Michał Albrycht
Bartosz Borkowski
Radosław Kujawa

Wirtualizacja

Systemy operacyjne

Agenda

- Wirtualizacja
  - Historia i współczesne wykorzystanie
  - Podziały wirtualizacji
- Wirtualizacja tablic stron
  - Shadow Page Tables
  - Nested Page Tables
- Wspomaganie sprzętowe wirtualizacji
  - XEN i KVM
- Zagrożenia wirtualizacji

Co to jest wirtualizacja?

- Wirtualizacja

  jest to abstrakcja zasobów maszyny fizycznej

- Przykład mówi więcej niż definicja
  - RAID
  - Pamięć wirtualna
  - Wirtualne napędy CD-ROM
  - itp.

Historia

- W latach 60 IBM wprowadził komputer M44 na którym były uruchamiane wirtualne maszyny 44X.
- Użykownik pracował na wirtualnej kopi systemu
- Chodziło o lepsze wykorzystanie procesora
- Zainteresowanie wirtualizacją malało wraz z popularyzacją PC

Po co nam teraz wirtualizacja?

- Tworzenie i rozwój systemów operacyjny
- Testowanie kompatybilności (np. programów z systemem operacyjnym)
- Symulowanie wielu komputerów przez jeden komputer fizyczny
- Testowanie programów w środowisku ze zmienioną ilością zasobów (np. aplikacje komórkowe, zachowanie programów przy małej ilości RAMu itp.)
- „Kontakt” ze szkodliwym oprogramowaniem.
- Korzystanie z kilku systemów operacyjnych jednocześnie
- Wykorzystywane podczas nauki/szkoleń dzięki łatwej odtwarzalności systemu

Co mieści się w pojęciu „wirtualizacja”?

- Wirtualizację można badać w wielu aspektach. My spojrzymy pod kątem sposobu tworzenia abstrakcji:
  - Emulacja (pełna)
  - Emulacja API
  - Wirtualizacja (pełna)
    - Parawirtualizacja

Emulacja (pełna) - charakterystyka

- Polega na symulacji sprzętu fizycznego w szczególności procesora, pamięci, rejestrów, czasu systemowego itd...
- Każda instrukcja w środowisku emulowanym musi być tłumaczona przez emulator
- Wady:
  - Bardzo duże narzuty na przetwarzanie pojedynczego żądania co skutkuje ogromnymi stratami wydajności
  - Zasobożerność
- Zalety:
  - Niezależność od architektury
  - Bezpieczeństwo

Emulacja (pełna) - przykłady

Programy działające na zasadzie pełnej emulacji:

- QEMU
- Bochs
- PearPC

Emulatory często wykorzystywane są w celu symulacji środowisk starszych komputerów i konsol (Atari, Commodore, Amiga, Nintendo) gdzie straty wydajności nie są odczuwalne.

Emulacja API - charakterystyka

- NIE symulujemy całego systemu
- Emulowany jest tylko interfejs udostępniany przez system
- Zalety:
  - Dużo szybsza od pełnej emulacji
- Wady:
  - Nieprzenośna
  - Częste problemy w działaniu
    (zmieniające się biblioteki, lub ich brak)
  - Konieczność przepisania całego API

Emulacja API – przykład: Wine

Najbardziej znany program emulujący API to Wine (Wine is not emulator)

- Oparty na WinAPI (interfejs systemu Windows)
- Posiada odpowiedniki znacznej ilości bibliotek
- Istnieje możliwość podłączenia oryginalnych bibliotek systemu Windows

Wirtualizacja

- Jest niejako hybrydą emulacji pełnej i emulacji API
- Systemy emulowane ograniczone do jednej architektury
- Bezpośrednie wykonywanie „bezpiecznych” instrukcji bezpośrednio na sprzęcie
- Warstwą pośrednią pomiędzy systemem gospodarzem a systemem gościem jest monitor maszyny wirtualnej (VMM, hypervisor)
- Szybszy od emulacji pełnej

Parawirtualizacja

- Technika wirtualizacji, która tworzy maszynę wirtualną podobną, ale nie identyczną z tą symulowaną.
- Pozwala na prostszą budowę monitora oraz na osiągnięcie wyników bliskich do tych niewirtualizowanego sprzętu.
- Czasami nie tyle symuluje sprzęt, co dostarcza API dla zmodyfikowanego gościnnego systemu operacyjnego.

Parawirtualizacja - przykłady

- Xen
- KVM
- OpenVZ

Pełna wirutalizacja

- Tworzy środowisko wirtualnej maszyny, która zapewnia pełna symulację zasobów sprzętowych
- Gość nie wie, że pracuje w środowisku wirtualnym
- Gość ma możliwość bezpośredniej pracy na sprzęcie
- Żadna z operacji wykonywanych przez gościa nie może zmienić stanu innych systemów wirtualnych, VMM czy gospodarza
- Możliwa tylko dla niektórych architektur

Twierdzenie Popek'a Goldberga

Wymagania względem wirtualizacji:

- **Równoważność**

  sprzęt symulowany zachowuje się identycznie (w najważniejszych punktach), co prawdziwy
- **Kontrola zasobów**

  maszyna wirtualna ma całkowitą kontrolę nad symulowanymi zasobami
- **Efektywność**

  większość instrukcji maszynowych powinna być wykonywana bez udziału maszyny wirtualnej

Twierdzenie Popek'a Goldberga cz.2

Aby stworzyć zestaw wymagań instrukcje podzielono na 3 grupy:

- instrukcje uprzywilejowane

  przerywane (wywołuje przerwanie systemowe) gdy procesor działa w trybie użytkownika, wykonywane w trybie systemowym

- instrukcje kontrolne

  próbujące zmienić konfigurację zasobów systemu

- instrukcje środowiskowe

  ich działanie i wynik zależy od ustawień środowiska

Twierdzenie Popek'a Goldberga cz.3

Twierdzenie:

Dla każdego komputera 3 generacji można stworzyć maszynę wirtualną jeśli zbiór instrukcji wrażliwych (kontrolne i środowiskowe) jest podzbiorem instrukcji uprzywilejowanych.

Instrukcje, które mogą zakłócić działanie monitora są wychwytywane, a sterowanie jest przekazywane do monitora.

Architektura x86 nie spełnia tego wymogu!

Problemy z wirtualizacją x86

W procesorze x86 istnieje 17 instrukcji które sprawiają problem przy wirtualizacji:

- SGDT, SIDT, SLDT – odczytanie rejestrów gospodarza
- SMSW – odczytanie informacji o monitorze wirtualizacji
- PUSHF, POPF – odczytanie stanu pracy procesora, zmiana rejestru EFLAGS przez Gościa

Problemy z wirtualizacją x86

- LAR, LSL, VERR, VERW – instrukcje te potrzebują informacji o aktualnym poziomie ochrony – poziom ochrony na maszynie wirtualnej inny od rzeczywistego
- POP, PUSH – problem z poziomem ochrony, np. PUSH jest w stanie wstrzymać proces wykonywany na maszynie wirtualnej
- CALL, JMP, RET – problem ze zmianą poziomu ochrony
- INT n, STR - dostęp do rejestru EFLAGS

W konsekwencji zaimplementowanie pełnej wirtualizacji na x86 jest niemożliwe!

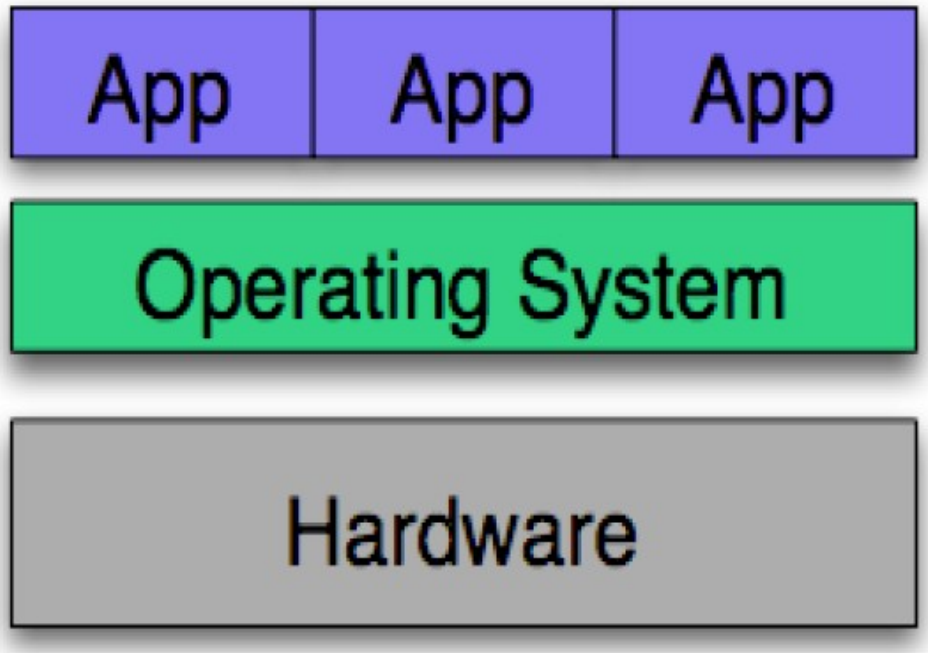
Poziomy ochrony w x86

- Zorganizowane za pomocą Ringów
- Cztery poziomy: od 0 do 3
  - 0 – poziom wykonywania jądra
  - 1 – poziom sterowników
  - 2 – poziom sterowników
  - 3 – poziom wykonywania aplikacji
- Uprawnienia poziomu z numerem n+1 zawierają się w uprawnieniach poziomu n

