

CONTAINER

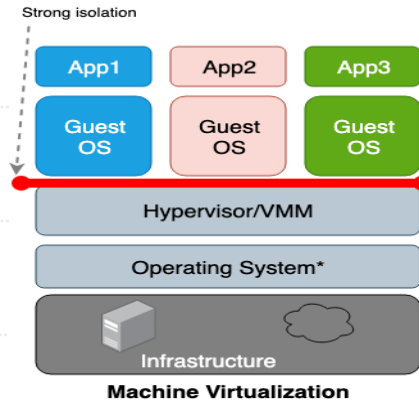
Ahmet Cemalettin Kumru

İnteraktif ve Web Projeleri – Yazılım
Geliştirme Stajyeri



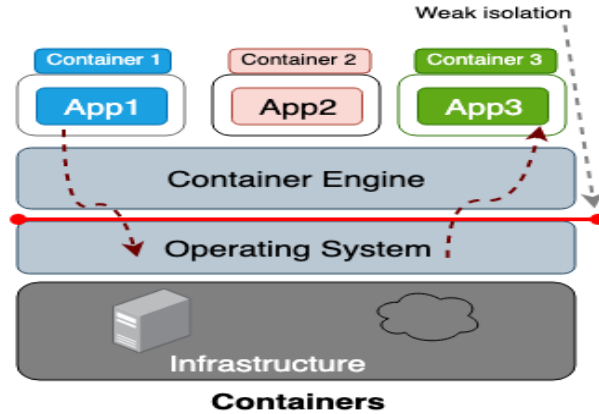
Sanallaştırma Nedir?

- Sanallaştırma teknolojisi, bir ara katman yazılım (hypervizör) kullanarak aynı fiziksel kaynaklar üzerinde birden fazla işletim sisteminin yönetilmesine imkan sağlayan bir teknolojidir. Bu teknoloji ile birlikte fiziksel kaynakların daha verimli kullanılması sağlanmıştır.
- Hypervizör, fiziksel donanım üzerinde oluşturulacak olan sanal makineler için kaynaklar oluşturulmasını ve yönetilmesini sağlayan bir yazılımdır. Bu ara katman yazılımı ile kullanıcıların direk donanımlara erişimi kısıtlanmış olur. Kullanıcıdan gelen talepleri uygun yöntem ve metodlarla donanıma iletmekle görevlidir.
- Bir yüksek kapasiteli sunucu üzerine kurulan sanal bilgisayarlardır. Her sanal bilgisayar sunucunun kaynaklarını kullanır ve içerisine farklı işletim sistemleri, uygulama ve uygulama gereksinimleri bulunmaktadır. Aşağıdaki görselde bir sanallaştırma altyapısı bulunmaktadır. Üç adet sanal makine bir sunucu üzerine kurulmuştur.



Container Nedir?

- Container yani kapsayıcılar, sanal makinelerle benzer rol oynayan bir uygulama dağıtım teknolojisidir. Container'lar da sanallaştırma gibi uygulamalar için yalıtılmış ortamlar sağlar. Ancak altyapı kaynaklarını bölmek için farklı bir yöntem kullanır.
- Virtual Machine'ler işletim sistemlerini taklit etmek için bir hiper yönetici kullanırken, kapsayıcılar ana bilgisayarın işletim sisteminin çekirdeğini diğer kapsayıcılarla paylaşır.
- Containerization, bir sunucu üzerindeki işletim sisteminin diğer containerler tarafından paylaşılması demektir. Birbirinden farklı uygulamalar, dağıtımlar ve envrionmentleri birbirinden izole şekilde çalıştırır. Aşağıdaki görselin bir Linux ubuntu sunucusu olduğunu varsayalım, ortamlar birbirinden farklı dağıtımlara (Fedora, CentOs, Debian), farklı environmentlere ve kütüphanelere sahip olabilirler. Dockerhub üzerinden istenen imageler indirilir ve Docker sunucu sistemindeki kerneli kullanarak belirtilen dağıtım üzerinde uygulamamızı çalıştırır.




Sanal Makineler ve Container


Sanallaştırma

- Kaynak kullanımı çok fazladır.
- Onlarca GB boyutundadır.
- Ana bilgisayar donanımını bölerek sanallaştırır.
- Başlatılması çok zaman alır.
- Sıfırdan kurulumu çok zaman alır.
- Donanım kaynakları düzeyinde sanallaştırma uygulanabilir.
- Sanal bilgisayarlar birbirinden tamamen izole şekildedir.
- Dağıtımı zordur.
- Farklı ortamlarda çalışmama sorunu ortaya çıkabilir.

Konteynerleştirme

- Kaynak kullanımı çok düşüktür.
- MB boyutundadır.
- Ana Bilgisayar donanımını paylaşır.
- Başlatılması çok kısa sürer.
- Kurulması saniyeler alır.
- Onlarca konteyner oluşturulabilir.
- Konteynerler işletim sistemi seviyesinde izole edilirler.
- Dağıtımı kolaydır.
- Her ortamda çalışacak durumda tutar.

