CTO’s Guide to Containers and Kubernetes — Answering the Top 10 FAQs

Published 27 July 2020 - ID G00726809 - 19 min read
By Analysts Arun Chandrasekaran, Wataru Katsurashima

Initiatives: Technology Innovation and 2 more

Kubernetes is emerging as a prominent platform for building cloud-native apps and modernizing legacy workloads. We answer the most common client questions on containers and Kubernetes that will guide enterprise architecture and technology innovation leaders to make sound decisions.

Overview

Key Findings

- Enterprise architecture and technology innovation leaders are keen to invest in container platform tools to enable improved developer productivity and software agility and reduce technical debt.

- The container ecosystem is rapidly evolving with new open-source projects and startups, which makes keeping abreast of changes very difficult.

- Kubernetes has become a popular platform for building cloud-native applications, but the key constraints are lack of adequate skills and mature DevOps practices to operationalize and succeed with large-scale production-grade deployments.

- Large cloud providers and CaaS/PaaS software vendors are competing to be the platform of choice for cloud-native apps. Multi-cloud management, DevOps toolchain integration, flexible consumption models and product maturity are among the key differentiators.

Recommendations

Enterprise architecture (EA) and technology innovation leaders driving business transformation through technology innovation should:

- Improve the likelihood of success by ensuring that a strong business case exists, identifying appropriate use cases and instituting a DevOps culture before scaling Kubernetes platform environments.

- Create a platform engineering team that curates platform selection, and empower them to leverage cloud-native architectures and build cohesive DevOps toolchains.
Choose packaged software distributions or cloud-managed services for production deployments that integrate different technology components, simplify the life cycle management of that stack and provide multi-cloud management rather than a DIY approach.

Measure the benefits accurately both in terms of technical metrics such as software velocity, release success and operational efficiency gain as well as through business metrics such as top-line growth and customer satisfaction.

**Strategic Planning Assumptions**

By 2025, more than 85% of global organizations will be running containerized applications in production, which is a significant increase from fewer than 35% in 2019.

By 2025, more than 80% of ISVs will offer their application software in container format, up from less than 10% today.

**Analysis**

Containers and Kubernetes have significantly grown in popularity and adoption over the past five years. Container technology has been in existence for more than a decade. But its recent meteoric rise can be attributed to changes in software architecture and development patterns, growing adoption of DevOps, and the fact that both containers and Kubernetes are open-source projects supported by a wide ecosystem of participants. In this research, we answer the most common questions about containers and Kubernetes (summarized in Table 1) that will help enterprise architecture and technology innovation leaders benefit from and operationalize these technologies.

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer Highlights</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>What are some of the key benefits and challenges that we should be aware of for containers and Kubernetes?</strong></td>
<td>Key benefits include agile software development and deployment, environmental consistency, immutability and resource efficiency, and open-source innovation. Key challenges include platform complexity, security, automation and governance, and cultural and skills challenges.</td>
</tr>
<tr>
<td><strong>What is the current state of market adoption of containers and Kubernetes?</strong></td>
<td>According to the Gartner IOCS conference survey in December 2019, close to two-thirds of enterprises have deployed containers within their environment, while production deployments are seen in close to one-third of enterprises.</td>
</tr>
<tr>
<td>Question</td>
<td>Answer</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>What are the common use cases for containers and Kubernetes?</td>
<td>Microservices, legacy app migration and application portability are common use cases.</td>
</tr>
<tr>
<td>Can commercial off-the-shelf (COTS) applications be deployed in containers?</td>
<td>Third-party OSS deployments are more common, while support from proprietary ISVs is nascent. Platform vendors are investing in marketplaces to enable better access and simplify deployment of containerized apps.</td>
</tr>
<tr>
<td>How do I measure the ROI on container and Kubernetes investments?</td>
<td>Key metrics include developer productivity, frequency of software releases, operational efficiency and IT operations productivity.</td>
</tr>
<tr>
<td>What roles do we need to succeed with our Kubernetes deployment?</td>
<td>Having requisite skills across platform engineering and operations and security engineering is critical. For software development use cases, build and release engineering and reliability engineering are vital roles that need training and/or role creation.</td>
</tr>
<tr>
<td>How do we deploy Kubernetes? What are the pros/cons of various deployment models?</td>
<td>Customers deploy Kubernetes platforms either using upstream open-source or commercial software distribution or via cloud managed services. Cloud services offer operational simplicity and rapid time to market while software distributions offer better multicloud support and manageability. Upstream open source offers customizability but is complex to operate.</td>
</tr>
<tr>
<td>What are the different architectural building blocks of a container management system?</td>
<td>While container runtime and Kubernetes are the foundation, vendors have added several infrastructure, security and application capabilities in a container management system.</td>
</tr>
<tr>
<td>Who are the key Kubernetes platform vendors? How do their capabilities compare?</td>
<td>The key cloud providers are Amazon Web Services (AWS), Microsoft Azure and Google Cloud, while the popular software vendors are IBM (Red Hat), Rancher and VMware. We have outlined their capabilities across multicloud support, service mesh integration, serverless containers, marketplaces and bare-metal support.</td>
</tr>
</tbody>
</table>
1. What are some of the key benefits and challenges that we should be aware of for containers and Kubernetes?

