Systemy Operacyjne — Wirtualizacja

Arkadiusz Chrobot, Grzegorz Łukawski

Katedra Systemów Informatycznych, Politechnika Świętokrzyska w Kielcach

Kielce, 24 stycznia 2025

Plan

Wstęp

Wirtualizacja jest techniką zarządzania zasobami, która umożliwia ich współdzielenie przez procesy lub nawet systemy komputerowe za pomocą:

- podziału,
- agregacji.

W węższym znaczeniu oznacza zastosowanie **maszyn wirtualnych**, celem umożliwienia wykonania oprogramowania, które przeznaczone jest dla tej samej platformy sprzętowej, ale wymaga specjalnego środowiska wykonawczego. Ten aspekt wirtualizacji oraz pojęcia pokrewne, takie jak **emulacja** i **symulacja** będą przedmiotem tego wykładu.

Symulacja

Symulacja jest pojęciem o szerszym znaczeniu niż emulacja. Oznacza ono odtwarzanie przez program komputerowy zachowania określonego modelu abstrakcyjnego. W przypadku informatyki model ten dotyczy najczęściej urządzenia przetwarzającego informację. Jako przykłady można podać symulatory Maszyny Turinga i Maszyny RAM.

Emulacja

Emulacja polega na symulowaniu przez program komputerowy uruchomiony w systemie komputerowym o danej strukturze pełnego zachowania innego systemu komputerowego, najczęściej o odmiennej konstrukcji. Program symulujący określany jest mianem **emulatora**.
Istnieją dwa rodzaje emulatorów:

1. **emulatory pełne** - działają zgodnie ze specyfikacją określonego fizycznego komputera (przykłady: QEMU, Bochs, UAE, Frodo),
2. **emulatory API** - emulują jedynie działanie interfejsu programistycznego określonego systemu operacyjnego i nazywane są często **warstwami kompatybilności** (np. WINE, DosBox, dosemu).

Warto zauważyć, że emulatory API nie są całkowicie zgodne z podaną wyżej definicją emulatora - symulują działanie tylko jednej z warstw systemu operacyjnego. W związku z tym niektórzy informatycy są zdania, że nie należy nazywać ich emulatorami. Szczególnie podkreślają to twórcy WINE, którego nazwa jest rozwijana rekurencyjnie do „WINE Is Not an Emulator".

Emulacja - możliwości realizacji

Emulatory pełne symulują działanie platformy sprzętowej do poziomu kodu maszynowego. Istnieją dwie możliwości realizacji tak dokładnego odwzorowania. Pierwsza polega na **interpretacji** programu przygotowanego dla danej platformy przez emulator. Druga oparta jest na dynamicznej rekompilacji, tzn. tłumaczeniu określonego bloku kodu zamiast pojedynczej instrukcji.

Emulatory - motywacja

Emulatory tworzone są z trzech powodów:

1. potrzeba użycia określonej platformy sprzętowej, która nie jest dostępna,
2. potrzeba uruchomienia oprogramowania przeznaczonego na starsze platformy sprzętowe,
3. stworzenie platformy testowej dla oprogramowania.

W latach 80. ubiegłego wieku termin „emulacja” oznaczał sprzętową imitację danej platformy z użyciem mikrokodu, natomiast „symulacja” oznaczała imitację na poziomie oprogramowania.

Wprowadzenie

Podczas gdy „emulacja" oznacza głównie symulację określonej platformy sprzętowej na innej platformie, to **wirtualizacja** najczęściej oznacza w tym kontekście stworzenie kopii takiej samej platformy na jakiej przeprowadzana jest wirtualizacja. Kopia ta, nazywana **maszyną wirtualną**, nie jest dokładna z racji tego, że nie może ona używać w całości zasobów należących do maszyny fizycznej. Niemniej jednak jest ona na tyle wierna, że pozwala uruchomić system operacyjny i oprogramowanie użytkowe.
W artykule z 1974 roku Gerald J. Popek i Robert P. Goldberg określili kryteria jakie powinna spełniać poprawnie działająca maszyna wirtualna.

