Chapter 1: Introduction to Rancher and Kubernetes

No Images…

Chapter 2: Rancher and Kubernetes High-Level Architecture

```yaml
! cluster.yaml
nodes:
  - address: 172.27.7.21
    user: root
    hostname_override: a1ubrancherl01
    internal_address: 172.27.7.21
    role: [controlplane,worker,etcd]
  - address: 172.27.7.22
    user: root
    hostname_override: a1ubrancherl02
    internal_address: 172.27.7.22
    role: [controlplane,worker,etcd]
  - address: 172.27.7.23
    user: root
    hostname_override: a1ubrancherl03
    internal_address: 172.27.7.23
    role: [controlplane,worker,etcd]

services:
  etcd:
    backup_config:
      enabled: true     # enables recurring etcd snapshots
      interval_hours: 12 # time increment between snapshots
      retention: 6     # time in days before snapshot purge
      s3backupconfig:
        access_key: "ABCDEFGHIJKLMNPO..."
        secret_key:  "123456789abcdefghijklmpo..."
        bucket_name: "etcd-backups"
        folder: "Cluster-Name-Here"
        endpoint: "s3.us-west-1.wasabisys.com"
        region: "us-west-1"

dns:
  provider: coredns
  upstreamnameservers:
  - 172.27.2.23
  - 172.27.2.24
```

Chapter 3: Creating a Single Node Rancher

```
root@a1ubranl00:~# docker info
Client:
 Context:    default
 Debug Mode: false
 Plugins:
  app: Docker App (Docker Inc., v0.9.1-beta3)
  buildx: Build with BuildKit (Docker Inc., v0.6.1-docker)

Server:
 Containers: 0
  Running: 0
  Paused: 0
  Stopped: 0
 Images: 0
 Server Version: 20.10.8
 Storage Driver: overlay2
  Backing Filesystem: extfs
  Supports d_type: true
  Native Overlay Diff: true
  userxattr: false
 Logging Driver: json-file
 Cgroup Driver: cgroupfs
 Cgroup Version: 1
 Plugins:
  Volume: local
  Network: bridge host ipvlan macvlan null overlay
  Log: awslogs fluentd gcplogs gelf journald json-file local logentries splunk syslog
 Swarm: inactive
 Runtimes: io.containerd.runc.v2 io.containerd.runtime.v1.linux runc
 Default Runtime: runc
 Init Binary: docker-init
 containerd version: e25210fe30a0a703442421b0f60afac609f950a3
 runc version: v1.0.1-0-g4144b63
 init version: de40ad0
 Security Options:
  apparmor
  seccomp
   Profile: default
 Kernel Version: 5.4.0-86-generic
 Operating System: Ubuntu 20.04.3 LTS
 OSType: linux
 Architecture: x86_64
 CPUs: 4
 Total Memory: 7.748GiB
 Name: a1ubranl00
 ID: KRPX:O5TH:3XMS:GS5L:UPP3:CEBZ:X25W:QCTA:RBJN:IG76:6L2U:JZC2
 Docker Root Dir: /var/lib/docker
 Debug Mode: false
 Registry: https://index.docker.io/v1/
 Labels:
 Experimental: false
 Insecure Registries:
  127.0.0.0/8
 Live Restore Enabled: false

WARNING: No swap limit support
root@a1ubranl00:~#
```

```
root@a1ubranl00:~# cat /etc/docker/daemon.json
{
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "10m",
    "max-file": "3"
  }
}
root@a1ubranl00:~#
```

```
-----BEGIN CERTIFICATE-----
Root certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Server certificate
-----END CERTIFICATE-----
```

```
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED DEK-Info: DES-EDE3-CBC,
....
-----END RSA PRIVATE KEY-----
```

```
-----BEGIN ENCRYPTED PRIVATE KEY-----
MIIJnDBOBgkqhkiG9w0BBQ0wQTApBgkqhkiG9w0BBQwwHAQIupZ5LBxkx4wCAggA
....
0+S3p1U3GAYMxdbZAcMtKnQzSVI4AalbF7a+7C1bFS4JIWBg/W1jkzZP/lc6klKq
b1J+hEz5rSoD+E/2ccsLpg==
-----END ENCRYPTED PRIVATE KEY-----
```

```
-----BEGIN RSA PRIVATE KEY-----
MIIJKQIBAAKCAgEAuozVjS468biRJAwRr0+LLW+fxoucw4u5vsI4UyFnLA2KnNMV
vc8idiTLTyOjukxgUYCABGxu0jzk5QP9Fgqs7p2+MmZn+lDBPF9zLAP/SJQVL2Jx
....
3r8vCDapjAgW2pazmpzDCv67C3G6yO4WTqBAphSpk4AIy7YJUgDQxnf3sfMgCiqV
ZIPf77ywmSOF5Ou6EyTYqglsGRXYXfUrR1j+HmqQ+EdMNEY1Kk9lWk1vIZ/p
-----END RSA PRIVATE KEY-----
```

```
docker run -d \
--name rancher_server \
--restart=unless-stopped \
-p 80:80 \
-p 443:443 \
-v /etc/rancher/ssl/tls.crt:/etc/rancher/ssl/cert.pem \
-v /etc/rancher/ssl/tls.key:/etc/rancher/ssl/key.pem \
--privileged \
rancher/rancher:v2.5.8 \
--no-cacerts
```

```
root@a1ubranl00:~# docker logs rancher_server 2>&1 | grep "Bootstrap Password:"
2021/09/28 01:40:54 [INFO] Bootstrap Password: p7brsdjzs4bnfbldxn6rqhq2mfgffz5ghdjddj7ptw5vkmjkqm7q2g
root@a1ubranl00:~#
```

Welcome to Rancher!

The first order of business is to set a strong password for the default `admin` user. We suggest using this random one generated just for you, but enter your own if you like.

- ○ Use a randomly generated password
- ● Set a specific password to use

New Password
••••••••

Confirm New Password
••••••••

What URL should be used for this Rancher installation? All the nodes in your clusters will need to be able to reach this.

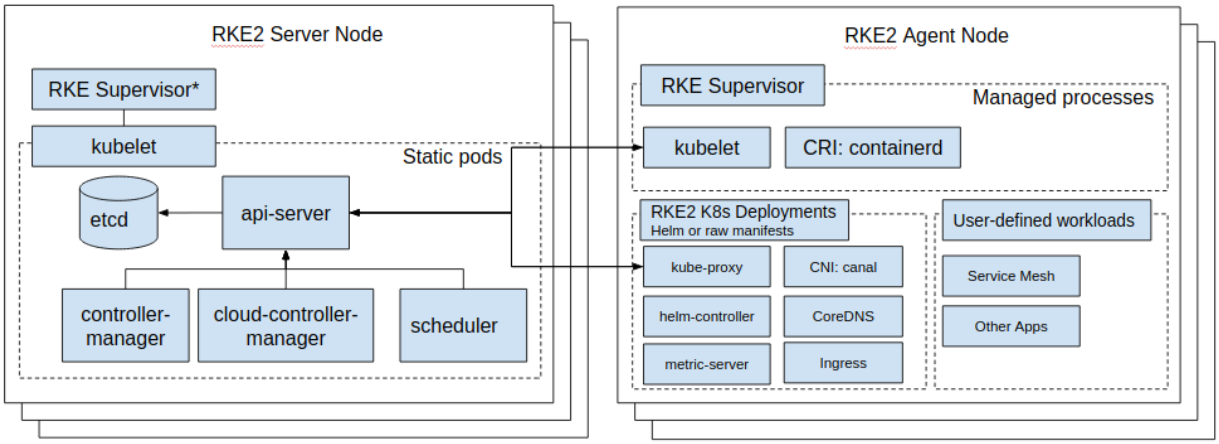
Server URL
https://a1ubranl00.support.tools

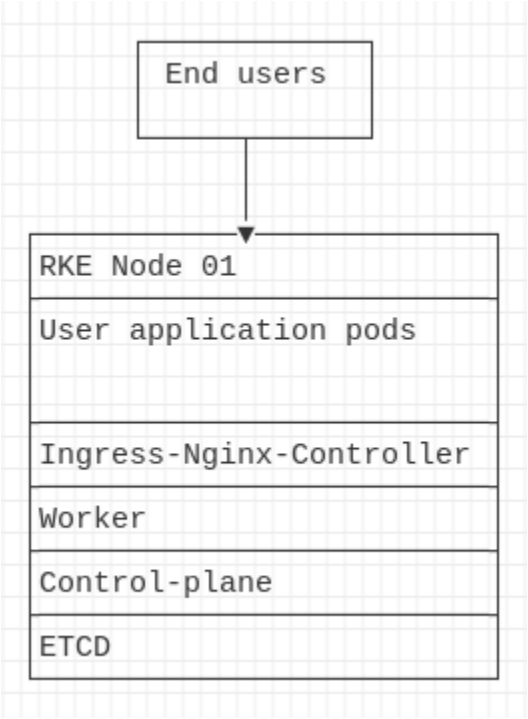
- [x] Allow collection of anonymous statistics to help us improve Rancher.
- [x] I agree to the terms and conditions for using Rancher.

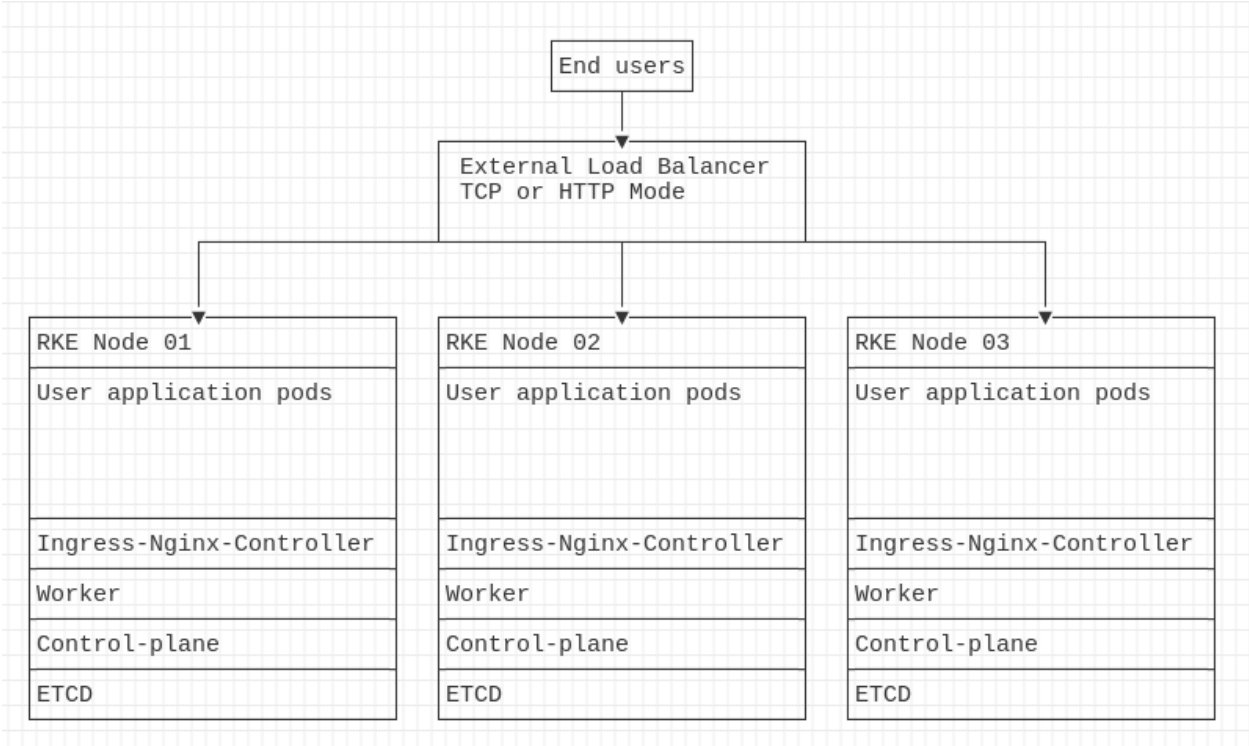
Continue

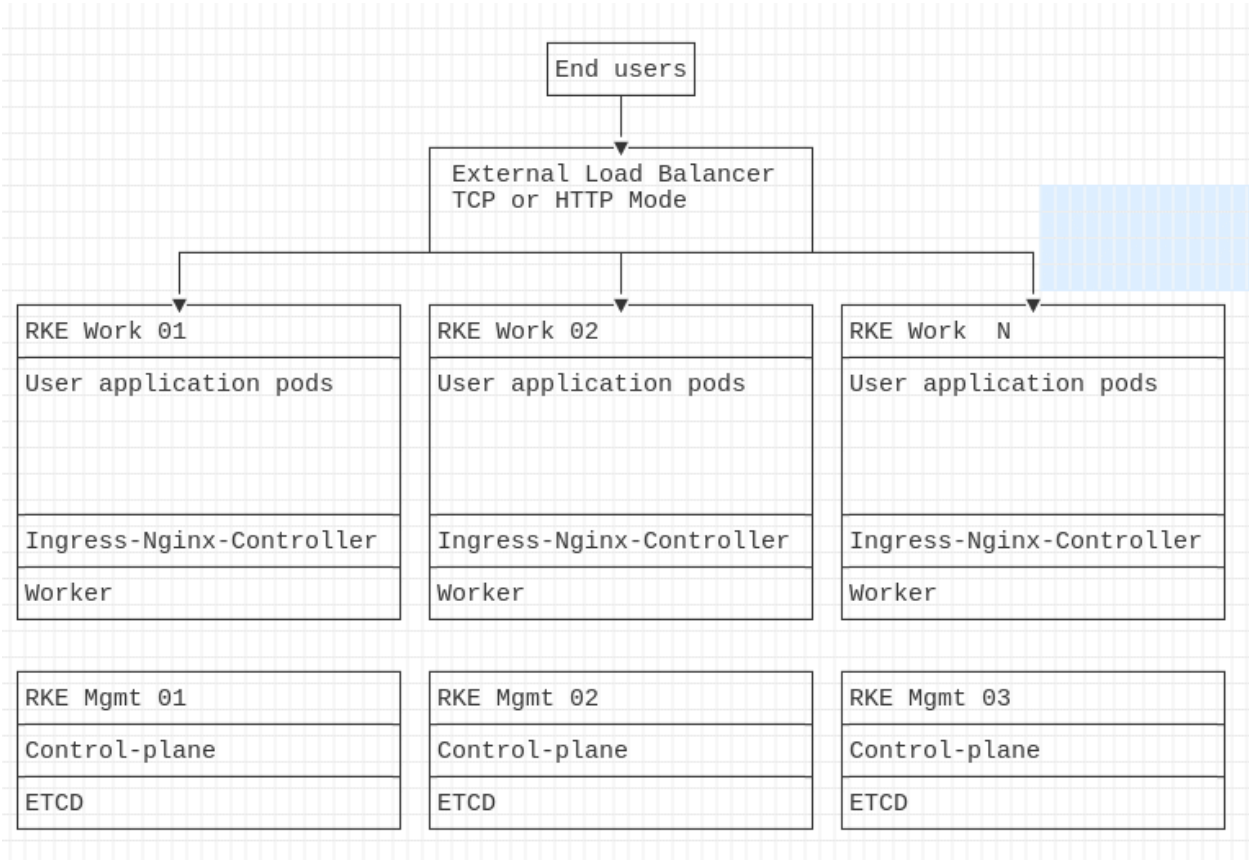
Chapter 4: Creating an RKE and RKE2 Cluster

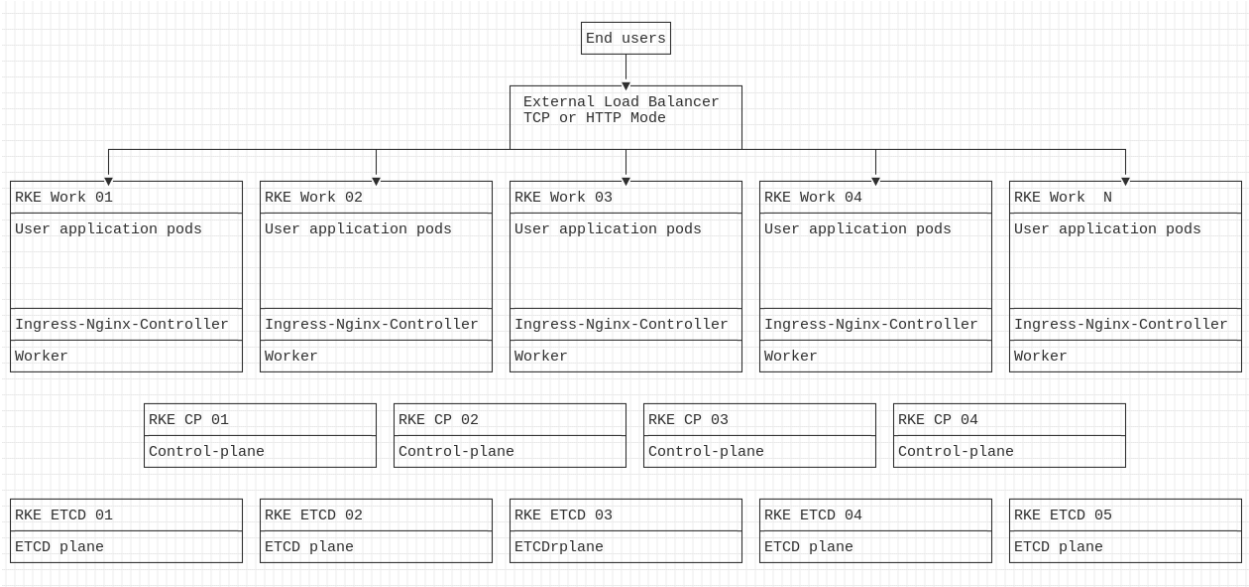
```yaml
nodes:
  - address: node01.support.tools
    hostname_override: node1
    internal_address: 192.168.1.101
    user: ubuntu
    role:
      - controlplane
      - worker
      - etcd
  - address: node02.support.tools
    hostname_override: node2
    internal_address: 192.168.1.102
    user: ubuntu
    role:
      - controlplane
      - worker
      - etcd
  - address: node03.support.tools
    hostname_override: node3
    internal_address: 192.168.1.103
    user: ubuntu
    role:
      - controlplane
      - worker
      - etcd

cluster_name: examplecluster
kubernetes_version: v1.21.5-rancher1-1

ingress:
  provider: nginx

services:
  kube-api:
    audit_log:
      enabled: true
  etcd:
    backup_config:
      enabled: true       # enables recurring etcd snapshots
      interval_hours: 3   # time increment between snapshots
      retention: 72       # time in days before snapshot purge
      # Optional S3
      s3backupconfig:
        access_key: "ABCDE....."
        secret_key:  "12345679....."
        bucket_name: "etcd"
        folder: "examplecluster"
        endpoint: "s3.us-west-1.amazonaws.com"
        region: "us-west-1"
```

