Technology Trends

Kubernetes

Enterprise Architecture, Chief Technology Branch

Version 0.1
Date 2019-5-9

Shared Services Canada
Services partagés Canada

Canada

Table of Contents

Business Brief

Application containers are increasingly being used in the infrastructure for cloud-native and microservice applications. Specifically, Docker is the engine most commonly used to create containers. Many application developers have noted the value of containers in dependency management as it packages the application and its dependencies, libraries, other binaries into a container to abstract away the differences in the OS distribution and underlying infrastructure. This solves many issues caused by differences in runtime environments when a software moves from one environment to another one.

Containers improve applications' portability and scalability that enable applications to be released and updated in an easy and fast way without downtime. However, there still exits a demand in the management of containers when the services provided need to be deployed across a cluster of host servers to achieve high-availability and disaster recovery. This is where container deployment in cluster and management tools like Kubernetes provide their value. Developers can now begin to deploy and orchestrate services as a collection of containers across a cluster of servers. Container resource requirements can be explicitly declared that allows developers to bundle application code with an environment configurations. Also by increasing container density, resources can be used more efficiently and thus it in turn improves hardware usage [1]. Containers provide applications with isolations, so that a development team can be made responsible for specific containers.

Kubernetes also known as K8s, is a portable, extensible open-source platform for managing containerized workloads and services that facilitates both declarative configuration and automation. Kubernetes provides a container-centric management environment. It orchestrates computing, networking, and storage infrastructure on behalf of user workloads. Kubernetes also offers self-healing, automatized rollout and rollback features, which greatly improve operation high-availability and flexibility. One of the biggest advantages of Kubernetes is the flexibility it provides. Many PaaS packaging dictate specific frameworks, are catered towards specific workloads, or impose limitations which language runtimes can be used [1]. These issues are all eliminated with Kubernetes. Therefore, if an organization's application is capable of being run on a container, Kubernetes is a viable option for container orchestration.

Technical Brief

The Kubernetes cluster or deployment can be broken down into several components. The Kubernetes "master" is the machine in charge of managing other "node" machines. The "node is the machine in charge of actually running tasks fed to it via the user or the "master". The master and nodes can be either a physical or virtual machines. In each Kubernetes cluster, there is one master and multiple nodes machines. The main goal of Kubernetes is to achieve "Desired State Management". The

"master" is fed a specific configuration through its RESTful API which it exposes to the user, and the "master" is then responsible for running this configuration across its set of "node". The nodes can be thought of as host of containers. They communicate with the "master" through the agent in each node --"Kubelet" process. To establish a specific configuration in Kubernetes, the "master is fed a deployment file with the ".yaml" extension. This file contains a variety of configuration information. Within this information are "Pods" and "replicas". There is a concept of Pod in Kubernetes and it can be described as a logic collection of containers which are managed as a single application. Resources can be shared within a Pod, these resources include shared storage (Volumes), a unique cluster of IP addresses, and information about how to run each container. A Pod can be thought of as the basic unit of the Kubernetes object model, it represents the deployment of a single instance of an application in Kubernetes [8]. A Pod can encapsulate one or more application containers. Two models exist for how Pods are deployed within a cluster. The "one-Pod-per-container" means a single pod will be associated with a single container. There can also be multiple containers that run within a single Pod, where these containers may need to communicate with one another as they share resources. In either model, the Pod can be thought of as a wrapper around the application containers. Kubernetes manages the Pod instances rather than managing the containers directly. The Pods are run on the Node machines to perform tasks. Replicas are simply instances of the Pods. Within the ".yaml" deployment file, specifications are instructing the "master" machine how many instances/replicas of each Pod to run, which is handled by a replication controller [8]. When a node dies or a running Pod experiences an unexpected termination, the replication controller will take note take care of this by creating the appropriate number of Pods [8].

