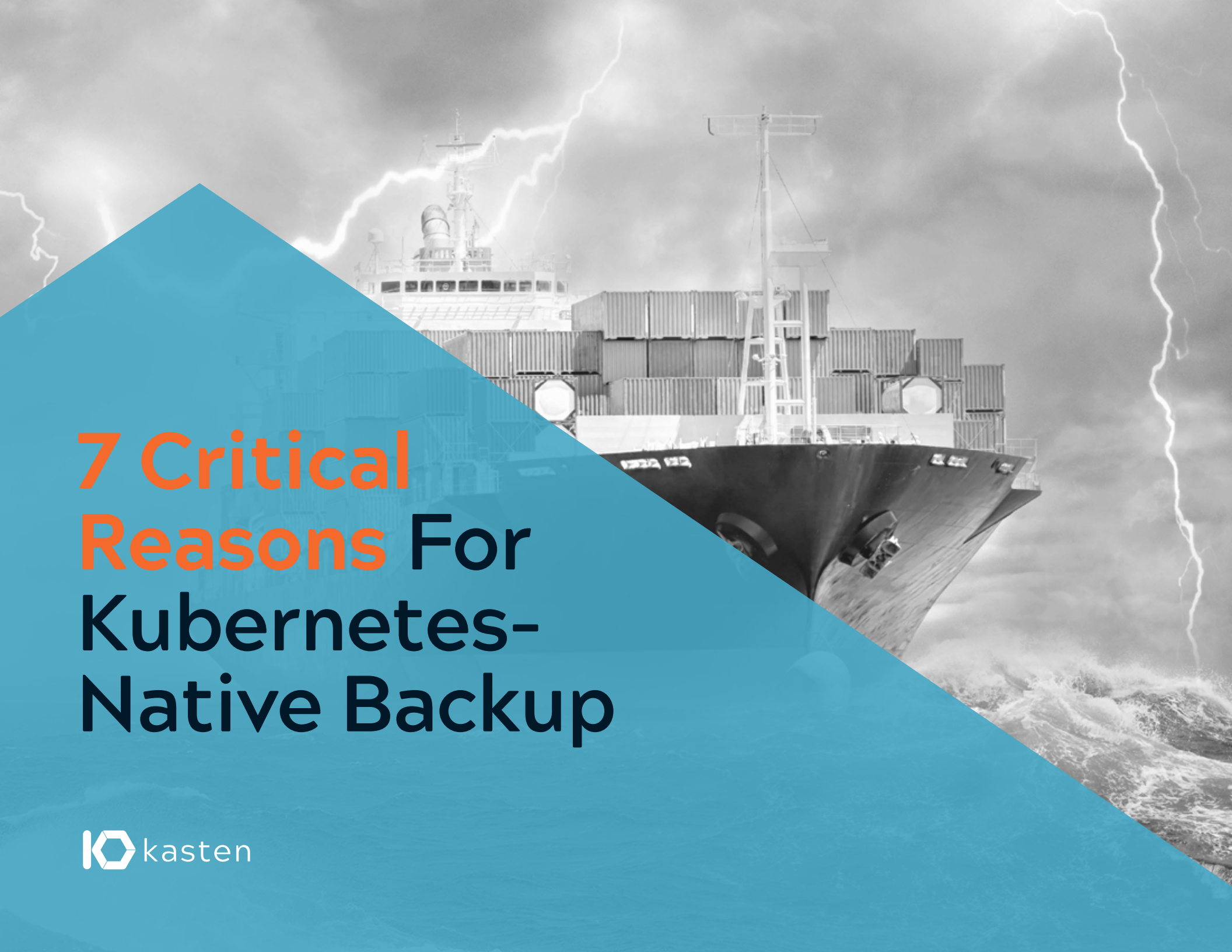


7 Critical Reasons For Kubernetes-Native Backup

kasten

Introduction

As cloud-native adoption and usage continues to explode, Kubernetes-native backup is one of the most critical, yet overlooked, challenges faced by Kubernetes operators today. Kubernetes, the fastest-growing infrastructure platform, is quickly becoming the foundation for all applications, no matter where they might be deployed. Its ubiquitous nature has it on track to be the next enterprise platform of choice, joining the ranks of Linux and vSphere. Given the increased agility, portability, and reliability seen by development teams using Kubernetes, there has been a sudden influx of applications onto the platform including stateful applications such as databases and NoSQL systems.

