



Kubernetes

Adaptive Training 3.0 - Day 1

Presented by:



Rafli Hadiana

Research Assistant at Adaptive Network Laboratory

Day 1 - Course Outline



- Kubernetes Introduction
- Kubernetes Main Component
- Kubernetes Architecture
- Kubectl Main Command
- Struktur Manifest File
- Struktur Kubeconfig File
- Kubernetes Namespace
- Kubernetes Services
- Kubernetes Ingress

Apa itu Cloud Native ?

Cloud Native adalah concept of building and running applications to take advantage of the distributed computing offered by the cloud delivery model.

Cloud Native apps are designed and built to exploit the scale, elasticity, resiliency, and flexibility the cloud provides.

Cloud Native menerapkan 5 prinsip teknologi berikut ini untuk bisa mengimplementasikan hal tersebut.

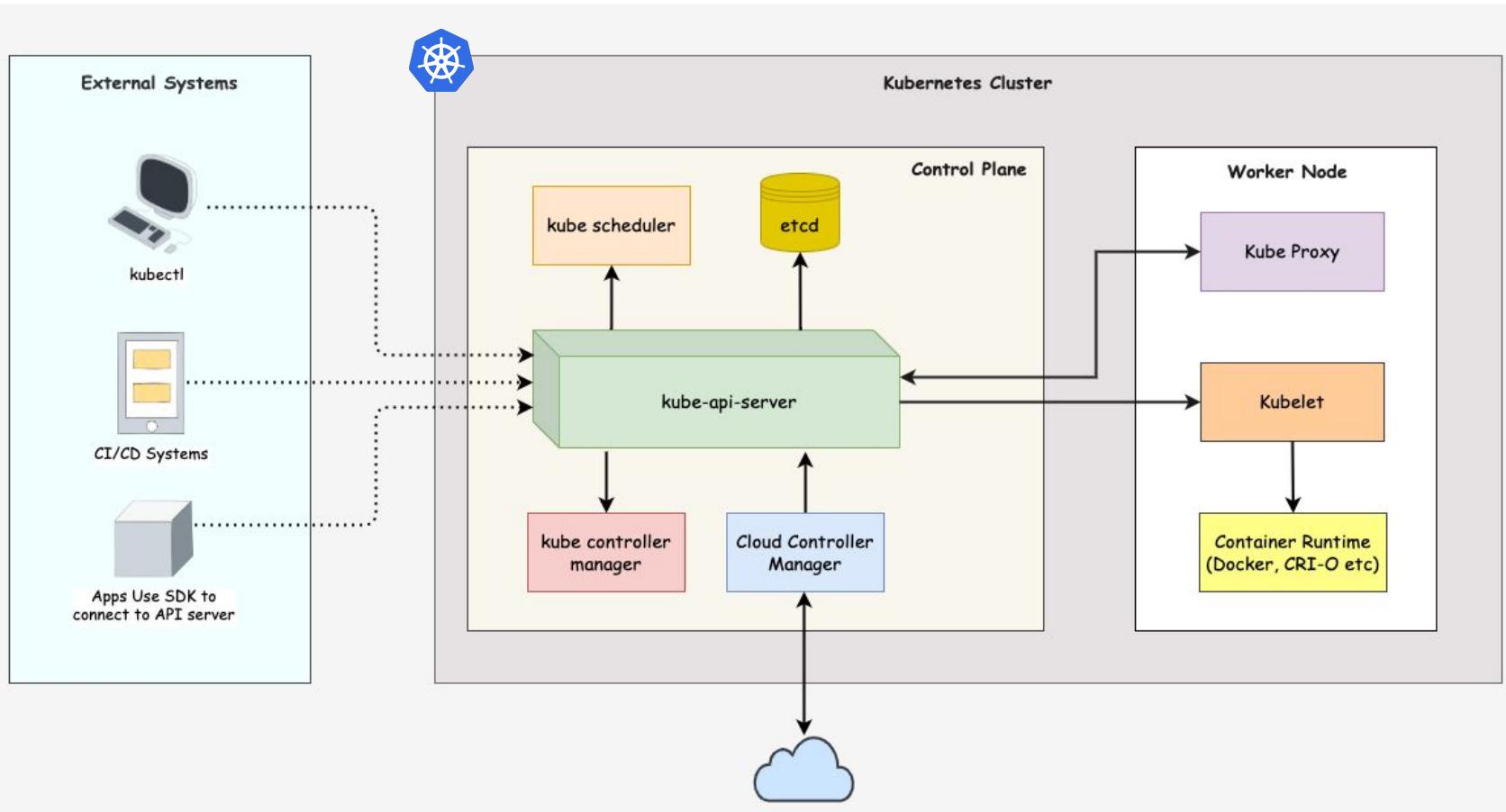


Kubernetes Introduction



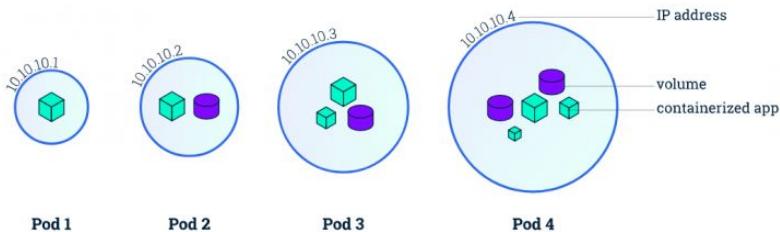
Kubernetes merupakan sebuah software container orchestration yang berguna untuk mengelola aplikasi berbasis container. Selain itu **Kubernetes** bersifat **High Availability, Scalability** dan **Disaster Recovery** sehingga aplikasi kita dapat diakses user dengan performa yang sangat baik.

Kubernetes Architecture



Kubernetes Object

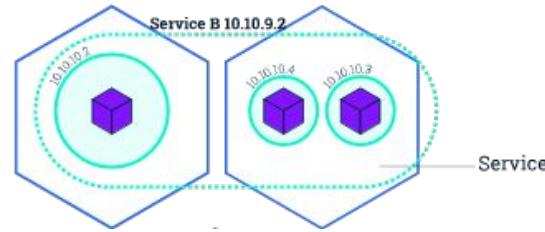
- **Pod**



Pod merupakan unit terkecil pada komponen Kubernetes.

Setiap **pod** memiliki IP address masing-masing dan satu pod berisi satu aplikasi yang biasanya terdiri dari satu container atau lebih.

- **Service**

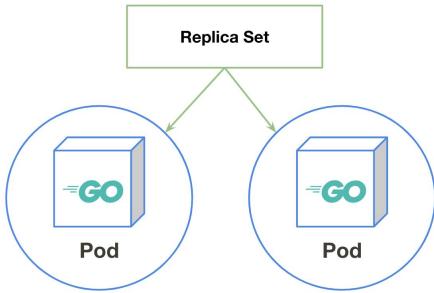


Service adalah permanent IP address yang bisa diletakan di setiap pod.

Object ini digunakan agar user bisa mengakses pod baik secara internal ataupun eksternal.

Kubernetes Object

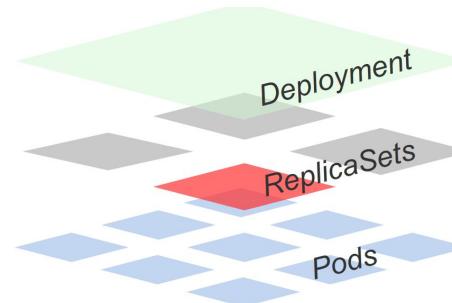
- **Replica Set**



Pod merupakan unit terkecil pada komponen Kubernetes.

Setiap **pod** memiliki IP address masing-masing dan satu pod berisi satu aplikasi yang biasanya terdiri dari satu container atau lebih.

- **Deployment**

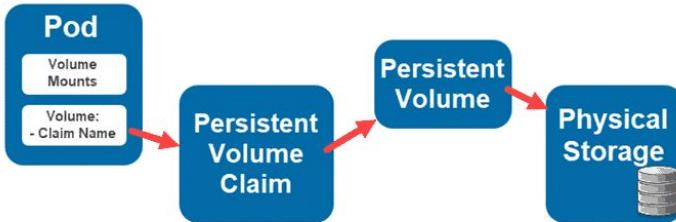


Service adalah permanent IP address yang bisa diletakan di setiap pod.

Object ini digunakan agar user bisa mengakses pod baik secara internal ataupun eksternal.

Kubernetes Object

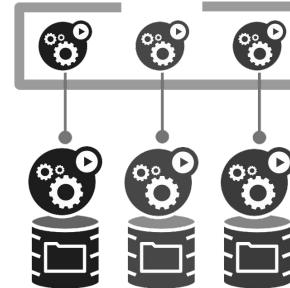
- **Volumes**



Pod merupakan unit terkecil pada komponen Kubernetes.

Setiap **pod** memiliki IP address masing-masing dan satu pod berisi satu aplikasi yang biasanya terdiri dari satu container atau lebih.

- **Statefulset**

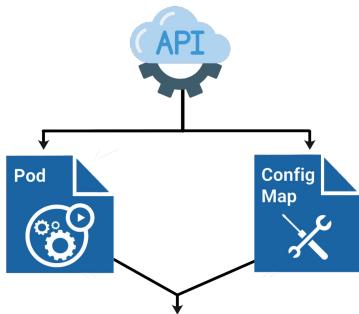


Service adalah permanent IP address yang bisa diletakan di setiap pod.

