Czym jest wirtualizacja

Wirtualizacja

- Szerokie pojęcie odnoszące się do abstrakcji zasobów w różnych aspektach komputeryzacji.

Odnosi się do:

- procesorów
- pamięci
- języków programowania

Cele wirtualizacji

- Niezależność od sprzętu - nie mając fizycznego dostępu do danego sprzętu, możemy emulować jego działanie
- Testowanie i bezpieczeństwo - możemy testować potencjalnie niebezpieczne programy i funkcje bez obawy o utratę danych, ponieważ stan OS, można zapisać; można poza tym w dość latwy sposób testować i debuggować działanie jądra systemu

Cele wirtualizacji

- Kilka OS na jednej maszynie - w prosty sposób, bez konieczności dzielenia dysku twardego na partycje i żmudnej instalacji, możemy mieć na jednym komputerze kilka OS
- Wygoda - nie ma potrzeby ciągłego restartowania systemu w przypadku potrzeby użycia innego OS
- Izolacja OS i użytkowników - OS-y nie wpływają na swoją pracę; podobnie użytkownicy, mający rozłączne środowiska

Historia

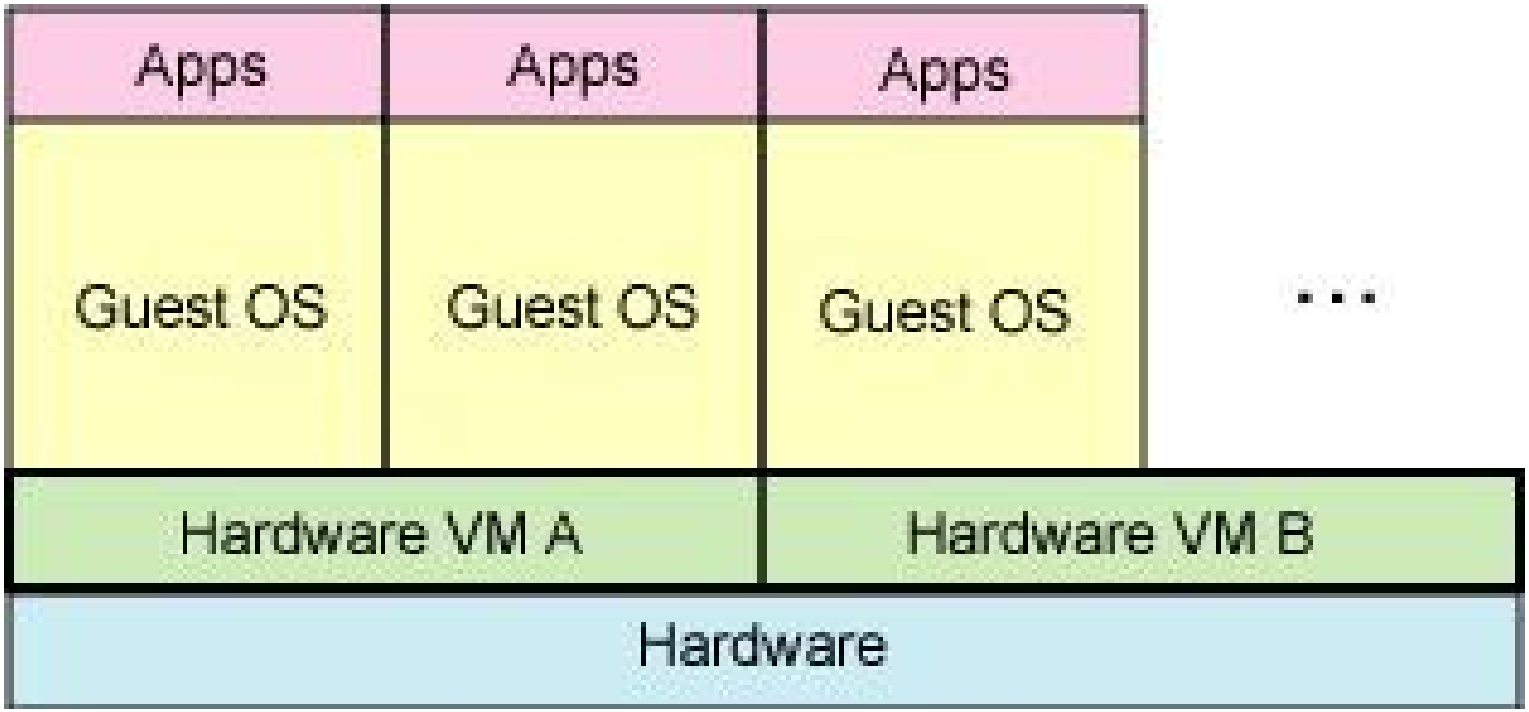
- IBM- lata '60 - projekt M44/44X - fizyczna maszyna M44 i wiele maszyn wirtualnych 44X, operujące na pamięci M44 (wieloprogramowanie i pamięć wirtualna); model 67 poprzez Monitor Wirtualnej Maszyny emulował hardware
- Wirtualizacja procesora- lata '70 - prekursor Wirtualnej Maszyny Javy; maszyna P-code, powstał kompilator, który przekształcał programy w Pascalu na programy P-code, które mogły być uruchamiane na wirtualnej maszynie
- Wirtualizacja na komputerach domowych- od lat '90 do dzisiaj - wraz z rozwojem komputerów domowych zaczęła być na nich możliwa wirtualizacja; powstaje firma VMware; Microsoft odkupuje Virtual PC

Wirtualizacja, a emulacja

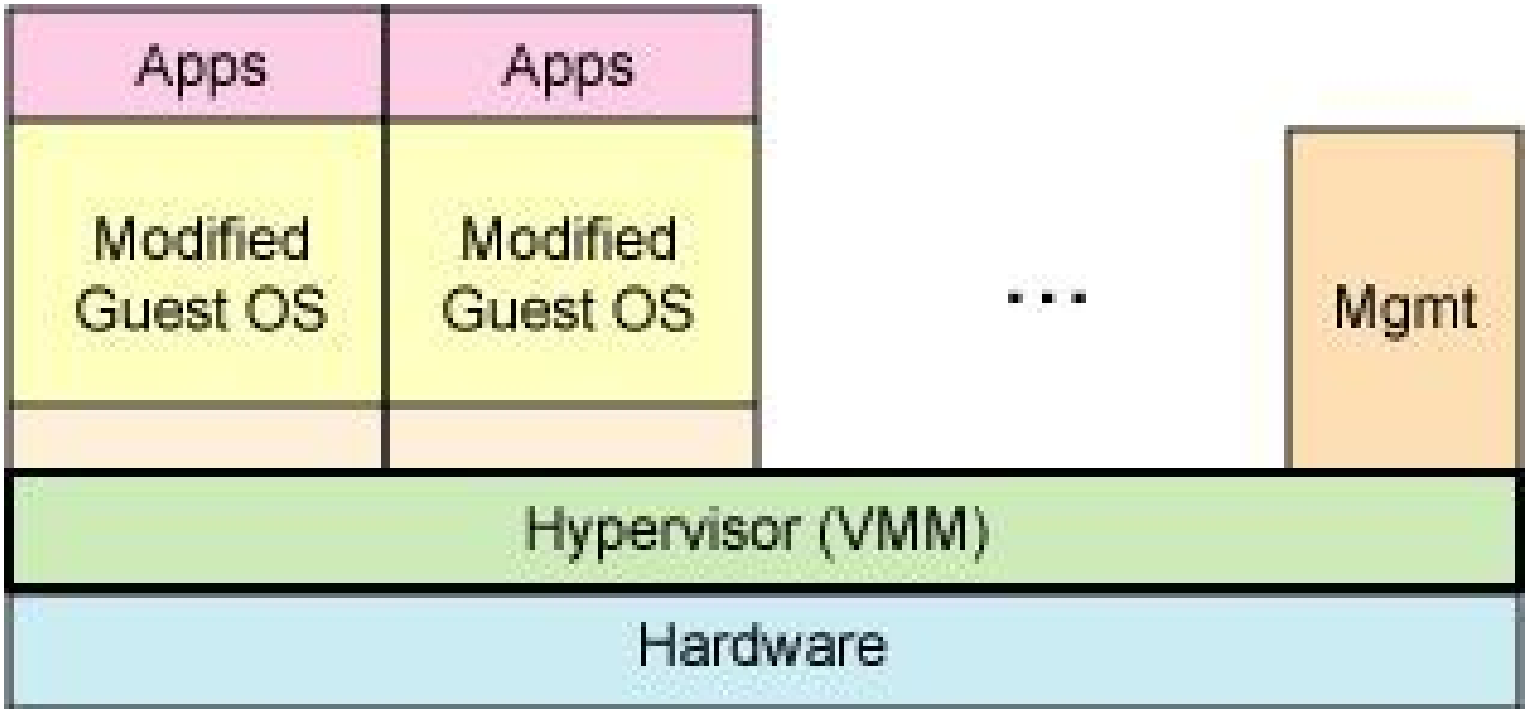
- Emulacja - emulowane jest działanie wszystkich podzespołów maszyny, w związku z czym można udawać, że się ma inny sprzęt niż w rzeczywistości
- Emulacja API - do systemu operacyjnego dodawany jest interfejs z innego systemu, dzięki czemu aplikacje mogą sie uruchamiać, tak jak w swoim macierzystym OS
- Wirtualizacja - wszystkie instrukcje, które mogą być wykonane bezpośrednio na sprzęcie są tak wykonywane; emulowane jest jedynie działanie instrukcji uprzywilejowane

- Parawirtualizacja - polega na zmianie jądra systemu operacyjnego, tak, by wspomagało ono przechwytywanie przez hypervisora instrukcji uprzywilejowanych
- Pelna wirtualizacja - system działa niezmieniony, to hypervisor jest odpowiedzialny za wykrywanie, przechwytywanie i emulowanie instrukcji uprzywilejowanych
- Wirtualizacja z poziomu OS - wirtualizowanie bezpośrednio z poziomu OS, można tworzyć niezależne serwery i środowiska użytkownika