Due to this rapid application growth and increased production deployments at scale, enterprises are now focused on “Day 2” production challenges such as data management, security, and observability. While Kubernetes takes away a lot of the pain of ensuring high availability and scalability of your application services, these benefits do not extend to your data, making data management of Kubernetes applications a critical priority.

To accelerate your Kubernetes journey, this report explores key considerations when evaluating a strategy to protect your business critical data in a developer-focused platform. It covers the pitfalls of trying to retrofit legacy backup architectures into a cloud-native ecosystem and, more importantly, focuses on the benefits of deploying a truly cloud-native backup solution.

“As soon we started to deploy production workloads on Kubernetes, providing backup and recovery for our cloud-native applications and data in a Kubernetes-native way was among the top requirements for our users. This is not something that we had expected.

MICHAEL COURCY
DevOps Architect
Sopra Steria

Kubernetes Backup Myths

One of the biggest Kubernetes myths is the belief that applications running on the platform should be architected to be stateless. While this might have been true in the project's infancy, the platform's support for storage and stateful applications has reached maturity and is driving users to adopt Kubernetes as the common control plane for both stateful and stateless applications. With low-friction and self-service dynamic storage provisioning available to application developers, you commonly find extensive storage use in Kubernetes clusters that weren't built with stateful applications in mind. The widespread deployment of databases and NoSQL systems on Kubernetes without the right data management system in place will not only leave your data exposed, but can also place your business at risk.

Running applications on Kubernetes, stateful or stateless, gained popularity because the platform works very hard to ensure high availability in the face of software, server, or region failures. For stateful applications, this increased reliability makes it significantly easier to run replicated data services across failure domains. However, although it is often confused for backup, the added reliability provided by replication only improves data availability. While it is true that data replication can protect against partial infrastructure failure, it will not protect against corruption or data loss, whether accidental or malicious. An error that accidentally deletes a database will get replicated as quickly as the infrastructure allows and cause catastrophic data loss. Truly protecting a Kubernetes application cannot rely on replication alone, it requires storing a copy of the data in a completely independent location.

50%

of Top 10 containers running in Kubernetes are stateful including relational databases and NoSQL systems[1]

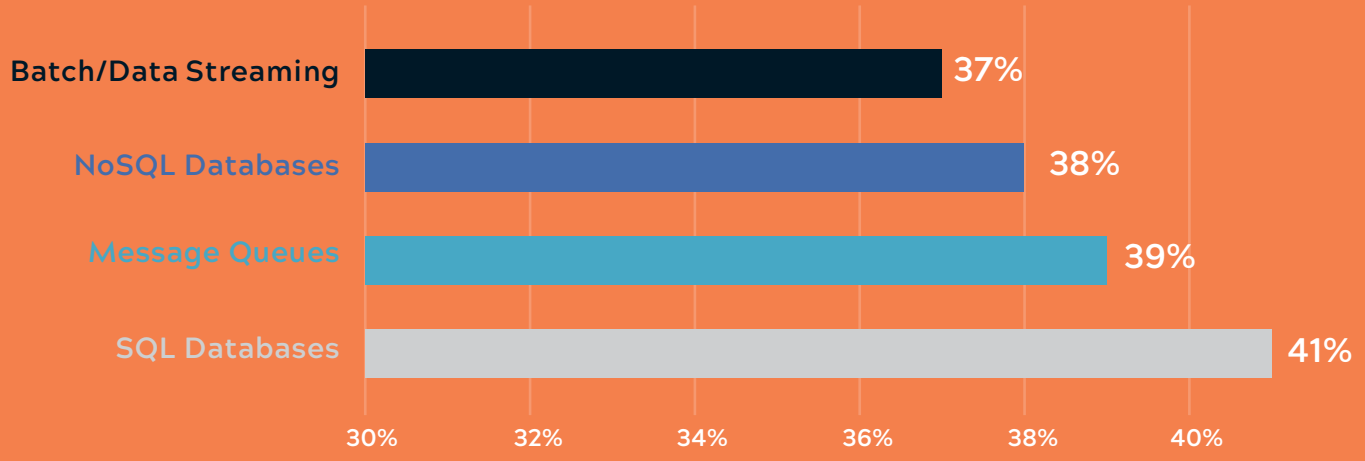
57%

of monitored clusters are running StatefulSets, a powerful Kubernetes primitive to manage stateful applications[1]