Poziomy ochrony w x86 - przykłady

System bez włączonej maszyny wirtualnej

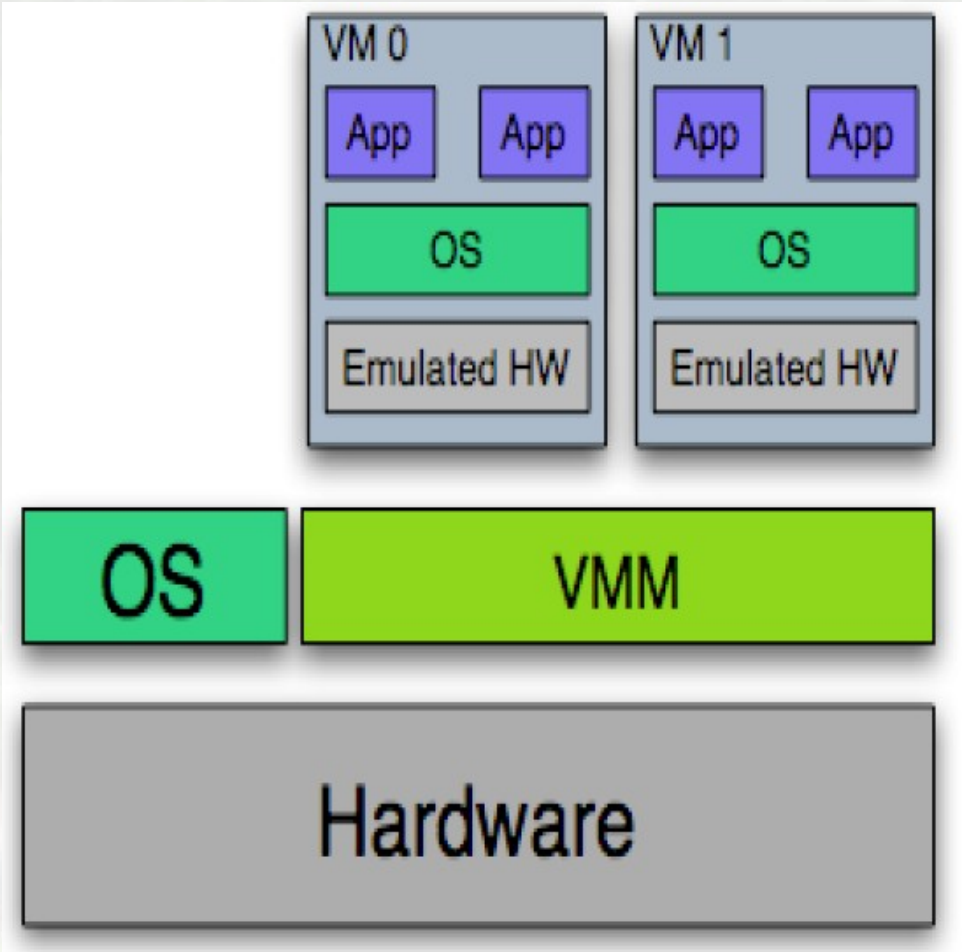
- App wykonują się w Ring 3
- OS wykonuje się w Ring 0



Poziomy ochrony w x86 - przykłady

Wirtualizacja programowa

- OS i VMM wykonują się na poziomie Ring 0
- VM0 i VM1 wykonuja się na poziomie Ring 1 przez co nie mają bezpośredniego kontaktu z procesorem.



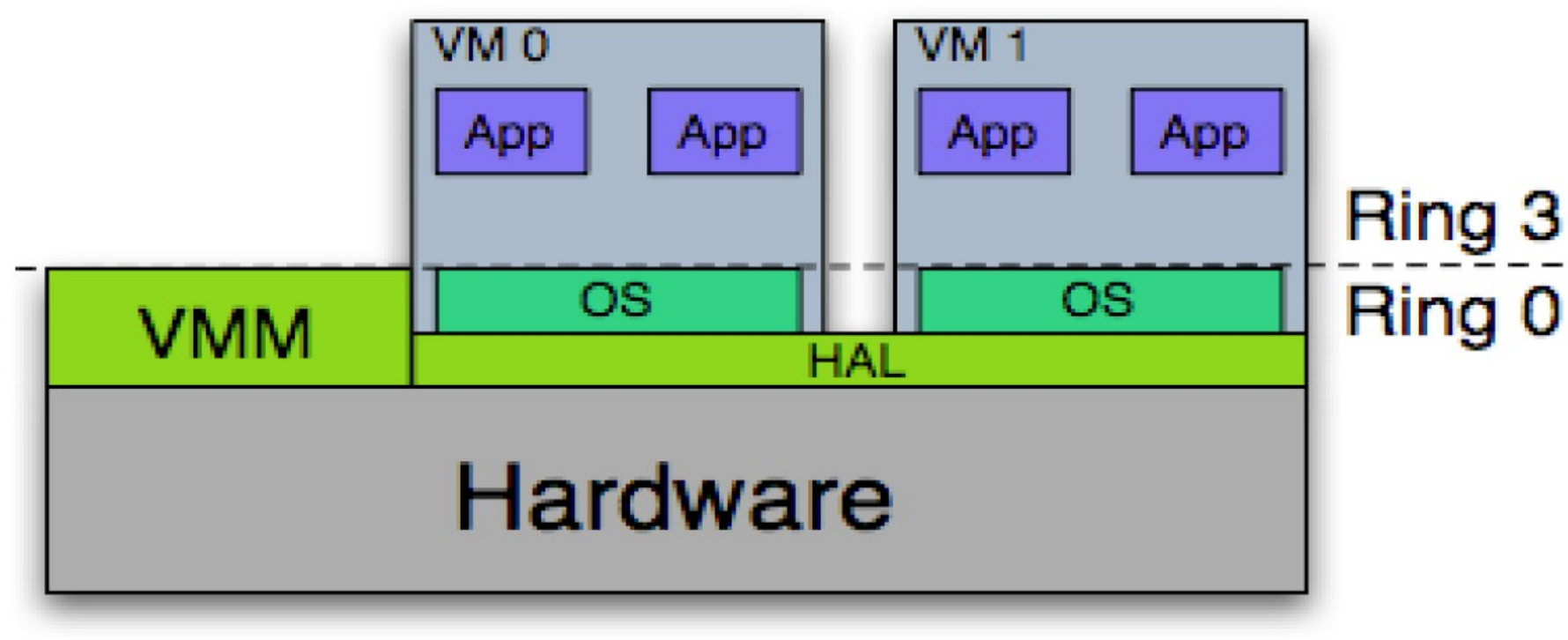
Wirtualizacja wspomagana sprzętowo

- Od 2005 r. niezależne rozwiązania od Intela i AMD
  - Intel VT
  - AMD-V

Usprawnienie architektury w celu zwiększenia szybkości działania wirtualnego systemu.

- Wprowadznie nowego poziomu ochrony Ring -1

Wirtualizacja na Intel VT



- VMM Pracującu w trybie VMX root (Ring 0)
- Maszyny wirtualne w trybie VMX non-root (Ring 0)

Jak to jest w Intel VT?

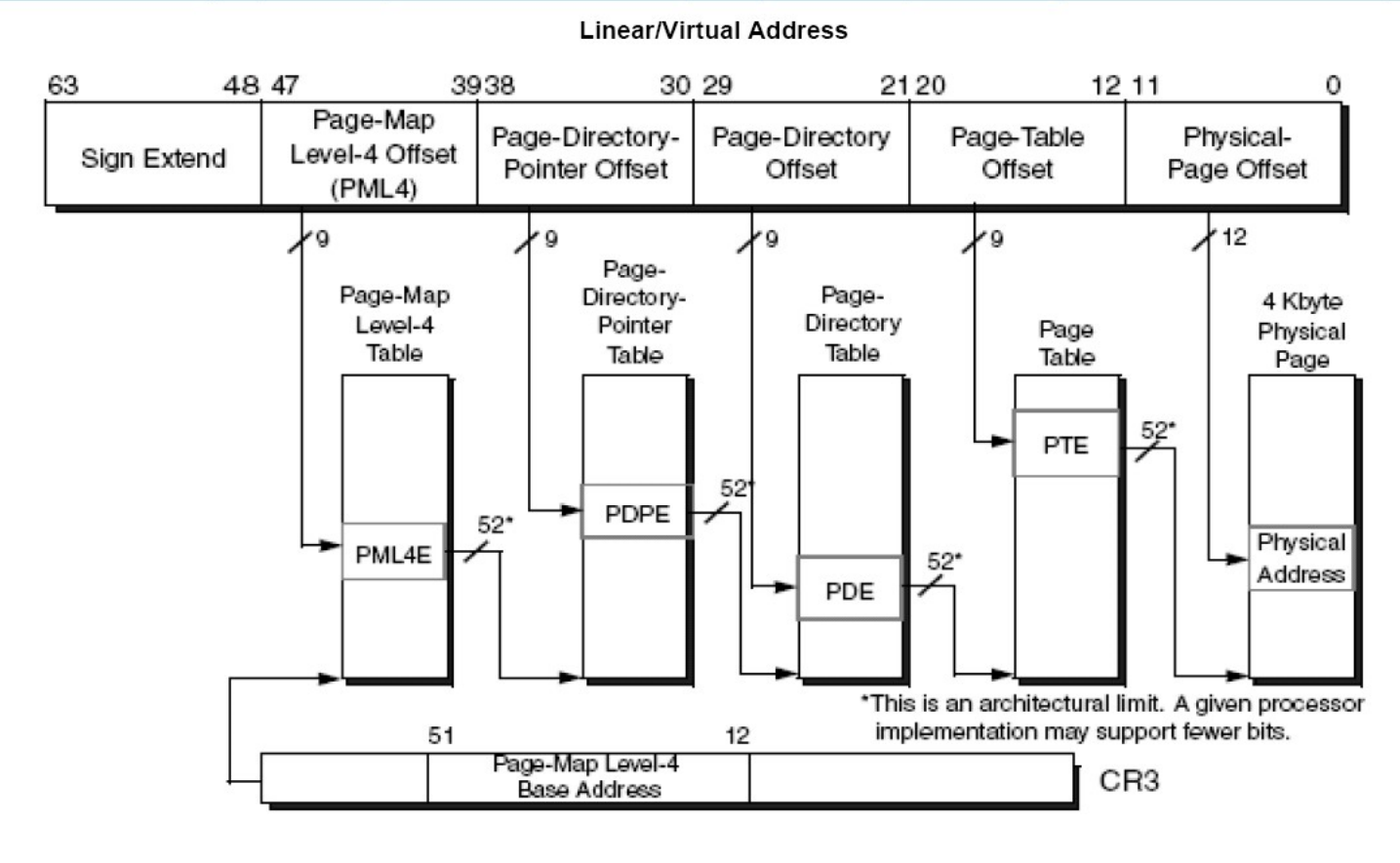
- 10 nowych instrukcji (VMPTRLD, VMPTRST, VMCLEAR, VMREAD, VMWRITE, VMCALL, VMLAUNCH, VMRESUME, VMXOFF oraz VMXON)
- Procesor działa w dwóch trybach
  - VMX root
  - VMX non-root
- Informacje przechowywane w specjalnej strukturze VMCS (Virtual Machine Control Structure)

Stronicowanie a wirtualizacja

- Czemu wirtualizacja stronicowania jest tak ważna
- Jak wirtualizuje się tablice stron
  - Para-wirtualizacja
  - Pełna wirtualizacja
    - Realizowana przez software
    - Realizowana przez software i hardware

Stronicowanie

- Tłumaczenie adresu wirtualnego na adres fizyczny
- “Spacer“ po tablicy stron



Stronicowanie

- Problem TLB
  - TLB – pamięć podręczna stosowana, by przyspieszyć translację adresu wirtualnego na fizyczny
  - Potrzeba aktualizacji TLB podczas:
    - dodawnia nowej strony
    - usuwania strony
    - operacji na architekturach SMP
      - gdy jakaś ramka w pamięci jest używana przez wiele procesorów, trzeba dopilnować, by po jej zwolnieniu każdy procesor miał o niej spójne informacje.

