Wirtualizacja QEMU

Spis treści:

Wstęp

Wybór tematu, który chcieliśmy opisać padł na narzędzie do wirtualizacji zwane QEMU. W dzisiejszych czasach wirtualizacja jest bardzo ważnym i przydatnym elementem pozwalającym na sprawdzanie działania różnych systemów operacynych zanim zdecydujemy się na ich bezpośrednią instalację na naszym komputerze lub np. testowanie tworzonych przez nas sieci bez konieczności posiadania w niej wielu fizycznych węzłów. Obecnie istnieje bardzo wiele różnych narzędzi do wirtualizacji, jak np. Bosch, PearPC, VirtualBox, VmWare czy chociażby Xen, jednak jak w porównaniu do nich wypana QEMU? W poniższym referacie postaramy się przedstawić czym jest program QEMU i jakie są jego charakterystyczne cechy.

Co to jest QEMU?

QEMU jest skrótem od Quick Emulator, a sam program jest jednym z wielu programów do wirtualizacji, jednak swoim działaniem znacząco wyróżnł się na tle konkurencji. QEMU poza wirtualizacją systemu służy głównie do emulacji działania procesorów o innej architekturze niż ta, na której został on uruchomiony. Jego autorem jest francuz o imieniu Fabrice Bellard. Bellard poza samym QEMU znany jest także z FFMpeg, Tiny C Compilatora czy Formuły Bellarda. Program wydany został na licencji GNU GPLV2 co oznacza, że można z niego korzystać za darmo i go rozpowszechniać w niezmodyfikowanej wersji, posiadamy całkowity dostęp do kodu i modyfikacji dla własnych potrzeb. Oficjalnie wiadomo, że QEMU dostępny jest dla następujących systemów: Linux, Windows, FreeBSD oraz Mac OS X, Solaris, OpenBSD, jednak ponieważ samo QEMU jest po prostu zbiorem bibliotek napisanych w C, to tak naprawdę będzie on działać na każdym systemie na którym da się go skompilować, przez co można w Internecie znaleźć informacje o tym jak np. uruchomić przy jego pomocy Windowsa XP na Androidzie.

Jak działa QEMU?

Wiemy już zatem czym jest i kto stworzył QEMU, jednak teraz postaramy się odpowiedzieć na pytanie w jaki sposób on działa i dlaczego obecnie jest uważany za jeden z najszybszych wirtualizatorów dostępnych w sieci.

Program ten wbrew pozorom nie pozostawał w tyle i na przestrzeni rozwoju wykorzystywał różne algorytmy i sztuczki do coraz to lepszego i bardziej optymalnego działania. Opis działania zaczniemy od trybów, w jakich może działać Qemu, a jest ich dwa: tryb użytkownika i tryb systemu.

Tryb użytkownika

Pierwszy z nich (tryb użytkownika) jest trybem na poziomie wirtualizacji aplikacji. Dzięki niemu

QEMU może uruchomić procesy skompilowane dla różnych procesorów na aktualnie uruchomionym procesorze, czyli np. możemy sprawdzić w jaki sposób wypada napisana przez nas aplikacja na procesorze o architekturze PowerPC, jeżeli sami posiadamy procesor x86.

Warunkiem działania tego trybu jest zgodność na poziomie systemu operacyjnego, co oznacza, że w tym trybie nie sprawdzimy jak działa nasza aplikacja na np. Windowsie, kiedy sami mamy zainstalowanego Linuxa. Niestety tryb ten aktualnie jest ograniczony tylko do procesów uruchamianych na systemach Linuxowych. Tryb użytkownika najczęściej jest wykorzystywany do sprawdzania działania pisanych aplikacji na różnych modelach procesorów, przez co lepszemu ich debugowaniu, bądź przeprowadzaniu różnego rodzaju benchmarków.

Tryb systemu

Drugi z trybów jest trybem na poziomie pełnej wirtualizacji platformy. Dzięki niemu QEMU może uruchomić całkowicie inny system operacyjny w trakcie działania naszego głównego systemu operacyjnego, co oznacza, że możemy bez problemu przetestować działanie np. Windowsa bez potrzeby instalacji go na naszym komputerze.

