Start part 2

sas.com

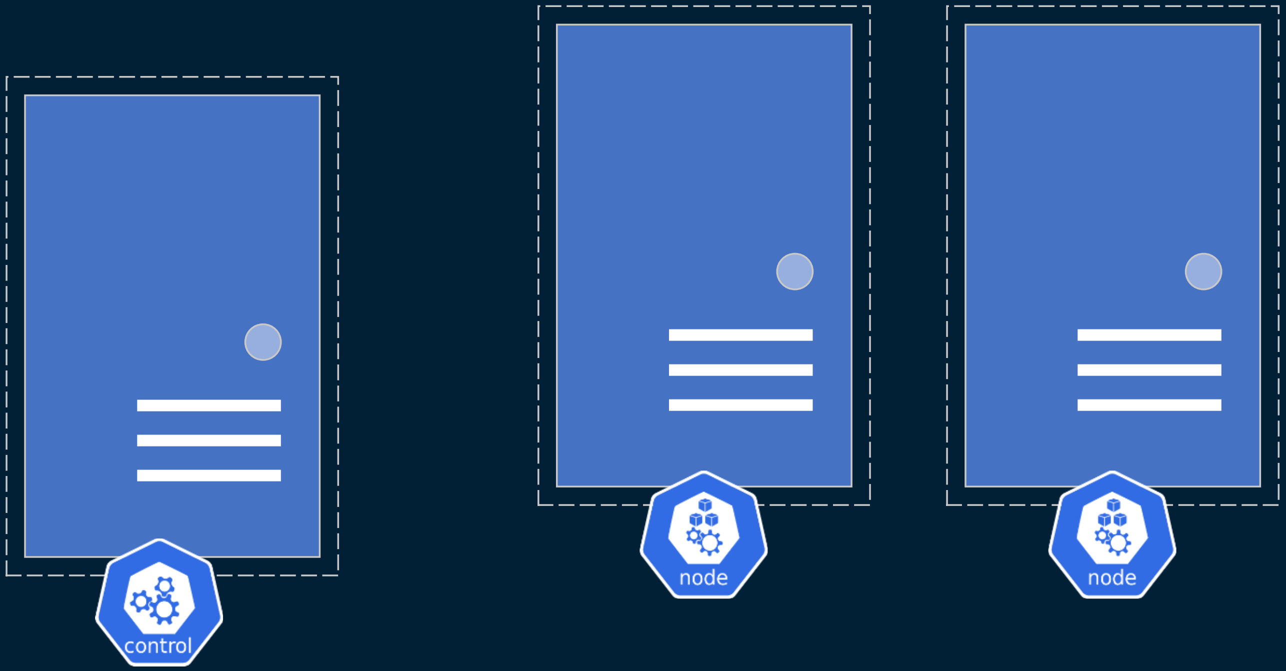
Agenda

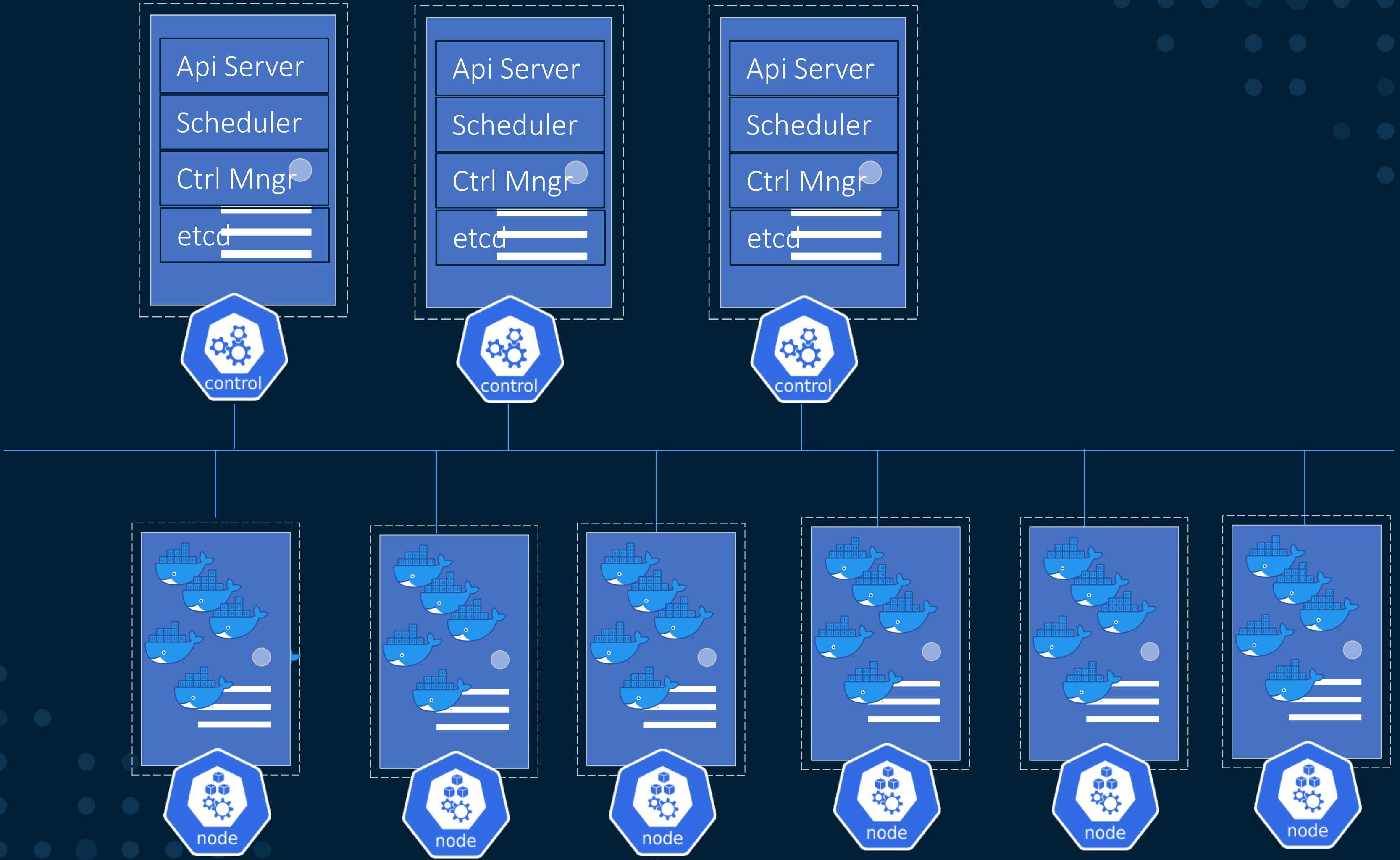
Kubernetes and components with SAS Viya4 in a Cloud environment

- What is Kubernetes (K8s)
- Main K8s Components
- **K8s Architecture**
- Organizing your components with K8s Namespaces
- SAS Viya in Kubernetes – helpful tools

Kubernetes Architecture

Basic concepts




Api Server
Scheduler
Ctrl Mngr
etcd
control
Api Server
Scheduler
Ctrl Mngr
etcd
control
Api Server
Scheduler
Ctrl Mngr
etcd
control
node
node
node
node
node
node

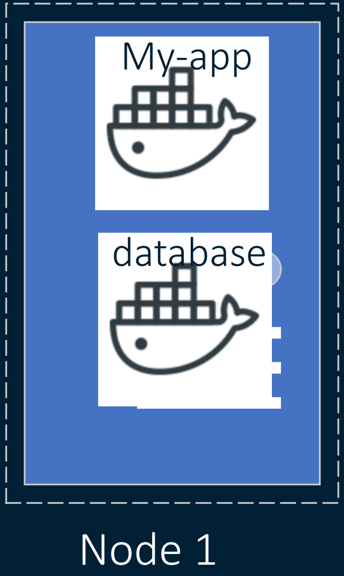
Kubernetes Architecture

Basic concepts

Node processes

Kubernetes Architecture

Basic concepts

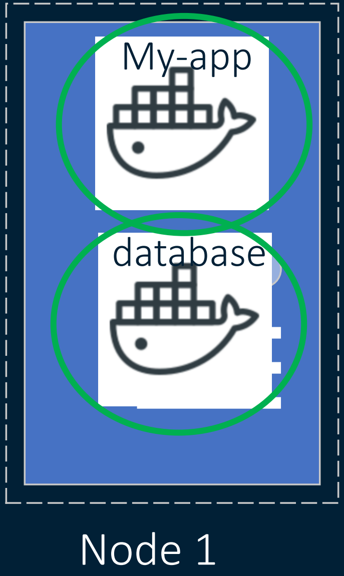


node

Worker machine in K8s cluster

Kubernetes Architecture

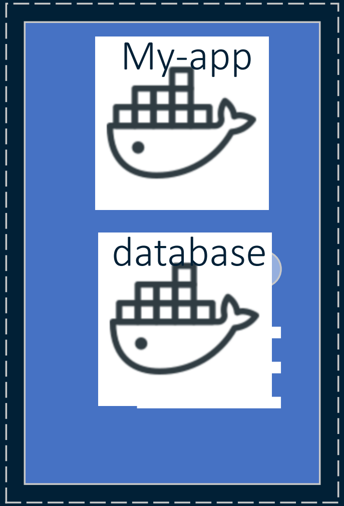
Basic concepts



- Each node has multiple Pods running on it

Kubernetes Architecture

Basic concepts



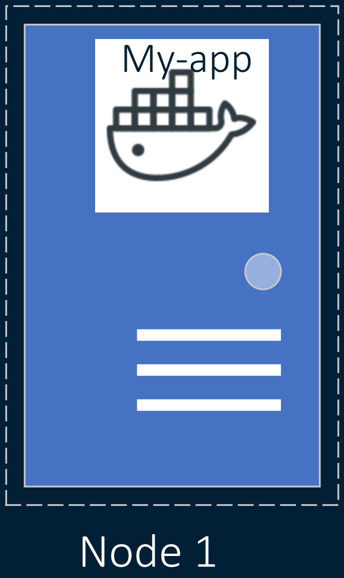
Node 1

- Each node has multiple Pods running on it
- 3 processes must be installed on every Node

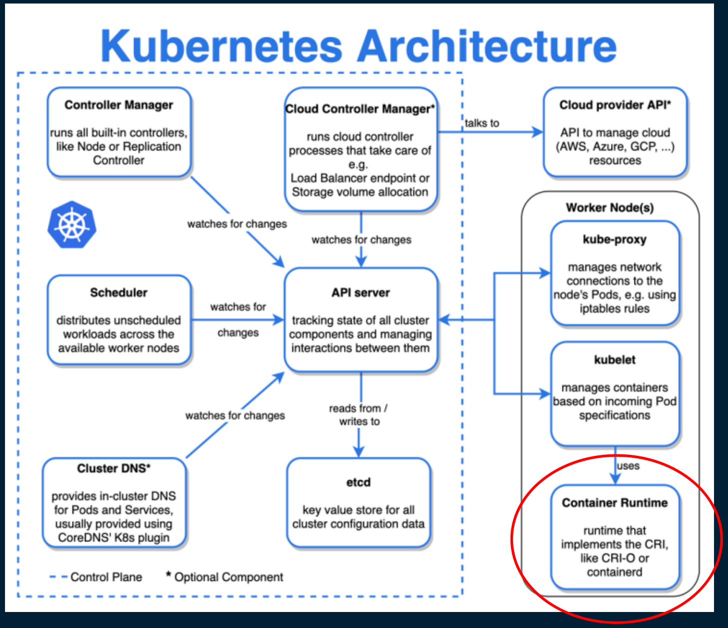
- Worker Nodes do the actual work

Kubernetes Architecture

Basic concepts



Container Runtime



Kubernetes Architecture

Controller Manager
runs all built-in controllers, like Node or Replication Controller

Cloud Controller Manager*
runs cloud controller processes that take care of e.g. Load Balancer endpoint or Storage volume allocation

Cloud provider API*
API to manage cloud (AWS, Azure, GCP, ...) resources

Scheduler
distributes unscheduled workloads across the available worker nodes

API server
tracking state of all cluster components and managing interactions between them

Cluster DNS*
provides in-cluster DNS for Pods and Services, usually provided using CoreDNS' K8s plugin

etcd
key value store for all cluster configuration data

Worker Node(s)

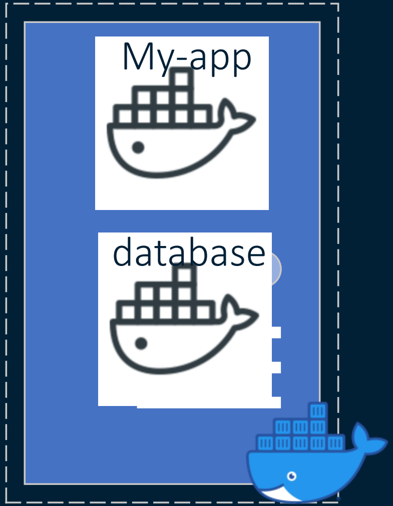
kube-proxy
manages network connections to the node's Pods, e.g. using iptables rules

kubelet
manages containers based on incoming Pod specifications

Container Runtime
runtime that implements the CRI, like CRI-O or containerd

Kubernetes Architecture

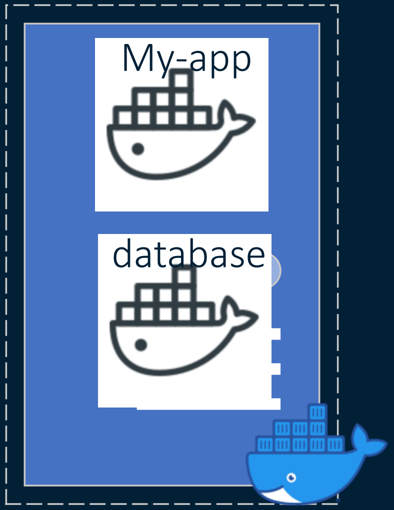
Basic concepts



Node 1

Kubernetes Architecture

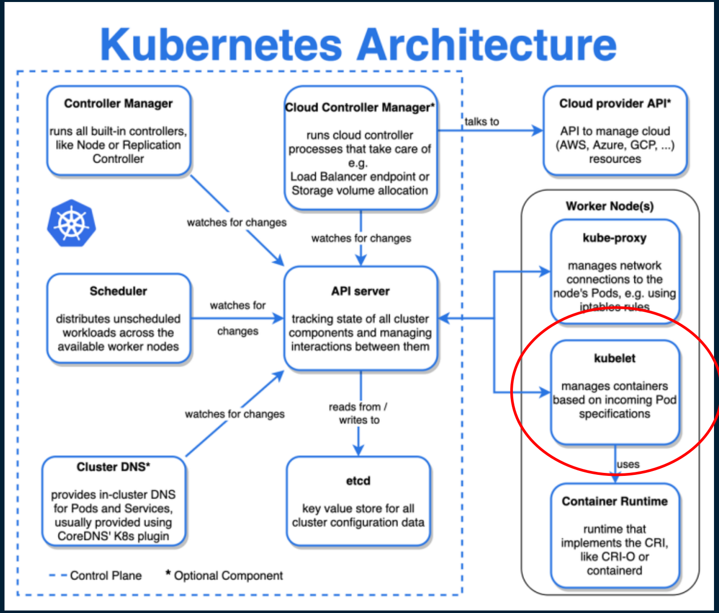
Basic concepts



Container Runtime

Node 1

Kubelet



Kubernetes Architecture

Controller Manager
runs all built-in controllers, like Node or Replication Controller

