

Introduction to Kubernetes

Agenda

1. Why Kubernetes?
2. What is Kubernetes?
3. The K8s Architecture
4. Creating a Kubernetes Cluster with K3d
5. Kubernetes Resource Management with kubectl
6. Q & A
7. The Main Kubernetes Objects
8. Q & A



Supporting Material

The screenshot shows a GitHub repository page for 'SUSE-Rancher-Community/intro-to-kubernetes'. The repository is private and has 3 watchers, 0 stars, and 0 forks. The main navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Security, Insights, and Settings. The repository is on the 'master' branch with 1 branch and 0 tags. A commit by 'LukeMwila' is shown, titled 'docs: modified the README', with a file list including 'manifests', '.gitignore', 'README.md', 'Vagrantfile', 'cluster.yml', 'node_script.sh', and 'rke.png'. The 'README.md' file is selected, showing the title 'Introduction to Kubernetes' and a description: 'This repository contains the source code for creating a local Kubernetes cluster setup using Vagrant and Rancher Kubernetes Engine (RKE). It also contains a set of manifest files to deploy a basic application to your local K8s cluster.' Below the description is a section titled 'Requirements/Prerequisites' with a bulleted list: 'Rancher Kubernetes Engine (RKE)', 'Virtual Box', and 'Vagrant'. On the right side, there are sections for 'About' (Introduction to Kubernetes), 'Releases' (No releases published), 'Packages' (No packages published), and 'Languages' (Shell 100.0%).

SUSE-Rancher-Community / intro-to-kubernetes (Private)

Watch 3 Star 0 Fork 0

<> Code Issues Pull requests Actions Projects Security Insights Settings

master 1 branch 0 tags

Go to file Add file Code

LukeMwila docs: modified the README fabe270 1 minute ago 2 commits

manifests	other: initial	6 minutes ago
.gitignore	other: initial	6 minutes ago
README.md	docs: modified the README	1 minute ago
Vagrantfile	other: initial	6 minutes ago
cluster.yml	other: initial	6 minutes ago
node_script.sh	other: initial	6 minutes ago
rke.png	other: initial	6 minutes ago

README.md

Introduction to Kubernetes

This repository contains the source code for creating a local Kubernetes cluster setup using Vagrant and Rancher Kubernetes Engine (RKE). It also contains a set of manifest files to deploy a basic application to your local K8s cluster.

Requirements/Prerequisites

- Rancher Kubernetes Engine (RKE)
- Virtual Box
- Vagrant

About Introduction to Kubernetes

Readme

Releases

No releases published
[Create a new release](#)

Packages

No packages published
[Publish your first package](#)

Languages

Shell 100.0%



Why Kubernetes?



