Wirtualizacja

Przegląd wybranych technik

Magda Michalska Krzysztof Kulewski Andrzej Pacuk

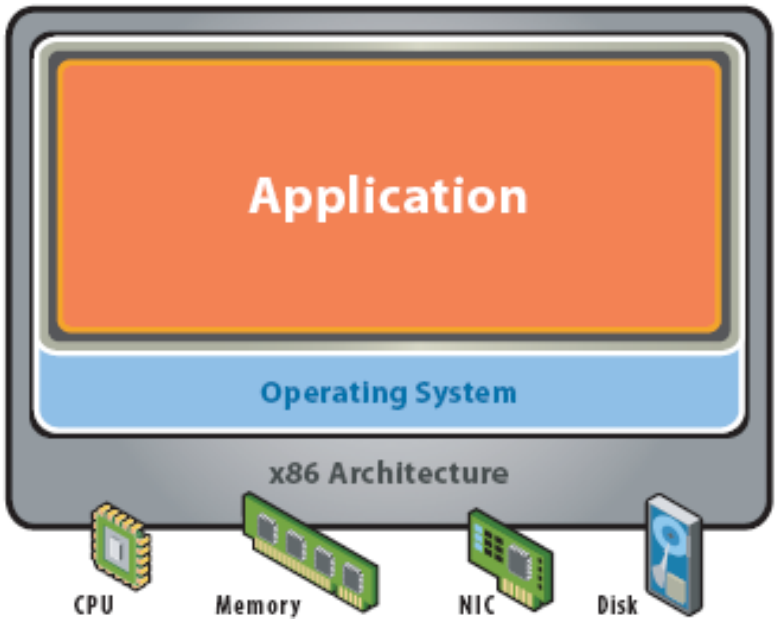
Systemy operacyjne 2006

Plan

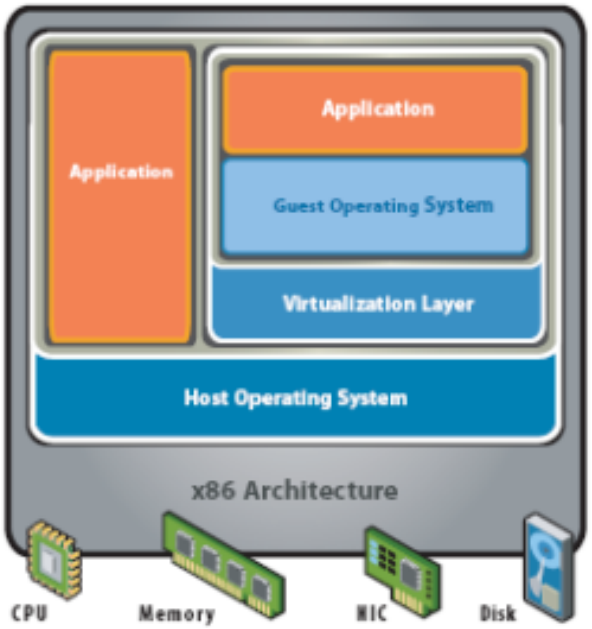
Plan

Czym jest wirtualizacja?

- Wirtualizacja to bogaty zestaw metod umożliwiających podział zasobów komputera miedzy wiele środowisk.



przed



po

Stosowane techniki

- partycjonowanie hardwarowe
- partycjonowanie softwarowe
- podział czasu
- maszyny wirtualne

Stosowane techniki

- partycjonowanie hardwarowe
- partycjonowanie softwarowe
- podział czasu
- maszyny wirtualne

Stosowane techniki

- partycjonowanie hardwarowe
- partycjonowanie softwarowe
- podział czasu
- maszyny wirtualne

Stosowane techniki

- partycjonowanie hardwarowe
- partycjonowanie softwarowe
- podział czasu
- maszyny wirtualne