Industry Use

Kubernetes is an open source system and many companies have begun to adopt it into their existing architecture as well as adapt it to their specific needs. It was originally developed by Google and was made an open source project in 2014. The Cloud Native Computing Foundation is a project of the Linux Foundation providing a community for different companies who are seeking to develop Kubernetes and other container orchestration projects. Several major cloud providers and platforms including Google Cloud Compute, HP Helion Cloud, RedHat Openshift, VMware Cloud, and Windows Azure all support the use of Kubernetes [7]. A survey, performed by iDatalabs in 2017, found 2,867 companies are currently using Kubernetes. These companies are generally located in the United States and are also most the computer software industry. Companies on the list hire between 50 and 200 employees, and accumulate 1M-100M in revenue per year. Some of the major companies on this list include GoDaddy inc, Pivotal Software inc, Globant SA, and Splunk inc. Kubernetes own approximately 8.6% of the market share within the virtualization management software category [9].

Canadian Government Use

There is a lack of documented Government of Canada (GC) initiatives and programs promoting the current and future use of Kubernetes technology. As a GC strategic IT item, Kubernetes is absent from both the GC's Digital Operations Strategic Plan: 2018-2022 and the GC Strategic Plan for Information Management and Information Technology 2017 to 2021. This may be due to the fact that the GC is currently grappling with the implementation of Cloud Services, and the majority of resources and efforts are occupied with implementation challenges, as well as security concerns related to the protection of the information of Canadians.

However, the inception of containers into the market has shown that large-scale organizations, who are involved in cloud-native application development as well as networking, can benefit greatly from the use of containers [7]. Although the infrastructure applications providing cloud services can be based solely on Virtual Machines (VMs), the maintenance costs associated with running different operating systems on individual VMs outweighs the benefit [6]. Containers and Containerization is a replacement and/or complimentary architecture for VMs. As the GC moves toward cloud services and development of cloud-native applications, the use of containers and orchestrating them with Kubernetes can become an integral part the GC IT architecture.

Implications for Shared Services Canada (SSC)

Value Proposition

The primary business value impact of Kubernetes is the technology's portability, and mobility independent of the environment. Its ability to manage, and orchestrate an organization's application containers is a marked benefit. Kubernetes secondary business value is that it enables enterprise high-velocity, meaning that every product team can safely ship updates many times a day, deploy instantly, observe results in real time, and use this feedback to roll containers forward or back with the goal to improve the customer experience as fast as possible.[i]

In the age of modern web services, users expect their applications to be available 24/7, and developers expect the ability to deploy new versions of those applications several times a day with minimal downtime. Containers have become one of the main ways in which to manage applications across enterprise IT infrastructure and also one of the most difficult areas to manage effectively.

Kubernetes, as an open source system, is a technology that can administer and manage a large number of containerized applications spread across clusters of servers while providing basic mechanisms for deployment, maintenance, and scaling of applications.[ii] An application container is a standard unit of software that packages code and all its dependencies so the application runs quickly and reliably from one computing environment to another.[iii] Kubernetes automates the distribution and scheduling of application containers across a cluster in a more efficient way.[iv]

Containers offer a logical packaging mechanism in which applications can be abstracted from the environment in which they actually run. This decoupling allows container-based applications to be deployed easily and consistently, regardless of whether the target environment is a private data center, the public cloud, or even a developer's personal laptop.[v] An additional benefit to containerization is that the Operating System (OS) is not running as hard.

Since Kubernetes is open source, it allows the enterprise freedom to take advantage of on-premises, hybrid, or public cloud infrastructure, and the ability to effortlessly move workloads.[vi] Containerized applications are more flexible and available than in past deployment models, where applications were installed directly onto specific machines as packages deeply integrated into the host. Kubernetes groups containers that make up an application into logical units for easy management and discovery. The abstractions in Kubernetes allows deployment of containerized applications to a cluster without tying them specifically to individual machines (i.e. Virtual Machines). Applications can be co-located on the same machines without impacting each other. This means that tasks from multiple users can be packed onto fewer machines. This provides greater efficiency and reduces the cost on hardware as less machines are used.

Kubernetes contains tools for orchestration, secrets management, service discovery, scaling and load balancing and includes automatic bin packing to place containers with the optimal resources, and it applies configurations via configuration management features.[vii] It protects container workloads by rolling out or rolling back changes and offers availability and quality checks for containers -- replacing or restarting failed containers. As requirements change, a user can move container workloads in Kubernetes from one cloud provider or hosting infrastructure to another without changing the code.[viii] This is a great value to developers as their work is protected and an audit trail of changes is available.