1. **Odpowiedniość** - program działający na maszynie wirtualnej musi zachowywać się dokładnie tak samo jakby był wykonywany na maszynie fizycznej.
2. **Kontrola zasobów** - maszyna wirtualna musi w pełni kontrolować wszystkie zasoby podlegające wirtualizacji.
3. **Wydajność** - większość instrukcji musi być wykonywana na maszynie fizycznej, bez udziału maszyny wirtualnej.

W praktyce warunki te nie muszą (i zazwyczaj nie są) dokładnie wypełnione. Powoduje to ograniczenia w działaniu maszyn wirtualnych oraz niższą wydajność.

Wirtualizacja - motywacja

Zastosowania wirtualizacji:

- zwiększenie niezawodności serwerów,
- prosta migracja maszyn wirtualnych,
- możliwość jednoczesnej pracy wielu wersji systemu operacyjnego,
- wirtualny hosting,
- testowanie systemów operacyjnych, aplikacji i oprogramowania sieciowego.

Wirtualizacja - motywacja

Zastosowania wirtualizacji:

- zwiększenie niezawodności serwerów,
- prosta migracja maszyn wirtualnych,
- możliwość jednoczesnej pracy wielu wersji systemu operacyjnego,
- wirtualny hosting,
- testowanie systemów operacyjnych, aplikacji i oprogramowania sieciowego.

Wirtualizacja - motywacja

Zastosowania wirtualizacji:

- zwiększenie niezawodności serwerów,
- prosta migracja maszyn wirtualnych,
- możliwość jednoczesnej pracy wielu wersji systemu operacyjnego,
- wirtualny hosting,
- testowanie systemów operacyjnych, aplikacji i oprogramowania sieciowego.

Wirtualizacja - motywacja

Zastosowania wirtualizacji:

- zwiększenie niezawodności serwerów,
- prosta migracja maszyn wirtualnych,
- możliwość jednoczesnej pracy wielu wersji systemu operacyjnego,
- wirtualny hosting,
- testowanie systemów operacyjnych, aplikacji i oprogramowania sieciowego.

Wirtualizacja - motywacja

Zastosowania wirtualizacji:

- zwiększenie niezawodności serwerów,
- prosta migracja maszyn wirtualnych,
- możliwość jednoczesnej pracy wielu wersji systemu operacyjnego,
- wirtualny hosting,
- testowanie systemów operacyjnych, aplikacji i oprogramowania sieciowego.

Hipernadzorca typu 1 - wymagania

G.J.Popek i R.P.Goldberg w przytoczonym wcześniej artykule podali również wymagania, które musi spełniać architektura, aby umożliwić realizację wirtualizacji z użyciem hipernadzorcy typu 1. Wyróżnili oni trzy grupy rozkazów procesora: **rozkazy zwykłe**, które mogą być wykonywane w obu trybach pracy procesora, **rozkazy uprzywilejowane**, których próba wykonania w trybie użytkownika powoduje wygenerowanie wyjątku oraz **rozkazy wrażliwe**, które powinny być wykonywane tylko w trybie jądra. Dana architektura umożliwia realizację wirtualizacji z użyciem hipernadzorcy typu 1, jeśli zbiór jej instrukcji wrażliwych jest podzbiorem zbioru instrukcji uprzywilejowanych. W przypadku architektur PC taką wirtualizację umożliwiają procesory firmy AMD z technologią SVM (ang. *Secure Virtual Machine*)[1] oraz firmy Intel z technologią VT (ang. *Virtualization Technology*), które umożliwiają hipernadzorcy określenie, które rozkazy powinny generować wyjątki przy próbie wykonania ich w trybie użytkownika, a które nie.

[1] AMD zdecydowało się później przemianować tę technologię na AMD - V.