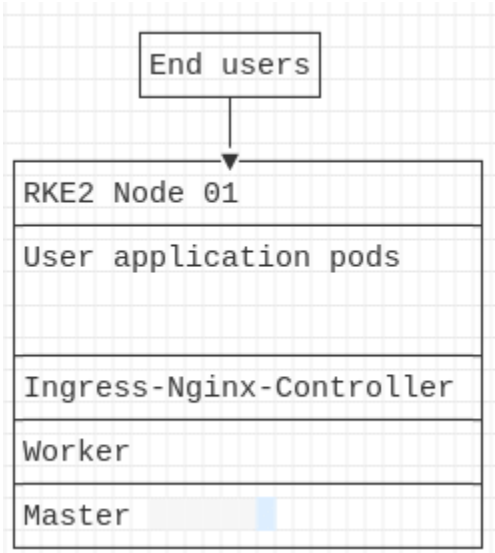

RKE2 Server Node
RKE Supervisor*
kubelet
Static pods
etcd
api-server
controller-manager
cloud-controller-manager
scheduler
RKE2 Agent Node
RKE Supervisor
Managed processes
kubelet
CRI: containerd
RKE2 K8s Deployments
Helm or raw manifests
User-defined workloads
kube-proxy
CNI: canal
helm-controller
CoreDNS
metric-server
Ingress
Service Mesh
Other Apps

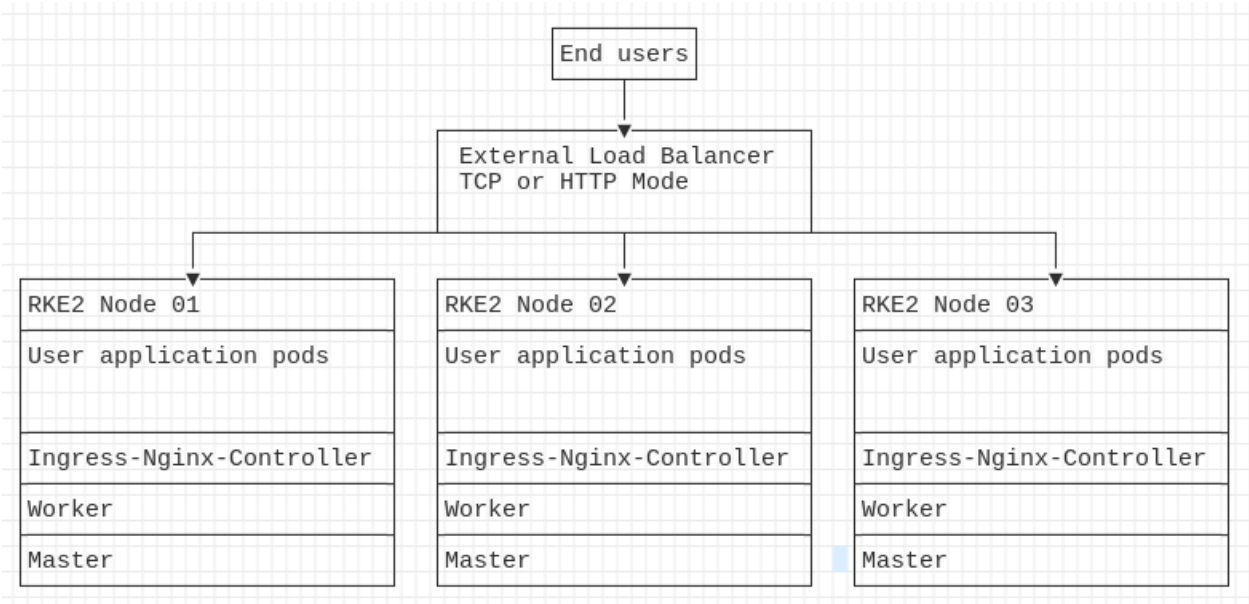

End users
RKE Node 01
User application pods
Ingress-Nginx-Controller
Worker
Control-plane
ETCD

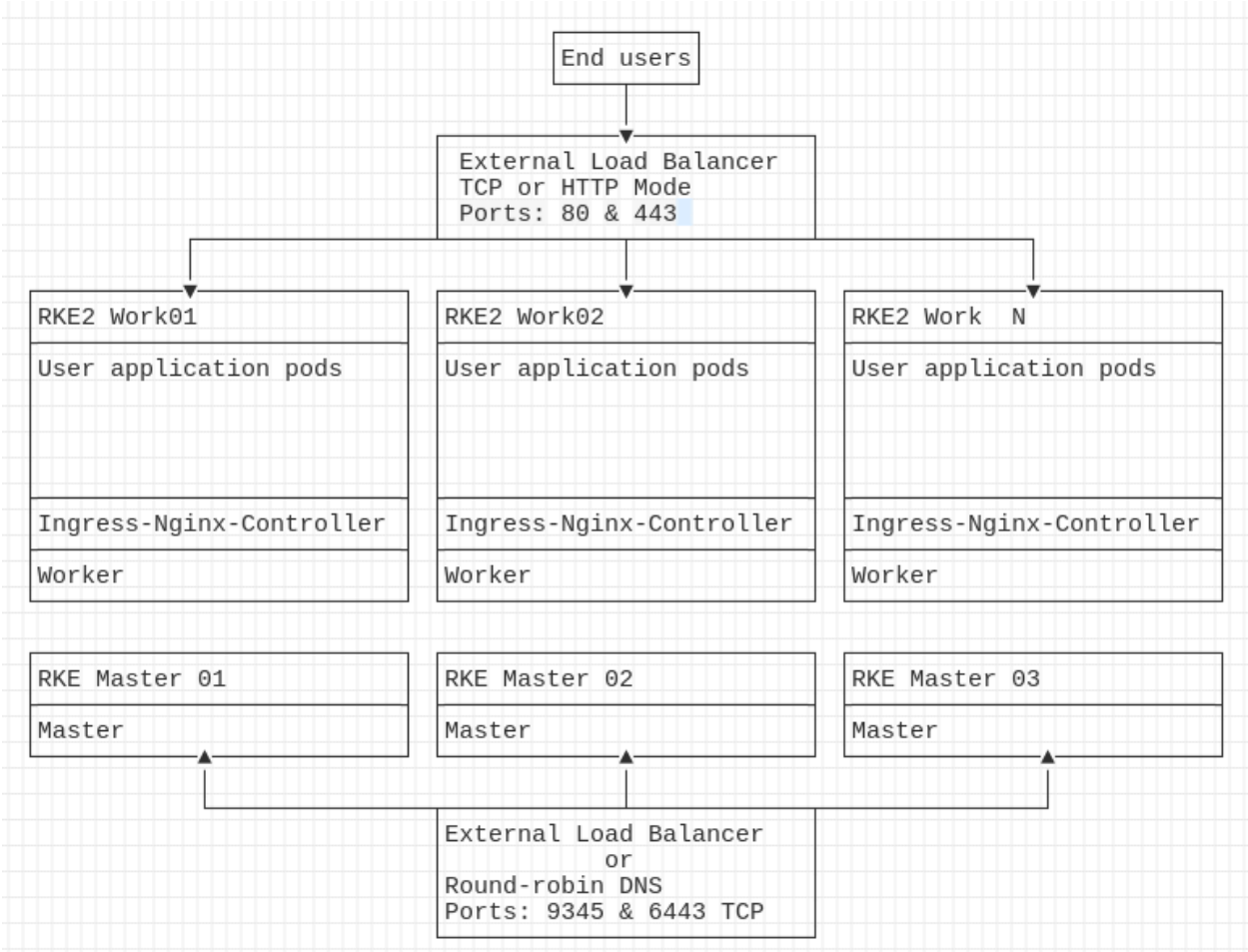

End users
External Load Balancer
TCP or HTTP Mode
RKE Node 01
User application pods
Ingress-Nginx-Controller
Worker
Control-plane
ETCD
RKE Node 02
User application pods
Ingress-Nginx-Controller
Worker
Control-plane
ETCD
RKE Node 03
User application pods
Ingress-Nginx-Controller
Worker
Control-plane
ETCD

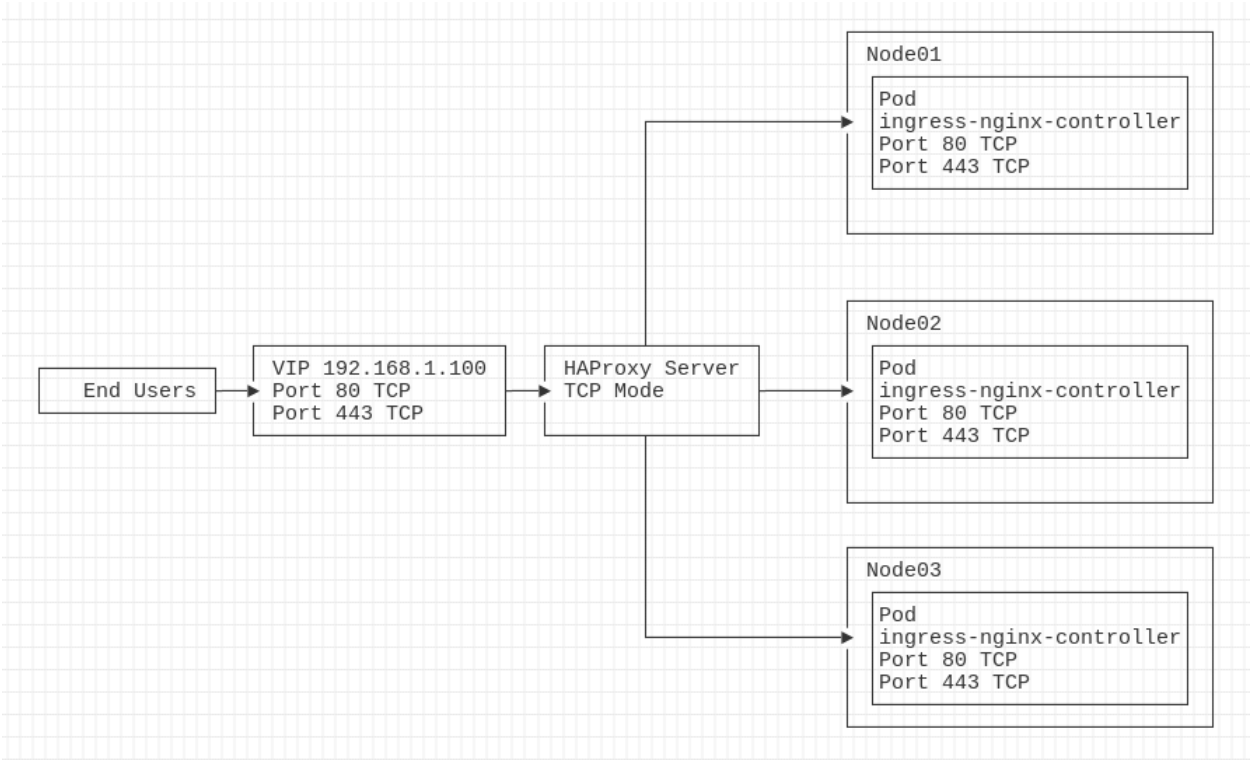

End users
External Load Balancer
TCP or HTTP Mode
RKE Work 01
User application pods
Ingress-Nginx-Controller
Worker
RKE Work 02
User application pods
Ingress-Nginx-Controller
Worker
RKE Work N
User application pods
Ingress-Nginx-Controller
Worker
RKE Mgmt 01
Control-plane
ETCD
RKE Mgmt 02
Control-plane
ETCD
RKE Mgmt 03
Control-plane
ETCD


End users
External Load Balancer
TCP or HTTP Mode
RKE Work 01
User application pods
Ingress-Nginx-Controller
Worker
RKE Work 02
User application pods
Ingress-Nginx-Controller
Worker
RKE Work 03
User application pods
Ingress-Nginx-Controller
Worker
RKE Work 04
User application pods
Ingress-Nginx-Controller
Worker
RKE Work N
User application pods
Ingress-Nginx-Controller
Worker
RKE CP 01
Control-plane
RKE CP 02
Control-plane
RKE CP 03
Control-plane
RKE CP 04
Control-plane
RKE ETCD 01
ETCD plane
RKE ETCD 02
ETCD plane
RKE ETCD 03
ETCDrplane
RKE ETCD 04
ETCD plane
RKE ETCD 05
ETCD plane

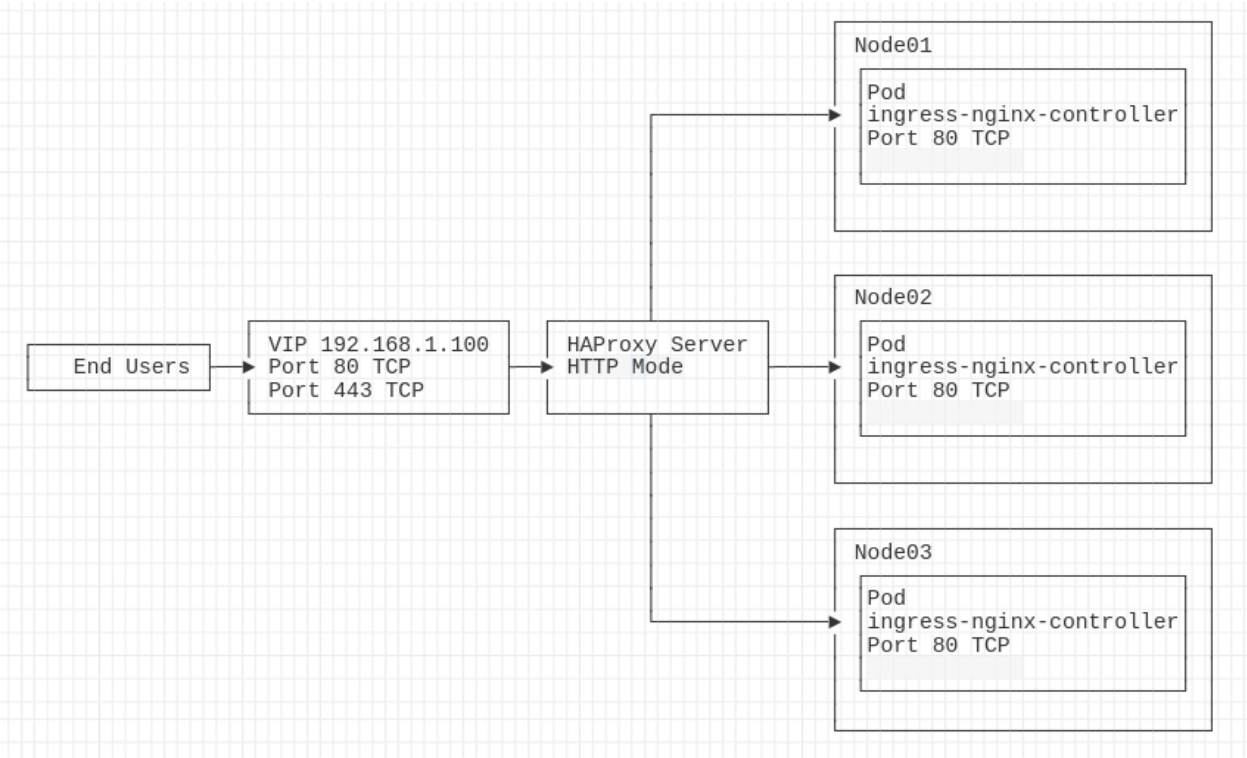

End users
RKE2 Node 01
User application pods
Ingress-Nginx-Controller
Worker
Master


End users
External Load Balancer
TCP or HTTP Mode
RKE2 Node 01
User application pods
Ingress-Nginx-Controller
Worker
Master
RKE2 Node 02
User application pods
Ingress-Nginx-Controller
Worker
Master
RKE2 Node 03
User application pods
Ingress-Nginx-Controller
Worker
Master