Benefits of Containers



Container images have a faster start up

New container image snapshots are much faster

Container images are smaller

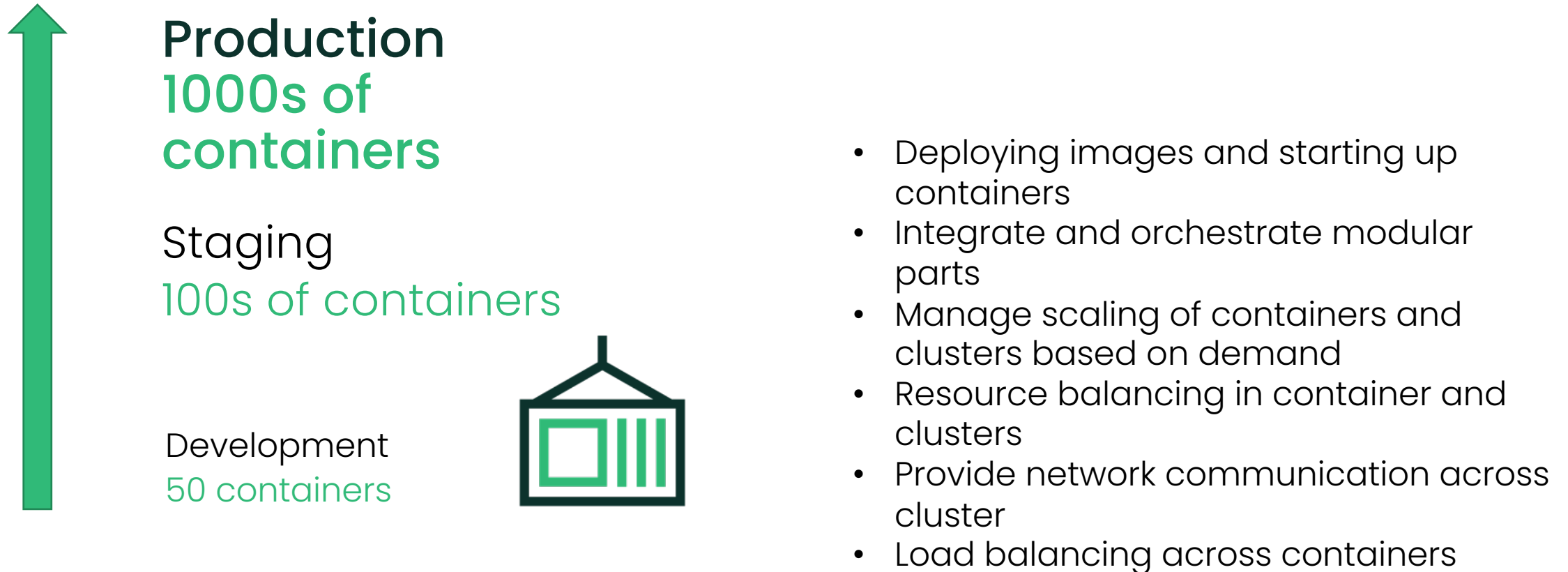
Containers are lighter

Containers can run anywhere once built

Containers have better resource utilization



Managing Containers



What is Kubernetes?



Container Orchestration with Kubernetes

Kubernetes is an open-source container orchestration tool or platform.

It is built to manage workloads that consist of 100s or 1000s of containers.

Niantic's gaming app, Pokemon Go, is powered by Kubernetes and scaled to manage traffic from millions of users across the globe.



