Kubernetes Application Management Based on Helm and Custom Controller

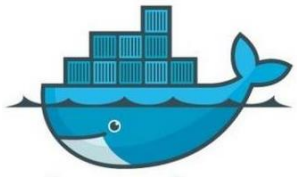
Peng Jiang/ Rancher Labs Inc.

LF ASIA, LLC

Agenda

- Helm介绍
- Rancher基于Helm的Catalog实现

应用容器化改造及部署



应用容器化 - 完成!

K8S集群部署 - 完成!

状态	名称	镜像	伸缩
命名空间：sockshop			
Active	carts	weaveworksdemos/carts:0.4.8 2个Pods / 创建于2 days ago	2
Active	carts-db	mongo 2个Pods / 创建于2 days ago	2
Active	catalogue	weaveworksdemos/catalogue:0.3.5 2个Pods / 创建于2 days ago	2
Active	front-end 80/http, 443/https, 30816/tcp	aaronji/front-end:7 2个Pods / 创建于2 days ago	2
Active	orders	weaveworksdemos/orders:0.4.7 2个Pods / 创建于2 days ago	2
Active	orders-db	mongo 2个Pods / 创建于2 days ago	2
Active	payment	weaveworksdemos/payment:0.4.3 2个Pods / 创建于2 days ago	2
Active	queue-master	weaveworksdemos/queue-master:0.3.1 2个Pods / 创建于2 days ago	2
Active	queue-master	weaveworksdemos/queue-master:0.3.1 2个Pods / 创建于2 days ago	2
Active	rabbitmq	rabbitmq:3.6.8 2个Pods / 创建于2 days ago	2
Active	session-db	redis:alpine 2个Pods / 创建于2 days ago	2
Active	shipping	weaveworksdemos/shipping:0.4.8 2个Pods / 创建于2 days ago	2
Active	user	weaveworksdemos/user:0.4.4 2个Pods / 创建于2 days ago	2
Active	user-db	weaveworksdemos/user-db:0.3.0 2个Pods / 创建于2 days ago	2

如何管理容器化的应用系统?

Deployment

ServiceAccount

RoleBinding **Role**

PersistentVolumeClaim

Secrets **Configmap**

Service **Ingress**

Statefulset

?

解决方案?

- Kubernetes下的包管理工具（类似于apt/pip/brew）
- 将应用部署包含的多种Kubernetes资源对象整合成单一对象（Chart）
- 模板提供默认部署配置，部署时可以进行变量替换修改
- 生命周期管理能力，升级、回滚、版本追踪
- Helm repo预置提供易于部署的多种应用

Helm的发展历史

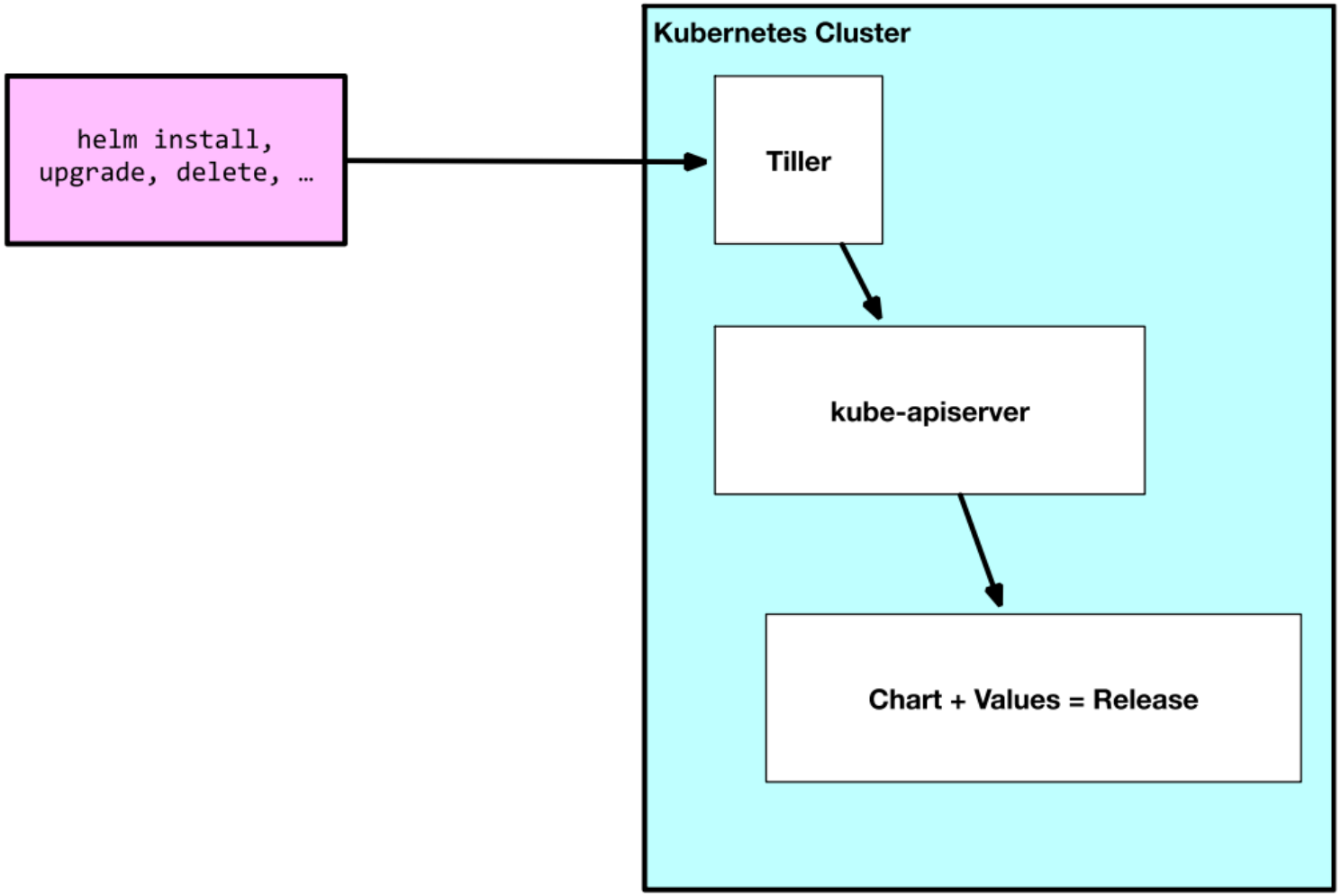
- 2015/10/15日，Deis公司内部的Hackathon项目
- 最早期的版本使用Python Jinja作为模板语言
- 最初的组件名称：k8splace, kpm, k3(kay cube)
- 用于Deis平台内部的工作流安装部署
- 在2015年11月旧金山的Kubecon正式公布
- 前期属于Kubernetes项目之下的子项目
- 2018年6月1日正式成为托管在CNCF之下的独立项目