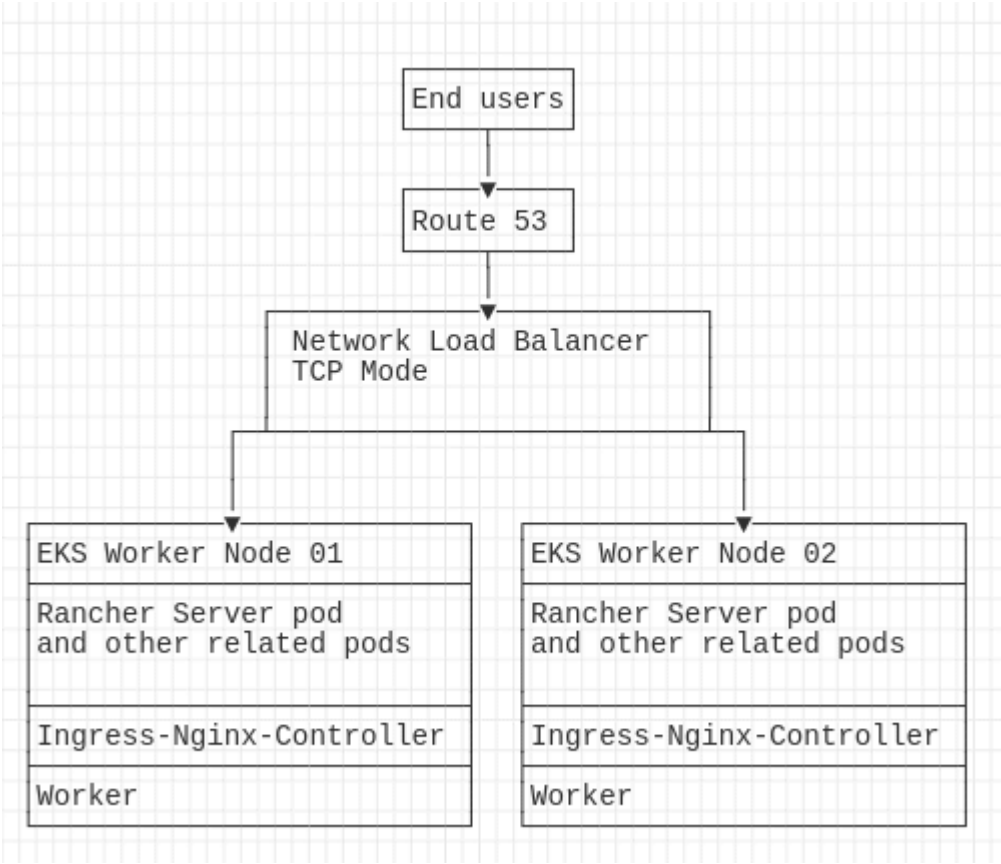

End users
External Load Balancer
TCP or HTTP Mode
Ports: 80 & 443
RKE2 Work01
User application pods
Ingress-Nginx-Controller
Worker
RKE2 Work02
User application pods
Ingress-Nginx-Controller
Worker
RKE2 Work N
User application pods
Ingress-Nginx-Controller
Worker
RKE Master 01
Master
RKE Master 02
Master
RKE Master 03
Master
External Load Balancer
or
Round-robin DNS
Ports: 9345 & 6443 TCP

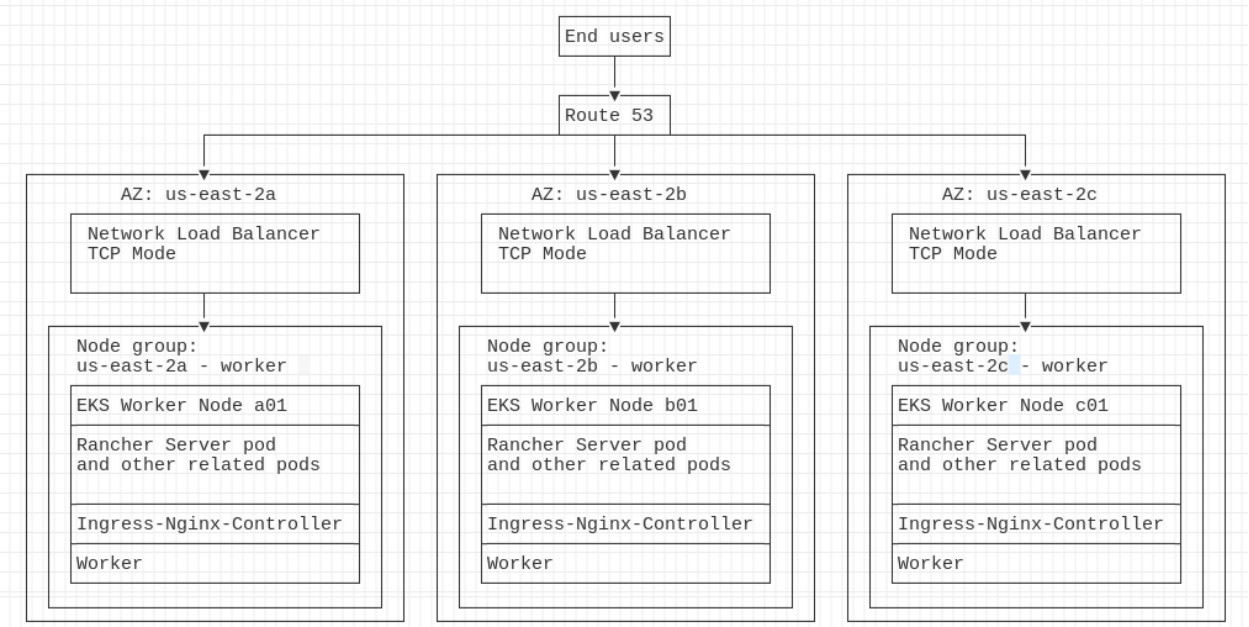

End Users
VIP 192.168.1.100
Port 80 TCP
Port 443 TCP
HAProxy Server
TCP Mode
Node01
Pod
ingress-nginx-controller
Port 80 TCP
Port 443 TCP
Node02
Pod
ingress-nginx-controller
Port 80 TCP
Port 443 TCP
Node03
Pod
ingress-nginx-controller
Port 80 TCP
Port 443 TCP

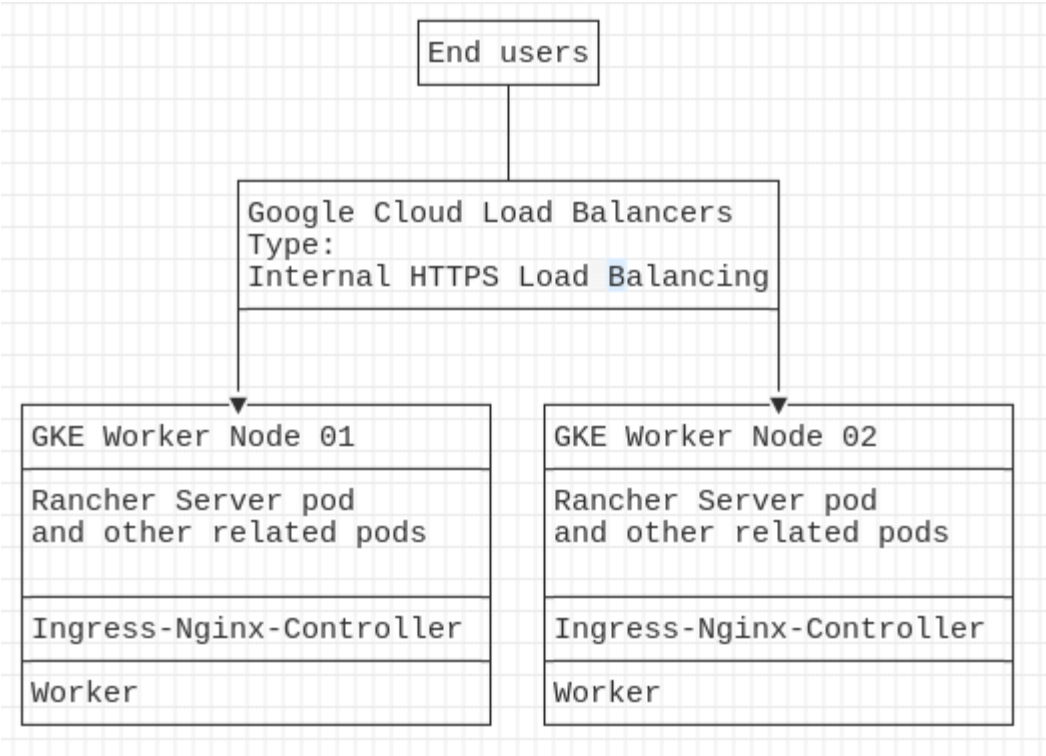
```
frontend prometheus
    bind *:8404
    option http-use-htx
    http-request use-service prometheus-exporter if { path /metrics }
    stats enable
    stats uri /stats
    stats refresh 10s

listen stats
    bind :9000
    mode http
    stats enable
    stats hide-version
    stats realm Haproxy\ Statistics
    stats uri /
    stats auth admin:Passw0rd

listen http
    bind *:80
    mode tcp
    balance leastconn
    stick match src
    stick-table type ip size 200k expire 30m
    server node01 192.168.1.101:80 check
    server node02 192.168.1.102:80 check
    server node03 192.168.1.103:80 check

listen https
    bind *:443
    mode tcp
    balance leastconn
    stick match src
    stick-table type ip size 200k expire 30m
    server node01 192.168.1.101:443 check ssl verify none
    server node02 192.168.1.102:443 check ssl verify none
    server node03 192.168.1.103:443 check ssl verify none
```

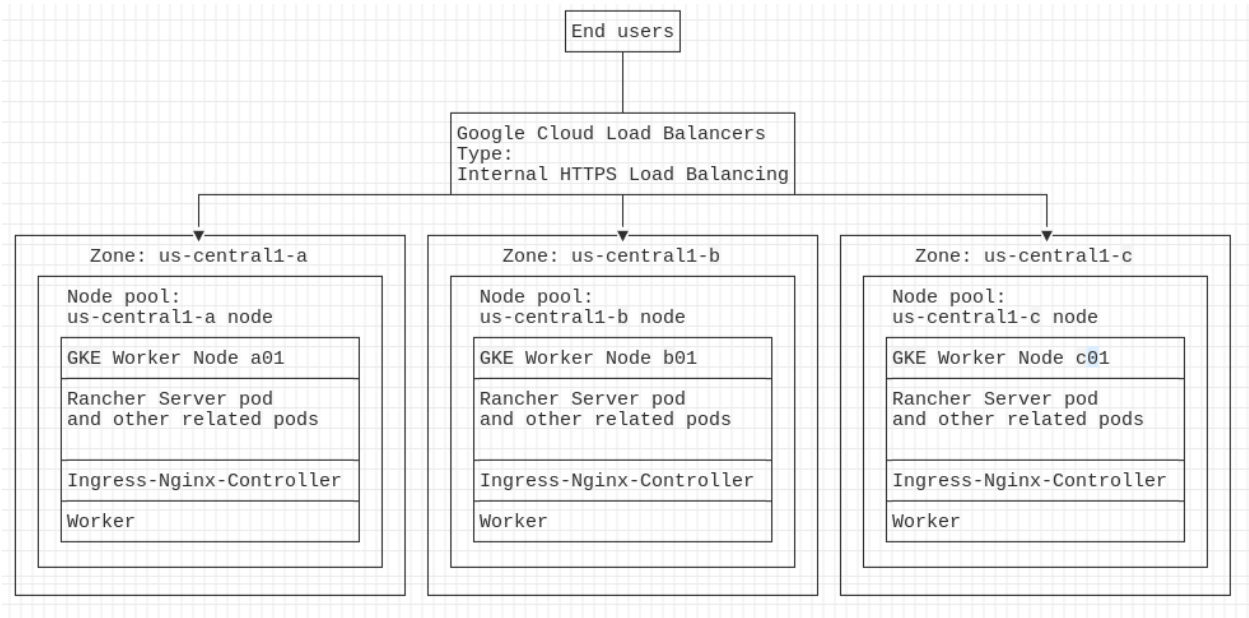

End Users
VIP 192.168.1.100
Port 80 TCP
Port 443 TCP
HAProxy Server
HTTP Mode
Node01
Pod
ingress-nginx-controller
Port 80 TCP
Node02
Pod
ingress-nginx-controller
Port 80 TCP
Node03
Pod
ingress-nginx-controller
Port 80 TCP

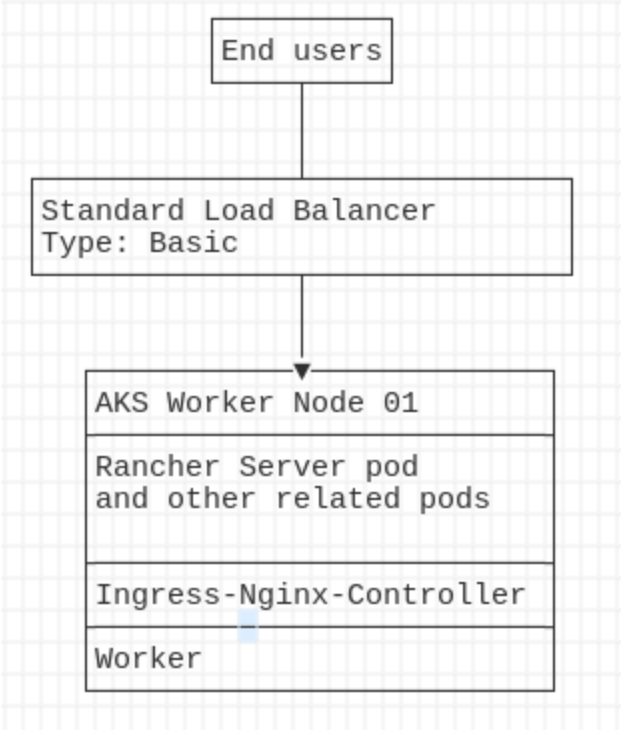
Chapter 5: Deploying Rancher on a Hosted Kubernetes Cluster

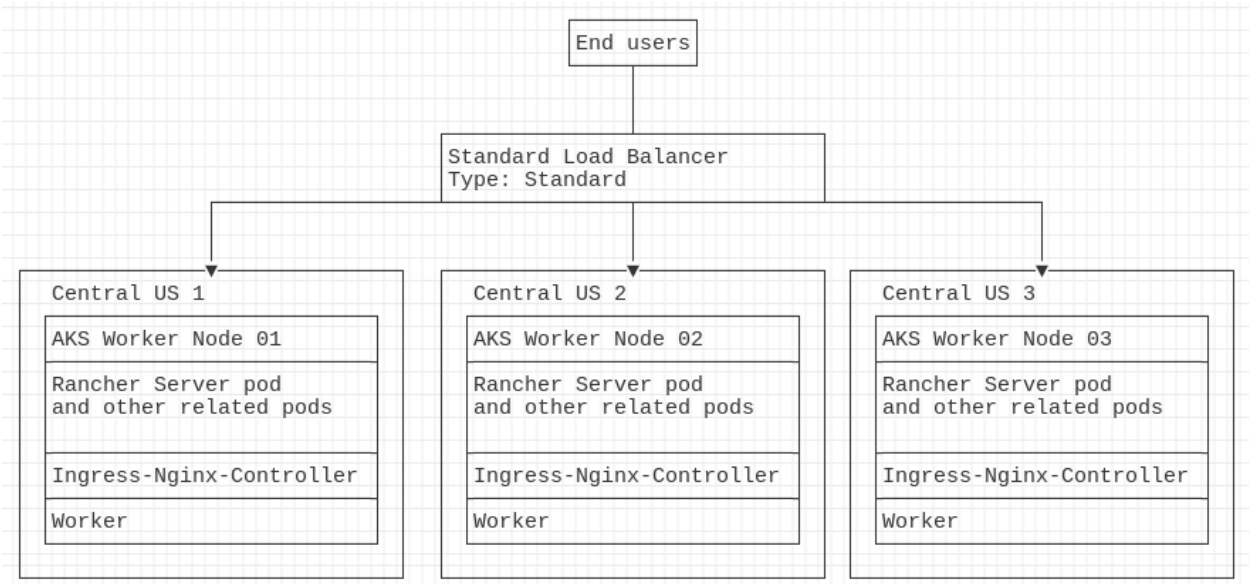





End users
Google Cloud Load Balancers
Type:
Internal HTTPS Load Balancing
GKE Worker Node 01
Rancher Server pod
and other related pods
Ingress-Nginx-Controller
Worker
GKE Worker Node 02
Rancher Server pod
and other related pods
Ingress-Nginx-Controller
Worker


End users
Google Cloud Load Balancers
Type:
Internal HTTPS Load Balancing
Zone: us-central1-a
Node pool:
us-central1-a node
GKE Worker Node a01
Rancher Server pod
and other related pods
Ingress-Nginx-Controller
Worker
Zone: us-central1-b
Node pool:
us-central1-b node
GKE Worker Node b01
Rancher Server pod
and other related pods
Ingress-Nginx-Controller
Worker
Zone: us-central1-c
Node pool:
us-central1-c node
GKE Worker Node c01
Rancher Server pod
and other related pods
Ingress-Nginx-Controller
Worker


End users
Standard Load Balancer
Type: Basic
AKS Worker Node 01
Rancher Server pod
and other related pods
Ingress-Nginx-Controller
Worker


End users
Standard Load Balancer
Type: Standard
Central US 1
AKS Worker Node 01
Rancher Server pod
and other related pods
Ingress-Nginx-Controller
Worker
Central US 2
AKS Worker Node 02
Rancher Server pod
and other related pods
Ingress-Nginx-Controller
Worker
Central US 3
AKS Worker Node 03
Rancher Server pod
and other related pods
Ingress-Nginx-Controller
Worker

