Kubernetes Security

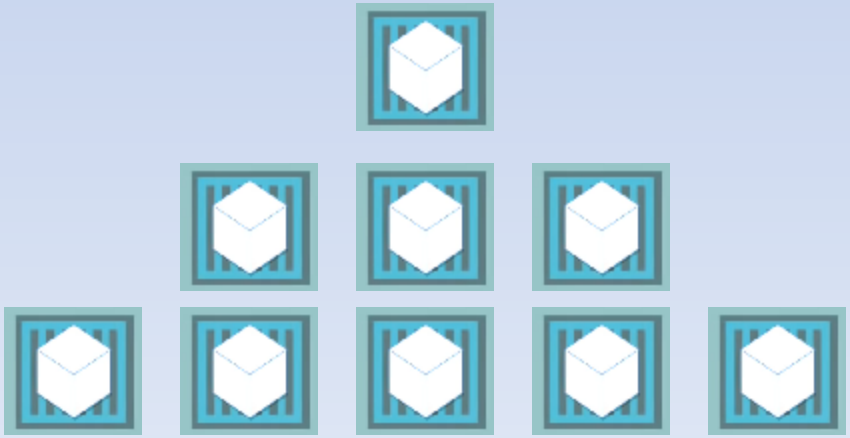
Chen Fei
Suneel Gandham
Ebhadulla Shaik

What is K8S?

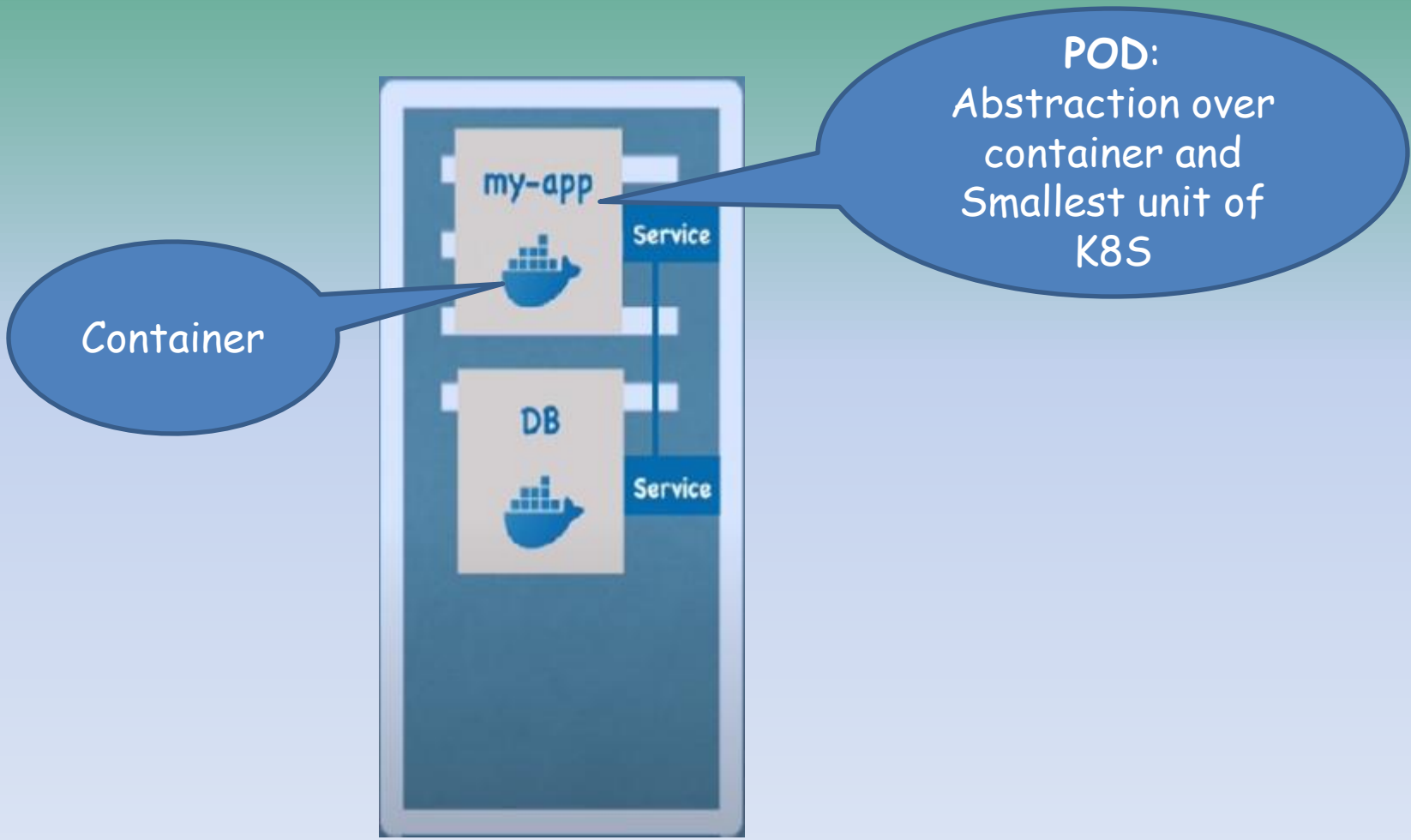
- **K8S** - Kubernetes
- **Open source** container orchestration tool
- **Developed by** Google by name Brog.
- **Helps you manage** containerized applications in different application environments

