SIMFORM

CTOs Handbook to Build an Enterprise Kubernetes Strategy

like GitHub, Spotify, and other tech superpowers

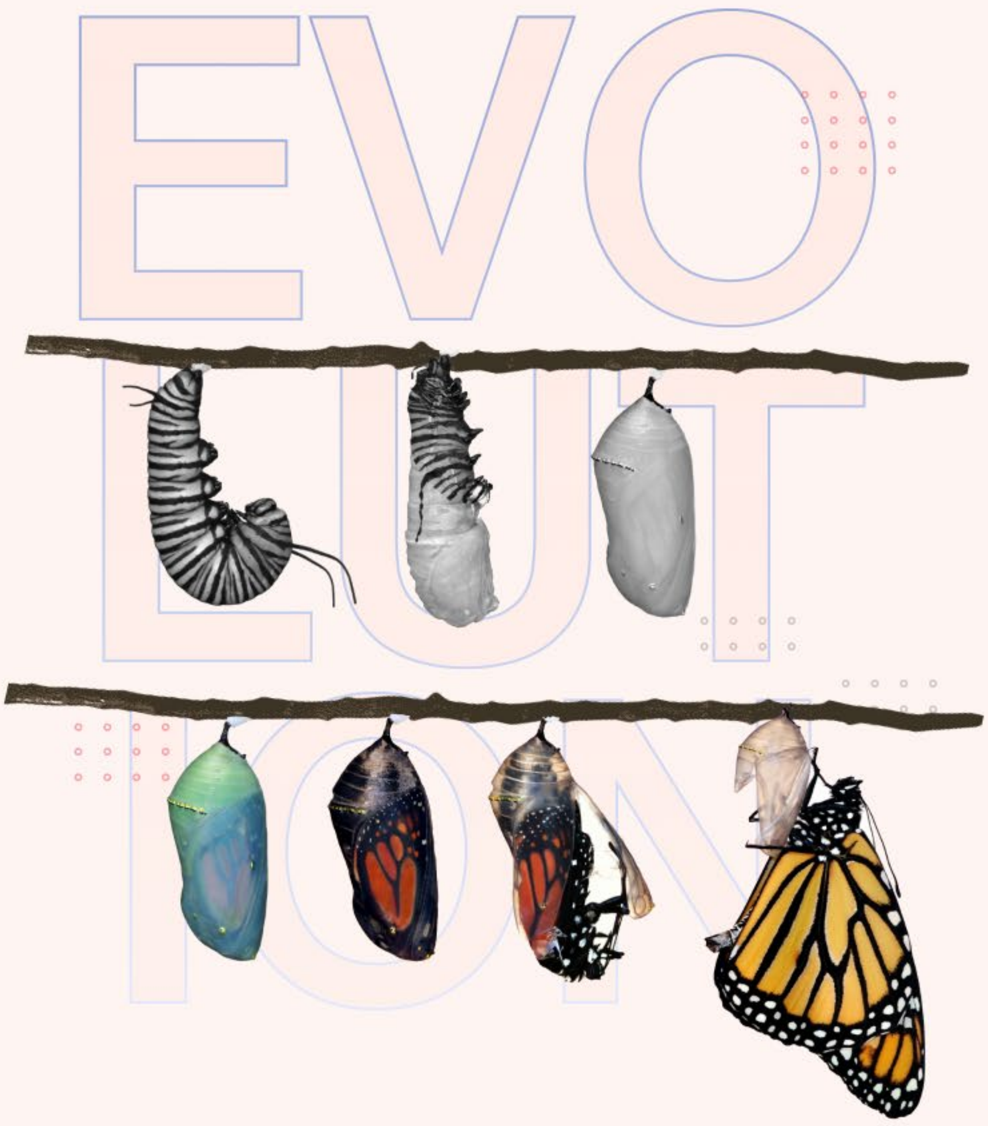
Contents

In today's ultra-competitive digital world, every enterprise expects its software developers to build and deploy applications rapidly. Users also want new features and updates across all their devices in real-time. To meet this need, the ever-agile IT operations teams are turning to container technology. And the best way to deploy these containerized applications at scale and create a highly available distributed system is to implement the enterprise Kubernetes strategy.

Kubernetes, also known as K8s, was built by Google. After its initial launch in 2014, Kubernetes gradually cracked mainstream success and adoption and became the de facto industry standard for open-source container orchestration software. And now, this revolutionary technology, which was later handed over to Cloud Native Computing Foundation (CNCF), is central to the digital transformation of 77% of organizations. Around 2017, many big names in tech like GitHub, Spotify, Pinterest, and many more jump on the Kubernetes train.

This ebook highlights the **benefits, challenges, real-world examples**, and the **necessary steps** to implement a successful enterprise Kubernetes strategy.

01



Container: Witness the historic shift in how modern applications run

Kubernetes' journey in enterprises does not generally start with a random **"Hey, let's move our applications into Kubernetes."** Instead, for most cases, it's more like, **"We need to containerize our applications!"** The reason is - containers are the foundation for Kubernetes. Let's see how.

Marc Andreessen penned his pivotal "Why Software Is Eating the World" essay in The Wall Street Journal in 2011. It's been over a decade, and it rings truer than ever, as software becomes even more essential for the functionality of IT as well as non-IT industries.

