## Wirtualizacja

Poniższy dokument objaśnia teoretyczne zagadnienia związane z technologią wirtualizacji. Nie zajmuje się żadnym z jej aspektu nazbyt szczegółowo, lecz stanowi przekrojowe omówienie tegoż zjawiska. Celem opracowania jest możliwie dobre wyrobienie intuicji u czytelnika na temat tego zjawiska i przygotowanie go do prezentacji praktycznych.

### Definicja

Aby móc swobodnie poruszać się po tematyce wirtualizacji, musi posiadać wspólną definicję omawianego zjawiska. W poniższym dokumencie przyjmiemy następującą:

> Wirtualizacją nazywamy użycie oprogramowania w celu stworzenia abstrakcji (iluzji) posiadanych zasobów.

Jest to definicja bardzo obszerna, mieszcząca w sobie wiele zjawisk związanych z komputerami i oprogramowaniem, my zaś będziemy zajmować się tylko jej wycinkiem. Jaki to będzie wycinek, uściślimy poniżej.

### Historia

By być możliwie ścisłym o pierwszym szeroko znanym zastosowaniu wirtualizacji można mówić przy tworzonym we współudziale Uniwersytetu w Manchesterze i firm Ferranti oraz Plessey komputerze Atlas. W zastosowanej architekturze część instrukcji zewnętrznych była przez sprzęt, a część zostawała przechwycona przez program nadzorujący. Dziś pełniłby on jeszcze dużo innych funkcji i zostałby nazwany systemem operacyjny. Wtedy jednak zajmował się wykonywaniem w odpowiedni sposób przechwyconych wywołań, a ta skromna wirtualizacja była wyłącznie efektem ubocznym dość rewolucyjnych rozwiązań zastosowanych w komputerze.

Faktyczne zastosowanie wirtualizacji pojawiło się dopiero w połowie lat 60 w wyniku prac laboratoriów IBM. Wtedy właśnie centrum badawcze Watsona zakończyło projekt M44/44X, który składał się z fizycznej maszyny M44, oznaczanej też IBM 7044 oraz maszyn wirtualnych 44X. Każda z nich operowała na pamięci M44, korzystając przy tym z wieloprogramowania i pamięci wirtualnej. Wkrótce potem stworzono kolejne maszyny wirtualne: CP40 dla IBM 860/40, CP67 dla 360/67, a także wiele innych. Zwykle maszyny wirtualne były idealnymi kopiami fizycznego sprzętu, obsługiwanymi przez monitor maszyn wirtualnych, kontaktujący się bezpośrednio z systemem komputerowym. W ten sposób mogło być tworzone wiele kopii tego samego systemu operacyjnego, co pozwalało znacząco zwiększyć wykorzystanie procesora na komputerach klasy Mainframe. Wielu użytkowników mogło korzystać z tej samej fizycznej maszyny, dzięki czemu – choć tak wiele czasu trzeba zabierała interakcja między komputerem i człowiekiem – procesor był w większym stopniu wykorzystywany.

Od tego czasu wirtualizacja była nadal rozwijana, lecz do momentu, gdy komputery osobiste uzyskały wystarczająco dużą moc obliczeniową, zastosowanie maszyn wirtualnych w gospodarstwach domowych lub niewyspecjalizowanych firmach było ograniczone. Druga połowa poprzedniego dziesięciolecia to jednak czas dość gwałtownego rozwoju technologii wirtualnych – pojawił się niezwykle dziś popularny język Java, korzystający z maszyn

wirtualnych, działalność zaczęła również firma VMWare, specjalizująca się w oprogramowaniu do wirtualizacji. Również Microsoft dołączył do rywalizacji w tej branży, podobnie jak niedawno Google. W wyniku konkurencji część swych produktów dotyczących wirtualnych serwerów Microsoft i VMWare zaczęły udostępniać za darmo. Wraz ze wzrostem mocy obliczeniowej i zmniejszania kosztów pamięci można oczekiwać popularyzacji maszyn wirtualnych i używania ich także w domowych stanowiskach.

**Kryteria wirtualizacji**

W 1974 roku Gerald J. Popek i Robert P. Goldberg przedstawili kryteria właściwego funkcjonowania maszyny wirtualnej oraz kryteria możliwości zbudowania takowej na zadanej architekturze komputera.

