliwych jest podzbiorem zbioru instrukcji uprzywilejowanych to wirtualna maszyna może zostać skonstruowana

Można jednak wirtualizować i bez tych założeń godząc się na spadek wydajności. Tak będzie w architekturze x86.

Przykłady architektur

- Architektury naturalnie wspierające wirtualizację:
  - IBM System/370
  - Motorola MC68020
- Problemy z wirtualizacją:
  - x86

Trudności techniczne z bliska (1/2)

- Poziomy uprzywilejowania (PL od *privilege level*)
  - 0 - najwyższy PL (dla systemu operacyjnego)
  - 3 - najniższy PL (dla zwykłego oprogramowania)
- OS zainstalowany na VM nie może działać w trybie 0, gdyż z tego trybu korzysta VMM, jednakże jako system operacyjny nieraz także chce wykonywać instrukcje na poziomie 0
- OS żądając dostępu na wyższym poziomie powoduje wystąpienie wyjątku, który może być obsłużony przez VMM poprzez emulację danej instrukcji na odpowiednim poziomie
- Pomijając wydajność, takie rozwiązanie jest problematyczne, gdyż nie zawsze jest generowany wyjątek (odczyt z niektórych rejestrów nie jest uprzywilejowany, podczas gdy zapis jest – odczyt jest wrażliwy!)

Trudności techniczne z bliska (2/2)

- Procesor wspomaga sprzętowo przełączanie kontekstu pomiędzy systemem operacyjnym a innymi aplikacjami dzięki instrukcjom takim jak SYSENTER czy SYSEXIT. Gdy jesteśmy skazani na emulowanie wykonania tych instrukcji szybkość działania funkcji systemowych znacznie spada.
- Goszczący OS chce mieć dostęp do całej pamięci mu przydzielonej, jednak częściowo musi ona być dzielona z VMM w celu komunikacji. Problemem jest także to, że gość nie ma dostępu do ważnych części pamięci fizycznej, takich jak te używane przez urządzenia wejścia/wyjścia.

Softwarowe pokonywanie przeszkód

- Można ingerować w kod źródłowy OS w ten sposób, aby stworzyć interfejs który jest łatwiejszy do wirtualizacji. Zabieg ten nosi nazwę **parawirtualizacji**. Takie podejście jest stosowane m. in. przez projekty Xen oraz Denali. Rozwiązanie wydajne, ale nie zawsze możliwe.
- Innym sposobem jest **zmiana binariów** systemu operacyjnego w momencie jego działania. Wrażliwe operacje są odpowiednio przekształcane. Taką technikę wykorzystują m.in. VMware, Microsoft VirtualPC oraz Microsoft VirtualServer.

Wsparcie sprzętowe

Intel i AMD niezależnie wprowadziły sprzętowe ulepszenia do architektury x86 pozwalające znacznie poprawić wirtualizację:

- **Intel Vitrual Technology (IVT)**
  - VT-x dla architektury 32-bitowej
  - VT-i dla architektury 64-bitowej (Itanium)
  - VT-d Directed I/O
- **AMD**
  - AMD-V
  - Direct Connect

Technologie te nie są ze sobą kompatybilne.