Przyczyny i zalety

- server consolidation
  (średnie wykorzystanie zasobów nieskonsolidowanego serwera to 5-10%)
- emulacja innych, często niedostępnych architektur lub systemów operacyjnych
- używanie kilku systemów operacyjnych jednocześnie
- testy i eksperymenty bez ryzyka
- szybkie wykonywanie i odtwarzanie kopii zapasowych
- symulacja dużych środowisk
- bezpieczne środowisko do uruchamiania podejrzanych aplikacji
- tworzenie nowych systemów operacyjnych
- wieksza przenonośność oprogramowania

Przyczyny i zalety

- server consolidation
  (średnie wykorzystanie zasobów nieskonsolidowanego serwera to 5-10%)
- emulacja innych, często niedostępnych architektur lub systemów operacyjnych
- używanie kilku systemów operacyjnych jednocześnie
- testy i eksperymenty bez ryzyka
- szybkie wykonywanie i odtwarzanie kopii zapasowych
- symulacja dużych środowisk
- bezpieczne środowisko do uruchamiania podejrzanych aplikacji
- tworzenie nowych systemów operacyjnych
- wieksza przenonośność oprogramowania

Przyczyny i zalety

- server consolidation
  (średnie wykorzystanie zasobów nieskonsolidowanego serwera to 5-10%)
- emulacja innych, często niedostępnych architektur lub systemów operacyjnych
- używanie kilku systemów operacyjnych jednocześnie
- testy i eksperymenty bez ryzyka
- szybkie wykonywanie i odtwarzanie kopii zapasowych
- symulacja dużych środowisk
- bezpieczne środowisko do uruchamiania podejrzanych aplikacji
- tworzenie nowych systemów operacyjnych
- wieksza przenonośność oprogramowania

Przyczyny i zalety

- server consolidation
  (średnie wykorzystanie zasobów nieskonsolidowanego serwera to 5-10%)
- emulacja innych, często niedostępnych architektur lub systemów operacyjnych
- używanie kilku systemów operacyjnych jednocześnie
- testy i eksperymenty bez ryzyka
- szybkie wykonywanie i odtwarzanie kopii zapasowych
- symulacja dużych środowisk
- bezpieczne środowisko do uruchamiania podejrzanych aplikacji
- tworzenie nowych systemów operacyjnych
- wieksza przenonośność oprogramowania

Przyczyny i zalety

- server consolidation
  (średnie wykorzystanie zasobów nieskonsolidowanego serwera to 5-10%)
- emulacja innych, często niedostępnych architektur lub systemów operacyjnych
- używanie kilku systemów operacyjnych jednocześnie
- testy i eksperymenty bez ryzyka
- szybkie wykonywanie i odtwarzanie kopii zapasowych
- symulacja dużych środowisk
- bezpieczne środowisko do uruchamiania podejrzanych aplikacji
- tworzenie nowych systemów operacyjnych
- wieksza przenonośność oprogramowania

Przyczyny i zalety

- server consolidation
  (średnie wykorzystanie zasobów nieskonsolidowanego serwera to 5-10%)
- emulacja innych, często niedostępnych architektur lub systemów operacyjnych
- używanie kilku systemów operacyjnych jednocześnie
- testy i eksperymenty bez ryzyka
- szybkie wykonywanie i odtwarzanie kopii zapasowych
- symulacja dużych środowisk
- bezpieczne środowisko do uruchamiania podejrzanych aplikacji
- tworzenie nowych systemów operacyjnych
- wieksza przenonośność oprogramowania

Przyczyny i zalety

- server consolidation
  (średnie wykorzystanie zasobów nieskonsolidowanego serwera to 5-10%)
- emulacja innych, często niedostępnych architektur lub systemów operacyjnych
- używanie kilku systemów operacyjnych jednocześnie
- testy i eksperymenty bez ryzyka
- szybkie wykonywanie i odtwarzanie kopii zapasowych
- symulacja dużych środowisk
- bezpieczne środowisko do uruchamiania podejrzanych aplikacji
- tworzenie nowych systemów operacyjnych
- wieksza przenonośność oprogramowania

