

KUBERNETES

INTRODUCTION TO DEVOPS

Anushka Jain 1MS19CS022
M Shashanka 1MS19CS063
Muskan Bajaj 1MS19CS076

INTRODUCTION

- Kubernetes is an open source container orchestration engine for automating deployment, scaling, and management of containerized applications.
- It groups containers that make up an application into logical units for easy management and discovery.

Problems with manual deployment

Manual deployment of Containers is hard to maintain, error-prone and annoying

(even beyond security and configuration concerns!)

Containers might
crash / go down and
need to be replaced



We might need more
container instances
upon traffic spikes



Incoming traffic
should be distributed
equally



Services Like AWS ECS Can Help!

Manual deployment of Containers is hard to maintain, error-prone and annoying

(even beyond security and configuration concerns!)



Containers might crash / go down and need to be replaced

Container health checks + automatic re-deployment



We might need more container instances upon traffic spikes

Autoscaling



Incoming traffic should be distributed equally

Load balancer

But That Locks Us In!

Using a specific cloud service locks us into that service

Of course, you might be fine with sticking to one provider though!

You need to learn about the specifics, services and config options of another provider if you want to switch

Just knowing Docker isn't enough!

Kubernetes To The Rescue



Kubernetes

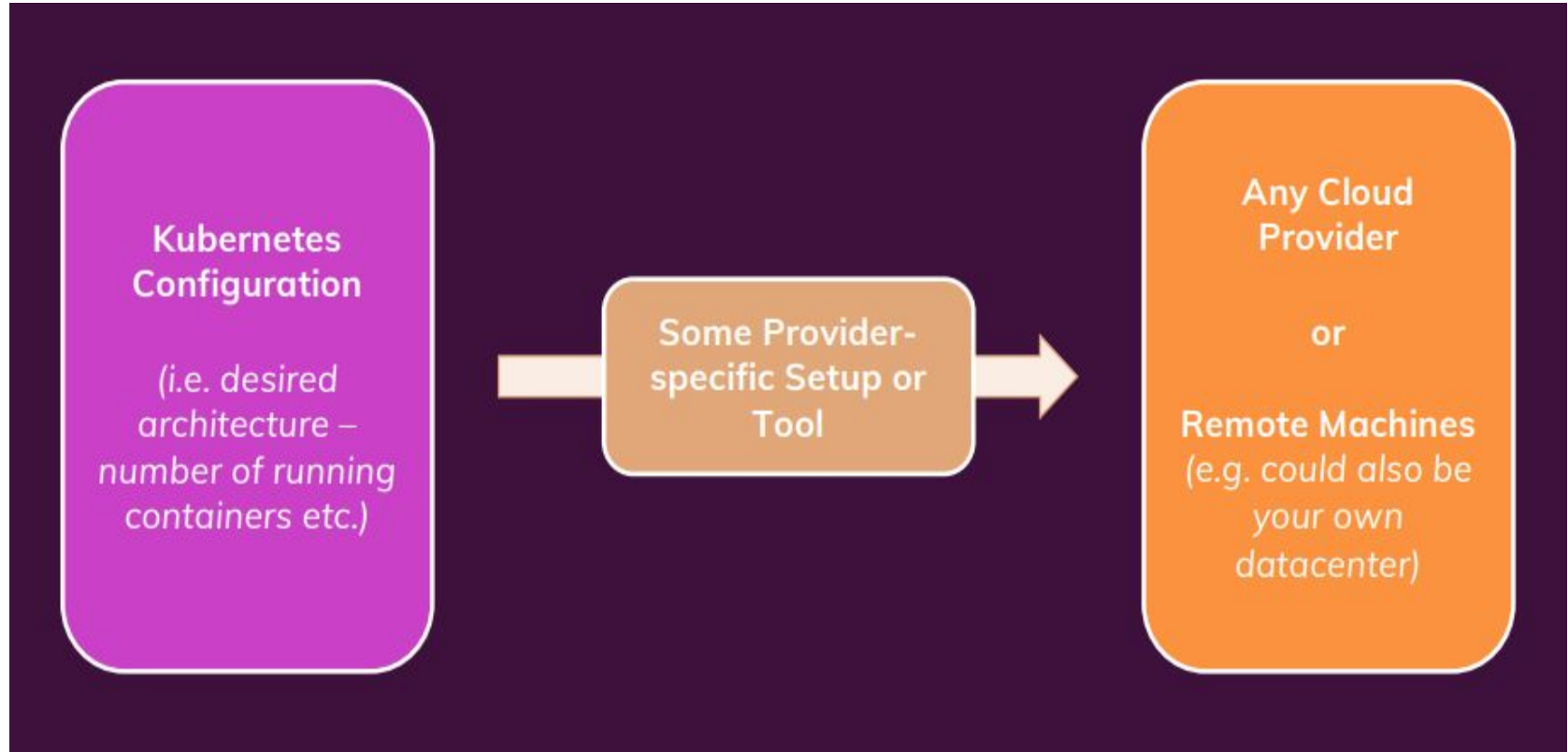
An open-source system (and de-facto standard) for orchestrating container deployments

Automatic Deployment

Scaling & Load Balancing

Management

Why Kubernetes?



