Introduction to Kubernetes

Kubernetes Pune Meetup | 29 July 2017 | Jakir Patel

Agenda

Container and Orchestration

Kubernetes - Architecture and Features

Pods and Labels

Controllers , replication sets and Deployments

Rolling updates

Services

Auto Scaling in Kubernetes

Persistence Storage

Containers are future deployment units.

How to deploy containers?

Manual Deployment: SSH to Machine and deploy manually

Automated Deployment: Chef, Puppet, Saltstack

Container Orchestration Tools: Kubernetes, Apache Mesos, Nomad

Container Orchestration

Scheduling

Handle Machine Failure

Inspection

Scaling

Replication

Service Discovery

Kubernetes

Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications.

Automated Scheduling

Self Healing

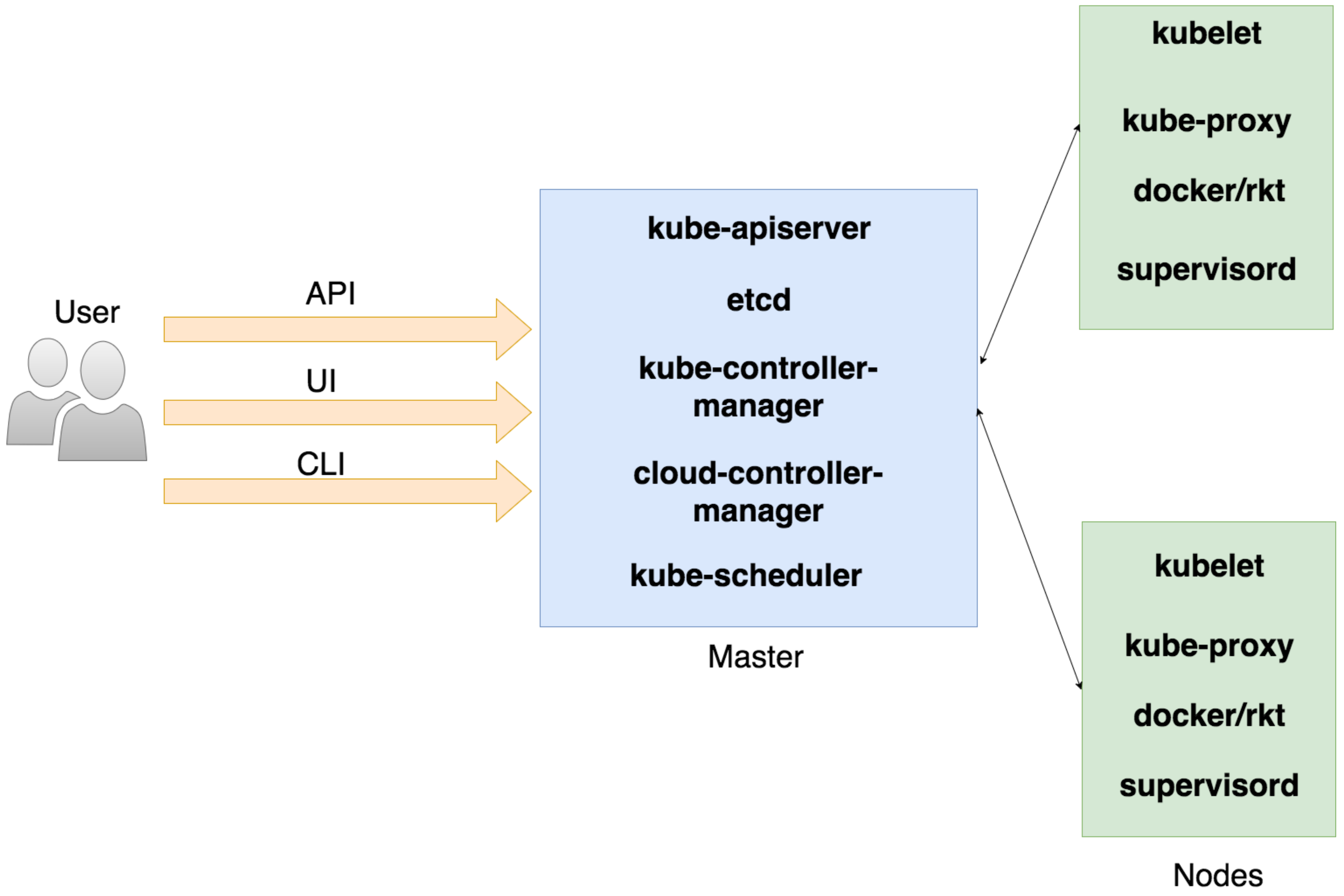
Horizontal Scaling

Service Discovery and Load Balancing

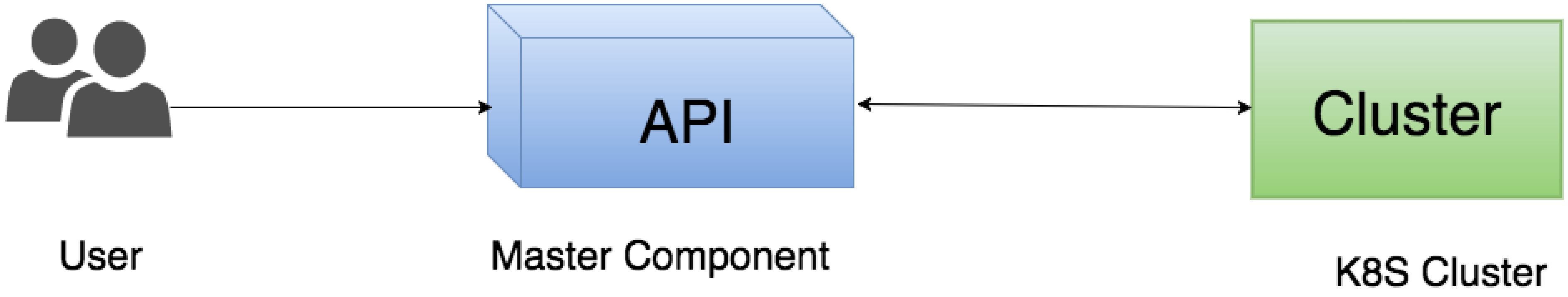
Secret and Configuration Management

Automated Rollouts and Rollback

Kubernetes Architecture View



User Interactions



Pods

Deployment Unit for Kubernetes

Run a single container

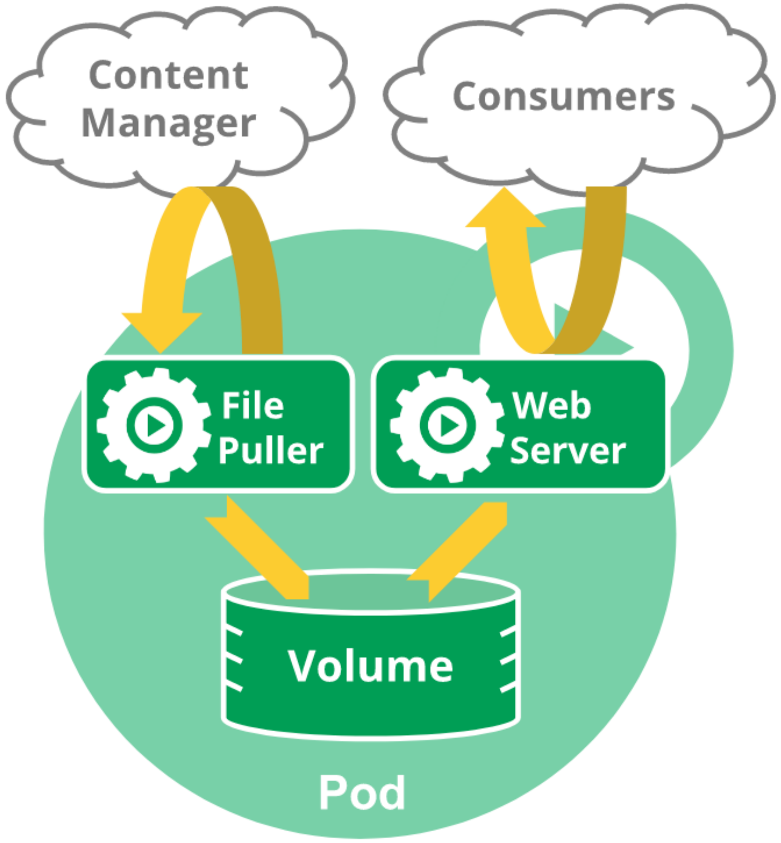
Run multiple containers

Shared namespaces: IP, IPC

Pod Template

Managed Lifecycle

- Restart in place
- Can die, UID is different for each Pod



Labels and Selectors

Identify the objects in Kubernetes. i.e. Object Identity

Grouping in Kubernetes

Represented with Key-Value pair

Query Filtering: LIST / WATCH API

Name: MyApp

Environment: Prod

Release: Stable

Name: MyApp

Environment: Dev

Release: Canary

