Wirtualizacja wspomagana sprzętowo

zalety, wady i zagrożenia

Bartosz Gęza Tomasz Rogozik Bartosz Szreder

19 listopada 2009

1. Wirtualizacja bez wspomagania sprzętowego
   - Wprowadzenie do wirtualizacji
   - Zasada działania pełnej wirtualizacji
   - Problemy i sposoby radzenia sobie z nimi
   - Przyczyny problemów z wydajnością
2. Pierwsze podejście do wirtualizacji wspomaganej sprzętowo
   - Ring 1
   - AMD-V
   - Intel-VT
3. Wirtualizacja MMU
   - Shadow page tables
   - Nested page tables
4. Dostęp do urządzeń z maszyny wirtualnej. IOMMU
   - Dostęp do urządzeń z maszyny wirtualnej
   - IOMMU
5. Wirtualizacja sprzętowa w XEN i KVM
   - Co daje sprzęt?
   - Sprzętowa wirtualizacja w Xen i KVM
6. Problemy związane z wirtualizacją
   - Ogólne problemy wirtualizacji
   - Matrix jak żywy

Co to jest wirtualizacja?

Wirtualizacja

to tworzenie abstrakcji zasobów w różnych aspektach komputeryzacji.

W kontekście systemów operacyjnych, jest to umożliwienie:

Co to jest wirtualizacja?

Wirtualizacja

to tworzenie abstrakcji zasobów w różnych aspektach komputeryzacji.

W kontekście systemów operacyjnych, jest to umożliwienie:

- pracy na wielu systemach operacyjnych jednocześnie

Co to jest wirtualizacja?

Wirtualizacja

to tworzenie abstrakcji zasobów w różnych aspektach komputeryzacji.

W kontekście systemów operacyjnych, jest to umożliwienie:

- pracy na wielu systemach operacyjnych jednocześnie
- efektywnego wykorzystania sprzętu

Co to jest wirtualizacja?

Wirtualizacja

to tworzenie abstrakcji zasobów w różnych aspektach komputeryzacji.

W kontekście systemów operacyjnych, jest to umożliwienie:

- pracy na wielu systemach operacyjnych jednocześnie
- efektywnego wykorzystania sprzętu
- jednoczesnego używania aplikacji stworzonych dla różnych systemów operacyjnych

Rodzaje wirtualizacji

- Emulacja
- Emulacja API
- Pełna wirtualizacja

Rodzaje wirtualizacji - emulacja

Emulacja

pozwala na uruchamianie aplikacji pochodzących z niekompatybilnego komputera (w stosunku do wykorzystywanego).

Rodzaje wirtualizacji - emulacja

Emulacja

pozwala na uruchamianie aplikacji pochodzących z niekompatybilnego komputera (w stosunku do wykorzystywanego).

- emulacja podstawowych podzespołów komputera (CPU, RAM, HDD, CD itp.)

Rodzaje wirtualizacji - emulacja

Emulacja

pozwala na uruchamianie aplikacji pochodzących z niekompatybilnego komputera (w stosunku do wykorzystywanego).

- emulacja podstawowych podzespołów komputera (CPU, RAM, HDD, CD itp.)
- emulacja systemu operacyjnego

Rodzaje wirtualizacji - emulacja

Emulacja

pozwala na uruchamianie aplikacji pochodzących z niekompatybilnego komputera (w stosunku do wykorzystywanego).

- emulacja podstawowych podzespołów komputera (CPU, RAM, HDD, CD itp.)
- emulacja systemu operacyjnego
- prawie każda operacja na wirtualnym systemie jest emulowana -> spadek wydajności

Rodzaje wirtualizacji - emulacja

Emulacja

pozwala na uruchamianie aplikacji pochodzących z niekompatybilnego komputera (w stosunku do wykorzystywanego).

- emulacja podstawowych podzespołów komputera (CPU, RAM, HDD, CD itp.)
- emulacja systemu operacyjnego
- prawie każda operacja na wirtualnym systemie jest emulowana -> spadek wydajności

Emulator pełny

- wirtualny odpowiednik całego komputera

Rodzaje wirtualizacji - emulacja

Emulacja

pozwala na uruchamianie aplikacji pochodzących z niekompatybilnego komputera (w stosunku do wykorzystywanego).

- emulacja podstawowych podzespołów komputera (CPU, RAM, HDD, CD itp.)
- emulacja systemu operacyjnego
- prawie każda operacja na wirtualnym systemie jest emulowana -> spadek wydajności

Emulator pełny

- wirtualny odpowiednik całego komputera
- wykonuje w pętli to, co robiłby rzeczywisty procesor maszyny emulowanej

Rodzaje wirtualizacji - emulacja API

Emulacja API

wykorzystuje sposób działania aplikacji jako rozdzielnych procesów w stosunku do systemu operacyjnego.

Rodzaje wirtualizacji - emulacja API

Emulacja API

wykorzystuje sposób działania aplikacji jako rozdzielnych procesów w stosunku do systemu operacyjnego.

- wykorzystanie interfejsu API do komunikacji z systemem operacyjnym

Rodzaje wirtualizacji - emulacja API

Emulacja API

wykorzystuje sposób działania aplikacji jako rozdzielnych procesów w stosunku do systemu operacyjnego.

- wykorzystanie interfejsu API do komunikacji z systemem operacyjnym
- wprowadzenie do SO otoczenia API pochodzącego z innego systemu

Rodzaje wirtualizacji - emulacja API

Emulacja API

wykorzystuje sposób działania aplikacji jako rozdzielnych procesów w stosunku do systemu operacyjnego.

- wykorzystanie interfejsu API do komunikacji z systemem operacyjnym
- wprowadzenie do SO otoczenia API pochodzącego z innego systemu
- eliminuje konieczność posiadania całego systemu operacyjnego, pod którym działa aplikacja

Rodzaje wirtualizacji - pełna wirtualizacja

Pełna wirtualizacja

- pozwala jednocześnie uruchomić wiele systemów operacyjnych na tej samej platformie sprzętowej i systemowej przy maksymalnej możliwej wydajności

Rodzaje wirtualizacji - pełna wirtualizacja

Pełna wirtualizacja

- pozwala jednocześnie uruchomić wiele systemów operacyjnych na tej samej platformie sprzętowej i systemowej przy maksymalnej możliwej wydajności
- jest połączeniem podstawowych zalet emulacji pełnej oraz emulacji API

Rodzaje wirtualizacji - pełna wirtualizacja

Pełna wirtualizacja

- pozwala jednocześnie uruchomić wiele systemów operacyjnych na tej samej platformie sprzętowej i systemowej przy maksymalnej możliwej wydajności
- jest połączeniem podstawowych zalet emulacji pełnej oraz emulacji API
- uruchamia w maszynie wirtualnej system operacyjny

Rodzaje wirtualizacji - pełna wirtualizacja

Pełna wirtualizacja

- pozwala jednocześnie uruchomić wiele systemów operacyjnych na tej samej platformie sprzętowej i systemowej przy maksymalnej możliwej wydajności
- jest połączeniem podstawowych zalet emulacji pełnej oraz emulacji API
- uruchamia w maszynie wirtualnej system operacyjny

- rezygnuje z uniwersalności emulowania wielu architektur komputerów

Rodzaje wirtualizacji - pełna wirtualizacja

Pełna wirtualizacja

- pozwala jednocześnie uruchomić wiele systemów operacyjnych na tej samej platformie sprzętowej i systemowej przy maksymalnej możliwej wydajności
- jest połączeniem podstawowych zalet emulacji pełnej oraz emulacji API
- uruchamia w maszynie wirtualnej system operacyjny

- rezygnuje z uniwersalności emulowania wielu architektur komputerów
- teoretycznie wirtualizator powinien wykonać bez emulacji wszystkie operacje w trybie nieuprzywilejowanym

Rodzaje wirtualizacji - pełna wirtualizacja

Pełna wirtualizacja

- pozwala jednocześnie uruchomić wiele systemów operacyjnych na tej samej platformie sprzętowej i systemowej przy maksymalnej możliwej wydajności
- jest połączeniem podstawowych zalet emulacji pełnej oraz emulacji API
- uruchamia w maszynie wirtualnej system operacyjny

- rezygnuje z uniwersalności emulowania wielu architektur komputerów
- teoretycznie wirtualizator powinien wykonać bez emulacji wszystkie operacje w trybie nieuprzywilejowanym
- wymaga to jednak specjalnej konstrukcji architektury (niestety nie spełnia jej x86)

Jak działa pełna wirtualizacja?

VMM – Virtual Machine Monitor (Hypervisor)

- arbiter dostępu do wszelkich zasobów takich jak I/O, CPU, RAM

Jak działa pełna wirtualizacja?

VMM – Virtual Machine Monitor (Hypervisor)

- arbiter dostępu do wszelkich zasobów takich jak I/O, CPU, RAM
- umożliwia wykonywanie pewnej ilości procesów systemu operacyjnego gościa bezpośrednio na zasobach sprzętowych komputera

Jak działa pełna wirtualizacja?

VMM – Virtual Machine Monitor (Hypervisor)

- arbiter dostępu do wszelkich zasobów takich jak I/O, CPU, RAM
- umożliwia wykonywanie pewnej ilości procesów systemu operacyjnego gościa bezpośrednio na zasobach sprzętowych komputera
- gdy operacje nie dadzą się bezpośrednio wykonać, wirtualizator emuluje je

Jak działa pełna wirtualizacja?

VMM – Virtual Machine Monitor (Hypervisor)

- arbiter dostępu do wszelkich zasobów takich jak I/O, CPU, RAM
- umożliwia wykonywanie pewnej ilości procesów systemu operacyjnego gościa bezpośrednio na zasobach sprzętowych komputera
- gdy operacje nie dadzą się bezpośrednio wykonać, wirtualizator emuluje je

Tryby maszyny wirtualnej

Type 1 - native na czystym sprzęcie, VMM przypomina SO (blisko sprzętu)

Jak działa pełna wirtualizacja?

VMM – Virtual Machine Monitor (Hypervisor)

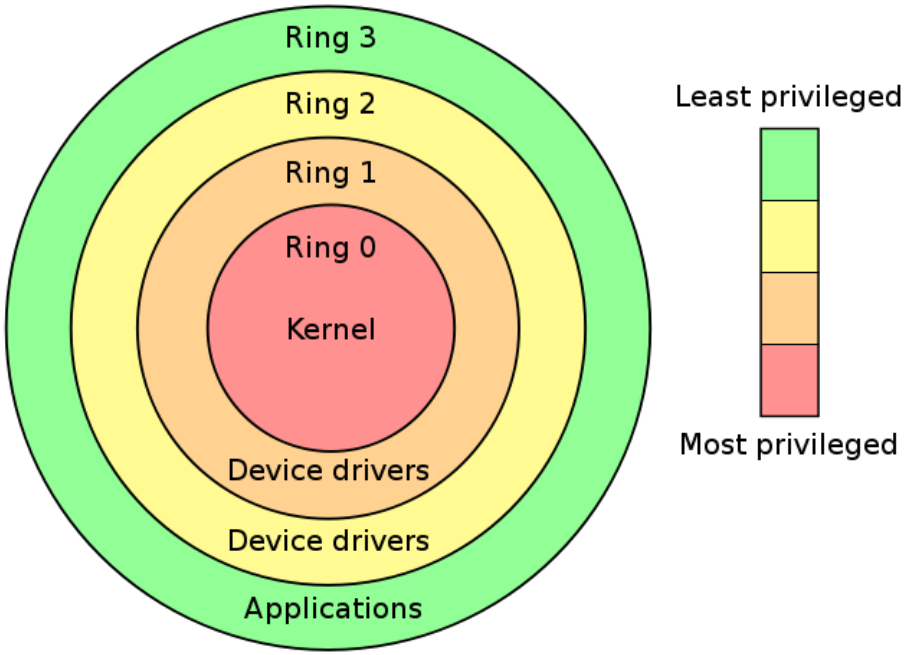
- arbiter dostępu do wszelkich zasobów takich jak I/O, CPU, RAM
- umożliwia wykonywanie pewnej ilości procesów systemu operacyjnego gościa bezpośrednio na zasobach sprzętowych komputera
- gdy operacje nie dadzą się bezpośrednio wykonać, wirtualizator emuluje je

Tryby maszyny wirtualnej

Type 1 - native na czystym sprzęcie, VMM przypomina SO (blisko sprzętu)

Type 2 - hosted jako aplikacja systemu operacyjnego, VMM prostszy, emulacja kosztowniejsza

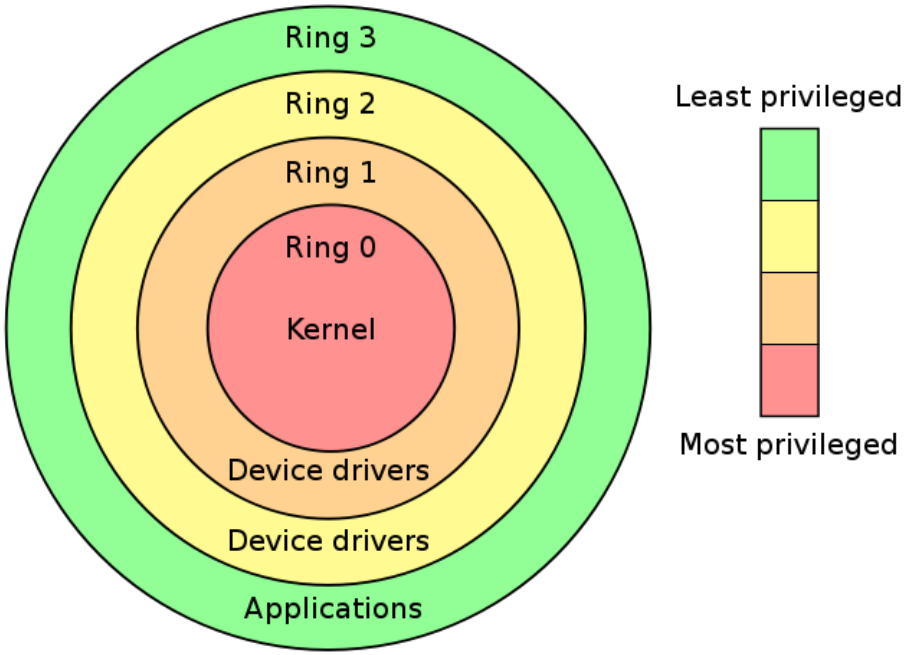
Poziomy uprawnień (protection rings)



Pierścienie ochrony

- dwa lub więcej poziomów uprawnień w architekturze komputera

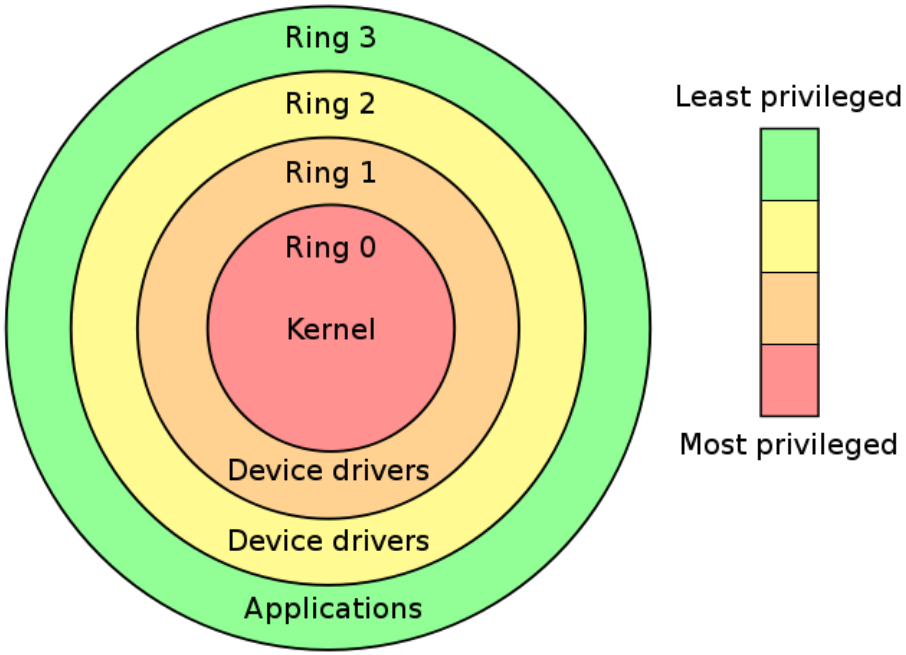
Poziomy uprawnień (protection rings)



Pierścienie ochrony

- dwa lub więcej poziomów uprawnień w architekturze komputera
- zazwyczaj są wymuszane przez niektóre architektury by zapewnić różne tryby CPU na poziomie firmware'u.