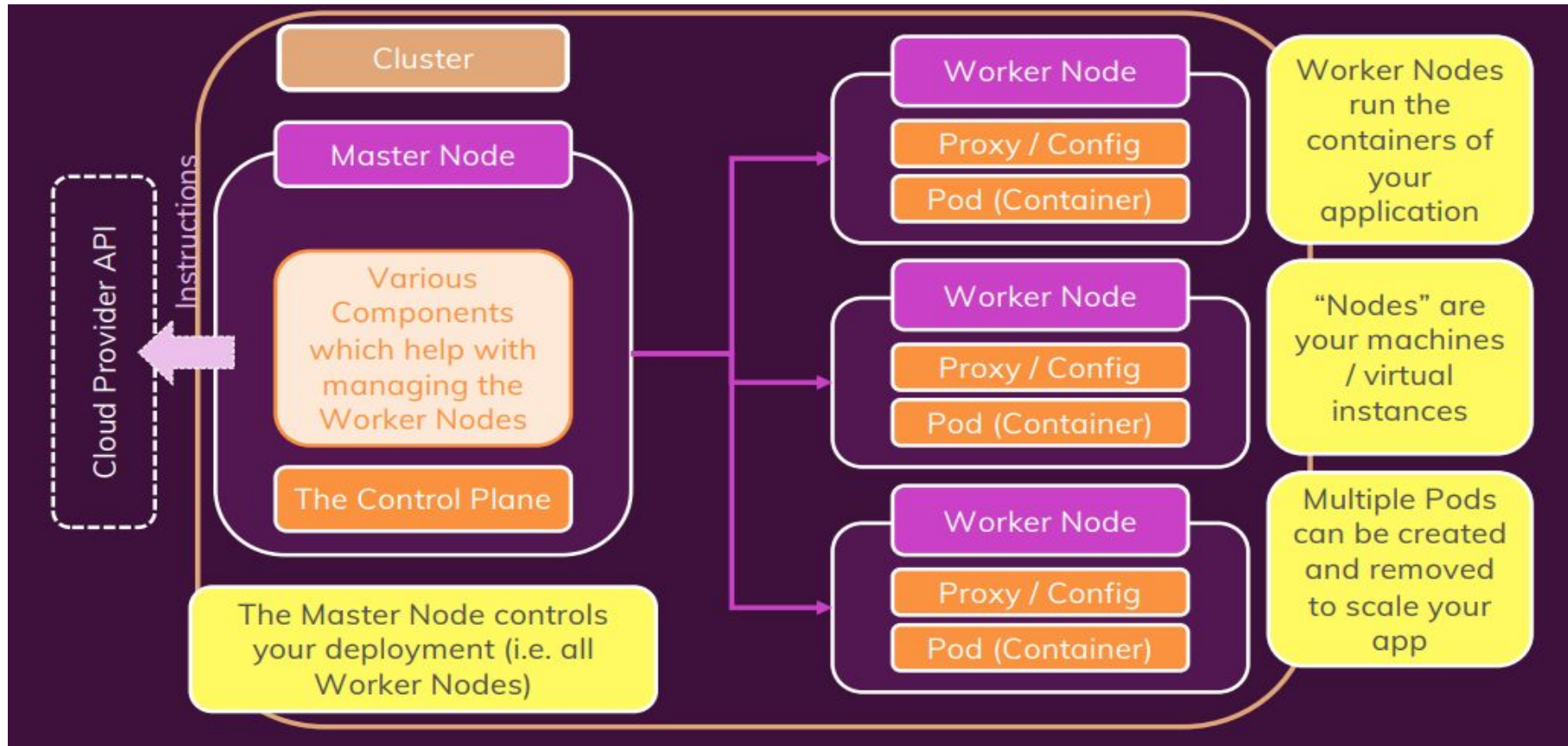
Extensible, Yet Standardized Configuration

```
apiVersion: v1
kind: Service
metadata:
  name: auth-service
  annotations:
    service.beta.kubernetes.io/aws-load-balancer-access-log-enabled: "true"
spec:
  selector:
    app: auth-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 8080
  type: LoadBalancer
...
```

