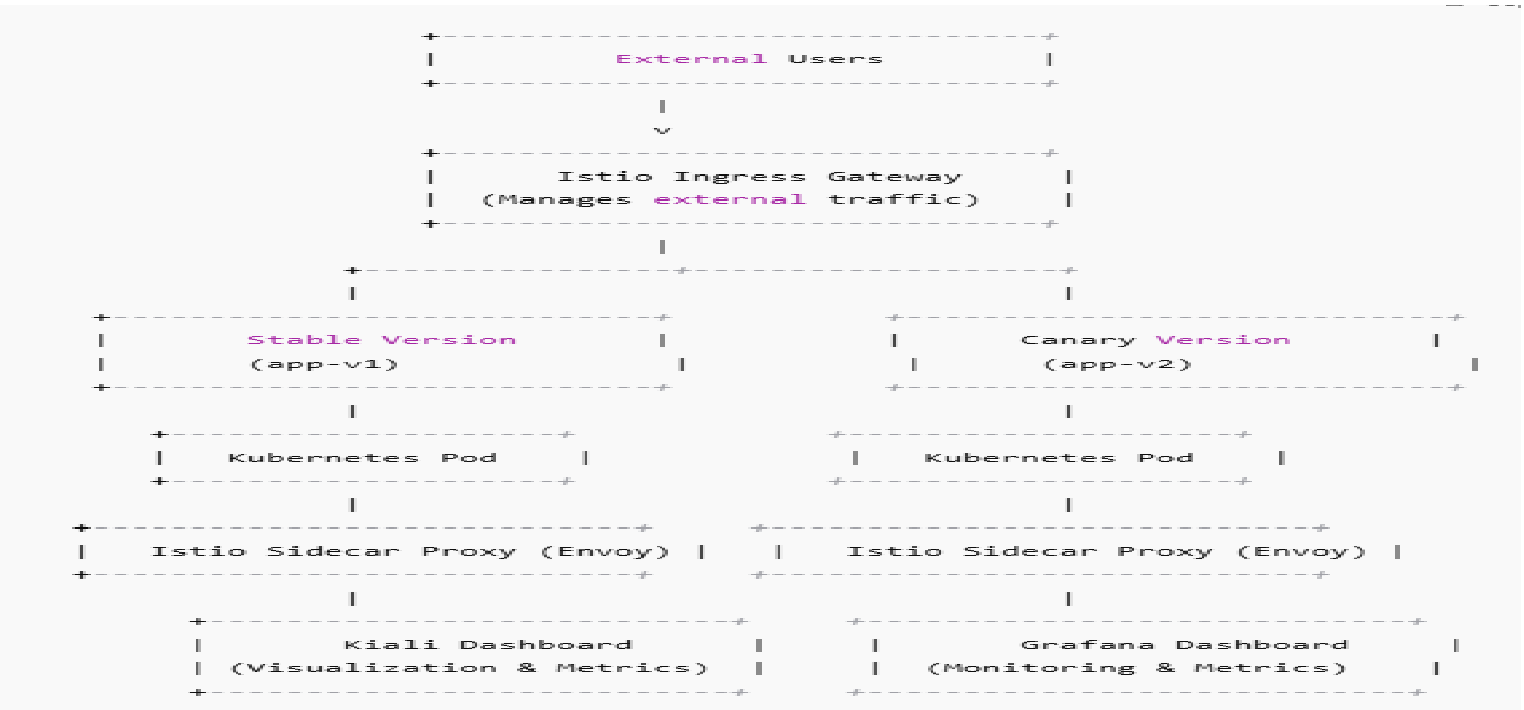
Kubernetes-Based Canary Deployment with K3s and Istio

Objective: Simulate modern canary deployments with traffic splitting between stable and new app versions.

Tools:

- K3s: Lightweight Kubernetes distribution
- Istio: Service mesh for traffic control and observability
- Docker: To containerize app
- Helm: (Optional) for managing Kubernetes apps
- App: Node.js or Python (2 versions)

Architecture



External Users

Istio Ingress Gateway
(Manages external traffic)

Stable Version
(app-v1)

Canary Version
(app-v2)

Kubernetes Pod

Kubernetes Pod

Istio Sidecar Proxy (Envoy)

Istio Sidecar Proxy (Envoy)

Kiali Dashboard
(Visualization & Metrics)

Grafana Dashboard
(Monitoring & Metrics)