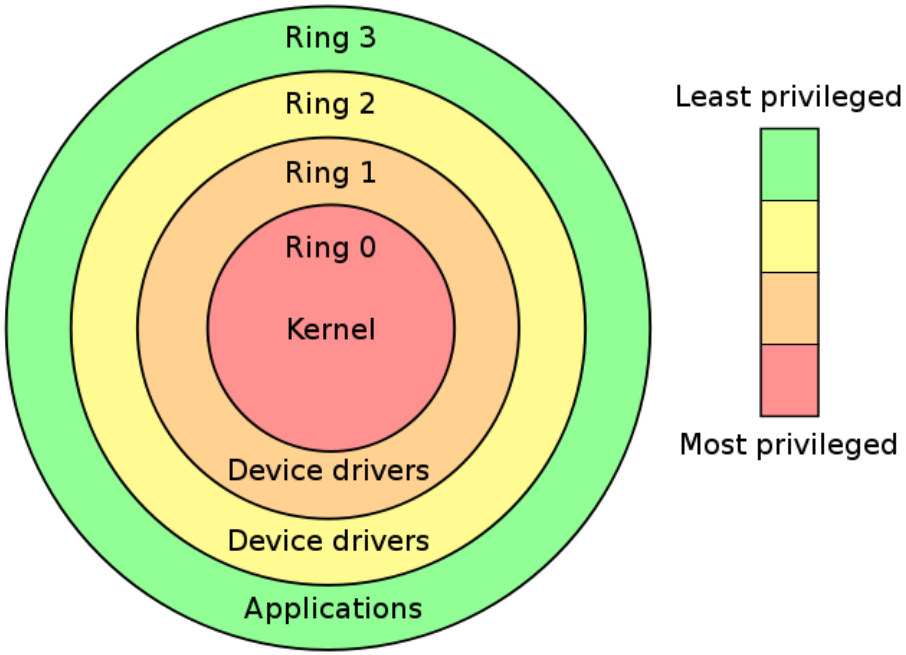
Poziomy uprawnień (protection rings)



Pierścienie ochrony

- dwa lub więcej poziomów uprawnień w architekturze komputera
- zazwyczaj są wymuszane przez niektóre architektury by zapewnić różne tryby CPU na poziomie firmware'u.
- uporządkowane są od najbardziej zaufanych (0) do najmniej uprzywilejowanych (tutaj 3)

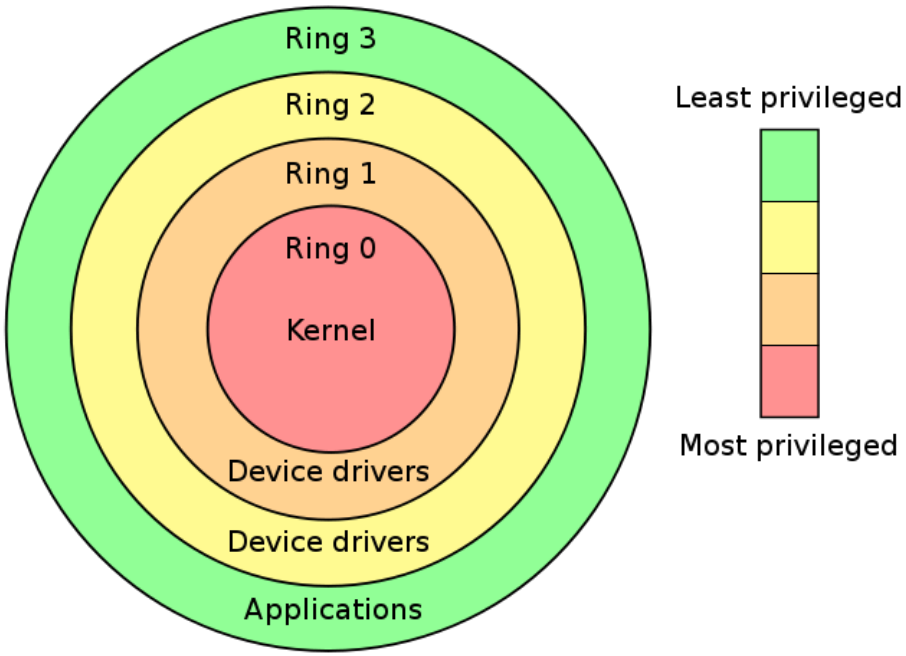
Poziomy uprawnień (protection rings)



Pierścienie ochrony

- dwa lub więcej poziomów uprawnień w architekturze komputera
- zazwyczaj są wymuszane przez niektóre architektury by zapewnić różne tryby CPU na poziomie firmware'u.
- uporządkowane są od najbardziej zaufanych (0) do najmniej uprzywilejowanych (tutaj 3)
- im niższy numer tym bliżej sprzętu

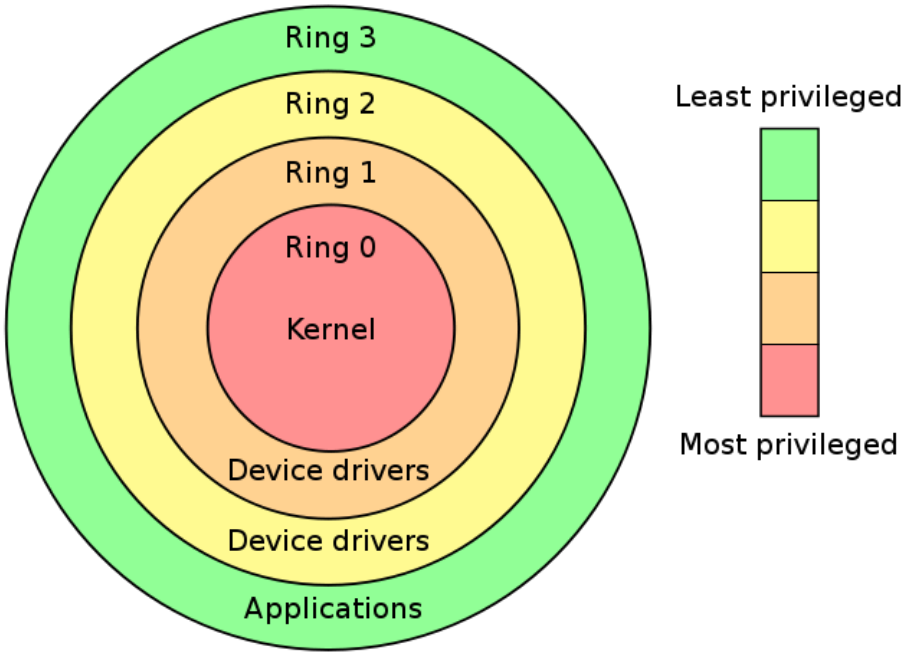
Poziomy uprawnień (protection rings)



Pierścienie ochrony

- dwa lub więcej poziomów uprawnień w architekturze komputera
- zazwyczaj są wymuszane przez niektóre architektury by zapewnić różne tryby CPU na poziomie firmware'u.
- uporządkowane są od najbardziej zaufanych (0) do najmniej uprzywilejowanych (tutaj 3)
- im niższy numer tym bliżej sprzętu
- bramy pomiędzy pierścieniami pozwalają zewnętrznemu pierścieniowi na korzystanie z zasobów pierścienia wewnętrznego

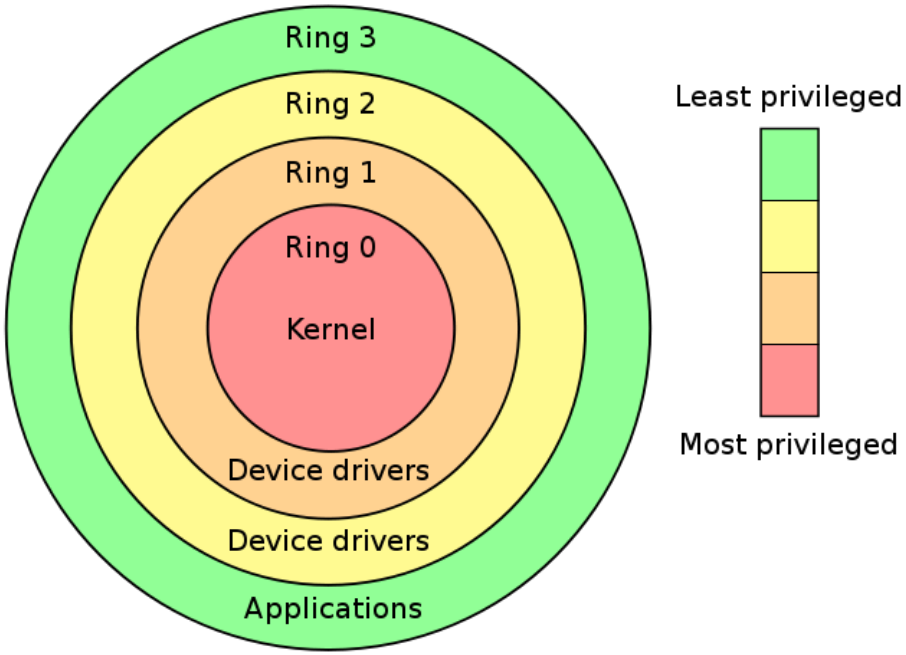
Poziomy uprawnień (protection rings)



Problem

- SO pracuje w Ring 0. Dwa systemy nie mogą pracować w Ring 0, jeden z nich musi działać jak aplikacja (Ring 3)

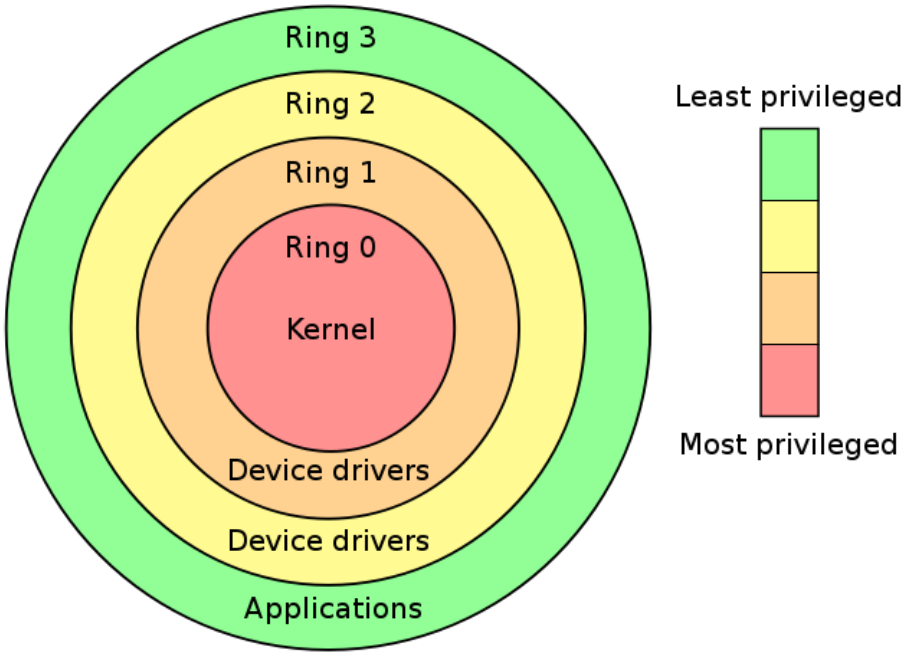
Poziomy uprawnień (protection rings)



Problem

- SO pracuje w Ring 0. Dwa systemy nie mogą pracować w Ring 0, jeden z nich musi działać jak aplikacja (Ring 3)
- SO żądając dostępu na wyższym poziomie powoduje wystąpienie wyjątku, który może być obsłużony przez VMM (emulacja instrukcji na odpowiednim poziomie)

Poziomy uprawnień (protection rings)



Problem

- SO pracuje w Ring 0. Dwa systemy nie mogą pracować w Ring 0, jeden z nich musi działać jak aplikacja (Ring 3)
- SO żądając dostępu na wyższym poziomie powoduje wystąpienie wyjątku, który może być obsłużony przez VMM (emulacja instrukcji na odpowiednim poziomie)
- Nie zawsze generowany jest wyjątek - odczyt z niektórych rejestrów nie jest uprzywilejowany podczas gdy zapis jest!

