



# Kubernetes (GKE)

# Kubernetes (GKE)

## Course Outline

### Day 1

- Google Container Registry
- Kubernetes Architecture
- Nodes
- Pods
- Services
- Replica Sets
- Deployments
- Daemon Sets
- Resource Quotas
- Health Check
- Horizontal Pod Autoscaler
- Labels
- Pod Disruption Budgets
- GCE PD

### Day 2

- Disks
- Stateful Sets
- Config Maps
- Secrets
- Ingresses
- GCP L7 Ingress
- NGINX Ingress
- Automated https let's encrypt
- Jobs & CronJobs
- Automated Deploy



**acoshift/  
course-kubernetes**

DAY I

# YAML

- stands for “YAML Ain’t Markup Language”
- is a human friendly data serialization standard for all programming language

# YAML

```
name: Courses
list:
- name: Go for Beginner
  price: 600
- name: Redis Fundamental
  price: 300
- name: RxJS for Beginner
  price: 500
```

# JSON

```
{
  "name": "Courses",
  "list": [
    {
      "name": "Go for Beginner",
      "price": 600
    },
    {
      "name": "Redis Fundamental",
      "price": 300
    },
    {
      "name": "RxJS for Beginner",
      "price": 500
    }
  ]
}
```

# Google Container Registry

<https://cloud.google.com/container-registry/>





## Container Registry

[REFRESH](#)[SHOW PULL COMMAND](#)[DELETE](#)

[gcr.io](#) / [acoshift-1362](#) / [acourse](#)

<input type="checkbox"/> Name	Tags	Virtual size	Uploaded
<input type="checkbox"/> 1f5261270fb3	7d9feb45ba5038b84856f51142e730a23fe9a3b9 latest	3.9 MB	2 days ago
<input type="checkbox"/> 6b81b1966f31	38f520dbf60aae34bb5528fd7559666e2c5f3eaf 	3.9 MB	2 days ago
<input type="checkbox"/> 4dd203deaac1	09f311488b5756981ed1f8089b225032c98d1d1d 	3.9 MB	2 days ago
<input type="checkbox"/> b6bb495787e3	049cc2b0944aba25cf2b71a7b051cc5d8a807d5c 	3.9 MB	2 days ago
<input type="checkbox"/> a1e2abf97afa	09695bdb99fcbe4114e9e120e36b3a35881c2228 	3.9 MB	4 days ago
<input type="checkbox"/> c81615a32bfb	6444528f0898dc7074b03b2155702178e0cee3a4 	3.9 MB	6 days ago
<input type="checkbox"/> 340072282343	390c17ac043b2f9881496b00430b451b49cb773a 	3.9 MB	7 days ago
<input type="checkbox"/> bf6d895e3d4f	b0b70dd5745c99969205a087b9b745c1fbdb6560 	3.9 MB	8 days ago
<input type="checkbox"/> d2d2aaa26269	5917485deaf70248110e2da0e6296e014967e1f5 	3.9 MB	8 days ago
<input type="checkbox"/> ef65e4cce15c	fe5253cdfa5a59e861fd2e61e7f77b7b7d97f1c5 	3.9 MB	9 days ago
<input type="checkbox"/> dfbc13dac286	977399a9770be959ae830da88933cd9e4884ad3f	3.9 MB	12 days ago



```
$ docker push acoshift/backend:1.0.0
```

```
$ gcloud docker -- push gcr.io/myproject/backend:1.0.0
```

```
$ docker pull acoshift/backend:1.0.0
```

```
$ gcloud docker -- pull gcr.io/myproject/backend:1.0.0
```

```
$ docker login -u _json_key -p "$(cat keyfile.json)" https://gcr.io
$ docker push gcr.io/myproject/backend:1.0.0
$ docker pull gcr.io/myproject/backend:1.0.0
```

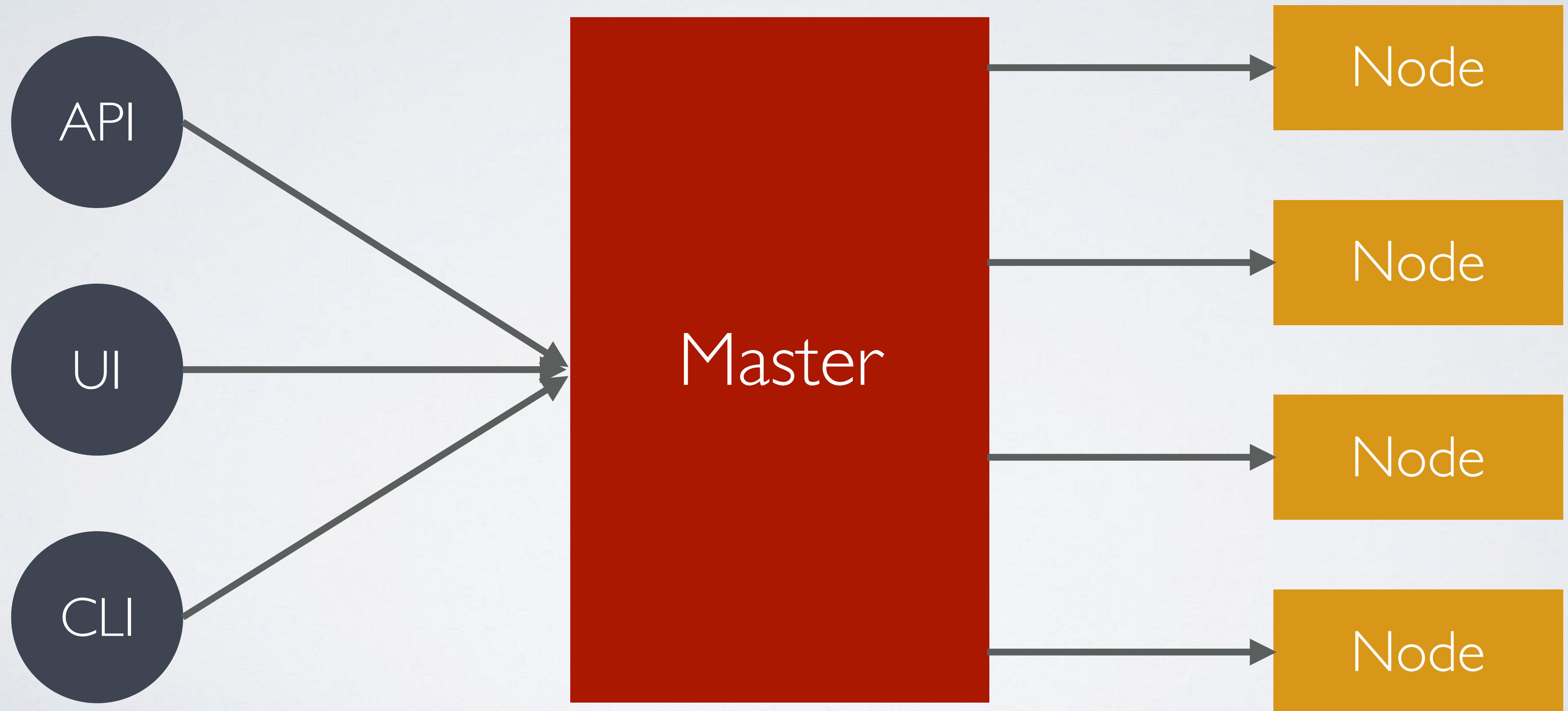
# docker-credential-gcr

```
$ gcloud components install docker-credential-gcr  
$ docker-credential-gcr configure-docker  
$ docker-credential-gcr gcr-login
```

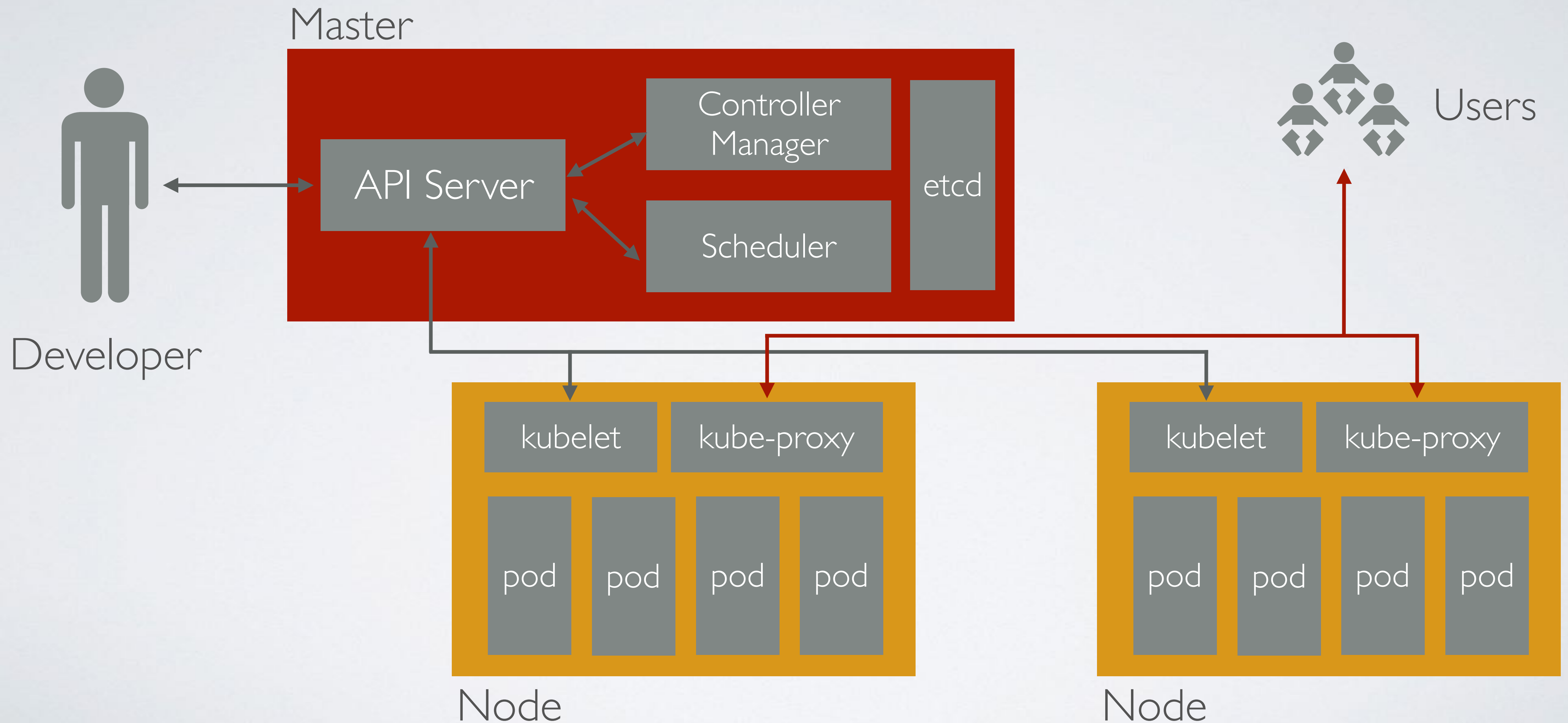
```
$ docker push gcr.io/myproject/backend:1.0.0  
$ docker pull gcr.io/myproject/backend:1.0.0
```

**<https://gcr.io/google-containers/global>**

# Kubernetes Architecture



# Kubernetes Architecture





# Nodes (no)

a worker machine in Kubernetes

```
$ kubectl get nodes
```

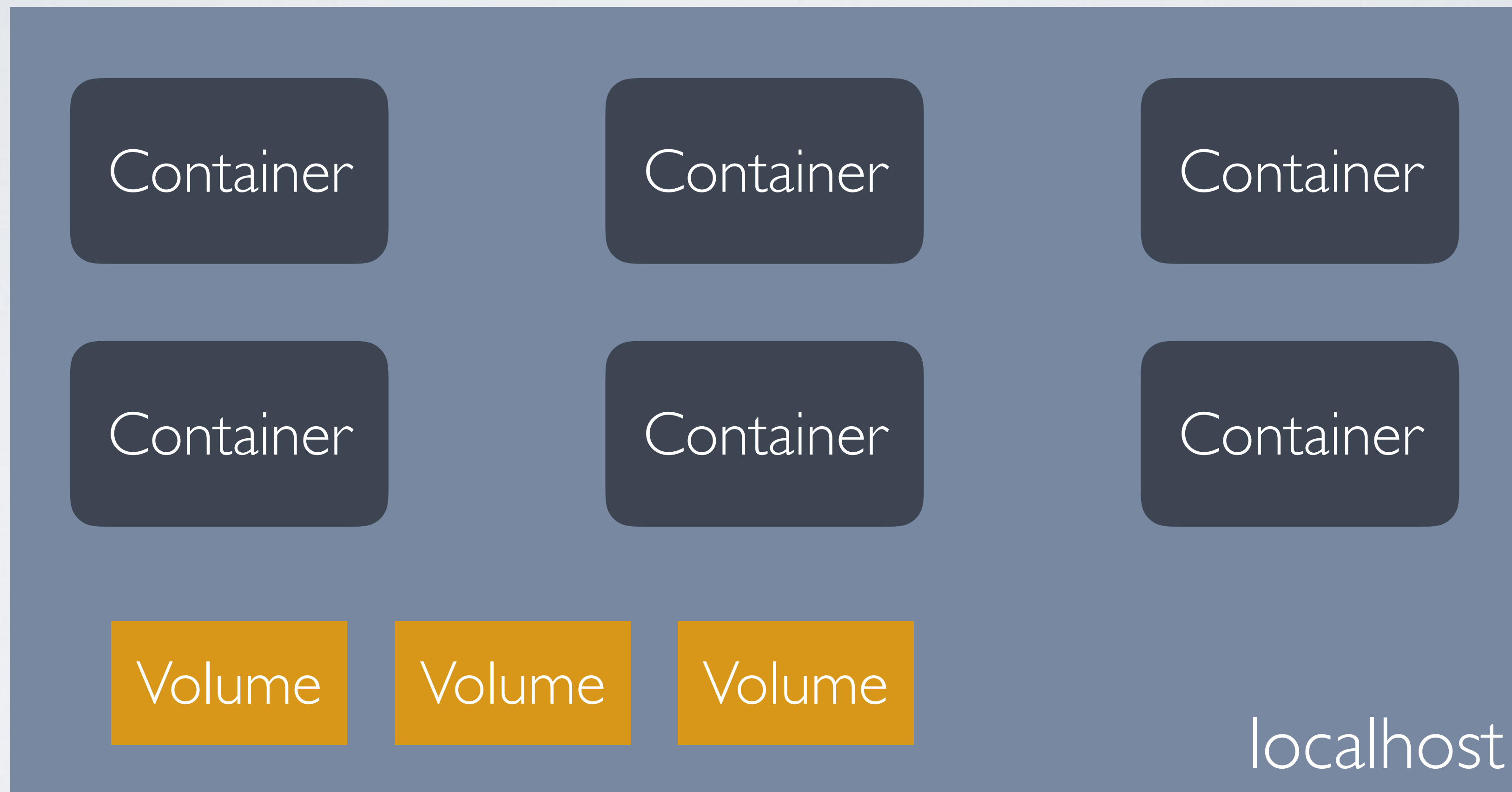
NAME	STATUS	AGE	VERSION
gke-cluster-sg-1-pool-1-3fada004-n6gj	Ready	4d	v1.7.0
gke-cluster-sg-1-pool-1-3fada004-pglr	Ready	4d	v1.7.0

```
$ kubectl describe nodes gke-cluster-sg-1-pool-1-3fada004-n6gj
```

# Pods (po)

a group of one or more containers

# Pod



10.0.1.4

```
kind: Pod
apiVersion: v1
metadata:
  name: echoserver
spec:
  containers:
  - name: echoserver
    image: gcr.io/google-containers/echoserver:1.6
    ports:
    - containerPort: 8080
```

just additional  
information

all ports listening on  
0.0.0.0 will be accessible  
from network

```
$ kubectl create -f pod.yaml  
pod "echoserver" created
```



```
$ kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
echoserver	1/1	Running	0	4m

```
$ kubectl port-forward echoserver 9000:8080  
Forwarding from 127.0.0.1:9000 -> 8080  
Forwarding from [::1]:9000 -> 8080
```

```
$ curl localhost:9000
Hostname: echoserver
```

```
Pod Information:
  -no pod information available-
```

```
Server values:
  server_version=nginx: 1.13.1 - lua: 10008
```

```
Request Information:
  client_address=127.0.0.1
  method=GET
  real path=/
  query=
  request_version=1.1
  request_uri=http://localhost:8080/
```

```
Request Headers:
  accept=/*/*
  host=localhost:9000
  user-agent=curl/7.51.0
```

```
Request Body:
  -no body in request-
```

```
$ kubectl delete pod echoserver  
pod "echoserver" deleted
```

```
kind: Pod
apiVersion: v1
metadata:
  name: web
spec:
  volumes:
  - name: www
    emptyDir: {}
  containers:
  - name: nginx
    image: gcr.io/google-containers/nginx-slim:0.8
    ports:
    - containerPort: 80
    volumeMounts:
    - name: www
      mountPath: /usr/share/nginx/html
  - name: ubuntu
    image: ubuntu
    volumeMounts:
    - name: www
      mountPath: /data
    command:
    - /bin/sh
    args:
    - -c
    - while true; do dd if=/dev/urandom bs=32 count=1 | base64 > /data/index.html; sleep 1; done
```

```
$ kubectl create -f multi-container.yaml  
pod "web" created
```

```
$ kubectl port-forward web 8080:80  
Forwarding from 127.0.0.1:8080 -> 80  
Forwarding from [::1]:8080 -> 80
```

