

## REPORT REPRINT

# A room with a view: a non-tech explanation of containers and Kubernetes

**MARCH 17 2021**

**By Owen Rogers, William Fellows**

Virtualization, containers and Kubernetes are big topics in tech, but the differences between them aren't always clear. Here, we present a simple analogy to aid understanding for a non-tech audience, and consider the role of these topics in a multicloud future.

---

THIS REPORT, LICENSED TO VMWARE, DEVELOPED AND AS PROVIDED BY 451 RESEARCH, LLC, WAS PUBLISHED AS PART OF OUR SYNDICATED MARKET INSIGHT SUBSCRIPTION SERVICE. IT SHALL BE OWNED IN ITS ENTIRETY BY 451 RESEARCH, LLC. THIS REPORT IS SOLELY INTENDED FOR USE BY THE RECIPIENT AND MAY NOT BE REPRODUCED OR RE-POSTED, IN WHOLE OR IN PART, BY THE RECIPIENT WITHOUT EXPRESS PERMISSION FROM 451 RESEARCH.

451 Research

---

**S&P Global**

Market Intelligence

### Introduction

Virtualization, containers and Kubernetes are big topics in tech, but the differences aren't always clear. Here, we present a simple analogy to aid understanding for a non-tech audience, and consider the role of these topics in a multicloud future.

### 451 TAKE

Containers are now a fundamental component of IT infrastructure: 53% of enterprises have at least some adoption today, with just 5% having no plans to implement, according to 451 Research's Voice of the Enterprise: DevOps, Organizational Dynamics 2020. Meanwhile, 43% of enterprises are using Kubernetes to, at least at some level, to manage their IT estates. The crucial benefit of containers is that applications can be decomposed into self-managed components that can live for as long or short as needed, depending on the demand at that time. These components can be updated independently across many physical servers without needing to rebuild the whole application, and components can be shared by multiple applications.

Orchestration platforms such as Kubernetes perform management of containers across many hosts, duplicating containers when needed to handle more requirements, and then scaling back down to save resources when not needed. Because of this, they are a logical enabler of multicloud applications. Being able to update and scale apps so users are always getting the best experience can make the difference between winning business and losing opportunities. In the post-COVID-19 era, the online experience has never been more important.

### Virtualization

Imagine a building with a number of floors – this represents a server in our analogy. An owner rents it out to a single tenant, but the tenant must pay a lot of rent for the whole building, even though they don't use it all. The landlord is only able to collect rent from a single person.

The owner could break the building into a number of apartments, all sharing the same physical building. Each apartment is a fully self-contained living space. Here, the landlord gets multiple revenue streams, and each tenant pays less than if they were to rent the whole house. This is akin to 'virtualization' in cloud computing, with tenants representing applications that are all hosted within the same building (server).

Virtualization enables the most basic attribute of a cloud service: a massive amount of computing or storage resources can be applied to many users simultaneously, with each user utilizing that service without regard to what other users are doing. Similarly, those users should be able to utilize the service without having to worry about the details of hardware-level implementation. Cloud suppliers virtualize compute, storage and other services on a massive scale using hardware virtualization, where a hypervisor layer (the abstraction layers) that runs on it enables operating systems (and their applications) to share hardware resources such as compute, storage and memory from a single asset, be it a server or even a pool of servers.

Originally, one server meant one operating system, typically delivering one workload. Through virtualization, one server (and its 'host' operating system) can hold multiple 'guest' operating systems, each one operating a logically separated workload, deployed together and contained within so-called virtual machines (the VM is the 'unit' of work in virtualization). Before virtualization, perhaps only a tiny fraction of the asset (the server and its resources) would be used at any one time. Through virtualization, multiple applications can be multiplexed together, so that resources are shared, and the asset is fully used. VMs contain a guest operating system, application payload, and anything else that may be needed (such as a database).

Common hypervisors include VMware ESX, Microsoft Windows Server, Citrix Xen, Red Hat Enterprise Virtualization and Linux KVMs. Cloud providers often have their own virtualization software. Increasingly, the hypervisors are less valuable than the software used to manage hypervisors across large numbers of servers, which allow virtual machines to rapidly be spun up and down, and configured to be resilient and performant – the essence of cloud.

### Containers

Back to our apartment building. Alternatively, the landlord could just break it up into a number of bedrooms, with a bathroom and kitchen shared among all tenants. In this model, the landlord is squeezing the most from the building, and the tenants are paying the least. Furthermore, a landlord that needs to perform maintenance on the kitchen or bathrooms only has to do it once, and will still satisfy all tenants. This is akin to 'containerization,' where each application isn't just sharing the physical server (the building), it is sharing code that is common across other applications (represented by tenants sharing some rooms). This code can be swapped out quickly across all applications at once, just like the kitchen can be repainted to satisfy all tenants. And if a room is unoccupied, the other tenants can take over that space temporarily, but can also vacate it quickly when a new tenant wants to move in.