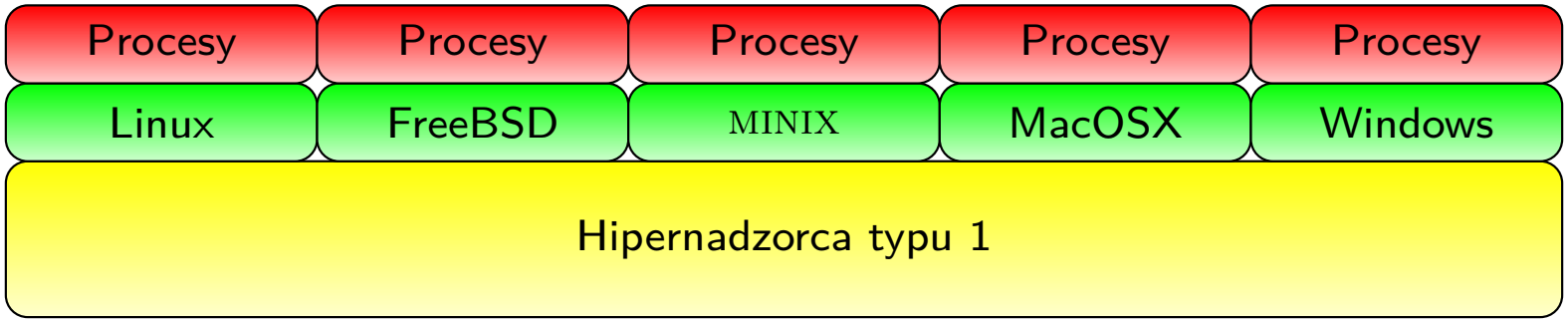
Hipernadzorca typu 1 - charakterystyka działania

Działanie wirtualizacji opartej o hipernadzorcę typu 1 można opisać następująco:

- w trybie jądra procesora fizycznego działa tylko hipernadzorca nazywany również **monitorem maszyn wirtualnych**,
- hipernadzorca kontroluje pracę wszystkich pracujących maszyn wirtualnych,
- system operacyjny maszyny wirtualnej, nazywany **systemem operacyjnym - gościem** działa w trybie użytkownika maszyny rzeczywistej oraz w trybie jądra maszyny wirtualnej,
- procesy użytkownika działają w trybie użytkownika zarówno maszyny wirtualnej, jaki i rzeczywistej,
- rozkazy nieuprzywilejowane są wykonywane przez system operacyjny - gościa i procesy użytkownika bezpośrednio,
- jeśli pojawi się wyjątek spowodowany przez próbę wykonania rozkazu uprzywilejowanego, to hipernadzorca sprawdza, co próbowało go wykonać; jeśli był to proces użytkownika, to uruchamiana jest obsługa wyjątku w systemie - gościu, jeśli był to system - gość, to hipernadzorca wykonuje ten rozkaz.

Wirtualzację tego typu zastosowano po raz pierwszy w komputerach typu mainframe firmy IBM. System operacyjny obsługujący to rozwiązanie początkowo nazwano CP/CMS, a później przemianowano na VM/370.

Hipernadzorca typu 1



Rysunek: Wirtualizacja z użyciem hipernadzorcy typu 1

Hipernadzorca typu 2 - wymagania

Wirtualizacja z użyciem hipernadzorcy typu 2 nie wymaga wsparcia ze strony procesora w postaci takich technologi jak Intel VT lub AMD - V. Przykładami takich hipernadzorców są VirtualBox i VirtualPC.