```
$ curl localhost:8080
```



```
$ kubectl exec web -itc ubuntu -- bash
```

```
root@web:/# apt-get update
```

```
root@web:/# apt-get install curl
```

```
root@web:/# curl localhost
```

```
s4zX9vA0juonZhYlCjfRiXUpIV54EsAfz+UwAgnrWhA=
```

```
root@web:/# curl localhost
```

```
n7dhG+ZDK//+vQm/M6upoA55JqK5lQ96tYsiDdGj+7M=
```

```
root@web:/# curl localhost -I
```

```
HTTP/1.1 200 OK
```

```
Server: nginx/1.11.1
```

```
Date: Fri, 21 Jul 2017 12:26:55 GMT
```

```
Content-Type: text/html
```

```
Content-Length: 45
```

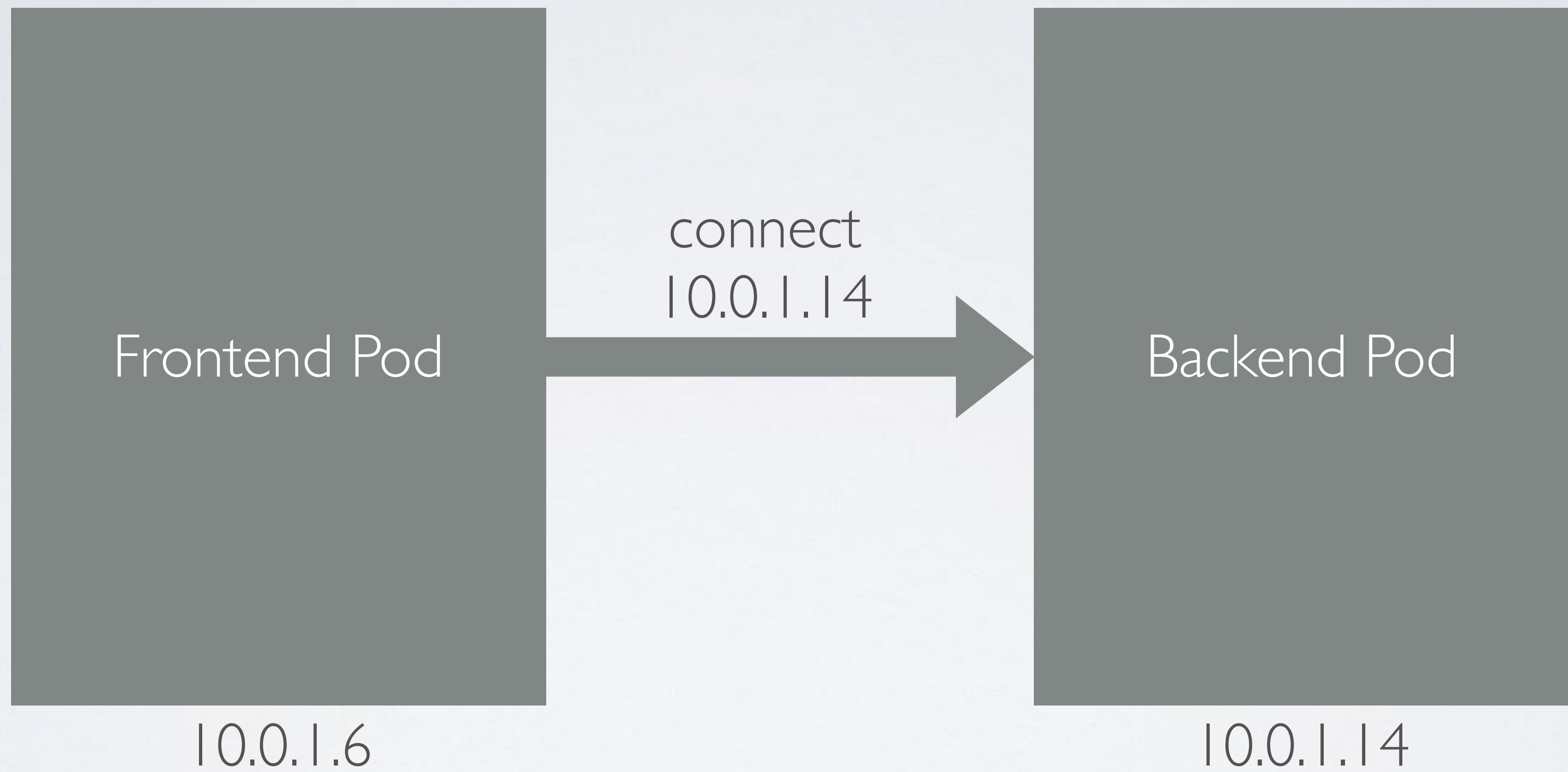
```
Last-Modified: Fri, 21 Jul 2017 12:26:54 GMT
```

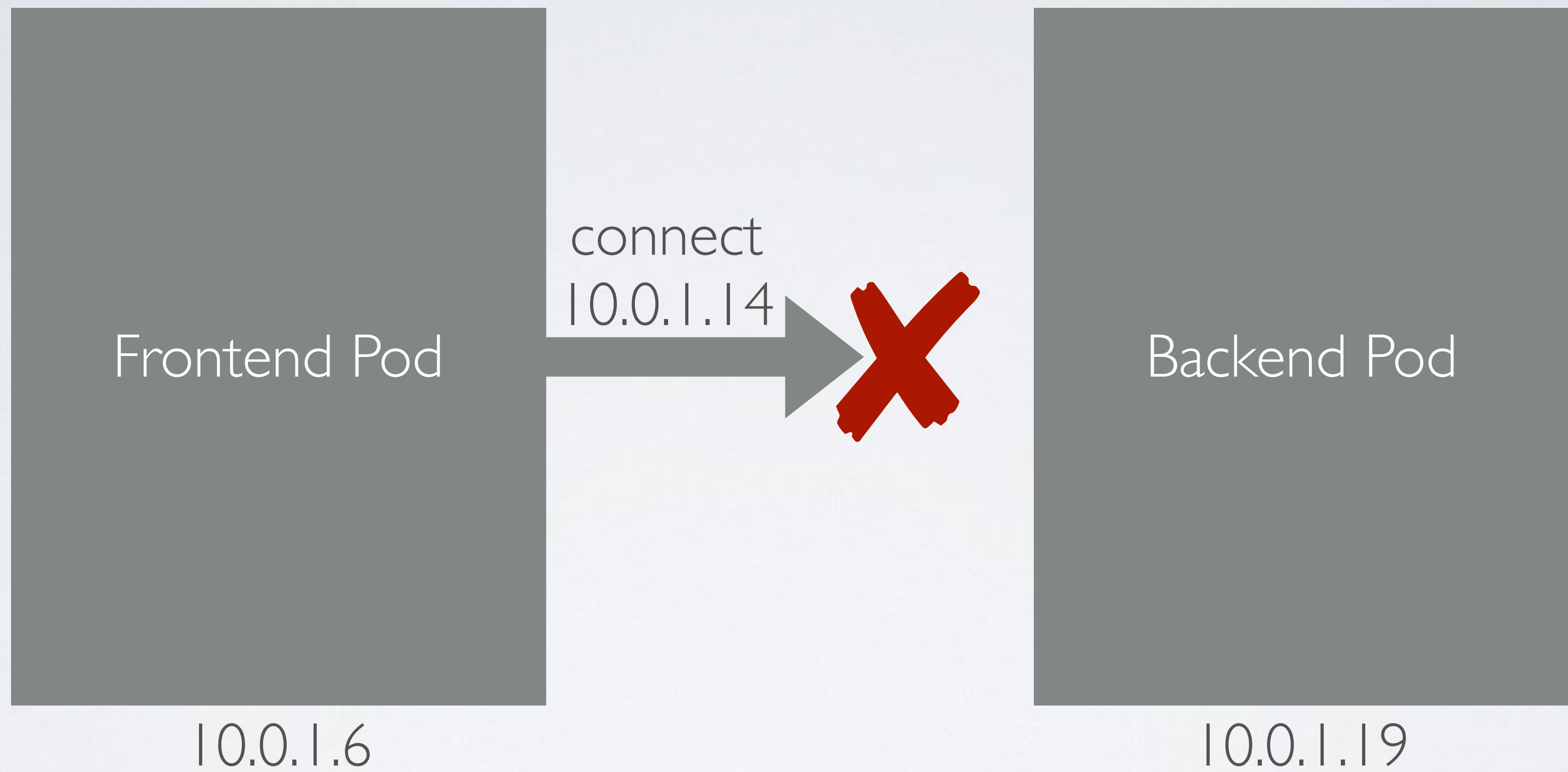
```
Connection: keep-alive
```

```
ETag: "5971f30e-2d"
```

```
Accept-Ranges: bytes
```

```
$ kubectl delete -f multi-container.yaml  
pod "web" deleted
```





# Services (svc)

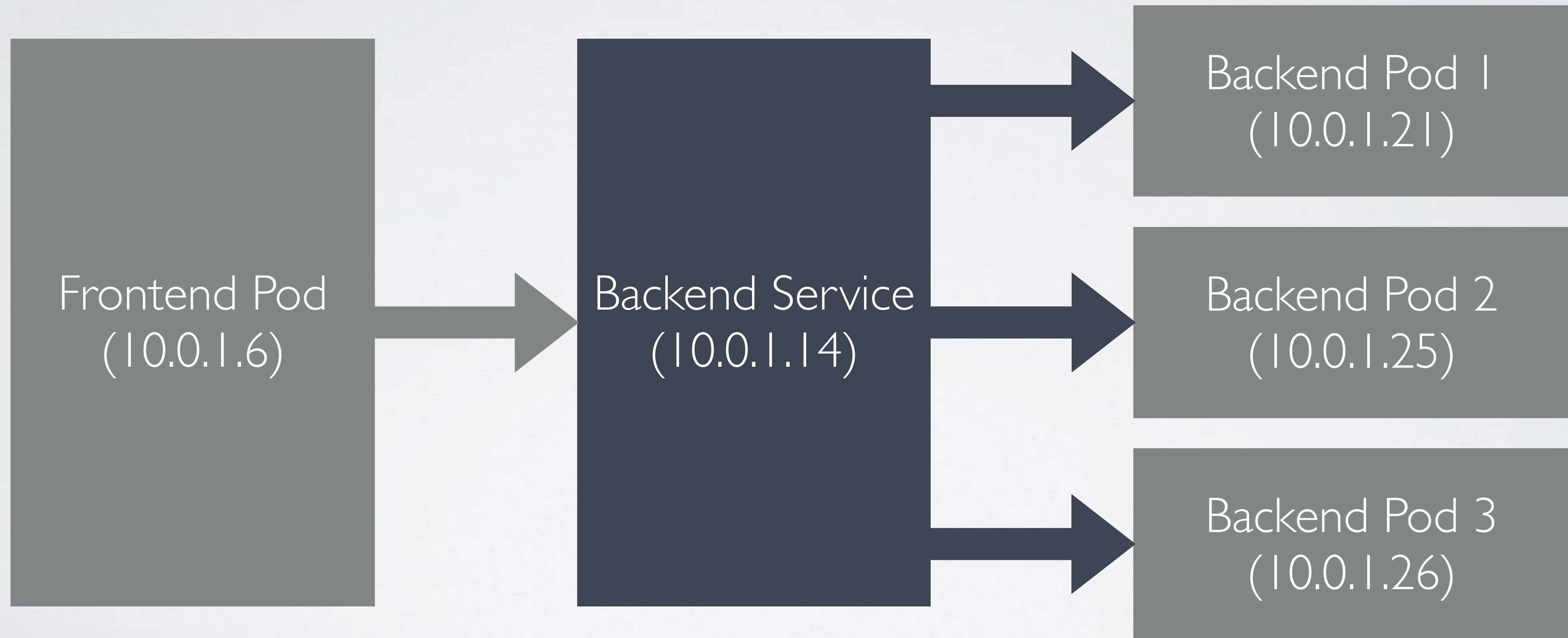
an abstraction which defines a logical set of Pods and a policy by which to access them

# Service Types

- ClusterIP
- NodePort
- LoadBalancer
- ExternalName



# ClusterIP



```
kind: Pod
apiVersion: v1
metadata:
  name: echoserver
  labels:
    app: echoserver
spec:
  containers:
  - name: echoserver
    image: gcr.io/google-containers/echoserver:1.6
    ports:
    - containerPort: 8080
```

```
kind: Service
apiVersion: v1
metadata:
  name: echoserver
spec:
  selector:
    app: echoserver
  ports:
    - port: 80
      targetPort: 8080
```

```
$ kubectl create -f clusterIp.yaml  
pod "echoserver" created  
service "echoserver" created
```

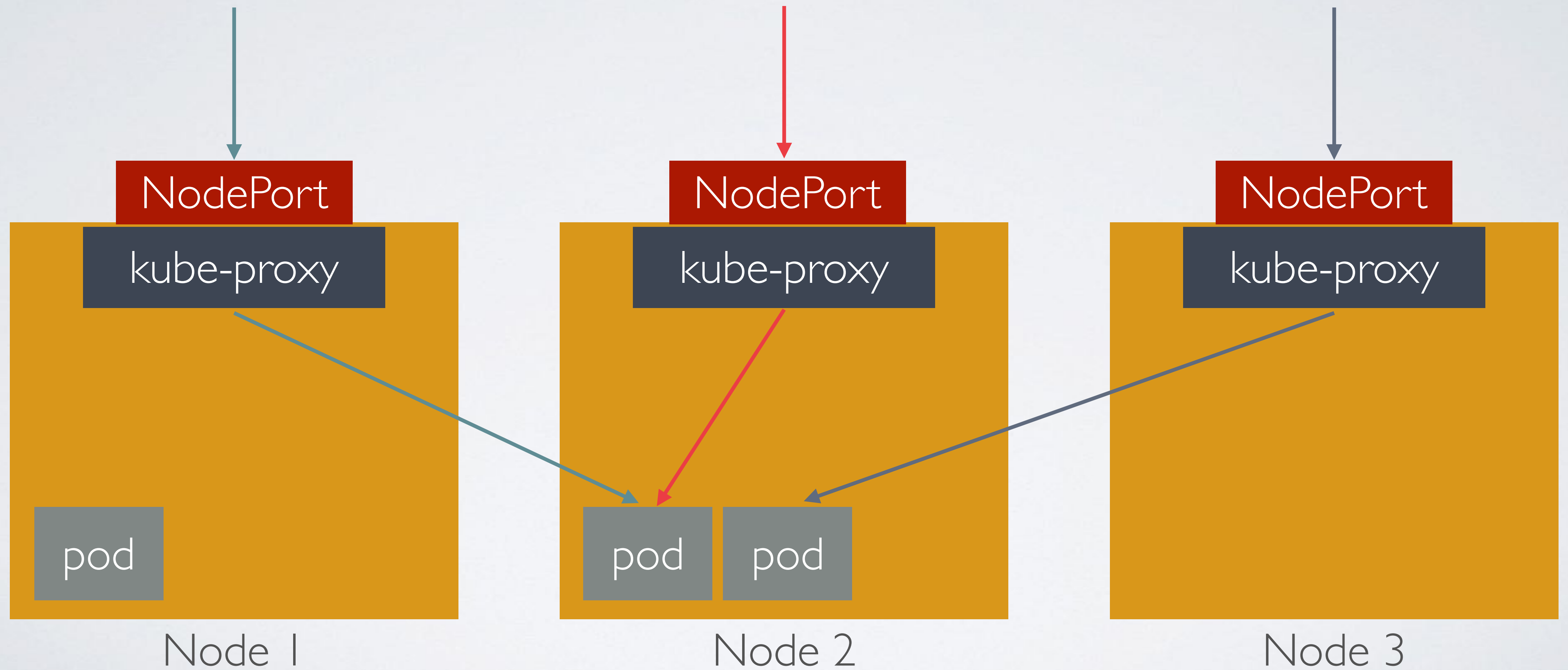
```
$ kubectl get services
```

NAME	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
echoserver	10.3.248.15	<none>	80/TCP	11s

```
$ kubectl run -it --rm busybox --image=busybox  
$ wget -O- http://echoserver
```

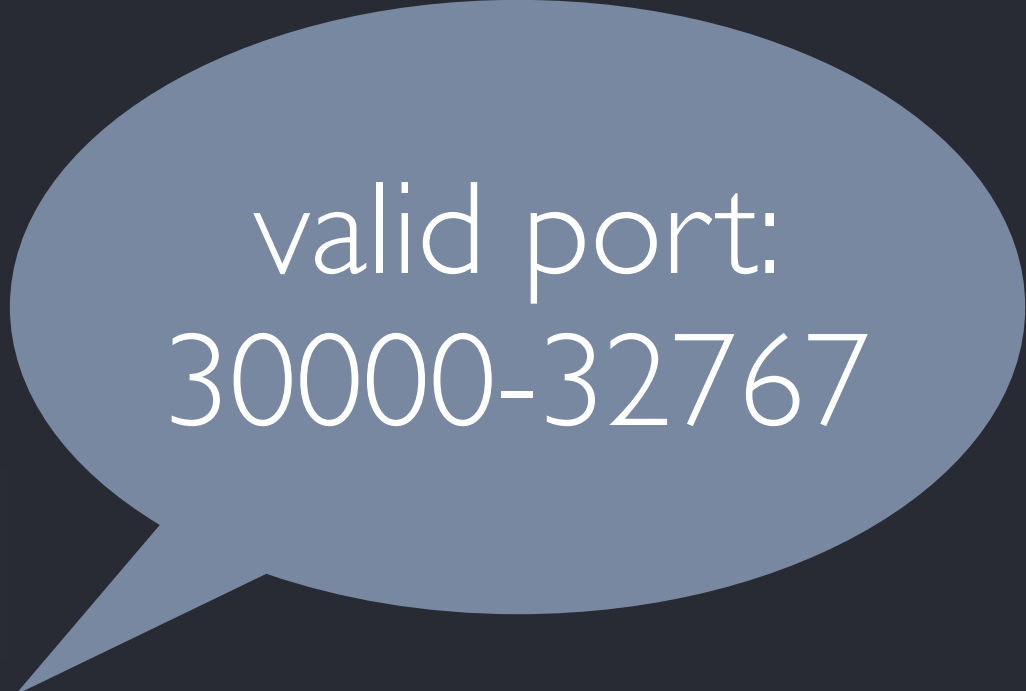
```
$ kubectl delete -f clusterIp.yaml
```

# NodePort





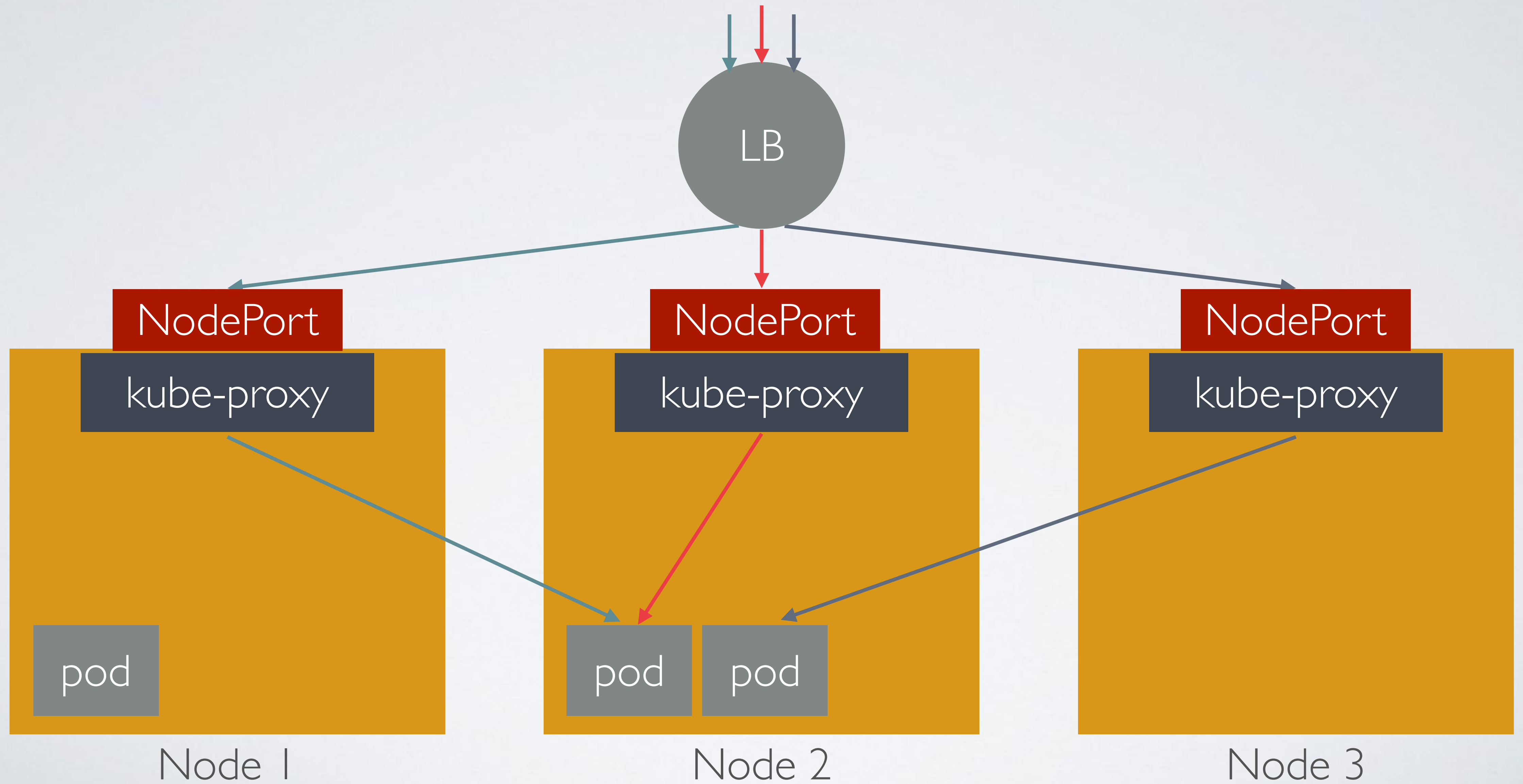
```
kind: Service
apiVersion: v1
metadata:
  name: echoserver
spec:
  type: NodePort
  selector:
    app: echoserver
  ports:
  - port: 80
    targetPort: 8080
    nodePort: 31000
```



valid port:  
30000-32767