Cloud Controller Manager*
runs cloud controller processes that take care of e.g. Load Balancer endpoint or Storage volume allocation

talks to

Cloud provider API*
API to manage cloud (AWS, Azure, GCP, ...) resources

watches for changes

watches for changes

Scheduler
distributes unscheduled workloads across the available worker nodes

watches for changes

API server
tracking state of all cluster components and managing interactions between them

watches for changes

reads from / writes to

Cluster DNS*
provides in-cluster DNS for Pods and Services, usually provided using CoreDNS' K8s plugin

etcd
key value store for all cluster configuration data

Worker Node(s)

kube-proxy
manages network connections to the node's Pods, e.g. using iptables rules

kubelet
manages containers based on incoming Pod specifications

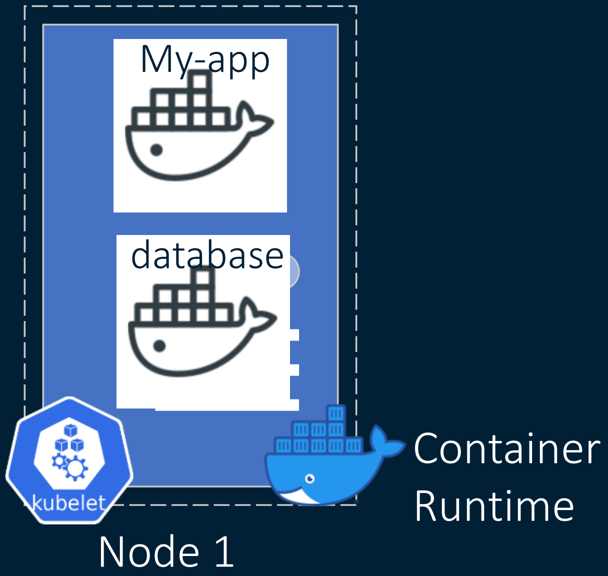
uses

Container Runtime
runtime that implements the CRI, like CRI-O or containerd

- - Control Plane * Optional Component

Kubernetes Architecture

Basic concepts

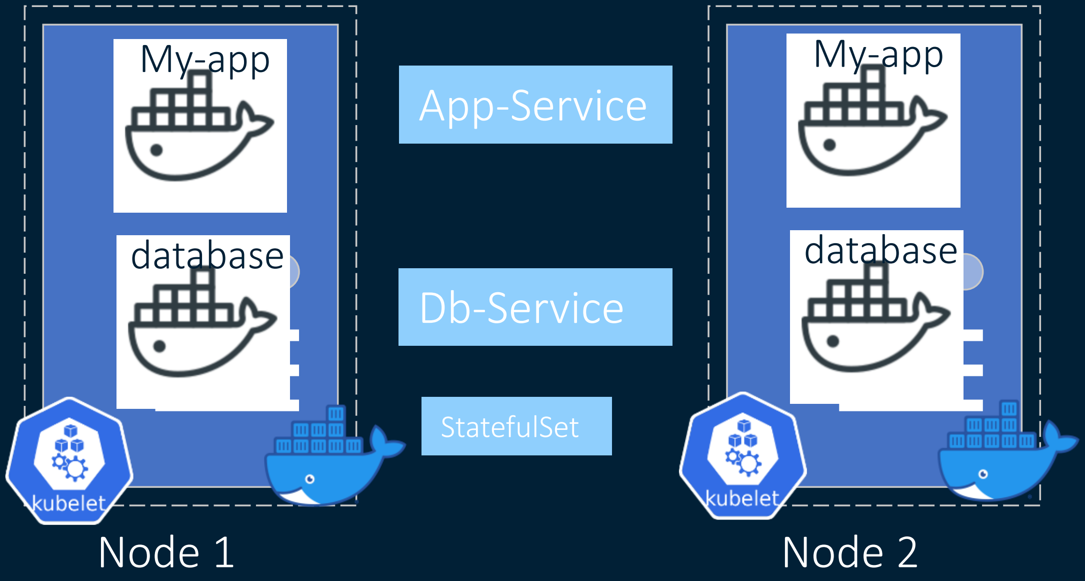


- Kubelet interacts with both –the container and the node
- Kubelet starts the pod with a container inside

Kubernetes Architecture

Deployment

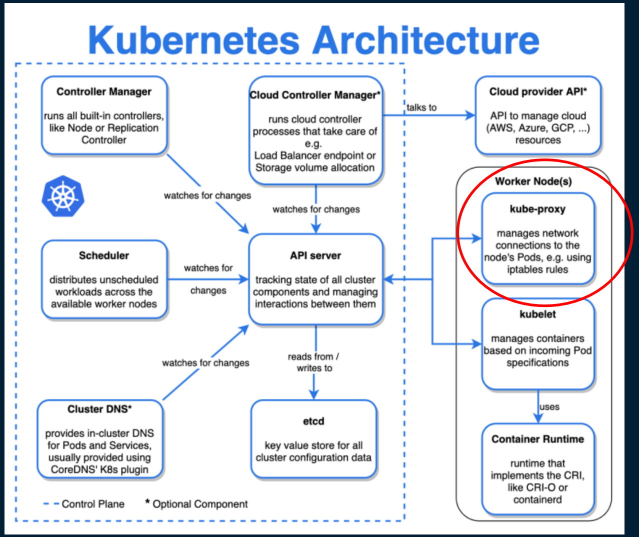
Basic concepts



k-proxy

Kube proxy

Communication via Services



Kubernetes Architecture

Controller Manager

runs all built-in controllers, like Node or Replication Controller

Cloud Controller Manager*

runs cloud controller processes that take care of e.g. Load Balancer endpoint or Storage volume allocation

Cloud provider API*

API to manage cloud (AWS, Azure, GCP, ...) resources

Scheduler

distributes unscheduled workloads across the available worker nodes

API server

tracking state of all cluster components and managing interactions between them

Cluster DNS*

provides in-cluster DNS for Pods and Services, usually provided using CoreDNS' K8s plugin

etcd

key value store for all cluster configuration data

Worker Node(s)

kube-proxy

manages network connections to the node's Pods, e.g. using iptables rules

kubelet

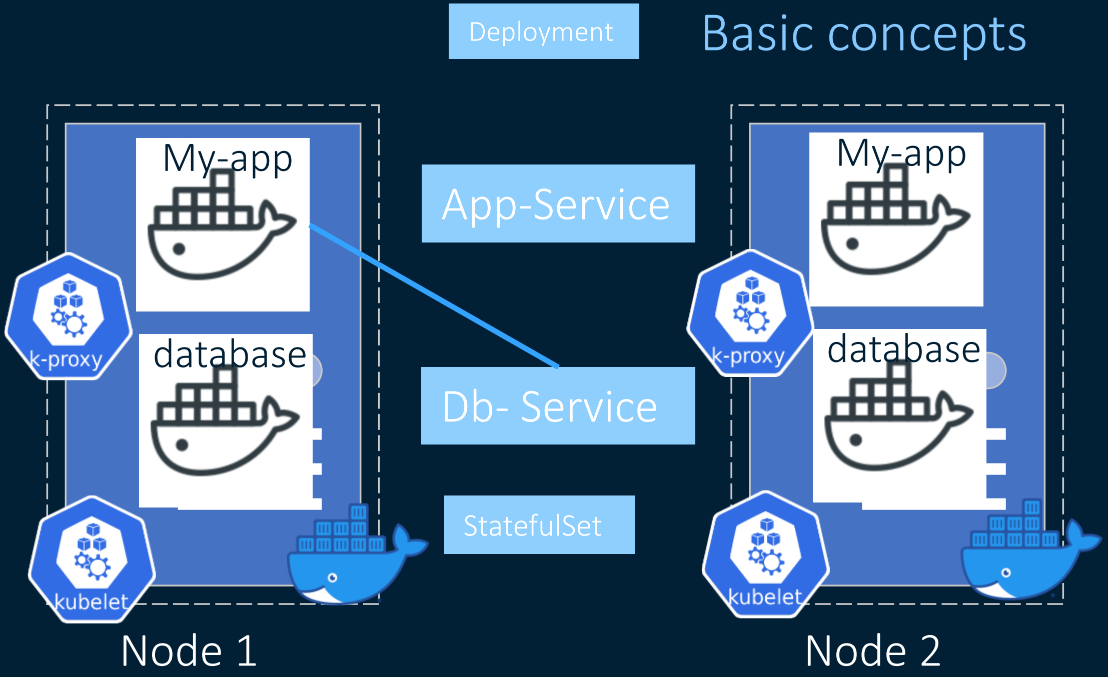
manages containers based on incoming Pod specifications

Container Runtime

runtime that implements the CRI, like CRI-O or containerd

Kubernetes Architecture

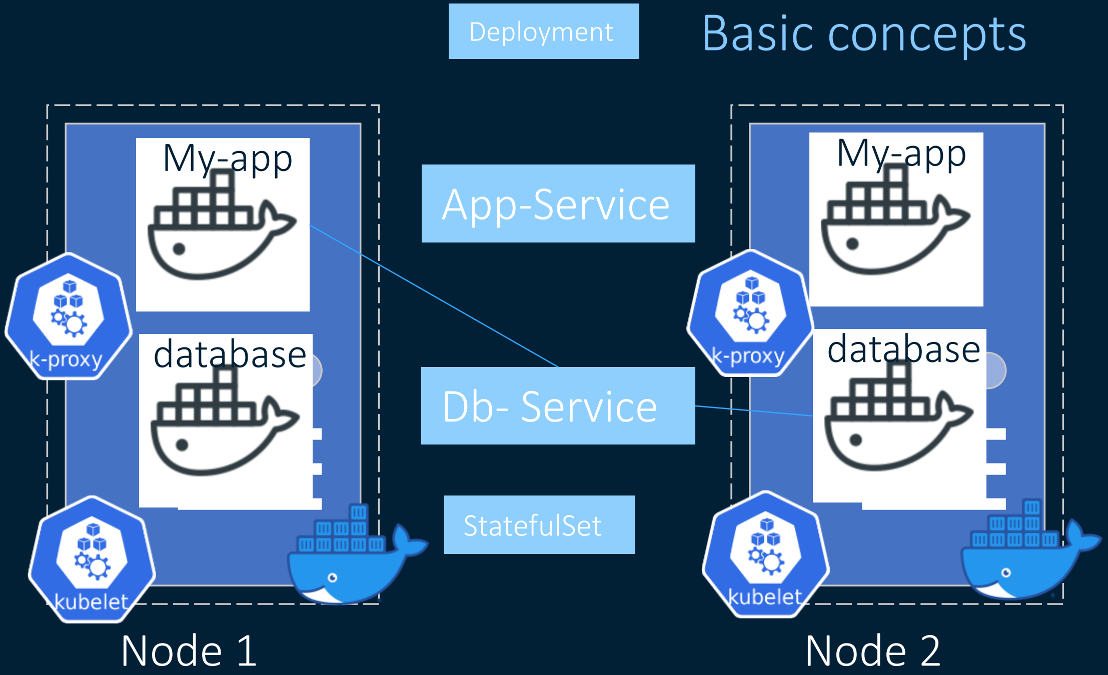
Basic concepts



Communication via Services

Kubernetes Architecture

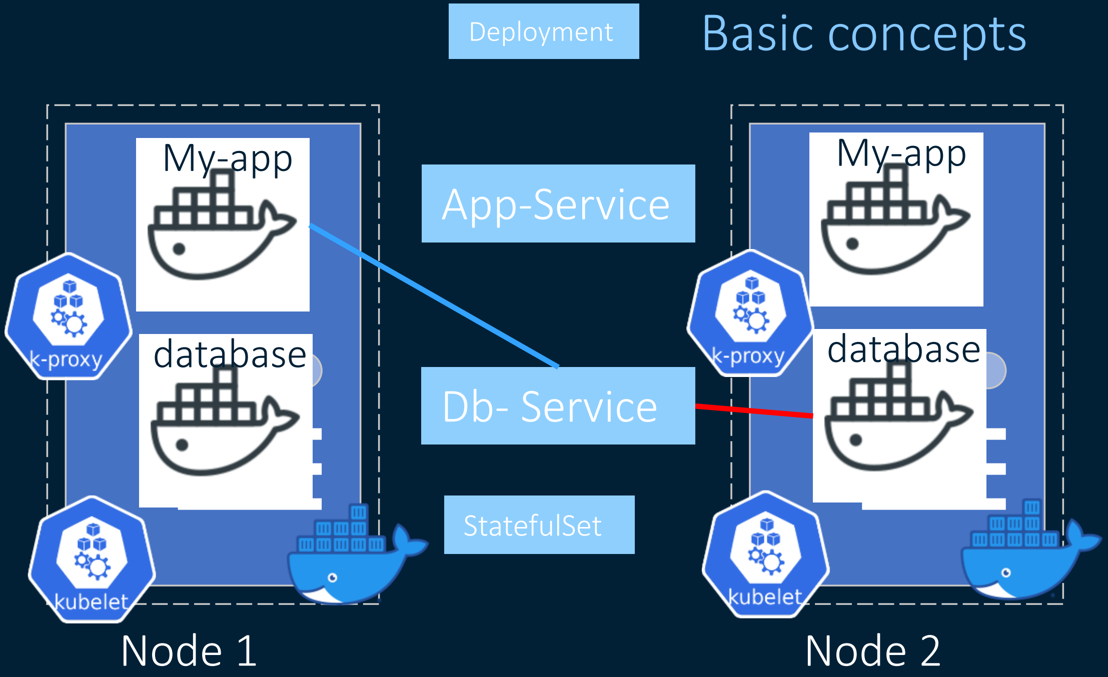
Basic concepts



Communication via Services

Kubernetes Architecture

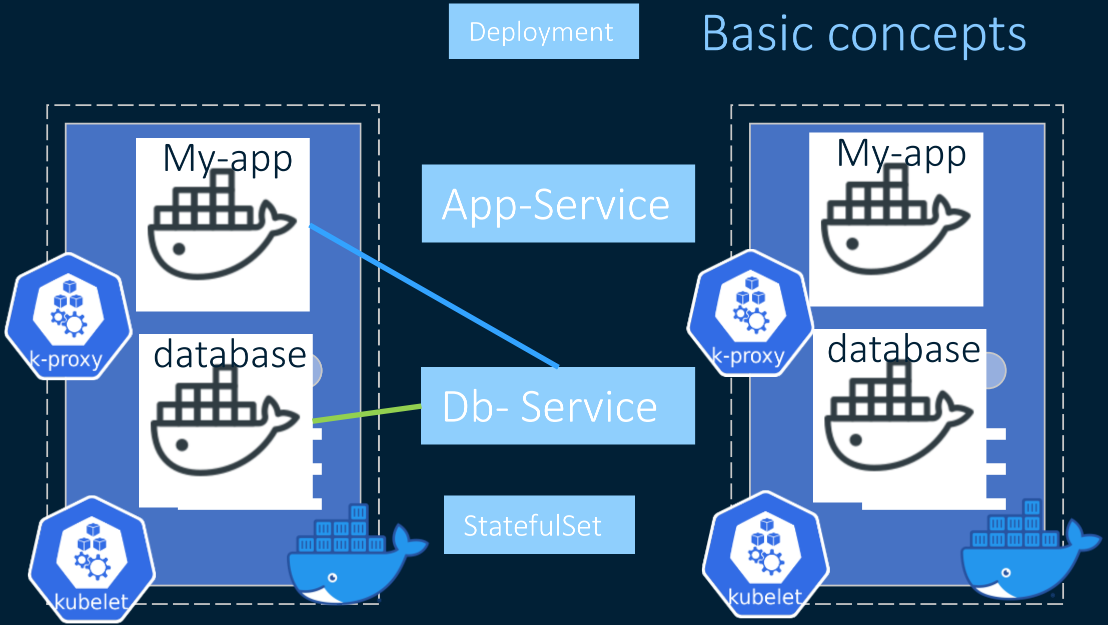
Basic concepts



Communication via Services

Kubernetes Architecture

Basic concepts



Node 1

Node 2

Communication via Services

Kubernetes Architecture

Interaction with the cluster

How to:

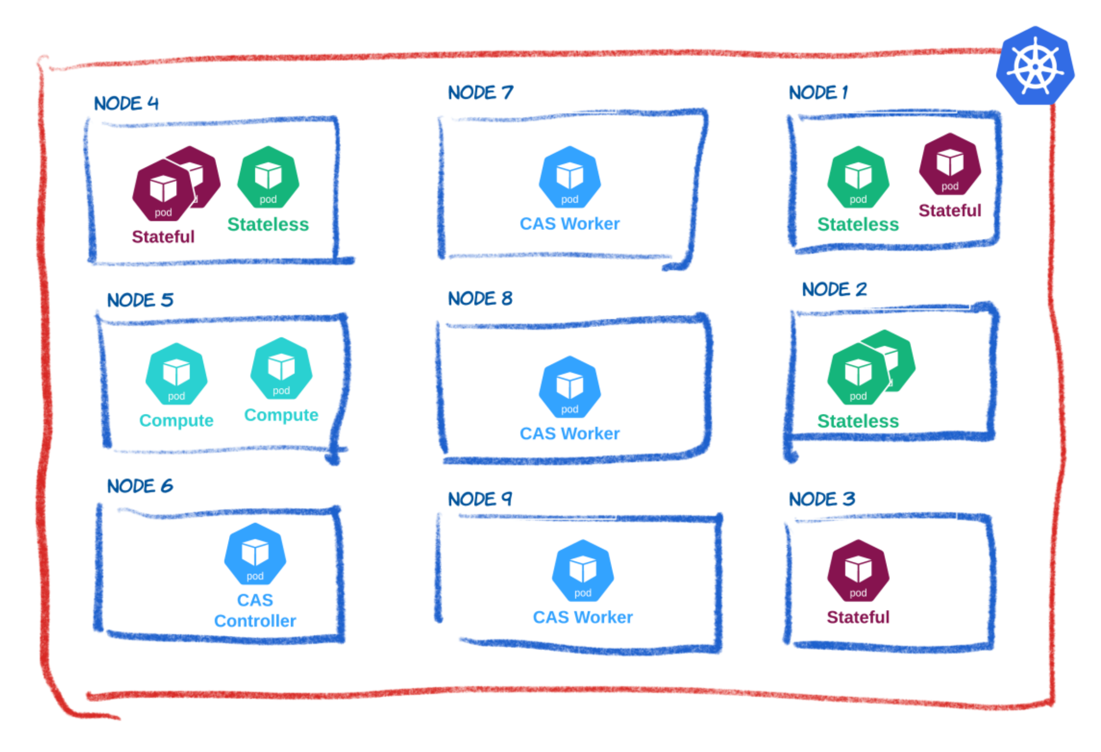
- Schedule pods on which Nodes?
- Monitor the cluster and the running of the pods?
- Re-schedule/restart pods that dies?
- Join a new Node to the cluster?

