INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact Factor: 6.078
(Volume 9, Issue 2 - V9I2-1192)
Available online at: https://www.ijariit.com

Performance evaluation of Kubernetes cluster federation using Kubefed

Ben-Salem Banguena E.
beloumbe@gitam.in
Gandhi Institute of Technology and Management University, Visakhapatnam, Andhra Pradesh

Dr. T. Uma Devi
luckypavankumar666@gmail.com
Gandhi Institute of Technology and Management University, Visakhapatnam, Andhra Pradesh

ABSTRACT

We have entered the multi-cloud and hybrid age. The inevitable trend in cloud computing is application-oriented multi-cloud and multi-cluster architecture. Today's cloud applications must abide by a wide range of laws and rules. It is doubtful that a single cluster can follow all the rules. The scope of compliance for each cluster is decreased by the multiple cluster technique. We can move workloads between Kubernetes suppliers to benefit from new features and costs. This paper aims to describe an integration between multiple clusters running on the same cloud and evaluate their performance based on the Kubernetes Cluster Federation system. Some experimental evaluations were carried out with this goal in mind (Cloud Evaluation Experiment Methodology – CEEM) to monitor system resource behavior and availability, including network, disk, CPU, and memory. The test environment consists of a manually deployed Kubernetes cluster that was created. Azure Kubernetes Service (AKS) is the Cloud service provider considered. The Cluster Federation was performed using the Kubernetes Cluster Federation (KubeFed).

Keywords: *Cluster, Container, Federation, Kubefed, Kubernetes, Virtualization.*

I. INTRODUCTION

The pay-per-demand service model used by cloud computing makes it more popular with users [1]. The primary benefit of on-demand service is generating efficiencies for both consumers and providers of cloud services [2]. As businesses grow, there is an eventual need to scale the system. This proliferation could also be due to other reasons such as multi-provider strategies, geographical constraints, and computational usage. Most of these systems, at present, exploit some container technology, such as Kubernetes, which help them manage and orchestrate their workloads on different worker nodes which constitute a cluster. Kubernetes (also known as K8s) is a portable, expandable, open-source platform to manage containerized workloads and services that support declarative configuration and automation. It has a large, rapidly growing ecosystem [3]. Kubernetes clusters are growing in number and size inside organizations. In recent years, Kubernetes has replaced containers as the de facto infrastructure management standard [4]. Resiliency, usability, and portability are the three most significant issues that are been currently faced, and these programs (de-facto) should address them.

A company may use two or more cloud computing platforms as part of a multi-cloud strategy to accomplish various objectives. Additionally, it enables businesses to effectively manage expenses, concentrate on capital and operational expenditures, and take advantage of affordable public cloud and infrastructure providers [5]. To maximize the advantages of each specific service, businesses that don't want to rely on a single cloud vendor might leverage resources from multiple vendors. A multi-cloud architecture can offer improved cost-effectiveness, dependability, and scalability, among other advantages. However, those advantages come with costs. The explanation is straightforward: configuring and managing more clouds makes things more difficult. Multi-Cloud models require more significant interaction between various clouds and services, more management of accounts, attention to vendor-specific tools and procedures, etc. Additionally, integrating and maintaining the complexity gets even more complicated if your multi-cloud strategy involves a hybrid cloud (as it does if you have on-premises infrastructures or private clouds running alongside public clouds). Numerous solutions have emerged in this situation, and Kubernetes has quickly emerged as the industry standard for container orchestration.

Configuring and maintaining a Kubernetes infrastructure can be discouraging, despite its proven effectiveness. Since many providers offer Kubernetes solutions that are more or less complicated, their evaluation is increasingly crucial, for example, for the orientation of future improvements.

