ISSN 2347 - 3983
Volume 8. No. 1.1, 2020
International Journal of Emerging Trends in Engineering Research
Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter2181.12020.pdf
https://doi.org/10.30534/ijeter/2020/2181.12020

A Self-Healing Data Management Architecture for IoT Applications: A Case Study of Smart Urban Farming on High-Rise Buildings

R. Hamzah [1], N. Jamil[1], S. K. N. A. Rahim[1], A. R. M Asmuei[2], M. A. Mushaimi[2], M.H. M. Noor[3], H.A.Hamzah[4]. K. A. F. A.Samah[5]

[1]Faculty of Computer and Mathematical Sciences, University of MARA Technology , 40450 Shah Alam Selangor Malaysia, raseeda@uitm.edu.my

[2]KGR Solutions, 1st Floor, Jalan Suarasa 8/4, Lake Valley, Bandar Tun Hussein Onn, Cheras, 43200, Selangor, Malaysia, aizzadmushaimi@gmail.com

[3]BeePlus Agro, No 121, Jalan GunungNuang U11/18 40170 Seksyen 11 Shah Alam Selangor, Malaysia, beepulsagro@gmail.com

[4]International Islamic University 25200 Kuantan, Pahang Malaysia, hairulaini@iium.edu.my

[5]Faculty of Computer and Mathematical Sciences, UniversitiTeknologi MARA Cawangan Melaka KampusJasin, Melaka, Malaysia, khyrina783@uitm.edu.my

ABSTRACT

This paper presents the containerization of virtualization method that is done to execute distributed application for smart urban farming system. Containerization is a virtualization method for executing distributed applications without the need of virtual machines. However, when dealing with container downtime, a backup solution is needed to ensure that no data lost and to ensure container is in functioning. A container downtime might be caused by connectivity failure or maximum CPU capacity utilization. Therefore, a micro services-based system was developed by leveraging on containerization technology such as Docker for its lightweight footprint in terms of infrastructure resources and container orchestration capabilities from Kubernetes to enable scalability and reliability of the overall system. The results showed that efficient data monitoring and management with the added advantage of self-healing nature of Kubernetes platform was achieved.

Key words: Kubernetes; IoT, Self-Healing, Smart Urban Farming, Container

1. INTRODUCTION

Currently, the increasing awareness of organic or pollution and pesticide-free vegetables importance can be seen in consumers around the world. However, several factors have been recognized as the problem that limiting consumers from getting healthy vegetables. One of the problems is limited land space. In fact, researchers around the world have significantly focused on increasing productivity and reduce the environmental footprint within a framework of urban, indoor, climate-controlled high-rise buildings [1]. Since the shortage of land affects the food supply that is needed to cater the increasing population, smart urban farming is introduced as one of the solutions. Urban farming is a concept of farming in the city by growing food in a highly populated town. Urban farming is not only safe [2] but it is also cost-effective to help the urban poor to grow their own vegetables or fruit [3].

Although urban farming is becoming famous in providing solution on many traditional farming problems, it still requires human intervention to manage the farming, starting from sowing to harvesting. For urban and modern style farming, time management is quite an issue. In average, citizens in large cities spent roughly 11 hours of commuting and working time per day. Therefore, it is quite impossible to fully care and maintain the urban farm. The integration of information and communication technology (ICT) implies Internet of Things (IoT) to help human being managing their daily task remotely [4].

Developer has always facing several challenges when deploying software to IoT devices. IoT devices (e.g. sensors and controllers) are lack of powerful computing and memory resources that limiting their abilities to process software updates. IoT devices are used and also scattered across a large geographic area and may encounter internet disconnection or limited network bandwidth that making it difficult to receive software updates. Logically, software deployment or delivering on each of the IoT devices from one central source is ineffective as it will take up a lot of hardware and software capacity. Therefore, a technology that is called container is currently being applied in IoT software development. Container is one of the technologies that can solve this issue. To install/maintain a container is a lot easier than setting up a virtual machine to control all of the IoT devices. The containers' image registry is easy to be altered and manipulated (i.e. when container images are updated, Docker that is located in multiple location will downloads only the parts of the image that have changed).