```
$ curl http://serverIP:31000
```

# LoadBalancer



```
kind: Service
apiVersion: v1
metadata:
  name: echoserver
spec:
  type: LoadBalancer
  selector:
    app: echoserver
  ports:
  - port: 80
    targetPort: 8080
  loadBalancerIP: 35.185.1.1
```



optional static ip

```
$ curl http://loadbalcnerIP
```

# ExternalName



```
kind: Service
apiVersion: v1
metadata:
  name: google
spec:
  type: ExternalName
  externalName: google.com
```

```
$ kubectl run -it --rm busybox --image=busybox  
$ wget -O- --header='Host: www.google.com' http://google
```

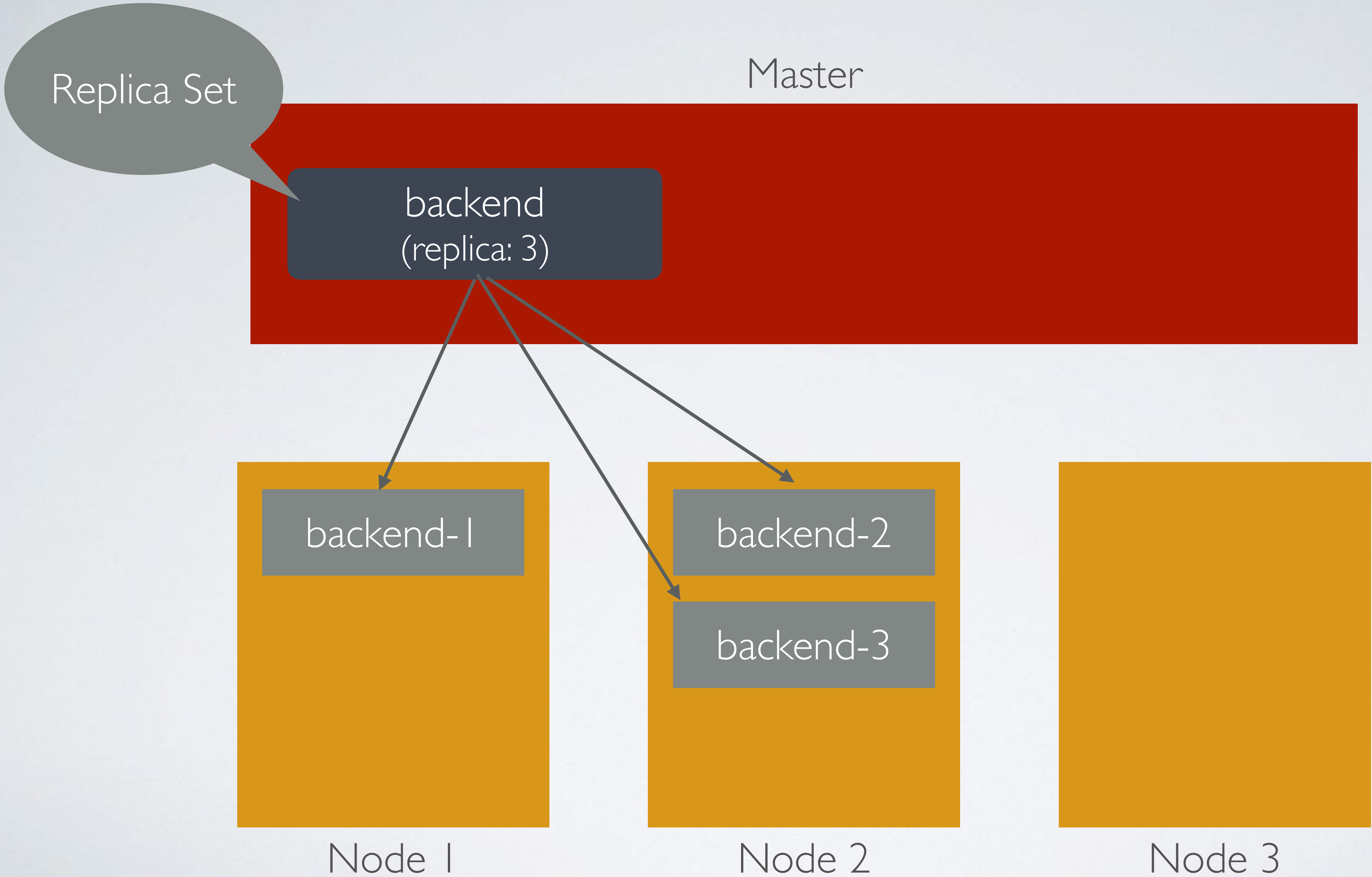


~~Replication Controller (rc)~~

# Replica Sets (rs)

the next-generation Replication Controller

ensures that a specified number of pod “replicas” are running at any given time



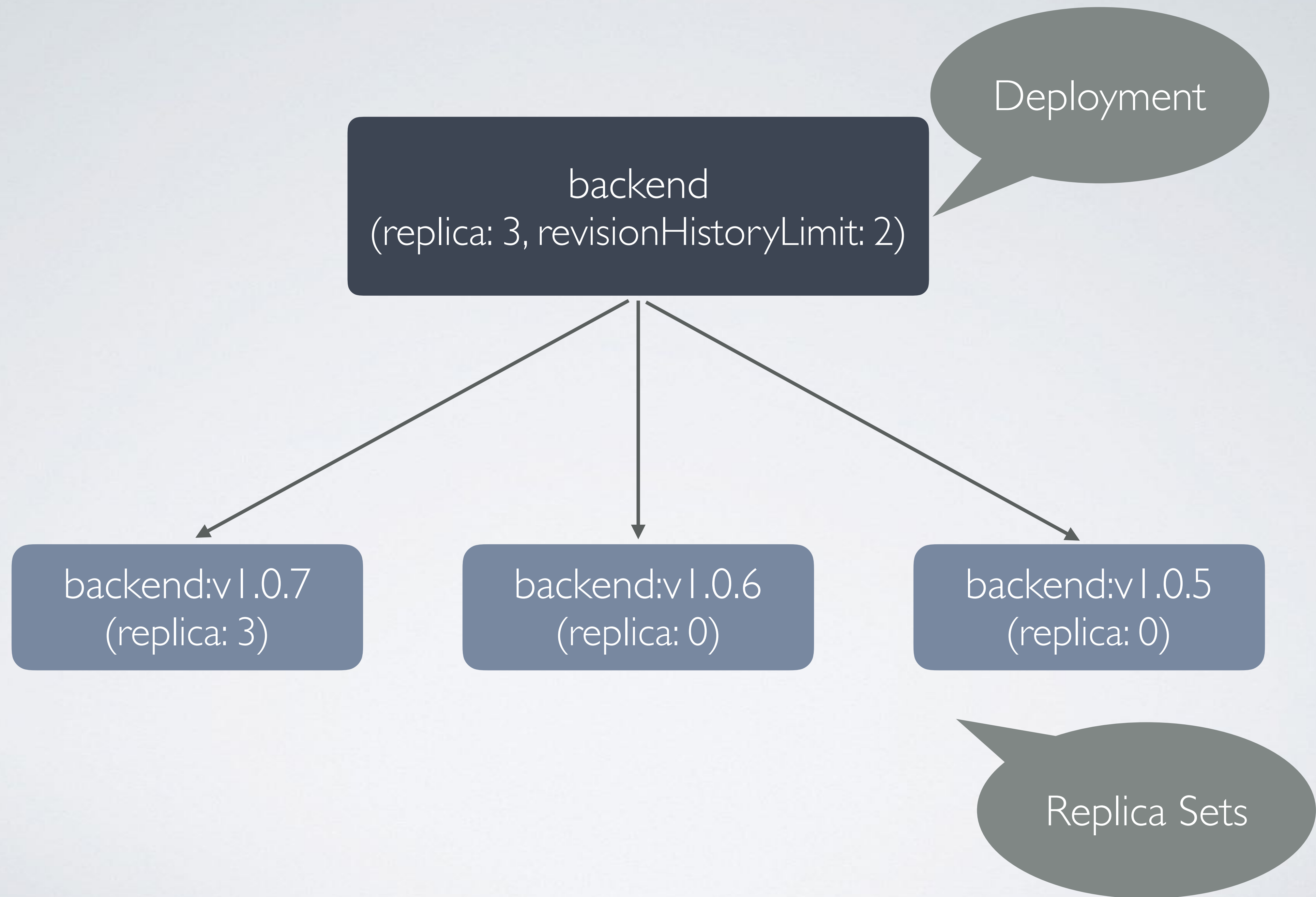
```
kind: ReplicaSet
apiVersion: apps/v1beta1
metadata:
  name: echoserver
spec:
  replicas: 3
```

```
  template:
    metadata:
      labels:
        app: echoserver
    spec:
      containers:
      - name: echoserver
        image: gcr.io/google-containers/echoserver:1.6
        ports:
        - containerPort: 8080
```

Pod

# Deployments (deploy)

provides declarative updates for Pods and ReplicaSets



# Update Strategy



Default

- RollingUpdate — updates one pod at a time
- Max Unavailable — maximum number of Pods that can be unavailable during the update process
- Max Surge — maximum number of Pods that can be created above the desired number of Pods
- Recreate — All existing Pods are killed before new ones are created



```
kind: Deployment
apiVersion: apps/v1beta1
metadata:
  name: echoserver
spec:
  replicas: 3
  revisionHistoryLimit: 2
  template:
    metadata:
      labels:
        app: echoserver
    spec:
      containers:
      - name: echoserver
        image: gcr.io/google-containers/echoserver:1.1
        ports:
        - containerPort: 8080
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxUnavailable: 1
      maxSurge: 1
```

```
$ kubectl create -f deployment.yaml --record=true  
deployment "echoserver" created
```

```
$ kubectl set image deployment/echoserver echoserver=gcr.io/google-containers/echoserver:1.2  
deployment "echoserver" image updated
```

```
$ kubectl rollout status deployment/echoserver  
Waiting for rollout to finish: 1 out of 3 new replicas have been updated...  
Waiting for rollout to finish: 1 out of 3 new replicas have been updated...  
Waiting for rollout to finish: 2 out of 3 new replicas have been updated...  
Waiting for rollout to finish: 2 out of 3 new replicas have been updated...  
Waiting for rollout to finish: 2 out of 3 new replicas have been updated...  
Waiting for rollout to finish: 1 old replicas are pending termination...  
Waiting for rollout to finish: 1 old replicas are pending termination...  
deployment "echoserver" successfully rolled out
```



```
$ kubectl set image deployment/echoserver echoserver=gcr.io/google-containers/echoserver:1.3  
deployment "echoserver" image updated
```

```
$ kubectl rollout history deployment/echoserver  
deployments "echoserver"
```

REVISION	CHANGE-CAUSE
1	kubectl create --filename=deployment.yaml --record=true
2	kubectl set image deployment/echoserver echoserver=gcr.io/google-containers/echoserver:1.2
3	kubectl set image deployment/echoserver echoserver=gcr.io/google-containers/echoserver:1.3

```
$ kubectl rollout history deployment/echoserver --revision=2
deployments "echoserver" with revision #2
Pod Template:
  Labels:      app=echoserver
              pod-template-hash=1885346732
Annotations:  kubernetes.io/change-cause=kubectl set image deployment/echoserver echoserver=gcr.io/google-containers/echoserver:1.2
Containers:
  echoserver:
    Image:      gcr.io/google-containers/echoserver:1.2
    Port:      8080/TCP
    Environment: <none>
    Mounts:     <none>
Volumes:      <none>
```

```
$ kubectl rollout undo deployment/echoserver  
deployment "echoserver" rolled back
```

```
$ kubectl rollout history deployment/echoserver  
deployments "echoserver"
```

REVISION	CHANGE-CAUSE
1	kubectl create --filename=deployment.yaml --record=true
3	kubectl set image deployment/echoserver echoserver=gcr.io/google-containers/echoserver:1.3
4	kubectl set image deployment/echoserver echoserver=gcr.io/google-containers/echoserver:1.2

```
$ kubectl rollout undo deployment/echoserver --to-revision=1
deployment "echoserver" rolled back
```

```
$ kubectl rollout history deployment/echoserver
deployments "echoserver"
```

REVISION	CHANGE-CAUSE
3	kubectl set image deployment/echoserver echoserver=gcr.io/google-containers/echoserver:1.3
4	kubectl set image deployment/echoserver echoserver=gcr.io/google-containers/echoserver:1.2
5	kubectl create --filename=deployment.yaml --record=true

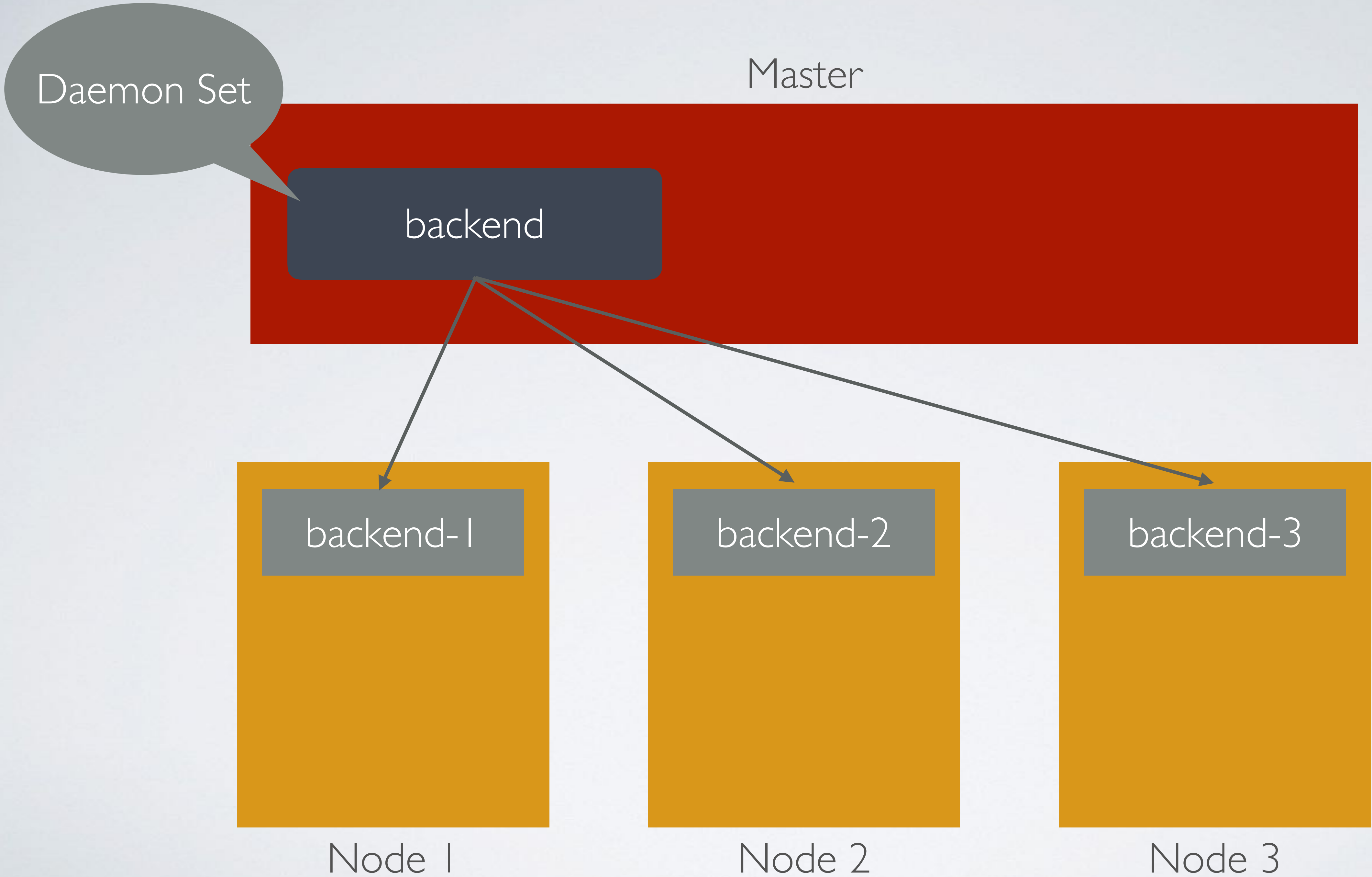
```
$ kubectl scale deployment/echoserver --replicas 6  
deployment "echoserver" scaled
```

```
$ kubectl get deployment/echoserver
```

NAME	DESIRED	CURRENT	UP-TO-DATE	AVAILABLE	AGE
echoserver	6	6	6	6	11m

# Daemon Sets (ds)

ensures that all (or some) nodes run a copy of a pod



```
kind: DaemonSet
apiVersion: extensions/v1beta1
metadata:
  name: echoserver
spec:
  template:
    metadata:
      labels:
        app: echoserver
    spec:
      containers:
      - name: echoserver
        image: gcr.io/google-containers/echoserver:1.6
        ports:
        - containerPort: 8080
  updateStrategy:
    type: RollingUpdate
    rollingUpdate:
      maxUnavailable: 1
```



# Strategy



Default

- OnDelete — new DaemonSet pods will only be created when you manually delete old DaemonSet pods
- RollingUpdate

# Resource Quotas (quota)

limit aggregate resource consumption