70+

Kubernetes storage drivers available to dynamically provision block and file volumes in public cloud and on-premises environments[2]

Stateful Application Type Deployed on Kubernetes[3]



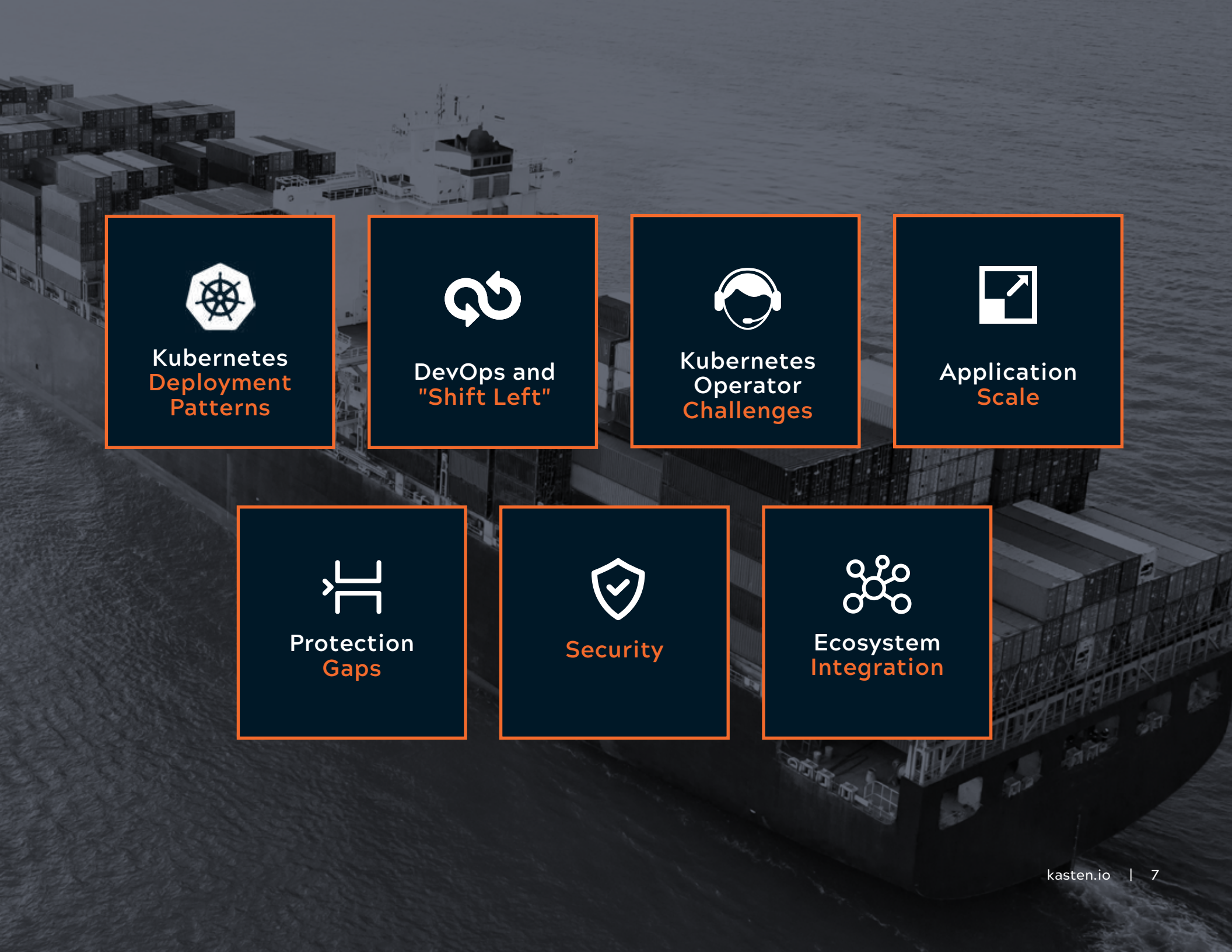
[1]2019 Container Usage Report, Sysdig
[2]Kubernetes CSI Project
[3]Kubernetes Application Survey, 2018