Kubernetes Architecture

Thinking about SAS Viya

For example, I need

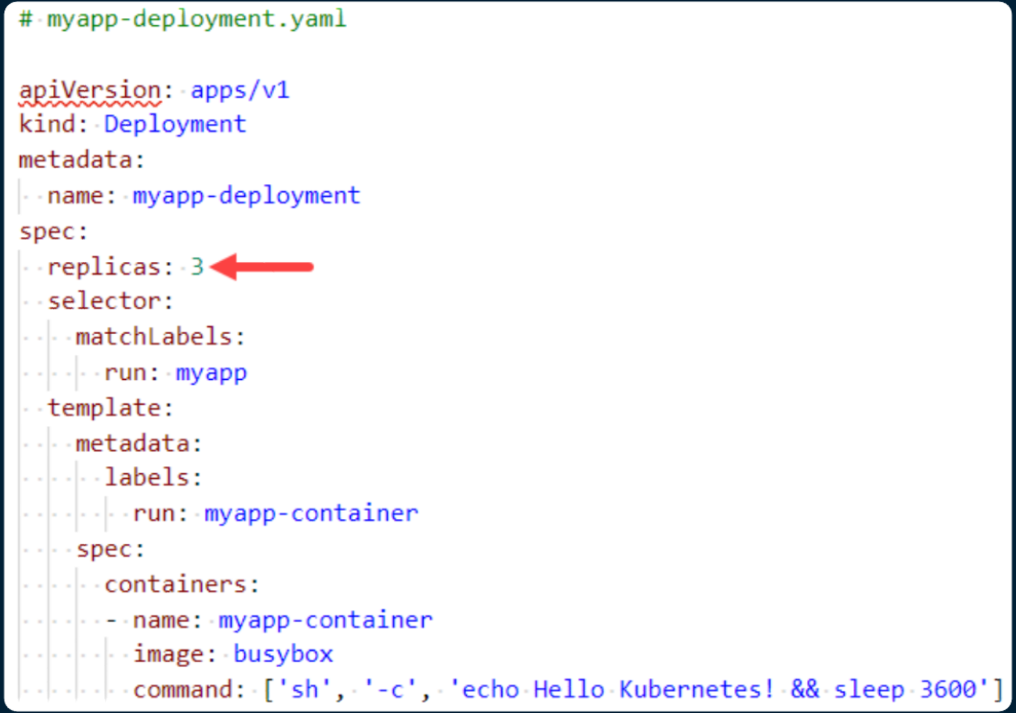
- 3 x CAS Workers
- 2 Compute Servers



Kubernetes Architecture

You declare the desired state

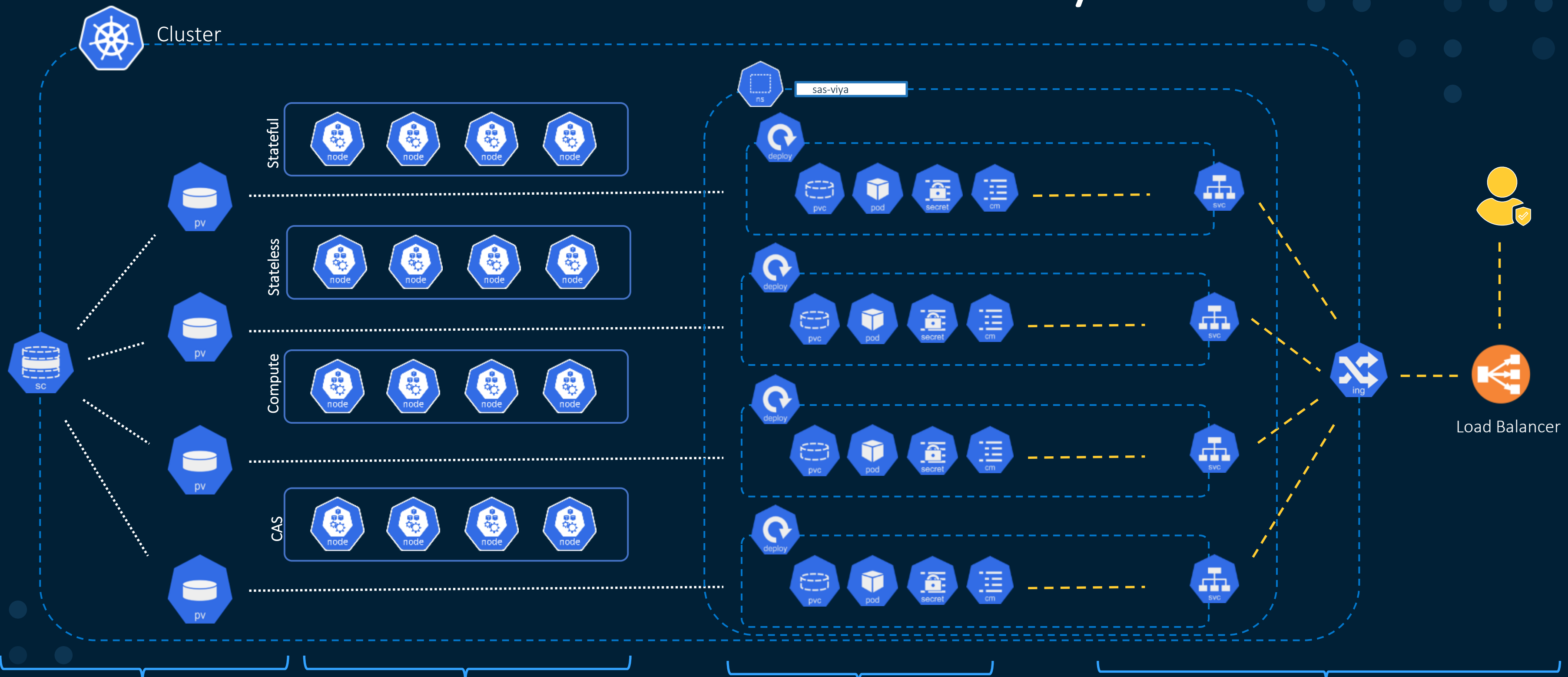
- It is Kubernetes job to ensure that the desired state is achieved
- That the required number of pod replicas are running
- In this example we are declaring that we want 3 replicas of the ‘myapp-container’



```yaml
# myapp-deployment.yaml

apiVersion: apps/v1
kind: Deployment
metadata:
  name: myapp-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      run: myapp
  template:
    metadata:
      labels:
        run: myapp-container
    spec:
      containers:
      - name: myapp-container
        image: busybox
        command: ['sh', '-c', 'echo Hello Kubernetes! && sleep 3600']
```

Kubernetes cluster & SAS Viya



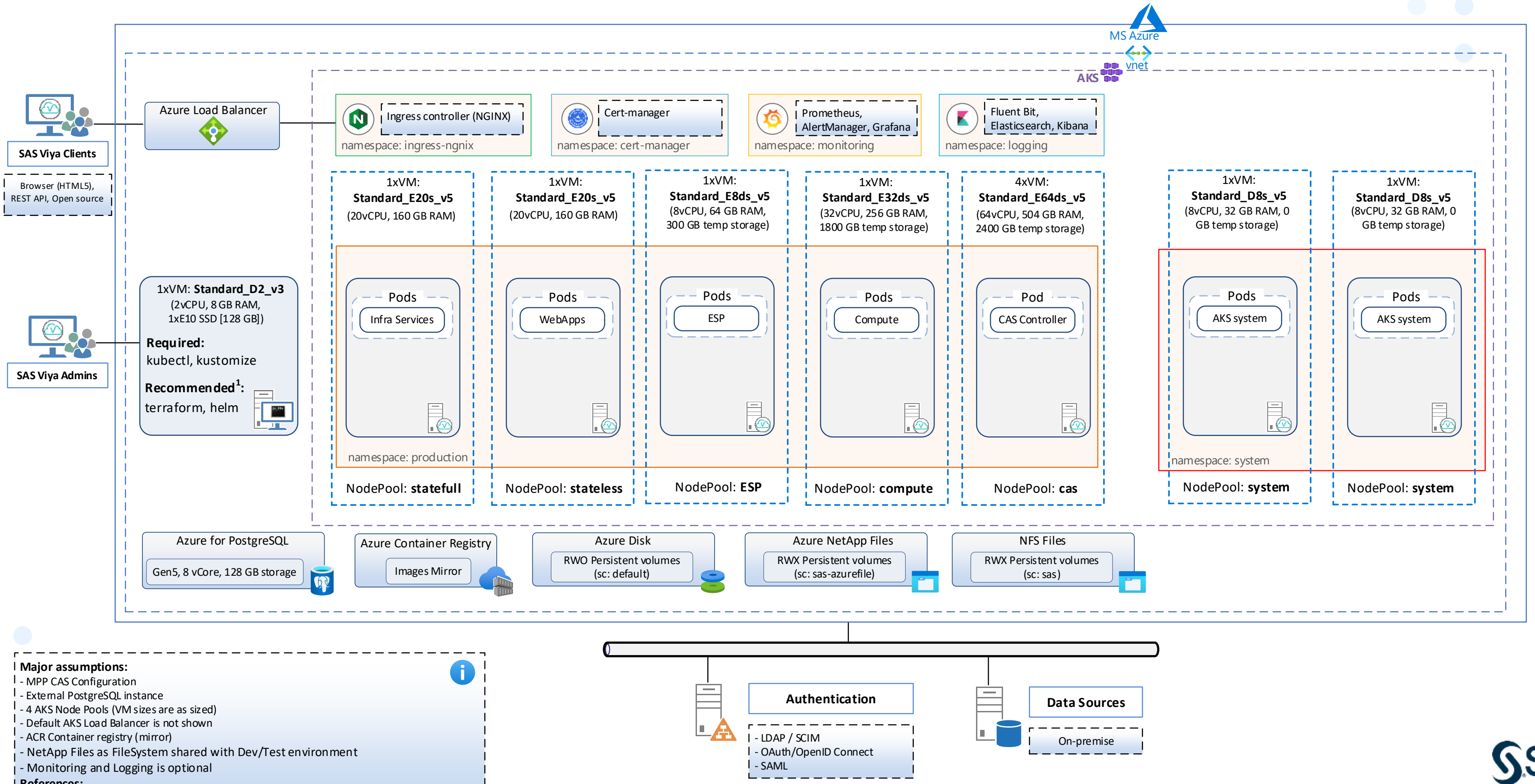
Storage, Physical Volumes & Claims
Different types of storage are access through StorageClasses, making external storage available to pods through PhysicalVolumes.

Nodes & workload classes
The cluster is composed of nodes, divided by pools that define which types of workloads run where.

Deployments, Pods & other resources
Pods, where micro-services run, are part of a Deployment, with Secrets, Configs, etc, inside a namespace. These are the backbone of SAS Viya.

Services, Ingress, Load Balancers
Each Deployment has a Service, which is accessed through an Ingress Controller (Layer 7) : this exposes the pods to the outside, using an external Load Balancer (Layer 4)

SAS Viya4 Deployment on Azure Kubernetes Service (AKS)
Customer A – Production


MS Azure
vnet
AKS
SAS Viya Clients
Browser (HTML5), REST API, Open source
SAS Viya Admins
Azure Load Balancer
Ingress controller (NGINX)
namespace: ingress-ngnix
Cert-manager
namespace: cert-manager
Prometheus, AlertManager, Grafana
namespace: monitoring
Fluent Bit, Elasticsearch, Kibana
namespace: logging
1xVM: Standard_D2_v3 (2vCPU, 8 GB RAM, 1xE10 SSD [128 GB])
Required: kubectl, kustomize
Recommended¹: terraform, helm
1xVM: Standard_E20s_v5 (20vCPU, 160 GB RAM)
Pods
Infra Services
NodePool: statefull
1xVM: Standard_E20s_v5 (20vCPU, 160 GB RAM)
Pods
WebApps
NodePool: stateless
1xVM: Standard_E8ds_v5 (8vCPU, 64 GB RAM, 300 GB temp storage)
Pods
ESP
NodePool: ESP
1xVM: Standard_E32ds_v5 (32vCPU, 256 GB RAM, 1800 GB temp storage)
Pods
Compute
NodePool: compute
4xVM: Standard_E64ds_v5 (64vCPU, 504 GB RAM, 2400 GB temp storage)
Pod
CAS Controller
NodePool: cas
namespace: production
1xVM: Standard_D8s_v5 (8vCPU, 32 GB RAM, 0 GB temp storage)
Pods
AKS system
NodePool: system
1xVM: Standard_D8s_v5 (8vCPU, 32 GB RAM, 0 GB temp storage)
Pods
AKS system
NodePool: system
namespace: system
Azure for PostgreSQL
Gen5, 8 vCore, 128 GB storage
Azure Container Registry
Images Mirror
Azure Disk
RWO Persistent volumes (sc: default)
Azure NetApp Files
RWX Persistent volumes (sc: sas-azurefile)
NFS Files
RWX Persistent volumes (sc: sas)
Authentication
- LDAP / SCIM
- OAuth/OpenID Connect
- SAML
Data Sources
On-premise

Major assumptions:
- MPP CAS Configuration
- External PostgreSQL instance
- 4 AKS Node Pools (VM sizes are as sized)
- Default AKS Load Balancer is not shown
- ACR Container registry (mirror)
- NetApp Files as FileSystem shared with Dev/Test environment
- Monitoring and Logging is optional

References:
1) others: Azure CLI, SAS Mirror Manager, SAS Viya Orders CLI, ansible, python3, pip3, jq, k9s, git