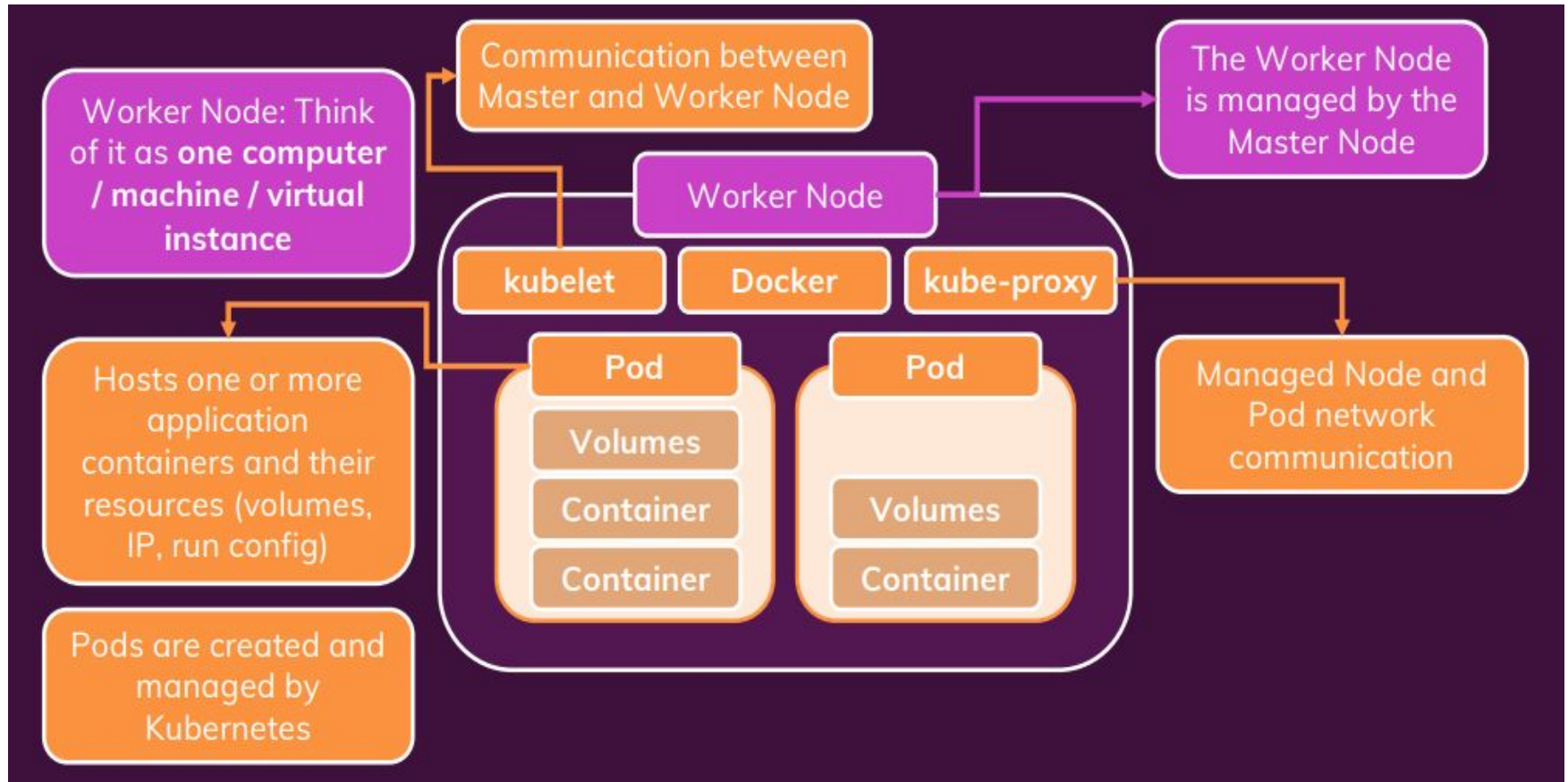
Standardized way of describing the
to-be-created and to-be-managed
resources of the Kubernetes Cluster