Helm架构

核心概念：Helm、Tiller、Charts

- Helm的核心组件
  - Helm Client
  - Tiller Server
- Helm Client是用户使用的CLI工具
  - Go语言开发，通过gRPC和Tiller Server交互
  - 将Charts和Values发送给Tiller Server进行部署、升级等操作
- Tiller Server
  - 运行在k8s集群之上
  - 与Helm CLI和K8s API Server进行交互
  - 管理应用包(chart)的部署实例(release)

Helm架构(续)


helm install,
upgrade, delete, …
Kubernetes Cluster
Tiller
kube-apiserver
Chart + Values = Release

如何安装使用?

- 安装部署
  - Helm.sh
- 使用
  - $helm init #安装Tiller
  - $helm install <chart> #部署应用
  - $helm upgrade <release>
  - $helm delete <release>
  - …

Chart

- 应用的定义描述
- 包含
  - Metadata
  - K8S 资源定义
  - 文档说明
- 有规范的命名及结构，包括文件和目录名
- 存在chart仓库中，可以保存为tar包进行分发

Chart目录结构


```
wordpress/
  Chart.yaml          # A YAML file containing information about the chart
  LICENSE             # OPTIONAL: A plain text file containing the license for the chart
  README.md           # OPTIONAL: A human-readable README file
  values.yaml         # The default configuration values for this chart
  charts/             # OPTIONAL: A directory containing any charts upon which this chart depends.
  templates/          # OPTIONAL: A directory of templates that, when combined with values,
                      # will generate valid Kubernetes manifest files.
  templates/NOTES.txt # OPTIONAL: A plain text file containing short usage notes
```

- $helm create demoapp

```
demoapp/
├── Chart.yaml
├── charts
├── templates
└── values.yaml
```

Chart.yml - 元数据


```
name: The name of the chart (required)
version: A SemVer 2 version (required)
description: A single-sentence description of this project (optional)
keywords:
  - A list of keywords about this project (optional)
home: The URL of this project's home page (optional)
sources:
  - A list of URLs to source code for this project (optional)
maintainers: # (optional)
  - name: The maintainer's name (required for each maintainer)
    email: The maintainer's email (optional for each maintainer)
engine: gotpl # The name of the template engine (optional, defaults to gotpl)
icon: A URL to an SVG or PNG image to be used as an icon (optional).
```

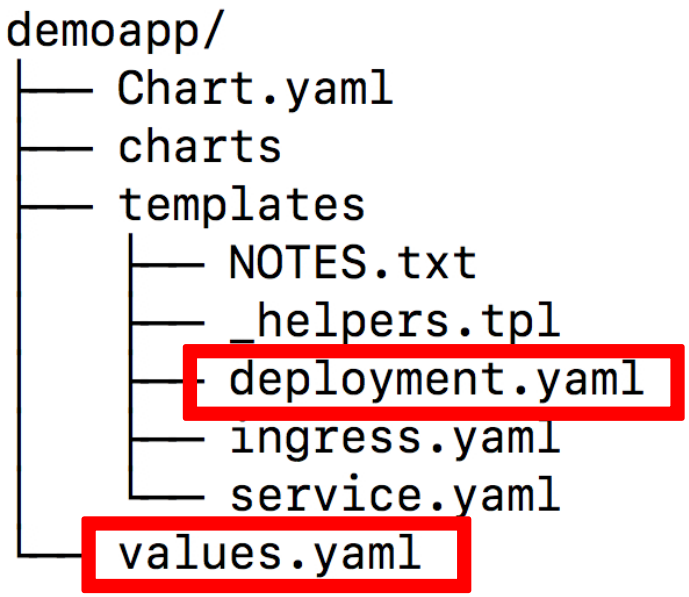
Templates – 模板


```
demoapp/
├── Chart.yaml
├── charts
├── templates
│   ├── NOTES.txt
│   ├── _helpers.tpl
│   ├── deployment.yaml
│   ├── ingress.yaml
│   └── service.yaml
└── values.yaml
```

