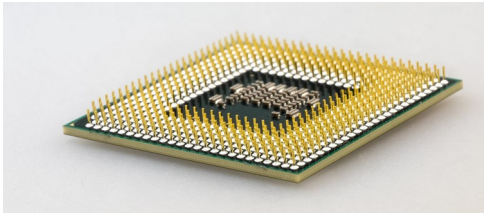
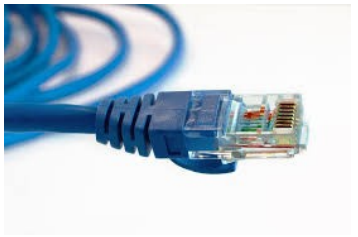
# W2

## CHMURY OBLICZENIOWE
## – podstawowe technologie: WIRTUALIZACJA

# W2 Przyczyny powstania i rozwoju wirtualizacji

Do opisania działania systemów obliczeniowych potrzebne są trzy podstawowe abstrakcje:





(1) interpretator / procesor (2) pamięć / pamięć masowa (3) łącza komunikacyjne

*Zagadnienia dotyczące zarządzania zasobami: :*
*- rezerwa na szczytowe zapotrzebowanie oraz tzw. overprovisioning*
*- heterogeniczność sprzętu i oprogramowania*
*- awarie sprzętu i metody reakcji na awarie*

**Wraz ze wzrostem skali systemu i wielkości użytkowników, zarządzanie jego zasobami staje się bardzo trudne**

# W2 Pojęcie wirtualizacji

*Na podstawie  Wikipedi*

Virtualization, in computing, refers to the act of creating a virtual (rather than actual) version of something, including but not limited to a virtual computer hardware platform, operating system (OS), storage device, or computer network resources.”

*Typowe “spojrzenie inżynierskie”*

Wirtualizacja wyodrębnia podstawowe zasoby; upraszcza ich użycie; izoluje użytkowników od siebie; i wspiera replikację, co zwiększa elastyczność systemu

# W2 Wirtualizacja a chmury obliczeniowe

Wirtualizacja jest podstawowym elementem Cloud Computing, który pozwala na uproszczenie zarządzania zasobami fizycznymi dla trzech abstrakcje, przedstawionych na pierwszym slajdzie

- przykład 1: stan maszyny wirtualnej (VM) działającej pod kontrolą monitora maszyny wirtualnej (VMM) można w prosty sposób zapisać i przenieść na inny serwer, aby zrównoważyć obciążenie,

- przykład 2: wirtualizacja pozwala użytkownikom działać w środowiskach, z którymi są zaznajomieni, zamiast zmuszać ich do konkretnych rozwiązań sprzętowo-programowych.

**Wirtualizacja zasobów w chmurze jest ważna ze względu na:**

- Izolację wydajności ponieważ umożliwia dynamicznie przydzielanie i rozliczanie zasobów dla różnych użytkowników i/lub aplikacji,
- bezpieczeństwo systemu ponieważ pozwala na izolację usług działających na tym samym sprzęcie
- wydajność i niezawodność ponieważ umożliwia migrację aplikacji z jednej platformy na drugą

# Wirtualizacja – podejścia do jej realizacji

Wirtualizacja symuluje określoną formę interfejs do obiektu fizycznego poprzez:

**Multipleksowanie:** tworzy wiele wirtualnych obiektów z jednej instancji obiektu fizycznego. Wiele wirtualnych obiektów w jednym fizycznym. Przykład - a procesor jest multipleksowany między wieloma procesami lub wątkami.

**Agregację:** tworzy jeden wirtualny obiekt z wielu obiektów fizycznych. Jeden wirtualny obiekt na wiele obiektów fizycznych. Przykładowo - pewna liczba dysków fizycznych jest agregowana w dysk RAID.

**Emulacja:** konstruuje wirtualny obiekt określonego typu z innego typu obiektu fizycznego. Przykładowo - dysk fizyczny emuluje pamięć o dostępie swobodnym (RAM).

**Multipleksowanie i emulację.** przykładowo - pamięć wirtualna ze stronicowaniem multipleksuje pamięć rzeczywistą i dysk; wirtualny adres emuluje prawdziwy adres.

# W2 Wrzut oka na historię wirtualizacji

**1960: IBM: program sterujący CP/CMS:** działająca maszyna wirtualna dla IBM System/360 Model 67

……

**2000,** IBM: seria z z 64-bitowymi wirtualnymi przestrzeniami adresowymi i wstecz kompatybilny z Systemem/360

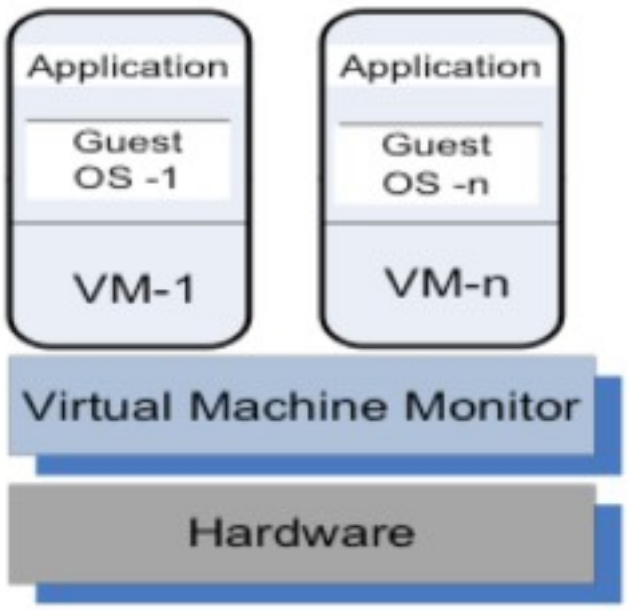
**1974: Popek and Golberg** z UCLA opublikowali pracę “Formal Requirements for Virtualizable Third Generation Architectures” gdzie wymienili warunki, jakie architektura komputerowa powinna spełniać, aby wydajnie wspierać wirtualizację. Popularna architektura x86, która powstała w latach 70, nie nie obsługiwała tych wymagań przez kolejne wiele lat.

**1990’s, badacze z uniwersytetu w Stanford, VMware:** Badacze ci opracowali nowy hiperwizor i założenie VMware, największej firmy zajmującej się wirtualizacją do dnia dzisiejszego. Opracowali pierwsze rozwiązanie do wirtualizacji x86 w 1999.

Obecnie wiele rozwiązań: Xen z Cambridge, Linux KVM, Microsoft Hyper-V,

# W2 Virtual Machine Monitor (VMM / Hypervisor) (1)

Monitor maszyny wirtualnej (VMM / hypervisor) jest odpowiedzialny za:




- przydział zasobów systemu komputerowego na jedną lub więcej maszyn wirtualnych (VM).

Tym samym umożliwia jednoczesne działanie kilku systemów operacyjnych na jednej platformie sprzętowej.