Object ini digunakan agar user bisa mengakses pod baik secara internal ataupun eksternal.

Kubernetes Object

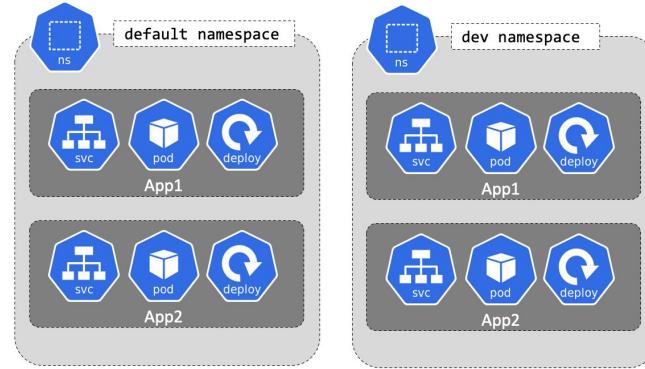
- **Configmap**



Configmap merupakan objek yang berisi konfigurasi eksternal variabel atau data dari aplikasi.

Setiap **pod** memiliki IP address masing-masing dan satu pod berisi satu aplikasi yang biasanya terdiri dari satu container atau lebih.

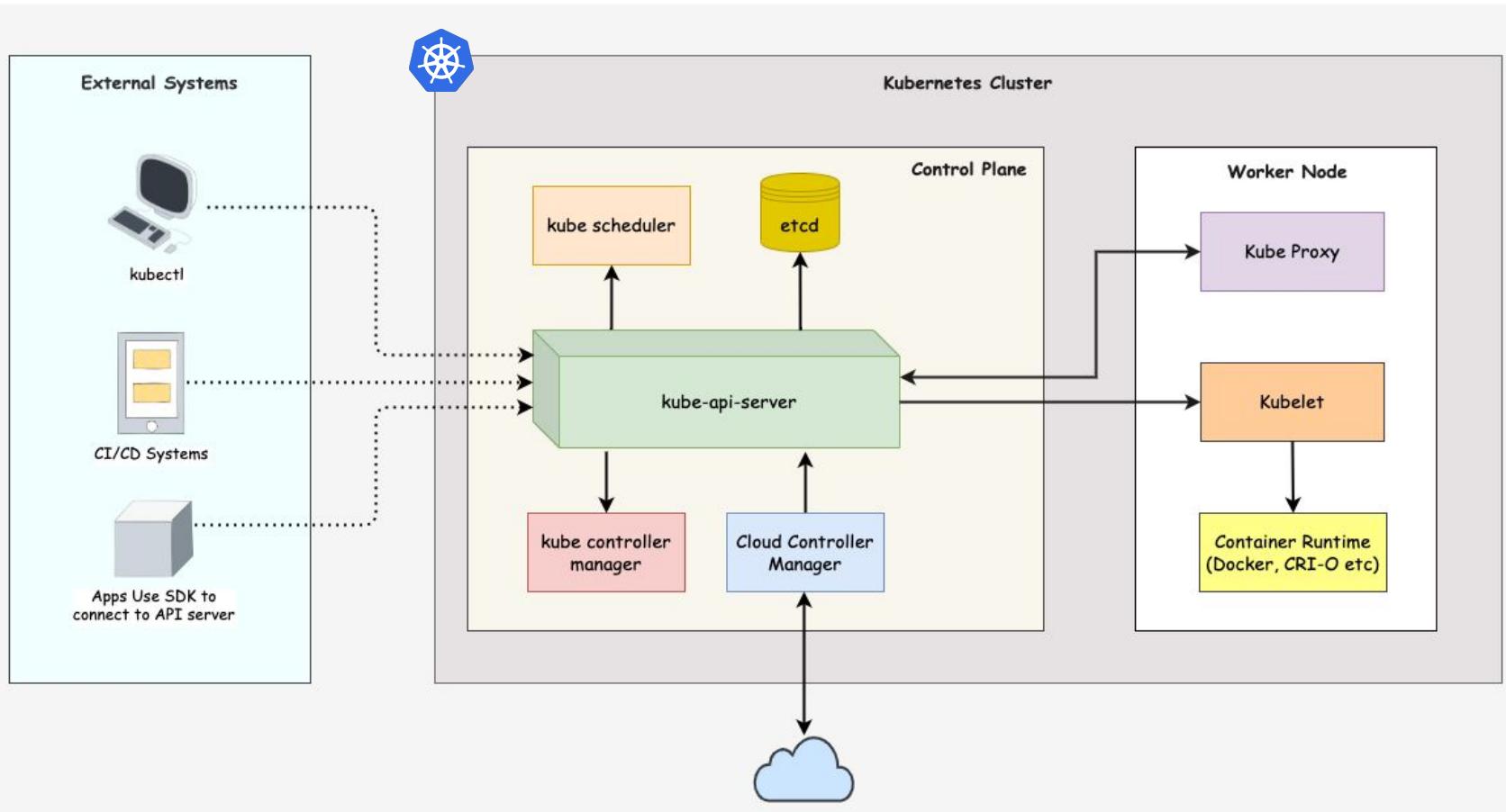
- **Namespace**



Service adalah permanent IP address yang bisa diletakan di setiap pod.

Object ini digunakan agar user bisa mengakses pod baik secara internal ataupun eksternal.

Kubernetes Architecture



Kubernetes Architecture

Komponen Master Node

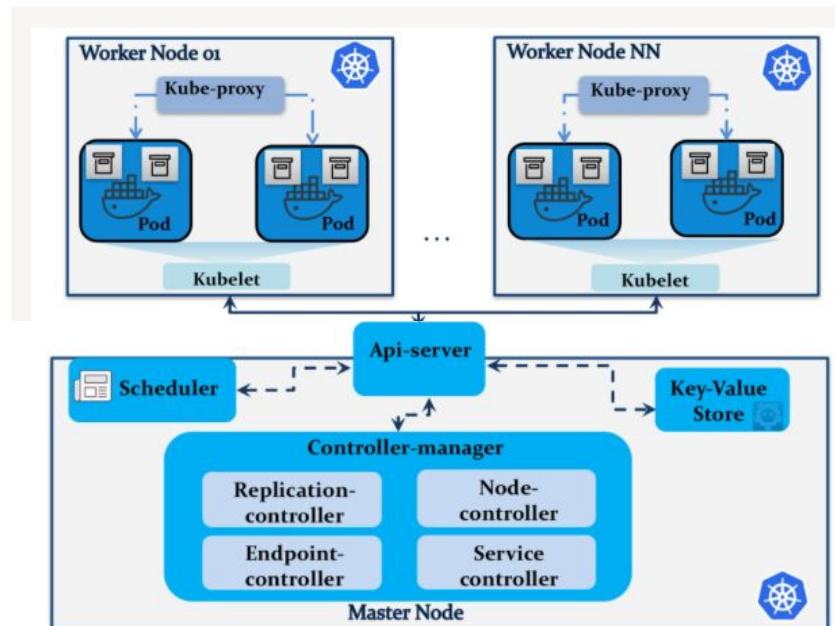
Master node adalah node yang mengelola semua workload dari cluster. Berikut beberapa komponen di dalamnya:

1. kube-apiserver
2. etcd
3. kube-scheduler
4. kube-controller-manager
5. cloud-controller-manager

Komponen Worker Node

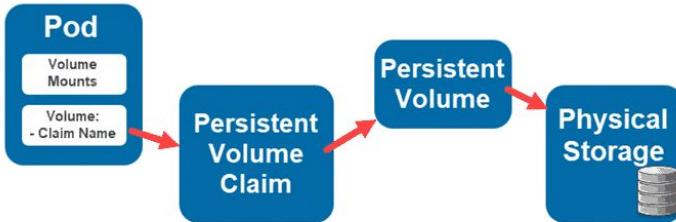
Worker node adalah node yang menjalankan aplikasi kita sebagai container. Berikut beberapa komponen di dalamnya:

1. kubelet
2. kube-proxy
3. Container runtime



Komponen Master Node

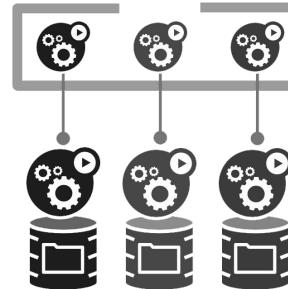
- Kube-API Server



Pod merupakan unit terkecil pada komponen Kubernetes.

Setiap **pod** memiliki IP address masing-masing dan satu pod berisi satu aplikasi yang biasanya terdiri dari satu container atau lebih.

- ETCD

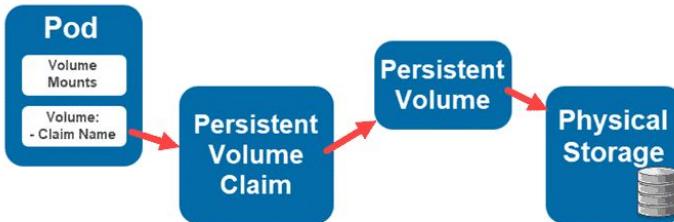


Service adalah permanent IP address yang bisa diletakan di setiap pod.

Object ini digunakan agar user bisa mengakses pod baik secara internal ataupun eksternal.

