Kubernetes for Executives

Presented by: VMware

VMware Tanzu

Table of Contents

VMware Tanzu

Executive Summary

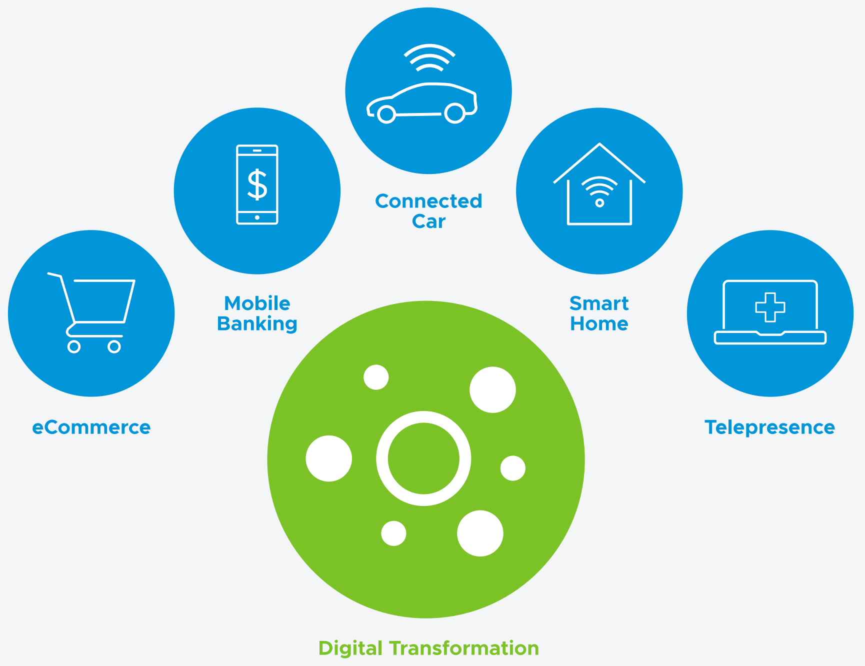
Because business success increasingly depends on a company's ability to deliver digital services and software quickly, enterprises are turning to containers and Kubernetes as essential tools for building, deploying, and running modern applications at scale.

Containers encapsulate applications and make them portable. Kubernetes works at a higher level as a container management system, making it possible to run complex, large-scale applications across multi-cloud environments. By providing uniform deployment, management, scaling, and availability services for applications—Kubernetes offers significant advantages for your IT and development efforts.

Software developers benefit because they can access the resources they need when they need them with no delays. Operators benefit because Kubernetes simplifies multi-cloud operations and eliminates time-consuming and error-prone manual tasks. Your business benefits because the IT organization as a whole becomes more responsive to business needs.

Kubernetes has a flexible architecture that is inherently extensible. Kubernetes operators embody the knowledge and logic traditionally captured in run books, allowing you to more easily deploy and manage large-scale applications on Kubernetes. This eBook acquaints you with some key concepts that all IT leaders should understand.

Modern apps power digital transformation



Containers and Kubernetes: Why You Should Care

For IT executives, the past few years have been particularly busy ones. Your IT and development teams are up to their elbows with digital transformation and working hard in the face of what seems like a mind-boggling pace of technology innovation. It's tough to keep up.

No matter what industry you're in, business success now depends on your company's ability to deliver digital services and software more quickly and with higher quality—and to operate those apps and services reliably at scale. Traditional methods of software development and deployment simply don't deliver new applications quickly enough or achieve the necessary cadence of new features and improvements.

The answer for most enterprises is agile software development methods (sometimes referred to as DevOps) and cloud native application architecture. If your organization is working to make the transition, there are two technologies you must understand: containers and Kubernetes. These are the tools that innovators are using to deliver software faster and more reliably—while using resources more efficiently and reducing costs.

The purpose of this eBook is to provide the foundation you need to get started on your journey to containers and Kubernetes, including what these technologies are, what they do, and why they matter.

Over the coming three years

The number of container instances worldwide will grow 6x

1.6

BILLION CONTAINERS

Enterprises will fully embrace Kubernetes

70%

KUBERNETES ADOPTION

Development teams will build modern applications

35%

CLOUD NATIVE APPS

Containers encapsulate an application in a form that's portable and easy to deploy. Containers can run on any compatible system—in any cloud—without changes. Containers consume resources efficiently, enabling high density and utilization.