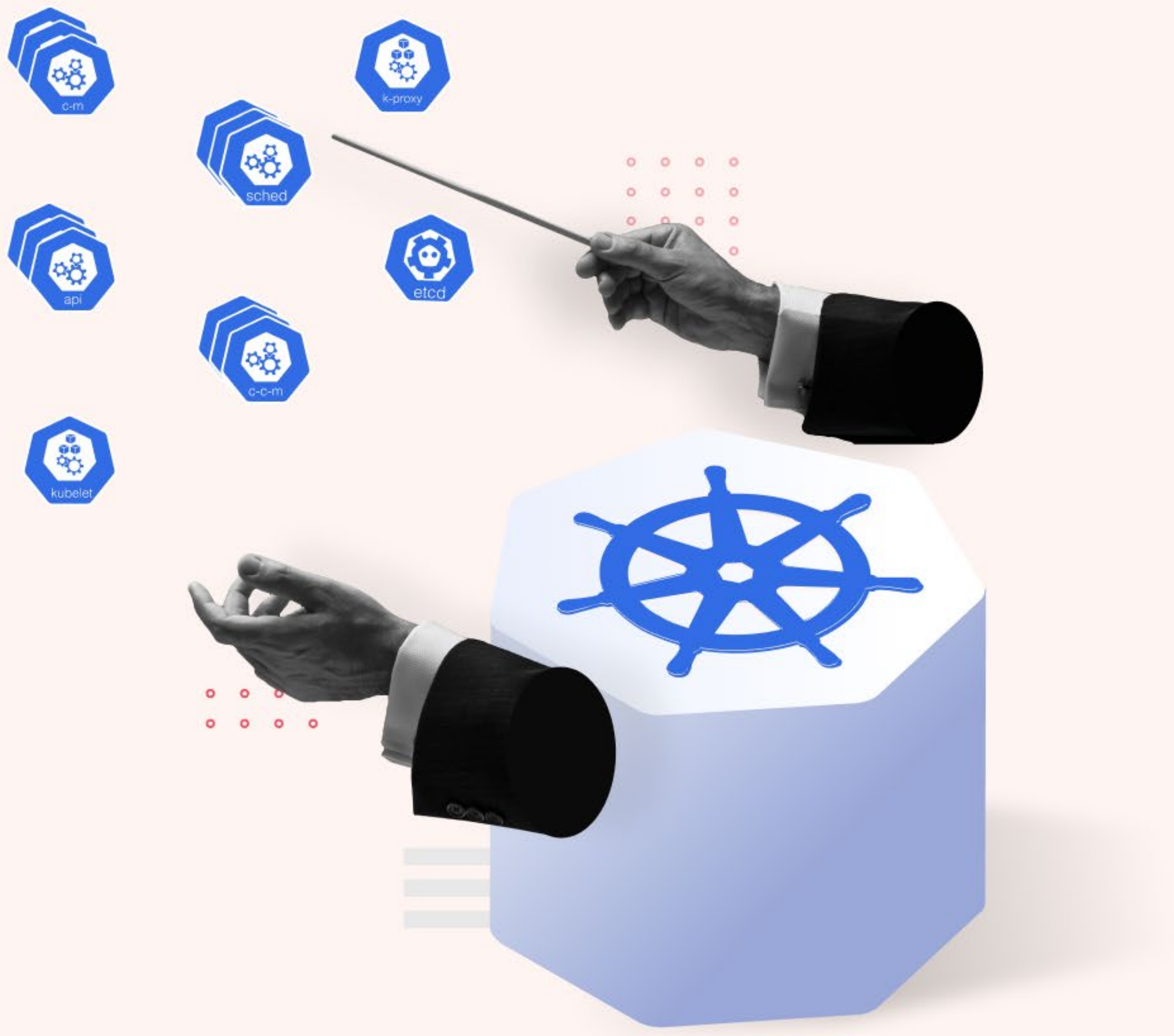
Meanwhile, the complexity of software also keeps growing, with dependencies creating havoc and new bugs popping up now and then. There are some essential complexities owing to your business domain, extremely complicated enterprise environments, and inherently complex problems that you're trying to solve. Then, you'd also have accidental complexities coming with the tooling, cloud-native era, and pervasive microservices environment. Dealing with these complexities can be pretty exhausting.

Come into the picture, containers! To give businesses "systems" that "work" all the time, irrespective of the environments and dependencies. Containers, a form of operating system virtualization, provide a consistent way to package application code and its libraries and dependencies into a single object. It can run in any isolated user environment, whether on traditional IT, desktop, or the cloud. Also, a container can start and terminate quickly, enabling applications to scale to any size.

Linux remains at the core of open-source software development and is foundational to containers as an application deployed into containers run on Linux.

For modern developers, 2013 marks the start of the contemporary container era with the introduction of Docker. Now containers have almost become the default packaging mechanism for applications, and many legacy workloads are also getting containerized.

02



Kubernetes: Meet an awesome conductor that manages everything in an orchestra

Containers do not have operating system images, so they are small, fast, and portable. However, when it is larger application deployments, you need to deploy multiple containers as one or more container clusters. Manual management of this would require superhuman powers and also a significant amount of time. Instead, container orchestration tools like Kubernetes can automate and manage these clusters and, overall, the lifecycle of containers and services.

Also, as we saw earlier, enterprise applications are becoming more complex, and so DevOps teams need a solution to launch all the services dependent on those applications.