```
kind: Deployment
apiVersion: apps/v1beta1
metadata:
  name: echoserver
spec:
  replicas: 3
  revisionHistoryLimit: 2
  template:
    metadata:
      labels:
        app: echoserver
    spec:
      containers:
      - name: echoserver
        image: gcr.io/google-containers/echoserver:1.6
        ports:
        - containerPort: 8080
        resources:
          requests:
            cpu: 200m
            memory: 300Mi
          limits:
            cpu: 1
            memory: 1Gi
```



1 CPU = 1000m

# Health Check

# Health Check

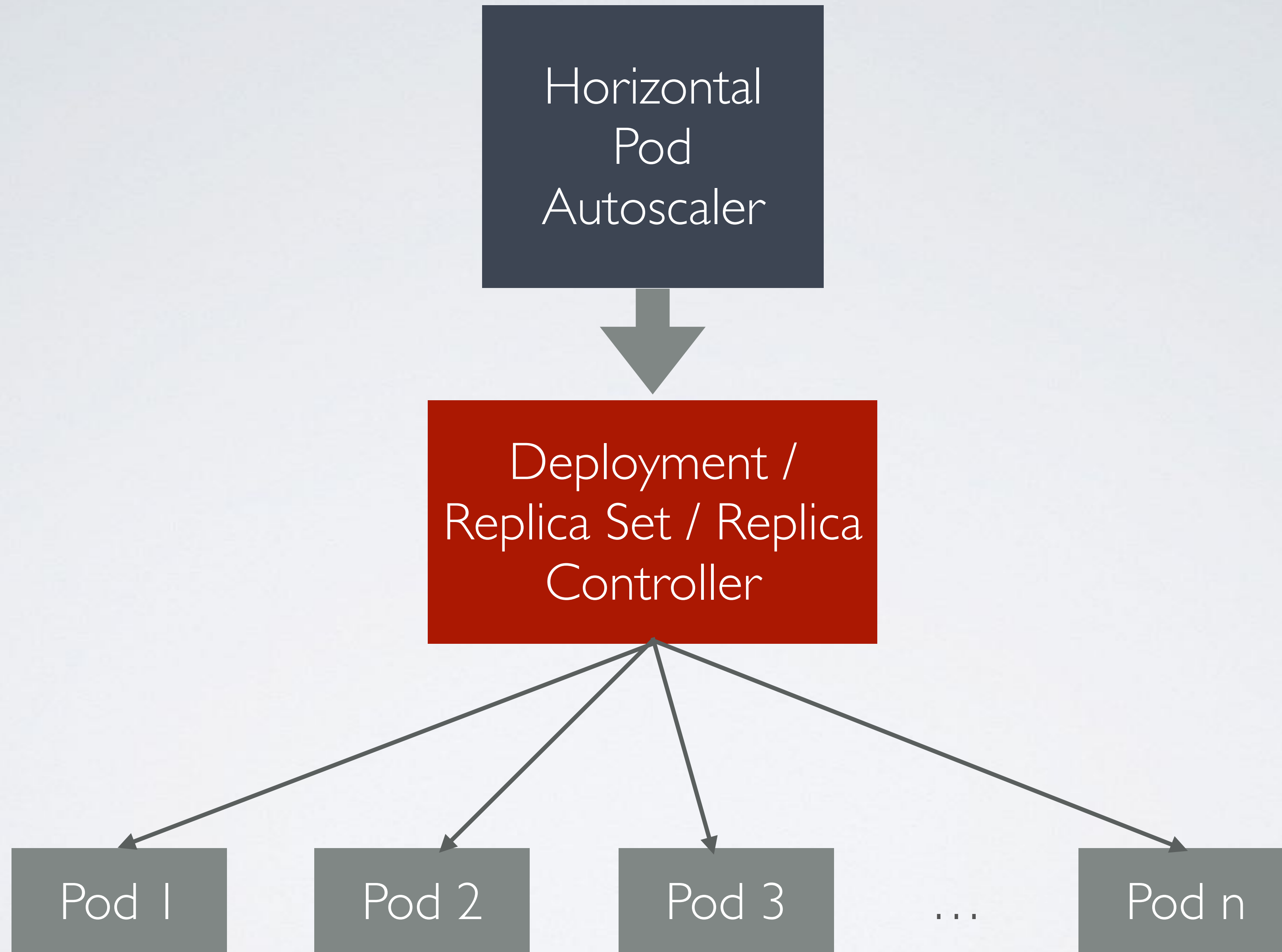
- Liveness Probes — know when to restart a Container
- Readiness Probes — don't send requests until application started

```
kind: Deployment
apiVersion: app/v1beta1
metadata:
  name: default-http-backend
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: default-http-backend
    spec:
      containers:
      - name: default-http-backend
        image: gcr.io/google-containers/defaultbackend:1.3
        ports:
        - containerPort: 8080
        readinessProbe:
          httpGet:
            path: /healthz
            port: 8080
            scheme: HTTP
          initialDelaySeconds: 30
          timeoutSeconds: 5
          periodSeconds: 10
          successThreshold: 1
          failureThreshold: 3
        livenessProbe:
          httpGet:
            path: /healthz
            port: 8080
            scheme: HTTP
          initialDelaySeconds: 30
          timeoutSeconds: 5
          periodSeconds: 10
          successThreshold: 1
          failureThreshold: 3
```

# Horizontal Pod Autoscaler (hpa)

automatically scales the number of pods in  
a replication controller, deployment or replica set







```
kind: Deployment
apiVersion: apps/v1beta1
metadata:
  name: hpa-example
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: hpa-example
    spec:
      containers:
      - name: hpa-example
        image: gcr.io/google-containers/hpa-example
        ports:
        - containerPort: 80
        resources:
          requests:
            cpu: 100m
```

```
apiVersion: v1
kind: Service
metadata:
  name: hpa-example
spec:
  selector:
    app: hpa-example
  ports:
    - port: 80
```

```
kind: HorizontalPodAutoscaler
apiVersion: autoscaling/v1
metadata:
  name: hpa-example
spec:
  scaleTargetRef:
    apiVersion: apps/v1beta1
    kind: Deployment
    name: hpa-example
  minReplicas: 1
  maxReplicas: 6
  targetCPUUtilizationPercentage: 50
```



50% of  
request

\$ kubectl get hpa --watch

NAME	REFERENCE	TARGETS	MINPODS	MAXPODS	REPLICAS	AGE
hpa-example	Deployment/hpa-example	0% / 50%	1	6	1	12m
hpa-example	Deployment/hpa-example	522% / 50%	1	6	1	12m
hpa-example	Deployment/hpa-example	522% / 50%	1	6	1	12m
hpa-example	Deployment/hpa-example	941% / 50%	1	6	1	13m
hpa-example	Deployment/hpa-example	941% / 50%	1	6	4	13m
hpa-example	Deployment/hpa-example	362% / 50%	1	6	4	14m
hpa-example	Deployment/hpa-example	362% / 50%	1	6	4	14m
hpa-example	Deployment/hpa-example	12% / 50%	1	6	4	15m
hpa-example	Deployment/hpa-example	12% / 50%	1	6	4	15m
hpa-example	Deployment/hpa-example	0% / 50%	1	6	4	16m

# Auto-scale Node on Container Engine

```
$ gcloud alpha container clusters update cluster-1 \  
  --enable-autoscaling \  
  --min-nodes=2 \  
  --max-nodes=6 \  
  --zone=asia-southeast1-b \  
  --node-pool=default-pool \  
  --project=acoshift-k8s
```



Name	default-pool
Current size	<input type="text" value="1"/>
Node version	1.7.0
Node image	Container-Optimized OS (cos)
Machine type	n1-standard-1 (1 vCPU, 3.75 GB memory)
Total cores	1 vCPU
Total memory	3.75 GB
Automatic node upgrades	Disabled
Automatic node repair	Disabled

Automatic node upgrades (beta) ?

Automatic node repair (beta) ?

Autoscaling (beta) ?

Minimal size

Maximal size

Preemptible nodes	Disabled
Boot disk size in GB (per node)	100
Local SSD disks (per node)	0
Instance groups	<a href="#">gke-cluster-1-default-pool-73cdab92-grp</a>

# Labels

key/value pairs that are attached to objects



```
kind: Deployment
apiVersion: apps/v1beta1
metadata:
  name: echoserver
  labels:
    app: echoserver
spec:
  template:
    metadata:
      labels:
        app: echoserver
    spec:
      containers:
      - name: echoserver
        image: gcr.io/google-containers/echoserver:1.6
        ports:
        - containerPort: 8080
```

---

```
kind: Service
apiVersion: v1
metadata:
  name: echoserver
  labels:
    app: echoserver
spec:
  selector:
    app: echoserver
  ports:
  - port: 80
    targetPort: 8080
```

\$ kubectl get all -l app=echoserver

NAME	READY	STATUS	RESTARTS	AGE
po/echoserver-3345770719-c5q61	1/1	Running	0	10s

NAME	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
svc/echoserver	10.3.240.126	<none>	80/TCP	9s

NAME	DESIRED	CURRENT	UP-TO-DATE	AVAILABLE	AGE
deploy/echoserver	1	1	1	1	10s

NAME	DESIRED	CURRENT	READY	AGE
rs/echoserver-3345770719	1	1	1	10s

# Node Selector

```
$ kubectl label no node-3 nodeType=highmem
```

```
$ kubectl get no --show-labels
```

```
kind: Deployment
apiVersion: apps/v1beta1
metadata:
  name: redis
spec:
  template:
    metadata:
      labels:
        app: redis
    spec:
      containers:
      - name: redis
        image: redis:3.2.9
        ports:
        - containerPort: 6379
      nodeSelector:
        nodeType: highmem
```

```
$ kubectl label no node-3 nodeType-
```

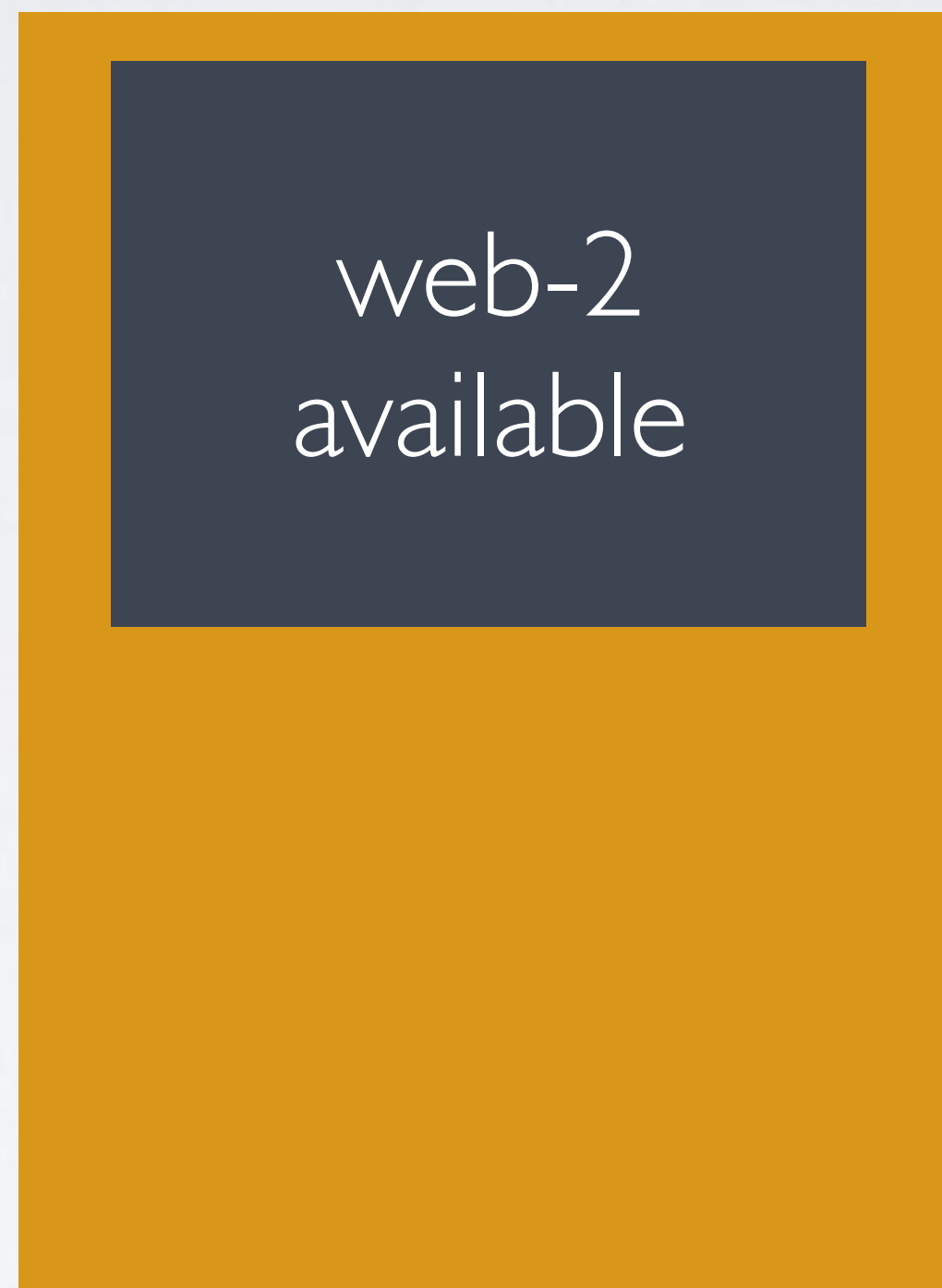
# Pod Disruption Budgets (pdb)

limits the number pods of a replicated application  
that are down simultaneously from voluntary disruptions

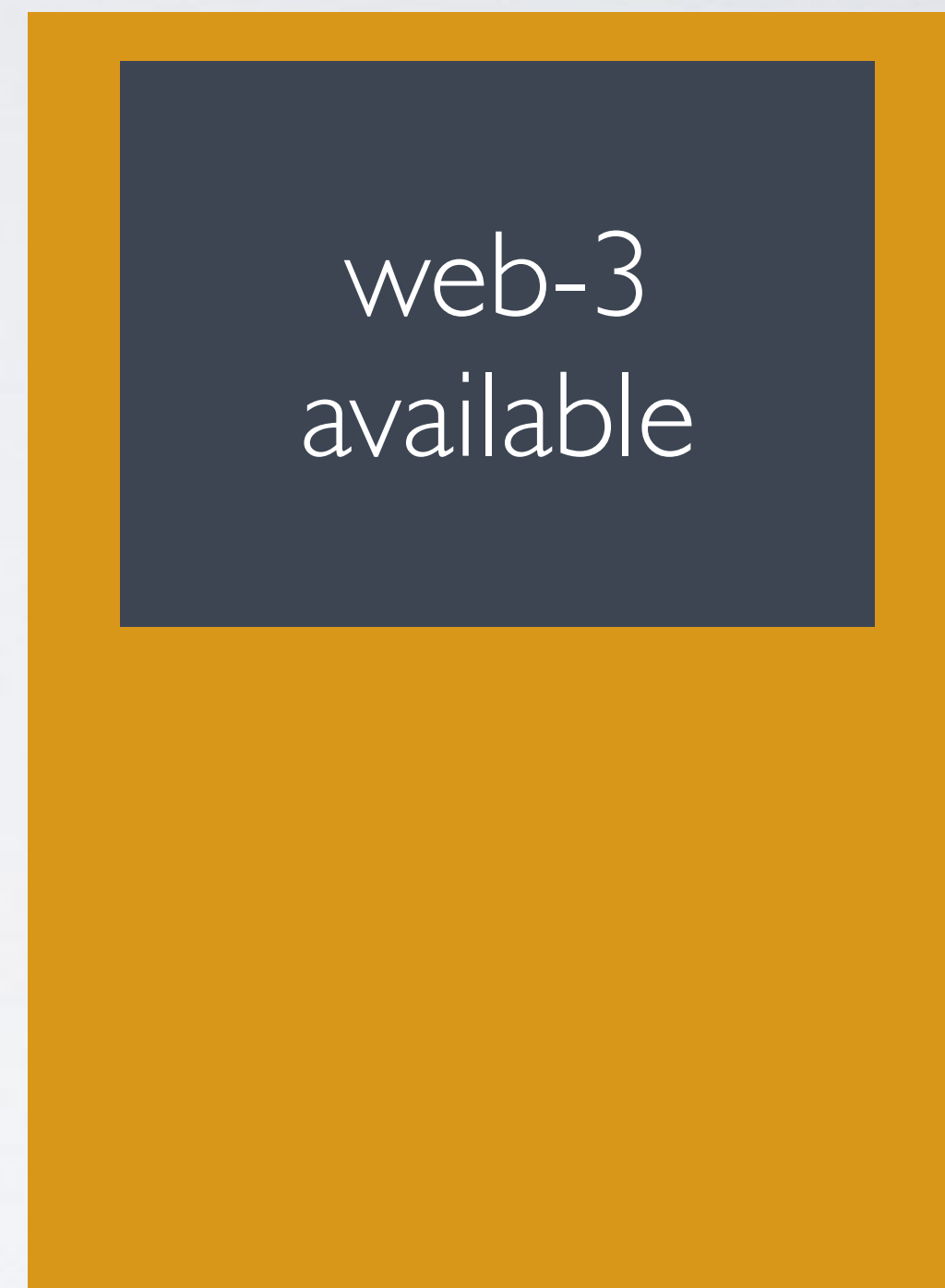
web: replicas=3, minAvailable=2



node-1



node-2



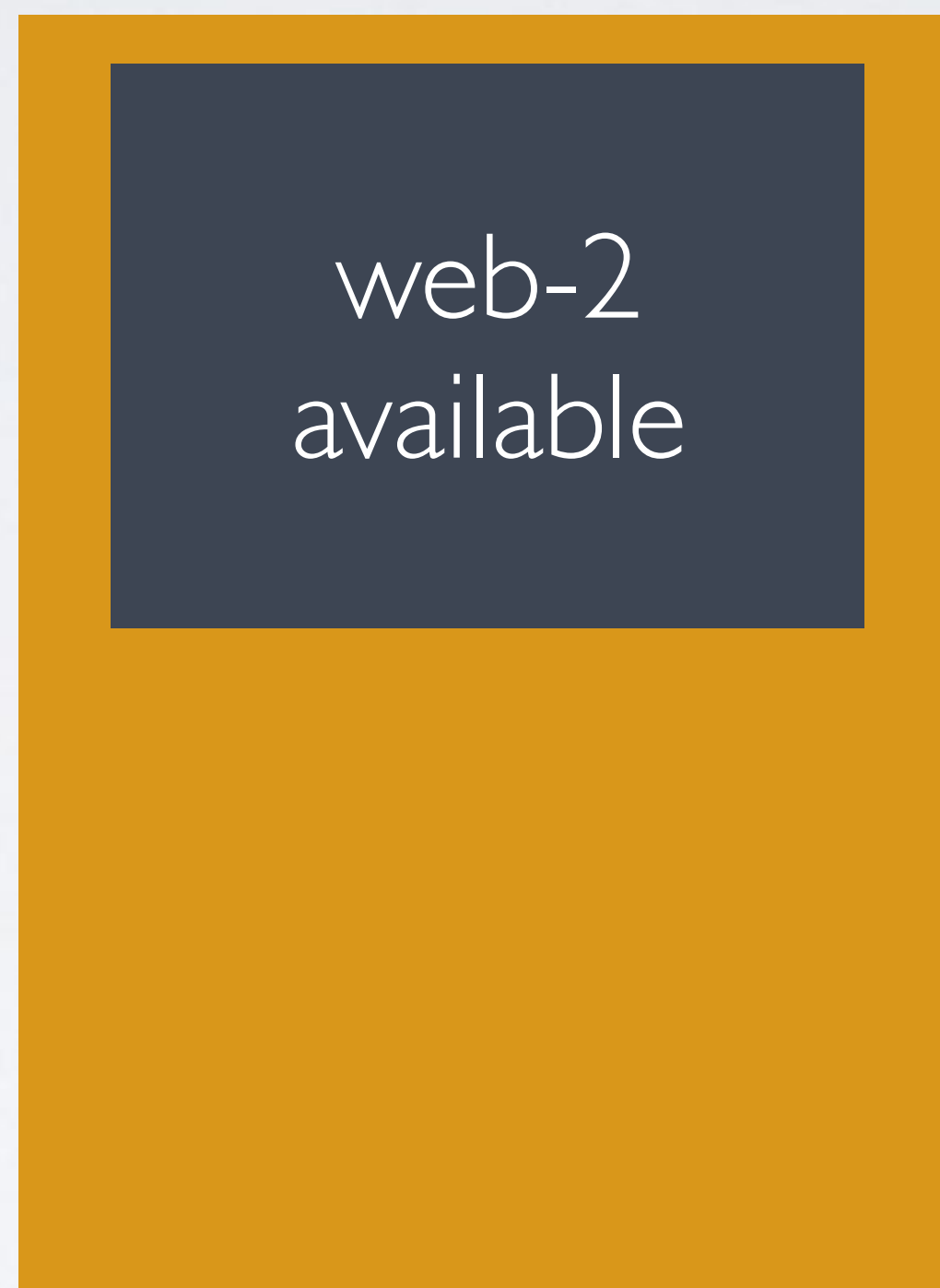
node-3



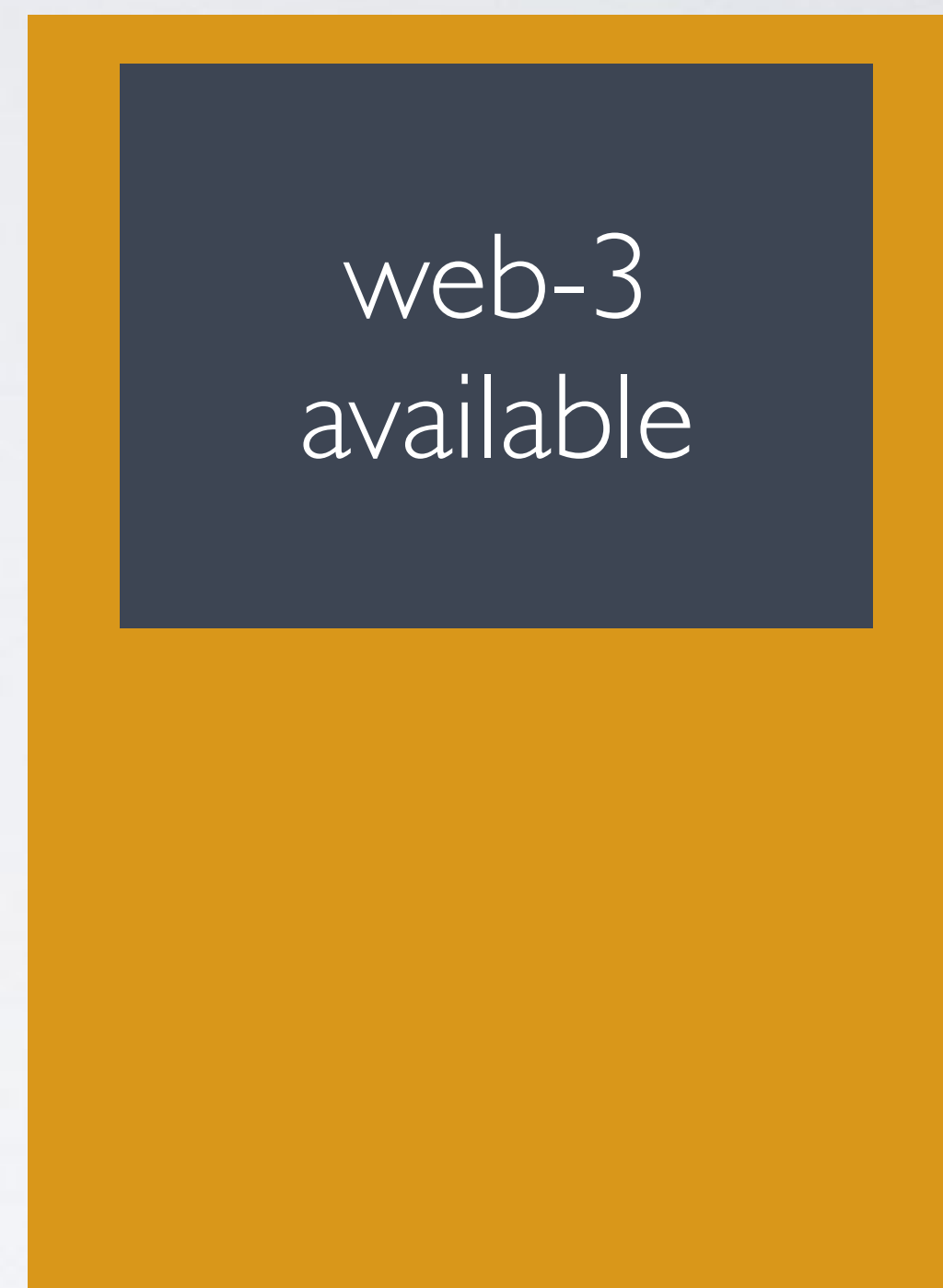
web: replicas=3, minAvailable=2



node-1  
draining



node-2



node-3

web: replicas=3, minAvailable=2



node-1  
draining



node-2



node-3

web: replicas=3, minAvailable=2



node-1  
drained



node-2



node-3

web: replicas=3, minAvailable=2



node-1  
drained



node-2



node-3

web: replicas=3, minAvailable=2



node-1  
drained



node-2  
draining



node-3

web: replicas=3, minAvailable=2

web-5  
pending



node-1  
drained



node-2  
drain blocked



node-3

# PodDisruptionBudget

- minAvailable
- maxUnavailable



percent/value