Containers and Kubernetes can bring several benefits to an organization. Some of the key benefits are:

- **Agile application development and deployment**: Containers help simplify application packaging and enable a rapid application deployment process with the ability to do frequent application builds, quick software release and granular rollbacks. The ability to develop and deploy software faster can have a significant impact on top-line growth and customer experience.

- **Environmental consistency**: Through tight application component packaging, containers enable platform consistency across development, testing, staging and production clusters. This is an important driver of developer productivity and service resiliency.

- **Immutability and resource efficiency**: Containers should be deployed in an immutable way — which means there won’t be any configuration changes or patches. This makes container deployments highly repeatable, automated and secure with lower operational overheads. In addition, containers have a smaller resource footprint, which enables a much higher tenant density on a host, thereby increasing infrastructure utilization.

- **Flexibility and choice**: Kubernetes, which has emerged as the de facto standard for orchestrating containers, provides a high degree of scalability and flexibility for containerized applications. Kubernetes can orchestrate containerized workloads in either bare-metal or VM-based environments and is supported by a huge ecosystem of cloud providers, ISVs and IHVs. This API and cross-platform consistency, open-source innovation and industry support offers a great degree of flexibility for CTOs.

The above benefits must be balanced with the following technology and organizational challenges:

- **Platform complexity**: While containers and Kubernetes can apply to many use cases today, they don’t need to be used in every case. Using Kubernetes to orchestrate static, COTS applications can be overkill due to the inherent complexity of Kubernetes that would trump any meaningful...
business benefits that may accrue. Similarly, deploying it for performance-sensitive stateful applications requires significant know-how and operational maturity.

- **Security**: While there is nothing inherent in the container technology that makes it unsecure, deploying it at scale requires a mature DevSecOps process and security strategy different from protecting monolithic apps running in VMs. The tooling and process around container security needs rethinking as well as an understanding that this will be a shared responsibility between developers, platform operations, SRE and security teams.

- **Automation and governance**: Successful container deployments require extensive curation of technology components, consistent operations, and upgrade of existing tools and processes to ensure automation and governance. This necessitates investments in new tooling and removing any constraints in enabling an agile environment, which can often lead to conflicting opinions across teams as well as a laborious time-consuming selection and integration process.

- **Culture and skills**: Organizations face steep challenges in building containerized apps and operationalizing them due to dearth of skills around containers and Kubernetes across development, security and operations teams. Moreover, deploying a common platform such as Kubernetes requires clearly defined shared responsibilities across teams and a growth mindset that can tolerate failures and iterate accordingly. Lack of cultural change can easily derail your efforts.

2. **What is the current state of market adoption of containers and Kubernetes?**

The market adoption of containers has crossed the chasm of early adopters into more mainstream enterprise customers. A poll at the 2019 Gartner IT Infrastructure, Operations and Cloud Strategies Conference in Las Vegas in December 2019 showed that 36% of respondents were using containers in production. Another 31% were piloting or evaluating containers (n = 170). See Figure 1.

**Figure 1: Containers Adoption**
It is worth noting that while adoption of containers is growing across enterprises, the ratio of containerized applications is not high yet. Gartner estimates that less than 5% of enterprise applications are containerized today.

3. What are the common use cases for containers and Kubernetes?

There are several common use cases for containers and Kubernetes, as outlined below:

- **Microservices** — Microservices application architecture is characterized by independent application components that are distributed and loosely coupled. Containers and Kubernetes enable a strong foundation architecture for microservices due to their ability to orchestrate these modular services, enable scaling and self-healing of the services, and creating a layer of service isolation. Containers can enable a CI/CD process due to their ability to isolate code into a discrete unit, which makes it easier for developers and/or SREs to modify and update the code across the software development life cycle.