The need for a container orchestration tool

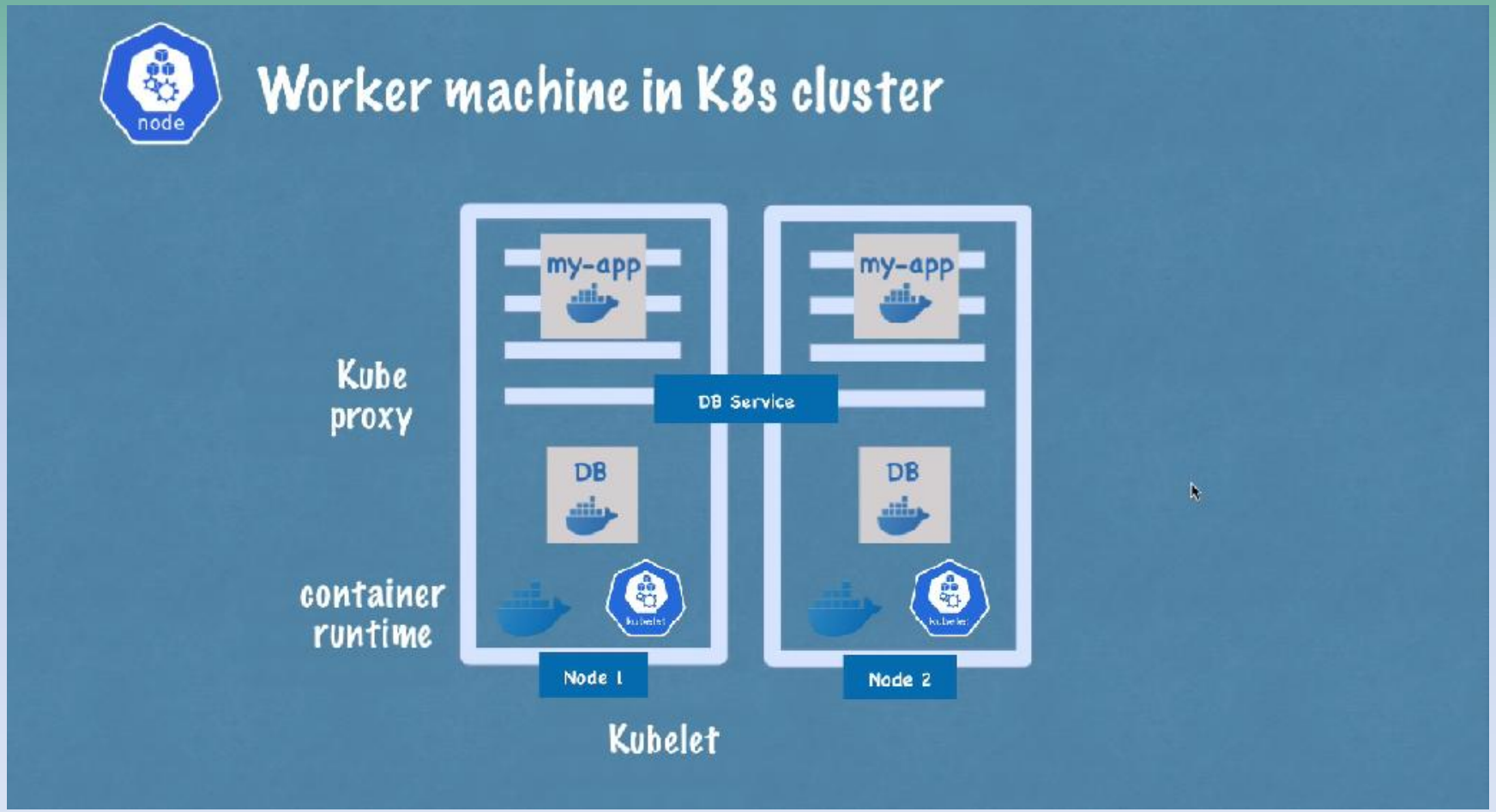
- Trend from Monolith to Microservices
- Increased usages of containers
- Need for managing hundreds of independent microservices
- Managing them self made tools is difficult
- Availability
- Scalability
- Disaster recovery



