

# Kubernetes集群管理的轻松之道（下）



# 大纲

- 如何部署应用
- 认证方式
- 审计日志

# 如何部署应用

- Kubernetes应用相关的几个基本概念
  - Pod
  - Replicaset
  - Service
  - Configmap
  - Ingresses
  - 使用Rancher管理应用

# Pod

- Kubenetes可以部署的最小单元
- 包含一个或多个可以容器
- 一个pod中的容器会调度到一起
- 每个pod有唯一的ip
- pod内的容器可以通过localhost沟通

# Resplicaset

- 确保Pod以你指定的副本数运行，即如果有容器异常退出，会自动创建新的 Pod 来替代；而异常多出来的容器也会自动回收。

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: frontend
  labels:
    app: guestbook
    tier: frontend
spec:
  # modify replicas according to your case
  replicas: 3
  selector:
    matchLabels:
      tier: frontend
    matchExpressions:
      - {key: tier, operator: In, values: [frontend]}
  template:
    metadata:
      labels:
        app: guestbook
        tier: frontend
    spec:
      containers:
        - name: php-redis
          image: gcr.io/google_samples/gb-frontend:v3
          resources:
            requests:
              cpu: 100m
              memory: 100Mi
          env:
            - name: GET_HOSTS_FROM
              value: dns
              # If your cluster config does not include a dns service, then to
              # instead access environment variables to find service host
              # info, comment out the 'value: dns' line above, and uncomment the
              # line below.
              # value: env
      ports:
        - containerPort: 80
```

# Services

- Service可以看作是一组提供相同服务的Pod对外的统一入口。
- 定义了可以访问一组pod的DNS入口
- 有不同类型：NodePort, ClusterIP, Loadbalancer

# Configmap

- 保存配置或配置文件
- 在pod中可以作为环境变量或本地文件访问
- 解耦了 pod spec 的状态定义和应用的配置

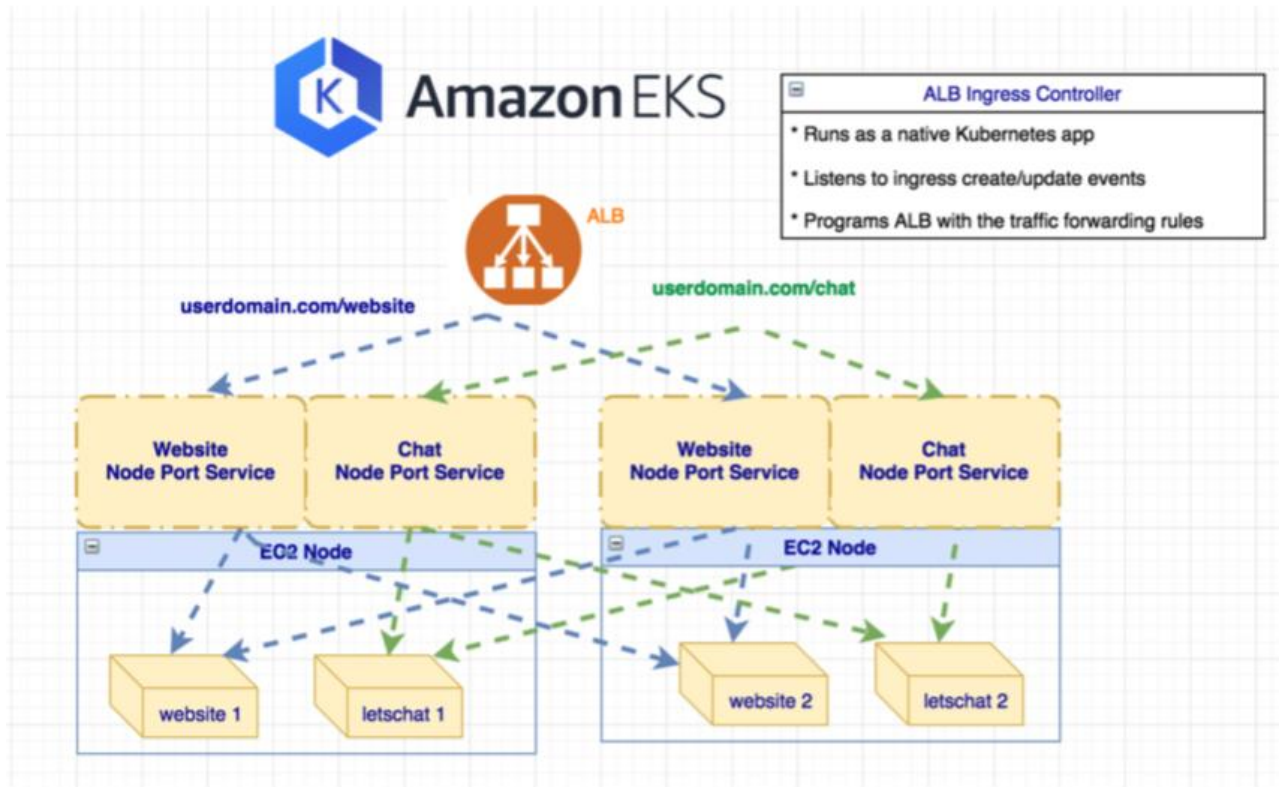
# Ingresses

- 定义Cluster外的流量怎么路由到Cluster中
- 用于暴露kubernetes service
- 可以通过host, path等属性进行路由
- 通常通过负载均衡实现, 例如Nginx, HAProxy, AWS ELB 等



# Ingress controllers for LBs deployed outside of Kubernetes cluster

## Ingress LB outside of Kubernetes cluster

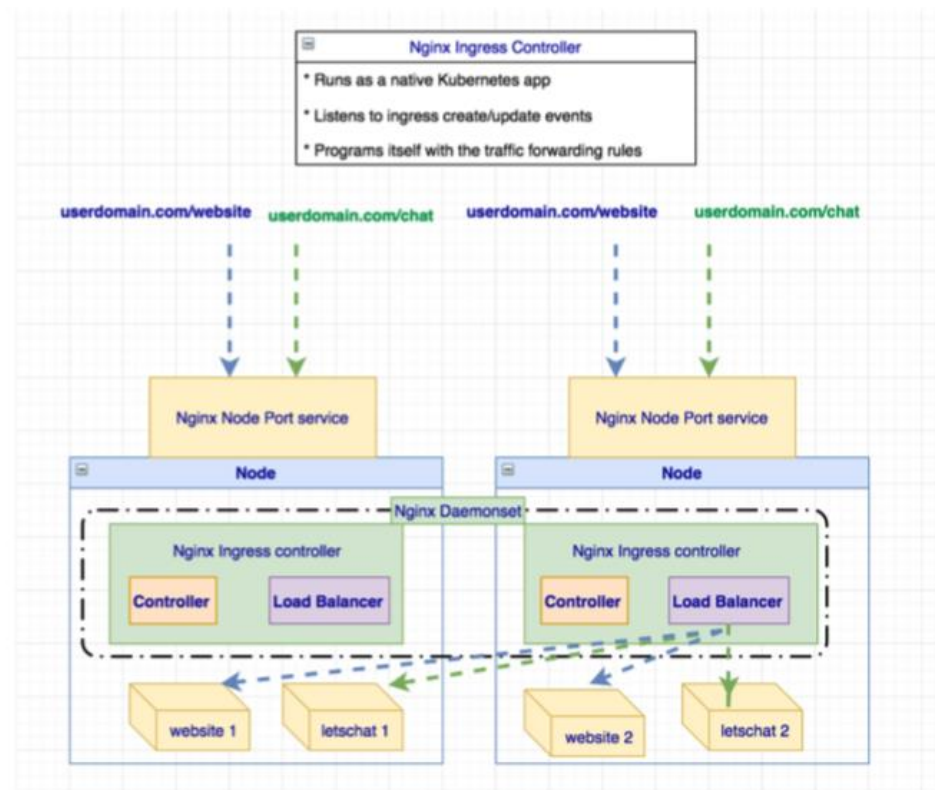


# Ingress controllers for LBs deployed outside of Kubernetes cluster

- Amazon ALB
  - <https://github.com/kubernetes-sigs/aws-alb-ingress-controller>
- Google GLBC
  - <https://github.com/kubernetes/ingress-gce>
- F5
  - <https://github.com/F5Networks/k8s-bigip-ctlr>
- Netscaler
  - <https://github.com/citrix/kube-ingress-citrix-netscaler>
- Openstack Octavia
  - <https://github.com/kubernetes/cloud-provider-openstack/blob/master/docs/using-octavia-ingress-controller.md>

# Ingress controllers for LBs deployed inside of Kubernetes cluster

## Ingress LB inside Kubernetes cluster



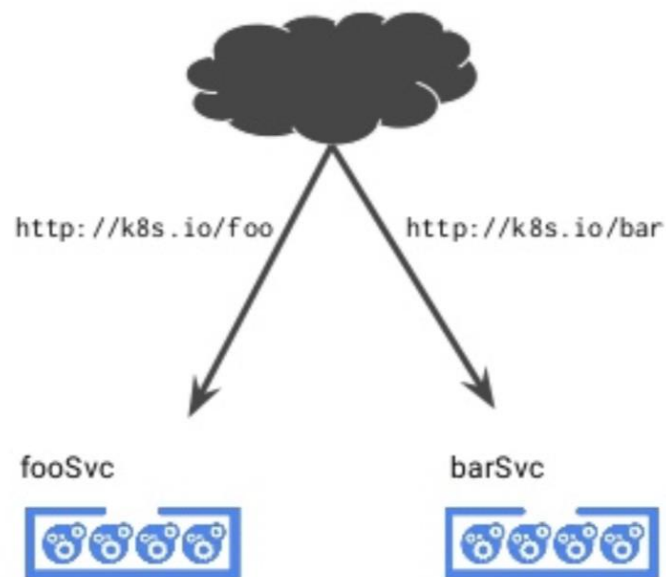
# Ingress controllers for LBs deployed inside Kubernetes cluster

- Nginx
  - <https://github.com/kubernetes/ingress-nginx>
- Haproxy
  - <https://github.com/jcmoraisjr/haproxy-ingress>
- Traefik
  - <https://github.com/containous/traefik/blob/master/docs/user-guide/kubernetes.md>
- Contour
  - <https://github.com/heptio/contour>

# Ingresses

## Ingress API

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: test
spec:
  rules:
  - host: k8s.io
    http:
      paths:
      - path: /foo
        backend:
          serviceName: fooSvc
          servicePort: 80
      - path: /bar
        backend:
          serviceName: barSvc
          servicePort: 80
```