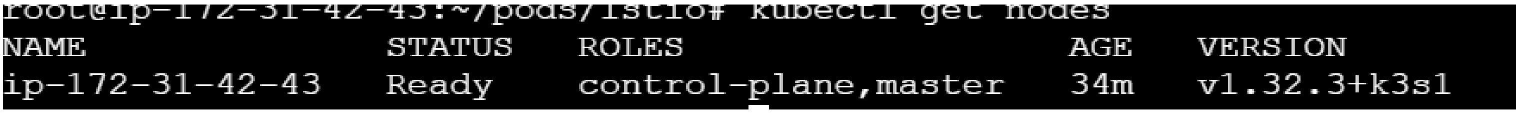
Step 1: Install K3s

curl -sfL https://get.k3s.io | sh -

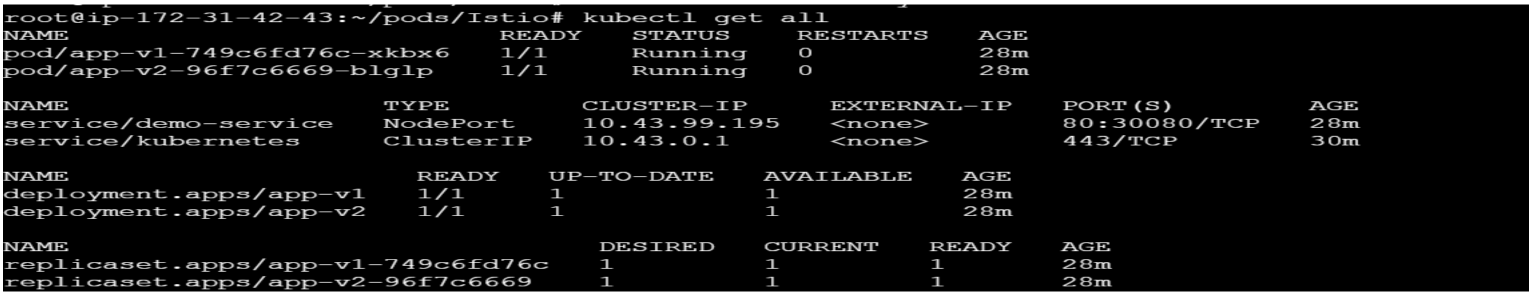
sudo chmod 644 /etc/rancher/k3s/k3s.yaml

export KUBECONFIG=/etc/rancher/k3s/k3s.yaml

kubectl cluster-info



```
root@ip-172-31-42-43:~/pods/Istio# kubectl get nodes
NAME               STATUS   ROLES                  AGE   VERSION
ip-172-31-42-43    Ready    control-plane,master   34m   v1.32.3+k3s1
```



```
root@ip-172-31-42-43:~/pods/Istio# kubectl get all
NAME                           READY   STATUS    RESTARTS   AGE
pod/app-v1-749c6fd76c-xkbx6    1/1     Running   0          28m
pod/app-v2-96f7c6669-blglp     1/1     Running   0          28m

NAME                     TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
service/demo-service     NodePort    10.43.99.195   <none>        80:30080/TCP   28m
service/kubernetes       ClusterIP   10.43.0.1      <none>        443/TCP        30m

NAME                     READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/app-v1   1/1     1            1           28m
deployment.apps/app-v2   1/1     1            1           28m

NAME                                DESIRED   CURRENT   READY   AGE
replicaset.apps/app-v1-749c6fd76c   1         1         1       28m
replicaset.apps/app-v2-96f7c6669    1         1         1       28m
```

Step 2: Install Istio with Helm

helm repo add istio https://istio-release.storage.googleapis.com/charts

helm repo update

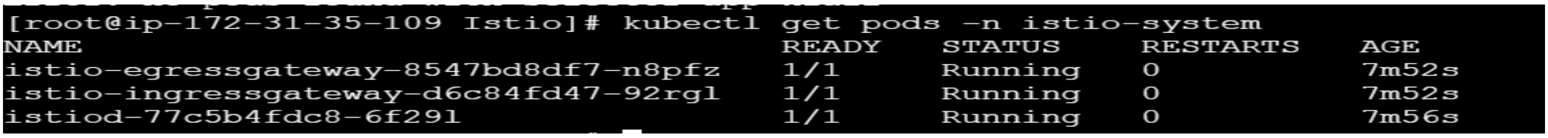
kubectl create namespace istio-system

helm install istio-base istio/base -n istio-system

helm install istiod istio/istiod -n istio-system --wait

helm install istio-ingress istio/gateway -n istio-system –wait

kubectl get pods -n istio-system # Verify Istio installation



```
[root@ip-172-31-35-109 Istio]# kubectl get pods -n istio-system
NAME                                    READY   STATUS    RESTARTS   AGE
istio-egressgateway-8547bd8df7-n8pfz    1/1     Running   0          7m52s
istio-ingressgateway-d6c84fd47-92rgl    1/1     Running   0          7m52s
istiod-77c5b4fdc8-6f291                 1/1     Running   0          7m56s
```

Step 3: Label Namespace for Istio Injection

kubectl create namespace app

kubectl label namespace app istio-injection=enabled

Step 4: Deploy Sample Application

Create a simple Node.js application with two versions (v1 and v2).

