Operation Manual for KubeManager — Sangfor PaaS Platform

(For Version 1.0)

Sangfor Technologies Inc.

Revision

Revision Number	Author	Date	Brief Introduction
1.0	Zhao Zhenyang	June 30, 2020	Initial draft

Contents

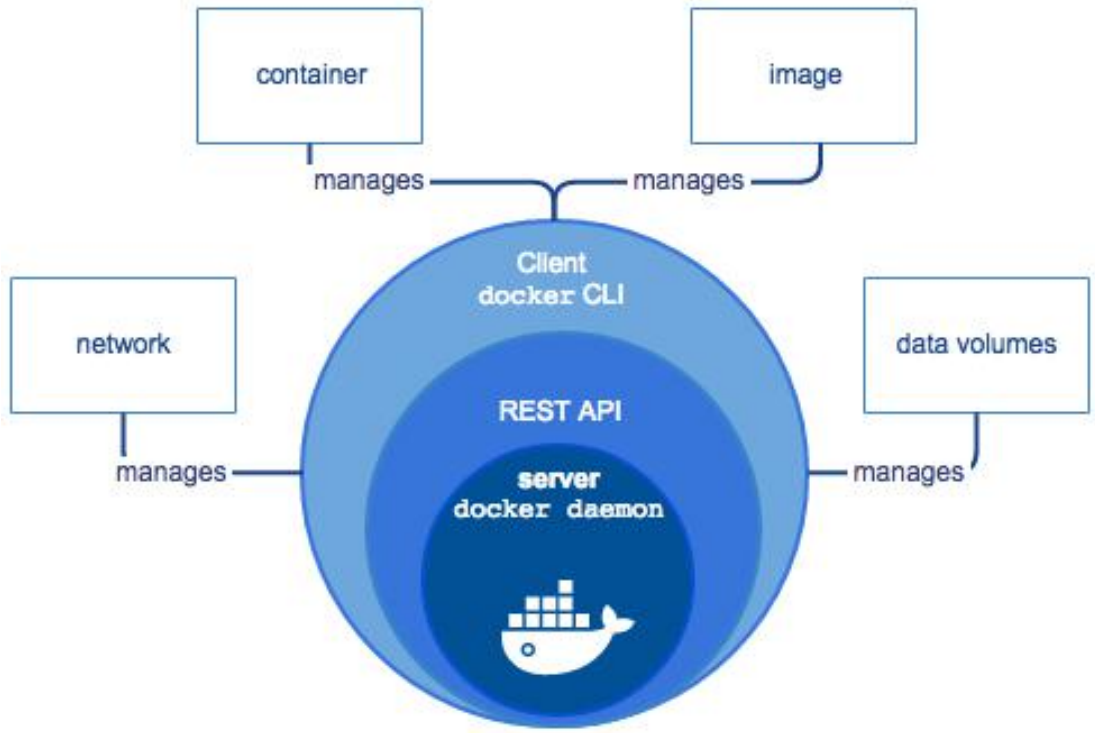
1 Overview of KubeManager

1.1 Development of Docker

The container technology originated from the namespace resource isolation technology and cgroup resource restriction technology of Linux kernel. Namespace saved path to Linux kernel in 2001. The container technology really became available after the release of LXC in 2008.

Linux Container is a kernel virtualization technology, which can provide lightweight virtualization and isolate processes and resources. Linux Container is abbreviated as LXC. LXC needs no instruction interpretation and full virtualization. LXC, the predecessor of docker, has been replaced by Libcontainer since version 0.9. Docker started with LXC, then optimized LXC, encapsulated components such as Libcontainerd and libnetwork, and managed them with runc.

The current Docker has three packages: containerd, docker-ce and docker-ce-cli. Docker-ce-cli replaces runc as the manager.

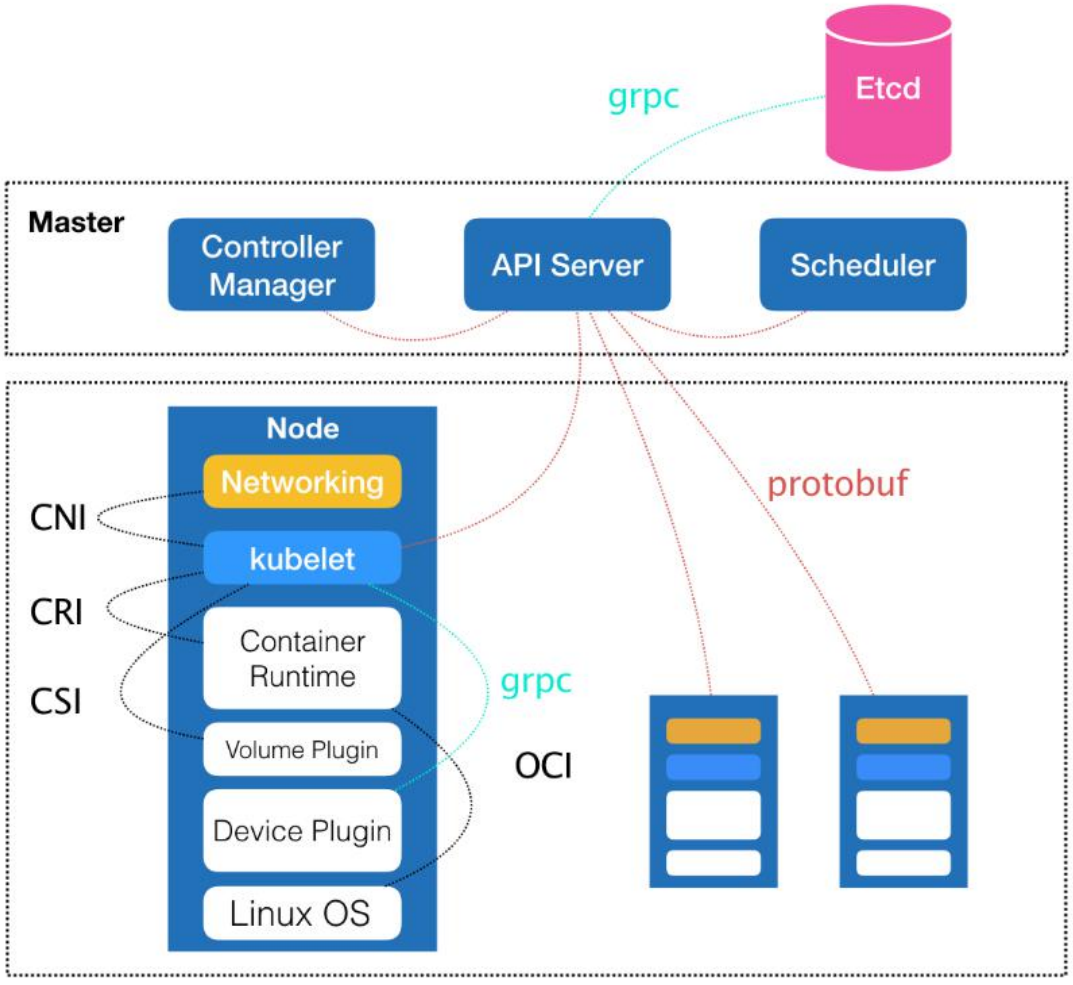


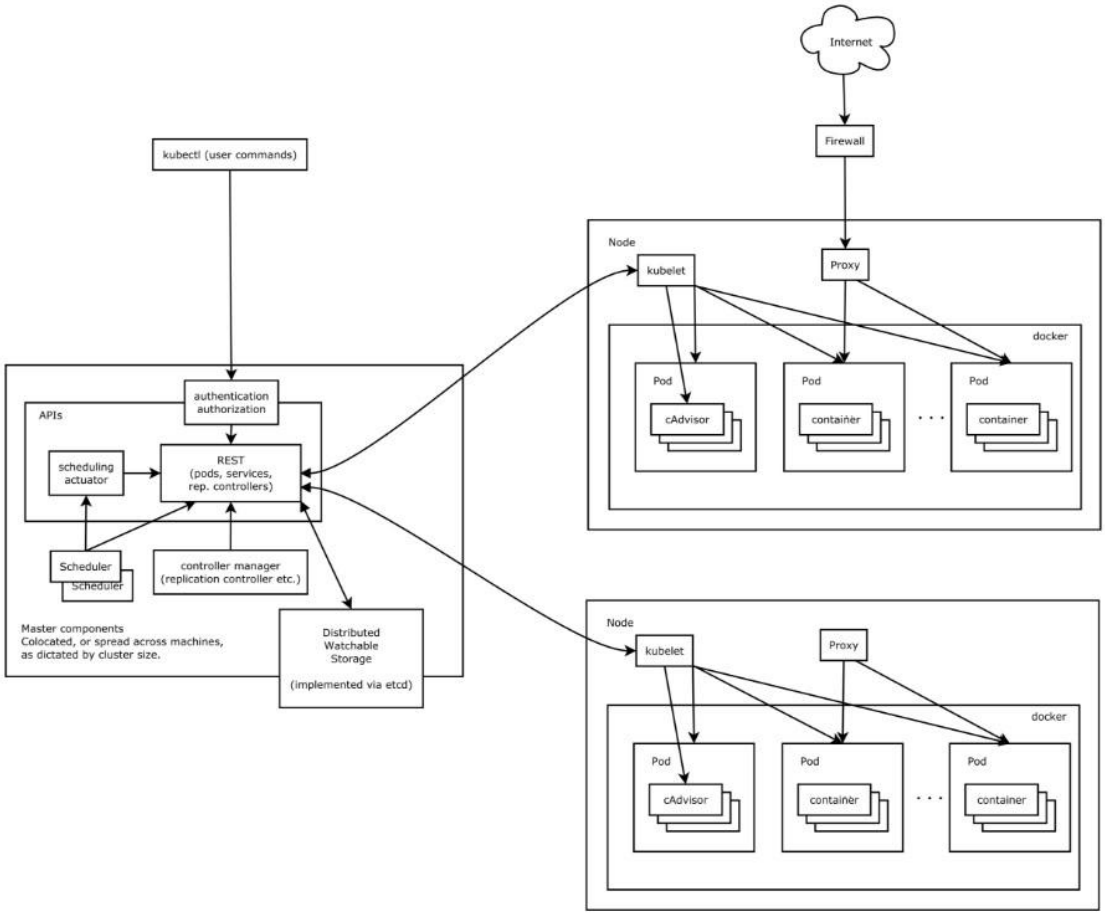
1.2 Kubernetes

Docker solves various problems caused by inconsistent environment in the process of

application deployment, but problems like container deployment, expansion and management on large clusters give birth to container orchestration engines, such as Kubernetes, Docker swarm and Mesos. Mesos was released by Twitter, swarm by Docker, and Kubernetes originated from the Borg system of Google. Kubernetes is abbreviated as K8S, in which 8 represents the 8 letters being omitted. K8S is portable, extensible and automatic.

K8S manages the above services with declarative API. K8S contains such components as ETCD, api server, controller manager, scheduler, kubectl, kubelet, kubeproxy, and docker. Generally, etcd, api server, controller manager, and scheduler are deployed in one node, which is usually called Master node. Other components are deployed in each node and become nodes. Master mainly stores the cluster configuration, schedules services in the cluster, and controls the cluster. Node is mainly responsible for the operation of containers, the hosting of services and the release of services.

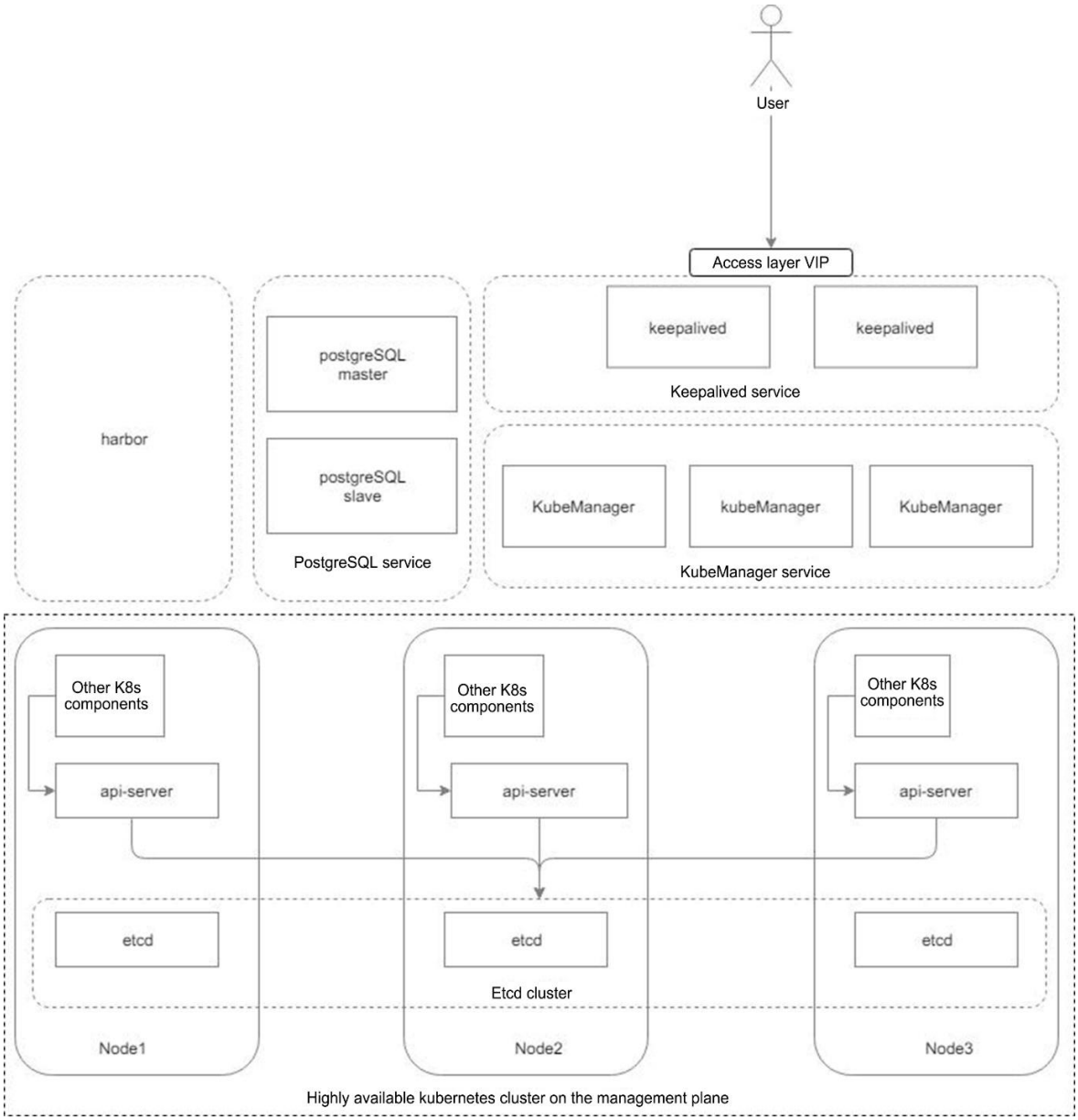




1.3 Sangfor PaaS Platform — KubeManager

KubeManager is a simple, easy-to-use and stable cloud native multi-K8S management platform released by Sangfor. By applying the cloud native technology, KubeManger runs all system components on K8S. KubeManager provides default registry and app store, supports integration with the third-party registry and app store, and supports third-party account system. KubeManager is a lightweight and stable commercial Kubernetes multi-cluster management system.