# What Rancher do?

- 可以指定ingress对应的服务为workload或者service
- 支持在ingress中自动配置xip.io为hostname
- 支持的workload中根据开放的端口类型自动配置loadbalance
- 在workload中可以获取到配置的ingress，从而可以直接访问

Add Ingress

Name Add a Description  
e.g. website

Namespace Add to a new namespace  
cattle-system

Rules

☒ Automatically generate a `.xip.io` hostname  
☐ Specify a hostname to use  
☐ Use as the default backend

Target Backend + Service + Workload

Path Target Port  
e.g. /foo Choose a Workload... e.g. 80

+ Add Rule

Expand All

SSL/TLS Certificates  
Configure the certificates that will be presented for requests to encrypted ports.

Labels & Annotations  
Key/Value pairs that can be used to label/annotate containers and make scheduling decisions. None

Workloads Load Balancing Service Discovery Volumes Pipelines			
Redeploy ↺ Pause Orchestration    Download YAML ⬇ Delete 🗑			
<input type="checkbox"/>	State ⚙	Name ⚙	Image ⚙
Namespace: default			
<input type="checkbox"/>	▶ Active	chat 🌐 /index.html	crccheck/hello-world 1 Pod / Created an hour ago
<input type="checkbox"/>	▶ Active	test 🌐	crccheck/hello-world 2 Pods / Created 37 minutes ago
<input type="checkbox"/>	▶ Active	web 🌐 /name.html	sangeetha/testnewhostrouting 1 Pod / Created an hour ago

# 应用生命周期管理

- Rancher容器云平台提供简单易用的图形化界面，用户可以通过图形界面便捷的进行容器应用部署。应用部署完成后，可以进行容器的全生命周期管理和各种操作，包括启动、暂停、删除、克隆容器等。



# 应用部署方式

## 1. Deployment

- Deployment's pod is ephemeral
- Easy to scale up and down
- No desired context

## 2. StatefulSet

- More like pets than cattle
- Mostly using for persistent storage
- Roll out in orders and implement rolling update

## 3. DaemonSet

## 4. CronJob/Job




```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx:1.7.9
          ports:
            - containerPort: 80
```



# 应用状态管理

- 通过图形界面，用户可以便捷的查看容器的信息和运行状态，包括健康状态、容器名称、IP地址、主机、镜像和所在节点等信息。


[工作负载](#) [负载均衡](#) [服务发现](#) [数据卷](#) [流水线](#)


   

[导入YAML](#) [部署服务](#)


[暂停编排](#) 

[下载YAML](#) 


[删除](#)  1个工作负载





☐ [状态](#) 

☐ [名称](#) 

☐ [镜像](#) 