Kubernetes Architecture

Interaction with the cluster

Managing processes are done by

Master Nodes - Control

Kubernetes Architecture

Interaction with the cluster

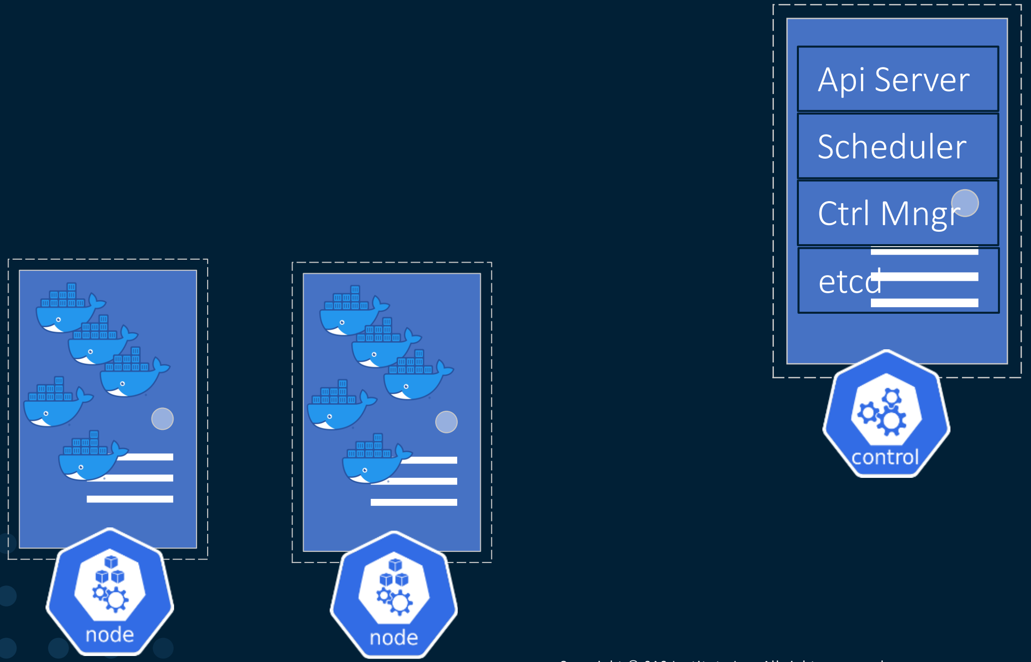
control

Master processes

Kubernetes Architecture

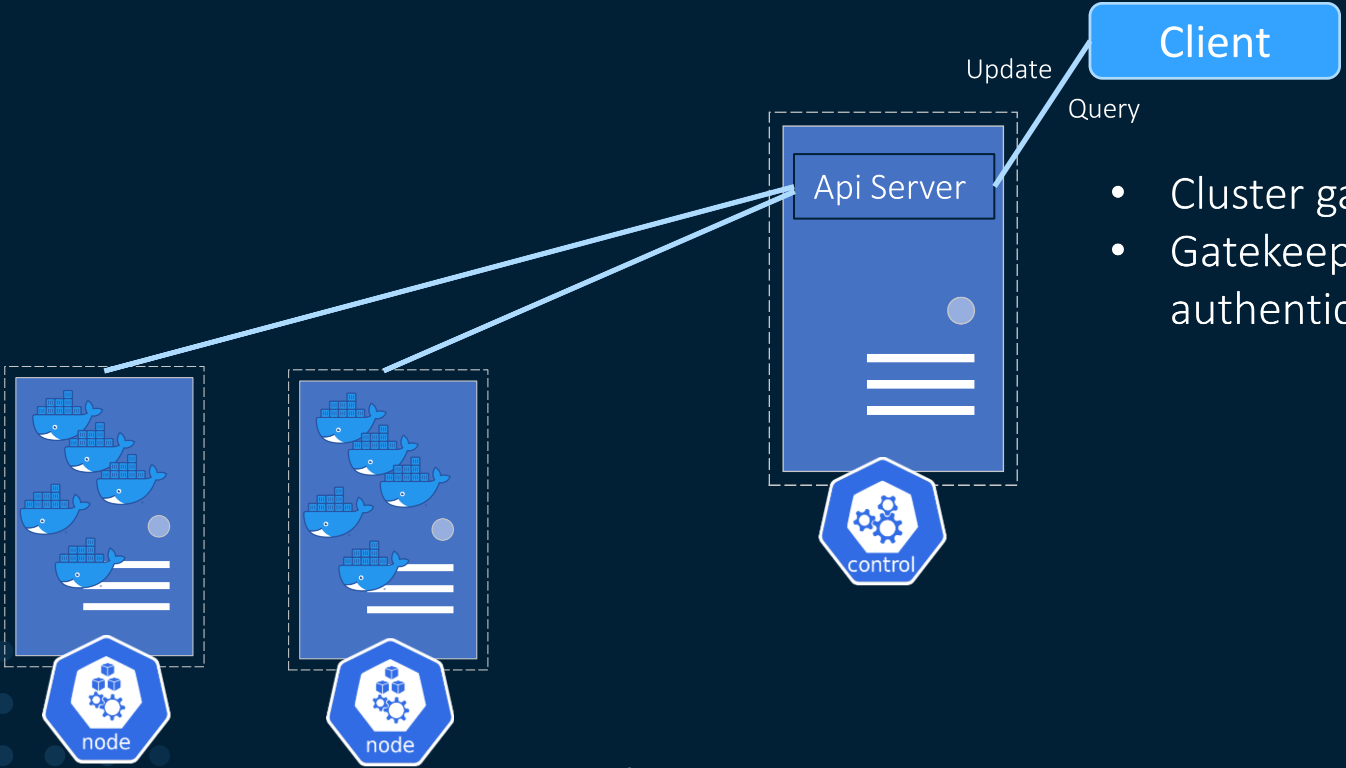
Interaction with the cluster

- 4 processes runs on every Master/Controller:



Kubernetes Architecture

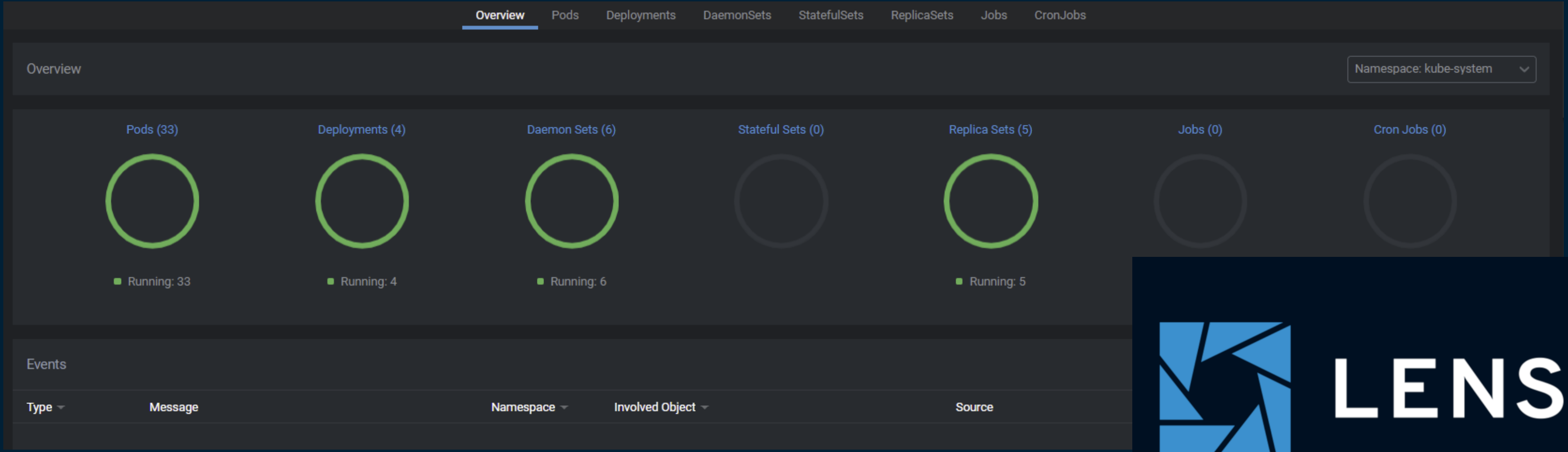
Interaction with the cluster



- Cluster gateway
- Gatekeeper for authentication ✓

Kubernetes Architecture

Interaction with the cluster



Overview Pods Deployments DaemonSets StatefulSets ReplicaSets Jobs CronJobs

Overview

Namespace: kube-system

Pods (33)	Deployments (4)	Daemon Sets (6)	Stateful Sets (0)	Replica Sets (5)	Jobs (0)	Cron Jobs (0)
🟩 Running: 33	🟩 Running: 4	🟩 Running: 6		🟩 Running: 5		

Events

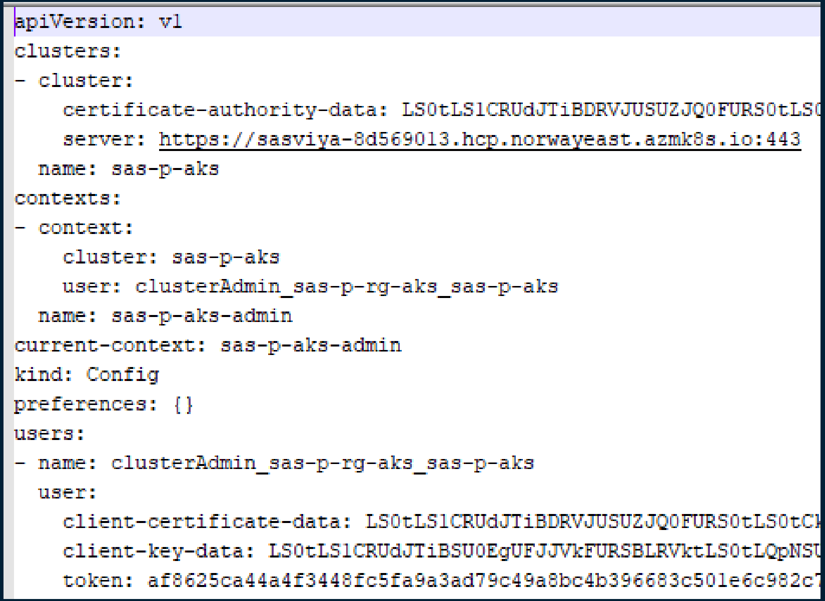
Type	Message	Namespace	Involved Object	Source

LENS

Kubernetes Architecture

Interaction with the cluster

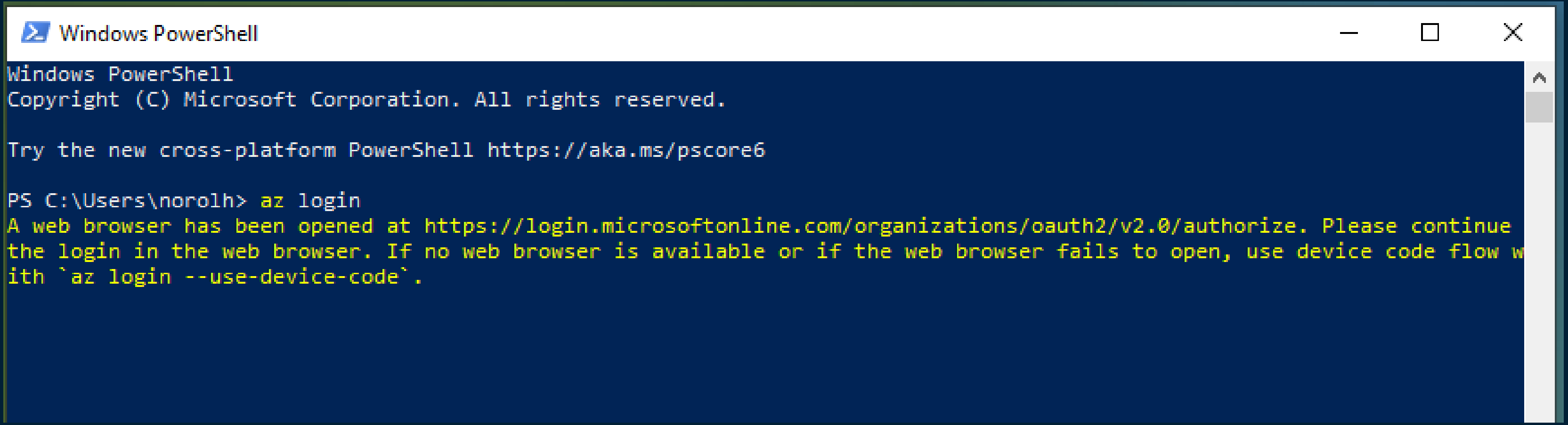
- Kubeconfig file – description on connection to K8s cluster – need this for lens



```
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0
    server: https://sasviya-8d569013.hcp.norwayeast.azmk8s.io:443
  name: sas-p-aks
contexts:
- context:
    cluster: sas-p-aks
    user: clusterAdmin_sas-p-rg-aks_sas-p-aks
  name: sas-p-aks-admin
current-context: sas-p-aks-admin
kind: Config
preferences: {}
users:
- name: clusterAdmin_sas-p-rg-aks_sas-p-aks
  user:
    client-certificate-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk
    client-key-data: LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQpNSU
    token: af8625ca44a4f3448fc5fa9a3ad79c49a8bc4b396683c501e6c982c7
```