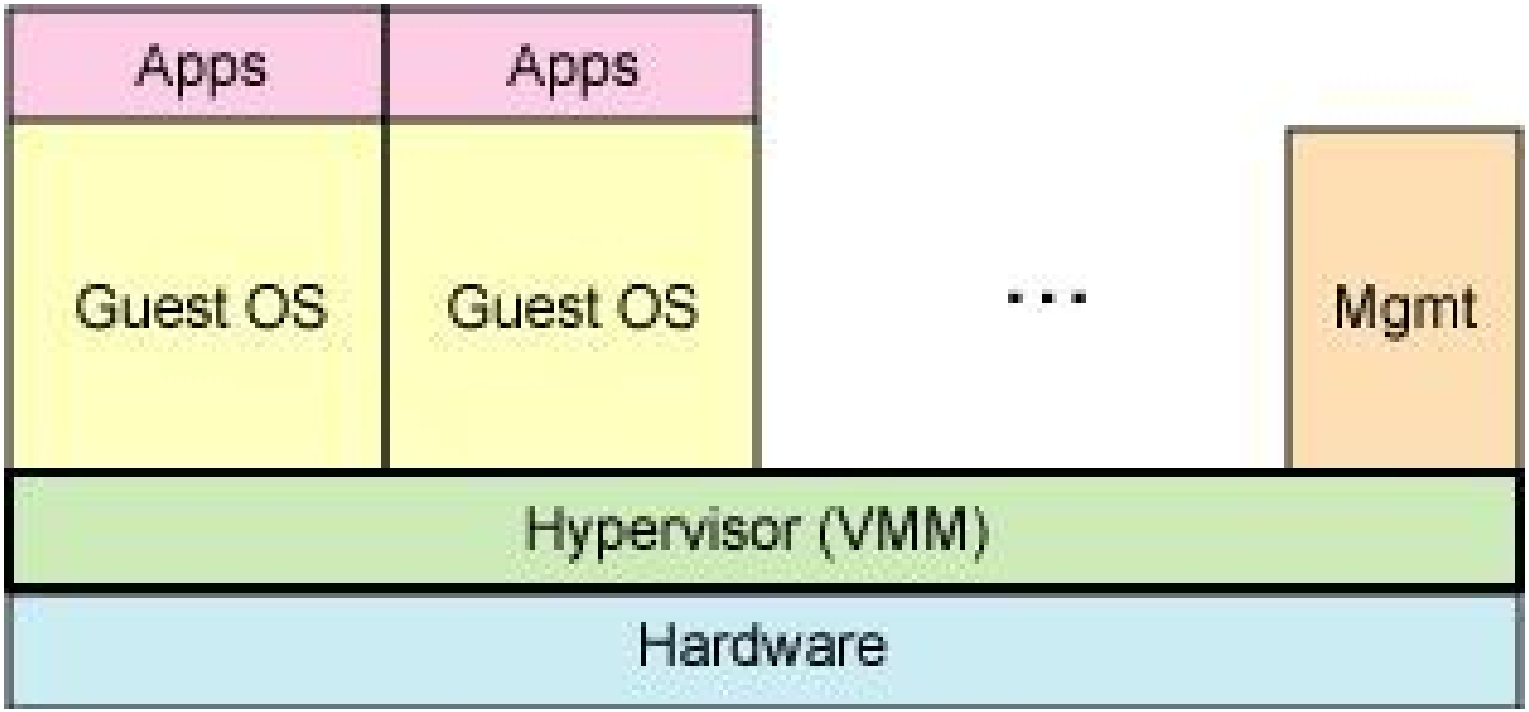
Typy wirtualizacji
Emulacja



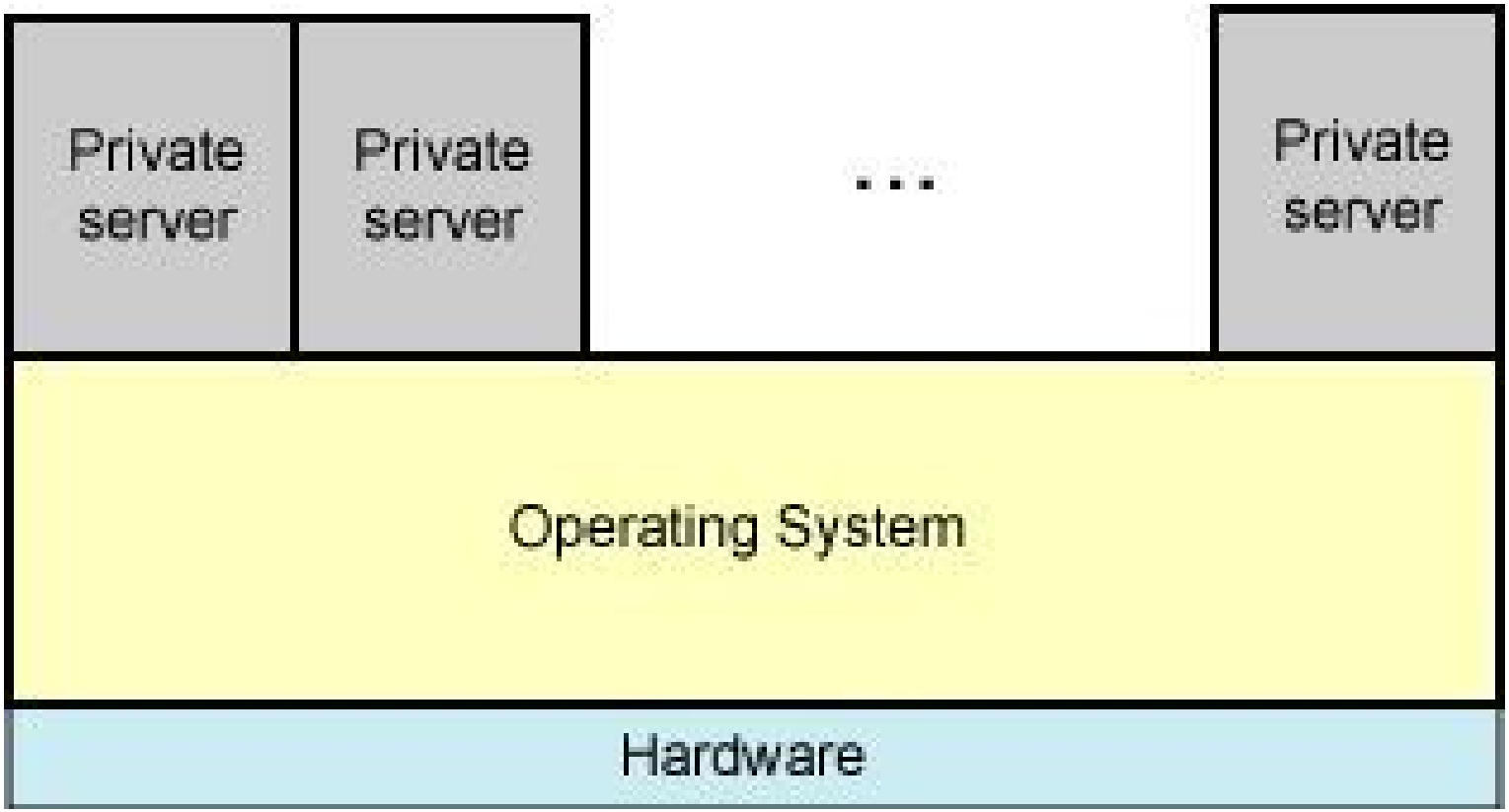
Typy wirtualizacji
Parawirtualizacja



Typy wirtualizacji
Pełna wirtualizacja



Typy wirtualizacji
Wirtualizacja z poziomu OS



Przyklady

- Bochs (emulacja)
- QEMU (emulacja)
- Wine (emulacja API)
- VMware Workstation/Server (pełna wirtualizacja)
- VirtualBox (pełna wirtualizacja)
- VirtualPC (pełna wirtualizacja)
- XEN (parawirtualizacja)
- UML (parawirtualizacja)
- Linux-VServer (wirtualizacja z poziomu OS)
- OpenVZ (wirtualizacja z poziomu OS)

Kryterium Popka-Goldberga

Warunki, które powinna spełniać wirtualna maszyna

- odpowiedniość - program na wirtualnej maszynie ma się zachowywać, tak jakby działał na rzeczywistym sprzęcie
- kontrola zasobów - wirtualna maszyna powinna w pełni kontrolować wszystkie zasoby, które są wirtualizowane
- wydajność - większośc instrukcji musi być wykonywana bez udziału wirtualnej maszyny

Instrukcje uprzywilejowane

- Powodują przerwania lub wywołania systemowe.

Instrukcje wrażliwe

- Mogą zmienić konfigurację zasobów systemu operacyjnego, bądź też korzystają z tej konfiguracji.

Twierdzenie Popka-Goldberga

- Dla każdego standardowego komputera trzeciej generacji wirtualna maszyna może zostać skonstruowana, jeśli zbiór instrukcji wrażliwych jest podzbiorem zbioru instrukcji uprzywilejowanych.

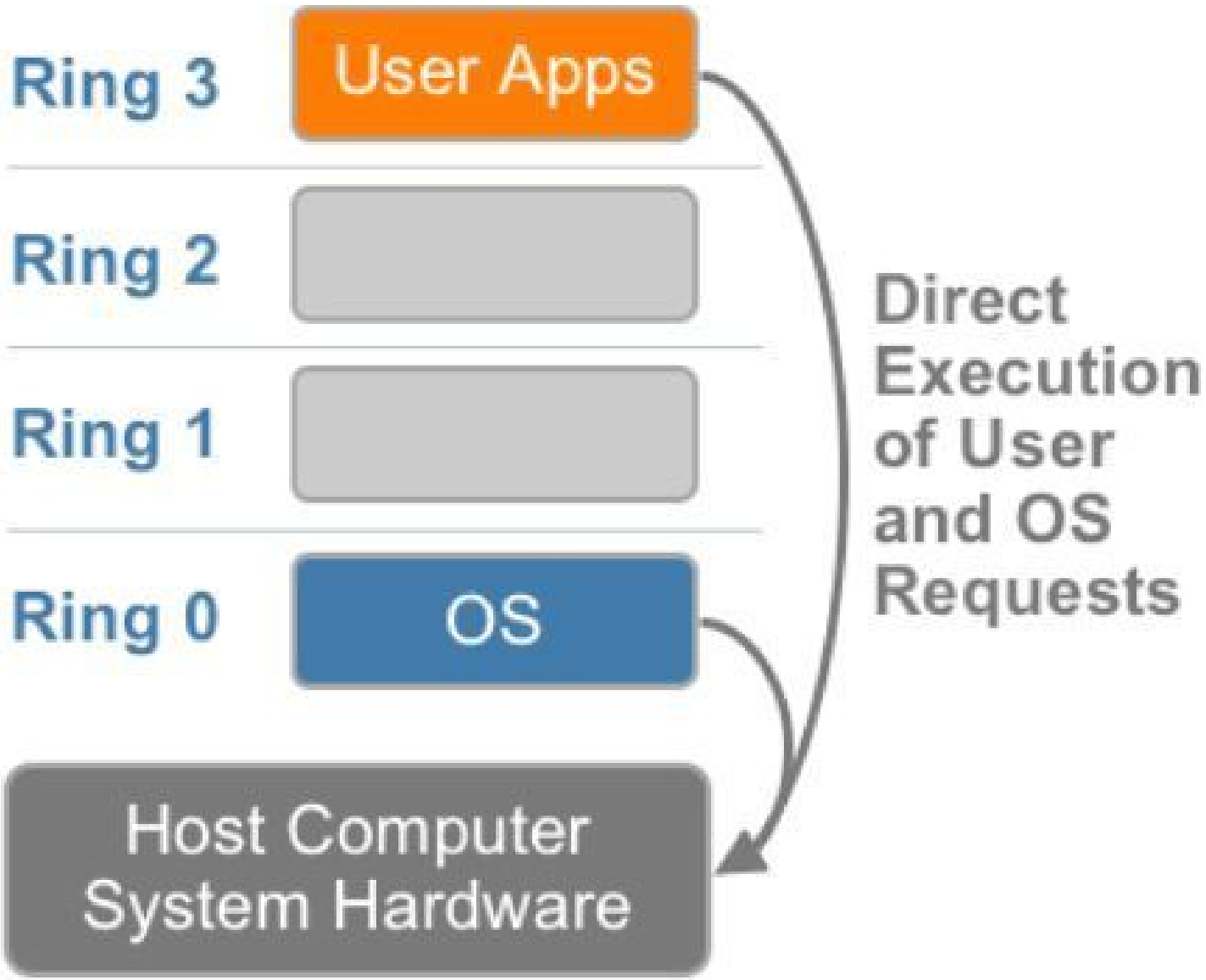
Procesy mogą wykonywać się na 4 poziomach uprzywilejowania