Stronicowanie

- Działania procesora na tablicy stron
  - ustawianie *bitu użycia*
    - gdy strona jest używana, procesor ustawia ten bit na 1
  - ustawianie *bitu modyfikacji*
    - gdy strona jest modyfikowana, procesor ustawia ten bit na 1
- System operacyjny zbiera informację o zmianiach tych bitów, tworzy statystyki I korzysta z nich podczas swapingu

Pierwsza próba wirtualizacji

- Najprostsza i najszybsza metoda wirtualizacji tablic stron to ich para-wirtualizacja
- Gość i monitor wspólnie dbają o to, by koszty wirtualizacji były jak najmniejsze
- Niestety para-wirtualizacja jest sprzeczna z ideą wirtualizacji

Wirtualizacja pełna z użyciem tablicy cienia

- Pomysł opiera się na dublowaniu wszystkich operacji wykonywanych przez gościa w jego tablicy stron (tzw. *guest page table – gPT*)
- Do owego dublowania wykorzystuje się tzw. *shadow page table (sPT)*
- Gość nie ma pojęcia, że jest gościem, działa jak najnormalniejszy system
- sPT pilnuje wszelkich modyfikacji dokonywanych na gPT
- to z sPT tak naprawdę korzysta procesor

Shadow page table – techniki działania

- Technika “ochrony przed zapisem”
  - Monitor ustanawia ochronę przed zapisem na wszystkie strony fizyczne zawierające elementy z gPT
  - Każda próba zapisu przez gościa kończy się podniesieniem wyjątku
  - Każdy wyjątek jest łapany przez monitor
  - Monitor obsługuje wyjątek
    - Emuluje operację, którą próbował wykonać gość
    - Ustawia wszystkie odpowienie dane w tablicach gPT i sPT

Shadow page table – technika “virtual TLB”

- Brak ochrony przed zapisem
- Gdy gość próbuje coś zapisać do gPT, po prostu to zapisuje
- Jednak gdy próbuje się odwołać pod adres, który widnieje w gPT jako “fizyczny”...
  - procesor dostaje od gościa adres wirtualny
  - próba “spaceru po stronach” po sPT
  - w sPT jeszcze nie ma żadnego wpisu
  - generowany jest wyjątek
  - dopiero wtedy monitor interweniuje uzupełniając braki w sPT

Problem z technikami software'owymi

- Duża liczba generowanych błędów stron
  - page faulty zwyczajne – takie, których gość jest świadom
  - page faulty wynikające z wirtualizacji
- Każdy błąd musi zostać rozpoznany i obsłużony przez monitor.
- Procesor zmienia bity użycia I modyfikacji tylko w sPT. Monitor musi zadbać o to, by odpowiednie zmiany znalazły się też w gPT

Problem z technikami software'owymi

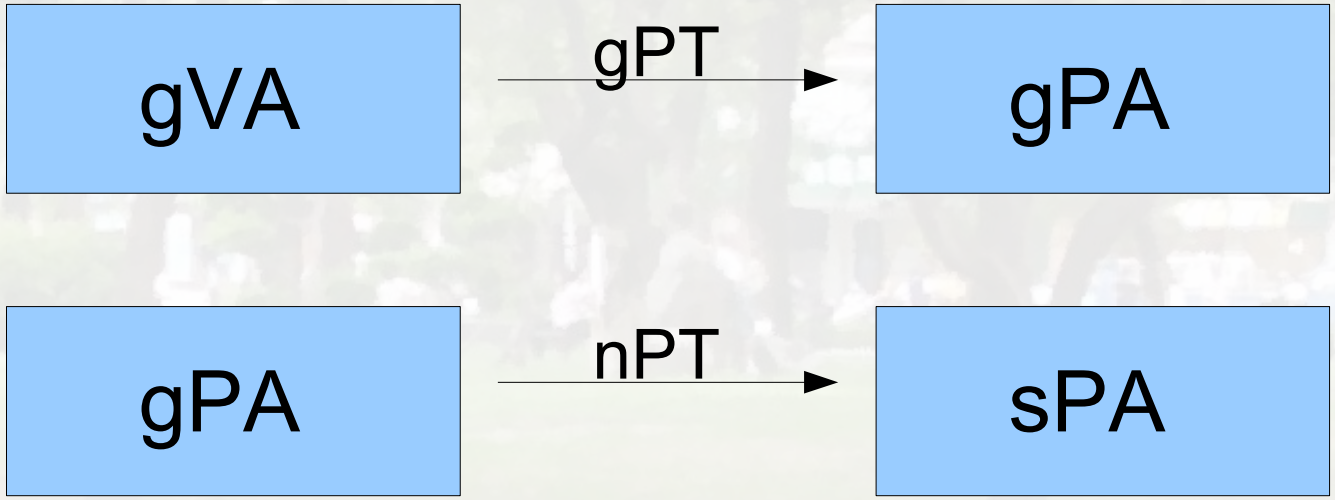
- w przypadku SMP istnieją co najmniej dwie możliwości tworzenia sPT
  - można tworzyć tyle sPT ile jest procesorów
    - narzut pamięciowy
  - można tworzyć tylko jedną
    - narzut synchronizacyjny
- szacuje się, że na niektórych maszynach nawet 75% czasu pracy monitora to operacje na sPT.

Nested Page Tables

- głównym problemem z sPT jest duża ilość operacji dokonywana przez monitor
- Należy skupić się na tym, by ograniczyć działania monitora
- AMD stworzyło technologię, która to umożliwia

Abstrakcja

- gPT ma za zadanie mapować adres wirtualny gościa na adres fizyczny gościa
- Nested Page Tables mapują adres fizyczny gościa na prawdziwy, systemowy adres fizyczny

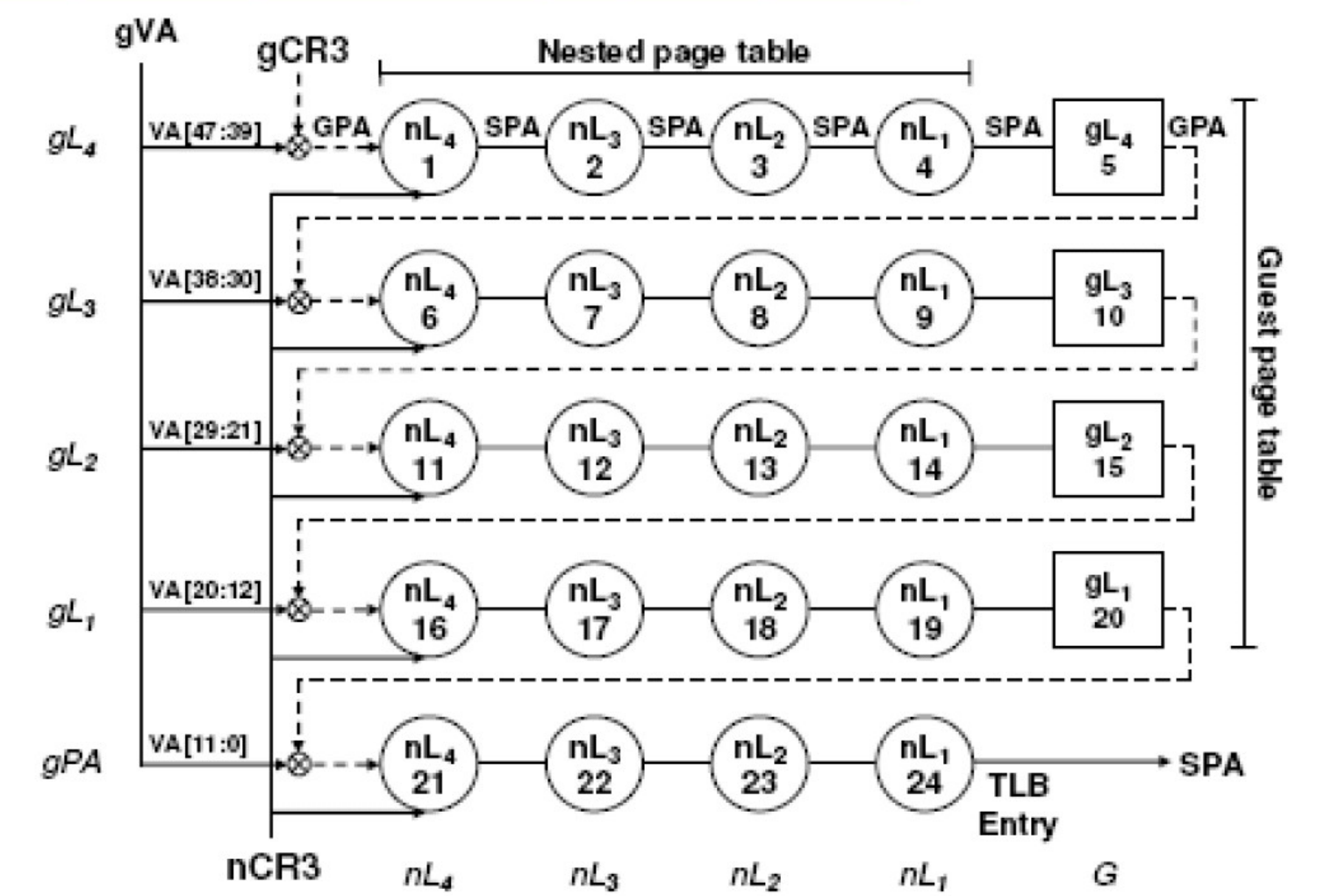


Abstrakcja

- sPT nie ma wcale. Jest zastąpiona przez nPT.
- na nPT nie trzeba symulować każdego ruchu gościa. Służą tylko do wyliczania adresów.
- Niestety sposób wyliczania adresu fizycznego jest dłuższy – w przypadku chybienia w TLB mogą się zdarzyć nawet 24 odwołania do pamięci dla pojedynczego adresu.