kubernetes



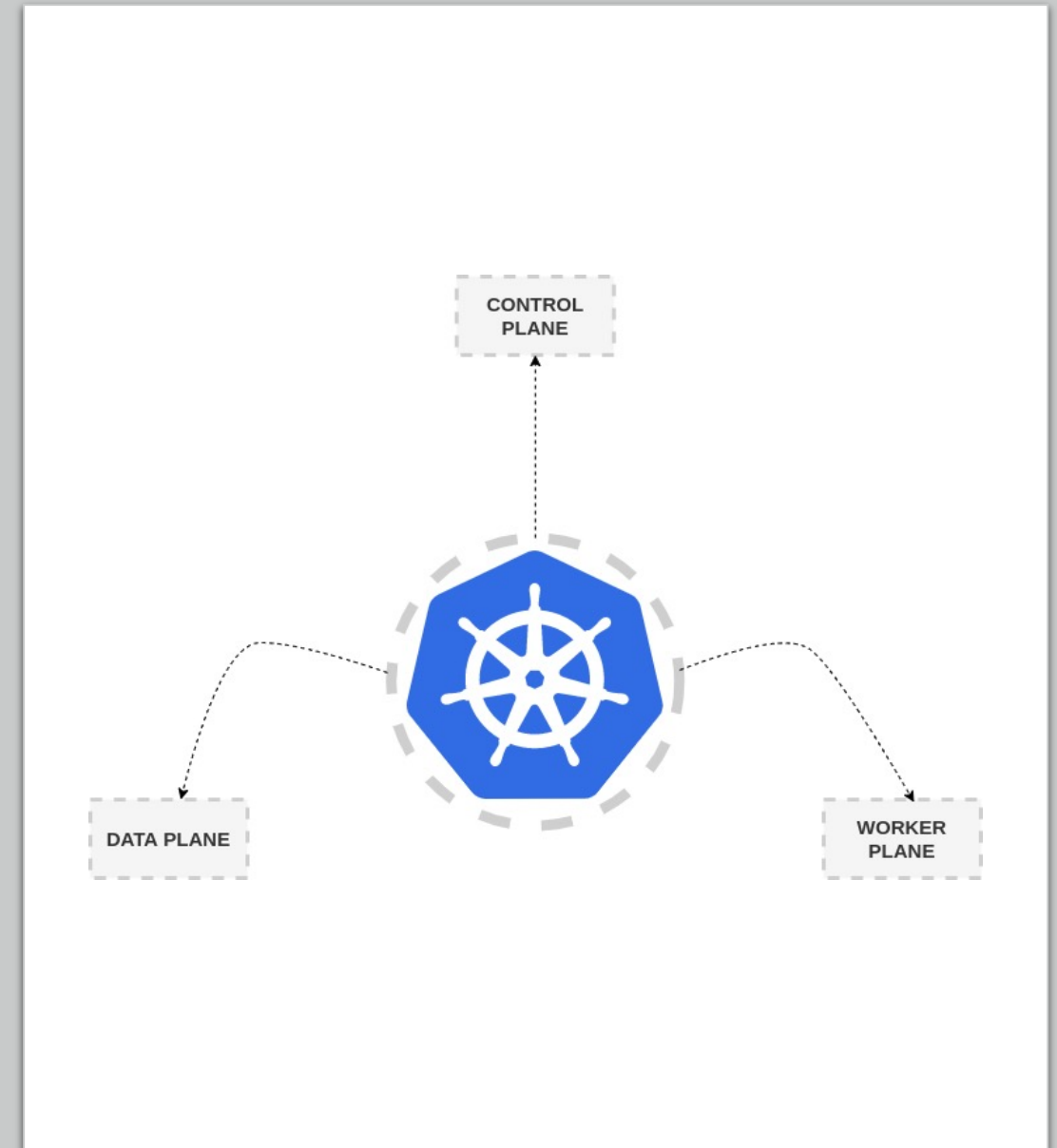
The K8s Architecture



The 3 Planes of K8s

- The Control Plane – This is the brain.
- The Data Plane – This is the memory or storage.
- The Worker Plane – This is the body responsible for running the application.

These 3 planes makeup what is known as a Kubernetes cluster. A cluster can be on one device or distributed across devices.



Creating a K8s Cluster

Setting up a Kubernetes cluster is complex and time consuming i.e. KH Kubernetes the hard way.

In most cases, Kubernetes administrators work with CNCF-certified Kubernetes distributions, hosted clusters or installers.