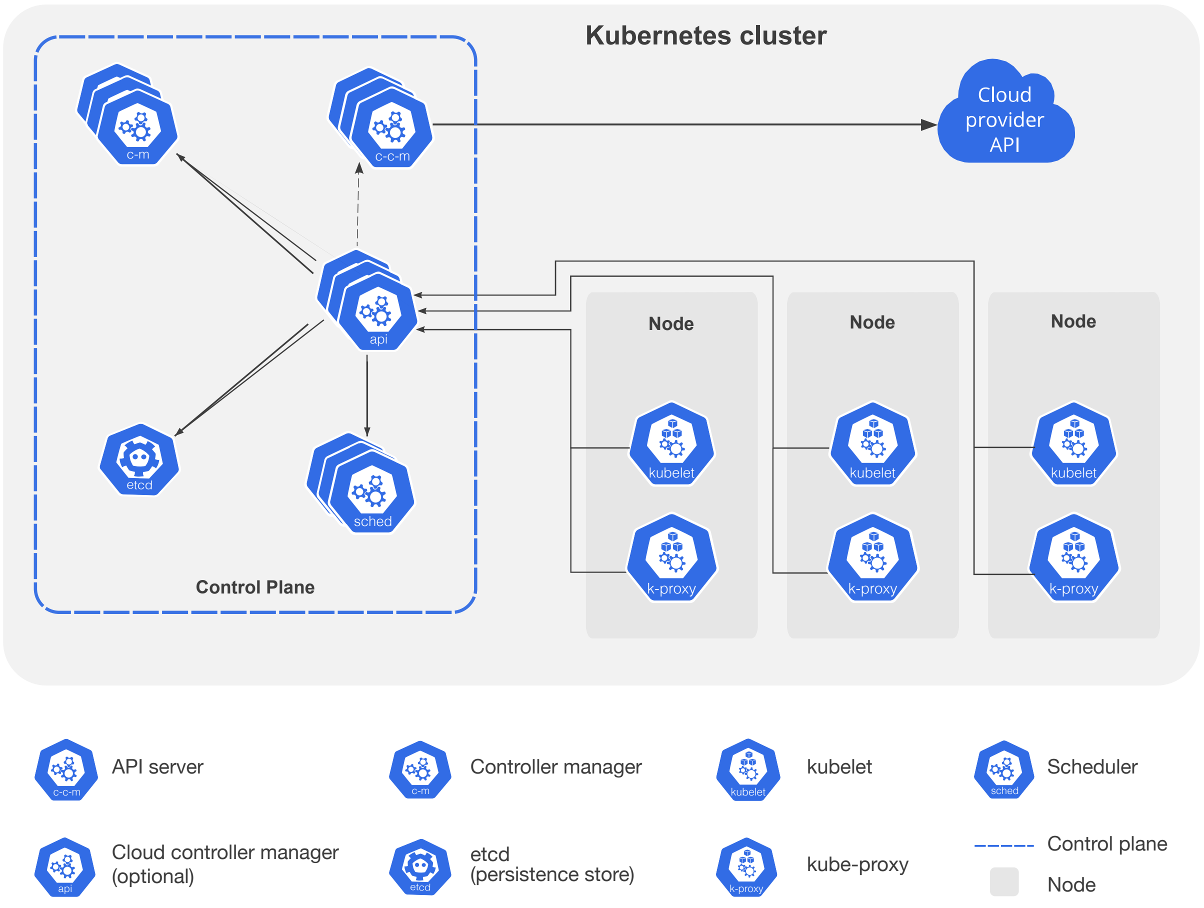
Again, Kubernetes is the tool that orchestrates the complexity and ensures these applications and services are healthy and can connect to one another.

You can run Kubernetes on any operating system, but it was built from Linux and uses key Linux libraries, features, and system calls. So, while selecting an OS for your Kubernetes environment, you must look for a leading, trusted Linux distribution.

Kubernetes is portable and extensible and enables developers to view, coordinate, and manage containerized workloads and services, facilitating both declarative configuration and automation. Though Kubernetes is most commonly used with Docker, it can orchestrate containers other than Docker and hence work with any container runtime.

1. Components of Kubernetes

Before we head deeper into the enterprise Kubernetes strategy, let's first understand the basic terms and components related to K8s.



Pods

A Pod is the smallest deployable unit of computing in a Kubernetes cluster, with shared network resources and storage and a specification for running the containers.

Nodes

A node is a worker machine (physical/virtual) where Kubernetes launches the containers that are inside the pods. Nodes are at the center of a Kubernetes cluster.

Cluster

A cluster in Kubernetes is a set of node machines, which run containerized applications, so in case of one node failure, the application remains accessible from other nodes.

Desired state

The desired state is a core concept where you describe the state of objects that will run the containers through a declarative or an imperative API.

Deployment controller

It continuously monitors the current state of the relevant resources.

Control plane

The control plane helps maintain the desired state of the cluster. It generally runs across multiple computers and consists of processes that manage the pods and the worker nodes in the cluster.

etcd

etcd is a distributed, reliable, and consistent key-value store that stores the configuration information used by every node in the cluster.

Kubelet

The Kubelet is a small application that communicates with the control plane and ensures that defined containers in pod configuration are running on a particular node.

Fun Fact!

Here is an interesting comic-book-style introduction to Kubernetes by Google!

03

Why do growing organizations love Kubernetes? Know the prime drivers

From removing barriers in infrastructure management to significantly increasing the agility and efficiency of software development teams, from minimizing time and risks associated with having new software into production to reducing overall costs, enterprises have many reasons to migrate their deployment setups to Kubernetes.

And thus, it doesn't come as a surprise that DevOps teams and businesses both hold Kubernetes in high regard when it comes to managing software complexity and achieving hybrid cloud goals.

Moreover, the K8s community is ever-growing, acting as a catalyst in Kubernetes adoption across the industry. Here are 3 prominent drivers for Kubernetes adoption.

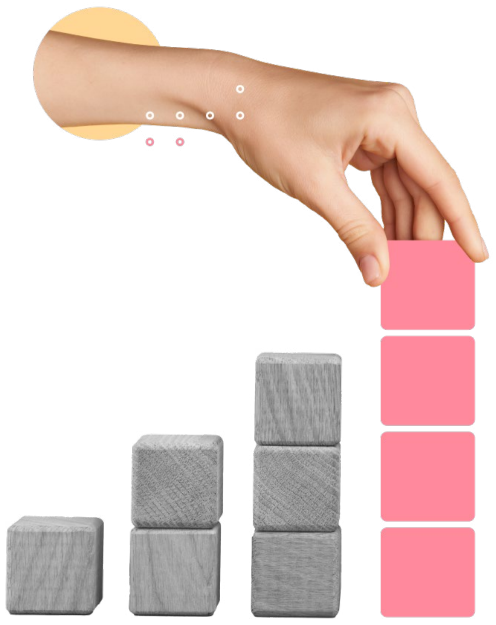
1. Improved scalability and availability

Developing a mobile or web application is an extensive and time-consuming process. A successful application cannot afford to hit a saturation point when its user base is growing.