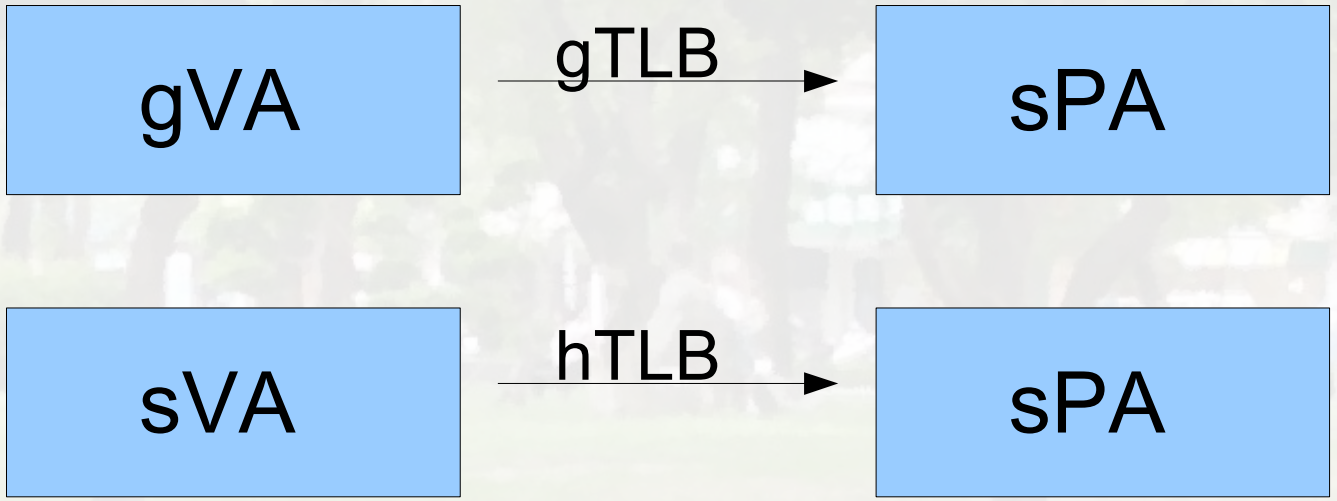
Schemat działania translacji

- Po gPT i nPT jest wykonywany “dwuwymiarowy” spacer.



Zasada działania nested page tables

- Wsparcie ze strony hardware'u w postaci dwutrybowości:
  - tryb gościa
  - tryb monitora
- tryby posiadają osobne rejestry, a także dwa oddzielne TLB



Zasada działania nested page tables

- Dodatkowe usprawnienia
  - Nested TLB – inna pamięć podręczna, tym razem tłumacząca adres fizyczny gościa w systemowy adres fizyczny
  - Page Walk Cache (PWC) – jeszcze inna pamięć podręczna
    Ma na celu skrócenie czasu spaceru po stronach
    - PWC równie dobrze może być stosowane przez gPT jak i sPT
    - Dzięki PWC schodząc w hierarchii stron cały czas mamy nadzieję, że uda nam się uniknąć dalszych działań

Podsumowanie nPT - analiza kosztów

- Pesymistyczny koszt obliczania adresu fizycznego dramatycznie rośnie
- Udaje się rzadko osiągać ów pesymistyczny koszt dzięki zastosowaniu tylu różnych pamięci podręcznych:
  - dwa rodzaje TLB
  - NTLB
  - PWC
- Liczbę 24 z dużą nawiązką rekompensuje nam fakt, że monitor niemal nic nie robi

Podsumowanie nPT jako pełnej wirtualizacji

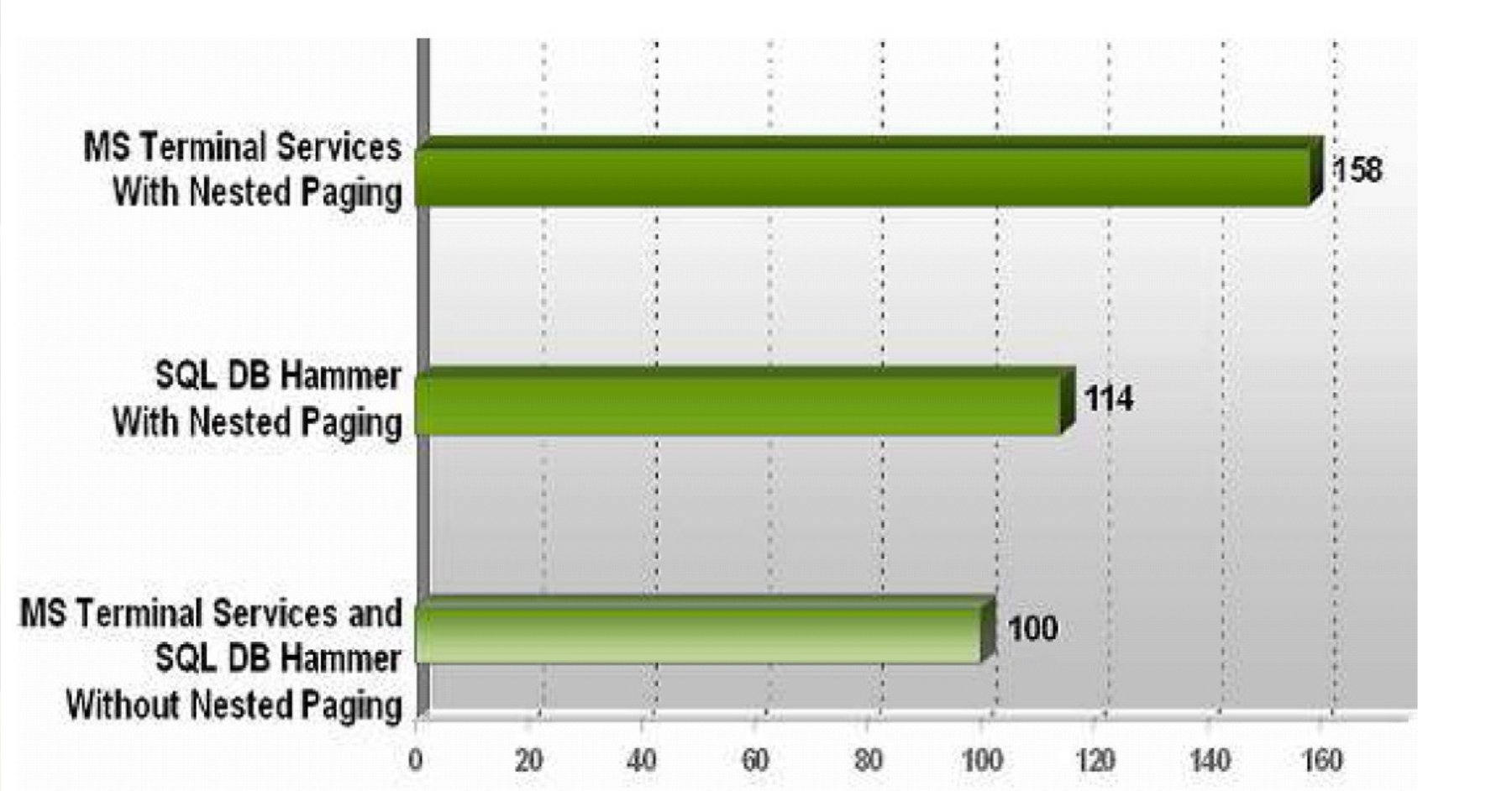
- Gość nie widzi absolutnie nic podejrzanego
- Jedyne co może go niepokoić to właśnie owo długie wyliczanie adresu fizycznego
- wniosek: nPT jest klasyczną techniką wspomaganej sprzętowo wirtualizacji pełnej

Namacalne korzyści z zastosowania nPT

- technologia nPT została poddana licznym testom, które potwierdziły jej zalety
- Statystyki, które przedstawione zostaną za moment zostały zebrane na maszynie o następujących parametrach:
  - Processors: 2-socket, 2.0GHz/1800MHz-NB Barcelona (Model 2350), 95W.
  - Memory: 32GB of 4GB DDR2-667MHz.
  - HBA: QLA2432 (dual-port PCIe 2Gb Fiber) HBA: 1 port used.
  - Disk Array: MSA1500, 1 controller, 1 fiber connection, 512MB cache. 15
  - drives 15K rpm SCSI 73GB disks.

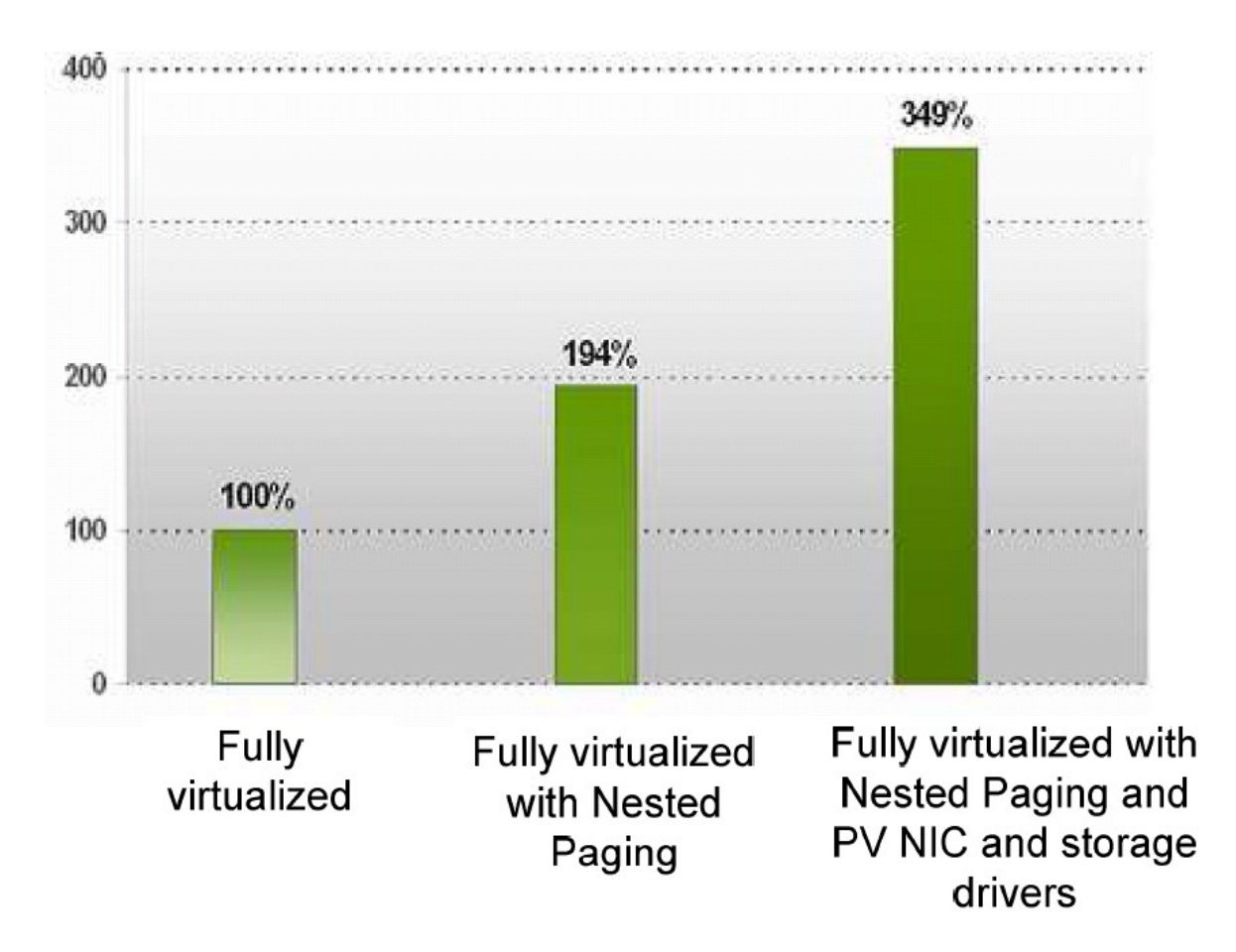
Namacalne korzyści z zastosowania nPT

- Wyniki testów używających do wirtualizacji Vmware ESX



Namacalne korzyści z zastosowania nPT

- Wyniki testów używających do wirtualizacji Xen 3.1, wirtualizujących Oracle 10G OLTP