The landlord in this case likes the predictability of tenants that have signed a contract to pay rent every month for a term commitment; and the tenants like knowing they have a roof over their head for the foreseeable future. But if there is a steady stream of people who only need a room for a few nights, the landlord might decide to turn the building into a hotel. Here, guests can stay for as long and as short as they need, and can rent as many rooms as they require. Containers can be spun up and consumed for a matter of seconds, to provide immediate capability where needed, but can be rapidly turned down to free up resources for other containers.

Container technology is essentially operating system virtualization – workloads share operating system resources such as libraries and code. Containers have the same consolidation benefits as any virtualization technology, but with one major benefit – there is less need to reproduce operating system code. Hardware virtualization means each workload must have all its underlying operating system technology. If the operating system takes up 10% of a workload's footprint, then in a hardware virtualized platform, 10% of the whole asset is spent on operating system code. This is regardless of the number of workloads being run. In the same environment utilizing containers, the operating system only takes up 10%, divided by the number of workloads.

In this case, a server runs 10 workloads, but only one operating system in the container environment. In the virtualized environment, the server would be running 10 workloads and 10 operating systems. An application container consists of an entire runtime environment: an application plus all of its dependencies, libraries and other binaries, as well as the configuration files needed to run it, bundled into a virtual container that can run on a variety of infrastructures – bare metal, traditional datacenter, virtual environment, or public, private or hybrid cloud. Application containerization represents a method of deploying software that is immune to changes in the underlying computing environment.

The software that actually runs these containers is called a container runtime, and these include Docker, containerd and CRO-O. The Open Container Initiative is an industry project to standardize container runtimes based on Docker, with partners that include AWS, CoreOS, Docker, Google, IBM, HP, Microsoft, VMware, Red Hat and Oracle. As with virtualization, these runtimes are less differentiated and valuable than the orchestration platforms that manage containers across multiple servers, and even clouds.

### Kubernetes and orchestration

At our hotel is a reception desk, staffed by a receptionist. The receptionist is responsible for keeping track of who is in what room, and allocating free rooms to new guests. Without the receptionist, the hotel would be static and underutilized. In containers, Kubernetes is the receptionist that manages the lifecycle of containers and automates the deployment, scaling and management of containerized applications.

Kubernetes is a standard today for container orchestration, due to its extensibility, powerful configuration options and ability to manage workloads independent of underlying infrastructure. It is an open source, container orchestration system for automating application deployment, scaling and management. Kubernetes was originally designed by Google, but wasn't the flagship project of the Cloud Native Computing Foundation (CNCF). It load balances the application load, just as the receptionist would allocate a block booking across many rooms, and monitors resource consumption so the hotel isn't overbooked. Kubernetes also allows new resources to be added, and can redistribute containers to different hosts should resources struggle.

Kubernetes 'clusters' are groups of machines referred to as nodes that are responsible for running containerised applications. A 'pod' is a single node in this cluster with one or multiple containers deployed onto it. Each node in a cluster can service pods or machines to run. Kubernetes can automate the management of clusters to perform at a desired state.

Many vendors are building Kubernetes compatibility into their own orchestration platforms, including Docker, Red Hat OpenShift, VMware Tanzu, IBM, Rancher Labs, Mirantis and Morpheus Data. Cloud providers, too, have created cloud services with support for Kubernetes, including AWS's Elastic Kubernetes Service, Google Kubernetes Engine, and Microsoft's Azure Kubernetes Services. There are also offerings from IBM, Oracle, Alibaba and OVH. But not everything is Kubernetes based: AWS's Elastic Container Service and Azure Container Instances are just two examples of container platforms that don't fit the mould.

### Multicloud

If a receptionist can manage rooms in a hotel, why can't a centralized receptionist handle rooms at a range of locations, balancing capacity and availability across cities far apart? Kubernetes and containers are being touted as an enabler of multicloud deployments – they provide a standard mechanism of managing and updating applications, regardless of the underlying infrastructure platform. This can help remove the risk of lock-in, because containers can be moved between venues without needing to be rewritten, and their lightweight nature (due to the sharing of code) means they can be moved from A to B far quicker than heavy-duty virtual machines.

## REPORT REPRINT

However, it is unlikely we will see all applications being decomposed to containers. Just like the property and leisure markets sustain houses, apartments, hotels and hostels, there will be demand for physical hosts, virtual machines and containers depending on specific requirements. Some enterprises are happy to pay for a whole physical host, knowing it is isolated and has full access to physical resources. Others might prefer to share resources but isolate code, like in a virtual machine, while others might have the appetite to abstract apps as far as possible into containers. Lots of applications are still monolithic, and slimming them down to VM size or breaking them into containers isn't an easy task. And – post-pandemic – many enterprises will struggle to find the money and the motivation to rebuild applications from scratch, rather than just move them to more powerful servers.

Anyway, it is often the case that physical, virtual and containerized apps run nested and hand-in-hand. Containers can be deployed on virtual machines, providing flexibility with the benefits of isolation. And no one said containers and Kubernetes would be easy. Virtual machines have reached a point of maturity where they are easy to deploy and easy to manage. Right now, containers are a hodgepodge that are best deployed and managed by experts. This will change over time, but for now, the container hotel is open for business – just make sure your receptionist knows what they're doing.