Podział instrukcji

Instrukcje dostępne na danej architekturze dzielimy na

uprzywilejowane

których efektem jest przerwanie lub wywołanie systemowe w trybie użytkownika lub ich brak w trybie jądra

Podział instrukcji

Instrukcje dostępne na danej architekturze dzielimy na

uprzywilejowane

których efektem jest przerwanie lub wywołanie systemowe w trybie użytkownika lub ich brak w trybie jądra

wrażliwe ze względu na kontrolę

w trakcie ich wykonywania może zostać zmieniona konfiguracja zasobów systemowych

Podział instrukcji

Instrukcje dostępne na danej architekturze dzielimy na

uprzywilejowane
: których efektem jest przerwanie lub wywołanie systemowe w trybie użytkownika lub ich brak w trybie jądra

wrażliwe ze względu na kontrolę
: w trakcie ich wykonywania może zostać zmieniona konfiguracja zasobów systemowych

wrażliwe ze względu na wykonanie
: ich działanie jest zależne od konfiguracji systemu

Podział instrukcji

Instrukcje dostępne na danej architekturze dzielimy na

uprzywilejowane
: których efektem jest przerwanie lub wywołanie systemowe w trybie użytkownika lub ich brak w trybie jądra

wrażliwe ze względu na kontrolę
: w trakcie ich wykonywania może zostać zmieniona konfiguracja zasobów systemowych

wrażliwe ze względu na wykonanie
: ich działanie jest zależne od konfiguracji systemu

Uwaga

Te wydzielone podzbiory instrukcji nie są ani rozłączne, ani nie sumują się do zbioru wszystkich instrukcji architektury.

Twierdzenie Popka-Goldberga

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

Twierdzenie Popka-Goldberga

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

Wnioski

- każda operacja mogąca powodować niepożądane działanie maszyny wirtualnej powoduje przerwanie lub wywołanie systemowe

Twierdzenie Popka-Goldberga

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

Wnioski

- każda operacja mogąca powodować niepożądane działanie maszyny wirtualnej powoduje przerwanie lub wywołanie systemowe
  - VMM może przechwycić takie przerwania lub wywołania

Twierdzenie Popka-Goldberga

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

Wnioski

- każda operacja mogąca powodować niepożądane działanie maszyny wirtualnej powoduje przerwanie lub wywołanie systemowe
  - VMM może przechwycić takie przerwania lub wywołania
- reszta instrukcji nieuprzywilejowanych jest przekazywana do fizycznego sprzętu

Twierdzenie Popka-Goldberga

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

Wnioski

- każda operacja mogąca powodować niepożądane działanie maszyny wirtualnej powoduje przerwanie lub wywołanie systemowe
  - VMM może przechwycić takie przerwania lub wywołania
- reszta instrukcji nieuprzywilejowanych jest przekazywana do fizycznego sprzętu

Można jednak zbudować maszynę wirtualizującą architekturę, która nie spełnia założeń twierdzenia Popka-Goldberga, godząc się na zmniejszoną wydajność lub inne ograniczenia.

Twierdzenie Popka-Goldberga

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

Wnioski

- każda operacja mogąca powodować niepożądane działanie maszyny wirtualnej powoduje przerwanie lub wywołanie systemowe
  - VMM może przechwycić takie przerwania lub wywołania
- reszta instrukcji nieuprzywilejowanych jest przekazywana do fizycznego sprzętu

W x86 instrukcje wrażliwe nie są podzbiorem instrukcji uprzywilejowanych

SGDT, SIDT, SLDT, SMSW, PUSHF, POPF, PUSHFD, POPFD, LAR, LSL, VERR, VERW, POP, PUSH, CALL, JMP, INT, RET, IRET, IRETD, STR, MOVE

Rozwiązania

Parawirtualizacja

- ingerencja w kod SO by stworzyć interfejs łatwiejszy do wirtualizacji
- wydajne, ale nie zawsze możliwe (licencja)
- przykład: XEN

Rozwiązania

Parawirtualizacja

- ingerencja w kod SO by stworzyć interfejs łatwiejszy do wirtualizacji
- wydajne, ale nie zawsze możliwe (licencja)
- przykład: XEN

Zmiana binariów (dynamiczna rekompilacja)

- wrażliwe operacje są przekształcane w czasie wykonania
- wiemy, w którym miejscu kodu rozpoczyna wykonanie, możemy znaleźć najbliższe miejsce rozgałęzienia lub instrukcji niebezbpiecznej i założyć na nim pułapkę (breakpoint)
- uzyskujemy wyjątki wykonania - VMM może je przechwycić
- przykład: VMWare, VirtualPC

Inne problemy

- goszczący SO chce mieć dostęp do całej pamięci mu przydzielonej - musi ona być dzielona z VMM w celu komunikacji
- gość (system wirtualizowany) nie ma dostępu do ważnych części pamięci fizycznej, takich jak te używane przez urządzenia wejścia/wyjścia

Inne problemy

- goszczący SO chce mieć dostęp do całej pamięci mu przydzielonej - musi ona być dzielona z VMM w celu komunikacji
- gość (system wirtualizowany) nie ma dostępu do ważnych części pamięci fizycznej, takich jak te używane przez urządzenia wejścia/wyjścia

Wniosek

VMM musi zarządzać również dostępem do pamięci - spadek wydajności

Problemy z wydajnością

- procesor posiada sprzętowe wspomaganie przełączania kontekstu pomiędzy systemem operacyjnym a innymi aplikacjami dzieki instrukcjom takim jak **SYSENTER** czy **SYSEXIT**

Problemy z wydajnością

- procesor posiada sprzętowe wspomaganie przełączania kontekstu pomiędzy systemem operacyjnym a innymi aplikacjami dzieki instrukcjom takim jak **SYSENTER** czy **SYSEXIT**

w momencie gdy trzeba je emulować, wydajność wirtualizowanego systemu drastycznie spada

Problemy z wydajnością

- procesor posiada sprzętowe wspomaganie przełączania kontekstu pomiędzy systemem operacyjnym a innymi aplikacjami dzieki instrukcjom takim jak **SYSENTER** czy **SYSEXIT**

w momencie gdy trzeba je emulować, wydajność wirtualizowanego systemu drastycznie spada

- dostęp do części zasobów jest pośredni

Problemy z wydajnością

- procesor posiada sprzętowe wspomaganie przełączania kontekstu pomiędzy systemem operacyjnym a innymi aplikacjami dzieki instrukcjom takim jak **SYSENTER** czy **SYSEXIT**

w momencie gdy trzeba je emulować, wydajność wirtualizowanego systemu drastycznie spada

- dostęp do części zasobów jest pośredni

wydłużenie czasu dostępu

Problemy z wydajnością

- procesor posiada sprzętowe wspomaganie przełączania kontekstu pomiędzy systemem operacyjnym a innymi aplikacjami dzieki instrukcjom takim jak **SYSENTER** czy **SYSEXIT**

w momencie gdy trzeba je emulować, wydajność wirtualizowanego systemu drastycznie spada

- dostęp do części zasobów jest pośredni

wydłużenie czasu dostępu

- VMM działające w trybie hosted jest jedną z aplikacji

Problemy z wydajnością

- procesor posiada sprzętowe wspomaganie przełączania kontekstu pomiędzy systemem operacyjnym a innymi aplikacjami dzieki instrukcjom takim jak **SYSENTER** czy **SYSEXIT**

w momencie gdy trzeba je emulować, wydajność wirtualizowanego systemu drastycznie spada

- dostęp do części zasobów jest pośredni

wydłużenie czasu dostępu

- VMM działające w trybie hosted jest jedną z aplikacji

procesor jest dzielony z innymi aplikacjami działającymi w tym czasie

Pierwsze podejście do wirtualizacji wspomaganej sprzętowo

Pomysł

Wprowadzenie sprzętowej warstwy pośredniczącej, wspomagającej wirtualizację poprzez poziom Ring 1.

Pierwsze podejście do wirtualizacji wspomaganej sprzętowo

Pomysł

Wprowadzenie sprzętowej warstwy pośredniczącej, wspomagającej wirtualizację poprzez poziom Ring 1.

Realizacja

- AMD-V
- Intel-VT

Ring 1

- W normalnym środowisku x86, OS pracuje w trybie Ring 0

Ring 1

- W normalnym środowisku x86, OS pracuje w trybie Ring 0
- Rozszerzenie zestawu instrukcji czy też wprowadzenie nowych elementów hardware'owych, działających w nowym uprzywilejowanym i chronionym trybie Ring 1 umożliwia wirtualnym maszynom przekierowanie komunikacji z Ring 0 do Ring 1

Ring 1

- W normalnym środowisku x86, OS pracuje w trybie Ring 0
- Rozszerzenie zestawu instrukcji czy też wprowadzenie nowych elementów hardware'owych, działających w nowym uprzywilejowanym i chronionym trybie Ring 1 umożliwia wirtualnym maszynom przekierowanie komunikacji z Ring 0 do Ring 1
  - Wirtualizowane systemy operacyjne współistnieją ze sobą bez świadomości o istnieniu pozostałych systemów

AMD-V

AMD-V

- rozszerzenie wprowadzone przez firmę AMD do swoich 64-bitowych procesorów (x86-64, nazwa kodowa: *Pacifica*)
- dodatkowo Direct Connect

AMD-V

AMD-V

- rozszerzenie wprowadzone przez firmę AMD do swoich 64-bitowych procesorów (x86-64, nazwa kodowa: *Pacifica*)
- dodatkowo Direct Connect

- wbudowany kontroler pamięci z obsługą wirtualizacji - znaczne usprawnienie wirtualizacji, separacja pamięci operacyjnej poszczególnych VM

AMD-V

AMD-V

- rozszerzenie wprowadzone przez firmę AMD do swoich 64-bitowych procesorów (x86-64, nazwa kodowa: *Pacifica*)
- dodatkowo Direct Connect

- wbudowany kontroler pamięci z obsługą wirtualizacji - znaczne usprawnienie wirtualizacji, separacja pamięci operacyjnej poszczególnych VM
- VMCB (Virtual Machine Control Block) - zawiera informacje o tym, jakie akcje mają być obsługiwane przez monitor VM oraz struktury opisujące, które informacje (zasoby) są widoczne dla systemu gościa.

AMD-V

AMD-V

- rozszerzenie wprowadzone przez firmę AMD do swoich 64-bitowych procesorów (x86-64, nazwa kodowa: *Pacifica*)
- dodatkowo Direct Connect

- wbudowany kontroler pamięci z obsługą wirtualizacji - znaczne usprawnienie wirtualizacji, separacja pamięci operacyjnej poszczególnych VM
- VMCB (Virtual Machine Control Block) - zawiera informacje o tym, jakie akcje mają być obsługiwane przez monitor VM oraz struktury opisujące, które informacje (zasoby) są widoczne dla systemu gościa.
- dodatkowe instrukcje VMRUN (entry i exit), VMCALL

AMD-V

AMD-V

- rozszerzenie wprowadzone przez firmę AMD do swoich 64-bitowych procesorów (x86-64, nazwa kodowa: *Pacifica*)
- dodatkowo Direct Connect

- wbudowany kontroler pamięci z obsługą wirtualizacji - znaczne usprawnienie wirtualizacji, separacja pamięci operacyjnej poszczególnych VM
- VMCB (Virtual Machine Control Block) - zawiera informacje o tym, jakie akcje mają być obsługiwane przez monitor VM oraz struktury opisujące, które informacje (zasoby) są widoczne dla systemu gościa.
- dodatkowe instrukcje VMRUN (entry i exit), VMCALL
- stan hosta jest zapamiętywany przy pomocy MSR (Model Specific Register)

AMD-V

AMD-V

- rozszerzenie wprowadzone przez firmę AMD do swoich 64-bitowych procesorów (x86-64, nazwa kodowa: *Pacifica*)
- dodatkowo Direct Connect

- wbudowany kontroler pamięci z obsługą wirtualizacji - znaczne usprawnienie wirtualizacji, separacja pamięci operacyjnej poszczególnych VM
- VMCB (Virtual Machine Control Block) - zawiera informacje o tym, jakie akcje mają być obsługiwane przez monitor VM oraz struktury opisujące, które informacje (zasoby) są widoczne dla systemu gościa.
- dodatkowe instrukcje VMRUN (entry i exit), VMCALL
- stan hosta jest zapamiętywany przy pomocy MSR (Model Specific Register)
- kontroler pamięci (IOMMU - I/O Memory Management Unit) - o tym więcej później :)

AMD-V

AMD-V

- rozszerzenie wprowadzone przez firmę AMD do swoich 64-bitowych procesorów (x86-64, nazwa kodowa: *Pacifica*)
- dodatkowo Direct Connect

- wbudowany kontroler pamięci z obsługą wirtualizacji - znaczne usprawnienie wirtualizacji, separacja pamięci operacyjnej poszczególnych VM
- VMCB (Virtual Machine Control Block) - zawiera informacje o tym, jakie akcje mają być obsługiwane przez monitor VM oraz struktury opisujące, które informacje (zasoby) są widoczne dla systemu gościa.
- dodatkowe instrukcje VMRUN (entry i exit), VMCALL
- stan hosta jest zapamiętywany przy pomocy MSR (Model Specific Register)
- kontroler pamięci (IOMMU - I/O Memory Management Unit) - o tym więcej później :)
- zmniejszenie obciążenia poprzez selektywne przechwytywanie instrukcji przeznaczonych dla środowisk gości

AMD-V: Direct Connect

Direct Connect

- bezpośrednio łączy procesory, zintegrowany kontroler pamięci oraz układ I/O z centralną jednostką obliczeniową, umożliwiając komunikację z pełną prędkością pracy procesora

AMD-V: Direct Connect

Direct Connect

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

Intel-VT

Intel-VT

- rozszerzenie wprowadzone przez firmę Intel do swoich procesorów (nazwa kodowa: *Vanderpool*)

Intel-VT

Intel-VT

- rozszerzenie wprowadzone przez firmę Intel do swoich procesorów (nazwa kodowa: *Vanderpool*)
- dla procesorów 32-bitowych: **VT-x**

Intel-VT

Intel-VT

- rozszerzenie wprowadzone przez firmę Intel do swoich procesorów (nazwa kodowa: *Vanderpool*)
- dla procesorów 32-bitowych: **VT-x**
- dla procesorów 64-bitowych (Itanium): **VT-i**

Intel-VT

Intel-VT

- rozszerzenie wprowadzone przez firmę Intel do swoich procesorów (nazwa kodowa: *Vanderpool*)
- dla procesorów 32-bitowych: **VT-x**
- dla procesorów 64-bitowych (Itanium): **VT-i**
- dodatkowo **VT-d**: Directed I/O - o tym więcej na późniejszych slajdach

Intel VT-x oraz VT-i

VT-x oraz VT-i

- architektura wzbogacona o 2 nowe formy operacji: VMX root operations oraz VMX non-root operations

Intel VT-x oraz VT-i

VT-x oraz VT-i

- architektura wzbogacona o 2 nowe formy operacji: VMX root operations oraz VMX non-root operations
  - VMX root operations - używane przez VMM do własnych potrzeb

Intel VT-x oraz VT-i

VT-x oraz VT-i

- architektura wzbogacona o 2 nowe formy operacji: VMX root operations oraz VMX non-root operations
  - VMX root operations - używane przez VMM do własnych potrzeb
  - VMX non-root operations - zaprojektowane by stworzyć środowisko kontrolowane przez VMM, ale zaprojektowane by wspierać VM.