Komponen Master Node

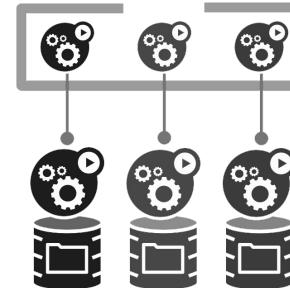
- **Kube-Controller Manager**



Pod merupakan unit terkecil pada komponen Kubernetes.

Setiap **pod** memiliki IP address masing-masing dan satu pod berisi satu aplikasi yang biasanya terdiri dari satu container atau lebih.

- **Kube-Scheduler**

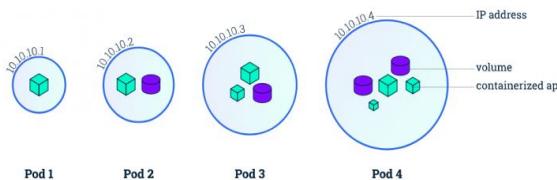


Service adalah permanent IP address yang bisa diletakan di setiap pod.

Object ini digunakan agar user bisa mengakses pod baik secara internal ataupun eksternal.

Komponen Worker Node

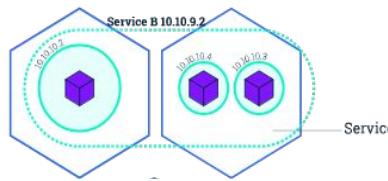
- Kubelet



Pod merupakan unit terkecil pada komponen Kubernetes.

Setiap **pod** memiliki IP address masing-masing dan satu pod berisi satu aplikasi yang biasanya terdiri dari satu container atau lebih.

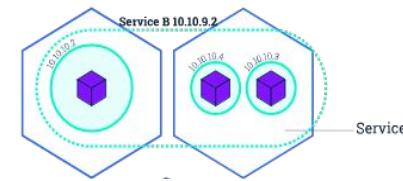
- Kube-Proxy



Service adalah permanent IP address yang bisa diletakan di setiap pod.

Object ini digunakan agar user bisa mengakses pod baik secara internal ataupun eksternal.

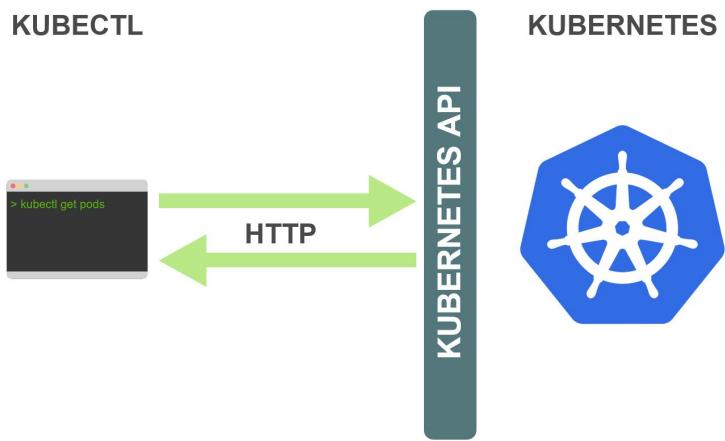
- Container Runtime



Service adalah permanent IP address yang bisa diletakan di setiap pod.

Object ini digunakan agar user bisa mengakses pod baik secara internal ataupun eksternal.

Kubectl



Apa itu Kubectl ?

Command line tool kubernetes yang digunakan user untuk menjalankan berbagai perintah untuk mengkonfigurasi cluster.

Kubectl Command:

- **Create** deployment (`kubectl create deployment [nama]`)
- **Edit** deployment (`kubectl edit deployment [nama]`)
- **Delete** deployment (`kubectl delete deployment [nama]`)
- **Run** deployment
- **Get** deployment
- **Apply** deployment

Di dalam k8s version 1.19+, kita bisa melakukan spesifikasi konfigurasi –replicas untuk membuat deployment dengan misal 4 replicas.

```
kubectl create deployment --image=nginx nginx  
--replicas=4 --dry-run=client -o yaml >  
nginx-deployment.yaml
```

Kubeconfig

```
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data: <ca-data-here>
    server: https://your-k8s-cluster.com
    name: <cluster-name>
contexts:
- context:
    cluster: <cluster-name>
    user: <cluster-name-user>
    name: <cluster-name>
current-context: <cluster-name>
kind: Config
preferences: {}
users:
- name: <cluster-name-user>
  user:
    client-certificate-data:
<secret-certificate>
```

File **Kubeconfig** adalah file yang berisi tentang semua detail Kubernetes Cluster sebagai password atau otentikasi masuk ke dalam cluster.

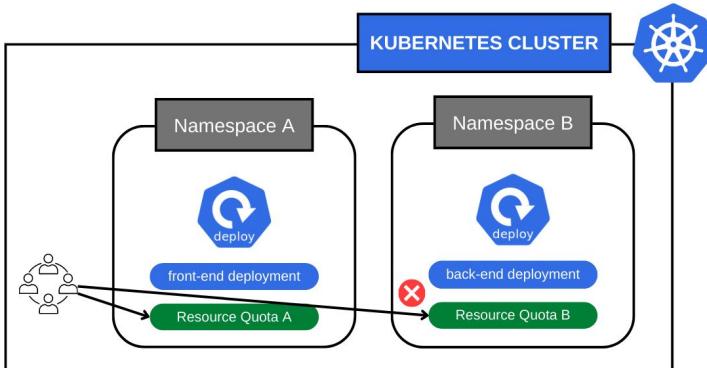
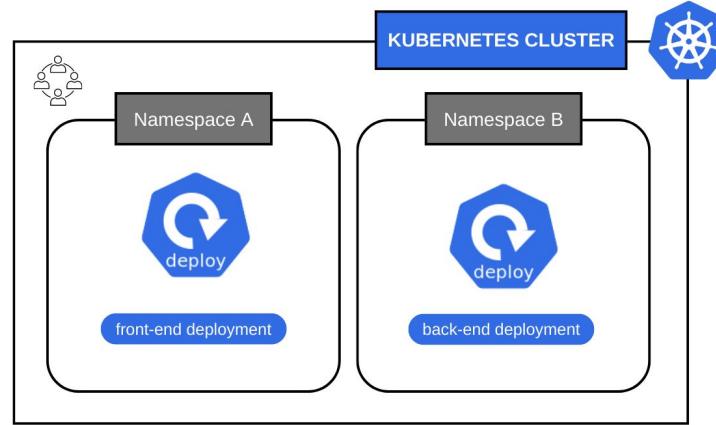
Minimal kita membutuhkan lima informasi penting untuk bisa melakukan koneksi, yaitu:

1. **certificate-authority-data:** Cluster CA.
2. **server:** Endpoint Cluster (IP/DNS master node).
3. **name:** Nama cluster.
4. **user:** Nama user.
5. **client-certificate-data:** Client certificate dari user.

Kubernetes Namespace

Namespace disebut sebagai **virtual cluster** di dalam cluster utama. Secara default Kubernetes memiliki empat namespace diantaranya yaitu:

1. Namespace **kube-system**
2. Namespace **kube-public**
3. Namespace **kube-node-release**
4. Namespace **default**

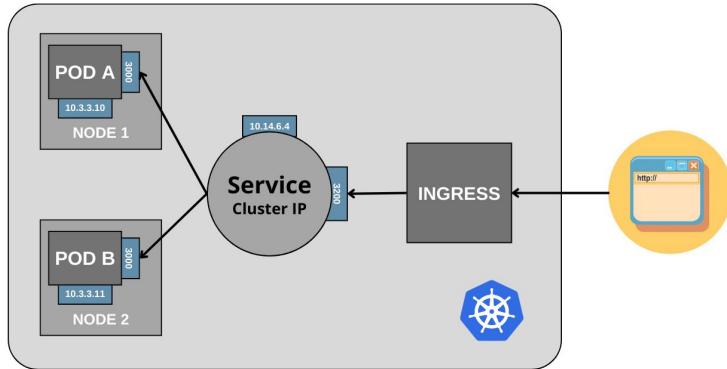


Berikut terdapat beberapa **manfaat use case** yang bisa diterapkan melalui penggunaan objek namespace:

1. Mudah untuk dikelola
2. Menghindari konflik antar tim
3. Melakukan resource sharing
4. Membatasi akses dan resource namespace

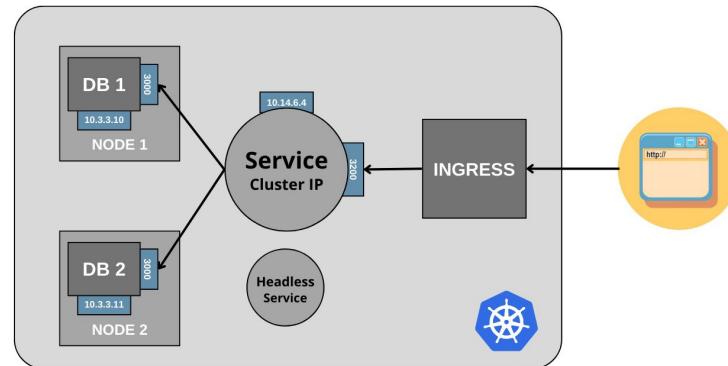
Kubernetes Service

- Cluster IP



Setiap **pod** memiliki IP address masing-masing dan satu pod berisi satu aplikasi yang biasanya terdiri dari satu container atau lebih.