Kubernetes horizontally auto-scales the containers depending on the application requirements, which usually changes over time. It provides not only infrastructure utilization but also a satisfactory user experience. You can use infrastructure metrics, custom metrics, and other metric-resource utilization metrics to trigger the scaling process.

When you're deploying a horizontally scalable application to Kubernetes, you need to ensure the configurations of the following:

- Node and pod, affinities and anti-affinities
- Resource requests and limits
- Horizontal pod autoscaling
- Deployment strategies
- Health checks
- Pod disruption budgets



→ For instance, **Pinterest** had 1,000 microservices, multiple layers of infrastructure, various setup tools, and platforms after eight years of existence. In 2016, the company wanted to create the fastest path from an idea to production without bothering engineers about the underlying infrastructure.

They first moved services to Docker containers, then to Kubernetes, which simplified overall deployment and enabled them to build on-demand scaling and new failover policies.

2. Application portability

Many businesses nowadays select a variety of cloud platforms and technologies to avoid single-provider dependency and have services that are tailored to their requirements. In addition, the hybrid or multi-cloud strategy also enables them to have better performance and greater cost efficiency.

However, various environments used for application development usually have system-specific dependencies and run on different operating systems. They also impose challenges like migration, provisioning, governance, security, and compliance.

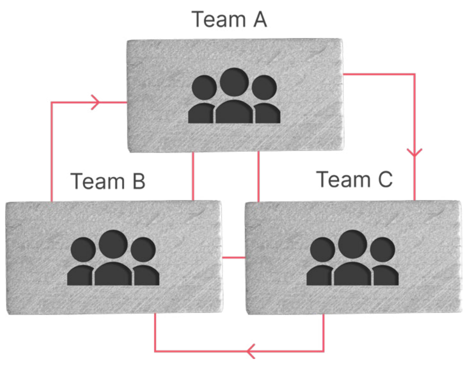
Thanks to its environment-agnostic approach, Kubernetes facilitates automation and configuration, in turn helping manage workloads and migration. Further, Kubernetes inherently supports portability, making it more suitable for hybrid and multi-cloud computing environments.

→ **Gannett/USA Today** uses Kubernetes to operate across the Google Cloud Platform and AWS. The adoption of Kubernetes allowed them to support their growing base of customers. Gannett deployed Kubernetes in AWS to run their apps seamlessly in GCP.

3. Faster release cycle

Kubernetes applies a microservices approach, allowing you to break down your development team into small teams, where each team focuses on specific functions. This way, you can scale various smaller, agile teams of specialized experts who can help support a fleet of thousands of machines. And your IT teams can efficiently manage larger applications across many containers.

Besides, APIs between these microservices reduce the amount of cross-team communication.



➜ One of the best examples of accelerated time to market is Tinder, which decided to move its platform to Kubernetes in 2017. Through immutable deployment, the disruptor tool led Tinder Engineering toward containerization and low-touch operation. Tinder, too, faced challenges of scale and stability owing to the high traffic volume. Tinder's engineering team migrated 200+ services and had a Kubernetes cluster of 15,000 pods, 1,000 nodes, and 48,000 running containers.

04

Kickstart your journey: Steps for a successful enterprise Kubernetes strategy

Every enterprise has a different journey to Kubernetes adoption. However, the basic steps remain almost the same.

1. Analyze the business scope

Despite Kubernetes being designed to run for various use cases, some use cases work better with the less complex yet effective alternatives. Kubernetes might not be a good choice if

- You have a small startup with a few resources requiring you to deliver a minimal version of its product to end-users quickly.
- You have established a cost-effective architecture that works for you without significant issues.
- You don't expect your simple application to scale to a large user base.

However, Kubernetes could be a better option, if

- Your enterprise is large with many development teams, where each team looks after different microservices of an application used by millions of daily users.
- You have a startup developing an application that requires heavy data processing, and you expect to increase your user base steadily.
- Your application or some parts of it are on-prem, but you want to avail similar features on the cloud platforms.

Organization-wide adoption of Kubernetes requires prioritization of goals in the context of this new technology that offers a rich set of functionalities to run modern, microservice-centric applications. And to prioritize your goals, you should first understand the potential of Kubernetes and visualize how your company might be using it in, let's say, five years.

For instance, if the aim is to reduce infrastructure costs, you should focus on building big clusters and getting the maximum possible density out of them. Instead, suppose you want to accelerate development and have continuous delivery across different computing platforms. In that case, it'd be better to take a different approach that emphasizes flexibility and delivers more tooling around Kubernetes like monitoring and CI/CD integration.

2. Align Kubernetes with enterprise's cloud strategy

Nowadays, software development teams are consistently building new applications, using Kubernetes to run them, and deploying them to various clouds. Fortunately, all major cloud providers - AWS, Azure, and Google have made it easy to deploy Kubernetes clusters within minutes. Then there are software giants creating their own standards and benchmarks, including Oracle, IBM, Rancher (SUSE Linux), etc. There are a few broad patterns in how enterprises can adopt Kubernetes.

Each approach has its advantages and constraints, and you should make the selection based on your enterprise's maturity and business strategy.

Managed Kubernetes - More and more companies are shying away from managing their own clusters due to the availability of varied and authenticated managed Kubernetes options. You can partner with third-party providers to assign some or all responsibilities of managing the successful setup and operation of K8s. Enterprises with an established cloud strategy can leverage Kubernetes-as-a-service (KaaS) provided by their cloud providers.

Currently, the most popular managed Kubernetes options are — Amazon Elastic Kubernetes Service (EKS), Azure Kubernetes Service (AKS), and Google Kubernetes Engine (GKE).

Cloud-agnostic managed Kubernetes - By selecting a managed Kubernetes service provider not tied to any cloud provider, you can have a more portable platform that can cope with future needs and is entirely cloud-native.