- 
- VM'ler, normal çalışma koşulları altında ana sistemi veya diğer VM'leri etkileyemeyen tamamen farklı bilgisayarlar olarak çalıştırıldığından, sanal makineler mükemmel yalıtım ve güvenlik sunar. Ancak, dezavantajları da vardır. Örneğin, tüm bir bilgisayarı sanallaştırmak, VM'lerin önemli miktarda kaynak kullanmasını gerektirir. Genel olarak sanal makineler, bir makinenin kaynaklarını daha küçük, bireysel bilgisayarlara bölmenize izin verir.
 - Konteynerler farklı bir yaklaşım benimser. Tüm bilgisayarı sanallaştırmak yerine, kapsayıcılar işletim sistemini doğrudan sanallaştırır. Ana bilgisayar işletim sisteminin çekirdeği tarafından yönetilen özel süreçler olarak çalışır, ancak sistem süreçleri ve kaynakları, ortamının kısıtlı ve yoğun bir şekilde manipüle edilmiş bir görünümü ile çalışır. Konteynerler, paylaşılan bir sistemde var olduklarının farkında olmazlar ve sanki bilgisayarın tam kontrolündeymiş gibi çalışırlar.

- 
- Kapsayıcılar, uygulamayı temel alınan altyapıdan ayırır. Bu, geliştiricilerin çabalarını barındırılacak ortam yerine kod yazmaya odaklayabildikleri için hayatı kolaylaştırır.
 - Örneğin, her sunucu işletim sisteminin aynı Linux çekirdeğini (veya kapsayıcı ortamıyla uyumlu olanı) kullanması koşuluyla, kapsayıcıları farklı yapılandırmalara sahip farklı sunucularda çoğaltabilirler. Bu, bir kodlayıcı ekibinin, her birinin kullandığı ana bilgisayar ortamından bağımsız olarak bir proje üzerinde işbirliği içinde çalışmasına olanak tanır.

Temel Container Kavramları

- **1.Container Engine:** Kapsayıcılarınızı oluşturmak, çalıştırmak ve yönetmek için ana makinenize yüklediğiniz uygulamadır. Kurulumunuzun özüdür ve konteyner sisteminizin diğer tüm bileşenlerini bir araya getirir.
- **2.Container Image:** Gerçek çalışan kapsayıcılar oluşturmak için salt okunur bir şablondur. Tam olarak çalışır durumda bir kapsayıcı ortamını yapılandırmak için gereken tüm temel öğeleri bir araya toplayan bir dosya koleksiyonundan oluşur.Görüntüyü oluşturan dosyaların her biri katman olarak bilinir. Bu katmanlar, aşamalı olarak birbiri üzerine inşa edilmiş bir dizi ara görüntü oluşturur. Kapsayıcı görüntüleri salt okunur olduğundan, geleneksel sunucu modelinde çok yaygın olan yapılandırma kayması gibi sorunlardan kaçınarak modern değişmez altyapının tüm güvenilirliğini ve tutarlılığını sunar. Container ortamlarınızda değişiklik yapmanız gerekiyorsa, önceki resminizi güncellenmiş bir resimle değiştirin ve yeni kapsayıcılarınızı bundan başlatın.

- **3.Parent Image:** Kapsayıcı görüntünüzün ilk katmanı ve başlangıç noktasıdır. Konteyner ortamlarınız için temel yapı taşlarını ve diğer tüm katmanların üzerine inşa edildiği temelleri sağlar. Parent image, genellikle sadeleştirilmiş bir Linux dağıtımdır. Ancak, bir uygulama framework'ü veya kullanıma hazır içerik yönetim sistemi (CMS) gibi tam bir uygulama yığını da olabilir. Docker Hub veya Google Container Registry gibi bir kapsayıcı kayıt hizmetinden önceden yapılandırılmış parent image'ları içe aktarabilirsiniz. Alternatif olarak, kendi mevcut resimlerinizden birini de kullanabilirsiniz.
- **4.Base Image:** Temel olarak kapsayıcı görüntülerinizi sıfırdan oluşturmanıza olanak tanıyan boş bir ilk katmandır. Temel görüntüler genellikle, görüntülerinin her parçası üzerinde tam kontrol sahibi olmak isteyen daha ileri düzey kullanıcılar için tasarlanmıştır.
- **5.Konteyner:** Bir kapsayıcı görüntüsünün canlı bir örneğidir. Çalışan bir kapsayıcı temel olarak aşağıdakilerden oluşur:
 - Başlatıldığı görüntü.
 - Kapsayıcı katmanı olarak bilinen ve çalışma süresi boyunca kapsayıcıda yapılan değişiklikleri depolamak için kullanılan, yazılabilir bir üst katman.
 - Bu katmanlı yapı, benzer kapların her biri kendi bireysel durumunu korurken aynı temel görüntüye erişimi paylaşmasına izin verdiği için kapsayıcı verimliliğinin anahtarıdır.



Docker Nedir ?

- Docker, open source bir container teknolojisidir. Docker aynı işletim sistemi üzerinde bulunan birbirinden bağımsız ve izole yüzlerce container sayesinde sanallaştırma sağlar.
- Docker, uygulamaların farklı ortamlarda çalıştırılması sürecinde kullanılan bir platformdur.
- Uygulamamızı derler, ölçeklendirir, paketler ve dağıtmamızı sağlar.
- Docker içerisinde barındırdığı container yapısını ve Docker Hub üzerinden indirilen imageleri kullanarak ürünü farklı işletim sistemi, versiyon ve environmentlerle birbirinden izole şekilde çalıştırır; yeterli kaynak ve gerekli yapılandırmalar ile birlikte kullanıldığında, yapısal problemleri ortadan kaldırarak uygulamamızı stabil bir şekilde çalışabilecek hale getirir.