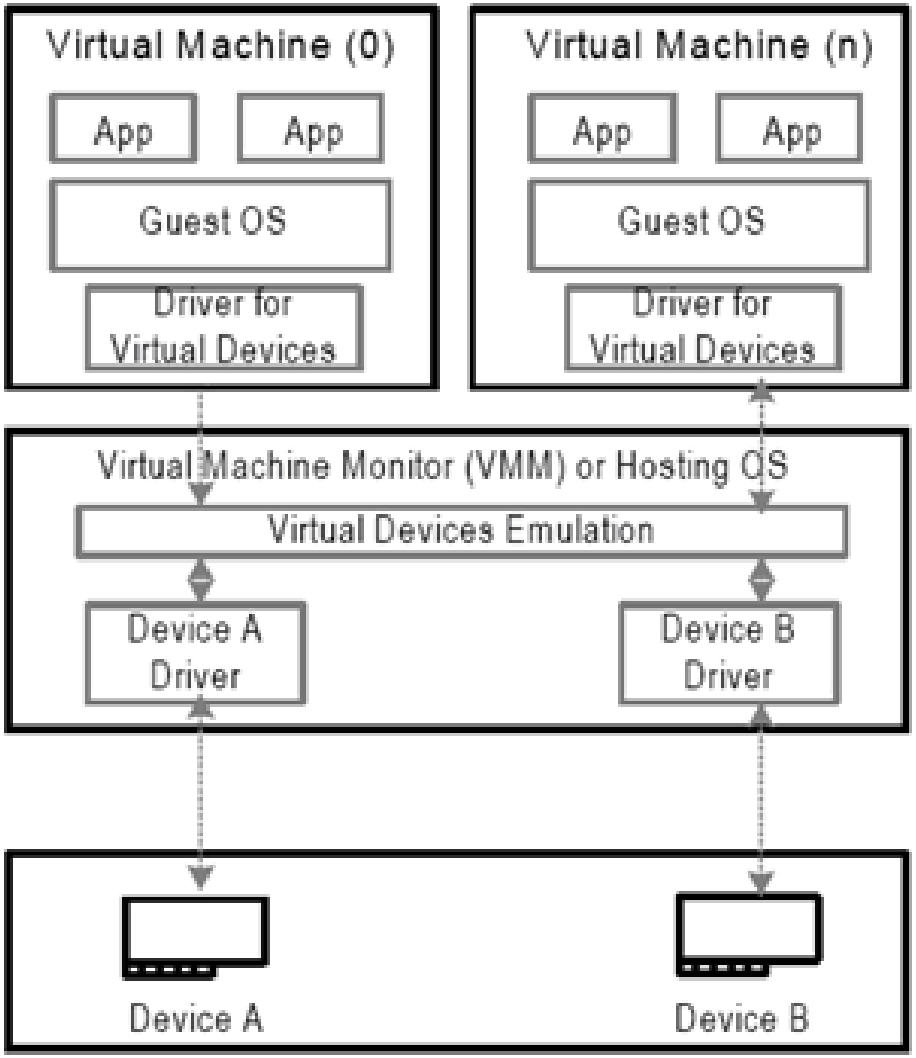
Wsparcie sprzętowe: VT-x oraz VT-i

- Wzbogacenie architektury o 2 nowe formy operacji: **VMX root operation** oraz **VMX non-root operation**. Pierwsza z nich jest używana przez VMM do jej własnych potrzeb, druga jest zaprojektowana aby stworzyć środowisko kontrolowane przez VMM ale zaprojektowane by wspierać VM. Obie formy operacji obsługują 4 poziomy uprzywilejowania co pozwala instrukcjom goszczącego OS wykonywać się na takim poziomie, dla jakiego oryginalnie zostały zaprojektowane.
- Zdefiniowanie nowych przejść pomiędzy stanami procesora: **VM entry** oraz **VM exit**. Przejścia są zarządzane przez strukturę danych o nazwie **Virtual Machine Control Structure (VMCS)**, która zawiera grupy pól **guest-state-area** oraz **host-state-area** pamiętające stan procesora dla poszczególnych maszyn.