- **Application portability** — The runtime parity that containers enable and the ubiquitous availability of Kubernetes enable developers to build apps that can run in a consistent way across hybrid or multicloud environments. However, true portability is only possible for simple use cases because stateful and complex workloads that have tight dependencies require a lot of
 legacy app migration — While this is not an end state that most customers desire, monolithic applications can be migrated to containers, where you can run multiple instances of the application effectively. You can also take advantage of the more efficient deployment and service isolation that containers can provide.

4. Can commercial off-the-shelf (COTS) applications be deployed in containers?

Today, most container images are based on open-source software. According to the Datadog annual survey on containers, Elasticsearch, Postgres and NGINX are the most common container images. Compared to open source, container support of commercial applications has been much slower in progression. The licensing and support can still be serious challenges when enterprise organizations containerize their existing applications, unless they have certification and strong support from ISVs. Moreover, container deployment of commercial applications may not be ideal if you don’t have past experience in container management.

Though container support is still immature (especially for Windows-based applications), some major independent software vendors (ISVs) have been improving container support. For example, IBM released IBM Cloud Paks, which packaged various IBM middleware and application portfolios as containers on Red Hat OpenShift Container Platform. SAP also released SAP HANA in-memory database that runs natively on Kubernetes clusters. In addition, Kubernetes platform vendors are working with ISVs to create software operators that simplify the life cycle management of containerized applications. Some of the software operators available for Kubernetes include Apache Spark, Cassandra, CockroachDB, Couchbase, Kafka, PostgreSQL, MongoDB, NuoDB and Redis.

Though the speed of container support varies greatly among ISVs, the container support of ISVs is expected to increase. An important driver is platform vendors’ container marketplaces, which have been quickly expanding. Google Cloud Platform (GCP) Marketplace provides 83 Kubernetes applications and 68 container images as of February 2020. AWS also announced AWS Marketplace for Containers at re:Invent 2018 in November 2018, and it has a total of 320 container-related entries, of which 258 support EKS and 152 support ECS, as of February 2020. Other examples include OperatorHub.io (backed by Red Hat, AWS, Microsoft and Google Cloud) and the Bitnami catalog, which is now owned by VMware.

5. How do I measure the ROI on container and Kubernetes investments?

Ensuring ROI by building a thorough business case is important to validate that you aren’t investing in containers and Kubernetes purely because it is a shiny new technology. Organizations need to take a realistic view of the costs incurred and potential benefits. Figure 2 summarizes the key benefits and costs to consider:
Benefits:

- **Developer productivity** — Organizations often measure this on a per developer basis annually.

- **Agile CI/CD environment** — This is often measured through deployment rate, release success rate, software cycle time as well as business metrics such as user satisfaction and top-line growth.

- **Infrastructure efficiency gain** — Because of their smaller resource footprint, containers can enable a much higher tenant density on a host. This increases utilization of host resources as well as ISV licensing costs in scenarios where commercial off-the-shelf (COTS) software is licensed on a per host basis. This is measured based on deferred hardware upgrades, better infrastructure utilization and potential ISV savings.

- **Reduced operational overhead** — Kubernetes abstracts away a fair amount of low-level operational management, freeing up valuable IT staff time. This is measured as IT staff productivity savings. However, organizations need to account for the steep initial learning curve for the operations team.

Costs:

- **CaaS/PaaS subscription and other licenses** — Most organizations consume downstream Kubernetes software distribution or cloud services, which involves paying for annual software subscription fees or for cluster management (cloud services). In addition, optional software licensing fees may include security, monitoring and automation tools.
Infrastructure acquisition and upgrades — These would include investments in compute, storage and networking infrastructure (software and hardware) and cloud-based IaaS required to support containerized applications.

Staff training and new hires — Container and Kubernetes investments often entail training programs to upskill existing employees across software development, operations and security. They also demand money to hire new employees to fill gaps in any of the aforementioned areas or in newer practice areas, such as release engineering, automation engineering or SRE.

Rollout and implementation costs — These are the initial costs of professional and implementation services to get the environment up and running.

6. What roles do we need to succeed with our Kubernetes deployment?

Success with containers and Kubernetes requires a variety of core and auxiliary roles, depending on the use cases and maturity of the organization. While the security and platform ops roles are critical across most enterprises, software development use cases certainly create the need for developer, platform, build, release and reliability engineering teams. Table 2 shows the required key skills and roles.