© 2023, www.IJARIIT.com All Rights Reserved

This paper describes integrating multiple clusters on Microsoft Azure Kubernetes Services (AKS) [6]. It evaluates their performance character while exploring the ability of the Kubernetes Federation project to perform the following scenarios: avoiding vendor lock-in, high availability, and simplified manageability, and. Azure is the only cloud provider left offering a free master node. First, Kubernetes clusters are manually deployed and connected. Following a defined procedure, the performance of common computer resources is then monitored using open-source benchmarking tools., including system memory, API server requests, etcd requests, and work queue processing times. The Cloud Evaluation Experiment Method (CEEM) [7] is used to direct the evaluation. To ensure its traceability and reproducibility, the work is organized according to a strict methodology (CEEM). The evaluation logic is easily adaptable to new settings and includes a complete description of how these experiments were carried out.

Besides this introduction, five additional sections are included in this paper. Section 2 aims to familiarize the reader with concepts related to federation and multi-cluster while summarizing related work. In addition to displaying a cluster architecture set-up and outlining the performance tests, Section 3 gives an evaluation approach. Section 4 presents the preliminary findings as well as the pertinent discussion. Finally, Section 5 summarizes the study and gives final remarks regarding this approach and potential areas for future work.

II. RELATED WORKS

The main advantage of Kubernetes technology, especially when optimizing cloud-native application development, is that it provides a dedicated platform for scheduling and running containers on cluster machines. A method for deploying an application on or across several Kubernetes (K8s) clusters is known as multi-clustering [8]. Today organizations increasingly deploy Kubernetes clusters, which they consider disposable [9]. There are many scenarios, depending upon requirements, where the need for multiple clusters becomes a necessity; a few such scenarios are as follows:
Low latency: Kubernetes clusters in multiple regions minimizes the latency as users are served content from clusters nearest to their locations.
Fault isolation: multiple small clusters instead of a single large cluster simplifies fault isolation in case of failure.
Scalability: user demand drives the scalability needs of a system.
Hybrid cloud: prevent provider lock-in by having multiple clusters on cloud providers or on-premises data centres.
Business isolation: maintaining separate clusters for different business domains facilitates the decoupling of services and provides better performance when compared to the multi-tenant architecture that relies solely on the presence of namespaces.

Strong separation ensures that essential operational activities like cluster and application upgrades are simplified. Isolation can also help to decrease the blast radius of a cluster failure. Tenants can be routed to their cluster in organizations with strict tenancy isolation requirements. Multi-cluster enables the deployment of global applications in or across various availability zones and regions, increasing application availability and improving regional performance. Today's cloud applications must adhere to a slew of rules and norms. It is improbable that a single cluster can comply with all regulations. The scope of compliance for each cluster is reduced when using a multi-cluster technique. A multi-cluster approach allows your company to move workloads across multiple Kubernetes suppliers to take advantage of new features and prices [10].

As individual clusters can be customized to conform to specific regional or certification regulations, multi-clustering may be necessary to comply with competing laws. With independent development teams delivering apps on segregated clusters and selectively exposing available services for testing and release, the speed and security of software delivery can also be boosted. However, multi-cloud architectures are highly complex and challenging to monitor and manage. K8s allows you to centralize multi-cloud management, making it convenient and efficient. Kubernetes allows for the extension of a cluster across several clusters and clouds, and for handling these multi-cluster deployments, the federated Kubernetes architecture is advised [11].
There are many cloud federation solutions. The most popular is KubeFed [12], [5], a native Kubernetes solution. Its operation is detailed in Fig 1.

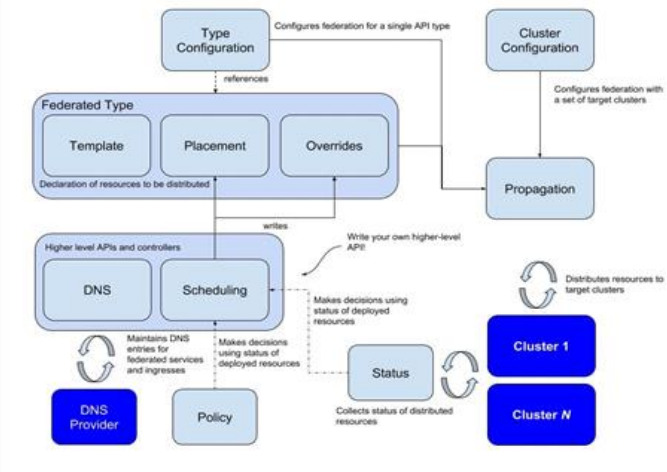

Fig. 1. Kubernetes cluster federation architecture (source: github.com)