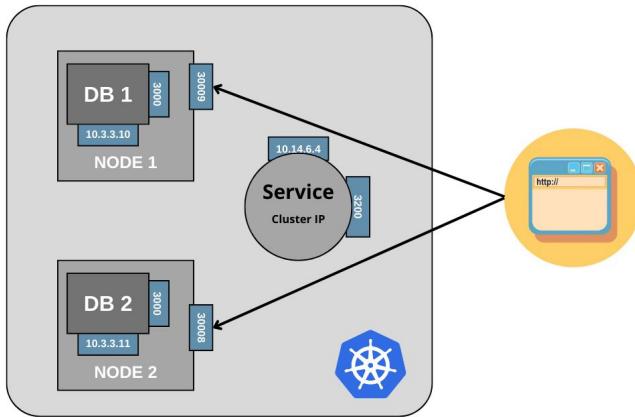
- Headless



Setiap **pod** memiliki IP address masing-masing dan satu pod berisi satu aplikasi yang biasanya terdiri dari satu container atau lebih.

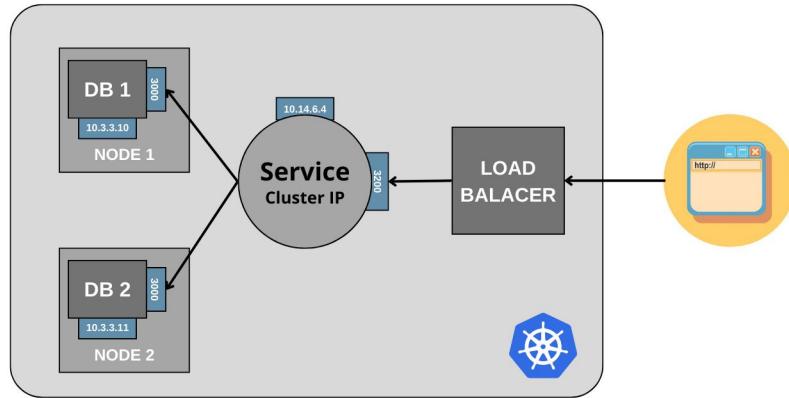
Kubernetes Service

- Node Port



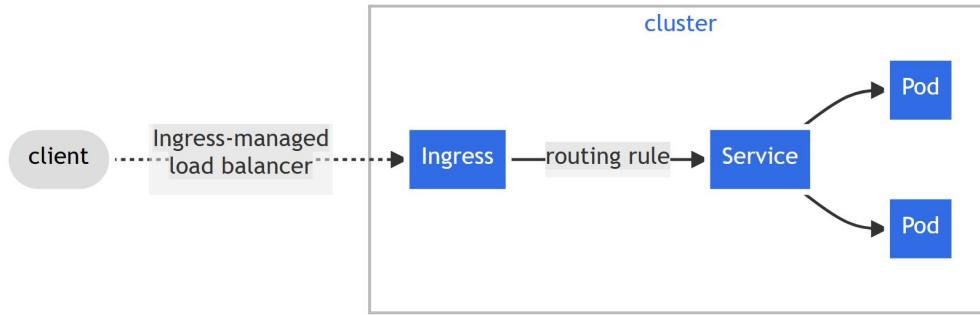
Setiap **pod** memiliki IP address masing-masing dan satu pod berisi satu aplikasi yang biasanya terdiri dari satu container atau lebih.

- Load Balancer



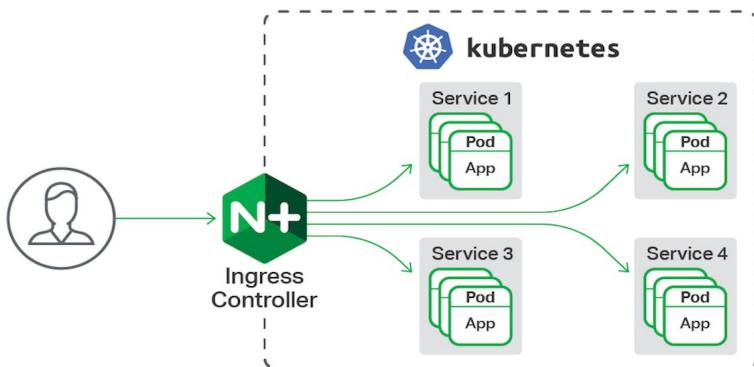
Setiap **pod** memiliki IP address masing-masing dan satu pod berisi satu aplikasi yang biasanya terdiri dari satu container atau lebih.

Kubernetes Ingress



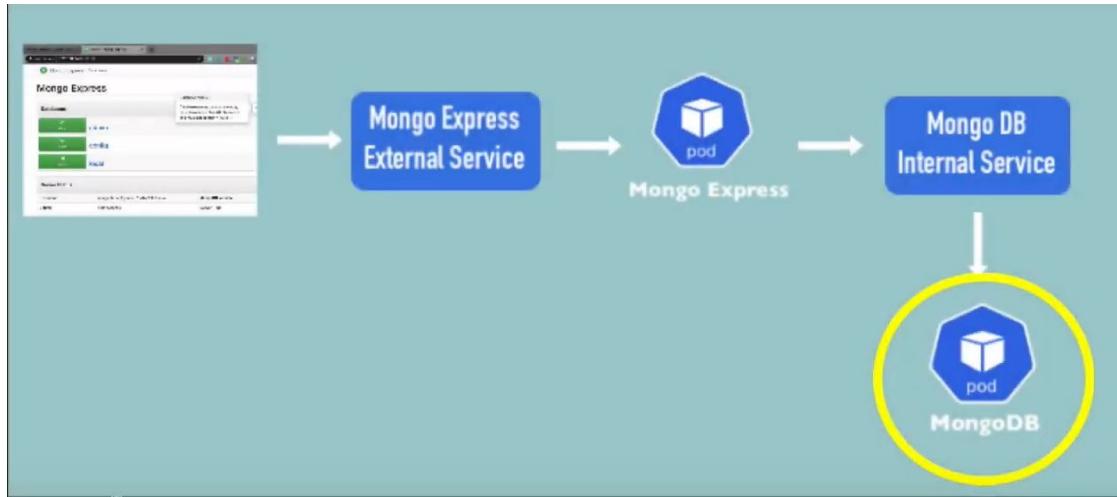
Dengan menggunakan object **ingress**, request datang dari browser ditangani oleh ingress dan di forward ke internal service.

Ingress Controller



Ingress controller berbentuk pod dan mengevaluasi juga memproses dari ingress routing rule, memanage redirection dan menjadi entry point ke dalam cluster, karena nanti rules ingress akan sangat banyak (misalkan 50 rules).

Hands-on 1 | Database Application



<https://github.com/raflihadiana/kube-class/tree/main/database-app>



Kubernetes

Adaptive Training 3.0 - Day 2

Presented by:



Rafli Hadiana

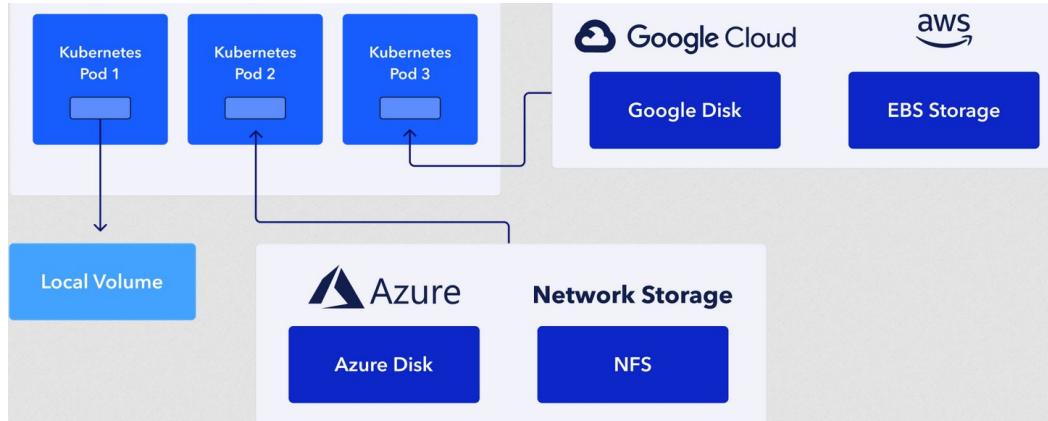
Research Assistant at Adaptive Network Laboratory

Day 2 - Course Outline



- Kubernetes Volume
- ConfigMap & Secret
- Kubernetes Statefulset
- RBAC & Service Account
- CR & CRD
- Helm Chart

Kubernetes Volume



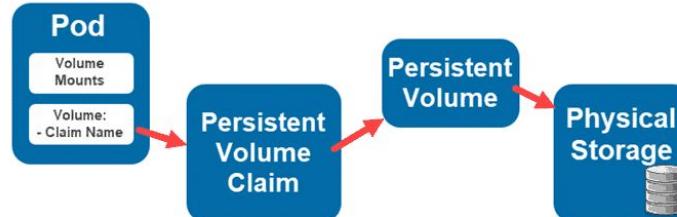
Volume sangat dibutuhkan untuk menyimpan data aplikasi pada database.

Persyaratan storage:

1. Tidak tergantung dengan pod lifecycle.
2. storage harus bisa tersedia oleh semua node.
3. Storage harus bisa survive ketika cluster crash.