**Maszyna wirtualna (VM)** to środowisko wykonawcze, w którym działa system operacyjny VM – izolowane środowisko, które wydaje się być całym komputerem, ale w rzeczywistości ma dostęp tylko do części zasobów komputera

# W2 Virtual Machine Monitor (VMM / Hypervisor) (2)

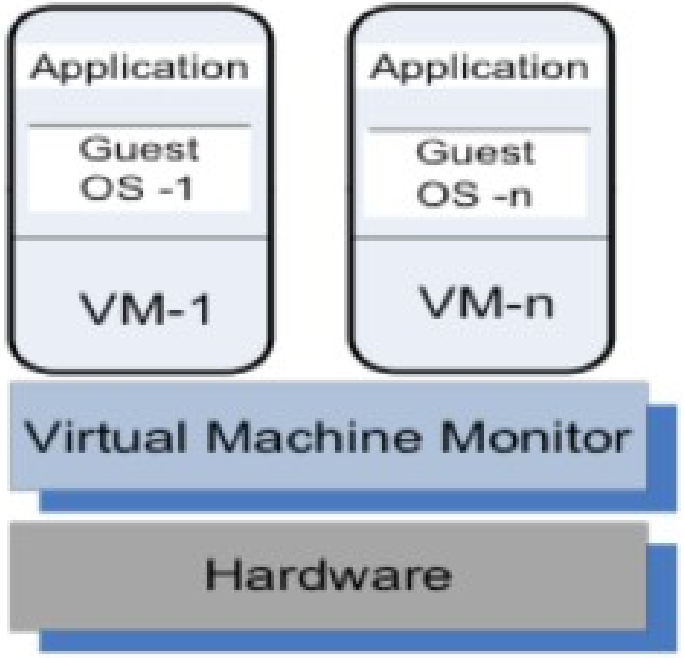
W większości praktycznych rozwiązań VMM pozwala na:

- udostępnianie wielu usługom tej samej platformy,

- migracjaęna żywo – przenoszenie serwera z jednej platformy na drugą,

- modyfikację systemu przy zachowaniu wsteczna kompatybilność z oryginalnym systemem,

- nadzorowanie I wymuszanie oczekiwanego poziomu izolacji pomiędzy systemami, a tym samym nadzorowanie bezpieczeństwa usług.




**System operacyjny gościa** to system operacyjny działający na maszynie wirtualnej pod kontrolą VMM.

# W2 Szablon działania hypervisor-a (obowiązki VMM) i

- Przechwytuje uprzywilejowane instrukcje wykonywane przez system gościa i wymusza poprawność i bezpieczeństwo operacji.

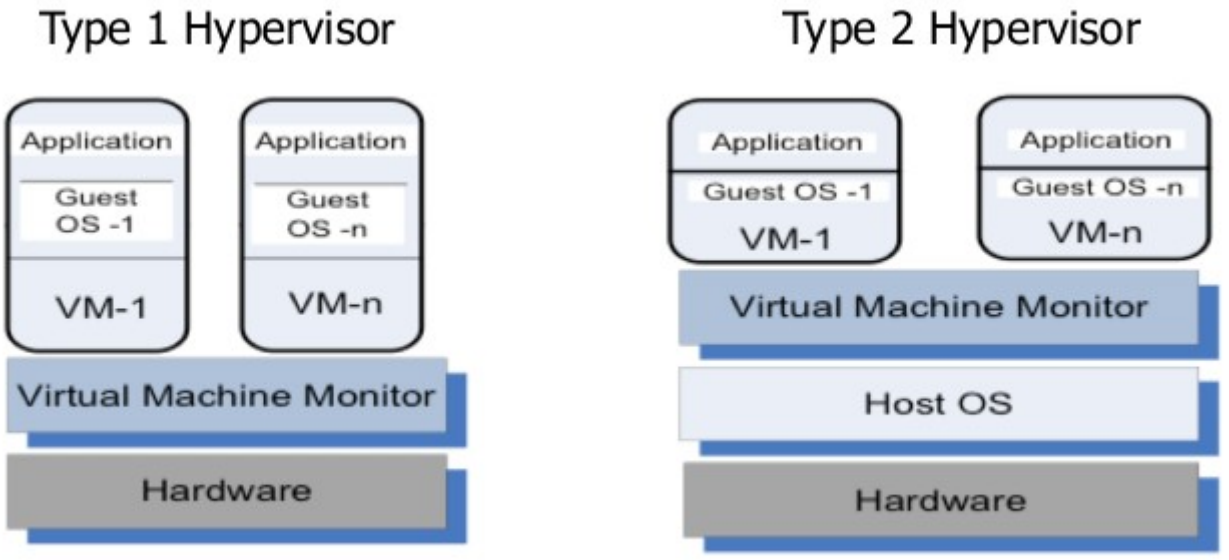
- Definiuje pułapki. Które przerywają i wysyłają te instrukcje  do poszczególnych systemów operacyjnych gości.

- Kontroluje zarządzanie pamięcią wirtualną.

- Utrzymuje tabelę stron (ang. shadow page) dla każdego systemu operacyjnego gościa i replikuje wszelkie modyfikacje wprowadzone przez system operacyjny gościa we własnej tabeli stron. Ta tabela stron wskazuje na rzeczywistą ramkę strony i jest używana przez jednostkę zarządzania pamięcią (MMU) do dynamicznej translacji adresów.

- Monitoruje wydajność systemu i podejmuje działania naprawcze, aby uniknąć obniżenia wydajności. Na przykład VMM może zamienić maszynę wirtualną, aby uniknąć thrashingu.

# W2 Taksonomia VMM. Hypervisor-y typu 1 oraz 2 i




1. Hypervisor typu 1 (bare metal, natywny): obsługuje wiele maszyn wirtualnych i działa bezpośrednio na sprzęcie (np. VMware ESX , Xen)

2. Hypervisor typu 2 (tzw. hosted) — działa w systemie operacyjnym hosta (np. Linux w trybie użytkownika)

# W2

## Przykładowe VMM

| Name | Host ISA | Guest ISA | Host OS | guest OS | Company |
|---|---|---|---|---|---|
| Integrity VM | *x86*-64 | *x86*-64 | HP-Unix | Linux,Windows HP Unix | HP |
| Power VM | Power | Power | No host OS | Linux, AIX | IBM |
| z/VM | z-ISA | z-ISA | No host OS | Linux on z-ISA | IBM |
| Lynx Secure | *x86* | *x86* | No host OS | Linux, Windows | LinuxWorks |
| Hyper-V Server | *x86*-64 | *x86*-64 | Windows | Windows | Microsoft |
| Oracle VM | *x86*, *x86*-64 | *x86*, *x86*-64 | No host OS | Linux, Windows | Oracle |
| RTS Hypervisor | *x86* | *x86* | No host OS | Linux, Windows | Real Time Systems |
| SUN xVM | *x86*, SPARC | same as host | No host OS | Linux, Windows | SUN |
| VMware EX Server | *x86*, *x86*-64 | *x86*, *x86*-64 | No host OS | Linux, Windows Solaris, FreeBSD | VMware |
| VMware Fusion | *x86*, *x86*-64 | *x86*, *x86*-64 | MAC OS *x86* | Linux, Windows Solaris, FreeBSD | VMware |
| VMware Server | *x86*, *x86*-64 | *x86*, *x86*-64 | Linux, Windows | Linux, Windows Solaris, FreeBSD | VMware |
| VMware Workstation | *x86*, *x86*-64 | *x86*, *x86*-64 | Linux, Windows | Linux, Windows Solaris, FreeBSD | VMware |
| VMware Player | *x86*, *x86*-64 | *x86*, *x86*-64 | Linux Windows | Linux, Windows Solaris, FreeBSD | VMware |
| Denali | *x86* | *x86* | Denali | ILVACO, NetBSD | University of Washington |
| Xen | *x86*, *x86*-64 | *x86*, *x86*-64 | Linux Solaris | Linux, Solaris NetBSD | University of Cambridge |