Kubernetes as a turnkey solution / Self-hosted Kubernetes platform -
Some technology-focused and regulations-driven enterprises are often inclined towards setting up Kubernetes on-premises or as part of a private cloud to avoid vendor lock-in and restricted control associated with managed K8s.

3. Make an informed decision based on current adoption patterns

Often development teams within the organizations have different preferences regarding the Kubernetes platform, based on their requirements for configuration flexibility and Kubernetes cluster management.

For instance, some teams may adopt a hosted K8s platform like EKS or GKE from their preferred cloud provider to reduce management overheads. In contrast, others might favor self-managed K8s clusters that give higher control and flexibility. A detailed analysis might reveal the preference helping you make the informed decision for a common platform with maximum support and adoption.

4. Determine the right governance approach

Governance, in general, refers to verifying, managing, and enforcing specific rules across teams, departments, or the entire organization. In the Kubernetes context, that means dictating rules across Kubernetes clusters and applications running in those clusters.

When you continue to grow the number of clusters in use without governance, they will be in separate pockets with different rules and implementations. And you'll need to invest a considerable amount of time and effort in the future to manage everything. Governance in Kubernetes helps synchronize clusters and enforce policy management.

Decentralized governance enables teams to wrap their Kubernetes clusters on-demand depending on the requirements.

While with **centralized governance**, IT teams can centrally control repetitive tasks for cluster lifecycle management like upgrades, patching, and applying security policies in a uniform manner.

	Decentralized governance	Centralized governance
BENEFITS	• Development teams have maximum flexibility to run Kubernetes clusters in terms of storage, security policies, and infrastructure on which they run. • A high degree of independence promotes innovation	• Great focus on security and compliance • Cost-effective
CHALLENGES	• Management of the development, tools, and security of your entire organization becomes difficult. • Difficulties identifying role violations, and performing compliance checks. • Often negative impacts on application availability and performance	• Addressing the fine line between an operator's ability to maintain the necessary procedures and governance and a developer's ability to innovate • Integration with the entire stack

The optimum choice here would be to blend both types of governance practices. You can have the best of a decentralized approach with enough management and controls to remove repetitive tasks and ensure compliance. Your primary focus should be on maintaining visibility into the provisioned clusters rather than insisting on the utilization of a set of pre-approved clusters.

5. Establish the right ownership model

The Kubernetes ownership model represents the responsibilities and accountabilities of the individuals or teams managing various aspects of the Kubernetes implementation. It ultimately simplifies governance. Let's understand how. Before deploying your containerized application, you must install various tools that enable different capabilities required by the application.
These capabilities include Layer 7 routing like Nginx Ingress, TLS certificates with Cert Manager, DNS management with External DNS, and many more.

And then, you'll be faced with many questions, like

- Who is responsible for these applications?
- Where should we deploy it?
- How do we track various application deployments?

Even before this, while starting, you'd wonder who is responsible for maintaining the Kubernetes clusters. Or if anyone could request or create a new Kubernetes cluster. The ownership model addresses all these questions and more.

For large enterprises, mostly, it is a shared services team and the central IT team that leads the container strategy. The shared services team contributes by providing insights on how this approach will modernize application development.

In contrast, central IT teams with a strong understanding of infrastructure, platform operations, multi-tenancy, security, and existing IT investments, focus on cloud computing and other computing platforms. Great results can be achieved if central IT teams work in collaboration with teams leading innovation around application development.

However, it might not be the right model for every business. For instance, small startups with a single application, one DevOps engineer, and a small development team might not require service ownership. Instead, larger organizations with hundreds or even thousands of clusters and a centralized Kubernetes platform can utilize this ownership model effectively.

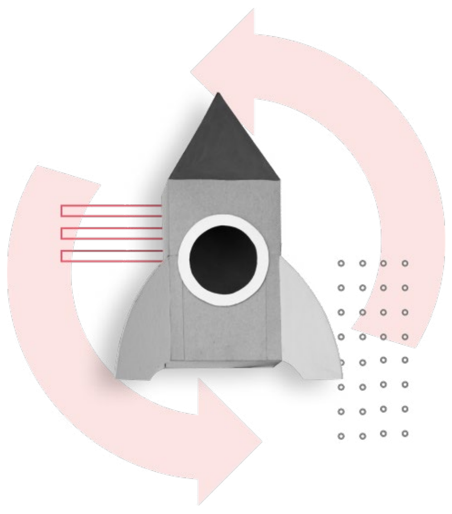
6. Adopt best Kubernetes security practices

We'll learn soon in the next section how Kubernetes workloads are vulnerable to several types of security threats related to the compromise of the nodes, pods, clusters, Kubernetes control plane, and network connections. Kubernetes documentation describes these the 4Cs of cloud-native security -- cloud, cluster, container, and code -- and outlines different control points for each.
Here are some of the best practices to secure Kubernetes clusters and workloads throughout the software development lifecycle.

- Apply automated image scanning at each stage of the CI/CD pipeline to ensure container images are free of vulnerabilities
- Have strong isolation between containers and let containers have minimal privileges on the host
- Create a container image from scratch to minimize the attack surface
- Use a minimal base image with truly essential libraries or OS components for containers.
- Review cluster configuration with automated tools and rectify any insecure configuration
- Build network security definitions into Kubernetes workloads using a declarative approach
- Enable data encryption in transit, auditing, and compliance reports, and have continuous compliance checks with automated remediation.

7. Configure a zero-impact deployment strategy

Application deployments or upgrades can disrupt service or downtime in a traditional software environment. Kubernetes deployments, by default, are rolling in nature. A zero-impact deployment strategy helps enterprises easily roll out updates and new features and improve time to market through frequent releases.