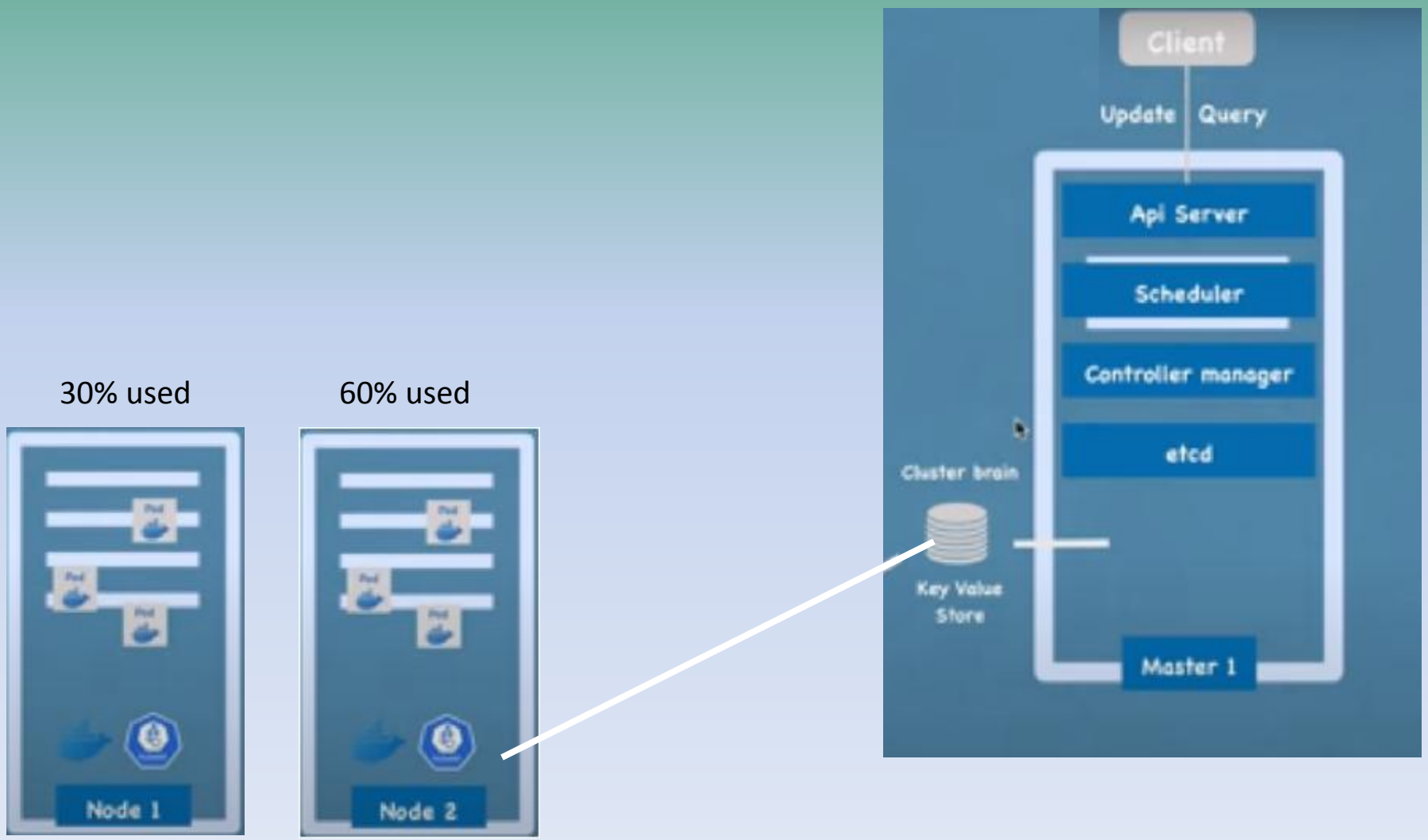
Kubernetes Basics



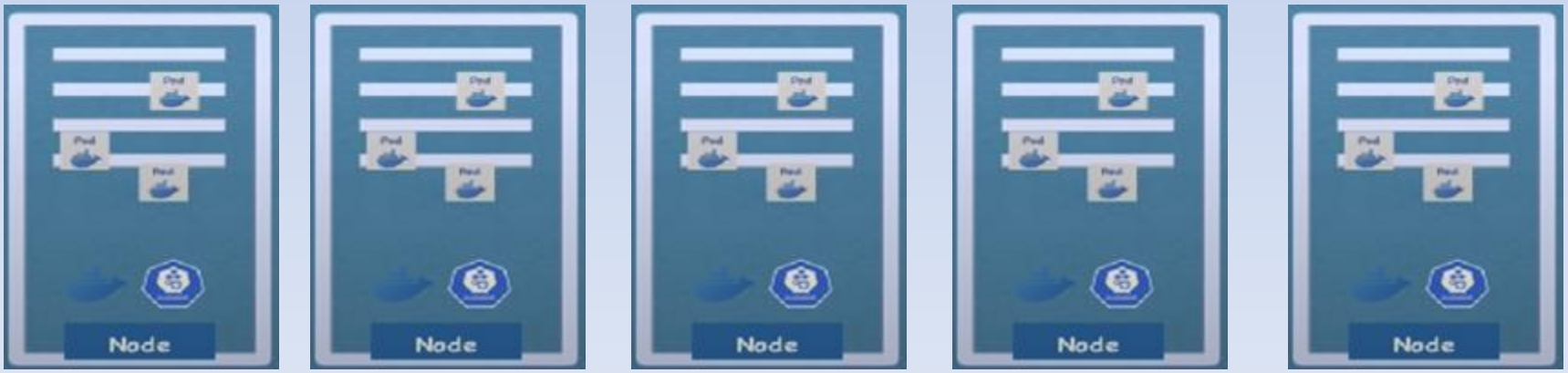
Kubernetes Architecture



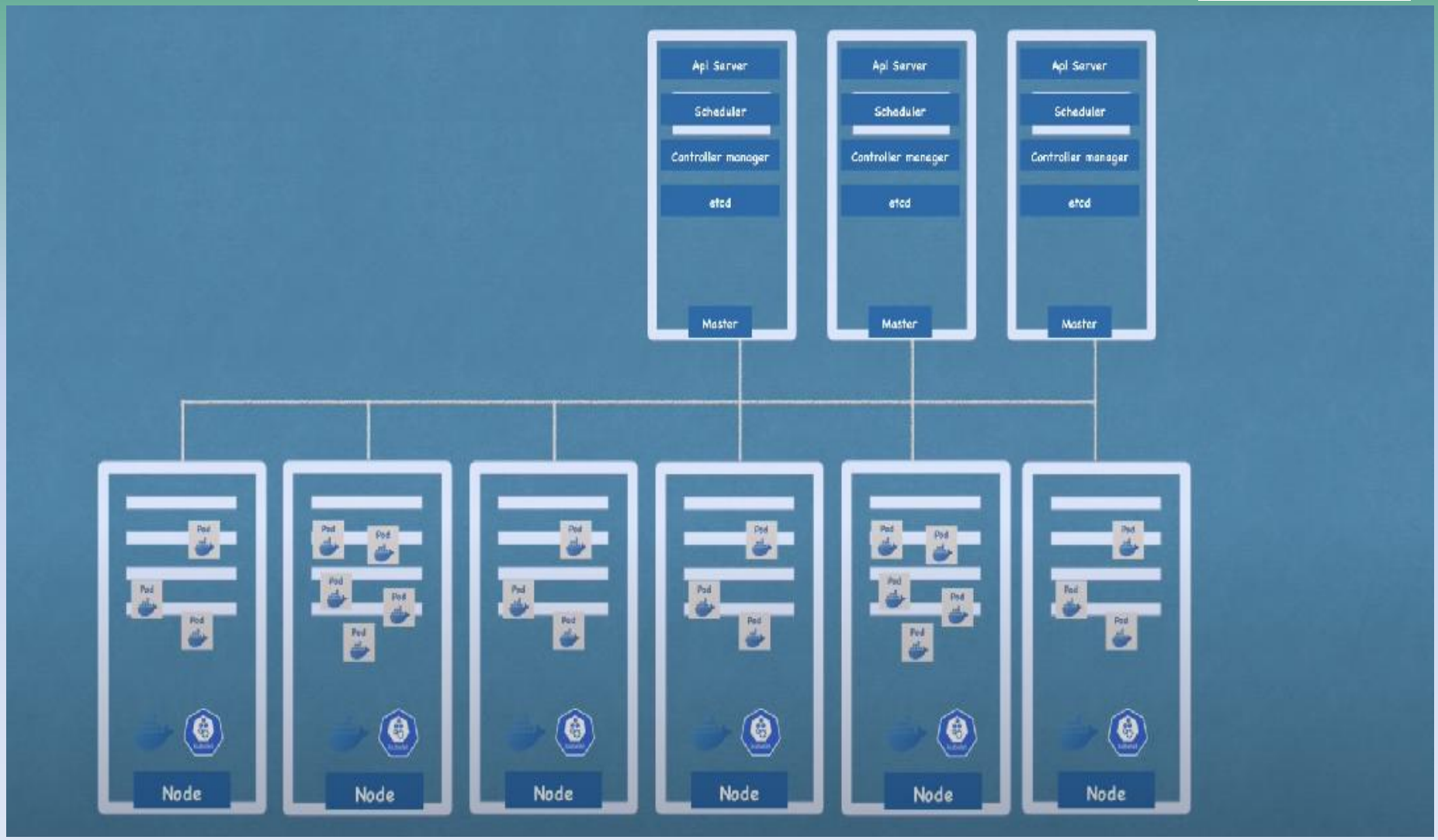
Kubernetes Architecture



Scale as needed



Cluster view



Few Key components

- Services
- Ingress
- ConfigMap
- Secrets
- Kube Proxy
- Kubelet
- Deployment
- Volumes

How Secure Kubernetes by default?

- **Misconception:** Cloud is always secure.
- It is just that, on cloud different security tools are available which we can leverage.
- K8s main aim is container orchestration and not much focus on security.
- Don't worry… we have tools to secure the infrastructure.

Building a secure image in CI/CD pipeline

Causes for insecure image:

- Code from untrusted registries
- Vulnerabilities in tools of OS or code libraries

Best Practices:

- Scanning the images in CI/CD pipeline using tools before pushing to registry.
- Scan images regularly that are pushed to registry.

Secure Container and Pods

- Running a container with root user will make attacker's job easy to escalate the privileges.
- Attacker can easily break out from the container and can access the underlying host.

Best Practice: Run container as non-root

- Create service user and run the container with that user

```
# create group and user
RUN groupadd -r myapp && useradd -g myapp myapp

# set ownership and permissions
RUN chown -R myapp:myapp /app

# switch to user
USER myapp
```

- Don't allow privilege escalation

```
apiVersion: v1
kind: Pod
metadata:
  name: my-app
spec:
  securityContext:
    runAsUser: 1000
```

```
apiVersion: v1
kind: Pod
metadata:
  name: my-app
spec:
  securityContext:
    allowPrivilegeEscalation: false
```

Secure Container and Pods

- Avoid running privileged Pods
- Set limits on cluster resources for containers
  - ❖ Resource quota
  - ❖ Limit ranges for the resources

Securing communication between Pods

- Control how the Pods communicate with each other using,

❖ NetworkPolicy

❖ Firewalls for pods.

Calico

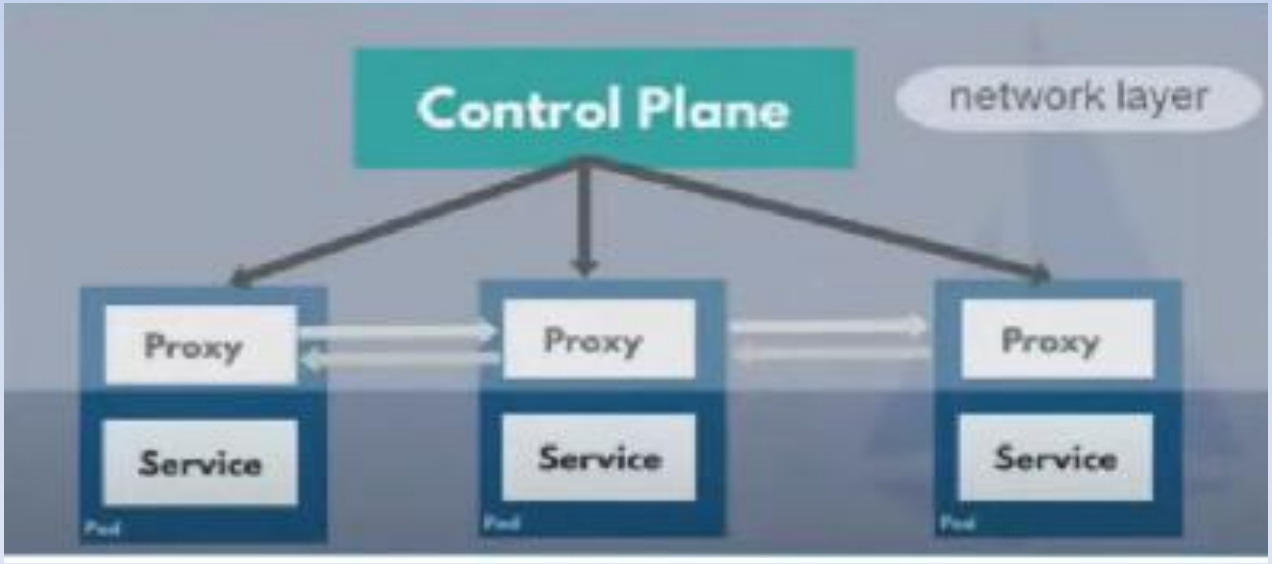
```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: my-app
spec:
  podSelector:
    matchLabels:
      name: my-app
  ingress:
  - from:
    - ipBlock:
        cidr: 172.17.0.0/16
        except:
        - 172.17.1.0/24
    - podSelector:
        matchLabels:
          name: my-db
    ports:
    - protocol: TCP
```

Maintaining Namespaces

- Namespaces help in separating the resources, there by limiting the scope in case of attacks.