Intel VT-x oraz VT-i

VT-x oraz VT-i

- architektura wzbogacona o 2 nowe formy operacji: VMX root operations oraz VMX non-root operations
  - VMX root operations - używane przez VMM do własnych potrzeb
  - VMX non-root operations - zaprojektowane by stworzyć środowisko kontrolowane przez VMM, ale zaprojektowane by wspierać VM.
- obie formy operacji obsługują 4 poziomy uprzywilejowania - instrukcje wirtualizowanego SO wykonują się na takim poziomie, dla jakiego zostały zaprojektowane

Intel VT-x oraz VT-i

VT-x oraz VT-i

- architektura wzbogacona o 2 nowe formy operacji: VMX root operations oraz VMX non-root operations
  - VMX root operations - używane przez VMM do własnych potrzeb
  - VMX non-root operations - zaprojektowane by stworzyć środowisko kontrolowane przez VMM, ale zaprojektowane by wspierać VM.
- obie formy operacji obsługują 4 poziomy uprzywilejowania - instrukcje wirtualizowanego SO wykonują się na takim poziomie, dla jakiego zostały zaprojektowane
- zdefiniowane nowe przejścia między stanami procesora: VM entry oraz VM exit

Intel VT-x oraz VT-i

VT-x oraz VT-i

- architektura wzbogacona o 2 nowe formy operacji: VMX root operations oraz VMX non-root operations
  - VMX root operations - używane przez VMM do własnych potrzeb
  - VMX non-root operations - zaprojektowane by stworzyć środowisko kontrolowane przez VMM, ale zaprojektowane by wspierać VM.
- obie formy operacji obsługują 4 poziomy uprzywilejowania - instrukcje wirtualizowanego SO wykonują się na takim poziomie, dla jakiego zostały zaprojektowane
- zdefiniowane nowe przejścia między stanami procesora: VM entry oraz VM exit
- przejścia zarządzane przez VMCS (Virtual Machine Control Structure, por. AMD VMCB), strukturę zawierającą grupy pól guest-state-area oraz host-state-area pamiętające stan procesora dla poszczególnych maszyn

Intel VT-x oraz VT-i

VT-x oraz VT-i

- architektura wzbogacona o 2 nowe formy operacji: VMX root operations oraz VMX non-root operations
  - VMX root operations - używane przez VMM do własnych potrzeb
  - VMX non-root operations - zaprojektowane by stworzyć środowisko kontrolowane przez VMM, ale zaprojektowane by wspierać VM.
- obie formy operacji obsługują 4 poziomy uprzywilejowania - instrukcje wirtualizowanego SO wykonują się na takim poziomie, dla jakiego zostały zaprojektowane
- zdefiniowane nowe przejścia między stanami procesora: VM entry oraz VM exit
- przejścia zarządzane przez VMCS (Virtual Machine Control Structure, por. AMD VMCB), strukturę zawierającą grupy pól guest-state-area oraz host-state-area pamiętające stan procesora dla poszczególnych maszyn
- dodatkowo wprowadzono instrukcje: VMPTRLD, VMPTRSD, VMCLEAR, VMREAD, VMWRITE, VMCALL, VMLAUNCH, VMRESUME, VMXOFF, VMXON

Pamięć wirtualna maszyny wirtualnej

Pamięć wirtualna maszyny wirtualnej

- Gościnny system operacyjny może korzystać z pamięci wirtualnej

Pamięć wirtualna maszyny wirtualnej

- Gościnny system operacyjny może korzystać z pamięci wirtualnej
- Oczekuje od MMU translacji adresów wirtualnych na fizyczne

Pamięć wirtualna maszyny wirtualnej

- Gościnny system operacyjny może korzystać z pamięci wirtualnej
- Oczekuje od MMU translacji adresów wirtualnych na fizyczne
- Otrzymane adresy są fizyczne tylko w z perspektywy gościa - są wirtualnymi adresami gospodarza

Pamięć wirtualna maszyny wirtualnej

- Gościnny system operacyjny może korzystać z pamięci wirtualnej
- Oczekuje od MMU translacji adresów wirtualnych na fizyczne
- Otrzymane adresy są fizyczne tylko w z perspektywy gościa - są wirtualnymi adresami gospodarza
- Potrzebujemy tablic stron tłumaczących guest-virtual $\rightarrow$ host-physical

Jak to działa

Jak to działa

- Gość utrzymuje własne tablice stron (guest page tables - gPT).

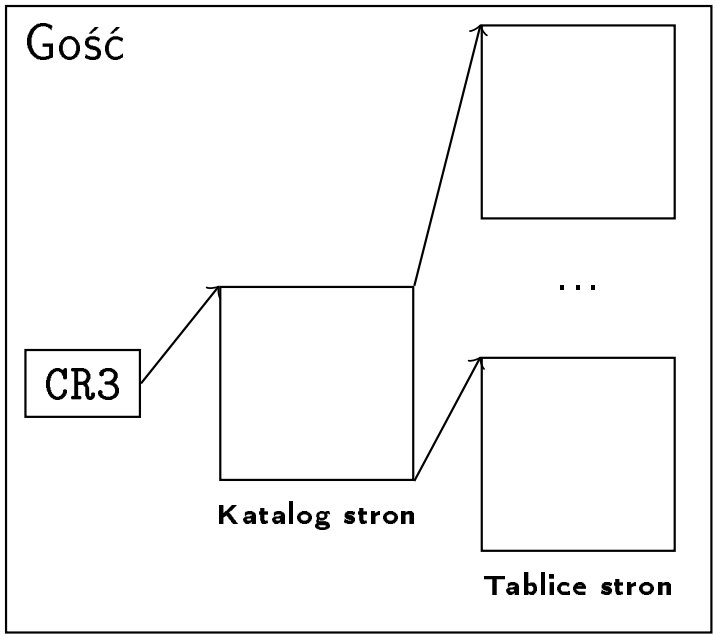
Jak to działa

- Gość utrzymuje własne tablice stron (guest page tables - gPT).
- VMM zmusza CPU do korzystania z tablic stron ukrytych przed gościem (shadow page tables - sPT).

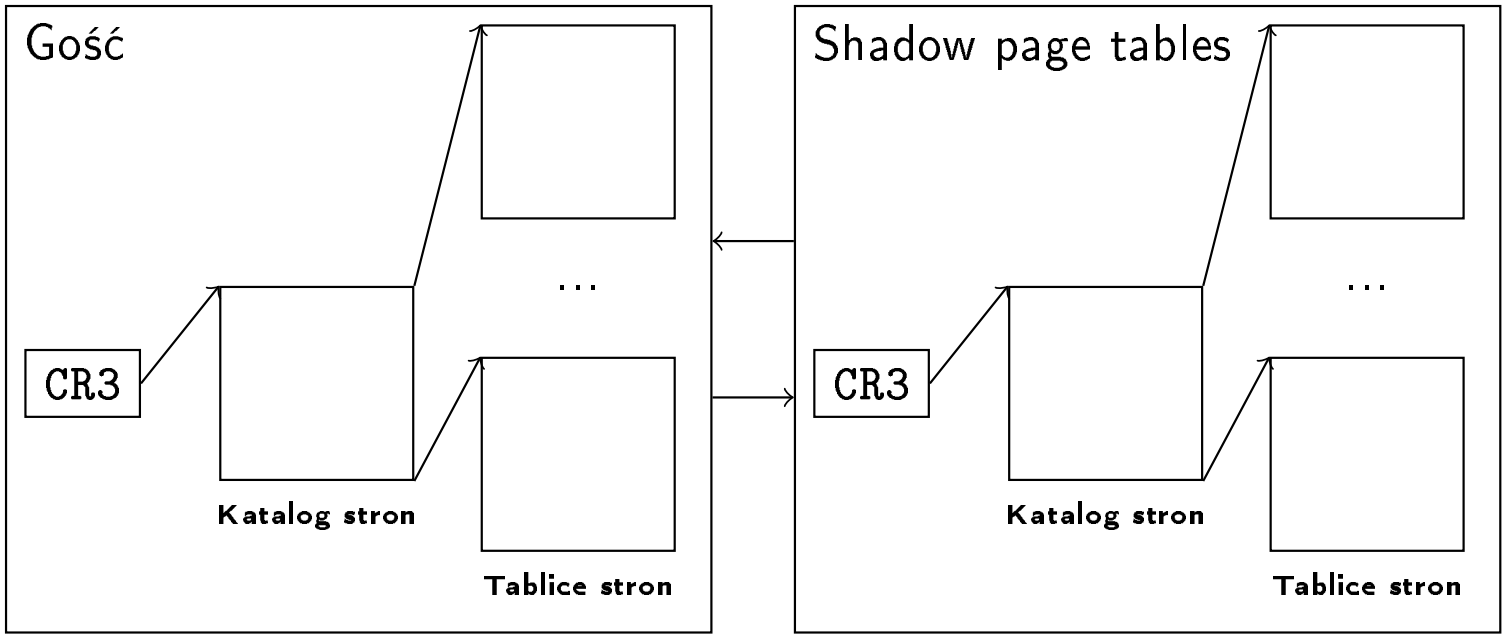
Jak to działa

- Gość utrzymuje własne tablice stron (guest page tables - gPT).
- VMM zmusza CPU do korzystania z tablic stron ukrytych przed gościem (shadow page tables - sPT).
- VMM utrzymuje sPT i TLB w spójności z gPT.

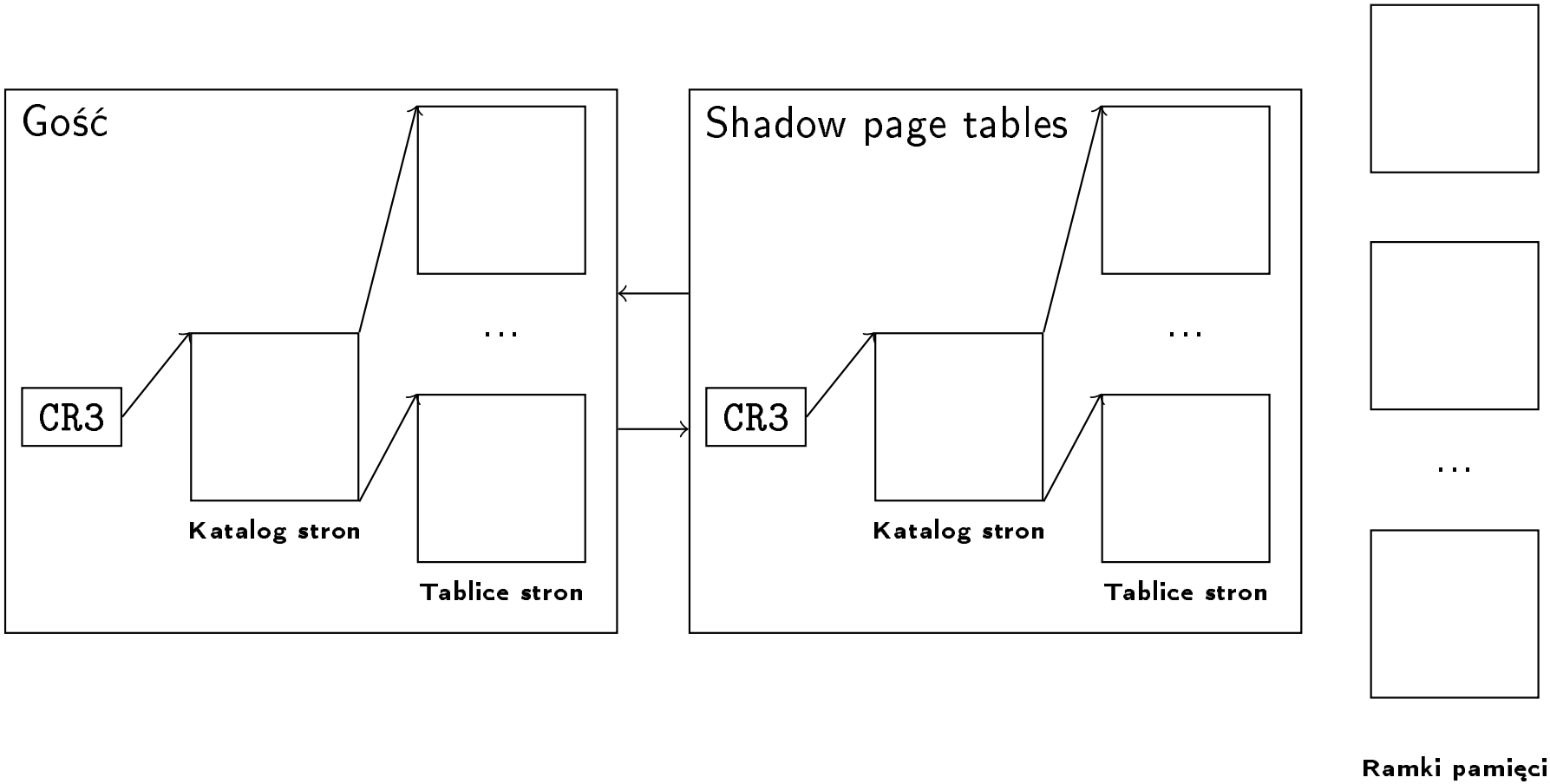
Shadow Page Tables - diagram



Shadow Page Tables - diagram



Shadow Page Tables - diagram



Przykład implementacji

Przykład implementacji

Virtual TLB

- Zakłada się, że system operacyjny dba o spójność wpisów w TLB

Przykład implementacji

Virtual TLB

- Zakłada się, że system operacyjny dba o spójność wpisów w TLB
- VMM przechwytuje operacje na rejestrze CR3 (wskaźnik katalogu stron), INVLPG (unieważnienie wpisu w TLB), oraz page fault

Przykład implementacji

Virtual TLB

- Zakłada się, że system operacyjny dba o spójność wpisów w TLB
- VMM przechwytuje operacje na rejestrze CR3 (wskaźnik katalogu stron), INVLPG (unieważnienie wpisu w TLB), oraz page fault
- W wypadku page fault przechodzi przez gPT i hostPT, pobiera potrzebną stronę do ramki i umieszcza odpowiedni wpis w sPT.

Przykład implementacji

Virtual TLB

- Zakłada się, że system operacyjny dba o spójność wpisów w TLB
- VMM przechwytuje operacje na rejestrze CR3 (wskaźnik katalogu stron), INVLPG (unieważnienie wpisu w TLB), oraz page fault
- W wypadku page fault przechodzi przez gPT i hostPT, pobiera potrzebną stronę do ramki i umieszcza odpowiedni wpis w sPT.
- Przy INVLPG czyści odpowiednie wpisy w sPT.