© 2023, www.IJARIIT.com All Rights Reserved

The Federation v2 project, headed by Red Hat, includes a controller for pushing federated items and a means to transform any Kubernetes API type to a multi-cluster federated variety. Role-Based Access Control (RBAC) [13] policies and other configuration information will be pushed to various clusters by Federation v2. These resource categories are fixed, and each cluster's configuration policy format is comparable. On the other hand, a multi-cloud delivery system frequently has a more complex decision-making logic. Overall, several experts advise against employing Kubernetes Federation Cluster (KubeFed) in real-world applications [14]. Due to their reliance on virtualization technologies, cloud-based infrastructures are an ideal environment for efficiently scaling up or down nodes, such as nodes in a Kubernetes cluster. Due to the flexibility the various Cloud service providers provide, consumers can supply computer resources as needed and only pay for what they use.

According to Gigi Sayfan [15], capacity overflow, sensitive workloads (opposite of capacity overflow), avoiding vendor lock-in, and geo-distributing high availability are the four use cases that benefit from cluster federation. The configuration support methods (for various K8s clusters) are provided by KubeFed [12] when used as a multi-cluster manager from a single control plane in a hosting cluster. Comparing the performance of workloads running in containers vs those running on virtual machines is a common theme in the literature. For instance, many studies have conducted a performance evaluation of containerized-based cloud systems. As in earlier research [16], [17], several benchmarking tools were utilized to access the performance overheads of various system resources, such as disc I/O, CPU, RAM, and network. Therefore, a specific approach to evaluating cloud services should have distinguished the various processes in detail. The study [18] specifically suggested a five-step process and extended the ASTAR method [19].

The systematic literature analysis found that most assessors did not precisely define or specify their evaluation methods. On the Microsoft Azure cloud platform, manually deployed clusters were used to establish a baseline for this study. The closest work to ours is [20]. We use a similar process to others and evaluate the effectiveness of the abovementioned services. In the following section, we provide the approach for doing this.

III. METHODOLOGY

The federation is the method used to spread World-Wide Applications across numerous areas and clouds. Using KubeFed as a multi-cluster manager provides configuration support mechanisms from a single control plane in a hosting cluster. It was determined that there is a need to assess the performance of this technology in the context of the federation because research on the use of Kubernetes (in the context of cloud computing) is still in an advanced stage. It is recognized that the evaluation of cloud services falls under the purview of experimental computer science, which necessitates using suitable evaluation methods to strategically direct experimental studies [18]. The Cloud Evaluation Experiment Methodology (CEEM) is necessary for this inquiry [7]. CEEM is an evaluating system for Cloud services, arranged in a ten-step methodology as illustrated in Fig 2.

1	Requirement Recognition	Recognize the problem, and state the purpose of a proposed evaluation
2	Service Feature Identification	Identify Cloud services and their features to be evaluated
3	Metrics and Benchmarks Listing	List al the metricsand benchmarks that may be used for the proposed evaluation
4	Metrics and Benchmarks Selection	Select suitable metricsand benchmarks for the proposed evaluation
5	Experimental Listing Factors	List all the factors that may be involved in the evaluation experiments
6	Experimental Factors Selection	Select lilited factors to study, and also choose level/ranges of these factors
7	Experimental Design	Design experiments based on the above work
8	Experimental Implementation	Prepare the experimental environment and perform the designed experiments
9	Experimental Analysis	Statistically analyze and interpret the experimental results
10	Conclusion and Reporting	Draw conslusions and report the overall evaluation procedure and results

Fig. 2. Different steps of the CEEM

The proposed system was implemented by creating three Kubernetes clusters on different hosted services. Deploying this system requires joining the clusters as a single logical cluster, especially using KubeFed and looking at its practical usage.