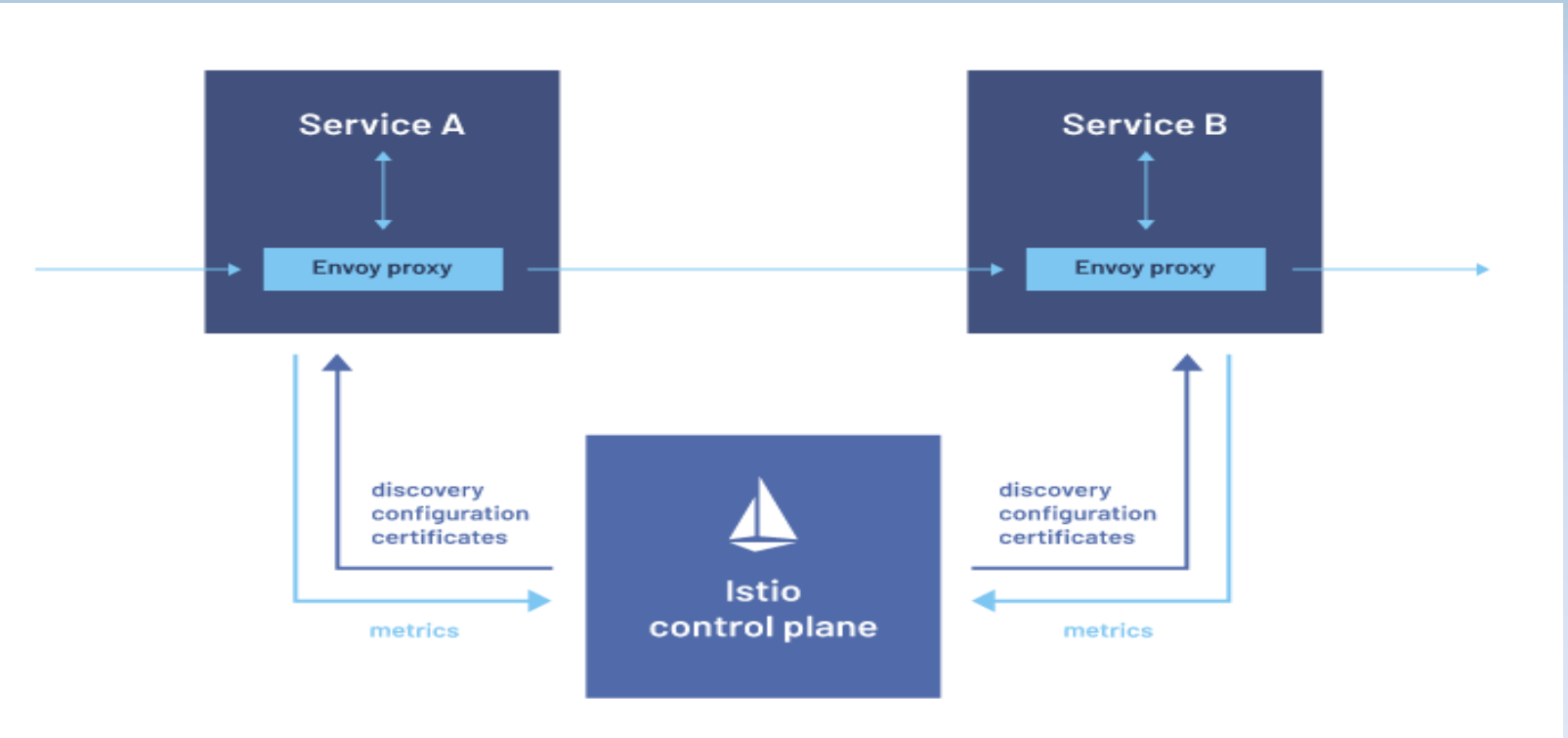
Securing communication between Pods

- Service Mesh
  - ❖ mTLS – Mutual TLS
  - ❖ Limits the communication with Network Rules
  - ❖ Abstracts the rules to a layer of infrastructure and out of individual services



Service Mesh Implementations

- Istio (native implementation) – Kiali dashboard
- Linkered
- Consul Connect(HashiCorp)



Securing etcd & secrets data

- By default secrets and configurations are stored as plain text
- Encoding them is not sufficient.
- Tools:
  - HashiCorp's Vault
  - Encryption tools from cloud providers
- Safe backup and recovery with K8s native K10.