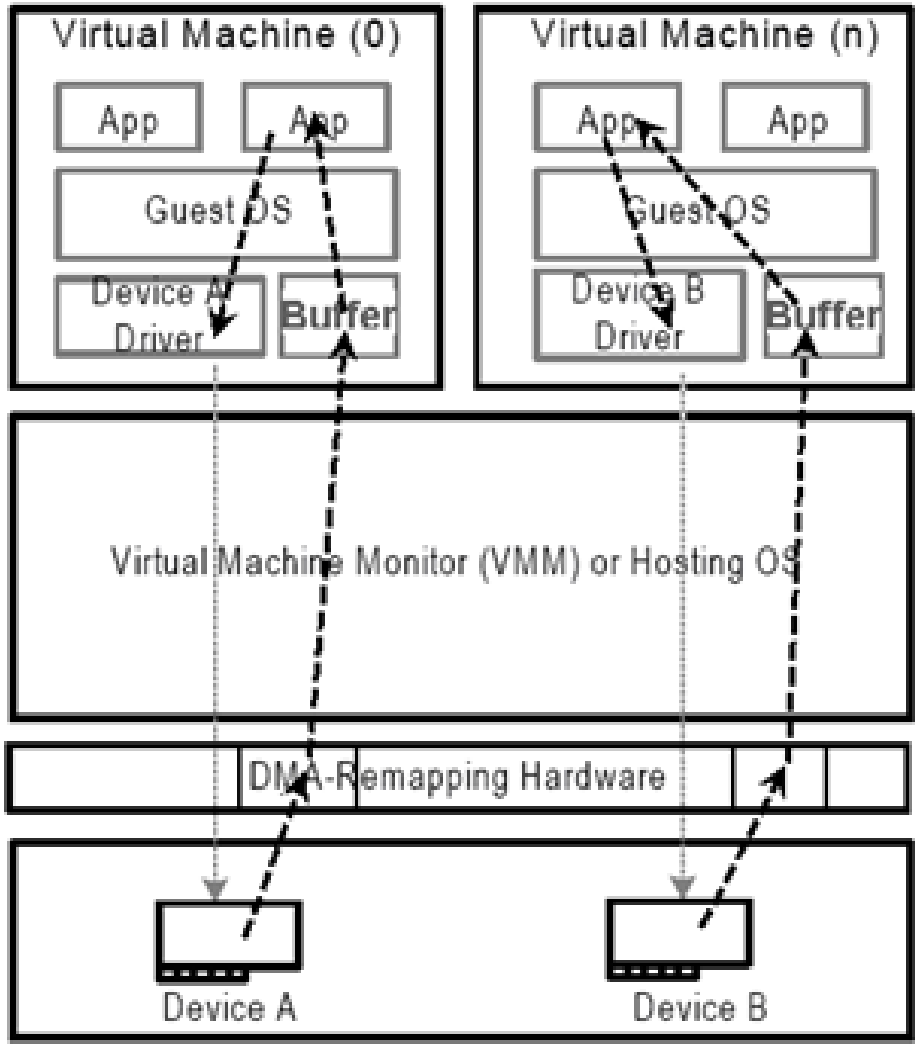
Wsparcie sprzętowe: VT-d

- Bez wsparcia sprzętowego operacje I/O są możliwe dzięki pełnemu emulowaniu urządzeń lub modyfikacji sterowników u gościa (przykład parawirtualizacji)
- Stosuje się też technikę **Direct Assignment**, dzięki czemu goszczący OS może pracować bezpośrednio na fizycznym sprzęcie. Wykorzystanie DMA jest jednak w tym wypadku utrudnione z powodu rozłącznych przestrzeni adresowych gościa i urządzenia. Urządzenie do komunikowania się używa adresów pamięci fizycznej do których VM nie ma dostępu. Softwerowo można rozwiązać ten problem wprowadzając do VMM moduł mapujący adresy.
- VT-d zwiększa wydajność poprzez wprowadzenie sprzętowego tłumaczenia adresów. Potrzebne do tego struktyry danych przechowywane są w specjalnym cache.

Ilustracja VT-d



Example Software-based I/O Virtualization



Direct Assignment of I/O Devices

Wsparcie sprzętowe:AMD-V

- Zintegrowany kontroler pamięci z obsługą wirtualizacji. Usprawnia wirtualizację i zapewnia efektywną separację pamięci operacyjnej poszczególnych maszyn wirtualnych, zwiększając bezpieczeństwo i jakość pracy użytkowników wirtualnych.
- Kontroler pamięci (IOMMU – I/O Memory Management Unit)– zapewnie translację adresów pamięci (również w trybie DMA) pozwalając na bezpośredni dostęp do pamięci dla goszczącego systemu (Intel - Direct Assignment)
- Mechanizm AMD-V™ zmniejsza obciążenie, selektywnie przechwytując instrukcje przeznaczone dla środowisk gości
- AMD stawia raczej na podział rdzeni między różne, goszczące OS.

AMD: Direct Connect

- Łączy ona bezpośrednio procesory, zintegrowany kontroler pamięci oraz układ We-Wy z centralną jednostką obliczeniową, umożliwiając komunikację z pełną prędkością pracy procesora.
- Technologia HyperTransport™ zapewnia skalowalną przepustowość łącza między procesorami, podsystemami We-Wy i innymi układami. Udostępnia do trzech łączy, zapewniających łączną przepustowość do 24,0 GB/s na procesor.

