

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Pandemi covid-19 yang telah berlangsung selama lebih dari 2 tahun belakangan ini telah mengakselerasi kebutuhan pembelajaran secara non-tatap muka. Pembelajaran non-tatap muka dapat dilakukan secara daring melalui *platform learning management system*, salah satu yang populer dan bersifat *open source* adalah Moodle [1]. *Public cloud* menjadi salah satu wadah yang cocok untuk melayani aplikasi moodle karena sifatnya yang fleksibel, efisien, aman dan hemat biaya. Dengan layanan *public cloud*, *server* dapat dengan mudah untuk diskalakan secara otomatis sesuai kebutuhan dan membayar sesuai *resource* yang digunakan [2]. Google Cloud Platform (GCP) dan Microsoft Azure merupakan dua dari tiga *public cloud* yang paling banyak digunakan berdasarkan *market share* pada Q1 2021 [3]. Dengan semakin merambahnya layanan *cloud* maka organisasi mendorong untuk beralih dari penggunaan *virtual private server* menuju *container* karena dapat meningkatkan efisiensi *workload* yang berjalan di *cloud* [4]. Untuk mengoptimalkan *container*, dibutuhkan *tool* untuk melakukan orkestrasi *container*. Salah satu *tool* orkestrasi *container* yang *open source* dan didukung langsung oleh Google adalah kubernetes [5].

Arsitektur kubernetes menerapkan konsep *master node* dan *worker node*. Dengan arsitektur tersebut, apabila salah satu *pod* berhenti berjalan pada *worker node*, *master node* akan secara otomatis memerintahkan *worker node* untuk membuat *pod* replika baru sebagai pengganti *pod* yang berhenti berjalan. Dengan konsep tersebut, arsitektur kubernetes cocok untuk diterapkan pada *learning management system* seperti moodle.

Public Cloud memberi kemudahan dalam membangun arsitektur kubernetes. Kemudahan tersebut membuat suatu *cluster* kubernetes dapat dibuat secara mudah dan kapan saja bisa diberhentikan. Terlepas dari kemudahan membangun suatu *cluster* kubernetes di-*public cloud*, ada kalanya kita perlu melakukan migrasi suatu *cluster* kubernetes dari suatu *public cloud* menuju ke *public cloud* lainnya. Salah

satu tujuan dari migrasi tersebut yaitu untuk memitigasi terjadinya *disaster* pada suatu *region* di *public cloud*. Misalnya ketika *public cloud* seperti Google Cloud mengalami *disaster* terhadap salah satu *region* mereka, maka aplikasi moodle yang ditempatkan di-*region* tersebut tidak dapat diakses [6]. Aplikasi tersebut bisa saja di-*route* ke *region* yang berbeda, namun hal tersebut akan menyebabkan *latency* yang tinggi. Permasalahan tersebut dapat diatasi dengan memigrasi *cluster* kubernetes pada Google Cloud menuju ke *region* yang sama di Microsoft Azure sebagai *public cloud* alternatif. Salah satu cara untuk melakukan migrasi dari Google Cloud menuju ke Microsoft Azure adalah dengan menggunakan metode *backup* dan *restore* [7]. Namun, terdapat tantangan untuk melakukan *backup* suatu *cluster* kubernetes yang memiliki *database stateful* seperti moodle. Database harus diletakan diluar *cluster* untuk tetap menjaga agar data tidak ikut hilang apabila suatu *pod* moodle pada *cluster* mengalami masalah [8] .

Velero merupakan salah satu *tool* yang dapat digunakan untuk melakukan *backup* dan *restore* pada *cluster* Kubernetes, namun dibutuhkan *plugin restic* untuk dapat melakukan *backup* dan *restore volume* diluar *cluster* [9]. Pada penelitian sebelumnya [10], velero digunakan untuk *backup* dan *restore* pada *cluster* kubernetes, namun hanya *object* didalam *cluster* yang dapat di-*backup* dan *restore*, tidak dengan *volume* yang terdapat diluar *cluster* dikarenakan penelitian tersebut tidak menggunakan *plugin restic*. Kemudian penelitian tersebut tidak melakukan skema migrasi antar *public cloud* melainkan antar *on-premise* ke *public cloud*.