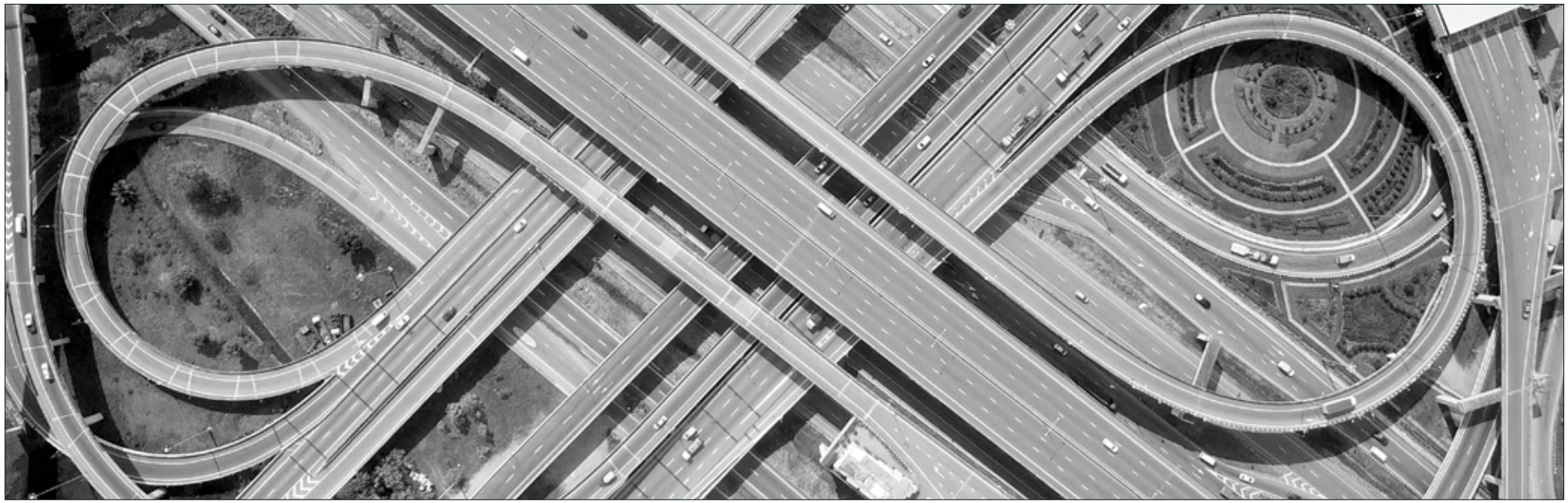
Seven Reasons Why Kubernetes-Native Backup is Critical

If it has not already, Kubernetes will soon increase its footprint within your infrastructure and will become the foundation for your next generation of applications. Ensuring that your critical applications have a cloud-native backup solution in place will allow your DevOps teams to move quickly by reducing work, also giving your infrastructure and ops teams a higher confidence in being able to recover from failure.

It might be tempting to delegate the backup and restore responsibility to legacy tools built for virtualization-based infrastructure or, in extreme cases, even assigning the responsibility to each application team. However, multiple customer journeys have shown that while this might be well intentioned, it is the wrong path to follow. Not only will you experience data loss if the backup responsibility is disaggregated, the high recovery time from manual and error-prone playbooks required with legacy approaches will add excessive risk. When the average hourly cost of critical application failure is $500,000, this is not a risk organizations can afford or should tolerate!

Our customer research, highlighted on the next page, shows that there are seven critical reasons why a Kubernetes-native backup solution will be the best fit for your expanding Kubernetes environment.


Kubernetes Deployment Patterns
DevOps and "Shift Left"
Kubernetes Operator Challenges
Application Scale
Protection Gaps
Security
Ecosystem Integration



Kubernetes Deployment Patterns

The most significant reasons motivating the need for a Kubernetes-native backup solution are the fundamental differences between the Kubernetes platform and all compute infrastructure that has come before it.

One of the biggest deployment changes for containerized applications is that there is no mapping of applications to servers or VMs. Kubernetes uses its own placement policy to distribute application components across all servers for fault-tolerance and performance. Further, different applications are often co-located on the same server. In comparison, traditional data management systems will fail as they can never independently capture the state of just a particular application without pulling in unrelated applications.