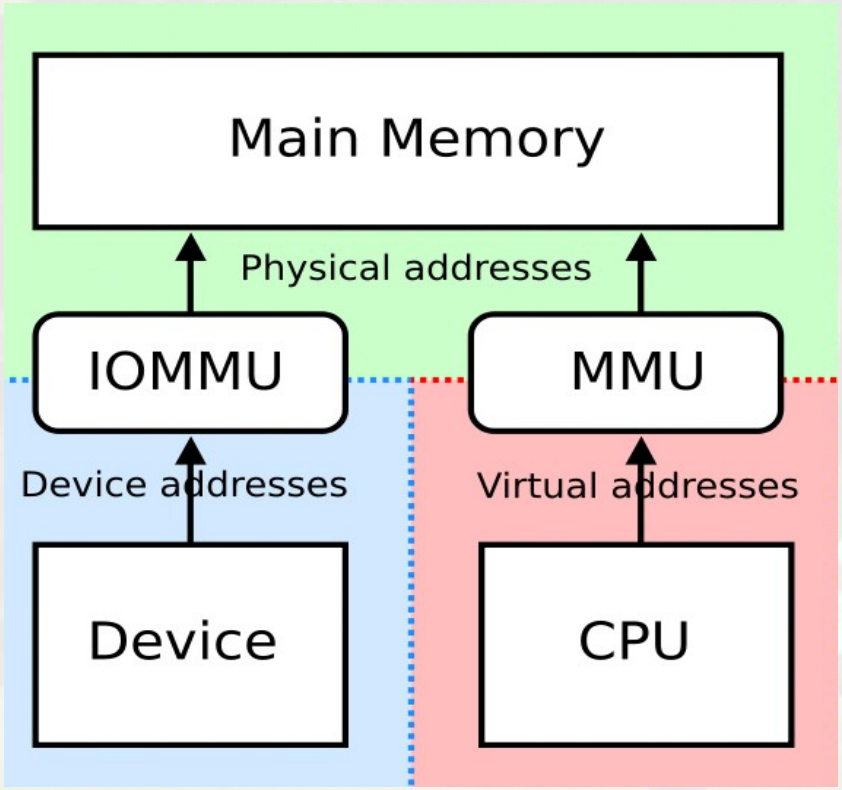


Co dalej?

- nPT jest technologią nową i jeszcze niedopracowaną – między innymi nie ma dobrego software'u implementującego monitor
- możliwe, że dobrze napisany monitor jeszcze bardziej usprawniłby działanie nPT
- Intel alternatywą dla AMD – technologia EPT (extended page tables)
  - do pewnego stopnia inne podejście
  - tak samo stosuje mnogość buforów (BTB, RSB)
  - swoją nową technologię zwaną *Nehalem* intel niedługo wprowadzi na rynek (o ile już nie wprowadził)

Słów kilka o IOMMU

- IOMMU – odpowiednik MMU dla wirtualnych adresów urządzeń



IOMMU – zalety i wady

- Zalety
  - Pomocne przy wirtualizacji
    - rozwiązuje problem, gdy gość próbuje użyć DMA
  - Obsługuje adresowanie pamięci większej niż 4GB i stanowi dzięki temu pomoc dla urządzeń zewnętrznych
  - Stosuje ochronę pamięci, gdy urządzenia zewnętrzne się źle zachowują
- Wady
  - Dodatkowy narzut czasowy związany z operacjami na IOMMU
  - Dodatkowy narzut pamięciowy dla IOPT (da się zniwelować)

XEN – Opis

- XEN jest monitorem maszyny wirtualnej dla architektur
  - x86 oraz x86-64
  - IA-32 oraz IA-64
  - PowerPC 970
  - AMD
- Stworzony na Uniwersytecie Cambridge w 2003 roku
- Dziś rozwijany przez konsorcjum, składające się z:
  - Citrix
  - IBM
  - Intel
  - Sun Microsystems
  - Hewlett-Packard

XEN – Parawirtualizacja

- XEN powstawał jako VMM oferujący parawirtualizację
  - systemy-goście mieli działać na niższych trybach uprzywilejowania niż monitor
  - nie potrzebował zmian w ABI
    - Application Binary Interface
  - oferował wsparcie dla multipexing'u aplikacji
    - jednoczesne korzystanie z tego samego kanału przesyłu danych
  - przewidywał małą skalę – do 100 maszyn wirtualnych
  - działał bardzo sprawnie

XEN – Pełna wirtualizacja

- XEN korzysta teraz z rozszerzeń VT-x oraz AMD-V
  - obie architektury są obsługiwane przez jedną abstrakcyjną warstwę
- Począwszy od wersji 3.0 XEN obsługuje niezmodyfikowane systemy (np. systemy o zamkniętym kodzie)

XEN – Pełna wirtualizacja

- Wspomaganie sprzętowe zapewnia nowe instrukcje
  - VMCALL – przerwanie w VM i oddanie kontroli do monitora
- Zapewnia nowe tryby wykonania programu
  - root mode
  - non-root mode
  - oba posiadające swoje własne zestawy Ring 0-3
- Gość nie widzi różnicy między VM a środowiskiem natywnym, ale jest ściśle kontrolowany przez monitor

XEN – Trochę o technologii

- Niektóre instrukcje zawsze powodują przerwanie w VM
  - RDMSR
  - WRMSR
  - CPUID
- Inne przerywają tylko dla odpowiednich ustawień kontroli
  - HLT
- Wykorzystywane są dwie mapy bitowe
  - mapa wyjątków
  - dwie mapy I/O
    - mapa A dla portów 0000-7999
    - mapa B dla portów 8000-FFFF
- VM exit obsługiwany jest przez vmx_vmexit_handler ()
- Istnieją 43 podstawowe przerwania, do wglądu w vmx.h

XEN – Ideologia

- System działający na maszynie wirtualnej korzysta bezpośrednio z zasobów procesora i pamięci.
- Pojawia się instrukcja powodująca przerwanie.
- Procesor jest w trybie non-root, zatem następuje przerwanie.
- Wywołany zostaje VM exit i kontrola zostaje oddana do monitora.
- Monitor w sposób bezpieczny wykonuje instrukcję.
- Kontrola zostaje oddana do procesora.
- Procesor wznawia pracę.

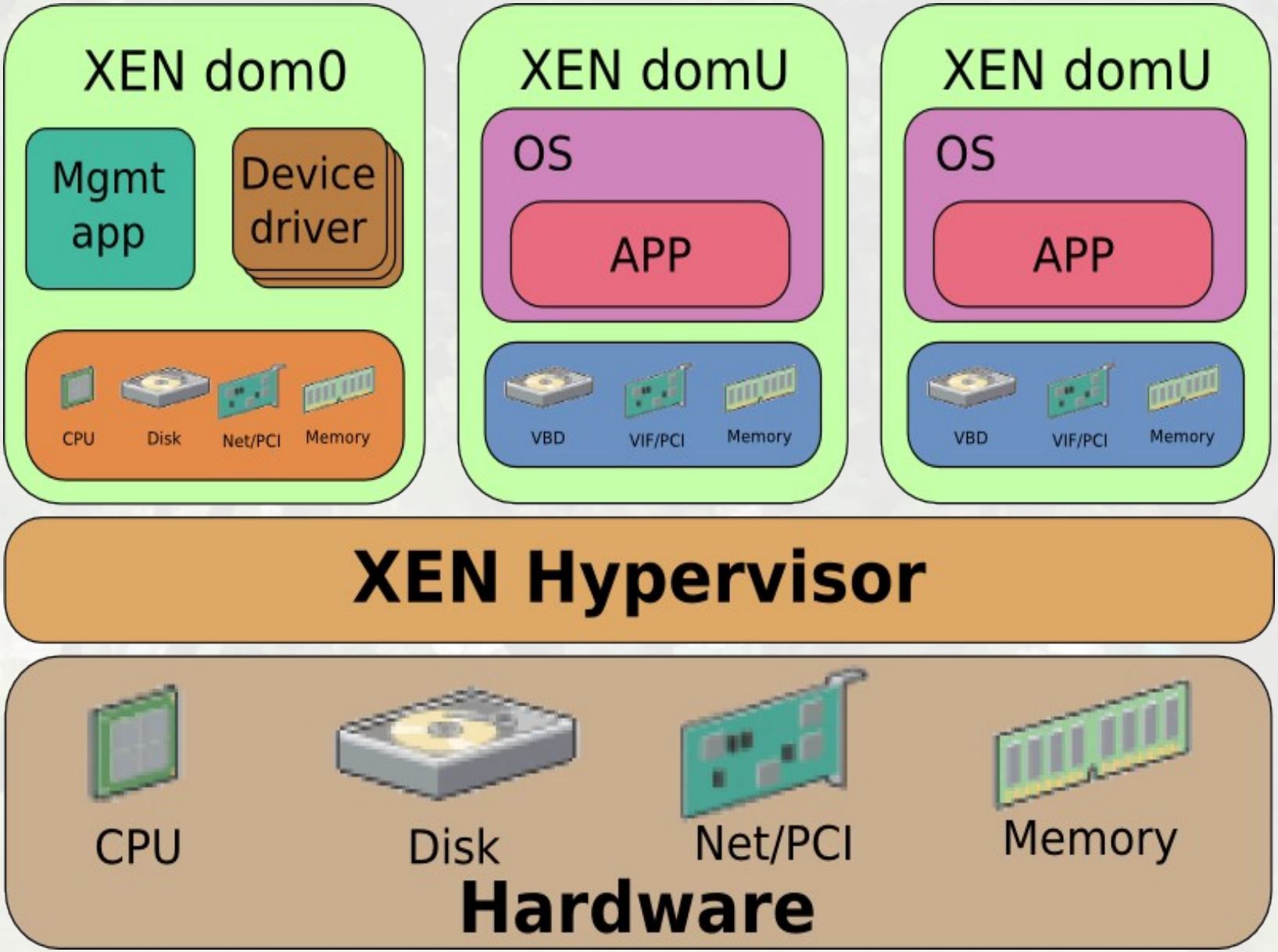
XEN – Migracja

- Jedna z najciekawszych możliwości XEM
- Migracja odbywa się „na żywo”
  - podczas działania maszyny następuje iteracyjne kopiowanie danych poprzez sieć LAN
  - następuje zatrzymanie maszyny i synchronizacja danych
    - ta część trwa zwykle bardzo krótko (~300ms)
  - wznowienie pracy już przez drugą maszynę