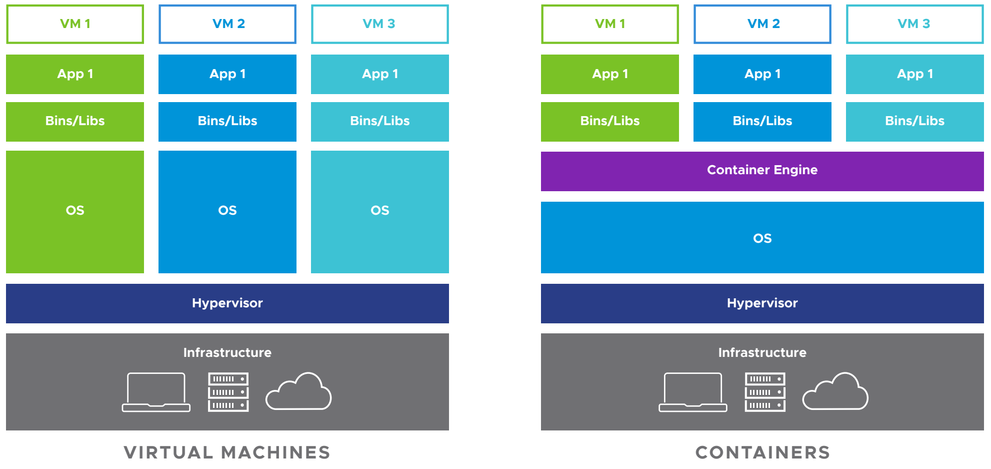
Kubernetes makes it possible to deploy and run complex applications requiring multiple containers by clustering physical or virtual resources for application hosting. Kubernetes is extensible, self-healing, scales applications automatically, and is inherently multi-cloud.

VMware Tanzu

Introduction to Containers

If your organization relies on virtualized infrastructure today—a pretty safe bet—your first questions regarding containers are likely to ask what they are and how they differ from VMs. The first thing you should understand is that it's not an either/or situation. Many organizations run both containers and VMs, combining the benefits of each technology while protecting existing infrastructure investments.

At the highest level, the difference between a VM and a container is that every VM runs a separate instance of an operating system. In a container environment, multiple containers share a single operating system instance. Each container encapsulates an application and dependencies like libraries, but it does not include the OS.



As a result, containers are lighter weight than VMs, making them easier and faster to start up and spin down. Multiple containers can run inside a single VM; the VM provides the OS and container runtime and your team can still take advantage of all the capabilities of familiar virtualization management tools.

SPEED

Boost Environments Rapidly

LIGHTWEIGHT

Minimal Resources Needed

PORTABILITY

Ability to Move Containers Freely

Why Containers Matter

While the public cloud by itself may seem like the solution to your company's agility problems, it's not easy to move applications between cloud providers or back on-premises from the cloud. By encapsulating all of an application's dependencies, containers make this much easier.

Once created, a container can move from a developer's laptop to QA to production—or from one cloud environment to another—without requiring any changes or reconfiguration.

That translates to greater agility, lower overhead, and fewer distractions for already busy teams.

VMware Tanzu

Container Benefits

While the differences between containers and VMs may not seem that major, they translate to significant benefits, especially for environments that need to deliver new software capabilities quickly and operate applications at scale:

- **Portability.** Containers can run without changes in a wide range of environments.
- **Microservices.** Instead of a single monolithic application, the cloud native approach breaks an application into its component services, each running in a separate container. This microservices architecture enables greater parallelism during both development and execution, reducing cycle time.
- **High density.** A container requires only minimal resources to run, increasing operating efficiency and start-up speed and reducing costs.

Over the coming three years

Public cloud cost control

"Cost is the top driver of public cloud adoption and also the number one pain post migration."

Reduce costs by 50%

451 Research, 2018

Development velocity

"Lowe's is hiring 2,000 developers. Siemens is hiring 1,500 developers. Financial Services companies are growing engineering teams by 15% every year."