+

Go template

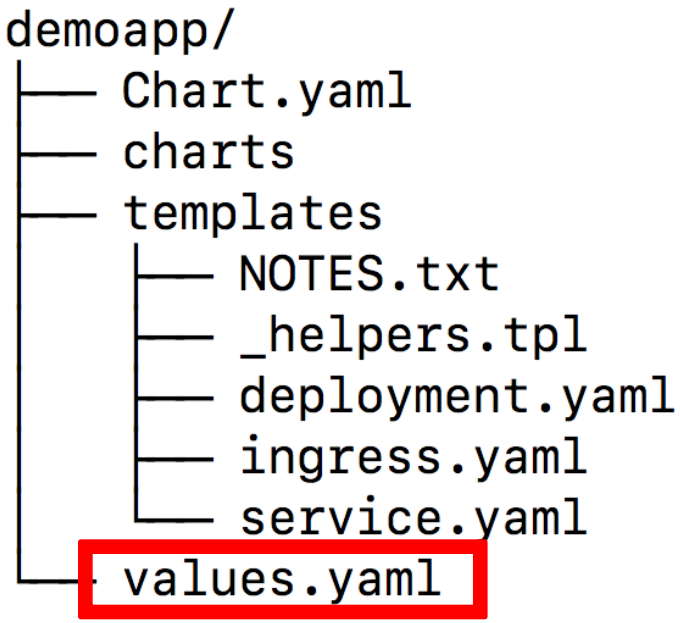
Values.yml – 配置选项




```
##### values.yaml
replicaCount: 1
image:
  repository: nginx
  tag: stable
  pullPolicy: IfNotPresent
...

##### deployment.yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  #...
spec:
  replicas: {{ .Values.replicaCount }}
  template:
    #...
    spec:
      containers:
        - name: {{ .Chart.Name }}
          image: "{{ .Values.image.repository }}:{{ .Values.image.tag }}"
          imagePullPolicy: {{ .Values.image.pullPolicy }}
...
```

Values.yml – 配置选项

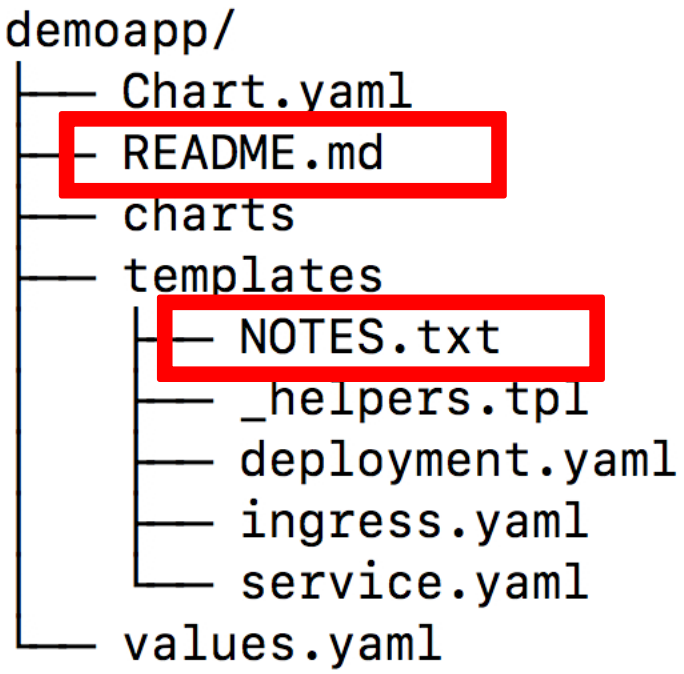



```
##### values.yaml
replicaCount: 1
image:
  repository: nginx
  tag: stable
  pullPolicy: IfNotPresent
...
```

$ helm install --set image.tag=latest ./demoapp

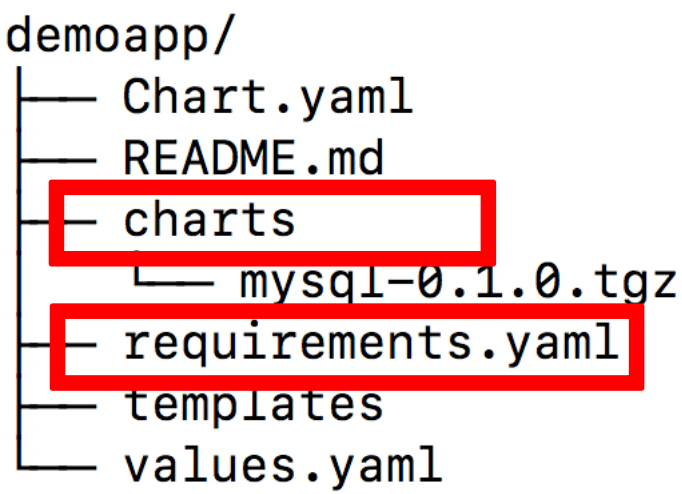
$ helm install -f stagingvalues.yaml ./demoapp

文档说明



- README.md一般用于Chart提供的应用及服务介绍，部署前提条件说明，values.yaml中的变量说明等。
- NOTES.txt会在部署后显示，可以用于向chart的使用者提示使用说明，下一步操作等信息。

应用依赖



- 在requirements.yaml中声明依赖
- Helm dependency update或者手工创建、拷贝依赖chart到charts子目录

```
##### requirements.yaml
dependencies:
- name: mysql
  version: 0.1.0
  repository: https://kubernetes-charts-incubator.storage.googleapis.com/
```

Chart仓库

- HTTP 服务器
- 提供 index.yaml索引文件和打包的charts
  - https://kubernetes-charts.storage.googleapis.com/

$ helm serve

例如 https://example.com/charts 的仓库目录结构

```
charts/
├── index.yaml
├── demoapp-0.1.0.tgz.prov
└── demoapp-0.1.0.tgz
...
```

Release管理

- 在 Kubernetes 集群上运行的 Chart 的实例
  - Release.Name
  - Release.Namespace
  - Release.IsInstall
  - Release.IsUpgrade

```
NAME    REVISION        UPDATED                         STATUS          CHART
db-test 1               Sun Jun 24 09:24:55 2018        DEPLOYED        mysql-0.8.1
```

- 对应Kubernetes资源对象的命名 releasename-chartname

```
NAME            DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
db-test-mysql   1         1         1            1           2m
```