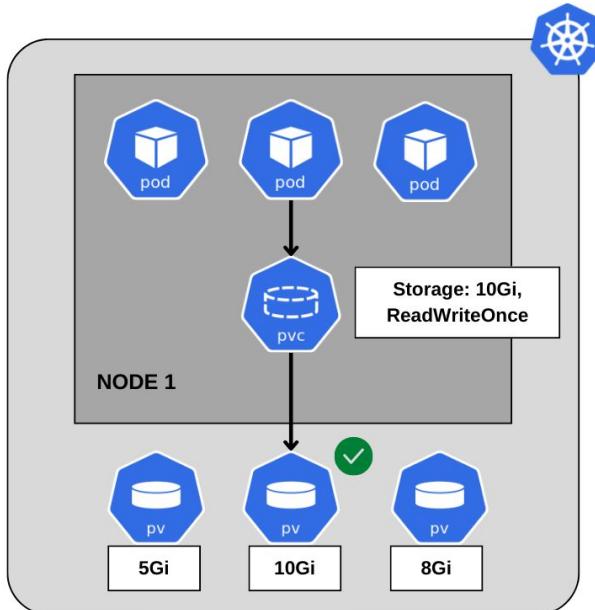
Di dalam kubernetes terdapat 3 Object yang dipakai sebagai Volume:

1. Persistent Volume (PV)
2. Persistent Volume Claim (PVC)
3. Storage Class (SC)



Kubernetes Volume

Persistent Volume (PV)



Storage Class (SC)

STORAGE CLASS CONFIG

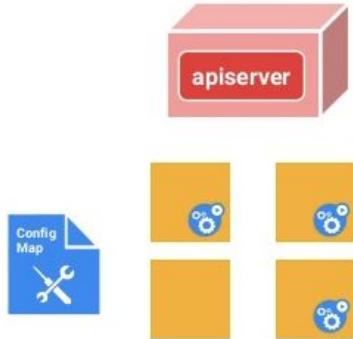
```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: my-storage-class
provisioner: kubernetes.io/glusterfs
parameters:
  resturl: "http://192.168.10.100:8080"
  restuser: ""
  secretNamespace: ""
  secretName: ""
allowVolumeExpansion: true
```

PVC Config

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: nama-pvc
spec:
  storageClassName: my-storage-class
  dataSource:
    name: existing-src-pvc-name
    kind: PersistentVolumeClaim
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi
```

Configmap & Secret

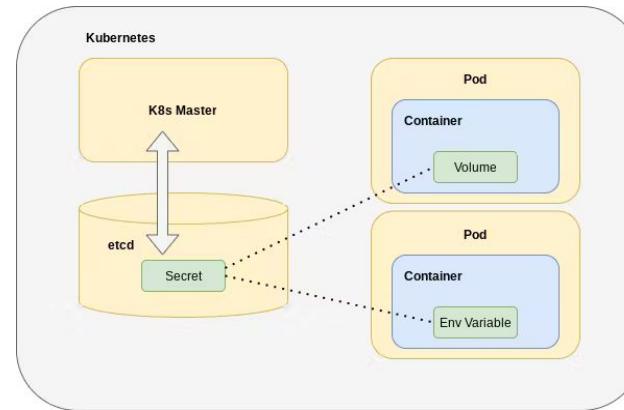
Configmap



ConfigMap adalah sebuah mekanisme untuk memisahkan konfigurasi dari sebuah aplikasi yang sudah di kontainerisasi.

ConfigMap cocok digunakan untuk menyimpan konfigurasi yang tidak sensitif atau konfigurasi yang tidak perlu dienkripsi, seperti ENV.

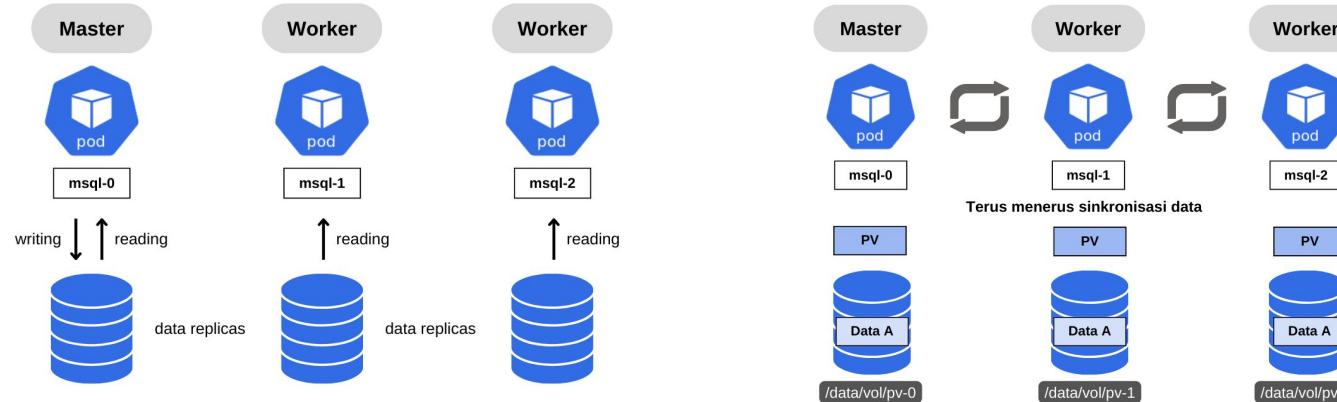
Secret



Secret berperan sebagai key atau cara otentikasi dengan protected computing resources dalam environment aplikasi.

Secret berada di dalam etcd, yang penyimpanan data center untuk cluster Kubernetes.

Kubernetes StatefulSet

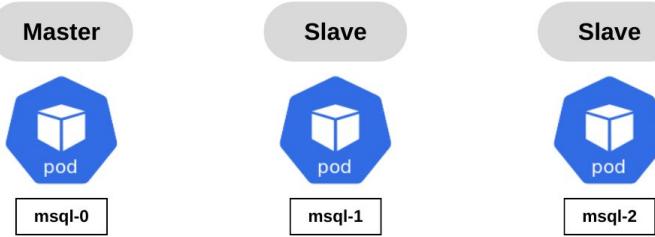


Komponen ini digunakan untuk mendeploy aplikasi stateful. setiap pod memiliki data storage (PV) nya masing-masing, dan di backup oleh physical storage nya, termasuk sinkronisasi data dan pod state.

Pod state berisi informasi tentang masterpod atau slavepod, jadi misal ketika pod mati maka Persistent Pod Identifier akan memastikan storage volume (PV) terhubung kembali dengan Pod Baru.

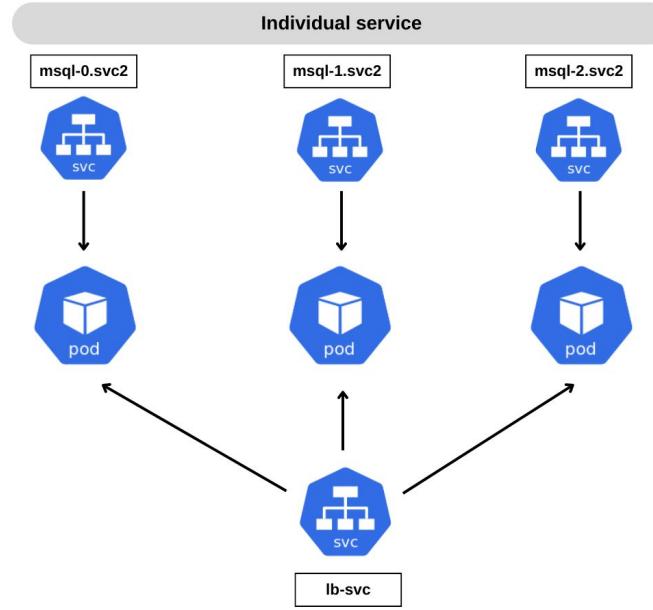
Kubernetes StatefulSet

Pod Identity



Setiap pod statefulset memiliki masing-masing identifier, dengan fixed ordered name bukan dengan hash. Pod selanjutnya akan berjalan ketika pod sebelumnya berjalan.

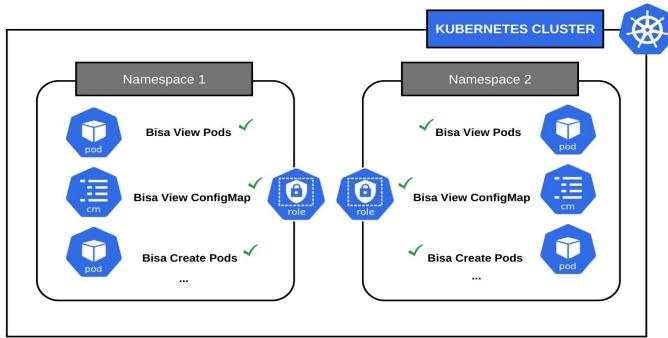
2 Pod Endpoint



Pod statefulset mempunyai karakteristik sticky identity, ketika pod mati maka hanya nama dan endpoint statik, sedangkan IP address berganti.

RBAC

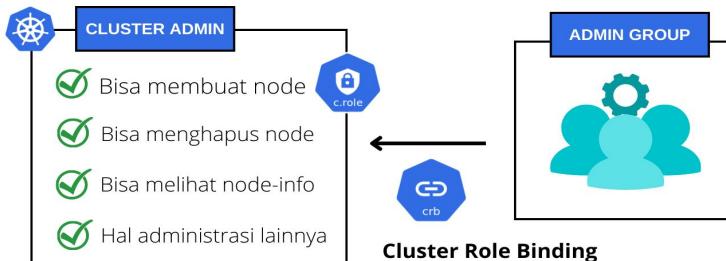
Role dan Role Binding



Role hanya konfigurasi resource dan access permission, tidak termasuk siapa yang dapat permission ini

Role Binding, yang memberikan akses Role kepada masing-masing atau sekelompok user. Jadi kita bisa mengetahui siapa-siapa saja yang dapat mengakses resource ini.

Cluster Role dan Cluster Role Binding

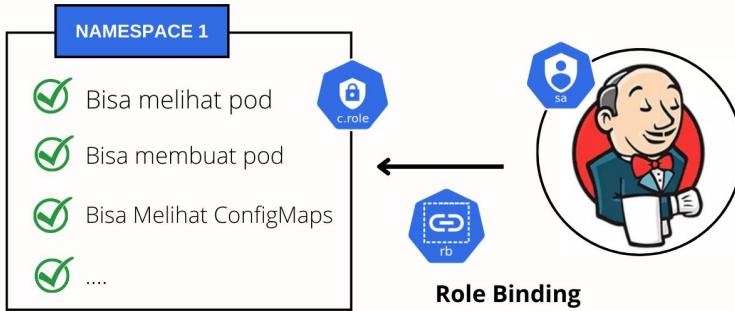


Cluster Role untuk mengkonfigurasi Cluster permission oleh admin tertentu pada Multi-Cluster Kubernetes.

Cluster Role Binding yang sama fungsinya dengan Role Binding untuk bisa mengetahui admin-admin mana saja yang diberikan akses pada cluster role tertentu.

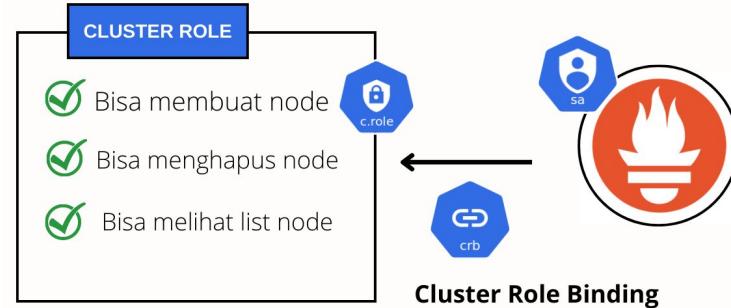
RBAC

Service Account



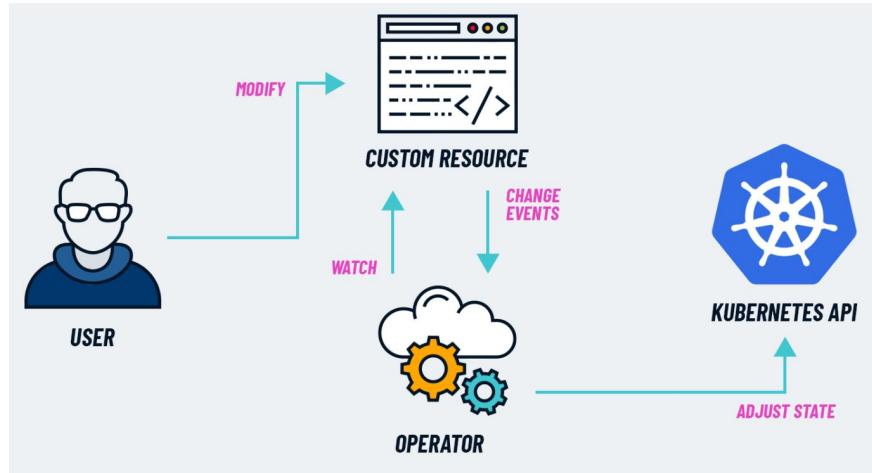
ServiceAccount melekat pada aplikasi eksternal yang mempunyai kebutuhan mengakses namespace di cluster, akan dihubungkan ke Role namespace dengan komponen RoleBinding.

Sedangkan aplikasi eksternal yang mempunyai kebutuhan mengakses cluster secara keseluruhan akan diberikan **ServiceAccount** dan akan dihubungkan ke ClusterRole dengan komponen **ClusterRoleBinding**.



CR & CRD

Custom Resource

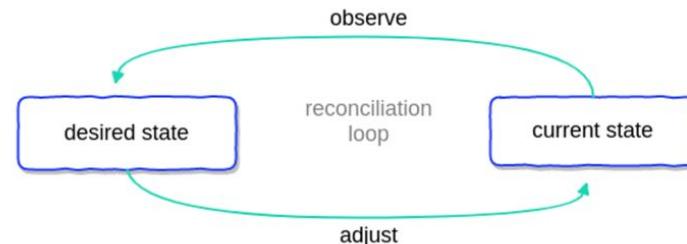


Kubernetes API dapat diextend untuk memanage aplikasi kompleks menggunakan custom resources (CR).

Di dalam CR kita bisa memodifikasi (Add, Update, Delete) struktur data sesuai dengan state aplikasi kita. Sebagai contoh kita bisa membuat objek kubernetes baru sesuai dengan kebutuhan custom user.

Custom Resource Definition

Cara menambahkan Custom Resource Salah satunya menggunakan Custom Resource Definitions (CRD)



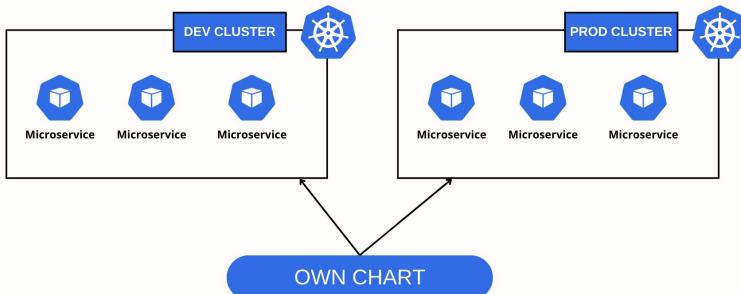
Kubernetes Helm

Helm Package Manager



Helm adalah package manager untuk Kubernetes, mengemas kumpulan YAML file dan di distribusikan ke publik atau private repository.

Helm Chart



values.yaml