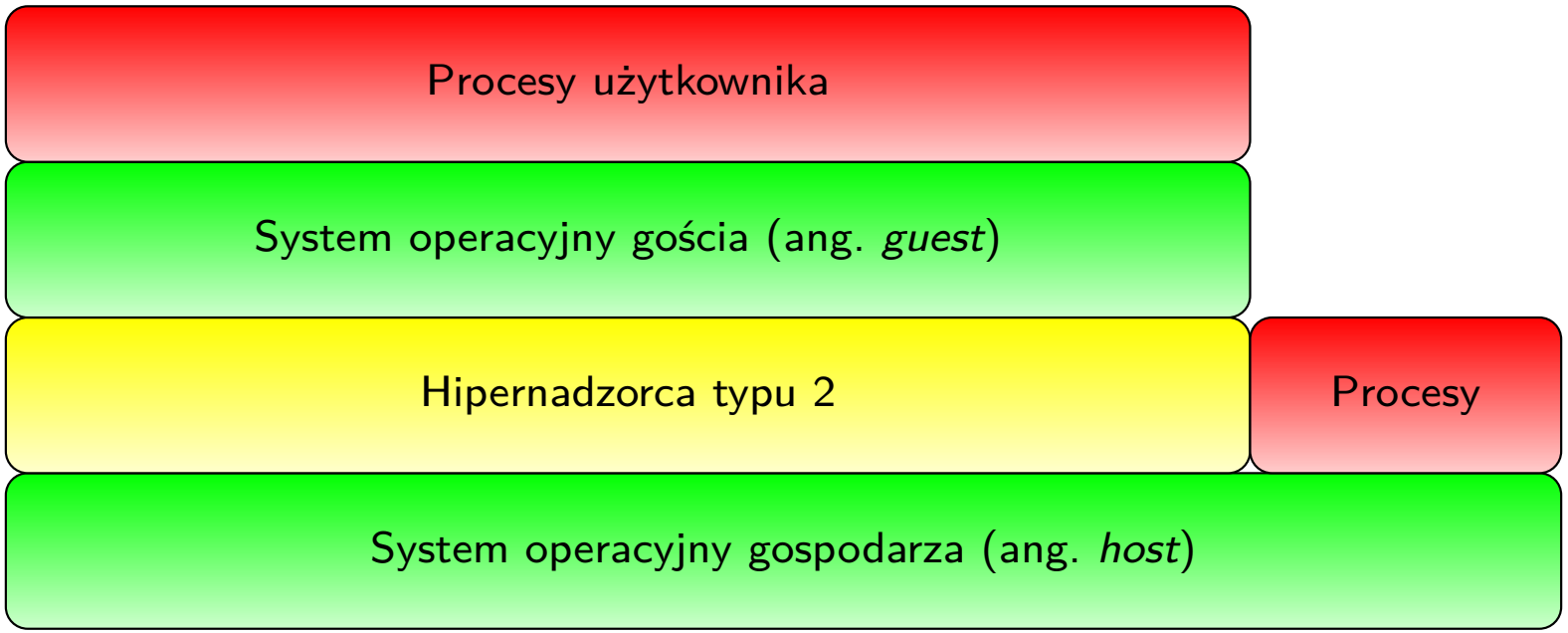
Hipernadzorca typu 2 - charakterystyka działania

Działanie wirtualizacji z użyciem hipernadzorcy typu 2 można opisać następująco:

- hipernadzorca jest oprogramowaniem działającym w trybie użytkownika pod kontrolną systemu operacyjnego nazywanego **systemem operacyjnym - gospodarzem**,
- z punktu widzenia użytkownika hipernadzorca zachowuje się jak emulator maszyny rzeczywistej na której jest uruchomiony; pozwala zainstalować system operacyjny (nazywany systemem operacyjnym - gościem) i uruchamiać procesy użytkownika,
- problem rozkazów wrażliwych hipernadzorca rozwiązuje stosując **translację binarną**, która polega na znalezieniu **bloków podstawowych**, czyli ciągów rozkazów zakończonych dowolnym rozkazem zmieniającym przepływ sterowania i zastąpieniu wszystkich instrukcji wrażliwych znajdujących się w takich blokach wywołaniami procedur hipernadzorcy,
- celem zwiększenia wydajności stosowana jest translacja z wyprzedzeniem oraz pamięci podręczne.

Wbrew intuicji, wirtualizacja z użyciem hipernadzorcy typu 2 może być w pewnych warunkach wydajniejsza, niż wirtualizacja z użyciem hipernadzorcy typu 1.

Hipernadzorca 2



Rysunek: Wirtualizacja z użyciem hipernadzorcy typu 2

Parawirtualizacja - wymagania

Parawirtualizacja jest wydajniejsza niż opisane wcześniej sposoby wirtualizacji, ale wymaga specjalnie przygotowanej wersji systemu operacyjnego - gościa. Przykładem hipernadzorcy umożliwiającego parawirtualizację jest Xen.