Cloud-provider-specific settings
can be added

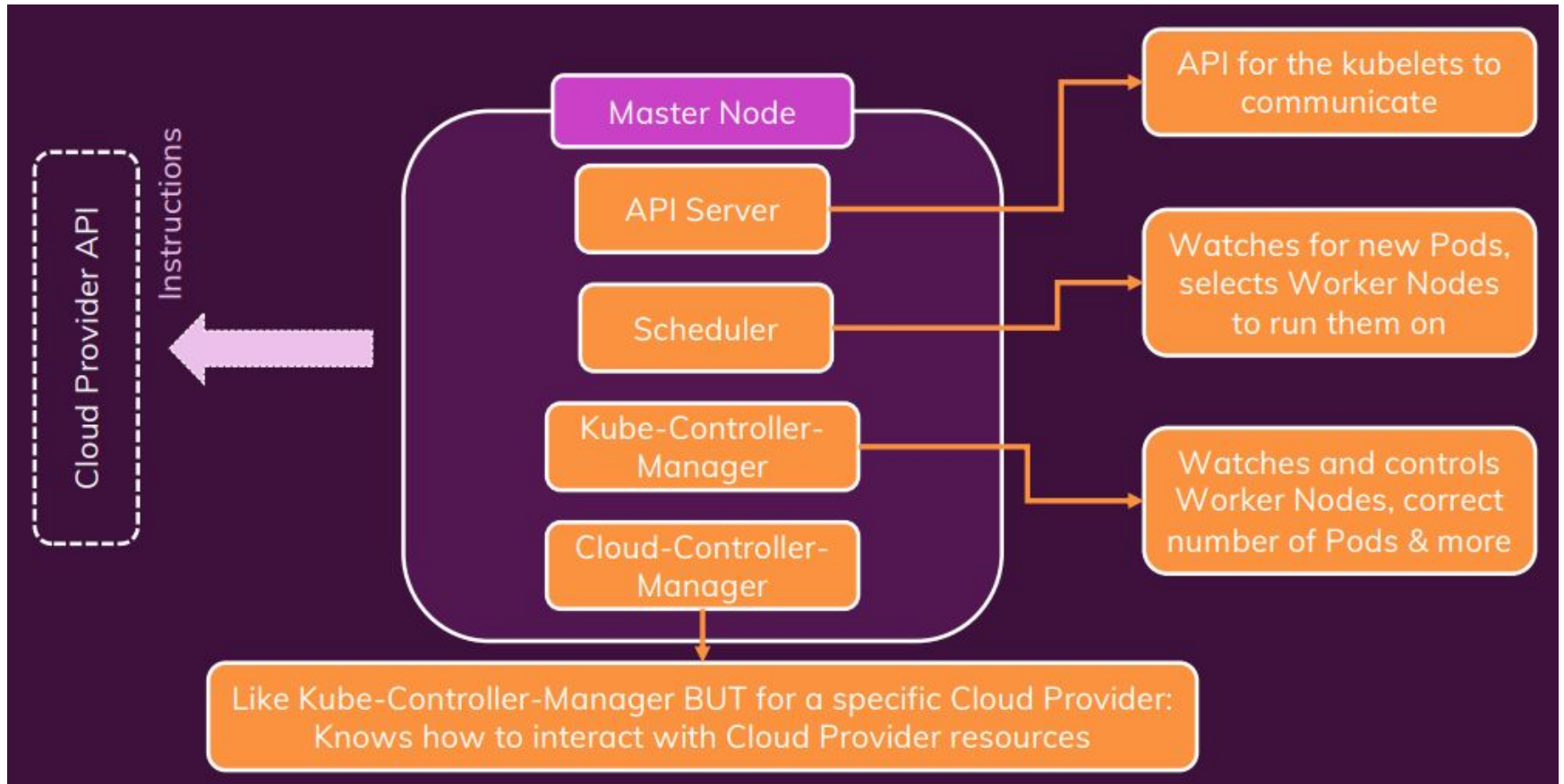
Kubernetes Architecture



A Closer Look At The Worker Nodes



A Closer Look At The Master Node



Core Components

Cluster

A set of **Node** machines which are running the **Containerized Application** (**Worker Nodes**) or control other Nodes (**Master Node**)

Nodes

Physical or virtual machine with a certain hardware capacity which hosts **one or multiple Pods** and **communicates** with the Cluster

Master Node

Cluster **Control Plane**, managing the **Pods** across Worker Nodes

Worker Node

Hosts Pods, running **App Containers** (+ **resources**)

Pods

Pods **hold the actual running App Containers** + their **required resources** (e.g. volumes).