Reduce cycles by 50%

World Economic Forum, 2019

VMware Tanzu

Principles of Kubernetes

Though containers have many benefits by themselves, it can be difficult to manage them at scale. Having a way to manage containerized applications uniformly and effectively is an essential element of cloud native architecture. Kubernetes has emerged as the solution to container management. Since it was first open-sourced in 2014, the Kubernetes system has become the most popular container orchestrator, eclipsing other solutions. It has already been adopted by more than 50% of the Fortune 500.

Built on a few core principles, Kubernetes has proven both more flexible and easier to configure and deploy at scale than alternative approaches to container orchestration.

Multi-cloud portability. Containers are portable across a range of environments from virtual environments to bare metal. Kubernetes is supported in all major public clouds, including AWS, Google Cloud Platform, and Microsoft Azure. As a result, you can run containerized applications on Kubernetes across many different environments.

Scalability. Cloud native applications scale horizontally. Kubernetes uses "auto-scaling," spinning up additional container instances and scaling out automatically in response to demand.

High availability. Kubernetes takes a self-healing approach to infrastructure that reduces the criticality of individual failures, making fire drills less common and your teams more productive.

API-based. The fundamental fabric of Kubernetes is its REST API. Everything in the Kubernetes environment can be controlled through programming.

Integration and extensibility. Kubernetes is extensible to work with the solutions you already rely on, including logging, monitoring, and alerting services. The Kubernetes community is working on a variety of open source solutions complementary to Kubernetes, creating a rich and fast-growing ecosystem.

Kuber—what—ees?
Tech visionaries are often fond of esoteric knowledge—and known for unusual senses of humor.

Kubernetes (pronounced koo-burr-NET-eez) is the Greek word for "helmsman," meaning the one who steers a ship. The common English words "governor" and "cybernetic" share the same Greek root.

Kubernetes is widely known as the leading system for container orchestration. However, whether spoken or written, the name is still a mouthful. That's why you'll often see and hear it shortened to K8s, where "8" stands in for the middle eight letters.

VMware Tanzu

How Does Kubernetes Help Your Business?

The Kubernetes environment—with uniform deployment, management, scaling, and availability services for containerized applications—offers significant advantages for your IT and development efforts, contributing directly to the success of your business.

Kubernetes Advantages for CIOs

- Empowers your organization to reach the next level
- Deliver new software and features more quickly
- Faster time to market
- Enable multi-cloud operations for greater agility and resilience

Kubernetes Advantages for Developers

- More productive and happier development teams
- Fewer impediments to development and deployment
- Faster time to market

How Kubernetes Helps Developers

The ticket-based infrastructure of conventional IT is replaced by self-service infrastructure that allows developers to access the resources they need when they need them. The Kubernetes community focuses on making the developer experience as clear and straightforward as possible. The result is more frequent code deployment, reduced time to production, and faster patches and bugfixes.

Metric	Before	After	Improvement
Code deployment frequency	Weekly	Daily	**5x more frequently**
Time from commit to deploy to production	7 days	Daily	**86% faster**
Mean Time to Repair (MTTR) for security patches	3 days	15 minutes	**99x decrease in TTR**
Time from request to provision resources	14 days	10-15 mintues	**450x faster**

VMware Tanzu

How Kubernetes Helps Operators

For operators, infrastructure resources are clustered and can be consumed and released elastically, enabling seamless scaling and higher resource utilization. Kubernetes eliminates many of the manual provisioning and other tasks of conventional enterprise IT.

Kubernetes clusters deployed in different private and public clouds provide a uniform (or very similar) management environment and identical principles of operation, reducing the learning curve associated with managing a multi-cloud environment and minimizing the risk of operator errors.

Kubernetes Advantages for Operators

- Easier multi-cloud operations
- More services delivered with less infrastructure
- Reduction of manual and task-oriented work
- Less operator error

By 2020, 75% of applications supporting digital business will be "built" not "bought"

—Gartner

PORTABILITY

Run Kubernetes everywhere.

Leverage a common upstream framework to run workloads on-premises, in public clouds or hybrid cloud.

INTEGRATION

Build on your SDDC infrastructure.

Apply Kubernetes as a practical path to public cloud adoption.

COMMUNITY

Move in lockstep with the community.

Work with a partner that demonstrates leadership in the open source community.