```
kind: Deployment
apiVersion: apps/v1beta1
metadata:
  name: echoserver
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: echoserver
    spec:
      containers:
      - name: echoserver
        image: gcr.io/google-containers/echoserver:1.6
        ports:
        - containerPort: 8080
        resources:
          requests:
            cpu: 250m
```

```
kind: PodDisruptionBudget
apiVersion: policy/v1beta1
metadata:
  name: echoserver
spec:
  minAvailable: 2
  # minAvailable: 67%
  selector:
    matchLabels:
      app: echoserver
```



```
$ kubectl get no
NAME                                     STATUS      AGE      VERSION
gke-cluster-2-default-pool-75546f17-39fq Ready      10m      v1.7.0
gke-cluster-2-default-pool-75546f17-9dc5 Ready      10m      v1.7.0
gke-cluster-2-default-pool-75546f17-xr57 Ready      10m      v1.7.0

$ kubectl get po
NAME                                     READY      STATUS      RESTARTS   AGE
echoserver-1994896057-4lk6s           1/1       Running    0          50s
echoserver-1994896057-f0rft           1/1       Running    0          51s
echoserver-1994896057-p0lpr           1/1       Running    0          42s

$ kubectl drain gke-cluster-2-default-pool-75546f17-39fq --force --ignore-daemonsets
node "gke-cluster-2-default-pool-75546f17-39fq" already cordoned
pod "kube-dns-autoscaler-3880103346-4qj3t" evicted
pod "kube-dns-1413379277-h96k0" evicted
node "gke-cluster-2-default-pool-75546f17-39fq" drained

$ kubectl get no
NAME                                     STATUS      AGE      VERSION
gke-cluster-2-default-pool-75546f17-39fq Ready,SchedulingDisabled 16m      v1.7.0
gke-cluster-2-default-pool-75546f17-9dc5 Ready      16m      v1.7.0
gke-cluster-2-default-pool-75546f17-xr57 Ready      16m      v1.7.0
```

```
$ kubectl drain gke-cluster-2-default-pool-75546f17-9dc5 --force --ignore-daemonsets
node "gke-cluster-2-default-pool-75546f17-9dc5" already cordoned
pod "heapster-v1.4.0-2764992688-5xp57" evicted
pod "kubernetes-dashboard-1962351010-2xtq1" evicted
pod "echoserver-1994896057-f0rft" evicted
pod "l7-default-backend-2954409777-d4x3g" evicted
pod "event-exporter-v0.1.4-1771975458-s86dg" evicted
# hang
pod "echoserver-1994896057-p0lpr" evicted
node "gke-cluster-2-default-pool-75546f17-9dc5" drained
```

```
$ kubectl get no
NAME                                     STATUS                                AGE      VERSION
gke-cluster-2-default-pool-75546f17-39fq Ready,SchedulingDisabled             18m      v1.7.0
gke-cluster-2-default-pool-75546f17-9dc5 Ready,SchedulingDisabled             18m      v1.7.0
gke-cluster-2-default-pool-75546f17-xr57 Ready                                18m      v1.7.0
```

```
$ kubectl get po
NAME                                     READY    STATUS    RESTARTS   AGE
echoserver-1994896057-1tg93            0/1     Pending   0           1m
echoserver-1994896057-1w5s7            1/1     Running   0           1m
echoserver-1994896057-4lk6s            1/1     Running   0           7m
```

```
$ kubectl describe po/echoserver-1994896057-1tg93
...
Events:
  FirstSeen  LastSeen  Count  From              SubObjectPath  Type            Reason           Message
  -----
  1m         1m        5      default-scheduler Warning         FailedScheduling  No nodes are available that match all of the
following predicates:: Insufficient cpu (1), PodToleratesNodeTaints (1).
...
```

```
$ kubectl uncordon gke-cluster-2-default-pool-75546f17-39fq
node "gke-cluster-2-default-pool-75546f17-39fq" uncordoned
```

```
$ kubectl get no
NAME                                     STATUS                                AGE      VERSION
gke-cluster-2-default-pool-75546f17-39fq Ready                                22m      v1.7.0
gke-cluster-2-default-pool-75546f17-9dc5 Ready,SchedulingDisabled            22m      v1.7.0
gke-cluster-2-default-pool-75546f17-xr57 Ready                                22m      v1.7.0
```

```
$ kubectl get po
NAME                                READY   STATUS    RESTARTS   AGE
echoserver-1994896057-1tg93        1/1     Running   0          4m
echoserver-1994896057-1w5s7        1/1     Running   0          4m
echoserver-1994896057-4lk6s        1/1     Running   0          10m
```

```
$ kubectl uncordon gke-cluster-2-default-pool-75546f17-9dc5
node "gke-cluster-2-default-pool-75546f17-9dc5" uncordoned
```

```
$ kubectl get no
NAME                                     STATUS    AGE      VERSION
gke-cluster-2-default-pool-75546f17-39fq Ready     25m      v1.7.0
gke-cluster-2-default-pool-75546f17-9dc5 Ready     25m      v1.7.0
gke-cluster-2-default-pool-75546f17-xr57 Ready     25m      v1.7.0
```

# GCE Persistent Disks

# Create Persistent Disk (pd) on GCP

```
$ gcloud compute disks create --size=20GB --zone=asia-southeast1-b --project=acoshift-k8s mysql-disk
Created [https://www.googleapis.com/compute/v1/projects/acoshift-k8s/zones/asia-southeast1-b/disks/mysql-disk].
```

NAME	ZONE	SIZE_GB	TYPE	STATUS
mysql-disk	asia-southeast1-b	20	pd-standard	READY

```
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: mysql
spec:
  replicas: 1
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
      - name: mysql
        env:
          - name: MYSQL_ROOT_PASSWORD
            value: mysqlpassword1234
        image: mysql:5.6.36
        imagePullPolicy: IfNotPresent
        ports:
          - containerPort: 3306
        volumeMounts:
          - mountPath: /var/lib/mysql
            name: mysql-disk
      volumes:
      - name: mysql-disk
        gcePersistentDisk:
          pdName: mysql-disk
          fsType: ext4
```



```
$ kubectl create -f pd.yaml  
deployment "mysql" created
```

```
$ kubectl get po
```

NAME	READY	STATUS	RESTARTS	AGE
mysql-1398320157-mgf6c	1/1	Running	0	3m

```
$ kubectl port-forward mysql-1398320157-mgf6c 3306:3306
```

```
$ mysql -u root -p -h 127.0.0.1
```

```
mysql> create database db1;  
Query OK, 1 row affected (0.05 sec)
```

```
mysql> use db1;  
Database changed
```

```
mysql> create table users (  
    -> id int auto_increment,  
    -> name varchar(255) not null,  
    -> created_at timestamp not null default now(),  
    -> primary key (id)  
    -> );  
Query OK, 0 rows affected (0.08 sec)
```

```
mysql> insert into users (name) values ('acoshift'), ('user1'), ('user2');  
Query OK, 3 rows affected (0.08 sec)  
Records: 3  Duplicates: 0  Warnings: 0
```

```
mysql> select * from users;
```

id	name	created_at
1	acoshift	2017-07-15 14:46:04
2	user1	2017-07-15 14:46:04
3	user2	2017-07-15 14:46:04

```
3 rows in set (0.03 sec)
```

```
mysql> exit  
Bye
```



```
$ kubectl get po
```

NAME	READY	STATUS	RESTARTS	AGE
mysql-1398320157-mgf6c	1/1	Running	0	19m

```
$ kubectl delete po/mysql-1398320157-mgf6c  
pod "mysql-1398320157-mgf6c" deleted
```

```
$ kubectl get po
```

NAME	READY	STATUS	RESTARTS	AGE
mysql-1398320157-d0scs	0/1	ContainerCreating	0	30s

Pods



Name

Status

Restarts

Age

CPU (cores)

Memory (bytes)



mysql-1398320157-d0scs

Waiting: Containe...

0

41 seconds

-

-



AttachVolume.Attach failed for volume "mysql-disk" : googleapi: Error 400: The disk resource 'projects/acoshift-k8s/zones/asia-southeast1-b/disks/mysql-disk' is already being used by 'projects/acoshift-k8s/zones/asia-southeast1-b/instances/gke-cluster-1-default-pool-73cdab92-hhk2'

```
$ kubectl get po
```

NAME	READY	STATUS	RESTARTS	AGE
mysql-1398320157-d0scs	1/1	Running	0	6m

```
$ kubectl port-forward mysql-1398320157-d0scs 3306:3306
```

```
Forwarding from 127.0.0.1:3306 -> 3306
```

```
Forwarding from [::1]:3306 -> 3306
```

```
$ mysql -u root -p -h 127.0.0.1
```

```
mysql> use db1;
```

```
Database changed
```

```
mysql> select * from users;
```

id	name	created_at
1	acoshift	2017-07-15 14:46:04
2	user1	2017-07-15 14:46:04
3	user2	2017-07-15 14:46:04

```
3 rows in set (0.04 sec)
```

```
mysql> exit
```

```
Bye
```

```
$ kubectl delete -f pd.yaml  
deployment "mysql" deleted
```

```
$ gcloud compute disks delete --zone=asia-southeast1-b --project=acoshift-k8s mysql-disk  
The following disks will be deleted:  
- [mysql-disk] in [asia-southeast1-b]
```

```
Do you want to continue (Y/n)? Y
```

```
Deleted [https://www.googleapis.com/compute/v1/projects/acoshift-k8s/zones/asia-southeast1-b/disks/mysql-disk].
```

Q&A

DAY 2

# Persistent Volumes (pv)

a piece of storage in the cluster that has been provisioned  
by an administrator

# Persistent Volume Claims (pvc)

a request for storage by a user

# Storage Classes

a way for administrators to describe the “classes” of storage they offer



# Provisioning

- Static
- Dynamic



```
$ kubectl get storageclass
```

NAME	TYPE
standard (default)	kubernetes.io/gce-pd

```
$ kubectl describe storageclass standard
```

Name: standard

IsDefaultClass: Yes

Annotations: storageclass.beta.kubernetes.io/is-default-class=true

Provisioner: kubernetes.io/gce-pd

Parameters: type=pd-standard

Events: <none>

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: ssd
provisioner: kubernetes.io/gce-pd
parameters:
  # type: pd-standard
  type: pd-ssd
  zone: asia-southeast1-b
```

```
$ kubectl create -f 01-storageclass.yaml  
storageclass "ssd" created
```

```
$ kubectl get storageclass
```

NAME	TYPE
ssd	kubernetes.io/gce-pd
standard (default)	kubernetes.io/gce-pd

# Access Modes

- `ReadWriteOnce` — the volume can be mounted as read-write by a single node
- `ReadOnlyMany` — the volume can be mounted read-only by many nodes
- `ReadWriteMany` — the volume can be mounted as read-write by many nodes

Volume Plugin	ReadWriteOnce	ReadOnlyMany	ReadWriteMany
AWSElasticBlockStore	✓	-	-
AzureFile	✓	✓	✓
AzureDisk	✓	-	-
CephFS	✓	✓	✓
Cinder	✓	-	-
FC	✓	✓	-
FlexVolume	✓	✓	-
Flocker	✓	-	-
GCEPersistentDisk	✓	✓	-
Glusterfs	✓	✓	✓
HostPath	✓	-	-
iSCSI	✓	✓	-
PhotonPersistentDisk	✓	-	-
Quobyte	✓	✓	✓
NFS	✓	✓	✓
RBD	✓	✓	-
VsphereVolume	✓	-	-
PortworxVolume	✓	-	✓
ScaleIO	✓	✓	-
StorageOS	✓	-	-

<https://kubernetes.io/docs/concepts/storage/persistent-volumes/>

# Reclaim Policy

- Retain

Default for static  
provisioning

- Recycle

Default for dynamic  
provisioning

- Delete

```
kind: PersistentVolume
apiVersion: v1
metadata:
  name: disk-1
  annotations:
    volume.beta.kubernetes.io/mount-options: discard
spec:
  storageClassName: standard
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteOnce
  gcePersistentDisk:
    fsType: ext4
    pdName: disk-1
```

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: mysql-pvc
spec:
  storageClassName: standard
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 5Gi
```



```
$ gcloud compute disks create --size=10GB --zone=asia-southeast1-b --project=acoshift-k8s disk-1
Created [https://www.googleapis.com/compute/v1/projects/acoshift-k8s/zones/asia-southeast1-b/disks/disk-1].
```

NAME	ZONE	SIZE_GB	TYPE	STATUS
disk-1	asia-southeast1-b	10	pd-standard	READY

```
$ kubectl create -f 02-pv.yaml
persistentvolume "disk-1" created
```

```
$ kubectl get pv
```

NAME	CAPACITY	ACCESSMODES	RECLAIMPOLICY	STATUS	CLAIM	STORAGECLASS	REASON	AGE
disk-1	10Gi	RWO	Retain	Available				12s