Dodatkowo poza emulacją konkretnego procesora możemy także emulować różne peryferia i tak w zależności od wybranej architektury procesora możemy dodatkowo emulować np. różnego rodzaju napędy, kartę graficzną, kartę sieciową (dzięki czemu możemy dodać kilka kart i łączyć je w podsieci), różnego rodzaju kontrolery bądź wejścia. Dzięki temu zabiegowi bez problemu możemy uruchomić niezmodyfikowany Mac OS X na zwykłym komputerze PC. Sam tryb systemu bardzo przydaje się do uruchamiania wielu różnych systemów operacyjnych bez potrzeby restartowania komputera w celu testowania tworzonych rozwiązań lub tworzonych sieci.

W jaki sposób Qemu umożliwia uruchomienie systemu operacyjnego bez potrzeby instalacji ich na naszym dysku twardym (poza oczywiście korzystaniem z Live CD)? Otóż tak jak większość innych programów do wirtualizacji, Qemu wykorzystuje obrazy twardych dysków zapisane w pojedynczych plikach. Taki obraz może być w postaci RAW (czyli np. 2GB dysk będzie 2GB plikiem) lub w postaci QCOW2 (początkowo zajmuje o wiele mniej miejsca od RAW i rozszerza je tylko wtedy, kiedy jest mu ono rzeczywiście potrzebne, jednak dane wczytywane są odrobinę wolniej). Inne formaty obsługiwane przez program do wirtualizacji to np. QCOW, COW, VMDK lub CLOOP.

Dynamiczna translacja

Znając już tryby, w jakich działa opisywany program pozostaje odpowiedzieć sobie na pytanie, czemu Qemu zawdzięcza swoją szybkość działania. QEMU swoje osiągi w emulacji działania różnych procesorów zawdzięcza algorytmom wymyślonym przez samego twórcę. Ponieważ same algorytmy są stosunkowo skomplikowane w działaniu postaramy się poniżej przedstawić ideę jaka im przyświeca. Początkowo wykorzystywanym rozwiązaniem był proces nazwany Portable Dynamic Translation (na potrzeby referatu będziemy go nazywać dynamiczną

translacją). Dynamiczna translacja jest procesem, w którym instrukcje z emulowanego procesora są konwertowane w czasie rzeczywistym na instrukcje rozumiane przez nasz fizyczny procesor. Aby osiągnąć powyższe rezultaty QEMU dzieli otrzymywane instrukcje na mikro operacje (napisane w języku C), które następnie są kompilowane w obiekty przy pomocy kompilatora GCC. Dzięki temu zabiegowi jesteśmy w stanie zbudować translator, w którym każda instrukcja emulowanego procesora ma swoje odzwierciedlenie w instrukcji (pod postacią mikro operacji) od naszego fizycznego procesora. Jednak to rozwiązanie nie satysfakcjonowało samego Bellarda, który nie był zadowolony z szybkości z jaką działa sam kompilator GCC.

Tiny Code Generator

Od wersji 0.9.1 QEMU przestał korzystać z dynamicznej translacji i zamienił ją na znacznie bardziej zoptymalizowaną wersję zwaną TCG. TCG, czyli Tiny Code Generator działa w bardzo podobny sposób, co dynamiczna translacja, jednak posiada on znaczące różnice. Instrukcje z emulowanego procesora są dzielone na bloki translacji, a następnie konwertowane do interpretera zwanego tcg-ops, który to następnie jest konwertowany na kod odpowiedni dla posiadanego przez nas procesora. Kod oczywiście następnie zostaje przez nasz procesor wykonany.

Dodatki

Ponieważ emulator QEMU został wydany na licencji GNU GPLV2, to oczywistym następstwem były dodatki tworzone przez zwolenników tego też programu. Poniżej chcielibyśmy przedstawić kilka ciekawych dodatków, które ułatwiają i usprawniają działanie samego programu. Pierwszym z nich będzie napisany przez samego twórcę akcelerator.