It should be made clear that this evaluation solely applies to the Kubernetes services offered by AKS, one of the numerous important cloud service providers active in the Infrastructure as a Service (IaaS) market [21]. Steps 1 through 7 of the pre-experimental process are carried out in this section. The section that comes after this one deals with steps 8 and 9. In part 5, the conclusions (step 10) are explained.

© 2023, www.IJARIIT.com All Rights Reserved

Requirement recognition
Due to the wide range of managed Kubernetes services offered across numerous Clouds, evaluating such solutions' performance is decisive for service providers and consumers, for example, to undertake cost-benefit evaluations and make plans for future improvements. A list of particular topics that future evaluation experiments would answer should be prepared according to the CEEM methodology. When conducting a performance evaluation, it is crucial to consider how dependable the performance is across various platforms.

Performance is defined in this context as the level of effectiveness we may anticipate from a containerized application while executing on a hosted cloud service.

Service feature identification
Performance, Economics, and Security have been the critical areas of concern when examining the procedures currently used to evaluate Cloud services [22]. However, depending on the formulation of the need, this evaluation is viewed from a performance angle.

Listing and selection of metrics and benchmarks
The choice of metrics plays a crucial role in evaluating evaluations, according to extensive research on evaluating classical computer systems [28]. A lookup capability or metrics and benchmarks have been built by employing Cloud service features. Prometheus [23] aids in keeping track of deployment activity and the resources available to Nodes, such as CPU, RAM, network latency, and disc I/O. Fig 3 illustrates the process. Benchmarking tools cannot be easily used across cluster deployments since containers do not offer an entire interactive desktop to execute applications. Prometheus was, therefore, a possibility. Prometheus is a high-scalable open-source monitoring framework and one of the projects managed by the Cloud Native Computing Foundation (CNCF). It provides out-of-the-box monitoring capabilities for the Kubernetes container orchestration platform. This study considers monitoring the API Server Request (latency and rate), the etcd request latency, and the work queue processing time in the clusters. In [20], a similar approach was discussed, in which different tools were used to run the benchmark. Measurements are taken from periods of static workload to those with the increased workload.

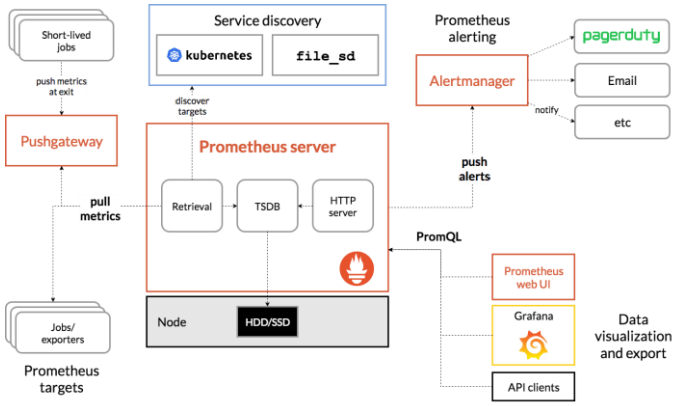

Fig. 3. Prometheus architecture (source: prometheus.io)

Listing and selection of experimental factors
This section evaluates the various elements that might affect the experiment's findings. The objective is to ensure that the federation project for Kubernetes accomplishes the factors mentioned above (cf. introduction). These elements must be appropriately identified to guarantee that the evaluation is traceable and reproducible.

High availability (HA)
Route traffic automatically away from hazardous clusters to maintain the health of services. Kubefed is not a workable HA option right now. When Kubernetes services are added or removed, it refreshes the DNS, however it does not update the DNS when a cluster goes down.

Simplified manageability
By controlling every cluster from a single kubectl context, you can minimize administrative work and guarantee consistency. It has been discovered that Kubernetes is a reliable and efficient way to update numerous clusters with a single "kubectl" command.

Avoiding vendor lock-in
Use your own private on-premises data centres, Azure, AWS, or GCP to run apps, or move them between data centres run by various organizations. We did not build Kubernetes clusters in AWS or GCP, but we do see a barrier to doing so. But an important distinction is that running clusters under various vendors is no longer necessitated with Kubernetes Federation.