VMware

Jest to firma, która w latach 90 podjęła wyzwanie wirtualizacji architektury x86. Jej pierwszy produkt **VMware Workstation** ujrzał światło dzienne w 1999 roku i był pierwszym krokiem w budowaniu przez WMvare pozycji lidera w dziedzinie wirtualizacji.

Wirtualizacja platformy x86 była pożądana z podobnych powodów, dla których w latach 60. wirtualizowano komputery mainframe. Wzrost mocy obliczeniowej PC-tów i zarazem ich popularność spowodowały, że z przyczyn ekonomicznych warto jest lepiej wykorzystać ich możliwości. W praktyce wiele serwerów wykorzystuje jedynie 15% swojej mocy. Wirtulaizacja pozwala w bezpieczny sposób lepiej wykorzystać zasoby sprzętowe.

VMware

VMware rozpoczęło prace nad wirtualizacją x86 jeszcze przed wsparciem ze strony technologii IVT i AMD-V. Wyodrębniono 17 instrukcji, które podczas wykonywania w środowisku wirtualnym generowały błędy. VMware jako pierwszy wprowadził technikę “wyłapywania” tych instrukcji gdy były generowane, a następnie konwertowania ich w bezpieczny kod, który mógł być wirtualizowany. Pozostałym instrukcjom pozwolono natomiast wykonywać się bez interwencji.

Najważniejsze produkty VMvare

- Dla serwerów
  - VMware Infrastructure 3
  - VMware ESX Server
  - VMware Virtual Center
  - VMware Consolidated Backup
  - VMware Converter
- Dla desktopów
  - VMware Workstation
  - VMware ACE
  - VMware Virtual Desktop Infrastructure
  - VMware Fusion
  - VMware Player

Workstation 6.0 - wspierane Host OS

- Windows (32 bity)
  - Windows Vista (Enterprise/Business/Home/Ultimate)
  - Windows Server 2008 SP1
  - Windows Server 2003 (Standard/Web/Small Business/Enterprise/R2)
  - Windows XP (Home/Professional)
  - Windows 2000 (Server/Professional/Advanced Server)
- Windows (64 bity)
  - Windows Vista (Enterprise/Business/Home/Ultimate)
  - Windows Server 2008 x64 SP1
  - Windows Server 2003 x64 (SP1/R2)
  - Windows XP Professional x64

Workstation 6.0 - wspierane Host OS

- Linux 32 bity
  - Mandriva (Corporate Desktop/Server 4.0)
  - Mandrake (10.1/9.0)
  - Red Hat Enterprise Linux (5.0 i inne)
  - Red Hat Linux (9.0/8.0/7.3/7.2/7.1/7.0)
  - OpenSUSE (10.3/10.2), SUSE (10.3 – 8.2)
  - Ubuntu (7.04 - 5.04)
- Linux 64 bity
  - Mandariva (2006/2007)
  - Mandariva (Corporate Desktop/Server 4.0)
  - Red Hat Enterprise Linux (5.0 i inne)
  - SUSE Linux Enterprise Server (10/9)
  - OpenSUSE (10.3/10.2), SUSE (10.1 – 9.1)
  - Ubuntu (7.04 - 5.04)

Workstation 6.0 - wspierane Guest OS

- Microsoft Windows (Vista/XP/Server/ME/98/95/3.1)
- MS-DOS
- Linux (Mandriva/Mandrake/RedHat/SUSE/Turbolinux/ Novell Linux/Sun Java Desktop System/Ubuntu)
- Novell NetWare
- Novell Open Enterprise Server
- FreeBSD
- Sun Solaris

Prezentacja Vmware

- Utworzenie maszyny wirtualnej
- Pliki tworzące maszynę wirtualną
- Przenoszenie
- Uruchomienie
- Wykorzystanie

VirtualBox

- Wersja OpenSource i Binary, zawierająca niewolne oprogramowanie
- Wspiera VT-x / AMD-V
- Schowek współdzielony
- Wirtualne Dyski
- Foldery współdzielone

VirtualBox. Systemy HOSTa

- Windows 2000
- Windows XP
- Windows Server 2003
- Windows Vista (od 1.5 również 64 bit)
- Debian GNU/Linux 3.1 i 4.0
- Fedora Core 4 do 7
- Gentoo Linux
- Redhat Enterprise Linux 3,4 i 5
- SUSE Linux 9 i 10
- Ubuntu 5.10, 6.06, 6.10, 7.04
- Linux64 (od 1.4)
- Apple Mac

VirtualBox. Systemy Gościa.

