IMPLEMENTATION OF GITOPS IN CONTAINERIZED INFRASTRUCTURE

1) Rayhan Gusty Alif, 2) Lulu Chaerani Munggaran
1,2) Graduate Program of Information System Management, Gunadarma University
1,2) Jl. Salemba No. 53 – Jakarta – Indonesia
E-mail: rayhanga@student.gunadarma.ac.id, lulu@staff.gunadarma.ac.id

ABSTRACT
IT Infrastructure is one of the core components of a business’ scalability and reliability, thus having an efficient way on managing the IT Infrastructure would be one of the core decisions for a business. IT Infrastructure itself has evolved throughout the years, with the rise of virtualization technology, containerization became more relevant as ever, one such example is Containerization Infrastructure, an IT Infrastructure that uses Containerization as its backbone. With the push of the technology, a new way of managing Containerization Infrastructure efficiently is needed. There are multiple researches regarding the implementation of GitOps already, but none of them explained the connection between GitOps and Containerized Infrastructure, this paper is intended to discuss the connection by implementing GitOps in a Containerized infrastructure. This resulted in a quite steep learning curve and preparation time, but in the end all the changes and deployment of the application would be done automatically, this resulted in maximized focus on the development of the application rather than reflecting the changes later on. Other than that, GitOps itself is not limited to Containerized Infrastructure, although since GitOps is designed with virtualization in mind, theoretically, the efficiency would be reduced if it’s implemented in other kind of IT Infrastructure.

Keyword: Containerized Infrastructure, GitOps, Implementation, IT Infrastructure Management.

INTRODUCTION
There are multiple researches regarding the implementation of GitOps already [1], [2]. Nevertheless, there are still no researches that explained the connection between Containerized Infrastructure and IT Infrastructure Management. This research is intended to discuss the connection between IT Infrastructure Management and the implementation of GitOps in a Containerized Infrastructure, this is done by implementing GitOps in a Containerized Infrastructure. IT Infrastructure itself is one of the core components of a business’ scalability and reliability [3]. With Virtual Infrastructures, a change has been brought in on how we manage a business's IT Infrastructure, these changes increase the efficiency in managing a deployed software/application, which led to a much better virtualization technology called Containerization. This Containerization method helps us make our application more portable, scalable, and isolated [4]. However, due to the rise of Containerization technology and the popularity of microservices architecture, managing Containerized applications increases the management complexity, therefore using traditional methods of infrastructure management would involve manual operations to scale and even to deploy it for the first time [5]. Using Traditional methods is more prone to human errors, time-consuming, and reduced scalability, not only that, but the rise of Cloud Computing also increases the need to reduce cost and full utilization of IT resources, and shorten software deployment time [6]. Introduce GitOps, a modern approach to managing IT Infrastructure, and a potential solution to address these challenges. GitOps combines the principle of Infrastructure as Code (IaC) and Continuous Delivery/Deployment (CD) practices, it uses Git repositories as the main source of truth for infrastructure configurations [7], [8], [9], [10], [11], [12]. This paper will be covering the connection between GitOps and IT Infrastructure Management to see whether it’s a viable IT Infrastructure Management method for a Containerized Infrastructure.

DOI: https://doi.org/10.36341/rabit.v9i1.3787 Some rights reserved BY-NC-SA 4.0 International License
METHOD

This research will explore the connection between GitOps and IT Infrastructure Management by implementing GitOps in a Kubernetes Cluster (Containerized Infrastructure), but will not cover all the available variations. By implementing GitOps using ArgoCD, Gitlab CI/CD Pipeline, and Rancher Desktop (K3s) on a local home lab with an internet connection to connect with Gitlab CI/CD repository.

To Setup a Kubernetes Cluster on a local home lab we can use Rancher Desktop to provision Kubernetes Cluster, which in this case it’s a K3s Cluster. To Setup CI/CD Pipeline, we can use Gitlab CI/CD, to do this we can create a YML file defining the build and deploy stages of the pipeline. The build stage will build all the docker images and update the Kubernetes manifests to use the recently built image. The deploy stage will send the docker images to Docker Hub (Image Registry) and commit the recently updated Kubernetes manifests. To Setup ArgoCD in the Kubernetes Cluster, we can apply the official ArgoCD Kubernetes Manifest directly into a new namespace called ArgoCD. Make sure to disable TLS since this will be done only in a local home lab which prioritizes giving ease of access without worrying too much about security, this can be done by applying a custom ConfigMap that will be discussed later on in the Implementation Chapter.

To Deploy an Example App, this research will use a slightly modified version of GoogleCloudPlatform/microservice-demo and ArgoCD to demonstrate the implementation, to deploy the example app using the ArgoCD, use the ArgoCD UI to register a new application in ArgoCD and then proceeds to sync up the manifests with the current deployment state from the Kubernetes.
RESULT

There is no formally defined implementation of GitOps in a Containerized Infrastructure, although based on previous researches and industry standards, we can generalize the existing design of GitOps implementation to be:


   [Figure 2. Rancher Desktop Main Page]

   Enabling the K3s Cluster inside the Rancher Desktop Settings as shown in Figure 2 and Figure 3.