Przyczyny i zalety

- server consolidation
  (średnie wykorzystanie zasobów nieskonsolidowanego serwera to 5-10%)
- emulacja innych, często niedostępnych architektur lub systemów operacyjnych
- używanie kilku systemów operacyjnych jednocześnie
- testy i eksperymenty bez ryzyka
- szybkie wykonywanie i odtwarzanie kopii zapasowych
- symulacja dużych środowisk
- bezpieczne środowisko do uruchamiania podejrzanych aplikacji
- tworzenie nowych systemów operacyjnych
- wieksza przenonośność oprogramowania

Przyczyny i zalety

- server consolidation
  (średnie wykorzystanie zasobów nieskonsolidowanego serwera to 5-10%)
- emulacja innych, często niedostępnych architektur lub systemów operacyjnych
- używanie kilku systemów operacyjnych jednocześnie
- testy i eksperymenty bez ryzyka
- szybkie wykonywanie i odtwarzanie kopii zapasowych
- symulacja dużych środowisk
- bezpieczne środowisko do uruchamiania podejrzanych aplikacji
- tworzenie nowych systemów operacyjnych
- wieksza przenonośność oprogramowania

Przyczyny i zalety

- server consolidation
  (średnie wykorzystanie zasobów nieskonsolidowanego serwera to 5-10%)
- emulacja innych, często niedostępnych architektur lub systemów operacyjnych
- używanie kilku systemów operacyjnych jednocześnie
- testy i eksperymenty bez ryzyka
- szybkie wykonywanie i odtwarzanie kopii zapasowych
- symulacja dużych środowisk
- bezpieczne środowisko do uruchamiania podejrzanych aplikacji
- tworzenie nowych systemów operacyjnych
- wieksza przenonośność oprogramowania

Wady

- prękość działania
  (mogą wystapić duże narzuty czasowe)
- trudność w implementacji architektur, które nie wspierają wirtualizacji
- brak idealnego rozwiązania

Wady

- prękość działania
  (mogą wystapić duże narzuty czasowe)
- trudność w implementacji architektur, które nie wspierają wirtualizacji
- brak idealnego rozwiązania

Wady

- prękość działania
  (mogą wystapić duże narzuty czasowe)
- trudność w implementacji architektur, które nie wspierają wirtualizacji
- brak idealnego rozwiązania

Plan

Charakterystyka

- Wirtualna maszyna symuluje kompletną architekturę
  (często zupłenie inną od macierzystej)
- Wszystkie urządzenia we/wy, dyski muszą być współdzielone z systemem gospodarza
- Instrukcje systemu gościa są tłumaczone często na wiele instrukcji gospodarza
  (wiaże się to z dużym narzutem czasowym)

Charakterystyka

- Wirtualna maszyna symuluje kompletną architekturę
  (często zupłenie inną od macierzystej)
- Wszystkie urządzenia we/wy, dyski muszą być współdzielone z systemem gospodarza
- Instrukcje systemu gościa są tłumaczone często na wiele instrukcji gospodarza
  (wiaże się to z dużym narzutem czasowym)

Charakterystyka

- Wirtualna maszyna symuluje kompletną architekturę
  (często zupłenie inną od macierzystej)
- Wszystkie urządzenia we/wy, dyski muszą być współdzielone z systemem gospodarza
- Instrukcje systemu gościa są tłumaczone często na wiele instrukcji gospodarza
  (wiaże się to z dużym narzutem czasowym)

Przykłady

- Bochs,
- PearPC,
- Virtual PC wersja na PPC,
- Qemu,
- emulatory ATARI, Amiga itp.

Przykłady

- Bochs,
- PearPC,
- Virtual PC wersja na PPC,
- Qemu,
- emulatory ATARI, Amiga itp.