VMware Tanzu

The Bottom Line

Kubernetes increases the velocity of your company's software development efforts. The direct benefits that Kubernetes delivers for developers and operators have a ripple effect that benefits your company, your partners and suppliers, and your customers.

- **More reliable infrastructure and applications.** The ability to create automation to handle various failure modes and unexpected events translates to higher service levels for all users of your company's digital services.
- **Auditable, visible, and debuggable.** Complex applications can be opaque. Cloud native tools provide more insight into what is happening within an application, making it easier to identify and fix problems, optimize digital services, and improve response times.
- **Deep security.** Cloud native capabilities enable application developers to play an active role in creating securable applications that protect your company's intellectual property and the private information of customers and employees.

Long-term business success depends on digital services that are reliable, fast, and secure. Kubernetes is a smart investment in your company's digital success.

How Kubernetes Works

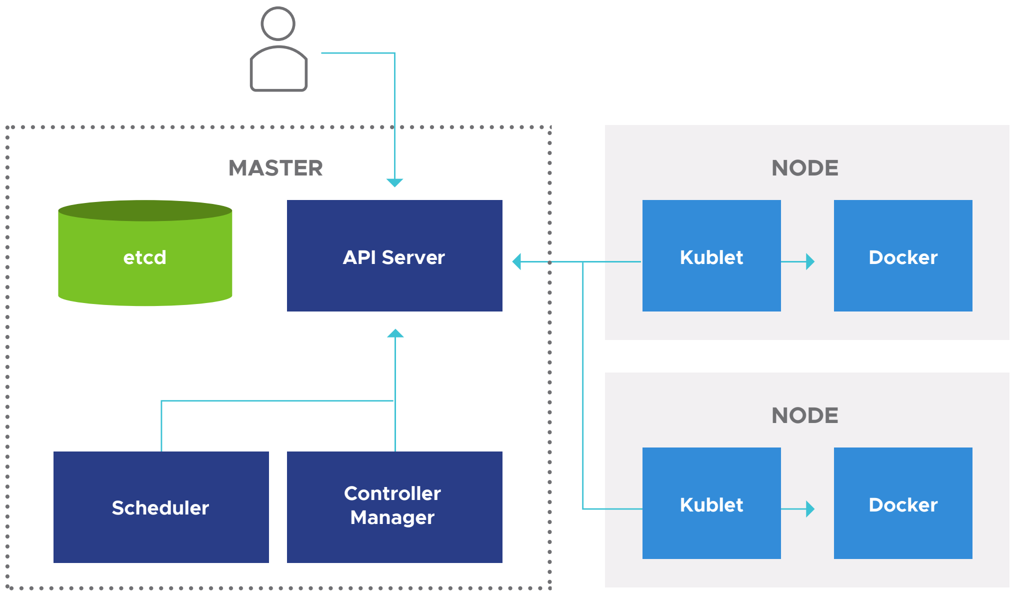
An understanding of some of the main concepts of Kubernetes will help you and your teams make smarter decisions as you experiment with and deploy containers. Although the term "orchestrator" implies a central conductor, Kubernetes is more like musicians playing jazz; the components of Kubernetes "play off" each other to coordinate activities and react to events.

At its core, Kubernetes is a database with some interesting features layered on top of it. These features enable a set of Controllers that each implement specific capabilities and work together to produce the end result. Each Kubernetes component is written using the same APIs that are available to users. As a result, components can be ripped out and replaced to extend the system and adapt it to new requirements and environments.

The diagram below shows the moving parts of a typical Kubernetes system at a high level. The core of the system is the database, *etcd*. The state of the cluster is stored there (and only there). In front of etcd is the *API Server*. Nothing else in Kubernetes talks to etcd directly. The API Server exposes a RESTful interface and provides the services necessary in a distributed system, including authentication, authorization, schema validation and translation, and resource initialization.

The *Scheduler* and the *Controller Manager* implement most of the orchestration logic of Kubernetes. Together, etcd, the API Server, Scheduler, and Controller Manager make up the Kubernetes control plane; they can run on a single node or across multiple nodes in an HA mode.

The worker nodes make up the data plane of Kubernetes; each worker node runs the container runtime (Docker in the diagram) and a local daemon called the *Kubelet* that communicates with the API Server.