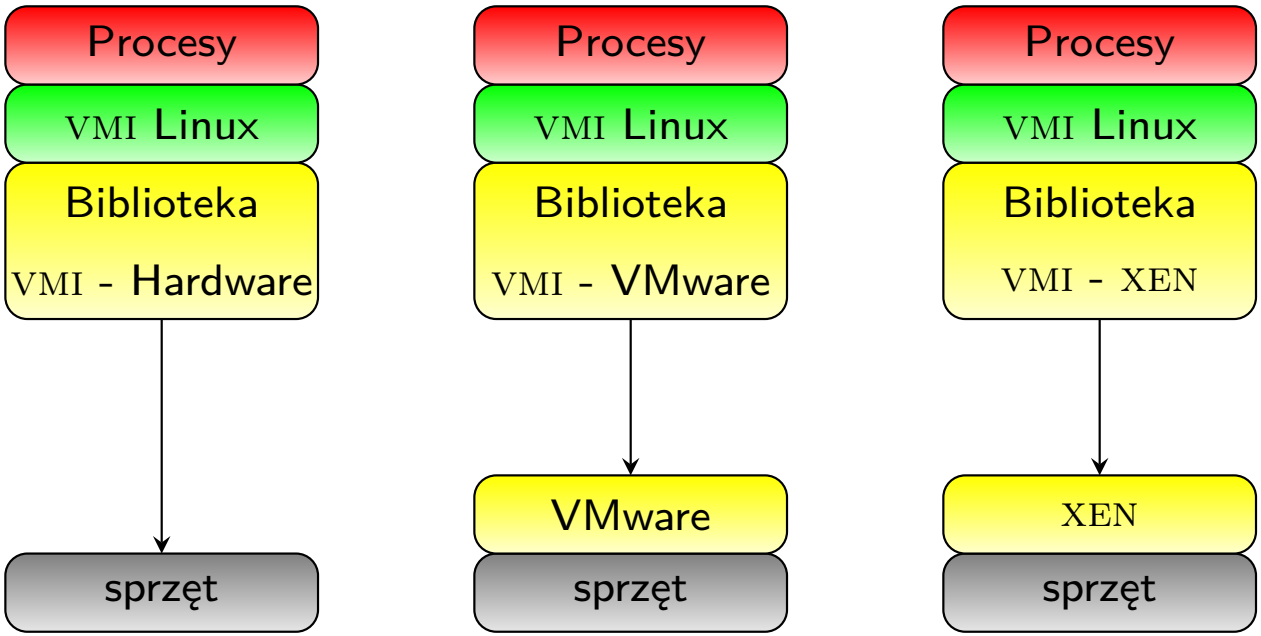
Parawirtualizacja - charakterystyka działania

Działanie parawirtualizacji można scharakteryzować następująco:

- hipernadzorca działa w trybie systemowym maszyny rzeczywistej, tak jak hipernadzorca typu 1,
- hipernadzorca posiada własne API,
- kod źródłowy systemu operacyjnego - gościa jest specjalnie spreparowany; każde odwołanie do instrukcji wrażliwej zastąpiono w nim wywołaniem podprogramu wchodzącego w skład API hipernadzorcy.
- wykonywalna wersja systemu - gościa jest uruchamiana w trybie użytkownika maszyny rzeczywistej; nie wykonuje ona żadnych (lub prawie żadnych) rozkazów wrażliwych, jedynie korzysta z API hipernadzorcy.

Parawirtualizacja może być stosowana na jednym komputerze, obok wcześniej wymienionych sposobów wirtualizacji. Hipernadzorców, którzy umożliwiają parawirtualizację jest wielu i każdy z nich posiada swoje API. Aby umożliwić systemowi - gościowi współpracę z nimi wszystkimi, bez konieczności wykonywania dodatkowych modyfikacji wprowadzono, warstwę pośredniczącą nazywaną **biblioteką VMI** (ang. Virtual Machine Interface).

Parawirtualizacja



Rysunek: Parawirtualizacja z użyciem techniki VMIL

Wirtualizacja pamięci

Oprócz procesora również pozostałe zasoby maszyny rzeczywistej muszą podlegać wirtualizacji. Najważniejszym z nich jest pamięć operacyjna. We współczesnych systemach najczęściej stosowanym sposobem zarządzania RAM jest stronicowanie na żądanie, co utrudnia gospodarowanie pamięcią na rzecz maszyn wirtualnych. Kłopotliwy scenariusz powstaje wtedy, gdy co najmniej dwie maszyny wirtualne próbują odwzorować swoje strony na te same ramki. Rozwiązaniem problemu jest utrzymywanie przez hipernadzorcę dla każdej maszyny wirtualnej specjalnej tablicy stron, która stanowi kolejną warstwę w procesie translacji numeru strony (zamienia adres ramki wirtualnej maszyny na adres ramki maszyny rzeczywistej). Inny problem polega na tym, że system operacyjny - gość może zmienić zawartość tablicy bez używania rozkazów wrażliwych, stosując jedynie zwykły zapis do pamięci. Ta modyfikacja powinna być wykryta przez hipernadzorcę, żeby mógł uaktualnić tablicę dla maszyny wirtualnej. W przypadku wirtualizacji hipernadzorca wykrywa, które strony zawierają tablicę strony systemu - gościa i oznacza je jako *tylko do odczytu*. Każda próba modyfikacji zawartości tych stron generuje wyjątek, który obsługuje hipernadzorca. W przypadku parawirtualizacji system - gość powiadamia hipernadzorcę o zmianie zawartości swojej tablicy stron.