# W2 Warunki konieczne efektywnej wirtualizacji

(Popek i Goldberg)

1. Program działający pod VMM powinien zasadniczo wykazywać zachowanie
identyczne z zachowaniem podczas pracy bezpośrednio na równoważnej maszynie.

2. VMM powinien mieć pełną kontrolę nad zwirtualizowanymi zasobami.

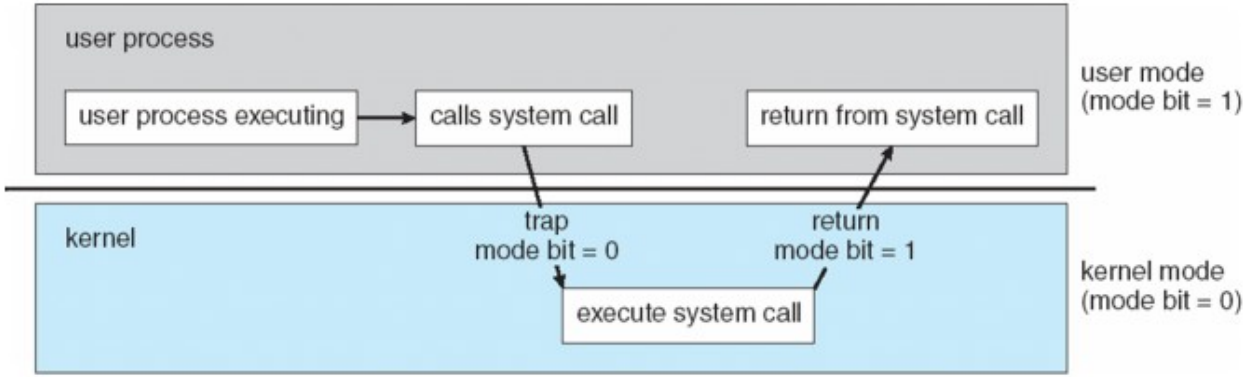
3. Należy wykonać statystycznie istotną część instrukcji maszynowych bez interwencji VMM.

# W2 Warunki konieczne efektywnej wirtualizacji - praktyka

(Dual-Mode Operations)

Praca w dwóch trybach pozwala systemowi operacyjnemu chronić siebie i inne komponenty systemu poprzez następujące mechanizmy:

- Tryb użytkownika i tryb jądra
- Bit trybu zapewniany przez sprzęt
- Możliwość rozróżnienia, kiedy system wykonuje kod w trybie użytkownika a kiedy jądra
- Niektóre instrukcje są uprzywilejowane, można je wykonywać tylko w trybie jądra
- Wywołanie systemowe (ang. system call) zmienia tryb na tryb jądra a powrót resetuje go do trybu użytkownika



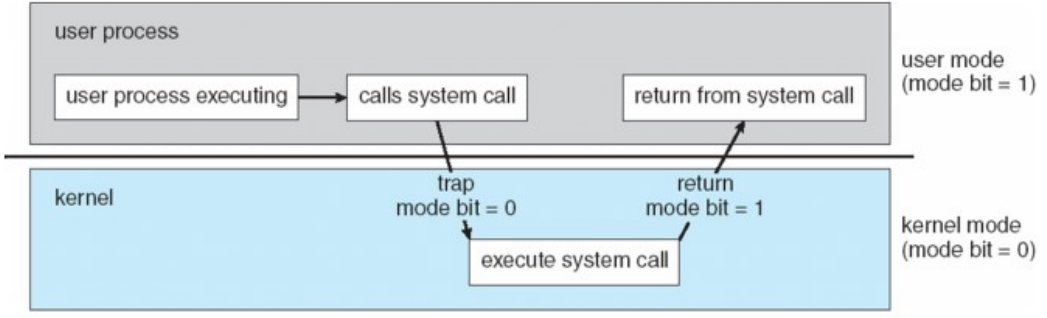
# W2 Tryb użytkownika vs tryb jądra i

Kod jądra (w szczególności obsługi przerwań) działa w trybie jądra (ang. Kernel mode)

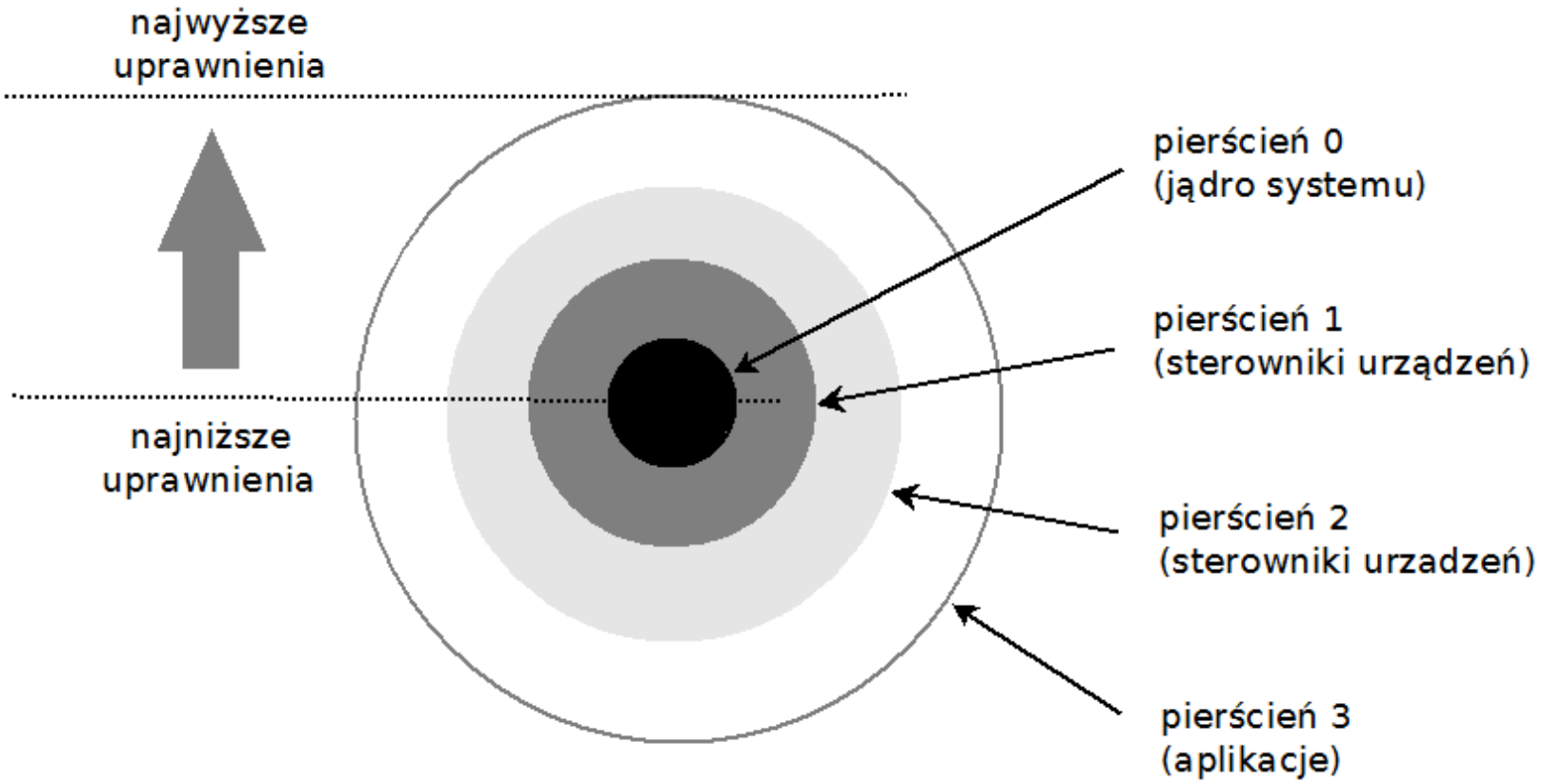
Sprzęt pozwala na wykonanie wszystkich instrukcji maszynowych i umożliwia nieograniczony dostęp do pamięci i portów I/O