Przykłady

- Bochs,
- PearPC,
- Virtual PC wersja na PPC,
- Qemu,
- emulatory ATARI, Amiga itp.

Przykłady

- Bochs,
- PearPC,
- Virtual PC wersja na PPC,
- Qemu,
- emulatory ATARI, Amiga itp.

Przykłady

- Bochs,
- PearPC,
- Virtual PC wersja na PPC,
- Qemu,
- emulatory ATARI, Amiga itp.

Charakterystyka

- Do odpalenia programu nie trzeba symulowac calego OS, wystarczy emulacja środowiska wykonawczego aplikacji.
- Nie instalujemy OS, więc nie musimy kupować licencji!
- Rozwiązanie szybsze od czystej emulacji

Charakterystyka

- Do odpalenia programu nie trzeba symulowac calego OS, wystarczy emulacja środowiska wykonawczego aplikacji.
- Nie instalujemy OS, więc nie musimy kupować licencji!
- Rozwiązanie szybsze od czystej emulacji

Charakterystyka

- Do odpalenia programu nie trzeba symulowac calego OS, wystarczy emulacja środowiska wykonawczego aplikacji.
- Nie instalujemy OS, więc nie musimy kupować licencji!
- Rozwiązanie szybsze od czystej emulacji

Przykłady

- DOSEMU,
- Wine,

Przykłady

- DOSEMU,
- Wine,

Charakterystyka

- System guesta musi obsługiwać tą samą architekture co host. *(ale ograniczenie do x86 nie jest aż tak bolesne!)*
- Emulacja prawdziwego (VMware) lub wirtualnego (VPC) sprzętu.
- Gość nie musi wiedzieć, że jest uruchamiany na nierzeczywistym systemie
  (ale czasem to pomaga -> VMware Tools)
- Bezpośrednio guest nie ma dostępu do żadnych zasobów sprzętowych
- Rozwiązanie najprostsze w obsłudze przez użytkownika

Charakterystyka

- System guesta musi obsługiwać tą samą architekture co host. *(ale ograniczenie do x86 nie jest aż tak bolesne!)*
- Emulacja prawdziwego (VMware) lub wirtualnego (VPC) sprzętu.
- Gość nie musi wiedzieć, że jest uruchamiany na nierzeczywistym systemie
  (ale czasem to pomaga -> VMware Tools)
- Bezpośrednio guest nie ma dostępu do żadnych zasobów sprzętowych
- Rozwiązanie najprostsze w obsłudze przez użytkownika

Charakterystyka

- System guesta musi obsługiwać tą samą architekture co host. *(ale ograniczenie do x86 nie jest aż tak bolesne!)*
- Emulacja prawdziwego (VMware) lub wirtualnego (VPC) sprzętu.
- Gość nie musi wiedzieć, że jest uruchamiany na nierzeczywistym systemie
  (ale czasem to pomaga -> VMware Tools)
- Bezpośrednio guest nie ma dostępu do żadnych zasobów sprzętowych
- Rozwiązanie najprostsze w obsłudze przez użytkownika

Charakterystyka

- System guesta musi obsługiwać tą samą architekture co host. *(ale ograniczenie do x86 nie jest aż tak bolesne!)*
- Emulacja prawdziwego (VMware) lub wirtualnego (VPC) sprzętu.
- Gość nie musi wiedzieć, że jest uruchamiany na nierzeczywistym systemie
  (ale czasem to pomaga -> VMware Tools)
- Bezpośrednio guest nie ma dostępu do żadnych zasobów sprzętowych
- Rozwiązanie najprostsze w obsłudze przez użytkownika

Charakterystyka

- System guesta musi obsługiwać tą samą architekture co host. *(ale ograniczenie do x86 nie jest aż tak bolesne!)*
- Emulacja prawdziwego (VMware) lub wirtualnego (VPC) sprzętu.
- Gość nie musi wiedzieć, że jest uruchamiany na nierzeczywistym systemie
  (ale czasem to pomaga -> VMware Tools)
- Bezpośrednio guest nie ma dostępu do żadnych zasobów sprzętowych
- Rozwiązanie najprostsze w obsłudze przez użytkownika