Release管理

- Release的版本管理

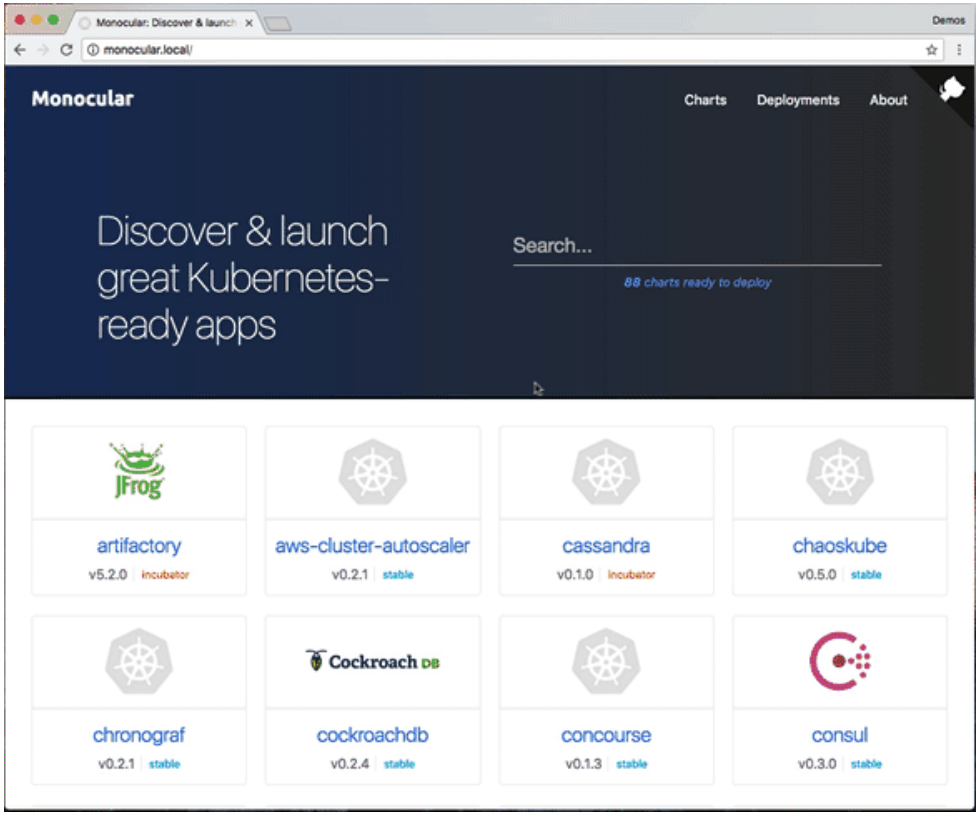
```
REVISION        UPDATED                         STATUS          CHART           DESCRIPTION
1               Sun Jun 24 09:24:55 2018        SUPERSEDED      mysql-0.8.1     Install complete
2               Sun Jun 24 09:31:49 2018        SUPERSEDED      mysql-0.8.1     Upgrade complete
3               Sun Jun 24 09:32:36 2018        SUPERSEDED      mysql-0.8.1     Upgrade complete
4               Sun Jun 24 09:34:27 2018        DEPLOYED        mysql-0.8.1     Rollback to 2
```

```
root@ip-172-31-20-51:~# kubectl get configmap -n kube-system
NAME                                 DATA      AGE
canal-config                         4         4d
cluster-state                        1         4d
db-test.v1                           1         10m
db-test.v2                           1         3m
db-test.v3                           1         2m
db-test.v4                           1         40s
```

- Helm delete --purge删除release对象

Monocular - Helm Web UI

- https://github.com/kubernetes-helm/monocular



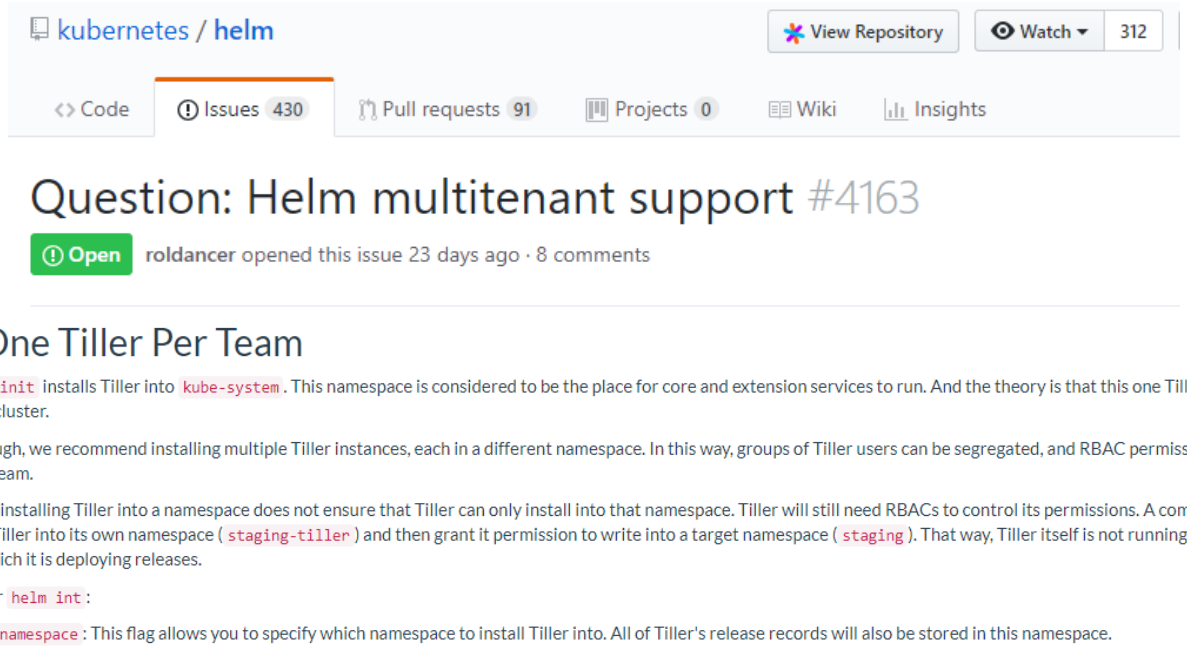
https://github.com/kubeapps/kubeapps

Helm的用途总结

- 利用已有的Chart快速部署进行实验
- 创建自定义Chart，方便地在团队间共享
- 便于管理应用的生命周期
- 便于应用的依赖管理和重用
- 将K8S集群作为应用发布协作中心