```
$ kubectl create -f 02-pvc.yaml
persistentvolumeclaim "mysql-pvc" created
```

```
$ kubectl get pv
```

NAME	CAPACITY	ACCESSMODES	RECLAIMPOLICY	STATUS	CLAIM	STORAGECLASS	REASON	AGE
disk-1	10Gi	RWO	Retain	Bound	default/mysql-pvc	standard		1m

```
$ kubectl get pvc
```

NAME	STATUS	VOLUME	CAPACITY	ACCESSMODES	STORAGECLASS	AGE
mysql-pvc	Bound	disk-1	10Gi	RWO	standard	34s

```
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: mysql
spec:
  replicas: 1
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
      - name: mysql
        env:
          - name: MYSQL_ROOT_PASSWORD
            value: mysqlpassword1234
        image: mysql:5.6.36
        imagePullPolicy: IfNotPresent
        ports:
          - containerPort: 3306
        volumeMounts:
          - mountPath: /var/lib/mysql
            name: mysql-disk
      volumes:
      - name: mysql-disk
        persistentVolumeClaim:
          claimName: mysql-pvc
```

```
$ kubectl create -f 02-mysql.yaml  
deployment "mysql" created
```

```
$ kubectl get po
```

NAME	READY	STATUS	RESTARTS	AGE
mysql-68058648-d3m8l	1/1	Running	0	48s

```
$ kubectl delete po/mysql-68058648-d3m8l  
pod "mysql-68058648-d3m8l" deleted
```

```
$ kubectl get po
```

NAME	READY	STATUS	RESTARTS	AGE
mysql-68058648-rsqk1	1/1	Running	0	4s

```
$ kubectl delete deploy/mysql pvc/mysql-pvc pv/disk-1  
deployment "mysql" deleted  
persistentvolumeclaim "mysql-pvc" deleted  
persistentvolume "disk-1" deleted
```

```
$ gcloud compute disks delete --zone=asia-southeast1-b --project=acoshift-k8s disk-1  
The following disks will be deleted:  
- [disk-1] in [asia-southeast1-b]
```

```
Do you want to continue (Y/n)? Y
```

```
Deleted [https://www.googleapis.com/compute/v1/projects/acoshift-k8s/zones/asia-southeast1-b/disks/disk-1].
```

# Stateful Sets

provides guarantees about the ordering of deployment and scaling

# Stateful Sets

- Stable, unique network identifier
- Stable, persistent storage
- Ordered, graceful deployment and scaling
- Ordered, graceful deletion and termination
- Ordered, automated rolling updates

```
$ kubectl create -f https://raw.githubusercontent.com/cockroachdb/cockroach/master/cloud/kubernetes/cockroachdb-statefulset.yaml
service "cockroachdb-public" created
service "cockroachdb" created
poddisruptionbudget "cockroachdb-budget" created
statefulset "cockroachdb" created
```

```
$ kubectl get po
```

NAME	READY	STATUS	RESTARTS	AGE
cockroachdb-0	1/1	Running	0	10m
cockroachdb-1	1/1	Running	0	9m
cockroachdb-2	1/1	Running	0	8m

```
$ kubectl port-forward cockroachdb-0 8080
```

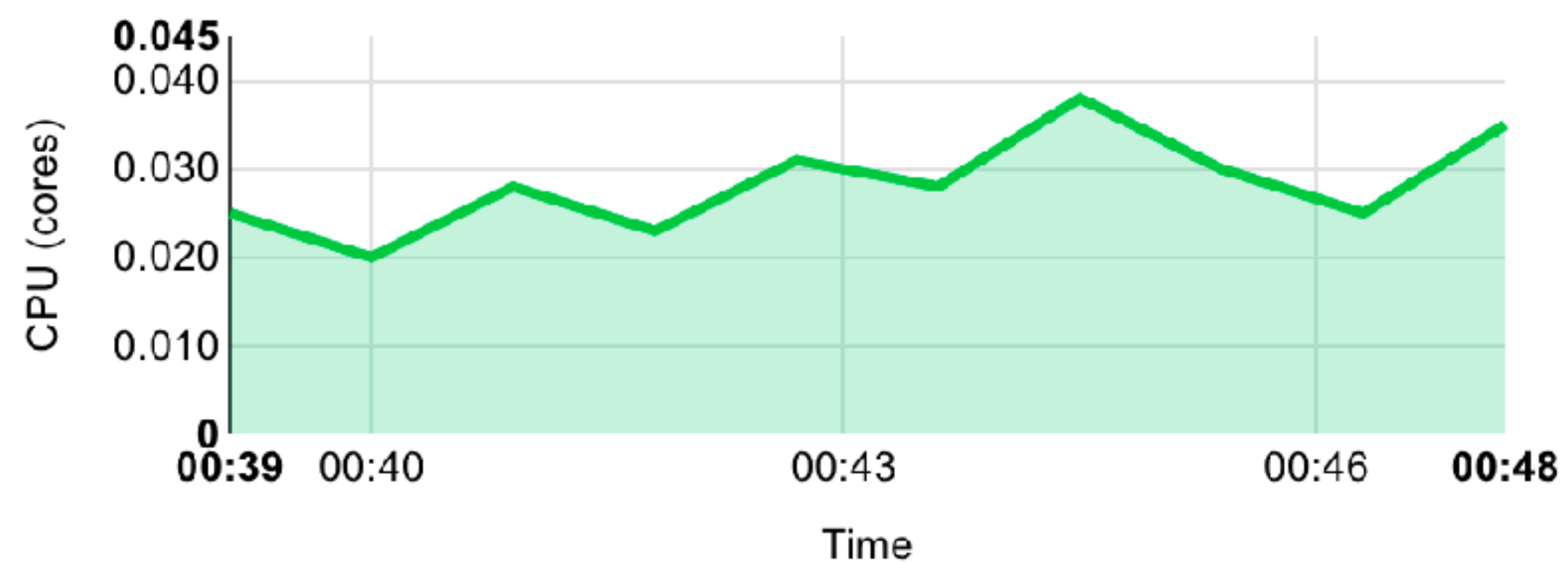


## Live Nodes

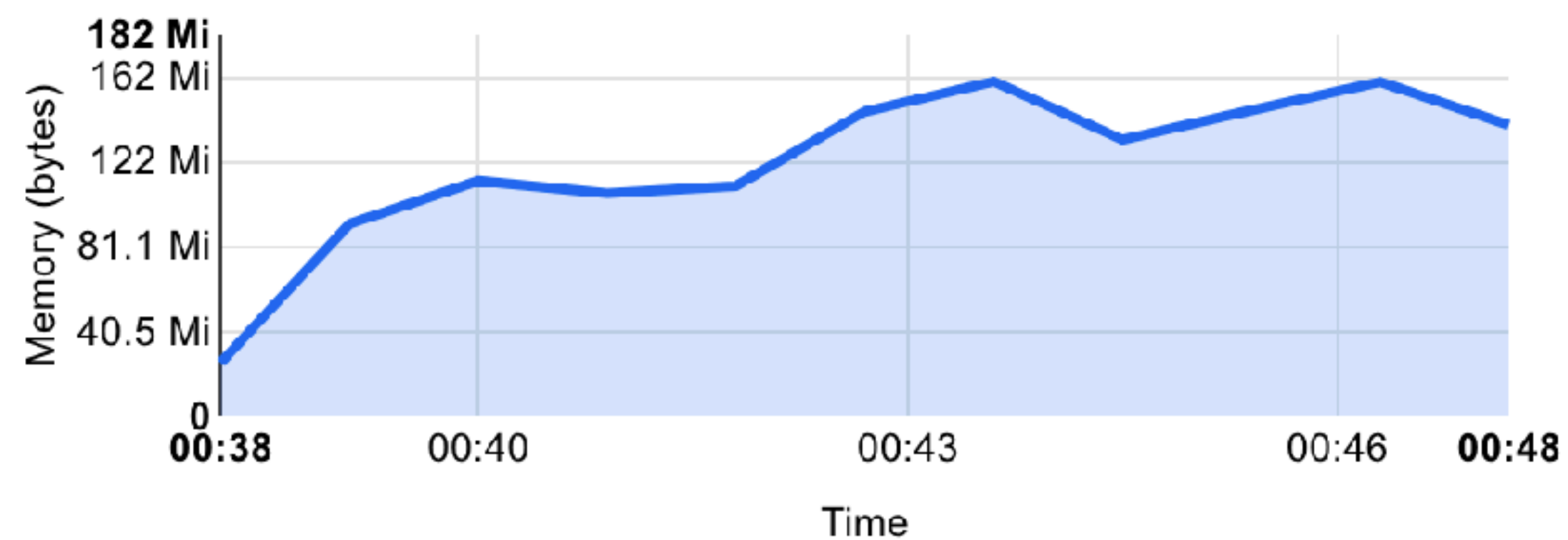
ID ▼	ADDRESS ▼	UPTIME ▼	BYTES ▼	REPLICAS ▼	MEM USAGE ▼	LOGS
1	<ul style="list-style-type: none"><li>cockroachdb-0.cockroachdb.default.svc.cluster.local:26257</li></ul>	10 minutes	3.2 MiB	10	95.6 MiB	<a href="#">Logs</a>
2	<ul style="list-style-type: none"><li>cockroachdb-1.cockroachdb.default.svc.cluster.local:26257</li></ul>	9 minutes	3.3 MiB	10	71.5 MiB	<a href="#">Logs</a>
3	<ul style="list-style-type: none"><li>cockroachdb-2.cockroachdb.default.svc.cluster.local:26257</li></ul>	9 minutes	3.3 MiB	10	70.4 MiB	<a href="#">Logs</a>









## CPU usage



## Memory usage ⓘ



## Pods

Name ↕	Status ↕	Restarts	Age ↕	CPU (cores)	Memory (bytes)		
✓ cockroachdb-4	Running	0	52 seconds	-	-	≡	⋮
✓ cockroachdb-3	Running	0	a minute	-	-	≡	⋮
✓ cockroachdb-2	Running	0	11 minutes	 0.009	 35.863 Mi	≡	⋮
✓ cockroachdb-1	Running	0	11 minutes	 0.011	 37.871 Mi	≡	⋮
✓ cockroachdb-0	Running	0	13 minutes	 0.015	 65.488 Mi	≡	⋮

# Live Nodes

ID ▾	ADDRESS ▾	UPTIME ▾	BYTES ▾	REPLICAS ▾	MEM USAGE ▾	LOGS
1	<ul style="list-style-type: none"><li>cockroachdb-0.cockroachdb.default.svc.cluster.local:26257</li></ul>	12 minutes	4.2 MiB	6	101.6 MiB	<a href="#">Logs</a>
2	<ul style="list-style-type: none"><li>cockroachdb-1.cockroachdb.default.svc.cluster.local:26257</li></ul>	11 minutes	84.4 KiB	7	73.5 MiB	<a href="#">Logs</a>
3	<ul style="list-style-type: none"><li>cockroachdb-2.cockroachdb.default.svc.cluster.local:26257</li></ul>	11 minutes	111.2 KiB	6	73.6 MiB	<a href="#">Logs</a>
4	<ul style="list-style-type: none"><li>cockroachdb-3.cockroachdb.default.svc.cluster.local:26257</li></ul>	a minute	4.1 MiB	5	60.4 MiB	<a href="#">Logs</a>
5	<ul style="list-style-type: none"><li>cockroachdb-4.cockroachdb.default.svc.cluster.local:26257</li></ul>	a minute	4.1 MiB	6	60.5 MiB	<a href="#">Logs</a>

```
$ kubectl run -it --rm cockroach-client --image=cockroachdb/cockroach --restart=Never --command -- ./cockroach sql --host cockroachdb-public --insecure
```

```
root@cockroachdb-public:26257/> create database db1;
CREATE DATABASE
```

```
root@cockroachdb-public:26257/> set database = db1;
SET
```

```
root@cockroachdb-public:26257/db1> create table users (
    -> id serial,
    -> name string not null default '',
    -> created_at timestamp not null default now(),
    -> primary key (id)
    -> );
```

CREATE TABLE

```
root@cockroachdb-public:26257/db1> insert into users (name) values ('acoshift'), ('user1'), ('user2');
INSERT 3
```

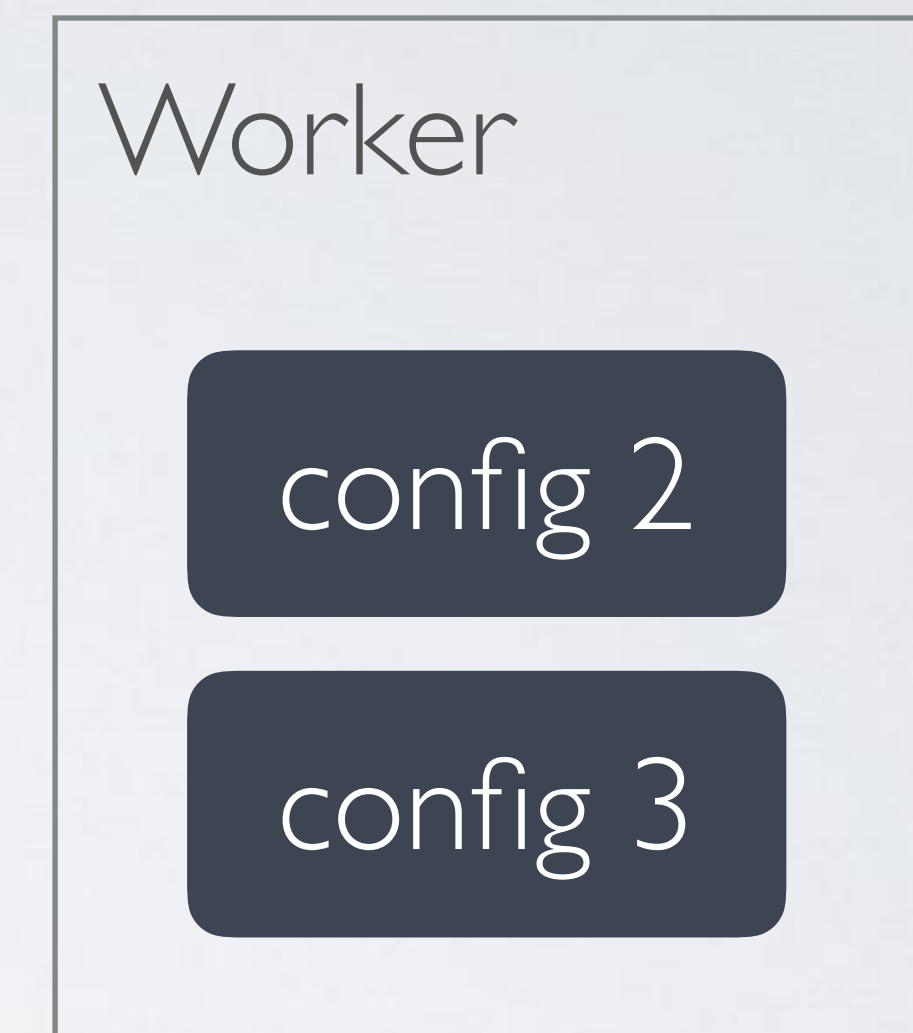
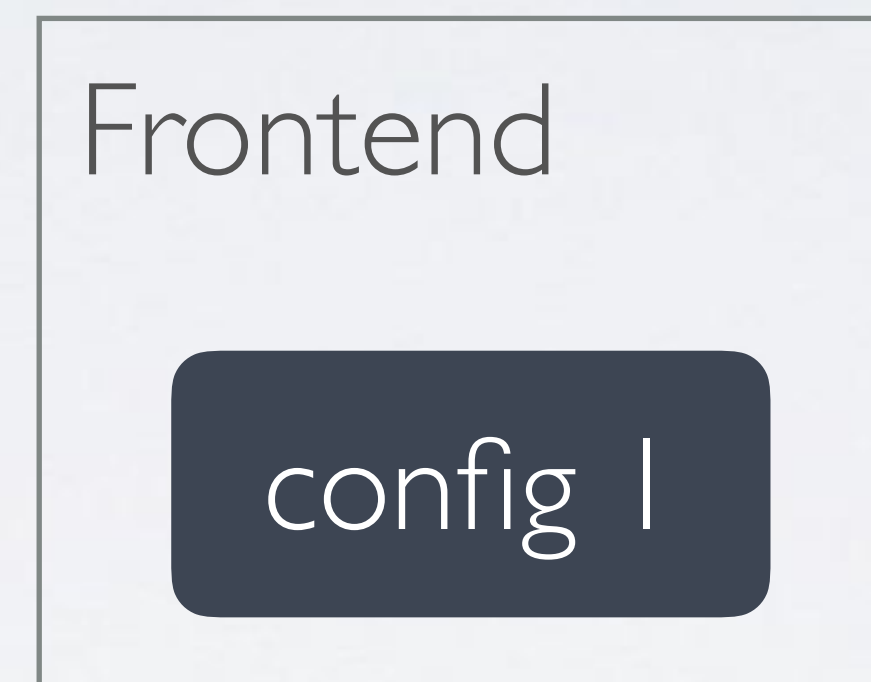
```
root@cockroachdb-public:26257/db1> select * from users;
```

id	name	created_at
262376372306051076	acoshift	2017-07-15 17:55:14.366042+00:00
262376372306247684	user1	2017-07-15 17:55:14.366042+00:00
262376372306345988	user2	2017-07-15 17:55:14.366042+00:00

(3 rows)

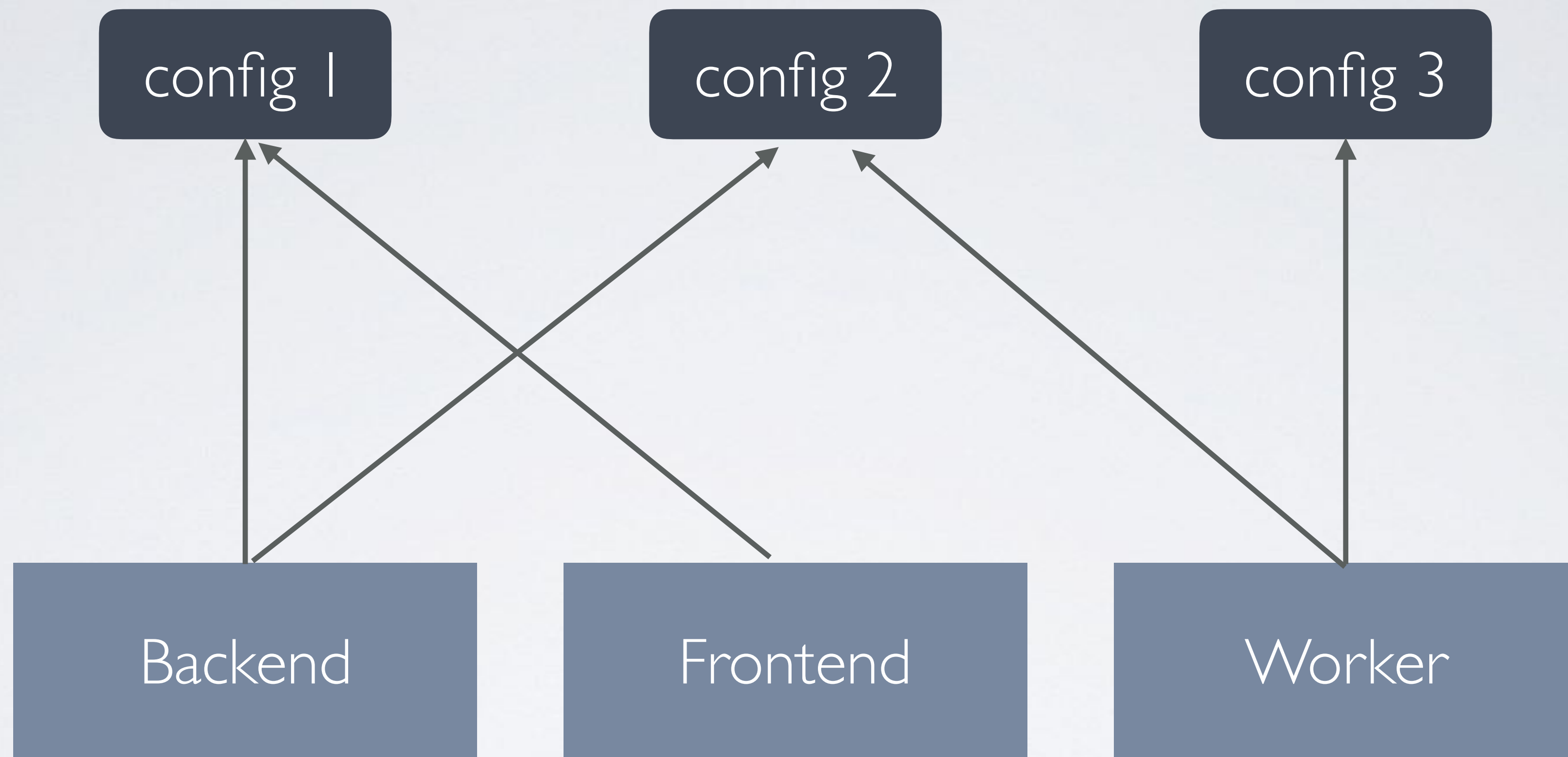
# Config Maps (cm)

decouple configuration artifacts from image content  
to keep containerized applications portable





Config Map