K8s distros: Rancher Kubernetes, RKE Government, K3s, K3d

K8s hosted clusters: Amazon EKS, Google Cloud GKE, Microsoft Azure AKS

K8s installers: RKE



Creating a Kubernetes Cluster with K3d

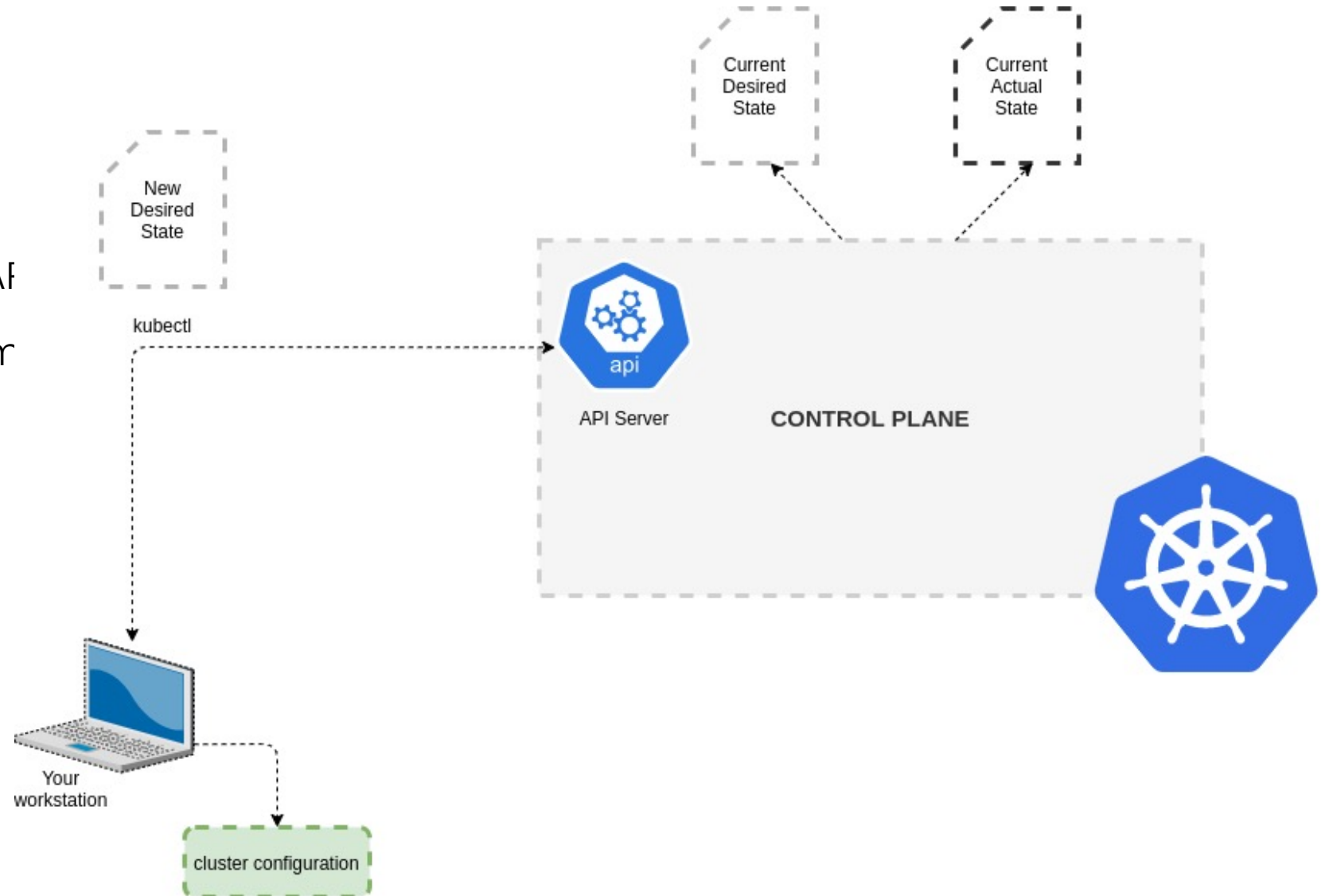


Kubernetes Resource Management with kubectl



Communication with K8s Cluster

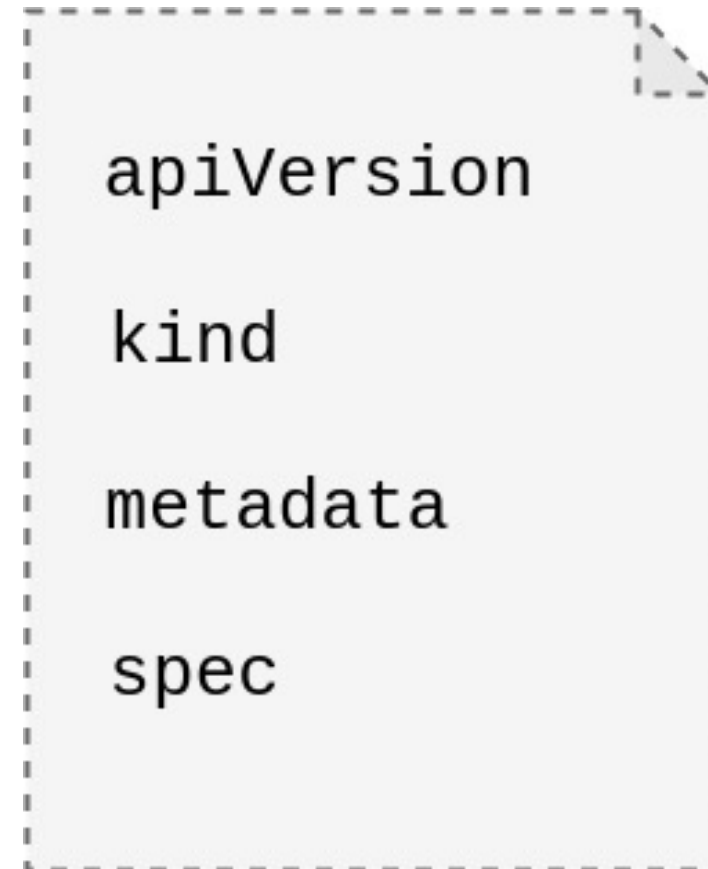
- Using kubectl CLI tool
- Client to server model
- All CRUD operations via API
- Kubernetes declarative model



Declaring State in K8s with Manifests

These are YAML configuration files. Most K8s objects consist of 4 top-level required fields:

- `apiVersion` – This field defines the API version number that the Kubernetes object belongs to.
- `kind` – This field specifies the type of Kubernetes object to be created.
- `metadata` – This field contains data that describes the object being created (i.e. name).
- `spec` – This field details the container configuration.