Kubernetes Controllers

In Kubernetes, controllers ensure that the observed state of the cluster is as close as possible to the desired state. Each controller monitors its configuration, stored as a resource in the API Server. It then looks at the state of the world and tries to make the state of the world match its configuration. If a controller can't fully achieve the desired state, it retries. Controllers are both patient and diligent, resulting in a very stable distributed system pattern that is self-healing. If something goes wrong, a controller will work to fix it. If the desired state changes while a controller is working, it changes course and works toward the new desired state.

Here's a simple example of how this works in practice. Suppose a user creates a Pod—a group of one or more containers that run on the same node—via a request to the API Server:

- The Scheduler notices the "unbound" Pod and decides which node to run it on.
- The Kubelet notices a change in the set of Pods bound to its node and starts the new Pod.
- The Kubelet monitors the Pod via the container runtime and sends the status back to the API Server.

Any controller can call the API Server and say, "let me know when the things I care about change." This enables controllers to react to each other very quickly, making Kubernetes extremely responsive. When this all comes together, the different controllers play off of each other. There is no grand plan from the start but rather a set of players reacting to each other. The actions of the system respond to the state of the world in real time.

Operators Adapt Kubernetes to Your Needs

The developers of Kubernetes (see sidebar) recognized early on that they couldn't build capabilities to cover every possible situation into Kubernetes from the outset. Kubernetes instead allows for Custom Resource Definitions or CRDs that provide a way to extend the resources the API Server is able to manage. You can dynamically register new types of resources and manage them just like built-in resources. CRDs are usually paired with a custom controller called an Operator.

Operators allow you to encapsulate domain-specific knowledge for a specific application (or operational need). You can think of this as embodying the knowledge and logic that is often traditionally captured in run books. By automating application-specific tasks that might otherwise have to be done manually, Operators allow you to more easily deploy and manage your applications on Kubernetes. The open source Operator Framework provides the necessary tools to facilitate Operator creation.

A recent example is the Oracle MySQL Operator. The Oracle Operator tracks four CRDs. The first is the MySQLCluster object, which describes the number of replicas and the HA mode (single primary, multi-primary). The remaining CRDs allow you to control backups, backup scheduling, and restores—all critical functions in database environments. Operators like this illustrate how diverse systems can become self-managing when running on top of Kubernetes.

Meet the K8s Creators

Kubernetes was founded by Craig McLuckie, Joe Beda, and Brendan Burns. Kubernetes is heavily influenced by Borg, Google's internal cluster management system. The codename for K8s during its Google days was Project Seven of Nine, the name of a friendlier Borg in Star Trek. To bring things full circle, the Kubernetes logo is a helm with seven spokes, a nod to the original name.

While the technology was designed at Google, it is now maintained by the Cloud Native Computing Foundation (CNCF). Google partnered with the Linux Foundation to form the CNCF and donated Kubernetes 1.0.

The team that created Kubernetes remain active in the open source and Kubernetes communities.

Craig McLuckie
Founder, Cloud Native Computing Foundation
Vice President of R&D, VMware

Joe Beda
Principal Engineer, VMware

Brendan Burns
Distinguished Engineer, Microsoft Azure

VMware Tanzu

What Should My Organization Do Next?

In coming years, more and more systems will be built on or leverage Kubernetes. If your organization is building distributed systems, many problems can be avoided simply by assuming they will be deployed on a Kubernetes cluster. Applications can become self-managing through Kubernetes, either through capabilities built in to the applications themselves—they'll recognize they are running on Kubernetes and adapt—or through add-on Operators.

The Kubernetes ecosystem is expanding quickly, enabling a wide variety of new capabilities built on top of Kubernetes as a foundation. If you're considering Kubernetes or in the process of evaluation and deployment, you can benefit from the hard-won wisdom of those who have gone before you. The move to cloud native technologies—including Kubernetes—is not yet as easy as it sounds, but it's clearly worth it for organizations that prioritize software development and need to achieve faster development cycles, better resource utilization, and access to best-in-class open source technology.

Ready to learn more?

To talk in depth about how VMware can help you on your journey to containers and Kubernetes, please get in touch.

VMware Tanzu