```
name: my-app
container:
  name: my-app-container
  image: my-app-image
  port: 9001
```

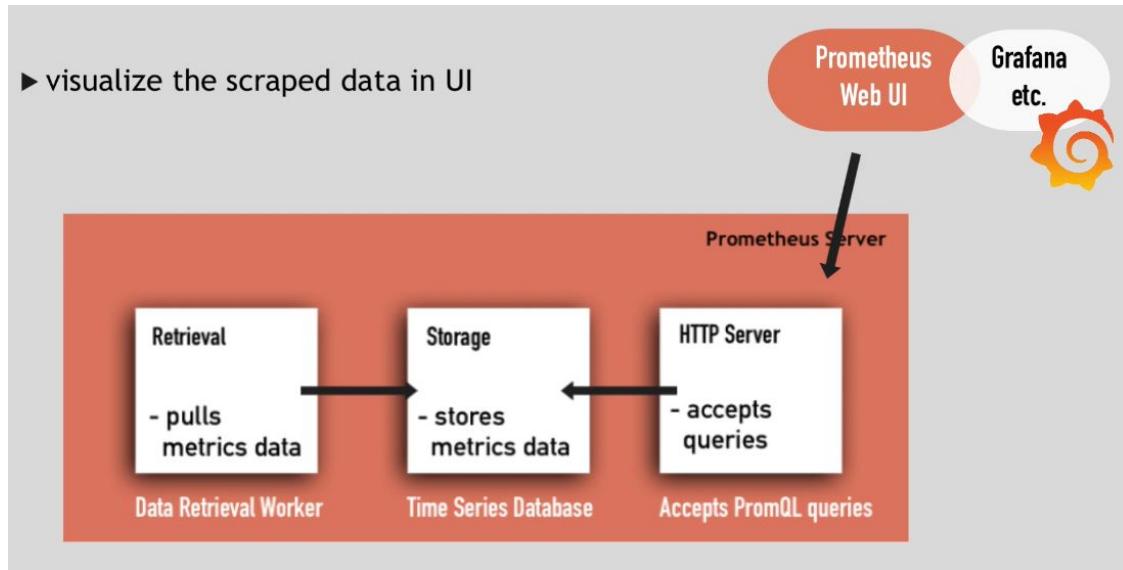
Template YAML Config