XEN – Zasada działania

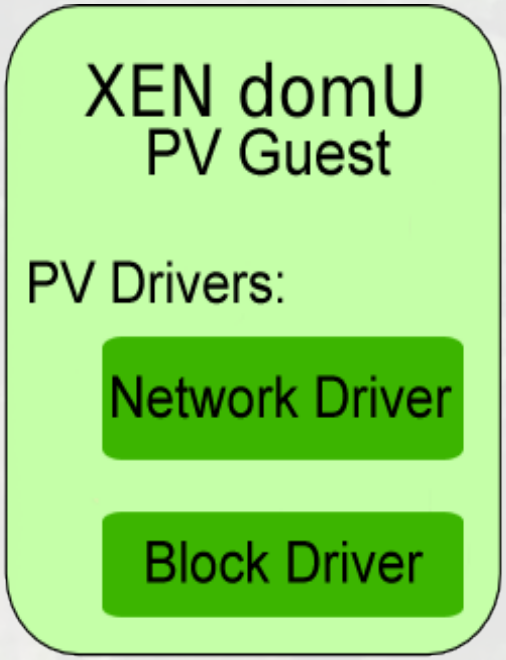
- XEN działa w dwóch warstwach
  - niższa – monitor, o wyższych uprawnieniach
  - wyższa – maszyny wirtualne
- Wyższa warstwa dzieli się na dwie części
  - dom0 – pierwsza wirtualna maszyna
  - domU – kolejne wirtualne maszyny

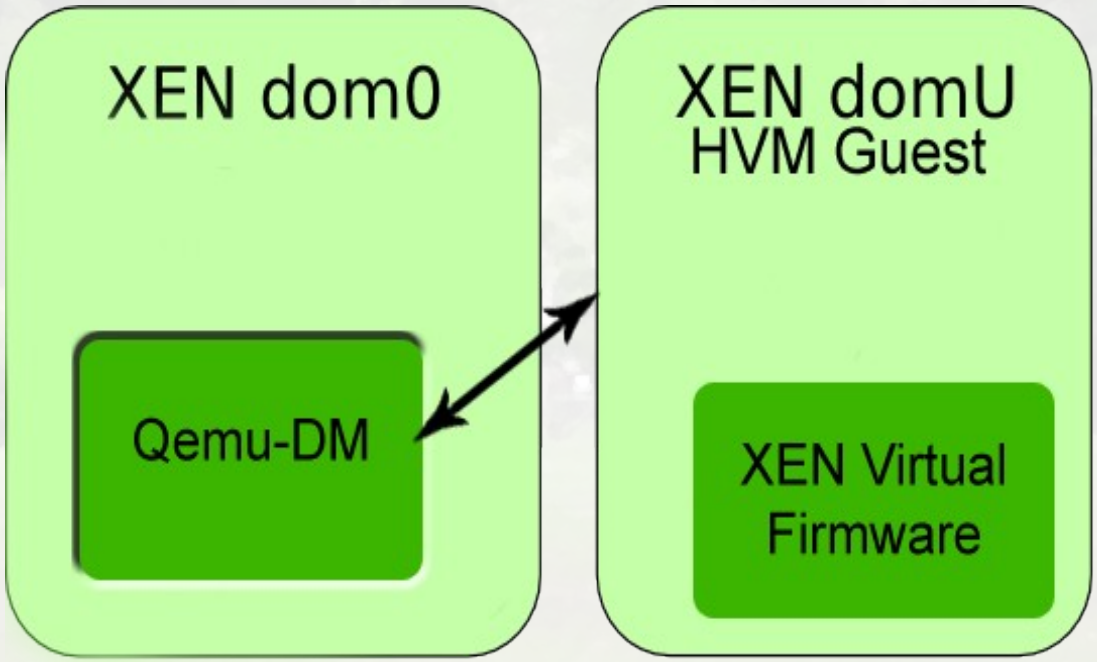
XEN – Schemat


XEN dom0
Mgmt app
Device driver
CPU
Disk
Net/PCI
Memory
XEN domU
OS
APP
VBD
VIF/PCI
Memory
XEN domU
OS
APP
VBD
VIF/PCI
Memory
XEN Hypervisor
CPU
Disk
Net/PCI
Memory
Hardware

XEN – Dostęp do zasobów

- XEN PV
  - maszyna parawirtualna posiada własne sterowniki
- XEN HVM
  - HVM korzysta z działającego w tle dom0 daemona





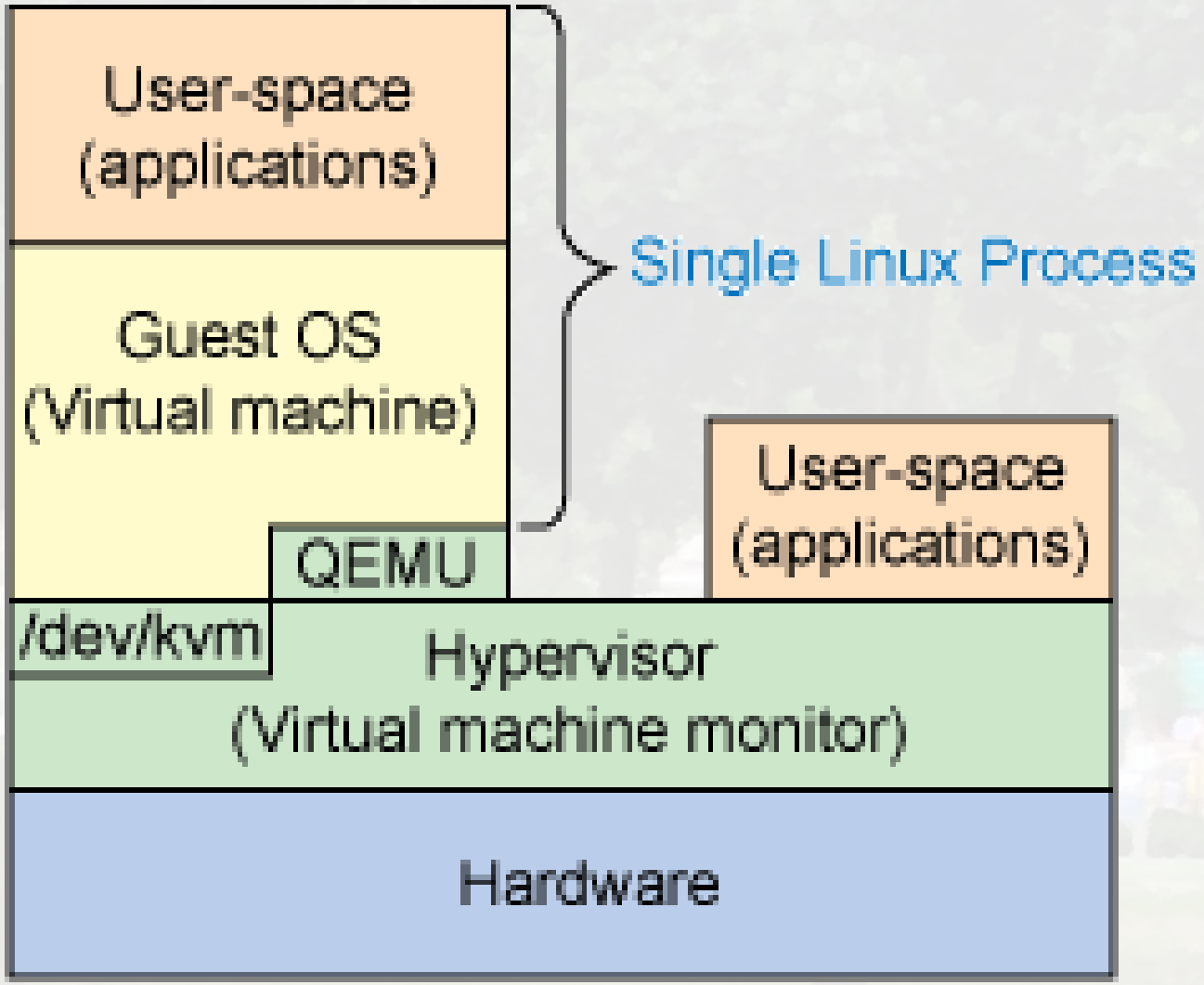
KVM – Opis

- Kernel-based Virtual Machine
  - podejście stosunkowo nowe
  - nie tyle metoda wirtualizacji, co część większego rozwiązania
  - korzysta z Intel VT oraz AMD-V
  - niewielki, ok. 10 000 linii kodu
    - w przyszłości ma być funkcja systemową

KVM – Zasada działania

- Zamienia jądro natywnego linuksa w monitor VM
- Systemy-goście są tworzeni w /dev/kvm
  - każdy gość traktowany jest jak pojedynczy proces linuksa
  - każdy ma swoją, oddzieloną od innych przestrzeń adresową
- KVM korzysta z zasobów natywnego linuksa
  - wirtualizacja pamięci odbywa się poprzez /dev/kvm
    - wsparcie procesora dla niemapowanych adresów
  - wirtualizacja I/O odbywa się poprzez QEMU
    - QEMU – emulator symulujący całe środowisko komputera
- Nowy tryb: guest