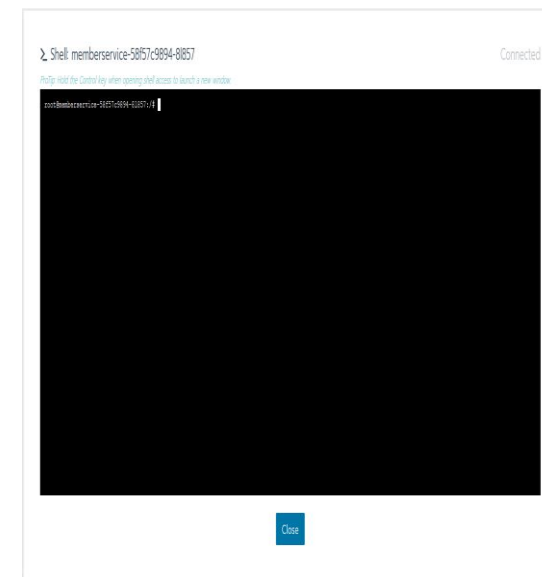
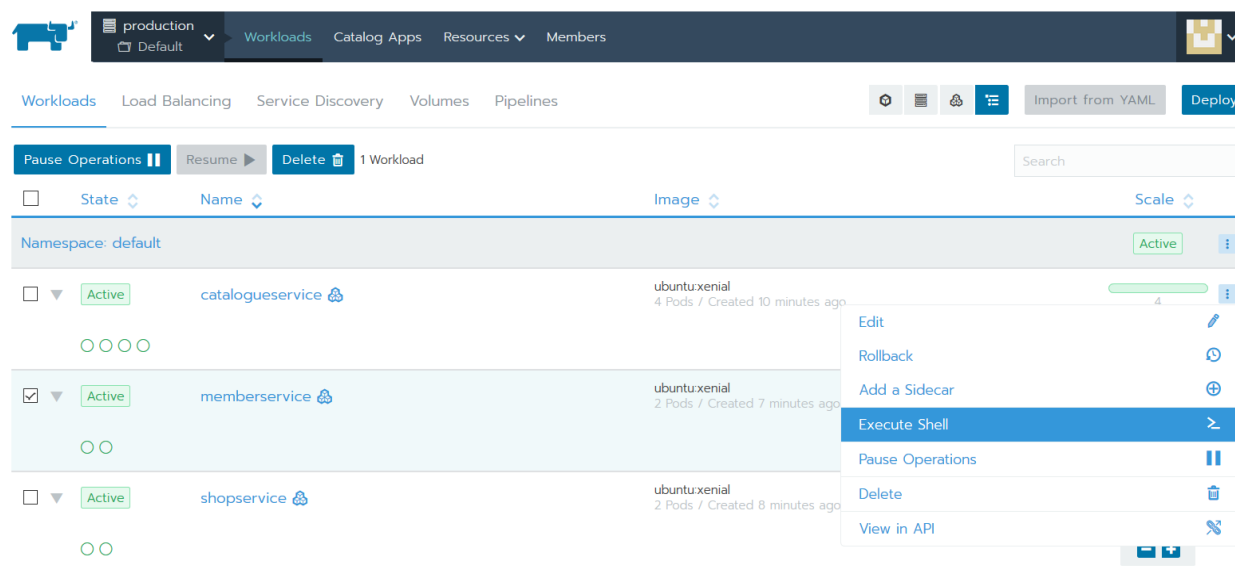
☐ [伸缩](#) 

命名空间: default 

<input checked="" type="checkbox"/> 	<div>Active</div> <div></div>	<div>podinfo </div> <div>31198/tcp, 80/http</div>	<div>stefanprodan/podinfo:0.0.1</div> <div>2个Pods / 创建于2 hours ago</div>	<div><div></div>2 </div> <div> </div>
<input type="checkbox"/> 	<div>Active</div> <div></div>	<div>test </div>	<div>nginx</div> <div>1个Pod / 创建于3 hours ago</div>	<div><div></div>1 </div> <div> </div>

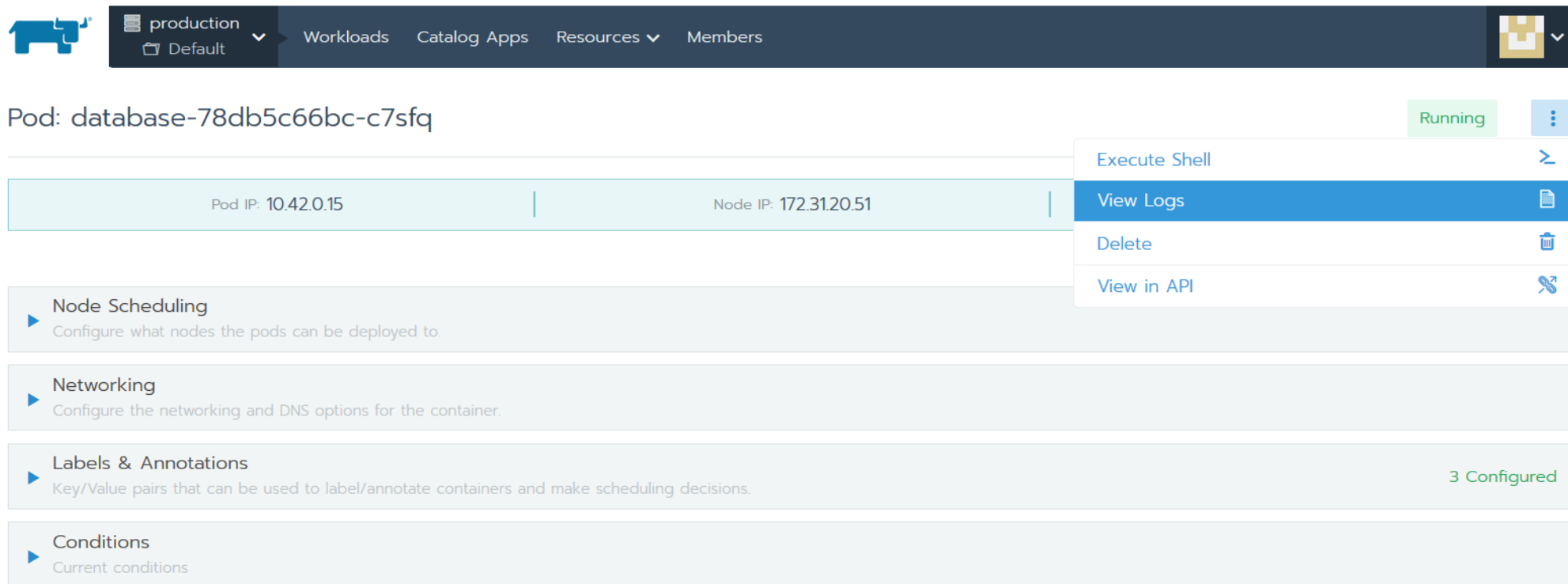
# 容器控制台

- Rancher容器云平台支持从图形界面直接访问容器Shell，从管理界面可以进入容器Shell支持命令，方便后台控制人员对容器进行相关操作。Rancher容器云平台支持从图形界面直接访问容器Shell，从管理界面可以进入容器Shell支持命令，方便后台控制人员对容器进行相关操作。



# 应用日志查看

- Rancher容器云平台支持从图形界面查看容器日志，日志支持实施刷新，方便管理和运维人员实时查看应用系统日志。



The screenshot displays the Rancher management interface. At the top, a dark navigation bar includes the Rancher logo, a dropdown menu set to 'production / Default', and links for 'Workloads', 'Catalog Apps', 'Resources', and 'Members'. On the right of this bar is a user profile icon. Below the navigation bar, the main content area shows a specific pod: 'Pod: database-78db5c66bc-c7sfq'. To the right of the pod name is a green 'Running' status label and a three-dot menu icon. A context menu is open, showing four options: 'Execute Shell' (with a terminal icon), 'View Logs' (highlighted in blue with a document icon), 'Delete' (with a trash icon), and 'View in API' (with a link icon). Below the pod information, there are four expandable configuration sections: 'Node Scheduling' (Configure what nodes the pods can be deployed to.), 'Networking' (Configure the networking and DNS options for the container.), 'Labels & Annotations' (Key/Value pairs that can be used to label/annotate containers and make scheduling decisions. - 3 Configured), and 'Conditions' (Current conditions).