Kubernetes Architecture

Interaction with the cluster



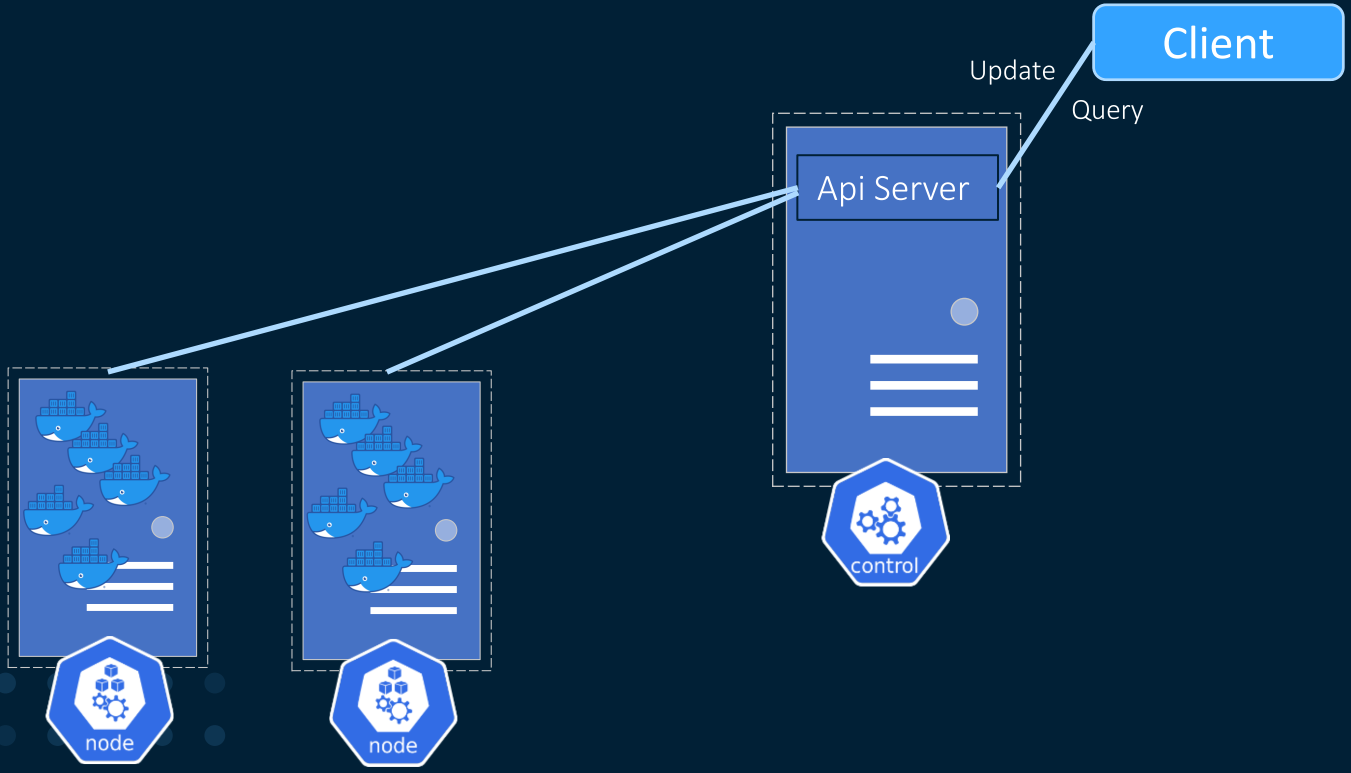
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\norolh> az login
A web browser has been opened at https://login.microsoftonline.com/organizations/oauth2/v2.0/authorize. Please continue
the login in the web browser. If no web browser is available or if the web browser fails to open, use device code flow w
ith `az login --use-device-code`.
```

Kubernetes Architecture

Interaction with the cluster



Client

Update

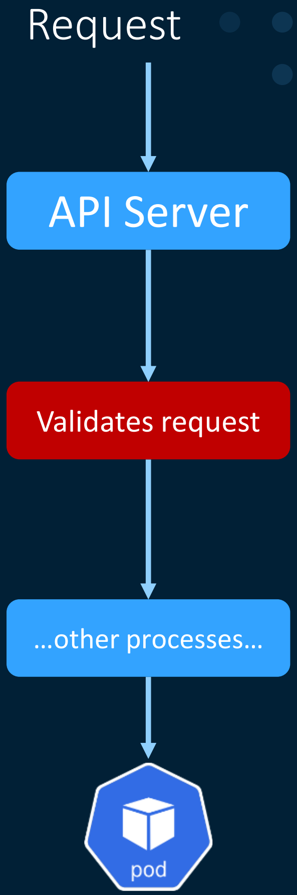
Query

Api Server

control

node

node



Request

API Server

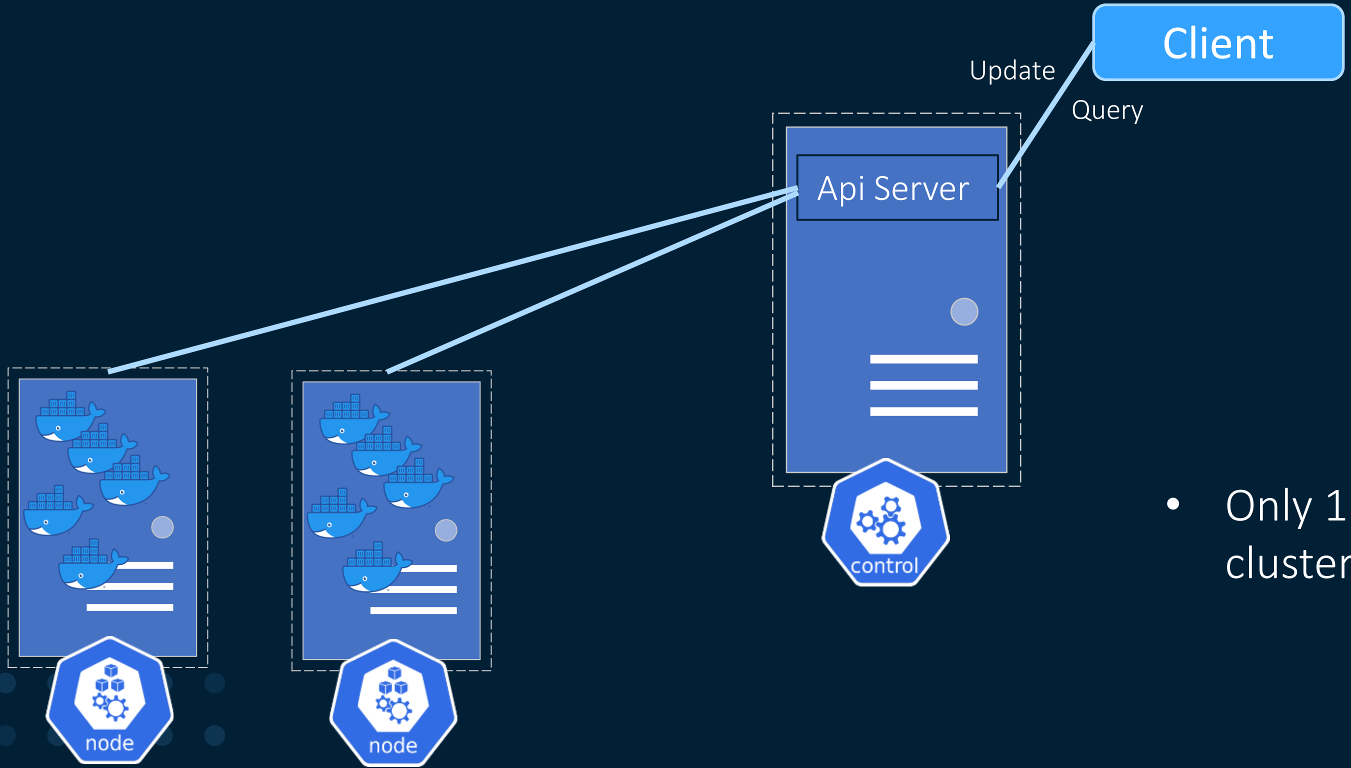
Validates request

...other processes...

pod

Kubernetes Architecture

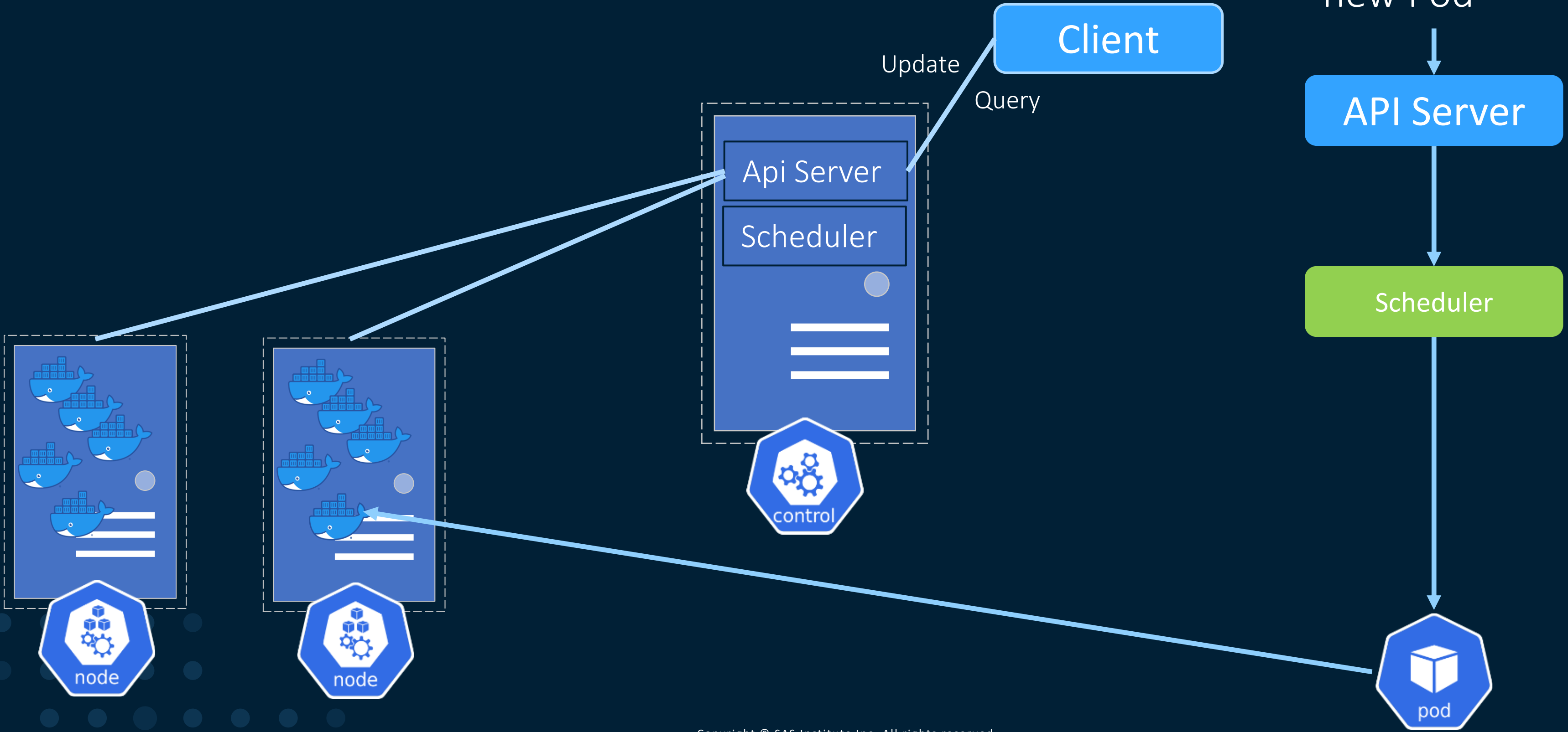