- Ring 3 - aplikacje użytkownika
- Ring 2 i ring 1 - sterowniki urządzeń
- Ring 0 - system operacyjny

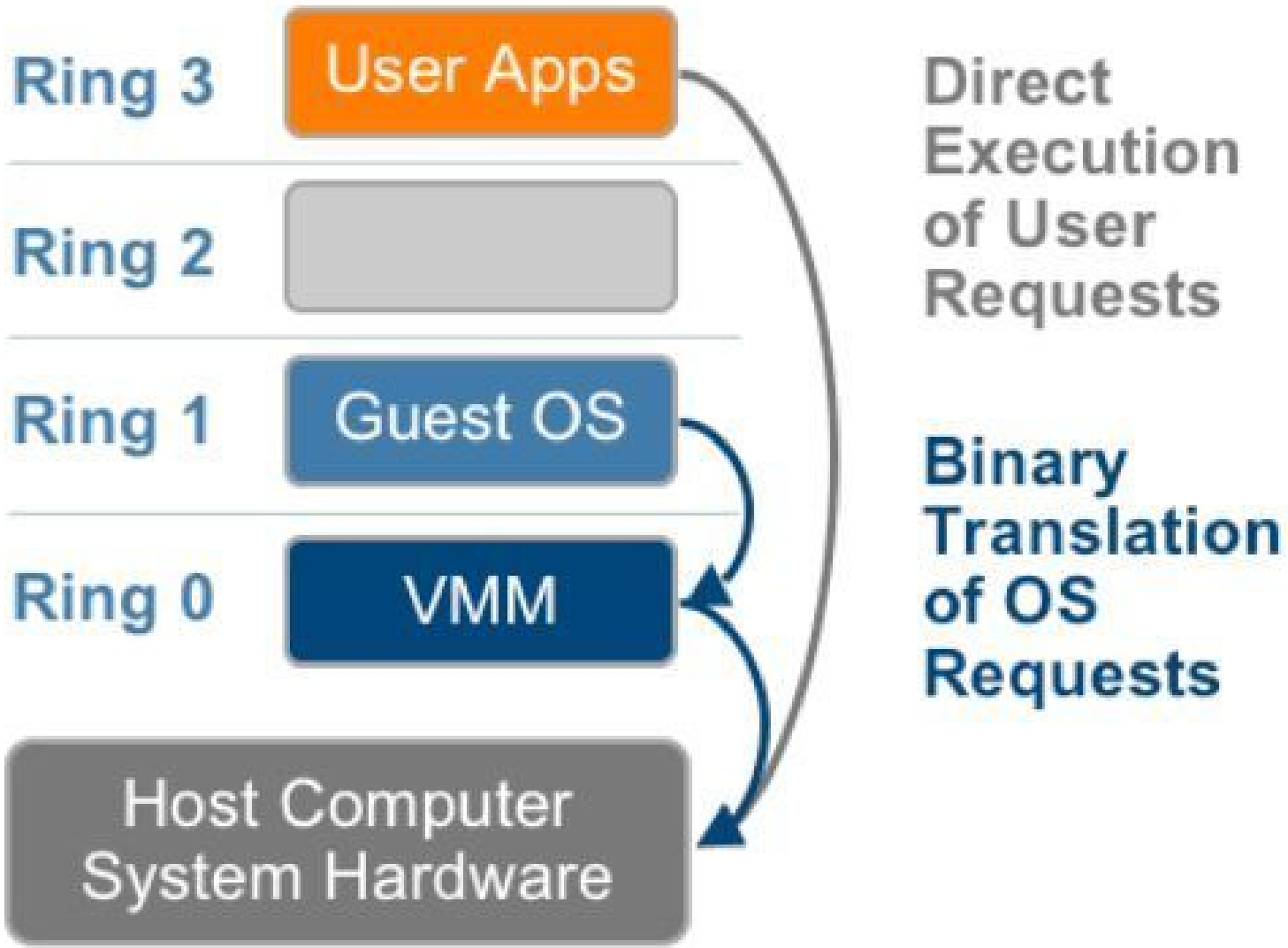
Poziomy uprzywilejowania, a wirtualizacja

- W przypadku działania wirtualnej maszyny to hypervisor działa w ring 0, a OS jest przeniesiony do ring 1, co jest dla niego sytuacją nietypową. Hypervisor musi przechwytywać instrukcje, które można wykonywać jedynie w ring 0.

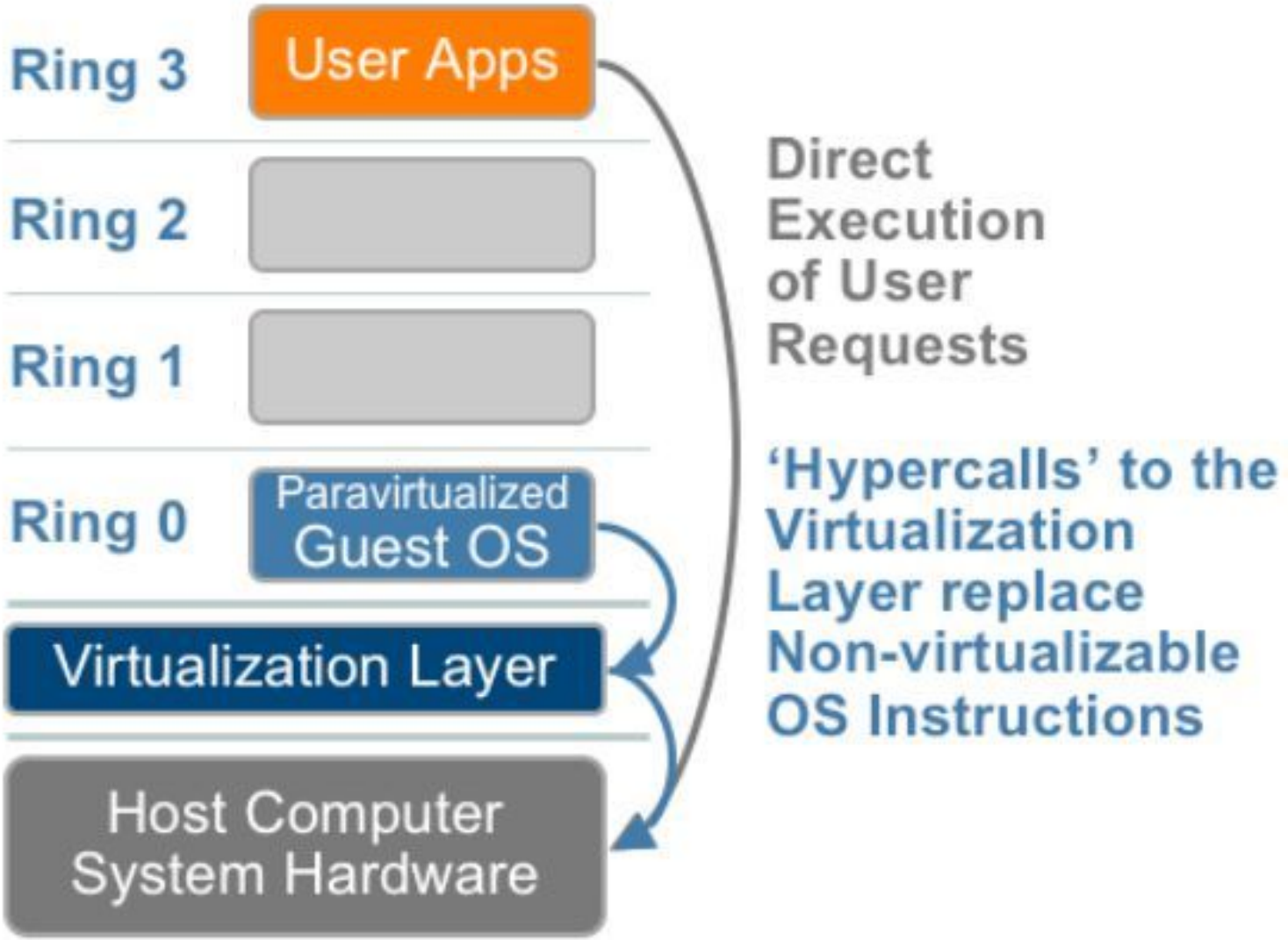
Problemy z wirtualizacją
OS na zwykłej maszynie