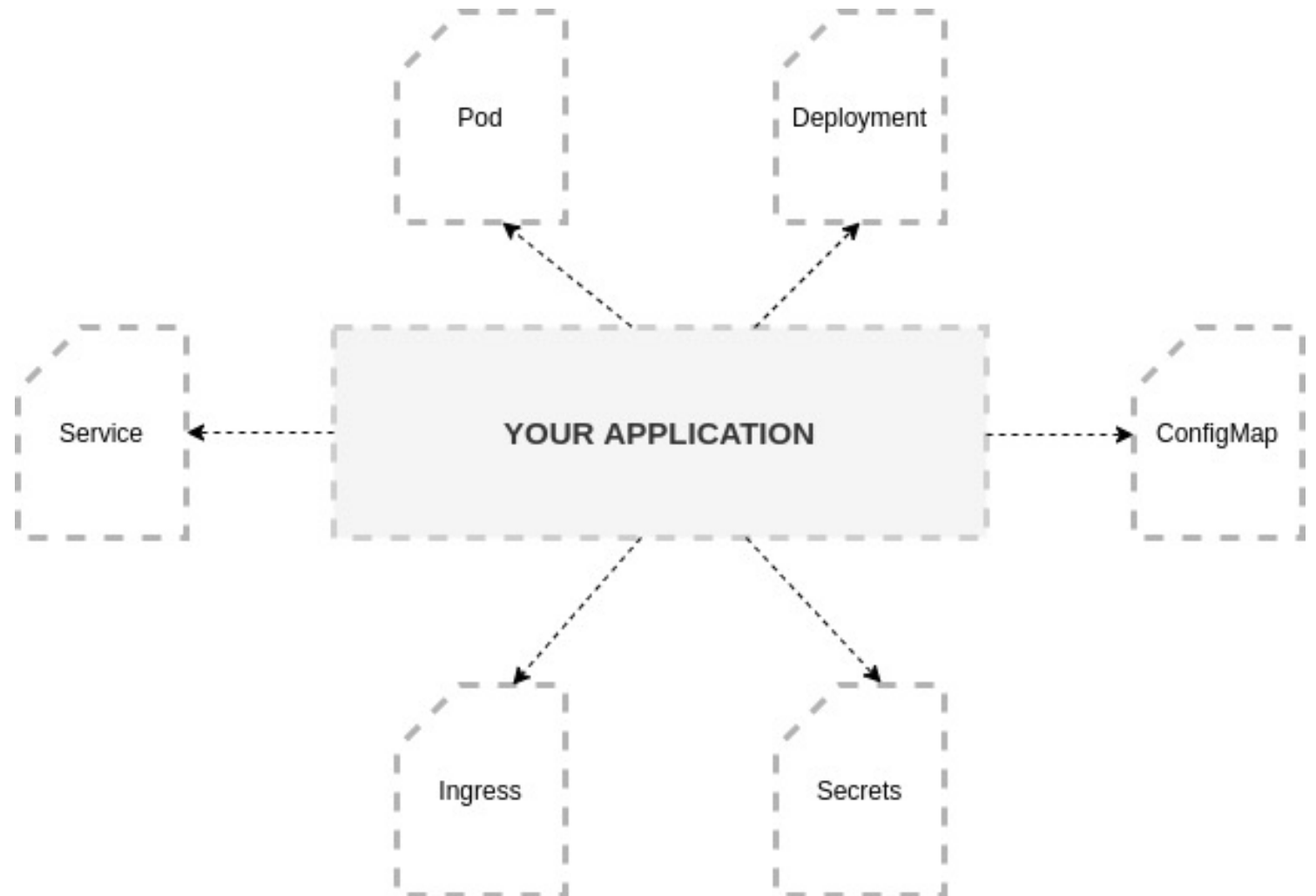


The Main Kubernetes Objects



Deploying a Kubernetes Application

- Pod
- Deployments
- Services
- Ingresses
- ConfigMaps
- Secrets



Pods

Pods are the smallest deployable artefacts in a Kubernetes cluster.

You can think of them as wrappers for one or more closely linked containers.

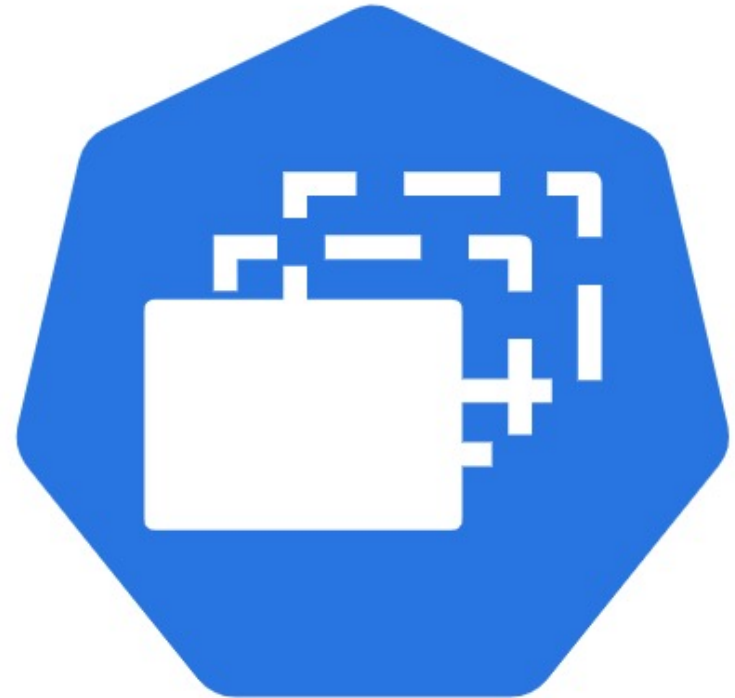


ReplicaSets

ReplicaSets are actually distinct Kubernetes objects but are typically not created with manifest files. Instead, they are created as part of Deployments.

In practice, you will want multiple replicas of a container running at a particular time, as opposed to just one. The main reasons for this are:

- Fault tolerance – the more instances you have of your container, the more fault-tolerant your application will be.
- Application scaling – the more instances you have of your container, the more requests that can be handled.



Deployments

Deployments are a special type of Kubernetes object categorically referred to as a controller.

Kubernetes uses controllers to ensure that the cluster's existing state is always updated to match the desired state.