<table>
<thead>
<tr>
<th>Role</th>
<th>Responsibilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software engineers</td>
<td>Coding, application design, implementation and testing</td>
</tr>
<tr>
<td></td>
<td>Source code management</td>
</tr>
<tr>
<td>Platform engineering and operations</td>
<td>Identify platform tools</td>
</tr>
<tr>
<td></td>
<td>Platform installation, configuration and administration</td>
</tr>
<tr>
<td></td>
<td>Automate container infrastructure provisioning</td>
</tr>
<tr>
<td></td>
<td>Integration of DevOps pipeline and automation of Day 2 operations</td>
</tr>
<tr>
<td></td>
<td>Enable self-service capabilities to developers</td>
</tr>
<tr>
<td></td>
<td>Capacity planning, workload isolation</td>
</tr>
</tbody>
</table>
7. How do we deploy Kubernetes? What are the pros/cons of various deployment models?

Kubernetes deployment models can be categorized into three patterns (see Figure 3):

- **Public cloud container services**: A deployment model where users consume a managed Kubernetes service offered by a public cloud IaaS provider. Users don’t have to build or manage Kubernetes master services. Examples are AWS ECS/EKS, Azure Kubernetes Service (AKS), Google Kubernetes Engine (GKE), IBM Cloud Kubernetes Service and Alibaba container service for Kubernetes. Some providers also offer serverless container services, where Kubernetes worker nodes are also automatically managed. Some services can run on-premises through distributed cloud products (such as AWS Outposts, Google Anthos, etc.).

- **Container management software**: A deployment model where users build and manage Kubernetes clusters on-premises and/or off-premises by using a packaged software solution. The software solutions combine a Kubernetes distribution with other management capabilities,
Figure 3: Pros/Cons of Various Kubernetes Deployment Models
Major advantages of public cloud container services are better operational simplicity and speed. With these models, users don't have to build or manage Kubernetes master services, which simplifies operations and improves time to value. Although not eliminating the need for learning Kubernetes, public cloud container services will considerably mitigate the learning curve. On the other hand, container management software can make it easier to achieve better consistency in hybrid and multicloud environments. Major advantages of the upstream version are better flexibility and customizability. New tools and technologies are released frequently in container management, so it can be an important requirement to have the flexibility to choose the tools.

8. What are the different architectural building blocks of a container management system?

Figure 4 references the architecture of container management systems.

**Figure 4: Architectural Building Blocks of a Container Management System**
Container management requires various capabilities across system infrastructure and application infrastructure management capabilities. Container orchestration, such as Kubernetes, is a core capability of the set. But other areas such as monitoring, security, policy management, API and management UI functionality also need to be considered when an enterprise organization manages containers at scale. Another key component is the registry, which is a sole repository for images and should include the key functionalities of vulnerability analysis and fine-grained access control. Container management software or services preintegrate these various elements to simplify building and managing container management systems. Many container management software products enable the management capabilities across hybrid cloud or multicloud environments by providing an abstraction layer across on-premises and public clouds. Most major cloud service providers also support hybrid cloud use cases by supporting running containers on their on-premises products.

The building blocks of container management are expanding. As management of Kubernetes becomes commonplace among vendor offerings, the competition focus is gradually shifting to higher-layer platform components to better support application life cycle processes and architecture, especially based on microservices. A notable example is service mesh. Container management vendors are integrating service mesh into their offerings.

9. Who are the key Kubernetes platform vendors? How do their capabilities compare?
The container management market consists of different types of market participants. Some of the sample vendors by various categories include:

- **Container management software vendors** — Diamanti, D2iQ, Hewlett Packard Enterprise (HPE), Mirantis, Rancher, Red Hat, Robin.io, SUSE, VMware

- **Cloud providers** — Alibaba Cloud, Amazon Web Services, Google Cloud, Huawei, IBM Cloud, Microsoft Azure, Oracle Cloud, Tencent Cloud

- **Managed services** — Giant Swarm, Platform9

Though management of Kubernetes becomes commonplace among vendor offerings, there are still competitive differences in Kubernetes platforms. Figure 5 shows some of the key selection factors and a comparative overview across vendors.