Problemy z wirtualizacją
Pełna wirtualizacja- podmiana binariów



Problemy z wirtualizacją
Parawirtualizacja



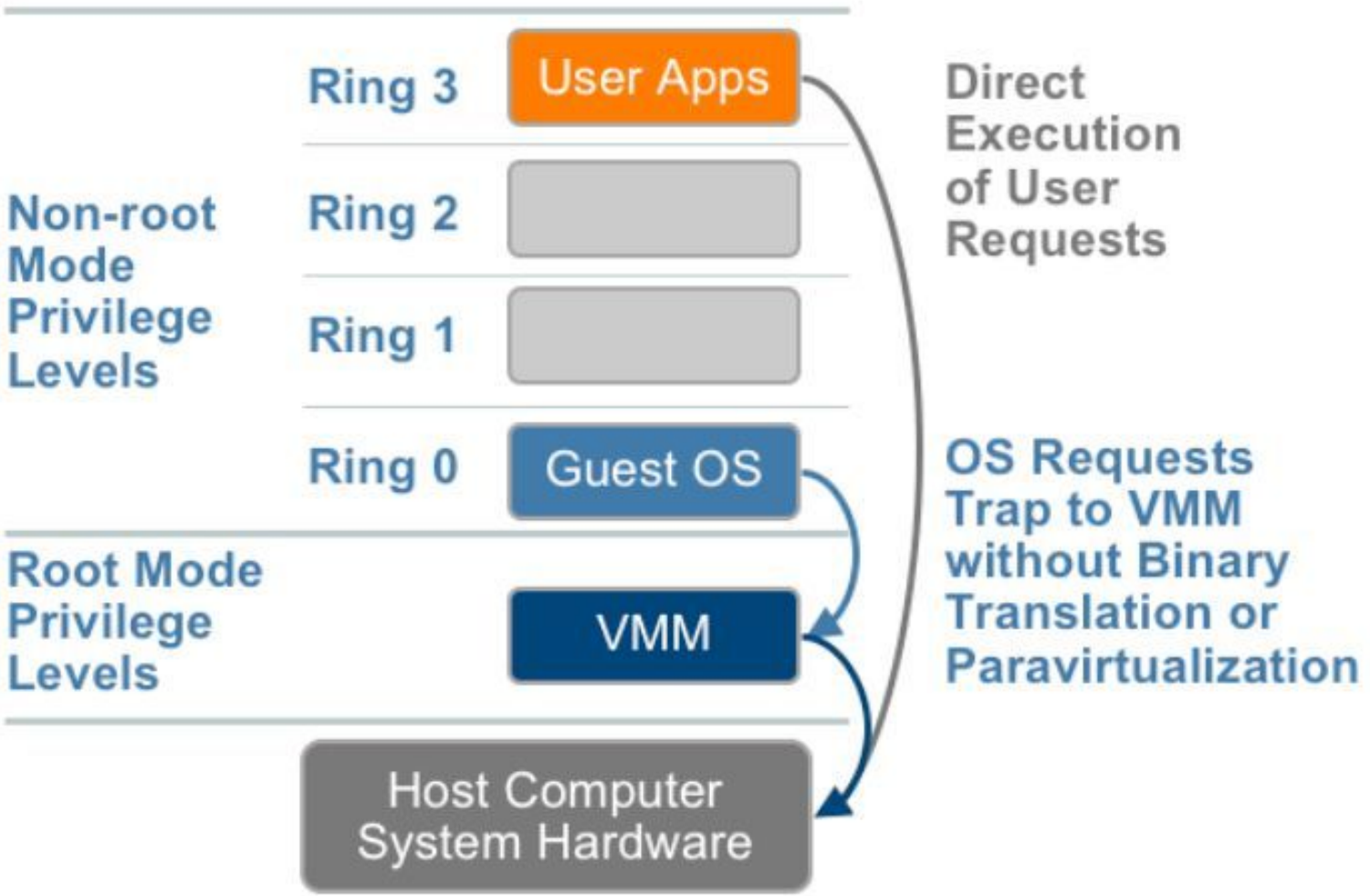
Ring -1

- Rozwiązanie problemu instrukcji uprzywilejowanych polega na wprowadzenie kolejnego poziomu uprzywilejowania (ring -1). Hypervisor działa własnie na tym poziomie, dzieki czemu system operacyjny może działać w ring 0.

Procesory z ring -1

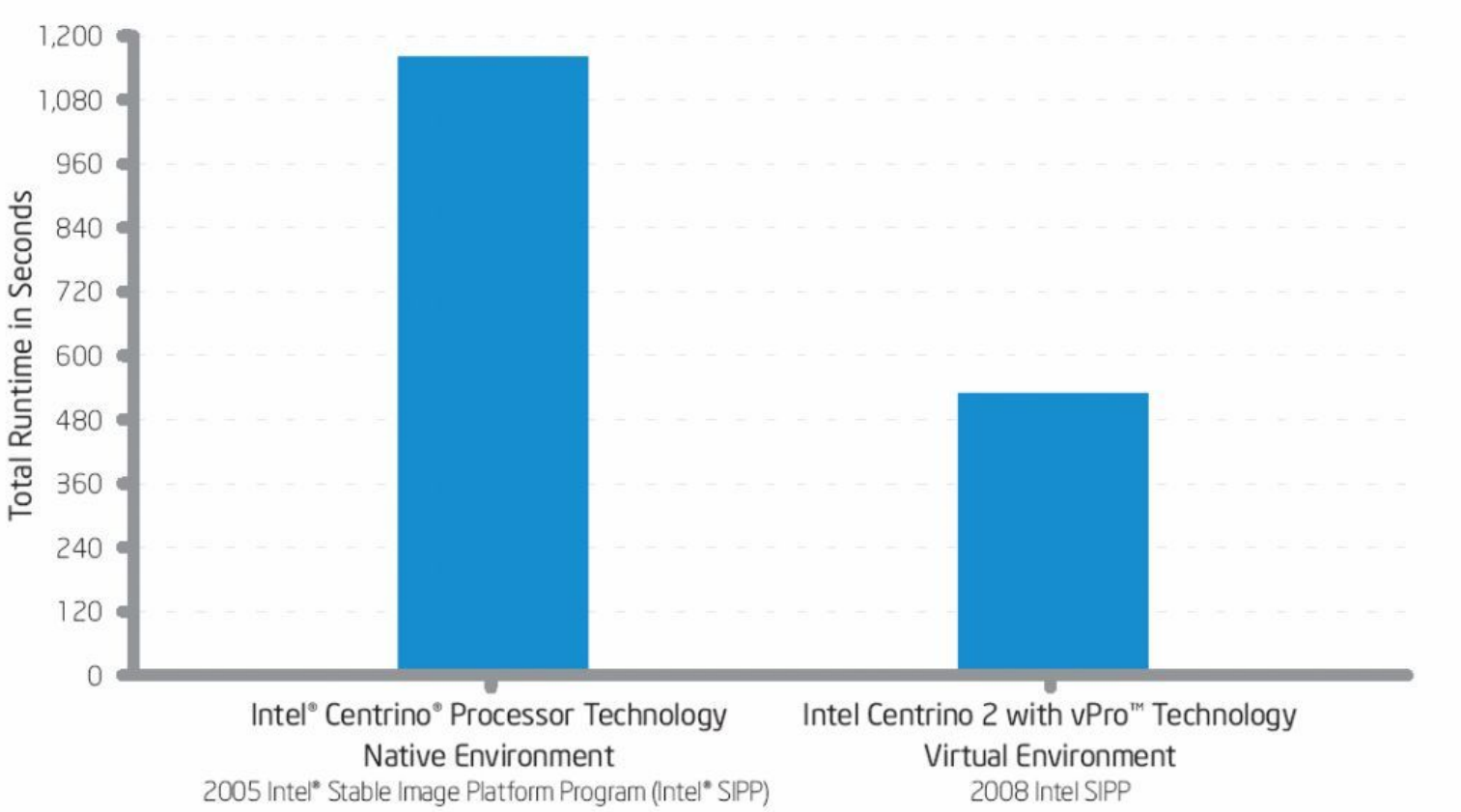
- AMD V
- Intel VT

Wirtualizacja wspomagana sprzętowo
Ring -1

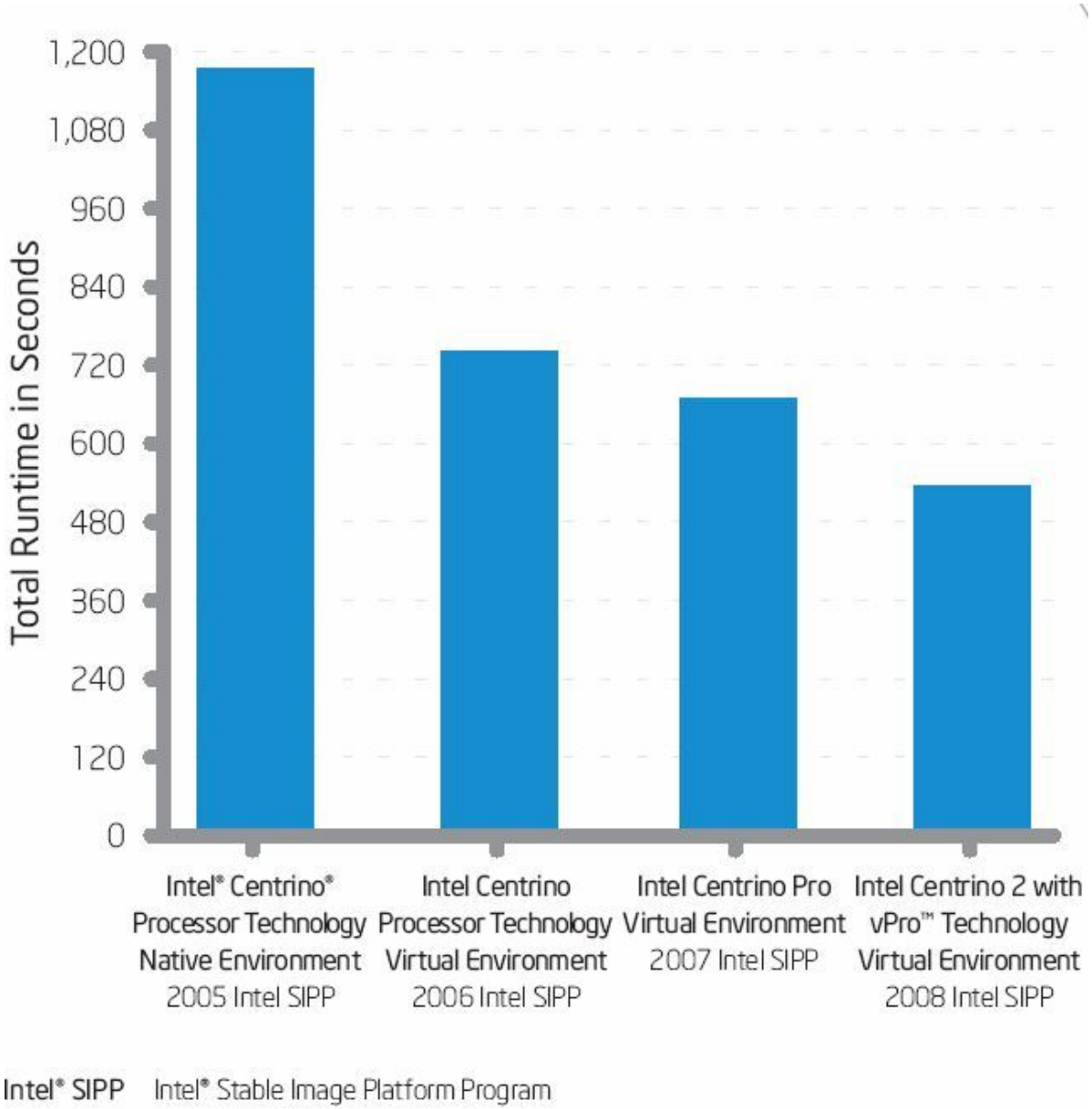


- Operacje główne i niegłówne (root i non-root) - tylko monitor maszyny wirtualnej wykonuje operacje główne, natomiast system operacyjny wykonuje operacje niegłówne
- Rozszerzenie listy rozkazów o 10 specyficznych dla wirtualizacji: VMPTRLD, VMPTRST, VMCLEAR, VMREAD, VMWRITE, VMCALL, VMLAUNCH, VMRESUME, VMXOFF(wyjście z trybu wirtualizacji) oraz VMXON(przejście w tryb wirtualizacji)
- Technologia IOMMU, zastępująca DMA - tłumaczenie adresów widocznych dla sytemu na adresy fizyczne