Containers are packaged with all dependencies and software for IoT applications are portable, light and secure [5]. However, even though containers offer many advantages in IoT software development, it cannot run from downtime issue that usually caused by updates activities, maximum CPU capacity utilization and internet connection problem. The scale of the smart urban farming accommodating multiple IoT applications also makes it difficult to monitor and coordinate the containers running in different applications. After-deployment maintenance is also in demand for container-based IoT. A platform to orchestrate all these containers along with their varying workloads, computing, networking, and storage are needed. A manually monitored bundle of containers are also troublesome as it requires extra-effort and time consuming. Therefore, this work developed a self-healing Docker containerization architecture by using Kubernetes orchestration capabilities for a smart urban farming that ensures containers reliability.

In this research we conducted a case study of smart urban farming in a high-rise building which is situated in Shah Alam, one of the large cities in Malaysia. As with other cities across Peninsular Malaysia, Shah Alam experiences a tropical rainforest climate according to the Köppen climate classification [6]. Temperatures are consistent throughout the year with an average high temperature of 31.9 °C and an average low temperature of 23.2 °C. The city is warmest in the month of March, and experiences heavy rains and showers during the month of November as the northeast monsoon moves in from October to March. The house development of Shah Alam can be seen in several categories ranging from terrace, condominium, apartment as well as flat. Other aspects such as humidity and temperature also have become an issue to grow vegetable in their house. As most of them are working, it added more problems in the monitoring. The President of Baiduri Apartment in Shah Alam stated that most of them do not do farming in the house as they found that it is hard to manage and maintain as most of them are working.

The remainder of this paper is structured as follows. Section 2 discusses the background study of system deployment framework. The proposed self-healing data management architecture for smart urban farming is illustrated in Section 3. Then, we explain the results and discussion in Section 4 and lastly concluded the research in section 5.

2. RELATED WORKS

This section presents the deployment chronology of system starting from conventional to Kubernetes system and previous works related to this research to illustrate the importance of self-healing data management by using Kubernetes.

2.1 Conventional System

In the previous, applications arerunning on physical servers and there are no resource boundaries for applications in physical server. Hence, resource allocation issues happened. It can cause one application to take up most of the resources that result in other application underperformed. Researchers has found a solution to run each application on different physical server [7]. However, the resources were underutilized and costly and not scalable. In software development, cost reduction is an important aim of any organization [8].

2.2 Virtualized System

Virtualization is the process of transforming a layer of actual hardware concept into virtual instance. Virtualization has been invented since 1974 [9]. In virtualization technology, multiple Virtual Machine (VM) can be operated on a single physical server's CPU that it can save a lot of hardware cost as a set of physical resources can be viewed as a cluster of disposable virtual machines. Each VM is a complete machine operating all the components, including its own operating system, over the virtualized hardware. Since VM allows applications to be isolated between VMs, it eventually provides a level of promising security. The information of one application cannot be easily retrieved by another application. Virtualization also allows better utilization of resources and scalability in a physical server since an application can be added or updated easily.

2.3 Container System

Multiple containers are more practical to run in isolation on a host Operating System (OS). This is one of the advantages of container system as it can reduce the faults damage and maintenance as if one container down, it will not affect the other. This isolation function can be provided by the container-based technology where multiple individual containers shared single OS. The root file system that is needed on each container is independent to each other and they can share binaries and libraries system safely. Based on the lightweight architecture of container-based virtualization, containers offer several advantages over virtual machines such as high performance, resource efficiency, and agile environment. As a result of these advantages, containers have been adopted in the Information Technology (IT) industry in areas such as cloud data centers, mobile systems, and networks [10].

Although container system provide usefulness in the IT technology that can be seen in a lot of IoT applications, developers need to ensure each container that contains applications are always run without downtime. When a container down, developers need to troubleshoot and find the root cause of the downtime. There are two (2) possibilities of a downtime container that are: (1) application break or/and (2) high software and hardware utilization. If the downtime is causes by application break, the coding need to be fixed. However, if the downtime is causes by a high utilization, the container needs to be replicated manually. An alternative way needs to be figured out on ensuring a backup container for a down container and it is really good if it can be done automatically.