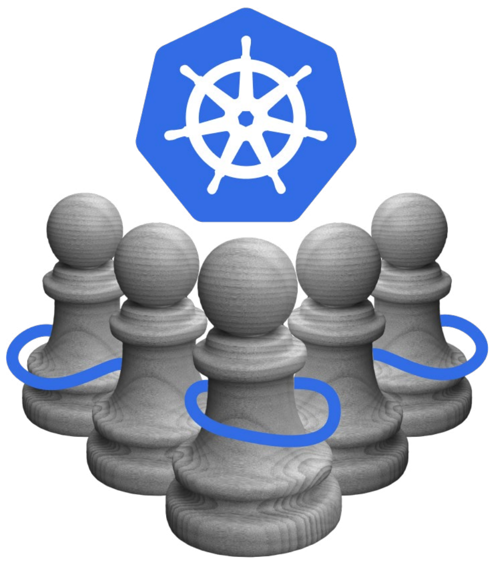


Below are the main Kubernetes deployment strategies in brief.

- **Rolling deployment:** One by one, pods running in the old application version are replaced with the new version without downtime to the cluster.
- **Recreate:** All the pods are terminated and replaced with the latest version.
- **Ramped slow rollout:** replicas of the latest version are rolled out while shutting down old replicas in parallel.
- **Canary deployment:** the older version of the application serves most users, while the more recent version serves a small pool of specific users based on geography, demography, etc. Once successful, the test deployment is rolled out to more users.

8. Prepare your teams for Kubernetes adoption

The learning curve for Kubernetes is quite high. Therefore, as you start building a core team of Kubernetes admins and users, you should attempt to certify and train the maximum members of your team. Also, if some of your staff members have expertise in Kubernetes and containers, you can ask them to provide support while defining requirements, developing policies, or evaluating tools.



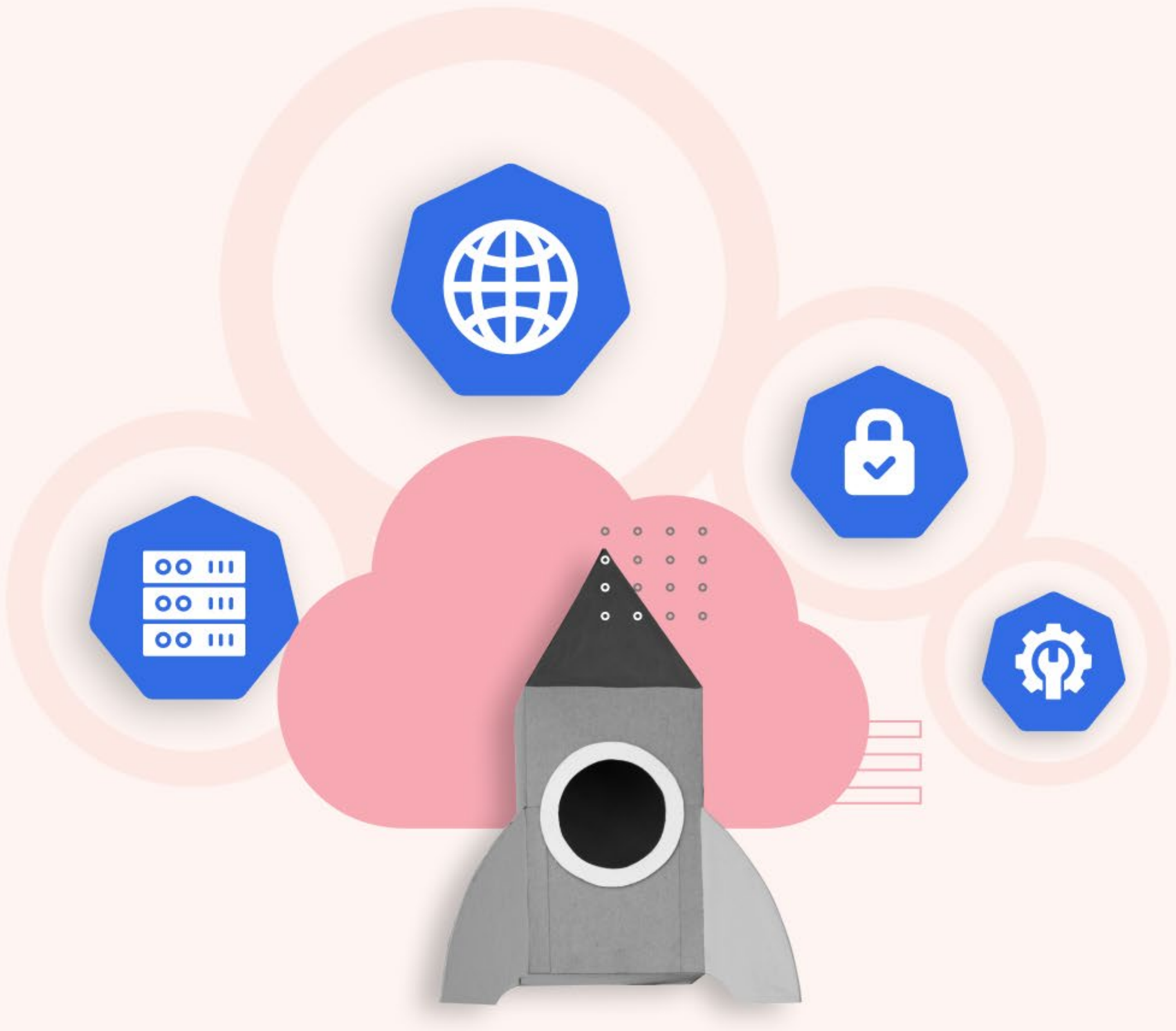
You can also have an internal team for building a digital platform to reduce the cognitive load for development teams regarding knowledge and support.
For example, companies like Airbnb and Uswitch minimized the cognitive load of developers with self-service capabilities.

Effective team communication and the behaviors promoted by enterprise culture are equally important as technical expertise while adopting Kubernetes. Failure in addressing how different teams must interact while using Kubernetes can put your entire endeavor at risk.

A team-focus approach for Kubernetes adoption can be achieved by

- Exploring your team's understanding of the Kubernetes abstractions, which they'll be using regularly.
- Defining how the platform team will provide on-call support and communication mechanisms.
- Clarifying collaboration methods in the new internal platform with self-service capabilities.