Wydajność
Intel VT

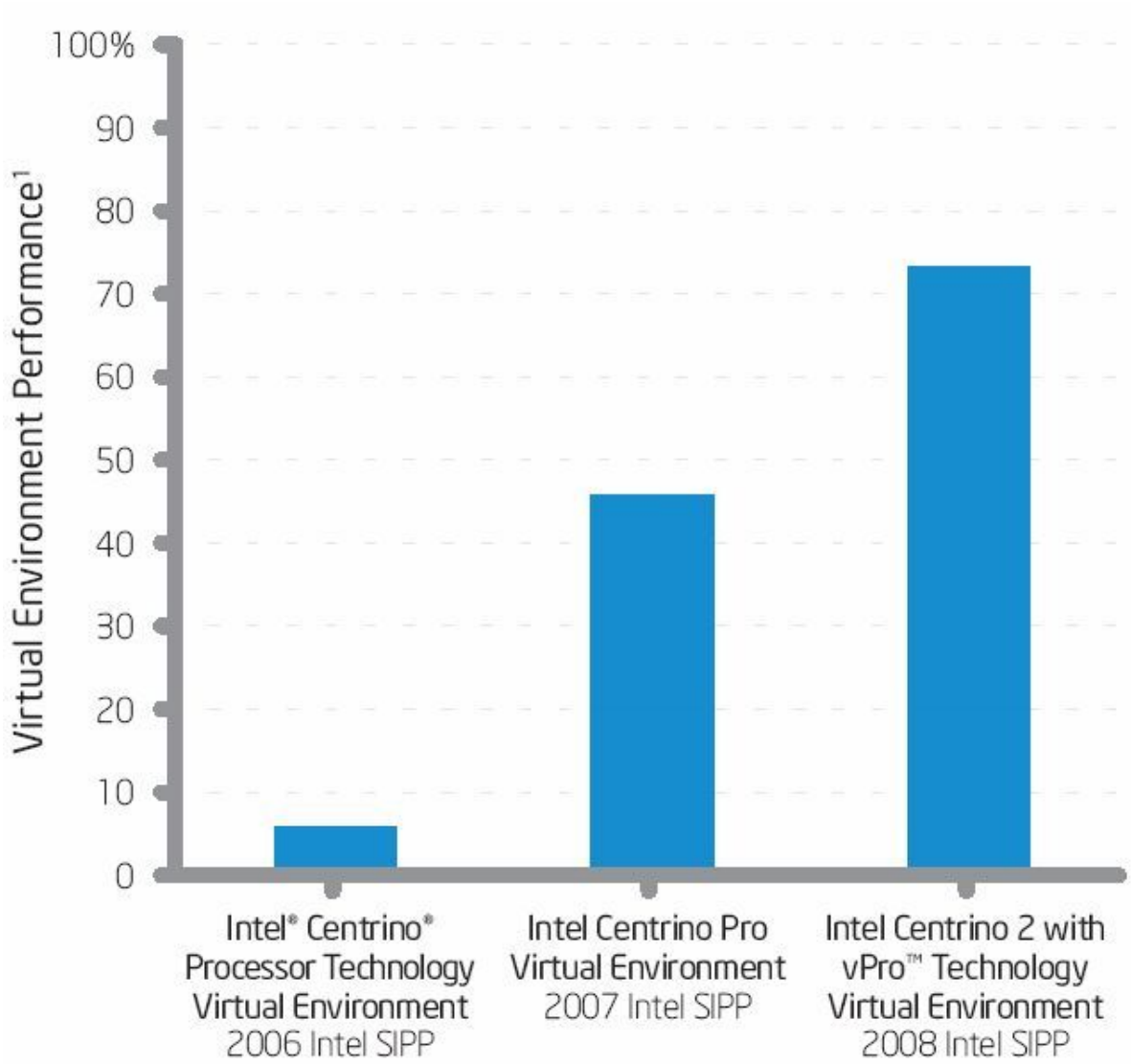


Wydajność
Intel VT



Intel® SIPP Intel® Stable Image Platform Program

Wydajność
Intel VT



Intel® SIPP Intel® Stable Image Platform Program

[1] Virtual environment performance improvement compared with the 2005 platform (baseline).

Zazwyczaj opłaca się wirtualizować urządzenia

- by lepiej wykorzystać zasoby sprzętowe
- by można było łatwo przenosić VM na inne maszyny

Do tego celu używa się:

- emulacji
- lub parawirtualizacji

Gospodarz implementuje i udaje prawdziwe urządzenie

- może całkowicie implementować urządzenie
- lub wykorzystywać inne do wykonywania zadań
- Gość nie jest świadomy, że nie rozmawia z urządzeniem

Zalety:

- nie wymaga modyfikacji Gości
- możemy udawać urządzenia, którego nie mamy
- możemy dzielić jedno fizyczne urządzenie między Gości
- łatwa migracja Wirtualnych Maszyn

Wady:

- bardzo niska wydajność

Bardzo niska wydajność spowodowana:

- udawaniem, że urządzenie istnieje
- przechwytywaniem przerwań (emulacja PIC)
- przechwytywaniem wszystkich odwołań urządzenia do pamięci
- tłumaczeniem wirtualnych rozkazów na prawdziwe akcje

Parawirtualizacja

- całkiem dobra wydajność
- ale kosztem modyfikacji Gości

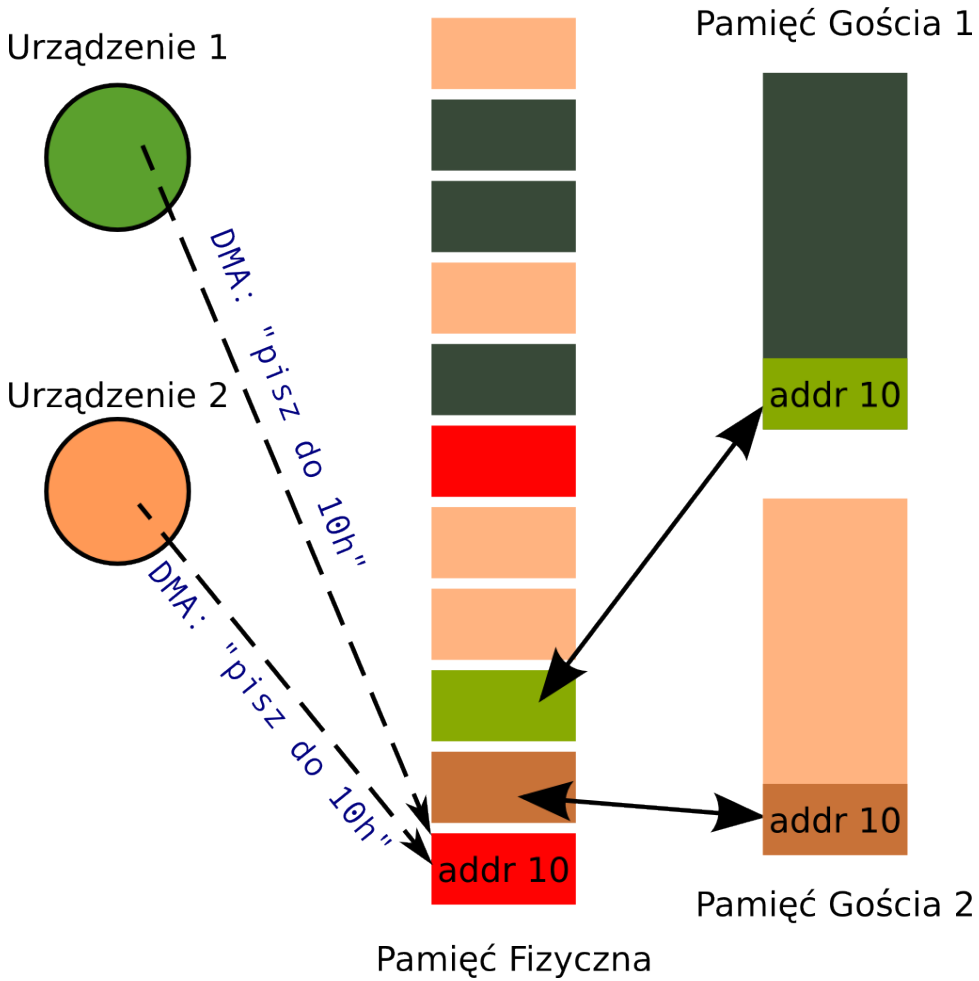
Więcej przy omawianiu technologii Xen.

Bezpośredni dostęp do urządzenia

- zdarzają się systemy wymagającej wysokiej wydajności
- głównie chodzi o dyski twarde i kontrolery sieciowe
- narzuty emulacji i parawirtualizacji są nieakceptowalne
- przy dużym transferze głownie obciążany jest procesor

Konieczna jest wtedy możliwość bezpośredniego dostępu do urządzenia.

Bezpośredni dostęp do urządzenia
główny problem - konflikty DMA



Bezpośredni dostęp do urządzenia

Konieczne jest sprzętowe wsparcie

- nie jest możliwe albo nie jest wydajne *softwarowe*
- tłumaczenie adresów, do których odwołuje się urządzenie
- trzeba też obsługiwać i przekazywać przerwania

Na szczęście z pomocą przychodzą nam twórcy sprzętu.

Input/Output Memory Management Unit

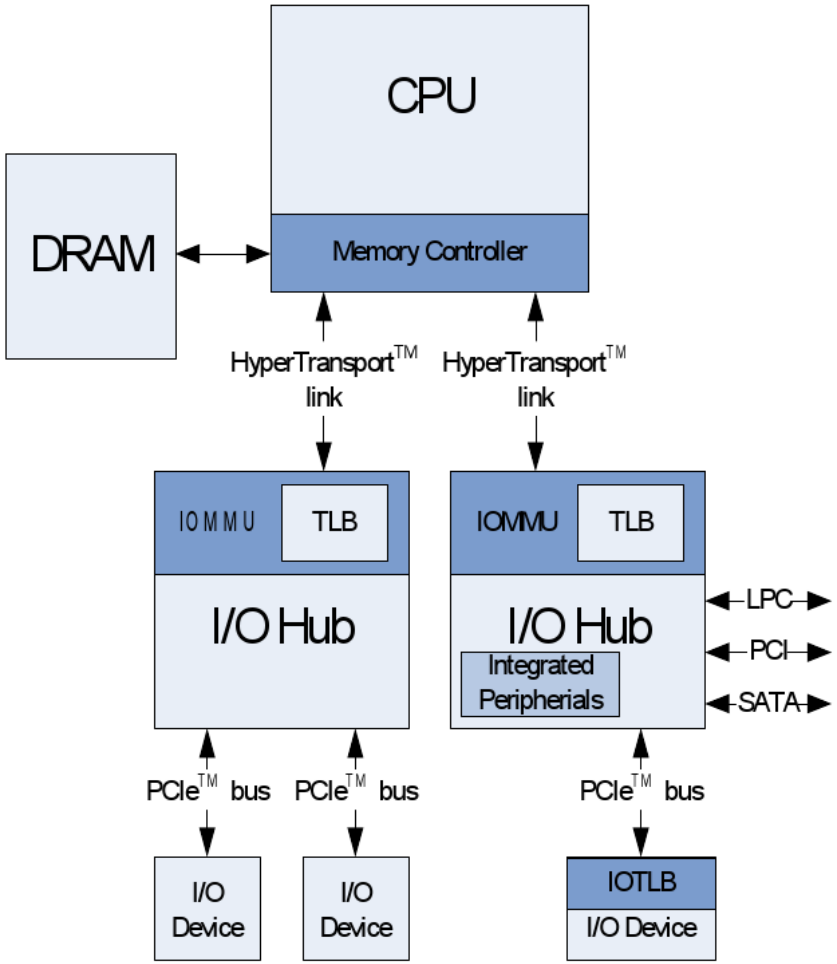
- urządzenie (część płyty głównej)
- tłumaczące adresy w transakcjach DMA
- i przerwania urządzeń Wej/Wyj
- zapewniające tym samym ochronę pamięci