Przykład implementacji

Virtual TLB

- Zakłada się, że system operacyjny dba o spójność wpisów w TLB
- VMM przechwytuje operacje na rejestrze CR3 (wskaźnik katalogu stron), INVLPG (unieważnienie wpisu w TLB), oraz page fault
- W wypadku page fault przechodzi przez gPT i hostPT, pobiera potrzebną stronę do ramki i umieszcza odpowiedni wpis w sPT.
- Przy INVLPG czyści odpowiednie wpisy w sPT.

write-protected gPT

- gPT jest chronione przed zapisem przez gościa

Przykład implementacji

Virtual TLB

- Zakłada się, że system operacyjny dba o spójność wpisów w TLB
- VMM przechwytuje operacje na rejestrze CR3 (wskaźnik katalogu stron), INVLPG (unieważnienie wpisu w TLB), oraz page fault
- W wypadku page fault przechodzi przez gPT i hostPT, pobiera potrzebną stronę do ramki i umieszcza odpowiedni wpis w sPT.
- Przy INVLPG czyści odpowiednie wpisy w sPT.

write-protected gPT

- gPT jest chronione przed zapisem przez gościa
- zapis powoduje page fault i VMM może zaemulować operacje dbając o spójność z sPT

Podsumowanie shadow page tables

Zalety

Wady

Podsumowanie shadow page tables

Zalety

- Nie ma potrzeby emulowania/rekompilacji każdej instrukcji - w wypadku stron znajdujących się w sPT MMU dokonuje translacji adresów

Wady

Podsumowanie shadow page tables

Zalety

- Nie ma potrzeby emulowania/rekompilacji każdej instrukcji - w wypadku stron znajdujących się w sPT MMU dokonuje translacji adresów

Wady

- Częsty page fault. Może być też powodowany przez gościa, który przeniósł stronę do pamięci pomocniczej. Żeby się o tym przekonać i tak trzeba przejść przez sPT, gPT i hostPT.

Podsumowanie shadow page tables

Zalety

- Nie ma potrzeby emulowania/rekompilacji każdej instrukcji - w wypadku stron znajdujących się w sPT MMU dokonuje translacji adresów

Wady

- Częsty page fault. Może być też powodowany przez gościa, który przeniósł stronę do pamięci pomocniczej. Żeby się o tym przekonać i tak trzeba przejść przez sPT, gPT i hostPT.
- Konieczność synchronizacji sPT $\rightarrow$ gPT (bit dirty i accessed).

Podsumowanie shadow page tables

Zalety

- Nie ma potrzeby emulowania/rekompilacji każdej instrukcji - w wypadku stron znajdujących się w sPT MMU dokonuje translacji adresów

Wady

- Częsty page fault. Może być też powodowany przez gościa, który przeniósł stronę do pamięci pomocniczej. Żeby się o tym przekonać i tak trzeba przejść przez sPT, gPT i hostPT.
- Konieczność synchronizacji sPT $\rightarrow$ gPT (bit dirty i accessed).
- Wysokie zużycie pamięci - dla każdego procesu gościa nowe sPT.

Podsumowanie shadow page tables

Zalety

- Nie ma potrzeby emulowania/rekompilacji każdej instrukcji - w wypadku stron znajdujących się w sPT MMU dokonuje translacji adresów

Wady

- Częsty page fault. Może być też powodowany przez gościa, który przeniósł stronę do pamięci pomocniczej. Żeby się o tym przekonać i tak trzeba przejść przez sPT, gPT i hostPT.
- Konieczność synchronizacji sPT $\rightarrow$ gPT (bit dirty i accessed).
- Wysokie zużycie pamięci - dla każdego procesu gościa nowe sPT.
- Obsługa sPT może stanowić do 75% czasu działania VMM.

AMD Nested Page Tables i Intel Extended Page Tables

AMD Nested Page Tables i Intel Extended Page Tables

- Sprzętowy page walker może dokonać podwójnej translacji adresów

AMD Nested Page Tables i Intel Extended Page Tables

- Sprzętowy page walker może dokonać podwójnej translacji adresów
- guest-virtual $\rightarrow$ guest-physical $=$ host-virtual $\rightarrow$ host-physical

Podsumowanie nested page tables

Wady

Zalety

Podsumowanie nested page tables

Wady

- Przejście przez nPT jest dwa razy dłuższe od przejścia przez sPT. Może to być odczuwalne przy bardzo niskiej wieloprogramowości gościa.

Zalety

Podsumowanie nested page tables

Wady

- Przejście przez nPT jest dwa razy dłuższe od przejścia przez sPT. Może to być odczuwalne przy bardzo niskiej wieloprogramowości gościa.
- Przejście przez nPT zajmuje więcej pozycji w TLB i zwiększa szanse na nie trafienie.

Zalety

Podsumowanie nested page tables

Wady

- Przejście przez nPT jest dwa razy dłuższe od przejścia przez sPT. Może to być odczuwalne przy bardzo niskiej wieloprogramowości gościa.
- Przejście przez nPT zajmuje więcej pozycji w TLB i zwiększa szanse na nie trafienie.

Zalety

- Mniejsze zużycie pamięci.

Podsumowanie nested page tables

Wady

- Przejście przez nPT jest dwa razy dłuższe od przejścia przez sPT. Może to być odczuwalne przy bardzo niskiej wieloprogramowości gościa.
- Przejście przez nPT zajmuje więcej pozycji w TLB i zwiększa szanse na nie trafienie.

Zalety

- Mniejsze zużycie pamięci.
- Zmiana kontekstu przez gościa (edycja CR3) nie wymaga obsługi VMM.

Podsumowanie nested page tables

Wady

- Przejście przez nPT jest dwa razy dłuższe od przejścia przez sPT. Może to być odczuwalne przy bardzo niskiej wieloprogramowości gościa.
- Przejście przez nPT zajmuje więcej pozycji w TLB i zwiększa szanse na nie trafienie.

Zalety

- Mniejsze zużycie pamięci.
- Zmiana kontekstu przez gościa (edycja CR3) nie wymaga obsługi VMM.
- Prostszy VMM - prostsza obsługa page fault.

Urządzania w maszynie wirtualnej

System operacyjny gościa będzie szukał urządzeń na swojej maszynie

Urządzania w maszynie wirtualnej

System operacyjny gościa będzie szukał urządzeń na swojej maszynie

- Może nie być świadomy uruchomienia na wirtualnej maszynia

Urządzania w maszynie wirtualnej

System operacyjny gościa będzie szukał urządzeń na swojej maszynie

- Może nie być świadomy uruchomienia na wirtualnej maszynia
- Przechowywanie danych (HDD, ROM)

Urządzania w maszynie wirtualnej

System operacyjny gościa będzie szukał urządzeń na swojej maszynie

- Może nie być świadomy uruchomienia na wirtualnej maszynia
- Przechowywanie danych (HDD, ROM)
- Komunikacja (karta sieciowa)

Urządzania w maszynie wirtualnej

System operacyjny gościa będzie szukał urządzeń na swojej maszynie

- Może nie być świadomy uruchomienia na wirtualnej maszynia
- Przechowywanie danych (HDD, ROM)
- Komunikacja (karta sieciowa)
- ...

Podejście pierwsze: emulacja

Podejście pierwsze: emulacja

- Gościnny system operacyjny korzysta ze zwykłych sterowników do urządzenia

Podejście pierwsze: emulacja

- Gościnny system operacyjny korzysta ze zwykłych sterowników do urządzenia
- Jednak zamiast komunikować się z urządzeniem, komunikuje się z VMM

Podejście pierwsze: emulacja

- Gościnny system operacyjny korzysta ze zwykłych sterowników do urządzenia
- Jednak zamiast komunikować się z urządzeniem, komunikuje się z VMM
- VMM emuluje operacje wykonywane przez urządzenie

Podejście pierwsze: emulacja

- Gościnny system operacyjny korzysta ze zwykłych sterowników do urządzenia
- Jednak zamiast komunikować się z urządzeniem, komunikuje się z VMM
- VMM emuluje operacje wykonywane przez urządzenie

Zalety

- Nie trzeba modyfikować gościnnego systemu operacyjnego, ani pisać nowych sterowników

Podejście pierwsze: emulacja

- Gościnny system operacyjny korzysta ze zwykłych sterowników do urządzenia
- Jednak zamiast komunikować się z urządzeniem, komunikuje się z VMM
- VMM emuluje operacje wykonywane przez urządzenie

Zalety

- Nie trzeba modyfikować gościnnego systemu operacyjnego, ani pisać nowych sterowników

Wady

- Częste przełączanie gość $\leftrightarrow$ VMM.

Podejście pierwsze: emulacja

- Gościnny system operacyjny korzysta ze zwykłych sterowników do urządzenia
- Jednak zamiast komunikować się z urządzeniem, komunikuje się z VMM
- VMM emuluje operacje wykonywane przez urządzenie

Zalety

- Nie trzeba modyfikować gościnnego systemu operacyjnego, ani pisać nowych sterowników

Wady

- Częste przełączanie gość $\leftrightarrow$ VMM.
- Sterownik w VMM musi interpretować niskopoziomowe polecenia od gościa.

Podejście pierwsze: emulacja

- Gościnny system operacyjny korzysta ze zwykłych sterowników do urządzenia
- Jednak zamiast komunikować się z urządzeniem, komunikuje się z VMM
- VMM emuluje operacje wykonywane przez urządzenie

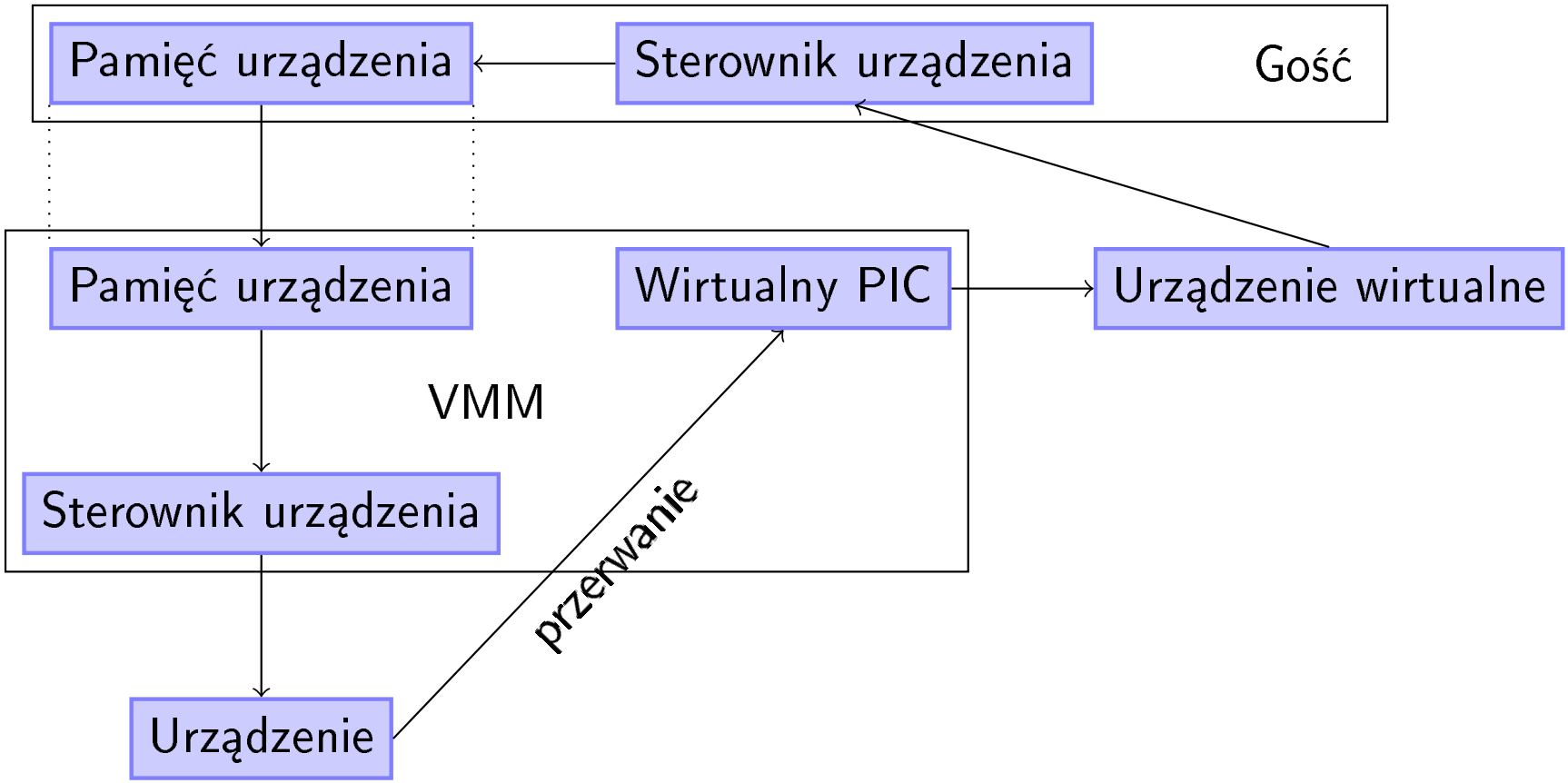
Zalety

- Nie trzeba modyfikować gościnnego systemu operacyjnego, ani pisać nowych sterowników

Wady

- Częste przełączanie gość $\leftrightarrow$ VMM.
- Sterownik w VMM musi interpretować niskopoziomowe polecenia od gościa.
- Czasami sterownik musi emulować bugi w urządzeniu.

Emulacja urządzeń: diagram



Podejście drugie: parawirtualizacja

Podejście drugie: parawirtualizacja

- Gościnny system operacyjny korzysta ze zmodyfikowanych sterowników do urządzenia.

Podejście drugie: parawirtualizacja

- Gościnny system operacyjny korzysta ze zmodyfikowanych sterowników do urządzenia.
- Komunikuje się z VMM, które zajmuje się obsługą urządzenia.

Podejście drugie: parawirtualizacja

- Gościnny system operacyjny korzysta ze zmodyfikowanych sterowników do urządzenia.
- Komunikuje się z VMM, które zajmuje się obsługą urządzenia.
- Komunikacja odbywa się na wyższym poziomie abstrakcji, niż z fizycznym urządzeniem.

Podejście drugie: parawirtualizacja

- Gościnny system operacyjny korzysta ze zmodyfikowanych sterowników do urządzenia.
- Komunikuje się z VMM, które zajmuje się obsługą urządzenia.
- Komunikacja odbywa się na wyższym poziomie abstrakcji, niż z fizycznym urządzeniem.