```
apiVersion: v1
kind: Pod
metadata:
  name: {{ .Values.name }}
spec:
  containers:
    - name: {{ .Values.container.name }}
      image: {{ .Values.container.image }}
      port: {{ .Values.container.port }}
```

Menggunakan packaging manager yang dinamakan Charts, helm chart dapat berisi sejumlah objek kubernetes, yang semuanya di-deploy sebagai bagian dari chart tersebut.

Hands-on 2 | Monitoring Application

- ▶ visualize the scraped data in UI



<https://github.com/raflihadiana/kube-class/tree/main/monitoring-app>



Kubernetes

Adaptive Training 3.0 - Day 3

Presented by:



Rafli Hadiana

Research Assistant at Adaptive Network Laboratory

Day 3 - Course Outline



- Kubernetes in Azure
- Terraform (IaC)
- Kubernetes Microservice

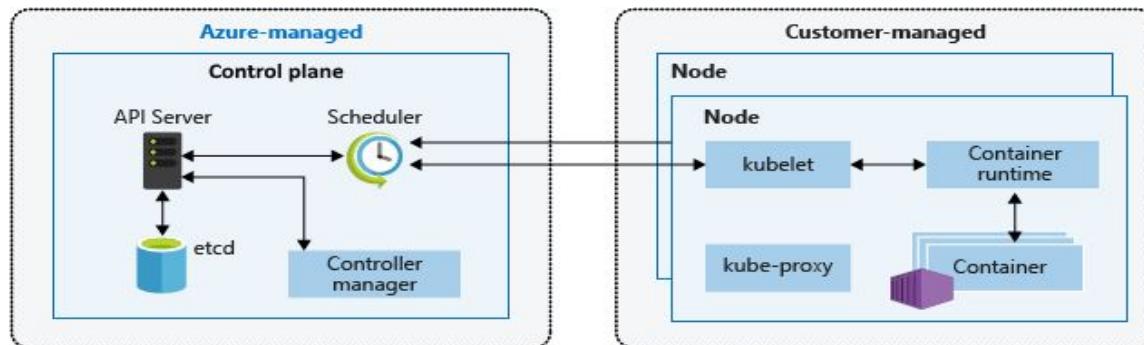
Kubernetes in Azure



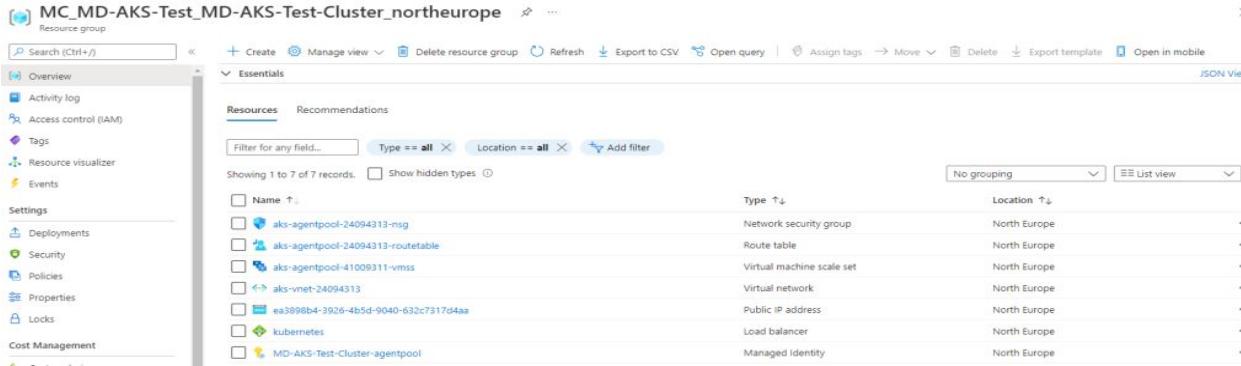
Managed Kubernetes Service di Public Cloud Platform. Terdapat beberapa banyak opsi seperti salah satunya adalah AKS.

Jadi user tidak perlu untuk membuat cluster dari awal / on-premis dan hampir semuanya diatur secara otomatis oleh Cloud platform.

Kubernetes di dalam AKS, node master nya secara otomatis akan dibuat dan dikonfigurasi. Jadi user hanya membayar untuk node pool worker kluster AKS.



Kubernetes in Azure



The screenshot shows the Azure portal interface for a resource group named 'MC_MD-AKS-Test_MD-AKS-Test-Cluster_northeurope'. The left sidebar contains navigation links for Overview, Activity log, Access control (IAM), Tags, Resource visualizer, Events, Settings, Deployments, Security, Policies, Properties, Locks, and Cost Management. The main content area displays a list of resources under the 'Resources' tab. The table includes columns for Name, Type, and Location. The resources listed are:

Name	Type	Location
aks-agentpool-24094313-nsg	Network security group	North Europe
aks-agentpool-24094313-routetable	Route table	North Europe
aks-agentpool-41009311-vms	Virtual machine scale set	North Europe
aks-vnet-24094313	Virtual network	North Europe
ea3898b4-3926-4b5d-9040-632c7317d4aa	Public IP address	North Europe
kubernetes	Load balancer	North Europe
MD-AKS-Test-Cluster-agentpool	Managed identity	North Europe

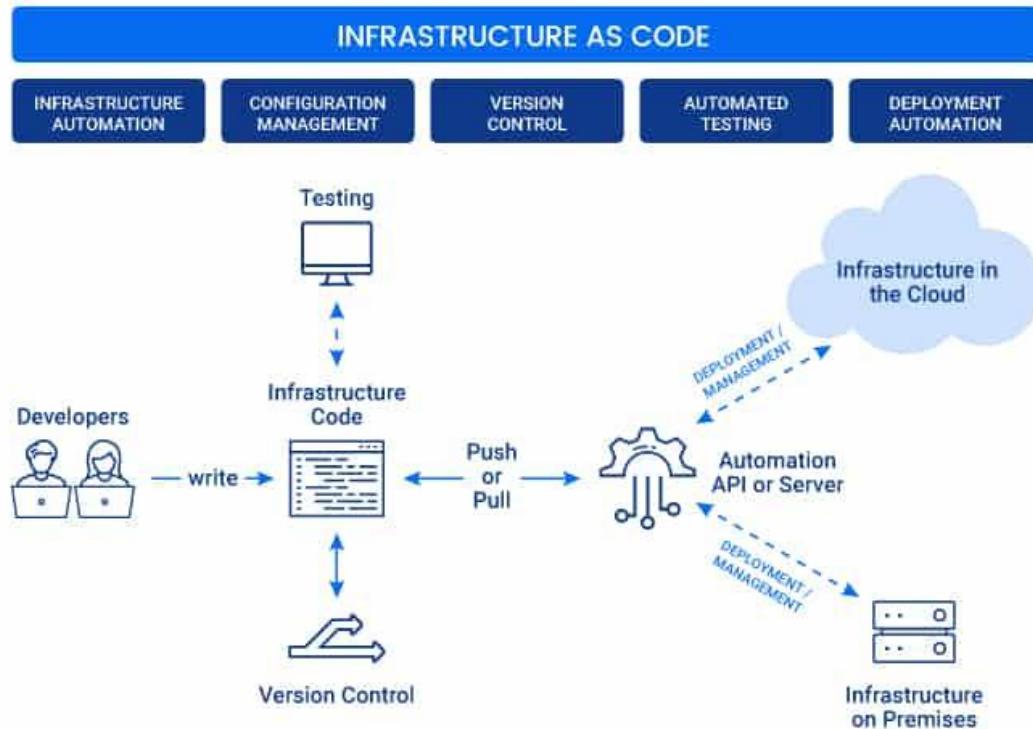
Dalam satu setup cluster terdapat NSG, Route Table, VNet, Load Balancer, Managed Identity dan Scale Set, jadi komponen underlying management components terpisah dari resource main cluster.

Mengakses Cluster