kubectl apply -f app-v1.yaml

kubectl apply -f app-v2.yaml

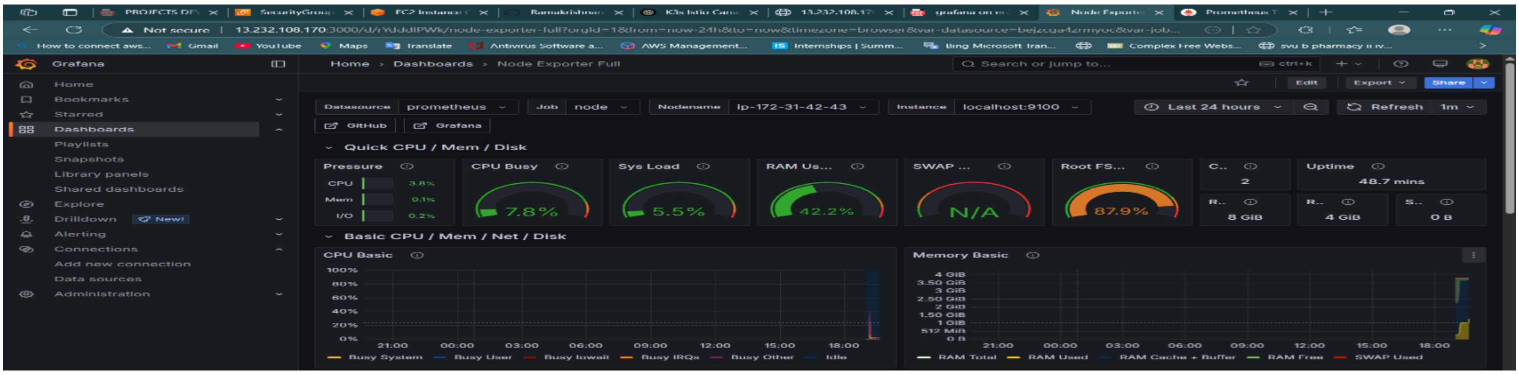
Step 5: Configure Istio Gateway and VirtualService

Vi gateway.yaml
Vi virtualservice.yaml
Vi destinationrule.yaml:

Apply the Istio configurations:

kubectl apply -f gateway.yaml
kubectl apply -f destinationrule.yaml
kubectl apply -f virtualservice.yaml

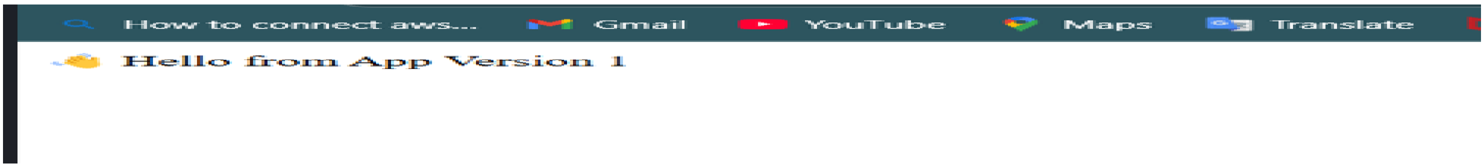
Monitoring & Metrics: Istio Telemetry (Prometheus + Grafana pre-installed with demo profile)



Datasource: prometheus · Job: node · Nodename: ip-172-31-42-43 · Instance: localhost:9100

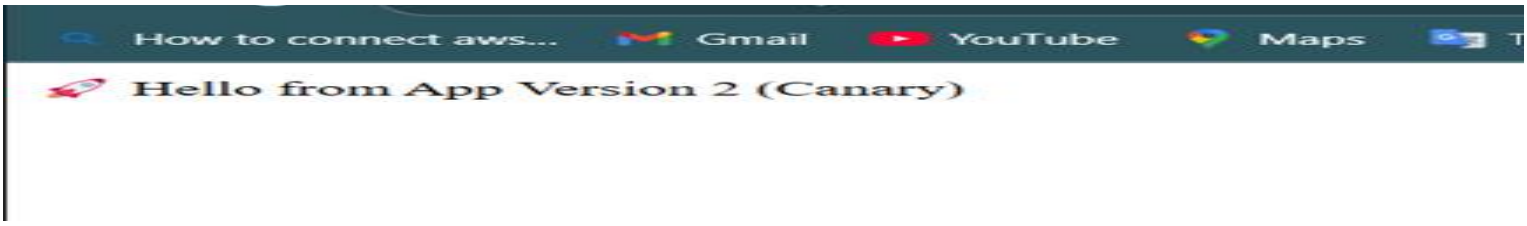
Promotion/Rollback Strategy:

Promote v1:



Hello from App Version 1

Rollback: Reverse weights or scale down v2 deployment



Hello from App Version 2 (Canary)