IOMMU umożliwia:

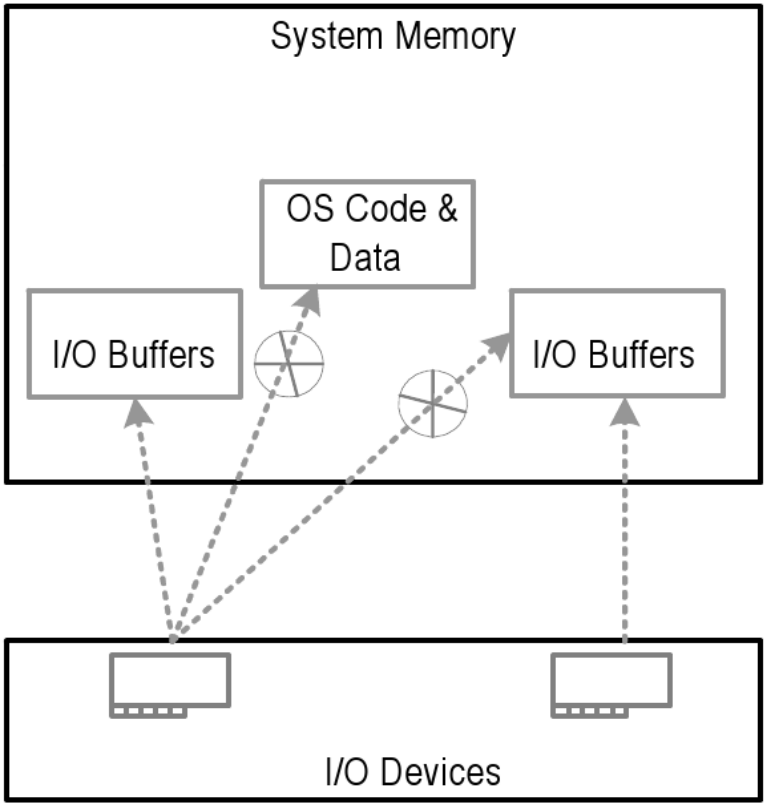
- przydzielenie każdemu urządzeniu *Protection Domain*,
- zestawu stron pamięci do którego urządzenie ma dostęp
- oraz tablic translacji

Gdy urządzenie zażąda dostępu do pamięci, IOMMU przechwytuje zapytanie. Tłumaczy je i pozwala lub nie na dostęp do fizycznej pamięci.