Securing access to K8s cluster

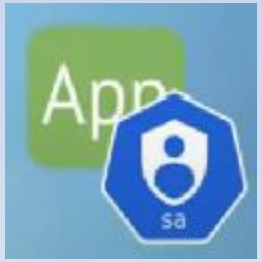
- Don’t allow access application by node port.

- LoadBalancer is better

- Ingress is even better

Securing access to K8s cluster

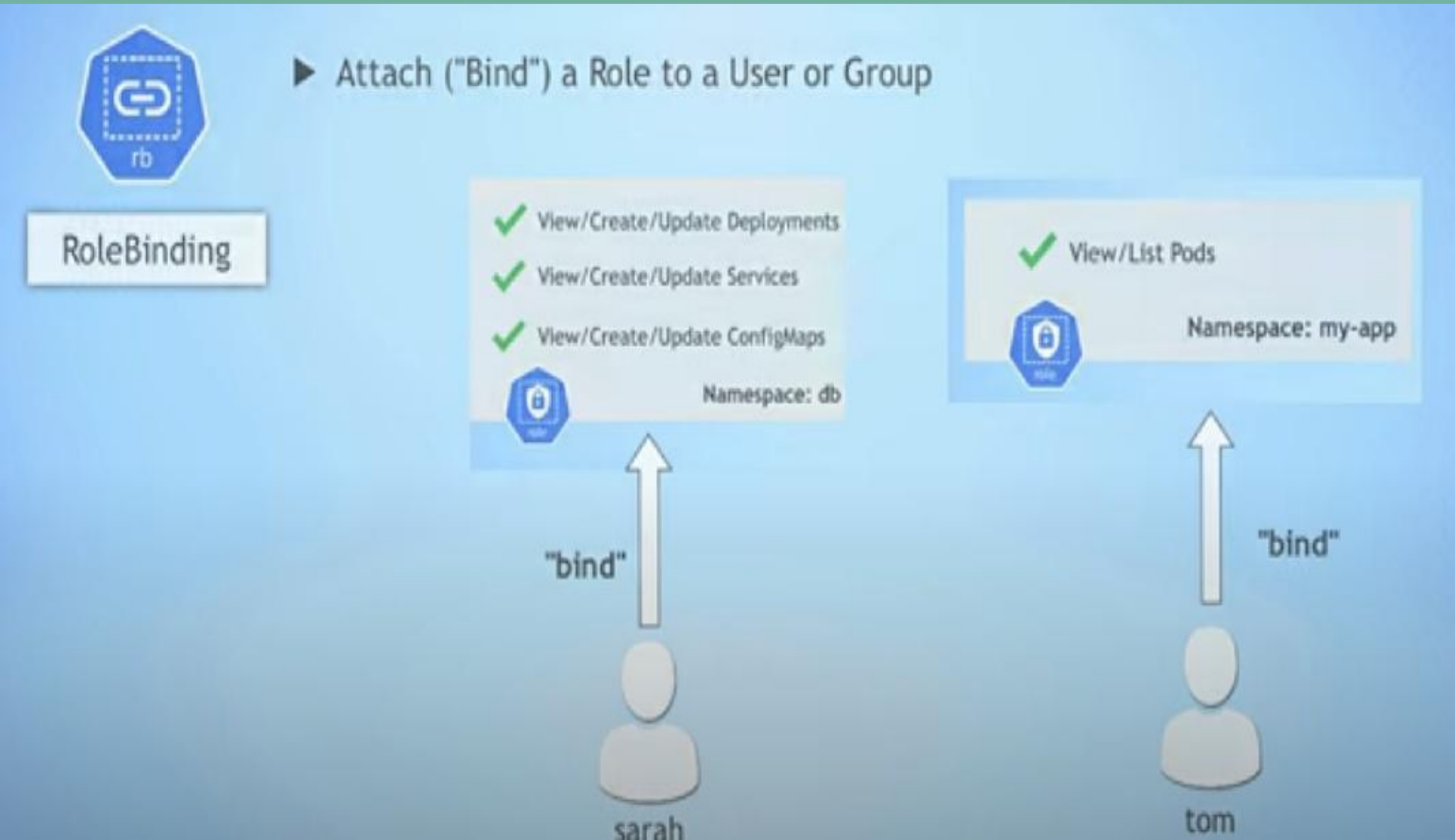
- Authentication
  - ❖Certificates for authenticating users
  - ❖Client certificates for human users
  - ❖Service Accounts for non-human users