Deployments are used to manage Pod releases. They manage Pods in the following ways:

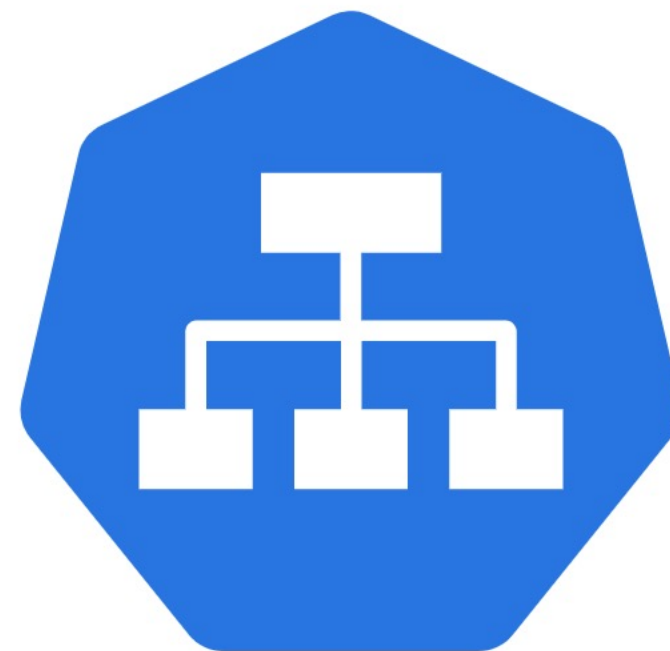
- Scaling of Pods – Are the correct number of Pods running?
- Monitoring the state of Pods – Are the Pods running or are they failing?
- Updates to Pods – Is there a new version that needs to be rolled out?



Services

We need a solution that will discover Pods and keep track of their continuously changing IPs. That solution is the Service object. Services balance traffic across Pods in the cluster based on a label attached to the Pods.

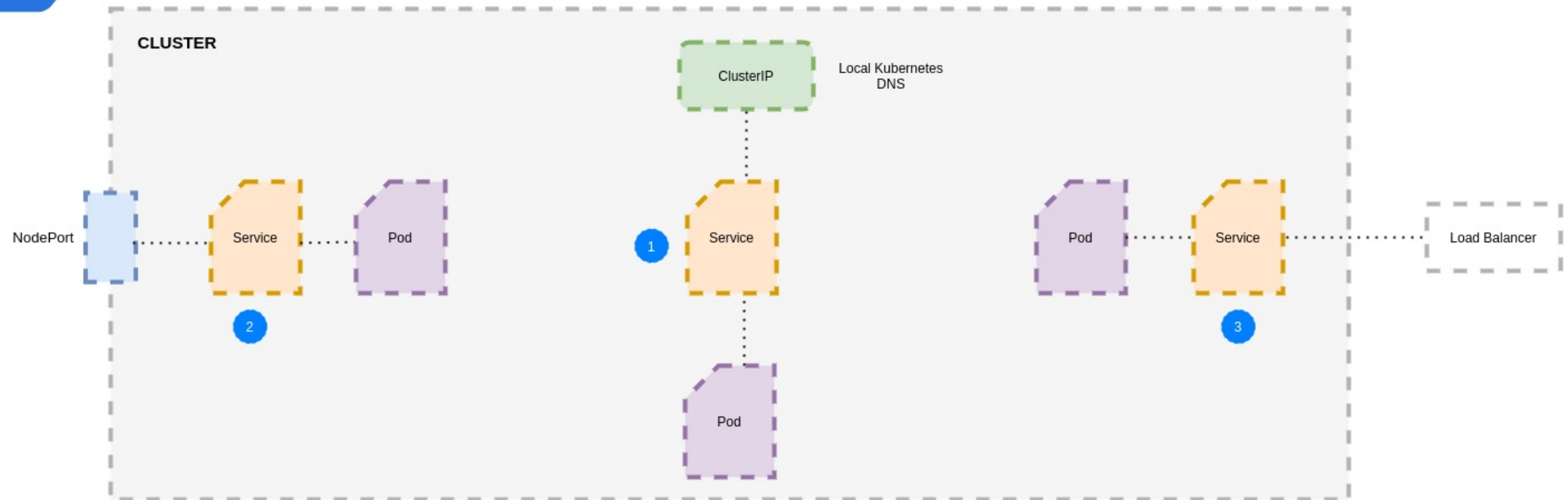
- ClusterIP – Only accessible from within the cluster.
- NodePort – Gets a cluster-wide port and is also accessible.
- LoadBalancer – Integrate with the public cloud to create a load balancer in a cloud environment.