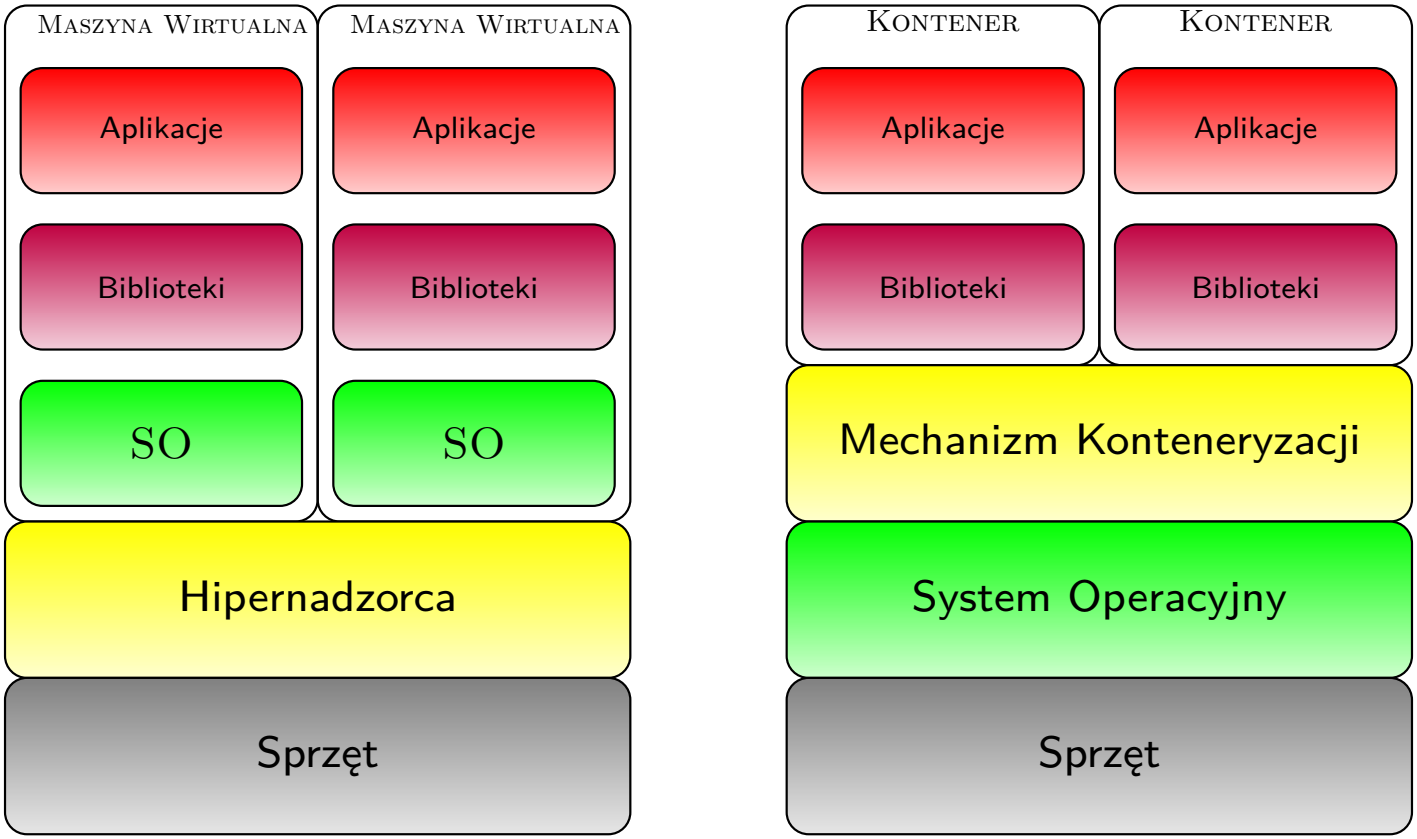
Wirtualizacja urządzeń wejścia-wyjścia

Hipernadzorca musi także przechwytywać polecenia sterowników urządzeń systemu operacyjnego - gościa. Konieczność ta wynika z potrzeby ochrony sprzętu oraz niemożliwości przydzielenia niektórych zasobów (np. dysku) w całości pojedynczej maszynie wirtualnej. Zaletą takiego rozwiązania jest również udogodnienie polegające na tym, że oprogramowanie obsługujące sprzęt starego lub innego typu można wykonać z użyciem urządzeń nowego typu. Konwersją poleceń między oboma typami urządzeń zajmie się ponownie hipernadzorca. Hipernadzorca musi zadbać również o poprawność przebiegu transmisji DMA. W wykonaniu tego zadania pomaga sprzęt w postaci IOMMU. Problem wirtualizacji obsługi wejścia-wyjścia można również rozwiązać przeznaczając jedną z maszyn wirtualnych do tego zadania i przekierowując do niej żądania wykonania operacji pochodzące z innych maszyn. To rozwiązanie jest szczególnie wygodne w przypadku parawirtualizacji i stosuje je Xen. Maszyna wirtualna, która zajmuje się wykonywaniem operacji I/O nazywa się w tej platformie **domeną 0**. Takie podejście do realizacji operacji wejścia-wyjścia korzystne jest również w wirtualizacji z użyciem hipernadzorcy typu 1, bo nie musi on zawierać sterownika do danego urządzenia. Hipernadzorca typu 2 może polegać na tym, że sterownik określonego urządzenia posiada system operacyjny - gospodarz.

Wirtualizacja a kontenery