05



Yes, there are a few caveats!

No technology adoption is free of complexities. From deployment failovers to reduced visibility into data flows, from the latency between services to optimizing resource usage, you have to overcome many challenges before you can get the above-stated benefits of Kubernetes. As per a survey, 95% of the organizations using cloud-native technology encounter challenges, mainly during the development phase (47%). Given below are 4 main challenges.

1. Security and compliance

47% of the enterprises have to face security concerns while adopting Kubernetes. The reason is - K8s can be quite complex and highly vulnerable if the process is not appropriately monitored. Even with monitoring, it becomes difficult to locate the container with vulnerability when you deploy more containers because of its distributed nature. These vulnerabilities can exist in production-accessible container registry, fail builds, images, and third-party admission controllers in Kubernetes clusters. And they refer to exploits on containers, including crypto mining, privilege escalation, malware installation, and host access.

➜ One of the best examples of a Kubernetes break-in is the Tesla cryptojacking attack, where the hackers infiltrated the company's Kubernetes admin console. The attack resulted in mining cryptocurrencies through Tesla's cloud resources on AWS.

Then there are other security concerns like threats during the runtime phase, compliance audit failures, ineffective patch management, etc.

2. Configuration

These include possible configuration issues involving namespaces, runtime privileges, persistent storage, container images, and control plane. On top of that, administrators must ensure that their Kubernetes configurations strike the right balance between usability and security.

→ For instance, Airbnb configured and deployed over 250 critical services to Kubernetes to support over 1,000 engineers. Kubernetes certainly worked for them, but they also had to face many issues. And the main one was configuration as the tooling used to interact with Kubernetes files, and the cluster is complex. Despite having a way to integrate with their infrastructure, the implementation was a bit vague as their infrastructure was a decade old at that point.

3. Networking

Kubernetes can't function properly with traditional networking approaches. It becomes even more challenging with the large-scale deployment. Here are the common networking challenges.

- The **complexity challenge** mainly arises while deploying in more than one cloud infrastructure. Separate policies of the individual cloud make already complex Kubernetes even more so across multiple infrastructures.
- Pods can use any number of IP addresses for one workload, making the communication between static IP addresses and ports and the implementation of IP-based policies more challenging.
- The **multi-tenancy challenge** occurs when one Kubernetes environment is shared among multiple workloads. If resources are not allocated properly, other workloads in the same environment naturally get affected.

The container network interface (CNI) plug-in enables seamless integration of Kubernetes into the infrastructure and gives application access on different platforms.

4. Storage

Many larger enterprises, especially those with on-premise servers, encounter the storage issue. The reason is they manage the entire storage infrastructure without relying on cloud resources, which can lead to memory crises and vulnerabilities.

Containerized applications in Kubernetes can be either stateless or stateful. Stateless applications have no persistent state and, as a result, are prone to data loss when the containerized application shuts down or accidentally crashes. However, Kubernetes has to attach persistent external volumes to store states for stateful applications.

06

But caveats can indeed be maneuvered! Real-world success stories of Kubernetes

Many large enterprises have shifted from heavily hardware-centric legacy infrastructure to the cloud, powered by containers and Kubernetes. Let us see some popular examples that can inspire and impart knowledge.

1. GitHub

GitHub was launched in 2008 upon Ruby on Rails. In eight years, the organization's infrastructure started to strain as GitHub was growing enormously in multiple ways, in terms of users, rate of requests, engineering staff, and more. In addition, with the increase in GitHub services, the Site Reliability Engineering (SRE) teams had to spend more and more time on server maintenance, provisioning, and tasks other than the main mission.

> **New services could take days, weeks, or even months to deploy, depending on their complexity and our team's availability. We very much needed to provide these other engineers with a self-service platform they could use to experiment with while building new services, and even deploy and scale them."**
>
> Jesse Newland
> Former Principal Production Engineer at GitHub

During their evaluation of container orchestration platforms, certain qualities of Kubernetes stood out from the other platforms, such as the first-run experience, the vibrant open source community supporting the project, and a wealth of information regarding the experience that motivated its design.

> **I can already see how the move to Kubernetes is creating an environment at GitHub for more rapid innovation — innovation that will benefit users and, ultimately, the software industry as a whole."**
>
> Jesse Newland
> Former Principal Production Engineer at GitHub

2. Spotify

Spotify is quite an early adopter of Docker and microservices. For years, a phenomenal music streaming service used an open-source, in-house container orchestration system called Helios. However, things started changing by late 2017.

> **Having a small team working on the Helios features was just not as efficient as adopting something that was supported by a much bigger community. We saw the amazing community that had grown up around Kubernetes, and we wanted to be part of that. We wanted to benefit from added velocity and reduced cost, and also align with the rest of the industry on best practices and tools."**
>
> Jai Chakrabarti
> Director Of Engineering, Core Infrastructure at Spotify

Spotify gradually started migrating to Kubernetes in 2018.

> **The biggest service currently running on Kubernetes takes about 10 million requests per second as an aggregate service and benefits greatly from auto scaling. Before, teams would have to wait for an hour to create a new service and get an operational host to run it in production, but with Kubernetes, they can do that on the order of seconds and minutes."**
>
> James Wen
> Senior Site Reliability Engineer at Spotify

3. Bloomberg

Bloomberg, one of the largest private networks globally, deals with numerous pieces of data and has 14,000 different applications on the Terminal. However, despite the infrastructure team working on delivering infrastructure as a service while spinning up many VMs, they were facing issues related to productivity, scaling, latency, scaling, efficiency, and automation.