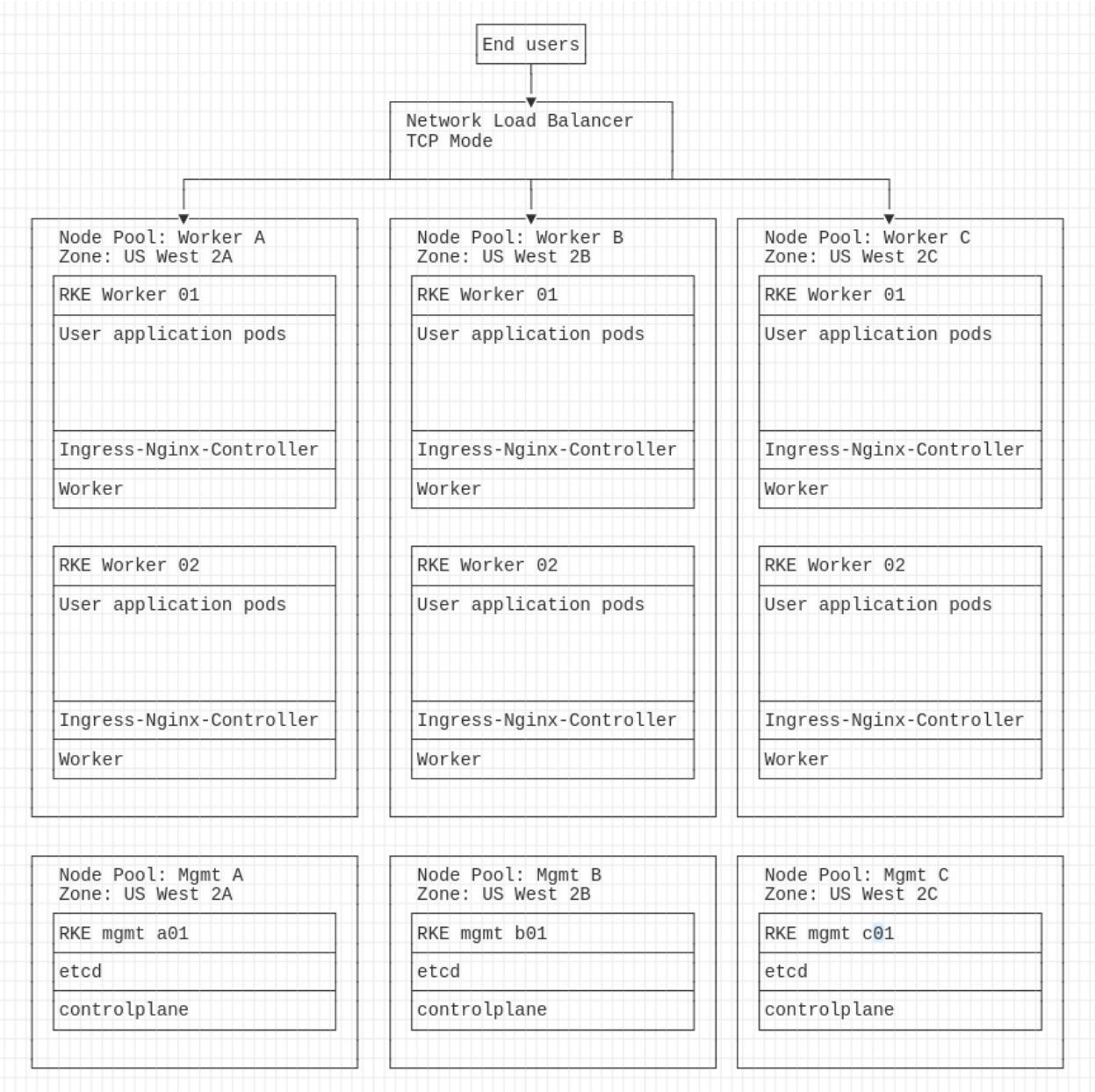
Chapter 6: Creating an RKE Cluster Using Rancher

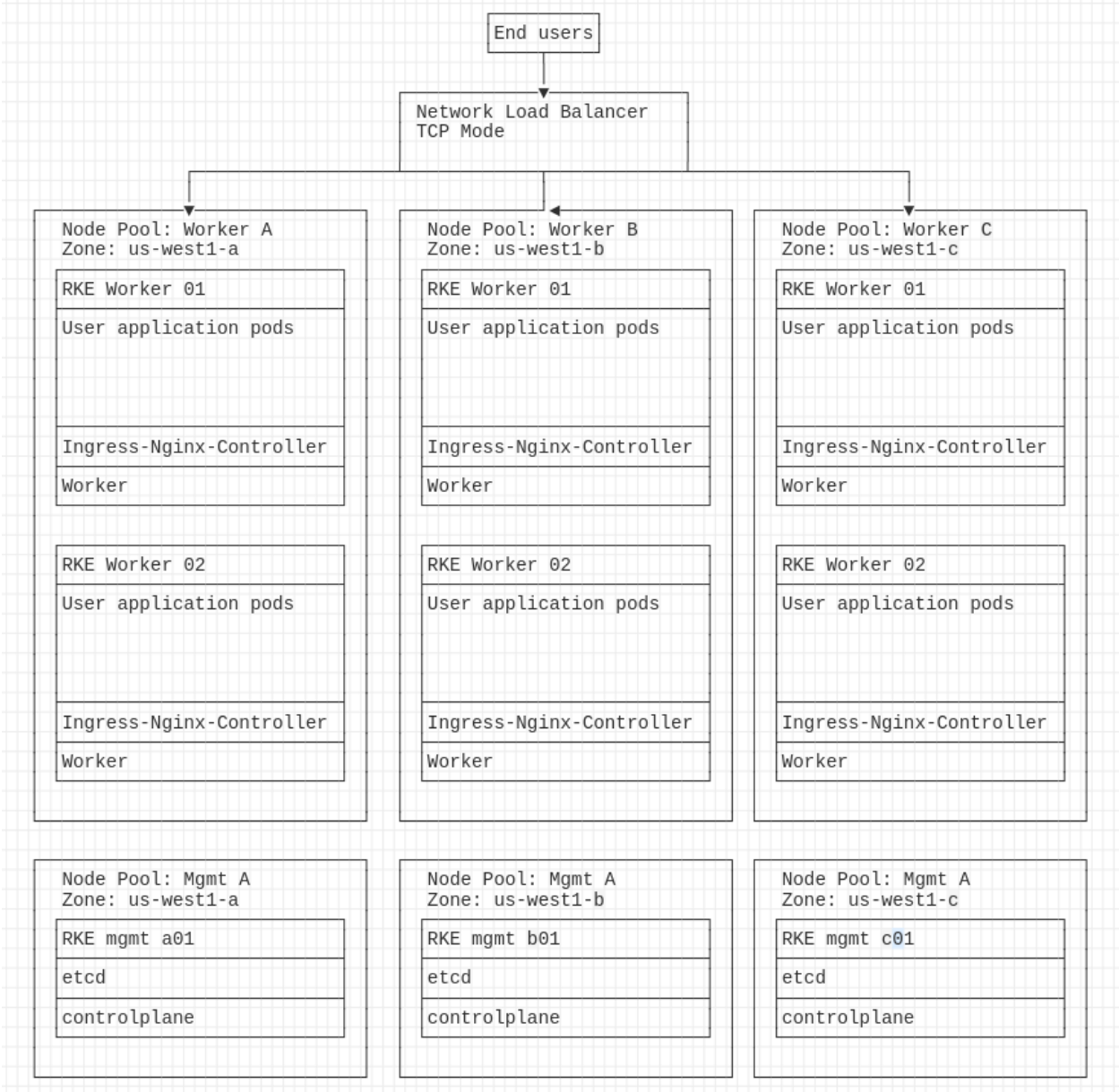
2 Run this command on one or more existing machines already running a supported version of Docker.

```
sudo docker run -d --privileged --restart=unless-stopped --net=host -v /etc/kubernetes:/etc/kubernetes -v /var/run:/var/run
rancher/rancher-agent:v2.4.4 --server https://example.rancher.com --token qkg67qfzzlqx8p4xg2gr2czt5ft52xqmmqtjfhwldtq52pb6s645lj
--ca-checksum 2bf373829900ec0c3f3539849577a122126040dccaea01556231b0eea813817d --worker
```

2 Run this command on one or more existing machines already running a supported version of Docker.

```
sudo docker run -d --privileged --restart=unless-stopped --net=host -v /etc/kubernetes:/etc/kubernetes -v /var/run:/var/run
rancher/rancher-agent:v2.4.4 --server https://example.rancher.com --token qkg67qfzzlqx8p4xg2gr2czt5ft52xqmmqtjfhwldtq52pb6s645lj
--ca-checksum 2bf373829900ec0c3f3539849577a122126040dccaea01556231b0eea813817d --node-name example-hostname --address 1.1.1.1 --
internal-address 10.0.0.1 --etcd --controlplane --worker
```


End users
Network Load Balancer
TCP Mode
Node Pool: Worker A
Zone: US West 2A
RKE Worker 01
User application pods
Ingress-Nginx-Controller
Worker
RKE Worker 02
User application pods
Ingress-Nginx-Controller
Worker
Node Pool: Worker B
Zone: US West 2B
RKE Worker 01
User application pods
Ingress-Nginx-Controller
Worker
RKE Worker 02
User application pods
Ingress-Nginx-Controller
Worker
Node Pool: Worker C
Zone: US West 2C
RKE Worker 01
User application pods
Ingress-Nginx-Controller
Worker
RKE Worker 02
User application pods
Ingress-Nginx-Controller
Worker
Node Pool: Mgmt A
Zone: US West 2A
RKE mgmt a01
etcd
controlplane
Node Pool: Mgmt B
Zone: US West 2B
RKE mgmt b01
etcd
controlplane
Node Pool: Mgmt C
Zone: US West 2C
RKE mgmt c01
etcd
controlplane


End users
Network Load Balancer
TCP Mode
Node Pool: Worker A
Zone: us-west1-a
RKE Worker 01
User application pods
Ingress-Nginx-Controller
Worker
RKE Worker 02
User application pods
Ingress-Nginx-Controller
Worker
Node Pool: Worker B
Zone: us-west1-b
RKE Worker 01
User application pods
Ingress-Nginx-Controller
Worker
RKE Worker 02
User application pods
Ingress-Nginx-Controller
Worker
Node Pool: Worker C
Zone: us-west1-c
RKE Worker 01
User application pods
Ingress-Nginx-Controller
Worker
RKE Worker 02
User application pods
Ingress-Nginx-Controller
Worker
Node Pool: Mgmt A
Zone: us-west1-a
RKE mgmt a01
etcd
controlplane
Node Pool: Mgmt A
Zone: us-west1-b
RKE mgmt b01
etcd
controlplane
Node Pool: Mgmt A
Zone: us-west1-c
RKE mgmt c01
etcd
controlplane

Cloud Credential: Create Amazon

Name *
A unique name

Description
Any text you want that better describes this resource

Access Key
Your AWS Access Key

SecretKey
Your AWS Secret Key

Default Region
us-west-2

The default region to use when creating clusters. Also contacted to verify that this credential works.

Cancel | Create

Add Node Template

Amazon EC2 | Azure | DigitalOcean | Rackspace | vSphere

AMAZON EC2 OPTIONS

1. Account Access
Choose the region and API Key that will be used to launch EC2 Instances

Region
us-west-2

Cloud Credentials
Rancher Example | Add New

Next: Authenticate & configure nodes | Cancel

4. Instance

Customize the EC2 Instance that will be created.

Instance Type

t2.medium

Spot Instance

☐ Request spot instance

Root Disk Size

16 GB

Encryption

☐ Encrypt EBS Volume

HTTP Endpoint

◉ Enabled ○ Disabled

Enables or disables the HTTP metadata endpoint on your instances

HTTP Tokens

○ Required ◉ Optional

Use HTTP Tokens for Instance Metadata Requests

AMI

An Ubuntu AMI

IAM Instance Profile Name

my-k8s-profile

SSH User

ubuntu

Private IP

☐ Use only private IP address

+ Add AWS Tag

Name *

Description

Name

Description

Labels

Labels are key/value pairs that can be used to annotate containers and make scheduling decisions.

+ Add Label

Taints

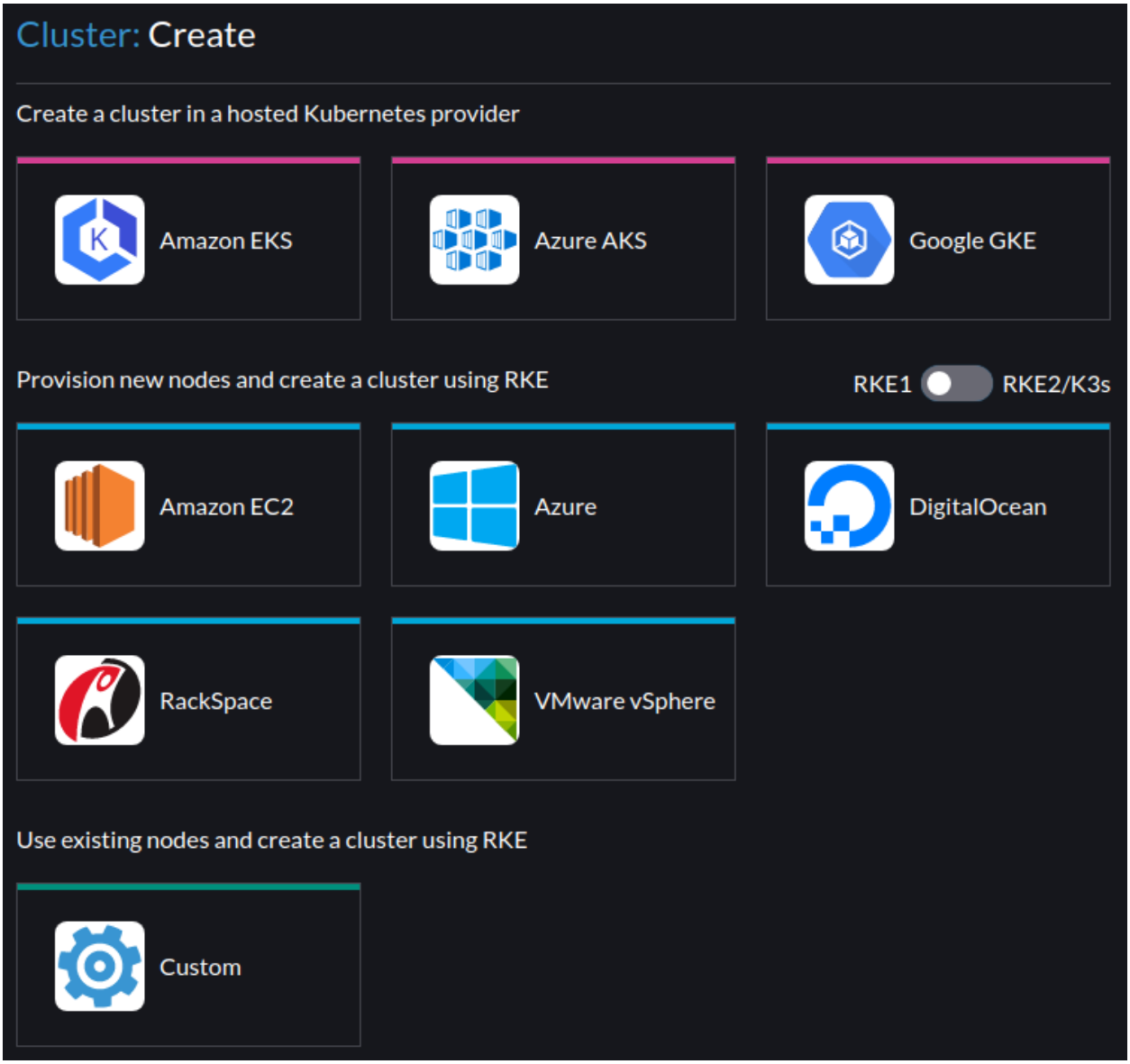
Taints mark that the node should not accept any pods that do not tolerate the taints.

Engine Options

Customize the configuration of the Docker daemon

Create

Cancel


Cluster: Create
Create a cluster in a hosted Kubernetes provider
Amazon EKS
Azure AKS
Google GKE
Provision new nodes and create a cluster using RKE
RKE1
RKE2/K3s
Amazon EC2
Azure
DigitalOcean
RackSpace
VMware vSphere
Use existing nodes and create a cluster using RKE
Custom

Add Cluster - Custom

Cluster Name *

Add a Description

Example Name

Member Roles
Control who has access to the cluster and what permission they have to change it.

Labels & Annotations
Configure labels and annotations for the cluster.

None

Cluster Options

Edit as YAML

Use an existing RKE Template and revision

Expand All

Kubernetes Options
Customize the kubernetes cluster options

Private Registry
Configure a default private registry for provisioning RKE cluster components. When enabled, all system images required for RKE cluster provisioning and system add-ons startup will be pulled from this registry.

Advanced Options
Customize advanced cluster options

Authorized Endpoint
Enabling the authorized cluster endpoint allows direct communication with the cluster, bypassing the API proxy. Authorized endpoints can be retrieved by generating a kubeconfig for the cluster.

Next

Cancel

Cluster Options

Customize Node Run Command

Editing node options will update the command you will run on your existing machines

1 **Node Options**

Choose what roles the node will have in the cluster.

Node Role

- [ ] etcd
- [ ] Control Plane
- [x] Worker

The cluster needs to have at least one node with each role in order for A1 Rancher Prod to finish provisioning.

