Czym jest wirtualizacja?

Zestaw metod umożliwiający:

- Podział zasobów komputera
  na wiele niezależnych środowisk
- Ukrywanie charakterystyki zasobów sprzętowych
- Symulację innego niż obecny stanu zasobów

Przykłady

- Pamięć wirtualna (zasoby)
- Partycje na dysku twardym (sprzęt)
- Inny typ procesora (sprzęt)
- Maszyny wirtualne (języki oprogramowania)
- Wirtualne systemy operacyjne (np. EyeOS)

Zastosowania

- Uniezależnienie od ograniczeń sprzętu
- Wyższe bezpieczeństwo
- Możliwość uruchomienia starego oprogramowania
- Możliwość równoległej pracy na wielu systemach operacyjnych
- Wysoka przenośność oprogramowania

Podejścia do wirtualizacji

- Emulacja pełna
- Emulacja API
- Wirtualizacja
  - Para wirtualizacja
  - Pełna wirtualizacja

Emulacja pełna

- Polega na całkowitej symulacji sprzętu (CPU, RAM, HDD, urządzenia I/O) wraz z systemem operacyjnym.
- Wirtualny odpowiednik całego komputera.
- Znaczny spadek wydajności. Każda operacja jest emulowana.

Emulacja API

- Emuluje API danego systemu operacyjnego.
- Wyższa wydajność niż przy emulacji pełnej.
- Brak konieczności posiadania pełnego systemu operacyjnego pod którym działa program.
- Bywa problematyczna.
  - Problemy z kompatybilnością
  - Nie wszystkie programy działają

Para wirtualizacja

- System operacyjny wie, że jest wirtualizowany
- Współpracuje ze środowiskiem operacyjnym komputera w zakresie obsługi tych elementów sprzętowych, których obsługa kolidowałaby z działaniem innych środowisk wirtualizowanych.
- Modyfikowane jest jądro, aby korzystało z dostarczanego API

Wirtualizacja pełna

- Technika, w której wirtualizowany system operacyjny ma wrażenie że działa na prawdziwym, fizycznym sprzęcie.
- Operacje, które mogłyby powodować kolizje są przechwytywane przez oprogramowanie wirtualizacyjne, a następnie emulowane.
- Wymagana jest odpowiednia architektura.

Kryteria właściwego działania maszyny wirtualnej

Przedstawione przez Geralda Popka i Roberta Goldberga kryteria:

- Odpowiedniość
  Program uruchomiony na maszynie wirtualnej ma zachowywać się tak samo, jak na rzeczywistym sprzęcie
- Kontrola zasobów
  Wirtualna maszyna w pełni kontroluje wirtualizowane zasoby
- Wydajność
  Większa część instrukcji wykonywana bez udziału maszyny wirtualnej

Rodzaje instrukcji

- Uprzywilejowane
  Ich efektem jest przerwanie lub wywołanie systemowe w trybie użytkownika lub ich brak w trybie jądra
- Wrażliwe ze względu na kontrolę
  W trakcie wykonywania mogą zmienić konfigurację zasobów systemowych
- Wrażliwe ze względu na wykonanie
  Ich działanie jest zależne od konfiguracji systemu

Twierdzenie Popka-Goldberga

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

Problemy na procesorach x86

- Nie wszystkie instrukcje wrażliwe wywołują przerwania
- Architektura zaprojektowana dla jednego SO
  Problem ze stronicowaniem i segmentacją
- Obsługa poziomów uprzywilejowania
- Brak sprzętowych mechanizmów wykrywania instrukcji nie działających poprawnie w wirtualnym środowisku
- Trudny do wykrycia samo modyfikujący się kod

Rozwiązywanie problemów

- Modyfikacja kodu
  Umieszczenie kodu w pamięci i ustawienie flagi zabezpieczającej przed zapisem. Próba modyfikacji wywoła przerwanie, które obsłuży nadzorca.
- Instrukcje, których nie można bezpiecznie wykonać bezpośrednio
  Tłumaczenie i stosowanie zwykłej emulacji sprzętu
- Główne przyczyny problemów z wydajnością
  Parawirtualizacja oraz dynamiczna rekompilacja z wykrywaniem wrażliwych instrukcji

Zagrożenia związane z wirtualizacją sprzętową

Niewykrywalne szkodliwe oprogramowanie

- BluePill
- SubVirt
- Virtriol

Blue Pill

- Rootkit zaprojektowany w 2006 roku przez Joannę Rutkowską
- Docelowy system: Microsoft Windows Vista
- Wykorzystuje technologię wirtualizacji AMD Pacifica

Blue Pill

- Wg autorki może umieścić SO w maszynie wirtualnej procesora działając w trybie nadzorcy i pozostając niewykrywalny
- Rozgorzała dyskusja czy jest to możliwe
- Upubliczniono kod źródłowy

SubVirt

- Stworzony w laboratoriach Microsoft Research wraz z pracownikami Uniwersytetu Michigan
- Wykorzystując luki w SO instaluje monitor maszyny wirtualnej
- Działa na systemach Windows i Linux
- Niemożliwy do wykrycia – oprogramowanie nie ma dostępu do maszyny wirtualnej
- Każdy kod uruchomiony na zainfekowanym systemie pozostaje niewidoczny
- Jeden z autorów sam nie wiedział, że jego komputer został zainfekowany.

Jak wykrywać rootkity?

- Brak możliwości sprawdzenia, że procesor pracuje w trybie VMX non-root
- Próba użycia VMX do stworzenia VM
- Próba wykrycia „VM exit latency”
- Red Pill

Pytania?