Wirtualna maszyna w myśli Popka i Goldberga by być skuteczna musi spełniać trzy warunki:

- Odpowiedniość – program działający na maszynie wirtualnej musi zachowywać się w dokładnie taki sam sposób, jak na rzeczywistym sprzęcie;
- Kontrola zasobów – wirtualna maszyna musi w pełni kontrolować wszystkie zasoby, które są wirtualizowane;
- Wydajność – większa część instrukcji musi być wykonywana bez udziału maszyny wirtualnej;

W systemie operacyjnym wyróżniamy trzy zbiory instrukcji dostępnych na danej architekturze:

- Instrukcje uprzywilejowane – ich efektem jest przerwanie lub wywołanie systemowe w trybie użytkownika lub ich brak w trybie jądra;
- Instrukcja wrażliwe ze względu na kontrolę – w trakcie wykonywania mogą zmienić konfigurację zasobów systemowych;
- Instrukcje wrażliwe ze względu na wykonanie – ich działanie jest zależne od konfiguracji systemu;

Warto pamiętać, że nie są ani rozłączne, ani nie sumują się do zbioru wszystkich instrukcji architektury.

Z tą wiedza możemy poznać twierdzenie przedstawione przez Popka i Goldberga:

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

Z tegoż twierdzenia wynika tyle, iż każda operacja, która mogła spowodować niepoprawne działanie maszyny wirtualnej, powoduje przerwanie lub wywołanie systemowe, dzięki czemu maszyna może podjąć odpowiednie kroki. Cała reszta instrukcji nieuprzywilejowanych jest przekazywana po prostu do fizycznego sprzętu, aby uzyskać odpowiednią wydajność.

Okazuje się jednak, że można zbudować maszynę wirtualizującą architekturę nie spełniającą założeń twierdzenia, choć zwykle powoduje to utratę wydajności lub nakłada inne ograniczenia. Przykładami są parawirtualizacja oraz dynamiczna rekompilacja z wykrywaniem wrażliwych instrukcji.

**Typy wirtualizacji**

Pojęcie wirtualizacji jako abstrakcji zasobów jest niezwykle szerokie i obejmuje wiele rozwiązań o zupełnie różnych zastosowaniach. Dlatego oddzielnie rozważymy kilka gałęzi technologicznych, które składają się na to zagadnienie. Przede wszystkim dokonamy podziału na wirtualizację oprogramowania i wirtualizację zasobów.

Pierwsze zastosowanie wirtualizacji dotyczy tworzenia oprogramowania udającego inny system operacyjny lub sprzęt komputerowy dla takowego. Takie programy wirtualizujące pozwalające na używanie oprogramowania przeznaczonego na inną platformę, tworzą dla niego izolację i pozwala uruchamiać inne systemy operacyjne na posiadanym. Zasadniczo sprowadza się więc to do umożliwienia wykorzystania oprogramowania, którego w normalnych warunkach nie możemy uruchomić. I w tym jednak istnieje kilka podejść:

- Emulacja (pełna wirtualizacja z dynamiczną rekompilacją)
  Program emulujący zazwyczaj udaje wyłącznie inny rodzaj sprzętu komputerowego bez jakiegokolwiek oprogramowania, czego efektem jest konieczność instalacji systemu operacyjnego na emulatorze. Oczywiście są wyjątki takie, jak choćby konsolę – tam wszak uruchamia się programy bez systemu operacyjnego. Bez względu na rozwiązanie, wywołania platformy lub innego oprogramowania są interpretowane przez emulator. Następnie są one zamieniane na wywołania sprzętu lub odwołania do bazowego systemu operacyjnego, cokolwiek by to miało znaczyć, bo przecież i on może być uruchamiany na maszynie wirtualnej. Warto dodać, że prędzej czy później emulatory będą pozwalać nam używać oprogramowania, do którego sprzęt może już fizycznie nawet nie istnieć.

- Wirtualizacja natywna (pełna wirtualizacja)
  Umożliwia uruchamianie systemu operacyjnego kompatybilnego z posiadanym systemem komputerowym i dokonuje wirtualizacji tylko części wywołań, potrzebnych do izolacji oprogramowania. Cała reszta jest przetwarzana przez fizyczny sprzęt, celem zwiększania wydajności wirtualizacji. Zwykle polega to na stworzeniu maszyny wirtualnej przechwytującej wszystkie wywołania, a następnie przekazywaniu wybranych instrukcji do systemu komputerowego. Przekazuje się przede wszystkim instrukcje wejścia/wyjścia i obsługi sieci.

- Parawirtualizacja (hipernadzorca)
  W tym przypadku oprogramowanie wirtualizujące nie tworzy iluzji sprzętu, a jedynie dostarcza API, z którego mogą korzystać odpowiednio zmodyfikowane systemy operacyjne. Jest to alternatywa szczególnie dla architektury x86, która jest trudna do pełnego zwirtualizowania, co przyczynia się do zmniejszenia wydajności działania oprogramowania.