Cała reszta działa w trybie użytkownika (ang. User mode)

System operacyjny bardzo mocno opiera się na tym wymuszonym sprzętowo mechanizmie ochrony



# W2 Model pierścieniowy



# W2 Typowa implementacja OS na sprzęcie


Trzy klasy instrukcji maszynowych I relacje pomiędzy innymi:

1. Uprzywilejowane instrukcje mogą być wykonywane w trybie jądra. Kiedy próbować będą być wykonywane w trybie użytkownika, uaktywnią pułapkę oraz i tak wykonywane zostaną w trybie jądra.

2. Instrukcje nieuprzywilejowane to te, które można wykonać w trybie użytkownika

3. Wrażliwe instrukcje mogą być wykonywane w trybie jądra lub użytkownika, ale wynikowe zachowywanie zależy od tego wyboru. Poufne instrukcje wymagają specjalnych środków ostrożności w czasie wykonywania.

4. Wrażliwe i nieuprzywilejowane instrukcje są trudne do wirtualizacji

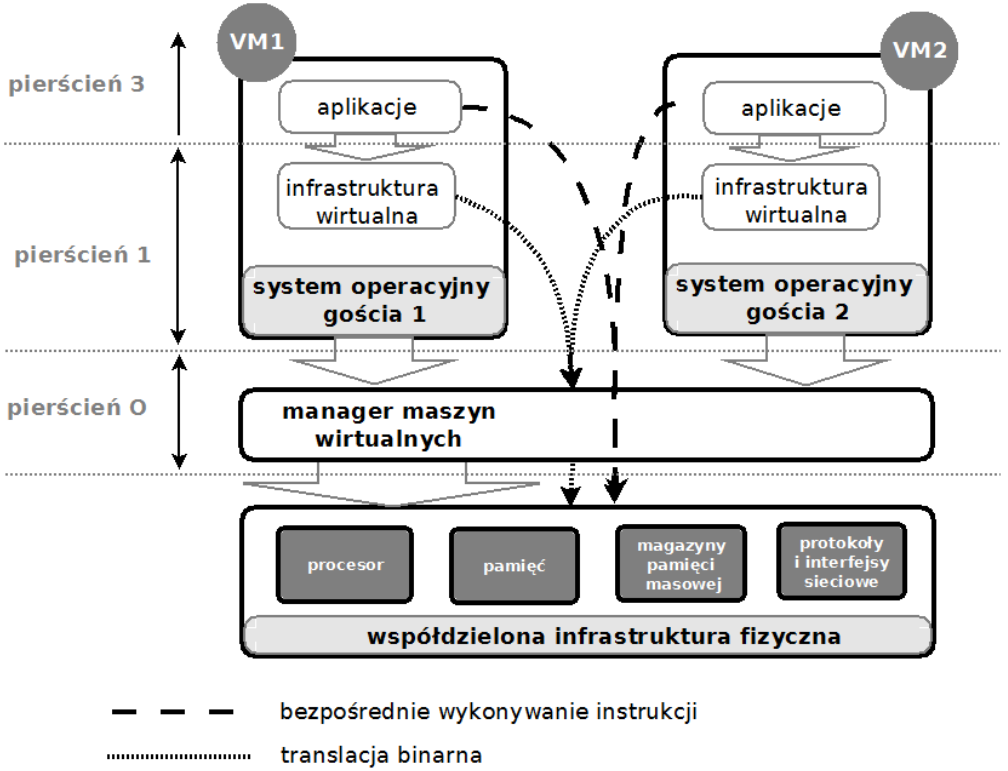
# W2 Techniki wirtualizacji CPU na x86

- metoda pełnej wirtualizacji (ang. full virtualization),
- metoda parawirtualizacji (ang. paravirtualization),
- metoda wirtualizacji sprzętowej (ang. hardware-assisted virtualization).

https://www.vmware.com/techpapers/2007/understanding-full-virtualization-paravirtualizat-1008.html

# W2 Pełna wirtualizacja (ang. full virtualization) (1)

**Pełna wirtualizacja** – system operacyjny gościa może działać bez zmian pod nadzorem VMM, tak jakby działał bezpośrednio na platformie sprzętowej. Każda maszyna wirtualna wykorzystuje dokładnie taką samą kopię rzeczywistego sprzętu.




Tłumaczenie binarne (ang. binary translation) przepisuje części kodu w locie, aby zastąpić wrażliwe ale nie uprzywilejowane instrukcje przez bezpieczny kod realizujący emulację oryginalnej instrukcji

# W2 Pełna wirtualizacja (ang. full virtualization) (2)

Hiperwizor tłumaczy w locie wszystkie instrukcje systemu operacyjnego i buforuje wyniki do wykorzystania w przyszłości, podczas gdy instrukcje na poziomie użytkownika działają bez modyfikacji z natywną szybkością.