   [Figure 3. Rancher Desktop Preferences Page]

   Pick the latest *stable* version of Kubernetes as shown in Figure 4.

   [Figure 4. Rancher Desktop Kubernetes Version Setting]

   Pick any port for the Kubernetes Port as shown in Figure 5.

   [Figure 5. Rancher Desktop Kubernetes Port Setting]

   [Figure 6. Rancher Desktop Kubernetes Traefik Setting]
Enable Traefik as the default network class for the Kubernetes Cluster as shown in Figure 6.

```
C:\Users\rayhan\kubernetes-demo\kubectl create namespace git-ops
namespace/git-ops created
```

**Figure 7. Creating Kubernetes Namespace called git-ops**

```
C:\Users\rayhan\kubernetes-demo\kubectl create namespace argocd
namespace/argocd created
```

**Figure 8. Creating Kubernetes Namespace called argocd**

Create new Kubernetes Namespaces as shown in Figure 7 and Figure 8.

2. Setup Gitlab CI/CD Pipeline

Define the stages for the CI/CD Pipeline as shown in Figure 9 which are build and push stages.

```
stages:
  - build
  - push
```

**Figure 9. Defined stages in .gitlab-ci.yml**

Define the build job template as shown in Figure 10 which will build said web app.

```
adservice: *build_template
cartservice: *build_template
collectionservice: *build_template
currencyservice: *build_template
generalservice: *build_template
loadgenerator: *build_template
paymentservice: *build_template
productcatalogservice: *build_template
recommendationservice: *build_template
shippingservice: *build_template
```

**Figure 10. Definition of build_template**

Define the push job template as shown in Figure 11.

```
1 adservice: *push_template
cartservice: *push_template
collectionservice: *push_template
currencyservice: *push_template
generalservice: *push_template
loadgenerator: *push_template
paymentservice: *push_template
productcatalogservice: *push_template
recommendationservice: *push_template
shippingservice: *push_template
```

**Figure 11. Definition of push_template**

Define each service build job as shown in Figure 12.

```
1 adservice: *build_template
2 cartservice: *build_template
3 collectionservice: *build_template
4 currencyservice: *build_template
5 generalservice: *build_template
6 loadgenerator: *build_template
7 paymentservice: *build_template
8 productcatalogservice: *build_template
9 recommendationservice: *build_template
10 shippingservice: *build_template
```

**Figure 12. Defined jobs that uses build_template**
13. Define each service push job as shown in Figure 13.

3. Setup ArgoCD

Apply the **Official ArgoCD Kubernetes Manifest** to the `argocd` namespace as shown in Figure 14.

Figure 14. Applying ArgoCD Manifest

Apply the custom ConfigMap (as shown in Figure 15) to `argocd` namespace and rollout-restart the ArgoCD deployment as shown in Figure 16.

Figure 15. ArgoCD ConfigMap Manifest

Figure 16. Applying ArgoCD ConfigMap and do a rollout-restart
Figure 17. ArgoCD Ingress Manifest

Apply the Ingress Manifest (as shown in Figure 17) to enable access through web browser to ArgoCD UI as shown in Figure 18.

Figure 18. Applying ArgoCD Ingress

Access the ArgoCD UI as shown in Figure 20 and Figure 21.

4. Deploy Example App

Create new App in ArgoCD UI as shown in Figure 22.

Figure 19. Getting the initial admin password for ArgoCD UI

Get the initial admin password using powershell command as shown in Figure 19.

Figure 20. ArgoCD UI Login Page

Fill in the GENERAL section as shown in Figure 23.
Each services of the application can be seen by click the application from the ArgoCD UI Homepage, the list of services will be shown as in Figure 27.

Create the application, and the created application will be listed in the ArgoCD UI Homepage as shown in Figure 26.

Once the APP HEALTH is Healthy as shown in Figure 28, proceed to open the deployed web app by accessing store.gitops.localhost as shown in Figure 29.

**CONCLUSION**

GitOps Implementation in a Containerized Infrastructure requires a lot of preparation in advance, but this preparation would pay off later on when changes are made to the application since all the changes would be automatically updated and all the needed IT Infrastructure would be provisioned.
automatically inside the Containerized Infrastructure.

GitOps is not limited to Containerized Infrastructure and can be implemented on any kind of IT infrastructure, but it would decrease the efficiency if it’s being implemented on other than virtualized infrastructure since the GitOps method is designed with virtualization in mind, and since containerization itself is the successor of virtualization technology GitOps works perfectly with Containerized Infrastructure.

The main weakness that we can see in this approach is that it requires a quite steep learning curve and heavy initial setup, other than that there are no other weaknesses that could be given from GitOps, although to prevent the heavy setup in the future we can use some automation script either by writing our own or use some provisioning tools like Terraform or Ansible.

BIBLIOGRAPHY