```
kind: ConfigMap
apiVersion: v1
metadata:
  name: redis-config
data:
  redis.conf: |
    databases 1
    save ""
    appendonly no
    maxmemory 2mb
    maxmemory-policy allkeys-lru
```

```
---
kind: Service
apiVersion: v1
metadata:
  name: redis
spec:
  selector:
    app: redis
  ports:
  - port: 6379
```

```
kind: StatefulSet
apiVersion: apps/v1beta1
metadata:
  name: redis
spec:
  serviceName: redis
  replicas: 1
  template:
    metadata:
      labels:
        app: redis
    spec:
      containers:
      - name: redis
        image: redis:3.2.9
        ports:
        - containerPort: 6379
        volumeMounts:
        - mountPath: /usr/local/etc/redis
          name: config
        command:
        - redis-server
        - /usr/local/etc/redis/redis.conf
      volumes:
      - name: config
        configMap:
          name: redis-config
          items:
          - key: redis.conf
            path: redis.conf
```

```
$ kubectl create -f cm.yaml  
configmap "redis-config" created  
service "redis" created  
statefulset "redis" created
```


```
$ kubectl run -it --rm redis-client --image=redis --restart=Never --command -- bash  
root@redis-client:/data# redis-cli -h redis
```

```
redis:6379> config get save  
1) "save"  
2) ""
```



# Secrets

hold sensitive information



respect access  
control and are not  
visible to users without  
read permissions

```
$ echo -n "mysqlpassword" | base64  
bXlzcWxwYXNzd29yZA==
```

```
kind: Secret
apiVersion: v1
metadata:
  name: mysql
data:
  password: bXlzcWxwYXNzd29yZA==
```

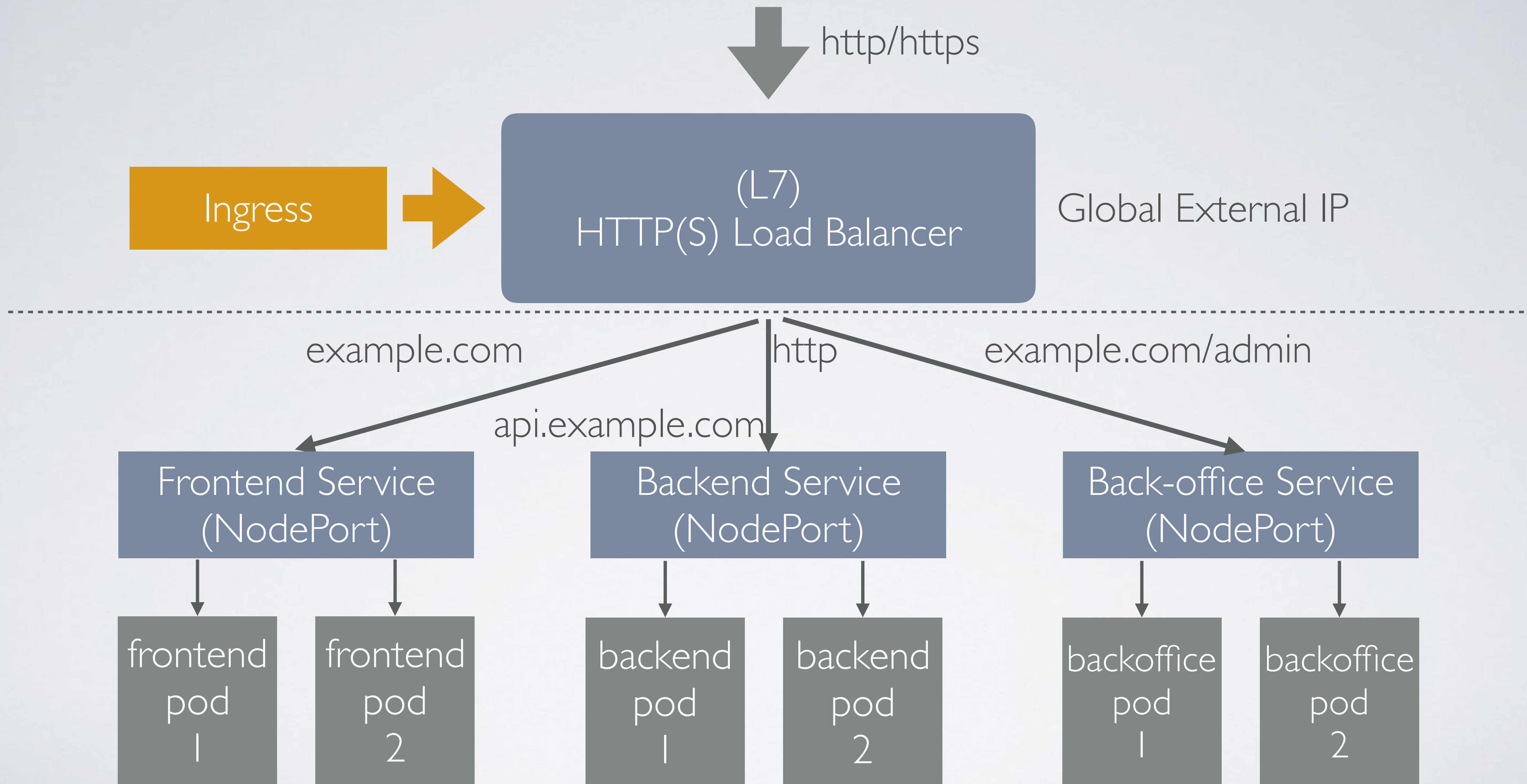
```
kind: Deployment
apiVersion: apps/v1beta1
metadata:
  name: mysql
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
      - name: mysql
        env:
          - name: MYSQL_ROOT_PASSWORD
            valueFrom:
              secretKeyRef:
                name: mysql
                key: password
        image: mysql:5.6.36
        ports:
          - containerPort: 3306
```



# Ingresses (ing)

a collection of rules that allow inbound connections to reach the cluster services

# Google Cloud HTTP(S) Load Balancer



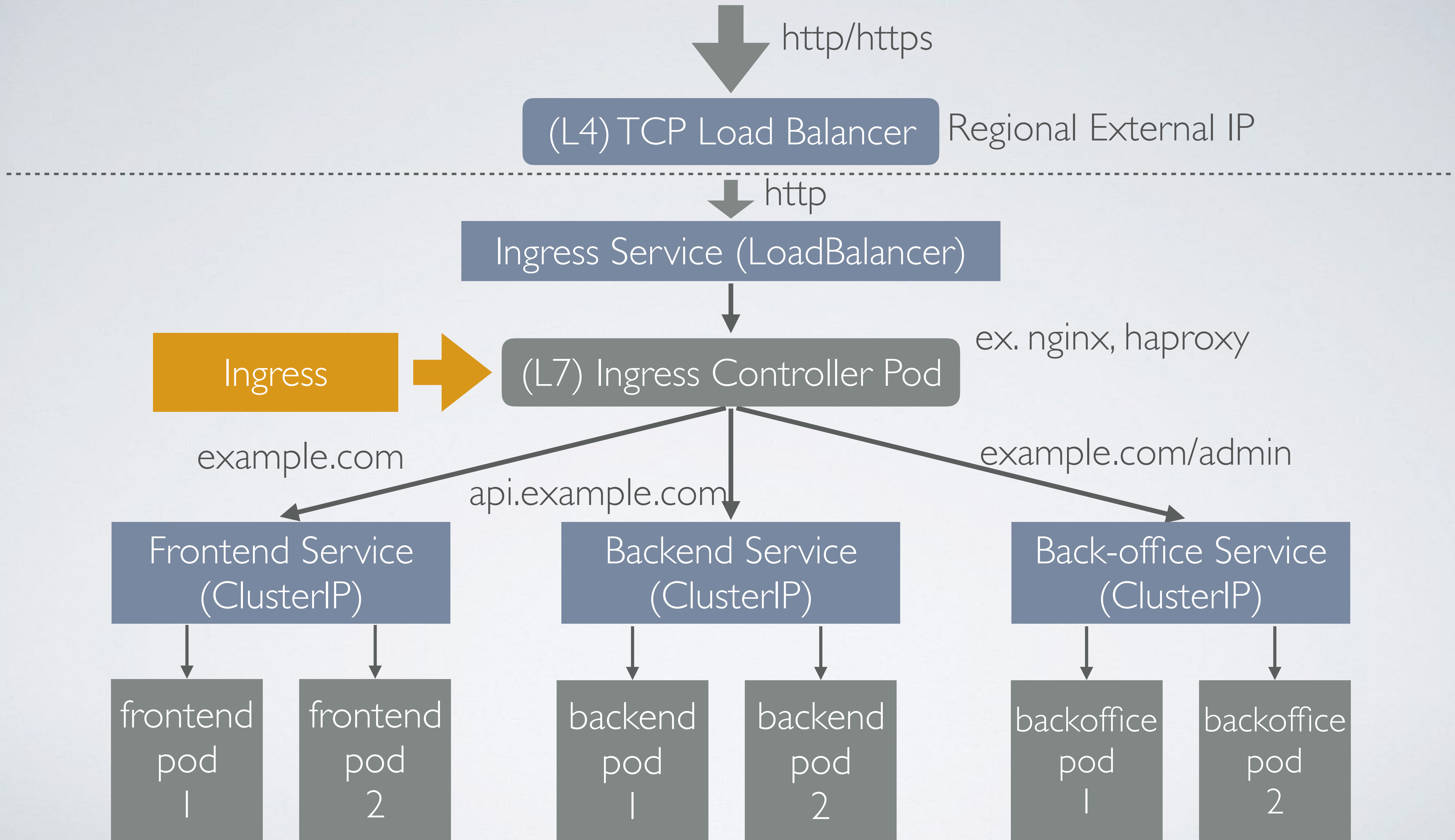


```
kind: Deployment
apiVersion: apps/v1beta1
metadata:
  name: echoserver
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: echoserver
    spec:
      containers:
      - name: echoserver
        image: gcr.io/google-containers/echoserver:1.6
        ports:
        - containerPort: 8080
        readinessProbe:
          httpGet:
            path: /
            port: 8080
```

```
apiVersion: v1
kind: Service
metadata:
  name: echoserver
spec:
  type: NodePort
  selector:
    app: echoserver
  ports:
  - port: 8080
```

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  annotations:
    kubernetes.io/ingress.class: gce
  name: gce-ingress
spec:
  rules:
    - host: echoserver-secure.acoshift.me
      http:
        paths:
          - backend:
              serviceName: echoserver
              servicePort: 8080
            path: /*
  tls:
    - secretName: echoserver-secure-acoshift-me-tls
      hosts:
        - echoserver-secure.acoshift.me
```

# Nginx Ingress Controller



```
kind: Deployment
apiVersion: apps/v1beta1
metadata:
  name: echoserver
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: echoserver
    spec:
      containers:
      - name: echoserver
        image: gcr.io/google-containers/echoserver:1.6
        ports:
        - containerPort: 8080

apiVersion: v1
kind: Service
metadata:
  name: echoserver
spec:
  ports:
  - port: 80
    targetPort: 8080
  selector:
    app: echoserver
```

```
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: default-http-backend
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: default-http-backend
    spec:
      containers:
      - name: default-http-backend
        image: gcr.io/google-containers/defaultbackend:1.0
        livenessProbe:
          httpGet:
            path: /healthz
            port: 8080
            scheme: HTTP
          initialDelaySeconds: 30
          timeoutSeconds: 5
        ports:
        - containerPort: 8080
        resources:
          limits:
            cpu: 10m
            memory: 20Mi
          requests:
            cpu: 10m
            memory: 20Mi
```

```
apiVersion: v1
kind: Service
metadata:
  name: default-http-backend
spec:
  ports:
  - port: 80
    targetPort: 8080
  selector:
    app: default-http-backend
```

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: nginx-config
data:
  client-max-body-size: 20m
  hsts: "false"
  keep-alive: "30"
  proxy-body-size: 20m
  server-tokens: "false"
  use-gzip: "true"
```

```
apiVersion: v1
kind: Service
metadata:
  name: nginx-ingress
spec:
  type: LoadBalancer
  selector:
    app: nginx-ingress
  ports:
    - name: http
      port: 80
    - name: https
      port: 443
```



```
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: nginx-ingress
spec:
  replicas: 1
  template:
    metadata:
      labels:
        app: nginx-ingress
    spec:
      containers:
      - name: nginx-ingress-controller
        image: gcr.io/google-containers/nginx-ingress-controller:0.9.0-beta.10
        imagePullPolicy: Always
        ports:
        - containerPort: 80
        - containerPort: 443
        env:
        - name: POD_NAME
          valueFrom:
            fieldRef:
              apiVersion: v1
              fieldPath: metadata.name
        - name: POD_NAMESPACE
          valueFrom:
            fieldRef:
              apiVersion: v1
              fieldPath: metadata.namespace
        args:
        - /nginx-ingress-controller
        - --default-backend-service=$(POD_NAMESPACE)/default-http-backend
        - --configmap=$(POD_NAMESPACE)/nginx-config
        - --publish-service=$(POD_NAMESPACE)/nginx-ingress
```