- DOS/Windows 3.x/95/98/ME – słabo.
- Windows NT 4.0 – pełne wsparcie, dostępne sterowniki
- Windows 2000/XP/Server 2003/Vista – pełne wsparcie, dostępne pełne sterowniki
- Linux 2.4 – częściowe wsparcie
- Linux 2.6 – pełne wsparcie ze stepownikami
- FreeBSD – częściowe wsparcie bez sterowników
- OpenBSD 3.7, 3.8 – pełne wsparcie bez sterowników
- OS/2 Warp 4.5 – wymaga VT-x

VirtualBox

- Wspacie dla:
  - ACPI
  - I/O APIC
  - USB Device
  - Multiscreen
  - iSCSI
  - PXE Network
- Snapshots

A teraz o wirtualizacji w kontekście serwerów...

Korzyści dla serwerów

- Praktyka wykazuje, że duża część mocy serwerów jest niewykorzystywana. Rozwiązaniem jest przeniesienie ich jako serwerów wirtualnych na mniej maszyn. Nie zmienia to całości systemu, a daje oszczędności na sprzęcie i ludziach (administracja itp.)
- Możliwość serwisowania lub przenoszenia np. systemu operacyjnego lub serwera bez przerywania jego pracy

Microsoft Virtual Server 2005

- Przeznaczony głównie do zastosowań serwerowych, szczególnie dobrze współpracuje z Windows 2003 Server
- Możliwośc darmowego pobrania ze strony Microsoftu, po uprzednim zarejestrowaniu się
- Obsługuje systemy Windows, a także Linux i Solaris i inne systemy na platformę x86

Microsoft Virtual Server 2005 - możliwości

- Tworzenie wirtualnych klastrów na jednej maszynie
- Tworzenie klastrów z wirtualnych serwerów działających na różnych maszynach fizycznych (przez iSCSI)
- Wirtualne dyski twarde – możliwość zrobienia konfigurowalnego obrazu systemu
- Obsługuje sprzętowe wsparcie wirtualizacji – IVT i AMD-V
- Do 64 wirtualnych maszyn na komputerach 32-bitowych, więcej na 64-bitowych
- Obsługa do 32 procesorów fizycznych