Zalety

- Znacząco mniejsza ilośc wysłanych komunikatów gość $\leftrightarrow$ VMM i mniej przełączeń kontekstu.
- Wyższy poziom abstrakcji upraszcza pisanie sterowników w VMM.

Podejście drugie: parawirtualizacja

- Gościnny system operacyjny korzysta ze zmodyfikowanych sterowników do urządzenia.
- Komunikuje się z VMM, które zajmuje się obsługą urządzenia.
- Komunikacja odbywa się na wyższym poziomie abstrakcji, niż z fizycznym urządzeniem.

Zalety

- Znacząco mniejsza ilośc wysłanych komunikatów gość $\leftrightarrow$ VMM i mniej przełączeń kontekstu.
- Wyższy poziom abstrakcji upraszcza pisanie sterowników w VMM.

Wady

- Dla każdego gościnnego systemu operacyjnego potrzebny jest nowy parasterownik.

Podejście drugie: parawirtualizacja

- Gościnny system operacyjny korzysta ze zmodyfikowanych sterowników do urządzenia.
- Komunikuje się z VMM, które zajmuje się obsługą urządzenia.
- Komunikacja odbywa się na wyższym poziomie abstrakcji, niż z fizycznym urządzeniem.

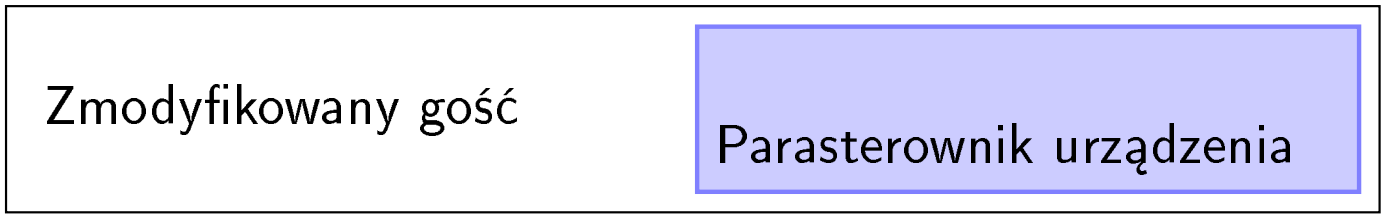
Zalety

- Znacząco mniejsza ilośc wysłanych komunikatów gość $\leftrightarrow$ VMM i mniej przełączeń kontekstu.
- Wyższy poziom abstrakcji upraszcza pisanie sterowników w VMM.

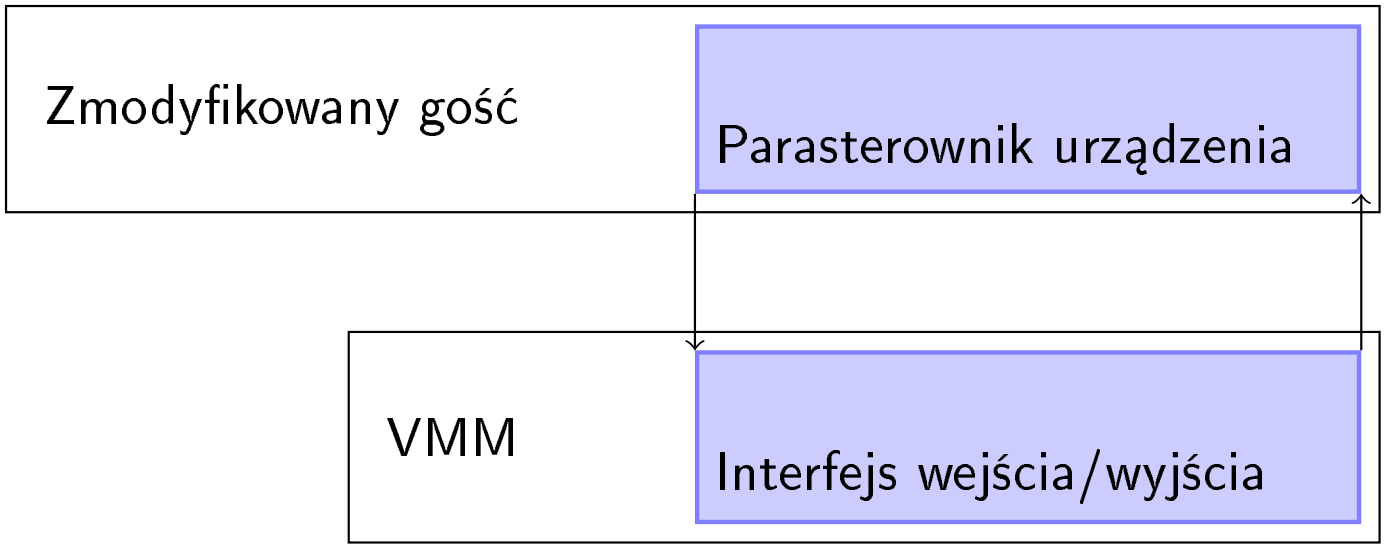
Wady

- Dla każdego gościnnego systemu operacyjnego potrzebny jest nowy parasterownik.
- Wciąż mniej wydajne od fizycznych urządzeń.

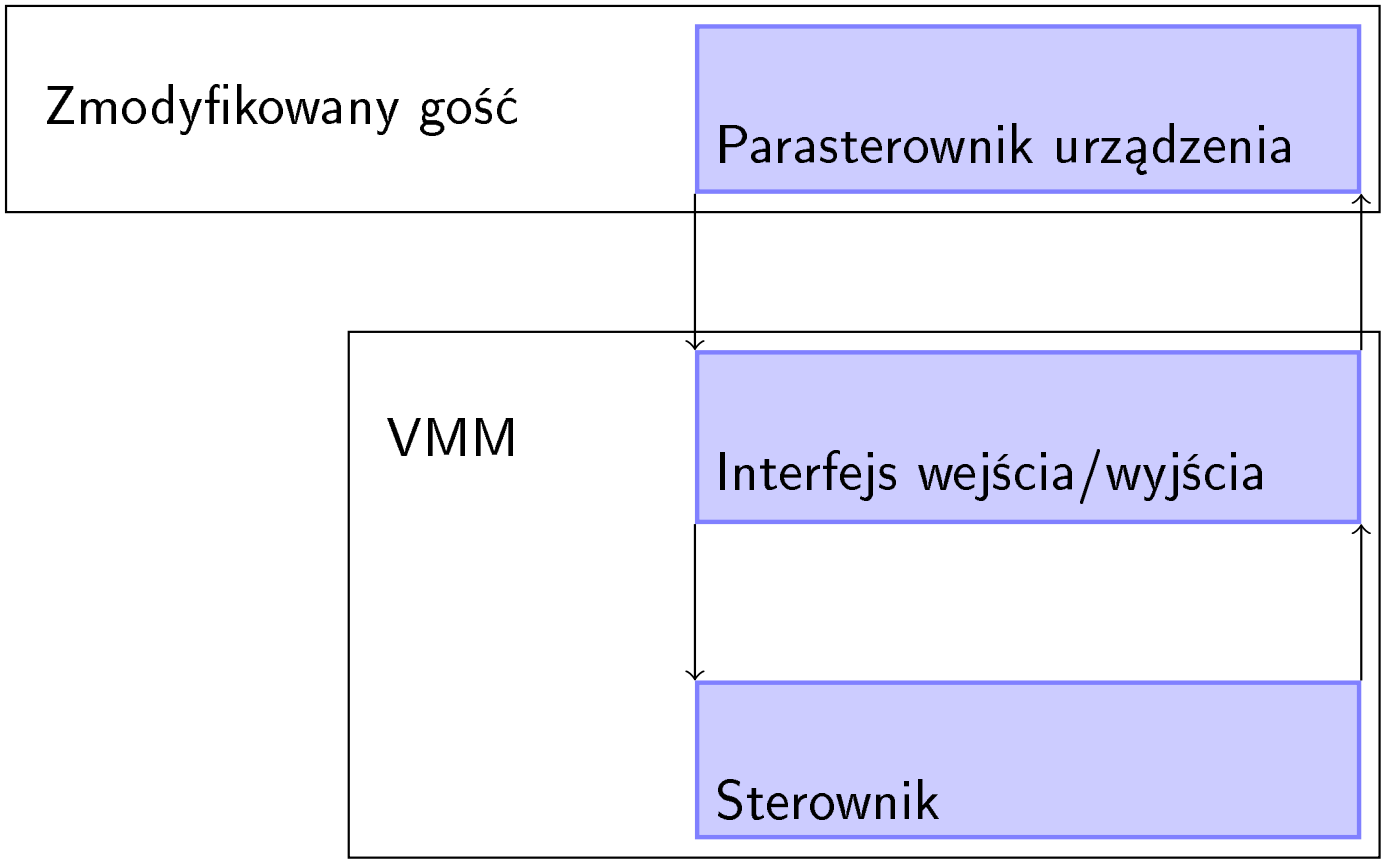
Parawirtualizacja urządzeń: diagram



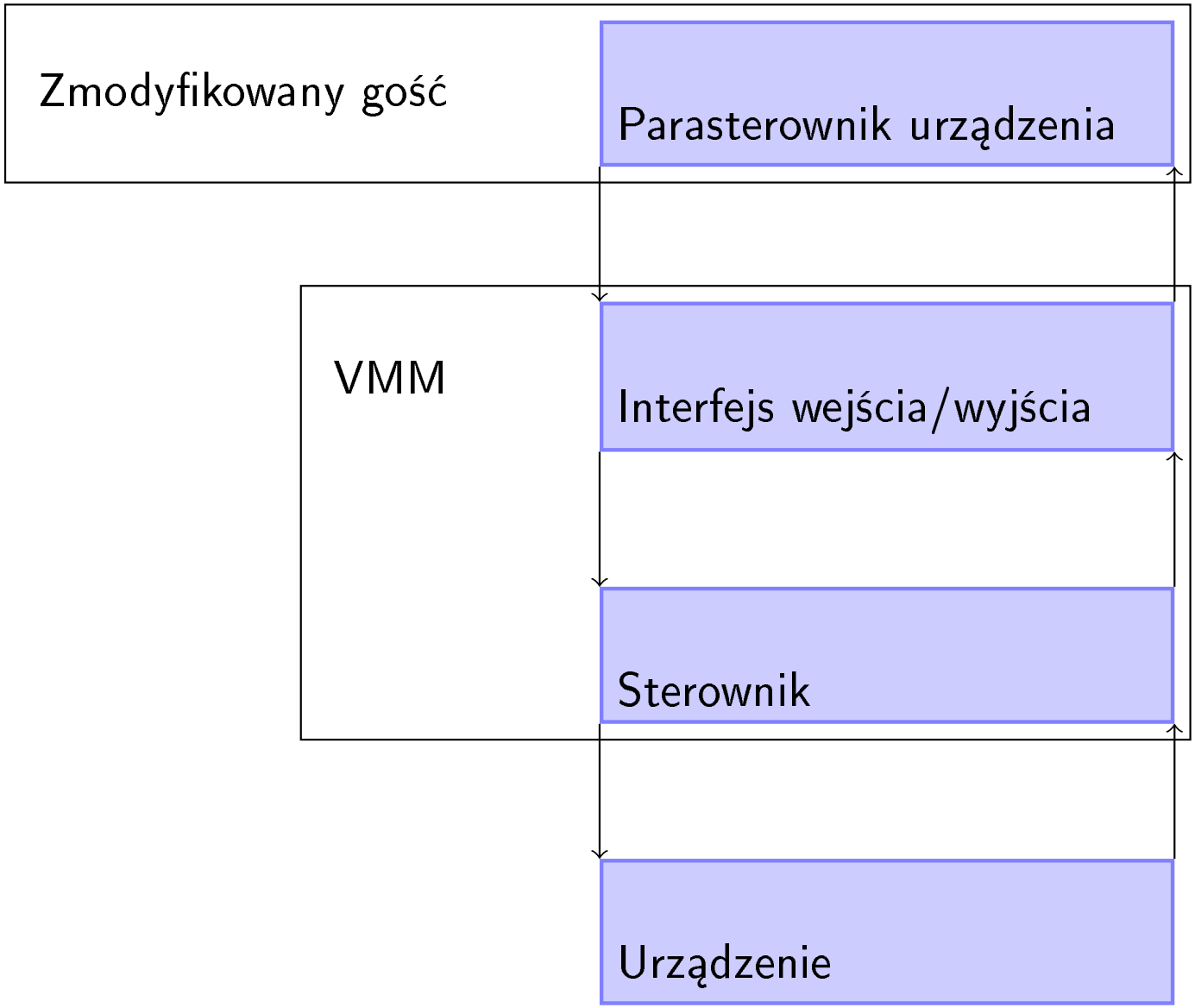
Parawirtualizacja urządzeń: diagram



Parawirtualizacja urządzeń: diagram



Parawirtualizacja urządzeń: diagram



Podejście trzecie: bezpośredni dostęp

Podejście trzecie: bezpośredni dostęp

- Pozwalamy na komunikację VM z wybranym urządzeniem.

Podejście trzecie: bezpośredni dostęp

- Pozwalamy na komunikację VM z wybranym urządzeniem.

Problemy

Podejście trzecie: bezpośredni dostęp

- Pozwalamy na komunikację VM z wybranym urządzeniem.

Problemy

- Nested Page Tables rozwiązują problem tłumaczenia adresów dla CPU, ale urządzenia nie korzystają z MMU.

Podejście trzecie: bezpośredni dostęp

- Pozwalamy na komunikację VM z wybranym urządzeniem.

Problemy

- Nested Page Tables rozwiązują problem tłumaczenia adresów dla CPU, ale urządzenia nie korzystają z MMU.
- Gość używając DMA będzie korzystał ze swoich adresów fizycznych, a nie gospodarza.

Podejście trzecie: bezpośredni dostęp

- Pozwalamy na komunikację VM z wybranym urządzeniem.

Problemy

- Nested Page Tables rozwiązują problem tłumaczenia adresów dla CPU, ale urządzenia nie korzystają z MMU.
- Gość używając DMA będzie korzystał ze swoich adresów fizycznych, a nie gospodarza.
- Dane z urządzeń znajdą się w dosyć losowym miejscu pamięci gospodarza.

Podejście trzecie: bezpośredni dostęp

- Pozwalamy na komunikację VM z wybranym urządzeniem.

Problemy

- Nested Page Tables rozwiązują problem tłumaczenia adresów dla CPU, ale urządzenia nie korzystają z MMU.
- Gość używając DMA będzie korzystał ze swoich adresów fizycznych, a nie gospodarza.
- Dane z urządzeń znajdą się w dosyć losowym miejscu pamięci gospodarza.
- Gość mógłby nawet użyć tego mechanizmu do ucieczki z maszyny wirtualnej.