1.3.1 KubeManager Architecture



KubeManager is a PaaS system running on kubernetes. KubeManager provides multi-K8S management, registry, app store and other functions. Every component of the KubeManager system meets the high availability mechanism. On the underlayer, the high availability of kubenetes is transplanted to this system, to ensure the high availability of system, so that the system can provide external services normally when the duplicate of any node or any service fails.

The following subsystems ensure the high availability of the entire system:

- The keepalived service ensures the high availability of interfaces through floating IP addresses;

- The daemonset service ensures the access performance expansion and high availability of services;
- The postgreSQL service component ensures the high availability of databases. The operator component allows automatic configuration of various database modes (single host, master-slave, and one-master-to-multiple-slave) and can be used to store audit logs or harbor user information.
- The ETCD cluster ensures the high availability of configuration data. All configuration information of KubeManager is stored in K8S at the underlayer.
- KubeManger runs on kubernetes as a service. Kubernetes guarantees the high availability of all components of KubeManager;

2 Environment

2.1 Environment Deployment

2.1.1 Environment Requirements

Non-highly available environment					
Host	**Hostname**	**CPU**	**Memory**	**Disk**	**Installation component**
Host 1	KubeMager	4C	8GB	80 GB system disk without partition; mount 50 GB	Manage clusters, registries, and app stores
Host 2	etcd,controller, worker	8C	8GB	100 GB system disk without partition; mount 150 GB	User cluster
Host 3	etcd,controller, worker	8C	8GB	100 GB system disk without partition; mount 150 GB	User cluster
Host 4	etcd,controller, worker	8C	8GB	100 GB system disk without partition; mount 150 GB	User cluster

Highly available environment					
Host	**Hostname**	**CPU**	**Memory**	**Disk**	**Installation component**
Host 1	KubeMager	4C	8GB	80 GB system disk without partition; mount 50 GB	Manage clusters, registries, and app stores
Host 2	KubeMager	4C	8GB	80 GB system disk without partition; mount 50 GB	Manage clusters, registries, and app stores
Host 3	KubeMager	4C	8GB	80 GB system disk without partition; mount 50 GB	Manage clusters, registries, and app stores
Host 4	etcd,controller, worker	8C	8GB	100 GB system disk without partition; mount 150 GB	User cluster
Host 5	etcd,controller, worker	8C	8GB	100 GB system disk without partition; mount 150 GB	User cluster
Host 6	etcd,controller, worker	8C	8GB	100 GB system disk without partition; mount 150 GB	User cluster
Host 7	worker	8C	8GB	100 GB system disk without partition; mount 150 GB	User cluster
Host 8	worker	8C	8GB	100 GB system disk without partition; mount 150 GB	User cluster

KubeManager can be installed on VM, bare metal, cloud server and other infrastructures, and supports various network topologies. For example, the deployment in VPC, single NIC deployment in classic network, multi-NIC deployment in classic network.

2.1.2 Installation of Manage Cluster

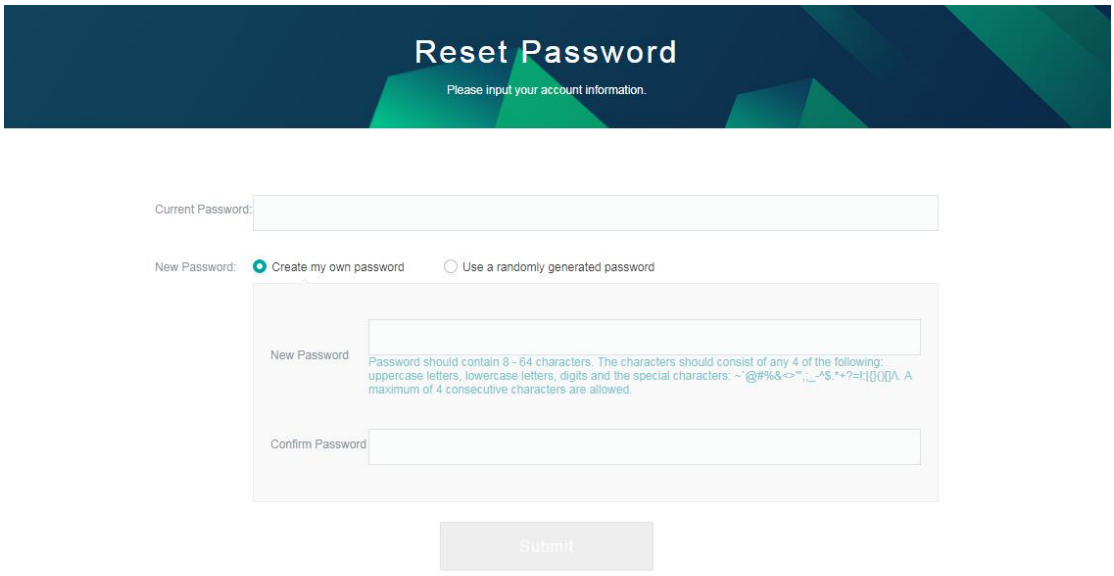
The installation of KubeManager cluster is very simple and convenient, including the following steps:

- Install the operating system

- Set disk partition and time
- Assign IP addresses and plan networks
- Configure ssh server
- Execute the installation program

The installation script has only one command. Then KubeManager can be installed based on the configured parameters. Once installation is completed, you can use the excellent service provided by KubeManager after simple initial setup.

You need to set the default password of admin when logging in for the first time:



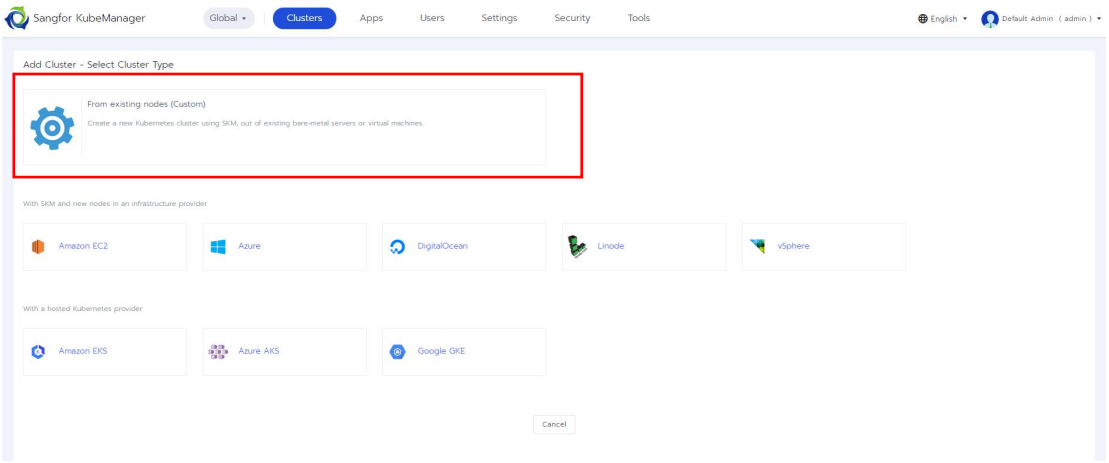
Select "Custom password" to define your own password or select "Random password" to generate a random password. After the password is set, it jumps to the homepage of KubeManager.

2.1.3 Installation of User Cluster

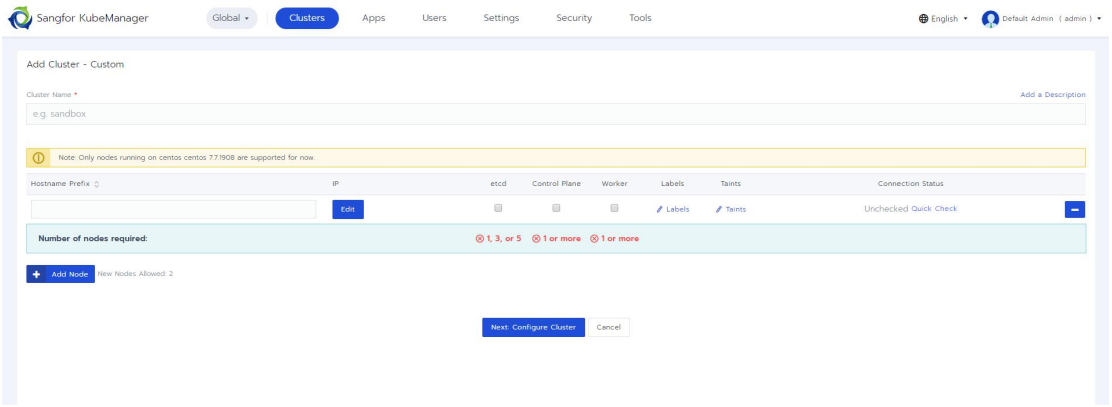
You can directly install the K8S cluster on the KubeManager interface, which is very convenient and quick.

First, login to the KubeManager platform, and then shift to the **Global** mode:

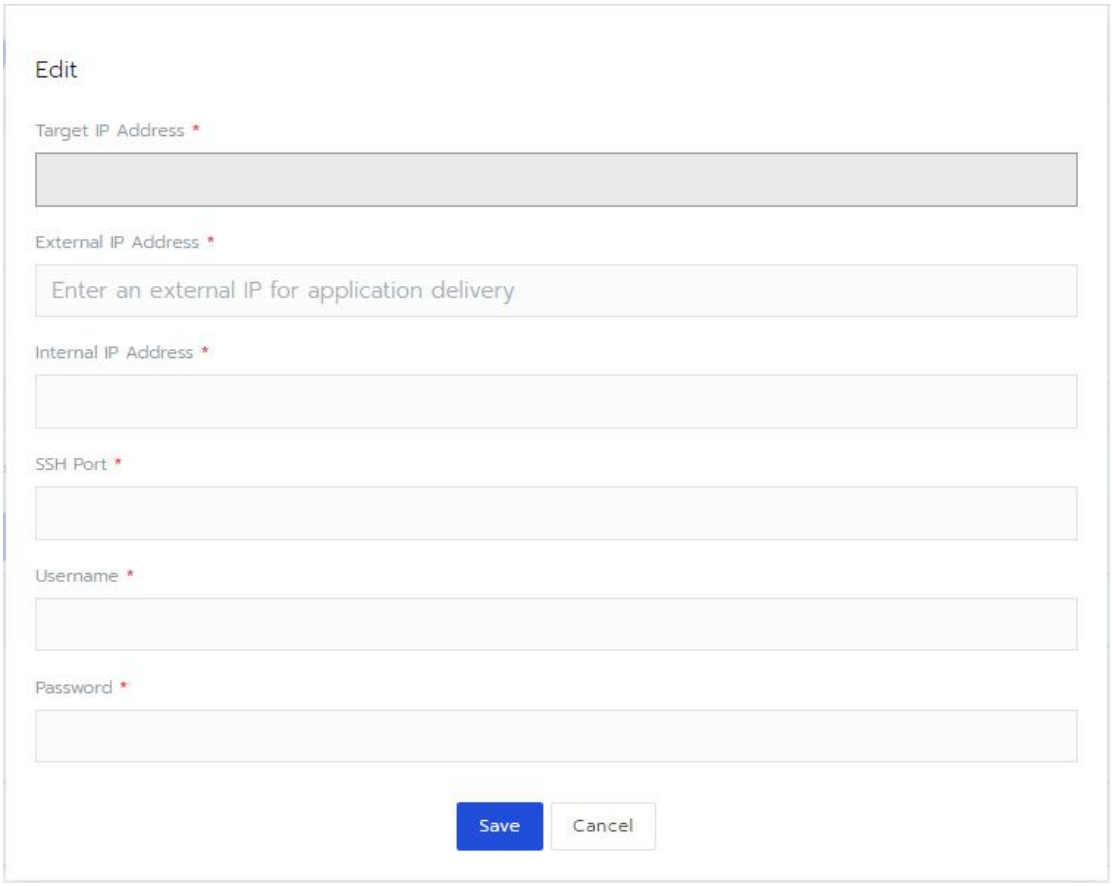
Click **Add cluster**, and select **Custom cluster**. If other clusters are required, select the corresponding options as prompted:



Add node for the cluster:

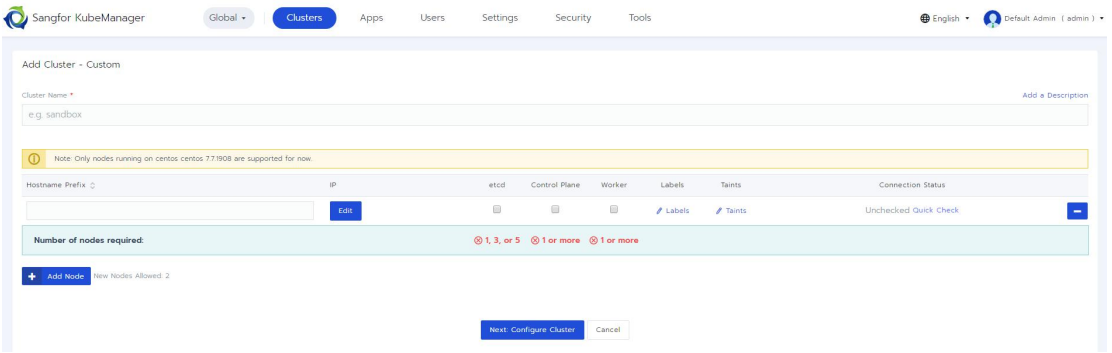


Configure the node:

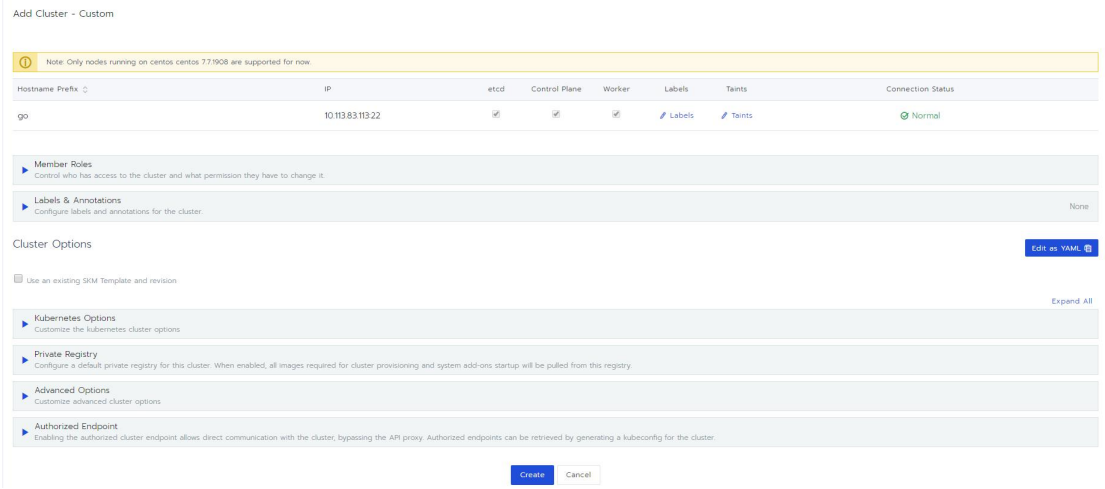


Check the connectivity of node after configuration.

Test the connectivity of the node and set the role of node:



Configure clusters, including network, ingress port, private registries;

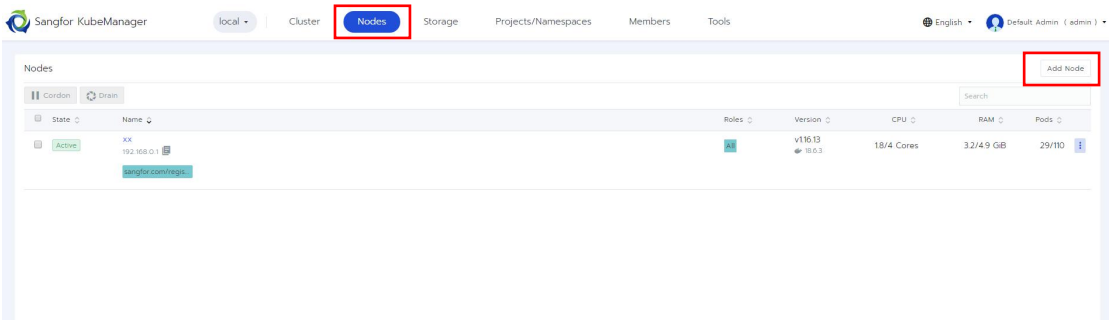


2.2 Environment Maintenance

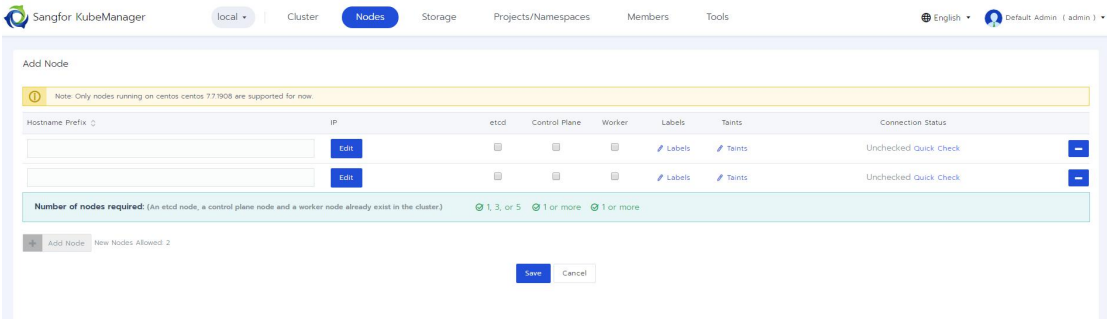
2.2.1 Expansion and Addition of User Cluster

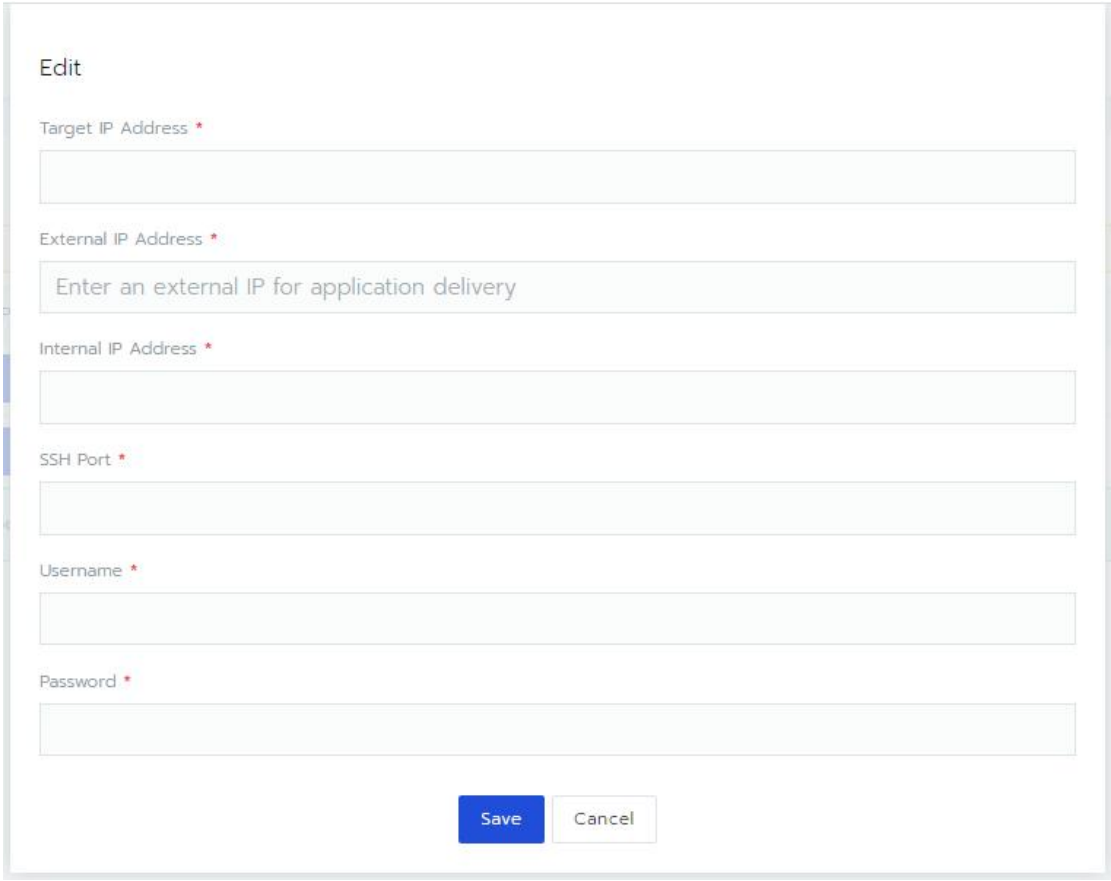
When users use K8S cluster, with the use of resources, they need to expand or add nodes. Add nodes following these steps:

Click a cluster to jump to the Nodes list page, and click the **Add Node** button:



As with cluster creation, add the intervention information and roles of nodes in configuration.



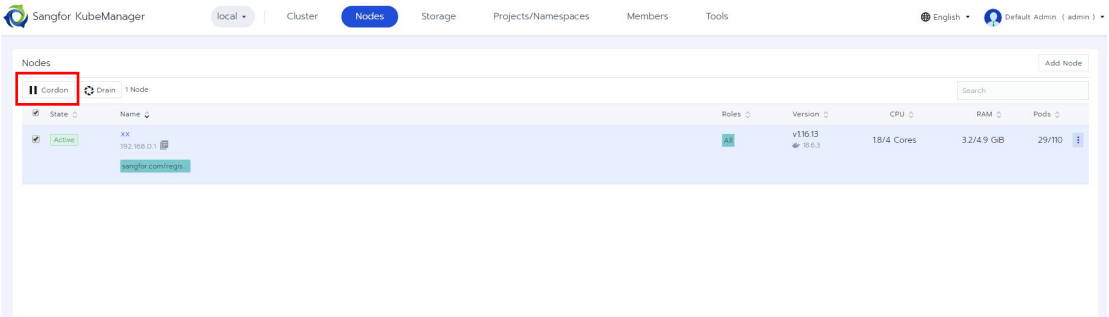


Then, after checking the connectivity of nodes, install and configure nodes for the system and expand the cluster.

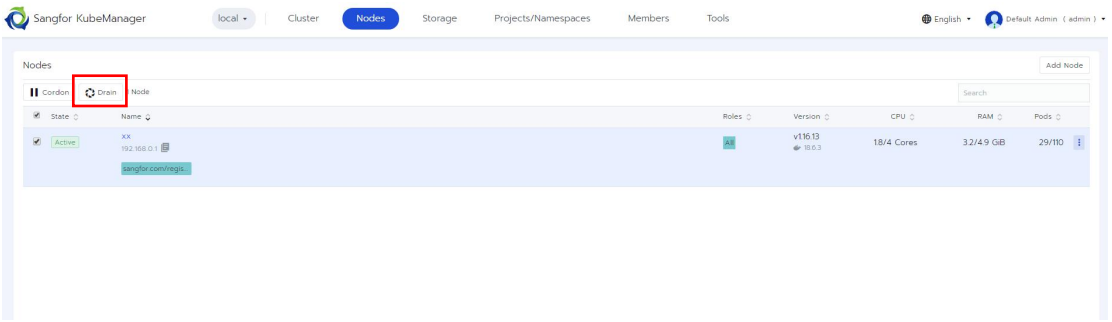
2.2.2 Maintenance and Deletion of Nodes

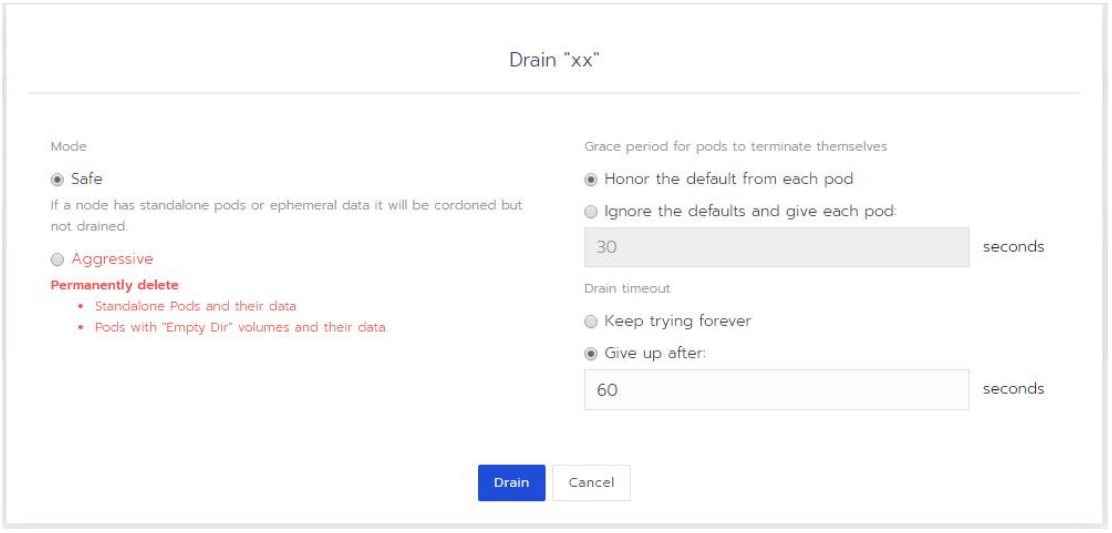
Maintain nodes when there is any fault. When a node failure cannot be fixed, delete this node and then add a new node to replace it.