The core concepts of Kubernetes which enables high velocity are immutability, declarative configuration and self-healing systems.[ix]

Containers and Kubernetes encourage developers to build distributed systems that adhere to the principles of immutable infrastructure. In immutable infrastructure an artifact created, will not be changed upon user modifications. To update applications

in an immutable infrastructure, a new container image is built with a new tag, and is deployed, terminating the old container with the old image version. In this way, the enterprise always has an artifact record of what was done and if there was an error in the new image. If an error is detected the container is rolled back to the previous image.[x] Anything that goes into a container has a text file. Text files can be treated like application source code and provisions immutability.

Declarative configuration enables the user to describe exactly what state the system should be in. Traditional tools of development such as source control, unit tests etc. can be used with declarative configurations in ways that are impossible with imperative configurations. Imperative systems describe how to get from point A to B, but rarely include reverse instructions to get back. Kubernetes declarative configuration makes rollbacks fairly easy which is impossible with imperative configurations.[xi]

Lastly, Kubernetes has a means of self-healing. When Kubernetes receives a desired state configuration, it does not simply take actions to make the current state match the desired state at a single time, but it will continuously take actions to ensure it stays that way as time passes by.[xii]

Challenges

The greatest challenge in regards to Kubernetes is its complexity. However, security, storage and networking, maturity, and competing enterprise transformation priorities are also challenges facing the Kubernetes technology.

Kubernetes Complexity and Analyst Experience

There is the challenge of a lack of organizational and analyst experience with container management and in using Kubernetes. Managing, updating, and changing a Kubernetes cluster can be operationally complex, more so if the analysts have a lack of experience. The system itself does provide a solid base of infrastructure for a Platform as a Service (PaaS) framework, which can reduce the complexity for developers. However, testing within a Kubernetes environment is still a complex task. Although its use cases in testing are well noted, testing several moving parts of an infrastructure to determine proper application functionality is still a more difficult endeavour [1]. This means a lot of new learning will be needed for operations teams developing and managing Kubernetes infrastructure. The larger the company, the more likely the Kubernetes user is to face container challenges.[xiii]

Security

In a distributed, highly scalable environment, traditional and typical security patterns will not cover all threats. Security will have to be aligned for containers and in the context of Kubernetes. It is critical for operations teams to understand Kubernetes security in

terms of containers, deployment, and network security. Security perimeters are porous, containers must be secured at the node level, but also through the image and registry. Security practices in the context of various deployment models will be a persistent challenge.[xiv]

Storage & Networking

Storage and networking technologies are pillars of data center infrastructure, but were designed originally for client/server and virtualized environments. Container technologies are leading companies to rethink how storage and networking technologies function and operate.[xv] Architectures are becoming more application-oriented and storage does not necessarily live on the same machine as the application or its services. Larger companies tend to run more containers, and to do so in scaled-out production environments requires new approaches to infrastructure.[xvi]

Some legacy systems can run containers and only sometimes can VMs can be replaced by containers. There may be significant engineering consequences to existing legacy systems if containerization and Kubernetes is implemented in a legacy system not designed to handle that change. Some Legacy systems may require refactoring and making it more suitable for containerization. Some pieces of a system may be able to be broken off and containerized. In general, anything facing the internet should be run in containers.

Maturity

Kubernetes maturity as a technology is still being tested by organizations. For now, Kubernetes is the market leader and the standardized means of orchestrating containers and deploying distributed applications. Google is the primary commercial organization behind Kubernetes; however they do not support Kubernetes as a software product. It offers a commercial managed Kubernetes service known as GKE but not as a software. This can be viewed as both a strength and a weakness. Without commercialization, the user is granted more flexibility with how Kubernetes can be implemented in their infrastructure; However, without a concrete set of standards of the services that Kubernetes can offer, there is a risk that Google's continuous support cannot be guaranteed. Its donation of Kubernetes code and intellectual property to the Cloud Native Computing Foundation does minimize this risk since there is still an organization enforcing the proper standards and verifying services Kubernetes can offer moving forward [1]. It is also important to note that the organizational challenges that Kubernetes users face have been more dependent on the size of the organization using it.