Containers

Normal (Docker) Containers

Services

A **logical set (group)** of **Pods** with a unique, Pod- and Container-independent **IP address**

Your Work / Kubernetes' Work



What Kubernetes Will Do

Create your objects (e.g. Pods) and manage them

Monitor Pods and re-create them, scale Pods etc.

Kubernetes utilizes the provided (cloud) resources to apply your configuration / goals



What You Need To Do / Setup *(i.e. what Kubernetes requires)*

Create the Cluster and the Node Instances (Worker + Master Nodes)

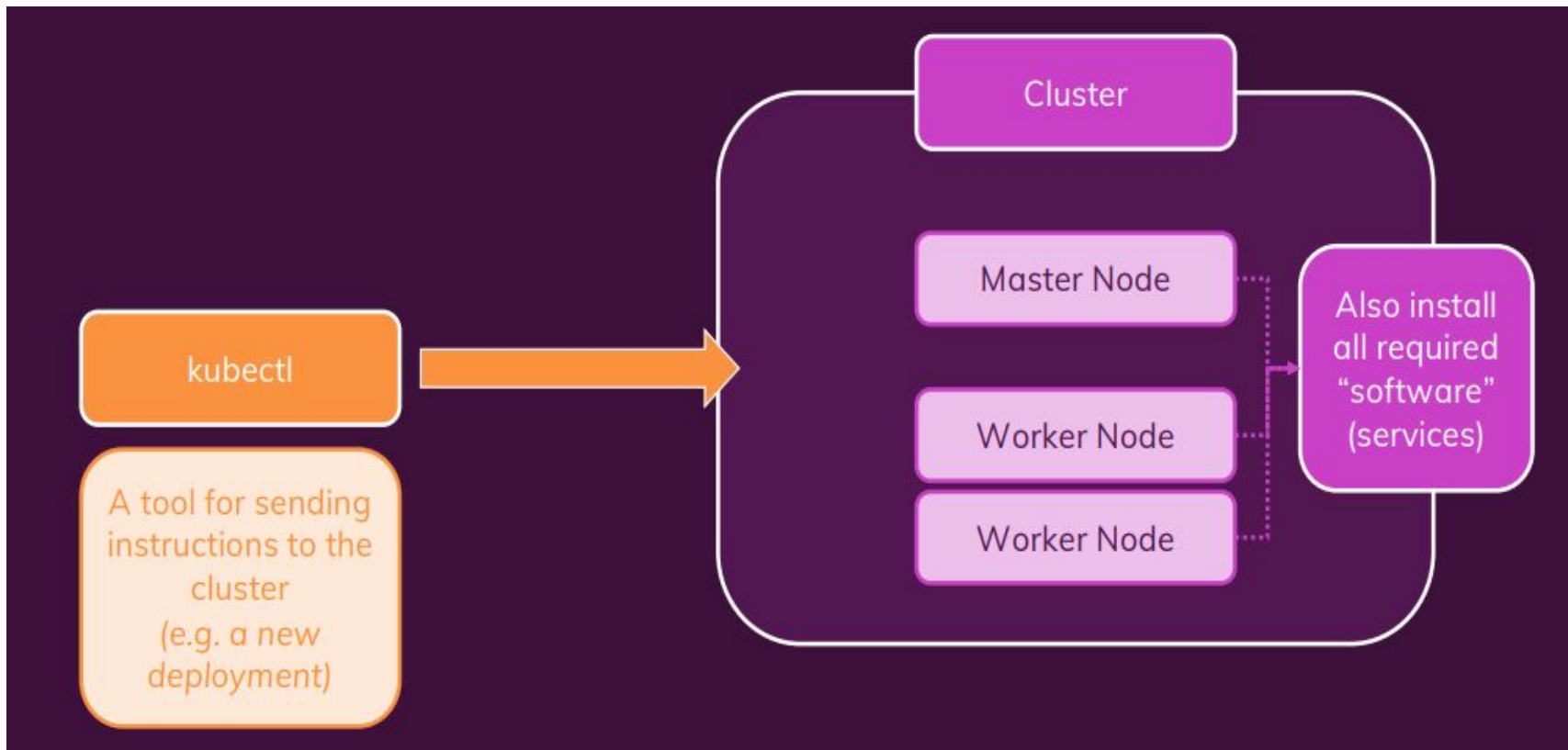
Setup API Server, kubelet and other Kubernetes services / software on Nodes

Create other (cloud) provider resources that might be needed (e.g. Load Balancer, Filesystems)