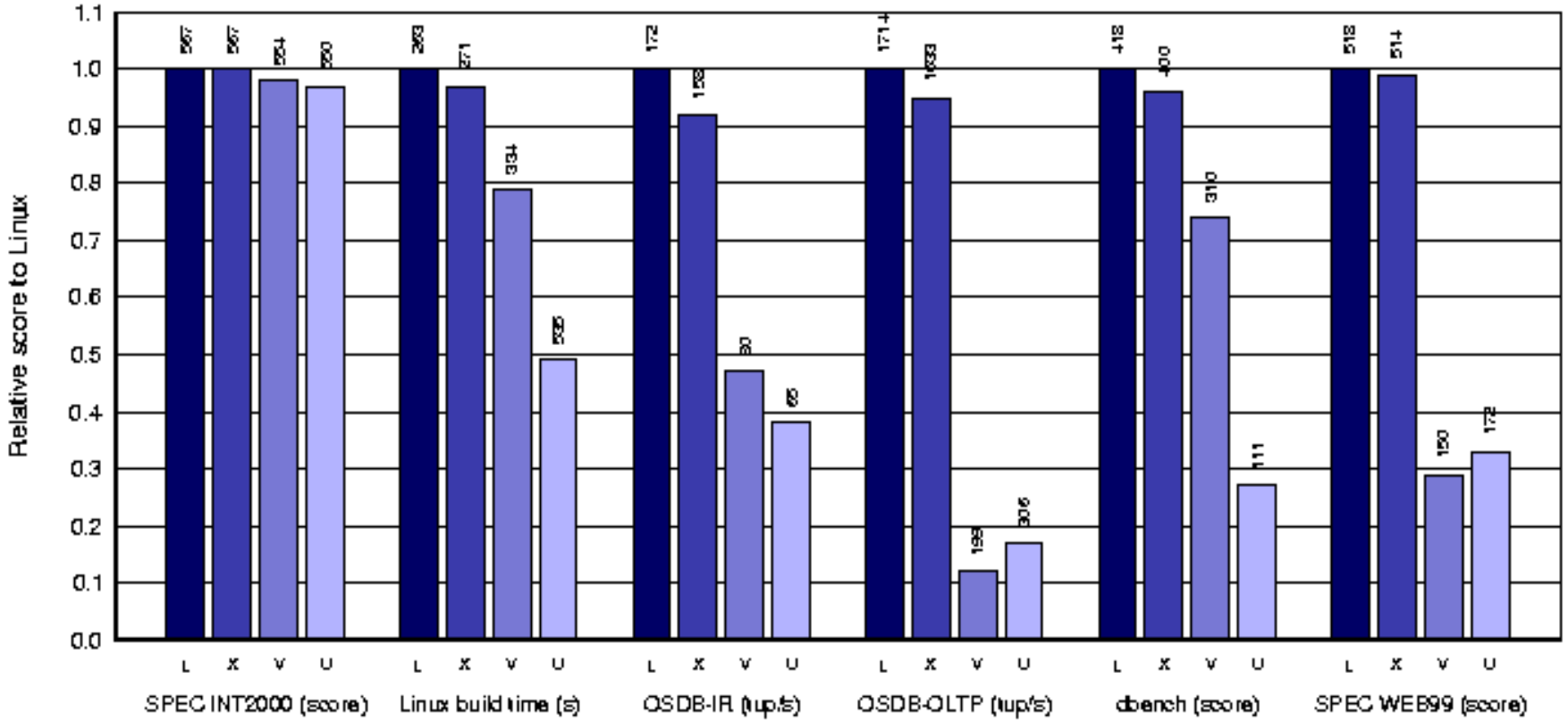
Xen

- Rozpowszechniany na licencji GNU GPL
- Stosuje parawirtualizację (wirtualizuje systemy specjalnie dla niego zmodyfikowane), dlatego najlepiej obsługuje systemy open-source
- Obsługuje także Microsoft Windows Server 2000, 2003 i XP, ale wymaga procesora wspierającego wirtualizację (IVT lub AMD-V)
- Microsoft i uniwersytet w Cambridge stworzyli także zmodyfikowaną wersję Windows XP, ale ze względów licencyjnych nie została ona opublikowana

Xen – trochę szczegółów

- Xen działa jako native VMM – nie jest instalowany jako program w systemie - gospodarzu
- Potrzebuje jednak systemu operacyjnego do zarządzania wirtualnymi maszynami (zmodyfikowane Linuksy, OpenBSD, OpenSolaris) – tak zwanego domain number 0 (dom0)
- Systemy wirtualizowane nazywane są unprivileged domain (domU)
- Przy parawirtualizacji, jądro dom0 działa na poziomie uprzywilejowania 0, a reszta systemu i wszystkie domU na PL 1
- Przy wirtualizacji niezmienionych systemów, wykorzystywane jest wsparcie procesora i dom0 pracuje w trybie „root”, a domU w „non-root”