LABELS

Kubernetes Pod Example

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
  - name: nginx
    image: nginx:1.7.9
    ports:
    - containerPort: 80
```

nginx-web.yml

Controllers

Manage the Pods

Handles replication and rollouts

Provides self-healing capabilities

Uses Pod templates to make actual Pods

Replica Sets

Deployments

Daemon Sets

Jobs

Replication Sets

Run N number of Pods

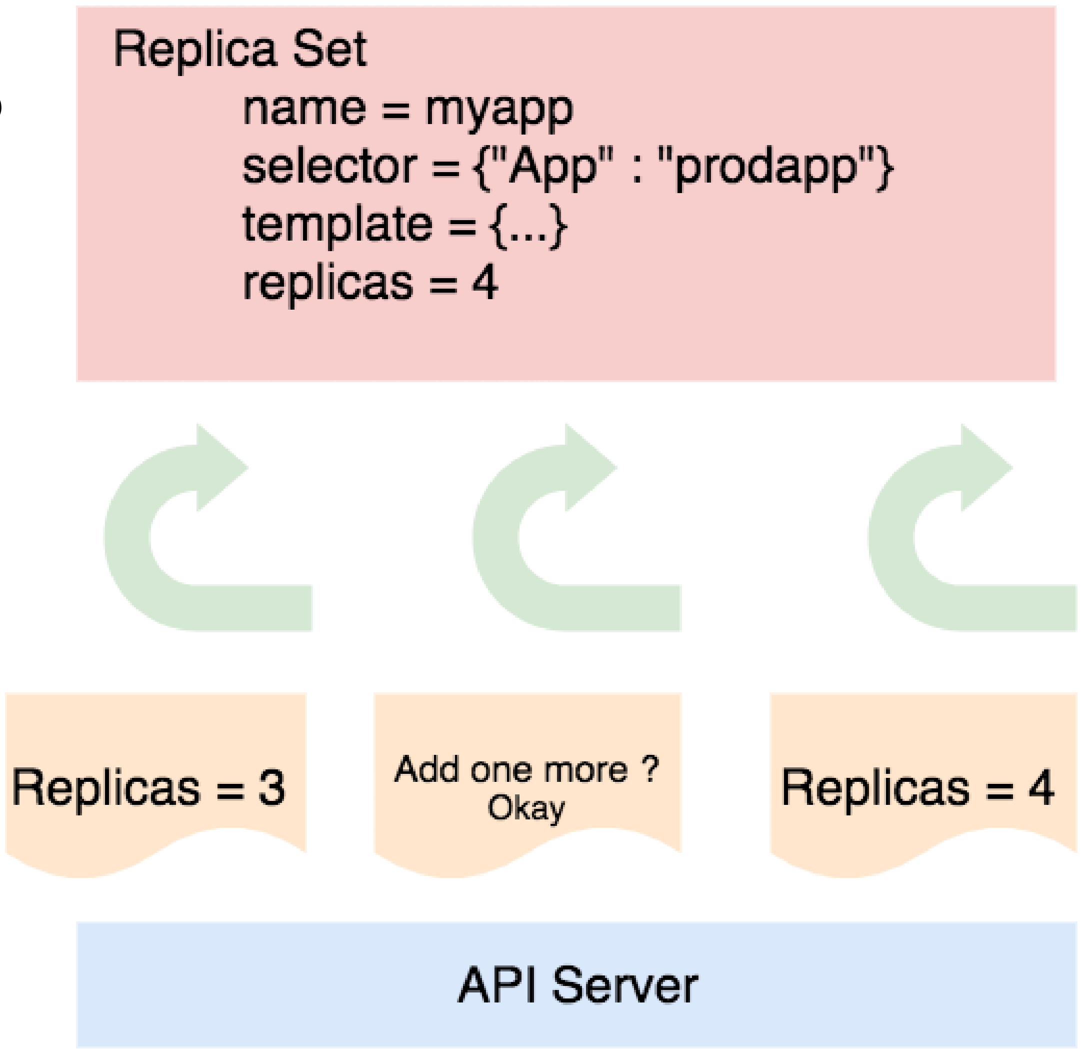
Ensures N number of Pods:

If few: start some

If more: kills some

Simple control loop

Work on top of pods



Deployment

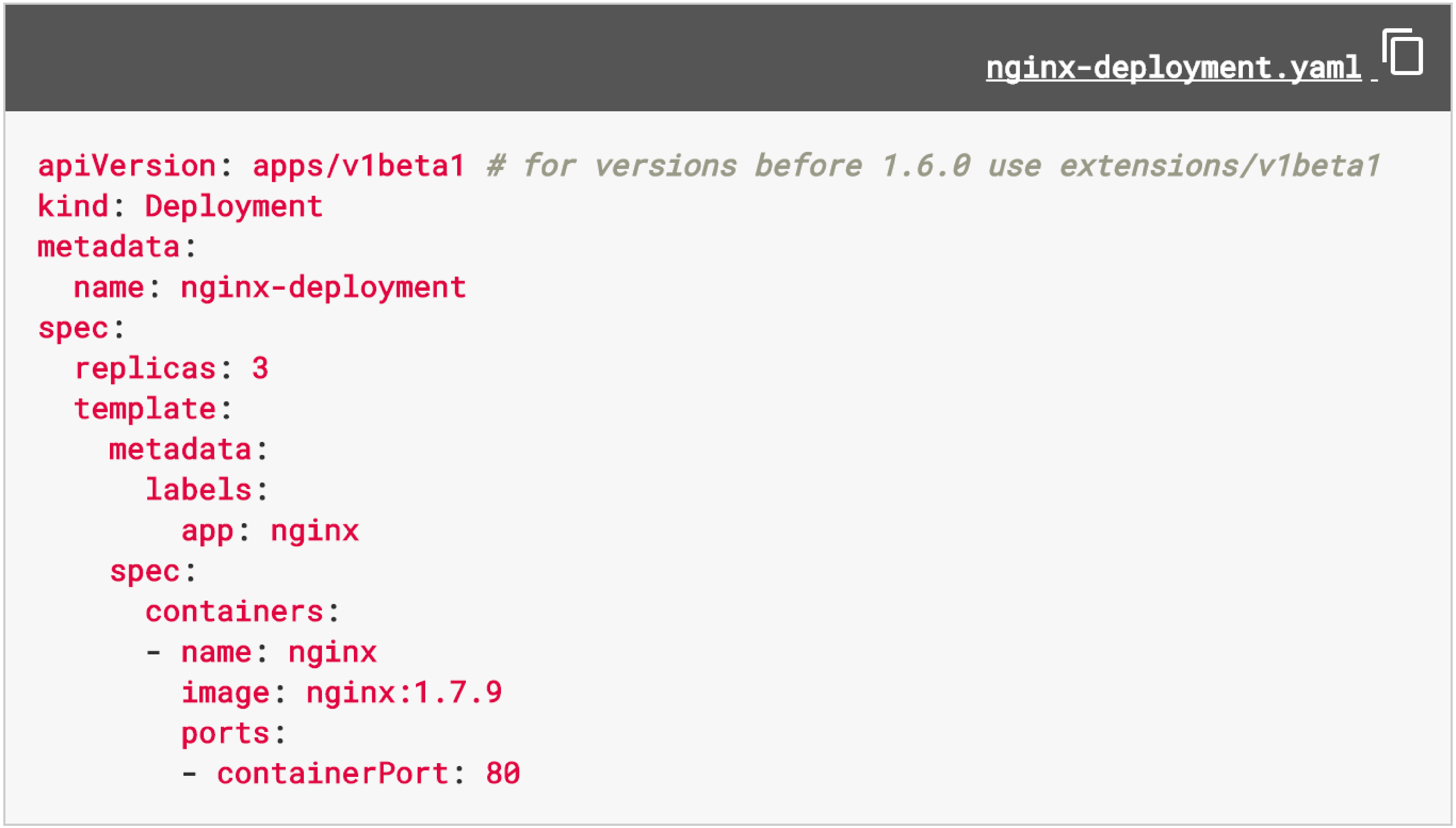
On top of Replica Sets

Declarative updates for Pods and ReplicaSets

Manage Replica changes for you

Multiple updates in flight

Kubernetes Deployment Example