First, **Cardon** the node. Then, the new service will not be scheduled to this node any more:

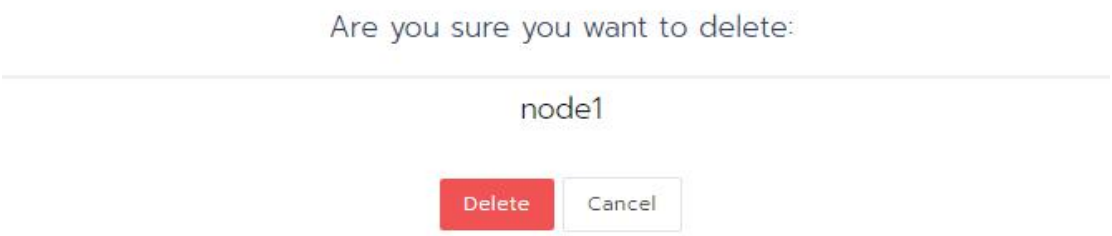


Then, **Drain** this node, to migrate the service on this node to other nodes seamlessly:

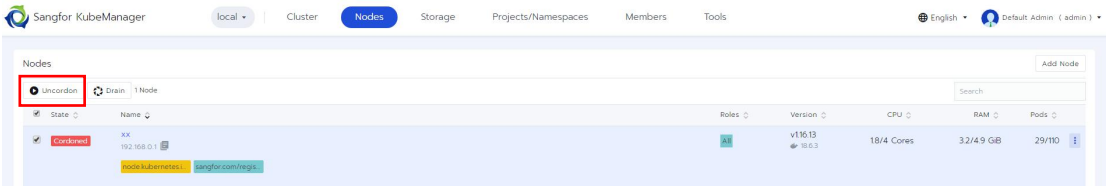




After the service on the node is evacuated, you can safely delete this node:



Nodes that have been **pause** and **scatter** can be **Uncordon** after being fixed, and service scheduling is supported again:



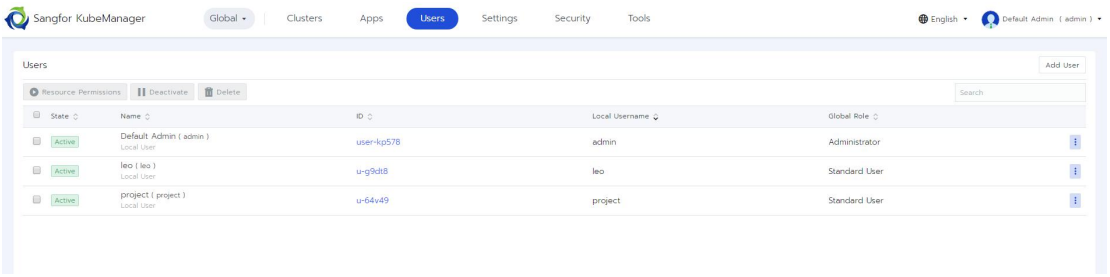
3 User System

3.1 Overview of User System

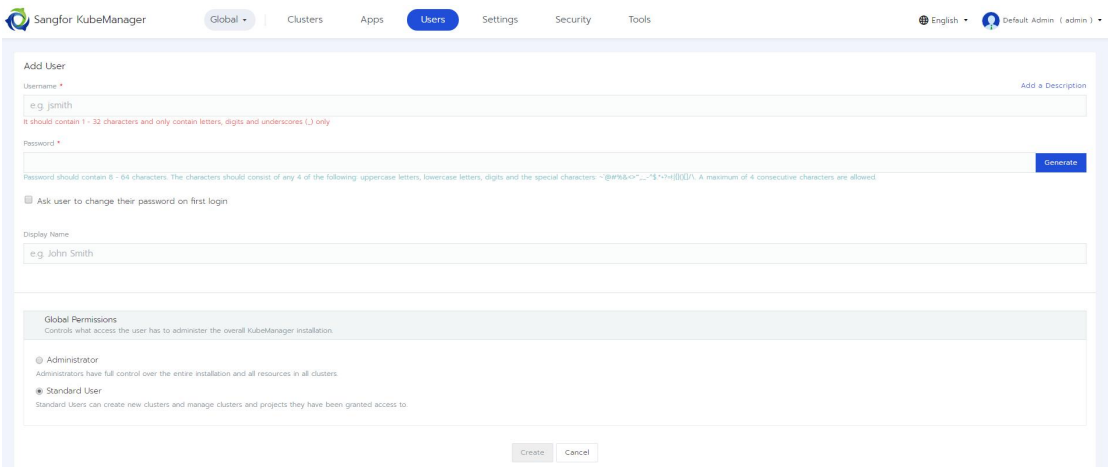
KubeManager supports a complete system of users and permissions. KubeManager supports the local user system, creation of local users, and integration with existing third-party permission systems, such as LADP and AD. Both local users and integrated third-party users have three-level permissions: **global**, **cluster** and **project**, and can customize permissions.

3.2 Local User

Click **Global** and view user list:

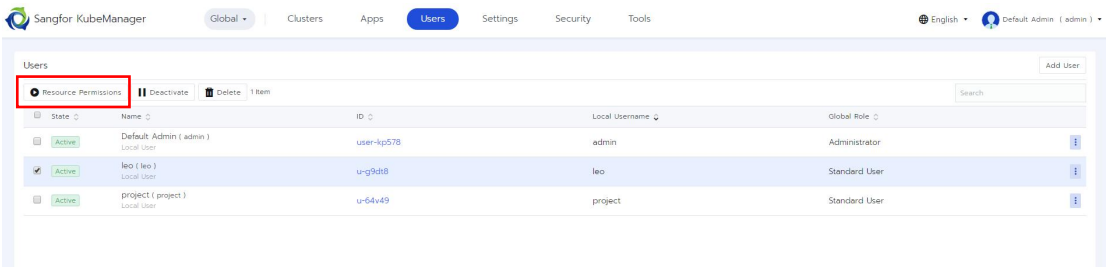


On the user list interface, add users:

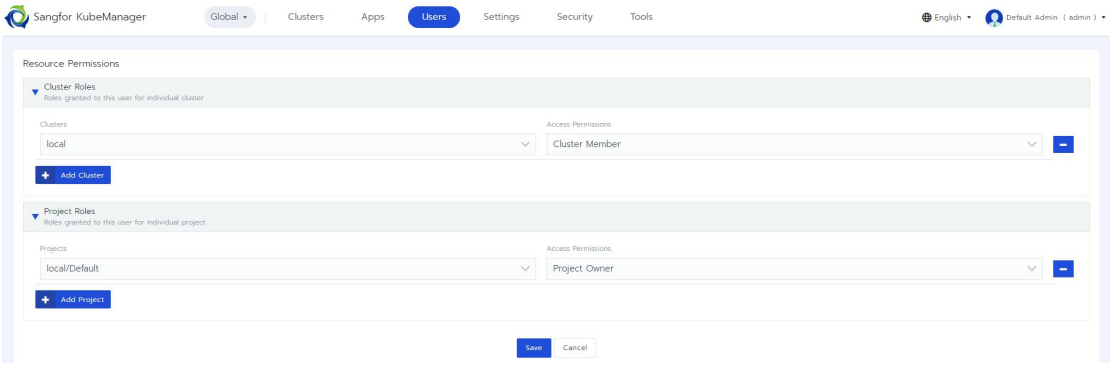


Create a local user, enter the username, set the password, or use the default password generated by the system, and set the user role to generate a local user.

Set accessible Resource Permissions for a user through resource authorization:



On the resource authorization page, select the cluster and project to be added, and choose the permissions for a user to access these resources:



After resource authorization, users can login to use and manage these resources and fulfill their service goals.

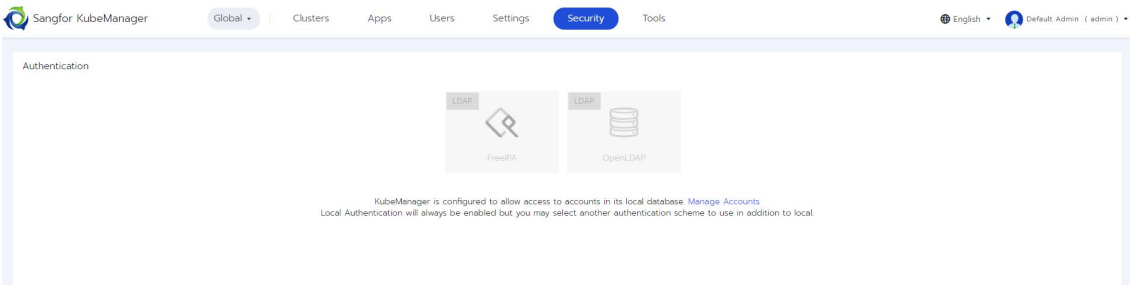
Similarly, for the user list at the **cluster** level, the global administrator can set an administrator and users for the cluster. The cluster administrator can set permissions for the cluster member list.

At the **Project** level, an administrator can add users and administrators for projects, and a project administrator can also set permissions for the project member list.

3.3 Integration with Third-party Account

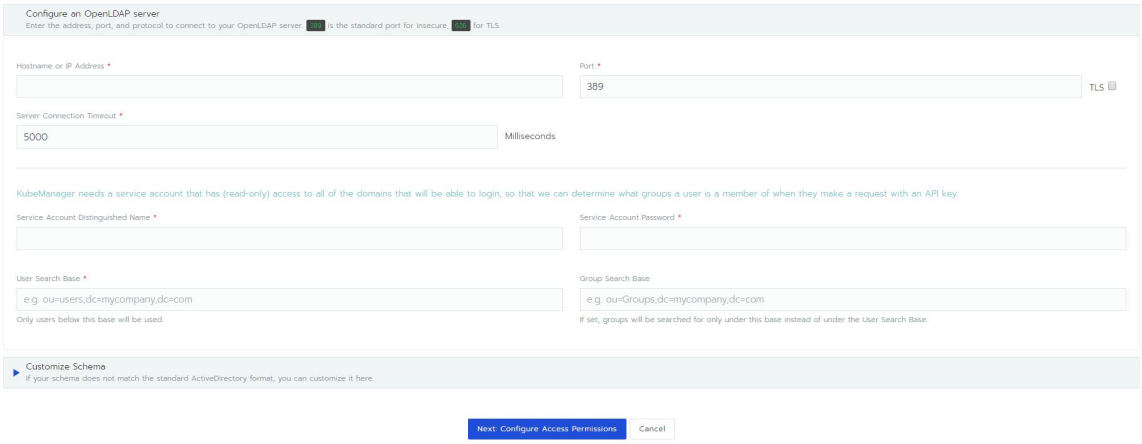
KubeManager supports the integration with third-party account systems such as LDAP, so that users can use their existing user systems directly without recreating users.

Choose **Authentication** from **Security** drop-down menu at the **Global** level, to integrate with the third-party user system:

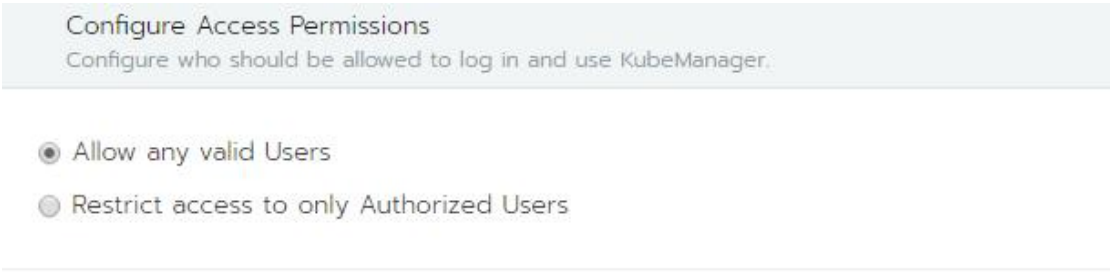


At present, accounts can be managed in a unified manner with LDAP. First, select **OpenLDAP**.

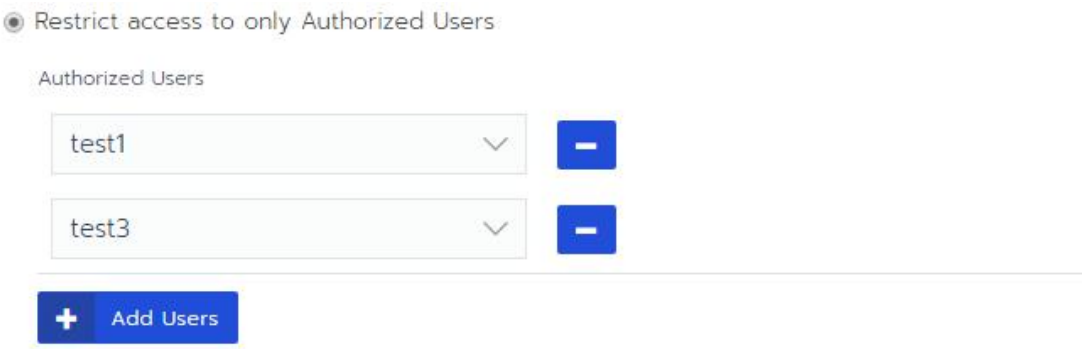
Configure the third-party account system:



Second, configure the access permission:



There are two types of access permissions. One is to allow all users on LDAP to access KubeManager, namely **Allow all valid users**. Another one is to allow the authorized LDAP users to access it, namely **Allow authorized users only**. For the latter one, you need to authorize the users on LDAP before they login to KubeManager. Follow these steps:



After importing the third-party accounts into this system, you can use them as the ordinary local account.

4 Registry

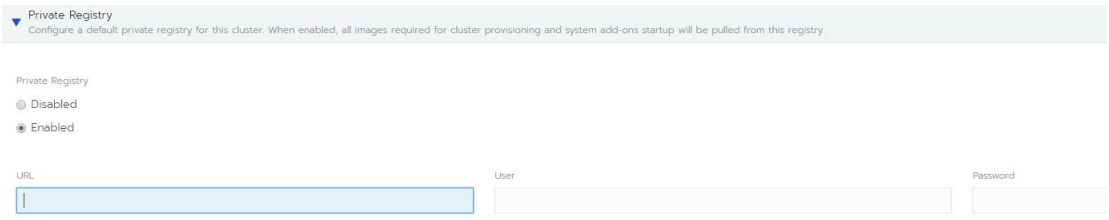
4.1 Deployment and High Availability

The registry of KubeManager system is deployed together with KubeManager, and it also runs on K8S. The database of the registry also runs in pod, but the high availability is maintained by our operator.

4.2 Private Registry

Private registries refer to the registries of system images such as the installation package used when installing the cluster. For hybrid cloud and multi-cloud scenarios, there are multiple registries distributed in different areas. To install the cluster quickly, it is the best way to choose the nearest registry.