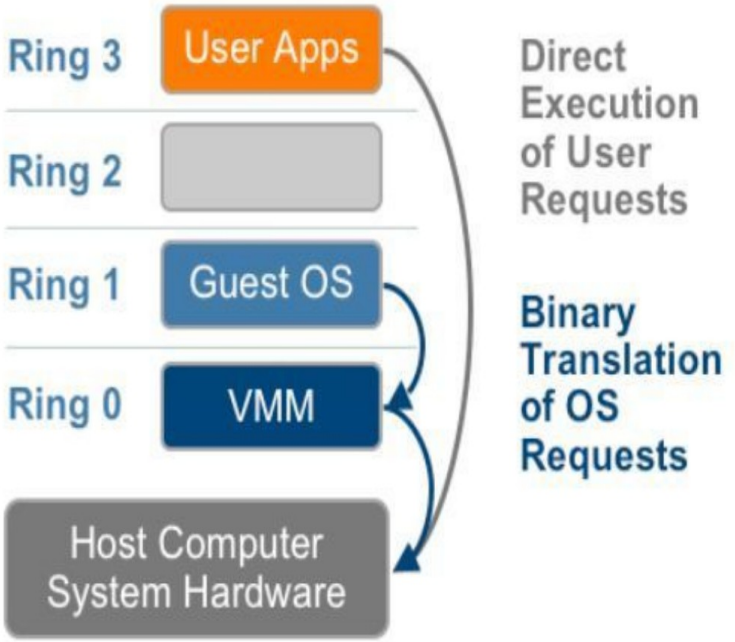
Zalety:

- Brak pomocy sprzętowej,
- Brak modyfikacji systemu gościa
- Izolacja, bezpieczeństwo

Wady:

- Szybkość wykonywania poleceń

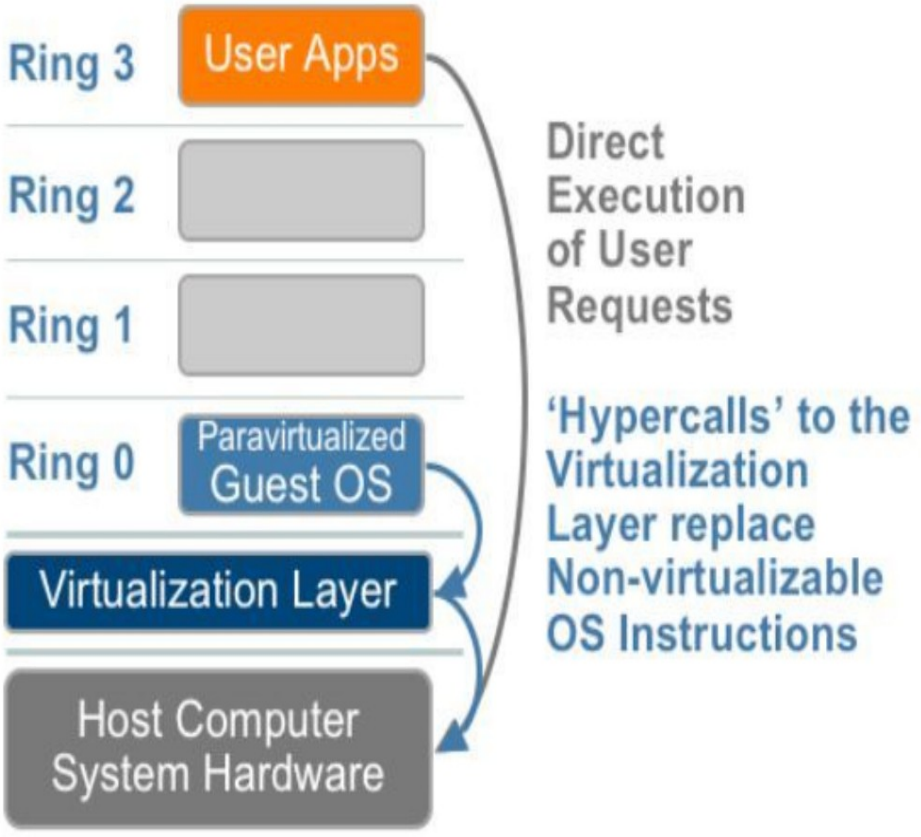
Przykłady: VMware, Microsoft Virtual Server



# W2 Parawirtualizacja (1)

**Parawirtualizacja** — obejmuje modyfikację jądra systemu operacyjnego w celu zastąpienia instrukcji niepodlegających wirtualizacji specjalnymi wywołaniami (ang. hypercalls), które komunikują się bezpośrednio z hiperwizorem warstwy wirtualizacji. Hiperwizor zapewnia również interfejsy do hypercall dla innych krytycznych operacji jądra, takich jak np. zarządzanie pamięci, obsługa przerwań

