Ordu Üniv. Bil. Tek. Derg., Cilt:6, Sayı:1, 2016,19-23/Ordu Univ. J. Sci. Tech., Vol:6, No:1,2016,19-23

ÖNYÜKLEME SÜRESİ VE DİSK KULLANIMI AÇISINDAN SANAL MAKİNELER İLE LİNUX KONTEYNERLERİNİN KARŞILAŞTIRILMASI

Emre Can Yılmaz[1*], Recai Oktaş[1], Bünyamin Karabulut[1]

[1] Ondokuz Mayıs Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Samsun

Özet

Son yıllarda, uygulama konuşlandırma yöntemlerinde büyük değişiklikler gerçekleşmekte olup, bu değişikliklerden en önemlilerinden biri, sanal makine tabanlı konuşlandırma yerine konteyner tabanlı konuşlandırılmaya geçiştir. Bu çalışma kapsamında, popüler konteyner tabanlı sanallaştırma yöntemlerinden olan Docker ve Docker'a alternatif olarak çıkan Rocket'in, Linux işletim sistemi çekirdeği üzerinde KVM (Kernal Based Virtual Machine) ile çalıştırılmış sanal makinelerin başarımları ölçülmüştür. Ölçümlerde yapılan testler; disk kullanımı ve önyükleme süresi testlerini kapsamaktadır. Elde edilen test sonuçlarına göre, hangi tip sanallaştırmanın hangi şartlarda daha iyi başarım sağlayacağı, sistem yöneticilerine faydalı olacağı düşünülmektedir.

Anahtar Kelimeler: Bulut Bilişim; Sanal Makine; Linux Konteyner; Hipervizör; Sistem Yönetimi.

Comparison Of Virtual Machines and Linux Containers In Terms Of Boot Time and Disk Usage

Abstract

In recent years, some big changes are occurring in application deployment processes. One of the most important change is the transition from the virtual machines to the container-based virtualization. In this study, we have made full virtualization with container-based virtualization performance benchmarks on the Linux operating system kernel. For full virtualization, we used the KVM (Kernel based Virtual Machine) hypervisor. For container-based virtualization, we used Docker and Rocket, which is emerged as an alternative to Docker. For each of the specified environment; disk usage comparisons, boot speed tests were performed. The results of these tests are expected to help administrators pick the most appropriate virtualization technology for the problem at hand .

Keywords: Cloud Computing; Virtual Machine; Linux Container; Hypervisor; System Administration.

*ecylmz@bil.omu.edu.tr

1. GİRİŞ

Sanal makinelerin en yaygın kullanım alanı bulut bilişim teknoljileridir. Sanal makinelerin veri tabanı gibi yazılımları servis olarak sunabilmeleri, bulut platformlarının en önemli özelliğidir. Sanal makinelerin yaygın olarak kullanılmaya başlanması sonucunda, bu makinelerin çalışma zamanları hakkında çeşitli çalışmalar yapılmıştır (Matthews et al 2007; Padala et al 2007). Literatürde, bu çalışmaların yanı sıra sanal makinelerin başarımlarının artırılmasına yönelik çalışmaların da yapıldığı görülmektedir (Kivity et al 2007).

Sanal makinelerin, Linux işletim sisteminde son yıllarda yaygın olarak kullanılmasının yanı sıra, bu işletim sisteminde isim uzayı kullanımına ilişkin öneriler ortaya çıkmış olup, Linux konteynerleri için, imaj formatı ve inşa sistemi olarak Docker uygulaması geliştirilmiştir (Biederman E W 2006, Docker 2015). Docker'ın yaygın kullanılması ve imaj formatlarındaki standartlaştırma ihtiyacına ek olarak güvenilirliğinin artırılması ihtiyacı sonucu Rocket (rkt) uygulaması araştırmacılar tarafından geliştirilmiştir (CoreOS 2015).

Linux konteyner ve sanal makine teknolojilerinin, performanslarıyla ilgili işlemci, depolama ve ağ başarımı testleri gerçekleştirilmiş (Yılmaz E C & Oktaş R 2016). Bu çalışmada Linux konteyner ve sanal makinelerin önyükleme süreleri, imaj boyutları ve disk kullanım kıyaslamaları gerçekleştirildi. Burada hedeflenen temel husus birbirinden bağımsız hale getirilmiş, sanal makine platformlarından biri olan Kernel Virtual Machine (KVM) ile Linux konteynerlerinden yaygın olarak kullanılan Docker ile Rocket'in önyükleme süreleri, imaj boyutları ve disk kullanımları karşılaştırılmıştır.

Linux konteynerler, üzerinde çalıştığı işletim sisteminin çekirdeği ile birlikte çekirdek isim uzayı özelliğini kullanmakta olup, Docker ve Rocket'in kullanımını yaygın hale getirmektedir.