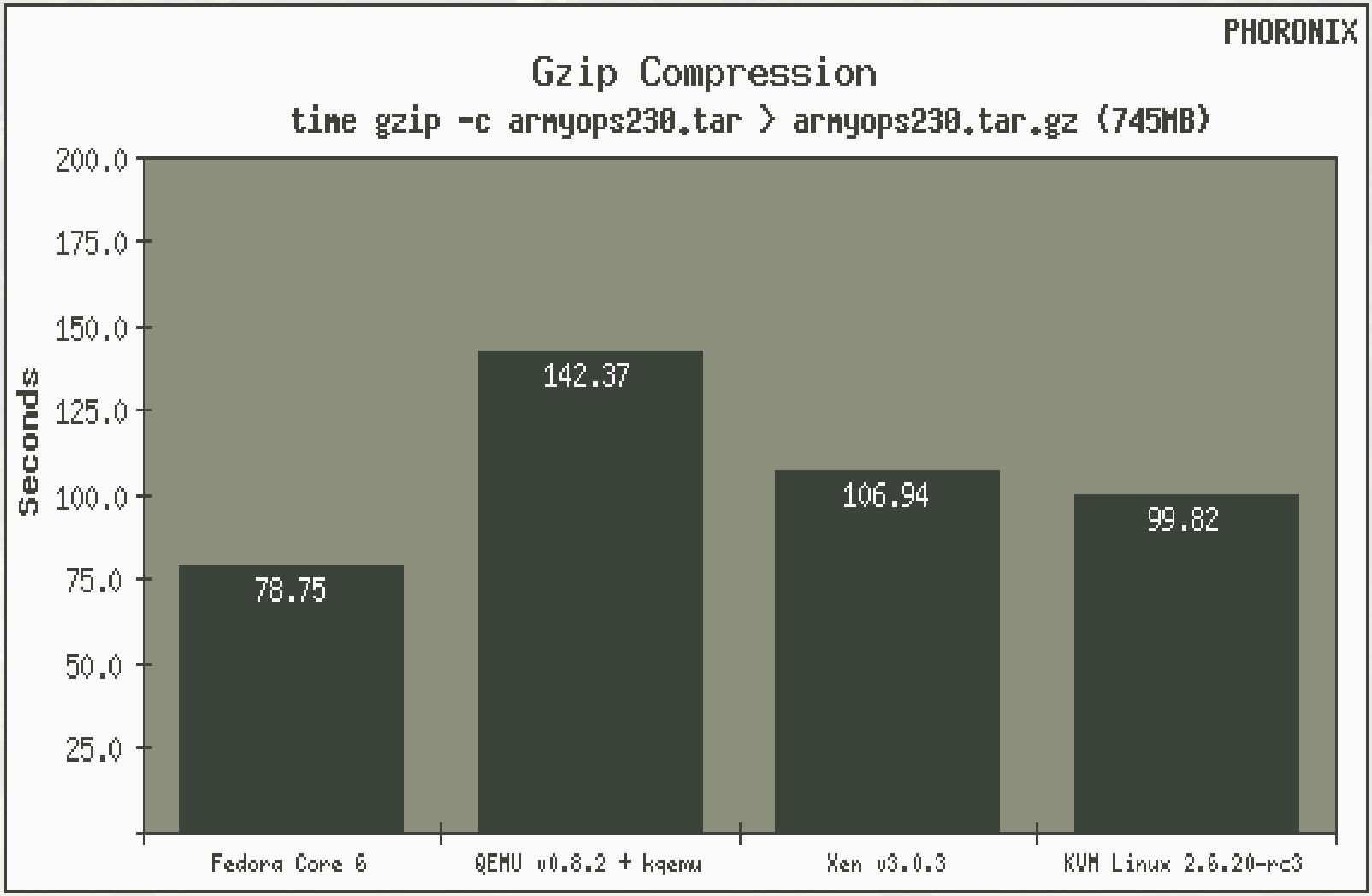
KVM – Schemat


User-space
(applications)
Guest OS
(Virtual machine)
QEMU
Single Linux Process
User-space
(applications)
/dev/kvm
Hypervisor
(Virtual machine monitor)
Hardware

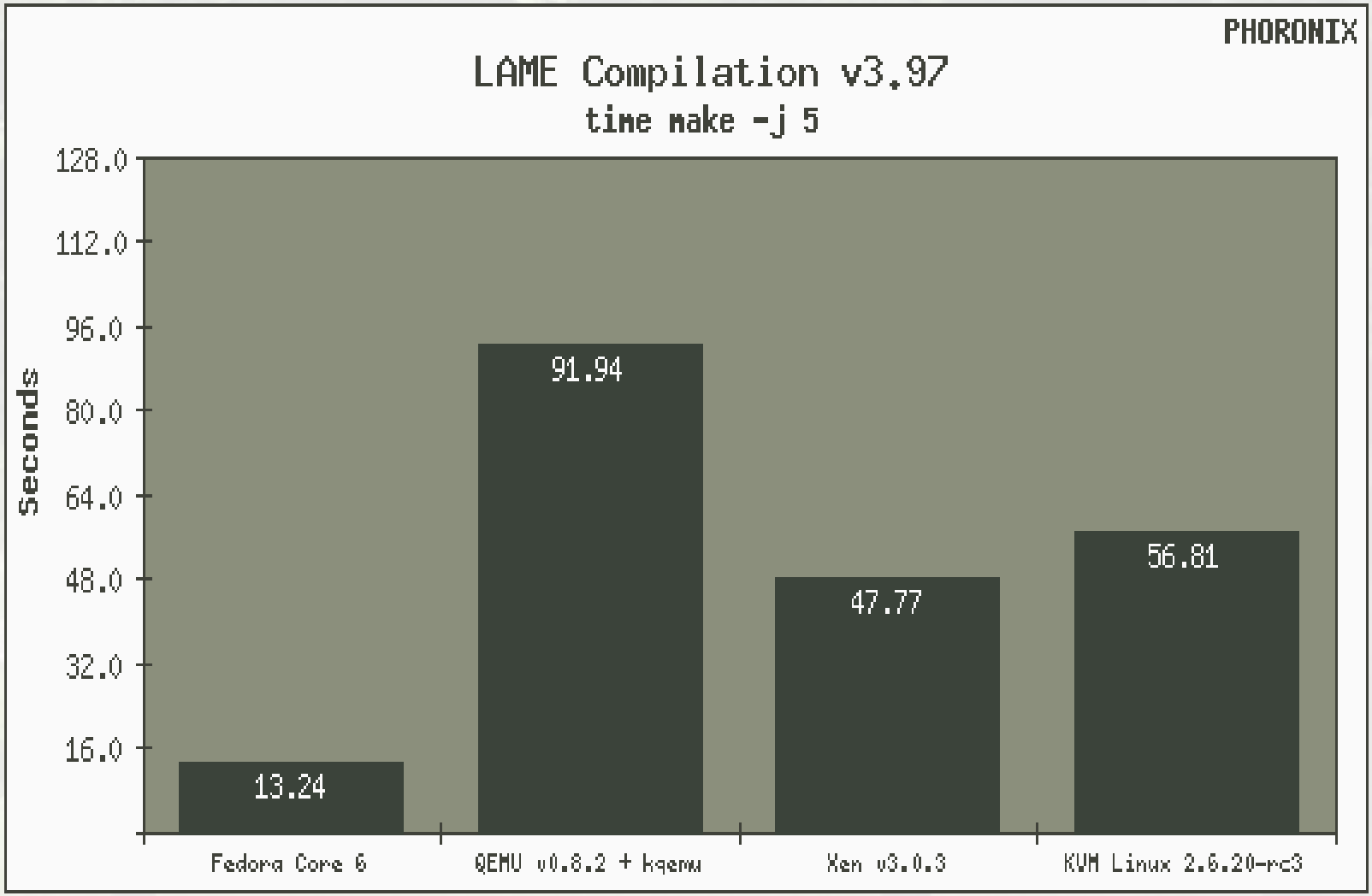
XEN i KVM – Obsługiwane systemy

- XEN
  - Windows XP
  - OpenBSD
  - OpenSolaris
  - Plan 9 (Bell Labs)
  - Linux
- KVM
  - Windows Server
  - Windows Vista
  - Windows XP
  - Windows 2000
  - Red Hat (poza 7 i 8)
  - Fedora
  - Ubuntu i Debian
  - SUSE
  - Gentoo