```yaml
apiVersion: apps/v1beta1 # for versions before 1.6.0 use extensions/v1beta1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
        ports:
        - containerPort: 80
```

How rolling update works with Deployments?

Services

Pods are mortal. Pods can die.

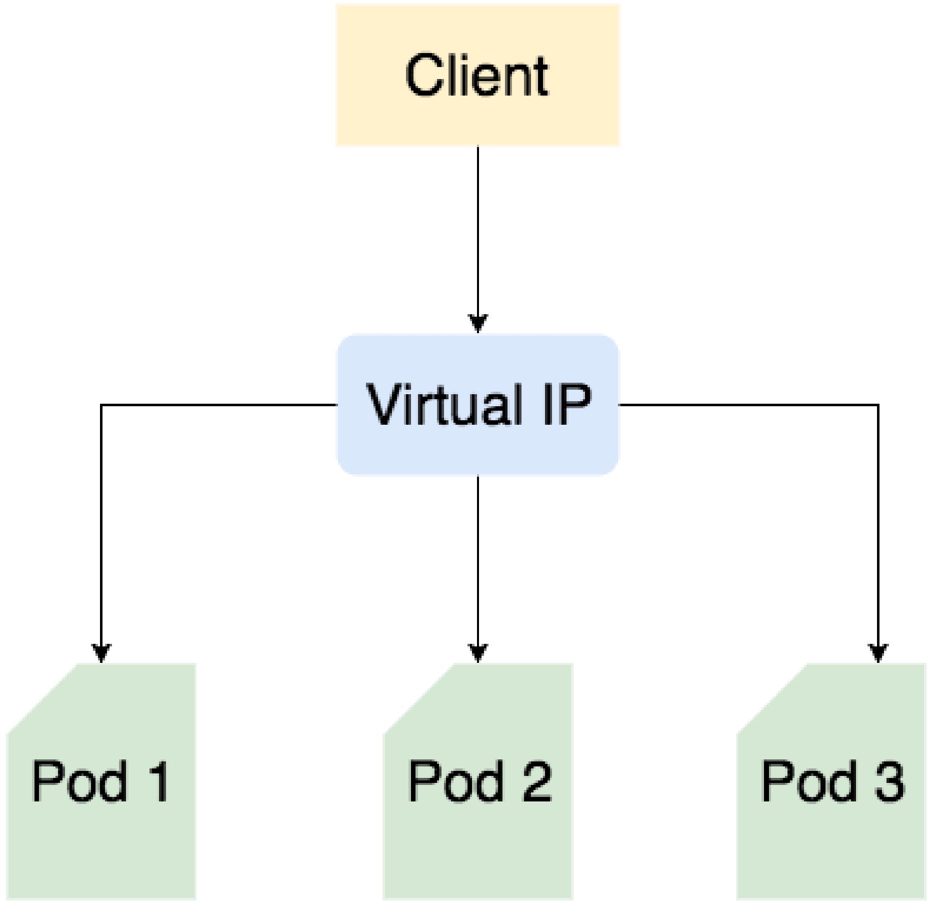
A logical set of Pods and policy to access them.

Uses label selector.

Can have Virtual IP and Port.
    Also DNS name.

Less complex.

Services Example