# 容器调度管理

- Rancher容器云平台支持多种调度策略，具体如下。
- 指定Node节点调度，容器云平台有三种方式指定 Pod 只运行在指定的 Node 节点上，包括：
  1. NodeSelector 只调度到匹配指定 label 的 Node 上
  2. NodeAffinity 功能更丰富的 Node 选择器，比如支持集合操作
  3. PodAffinity 调度到满足条件的 Pod 所在的 Node 上

# 应用健康检查

- Rancher容器云平台支持对部署的容器Pod进行健康检查并跟进检查结果进行相应操作，以确保应用处于健康运行状态。

## 健康检查



周期性的向容器发出请求, 以查看它是否存在并正确响应。

### 准备检查



无



检查TCP连接是否正常



HTTP请求返回成功状态(2xx或3xx)



HTTPS请求返回成功状态(2xx或3xx)



在容器内运行的命令退出状态为0

# 应用升级策略

- Rancher容器云平台支持多种升级策略
  - 滚动升级：启动新的，然后停止旧的
  - 滚动升级：停止旧的pod，然后开始新的pod
  - 杀掉所有旧的pod，然后启动新的pod

缩放/升级策略

配置升级过程中替换Pod的策略。

☒ 滚动：启动新的pod，然后停止旧的pod

☐ 滚动：停止旧的pod，然后开始新的pod

☐ 杀死所有pod，然后重新开始

☐ 自定义

批量大小

1

Pod

Pod将被一次启动或停止如下数量

最短准备时间

0

秒

Pod内的容器需至少启动运行以下时长才被视为可用

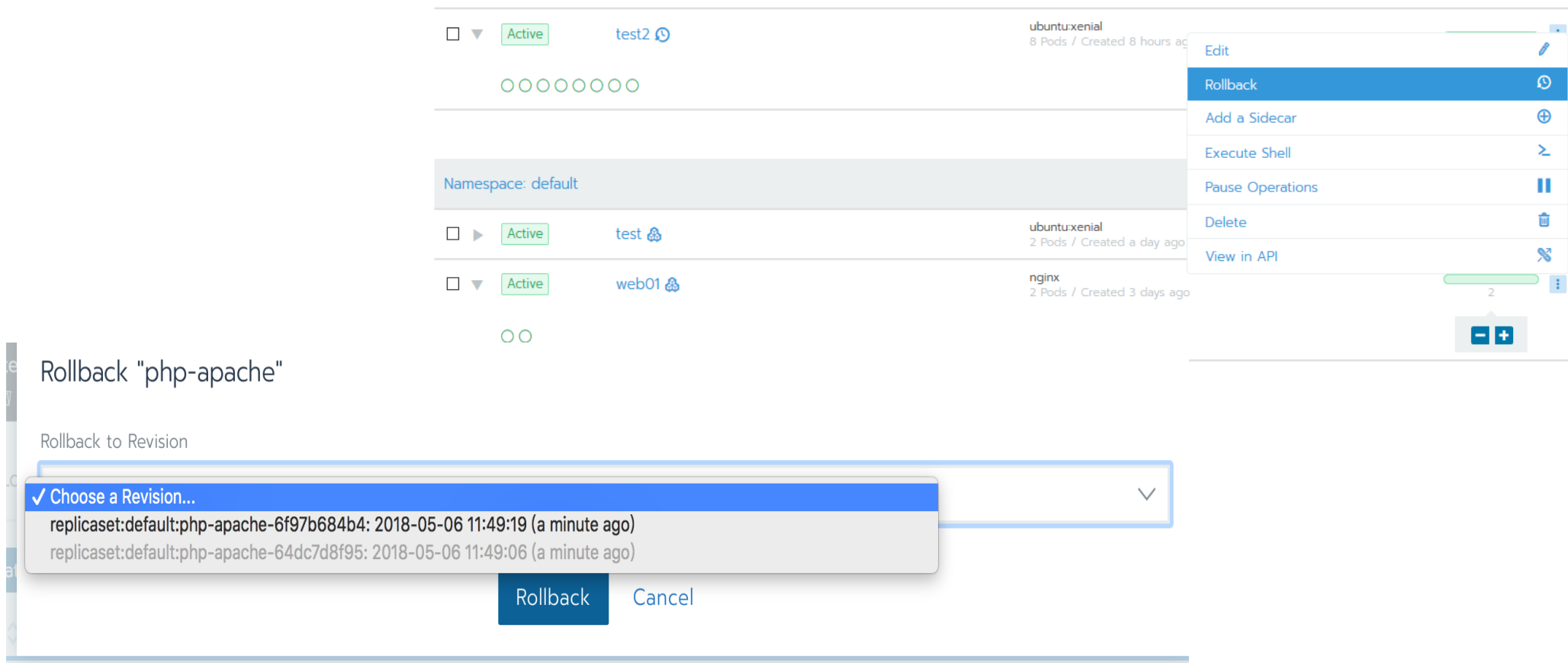
进度截止日期

秒

将进度不可见的部署标记为停滞状态的等待时间

# 应用灰度发布

- 容器云支持灰度升级，升级过程中服务不中断。灰度升级阶段，可以点击升级成功或回滚到原来版本。



# 应用资源管理

- Rancher容器云平台支持对应用的资源管理，主要分为应用调度最小资源需求和应用运行过程中最大资源限制，限制的资源主要分为CPU、memory、GPU

内存预留

例如： 128 MiB

内存限制

☒ 无限制

☐ 限制为 128 MiB

CPU预留

例如： 1000 milli CPUs

CPU 限制

☒ 无限制

☐ 限制为 1000 milli CPUs

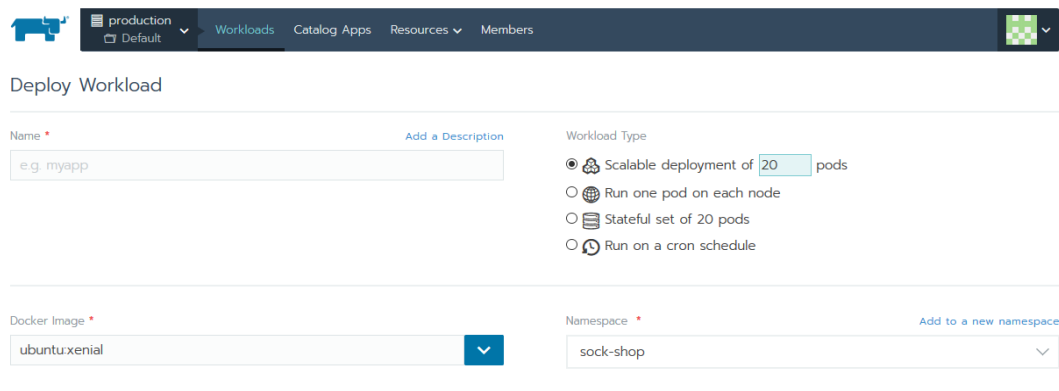
NVIDIA GPU预留

例如： 1 GPUs



# 应用的高可靠

- 容器云部署应用时可以设定服务容器的需要保证的有效数量。当系统主机故障时，云平台的控制器可保证有效的的容器数量，在健康主机上重新创建容器。可以保证应用服务维持指定的容器数量，以保证服务正常。