Experimental Design

© 2023, www.IJARIIT.com All Rights Reserved

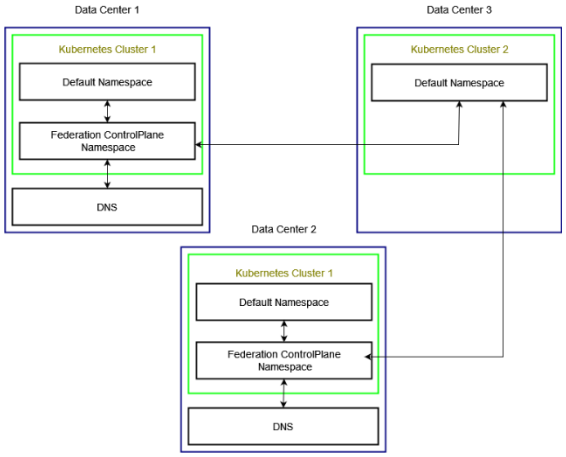

Fig. 4. Cluster federation via Kubefed

Understanding the solution requires understanding the topology that Kubefed produced. Fig 4 displays two Kubernetes clusters that are active in different Azure Data Centres, although Kubefed can federate 2-n clusters that are active anywhere. Any mix of on-premises (private), Azure, AWS, or GCP data centres, or two clusters in a single data centre, are acceptable topologies. On the Microsoft Azure cloud platform, two Kubernetes clusters were set up in different regions, and a federation was built between them. Azure DNS was used as the federation's external DNS even though the federation service still does not natively support Azure as a DNS provider. Deployment definition files (.yaml) were established after the Kubernetes environment was.

IV. RESULTS AND DISCUSSION

The evaluation findings are presented in this section, together with assessments of the critical results. Fig 5 and Fig 6 depict how the API server and the etcd are monitored. They include metrics like the processing time for the work queue and request latency.