Setting up the Infrastructure

- EKS (Amazon)
- AKS (Microsoft)
- Kubermatic

kubectl



kubectl Installation

- `choco install kubernetes-cli`

```
C:\Windows\system32>choco install kubernetes-cli
```

```
Chocolatey v1.1.0
```

```
Installing the following packages:
```

```
kubernetes-cli
```

```
By installing, you accept licenses for the packages.
```

```
Progress: Downloading kubernetes-cli 1.25.3... 100%
```

```
kubernetes-cli v1.25.3 [Approved]
```

```
kubernetes-cli package files install completed. Performing other installation steps.
```

```
The package kubernetes-cli wants to run 'chocolateyInstall.ps1'.
```

```
Note: If you don't run this script, the installation will fail.
```

```
Note: To confirm automatically next time, use '-y' or consider:
```

```
choco feature enable -n allowGlobalConfirmation
```

```
Do you want to run the script?([Y]es/[A]ll - yes to all/[N]o/[P]rint): y
```


- `kubectl version --client`

```
C:\Windows\system32>kubectl version --client
```

```
WARNING: This version information is deprecated and will be replaced with the output from kubectl version --short. Use --output=yaml|json to get the full version.
```

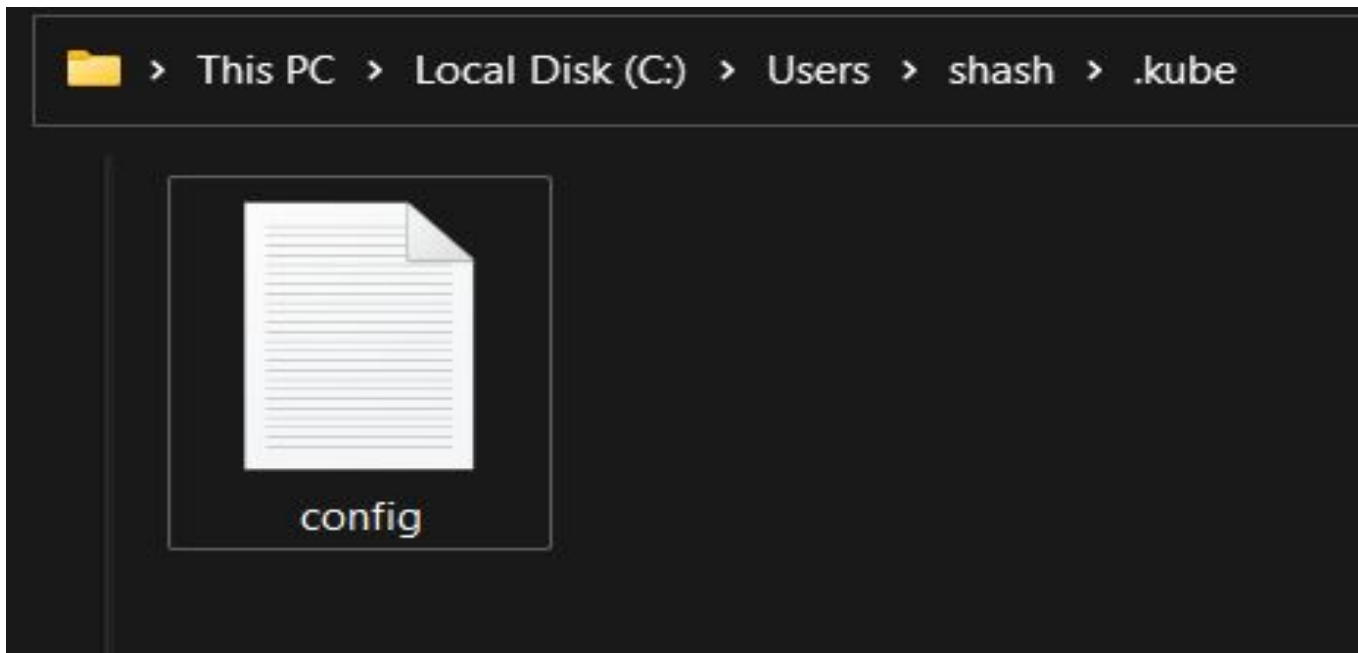
```
Client Version: version.Info{Major:"1", Minor:"25", GitVersion:"v1.25.3", GitCommit:"434bfd82814af038ad94d62ebe59b133fcb50506", GitTreeState:"clean", BuildDate:"2022-10-12T10:57:26Z", GoVersion:"go1.19.2", Compiler:"gc", Platform:"windows/amd64"}
```

```
Kustomize Version: v4.5.7
```

- Navigate to your home directory and create a ".kube" folder

```
C:\Windows\system32>cd %USERPROFILE%  
C:\Users\shash>mkdir .kube
```

- Create a config file inside this folder. (This file will later tell the kubectl command to which Cluster it should connect. It'll be populated automatically.)