Helm的限制

- 多租户环境下不便于实现权限控制和隔离：
  - 无内建认证授权机制，应用以Tiller运行使用的权限部署
  - Release对象集中存储在Tiller运行的namespace之下



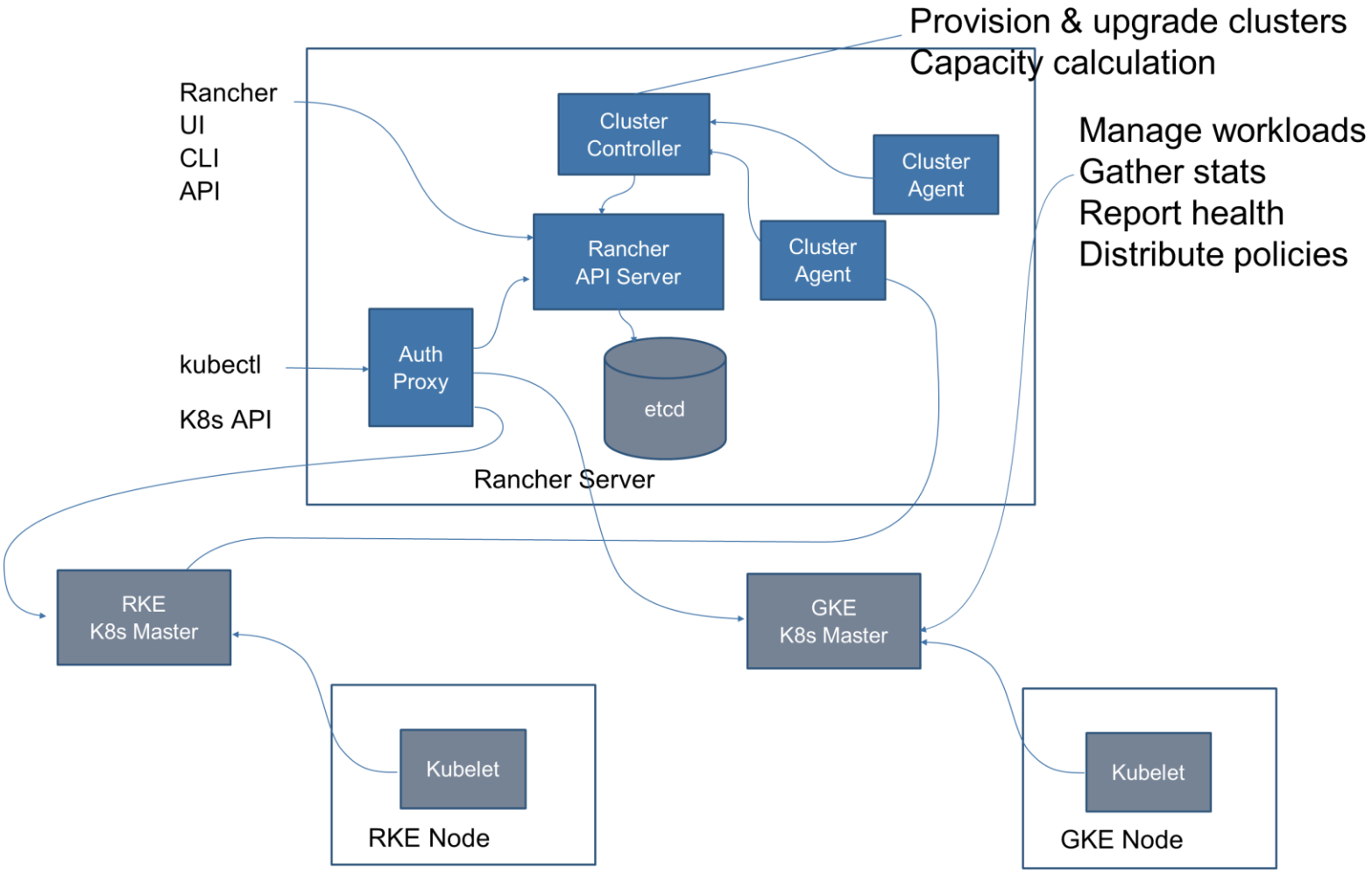
Rancher 2.0

Workload Management
User Interface • App Catalog • CI/CD • Monitoring • Logging

Unified Cluster Management
Provisioning • Authentication • RBAC • Policy • Security • Capacity • Cost