Porównanie Xen i innych VMM

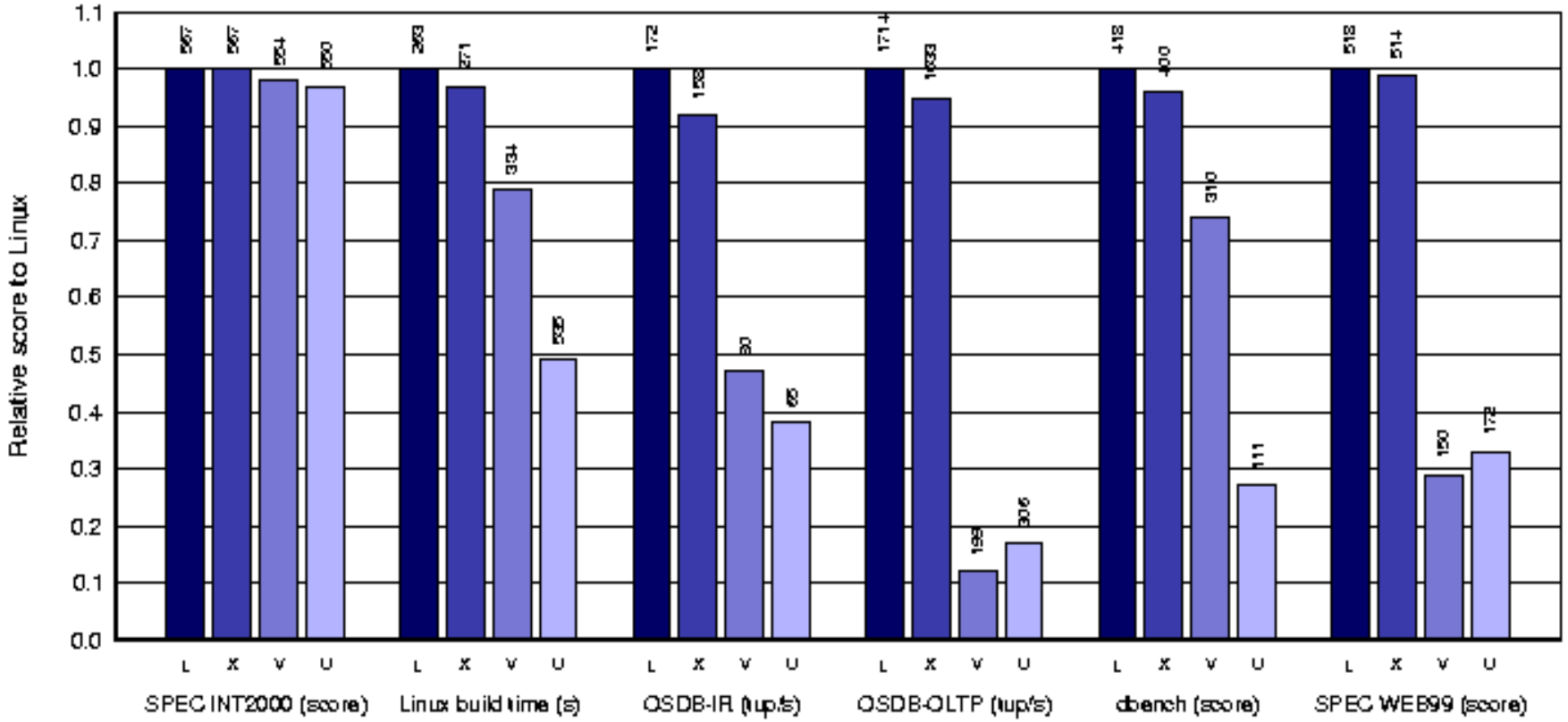


- L – Linux, X – wirtualny Linux na Xen, V – Vmware Workstation 3.2, U – User Mode Linux

XEN – skąd wziąć

- Ze strony http://xen.xensource.com/download/ można pobrać darmowy Xen-Express z różnymi hostami, a także kilka systemów – guestów
- Wszystkie Linuksy od wersji 2.6.23 wspierają parawirtualizację i mogą być guestami

Porównanie Xen i innych VMM



- L – Linux, X – wirtualny Linux na Xen, V – Vmware Workstation 3.2, U – User Mode Linux

Czy na pewno same zalety wirtualizacji?

- Spadek prędkości działania
- Większa złożoność systemów (trudniej mierzyć wydajność, szukać błędów itp.)
- Bezpieczeństwo – np. możliwość ataku od strony VMM, większa złożoność -> więcej dziur
- Potrzeba administratorów znających się na wirtualizacji
- Problemy z licencjami komercyjnymi (np. Windows Vista Home nie moze byc instalowana na maszynach wirtualnych)

Bibliografia

- www.intel.com
- www.vmware.com
- www.wikipedia.com
- www.microsoft.com
- www.xensource.com
- http://www.btquarterly.com/?mc=pros-cons-virtualization&page=virt-viewresearch