Services



- 1 ClusterIP Service
- 2 NodePort Service
- 3 LoadBalancer Service



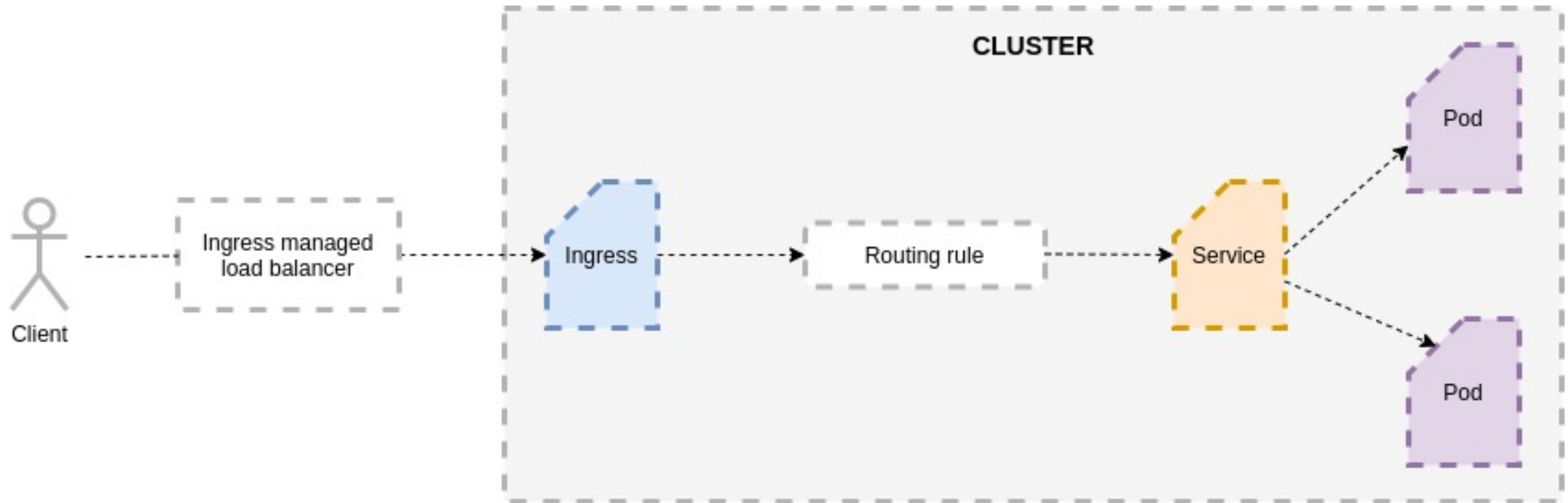
Ingresses

When it comes to getting external network traffic to and from your application, Services are only sufficient to a certain degree. They have issues that Ingress solves.

- HTTP Traffic
- Routing
- Security
- Cost



Ingresses



ConfigMaps & Secrets

In K8s, you may want to pass dynamic values to your applications at runtime to control how they behave. This is known as application configuration.

There are two primary ways to store configuration data in Kubernetes: ConfigMaps and Secrets

You can pass ConfigMap and Secret data to your containers as environment variables. These variables will be visible to your container process at runtime.





Thank You

© 2020 SUSE LLC. All Rights Reserved. SUSE and the SUSE logo are registered trademarks of SUSE LLC in the United States and other countries. All third-party trademarks are the property of their respective owners.