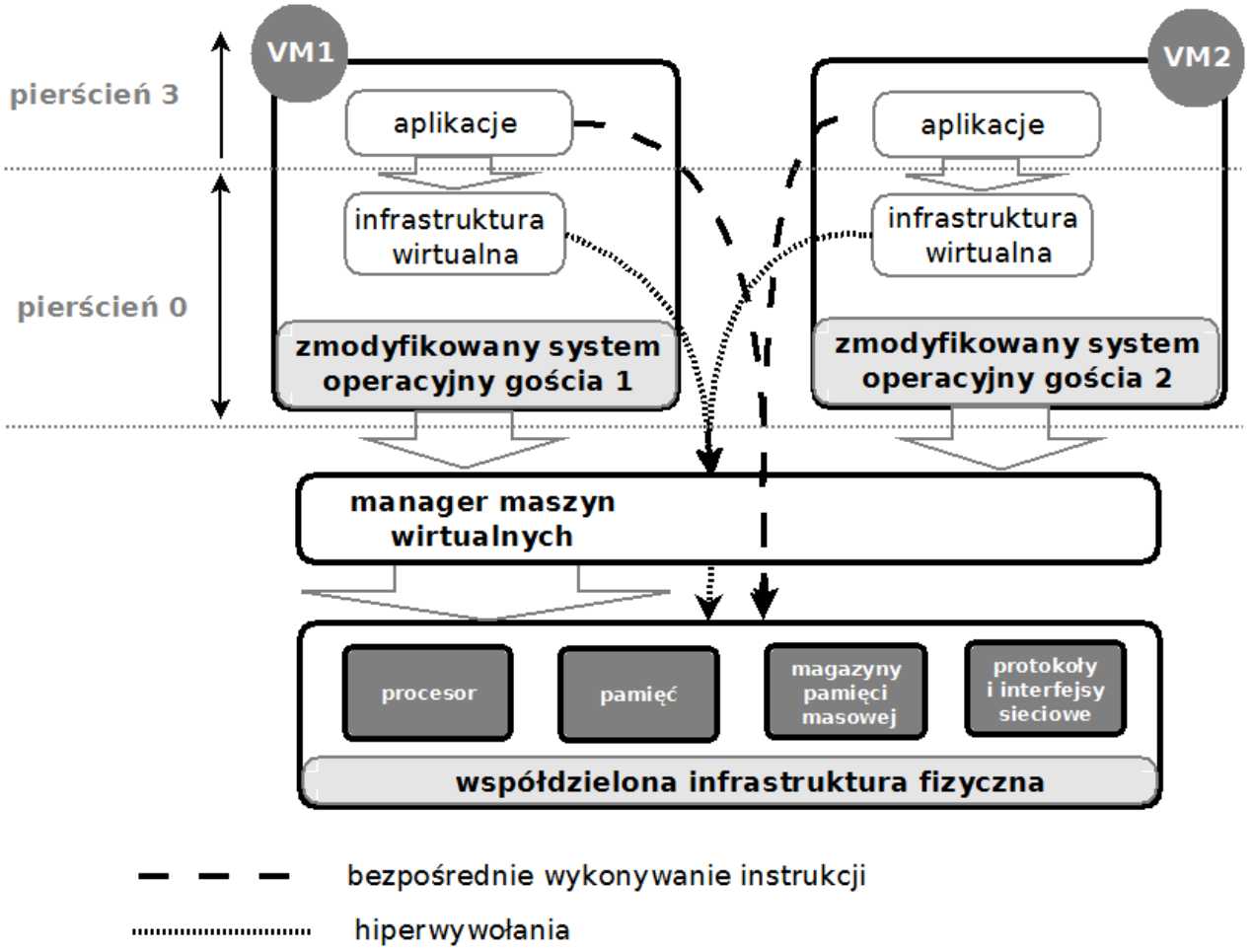
Przykłady: Xen



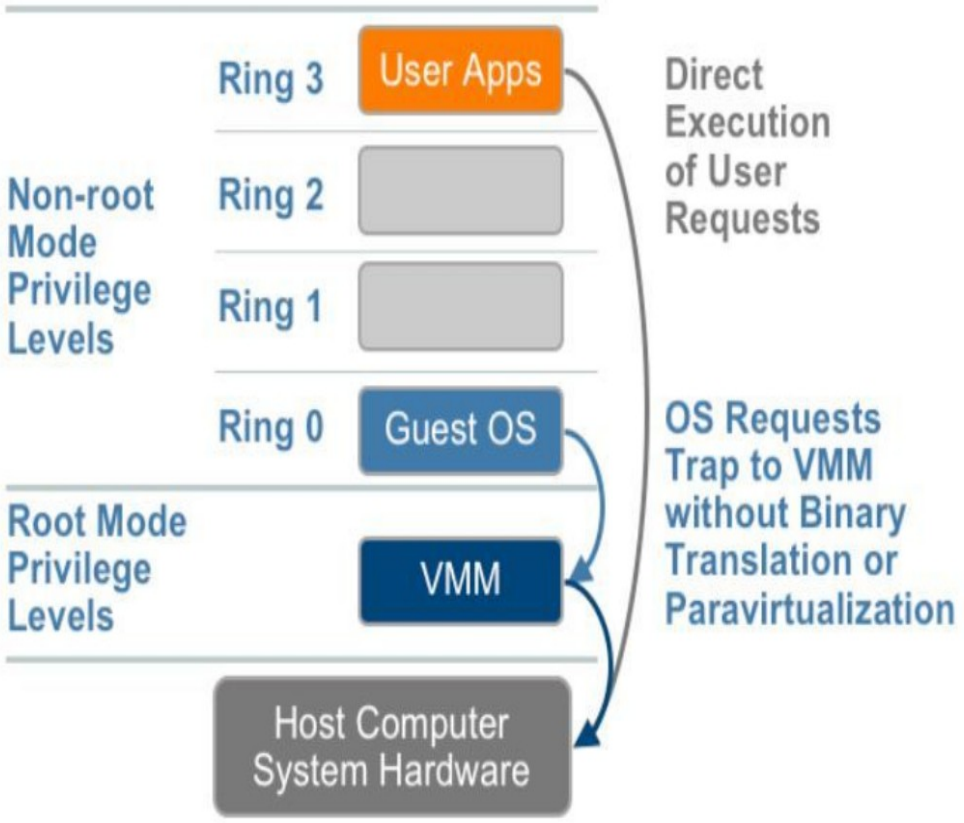
# W2 Parawirtualizacja (2)

Zaletą parawirtualizacji jest duża wydajność, którą osiąga się dzięki wyeliminowaniu potrzeby emulacji sprzętu.

Istotną wadą natomiast jest, wspomniana wyżej, konieczność modyfikacji jądra systemu operacyjnego gościa. Uniemożliwia ona również migracje VM do innych środowisk wirtualnych lub uruchamianie takich zmodyfikowanych systemów na fizycznych systemach komputerowych.