The screenshot shows the 'Deploy Workload' form in the Kubernetes Dashboard. The top navigation bar includes 'production', 'Default', 'Workloads', 'Catalog Apps', 'Resources', and 'Members'. The form has two main sections: 'Name' and 'Workload Type'. The 'Name' section has a text input field with 'e.g. myapp' and a link 'Add a Description'. The 'Workload Type' section has four radio button options: 'Scalable deployment of 20 pods' (selected), 'Run one pod on each node', 'Stateful set of 20 pods', and 'Run on a cron schedule'. Below these, there are two dropdown menus: 'Docker Image' with 'ubuntu:xenial' and 'Namespace' with 'sock-shop'.

production  
Default

Workloads Catalog Apps Resources Members

Deploy Workload

Name \* [Add a Description](#)

e.g. myapp

Workload Type

☒ Scalable deployment of 20 pods

☐ Run one pod on each node

☐ Stateful set of 20 pods

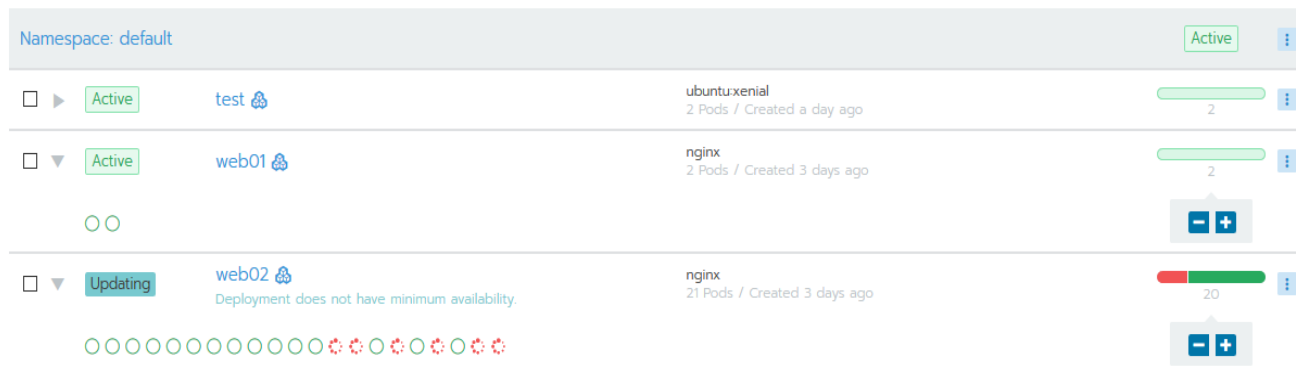
☐ Run on a cron schedule

Docker Image \* [Add to a new namespace](#)

ubuntu:xenial

Namespace \* [Add to a new namespace](#)

sock-shop



The screenshot shows the 'Workloads' page in the Kubernetes Dashboard. The top navigation bar includes 'Namespace: default', 'Active', and a menu icon. The table lists three workloads: 'test', 'web01', and 'web02'. Each workload has a status (Active or Updating), a name, a description, a container image, and a pod count. The 'web02' workload is in an 'Updating' state with a message 'Deployment does not have minimum availability'.

Namespace: default		Active	
<input type="checkbox"/>	Active	test	ubuntu:xenial 2 Pods / Created a day ago
<input type="checkbox"/>	Active	web01	nginx 2 Pods / Created 3 days ago
<input type="checkbox"/>	Updating	web02	nginx 21 Pods / Created 3 days ago

# Rancher2.0服务暴露方式

1. NodePort 在集群内每台宿主机上暴露
2. HostPort 在运行pod的节点上
3. ClusterIP 集群内部
4. LoadBlance 调用云厂商负载均衡器

## 部署工作负载

名称 *	<a href="#">添加描述</a>	工作负载类型	<a href="#">更多选项</a>
test		可扩展的部署 1 pod	
Docker镜像 *		命名空间 *	
nginx		default	
<b>端口映射</b>			
容器端口 *	协议		主机端口 *
80	TCP		例如: 80
<a href="#">+ 添加规则</a>			

# 部署一个应用

- Demo

# Authentication and Authorization in k8s & Rancher

1. Kubernetes RBAC Authorization
2. Rancher2.0 多集群RBAC管理
3. Rancher 2.0 的角色管理

# Kubernetes RBAC – role based authorization

1. 通过RBAC API控制 – Role, ClusterRole, RoleBinding, ClusterRoleBinding
2. Role
  1. 一系列的授权定义
  2. 通过添加更多的规则来定义操作更多资源的权限
3. Role Binding
  1. Binding User, Group, Service and Service Account

# Kubernetes Role & ClusterRole

1. Role: 在namespace内生效
2. ClusterRole: 在集群内生效, 不分Namespace
  1. Cluster Resources (e.g. Nodes)
  2. None-standard type of endpoint (e.g. “/health”)
  3. Resource like pod deployed in all namespaces

# Role

```
kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  namespace: default
  name: pod-reader
rules:
- apiGroups: ["" ] # "" indicates the core API group
  resources: ["pods"]
  verbs: ["get", "watch", "list"]
```

```
rules:
- apiGroups: ["" ]
  resources: ["configmaps"]
  resourceNames: ["my-config"]
  verbs: ["get"]
```

# Cluster Role

```
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  # "namespace" omitted since ClusterRoles are not namespaced
  name: secret-reader
rules:
- apiGroups: [""]
  resources: ["secrets"]
  verbs: ["get", "watch", "list"]
```

```
rules:
- apiGroups: [""]
  resources: ["nodes"]
  verbs: ["get", "list", "watch"]
```

```
rules:
- nonResourceURLs: ["/healthz", "/healthz/*"]
  verbs: ["get", "post"]
```



# Role/ClusterRole Binding

```
# This role binding allows "jane" to read pods in the "default" namespace.
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: read-pods
  namespace: default
subjects:
- kind: User
  name: jane # Name is case sensitive
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role #this must be Role or ClusterRole
  name: pod-reader # this must match the name of the Role or ClusterRole you wish to bind to
  apiGroup: rbac.authorization.k8s.io
```

```
# This cluster role binding allows anyone in the "manager" group to read secrets in any namespace
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: read-secrets-global
subjects:
- kind: Group
  name: manager # Name is case sensitive
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: secret-reader
  apiGroup: rbac.authorization.k8s.io
```