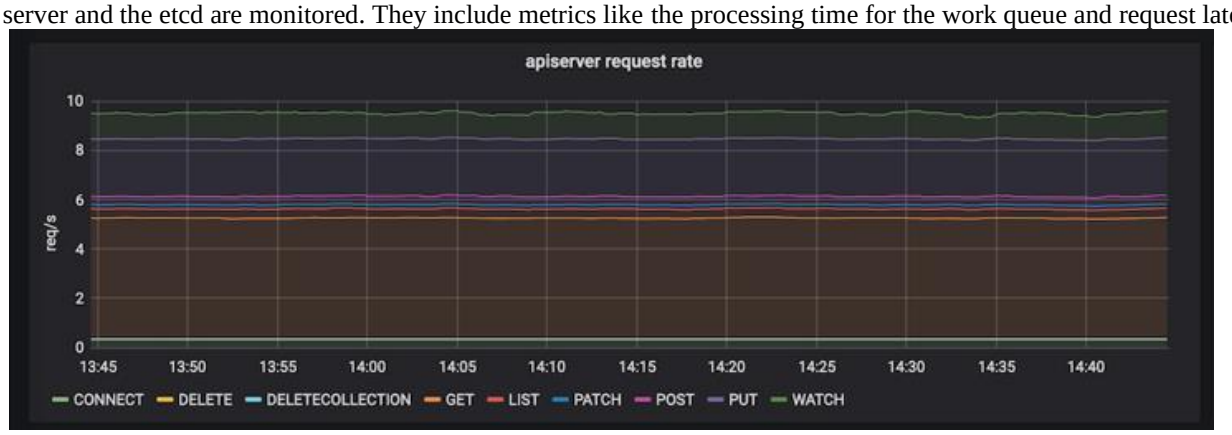

Fig. 5. API server request rate

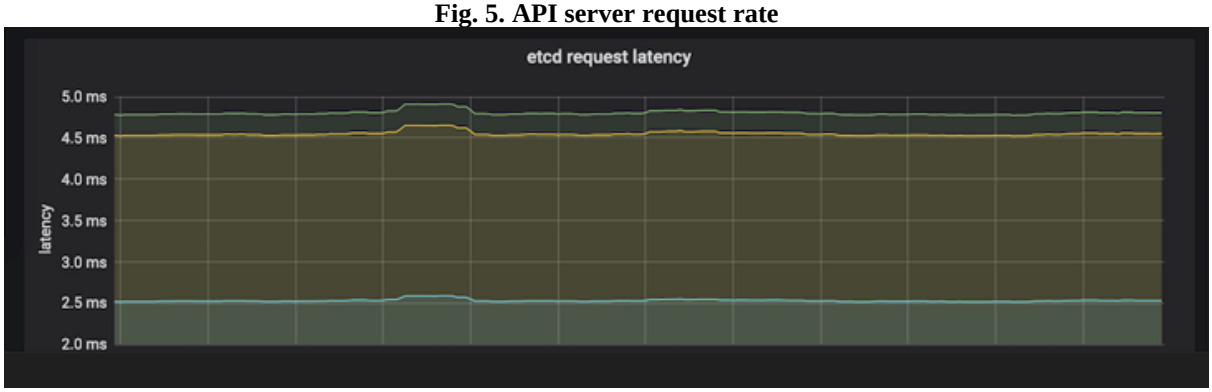

Fig. 6. Etcd request latency

High Availability (HA)

Kubefed does not have a High Availability solution, but "kubeadm" [24] does. Every public endpoint must be actively monitored for health like other DNS-based high availability solutions do. The recommendation is to use a DNS traffic management solution instead.

Simplified manageability

We question whether the Kubernetes federation simplifies management, even though it functions as stated, especially in comparison to the more directive and adaptable management made possible by continuous delivery solutions.

© 2023, www.IJARIIT.com All Rights Reserved

Avoiding vendor lock-in

Multi-vendor deployments do not seem to be any easier with Kubernetes Federation. In order to ensure cross-cluster communication, the federation may complicate deployments more than merely using CI/CD technologies.

V. CONCLUSION

This paper is all about an evaluation of a Kubernetes federation service deployed on a Cloud environment. With the emergence of numerous cloud-oriented solutions, the goal is to use the Kubefed project to investigate the behaviour of a cluster federation running on Microsoft Azure. The experimental evaluation indicates that the Kubernetes federation is an exciting concept. Still, we should be clear on what we're trying to solve by using it can avoid vendor lock-in and provide high availability and manageability. The difficulties include employing several methods to achieve the same outcomes, operations overhead, and weighing the advantages of using a single control location to manage jointed clusters under the federation against the difficulty of dealing with another abstraction. All of these things should be taken into account when determining the precise parameters for using Kubernetes to implement federation. This experiment adheres to the Cloud Evaluation Experiment Methodology (CEEM) process, which enables traceability and experimental repeatability. Its logical evaluation can be used in various scenarios. Measuring the performance of clusters federation, e.g., across different Cloud providers, could be a possible approach for future work.