- Wirtualizacja systemu operacyjnego
  To podejście polega na dzieleniu jednej fizycznej maszyny na kilka wirtualnej opartych o bazowy system operacyjny. Wszystkie platformy muszą być tego samego typu, co gospodarz, ponieważ obsługa sprzętu jest dokonywana przez jego jądro. Dopuszczalne są jednakowoż inne dystrybucje. Ten rodzaj wirtualizacji pozwala ograniczyć utratę wydajności, bo nie wymaga żadnego tłumaczenia w trakcie działania systemu – wywołania systemów wirtualizowanych są identyczne z wywołaniami rzeczywistego. Wykorzystuje się to rozwiązania szczególnie do konsolidacji serwerów i w dostarczaniu prywatnych serwerów stron internetowych.

- Wirtualizacja aplikacji
  Wirtualna maszyna zapewnia uruchamianej na niej aplikacji, która czasem nie musi być nawet wirtualizowana, wirtualne rejestry i dostęp do plików. Taka izolacja ma wyeliminować konflikty między aplikacjami oraz na linii aplikacja – system operacyjny, a także zapewnić większą przenośność.

Jak widać różne typy wirtualizacji nakładają na użytkownika coraz większe ograniczenia. Wszystko to ma na celu odnalezienie kompromisu między dostarczanymi usługami wirtualizującymi i wydajnością.

Drugi biegun technologii wirtualizujących polega na tworzeniu specyficznych warunków sprzętowych, do których akurat nie mamy dostępu. Prostym przykładem takiego narzędzia jest RAID, pozwalający postrzegać grupę dysków jako jeden lub narzędzia do obsługi sieci, dzięki którym kilka oddzielnych linii przemysłowych może udawać jedną o większej przepustowości. Wszelkiego rodzaju partycjonowanie zasobów może być również uznane za wirtualizację, w szczególności tyczy się to również tworzenia iluzji na poziomie sieci komputerowych. Wtedy grupa maszyn połączona siecią może być wirtualizowana do pojedynczego serwera.

W tym dokumencie zajmujemy się prawie wyłącznie pierwszym podejściem.

**Wsparcie sprzętowe**

Zajmiemy się tu tylko architekturą x86, która nie spełnia w swej pierwotnej formie kryteriów Popka i Goldberga. Dlatego dwie największe firmy zajmujące się produkcją procesorów zdecydowały się wprowadzić na rynek rozszerzone wersje swoich architektur. Miały one pozwolić na wirtualizację bez znaczącej utraty wydajności.

Intel w 2005 roku na Wiosennym Forum Rozwoju Intela zaprezentował rozszerzenie dla swoich architektur 32- i 64-bitowych zwane IVT (Intel Virtualization Technology) lub Vanderpool. Również dla procesorów Itanium została opublikowana nowa specyfikacja techniczna, choć wcześniej stosowana była inna, oznaczana kryptonimem Silvervale. Technologia ta ma być na tyle elastyczna, że może być deaktywowana na życzenie użytkownika.

AMD przygotowało rozszerzenia wirtualizacyjne tylko dla procesorów 64-bitowych, a używana technologia otrzymała nazwę Pacifica. W maju wprowadzono rozszerzone procesory Athlon 64 i Turion 64, w sierpniu miały pojawić się takież Opterony. Pacifica jest nadzbiorem instrukcji Vanderpoola, co nie oznacza, że są ze sobą kompatybilne.

Wszyscy najpoważniejsi gracze na rynku wirtualizacji: Microsoft, VMWare, Xen wprowadzili wykorzystywanie wsparcia sprzętowego w swoich produktach. Badania wskazują jednak, że póki co rozszerzenia architektury nie dają znaczącego wzrostu wydajności w stosunku do wirtualizacji samym oprogramowaniem.

Zastosowania wirtualizacji

Dzisiejsze zastosowania wirtualizacji są inne od tych, którym służyła na początku. Zwiększanie efektywności działania sprzętu komputerowego sprawiło, że wirtualizacji da się używać na co dzień:

- Konsolidacja serwerów
  Jeśli używa się wielu serwerów o niskim zapotrzebowaniu na moc obliczeniową, to stawianie każdego z nich na oddzielnej fizycznej maszynie jest nieopłacalne. Zamiast tego można użyć jednego komputera, a na nim trzymać wiele maszyn wirtualnych, z których każda obsługuje jeden serwer. Pozwala to na oszczędność miejsca, serwisowania, zarządzania, energii i sprzętu.

- Szkolenia
  Wirtualne maszyny stanowią skuteczną izolację przy szkoleniu nowych użytkowników, szczególnie gdy mają otrzymać wysokie uprawnienia. Dzięki wirtualizacji możliwie jest nawet udostępnienie uprawnień administracyjnych bez obaw o utratę danych lub konfiguracji.