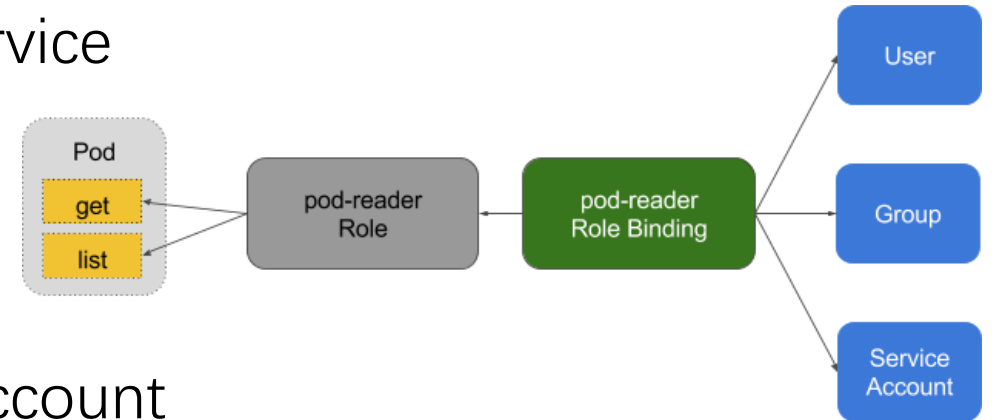
# Kubernetes Users

## 1. Normal Users

1. Managed by other independent auth service
2. Kubernetes didn't contains APIs for it

## 2. Service Accounts

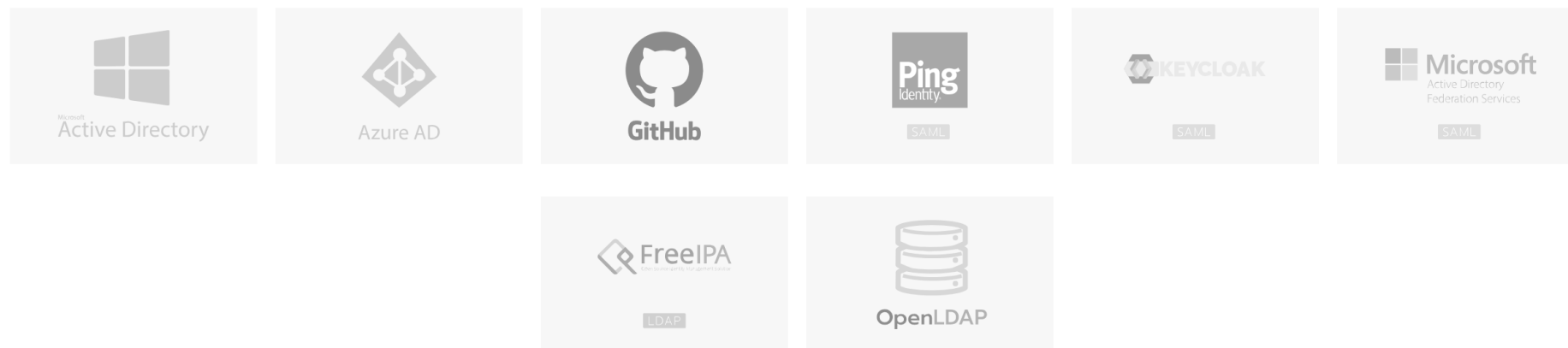
1. Managed by Kubernetes API's Service Account
2. Bind to the namespace



# Rancher 2.0 支持的认证方式



## Authentication



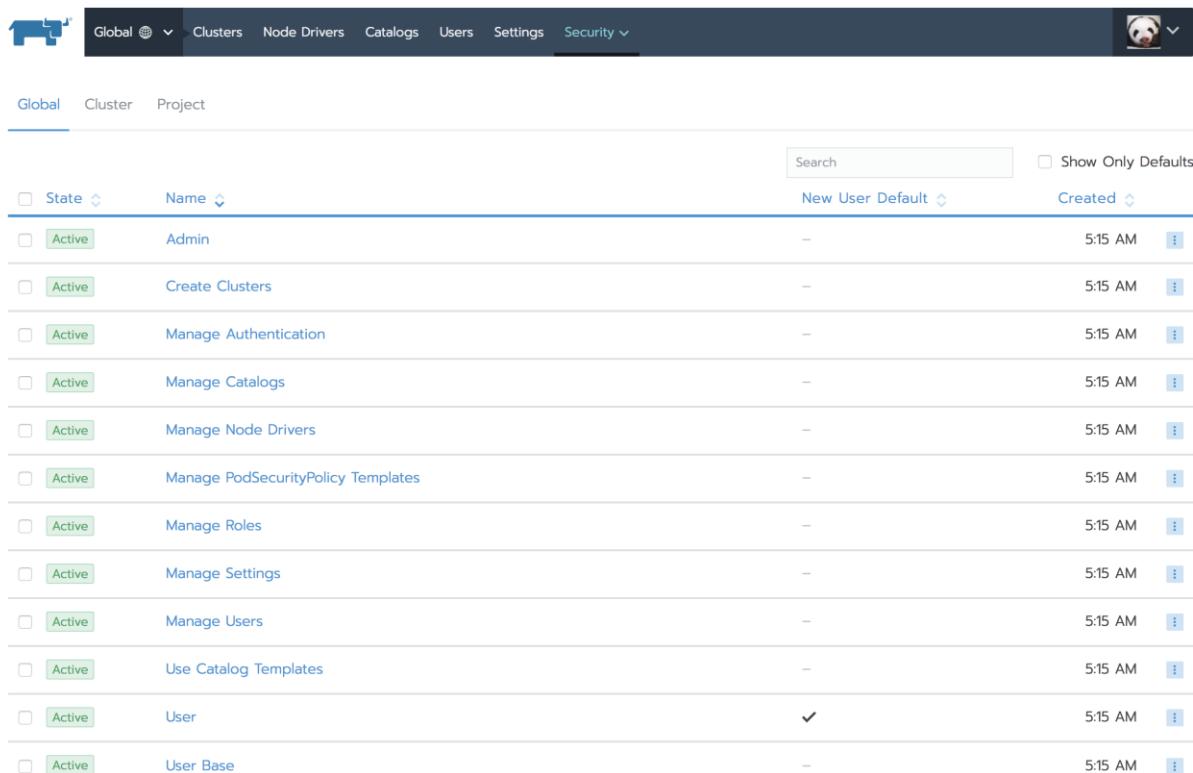
Rancher is configured to allow access to accounts in its local database. [Manage Accounts](#)  
Local Authentication will always be enabled but you may select another authentication scheme to use in addition to local.