# W2 Wirtualizacja wspomagana sprzętowo (1)

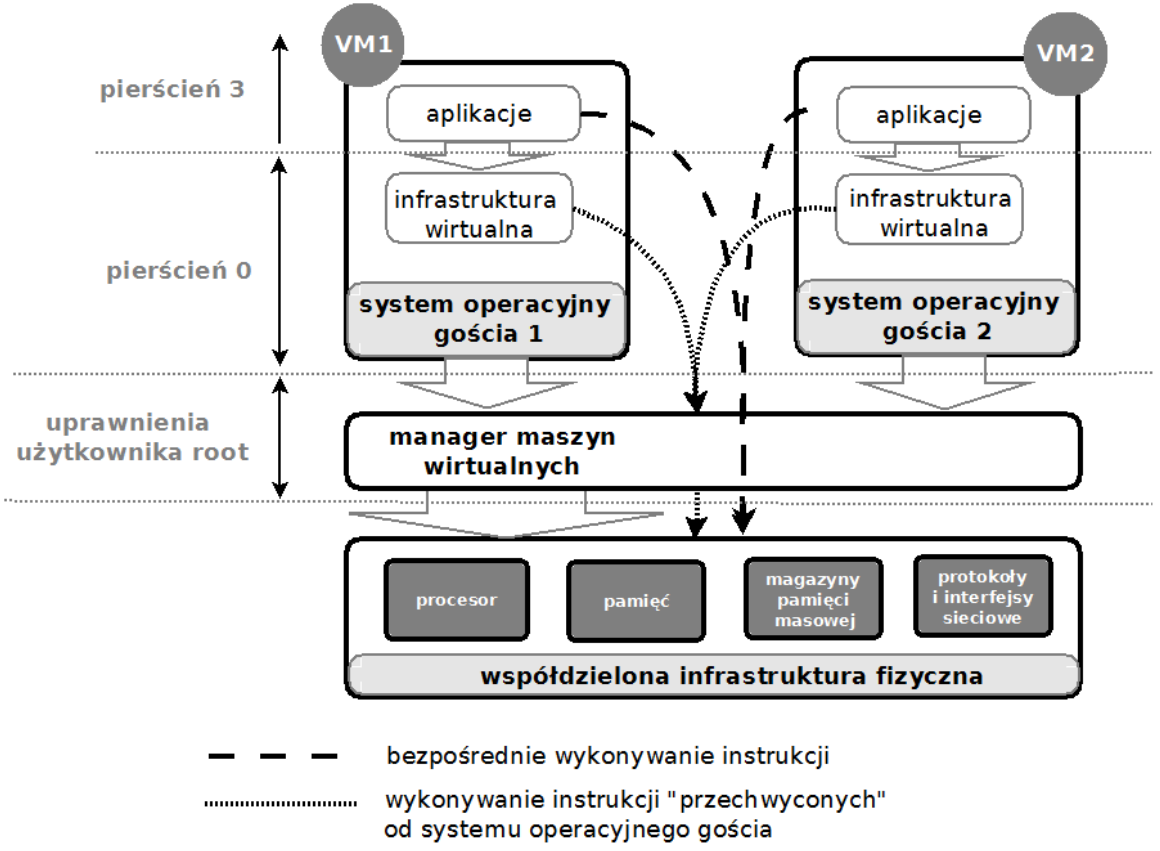



Ring -1
Rozwiązanie problemu instrukcji uprzywilejowanych polega na wprowadzenie kolejnego poziomu uprzywilejowania (tzw. ring -1). Hypervisor działa własnie na tym poziomie, dzieki czemu system operacyjny może działać w ring 0.

Procesory wykorzystujące rozwiązanie Ring -1

- AMD V
- Intel VT

# W2 Wirtualizacja wspomagana sprzętowo (2)




Podstawowe zalety metody wirtualizacji sprzętowej to możliwość bezpośredniego wykonywania rozkazów na fizycznym sprzęcie maszyny macierzystej . Niestety, nawet obecnie stosowane procesory ze wsparciem dla tej metody wirtualizacji cechują się obniżaniem wydajność wraz z rosnącą liczbą uruchamianych systemów operacyjnych gości.

# W2 Porównanie metod twardej (ciężkiej) wirtualizacji

| Cecha | Pełna wirtualizacja | Para-wirtualizacja | Wirtualizacja sprzętowa |
|---|---|---|---|
| Wykonywanie rozkazów | Translacja binarna i bezpośrednie wykonywanie instrukcji | Hiperwywołania i bezpośrednie wykonywanie instrukcji | Model przechwytywania i bezpośrednie wykonywanie instrukcji |
| System operacyjny gościa | Brak modyfikacji | Zmodyfikowany | Brak modyfikacji |
| Wydajność | Dobra | Najlepsza dla określonych przypadków | Zadawalający |
| Położenie VMM oraz poziom uprawnień | Pierścień 0 Uprawnienia root | Lokalizacja poniżej pierścienia 0 | Lokalizacja poniżej pierścienia 0 Uprawnienia root |
| Położenie systemu gościa oraz poziom uprawnień | Pierścień 1 Uprawnienia poniżej root | Pierścień 0 Uprawnienia root | Pierścień 0 Uprawnienia poniżej root |
| Przykład | Vmware ESX | Xen | |

# W2 Lekka wirtualizacja – kontenery (1)

Istotne przesłanki popularności lekkiej wirtualizacji:

- Każda z metod wirtualizacji pozwala na uruchomienie systemów operacyjnych gości a każdy z tych systemów gości wymaga zapewnienia określonego zbioru i określonej wielkości zasobów. Przykładowo, każdy z systemów uruchamia własny zestaw sterowników i usług, niezależnie czy są one faktycznie wykorzystywane czy też nie.

- W systemach chmur obliczeniowych niezwykle istotną kwestią jest też czas niezbędny do uruchamiania, zatrzymywania czy też migracji maszyn wirtualnych. W przypadku wirtualizacji wykorzystującej hipernadzorców operacje te zajmują dziesiątki sekund co jest wartością nieakceptowalną lub silnie ograniczającą wydajność usług.

# W2 Lekka wirtualizacja – kontenery (2)

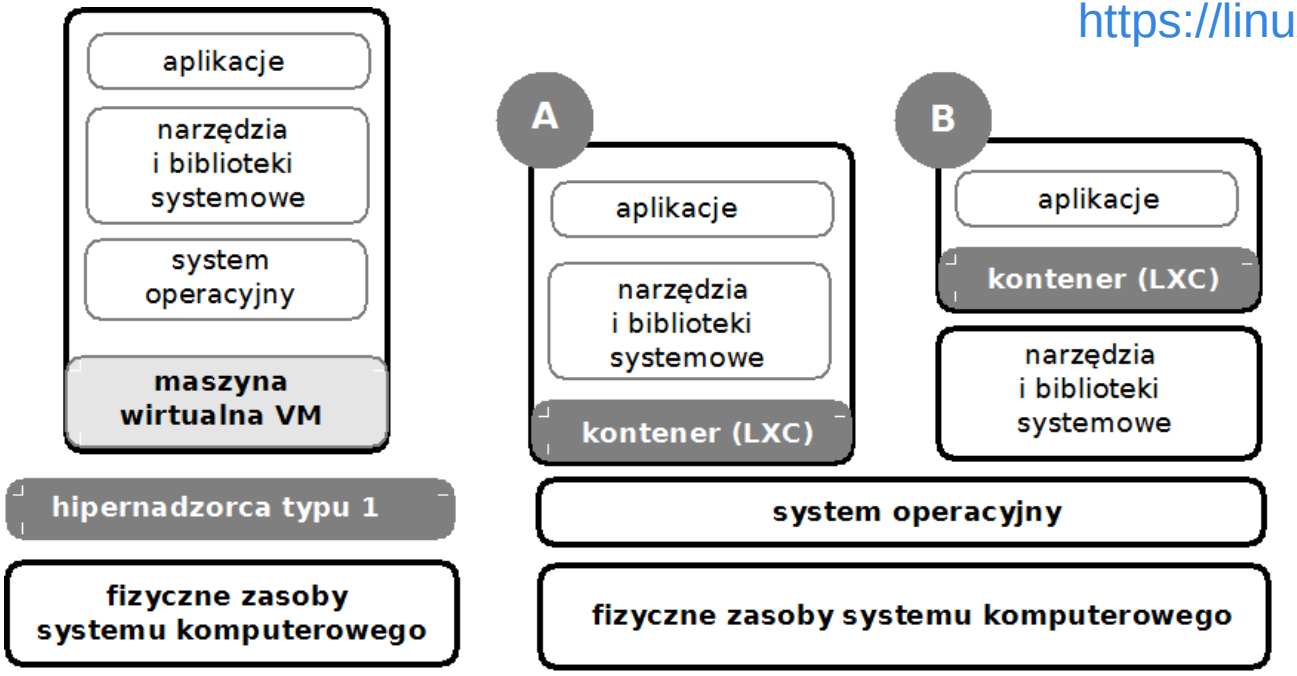
Istotne przesłanki popularności lekkiej wirtualizacji – cd:

W przeciwieństwie do typowych VM (do ciężkiej wirtualizacji), kontenery nie wymagają do działania obecności hipernadzorcy ani zmodyfikowanego systemu operacyjnego. Podstawowa idea funkcjonowania lekkiej wirtualizacji polega na wielokrotnym wykorzystaniu tych samych zasobów systemu macierzystego (współdzielenie jądra systemu macierzystego, systemu plików, stosu obsługi urządzeń wejścia i wyjścia) przy zachowaniu pełnej izolacji pomiędzy poszczególnymi kontenerami.