Rancher Kubernetes Engine (RKE)
vSphere • Bare metal

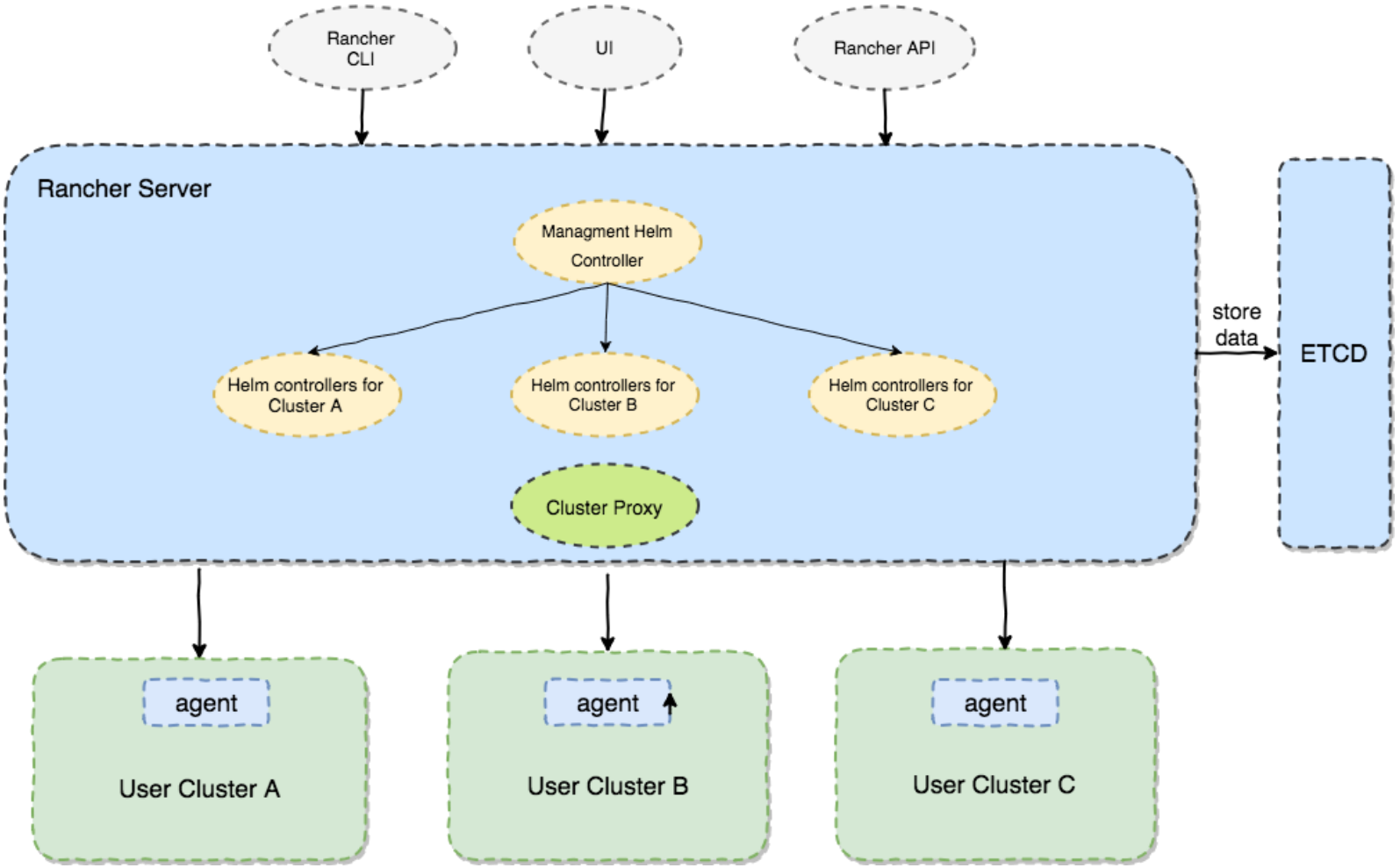
Architecture



Rancher 2.0 Catalog

- 基于标准的Helm规范进行增强
- 更好的用户使用体验
- 支持git和http/s server
- Rancher负责追踪和管理应用创建的资源对象

Catalog Architecture



Rancher 2.0 Charts目录结构


```
charts/wordpress/<app version>/
  app-readme.md            # Rancher Specific: Readme file for display in Rancher 2.0 UI
  charts/                  # Directory containing dependency charts
  Chart.yaml               # Required Helm chart information file
  questions.yml            # Rancher Specific: File containing questions for Rancher 2.0 UI
  README.md                # Optional: Helm Readme file (will be rendered in Rancher 2.0 UI as well)
  requirements.yaml        # Optional YAML file listing dependencies for the chart
  templates/               # A directory of templates that, when combined with values.yml will generate K8
  values.yaml              # The default configuration values for this chart
```