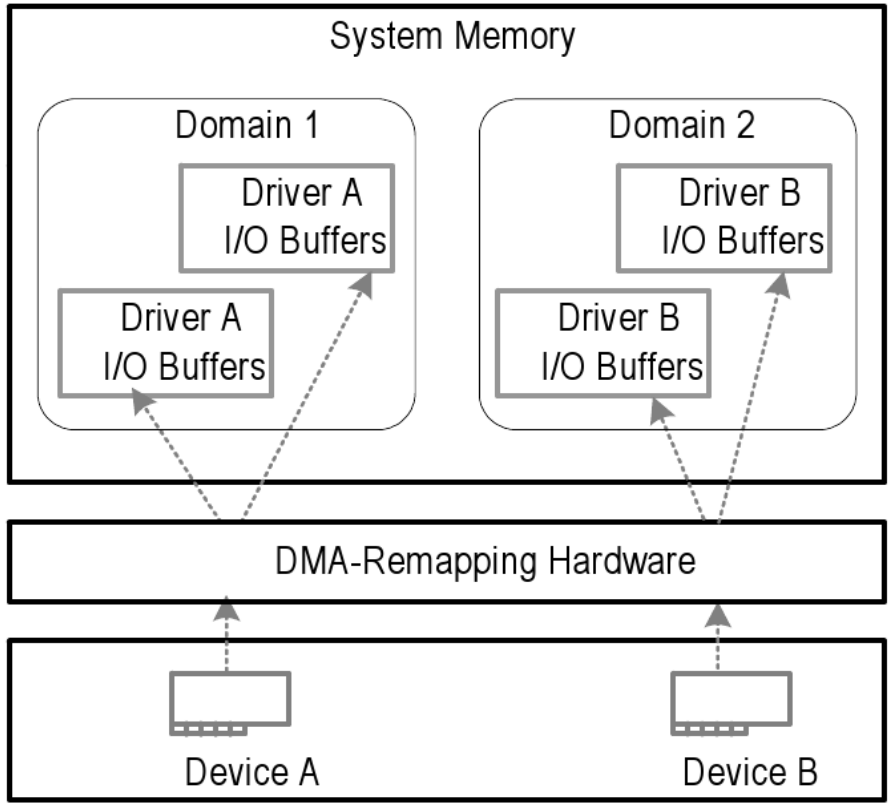
IOMMU
schemat



IOMMU
ochrona pamięci/Protection Domains

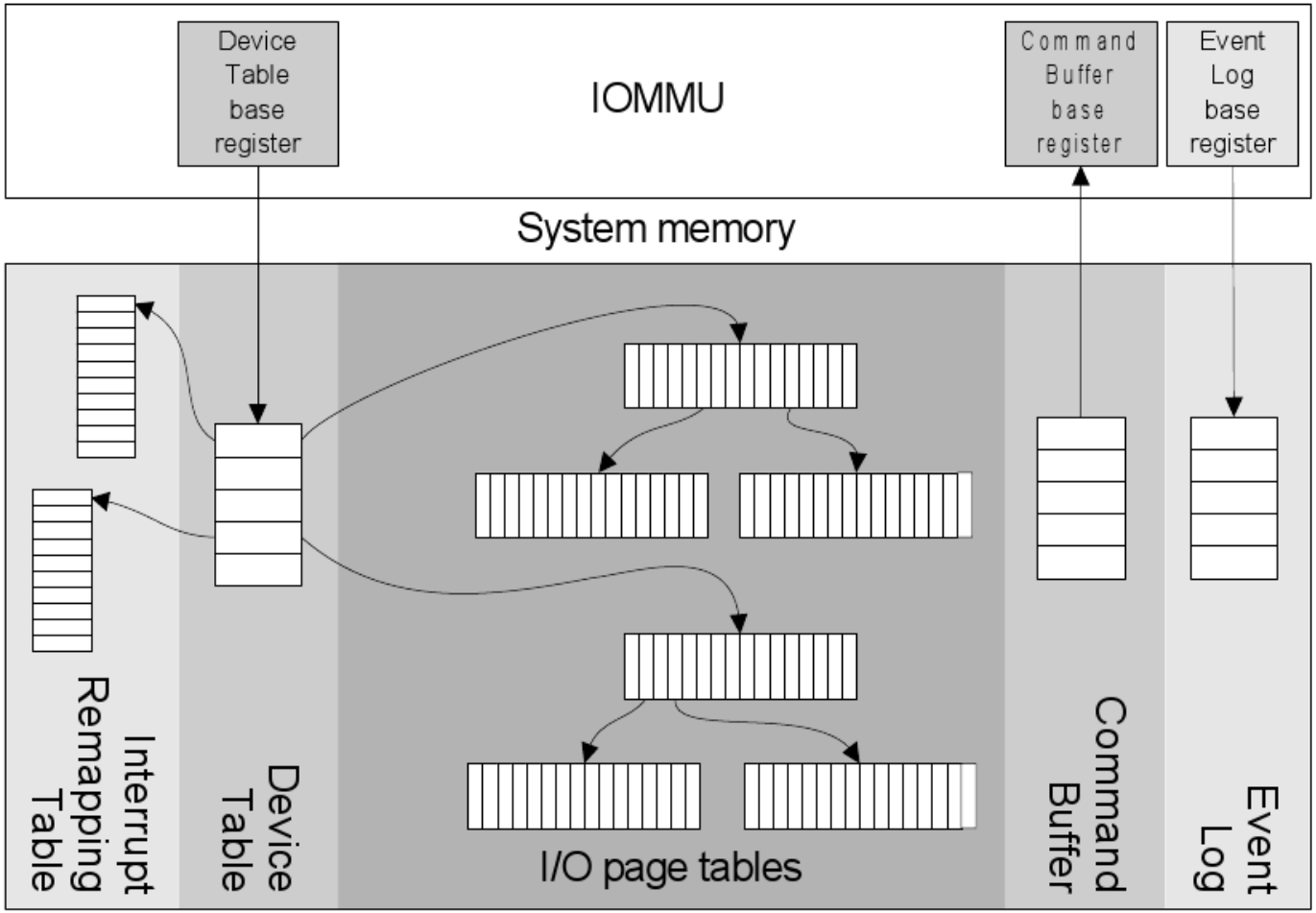


Device DMA without isolation



Device DMA isolated using DMA remapping hardware

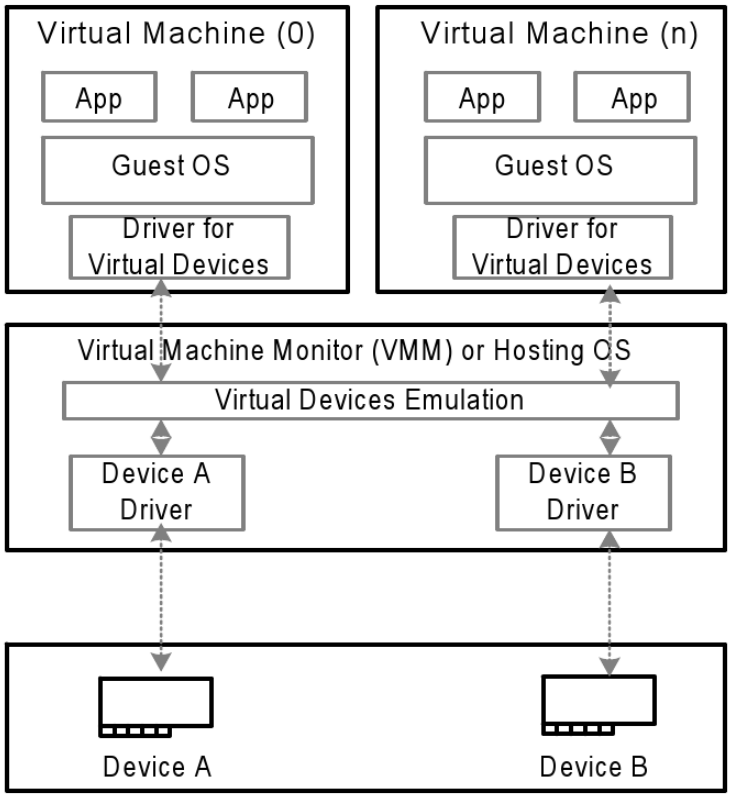
IOMMU
schemat struktur danych



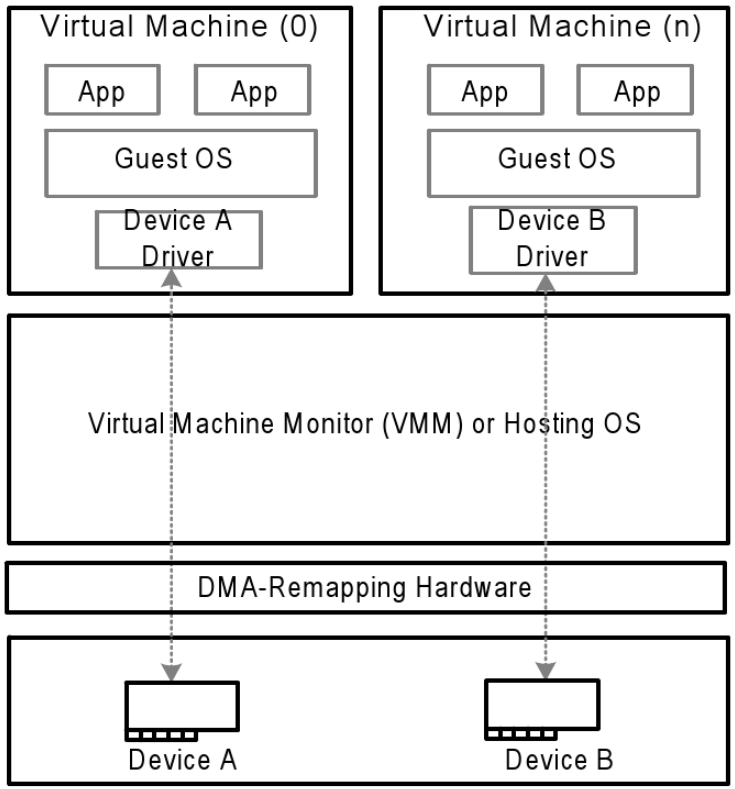
- urządzenia mogą posiadać własne bufory IO TLB
- tłumaczone mogą być też przerwania
- dzięki *Domenom Ochrony* współdzielone są tablice stron i cache
- IOMMU posiada dziennik zdarzeń/błędów

IOMMU

- pozwala na bezpośredni dostęp do urządzeń niezmodyfikowanym Gościom
  - cała pamięć wirtualna gościa w jednej Domenie Ochrony
- trochę większe możliwości przy parawirtualizacji
- wyeliminowanie narzutów VMM przy dostępie do urządzeń
- praktycznie 100% wydajność



Example Software-based I/O Virtualization



Direct Assignment of I/O Devices

- obsługa urządzeń bez obsługi 64-bitowego adresowania
  - brak konieczności używania *bounce buffers*
- zastąpienie pierwowzorów AMD IOMMU: GART i DEV
- bezpośredni dostęp do urządzeń z przestrzeni użytkownika
  - strony procesu w jednej Domenie Ochrony
  - pozostaje obsługa przerwań po stronie jądra

Intel VT-d

Intel Virtualization Technology for Directed I/O

- wspomaganie bezpośredniego dostępu do urządzeń
- czyli specyfikacja IOMMU od Intela
- funkcjonalność jak IOMMU od AMD
- ale urządzenie może należeć do wielu Dziedzin Ochrony
- **VT-d2** - ulepszone mapowanie przerwań

Intel Virtualization Technology for Connectivity

- kolejna technologia Intela wspierająca wirtualizację
- której zadaniem jest ogólna poprawa wydajności
- dzięki ulepszeniu komunikacji między urządzeniami I/O a CPU
- zapewnia zmniejszenie użycia CPU i zwiększenie przepustowości

W skład Intel VT-c wchodzi:

- Virtual Machine Device Queues (VMDq)
- Intel I/O Acceleration Technology
- Intel Single Root I/O Virtualization (SR-IOV)

Xen

- Monitor Maszyn Wirtualnych (VMM)
- wydany na licencji Open Source (GPL2)
- działający na IA-32/64, x86(-64) i PowerPC 970
- na początku używał tylko parawirtualizacji
- teraz potrafi też uruchamiać niezmodyfikowane systemy
- ale tylko na maszynach ze wspomaganiem sprzętowym

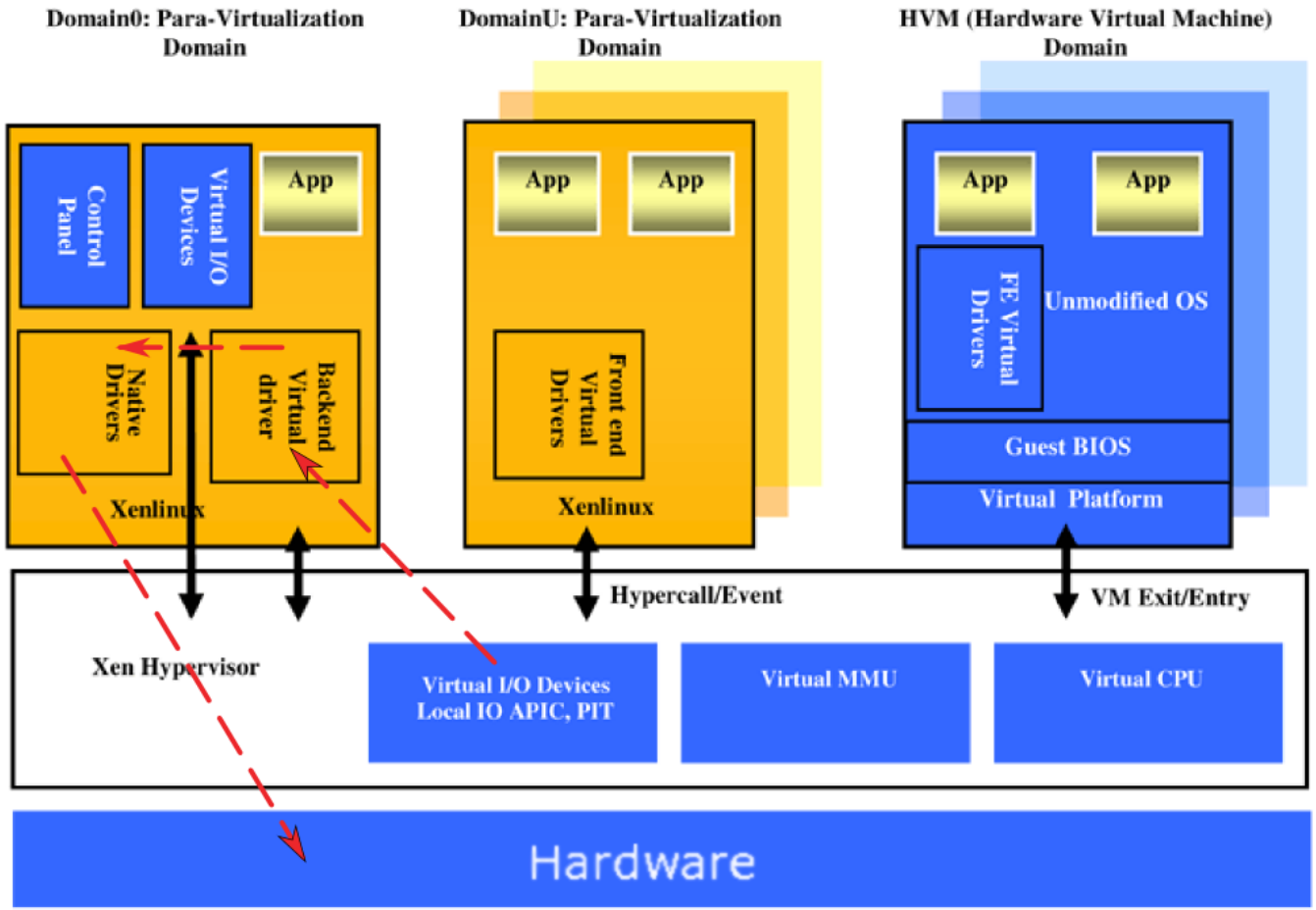
Parawirtualizacja

- polega na zmodyfikowaniu systemu operacyjnego Gościa
- tak by ten był świadom istnienia VMM
- i współpracował z nim za pomocą specjalnego API
- omijając w ten sposób wiele problemów
- oraz zyskując **znaczną** poprawę wydajności

Działanie:

- Xen posiada własne jądro
- posiada własny CPU scheduler, MM
- uruchamia *dom0* na niższym poziomie uprzywilejowania
- która jest wykorzystywana do zarządzania resztą VM
- oraz jako jedyna ma bezpośredni dostęp do urządzeń

Xen
zasada działania :: schemat



Kernel-based Virtual Machine

- infrastruktura dla wirtualizacji
- bazująca na jądrze Linuksa
- działająca na x86 z VT-x lub AMD-V
- ze szczątkowym wsparciem dla parawirtualizacji

KVM i Xen - różnice

- Xen ma własne jądro, z Linuksa korzysta tylko do obsługi I/O
- KVM używa Linuksowego schedulera i zarządzania pamięcią
- KVM koniecznie potrzebuje sprzętowego wsparcia wirtualizacji
- jest dzięki temu mały (ok. 12 000 linii kodu) i stosunkowo prosty
- KVM umożliwia bardzo prostą parawirtualizację (zmodyfikowane sterowniki)

- umożliwia pełną wirtualizację niezmodyfikowanych Gości
- pozwala przydzielić VM do 4 procesorów
- wspiera AMD IOMMU, Intel VT-d
- jest to znacznie mniejszy projekt od Xen
- jest także mniej dojrzały, lecz rozwój nabiera rozmachu

Porównanie wydajności

- *Oprogramowanie*:
  - Linux kernel 2.6.24-18
  - KVM 62
  - Xen 3.2.1+2.6.24-18-xen
- *Gość*: Ubuntu Linux 8.04
- *Sprzęt*: 2.4 GHz Intel Core 2 CPU 6600, 4 GB RAMu, 250 GB dysku

Porównanie wydajności

Test	**Linux**	**Xen**	**KVM**
CPU	1.000	0.999	0.993
Kompilacja jądra	1.000	0.487	0.384
IOzone Write	1.000	0.855	0.934
IOzone Read	1.000	0.852	0.994