Kubernetes faces competition from other scheduler and orchestrator technologies, such as Docker Swarm and Mesosphere DC/OS. While Kubernetes is sometimes used to manage Docker containers, it also competes with the native clustering capabilities of Docker Swarm.[xvii] However, Kubernetes can be run on a public cloud service or on-

premises, is highly modular, open source, and has a vibrant community. Companies of all sizes are investing into it, and many cloud providers offer Kubernetes as a service.[xviii]

Competing Enterprise Transformation Priorities

The last challenge facing Kubernetes initiative development and implementation is its place in an organization's IT transformation priority list. Often there are many higher priority initiatives that can take president over Kubernetes projects.

Considerations

Strategic Resourcing and Network Planning

A strategic approach to Kubernetes investments will need to be developed to ensure opportunities are properly leveraged. The GC invests a significant portion of its annual budget on IT and supporting infrastructure. Without strategic Kubernetes direction the fragmented approaches to IT investments, coupled with rapid developing technology and disjointed business practices, can undermine effective and efficient delivery of GC programs and services.[xix] A clear vision and mandate for how Kubernetes will transform services, and what the end-state Kubernetes initiative is supposed to look like, is a prominent consideration.

SSC should consider defining a network strategy for Kubernetes adoption. Multiple factors should be taken into account, including the amount of resources, funding, and expertise that will be required for the development and experimentation with Kubernetes technologies. Calculation of resource requirements including CPU, memory, storage, etc. at the start of Kubernetes projects is imperative. Considerations include whether or not an in-house Kubernetes solution is required or if a solution can be procured. Other strategy considerations include analyzing different orchestration approaches for different application use cases.

Complexity and Skills Gap

Kubernetes is a good technology and the de facto standard for orchestrating containers, and containers are the future of modern software delivery. But it is notoriously complex to manage for enterprise workloads, where Service Level Agreements (SLAs) are critical. The operational pain of managing production-grade Kubernetes is further complicated by the industry-wide talent scarcity and skills gap. Most organizations today struggle to hire Kubernetes experts, and even these "experts" lack advanced Kubernetes experience to ensure smooth operations at scale. SSC will need to be cautious in implementing Kubernetes and having the right staff experienced and comfortable in its use.

Customization and Integration Still Required

Kubernetes technology and ecosystem are evolving rapidly, because of its relatively new state, it is hard to find packaged solutions with complete out-of-the-box support for complex, large-scale enterprise scenarios. As a large and sophisticated enterprise organization, SSC will need to devote significant resources on customization and training. Enterprise Architecture pros will need to focus on the whole architecture of cloud-native applications as well as keep a close watch on technology evolution and industry.

Implementation usually takes longer than expected, however the consensus in the New Stack's Kubernetes User Experience Survey is that Kubernetes reduces code deployment times, and increases the frequency of those deployments.[xx] However, in the short run, the implementation phase does consume more human resources. Additionally, implementation takes longer than expected. The consensus is that Kubernetes reduces code deployment times, and increases the frequency of those deployments. However, in the short run, the implementation phase does consume more human resources.

Pilot Small and Scale Success

SSC may wish to consider evaluating the current Service Catalogue in order to determine where Kubernetes can be leveraged first to improve efficiencies, reduce costs, and reduce administrative burdens of existing services as well as how a new Kubernetes service could be delivered on a consistent basis. Any new procurements of devices or platforms should have high market value and can be on-boarded easily onto the GC network. SSC should avoid applying in-house Kubernetes for production mission-critical apps. Failure of in-house deployments is high and thus should be avoided. SSC should pilot and establish a Kubernetes test cluster. With all new cloud-based technologies, piloting is preferred. Focus should first be on a narrow set of objectives and a single application scenario to stand up a test cluster.

Implement Robust Monitoring, Logging, and Audit Practices and Tools

Monitoring provides visibility and detailed metrics of Kubernetes infrastructure. This includes granular metrics on usage and performance across all cloud providers or private data centers, regions, servers, networks, storage, and individual VMs or containers. Improving data center efficiency and utilization on both on-premises and public cloud resources is the goal. Additionally, logging is a complementary function and required capability for effective monitoring is also a goal. Logging ensures that logs at every layer of the architecture are all captured for analysis, troubleshooting and diagnosis. Centralized, distributed, log management and visualization is a key capability.[xxi] Lastly, routine auditing, no matter the checks and balances put in place, will cover topics that normal monitoring will not cover. Traditionally, auditing is as a manual process, but the automated tooling in the Kubernetes space is quickly improving.