> **We needed to deploy things in a uniform way across the entire network, and we wanted our private cloud to be as easy to use as the public cloud.”**
>
> Andrey Rybka
> Head of CTO Compute Architecture at Bloomberg

The team evaluated multiple frameworks and zeroed in on Kubernetes in 2016, even though it was still in alpha then. It turned out to be a great move. Bloomberg benefited in terms of improved resiliency of services, more productive developers, fewer errors with applications deployment, automation without much effort, and improved resource management. Ultimately,

> **Moving to Kubernetes is only a small piece of the puzzle. What you're talking about is moving to cloud native application development, deployment, and maintenance. So you still have to have all the rest of the stuff in place to enable people to do continuous integration and deployment, build container images and deploy them.”**
>
> Steven Bower
> Data and Analytics Infrastructure Lead at Bloomberg

4. Booking.com

Booking.com migrated to an OpenShift platform in 2016. However, they found that scaling the necessary support was not sustainable or feasible. The search for a new solution led the platform team to build its own vanilla Kubernetes platform and customize it to suit their needs.

Booking runs Kubernetes in many clusters in multiple data centers across the regions where it has been computed.

> **This is not a magical platform. We're not claiming that you can just use it with your eyes closed. Developers need to do some learning, and we're going to do everything we can to make sure they have access to that knowledge.”**
>
> Ben Tyler
> Principal Developer, B Platform Track At Booking.Com

The platform also includes CNCF technologies like Helm, Envoy, and Prometheus. Most of the critical service traffic is routed through Envoy, and Prometheus mainly monitors infrastructure components. The team also developed and open-sourced an extension for Kubernetes, Shipper, to add more complex rollout strategies and multi-cluster orchestration.

They also had internal discussions about the sensibility of building a Kubernetes platform from the ground up.

> **This is not really our core competency—Kubernetes and travel, they're kind of far apart, right? But we've made a couple of bets on CNCF components that have worked out really well for us. Envoy and Kubernetes, in particular, have been really beneficial to our organization. We were able to customize them, either because we could look at the source code or because they had extension points, and we were able to get value out of them very quickly without having to change any paradigms internally."**
>
> Ben Tyler
>
> Principal Developer, B Platform Track At Booking.Com

5. Capital One

As one of the largest banks in the US, Capital One has applications managing millions of transactions a day and making big-data decisions—for credit approvals, fraud detection, and beyond. The cloud team started with Docker first and then two years down the line embraced Kubernetes for its provisioning platform.

> **Kubernetes and its entire ecosystem are very strategic for us. We use Kubernetes as a substrate or an operating system, if you will. There's a degree of affinity in our product development."**
>
> John Swift
>
> Former Senior Director of Software Engineering at Capital One

The team wanted to consistently have the tools in the same ecosystem rather than having a large custom snowflake ecosystem where every tool required its own custom deployment. Now, they possess the tools to be autonomous in deployments, increasing deployments by two orders of magnitude.

> **And that was with just seven dedicated resources, without needing a whole group sitting there watching everything. That's a huge cost savings. With the scalability, the management, the coordination, Kubernetes really empowers us and gives us more time back than we had before."**
>
> Jamil Jadallah
> Former Scrum Master at Capital One

Are you ready for the big leap now!?

We have seen that organizations, varying in size and industry like retail, financial services, media, eCommerce, technology, are taking advantage of various technology services that the cloud-native landscape offers. And the foundation of many cloud-native journeys is Kubernetes. As a result, we are witnessing an enormous change in how software is built and deployed.

Before adopting Kubernetes, you should ask yourself the following questions to affirm its practicality.

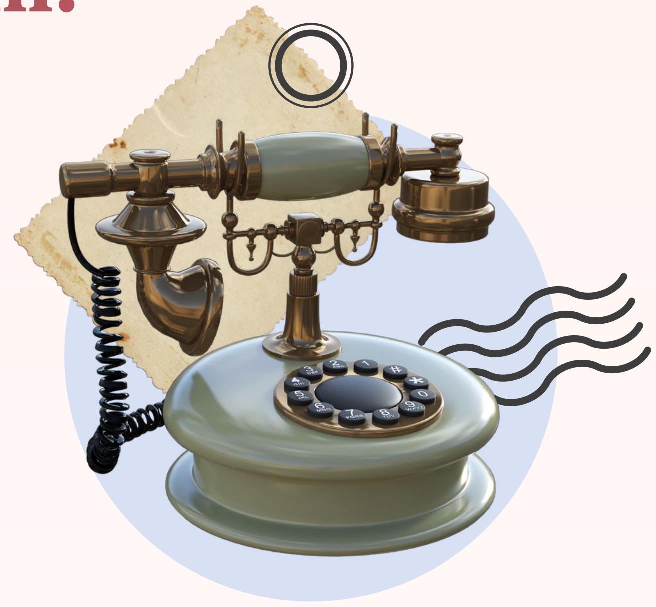
- Is your development workflow clearly defined?
- Is Kubernetes better than what you have? Do you actually need to stop using your monolithic architecture (if that's what you have currently)? Will it help break down your monolith?
- Can you keep pace with the Kubernetes release cycle?
- Does your enterprise have a reliable cloud foundation?
- What plan do you have to manage your databases and monitor your apps?
- Will Kubernetes adoption reduce your costs and improve service delivery speed?

Still, something as complex as automating application deployment can be quite challenging. Now you know what can potentially go wrong while adopting Kubernetes. And also how to mitigate those risks and rip the benefits.

However, the right Kubernetes consulting services can provide better quality and efficiency of software development and infrastructure management. Simform is an accomplished Kubernetes services provider with experience in delivering unmatched IT solutions to diverse industries. Apart from the K8s consulting service, we have extensive knowledge and expertise in cloud migration and services, microservices architecture, data and analytics consulting, DevOps, and more. So let Simform be your IT partner in embracing Kubernetes and container technology!

We are Simform!

With over 10+ years of experience under our belt, we are more than ready to supercharge your project with extraordinary code. 10 years ago, Simform was one person. Today, we're over 600+ people strong and growing.



Simform is a custom software development powerhouse. Let's get in touch to discuss your next project!

Contact Us

Managed software engineering teams

Quality Engineering and Testing

Cloud native development and modernization

DevOps