2.4 Kubernetes-Container System

A Kubernetes-container technology come into handy to solve the downtime issue of containers. Kubernetes is a framework that can offer service discovery and load balancing. Figure 1shows an architecture of Kubernetes cluster.

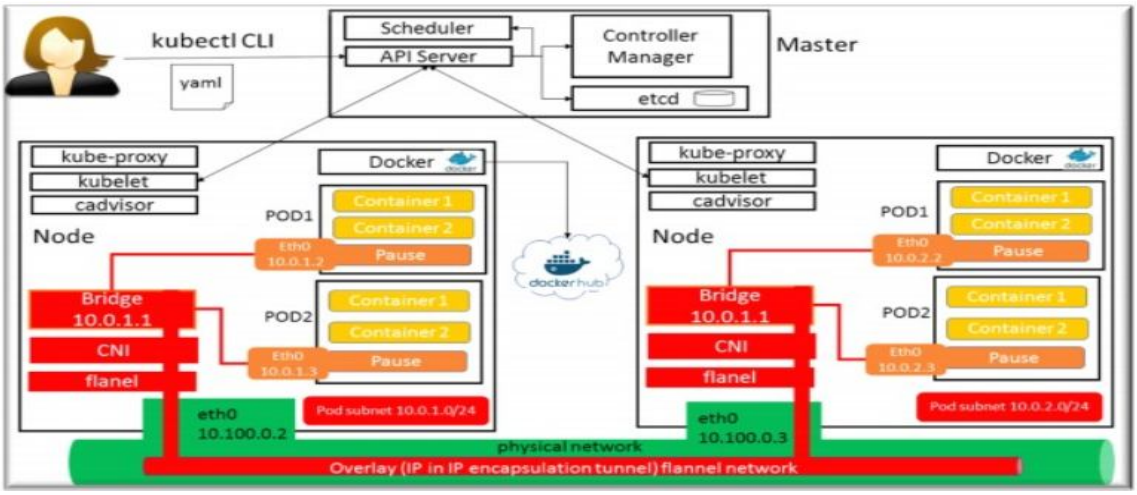

Figure 1: Kubernetes Cluster Architecture

3. PROPOSED SELF-HEALING DATA MANAGEMENT ARCHITECTURE

In this section, the architecture of proposed self-healing data management architecture of using Kubernetes is discussed. Prior to that, the requirements involved in this research regarding the space and system is presented.

3.1 Requirement of smart urban farming of a high-rise building

High Rise Building Farming Space Requirement

i. Farming kit can be placed anywhere regardless the source of the natural sunlight as long as it is exposed to some source of light (e.g. light bulb)

ii. Source of light is needed to store the energy in the battery so that it can be used to provide power to all sensors and controllers

System Requirement

i. System or application that is always up and running and run normal without breakdown.

ii. Continuous monitoring mechanism that can ensure the system runs in a normal behavior and can reports any abnormalities of the system

iii. Restore mechanism to ensure system back to normal functionalities without external assistance

3.2 Design of proposed self-healing data management architecture

The proposed self-healing data management architecture is shown in Figure2. In overall, the architecture is partitioned into hardware (i.e. shelf, sensors, controllers, solar and 3G sim card), and software (cloud, web applications and mobile application). First of all, the input sensor will detect and readthe physical parameter.

The analog digital converter (ADC) will then converts the analog electrical signal into a digital signal. A micro-computer that will publish the signal received from ADC and react according to the instruction received from IoT Cloud. In the cloud programming, an infra is created that consists of Kubernetes and database (dB) clusters. The A protocol is used to send data from all the devices to the cloud which is MQTT protocol that is based on publish or subscribe (pub/sub) model. All of the data received is stored in a database.

In the cloud, a MQTT broker is ready to receive the sending data of the devices to be filtered and published to the subscriber. Then, data acquisition system is used that implies set of functions to control the whole system based on the set rules and schedules. A web interface that extend the control of the system to admins and users and mobile application for the user to manage their own devices is also created. The database cluster is separated from the Kubernetes Cluster to avoid single point of failure. For future, the application can be easily updated without affecting the database.