Podejście trzecie: bezpośredni dostęp

- Pozwalamy na komunikację VM z wybranym urządzeniem.

Problemy

- Nested Page Tables rozwiązują problem tłumaczenia adresów dla CPU, ale urządzenia nie korzystają z MMU.
- Gość używając DMA będzie korzystał ze swoich adresów fizycznych, a nie gospodarza.
- Dane z urządzeń znajdą się w dosyć losowym miejscu pamięci gospodarza.
- Gość mógłby nawet użyć tego mechanizmu do ucieczki z maszyny wirtualnej.

- Problem rozwiązuje wykorzystanie IOMMU.

IOMMU

input/output memory management unit

- AMD IOMMU
- Intel VT-d

IOMMU

input/output memory management unit

- AMD IOMMU
- Intel VT-d
- Sprzętowe rozwiązanie kontrolujące dostęp urządzeń do pamięci.

IOMMU

input/output memory management unit

- AMD IOMMU
- Intel VT-d
- Sprzętowe rozwiązanie kontrolujące dostęp urządzeń do pamięci.

Możliwości

IOMMU

input/output memory management unit

- AMD IOMMU
- Intel VT-d
- Sprzętowe rozwiązanie kontrolujące dostęp urządzeń do pamięci.

Możliwości

- Pamięć wirtualna dla urządzeń - niespójne fragmenty pamięci fizycznej widoczne jako pamięć liniowa.

IOMMU

input/output memory management unit

- AMD IOMMU
- Intel VT-d
- Sprzętowe rozwiązanie kontrolujące dostęp urządzeń do pamięci.

Możliwości

- Pamięć wirtualna dla urządzeń - niespójne fragmenty pamięci fizycznej widoczne jako pamięć liniowa.
- Grupowanie urządzeń w domeny, które współdzielą pamięć wirtualną.

IOMMU

input/output memory management unit

- AMD IOMMU
- Intel VT-d
- Sprzętowe rozwiązanie kontrolujące dostęp urządzeń do pamięci.

Możliwości

- Pamięć wirtualna dla urządzeń - niespójne fragmenty pamięci fizycznej widoczne jako pamięć liniowa.
- Grupowanie urządzeń w domeny, które współdzielą pamięć wirtualną.
- Przemapowanie i kontrola przerwań

Zastosowanie w wirtualizacji

Zastosowanie w wirtualizacji

- IOMMU korzysta z tablic stron.

Zastosowanie w wirtualizacji

- IOMMU korzysta z tablic stron.
- Urządzeniom bezpośrednio wykorzystywanym przez VM daje się dostęp wyłącznie do pamięci gościa.

Zastosowanie w wirtualizacji

- IOMMU korzysta z tablic stron.
- Urządzeniom bezpośrednio wykorzystywanym przez VM daje się dostęp wyłącznie do pamięci gościa.
- IOMMU dokonuje translacji adresów wykorzystywanych w DMA.

Zastosowanie w wirtualizacji

- IOMMU korzysta z tablic stron.
- Urządzeniom bezpośrednio wykorzystywanym przez VM daje się dostęp wyłącznie do pamięci gościa.
- IOMMU dokonuje translacji adresów wykorzystywanych w DMA.
- Nie ma potrzeby ingerencji VMM w czasie korzystania z urządzenia.

Zastosowanie w wirtualizacji

- IOMMU korzysta z tablic stron.
- Urządzeniom bezpośrednio wykorzystywanym przez VM daje się dostęp wyłącznie do pamięci gościa.
- IOMMU dokonuje translacji adresów wykorzystywanych w DMA.
- Nie ma potrzeby ingerencji VMM w czasie korzystania z urządzenia.

Zalety

Zastosowanie w wirtualizacji

- IOMMU korzysta z tablic stron.
- Urządzeniom bezpośrednio wykorzystywanym przez VM daje się dostęp wyłącznie do pamięci gościa.
- IOMMU dokonuje translacji adresów wykorzystywanych w DMA.
- Nie ma potrzeby ingerencji VMM w czasie korzystania z urządzenia.

Zalety

- VM ma szybki dostęp do urządzenia.

Zastosowanie w wirtualizacji

- IOMMU korzysta z tablic stron.
- Urządzeniom bezpośrednio wykorzystywanym przez VM daje się dostęp wyłącznie do pamięci gościa.
- IOMMU dokonuje translacji adresów wykorzystywanych w DMA.
- Nie ma potrzeby ingerencji VMM w czasie korzystania z urządzenia.

Zalety

- VM ma szybki dostęp do urządzenia.

Wady

Zastosowanie w wirtualizacji

- IOMMU korzysta z tablic stron.
- Urządzeniom bezpośrednio wykorzystywanym przez VM daje się dostęp wyłącznie do pamięci gościa.
- IOMMU dokonuje translacji adresów wykorzystywanych w DMA.
- Nie ma potrzeby ingerencji VMM w czasie korzystania z urządzenia.

Zalety

- VM ma szybki dostęp do urządzenia.

Wady

- Potrzebujemy osobnego urządzenia dla każdego VM.

Zastosowanie w wirtualizacji

- IOMMU korzysta z tablic stron.
- Urządzeniom bezpośrednio wykorzystywanym przez VM daje się dostęp wyłącznie do pamięci gościa.
- IOMMU dokonuje translacji adresów wykorzystywanych w DMA.
- Nie ma potrzeby ingerencji VMM w czasie korzystania z urządzenia.

Zalety

- VM ma szybki dostęp do urządzenia.

Wady

- Potrzebujemy osobnego urządzenia dla każdego VM.
- Trwają prace nad rozszerzeniem PCI Express, aby wspierało dzielenie urządzenia między wirtualne maszyny.

Inne zastosowania IOMMU

Inne zastosowania IOMMU

- Bezpośredni dostęp aplikacji do urządzenia.

Inne zastosowania IOMMU

- Bezpośredni dostęp aplikacji do urządzenia.
- Zastąpienie Graphics Address Remapping Table - urządzenie mapujące pamięć dla kart graficznych.

Inne zastosowania IOMMU

- Bezpośredni dostęp aplikacji do urządzenia.
- Zastąpienie Graphics Address Remapping Table - urządzenie mapujące pamięć dla kart graficznych.
- Zastąpienie Device Exclusion Vector - odmowa dostępu wybranych urządzeń do wybranych ramek pamięci.

Inne zastosowania IOMMU

- Bezpośredni dostęp aplikacji do urządzenia.
- Zastąpienie Graphics Address Remapping Table - urządzenie mapujące pamięć dla kart graficznych.
- Zastąpienie Device Exclusion Vector - odmowa dostępu wybranych urządzeń do wybranych ramek pamięci.
- Wspomaganie urządzeń 32-bitowych - dostęp urządzeń do wybranej pamięci bez potrzeby używania bounce buffer.

Inne zastosowania IOMMU

- Bezpośredni dostęp aplikacji do urządzenia.
- Zastąpienie Graphics Address Remapping Table - urządzenie mapujące pamięć dla kart graficznych.
- Zastąpienie Device Exclusion Vector - odmowa dostępu wybranych urządzeń do wybranych ramek pamięci.
- Wspomaganie urządzeń 32-bitowych - dostęp urządzeń do wybranej pamięci bez potrzeby używania bounce buffer.
- Wirtualizacja IOMMU - shadow page tables dla IOMMU.

Wirtualizacja wspomagana sprzętowo

Podejście pozwalające na efektywne uzyskanie pełnej wirtualizacji z pomocą dodatkowych funkcjonalności sprzętu, głównie procesora. Pełna wirtualizacja symuluje pełne środowisko komputerowe wraz ze sprzętem, na którym niezmodyfikowane oprogramowanie napisane dla tego sprzętu (architektury) wykonuje się w całkowitej izolacji.

Wirtualizacja wspomagana sprzętowo

Podejście pozwalające na efektywne uzyskanie pełnej wirtualizacji z pomocą dodatkowych funkcjonalności sprzętu, głównie procesora. Pełna wirtualizacja symuluje pełne środowisko komputerowe wraz ze sprzętem, na którym niezmodyfikowane oprogramowanie napisane dla tego sprzętu (architektury) wykonuje się w całkowitej izolacji.

Dopiero niedawno (2007) dodano dodatkowe instrukcje do procesorów serii x86, pozwalające na uzyskanie pełnej wirtualizacji.

Wirtualizacja wspomagana sprzętowo

Podejście pozwalające na efektywne uzyskanie pełnej wirtualizacji z pomocą dodatkowych funkcjonalności sprzętu, głównie procesora. Pełna wirtualizacja symuluje pełne środowisko komputerowe wraz ze sprzętem, na którym niezmodyfikowane oprogramowanie napisane dla tego sprzętu (architektury) wykonuje się w całkowitej izolacji.

Dopiero niedawno (2007) dodano dodatkowe instrukcje do procesorów serii x86, pozwalające na uzyskanie pełnej wirtualizacji.

Wspomaganie sprzętowe pozwala na wykonywanie natywnie większości instrukcji procesora zawartych w pliku wykonywalnym.

Wirtualizacja wspomagana sprzętowo

Podejście pozwalające na efektywne uzyskanie pełnej wirtualizacji z pomocą dodatkowych funkcjonalności sprzętu, głównie procesora. Pełna wirtualizacja symuluje pełne środowisko komputerowe wraz ze sprzętem, na którym niezmodyfikowane oprogramowanie napisane dla tego sprzętu (architektury) wykonuje się w całkowitej izolacji.

Dopiero niedawno (2007) dodano dodatkowe instrukcje do procesorów serii x86, pozwalające na uzyskanie pełnej wirtualizacji.

Wspomaganie sprzętowe pozwala na wykonywanie natywnie większości instrukcji procesora zawartych w pliku wykonywalnym.

Na architekturach x86 dwa konkurencyjne rozszerzenia instrukcji procesorów, zaproponowane przez Intel i AMD, odpowiednio Vanderpool i Pacifica.

Jest pięknie?

Jest pięknie?

Nawet przy sprzętowej wirtualizacji zostaje sporo instrukcji, które muszą być przechwytywane przez nadzorcę.

Jest pięknie?

Nawet przy sprzętowej wirtualizacji zostaje sporo instrukcji, które muszą być przechwytywane przez nadzorcę.

Pomysł na radzenie sobie z tym kłopotem, to zastosowanie hybrydowej wirtualizacji, tj. sprzętowej wirtualizacji + parawirtualizacji niektórych fragmentów oprogramowania systemowego, np. sterowników sprzętu.

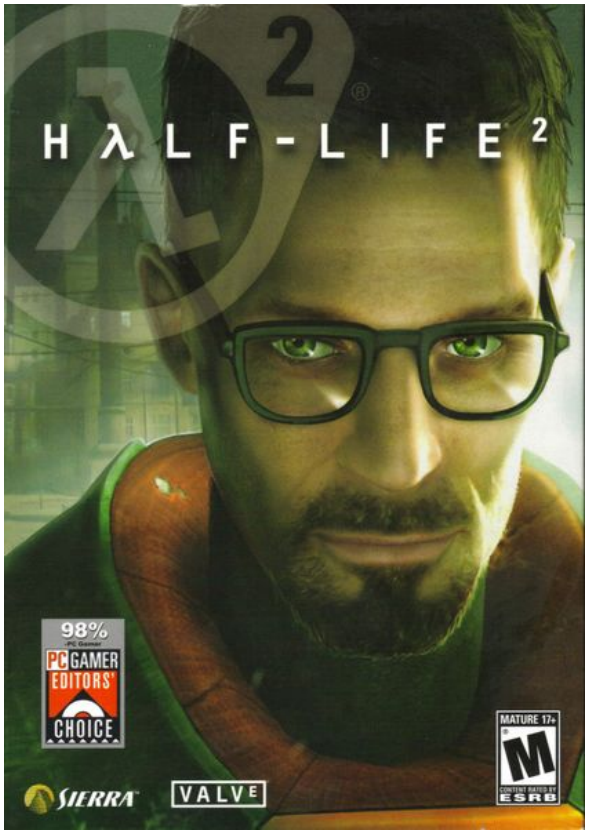
Jest pięknie?

Nawet przy sprzętowej wirtualizacji zostaje sporo instrukcji, które muszą być przechwytywane przez nadzorcę.

Pomysł na radzenie sobie z tym kłopotem, to zastosowanie hybrydowej wirtualizacji, tj. sprzętowej wirtualizacji + parawirtualizacji niektórych fragmentów oprogramowania systemowego, np. sterowników sprzętu.

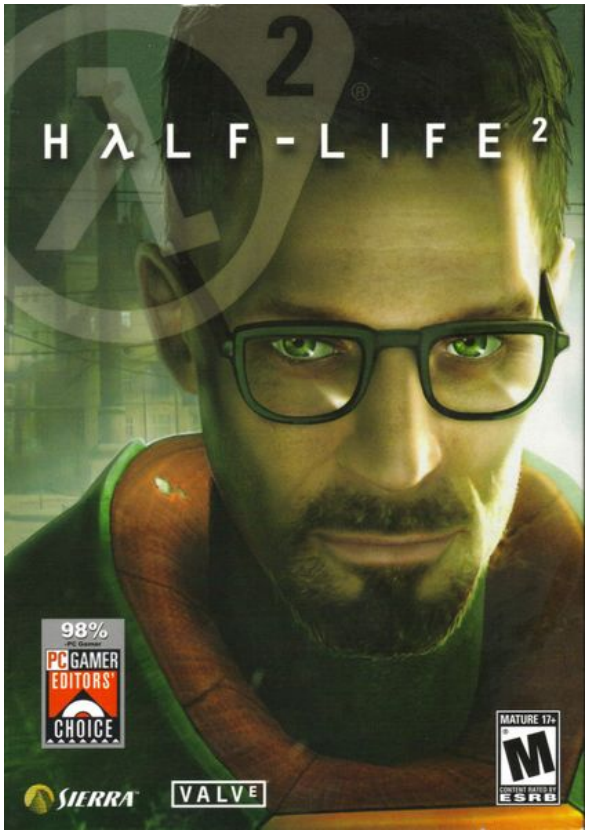
Coś za coś: albo mamy pełną wirtualizację z powolnym przechwytywaniem instrukcji procesora, albo mamy szybką parawirtualizację, która wymaga ingerencji w oprogramowanie.