```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-nginx
  labels:
    run: my-nginx
spec:
  ports:
  - port: 80
    protocol: TCP
  selector:
    run: my-nginx
```

Need to expose the traffic outside world.

Services (External)

VIP's only available inside cluster.

How to expose traffic outside the world?
- NodePort
- LoadBalancer

HAProxy

Nginx

Auto Scaling

Horizontal Pod Autoscale

- Stat based
- CPU Utilization
- Set Min and Max Value

ClusterAutoscale

- Scale the number of nodes
- Scheduler stat based
- Set Min and Max Value

HorizontalPod Autoscaler

```yaml
apiVersion: extensions/v1beta1
kind: HorizontalPodAutoscaler
metadata:
  name: php-apache
  namespace: default
spec:
  scaleRef:
    kind: ReplicationController
    name: php-apache
    namespace: default
  minReplicas: 1
  maxReplicas: 10
  cpuUtilization:
    targetPercentage: 50
```

Persistent Storage

Stateful Applications

Supported plugins:
- GCEPersistentDisk
- AWSElasticBlockStore
- AzureFile
- AzureDisk
- NFS
- Cinder (OpenStack block storage)

Access Policies with Volumes

Persistent Volume

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv0003
spec:
  capacity:
    storage: 5Gi
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Recycle
  storageClassName: slow
  nfs:
    path: /tmp
    server: 172.17.0.2
```

Get Involved.

Post questions (or answer questions) on Stack Overflow

Join the community portal for advocates on K8sPort

Follow us on Twitter @Kubernetesio for latest updates

Connect with the community on Slack
Share your Kubernetes story.

Thank you.

Please don't hesitate to contact us if you have any questions

jakirpatel@outlook.com