Akcelerator

Qemu pomimo swojej szybkości sam w sobie jest względnie wolny, jednak istnieje pewien sposób dzięki któremu możemy znacznie przyspieszyć jego działanie. Jeżeli architektura naszego procesora jest taka sama jak architektura emulowanego procesora, to możemy przyspieszyć szybkość działania do prawie stuprocentowej prędkości przy pomocy rozwiązań zwanych akceleratorami. Akceleratory pozwalają na wykonywanie kodu bezpośrednio na fizycznym procesorze. Początkowo do akceleracji QEMU wykorzystywano dodatek napisany przez Bellarda zwany KQEMU, który działał na większości wirtualizowanych platform z włączoną emulacją x86 bądź x86-64, jednak z powodów zmian w samym QEMU, które zaszły od wersji 0.12.0 samo KQEMU przestało być rozwijane i zostało zastąpione przez KVM. KVM jest skrótem od Kernel-based Virtual Machine i umożliwia on pełną wirtualizację wyłącznie na systemie Linux z podzespołami x86 (wymagane są rozszerzenia wirtualizacyjne Intel VT lub AMD-V). Obecnie QEMU odchodzi od wykorzystywania KVM na rzecz Upstream QEMU wykorzystując wirtualizator Xen.

QtEmu

Kolejnym z ciekawszych dodatków jest program zwany QtEmu. QtEmu jest to interfejs graficzny opierający się na QEMU. Znacznie ułatwia to pracę z samym QEMU, ponieważ ten obsługiwany jest przy pomocy komend z poziomu konsoli. Aplikacja ta jest dostępna dla następujących systemów operacyjnych: Windowsa, Linuxa i FreeBSD. Dzięki QtEmu możemy przełączać się pomiędzy kilkoma wirtualizowanymi systemami operacyjnymi przy pomocy wygodnych zakładek. Dodatkowo istnieje w nim możliwość ustawienia wielu różnych języków, w tym np. języka polskiego.

Qemudo

Kolejną ciekawostką, którą można zainstalować, aby wygodniej obsługiwać QEMU jest Qemudo. W działaniu przypomina on dodatek opisywany wyżej, jednak posiada on dosyć znaczącą różnicę. Otóż Qemudo jest interfejsem webowym do obsługi QEMU lub KVM, co mniej więcej oznacza, że dzięki niemu możemy zarządzać naszymi maszynami wirtualnymi z poziomu przeglądarki.

Podsumowanie

Widzimy zatem, że QEMU jest narzędziem bardzo rozbudowanym, wyróżniającym się mnogością opcji i szybkością działania, a rozwiązania w nim zastosowane zostały zaporzyczone w wielu innych programach do wirtualizacji. Wspomaga on testowanie działania sieci oraz programów na różnych konfiguracjach. Jego największe zalety to darmowa licencja, społeczność rozwijająca sam moduł i dodatki, ogromne możliwości oraz w niektórych przypadkach szybkość działania.

Bibliografia

Informacje do referatu oraz prezentacji zaczerpnięte z:
https://wiki.archlinux.org/index.php/QEMU
http://qemu.weilnetz.de/qemu-tech.html
http://en.wikipedia.org/wiki/QEMU
http://osworld.pl/qemu-emulator-procesora/
http://static.usenix.org/event/usenix05/tech/freenix/full_papers/bellard/bellard.pdf
http://www.claunia.com/qemu/old/index.php
http://osworld.pl/qemu-emulator-procesora/
http://qemudo.sourceforge.net/
http://qtemu.org/
http://www.ibm.com/developerworks/library/l-qemu/
http://wiki.qemu.org/Documentation/TCG

Obrazki wykorzystane w prezentacji (w nawiasach numer slajdu):
(2) http://true-wildlife.blogspot.com/2011/02/emu.html
(3) http://osworld.pl/wp-content/uploads/qemu-logo.png
(4) http://www.weblogit.pl/wp-content/vmware_fusion.jpg
(5) http://commons.wikimedia.org/wiki/File:2000px-ok_x_nuvola_green.png
(6) http://upload.wikimedia.org/wikipedia/commons/2/22/Spur_gears_animation.gif
(7) http://www.proginosko.com/wordpress/wp-content/uploads/question-mark.jpg
(8) http://www.mobo.pl/wp-content/uploads/2012/03/procesory1.png
(11) http://www.i-slownik.pl/446,dysk-twardy-lub-hdd-hd/
(12) http://jengbuas.files.wordpress.com/2011/01/puzzle.jpg
(13) http://www.greenlan.pl/uploads/media_items/bardzo-szybki-internet.630.204.s.jpg
(14) http://qtemu.org/preview_2.png
(15) http://qemudo.sourceforge.net/screen_vm.png