**Figure 5: Comparison of Kubernetes Platform Vendors**

<table>
<thead>
<tr>
<th>Analysis Criteria</th>
<th>Alibaba</th>
<th>AWS</th>
<th>Google</th>
<th>Microsoft</th>
<th>Mirantis</th>
<th>Rancher</th>
<th>Red Hat</th>
<th>VMware</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main Products</td>
<td>Alibaba Cloud Container Service for Kubernetes (ACK)</td>
<td>ECS, EKS, Fargate</td>
<td>GKE, Google Anthos</td>
<td>AKS</td>
<td>Docker Enterprise</td>
<td>Rancher</td>
<td>Red Hat OpenShift</td>
<td>Tanza Kubernetes Grid, Tanza Application Service, Tanza Mission Control</td>
</tr>
<tr>
<td>Main Deployment Type</td>
<td>Public Cloud Container Services</td>
<td>Public Cloud Container Services</td>
<td>Public Cloud Container Services</td>
<td>Public Cloud Container Services</td>
<td>Container Management Software</td>
<td>Container Management Software</td>
<td>Container Management Software</td>
<td>Container Management Software</td>
</tr>
<tr>
<td>Kubernetes</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>Non-Kubernetes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>•</td>
</tr>
<tr>
<td>Multicloud Deployment</td>
<td></td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>Service Mesh (Preintegrated/ Managed)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>•</td>
</tr>
<tr>
<td>Serverless Containers</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>Marketplace</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>Bare-Metal Containers</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
</tbody>
</table>

Source: Gartner
*Docker Hub

726809_C
Revenue growth and overall top-line revenue are also becoming important factors because the container management market is overcrowded. There are more than 90 Kubernetes management solutions certified by CNCF in addition to the hyperscale cloud providers. The number is too many to sustain, so market consolidation is inevitable.

0. Who are some of the key vendors and emerging startups in this ecosystem?

The container ecosystem is rapidly evolving with a slew of new projects and vendors. We have already outlined the key Kubernetes platform vendors that provide a software distribution service, cloud service or managed service that integrates many of the technology components together. Most customers will most likely start off with a platform distribution that may provide many of the capabilities outlined in Table 3. However, scale, use case, multiplatform deployments and need for best-of-breed capabilities may often steer technology innovation leaders to augment the capabilities of platform vendors with products from other established vendors and startups.

There is a growing ecosystem of established vendors and startups that offer commercial solutions in specific market categories. The CNCF interactive landscape chart provides an updated list of vendors and projects in this ecosystem.

Table 3 shows some of the key sample vendors that Gartner has seen come up in client interactions.

<table>
<thead>
<tr>
<th>Market Category</th>
<th>Vendors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security</td>
<td>Alcide, Anchore, Aqua Security, NeuVector, Palo Alto Networks, Portshift, Qualys, Snyk, StackRox, Styra, Sysdig, Tigera</td>
</tr>
<tr>
<td>Monitoring and Observability</td>
<td>Datadog, Dynatrace, Instana, New Relic, Sensu, Splunk, Sysdig, Turbonomic</td>
</tr>
<tr>
<td>Networking</td>
<td>Cisco, Juniper Networks, Tigera, VMware</td>
</tr>
<tr>
<td>Infrastructure Automation</td>
<td>Chef, HashiCorp (Terraform), Pulumi, Puppet, Red Hat (Ansible)</td>
</tr>
<tr>
<td>CI/CD</td>
<td>Atlassian, CircleCI, CloudBees, Codefresh, GitLab, Microsoft, Travis CI</td>
</tr>
<tr>
<td>Service Mesh</td>
<td>Aspen Mesh, Buoyant, Decipher Technology Studios, HashiCorp (Consul), Tetrate</td>
</tr>
<tr>
<td>Storage</td>
<td>Arrikto, Diamanti, Kasten, NetApp, Portworx, Robin.io, StorageOS</td>
</tr>
</tbody>
</table>
Evidence

The co-authors have spoken to most of the vendors listed in this document via briefings and inquiries. In addition, the authors have handled more than 500 client inquiries on this topic over the past 12 months.

1 “8 Facts About Real-World Container Use.” Datadog.


3 “CNCF Certified Kubernetes Offerings.” CNCF.

4 “CNCF Cloud Native Interactive Landscape.” CNCF.

Recommended by the Authors

Evolution of Virtualization: VMs, Containers, Serverless — Which to Use When?
Market Guide for Container Management
Best Practices for Running Containers and Kubernetes in Production
Top Emerging Trends in Cloud-Native Infrastructure
Innovation Insight for Microservices
Hype Cycle for Cloud Computing, 2019

Recommended For You

Market Guide for Small Cell Equipment
Summary Translation: How to Respond to the 2020 Threat Landscape
Ignition Guide to Creating a Functional Health Dashboard for Information Security
Building an Enterprise-Relevant Risk Assessment Process (McDonald's)
Build Controls Hygiene Discipline (DTCC)