Another constant for cloud-native applications is the dynamic nature of their environment. Containers can be dynamically rescheduled or scaled on different nodes for better load balancing. New deployments, that happen on an hourly basis, can involve rolling upgrades, and new application components can be added or removed at any time. In short, the definition of a cloud-native application is constantly shifting. A backup solution needs to understand this cloud-native architectural pattern, be able to work with a lack of IP address stability, and be able to deal with continuous change.

With these major changes in the compute environment, trying to adapt appliance-centric solutions designed for VMs and use them with cloud-native platforms will fit as well as a square peg in a round hole. A Kubernetes-native backup solution is needed so that requirements such as dynamic application discovery, instantaneous backup, platform-integrated recovery, and capturing all application context can be met.

DevOps and "Shift Left"

The success of Kubernetes has been driven by its focus on developers, their applications, and high-velocity application development cycles. While seemingly subtle, this decision permeates the platform's design and consequently requires that backup solutions be application-centric and not infrastructure-focused. The application is what both the developer and operator ultimately care about.

The DevOps philosophy adopted in parallel with Kubernetes also cedes control over both infrastructure and deployments to the developer (known as "shift left"). Developers define both application components and infrastructure requirements (e.g., storage or load balancers) as code. These programmatic requests are provisioned dynamically, via a CI/CD pipeline, and without an extensive change management process.

This great power comes with the increased risk that a simple configuration error could delete critical data! Any backup system today needs to not just integrate into CI/CD systems but, more importantly, be able to discover new and changed applications automatically in order to instantly protect them. This process also needs to be completely transparent to developers, and it shouldn't require them to make changes to their applications, tools, packaging, or deployment pipeline.

Finally, as developers take on more responsibility, a backup platform not only needs to be API-first, but its API needs to be cloud-native itself. Using a Kubernetes-native API, instead of older REST or SOAP APIs, allows for seamless authentication and authorization, simplifies application and workflow integration, and allows for the use of tools (e.g., kubectl) developers and operators are already intimately familiar with.

Kubernetes Operator Challenges

Infrastructure teams moving past early Kubernetes adoption and being tasked with providing Kubernetes infrastructure at scale are frequently running into a skills gap. For those teams migrating to Kubernetes from a Linux or vSphere background, a backup tool that provides CLI access and a clean API along with a powerful yet easy-to-use dashboard is critical. A well-designed backup solution will accelerate an IT team's production journey by providing a bridge to greater Kubernetes understanding. Deep Kubernetes integration will hide underlying platform complexity, and a sharp focus on UX and revisiting backup workflows for cloud-native applications will reduce or eliminate manual or integration work.

A backup solution for containerized applications also needs to be application-centric and understand Kubernetes constructs instead of being infrastructure-focused. A single application that might have been composed of a few VMs is now, on average, composed of 100s of distinct Kubernetes resources (configuration, disks, secrets, etc.). When multiplied across all applications in a cluster, an operator is faced with understanding and protecting millions of components, unless the operational unit for backup is switched over to be the application. Ignoring Kubernetes resources is not an option as a legacy backup solution that restricts itself to just infrastructure such as disks and volumes will suffer from error-prone recovery playbooks with very high recovery times because of the missing relationships. Even with an unrealistic assumption that there has been no drift in Kubernetes objects between backup and restore, the initial manual process to determine the backups required for restore will have to be followed by another complicated manual process to graft these restored volumes back into the Kubernetes application, placing undue burden on operations teams.

Application Scale

With the rise of microservices and first-class Kubernetes support for functions such as configuration and secret handling, applications have been broken up into hundreds of discrete components that have independent lifecycles and are only visible to Kubernetes. Only a cloud-native backup solution built to handle the millions of components found in large clusters will understand the relationships between applications, their data, and related Kubernetes state, and be able to consistently capture all of it together at scale.

Additionally, both Kubernetes and cloud-native applications have been architected to scale up (or down) in response to load. An effective backup solution must: adopt the same cloud-native architectural pattern to scale with application and cluster changes, be able to effectively "scale to zero" when not in use, and do it automatically without manual operator input. This will result in better performance as the backup platform grows with the cluster, plus cost savings as the backup system's resource footprint is correlated to instantaneous requirements and not peak load. Scaling of the backup system will also be linear with cluster and application growth and will not exhibit the step-function jumps seen with an appliance-based model.