Konteneryzacja jest formą wirtualizacji, ale nie sprzętu lecz systemu operacyjnego (Rys. 4). Kontener, w przeciwieństwie do maszyny wirtualnej, nie ma systemu operacyjnego, ale dostarcza wykonywanym w nim procesom (aplikacjom) środowiska uruchomieniowego w postaci bibliotek i innych zasobów, które są niezbędne do ich działania. Kontenery są bardziej przenośne od maszyn wirtualnych, mają zazwyczaj mniejszy rozmiar i łatwiej jest je modyfikować. Stanowią one dosyć często komponenty rozwiązań bazujących na mikrousługach (ang. *microservices*). Ich wadą jest większa podatność na problemy związane z bezpieczeństwem, gdyż nie są odizolowane od siebie w takim stopniu jak maszyny wirtualne.

Wirtualizacja a kontenery



Rysunek: Porównanie maszyn wirtualnych i kontenerów na podstawie „Containers Vs. Virtual Machines”

Wirtualizacja - podsumowanie

Wirtualizacja w przypadku sprzętu klasy PC i stacji roboczych jest nowością. Wiele problemów, które obecnie powoduje gorszą wydajność maszyn wirtualnych w przyszłości powinno być rozwiązanych za pomocą odpowiedniego wsparcia sprzętowego. Należy również nadmienić, że pojęcie maszyny wirtualnej nie jest ściśle związane z opisanym rodzajem wirtualizacji. Przykładem są tu maszyny wirtualne stosowane w językach programowania Java (JVM, Dalvik, ART) i C# (CLR). Pewną formę wirtualizacji wykorzystują systemy rodziny Windows do wykonywania programów napisanych dla systemu DOS. Rolę hipernadzorcy pełni tu procesor, który tworzy maszyny wirtualne dla których zachowuje się jak swój 16-bitowy poprzednik (procesor 8086). Tryb pracy procesora, w którym imituje działanie starych procesorów nazywany jest **trybem wirtualnym**.

Źródła

Do przygotowania niniejszego tekstu wykorzystano artykuł w Wikipedii na temat emulatorów: http://en.wikipedia.org/wiki/Emulate oraz informacje zawarte w literaturze podanej na stronie przedmiotu.

Pytania

?

Koniec

Dziękuję Państwu za uwagę!