VI. REFERENCES

[1] D. Rountree and I. Castrillo, "Introduction to the Cloud," The Basics of Cloud Computing, pp. 1–17, 2014, DOI: 10.1016/B978-0-12-405932-0.00001-3.
[2] P. Mell and T. Grance, "The NIST definition of cloud computing," in Cloud Computing and Government: Background, Benefits, Risks, 2011. DOI: 10.1016/b978-0-12-804018-8.15003-x.
[3] "Kubernetes." https://kubernetes.io/ (accessed Jul. 15, 2022).
[4] "Experts Discuss Top Kubernetes Trends and Production Challenges." https://www.infoq.com/articles/kubernetes-trends-and-challenges/ (accessed Jun. 14, 2022).
[5]"design-proposals-archive/architecture.md at main · kubernetes/design-proposals-archive." https://github.com/kubernetes/design-proposals-archive/blob/main/architecture/architecture.md (accessed Jul. 15, 2022).
[6] "Managed Kubernetes Service (AKS) | Microsoft Azure." https://azure.microsoft.com/en-us/services/kubernetes-service/ (accessed Jul. 15, 2022).
[7] Z. Li, L. Obrien, and H. Zhang, "CEEM: A practical methodology for cloud services evaluation," Proceedings - 2013 IEEE 9th World Congress on Services, SERVICES 2013, pp. 44–51, 2013, DOI: 10.1109/SERVICES.2013.73.
[8] M. Lukša, "Kubernetes in Action," in Kubernetes in Action, 2018. DOI: 10.3139/9783446456020.fm.
[9] "Kubernetes Is Everywhere, So What's Next? | DEVOPSdigest." https://www.devopsdigest.com/kubernetes-is-everywhere-so-whats-next (accessed Jul. 15, 2022).
[10] "Understanding Multi-Cluster Kubernetes | Ambassador Labs." https://www.getambassador.io/learn/multi-cluster-kubernetes/ (accessed Jun. 14, 2022).
[11] P. Kayal, "Kubernetes in Fog Computing: Feasibility Demonstration, Limitations and Improvement Scope : r," 2020. DOI: 10.1109/WF-IoT48130.2020.9221340.
[12] "kubernetes-sigs/kubefed: Kubernetes Cluster Federation." https://github.com/kubernetes-sigs/kubefed (accessed Jul. 15, 2022).
[13] "What is Kubernetes role-based access control (RBAC)." https://www.redhat.com/en/topics/containers/what-kubernetes-role-based-access-control-rbac (accessed Jul. 15, 2022).
[14] "aljannuzzi/Kubernetes-federation: A quick overview on how to federate multiple Kubernetes clusters on Azure." https://github.com/aljannuzzi/Kubernetes-federation (accessed Jul. 15, 2022).
[15] G. Sayfan, Mastering Kubernetes, 2nd ed. Packt, 2018.
[16] A. O. Dayo, "A Multi-Containerized Application using Docker Containers and Kubernetes Clusters," International Journal of Computer Applications, vol. 183, no. 44, pp. 55–60, 2021, DOI: 10.5120/ijca2021921843.
[17] L. Espe, A. Jindal, V. Podolskiy, and M. Gerndt, "Performance evaluation of container runtimes," 2020. DOI: 10.5220/0009340402730281.
[18] V. Stantchev, "Performance evaluation of cloud computing offerings," 2009. DOI: 10.1109/ADVCOMP.2009.36.
[19] V. Stantchev and C. Schröpfer, "Techniques for service level enforcement in web-services based systems," 2008. DOI: 10.1145/1497308.1497316.
[20] A. P. Ferreira and R. O. Sinnott, "A Performance Evaluation of Containers running on Managed Kubernetes Services", DOI: 10.1109/CloudCom.2019.00038.
[21]"Gartner Reprint." https://www.gartner.com/doc/reprints?id=1-271OE4VR&ct=210802&st=sb (accessed Jul. 15, 2022).
[22] Z. Li, H. Zhang, L. O'Brien, R. Cai, and S. Flint, "On evaluating commercial Cloud services: A systematic review," Journal of Systems and Software, vol. 86, no. 9, 2013, DOI: 10.1016/j.jss.2013.04.021.
[23]"Overview | Prometheus." https://prometheus.io/docs/introduction/overview/ (accessed Jul. 15, 2022).
[24]"Kubeadm | Kubernetes." https://kubernetes.io/docs/reference/setup-tools/kubeadm/ (accessed Jul. 15, 2022).

Authors Profile

© 2023, www.IJARIIT.com All Rights Reserved

Ben-Salem Banguena, obtained his Master Degree in Systems Management in 2016, at 3iL Limoges (France). Since September 2020, he pursuing a PhD in Department of Computer Science, School of Science, GITAM University, Andhra Pradesh, Visakhapatnam, India. His research interests include Big Data Analytics, Virtualization, Cloud Computing and Internet of Things. He is fluent in three languages (English, French, and Arabic), which allowed him to adapt in his multiple workplaces.

Associate Professor, Dr. T. Uma Devi is currently working as the Head of Department of Computer Science, GITAM School of Science, GITAM University, Andhra Pradesh, Visakhapatnam, India. Her major areas of interest are Neural Networks, Computer Security and Reliability, Human Computer Interaction.

© 2023, www.IJARIIT.com All Rights Reserved