When creating or editing a cluster, you can configure the registry as follows:

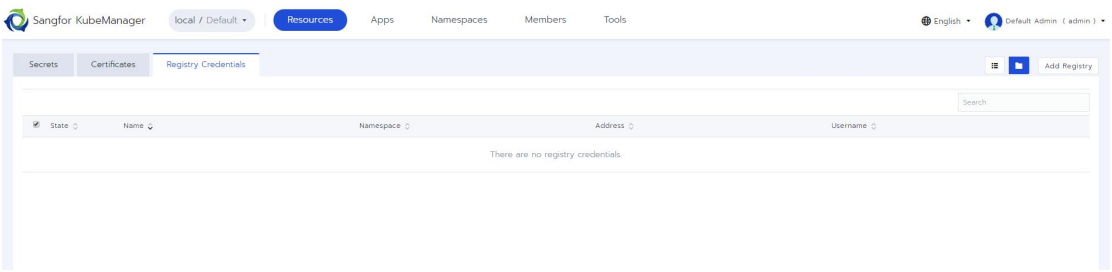


4.3 Registry Configuration

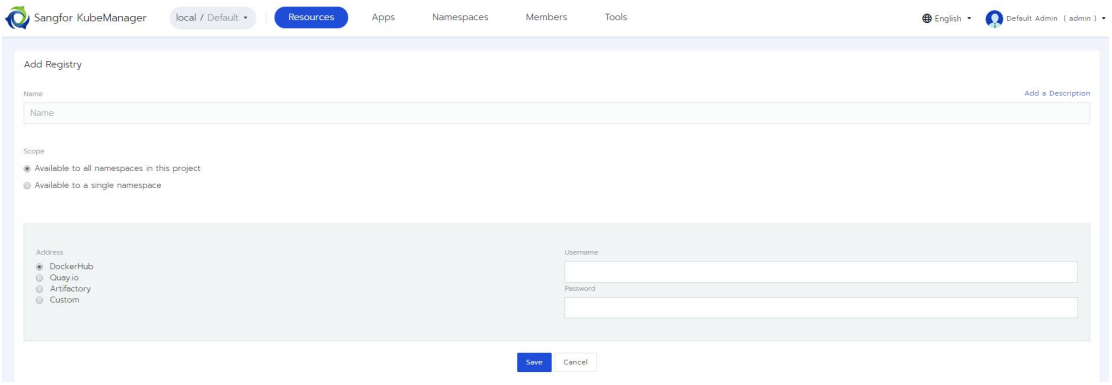
The private registry refers to the registry where system components are located when the cluster is installed. In reality, users often use multiple registries. The KubeManager successfully installed has a default registry, which can be accessed globally by default.

KubeManager allows customers to configure multiple different registries, that is, configuring one or more different registries in different projects. The registry can be either a public or a private one. For a private one, you need to provide a username and password.

Click into the project where the registry is to be configured. Choose **Secerets** from the drop-down menu of **Resources**, and click **Add** in **Registry Credentials** to configure the registry:



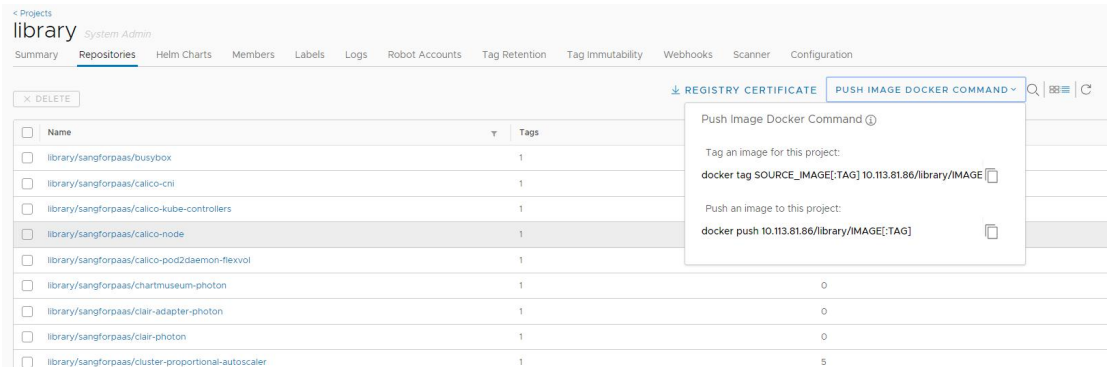
The configuration process is as follows:



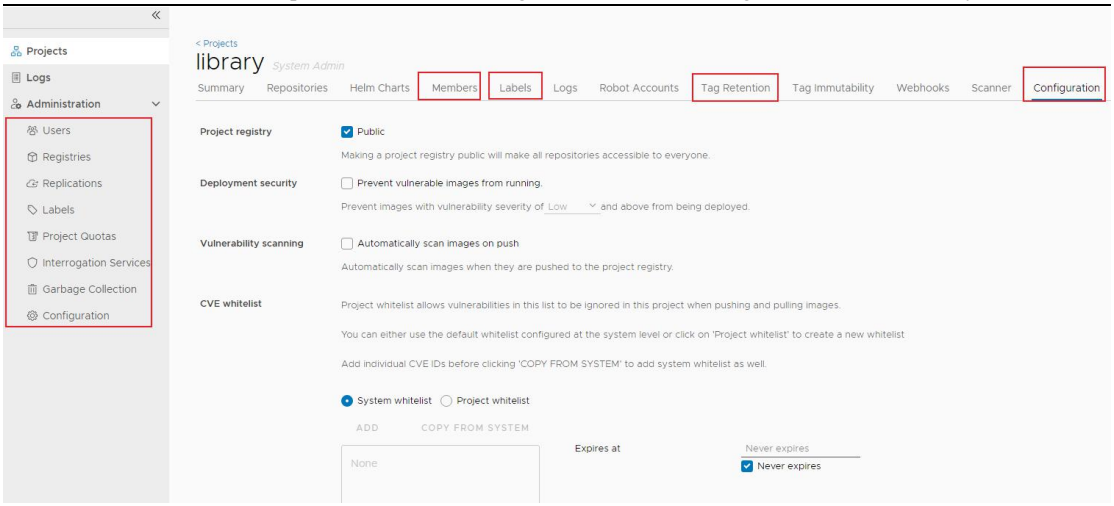
You can configure an effective namespace for the registry and set the registry type (any type is acceptable). We recommend using https registry as much as possible.

4.4 Image Upload and Management

We recommend uploading the image by docker push. First, download the https certificate from the registry, label the image with docker's command, and then push it to the registry.



In the registry, you can manage the image version, registry's permissions, and security policies. The permission and management system of the registry are independent of the use system of KubeManager.



5 App Store

5.1 Deployment and High Availability

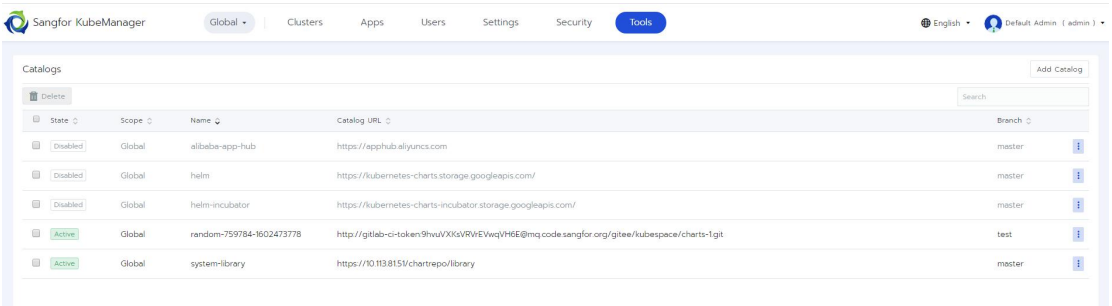
The app store of KubeManager system is deployed together with KubeManager, and it also runs on K8S. The database of the App Store also runs in pod, but the high availability is maintained by our operator.

By default, the system's app store and the registry are implemented through the same Harbor.

5.2 Configuration of App Store

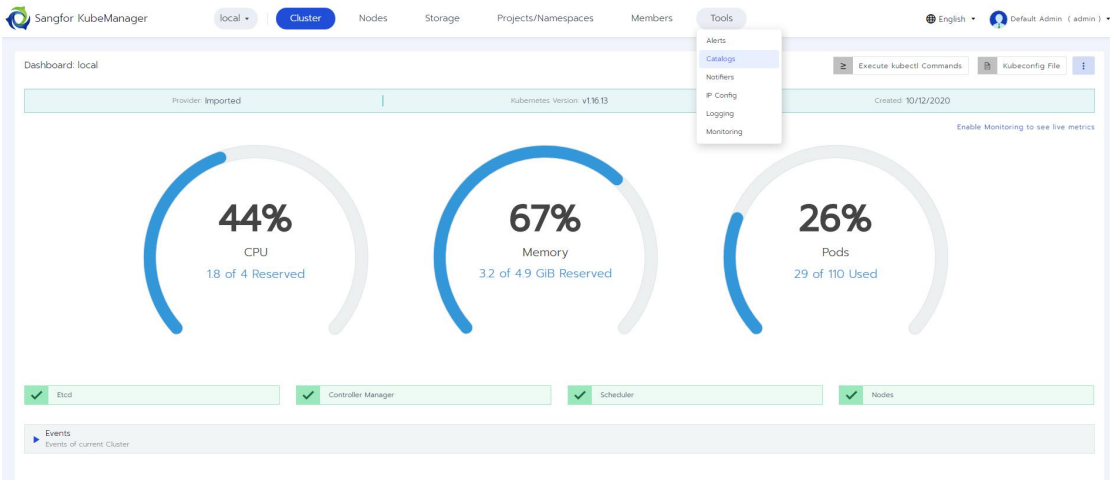
Like other resources, the app store can be set at the **global**, **cluster** and **project** levels. The app store can be public or private.

Choose **Catalogs** from the drop-down menu of **Tools** at the **Global** level, to configure the app store at the global level:

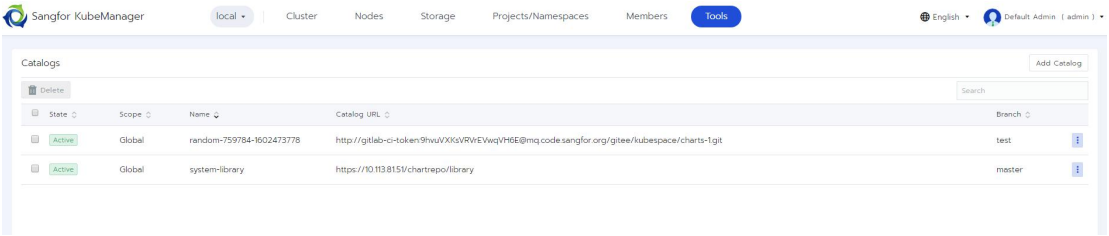


The app store at the global level can be used in all clusters. If some clusters cannot get some images from the app store at the global level, there will be an error of image acquisition failure.

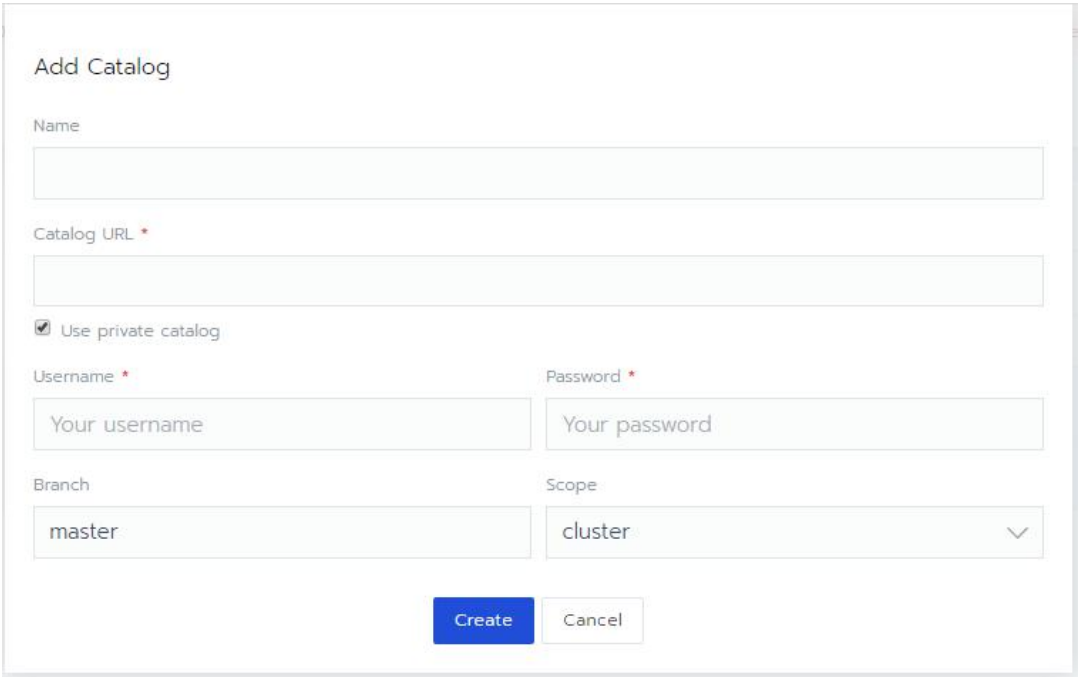
Select the cluster, and choose **Catalogs** from the drop-down menu of **Tools**, to set the app store at the **cluster** level:



Then choose **Add Catalog** to set the store in the project:



Configure the app store as follows:

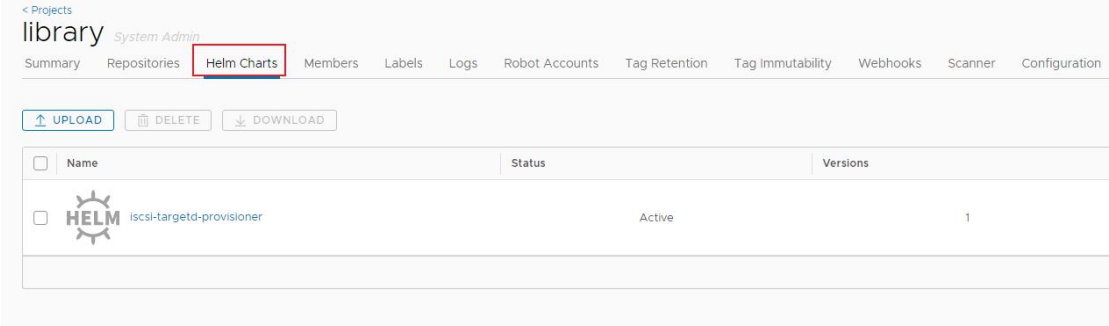


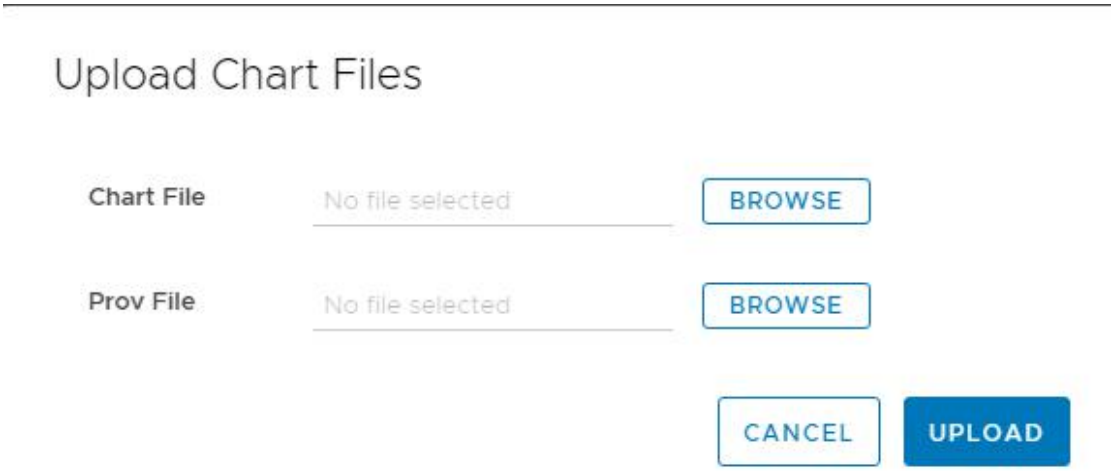
5.3 Application Upload and Management

To upload a new App developed by the existing application to the app store, follow these two steps:

- Upload the image required by the application to the corresponding registry
- Upload chat package to app store

The first step is the same as that for the registry. The second step is to operate on the Harbor page as follows:

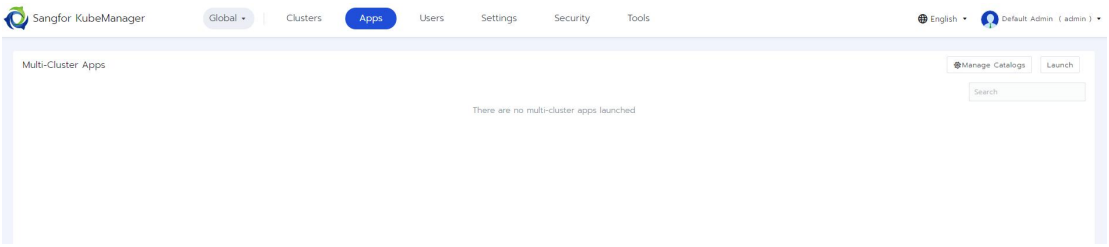




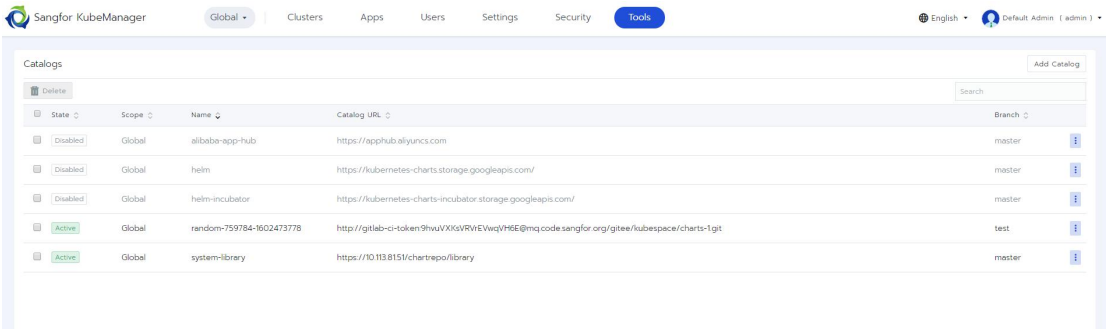
5.4 Multi-cluster Applications

KubeManager can create multi-cluster applications in addition to the creation of applications in the app store. The multi-cluster application allows the same application to be deployed in different projects of multiple clusters, and supports the unified management, upgrade, rollback and other maintenance operations. Multi-cluster applications at the global level: the application in the app store at the global level can be deployed in multi-cluster applications.

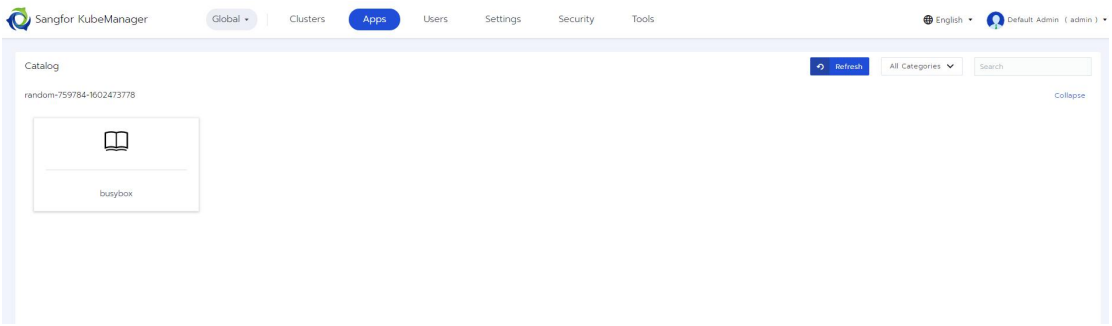
Click the **Apps** menu at the **Global** level:



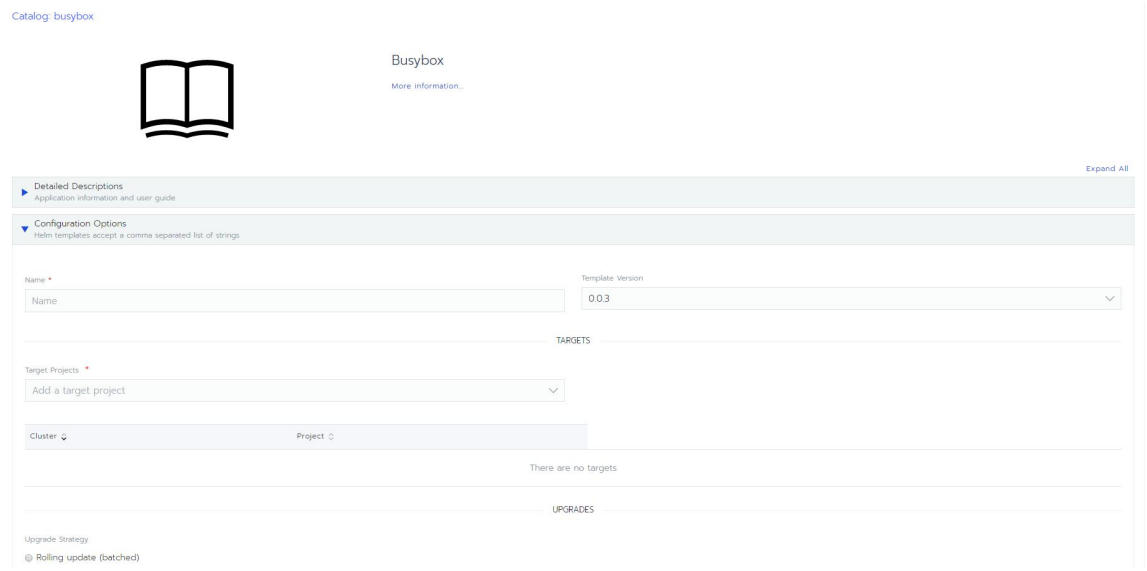
Click **Manage Catalogs** to configure and manage the applications in app store at the global level:



Click **Launch** to load all applications in the app store and classify them according to the store:



Click on the application in the app store to deploy multi-cluster applications:



6 Storage and Use

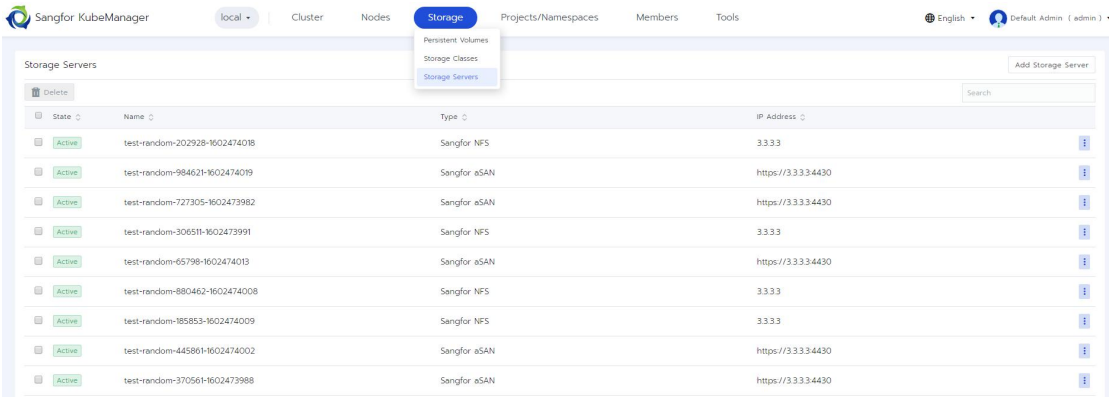
KubeManager supports a variety of common storage services through the built-in CSi driver, such as Amazon EBS Disk, Azure Disk, google persistent Disk, VMWare vSphere Volume, NFS, and sangfor asan. If deployed with Sangfor cloud devices such as Sangfor HCI, it could provide services by directly using HCI's virtual machine storage.

Moreover, it supports other types of storage too, such as glusterfs and ceph, as long as the user provides provisioner.

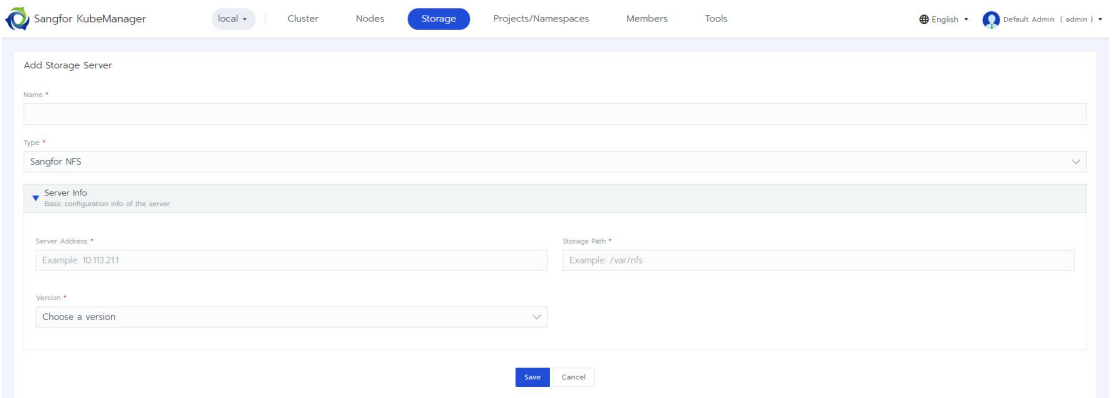
6.1 Storage Server

For NFS and sangfor asan, for the management convenience, we manage them via storage servers. A storage server can manage multiple PVs and SCs.

Click a cluster, and choose **Storage Servers** from the drop-down menu of **Storage**. The list of storage servers is displayed:



Click **Add Storage Server** to add a storage server:



The storage server supports two types of servers, namely sangfor asan and sangfor nfs. The

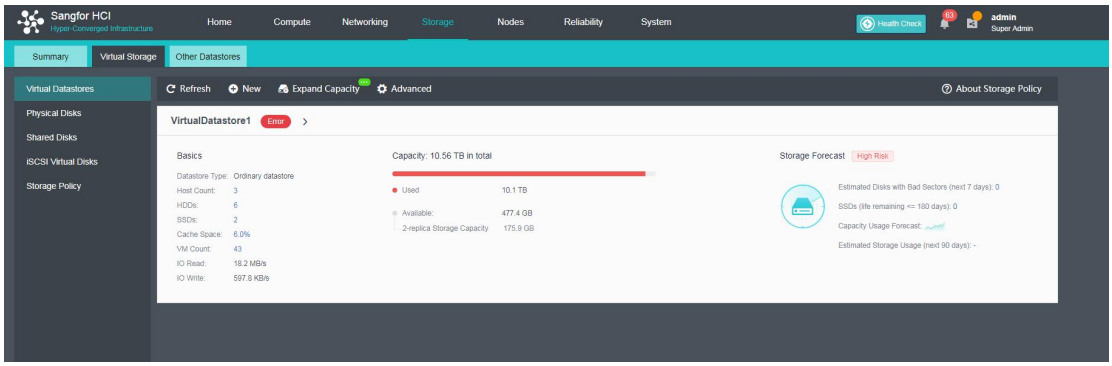
sangfor asan is the asan storage provided by Sangfor. The angfor nfs is the standard NFS CSI provided by us, being able to integrate with general NFS servers and provide NFS CSI support.