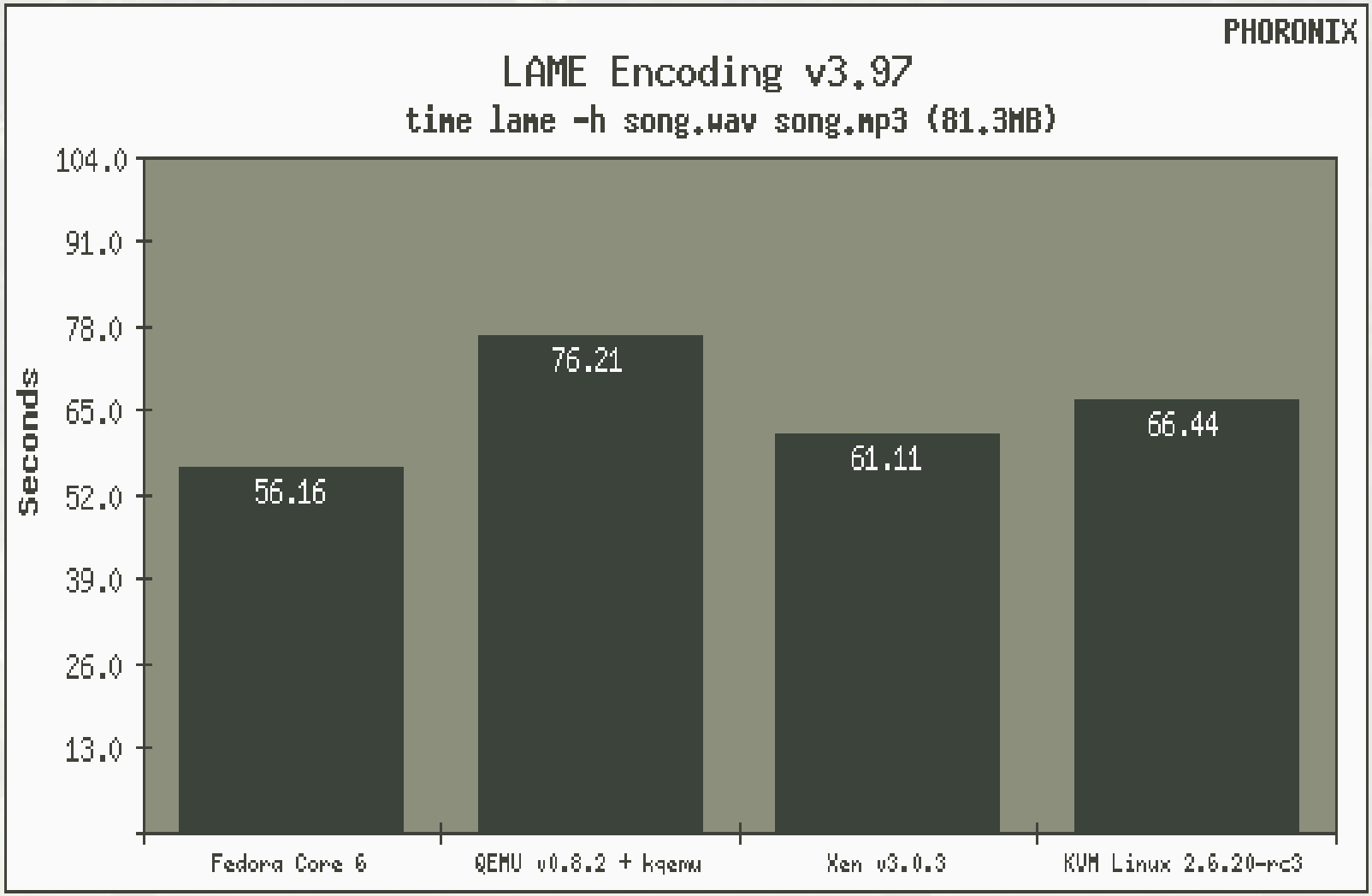
XEN i KVM – Testy



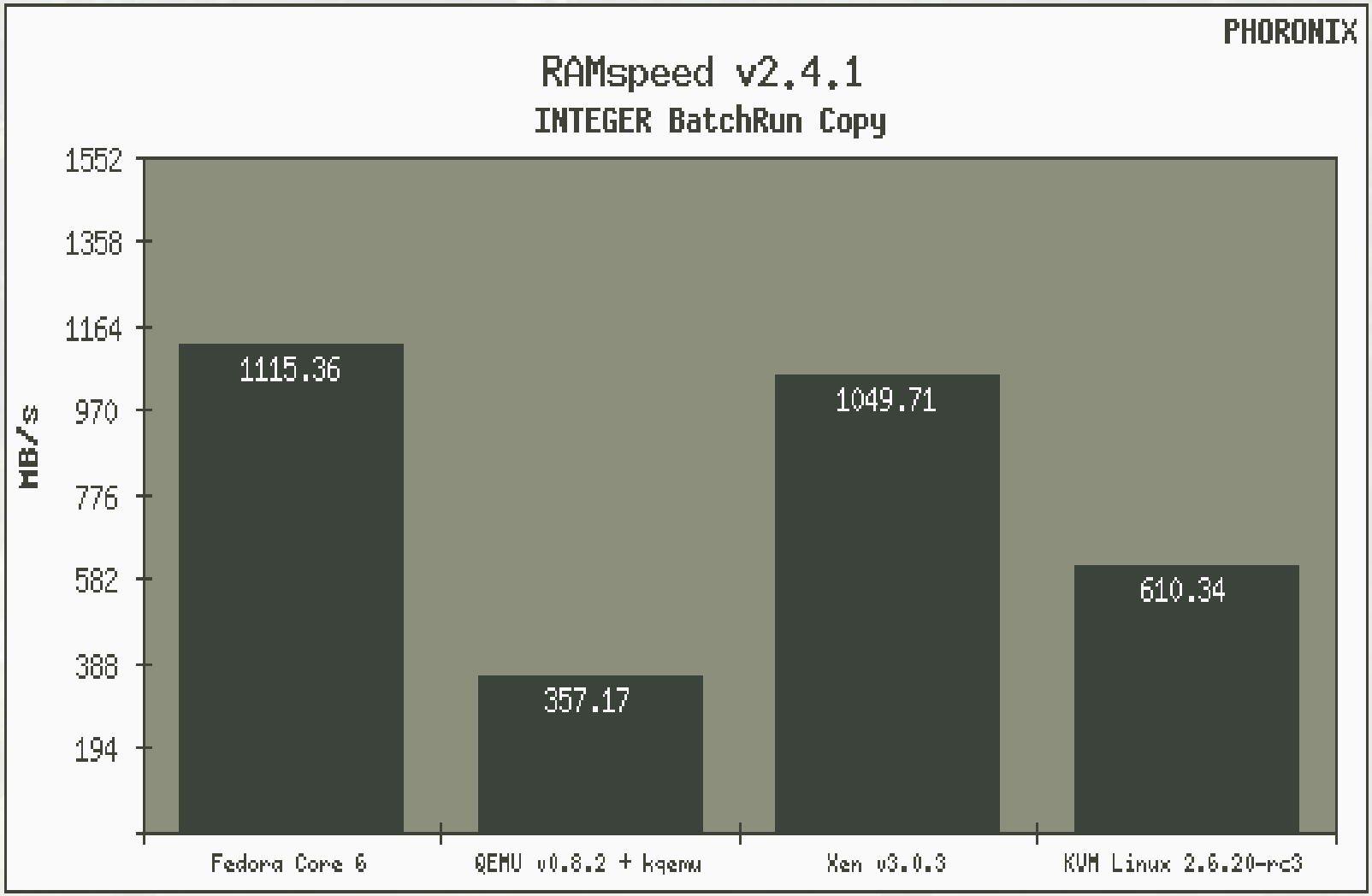
XEN i KVM – Testy



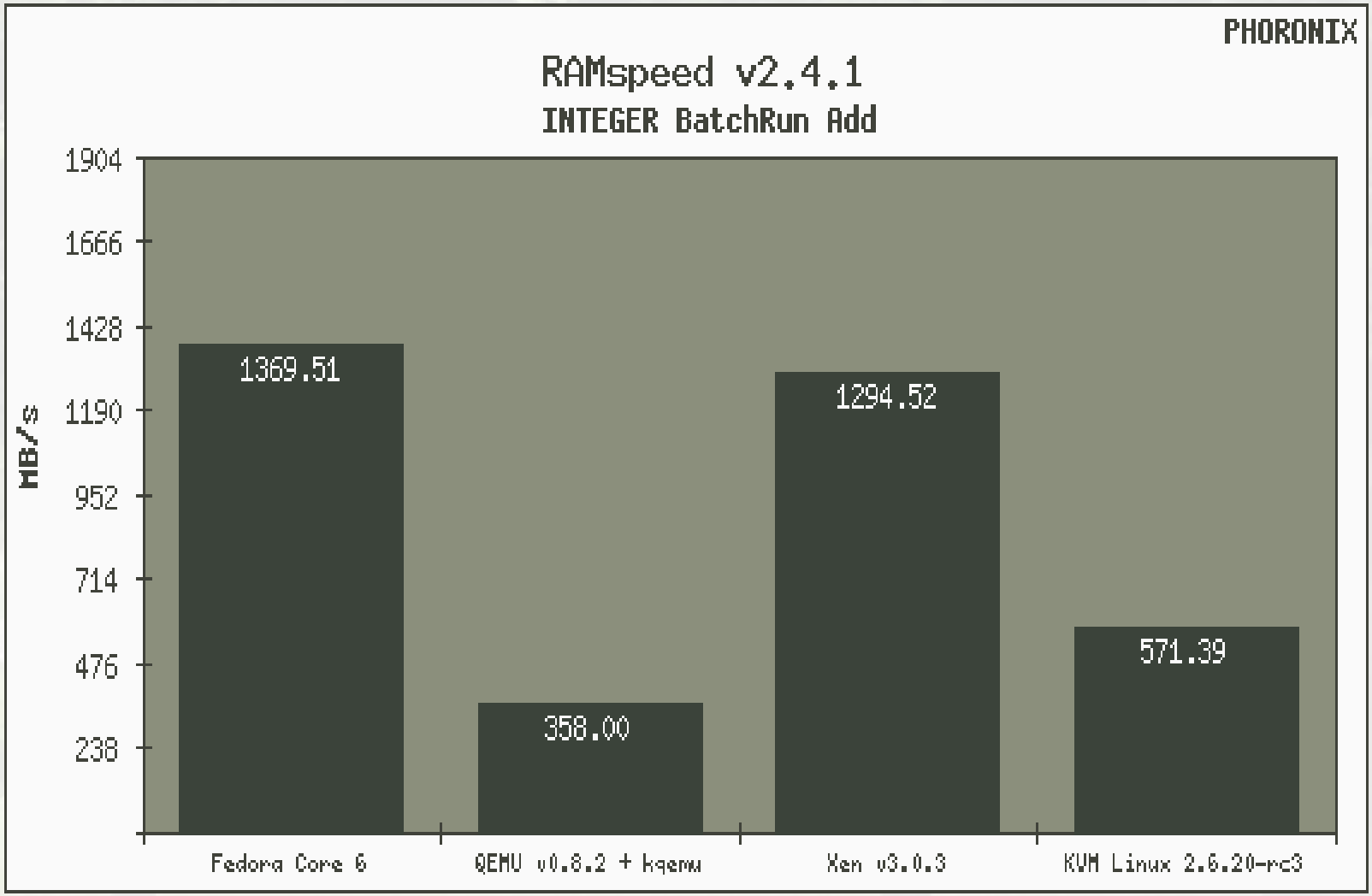
XEN i KVM – Testy



XEN i KVM – Testy



XEN i KVM – Testy



Niebezpieczeństwa wirtualizacji

- Malware musi wybierać pomiędzy siła a niewykrywalnością
- Wirtualizacja udostępnia wiele możliwości ukrycia się
- Łatwiej uzyskać całkowitą kontrolę nad komputerem
- Trudniej pozbyć się malware

Hardware VM Rootkits

- Rootkit – malware, którego zadaniem jest przejęcie kontroli nad komputerem
- Strategia działania
  - rozpoczyna prace w jądrze
  - instaluje rootkit hypervisor
  - zdobywa dla siebie trochę pamięci
  - migruje natywny system do VM
  - przejmuje dostęp do monitora
- Prosty w budowie i działaniu
  - rozpoczyna jako ładowalny moduł jądra
  - korzysta głównie z trzech funkcji
- Teoretycznie jest niewykrywalny

Przeciwdziałanie

- Aby przeciwdziałać trzeba rozpoznać, że zaistniał problem
  - żaden rejestr ani bit nie przechowują informacji czy system działa natywnie czy jako VM
- Sposoby wykrywania problemu
  - stworzenie nowej maszyny wirtualnej
  - timing attack

Przykłady

- Blue Pill
  - atakuje Microsoft Windows Vista na AMD Pacifica
    - podobno można przeportować na x86
- Vitriol
  - atakuje MacOS X na Intel VT-x (Intel Core Duo/Solo)
- SubVirt
  - atakuje Windows oraz Linux

Bibliografia

- http://www.floobydust.com/virtualization/lawton_1999.txt
- http://en.wikipedia.org/wiki/Popek_and_Goldberg_virtualization_requirements
- http://en.wikipedia.org/wiki/Partial_virtualization
- http://en.wikipedia.org/wiki/Full_virtualization
- http://en.wikipedia.org/wiki/Platform_virtualization
- http://pl.wikipedia.org/wiki/Wirtualizacja
- http://www.virtualization.info/2008/03/intel-to-introduce-new-virtualization.html
- http://software.intel.com/en-us/articles/
- http://developer.amd.com/assets/NPT-WP-1%201-final-TM.pdf
- http://www.intel.com/pressroom/archive/reference/whitepaper_Nehalem.pdf
- http://en.wikipedia.org/wiki/IOMMU
- http://www.webopedia.com/DidYouKnow/Computer_Science/
- http://www.xen.org/
- http://kvm.qumranet.com/kvmwiki
- http://www.phoronix.com