```
durkanm@Azure:~$ kubectl get nodes
NAME                               STATUS   ROLES      AGE     VERSION
aks-agentpool-41009311-vmss000000  Ready    agent      12m    v1.22.6
aks-agentpool-41009311-vmss000001  Ready    agent      12m    v1.22.6
durkanm@Azure:~$ 
```

Infrastructure as Code (IaC)

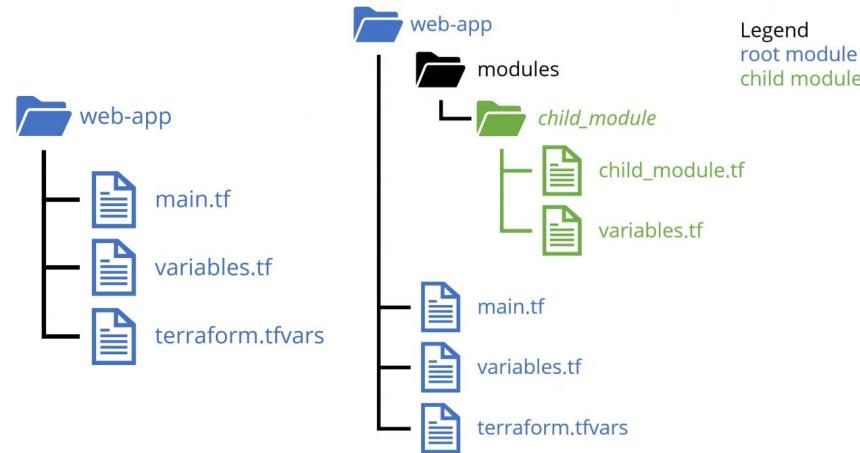


It is the process of managing and provisioning the complete IT infrastructure (comprising both physical and virtual machines) using machine-readable definition files. It helps in automating the complete data center by using programming scripts.

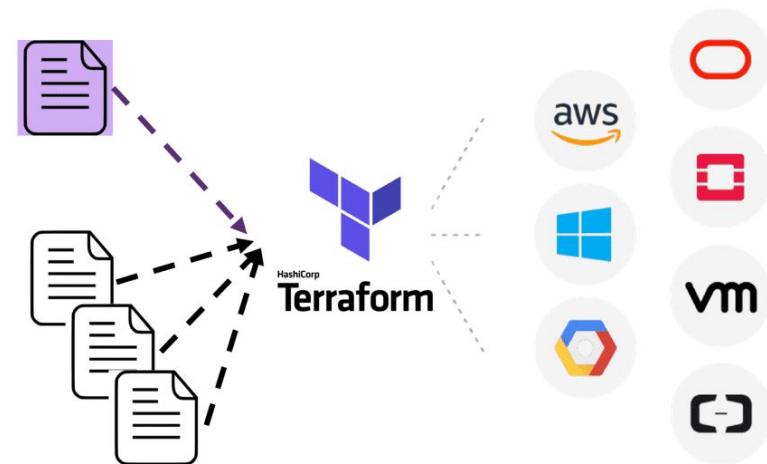
Terraform



Terraform Configuration Files

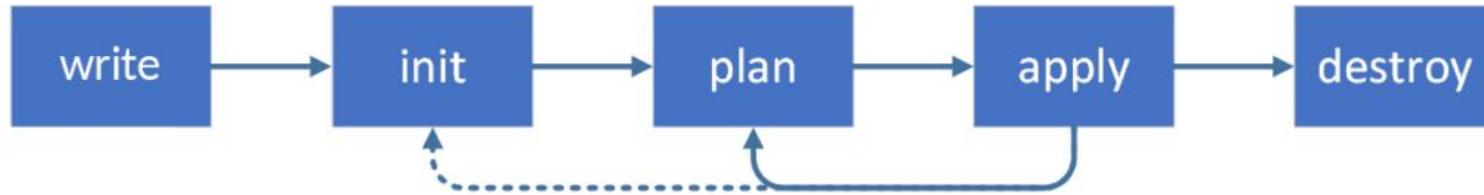


Terraform is one of the most popular **Infrastructure-as-code (IaC)** tool, used by DevOps teams to automate infrastructure tasks. It is used to automate the provisioning of your cloud resources.



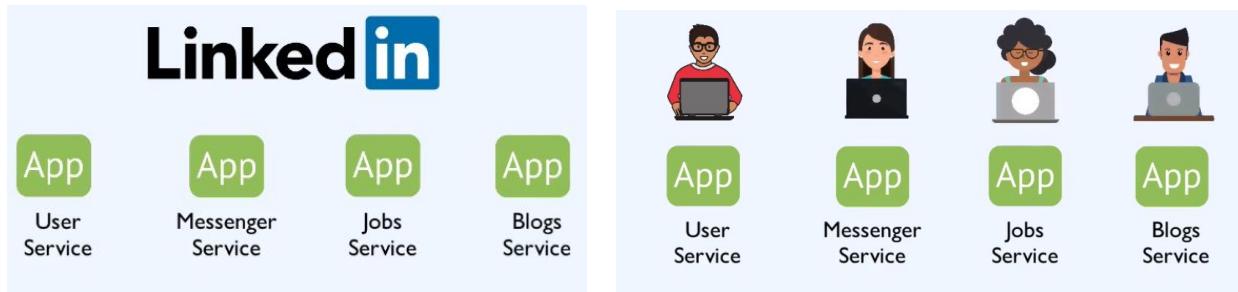
Terraform

Terraform Lifecycle



1. **Terraform init** initializes the (local) Terraform environment. Usually executed only once per session.
2. **Terraform plan** compares the Terraform state with the as-is state in the cloud, build and display an execution plan. This does not change the deployment (read-only).
3. **Terraform apply** executes the plan. This potentially changes the deployment.
4. **Terraform destroy** deletes all resources that are governed by this specific terraform environment.

Microservice in Kubernetes

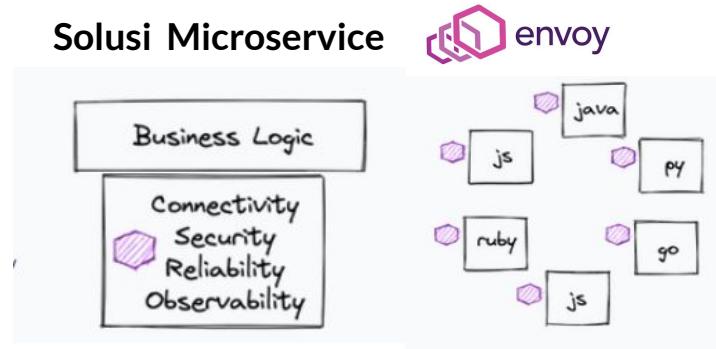


Microservice adalah memecah aplikasi monolith ke dalam service yang lebih kecil. Misal untuk aplikasi besar media sosial terbagi menjadi user service, message service, network service, blog service, dll.

Challenge Microservice

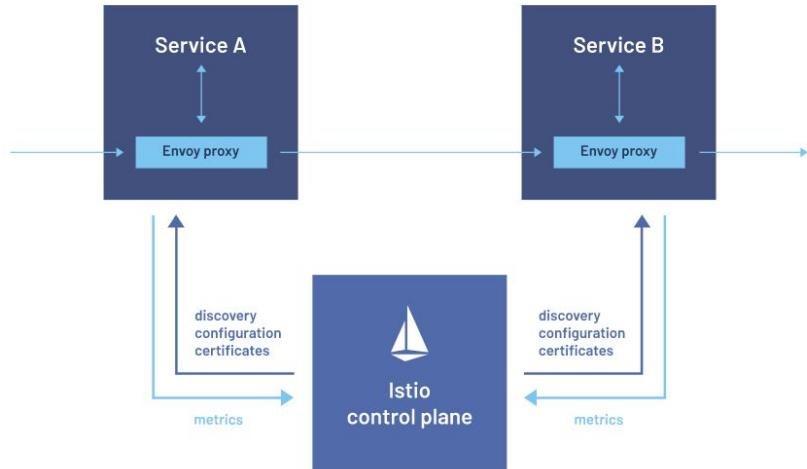


Solusi Microservice



Microservice in Kubernetes

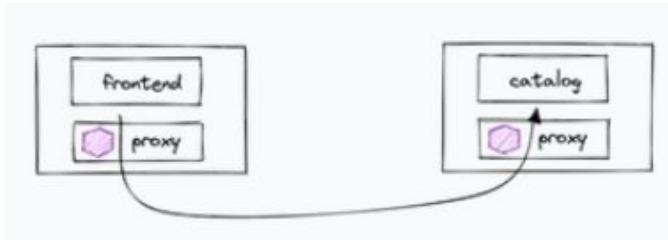
Istio Service Mesh



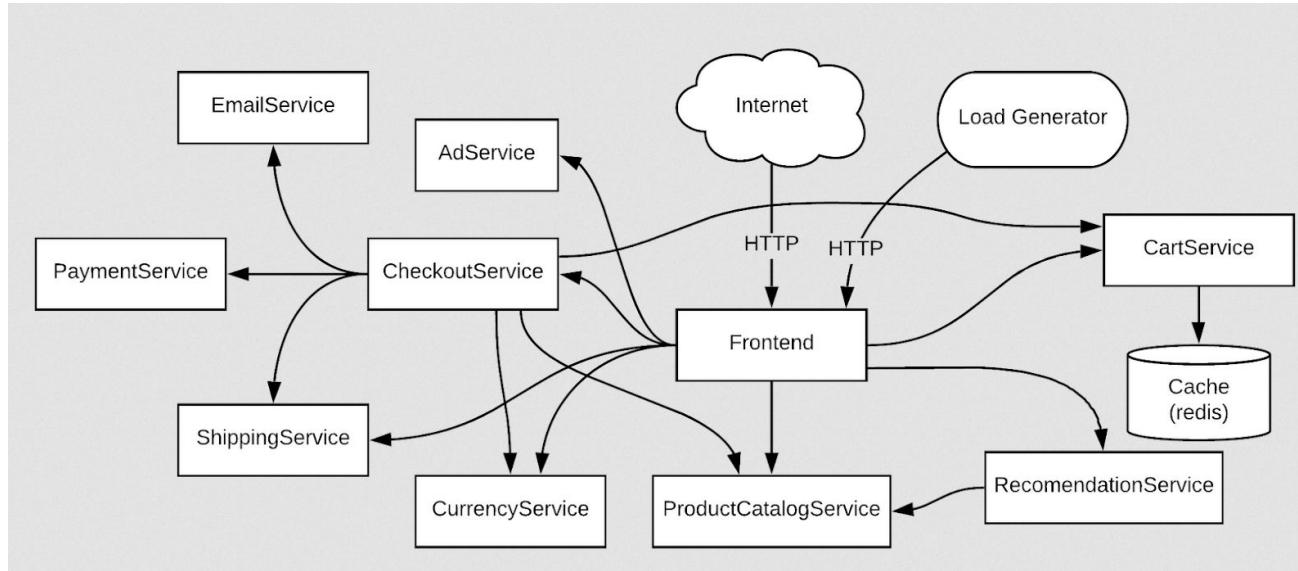
Istio menjadi salah aplikasi open source yang mengontrol traffic management, security, dan observability tanpa harus mengubah kode aplikasi.

Arsitektur ini disebut sebagai **service mesh** yang membagi istio sebagai control plane dan envoy sebagai data plane sidecar proxy.

Ketika melakukan deployment aplikasi microservice dengan arsitektur service mesh, Istio akan secara otomatis menambahkan proxy di sebelah aplikasi container dengan cara melakukan **inject proxy**.



Hands-on 3 | Microservice Application



<https://github.com/raflihadiana/kube-class/tree/main/microservice-app>