```
livenessProbe:
  failureThreshold: 3
  httpGet:
    path: /healthz
    port: 10254
    scheme: HTTP
  initialDelaySeconds: 10
  periodSeconds: 10
  successThreshold: 1
  timeoutSeconds: 5
readinessProbe:
  failureThreshold: 3
  httpGet:
    path: /healthz
    port: 10254
    scheme: HTTP
  periodSeconds: 10
  successThreshold: 1
  timeoutSeconds: 1
```

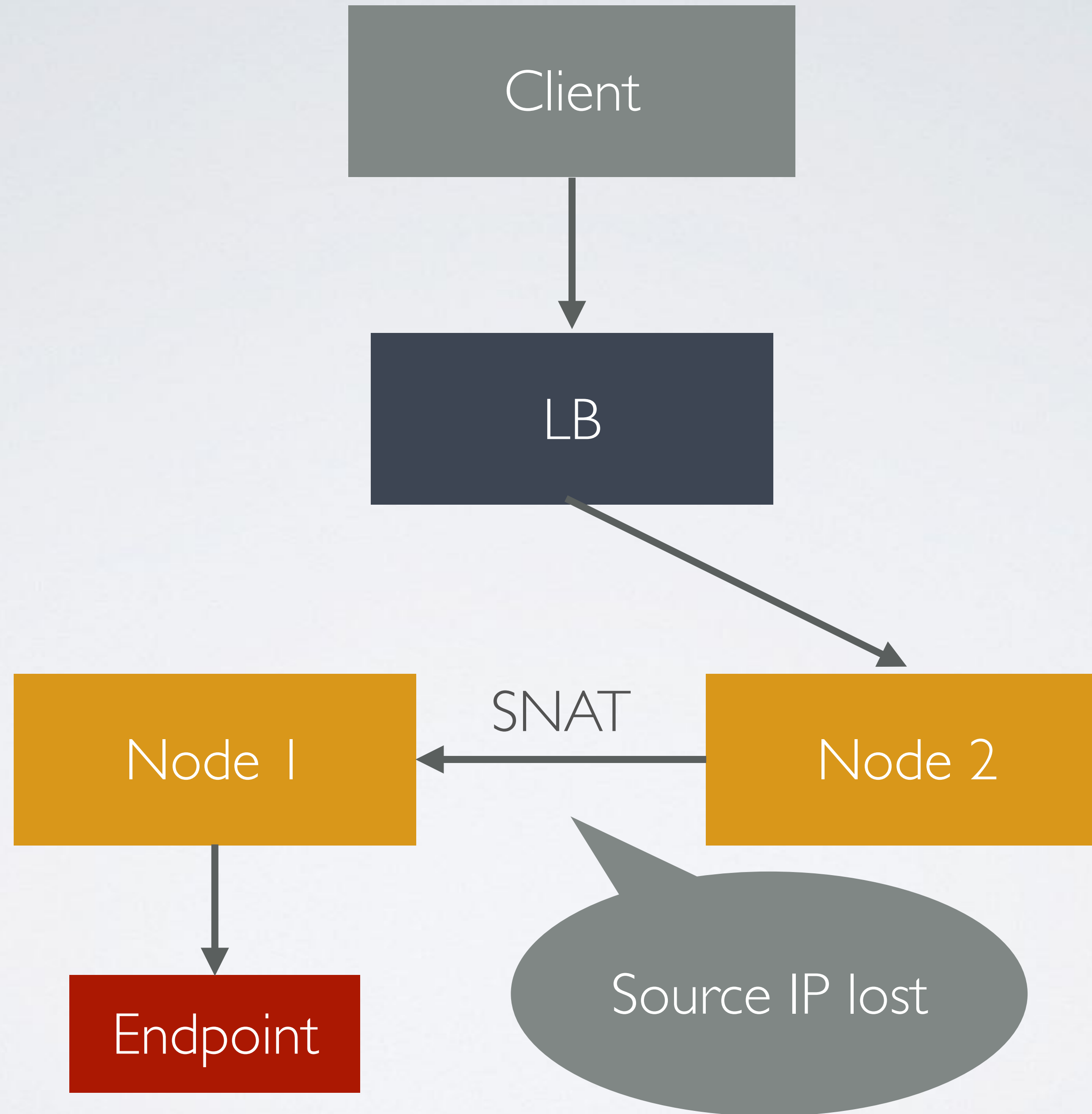
```
kind: Ingress
apiVersion: extensions/v1beta1
metadata:
  name: nginx-ingress
  annotations:
    kubernetes.io/ingress.class: nginx
spec:
  rules:
    - host: echoserver-secure.acoshift.me
      http:
        paths:
          - path: /
            backend:
              serviceName: echoserver
              servicePort: 80
    - host: echoserver.acoshift.me
      http:
        paths:
          - path: /
            backend:
              serviceName: echoserver
              servicePort: 80
  tls:
    - secretName: echoserver-secure-acoshift-me-tls
      hosts:
        - echoserver-secure.acoshift.me
```

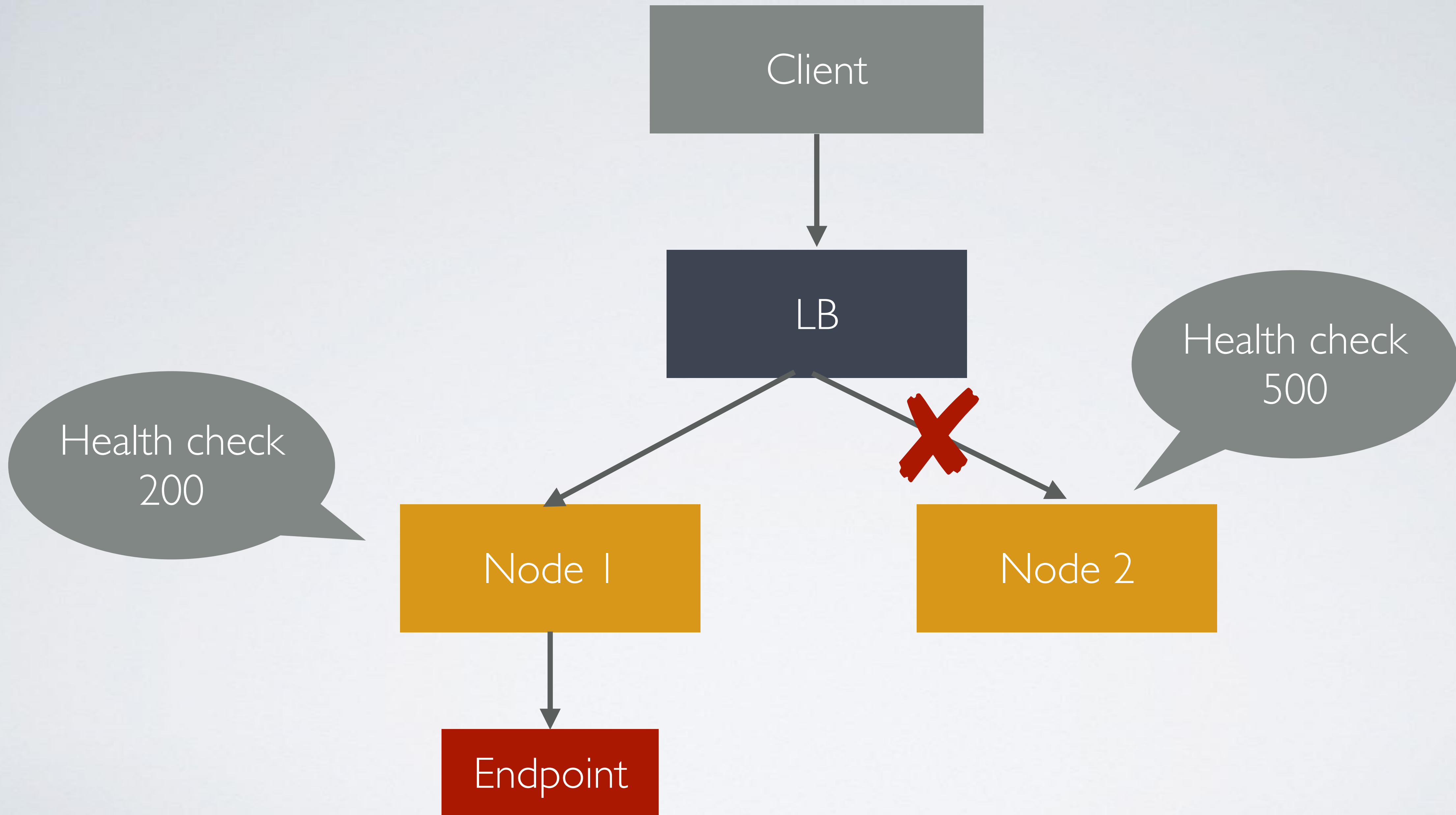
# External Traffic Policy

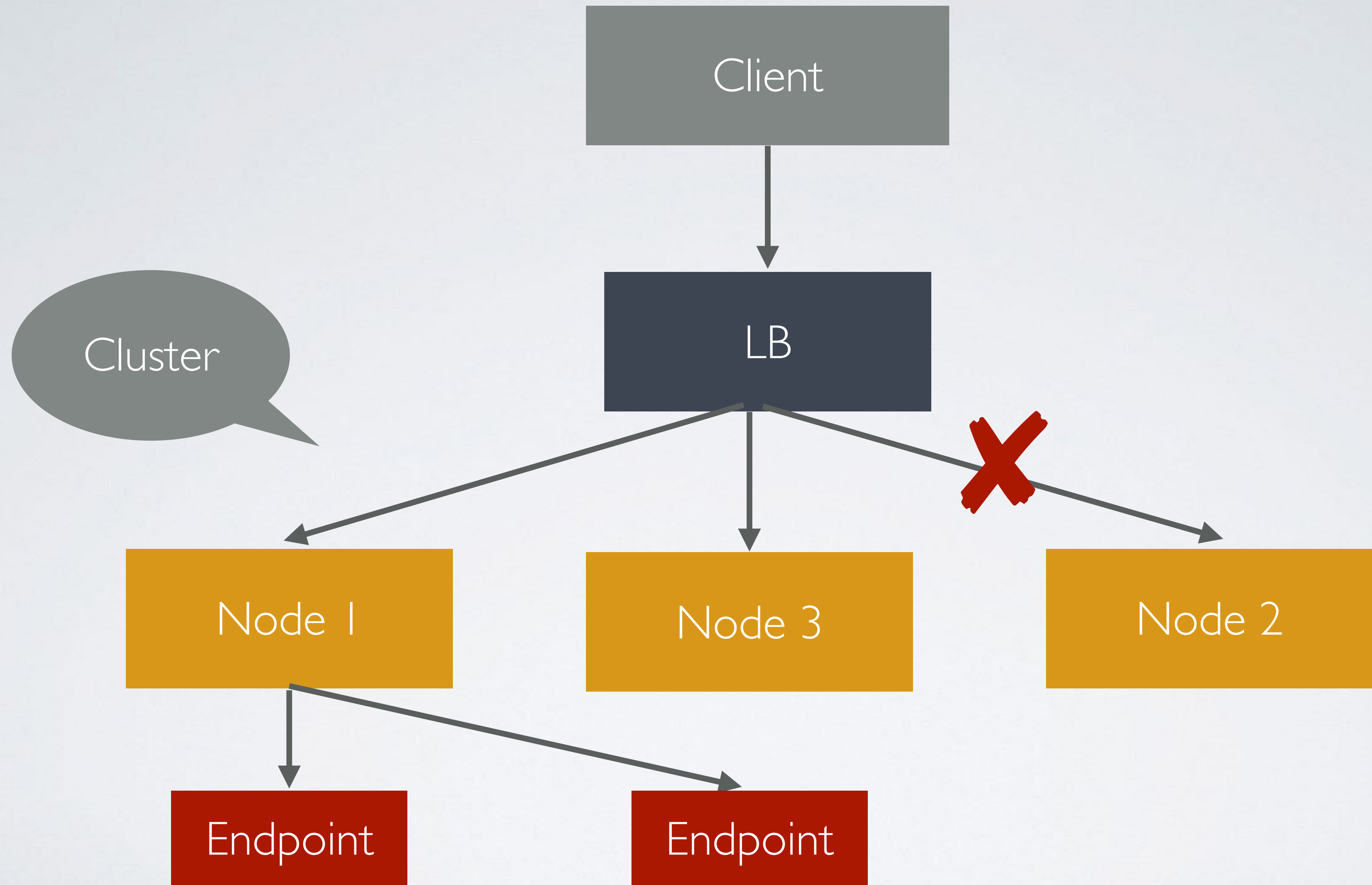
- Cluster
- Local



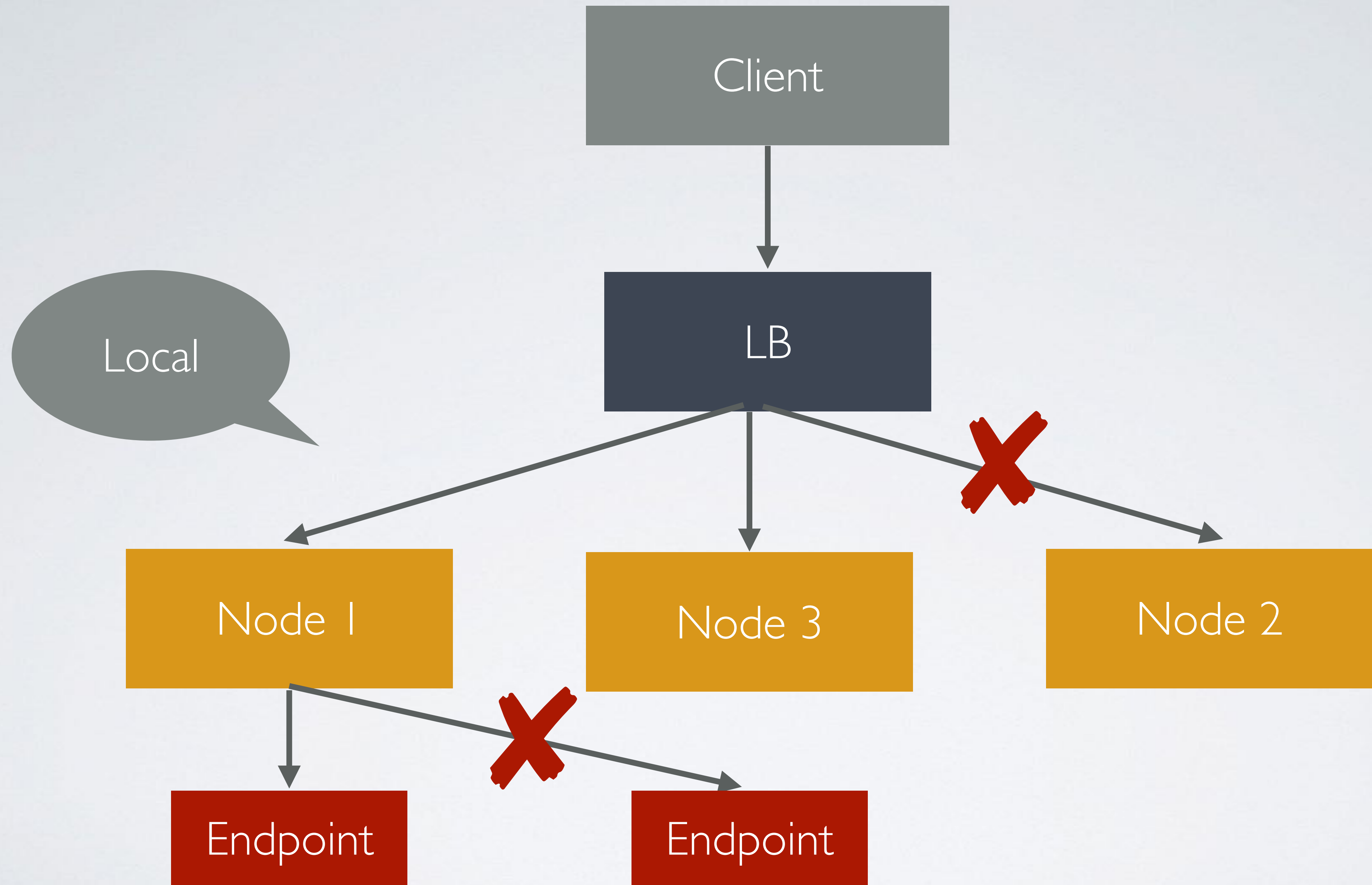
Default












```
kind: Service
apiVersion: v1
metadata:
  name: nginx-ingress
spec:
  type: LoadBalancer
  externalTrafficPolicy: Local
  selector:
    app: nginx-ingress
...
```





 **a46023bb4699311e7a0bf42010a94002**

Frontend

Protocol ^	IP:Port
TCP	35.187.231.192:80-443

Backend

Name: **a46023bb4699311e7a0bf42010a94002**    Region: **asia-southeast1**    Session affinity: **None**    Health check: **a46023bb4699311e7a0bf42010a94002**

Instances ^	35.187.231.192
gke-cluster-1-default-pool-73cdab92-hhk2	
gke-cluster-1-default-pool-73cdab92-k7np	

# kube-lego

automatically requests certificates for Kubernetes Ingress resources from Let's Encrypt



<https://github.com/jetstack/kube-lego>

# cert-manager

Automatically provision and manage TLS certificates in Kubernetes



<https://github.com/jetstack/cert-manager>

# helm

The Kubernetes Package Manager



<https://github.com/kubernetes/helm>

# Jobs

creates one or more pods and ensures that  
a specified number of them successfully terminate

```
apiVersion: batch/v1
kind: Job
metadata:
  name: backup-postgres
spec:
  template:
    spec:
      restartPolicy: OnFailure
      volumes:
      - name: data
        gcePersistentDisk:
          pdName: backup-disk
          fsType: ext4
      containers:
      - name: postgres
        image: postgres:9
        imagePullPolicy: Always
        env:
        - name: PGPASSWORD
          valueFrom:
            secretKeyRef:
              name: postgres
              key: backup
        command:
        - /bin/sh
        - -c
        args:
        - pg_dumpall -U backup -h postgres > /data/$(date +"%Y%m%d%H%M%S")-postgres
      volumeMounts:
      - name: data
        mountPath: /data
```



# Cron Jobs

manages time based Jobs

```
apiVersion: batch/v1beta1
kind: CronJob
metadata:
  name: backup-cronjob
spec:
  schedule: "0 21 * * *"
  successfulJobsHistoryLimit: 7
  failedJobsHistoryLimit: 7
  jobTemplate:
    spec:
      template:
        spec:
          restartPolicy: OnFailure
          volumes:
            - name: data
              gcePersistentDisk:
                pdName: backup-disk
                fsType: ext4
          containers:
            - name: postgres
              image: postgres:9
              imagePullPolicy: Always
              env:
                - name: PGPASSWORD
                  valueFrom:
                    secretKeyRef:
                      name: postgres
                      key: backup
              command:
                - /bin/sh
                - -c
              args:
                - pg_dumpall -U backup -h postgres > /data/$(date +"%Y%m%d%H%M%S")-postgres
          volumeMounts:
            - name: data
              mountPath: /data
```



# Google Container Builder

Fast, consistent, reliable builds on Google Cloud Platform

Source: GitHub    Repository: <https://github.com/acoshift/acourse> [↗](#)

[View triggered builds](#)

**Name** (Optional)

My trigger

**Trigger type** [?](#)

☒ Branch

☐ Tag

**Branch (regex)** [?](#)

Matches 2 branches: master, staging

master|staging

**Build configuration**

☐ Dockerfile

Specify the path within the Git repo

☒ cloudbuild.yaml

Specify the path to a Cloud Build configuration file in the Git repo [Learn more](#)

**cloudbuild.yaml location** [?](#)

/ cloudbuild.yaml

**Substitution variables** (Optional)

Substitutions allow to re-use a cloudbuild.yaml file with different variable values [Learn more](#)

[+ Add item](#)

**Summary**

Changes pushed to master|staging branch will trigger a build defined by the "cloudbuild.yaml" file.

Save

Cancel

```
steps:
- name: 'gcr.io/cloud-builders/docker'
  args: ['build', '-t', 'gcr.io/$PROJECT_ID/myapp:$COMMIT_SHA', '.']
- name: 'gcr.io/cloud-builders/docker'
  args: ['push', 'gcr.io/$PROJECT_ID/myapp:$COMMIT_SHA']
- name: 'gcr.io/cloud-builders/kubectl'
  args: ['set', 'image', 'deploy/myapp', 'myapp=gcr.io/$PROJECT_ID/myapp:$COMMIT_SHA']
  env:
  - 'CLOUDSDK_COMPUTE_ZONE=asia-southeast1-b'
  - 'CLOUDSDK_CONTAINER_CLUSTER=cluster-1'
images:
- 'gcr.io/$PROJECT_ID/myapp:$COMMIT_SHA'
```

```
steps:
- name: 'gcr.io/cloud-builders/npm'
  args: ['install']
- name: 'gcr.io/cloud-builders/npm'
  args: ['run', 'build']
- name: 'gcr.io/cloud-builders/go'
  args: ['build', '-o', 'entrypoint', '-a', '-ldflags', '-w -s', 'cmd/acourse/main.go']
  env:
  - 'PROJECT_ROOT=github.com/acoshift/acourse'
  - 'GOOS=linux'
  - 'GOARCH=amd64'
  - 'CGO_ENABLED=0'
- name: 'gcr.io/cloud-builders/docker'
  args: ['build', '-t', 'gcr.io/$PROJECT_ID/acourse:$COMMIT_SHA', '.']
- name: 'gcr.io/cloud-builders/docker'
  args: ['push', 'gcr.io/$PROJECT_ID/acourse:$COMMIT_SHA']
- name: 'gcr.io/cloud-builders/kubectl'
  args: ['set', 'image', 'deploy/acourse', 'acourse=gcr.io/$PROJECT_ID/acourse:$COMMIT_SHA']
  env:
  - 'CLOUDSDK_COMPUTE_ZONE=asia-southeast1-b'
  - 'CLOUDSDK_CONTAINER_CLUSTER=cluster-sg-1'
images:
- 'gcr.io/$PROJECT_ID/acourse:$COMMIT_SHA'
```

# Google Stackdriver

Monitoring, logging, and diagnostics for applications on Cloud Platform and AWS



Stackdriver

acoshift

UPGRADE

Thanatst

Monitoring Overview

Resources

Alerting

Uptime Checks

Groups

Dashboards

Debug

Trace

Logging

Error Reporting

Kubernetes Dashboard

TIME 1h 6h 1d 1w 1m 6w custom

GKE Container - CPU Usage

GKE Container - Disk Usage

GKE Container - Used Memory

Instance (GCE) - Network Inbound Traffic (GCE Monitoring)

Instance (GCE) - Network Outbound Traffic (GCE Monitoring)

Log Metrics

Pub/Sub Topic - Publish Request Count

URL Uptime Check - Endpoint Latency

Google Cloud Platform

Send Feedback

Q&A