Wycieczka do Xen



Xen — świat na pograniczu wymiarów w grze *Half–Life*

Wycieczka do Xen



Xen — świat na pograniczu wymiarów w grze *Half–Life* ...lub VMM na licencji GPL

Wycieczka do Xen

Możliwości:

Wycieczka do Xen

Możliwości:

- parawirtualizacja odpowiednio zmodyfikowanych systemów (Linux, *BSD, Solaris)

Wycieczka do Xen

Możliwości:

- parawirtualizacja odpowiednio zmodyfikowanych systemów (Linux, *BSD, Solaris)
- pełna wirtualizacja na komputerach wyposażonych w odpowiednie rozszerzenia instrukcji procesora (wspólny interfejs dla instrukcji Intela i AMD)

Wycieczka do Xen

Możliwości:

- parawirtualizacja odpowiednio zmodyfikowanych systemów (Linux, *BSD, Solaris)
- pełna wirtualizacja na komputerach wyposażonych w odpowiednie rozszerzenia instrukcji procesora (wspólny interfejs dla instrukcji Intela i AMD)
- ciekawy dodatek — przenoszenie "na żywo" działającej wirtualnej maszyny na inny komputer poprzez połączenie sieciowe.

Kernel–based Virtual Machine

KVM to zestaw interfejsów kernela linuksowego, umożliwiający wykorzystanie rozszerzeń Vanderpool i Pacifica do uzyskania pełnej wirtualizacji.

Kernel–based Virtual Machine

KVM to zestaw interfejsów kernela linuksowego, umożliwiający wykorzystanie rozszerzeń Vanderpool i Pacifica do uzyskania pełnej wirtualizacji.

- KVM działa jako moduł kernela, udostępniający przez urządzenie `/dev/kvm` interfejs do obsługi wirtualizowanych systemów.

Kernel–based Virtual Machine

KVM to zestaw interfejsów kernela linuksowego, umożliwiający wykorzystanie rozszerzeń Vanderpool i Pacifica do uzyskania pełnej wirtualizacji.

- KVM działa jako moduł kernela, udostępniający przez urządzenie `/dev/kvm` interfejs do obsługi wirtualizowanych systemów.
  - Zalety takiego podejścia, to możliwość obsługi wirtualizacji z poziomu user–space.

Kernel–based Virtual Machine

KVM to zestaw interfejsów kernela linuksowego, umożliwiający wykorzystanie rozszerzeń Vanderpool i Pacifica do uzyskania pełnej wirtualizacji.

- KVM działa jako moduł kernela, udostępniający przez urządzenie `/dev/kvm` interfejs do obsługi wirtualizowanych systemów.
  - Zalety takiego podejścia, to możliwość obsługi wirtualizacji z poziomu user–space.
- Natywna obsługa niektórych starożytnych systemów nie jest aktualnie możliwa (Windows '95, DOS) i trzeba uciekać się do emulacji — brak zaimplementowanego trybu rzeczywistego.

Kernel–based Virtual Machine

KVM to zestaw interfejsów kernela linuksowego, umożliwiający wykorzystanie rozszerzeń Vanderpool i Pacifica do uzyskania pełnej wirtualizacji.

- KVM działa jako moduł kernela, udostępniający przez urządzenie `/dev/kvm` interfejs do obsługi wirtualizowanych systemów.
  - Zalety takiego podejścia, to możliwość obsługi wirtualizacji z poziomu user–space.
- Natywna obsługa niektórych starożytnych systemów nie jest aktualnie możliwa (Windows '95, DOS) i trzeba uciekać się do emulacji — brak zaimplementowanego trybu rzeczywistego.
- Tak jak Xen, wspiera migrację działających maszyn, w dodatku dobrze sobie radzi robiąc to przez `ssh`.

Jeszcze będzie przepięknie...

KVM to dość młody projekt, chociaż ma już na koncie zdumiewającą listę obsługiwanych systemów, m.in. Windows Vista i Windows Server 2008.

Jeszcze będzie przepięknie...

KVM to dość młody projekt, chociaż ma już na koncie zdumiewającą listę obsługiwanych systemów, m.in. Windows Vista i Windows Server 2008.

Wciąż jest sporo miejsca na poprawę szybkości działania. W niektórych sytuacjach szybciej działa software'owa emulacja poprzez QEMU (emulator procesora) z dołączonym modułem kernela KQEMU, niż teoretycznie lepsza wirtualizacja poprzez KVM.

Jeszcze będzie przepięknie...

KVM to dość młody projekt, chociaż ma już na koncie zdumiewającą listę obsługiwanych systemów, m.in. Windows Vista i Windows Server 2008.

Wciąż jest sporo miejsca na poprawę szybkości działania. W niektórych sytuacjach szybciej działa software'owa emulacja poprzez QEMU (emulator procesora) z dołączonym modułem kernela KQEMU, niż teoretycznie lepsza wirtualizacja poprzez KVM.

Ogólny kierunek rozwoju KVM i Xen skupia się na żmudnym poprawianiu wydajności poprzez poprawę kodu oraz rozwijaniem emulacji dostępu do pamięci, dodawaniu obsługi różnych sprzętowych bajerów niekoniecznie związanych z wydajnością (np. ACPI) oraz nowych urządzeń (USB, SCSI), tudzież gruntownym remoncie niektórych istotnych kawałków kodu źródłowego — QEMU, wykorzystywany w KVM, Xen, a nawet VirtualBox, wciąż nie można skompilować za pomocą GCC 4.

Jeszcze będzie przepięknie...

KVM to dość młody projekt, chociaż ma już na koncie zdumiewającą listę obsługiwanych systemów, m.in. Windows Vista i Windows Server 2008.

Wciąż jest sporo miejsca na poprawę szybkości działania. W niektórych sytuacjach szybciej działa software'owa emulacja poprzez QEMU (emulator procesora) z dołączonym modułem kernela KQEMU, niż teoretycznie lepsza wirtualizacja poprzez KVM.

Ogólny kierunek rozwoju KVM i Xen skupia się na żmudnym poprawianiu wydajności poprzez poprawę kodu oraz rozwijaniem emulacji dostępu do pamięci, dodawaniu obsługi różnych sprzętowych bajerów niekoniecznie związanych z wydajnością (np. ACPI) oraz nowych urządzeń (USB, SCSI), tudzież gruntownym remoncie niektórych istotnych kawałków kodu źródłowego — QEMU, wykorzystywany w KVM, Xen, a nawet VirtualBox, wciąż nie można skompilować za pomocą GCC 4.

Autor tego slajdu osobiście testował działanie KQEMU, instalując system Windows '98 i swoją ulubioną aplikację "użytkową".

Odpowiedzialny stress–testing

Czy to wszystko jest zawsze fajne?

Trudności implementacyjne

Czy to wszystko jest zawsze fajne?

Trudności implementacyjne

- spadek wydajności dla operacji uprzywilejowanych (IO, modyfikacje tablic stron)

Czy to wszystko jest zawsze fajne?

Trudności implementacyjne

- spadek wydajności dla operacji uprzywilejowanych (IO, modyfikacje tablic stron)
- trudności z dobrym planowaniem przydziału czasu przez monitor (np. VMM przydziela czas maszynie wirtualnej, która wykonuje jałową pętlę)

Czy to wszystko jest zawsze fajne?

Trudności implementacyjne

- spadek wydajności dla operacji uprzywilejowanych (IO, modyfikacje tablic stron)
- trudności z dobrym planowaniem przydziału czasu przez monitor (np. VMM przydziela czas maszynie wirtualnej, która wykonuje jałową pętlę)
- wiarygodna emulacja fizycznej architektury utrudniania przez różnorodność BIOS–ów, metod segmentacji czy urządzeń wejścia–wyjścia

Czy to wszystko jest zawsze fajne?

Trudności implementacyjne

- spadek wydajności dla operacji uprzywilejowanych (IO, modyfikacje tablic stron)
- trudności z dobrym planowaniem przydziału czasu przez monitor (np. VMM przydziela czas maszynie wirtualnej, która wykonuje jałową pętlę)
- wiarygodna emulacja fizycznej architektury utrudniania przez różnorodność BIOS–ów, metod segmentacji czy urządzeń wejścia–wyjścia

Większość tych problemów przezwycięża *parawirtualizacja* — upraszcza interfejs do sprzętu, dzięki czemu pisanie VMM staje się prostsze, a wirtualizowane systemy osiągają wyższą wydajność.

Czy to wszystko jest zawsze fajne?

Trudności implementacyjne

- spadek wydajności dla operacji uprzywilejowanych (IO, modyfikacje tablic stron)
- trudności z dobrym planowaniem przydziału czasu przez monitor (np. VMM przydziela czas maszynie wirtualnej, która wykonuje jałową pętlę)
- wiarygodna emulacja fizycznej architektury utrudniania przez różnorodność BIOS–ów, metod segmentacji czy urządzeń wejścia–wyjścia

Większość tych problemów przezwycięża *parawirtualizacja* — upraszcza interfejs do sprzętu, dzięki czemu pisanie VMM staje się prostsze, a wirtualizowane systemy osiągają wyższą wydajność.
Problem: parawirtualizacja wymaga specjalnie zmienionych wersji systemu operacyjnego, który wirtualizujemy.

Czy to wszystko jest zawsze fajne?

Wady i niebezpieczeństwa

Czy to wszystko jest zawsze fajne?

Wady i niebezpieczeństwa

- awaria jednego komputera pociąga wyłączenie wielu systemów

Czy to wszystko jest zawsze fajne?

Wady i niebezpieczeństwa

- awaria jednego komputera pociąga wyłączenie wielu systemów
- potrzeba wykwalifikowanego personelu do rozpoznawania przyczyn błędów na wirtualizowanych systemach

Czy to wszystko jest zawsze fajne?

Wady i niebezpieczeństwa

- awaria jednego komputera pociąga wyłączenie wielu systemów
- potrzeba wykwalifikowanego personelu do rozpoznawania przyczyn błędów na wirtualizowanych systemach

Ale przede wszystkim **nowe techniki crackingu**

Czy to wszystko jest zawsze fajne?

Wady i niebezpieczeństwa

- awaria jednego komputera pociąga wyłączenie wielu systemów
- potrzeba wykwalifikowanego personelu do rozpoznawania przyczyn błędów na wirtualizowanych systemach

Ale przede wszystkim **nowe techniki crackingu**

- Vitriol, korzystający z rozszerzeń Intel VT-x
- Blue Pill, korzystający z rozszerzeń AMD Pacifica

Witamy w Matriksie

W roku 2006 polska specjalistka w dziedzinie bezpieczeństwa komputerowego, Joanna Rutkowska, przedstawiła sposób na załadowanie niepodpisanego kodu do kernela systemu Windows Vista. Metodę tę wykorzystano do podania systemowi "niebieskiej pigułki".

Witamy w Matriksie

W roku 2006 polska specjalistka w dziedzinie bezpieczeństwa komputerowego, Joanna Rutkowska, przedstawiła sposób na załadowanie niepodpisanego kodu do kernela systemu Windows Vista. Metodę tę wykorzystano do podania systemowi "niebieskiej pigułki".

Blue Pill to nazwa rootkita, który w dyskretny sposób przejmuje kontrolę nad systemem operacyjnym jako nadzorca. System–ofiara jest wirtualizowany przez rootkit.

Witamy w Matriksie

W roku 2006 polska specjalistka w dziedzinie bezpieczeństwa komputerowego, Joanna Rutkowska, przedstawiła sposób na załadowanie niepodpisanego kodu do kernela systemu Windows Vista. Metodę tę wykorzystano do podania systemowi "niebieskiej pigułki".

Blue Pill to nazwa rootkita, który w dyskretny sposób przejmuje kontrolę nad systemem operacyjnym jako nadzorca. System–ofiara jest wirtualizowany przez rootkit.

Z założenia wirtualizacja uniemożliwia wirtualizowanemu systemowi wykrycie, że jest wirtualizowany. Oznacza to, że niebieska pigułka może być całkowicie niewykrywalna.

Witamy w Matriksie

W roku 2006 polska specjalistka w dziedzinie bezpieczeństwa komputerowego, Joanna Rutkowska, przedstawiła sposób na załadowanie niepodpisanego kodu do kernela systemu Windows Vista. Metodę tę wykorzystano do podania systemowi "niebieskiej pigułki".

Blue Pill to nazwa rootkita, który w dyskretny sposób przejmuje kontrolę nad systemem operacyjnym jako nadzorca. System–ofiara jest wirtualizowany przez rootkit.

Z założenia wirtualizacja uniemożliwia wirtualizowanemu systemowi wykrycie, że jest wirtualizowany. Oznacza to, że niebieska pigułka może być całkowicie niewykrywalna.

Oryginalna wersja Blue Pill korzysta z rozszerzeń AMD Pacifica i celuje w system Windows Vista, ale możliwe jest wykorzystanie także rozszerzeń Intel Vanderpool i atakowanie innych systemów operacyjnych.

Matrix: Rewolucje?

Skuteczność działania pigułki jest kwestionowana.

Matrix: Rewolucje?

Skuteczność działania pigułki jest kwestionowana.

- Tom Yager: *Blue Pill is an attention-whoring non-threat, period*

Matrix: Rewolucje?

Skuteczność działania pigułki jest kwestionowana.

- Tom Yager: *Blue Pill is an attention-whoring non-threat, period*
- Anthony Liguori, developer Xen: *I assume most vendors provide a BIOS setting to enable or disable VT/SVM. If a problem were found, vendors could simply disable the extension until AMD or Intel fixed the problem.*

Matrix: Rewolucje?

Najczęściej sugerowaną metodą przejrzenia przez Matrix jest *timing attack*. Jego skuteczność wynika z konieczności powolnego emulowania uprzywilejowanych operacji.

Matrix: Rewolucje?

Najczęściej sugerowaną metodą przejrzenia przez Matrix jest *timing attack*. Jego skuteczność wynika z konieczności powolnego emulowania uprzywilejowanych operacji.

Ale czemu właściwie Matrix miałby pozwalać wirtualizowanemu systemowi na odczytywanie prawdziwych informacji dotyczących czasu?

Matrix: Rewolucje?

Najczęściej sugerowaną metodą przejrzenia przez Matrix jest *timing attack*. Jego skuteczność wynika z konieczności powolnego emulowania uprzywilejowanych operacji.

Ale czemu właściwie Matrix miałby pozwalać wirtualizowanemu systemowi na odczytywanie prawdziwych informacji dotyczących czasu?

Potrzebne jest wiarygodne, **zewnętrzne** źródło informacji, np. serwery NTP.

Matrix: Rewolucje?

Najczęściej sugerowaną metodą przejrzenia przez Matrix jest *timing attack*. Jego skuteczność wynika z konieczności powolnego emulowania uprzywilejowanych operacji.

Ale czemu właściwie Matrix miałby pozwalać wirtualizowanemu systemowi na odczytywanie prawdziwych informacji dotyczących czasu?

Potrzebne jest wiarygodne, **zewnętrzne** źródło informacji, np. serwery NTP.

Ale ponownie — dlaczego Matrix pozwalać systemom na korzystanie z tego typu usług lub nie fałszować nadchodzących informacji?