- AMD IOMMU Specification - `http://www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/34434.pdf`
- Intel VT-d Specification - `ftp://download.intel.com/technology/computing/vptech/Intel(r)_VT_for_Direct_IO.pdf`

Zagrożenia związane z wirtualizacją

Skąd się biorą zagrożenia

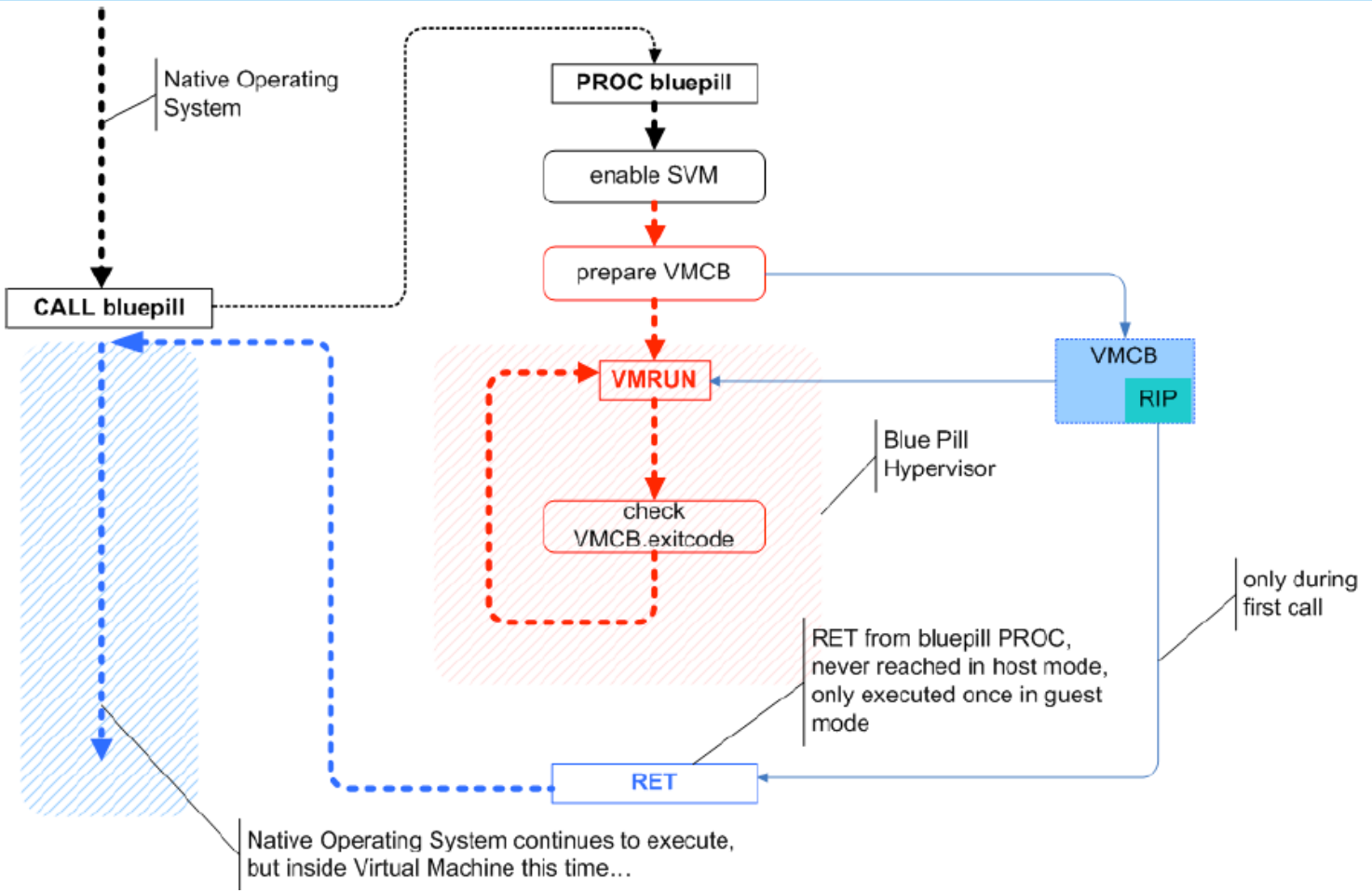
- w założeniu wirtualizacja ma być niewidoczna dla zwirtuakizowanego systemu
- jeżeli napisalibyśmy wirusa, który wirtualizuje (jest nadzorcą, *hypervisor*) prawdziwego systemu operacyjnego to zgodnie z powyższym powinien być niewykrywalny z wnętrza tego systemu
- jeżeli uda nam się zrobić to w sposób niewidoczny dla użytkownika, to wygraliśmy

Zagrożenia związane z wirtualizacją
Przykład-Bluepill

Ogólne informacje:

- najbardziej znany wirus wykorzystujący wirtualizację (AMD SVM)
- przeznaczony dla Windows Vista
- *przy okazji* obchodzi system zabezpieczenia pamięci przed nieuprawnionym zapisem zapisując swój kod do pliku swap w miejscu chwilowo nieużywanego sterownika
- po przejęciu kontroli poddaje wirtualizacji działający system **bez** ponownego uruchamiania komputera
- a użytkownik nawet nie widzi zmiany

Zagrożenia związane z wirtualizacją
Jak zwirtualizować działający system operacyjny



Zagrożenia związane z wirtualizacją I
Obrona

Wykrywanie:

- *teoretycznie* nie powinno się dać stwierdzić z wnętrza zwirtualizowanego systemu czy jest on poddany wirtualizacji, czy nie
- można umożliwić wirtualizację *wewnątrz* zwirtualizowanego systemu
- ale wszystkie operacje które przechwytujemy trwają dłużej, niż gdybyśmy ich nie przechwytywali
- ale możemy oszukiwać system który kontrolujemy co do upływu czasu
- ...

Zapobieganie:

- skuteczna ochronan systemy operacyjnego przed **zainstalowaniem** wirusa
- wyłączenie wirualizacji (ale powoduje utratę funkcjonalności)
- proponowane rozwiązania: ochrona hasłem instrukcji VMRUN

Shadow page tables
Dlaczego musimy wiedzieć, co gość robi ze stronami

- gość ma własne tablice stron, ale nie są one używane przez procesor
- nadzorca musi zapewniać spójność między stronami we właściwej tablicy stron a tymi z tablicy stron gościa
- zatem nadzorca musi kontrolować zapisy gościa do jego tablicy stron

Shadow page tables
Jak kontrolować gościa

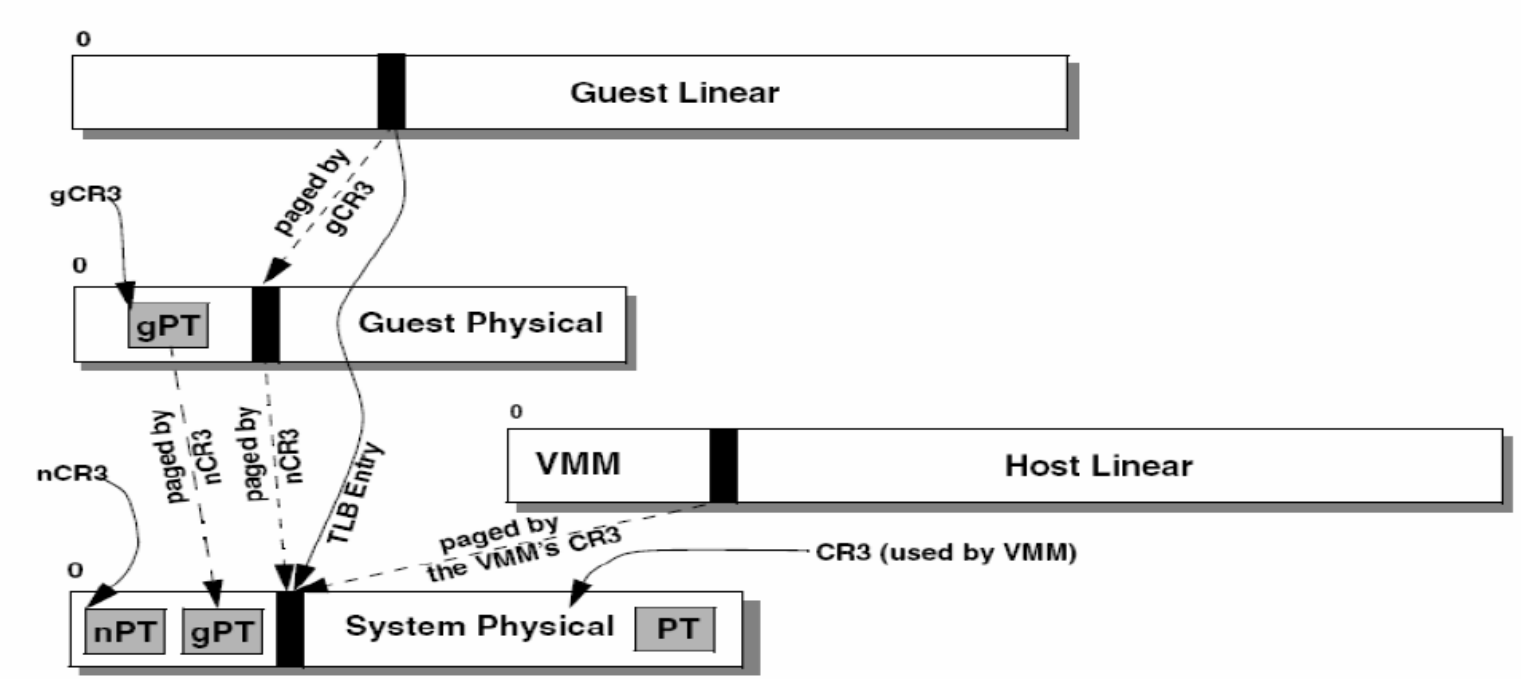
- zabezpieczamy *jego* tablicę stron przed zapisem
- każda próba zapisu powoduje page fault, który jest przechwytywany przez nadzorcę
- nadzorca aktualizuje od razu obie tablice stron
- co powoduje spory narzut na działanie programów które używają dużo pamięci

Nested page tables
Motywacja za zagdnieżdżonymi tablicami stron

- w zwykłym rozwiązaniu błędy braku strony są obsługiwane przez nadzorcę, który musi przekazać infromację właściwemu zwirtualizowanemu systemowi
- podowuje to duży narzut na obsługę takiego błędu (dużo VMWEXIT)
- Nested/Extended Page Tables pozwalają na bezpośrednie mapowanie stron widocznych dla systemu-gościa jako fizyczne na fizyczne strony pamięci bez pomocy nadzorcy
- błędy braku stron są przekazywane systeomwi-gościowi bezpośrednio przez procesor

Nested page tables
Jak działają Nested Page Tables

- nadzorca buduje tablicę stron a potem ustawia odpowiedni bit rejestru
- translacja wygląda następująco:



Nested page tables
Nested vs Shadow