Oleh Karena itu, dalam tugas akhir ini penulis melakukan penelitian mengenai implementasi dan analisis migrasi kubernetes *cluster* beserta *persistent volume* dari Google Cloud menuju ke Microsoft Azure menggunakan mekanisme *backup* dan *restore*. Skema *backup* dan *restore* menggunakan velero dengan *plugin restic* sebagai *tool* bantu untuk *backup cluster* kubernetes beserta *persistent volume* diluar cluster kedalam Google Cloud Storage Bucket lalu memigrasi *object backup* tersebut ke Azure Blob Storage sehingga *cluster* kubernetes pada Azure dapat melakukan *restore cluster* yang sudah dicadangkan oleh GCP beserta dengan *persistent volume*.

1.2 Rumusan Masalah

Berdasarkan uraian pada latar belakang penelitian, penulis mengidentifikasi beberapa masalah penelitian sebagai berikut:

1. Bagaimana merancang suatu migrasi LMS beserta *persistent volume database* pada *cluster* kubernetes antar *public cloud provider* menggunakan mekanisme *backup and restore* yang berfungsi untuk memitigasi terjadinya *disaster*?
2. Bagaimana cara melakukan mekanisme *backup and restore* kubernetes *cluster* beserta *persistent volume database* pada suatu kubernetes cluster antara Google Cloud dan Microsoft Azure?
3. Bagaimana performansi, waktu *backup, restore, migration* dan *cpu usage* pada mekanisme *backup and restore* di kedua *cluster* Kubernetes tersebut?

1.3 Tujuan dan Manfaat

Sesuai permasalahan yang diajukan, maka tujuan dan manfaat dari penelitian ini adalah:

1. Merancang migrasi LMS pada Kubernetes *cluster* beserta *persistent volume data* antar *public cloud provider* menggunakan mekanisme *backup and restore* dalam upaya memitigasi terjadinya *disaster* pada suatu *public cloud provider*.
2. Mengimplementasikan *tool* velero dengan *plugin* restic untuk melakukan *backup and restore* terhadap suatu *cluster* beserta *persistent volume data* kubernetes antar Google Cloud dan Microsoft Azure.
3. Mengidentifikasi nilai performansi, *backup, restore, migration time*, dan *cpu usage* saat melakukan *backup and restore*.

1.4 Batasan Masalah

Adapun batasan masalah dalam penelitian ini adalah sebagai berikut:

1. Kubernetes digunakan sebagai *tool* orkestrasi *container* yang bersifat *open source*.
2. Terdapat dua *public cloud platform* yang digunakan, yaitu Google Cloud Platform dan Microsoft Azure.

3. Mekanisme *backup* dan *restore* database pada Kubernetes *cluster* antar *public cloud* menggunakan Velero-restic sebagai *tool third party*.
4. Layanan yang digunakan berupa LMS dengan menggunakan Moodle dan MariaDB sebagai *database*.
5. Kubernetes *cluster* disetiap *public cloud* tidak menggunakan *horizontal pod autoscaler*.

1.5 Metode Penelitian

Metode penelitian yang digunakan untuk mencapai hasil dalam proses penyelesaian tugas akhir adalah sebagai berikut:

1. Studi literatur
Mencari jurnal, *paper*, buku dan dokumentasi terkait dengan penelitian yang akan dilakukan oleh penulis.
2. Analisa masalah
Melakukan analisa terhadap masalah yang terkait dengan penelitian, serta berdiskusi dengan dosen pembimbing.
3. Perancangan sistem
Melakukan perancangan terkait dengan penelitian sebelum melakukan implementasi, pengujian dan analisis.
4. Implementasi sistem
Melakukan implementasi rancangan terkait dengan penelitian, selanjutnya melakukan pengujian dan analisis.
5. Pengujian dan analisis
Pengujian terhadap rancangan sistem yang dibuat serta menganalisis sistem tersebut berdasarkan parameter yang telah direncanakan.
6. Penyusunan Tugas Akhir
Menyusun dan mencatat laporan secara sistematis terhadap hasil penelitian untuk dijadikan sebagai laporan tugas akhir.