Minikube Installation

- Choco install minikube

```
C:\Windows\system32>choco install minikube
```

```
Chocolatey v1.1.0
```

```
Installing the following packages:
```

```
minikube
```

```
By installing, you accept licenses for the packages.
```

```
Progress: Downloading Minikube 1.28.0... 100%
```

```
Minikube v1.28.0 [Approved]
```

```
minikube package files install completed. Performing other installation steps.
```

```
ShimGen has successfully created a shim for minikube.exe
```

```
The install of minikube was successful.
```

```
Software installed to 'C:\ProgramData\chocolatey\lib\Minikube'
```

```
Chocolatey installed 1/1 packages.
```

```
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
```

- minikube start --driver=docker

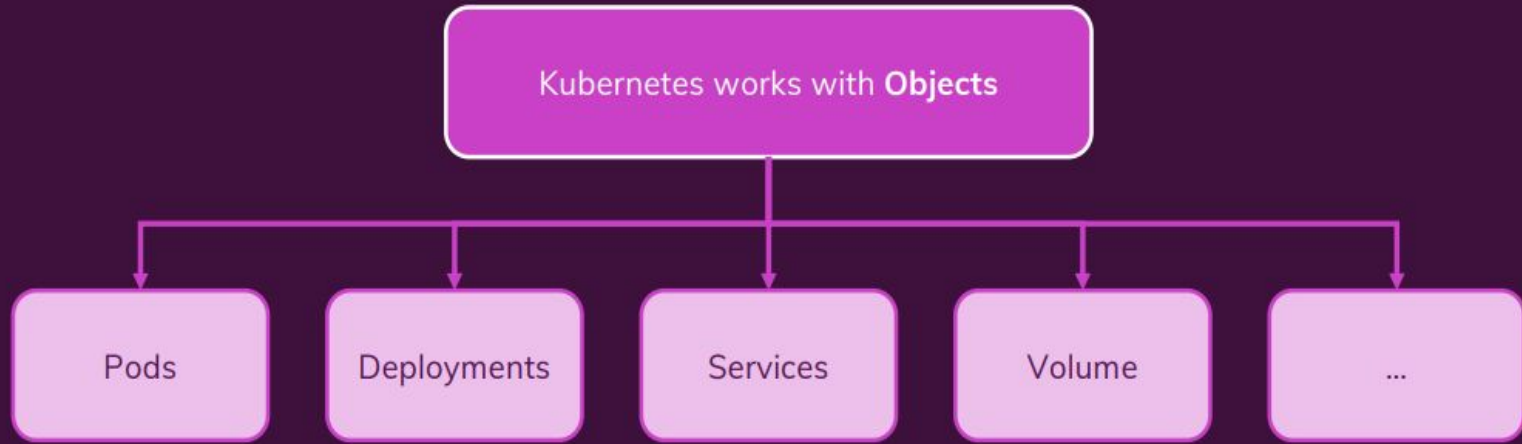
```
C:\Windows\system32>minikube start --driver=docker
* minikube v1.28.0 on Microsoft Windows 11 Home Single Language 10.0.22000 Build 22000
* Using the docker driver based on existing profile
* Starting control plane node minikube in cluster minikube
* Pulling base image ...
  > gcr.io/k8s-minikube/kicbase: 0 B [_____] ?% ? p    > gcr.io,
```

```
* docker "minikube" container is missing, will recreate.
* Creating docker container (CPUs=2, Memory=2200MB) ...
* Preparing Kubernetes v1.25.3 on Docker 20.10.20 ...
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Enabled addons: storage-provisioner, default-storageclass
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

- minikube status