Przykłady

- VMWare Workstation,
- Virtual PC,
- CoLinux,
- Parallels Desktop,

Przykłady

- VMWare Workstation,
- Virtual PC,
- CoLinux,
- Parallels Desktop,

Przykłady

- VMWare Workstation,
- Virtual PC,
- CoLinux,
- Parallels Desktop,

Przykłady

- VMWare Workstation,
- Virtual PC,
- CoLinux,
- Parallels Desktop,

Charakterystyka

- Gość musi wiedzieć, że nie jest uruchamiany na systemie rzeczywistym, jego kod musi być przeportowany.
- Dzięki bibliotekom API maszyna wirtualna może korzystać z zasobów rzeczywistych.
- Konieczność modyfikacji jądra systemu ogranicza zbiór guestów do systemów Open Source

Charakterystyka

- Gość musi wiedzieć, że nie jest uruchamiany na systemie rzeczywistym, jego kod musi być przeportowany.
- Dzięki bibliotekom API maszyna wirtualna może korzystać z zasobów rzeczywistych.
- Konieczność modyfikacji jądra systemu ogranicza zbiór guestów do systemów Open Source

Charakterystyka

- Gość musi wiedzieć, że nie jest uruchamiany na systemie rzeczywistym, jego kod musi być przeportowany.
- Dzięki bibliotekom API maszyna wirtualna może korzystać z zasobów rzeczywistych.
- Konieczność modyfikacji jądra systemu ogranicza zbiór guestów do systemów Open Source

Przykłady

- XEN,

Charakterystyka

- Umożliwia uruchamianie kilku kopii systemu identycznego (ten sam kernel) jak host, ale mogą to byc np. różne dystrybucje z tym samym kernelem.
- Uzyskuje się przez podział zasobów hosta na osobne partycje, dzięki czemu dostajemy izolację maszyn wirtualnych.
- Najszybsze, zaledwie 1-3% (w przypadku OpenVZ) wolniejsze niż natywny system hosta

Charakterystyka

- Umożliwia uruchamianie kilku kopii systemu identycznego (ten sam kernel) jak host, ale mogą to byc np. różne dystrybucje z tym samym kernelem.
- Uzyskuje się przez podział zasobów hosta na osobne partycje, dzięki czemu dostajemy izolację maszyn wirtualnych.
- Najszybsze, zaledwie 1-3% (w przypadku OpenVZ) wolniejsze niż natywny system hosta

Charakterystyka

- Umożliwia uruchamianie kilku kopii systemu identycznego (ten sam kernel) jak host, ale mogą to byc np. różne dystrybucje z tym samym kernelem.
- Uzyskuje się przez podział zasobów hosta na osobne partycje, dzięki czemu dostajemy izolację maszyn wirtualnych.
- Najszybsze, zaledwie 1-3% (w przypadku OpenVZ) wolniejsze niż natywny system hosta

Przykłady

- UML,
- OpenVZ,
- Linux-VServer,
- VMware ESX,

Przykłady

- UML,
- OpenVZ,
- Linux-VServer,
- VMware ESX,

Przykłady

- UML,
- OpenVZ,
- Linux-VServer,
- VMware ESX,

Przykłady

- UML,
- OpenVZ,
- Linux-VServer,
- VMware ESX,

Plan

Polecamy

- VMware Server *www.vmware.com*
- Qemu *www.qemu.com*
- UML *www.usermodelinux.org*

Plan

Bibliografia

- strona Wikipedii *http://en.wikipedia.org*
- strona VMware *http://www.vmware.com*
- strona Qemu *http://fabrice.bellard.free.fr/qemu/*
- strona UML *http://www.usermodelinux.org*
- strona z dużą ilością teorii *http://www.kernelthread.com/publications/virtualization/*