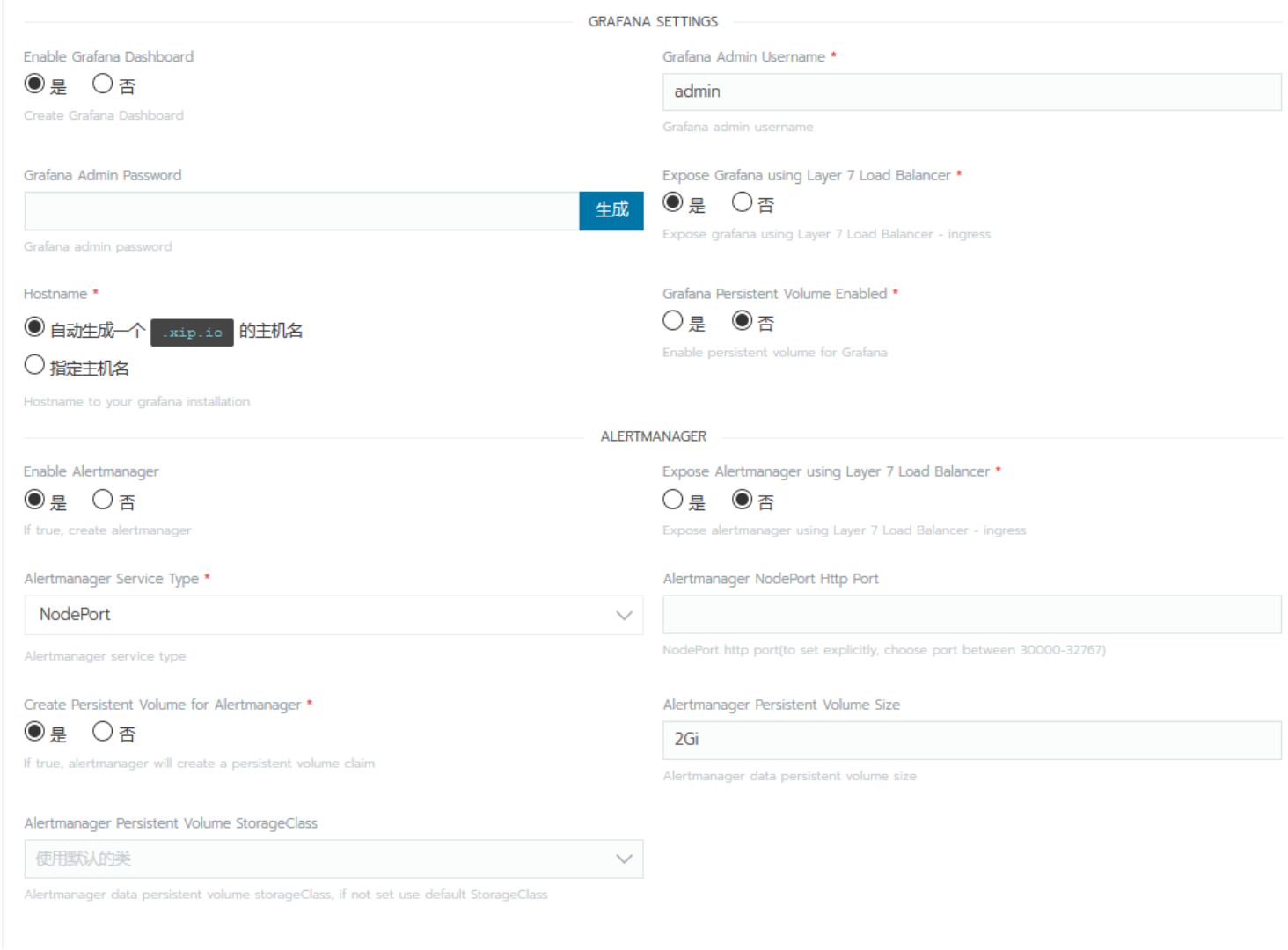
Repo: https://github.com/rancher/charts

Question.yaml


```yaml
- variable: server.ingress.enabled
  default: true
  description: "Expose prometheus using Layer 7 Load Balancer - ingress"
  type: boolean
  group: "Prometheus Server"
  label: Expose Prometheus using Layer 7 Load Balancer
  show_subquestion_if: true
  required: true
  subquestions:
  - variable: server.ingress.hosts[0]
    default: "xip.io"
    description: "Prometheus server ingress hostname"
    type: hostname
    required: true
    label: Hostname
- variable: server.service.type
  default: "NodePort"
  description: "Server service type"
  group: "Prometheus Server"
  type: enum
  show_if: "server.ingress.enabled=false"
  options:
    - "ClusterIP"
    - "NodePort"
  required: true
  label: Prometheus Service Type
  show_subquestion_if: "NodePort"
  subquestions:
  - variable: server.service.nodePort
    default: ""
    description: "NodePort http port(to set explicitly, choose port between 30000-32767)"
    type: int
    min: 30000
    max: 32767
    label: Prometheus NodePort Http Port
    show_if: "server.ingress.enabled=false&&server.service.type=NodePort"
```

- Go模板语言
- 变量的高级参数定义
  - Group
  - Description
  - Label
  - Type
  - Min/Max
  - Condition
  - ……

Question.yaml



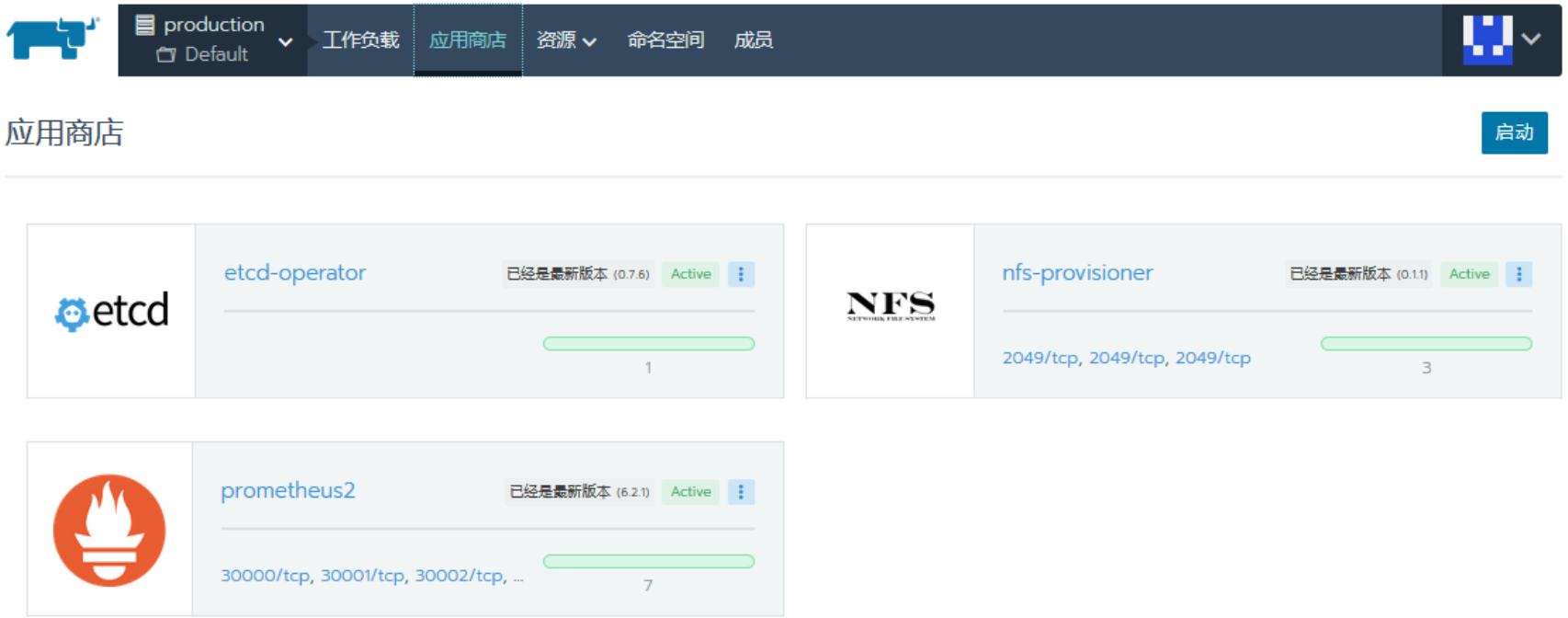
CRD - App


```
root@0e879f5e7550:/var/lib/rancher# kubectl get crd
NAME                                                                AGE
apprevisions.project.cattle.io                                      5d
apps.project.cattle.io                                              5d
authconfigs.management.cattle.io                                    5d
catalogs.management.cattle.io                                       5d
clusteralerts.management.cattle.io                                  5d
clustercomposeconfigs.management.cattle.io                          5d
clusterevents.management.cattle.io                                  5d
clusterloggings.management.cattle.io                                5d
clusterpipelines.management.cattle.io                               5d
clusterregistrationtokens.management.cattle.io                      5d
clusterroletemplatebindings.management.cattle.io                    5d
clusters.management.cattle.io                                       5d
composeconfigs.management.cattle.io                                 2d
dynamicschemas.management.cattle.io                                 5d
globalcomposeconfigs.management.cattle.io                           5d
globalrolebindings.management.cattle.io                             5d
globalroles.management.cattle.io                                    5d
groupmembers.management.cattle.io                                   5d
groups.management.cattle.io                                         5d
listenconfigs.management.cattle.io                                  5d
namespacecomposeconfigs.project.cattle.io                           5d
nodedrivers.management.cattle.io                                    5d
nodepools.management.cattle.io                                      5d
nodes.management.cattle.io                                          5d
nodetemplates.management.cattle.io                                  5d
notifiers.management.cattle.io                                      5d
pipelineexecutionlogs.management.cattle.io                          5d
pipelineexecutions.management.cattle.io                             5d
pipelines.management.cattle.io                                      5d
podsecuritypolicytemplateprojectbindings.management.cattle.io       5d
podsecuritypolicytemplates.management.cattle.io                     5d
preferences.management.cattle.io                                    5d
projectalerts.management.cattle.io                                  5d
projectloggings.management.cattle.io                                5d
projectnetworkpolicies.management.cattle.io                         5d
projectroletemplatebindings.management.cattle.io                    5d
```