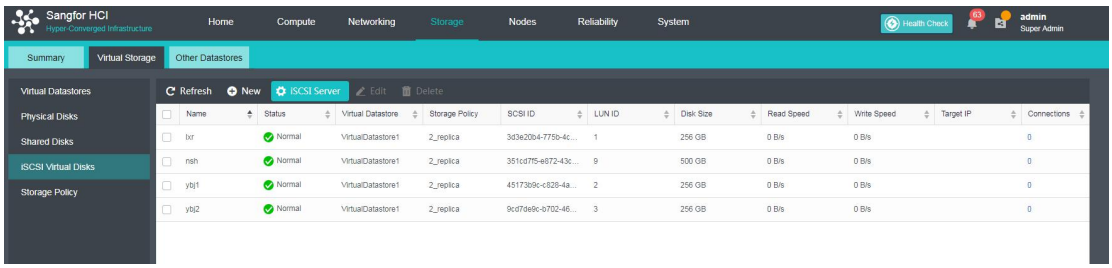
6.2 Creation of Storage Class

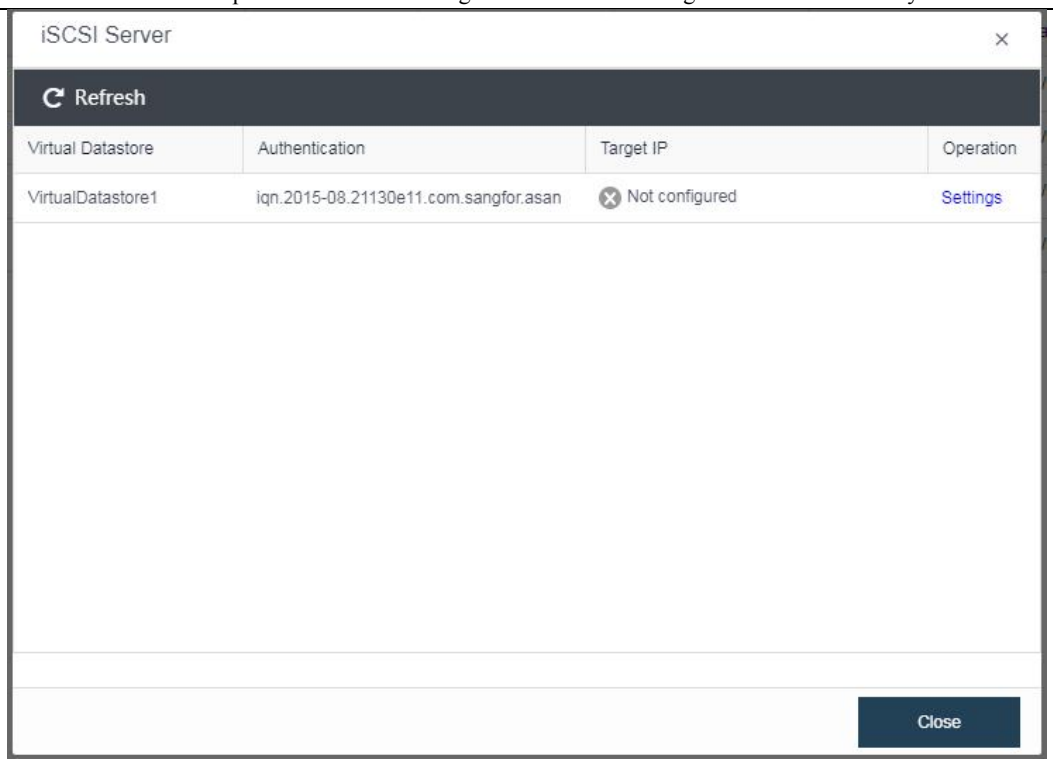
The storage class is the most common way for PaaS platform to use storage. The process of creating a storage class is as follows:

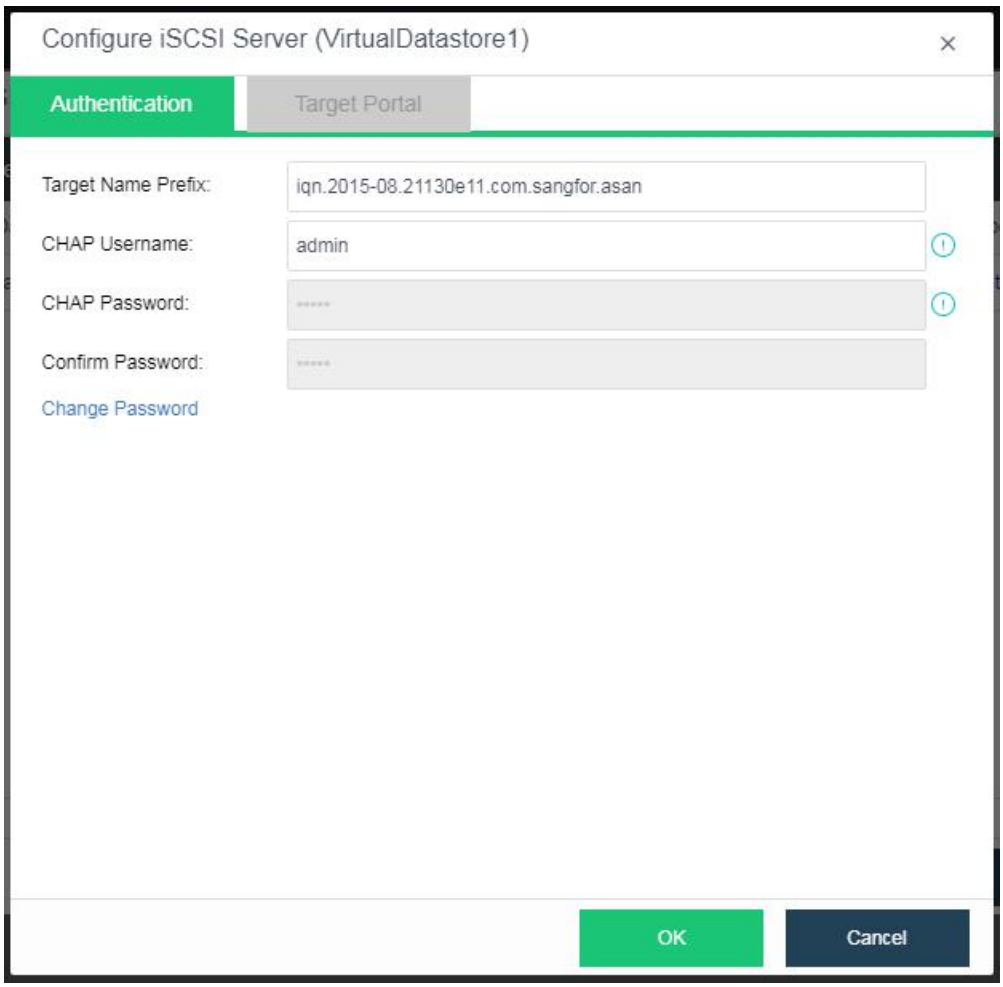
1. Create a storage server, as described above;

2. Go to sangfor HCI, create storage volume, and configure ISCSI server:

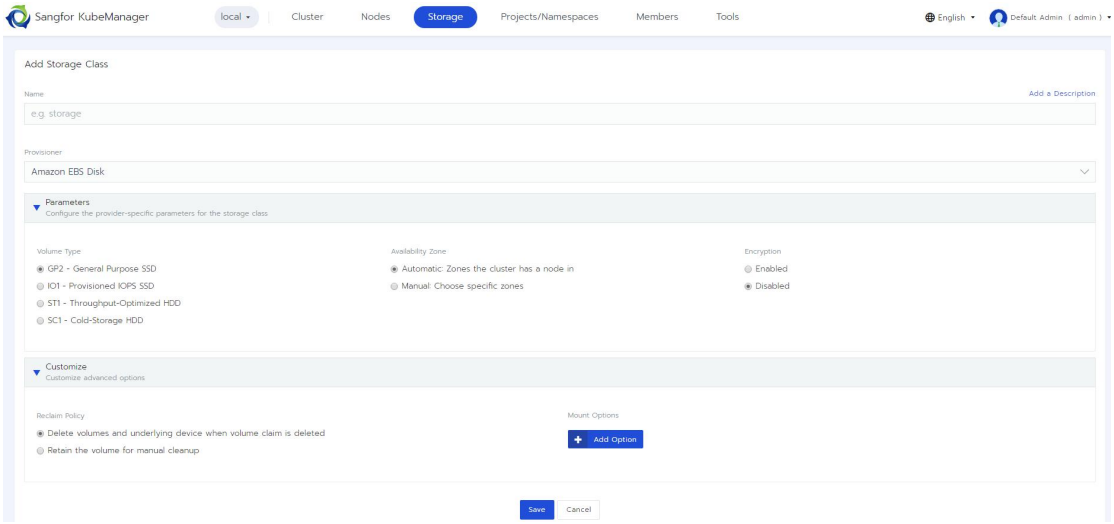





iSCSI Server
Refresh
Virtual Datastore
Authentication
Target IP
Operation
VirtualDatastore1
iqn.2015-08.21130e11.com.sangfor.asan
Not configured
Settings
Close


Configure iSCSI Server (VirtualDatastore1)
Authentication
Target Portal
Target Name Prefix:
iqn.2015-08.21130e11.com.sangfor.asan
CHAP Username:
admin
CHAP Password:
Confirm Password:
Change Password
OK
Cancel

3. Create a storage class based on the created storage server, storage volume and ISCSI server:

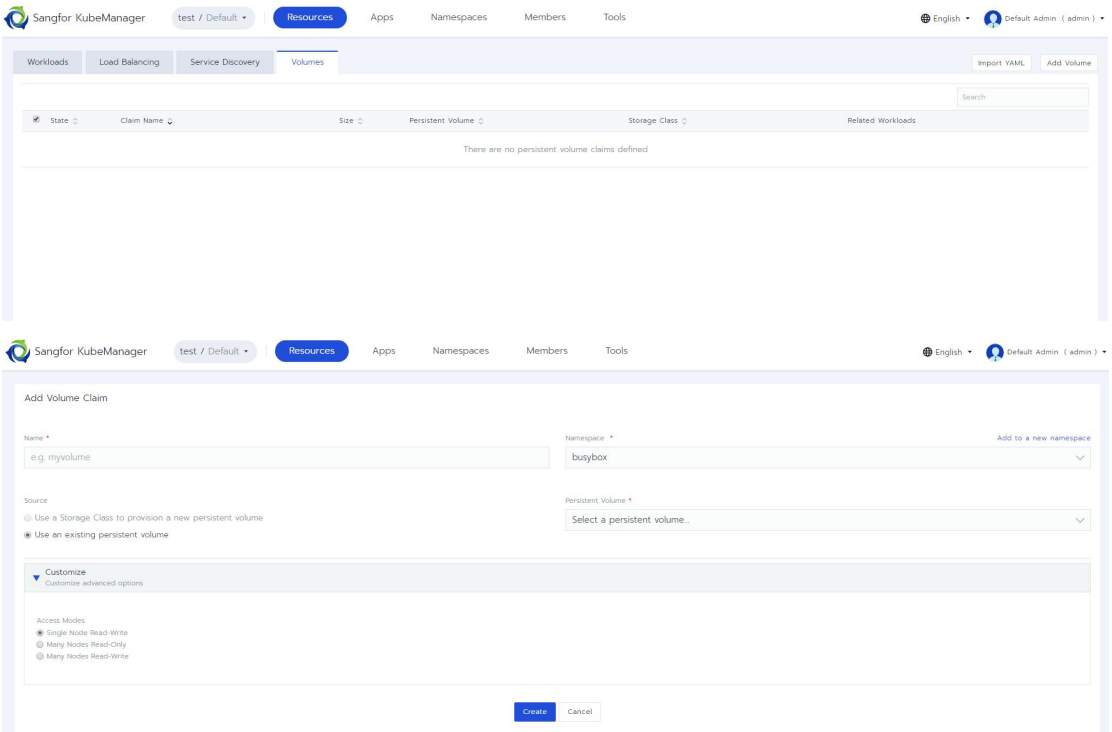


Configure the storage class according to the configuration in HCI.

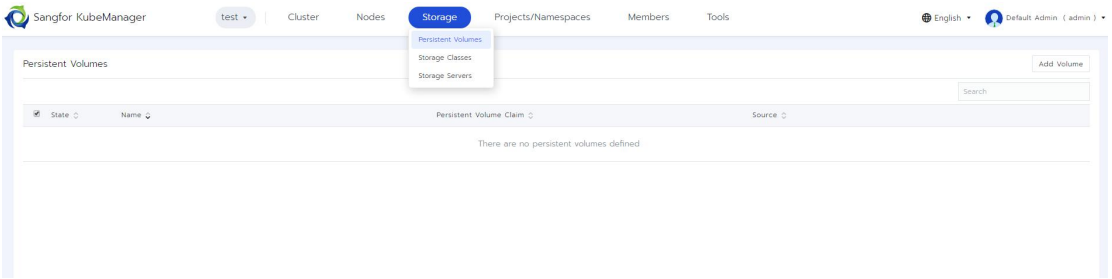
6.3 Creation of PV

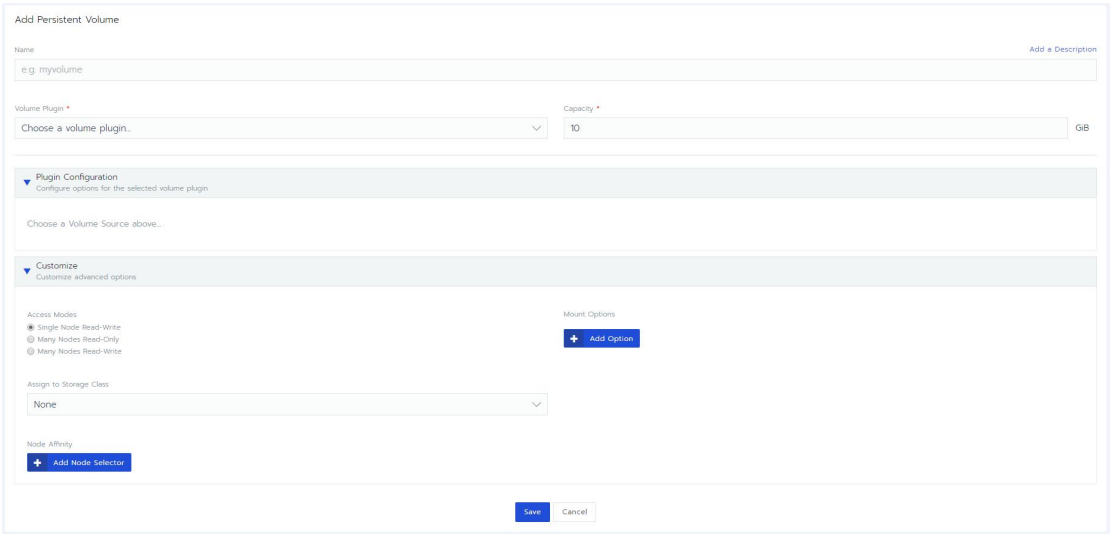
You can create PV through SC or the static volume.