# Rancher 2.0 RBAC

1. 在多集群管理中使用统一的认证系统
2. 使用Kubernetes RBAC
3. 通过Role+RoleBinding实现
4. 3个层级
  1. Global
  2. Cluster
  3. Project

# Global Level Authorization

- 针对像Users, Authentication, etc.这样的全局资源
- 包含预定义的全局权限

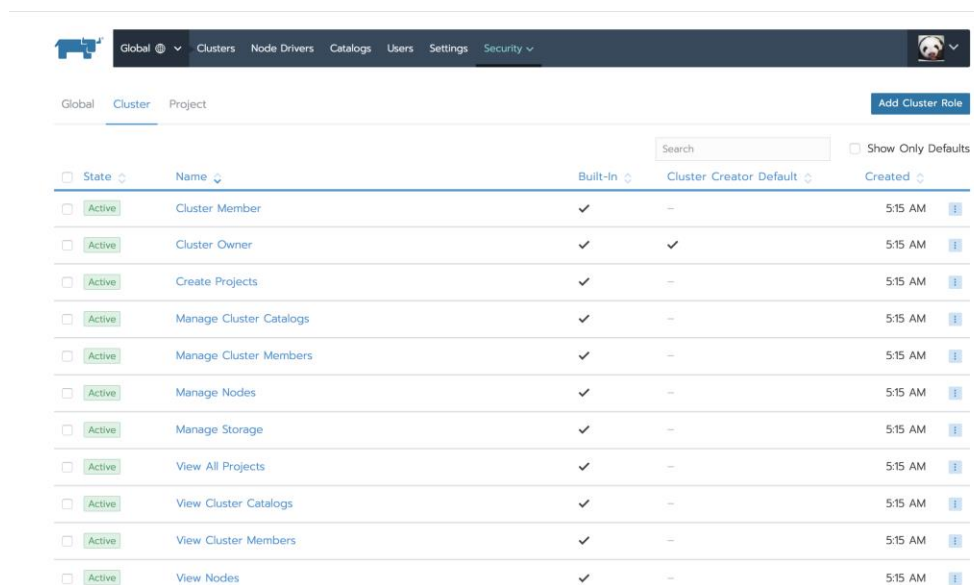


The screenshot shows the Databricks Global Level Authorization interface. At the top, there is a navigation bar with tabs for Global, Clusters, Node Drivers, Catalogs, Users, Settings, and Security. Below the navigation bar, there are tabs for Global, Cluster, and Project. The main content area displays a table of global resources with columns for State, Name, New User Default, and Created. The table lists various resources such as Admin, Create Clusters, Manage Authentication, Manage Catalogs, Manage Node Drivers, Manage PodSecurityPolicy Templates, Manage Roles, Manage Settings, Manage Users, Use Catalog Templates, User, and User Base. Each resource has a checkbox for selection and a green 'Active' status indicator. The 'User' resource is marked as the 'New User Default' with a checkmark.

State	Name	New User Default	Created
<input type="checkbox"/> Active	Admin	—	5:15 AM
<input type="checkbox"/> Active	Create Clusters	—	5:15 AM
<input type="checkbox"/> Active	Manage Authentication	—	5:15 AM
<input type="checkbox"/> Active	Manage Catalogs	—	5:15 AM
<input type="checkbox"/> Active	Manage Node Drivers	—	5:15 AM
<input type="checkbox"/> Active	Manage PodSecurityPolicy Templates	—	5:15 AM
<input type="checkbox"/> Active	Manage Roles	—	5:15 AM
<input type="checkbox"/> Active	Manage Settings	—	5:15 AM
<input type="checkbox"/> Active	Manage Users	—	5:15 AM
<input type="checkbox"/> Active	Use Catalog Templates	—	5:15 AM
<input type="checkbox"/> Active	User	✓	5:15 AM
<input type="checkbox"/> Active	User Base	—	5:15 AM

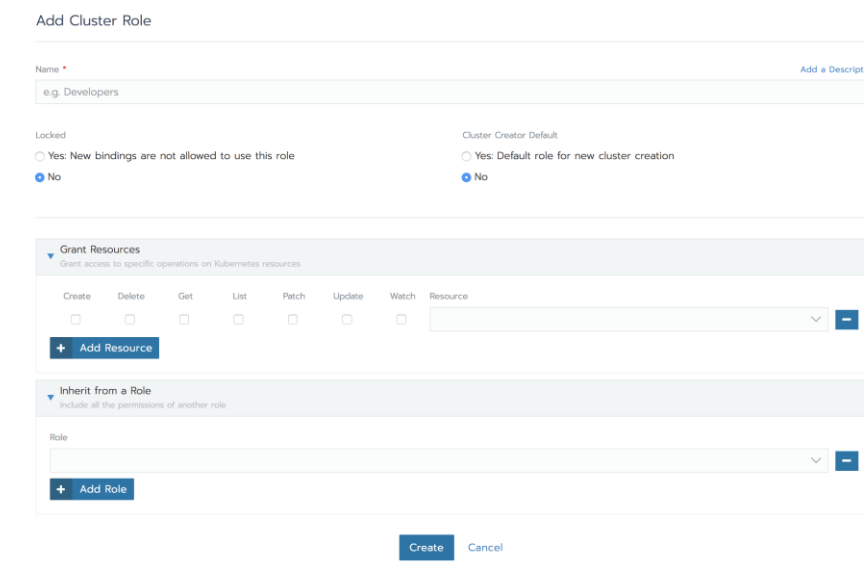
# Cluster Level Authentication

- 控制Cluster级别的资源
- 在实现上Cluster级别的权限控制通过ClusterRoleBinding来暴露求权限



The screenshot shows the Kubernetes dashboard with the 'Cluster' tab selected. A table lists built-in roles with columns for State, Name, Built-In, Cluster Creator Default, and Created. All roles are 'Active' and were created at '5:15 AM'.

State	Name	Built-In	Cluster Creator Default	Created
Active	Cluster Member	✓	—	5:15 AM
Active	Cluster Owner	✓	✓	5:15 AM
Active	Create Projects	✓	—	5:15 AM
Active	Manage Cluster Catalogs	✓	—	5:15 AM
Active	Manage Cluster Members	✓	—	5:15 AM
Active	Manage Nodes	✓	—	5:15 AM
Active	Manage Storage	✓	—	5:15 AM
Active	View All Projects	✓	—	5:15 AM
Active	View Cluster Catalogs	✓	—	5:15 AM
Active	View Cluster Members	✓	—	5:15 AM
Active	View Nodes	✓	—	5:15 AM



The 'Add Cluster Role' form includes a 'Name' field (e.g., 'Developers'), 'Locked' and 'Cluster Creator Default' checkboxes (all set to 'No'), and two sections for granting permissions: 'Grant Resources' and 'Inherit from a Role'. The 'Grant Resources' section has checkboxes for Create, Delete, Get, List, Patch, Update, Watch, and Resource, with a dropdown menu and an 'Add Resource' button. The 'Inherit from a Role' section has a 'Role' dropdown menu and an 'Add Role' button. 'Create' and 'Cancel' buttons are at the bottom.