Show advanced options

2 Run this command on one or more existing machines already running a supported version of Docker.

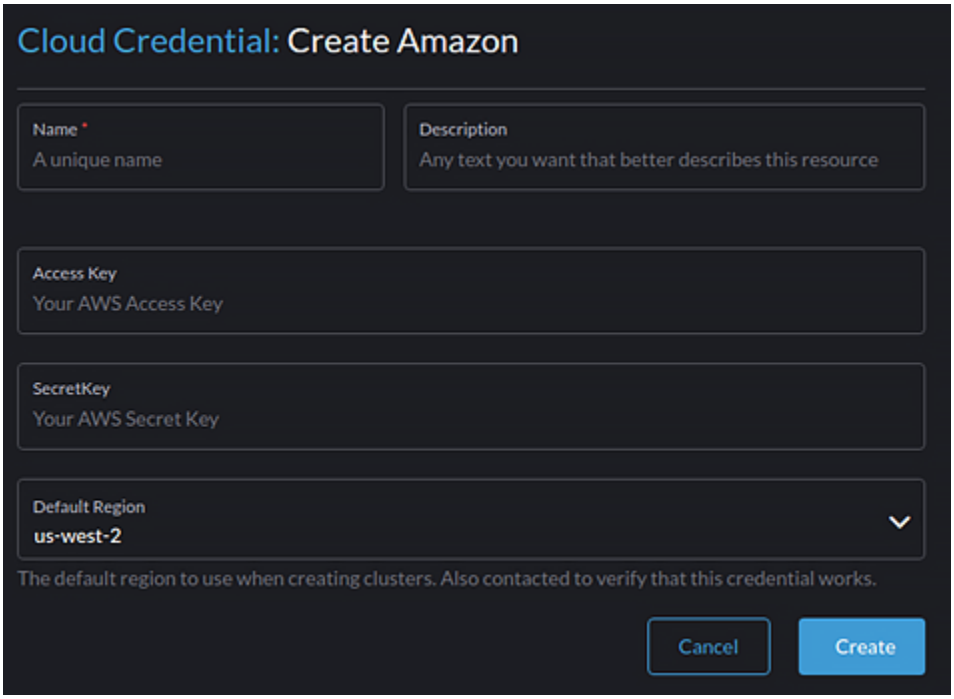
```
sudo docker run -d --privileged --restart=unless-stopped --net=host -v
/etc/kubernetes:/etc/kubernetes -v /var/run:/var/run  rancher/rancher-
agent:v2.6.2 --server https://example.rancher.com --token
kfn7m6qp7gkjh2l7m7tqjfj79b5dcfs6gbc62zwf2p7s6m9gwwmzsq --worker
```

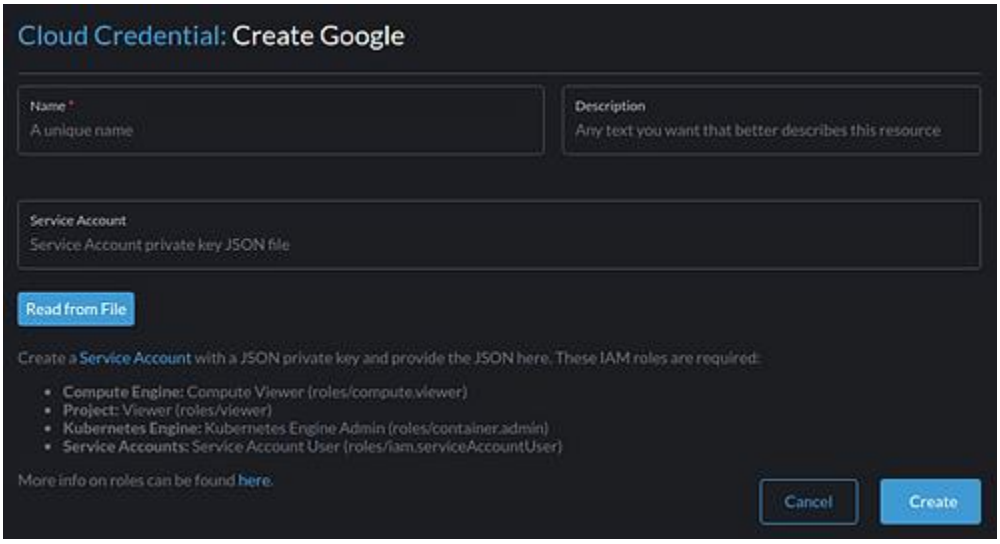
Name Prefix	Count	Template	Auto Replace	Drain Before Delete	etcd	Control Plane	Worker	Taints
	1	I	0 minutes	☐	☐	☐	☑	Taints

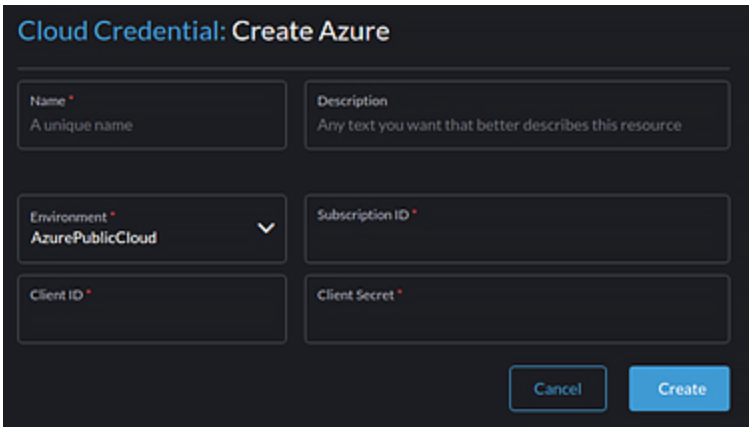
Number of nodes required: ⊗ 1, 3, or 5 ⊗ 1 or more ✓ 1 or more

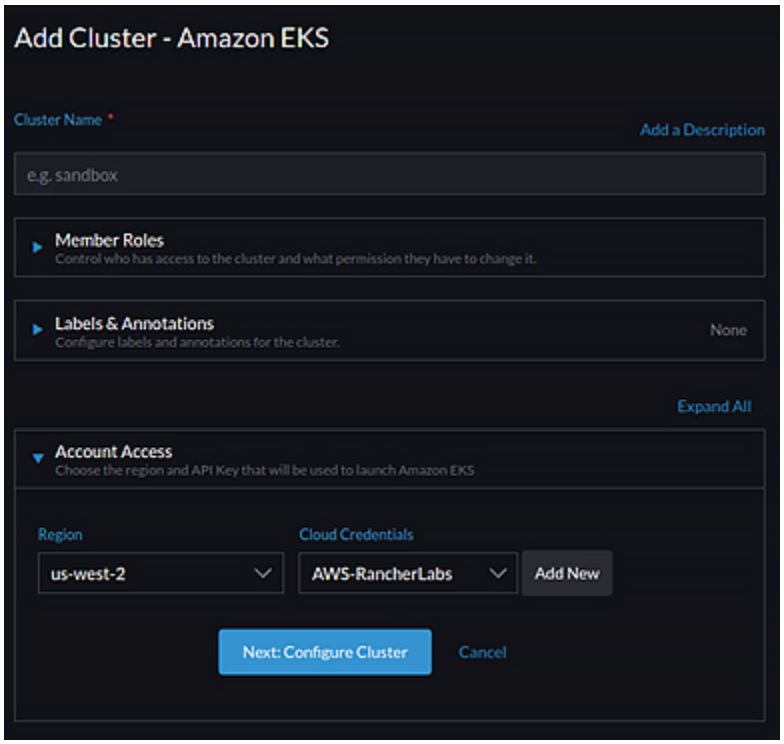
+ Add Node Pool

Chapter 7: Deploying a Hosted Cluster with Rancher






Cloud Credential: Create Azure
Name *
A unique name
Description
Any text you want that better describes this resource
Environment *
AzurePublicCloud
Subscription ID *
Client ID *
Client Secret *
Cancel
Create


Add Cluster - Amazon EKS
Cluster Name *
Add a Description
e.g. sandbox
Member Roles
Control who has access to the cluster and what permission they have to change it.
Labels & Annotations
Configure labels and annotations for the cluster.
None
Expand All
Account Access
Choose the region and API Key that will be used to launch Amazon EKS
Region
us-west-2
Cloud Credentials
AWS-RancherLabs
Add New
Next: Configure Cluster
Cancel

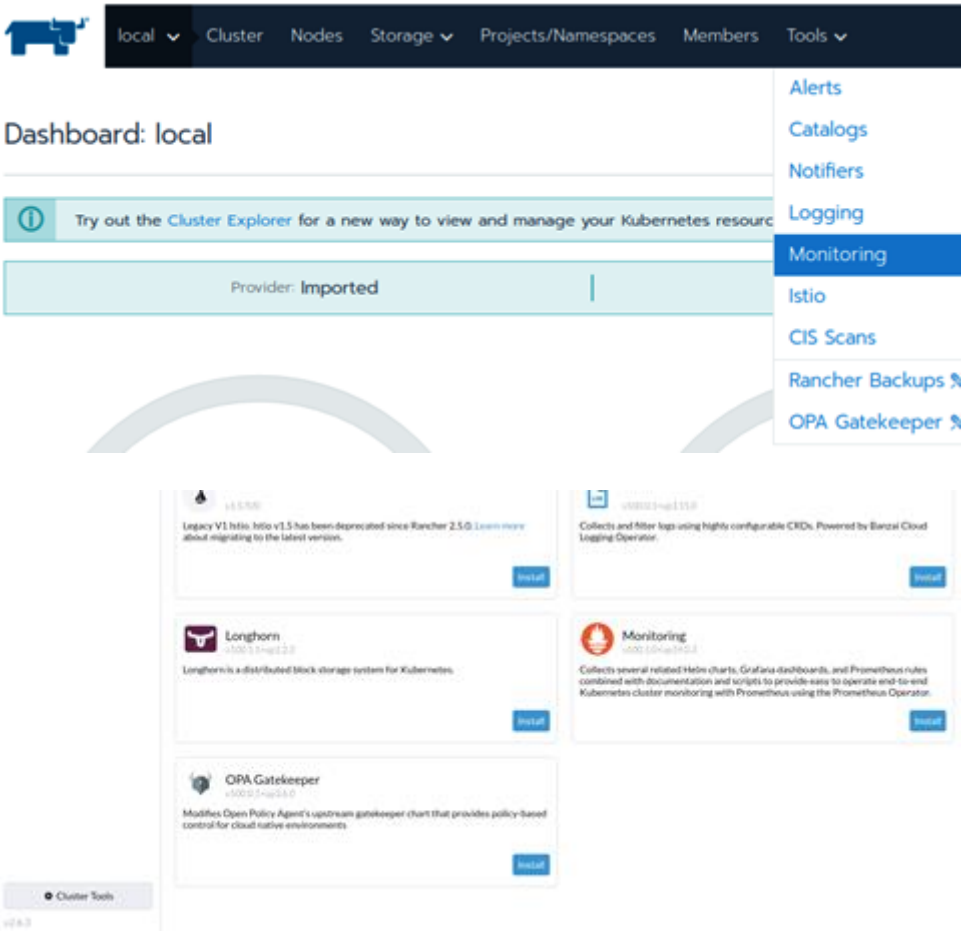
Chapter 8: Importing an Externally Managed Cluster into Rancher

No Images…

Chapter 9: Cluster Configuration Backup and Recovery

No Images…

Chapter 10: Monitoring and Logging



Chapter 11: Bring Storage to Kubernetes

```
                                        +---------+
                                        |End users|
                                        +----+----+
                                             |
                                +------------v------------+
                                | External Load Balancer  |
                                | TCP or HTTP Mode        |
                                |                         |
             +----------------+------------+------------+----------------+
             |                             |                             |
             |                             |                             |
+-----------v-------------+  +------------v------------+  +-------------v-----------+
|RKE Node 01              |  |RKE Node 02              |  |RKE Node 03              |
+-------------------------+  +-------------------------+  +-------------------------+
|User application pods    |  |User application pods    |  |User application pods    |
|                         |  |                         |  |                         |
|                         |  |                         |  |                         |
|                         |  |                         |  |                         |
|                         |  |                         |  |                         |
+-------------------------+  +-------------------------+  +-------------------------+
|Ingress-Nginx-Controller |  |Ingress-Nginx-Controller |  |Ingress-Nginx-Controller |
+-------------------------+  +-------------------------+  +-------------------------+
|Longhorn                 |  |Longhorn                 |  |Longhorn                 |
+-------------------------+  +-------------------------+  +-------------------------+
|Worker                   |  |Worker                   |  |Worker                   |
+-------------------------+  +-------------------------+  +-------------------------+
|Control-plane            |  |Control-plane            |  |Control-plane            |
+-------------------------+  +-------------------------+  +-------------------------+
|ETCD                     |  |ETCD                     |  |ETCD                     |
+-------------------------+  +-------------------------+  +-------------------------+
```

```
                                        +---------+
                                        |End users|
                                        +----+----+
                                             |
                                +------------v------------+
                                | External Load Balancer  |
                                | TCP or HTTP Mode        |
                                |                         |
             +----------------+------------+------------+----------------+
             |                             |                             |
             |                             |                             |
+-----------v-------------+  +------------v------------+  +-------------v-----------+
|RKE2 Node 01             |  |RKE2 Node 02             |  |RKE2 Node 03             |
+-------------------------+  +-------------------------+  +-------------------------+
|User application pods    |  |User application pods    |  |User application pods    |
|                         |  |                         |  |                         |
|                         |  |                         |  |                         |
|                         |  |                         |  |                         |
|                         |  |                         |  |                         |
+-------------------------+  +-------------------------+  +-------------------------+
|Ingress-Nginx-Controller |  |Ingress-Nginx-Controller |  |Ingress-Nginx-Controller |
+-------------------------+  +-------------------------+  +-------------------------+
|Longhorn                 |  |Longhorn                 |  |Longhorn                 |
+-------------------------+  +-------------------------+  +-------------------------+
|Worker                   |  |Worker                   |  |Worker                   |
+-------------------------+  +-------------------------+  +-------------------------+
|Master                   |  |Master                   |  |Master                   |
+-------------------------+  +-------------------------+  +-------------------------+
```

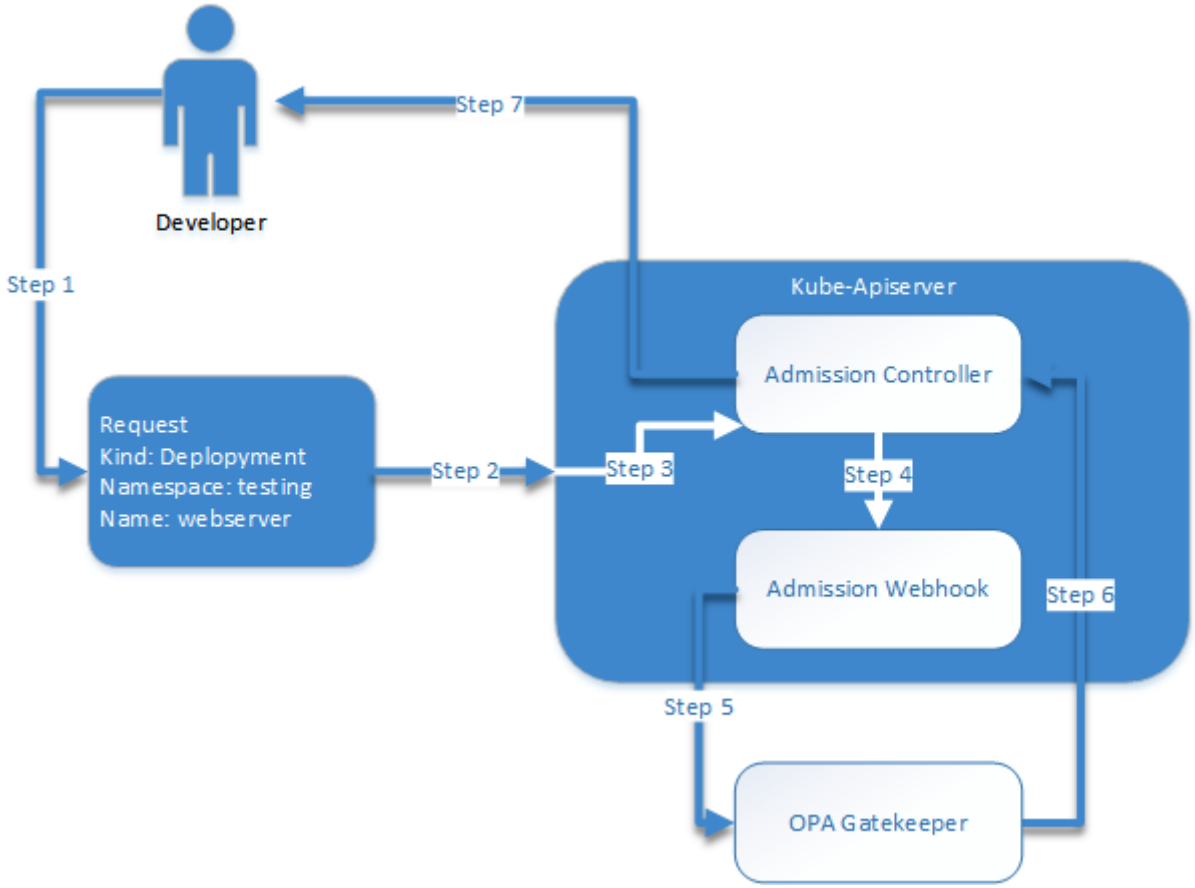
```
                                  +---------+
                                  |End users|
                                  +----+----+
                                       |
                             +------------v------------+
                             | External Load Balancer  |
                             | TCP or HTTP Mode        |
                             |                         |
           +----------------+------------+------------+--------------+
           |                             |                           |
+----------v--------------+ +------------v------------+ +------------v------------+
|RKE Work 01              | |RKE Work 02              | |RKE Work 03              |
+-------------------------+ +-------------------------+ +-------------------------+
|User application pods    | |User application pods    | |User application pods    |
|                         | |                         | |                         |
|                         | |                         | |                         |
|                         | |                         | |                         |
|                         | |                         | |                         |
+-------------------------+ +-------------------------+ +-------------------------+
|Ingress-Nginx-Controller | |Ingress-Nginx-Controller | |Ingress-Nginx-Controller |
+-------------------------+ +-------------------------+ +-------------------------+
|Longhorn                 | |Longhorn                 | |Longhorn                 |
+-------------------------+ +-------------------------+ +-------------------------+
|Worker                   | |Worker                   | |Worker                   |
+-------------------------+ +-------------------------+ +-------------------------+

+-------------------------+ +-------------------------+ +-------------------------+
|RKE Mgmt 01              | |RKE Mgmt 02              | |RKE Mgmt 03              |
+-------------------------+ +-------------------------+ +-------------------------+
|Control-plane            | |Control-plane            | |Control-plane            |
+-------------------------+ +-------------------------+ +-------------------------+
|ETCD                     | |ETCD                     | |ETCD                     |
+-------------------------+ +-------------------------+ +-------------------------+
```