The above scale challenges are further compounded with the growth of Kubernetes multi-cluster use. Multiple clusters are found not just across environments (dev, staging, prod, etc.) but are often being split across application, security, and team boundaries, and being deployed across multiple availability zones, regions, clouds, and on-premises data centers. Reducing the operational burden for ops teams is only possible with a cloud-native backup platform that seamlessly handles multi-cluster operations and provides global visibility.

Protection Gaps

Kubernetes is architected for fault-tolerance, which makes it dramatically easier to ensure application uptime when faced with partial infrastructure outages. However, high availability or replication is still not a backup! Data corruption or deletion, accidental or malicious, will spread to all replicas and cause catastrophic data loss.

Given the popularity of running Kubernetes on public clouds, there is often the belief that the underlying storage is failure-proof. However, this is not true, and even AWS's battle-hardened Elastic Block Storage (EBS) advertises a non-zero annual failure rate! Similarly, on-premises storage vendors also provide volume snapshots. However, these volume snapshots are often not resistant to hardware failure and, even worse, the deletion of a volume usually leads to simultaneous and automatic deletion of all related snapshots.

Looking under the hood, actions such as quiescing file system activity require elevated Kubernetes security privileges and aren't normally available. A Kubernetes-native backup platform can provide database and Kubernetes workload quiescing hooks, achieving the same result without sacrificing security. Finally, while it is tempting to push the backup and recovery responsibility to development teams, we find that not all developers understand backup and disaster recovery. Their justified focus on their application has the unfortunate side effect of creating custom and poorly maintained backup solutions. Further compounding this issue, these ad-hoc systems often suffer from silent or noisy but ignored failures as the application and infrastructure continues to evolve and complexity grows. To mitigate this risk, a backup solution needs to work transparently against a wide range of Kubernetes application stacks and deployment methods while integrating into development workflows whenever required.

Security

Kubernetes includes a number of security features such as network policies that deny access to internal application components and their associated data services from not just outside the cluster but also to other untrusted applications running in the same cluster. Running backup solutions outside of your Kubernetes clusters becomes a non-starter as they cannot discover, let alone access (e.g., to quiesce), applications without weakening isolation policies. A well-architected Kubernetes-native solution that can embed itself into the control plane will not suffer from these limitations.

Further, with developers taking on more infrastructure responsibilities, traditional IT is transitioning to an "ITOps" model and needs to provide self-service capabilities. A backup platform now needs to allow for scoped control to be handed off to developers for their applications. Not only is fine-grained Role-Based Access Control (RBAC) a requirement but this scoped access needs to be granted using the same roles and tools defined by Kubernetes instead of introducing additional role management systems that operators and developers need to learn.

Kubernetes also delegates data encryption to the underlying storage system and backup platform. To ensure that application data is never transferred or stored in plain text, a backup platform needs to understand Kubernetes certificate management, work with storage-integrated Key Management Systems (KMSs), and support Customer Managed Encryption Keys (CMEKs) through the Kubernetes Secrets interface.

Finally, given the popularity of Kubernetes and the fact that a number of Kubernetes applications are external customer-facing, Kuberentes-targeted malware and ransomware attacks will remain a persistent threat. A Kubernetes-native solution that creates reliable backups independent of Kubernetes and the storage system and has deep platform integrations for quick automated restores will be essential.

Ecosystem Integration

The rise of polyglot persistence, where multiple data services (e.g., MongoDB, MySQL, and Cassandra) are used within the same application, has coincided with the growth of Kubernetes. Richer backups for these workloads are now possible by integrating against Kubernetes for automated workload discovery. Workload knowledge enables the backup solution to select the capture primitive (e.g., one or more of volume snapshots, application-consistent backups, logical dumps) best suited to the application's requirements.

With the shift from a single data service to multiple independent ones, the relationship between these workloads can be derived automatically from Kubernetes metadata. With this application topology in hand, a Kubernetes-native backup solution can capture a consistent copy (both within and across services) of the entire application stack, identify and gather data from replicas to reduce application impact and improve performance and efficiency, optimize restore performance by leveraging Kubernetes parallelism, and much more!