2. MATERYAL VE YÖNTEM

Bu çalışmada materyal olarak Linux işletim sistemi üzerinde çalışan KVM, Docker ve Rocket uygulamaları kullanılmıştır. Bunun sebebi, sanal makinelerin, işletim sistemi içerisinde çalışan bir süreç olmasıdır.

Yöntem olarak başarım değerlendirmede sanallaştırma teknolojileri arasında iş yükü durumları incelenmiş olup sonucu araştırılmıştır. Bu iş yükü başarımı çizelge 1'de verilen donanım ve çizelge 2'de verilen yazılımlarla gerçekleştirilmiştir. Ayrıca önyükleme süresi başarım testi ve imaj boyutları ve disk kullanımı başarım testleri gerçekleştirilerek, sonuçları yazılan betik kodlar ile (GitHub 2016) elde edilmiştir.

Çizelge 1. Donanım Bilgileri

İşlemci	Bellek	Disk
Intel® Core™ i5-4570T CPU @ 2.90GHz × 4	12 GB DDR3	120 GB SSD Samsung Evo 850

Çizelge 2. Yazılım Bilgileri

İşletim Sistemi	Çekirdek Sürümü	Docker	Rocket	QEMU	libvirt
Ubuntu 14.04(Trusty) 64 bit	V3.13.0-62	V1.8.2	V0.5.5	V2.0.0	V1.2.2

2.1 Önyükleme Süreleri

Sanal makinelerin çalışma mekanizması gereği, disk imajından bir işletim sistemini belleğe yükleyip init mekanizmasını çalıştırması beklenmektedir. Bu da önyükleme süresinin ciddi anlamda artması anlamına gelmektedir. Gerek girdi-çıktı işlemleri gerek bellek hızı bu sürenin artmasına neden olmaktadır. Sanal makinelerin aksine, konteynerlerde tüm bir işletim sistemini belleğe yüklemek ve init sistemini çalıştırmak yerine, üzerinde çalıştığı sistemin çekirdeğini, isim uzayları, cgroups gibi araçları kullanarak konteyner üzerinde çalışacak süreci belleğe yüklemektedir.

2.2 İmaj Boyutları ve Disk Kullanımı

KVM, her bir sanal makine için tüm işletim sistemini barındıran bir imaj kullanmaktadır. Yani aynı işletim sistemi dosyaları her bir sanal makine içerisinde tekrar etmektedir. Bu tekrarlı veri, mevcut depolama alanında israfa yol açmaktadır. Buna karşılık, Docker dosya sistemi olarak "Union File System(UFS)" kullanmakta olup, kullanıcı tarafından oluşturulabilen katmanlı bir yapıya sahip olduğundan hafif ve hızlıdır (Docker 2016). Bunun aksine Rocket'te ise standart bir katman yapısı mevcuttur (Stage 0, Stage 1, Stage 2).

3. BULGULAR VE TARTIŞMA

3.1 Önyükleme Süreleri

Çizelge 3'te görünen sonuçlar Docker'ın milisaniyeler düzeyinde başlatma-durdurma sürelerinin olduğunu göstermektedir. Aynı işlem için Rocket'de bu süreler daha fazladır. Bunun nedeni, Docker'da, "Docker Daemon Engine" bulunması konteynerlerin açılış hızlarını milisaniyeler düzeyinde tutarken, Rocket'de böyle bir yapının olmaması,

doğrudan ikili dosya sistemini çalıştırması, önyükleme süresinin artmasına neden olmaktadır. Sanal makineler ise beklenildiği üzere en fazla önyükleme süresine sahip olduğu çizelge 3'te gözükmektedir.

Çizelge 3. Platformların Önyükleme Süreleri

Platform	Docker	Rocket	KVM
Yükleniş Süresi	0.81 saniye	18.01 saniye	24.2 saniye

3.2 İmaj Boyutları Ve Disk Kullanımı

Çizelge 4 göz önüne alındığında, üretilen 10 konteyner için disk alanı "(10 * imaj boyutu)" değil, sadece konteyner içerisinde üretilen veri miktarı kadardır. Bunun anlamı, sanal makinelerde varolan depolama alanı israfının Docker'da bulunmadığı görülmemektedir. Docker, üretilen her bir konteyner için sistemde salt-okunur halde duran imajların üzerine o konteyner tarafından yazılıp okunabilir bir katman eklemektedir. Rocket ise dosya sisteminde "Stage 0, Stage 1, Stage 2" denen aşamalar kullanan bir yapıya sahiptir. Bu yapılar birleşerek bir imaj dosyası meydana getirmektedir. Bu imaj dosyası sadece dizin yapısı, kitaplıkları ve kullanılacak ikili dosyaları barındırmaktadır ancak işletim sistemi çekirdeği bulundurmamaktadır. Bu nedenlerden ötürü, konteyner imajlarının boyutu, sanal makineler kadar çok depolama alanı tüketmemektedir. Platformların kullanmış olduğu imaj boyutları çizelge 4'te verilmiştir. Bu değerler, platformların tasarımlarının sonuçlarına göre beklenen sonuçlardır.