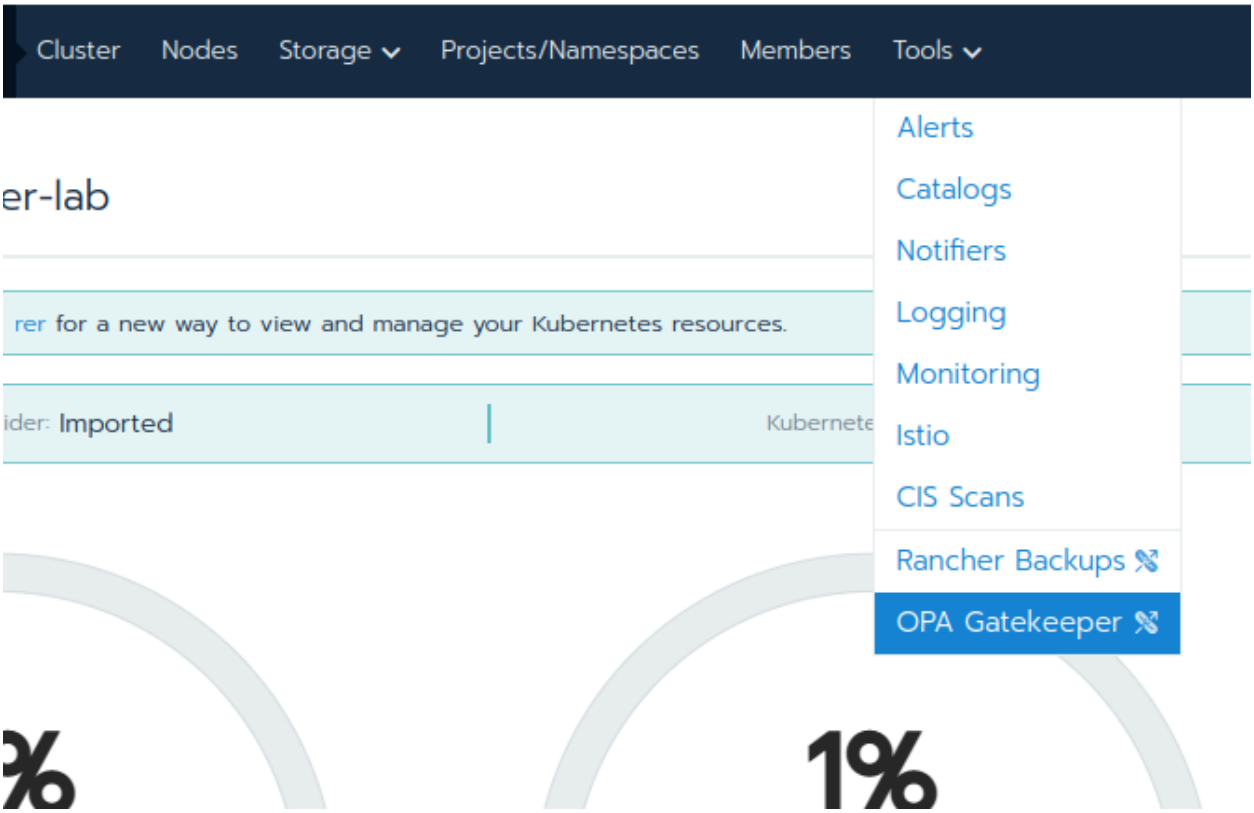
```
                                  +---------+
                                  |End users|
                                  +----+----+
                                       |
                             +------------v------------+
                             | External Load Balancer  |
                             | TCP or HTTP Mode        |
                             | Ports: 80 & 443         |
           +----------------+------------+------------+--------------+
           |                             |                           |
+----------v--------------+ +------------v------------+ +------------v------------+
|RKE2 Work01              | |RKE2 Work02              | |RKE2 Work  N             |
+-------------------------+ +-------------------------+ +-------------------------+
|User application pods    | |User application pods    | |User application pods    |
|                         | |                         | |                         |
|                         | |                         | |                         |
|                         | |                         | |                         |
|                         | |                         | |                         |
+-------------------------+ +-------------------------+ +-------------------------+
|Ingress-Nginx-Controller | |Ingress-Nginx-Controller | |Ingress-Nginx-Controller |
+-------------------------+ +-------------------------+ +-------------------------+
|Longhorn                 | |Longhorn                 | |Longhorn                 |
+-------------------------+ +-------------------------+ +-------------------------+
|Worker                   | |Worker                   | |Worker                   |
+-------------------------+ +-------------------------+ +-------------------------+

+-------------------------+ +-------------------------+ +-------------------------+
|RKE Master 01            | |RKE Master 02            | |RKE Master 03            |
+-------------------------+ +-------------------------+ +-------------------------+
|Master                   | |Master                   | |Master                   |
+-----------^-------------+ +------------^------------+ +-------------^-----------+
            |                             |                           |
            +---------------+------------+------------+---------------+
                             |External Load Balancer   |
                             |           or            |
                             |Round-robin DNS          |
                             |Ports: 9345 & 6443 TCP   |
                             +-------------------------+
```

```
                                                  +---------+
                                                  |End users|
                                                  +----+----+
                                                       |
                                             +------------v------------+
                                             | External Load Balancer  |
                                             | TCP or HTTP Mode        |
                                             |                         |
          +----------------------------+--------------+------------+--------------+-------------------------+
          |                            |              |            |              |                         |
+----------v-------------+ +------------v------------+ +------------v------------+ +------------v------------+ +------------v------------+
|RKE Work 01             | |RKE Work 02              | |RKE Work 03              | |RKE Work 04              | |RKE Work  N              |
+------------------------+ +-------------------------+ +-------------------------+ +-------------------------+ +-------------------------+
|User application pods   | |User application pods    | |User application pods    | |User application pods    | |User application pods    |
|                        | |                         | |                         | |                         | |                         |
|                        | |                         | |                         | |                         | |                         |
|                        | |                         | |                         | |                         | |                         |
|                        | |                         | |                         | |                         | |                         |
+------------------------+ +-------------------------+ +-------------------------+ +-------------------------+ +-------------------------+
|Ingress-Nginx-Controller| |Ingress-Nginx-Controller | |Ingress-Nginx-Controller | |Ingress-Nginx-Controller | |Ingress-Nginx-Controller |
+------------------------+ +-------------------------+ +-------------------------+ +-------------------------+ +-------------------------+
|Worker                  | |Worker                   | |Worker                   | |Worker                   | |Worker                   |
+------------------------+ +-------------------------+ +-------------------------+ +-------------------------+ +-------------------------+

+----------v-------------+ +------------v------------+ +------------v------------+ +------------v------------+ +------------v------------+
|Longhorn 01             | |Longhorn 02              | |Longhorn 03              | |Longhorn 04              | |Longhorn  5              |
+------------------------+ +-------------------------+ +-------------------------+ +-------------------------+ +-------------------------+
|Longhorn                | |Longhorn                 | |Longhorn                 | |Longhorn                 | |Longhorn                 |
+------------------------+ +-------------------------+ +-------------------------+ +-------------------------+ +-------------------------+

               +------------------------+ +------------------------+ +------------------------+ +------------------------+
               |RKE CP 01               | |RKE CP 02               | |RKE CP 03               | |RKE CP 04               |
               +------------------------+ +------------------------+ +------------------------+ +------------------------+
               |Control-plane           | |Control-plane           | |Control-plane           | |Control-plane           |
               +------------------------+ +------------------------+ +------------------------+ +------------------------+

+------------------------+ +------------------------+ +------------------------+ +------------------------+ +------------------------+
|RKE ETCD 01             | |RKE ETCD 02             | |RKE ETCD 03             | |RKE ETCD 04             | |RKE ETCD 05             |
+------------------------+ +------------------------+ +------------------------+ +------------------------+ +------------------------+
|ETCD plane              | |ETCD plane              | |ETCDrplane              | |ETCD plane              | |ETCD plane              |
+------------------------+ +------------------------+ +------------------------+ +------------------------+ +------------------------+
```

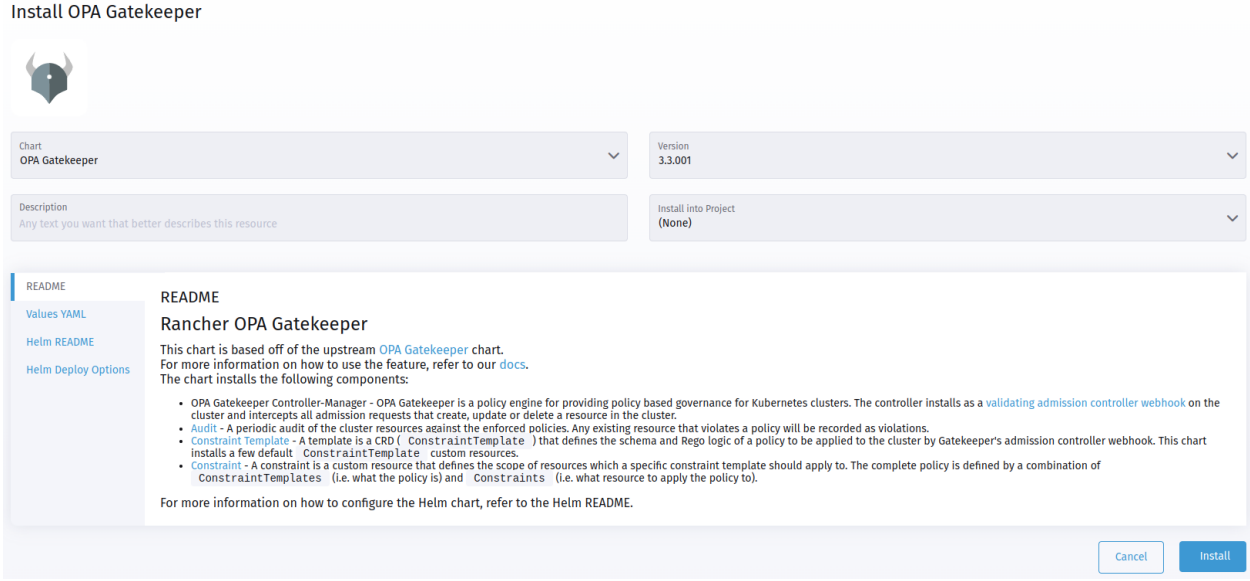
```
                                                  +---------+
                                                  |End users|
                                                  +----+----+
                                                       |
                                             +------------v------------+
                                             | External Load Balancer  |
                                             | TCP or HTTP Mode        |
                                             | Ports: 80 & 443         |
                                       +----------------+------------+------------+--------------+
                                       |                             |                          |
                                 +----------v-------------+ +------------v------------+ +------------v------------+
                                 |RKE2 Work01             | |RKE2 Work02              | |RKE2 Work  N             |
                                 +------------------------+ +-------------------------+ +-------------------------+
                                 |User application pods   | |User application pods    | |User application pods    |
                                 |                        | |                         | |                         |
                                 |                        | |                         | |                         |
                                 |                        | |                         | |                         |
                                 |                        | |                         | |                         |
                                 +------------------------+ +-------------------------+ +-------------------------+
                                 |Ingress-Nginx-Controller| |Ingress-Nginx-Controller | |Ingress-Nginx-Controller |
                                 +------------------------+ +-------------------------+ +-------------------------+
                                 |Longhorn                | |Longhorn                 | |Longhorn                 |
                                 +------------------------+ +-------------------------+ +-------------------------+
                                 |Worker                  | |Worker                   | |Worker                   |
                                 +------------------------+ +-------------------------+ +-------------------------+

          +----------------------------+--------------+------------+------------+--------------+-------------------------+
          |                            |              |            |              |                         |
+----------v-------------+ +------------v------------+ +------------v------------+ +------------v------------+ +------------v------------+
|RKE Work 01             | |RKE Work 02              | |RKE Work 03              | |RKE Work 04              | |RKE Work  N              |
+------------------------+ +-------------------------+ +-------------------------+ +-------------------------+ +-------------------------+
|User application pods   | |User application pods    | |User application pods    | |User application pods    | |User application pods    |
|                        | |                         | |                         | |                         | |                         |
|                        | |                         | |                         | |                         | |                         |
|                        | |                         | |                         | |                         | |                         |
|                        | |                         | |                         | |                         | |                         |
+------------------------+ +-------------------------+ +-------------------------+ +-------------------------+ +-------------------------+
|Ingress-Nginx-Controller| |Ingress-Nginx-Controller | |Ingress-Nginx-Controller | |Ingress-Nginx-Controller | |Ingress-Nginx-Controller |
+------------------------+ +-------------------------+ +-------------------------+ +-------------------------+ +-------------------------+
|Worker                  | |Worker                   | |Worker                   | |Worker                   | |Worker                   |
+------------------------+ +-------------------------+ +-------------------------+ +-------------------------+ +-------------------------+

+------------------------+ +------------------------+ +------------------------+ +------------------------+ +------------------------+
|Longhorn 01             | |Longhorn 02             | |Longhorn 03             | |Longhorn 04             | |Longhorn  5             |
+------------------------+ +------------------------+ +------------------------+ +------------------------+ +------------------------+
|Longhorn                | |Longhorn                | |Longhorn                | |Longhorn                | |Longhorn                |
+------------------------+ +------------------------+ +------------------------+ +------------------------+ +------------------------+

+------------------------+ +------------------------+ +------------------------+ +------------------------+ +------------------------+
|Master 01               | |Master 02               | |Master 03               | |Master 04               | |Master 05               |
+------------------------+ +------------------------+ +------------------------+ +------------------------+ +------------------------+
|Master                  | |Master                  | |Master                  | |Master                  | |Master                  |
+------------------------+ +------------------------+ +------------------------+ +------------------------+ +------------------------+
```

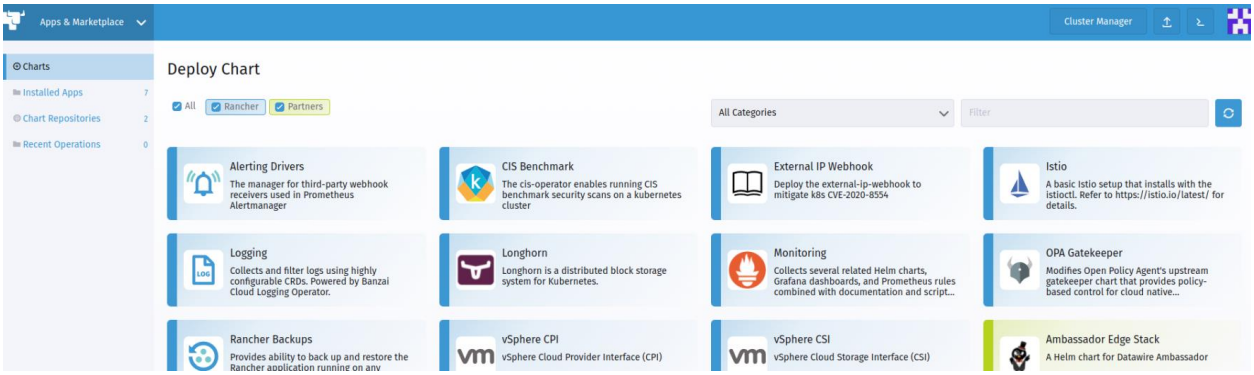
Chapter 12: Security and Compliance (OPA Gatekeeper)

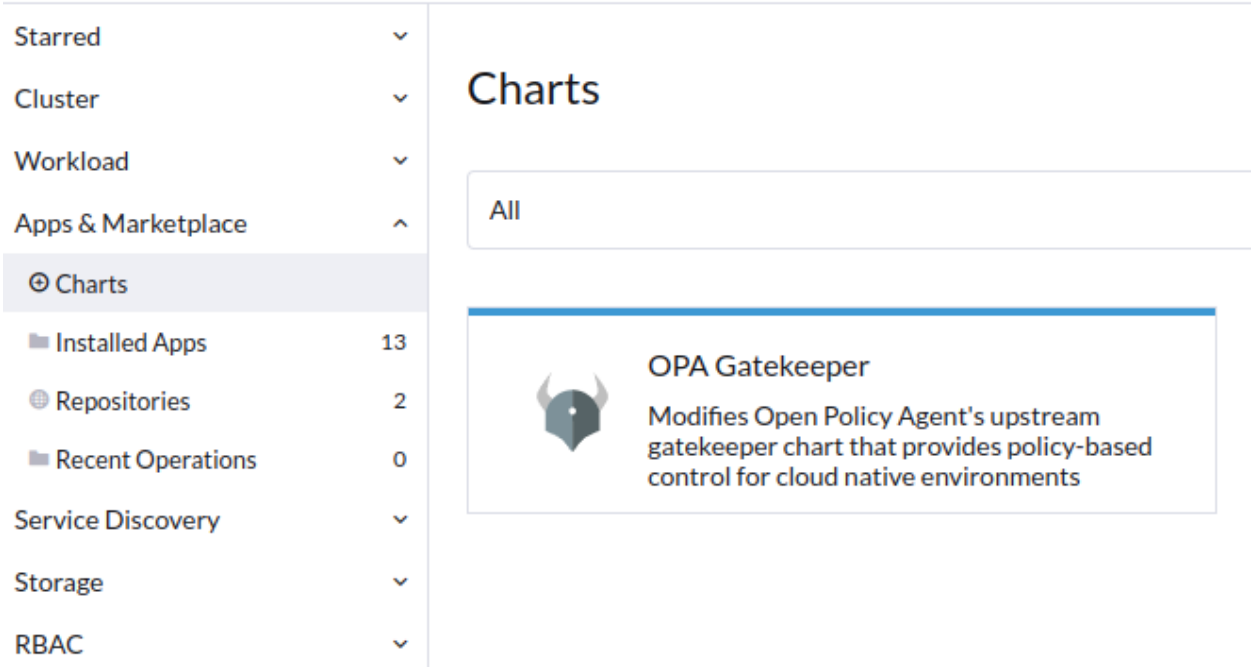




Install OPA Gatekeeper




Apps & Marketplace
Cluster Manager
Charts
Installed Apps 7
Chart Repositories 2
Recent Operations 0
Deploy Chart
All
Rancher
Partners
All Categories
Filter
Alerting Drivers
The manager for third-party webhook receivers used in Prometheus Alertmanager
CIS Benchmark
The cis-operator enables running CIS benchmark security scans on a kubernetes cluster
External IP Webhook
Deploy the external-ip-webhook to mitigate k8s CVE-2020-8554
Istio
A basic Istio setup that installs with the istioctl. Refer to https://istio.io/latest/ for details.
Logging
Collects and filter logs using highly configurable CRDs. Powered by Banzai Cloud Logging Operator.
Longhorn
Longhorn is a distributed block storage system for Kubernetes.
Monitoring
Collects several related Helm charts, Grafana dashboards, and Prometheus rules combined with documentation and script...
OPA Gatekeeper
Modifies Open Policy Agent's upstream gatekeeper chart that provides policy-based control for cloud native...
Rancher Backups
Provides ability to back up and restore the Rancher application running on any
vSphere CPI
vSphere Cloud Provider Interface (CPI)
vSphere CSI
vSphere Cloud Storage Interface (CSI)
Ambassador Edge Stack
A Helm chart for Datawire Ambassador


Starred
Cluster
Workload
Apps & Marketplace
Charts
Installed Apps 13
Repositories 2
Recent Operations 0
Service Discovery
Storage
RBAC
Charts
All
OPA Gatekeeper
Modifies Open Policy Agent's upstream gatekeeper chart that provides policy-based control for cloud native environments

Charts: OPA Gatekeeper (100.0.1+up3.6.0)

Install

Modifies Open Policy Agent's upstream gatekeeper chart that provides policy-based control for cloud native environments

This chart is based off of the upstream OPA Gatekeeper chart.
For more information on how to use the feature, refer to our docs.
The chart installs the following components:

- OPA Gatekeeper Controller-Manager - OPA Gatekeeper is a policy engine for providing policy based governance for Kubernetes clusters. The controller installs as a validating admission controller webhook on the cluster and intercepts all admission requests that create, update or delete a resource in the cluster.
- Audit - A periodic audit of the cluster resources against the enforced policies. Any existing resource that violates a policy will be recorded as violations.
- Constraint Template - A template is a CRD (`ConstraintTemplate`) that defines the schema and Rego logic of a policy to be applied to the cluster by Gatekeeper's admission controller webhook. This chart installs a few default `ConstraintTemplate` custom resources.
- Constraint - A constraint is a custom resource that defines the scope of resources which a specific constraint template should apply to. The complete policy is defined by a combination of `ConstraintTemplates` (i.e. what the policy is) and `Constraints` (i.e. what resource to apply the policy to).