```
C:\Users\shash>minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubecfg: Configured
```

Kubernetes Objects



Objects can be created in two ways:
Imperatively or Declaratively

The “Pod” Object



The smallest “unit” Kubernetes interacts with

Contains and runs one or multiple containers

The most common use-case is “one container per Pod”

Pods contain shared resources (e.g. volumes) for all Pod containers

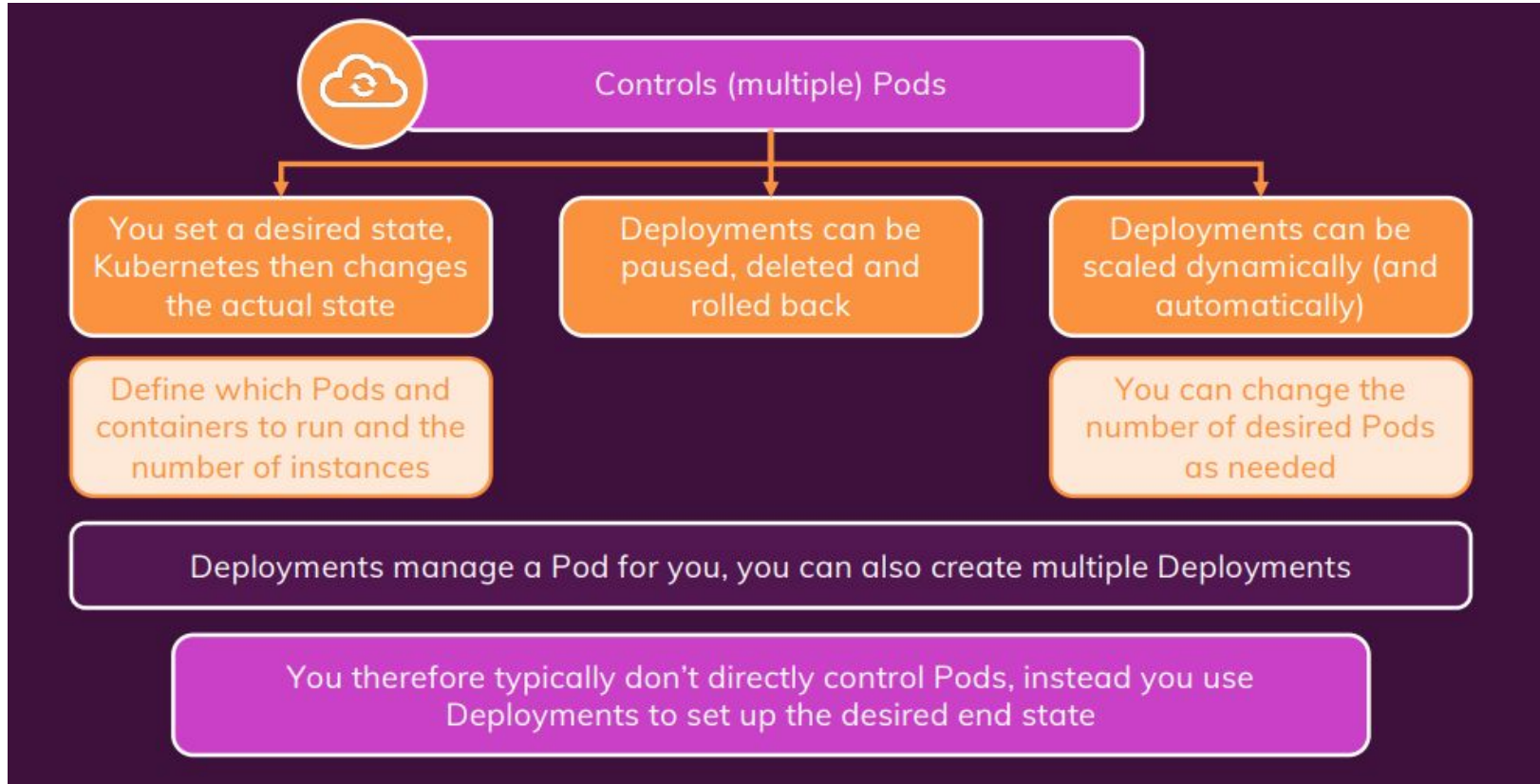
Has a cluster-internal IP by default

Containers inside a Pod can communicate via localhost

Pods are designed to be **ephemeral**: Kubernetes will start, stop and replace them as needed.

For Pods to be managed for you, you need a “**Controller**” (e.g. a “Deployment”)

The “Deployment” Object



The “Service” Object



Exposes Pods to the Cluster or Externally

Pods have an internal IP by default – it changes when a Pod is replaced

Finding Pods is hard if the IP changes all the time

Services group Pods with a shared IP

Services can allow external access to Pods

The default (internal only) can be overwritten

Without Services, Pods are very hard to reach and communication is difficult

Reaching a Pod from outside the Cluster is not possible at all without Services

Let's see a demo now

ADVANTAGES

- Simplified DevOps
- Adaptive Nature
- Universal Support
- Replicated Applications
- No Vendor Lock-Ins

DISADVANTAGES

- Steep Learning Curve
- Diverse Knowledge Needed
- Initial Configuration Difficult
- Migrating Existing Applications to Kubernetes Can Be a Pain
- Reduces Productivity