The components of the Smart Urban Farming that involve in the replication mechanism of Kubernetes is proposed as in this research as shown in Figure3. In overall there are three elements involved which are Kubernetes dashboard that is used to setup and monitor Kubernetes, Master which is used as a scheduler, controller manager and API server, and Worker 1 and 2 which is the location of pods that contains containers.

The worker 1 and 2 has 1 vCPU and 2 GB memory each. The choice of Worker 1 and 2 is after considering the amount of data received from devices which is adequate to be catered. The specification of each worker is based on resources that is needed of the application that need to deploy. An alert policy is set up at the provider site to notify us when the data supplied is near to the threshold.

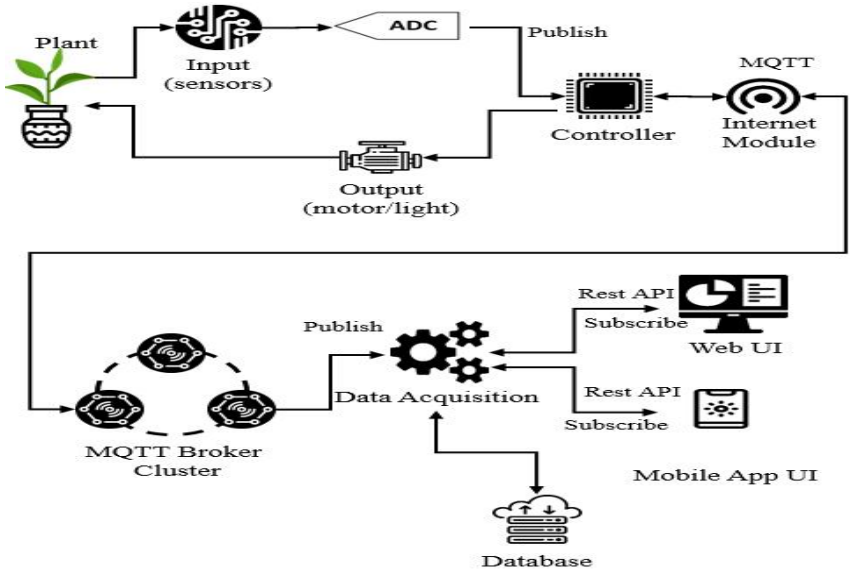

Figure 2: Smart Urban Farming Architecture

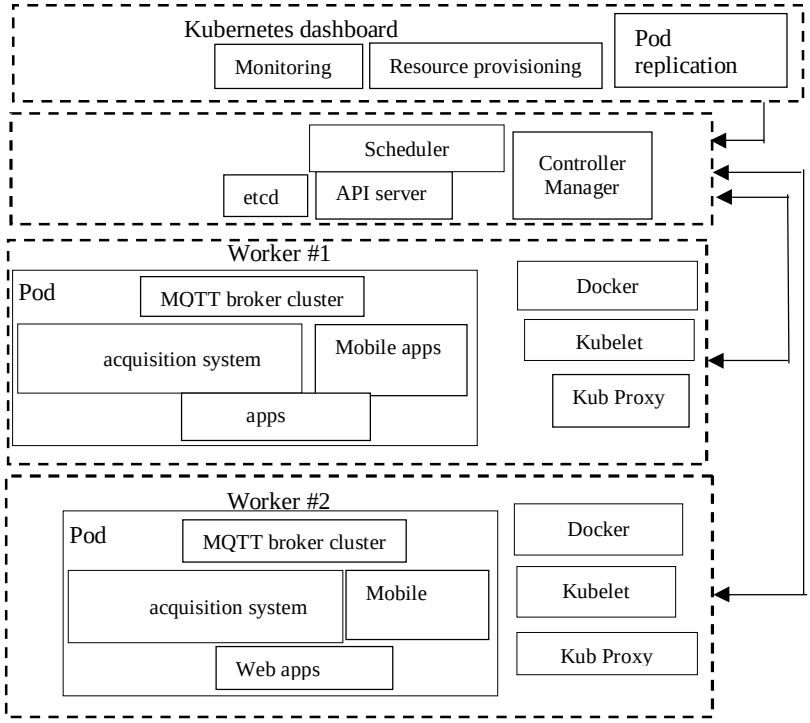

Figure 3: Proposed Replication Mechanism of Kubernetes for Smart Urban Farming

4. RESULTS AND DISCUSSION

This section presents the results gathered from the experiments as shown in Table 1.It shows that, Kubernetes will allocate new container for MQTT broker to ensure workload will be well-distributed. Once the allocation is done, the container will be instantiated by pulling image container for MQTT broker (Emqx). Finally, the emqx-container starts its function. This is known as pod instance in Kubernetes deployment. The configuration was done by using Horizontal Pod Autoscaler (HPA). One of the needs for pods replication configuration is "targetCPUUtilizationPercentage". For demonstration purpose, "targetCPUUtilization" value is set to 30%, 65%, 75%, and 90%. It is observable that