# Project Level Authentication

- 控制Project级别的资源， 比如application.
- Project级别通过Rolebinding来实现

Global Cluster Project					Add Project Role	
					Search	<input type="checkbox"/> Show Only Defaults
<input type="checkbox"/> State	Name	Built-In	Project Creator Default	Created		
<input type="checkbox"/> Active	Create Namespaces	✓	–	5:15 AM	<a href="#">i</a>	
<input type="checkbox"/> Active	Manage Config Maps	✓	–	5:15 AM	<a href="#">i</a>	
<input type="checkbox"/> Active	Manage Ingress	✓	–	5:15 AM	<a href="#">i</a>	
<input type="checkbox"/> Active	Manage Project Catalogs	✓	–	5:15 AM	<a href="#">i</a>	
<input type="checkbox"/> Active	Manage Project Members	✓	–	5:15 AM	<a href="#">i</a>	
<input type="checkbox"/> Active	Manage Secrets	✓	–	5:15 AM	<a href="#">i</a>	
<input type="checkbox"/> Active	Manage Service Accounts	✓	–	5:15 AM	<a href="#">i</a>	
<input type="checkbox"/> Active	Manage Services	✓	–	5:15 AM	<a href="#">i</a>	
<input type="checkbox"/> Active	Manage Volumes	✓	–	5:15 AM	<a href="#">i</a>	
<input type="checkbox"/> Active	Manage Workloads	✓	–	5:15 AM	<a href="#">i</a>	
<input type="checkbox"/> Active	Project Member	✓	–	5:15 AM	<a href="#">i</a>	
<input type="checkbox"/> Active	Project Owner	✓	✓	5:15 AM	<a href="#">i</a>	
<input type="checkbox"/> Active	Read-only	✓	–	5:15 AM	<a href="#">i</a>	
<input type="checkbox"/> Active	View Config Maps	✓	–	5:15 AM	<a href="#">i</a>	
<input type="checkbox"/> Active	View Ingress	✓	–	5:15 AM	<a href="#">i</a>	

## Add Project Role

Name \*

e.g. Developers

Add a Description

Locked

☐ Yes: New bindings are not allowed to use this role

☒ No

Project Creator Default

☐ Yes: Default role for new project creation

☒ No

Grant Resources

Grant access to specific operations on Kubernetes resources

+ Add Resource

Inherit from a Role

Include all the permissions of another role

+ Add Role

Create

Cancel

# Kubeconfig file with RBAC configured

```
users:
- name: "user-slzjw"
  user:
    token: "kubeconfig-user-slzjw:br2qkpvnbwk2mzl5tc5bd8p7b9qj7r4tv879qtncljvmmf5rsw8z4"

contexts:
- name: "satomic"
  context:
    user: "user-slzjw"
    cluster: "satomic"

current-context: "satomic"
```



```
root@jenkins:~# kubectl get pods --all-namespaces
NAMESPACE   NAME                                     READY   STATUS    RESTARTS   AGE
cattle-alerting  alertmanager-649c5fc4d7-h7h9q         2/2     Running   4           4d
cattle-system   cattle-cluster-agent-55cb6f845-gd56v   1/1     Running   1           24d
cattle-system   cattle-node-agent-q9dl8                1/1     Running   6           22d
cattle-system   cattle-node-agent-vgkmv                1/1     Running   2           24d
default         nginx-668f465c85-jf9kg                 1/1     Running   0           1d
ingress-nginx   default-http-backend-564b9b6c5b-zs9rq  1/1     Running   6           24d
ingress-nginx   nginx-ingress-controller-dvhpv         1/1     Running   1           24d
```

```
users:
- name: "u-p8kth"
  user:
    token: "kubeconfig-u-p8kth:hhqfvx5qh9n2kjbz56m189j2nxvm7mjd7bmsm2qk74shxmcdchh"

contexts:
- name: "satomic"
  context:
    user: "u-p8kth"
    cluster: "satomic"

current-context: "satomic"
```



```
root@jenkins:~# kubectl get service --kubeconfig kubeconfig-test.yaml -n test-ns
NAME      TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
nginx     NodePort    10.43.223.196 <none>        80:30808/TCP     5h
root@jenkins:~# kubectl get pods --kubeconfig kubeconfig-test.yaml
Error from server (Forbidden): pods is forbidden: User "u-p8kth" cannot list pods in the namespace "default"
root@jenkins:~# kubectl get service --kubeconfig kubeconfig-test.yaml
Error from server (Forbidden): services is forbidden: User "u-p8kth" cannot list services in the namespace "default"
```




# 如何启动Github登陆并配置权限

- Demo

## Authentication

① Note: Only a single authentication provider, in addition to local authentication, may be enabled at any time. If you had multiple authentication providers enabled prior to 2.1, you may still edit or disable these providers but you can not enable additional providers. Nor will you be able re-enable a provider you previously disabled. Proceed with caution when disabling because you can not get them back.

  
GitHub

Authentication

Client ID: ef5f6b3329dc848f5441

Site Access

Configure who should be allowed to log in and use Rancher.


☐ Allow any valid Users


☒ Allow members of Clusters, Projects, plus Authorized Users and Organizations


☐ Restrict access to only Authorized Users and Organizations


Save


Your Groups


 rancher ( rancher )  
Organization

 rancherlabs ( rancherlabs )  
Organization


 dev ( dev )  
Team

 ECOS ( ecos )  
Team

 Employees ( employees )  
Team

 HNA ( hna )

Authorized Users and Organizations

 Default Admin ( admin )  
Local User

# Audit Log in Kubesnetes && Rancher

- 开启Kubernete的Audit Log
- 开启Rancher的Audit Log

# Audit Log in Kubernetes

- <https://rancher.com/docs/rke/v0.1.x/en/example-yamls/>

# Audit Log in Rancher

- <https://rancher.com/docs/rancher/v2.x/en/admin-settings/api-audit-log/>
- configure
  - AUDIT\_LOG\_PATH - Log path for Rancher Server API. Default path is /var/log/auditlog/rancher-api-audit.log
  - AUDIT\_LOG\_MAXAGE - Defined the maximum number of days to retain old audit log files, default is 10 days.
  - AUDIT\_LOG\_MAXBACKUP - Defines the maximum number of audit log files to retain, default is 10
  - AUDIT\_LOG\_MAXSIZE - Defines the maximum size in megabytes of the audit log file before it gets rotated, default size is 100M
  - 
  - AUDIT\_LEVEL - 0 - disable audit log, 1 - log event metadata, 2 - log event metadata and request body, 3 - log event metadata, request body and response body
  -
- Run Sample
  - `docker run -v /root/var/log/auditlog:/var/log/auditlog -e AUDIT_LEVEL=1 -e AUDIT_LOG_PATH=/var/log/auditlog/rancher-api-audit.log -e CATTLE_AGENT_IMAGE=rancher/rancher-agent:master -p 8080:80 -p 8443:443 micheliac/rancher:dev`

# Thank you

---