Securing access to K8s cluster

- Authorization
  - Role Based Access Control (RBAC)
  - Cluster-wide
    - ❖ Cluster Role
    - ❖ Cluster Role Binding
  - Namespace-scoped
    - ❖ Role
    - ❖ Role Binding
  - Give least possible privileges for an user

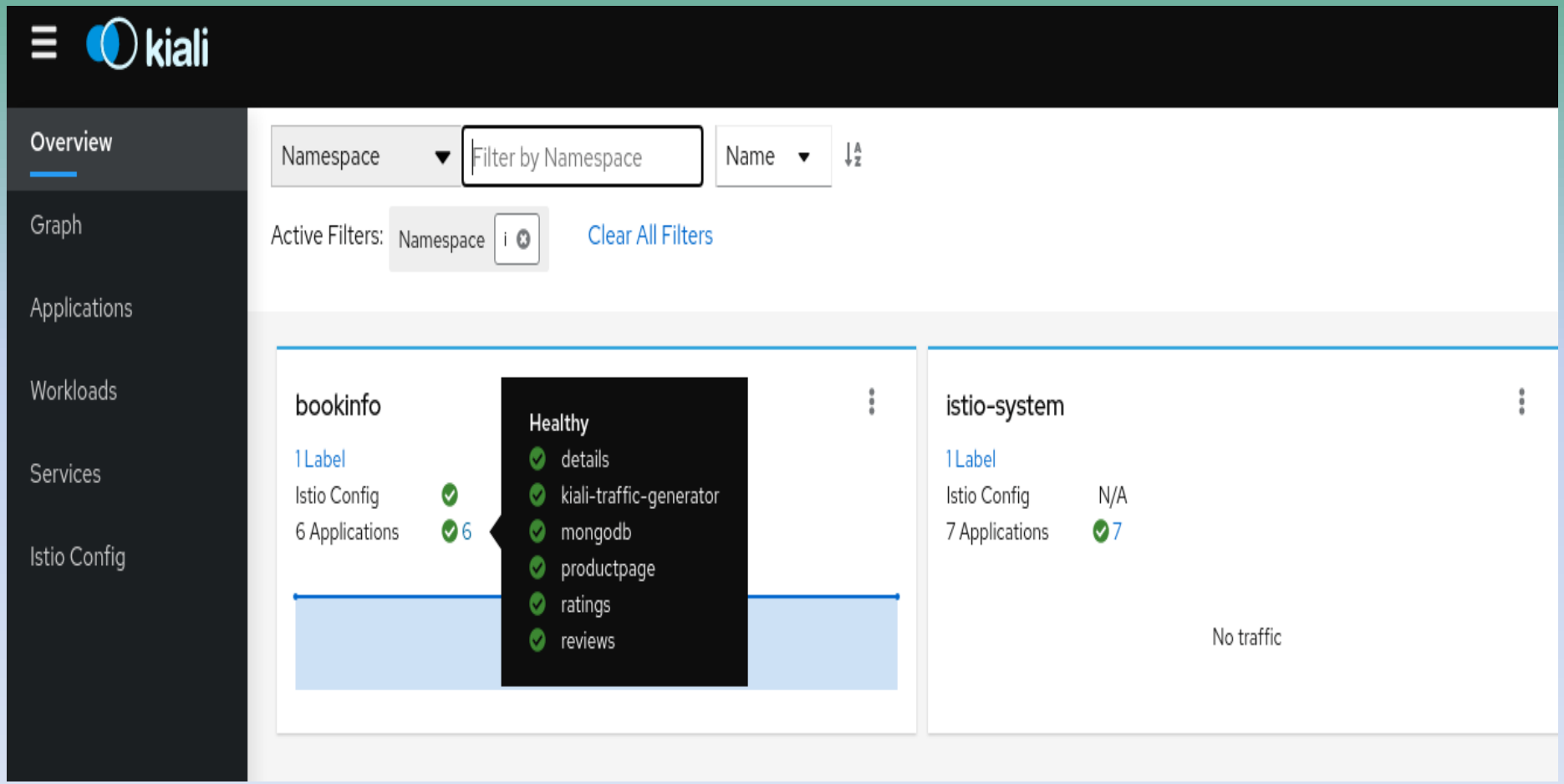
Role Based Access Control



Monitor and Mitigate Runtime Attacks

- Monitoring communications of Pods
- Using monitoring and logging tools
  - Kiali
  - Prometheus
  - Fluent Bit
  - Grafana

Kiali dashboard



Thank you