- Symulacja sprzętu
  Maszyny wirtualne mogą z powodzeniem udawać sprzęt, którego fizycznie nie posiadamy i pozwalają prowadzić działalność w warunkach, którymi nie dysponujemy – od maszyn wieloprocesorowych, po sieci komputerowe.

- Bezpieczne środowisko
  Dzięki izolacji, którą zapewnia maszyna wirtualna, możliwe jest zupełnie bezpieczne uruchamianie potencjalnie niebezpiecznego oprogramowania. Tyczy się to zarówno programów, co do których nie mamy zaufania, jak i specjalnie przygotowanych scenariuszy działania systemu operacyjnego, których efekty chcemy poznać.

- Przywracanie działania
  W razie utraty serwera maszyna wirtualna może stanowić urządzenie zastępcze, na którym można uzyskać oczekiwaną funkcjonalność. Jeśli utracony zostaje serwer wirtualny, to staje się to jeszcze prostsze, bo wirtualne maszyny i ich konfiguracje są przechowywane w jednym katalogu. Wystarczy je skopiować i przenieść na inną maszynę.

- Emulowanie nieposiadanego sprzętu lub systemu operacyjnego
  Dzięki maszynom wirtualnym możemy uruchamiać przestarzałe systemy operacyjne, które nie są już kompatybilne z posiadanym sprzętem. Podobnież przestarzałe lub obce oprogramowanie może być uruchamiane na nieobsługującym go sprzęcie lub systemie operacyjnym.

- Przenośność
  Przenośność nie jest najmocniejszą stroną aplikacji, ze względu na mnogość i rozproszenie wykorzystywanych przez nie zasobów. Można jednak zatrudnić maszynę wirtualną do przechwytywania wywołań z aplikacji. Dzięki temu możemy zapewnić przechowywanie wszystkich tworzonych przez nią zasobów w jednym, łatwym do przenoszenia katalogu. Pozwala to na uzyskanie przenośności bez zmian w kodzie.

- Środowiska wykonawcze dla języków programowania
  Maszyny wirtualne dla Javy i .Net zapewniają możliwość uruchamiania oprogramowania na każdej (obsługiwanej) platformie. Jednocześnie kod jest wykonywany w ten sam sposób na każdym systemie operacyjnym.

## Wady wirtualizacji

Najpoważniejszą wadą tej technologii jest oczywiście spadek wydajności związany z koniecznością tłumaczenia rozkazów. Różne rozwiązania próbują obchodzić to ograniczenie, lecz efektem tego typu działań jest zawężenie zastosowań danej maszyny wirtualnej. Jednak w dobie coraz potężniejszych maszyn i tańszej pamięci komputer zwykle dysponuje pewnymi nadwyżkami, które można przeznaczyć na wirtualizację. To z kolei prowadzi do innego pytania – dziś praca komputera jest niezwykle tania, nie ma więc potrzeby dzielenia czasu procesora między wielu użytkowników. Trzeba sobie zdawać sprawę z tego, że to właśnie temu celowi służyły maszyny wirtualne na komputerach klasy Mainframe. Te były zaś niebotycznie drogie i nie można było pozwolić, by pozostawały bezczynne, podczas na przykład operacji wejścia/wyjścia lub interakcji z użytkownikiem. Dziś nie zależy nam na ograniczaniu bezczynności komputera, a na zwiększaniu jego responsywności, co jest dużo istotniejsze z punktu widzenia użytkownika.

Zupełnie odrębnym problemem jest kwestia architektury x86. Ona zwyczajnie nie nadaje się do wirtualizacji, szczególnie przy choćby operacjach wejścia/wyjścia. Jeśli dane zastosowanie ma się opierać właśnie na nich, to wirtualizacja z pewnością nie jest właściwą odpowiedzią.

Jeśli wirtualizacja oprogramowania i dzielenie zasobów na mniejsze części trudno uznać za faktycznie potrzebną dzisiejszym użytkownikom technologię, to może choć proces odwrotny jest godzien rozważenia. Tu jest dużo lepiej, wszak klastry funkcjonują nie od dziś i są dość powszechnie używane (w wyspecjalizowanych ośrodkach, rzecz jasna). Możliwość uzyskania wirtualnej maszyny o potężnej mocy obliczeniowej będącej przy tym prostą w obsłudze jest co najmniej kuszące i zapewne wiele organizacji oraz firm mogłoby skorzystać na tej technologii. Pytanie brzmi: do czego klaster przyda się (i jak ma go zbudować) przeciętny użytkownik albo firma spoza branży IT?

Wszystkie wady wirtualizacji można sprowadzić do jednego zagadnienia: czy jej zastosowania praktyczne są nam dziś do czegokolwiek faktycznie potrzebne? Na to każdy potencjalny użytkownik musi sobie odpowiedzieć sam.