Security

Security is a critical part of cloud native applications and Kubernetes is no exception. Security is a constant throughout the container lifecycle and it is required throughout the design, development, DevOps, and infrastructure choices for container-based applications. A range of technology choices are available to cover various areas such as application-level security and the security of the container and infrastructure itself. Different tools that provide certification and security for what goes inside the container itself (such as image registry, image signing, packaging), Common Vulnerability Exposures/Enumeration (CVE) scans, and more.[xxii] SSC will need to ensure appropriate security measures are used with any new Kubernetes initiatives, including the contents of the containers being orchestrated.

References

[i] Jayanandana, Nilesh. (May 2nd, 2018). *Benefits of Kubernetes*. Medium Newspaper. Retrieved 16-May-2019 from: https://medium.com/platformer-blog/benefits-of-kubernetes-e6d5de39bc48

[ii] GitHub. (2019). *Production-Grade Container Scheduling and Management*. GitHub. 2019. Retrieved 16-May-2019 from: https://github.com/kubernetes/kubernetes

[iii] Docker. (2019). *What is a Container? A Standardized Unit of Software.* Docker Inc. 2019.Retrieved 16-May-2019 from: https://www.docker.com/resources/what-container

[iv] Kubernetes. (2019). *Using Minikube to Create a Cluster*. Kubernetes. 2019. ICP license: 京ICP备17074266号-3. Retrieved 16-May-2019 from: https://kubernetes.io/docs/tutorials/kubernetes-basics/create-cluster/cluster-intro/

[v] https://cloud.google.com/containers/

[vi] Kubernetes. (2019). *Production-Grade Container Orchestration.* Kubernetes. 2019. ICP license: 京ICP备17074266号-3. Retrieved 16-May-2019 from: https://kubernetes.io/

[vii] Rouse, Margaret, et al. (August 2017). *Kubernetes*. TechTarget Inc. 2019. Retrieved 16-May-2019 from: https://searchitoperations.techtarget.com/definition/Google-Kubernetes

[viii] Rouse, Margaret, et al. (August 2017). *Kubernetes*. TechTarget Inc. 2019. Retrieved 16-May-2019 from: https://searchitoperations.techtarget.com/definition/Google-Kubernetes

[ix] Jayanandana, Nilesh. (May 2nd, 2018). *Benefits of Kubernetes*. Medium Newspaper. Retrieved 16-May-2019 from: https://medium.com/platformer-blog/benefits-of-kubernetes-e6d5de39bc48

[x] Jayanandana, Nilesh. (May 2nd, 2018). *Benefits of Kubernetes*. Medium Newspaper. Retrieved 16-May-2019 from: https://medium.com/platformer-blog/benefits-of-kubernetes-e6d5de39bc48

[xi] Jayanandana, Nilesh. (May 2nd, 2018). *Benefits of Kubernetes*. Medium Newspaper. Retrieved 16-May-2019 from: https://medium.com/platformer-blog/benefits-of-kubernetes-e6d5de39bc48

[xii] Jayanandana, Nilesh. (May 2nd, 2018). *Benefits of Kubernetes*. Medium Newspaper. Retrieved 16-May-2019 from: https://medium.com/platformer-blog/benefits-of-kubernetes-e6d5de39bc48

[xiii] Williams, Alex, et al. *Kubernetes Deployment & Security Patterns.* The New Stack. 2019. 20180622. thenewstack.io. Retrieved 15-May-2019 from: https://thenewstack.io/ebooks/kubernetes/kubernetes-deployment-and-security-patterns/

[xiv] Williams, Alex, et al. *Kubernetes Deployment & Security Patterns.* The New Stack. 2019. 20180622. thenewstack.io. Retrieved 15-May-2019 from: https://thenewstack.io/ebooks/kubernetes/kubernetes-deployment-and-security-patterns/