Interaction with the cluster



- Only 1 entrypoint into the cluster

Kubernetes Architecture

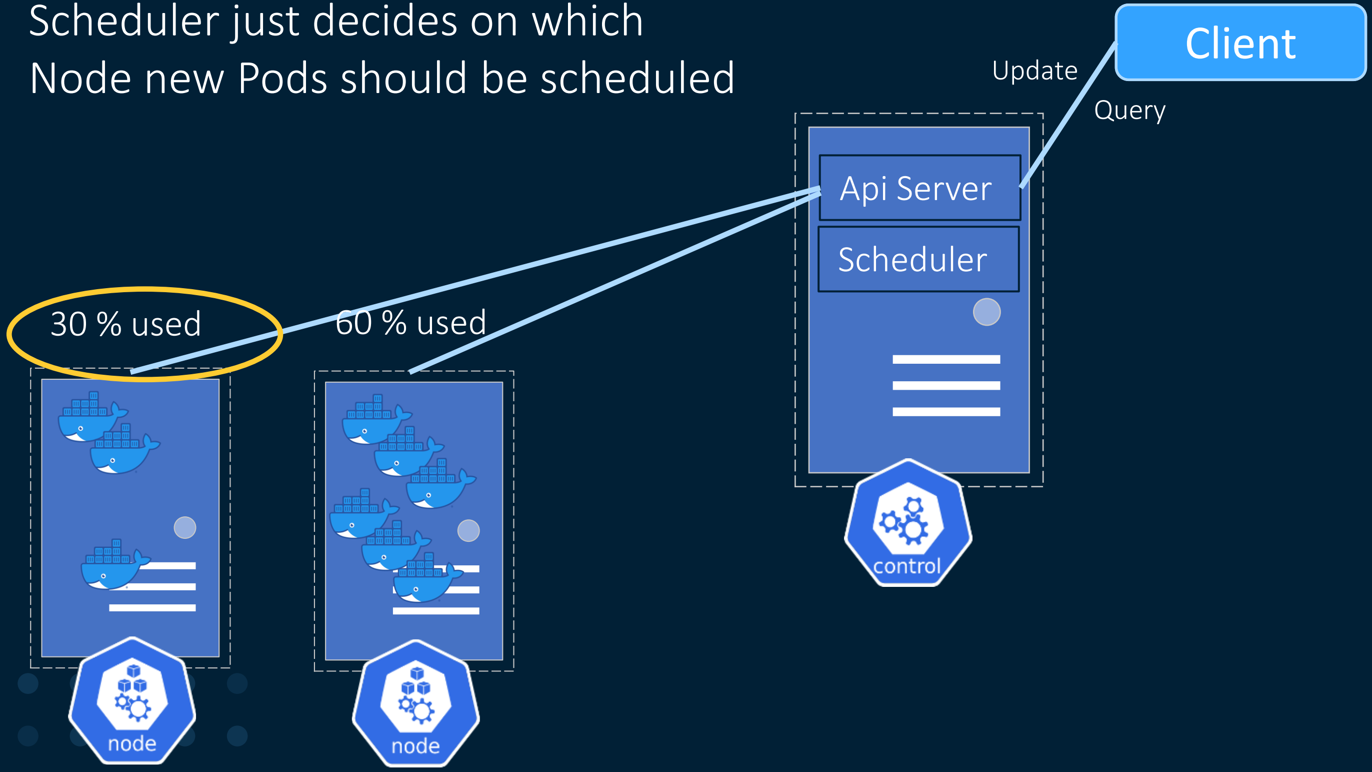
Interaction with the cluster

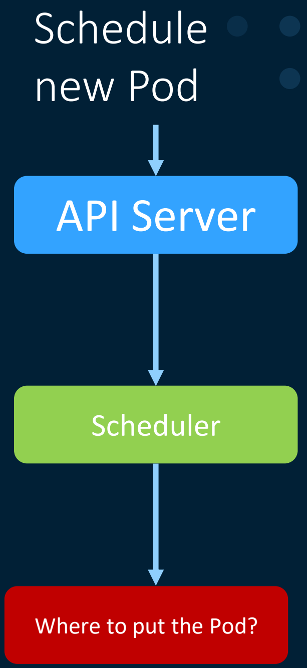


Kubernetes Architecture

Interaction with the cluster

Scheduler just decides on which Node new Pods should be scheduled

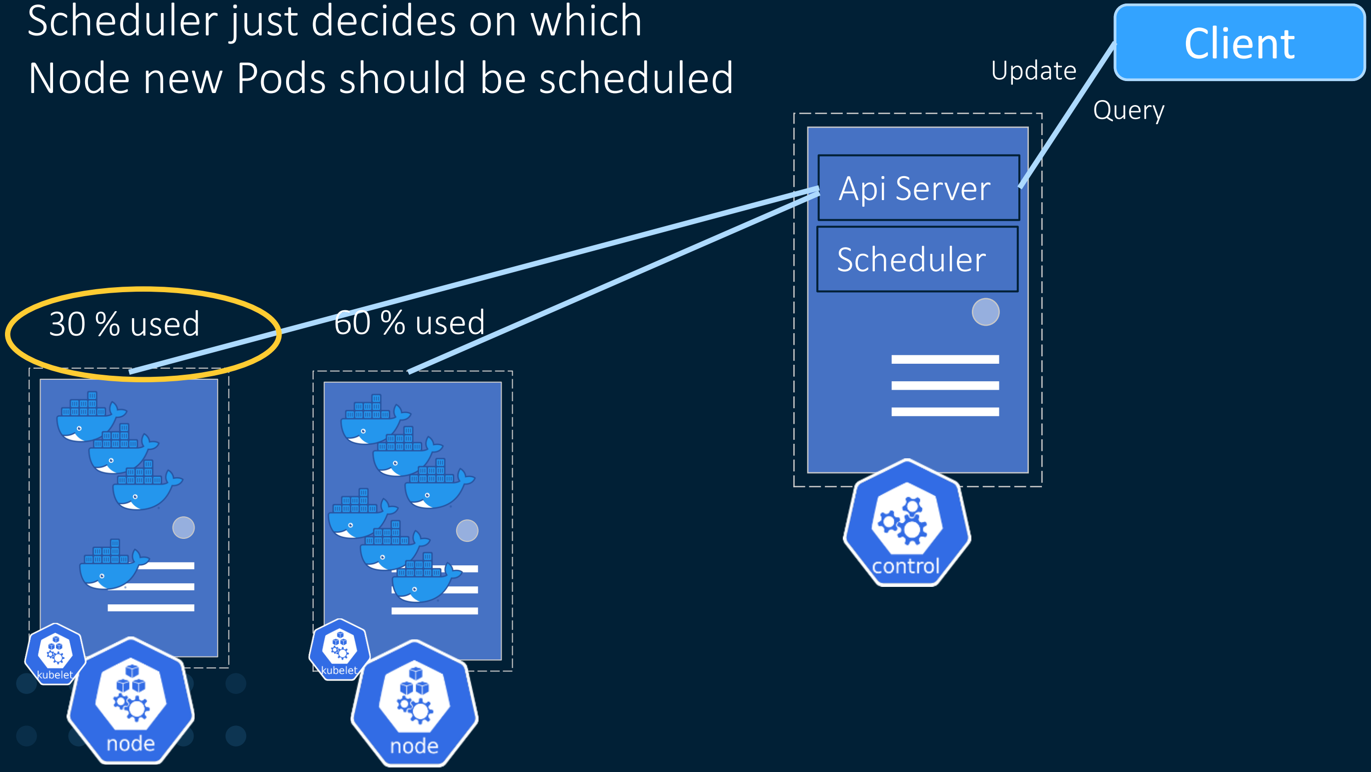


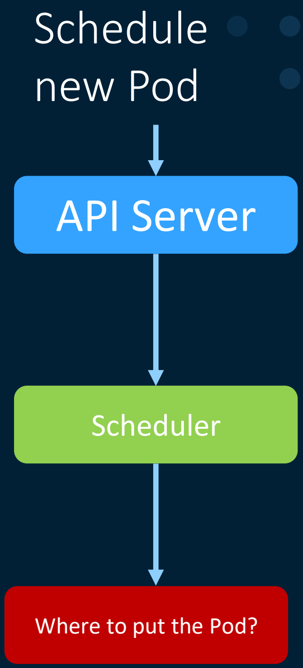


Kubernetes Architecture

Interaction with the cluster

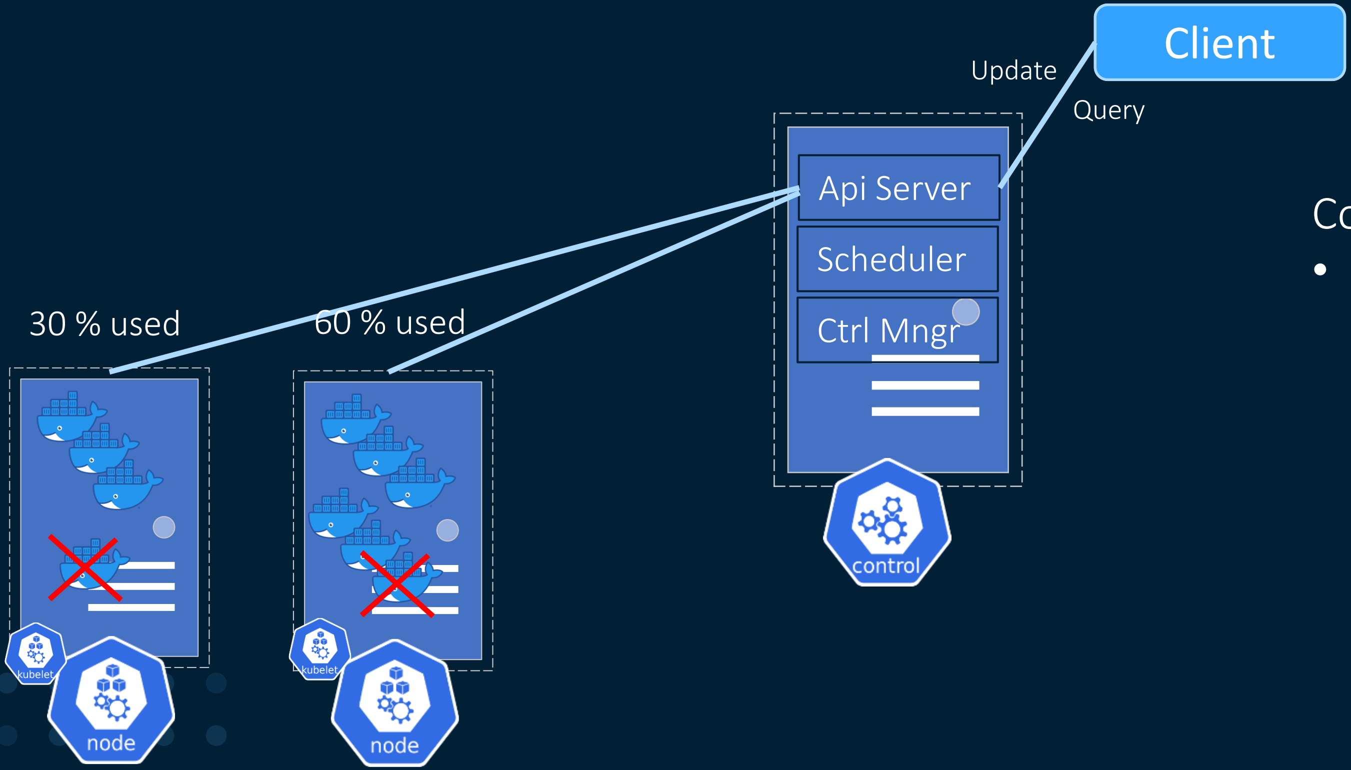
Scheduler just decides on which Node new Pods should be scheduled





Kubernetes Architecture

Interaction with the cluster

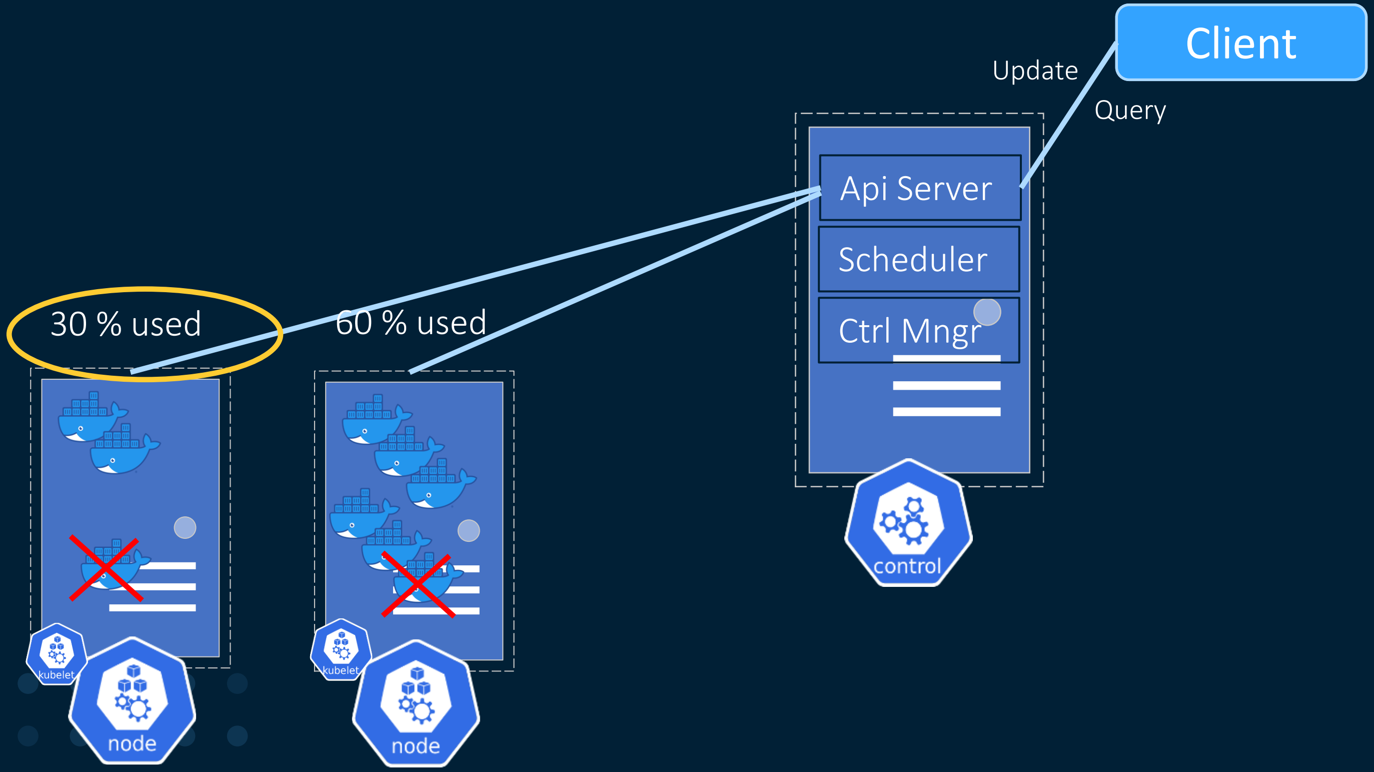


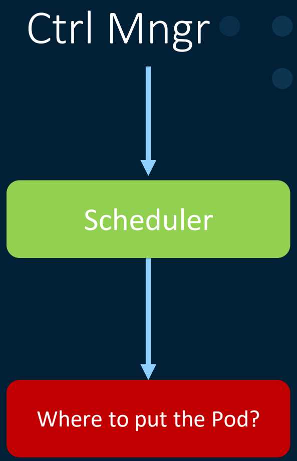
Control Manager – detects:

- Cluster State changes

Kubernetes Architecture

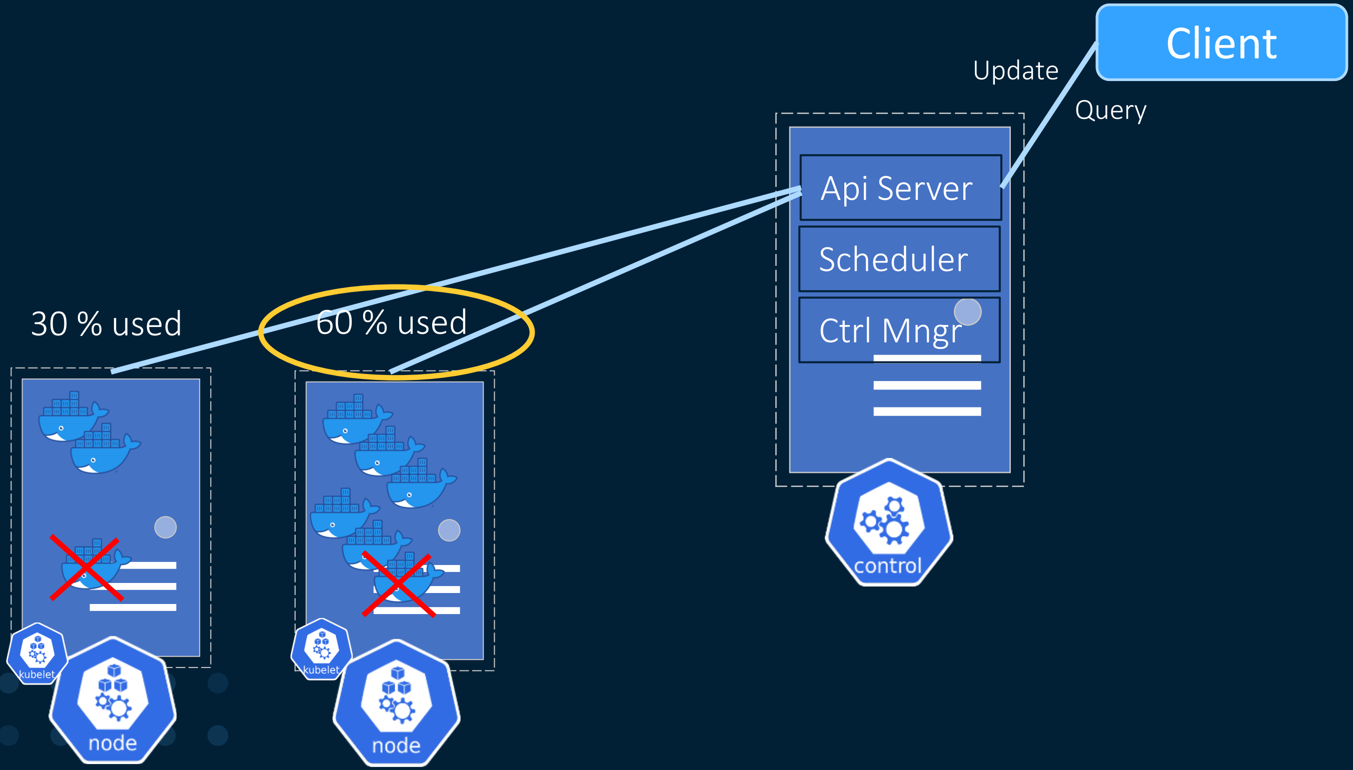
Interaction with the cluster

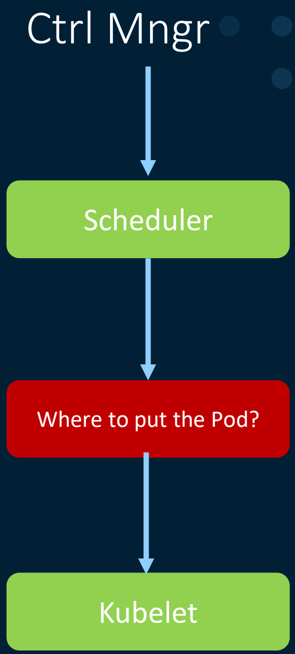




Kubernetes Architecture

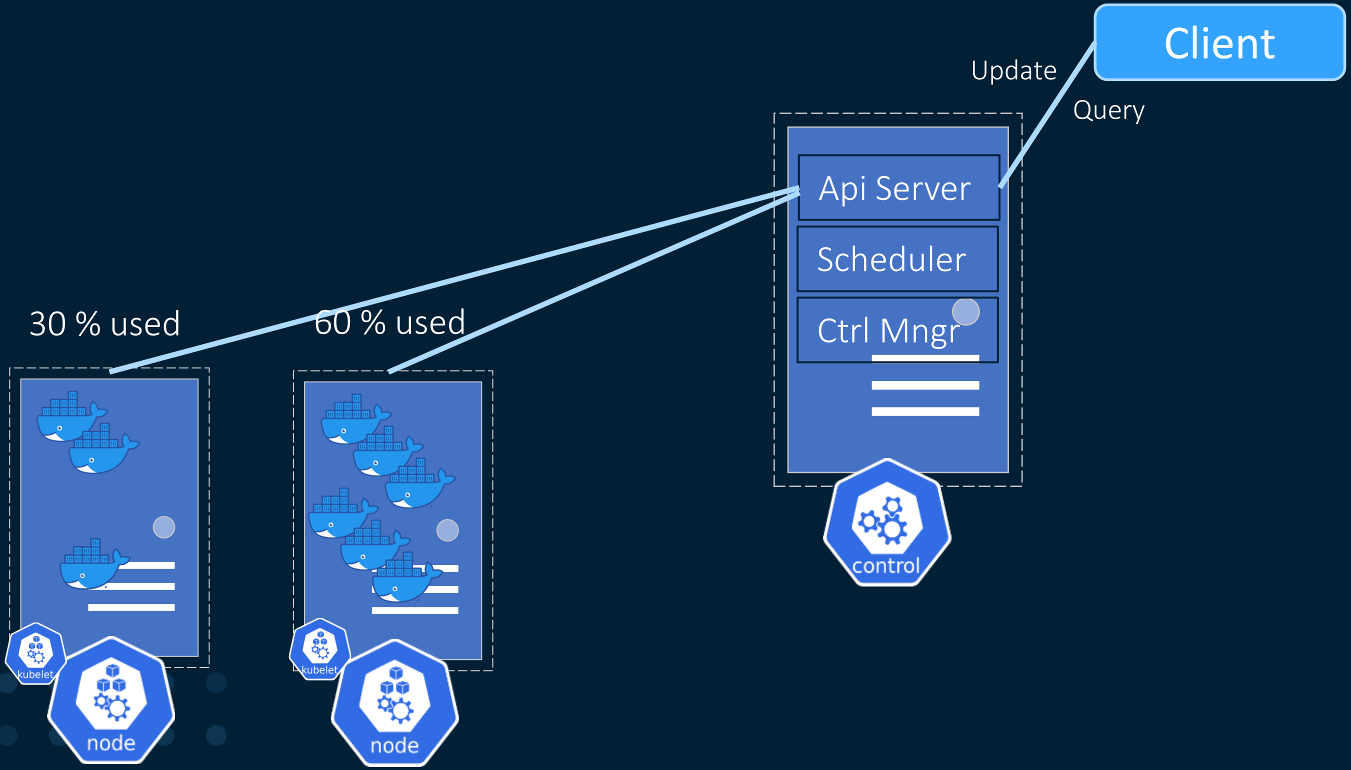
Interaction with the cluster

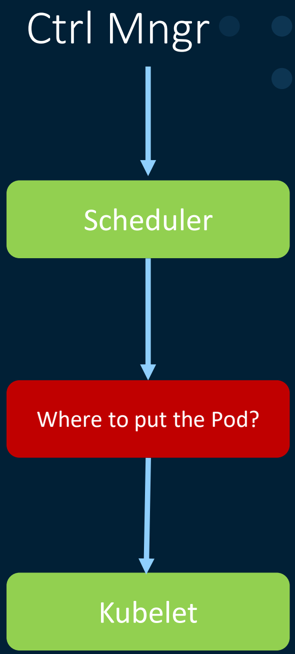




Kubernetes Architecture

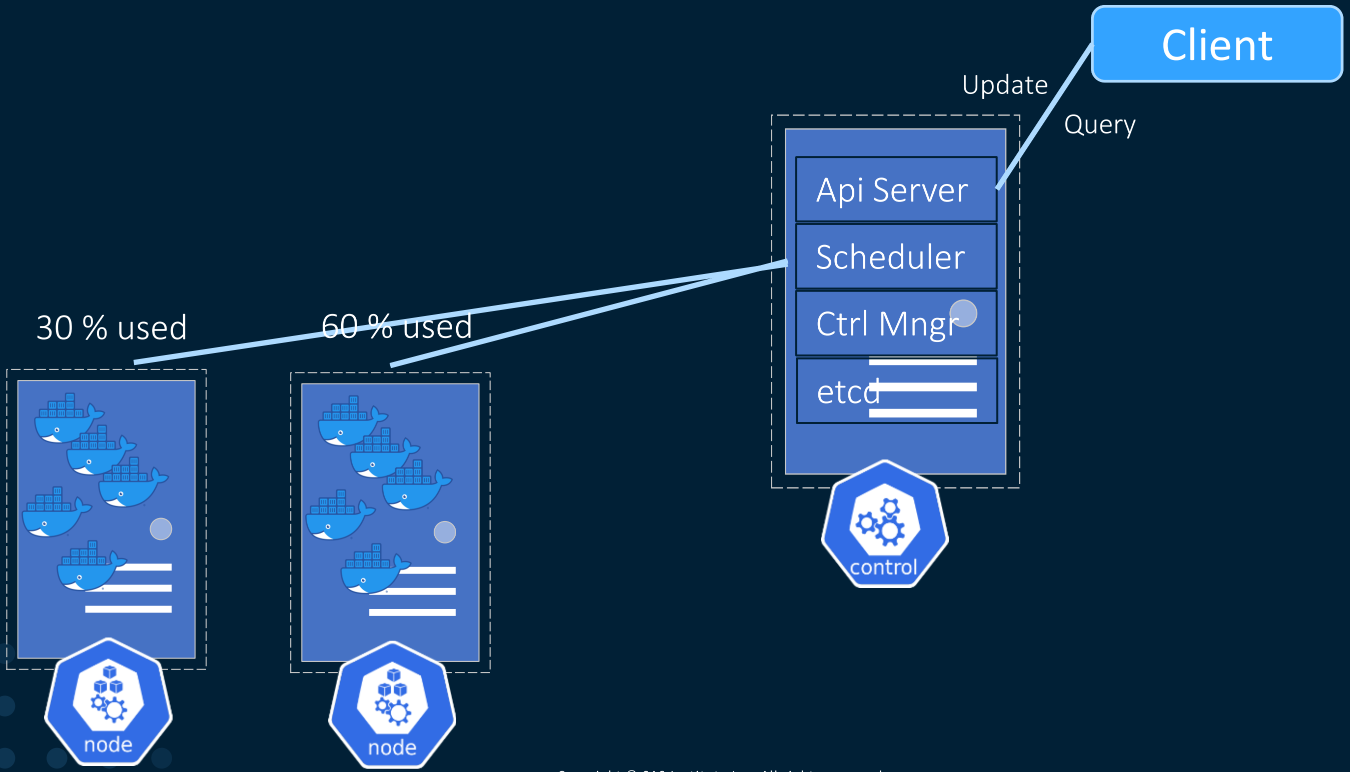
Interaction with the cluster





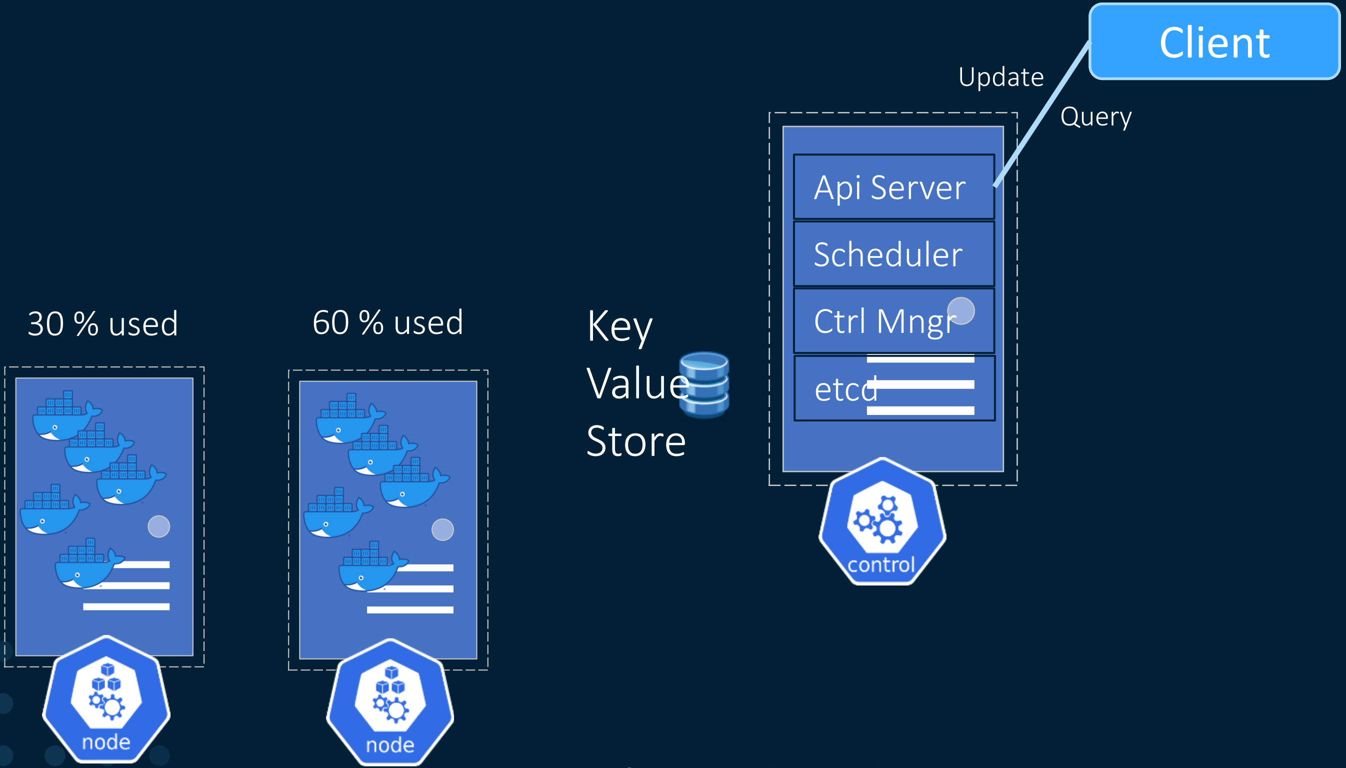
Kubernetes Architecture

Interaction with the cluster



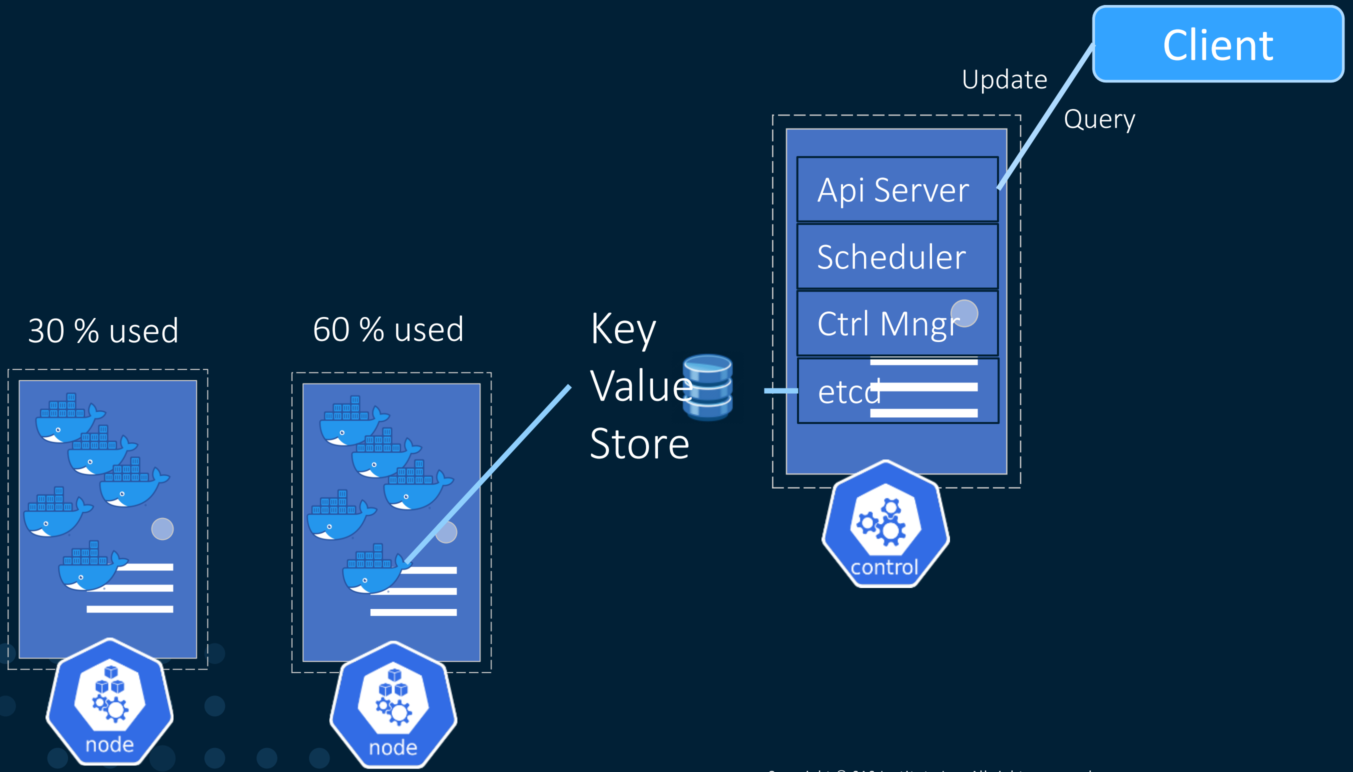
Kubernetes Architecture

Interaction with the cluster



Kubernetes Architecture

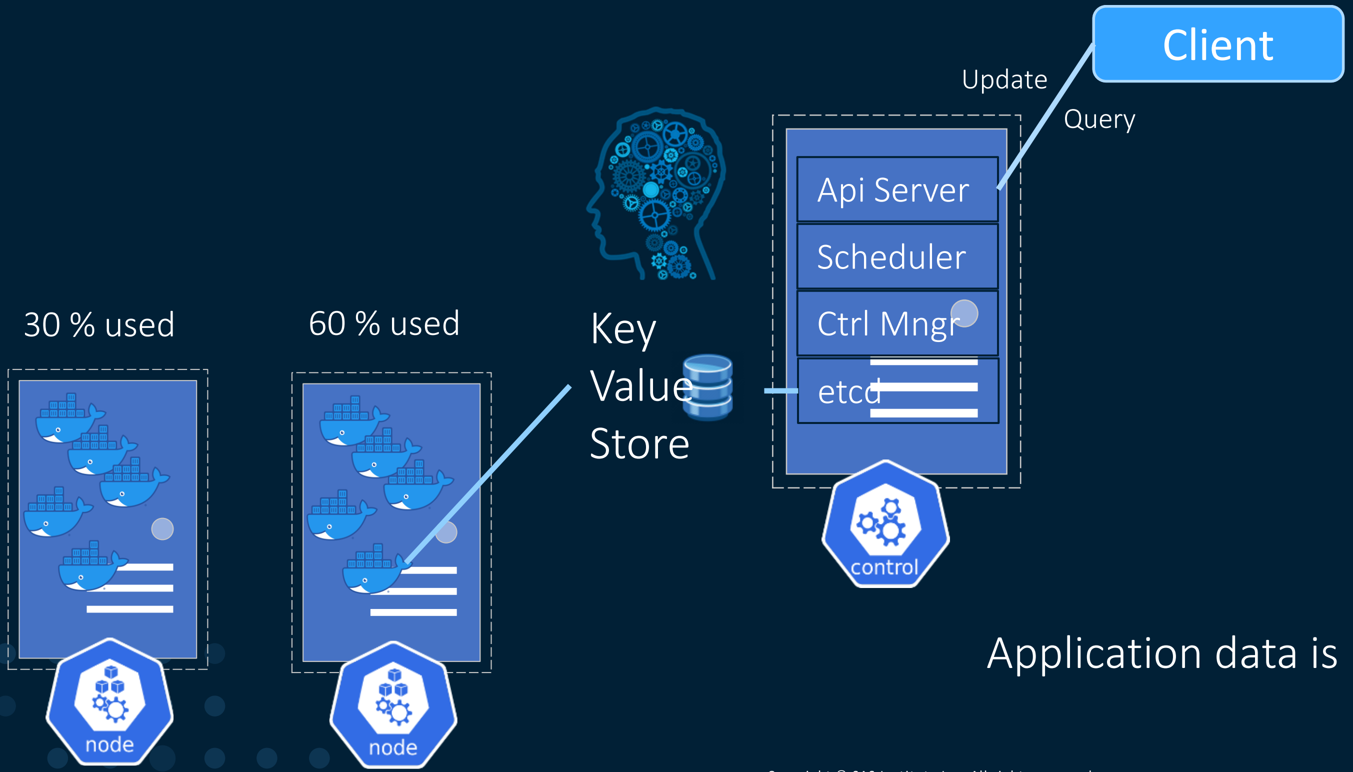
Interaction with the cluster



- Etcd is the cluster brain
- Cluster changes get stored in the key value store