Similarly, with an increasing number of organizations running a large number of Kubernetes clusters across different environments, it is crucial for a backup platform to interoperate with the rest of the cloud-native infrastructure ecosystem. This not only improves the UX, reduces costs, and improves the efficiency of ops teams, it also provides integration with the cloud-native tools both developers and operators have become accustomed to. Examples include integration with Prometheus for monitoring and alerting or with the Kubernetes APIs for RBAC, logging, and auditing for root-cause analysis.

What Is My Risk?

The reality is that there are far more stateful workloads running on Kubernetes today than people expect. The public case studies are the proverbial tip of the iceberg and this was driven home in a recent survey by Cloud Native Computing Foundation (CNCF) that showed that over 40% of respondents were already using storage with Kubernetes today while 55% of the remaining respondents were planning on using it. As additional context, the vast majority of these respondents (84%) were already using containers in production.

At the end of the day, Kubernetes is a fundamentally different compute platform. Apart from the traditional reasons for data loss in a cloud environment, Kubernetes increases the risk of accidental data loss due to higher complexity, relative unfamiliarity, and dispersed administrative responsibility. If you don't already have a production-grade backup and recovery solution deployed that is built for Kubernetes, we strongly recommend that you use the insights gathered from this report to investigate potential solutions today.

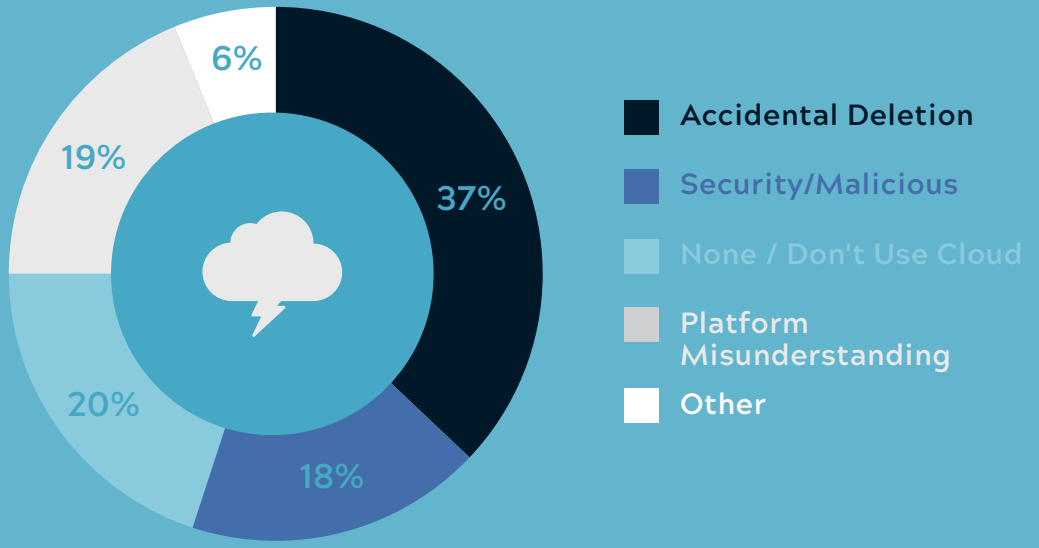
95%

2019 Cloud Native Computing Foundation (CNCF) survey respondents already using, evaluating, or planning storage use in Kubernetes

41%

Companies with > 1,000 employees who experienced a significant infrastructure outage or degradation in the past 12 months (451 Research)

Reasons for Data Loss in the Cloud



Conclusion

You have already made a smart infrastructure decision by deploying Kubernetes within your environment. As you broaden your production deployments, we encourage you to use the guidelines covered in this report to take a second look at your Kubernetes Data Management strategy and ensure that you have no gaps when it comes to application and data protection. We are confident that, on further discovery, you will find a Kubernetes-native backup solution to be a superior solution for your containerized applications.

If you found this report useful, we encourage you to share it with your network:

Learn more about K10, Kasten's Kubernetes backup and disaster recovery platform at:
https://kasten.io/product/

kasten

Kasten
289 S. San Antonio Road
Suite 204
Los Altos, CA, 94022

+1 (415) 851-1767
info@kasten.io
@kastenhq