[xv] Williams, Alex, et al. *Kubernetes Deployment & Security Patterns.* The New Stack. 2019. 20180622. thenewstack.io. Retrieved 15-May-2019 from: https://thenewstack.io/ebooks/kubernetes/kubernetes-deployment-and-security-patterns/

[xvi] Williams, Alex, et al. *Kubernetes Deployment & Security Patterns.* The New Stack. 2019. 20180622. thenewstack.io. Retrieved 15-May-2019 from: https://thenewstack.io/ebooks/kubernetes/kubernetes-deployment-and-security-patterns/

[xvii] Rouse, Margaret, et al. (August 2017). *Kubernetes*. TechTarget Inc. 2019. Retrieved 16-May-2019 from: https://searchitoperations.techtarget.com/definition/Google-Kubernetes

[xviii] Tsang, Daisy. (February 12th, 2018). *Kubernetes vs. Docker: What Does It Really Mean?* Sumo Logic. 2019. Retrieved 16-May-2019 from: https://www.sumologic.com/blog/kubernetes-vs-docker/

[xix] Treasury Board of Canada Secretariat. December 3, 2018. *Directive on Management of Information Technology*. Treasury Board of Canada Secretariat. Government of Canada. Retrieved 27-Dec-2018 from: https://www.tbs-sct.gc.ca/pol/doc-eng.aspx?id=15249

[xx] Williams, Alex, et al. *The State of the Kubernetes Ecosystem.* The New Stack. thenewstack.io. Retrieved 15-May-2019 from: https://thenewstack.io/ebooks/kubernetes/state-of-kubernetes-ecosystem/

[xxi] Chemitiganti, Vamsi, and Fray, Peter. (February 20th, 2019). *7 Key Considerations for Kubernetes in Production*. The New Stack. 2019. Retrieved 16-May-2019 from: https://thenewstack.io/7-key-considerations-for-kubernetes-in-production/

[xxii] Chemitiganti, Vamsi, and Fray, Peter. (February 20th, 2019). *7 Key Considerations for Kubernetes in Production*. The New Stack. 2019. Retrieved 16-May-2019 from: https://thenewstack.io/7-key-considerations-for-kubernetes-in-production/

1. Clayton, T. and Watson, R. (2018). Using Kubernetes to Orchestrate Container-Based Cloud and Microservices Applications. [online] Gartner.com. Available at: https://www.gartner.com/doc/3873073/using-kubernetes-orchestrate-containerbased-cloud [Accessed 6 Jul. 2018].

2. Anon, (2018). [online] Available at: https://www.infoworld.com/article/3173266/containers/4-reasons-you-should-use-kubernetes.htm [Accessed 6 Jul. 2018].

3. Contino. (2018). Who's Using Docker?. [online] Available at: https://www.contino.io/insights/whos-using-docker [Accessed 6 Jul. 2018].

4. Team, S. and Team, S. (2018). 3 Open Source Container Management Tools (Comparison & Review). [online] Shadow-Soft. Available at: https://shadow-soft.com/open-source-container-management-tools/ [Accessed 6 Jul. 2018].

5. MSV, J. (2018). Kubernetes: An Overview - The New Stack. [online] The New Stack. Available at: https://thenewstack.io/kubernetes-an-overview/ [Accessed 6 Jul. 2018].

6. Heron, P. (2018). Experimenting with containerised infrastructure for GOV.UK - Inside GOV.UK. [online] Insidegovuk.blog.gov.uk. Available at: https://insidegovuk.blog.gov.uk/2017/09/15/experimenting-with-containerised-infrastructure-for-gov-uk/ [Accessed 6 Jul. 2018].

7. CENGN. (2018). CENGN and CloudOps Collaborate to Train Industry on Docker and Kubernetes. [online] Available at: https://www.cengn.ca/docker-kubernetes-training-jan18/ [Accessed 13 Jul. 2018].

8. Kubernetes.io. (2018). Kubernetes Basics - Kubernetes. [online] Available at: https://kubernetes.io/docs/tutorials/kubernetes-basics/ [Accessed 13 Jul. 2018].

9. Idatalabs.com. (2018). Kubernetes commands 8.62% market share in Virtualization Management Software. [online] Available at: https://idatalabs.com/tech/products/kubernetes [Accessed 13 Jul. 2018].