Kubernetes Architecture

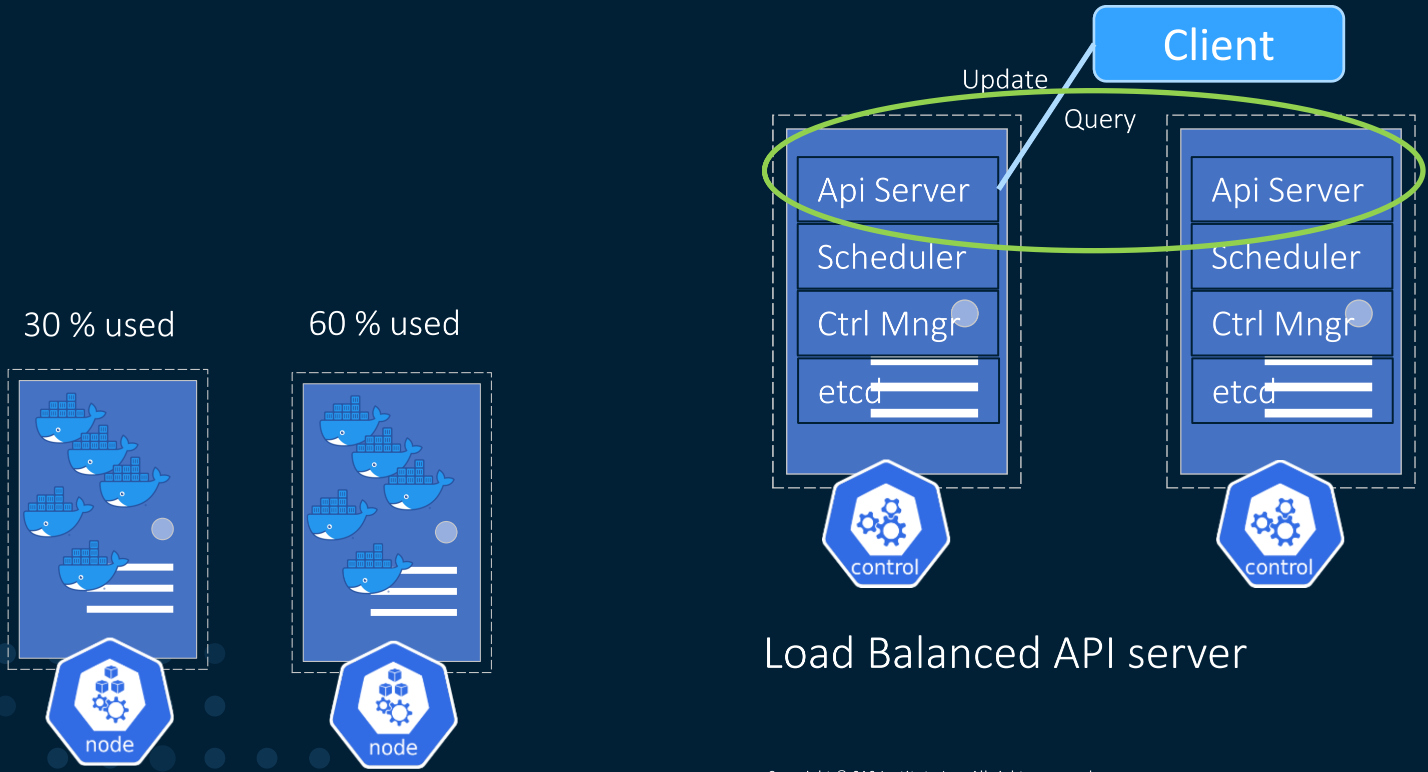
Interaction with the cluster



Application data is NOT stored in etcd

Kubernetes Architecture

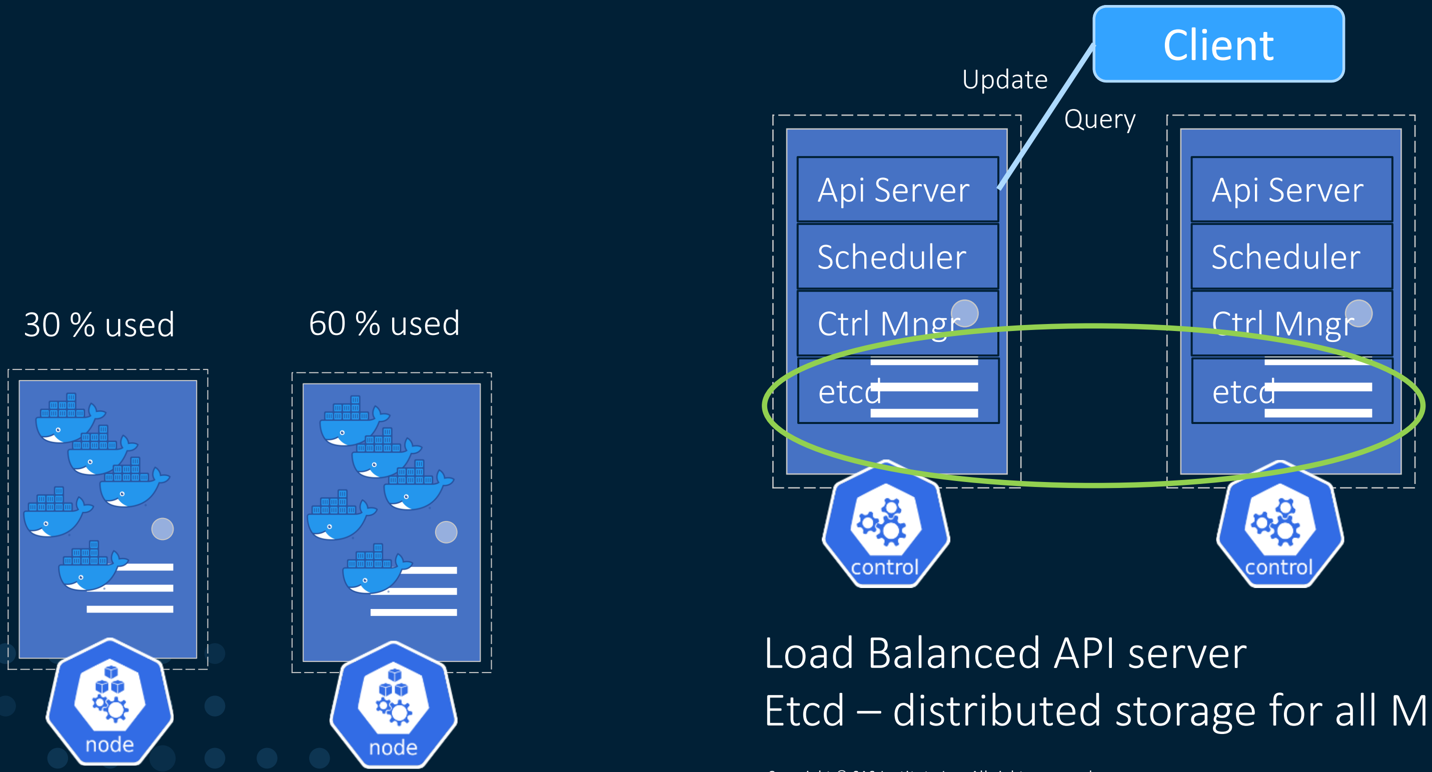
Interaction with the cluster



Load Balanced API server

Kubernetes Architecture

Interaction with the cluster



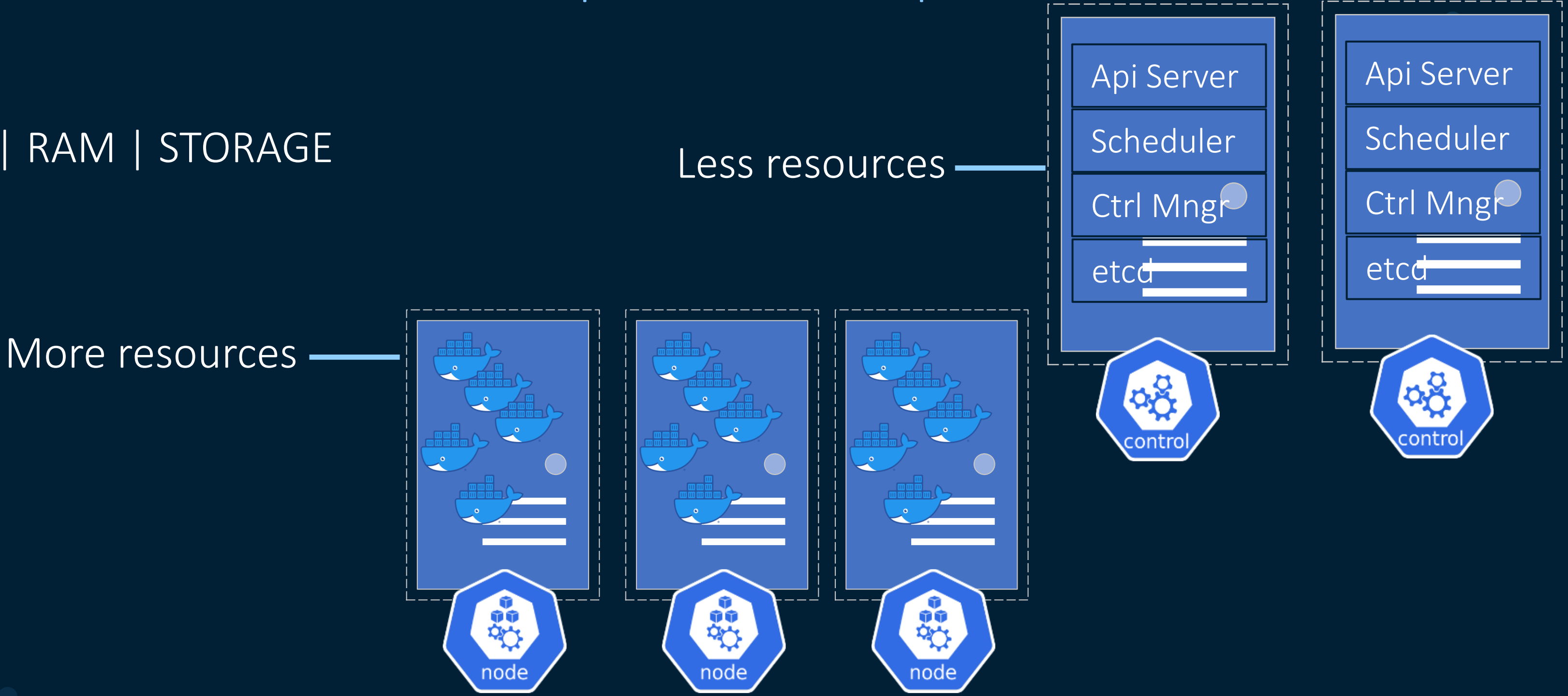
Load Balanced API server

Etcd – distributed storage for all Master nodes

Kubernetes Architecture

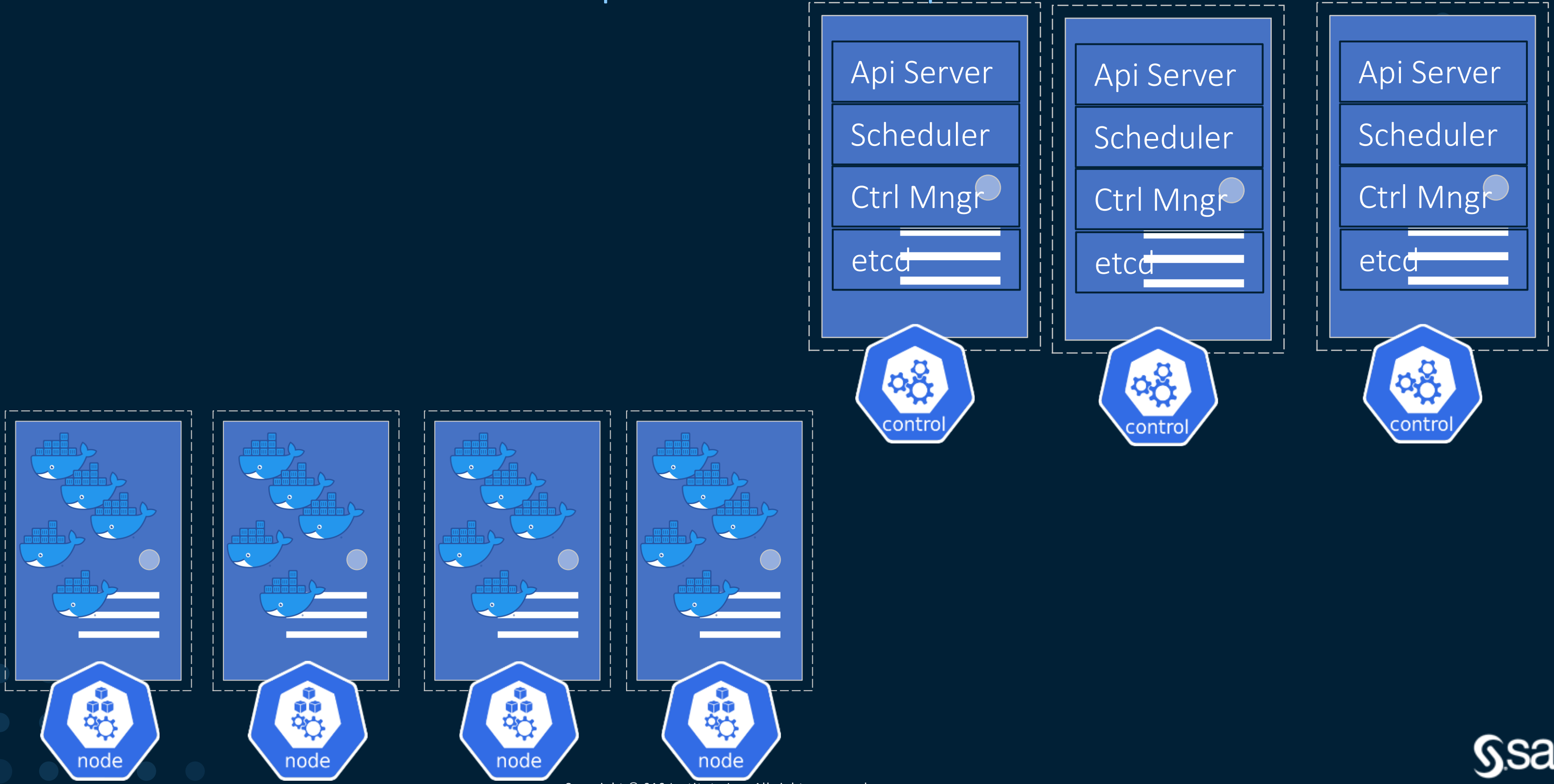
Example Cluster setup

CPU | RAM | STORAGE



Kubernetes Architecture

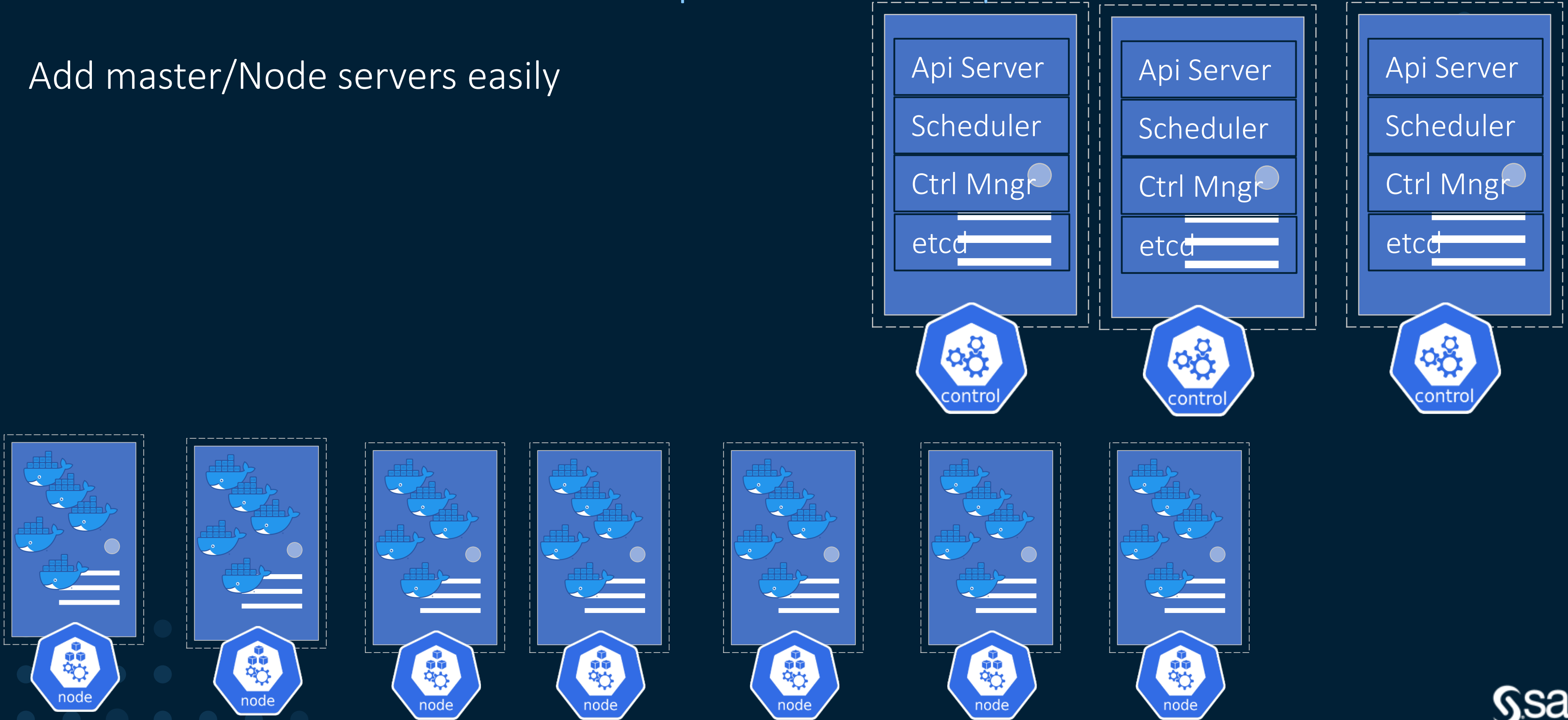
Example Cluster setup



Kubernetes Architecture

Example Cluster setup

Add master/Node servers easily



The end part 2
Pause for 10 min.

sas.com