when there are too many data requests, when the CPU utilization increased to 90% it activates and triggers to create new pod automatically. Based on observation, Kubernetes is used to scale in/out depending on the volume of data request/transaction received. This is done based on new pods creation when a certain threshold is triggered (in our case CPU utilization). It means that when volume of data requests increase, the number of pods will increase as well to handle the load. This can be resource intensive because we require a bigger size of Kubernetes cluster initially. However, to control this aspect (for economic reason), we can make the pod instance to be more efficient in handling transactions, simply by using optimized (low level) language for our program.In our case, initially we use python but it is not able to handle huge load and requires more pods to be spun up. But we already switched to Golang which is a more optimized and high-performance language. Hence, each pod is able to handle a larger number of requests and overall number of pods required in the Kubernetes cluster can be significantly reduced. From development perspective, coding in python is much simpler and faster, however resource can be intensive (e.g. Kubernetes cluster size). Another option is to implement a high performance Golang language. Despite its complex language structure compared to Python and requires more control, the resources can be optimized which is an added benefit to this research and other IoT application development.

.

Table 1: CPU utilization experiment on new pod creation

CPU utilization	*New pod creation trigger*
30%	No
65%	No
75%	No
90%	Yes

5. CONCLUSION

This paper presents the self-containerization that can ensure the data availability and functioning system at the user side. The main objective is to evaluate the effectiveness of self-healing data management of Kubernetes in dealing with downtime error. It shows that, when the CPU utilization hit the CPU utilization that was set, a new pod is produced. In comparison, this self-healing data management is helpful to solve the downtime error automatically. Conventionally, the system needs human availability to overcome the situation.

ACKNOWLEDGMENTS

We would like to thank the Ministry of Higher Education of Malaysia and University of MARA technology (UiTM) Shah Alam for sponsoring this research under 600-IRMI/FRGS-RACER 5/3 (081/2019) grant.

REFERENCES

1. N.V. Fedoroff. ***Food in a future of 10 billion***. Agriculture & Food Security, 2015, 4(1), pp.1-10.
2. K. Benke,&B. Tomkins. ***Future food-production systems: vertical farming and controlled-environment agriculture,***Sustainability: Science, Practice and Policy, 2017, 13(1), 13-26.
3. S. Golden.***Urban Agriculture Impacts: Social, Health, and Economic: A Literature Review***,University of California, 2016.
4. A. Siegner, J. Sowerwine, &C. Acey. ***Does urban agriculture improve food security? Examining the nexus of food access and distribution of urbanproduced foods in the United States: A systematic review,***Sustainability, 2018, 10(9), 2988.
5. C. Kerang, H. Lee, and H. Jung. **Task Management System According to Changes in the Situation Based on IoT**, Journal of Information Processing System, Vol.13, No.6, pp.1459-1466, December 2017.
6. S. Muralidharan, G. Song, &H. Ko, H. ***Monitoring and managing iot applications in smart cities using kubernetes***, CLOUD COMPUTING, 2019, 11.
7. P. J. Armington. ***Systems and methods for migrating a server from one physical platform to a different physical platform,*** US7769720B2 A1, United States Patent and Trademark Office, 12 January 2006.
8. W. Vogels. ***Beyond server consolidation***, Queue, 2008, 6(1), 20-26.
9. R.P. Goldberg.***Survey of Virtual Machine Research,*** Computer, June 1974, pp. 34-45.
10. K.Lee, Y. Kim, & C. Yoo, C. ***The Impact of Container Virtualization on Network Performance of IoT Devices***, Mobile Information Systems, 2018.