prometheus-k8s-cm.yaml

```yaml
- apiVersion: project.cattle.io/v3
  kind: App
  metadata:
    annotations:
      field.cattle.io/creatorId: user-tfkk4
      lifecycle.cattle.io/create.helm-controller_c-bmqbv: "true"
    clusterName: ""
    creationTimestamp: 2018-06-16T02:05:21Z
    finalizers:
    - clusterscoped.controller.cattle.io/helm-controller_c-bmqbv
    generation: 0
    initializers: null
    name: nfs-provisioner
    namespace: project-qqv88
    resourceVersion: "712370"
    selfLink: /apis/project.cattle.io/v3/namespaces/project-qqv88/apps/nfs-provisioner
    uid: b9841c68-7109-11e8-866e-0242ac110002
  spec:
    answers:
      defaultImage: "true"
      image.repository: quay.io/kubernetes_incubator/nfs-provisioner
      image.tag: v1.0.8
      nfs.hostPort: "2049"
      persistence.defaultClass: "true"
      persistence.hostPath: /srv
    appRevisionName: apprevision-7q4h2
    externalId: catalog://?catalog=dev&template=nfs-provisioner&version=0.1.1
    projectName: c-bmqbv:project-qqv88
    targetNamespace: nfs-provisioner
  status:
    conditions:
    - lastUpdateTime: 2018-06-16T02:05:22Z
      status: "True"
      type: Installed
    notes: |+
      ---
      # Source: nfs-provisioner/templates/NOTES.txt
      1. Get the storageClass URL by running these commands:
        kubeclt get storageClass
```

应用管理




```
externalId: catalog://?catalog=library&template=prometheus&version=6.2.1
projectName: c-mig6h:project-87ixq
```

应用商店


应用商店
添加应用商店
官方认证
禁用 启用
由Rancher官方维护的模板仓库。
Helm Stable
禁用 启用
由Kubernetes社区维护的模板仓库。
Helm Incubator
禁用 启用
未经测试验证的应用模板仓库。
自定义
您可以自定义其他应用商店地址，每个自定义应用商店需要有以下内容:
唯一的名称
仓库URL地址:
基于Git的商店URL地址, 例如: https://github.com/rancher/charts.git
Helm Charts server URL, 例如: https://kubernetes-charts.storage.googleapis.com/ (详细信息可查看Chart托管仓库)
可选 : 如果使用基于Git的应用商店，则还需要设置分支名称
删除
搜索
状态 名称 类型 URL 分支
Active demo Helm https://github.com/gitdemo-cn/catalog master
Active istio Helm https://github.com/fyery-chen/charts-1 istio

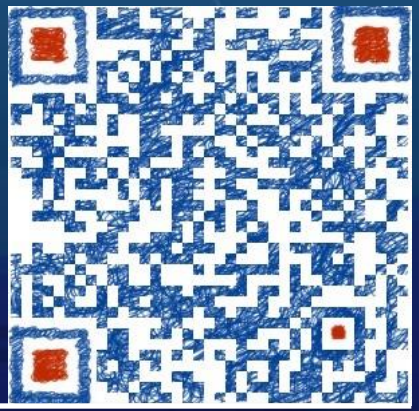
Try Rancher


```
docker run -d -p 80:80 -p 443:443 --restart always
-v /home/data:/var/lib/rancher
rancher/rancher:stable
```

LINUXCON
containercon
CLOUDOPEN
CHINA 中国
THINK OPEN
开放性思维

THANK YOU

Contact Me




LINUXCON
containercon
CLOUDOPEN
CHINA 中国
THINK OPEN
开放性思维
LF ASIA, LLC