Kontenery stanowią lekką formę VM i są uruchamiane w izolowanych środowiskach (ang. sandbox), którym przydziela się zasoby niezbędne do funkcjonowania aplikacji użytkowników. Należy również podkreślić, że ta izolacja jest osiągana programowo a nie w wyniku wsparcia przez dedykowaną architekturę procesorów. Zapewnia to przenośność i łatwość implementacji tego rodzaju wirtualizacji (ale z drugiej strony stwarza nowe zagrożenia z punktu widzenia bezpieczeństwa usług uruchamianych na bazie kontenerów).

# W2 Kontenery LXC (ang. Linux containers)

https://linuxcontainers.org/



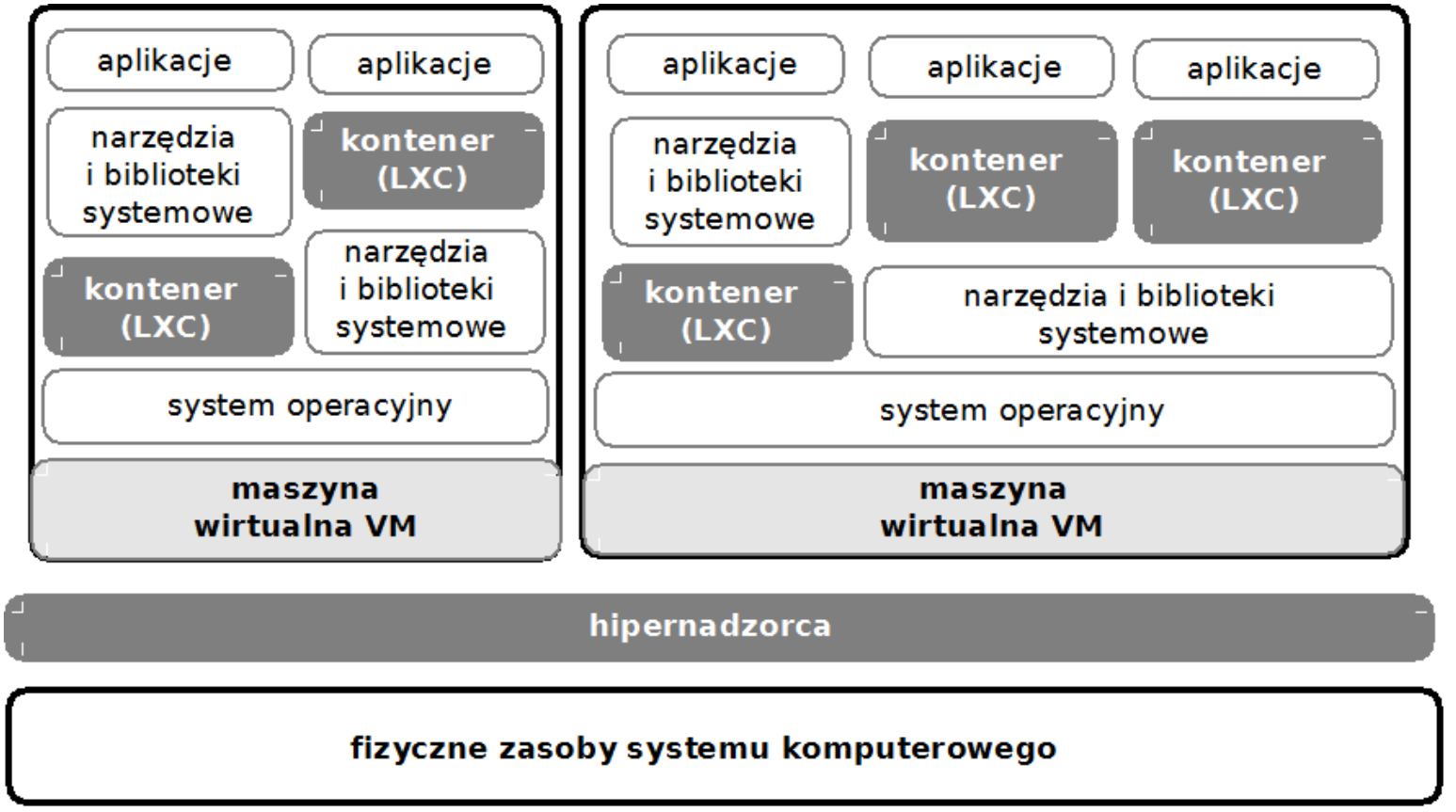

We współczesnych systemach Linux dostępne są dwie implementacje kontenerów LXC. Pierwszy bazuje na jądrze Linux-a ( A) a drugi opiera się o implementacje biblioteki libvirt ( B). Pierwsze z tych rozwiązań jest zdecydowanie popularniejsze (szczególnie jeżeli pod uwagę muszą być brane kwestie bezpieczeństwa systemów wirtualnych)

# W2 Współpraca hipervidor-ów i kontenerów (1)

W przypadku środowisk chmur obliczeniowych, kontenery oferują efektywną, tak pod względem kosztów jak i wydajności, implementację systemów usług na żądanie. W prosty sposób pozwalają uzyskać większą koncentrację izolowanych VM w oparciu o posiadany sprzęt. Jednakże poziom gwarantowanego bezpieczeństwa tej izolacji może być Niewystarczający szczególnie, gdy wiele organizacji przechowuje i przetwarza swoje dane w tym samym środowisku chmurowym.

Z kolei wirtualizacja oparta na hipernadzorcach pozwala zwiększyć poziom bezpieczeństwa kosztem zalet oferowanych przez kontenery. Z tego względu, w ostatnim czasie można zaobserwować tendencję do projektowania środowisk chmur obliczeniowych, tak by móc jednocześnie wykorzystywać zalety obu metod implementacji wirtualizacji. Takie połączenie oferuje cechy izolacji zasobów pomiędzy VM wykorzystującymi hipernadzorców a jednocześnie wysoką efektywność wykorzystania zasobów, jakie cechuje lekka wirtualizacja

# W2 Współpraca hipervidor-ów i kontenerów (2)



# W2 Przegląd wybranych rozwiązań rynkowych (1)


https://xenproject.org/

https://wiki.xenproject.org/wiki/Xen_Project_Software_Overview#What_is_the_Xen_Project_Hypervisor.3F

# W2 Przegląd wybranych rozwiązań rynkowych (2)


https://www.vagrantup.com/

https://medium.com/swlh/hashicorp-vagrant-101-6f7e613d8af

# W2 Przegląd wybranych rozwiązań rynkowych (3)


https://www.proxmox.com/en/

https://medium.com/devops-dudes/proxmox-101-8204eb154cd5

# W2

# PYTANIA / UWAGI ?