Chart Versions

100.0.1+up3.6.0 Wed, Jan 5 2022

Application Version

v3.6.0

Home

https://github.com/open-policy-agent/gatekeeper

Related

https://github.com/open-policy-agent/gatekeeper.git

OPA Gatekeeper
100.0.1+up3.6.0

Install: Step 1
Set App metadata

Metadata

This process will help create the chart. Start by setting some basic information used by A1 Rancher Prod to manage the App.

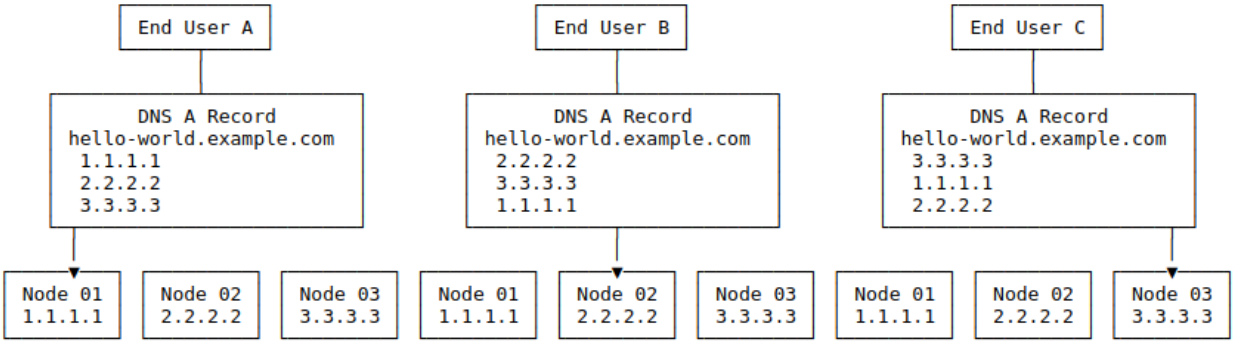
Install into Project
System

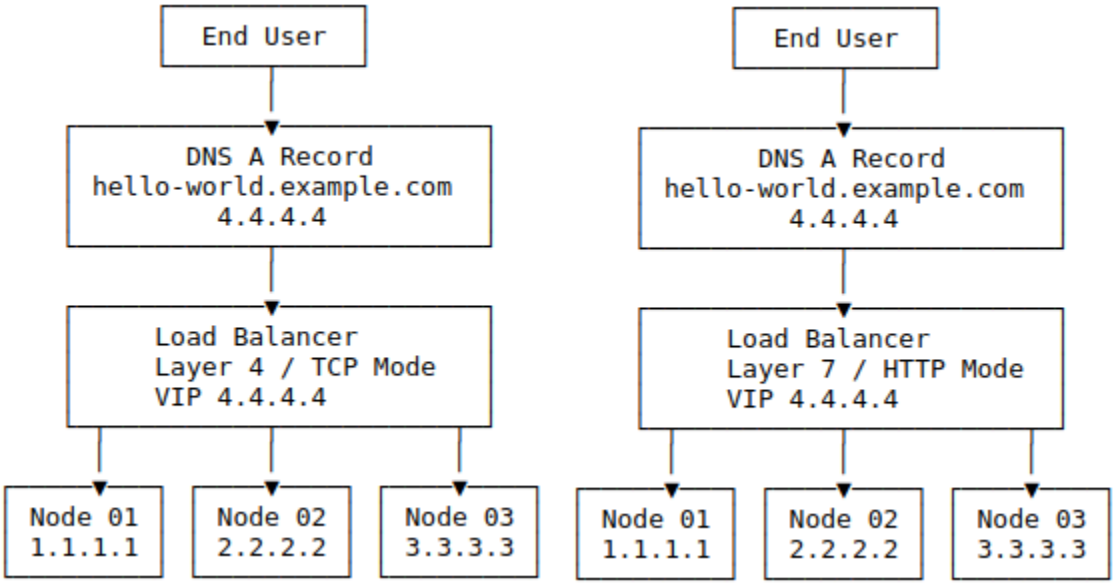
Chapter 13: Scaling in Kubernetes

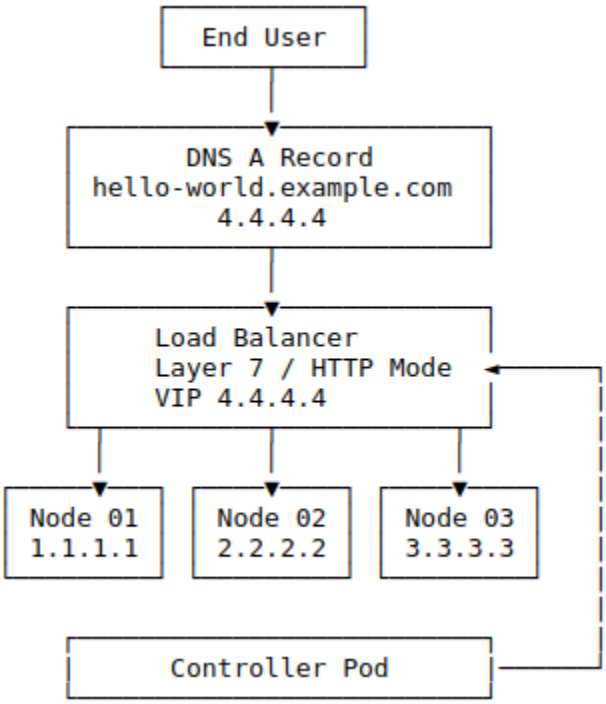
```
 Reason                       Age                  From                       Message
 ------                       ----                 ----                       -------
 SuccessfulRescale            11m                  horizontal-pod-autoscaler  New size: 8; reas
 SuccessfulRescale            11m                  horizontal-pod-autoscaler  New size: 10; rea
 SuccessfulRescale            10m (x2 over 11m)    horizontal-pod-autoscaler  New size: 4; reas
 SuccessfulRescale            9m53s                horizontal-pod-autoscaler  New size: 8; reas
 SuccessfulRescale            9m37s                horizontal-pod-autoscaler  New size: 10; rea
 FailedGetResourceMetric      4m19s                horizontal-pod-autoscaler  failed to get cpu
 FailedComputeMetricsReplicas 4m18s                horizontal-pod-autoscaler  invalid metrics (
ics returned from resource metrics API
 FailedGetResourceMetric      3m18s (x6 over 10m)  horizontal-pod-autoscaler  failed to get cpu
 FailedComputeMetricsReplicas 3m18s (x6 over 10m)  horizontal-pod-autoscaler  invalid metrics (

 SuccessfulRescale            3m3s                 horizontal-pod-autoscaler  New size: 1; reas
```

```
Status:
  Conditions:
    Last Transition Time:  2022-02-20T11:08:45Z
    Status: True
    Type: RecommendationProvided
  Recommendation:
    Container Recommendations:
      Container Name: hello-world
      Lower Bound:
        Cpu: 12m
        Memory: 247917233
      Target:
        Cpu: 63m
        Memory: 380258472
      Uncapped Target:
        Cpu: 63m
        Memory: 380258472
      Upper Bound:
        Cpu: 137m
        Memory: 561393834
      Container Name:  hello-world
      Lower Bound:
        Cpu: 12m
        Memory: 131072k
      Target:
        Cpu: 12m
        Memory: 131072k
      Uncapped Target:
        Cpu: 12m
        Memory: 131072k
      Upper Bound:
        Cpu: 16m
        Memory: 131072k
Events: <none>
```

Chapter 14: Load Balancer Configuration and SSL Certificates






End User
DNS A Record
hello-world.example.com
4.4.4.4
Load Balancer
Layer 7 / HTTP Mode
VIP 4.4.4.4
Node 01
1.1.1.1
Node 02
2.2.2.2
Node 03
3.3.3.3
Controller Pod

Configuration: Basic

Name	Rancher_80
Description	
Health Monitors	Active: /Common http << >> Available: /Common gateway_icmp, http2, http2_head_f5, http_head_f5

Resources

Load Balancing Method	Round Robin
Priority Group Activation	Disabled
New Members	(●) New Node ( ) New FQDN Node Node Name: node03 (Optional) Address: 192.168.1.103 Service Port: 80 HTTP Add

Node Name	Address/FQDN	Service Port	Auto Populate	Priority
node01	192.168.1.101	80		0
node02	192.168.1.102	80		0
node03	192.168.1.103	80		0

Edit Delete

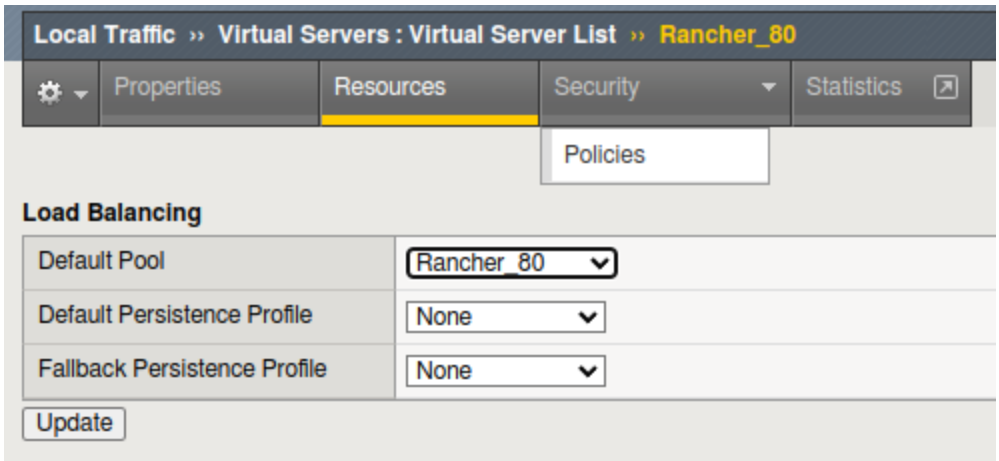
Cancel Repeat Finished

General Properties

Name	Rancher_80
Partition / Path	Common
Description	
Type	Performance (Layer 4)
Source Address	Host Address List 0.0.0.0/0
Destination Address/Mask	Host Address List 192.168.1.99
Service Port	Port Port List 80 HTTP
Notify Status to Virtual Address	☑
Availability	Unknown (Enabled) - The children pool member(s) either don't have service check
Syncookie Status	Inactive
State	Enabled

Configuration: Basic

DoH Profile Type	None
Protocol	TCP
Protocol Profile (Client)	fastL4
HTTP Profile (Client)	None
HTTP Profile (Server)	(Use Client Profile)
HTTP Proxy Connect Profile	None
VLAN and Tunnel Traffic	All VLANs and Tunnels
Source Address Translation	None



Debian/Ubuntu HAProxy packages

The Debian HAProxy packaging team provides various versions of HAProxy packages for use on different Debian or Ubuntu systems. The following wizard helps you to find the package suitable for your system.

« *I am running* Ubuntu Focal (20.04 LTS) *and I want to install HAProxy version* 2.5-stable.

Instructions for latest release

You need to enable a dedicated PPA with the following command:

```
# apt-get install --no-install-recommends software-properties-common
# add-apt-repository ppa:vbernat/haproxy-2.5
```

Then, use the following command:

```
# apt-get install haproxy=2.5.\*
```

You will get the *latest* release of HAProxy 2.5 (and stick to this branch).

```
frontend http_frontend
    mode tcp
    bind :80
    default_backend http_backend

backend http_backend
    mode tcp
    balance leastconn
    server node01 192.168.1.101:80 check
    server node02 192.168.1.102:80 check
    server node03 192.168.1.103:80 check

frontend https_frontend
    mode tcp
    bind :80
    default_backend https_backend

backend https_backend
    mode tcp
    balance leastconn
    server node01 192.168.1.101:443 check
    server node02 192.168.1.102:443 check
    server node03 192.168.1.103:443 check
```

```
frontend frontend:
    bind *:80
    bind *:443 ssl crt /etc/haproxy/certs/star.example.com.pem
    http-request redirect scheme https unless { ssl_fc }
    mode http

    acl host_rke-cluster-npd hdr(host) -i dev.example.com
    acl host_rke-cluster-npd hdr(host) -i qas.example.com
    use_backend rke-cluster-npd-https if host_a1-rke-cluster-npd

    acl host_rke-cluster-prd hdr(host) -i example.com
    acl host_rke-cluster-prd hdr(host) -i www.example.com
    use_backend rke-cluster-prd-https if host_a1-rke-cluster-prd
```

```
backend rke-cluster-npd-https
    mode http
    option httpchk HEAD /healthz HTTP/1.0
    server node01 192.168.1.101:443 check weight 1 maxconn 1024 ssl verify none
    server node02 192.168.1.102:443 check weight 1 maxconn 1024 ssl verify none
    server node03 192.168.1.103:443 check weight 1 maxconn 1024 ssl verify none

backend rke-cluster-prd-https
    mode http
    option httpchk HEAD /healthz HTTP/1.0
    server node04 192.168.1.104:443 check weight 1 maxconn 1024 ssl verify none
    server node05 192.168.1.105:443 check weight 1 maxconn 1024 ssl verify none
    server node06 192.168.1.106:443 check weight 1 maxconn 1024 ssl verify none
```

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  namespace: metallb-system
  name: config
data:
  config: |
    address-pools:
    - name: default
      protocol: layer2
      addresses:
      - 192.168.2.2-192.168.2.125
```

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  namespace: metallb-system
  name: config
data:
  config: |
    peers:
    - peer-address: 10.0.0.1
      peer-asn: 64501
      my-asn: 64500
    address-pools:
    - name: default
      protocol: bgp
      addresses:
      - 192.168.10.0/24
```

```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: ingress-wildcard-host
spec:
  rules:
  - host: "foo.bar.com"
    http:
      paths:
      - pathType: Prefix
        path: "/bar"
        backend:
          service:
            name: service1
            port:
              number: 80
  - host: "*.foo.com"
    http:
      paths:
      - pathType: Prefix
        path: "/foo"
        backend:
          service:
            name: service2
            port:
              number: 80
```

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: testsecret-tls
  namespace: default
data:
  tls.crt: base64 encoded cert
  tls.key: base64 encoded key
type: kubernetes.io/tls
```

```yaml
tls:
- hosts:
    - https-example.foo.com
  secretName: testsecret-tls
```

Chapter 15: Rancher and Kubernetes Troubleshooting

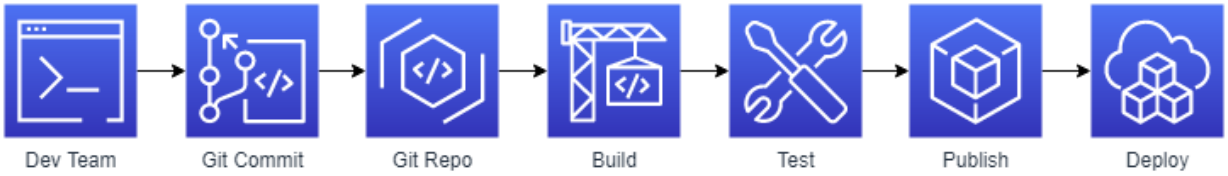
```
etcd-a1ubrked01
+-------------------------+---------+------------+-------+
|        ENDPOINT         | HEALTH  |   TOOK     | ERROR |
+-------------------------+---------+------------+-------+
| https://127.0.0.1:2379  |   true  | 6.96521ms  |       |
+-------------------------+---------+------------+-------+

etcd-a1ubrked02
+-------------------------+---------+-------------+-------+
|        ENDPOINT         | HEALTH  |    TOOK     | ERROR |
+-------------------------+---------+-------------+-------+
| https://127.0.0.1:2379  |   true  | 70.546814ms |       |
+-------------------------+---------+-------------+-------+

etcd-a1ubrked03
+-------------------------+---------+------------+-------+
|        ENDPOINT         | HEALTH  |    TOOK    | ERROR |
+-------------------------+---------+------------+-------+
| https://127.0.0.1:2379  |   true  | 14.07386ms |       |
+-------------------------+---------+------------+-------+

etcd-a1ubrked04
+-------------------------+---------+------------+-------+
|        ENDPOINT         | HEALTH  |    TOOK    | ERROR |
+-------------------------+---------+------------+-------+
| https://127.0.0.1:2379  |   true  | 8.447933ms |       |
+-------------------------+---------+------------+-------+

etcd-a1ubrked05
+-------------------------+---------+---------+-------+
|        ENDPOINT         | HEALTH  |  TOOK   | ERROR |
+-------------------------+---------+---------+-------+
| https://127.0.0.1:2379  |   true  | 7.038ms |       |
+-------------------------+---------+---------+-------+
```

```
apiVersion: v1
kind: Namespace
metadata:
  annotations:
    cattle.io/status: '{"Conditions":[{"Type":"InitialRolesPopulate
      resource quota","LastUpdateTime":""},{"Type":"ResourceQuotaIn
    field.cattle.io/creatorId: u-zbw65mdtej
    field.cattle.io/projectId: c-x8rzf:p-pr7zr
    field.cattle.io/resourceQuota: "null"
  creationTimestamp: "2022-03-19T14:00:17Z"
  deletionTimestamp: "2022-03-19T14:02:05Z"
  labels:
    cattle.io/creator: norman
    field.cattle.io/projectId: p-pr7zr
  name: cattle-logging
  resourceVersion: "5872"
  selfLink: /api/v1/namespaces/t
  uid: 6a975c42-0396-11e9-bd3b-aaaaaaaaaa4a
spec:
  finalizers:
  - kubernetes
status:
  phase: Terminating
```