Çizelge 4. Platformların İmaj Boyutları

Platform	1 imaj için tipik boyut	10 imaj için tipik boyut
Docker	253 MB	253 MB (yaklaşık)
Rocket	90 MB	90x10 = 900 MB
KVM	773 MB	773x10 = 7730 MB

4. SONUÇLAR

Sanallaştırma platformlarının depolama alanı kullanımı kıyaslandığında, konteyner teknolojilerinin sanal makinelere göre bariz bir üstünlüğü bulunduğu gözlenmektedir (Çizelge 4). Konteyner teknolojilerinin kullandığı katmanlı yapıdan dolayı, depolama alanı israfları minimuma indirgenmektedir. Konteyner teknolojilerinin aksine sanal

makine teknolojileri, her bir sanal makine için tam işletim sistemi imajını barındırmasından dolayı tekrarlı veriye sebep olmaktadır. Bunun sonucu olarak da depolama alanı israfı meydana gelmektedir.

Sanal makineler, çalıştırdıkları işletim sistemini belleğe yüklerken, konteynerler sadece içerisinde çalışacak süreci belleğe yüklemektedir. Bunun sonucu olarak, konteynerlerin önyükleme süreleri, sanal makinelere göre daha azdır (Çizelge 3). Bu sonucun yanı sıra depolama alanında olan israf, bellek tüketiminde de söz konusu olmaktadır. Konteynerler, belleğin verimli kullanılması açısından, sanal makinelere karşı bariz şekilde üstünlük sağlamaktadır.

Bu çalışma kapsamında yapılan karşılaştırmalarda, bir tane sunucu üzerinde tek bir sanal makine ve tek bir konteyner çalıştırılmıştır. Günümüzde sunucular daha farklı kullanılmaktadır. Bulut platform sağlayıcıları, sahip oldukları sunucuları daha verimli kullanabilmek için daha küçük birimlere bölmektedir. Sağlayıcılar, bir fiziksel sunucu üzerinde sunucunun özelliklerine göre yüzlerce sanal makine, binlerce konteyner barındırabilir. Ayrıca bulut platform sağlayıcılarının kullandıkları sunucuların, bu çalışmada kullanılan sunuculardan çok daha yüksek başarıma sahip olması gerekmektedir. Tüm bu nedenlerle, gerçek hayatta kullanılacak bir sunucu, bu çalışmada kullanılan sunucudan çok daha fazla kaynak barındıracağından, çalışmada ortaya çıkan sayısal değerler sanallaştırma teknolojileri arasındaki başarım farklılıklarını ortaya koyacak şekilde oransal olarak yorumlanmalıdır.

KAYNAKLAR

Biederman E W (2006). Multiple instances of the global Linux namespaces. In Proceedings of the 2006 Ottawa Linux Symposium

CoreOS (2015). Rocket – https://coreos.com/blog/rocket/ (Erişim tarihi: 05.08.2015)

Docker (2015). Solomon Hykes and others. What is Docker? https://www.docker.com/whatisdocker/ (Erişim Tarihi: 05.08.2015)

Docker (2016). Union File System https://docs.docker.com/engine/ reference/glossary/#union-file-system (Erişim tarihi: 01.02.2016)

GitHub (2016). https://github.com/ecylmz/docker-rkt-kvm-comp/ (Erişim tarihi: 03.03.2016)

Kivity A, Kamay Y, Laor D, Lublin U & Liguori A (2007). KVM: the Linux virtual machine monitor. In Proceedings of the Linux Symposium, Ottawa, Ontario, Canada, volume 1, pp. 225-230

Yılmaz E C & Oktaş R (2016). Sanal Makineler ve Linux Konteynerlerin Performans Karşılaştırması, http://ab.org.tr/ab16/ozet/47.html, XVIII. Akademik Bilişim Konferansı Adnan Menderes Üniversitesi, Aydın, Türkiye

Matthews J N, Hu W, Hapuarachchi M, Deshane T, Dimatos D, Hamilton G, McCabe M & Owens J (2007). Quantifying the performance isolation properties of virtualization systems. In Proceedings of the 2007 Workshop on Experimental Computer Science, ExpCS ’07

Padala P, Zhu X, Wang Z, Singhal S, Shin K G, et al (2007). Performance evaluation of virtualization technologies for server consolidation. HP Labs Technical Report