```yaml
apiVersion: v1
 kind: Namespace
 metadata:
   creationTimestamp: 2022-03-19T18:48:30Z
   deletionTimestamp: 2022-03-19T18:59:36Z
   name: test-namespace
   resourceVersion: "1385077"
   selfLink: /api/v1/namespaces/test-namespace
   uid: b50c9ea4-ec2b-11e8-a0be-fa163eeb47a5
 spec:
   finalizers:
   - kubernetes
 status:
   phase: Terminating
```

```json
{
    "apiVersion": "v1",
    "kind": "Namespace",
    "metadata": {
        "annotations": {
            "cattle.io/status": "{\"Conditions\":[{\"Type\":\"InitialRo
            "field.cattle.io/creatorId": "user-sw4mg",
            "field.cattle.io/projectId": "c-gkz6s:p-48tst",
            "kubectl.kubernetes.io/last-applied-configuration": "{\"apit
            "lifecycle.cattle.io/create.namespace-auth": "true"
        },
        "creationTimestamp": "2022-03-01T17:17:20Z",
        "deletionTimestamp": "2022-03-18T12:30:14Z",
        "labels": {
            "cattle.io/creator": "norman",
            "field.cattle.io/projectId": "p-48tst"
        },
        "name": "longhorn-system",
        "resourceVersion": "15206257",
        "selfLink": "/api/v1/namespaces/longhorn-system",
        "uid": "9673789f-14fb-11e9-ba68-005056b171b1"
    },
    "spec": {
        "finalizers": [
            "kubernetes"
        ]
    },
    "status": {
        "phase": "Terminating"
    }
}
```

Chapter 16: Setting up a CI/CD pipeline and image registry



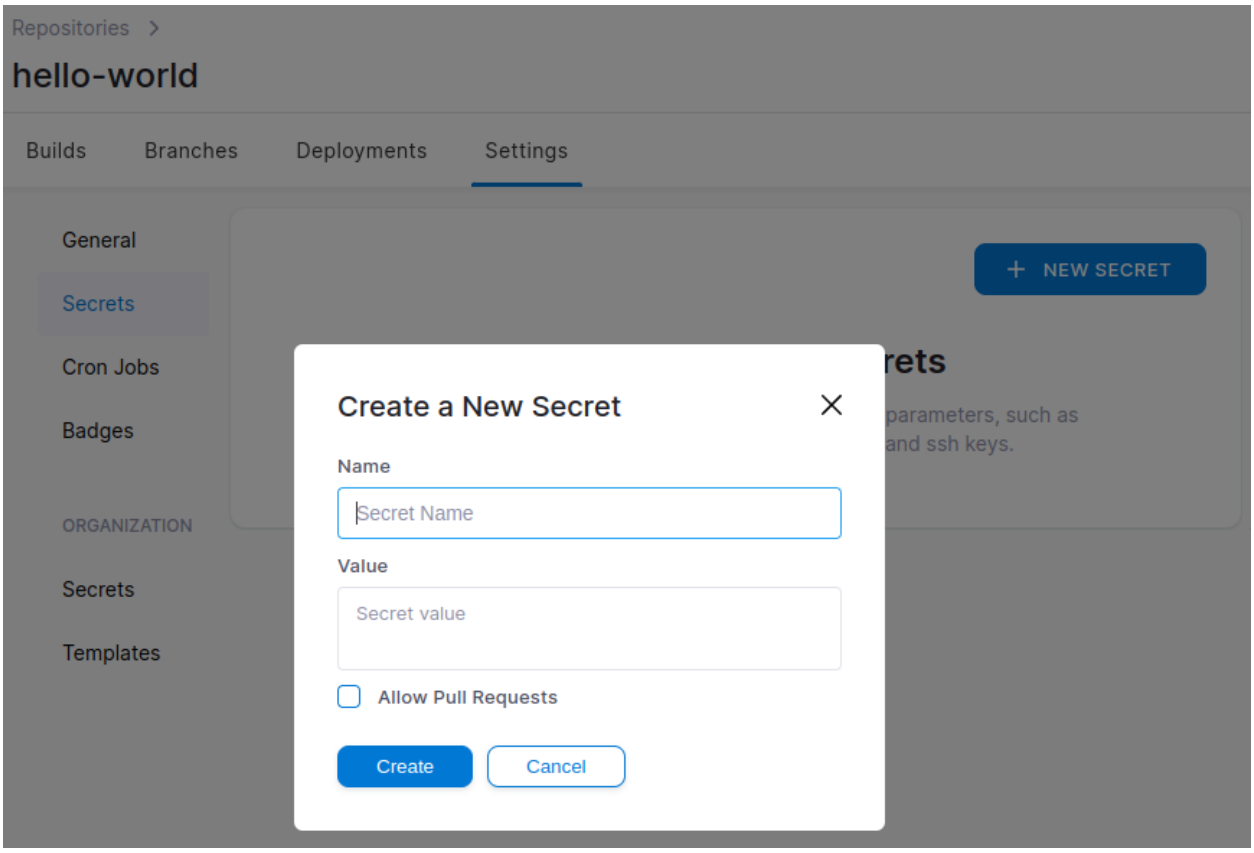

```
## Create drone namespace
kubectl create ns drone --dry-run=client | kubectl apply -f

## Setup helm repo
helm repo add bitnami https://charts.bitnami.com/bitnami

## Install the chart
helm install drone-db bitnami/postgresql -n drone \
--set global.storageClass=longhorn \
--set global.postgresql.auth.postgresPassword=drone \
--set global.postgresql.auth.username=drone \
--set global.postgresql.auth.password=drone \
--global.postgresql.auth.database=drone
```

```
## Setup helm repo
helm repo add drone https://charts.drone.io
helm repo update

## Upload ssl cert
kubectl -n drone create secret tls ssl-cert --key="tls.key" --cert="tls.crt"

## Create RPC Secret
openssl rand -hex 16
bea26a2221fd8090ea38720fc445eca6

## Install the chart
helm install drone-server drone/drone -n drone \
--set ingress.enabled=true \
--set ingress.hosts[0].host=drone.example.com \
--ingress.tls.[0].hosts=drone.example.com \
--set ingress.tls.[0].secretName=ssl-cert \
--set persistentVolume.enabled=true \
--set persistentVolume.storageClass=longhorn \
--set env.DRONE_SERVER_HOST=drone.example.com \
--set env.DRONE_SERVER_PROTO=https \
--set env.DRONE_DATABASE_DRIVER=postgres \
--set env.DRONE_DATABASE_DATASOURCE="postgres://drone:drone@drone-db:5432/drone?sslmode=disable" \
--set env.DRONE_RPC_SECRET=bea26a2221fd8090ea38720fc445eca6 \
--set env.DRONE_GIT_ALWAYS_AUTH=true \
--set env.DRONE_GITHUB_CLIENT_ID=your-id \
--set env.DRONE_GITHUB_CLIENT_SECRET=github-secret \
--set env.DRONE_USER_CREATE=username:Your-GitHub-Username,admin:true
--set env.DRONE_USER_FILTER=Your-GitHub-ORG
```

```
## Create drone namespace
kubectl create ns drone-runner --dry-run=client | kubectl apply -f

## Setup helm repo
helm repo add drone https://charts.drone.io
helm repo update

## Install the chart
helm install kube-runner drone/drone -n drone-runner \
--set env.DRONE_SERVER_HOST=drone.example.com \
--set env.DRONE_SERVER_PROTO=https \
--set env.DRONE_RPC_SECRET=bea26a2221fd8090ea38720fc445eca6
```

```
## Create a ServiceAccount and assign it ClusterAdmin permissions
kubectl -n kube-system create serviceaccount drone
kubectl create clusterrolebinding --clusterrole=cluster-admin --serviceaccount=kube-system:drone
TOKENNAME=`kubectl -n kube-system get serviceaccount/drone -o jsonpath='{.secrets[0].name}'`
TOKEN=`kubectl -n kube-system get secret $TOKENNAME -o jsonpath='{.data.token}'| base64 --decode`
echo $TOKEN
```

```yaml
apiVersion: v1
kind: Config
clusters:
- name: "example-cluster"
  cluster:
    server: "https://rancher.example.com/k8s/clusters/c-m-abcdefgj"

users:
- name: "example-cluster"
  user:
    token: "kubeconfig-user-abcdefgj:1234678912346789123467891234678912346789"


contexts:
- name: "example-cluster"
  context:
    user: "example-cluster"
    cluster: "example-cluster"

current-context: "example-cluster"
```



```
docker-username: DockerHub-Username
docker-password: DockerHub-PAT
kubernetes_server: https://rancher.example.com/k8s/clusters/c-m-abcdefgj
kubernetes_token: abcdefghiklmnopqrstuvwxyz1234567890.......
```

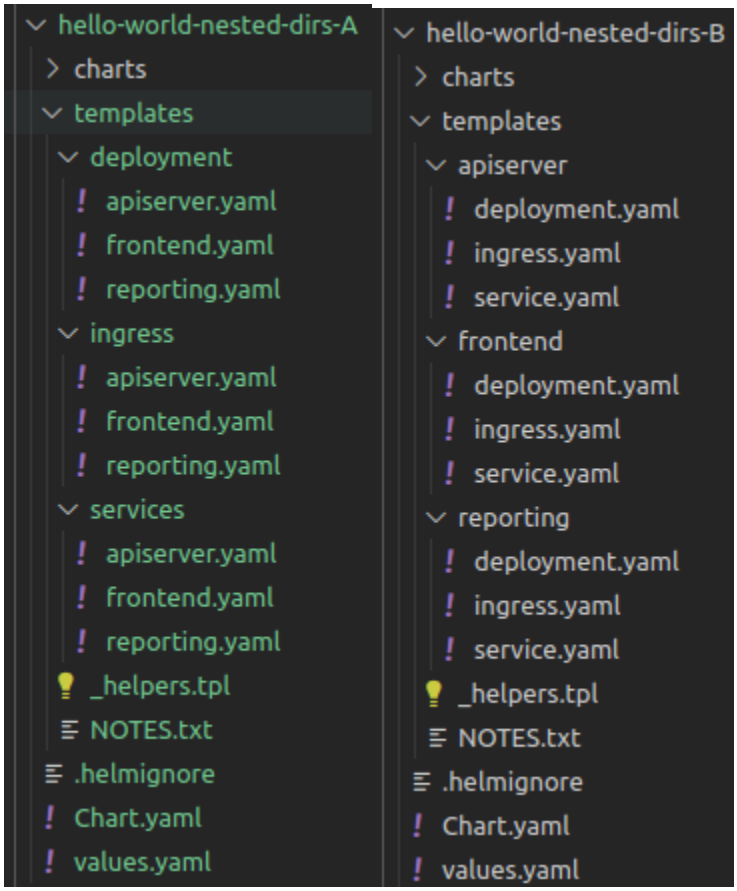
```
kind: pipeline
type: kubernetes
name: Example-App

steps:
  - name: Docker-Build
    image: plugins/docker
    settings:
      tags:
        - ${DRONE_BUILD_NUMBER}
        - ${DRONE_COMMIT_BRANCH}
        - latest
      username:
        from_secret: docker-username
      password:
        from_secret: docker-password

  - name: Deploy-to-k8s
    image: supporttools/kube-builder:latest
    settings:
      kubernetes_server:
        from_secret: kubernetes_server
      kubernetes_token:
        from_secret: kubernetes_token
    commands:
      - bash /usr/local/bin/init-kubectl
      - kubectl create ns hello-world --dry-run=client | kubectl apply -f
      - kubectl apply -n hello-world -f deploy.yaml
```

```
## Setup helm repo
helm repo add harbor https://helm.goharbor.io
helm repo update

## Create a namespace
kubectl create ns harbor --dry-run=client | kubectl apply -f

## Upload ssl cert
kubectl -n harbor create secret tls ssl-cert --key="tls.key" --cert="tls.crt"

## Install the chart
helm upgrade --install harbor harbor/harbor -n harbor \
--set expose.type=ingress \
--set expose.tls.secret.secretName=ssl-cert \
--set expose.tls.secret.notarySecretName=ssl-cert \
--set expose.ingress.hosts.core=harbor.example.com \
--set expose.ingress.hosts.notary=harbor-notary.example.com \
--set persistence.enabled=true \
--set persistence.persistentVolumeClaim.registry.storageClass=longhorn \
--set persistence.persistentVolumeClaim.chartmuseum.storageClass=longhorn \
--set persistence.persistentVolumeClaim.jobservice.storageClass=longhorn \
--set persistence.persistentVolumeClaim.database.storageClass=longhorn \
--set persistence.persistentVolumeClaim.redis.storageClass=longhorn \
--set persistence.persistentVolumeClaim.trivy.storageClass=longhorn
```

Chapter 17: Creating and using Helm charts

```
hello-world/
  charts/             # A directory containing any charts upon which this chart depends.
  Chart.yaml          # A YAML file containing information about the chart
  values.yaml         # The default configuration values for this chart
  templates/          # A directory of templates that, when combined with values,
                      # will generate valid Kubernetes manifest files.
```

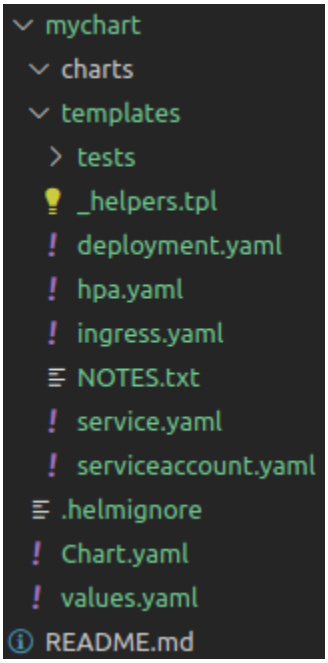
```yaml
apiVersion: v2
name: hello-world
description: hello-world
type: application
version: 0.1.0
appVersion: 1.0.0
keywords:
  - hello-world
sources:
  - https://github.com/SupportTools/hello-world
maintainers:
  - name: Matthew Mattox
    email: mmattox@support.tools
    url: https://github.com/mattmattox/
icon: https://raw.githubusercontent.com/SupportTools/hello-world/main/img/logo.svg
```


hello-world-nested-dirs-A
charts
templates
deployment
apiserver.yaml
frontend.yaml
reporting.yaml
ingress
apiserver.yaml
frontend.yaml
reporting.yaml
services
apiserver.yaml
frontend.yaml
reporting.yaml
_helpers.tpl
NOTES.txt
.helmignore
Chart.yaml
values.yaml
hello-world-nested-dirs-B
charts
templates
apiserver
deployment.yaml
ingress.yaml
service.yaml
frontend
deployment.yaml
ingress.yaml
service.yaml
reporting
deployment.yaml
ingress.yaml
service.yaml
_helpers.tpl
NOTES.txt
.helmignore
Chart.yaml
values.yaml

```yaml
frontend:
  image: supporttools/hello-world
  tag: v0.1.2

  ingress:
    enabled: true
    hosts:
      - host: hello-world.example.local
        paths:
          - path: /
            pathType: ImplementationSpecific

apiserver:
  image: supporttools/hello-world-api
  tag: v0.1.1-rc1

  ingress:
    enabled: false
    hosts:
      - host: chart-example.local
        paths:
          - path: /
            pathType: ImplementationSpecific
```

```
{{- if .Values.ingress.enabled }}
{{- if or (.Capabilities.APIVersions.Has "networking.k8s.io/v1/Ingress") (not (.Capabilities.APIVersions.Has "networking.k8s.io/v1beta1/Ingress")) }}
apiVersion: networking.k8s.io/v1
{{- else }}
apiVersion: networking.k8s.io/v1beta1
{{- end }}
kind: Ingress
metadata:
```

```
"Error: UPGRADE FAILED: rendered manifests contain a resource that already exists.
Unable to continue with update: Deployment "sentry-relay" in namespace "infra-sentry" exists and cannot
be imported into the current release: invalid ownership metadata; label validation error: missing key
"app.kubernetes.io/managed-by": must be set to "Helm"; annotation validation error: missing key
"meta.helm.sh/release-name": must be set to "sentry"; annotation validation error: missing key
"meta.helm.sh/release-namespace": must be set to "infra-sentry"
```




```
echo "Deploying Portal"
helm upgrade --install portal ./chart \
--namespace ${namespace} \
-f ./chart/values.yaml \
--set image.tag=${DRONE_BUILD_NUMBER} \
--set ingress.host=${ingress}
```

```yaml
- name: Deploy-to-Dev
  image: supporttools/kube-builder:latest
  environment:
    DOCKER_USERNAME:
      from_secret: harbor-username
    DOCKER_PASSWORD:
      from_secret: harbor-password
  settings:
    kubernetes_server:
      from_secret: k8s_dev_server
    kubernetes_token:
      from_secret: k8s_dev_token
  commands:
    - bash deploy.sh dev ${TAG}
```

Chapter 18: Resource management

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: hello-world
spec:
  containers:
  - name: webserver
    image: supporttools/hello-world
    resources:
      requests:
        memory: "64Mi"
        cpu: "250m"
      limits:
        memory: "128Mi"
        cpu: "500m"
```

```yaml
apiVersion: v1
kind: ResourceQuota
metadata:
  name: compute-resources
spec:
  hard:
    requests.cpu: "1"
    requests.memory: 1Gi
    limits.cpu: "2"
    limits.memory: 2Gi
    requests.nvidia.com/gpu: 4
    configmaps: "10"
    persistentvolumeclaims: "4"
    pods: "4"
    replicationcontrollers: "20"
    secrets: "10"
    services: "10"
    services.loadbalancers: "2"
```

```
kubectl create namespace kubecost
helm repo add kubecost https://kubecost.github.io/cost-analyzer/
helm install kubecost kubecost/cost-analyzer --namespace kubecost --set kubecostToken="abc123......"
```