Creation of PV via SC: When SC is referenced by creating **Volumes**, PV will be automatically created when **Volumes** is used. The operating steps are as follows:



Directly create PV with the storage volume, as shown below:



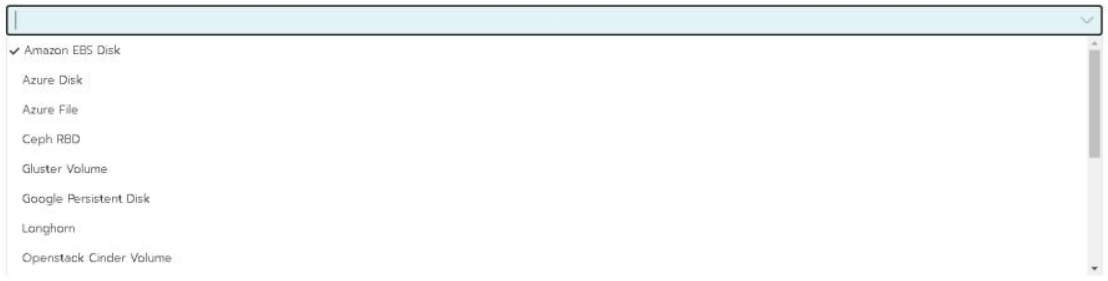


6.4 Support to Other Storage

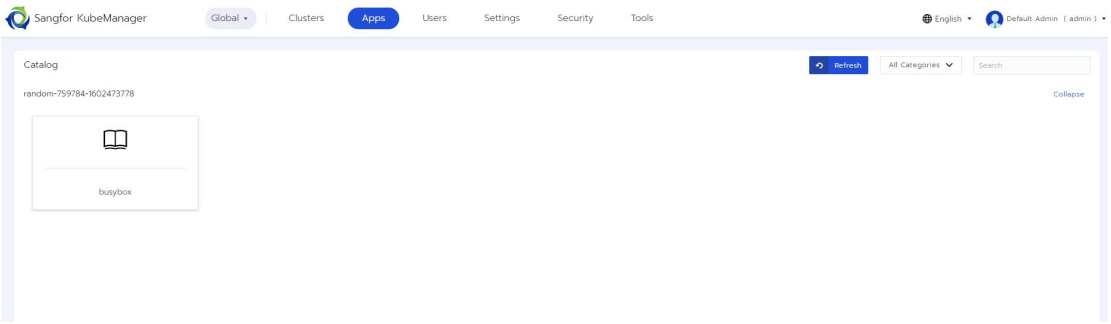
KubeManager allows to create SC and PV via a variety of common storage services, and provides built-in CSI plug-ins.

Commonly used volume plug-ins for creating PV are as follows:

Commonly used provisioners for creating SC are as follows:



Custom is a user-defined provisioner, which can be provided in the app store, for example:



After creating provisoner, you can provide the name of provisioner in a Custom way, create a storage class and then use the specific storage.

7 Multi-cluster Management

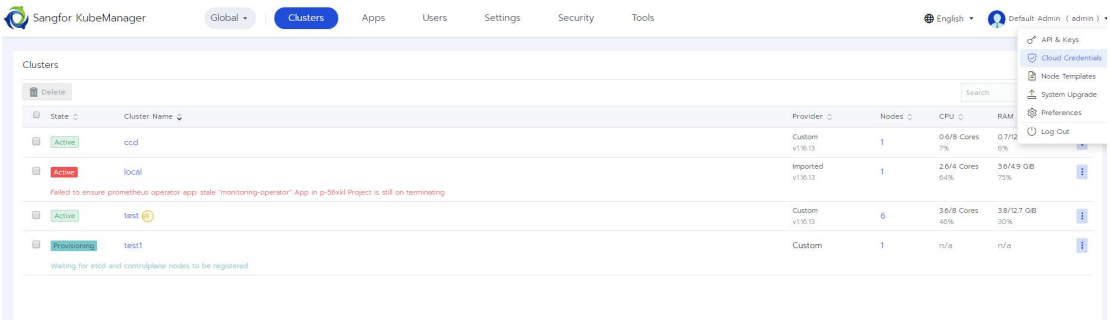
KubeManager provides powerful multi-cluster management capability, which is one of the core functions of KubeManager, including multi-cluster application, creating multi-cluster and supporting various K8S cluster forms.

As the multi-cluster applications have been described in the section of **App Store**, this section mainly describes the other two. KubeManager allows a variety of K8S clusters to provide PaaS layer services for users, including custom clusters, hosts from cloud service providers, and Kubernetes hosting services.

7.1 K8S Clusters on "Hosts from Cloud Service Providers"

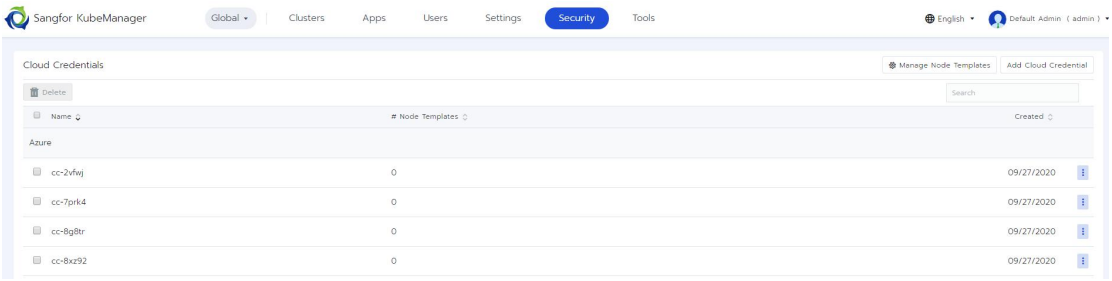
For the hosts provided by cloud service providers, we only need to integrate the drivers provided by cloud service providers through cloud provisioner, and configure the corresponding accounts to KubeManager. In this way, we can seamlessly create and manage the virtual machines on another platform and the Kubernetes on the virtual machines through the KubeManager platform.

Create **Cloud Credentials**:

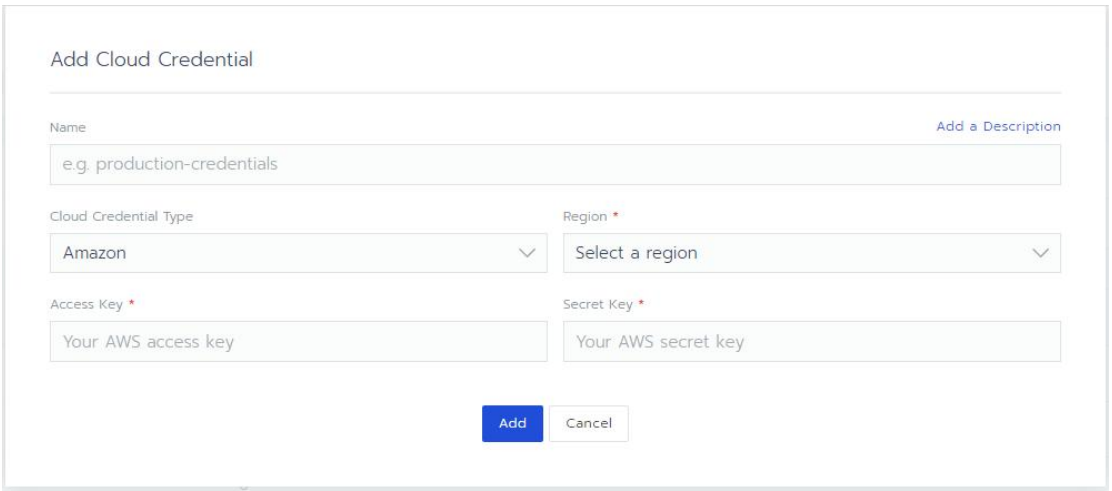


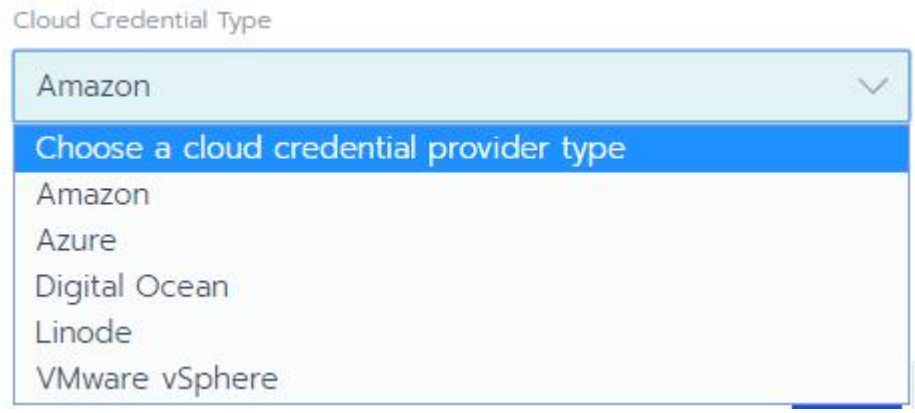
The cloud credential page has two options: **Add Cloud Credential** and **Manage Node Templates**.

The cloud credential is the authentication credential for accessing cloud service providers:

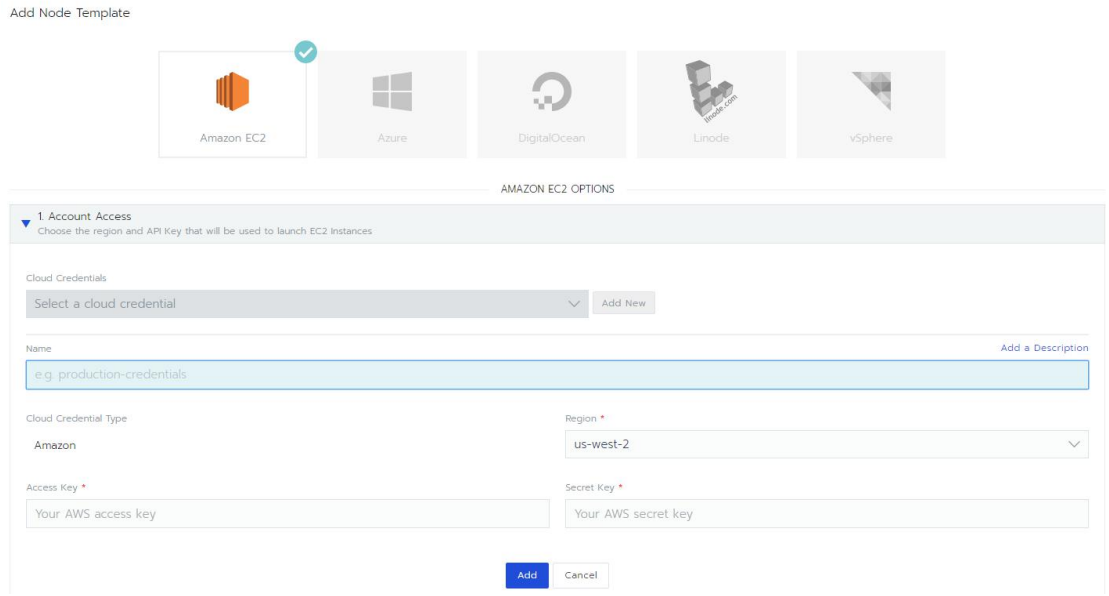


Add Cloud Credential:



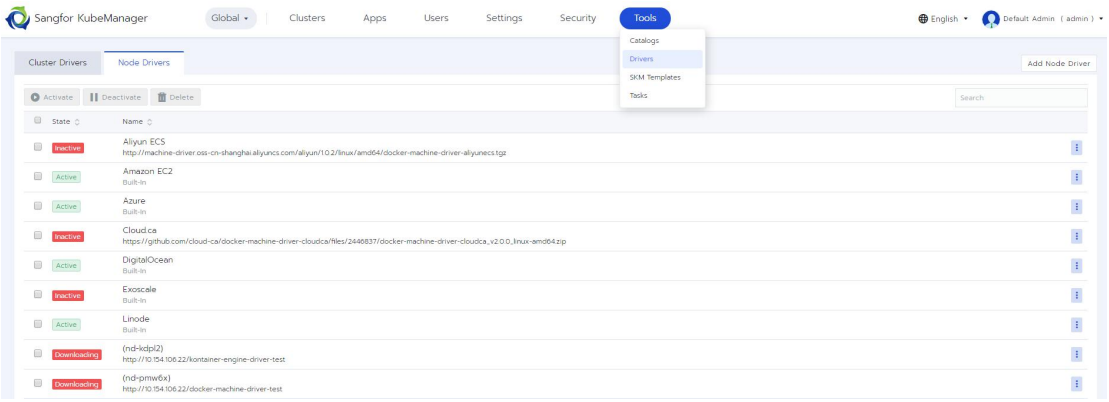


Manage Node Templates, which is used for creating a host at a cloud service provider, including such information as region and model:

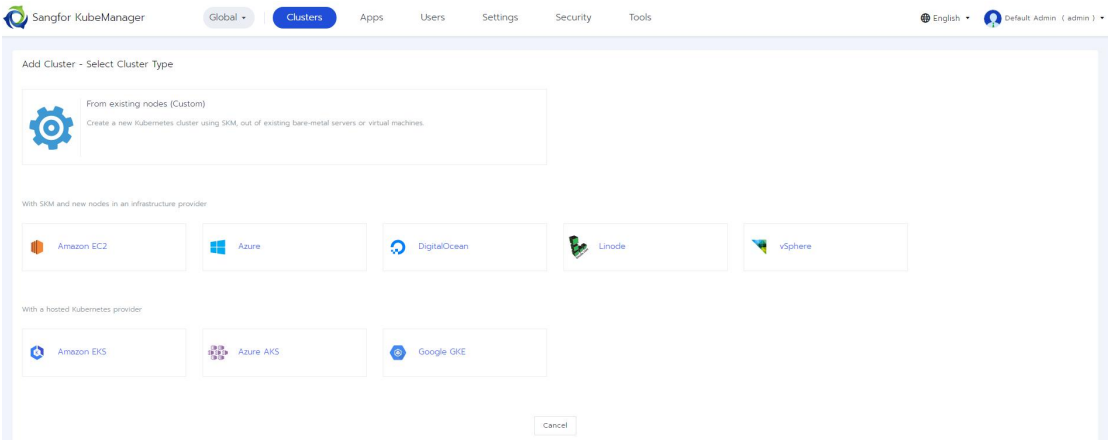


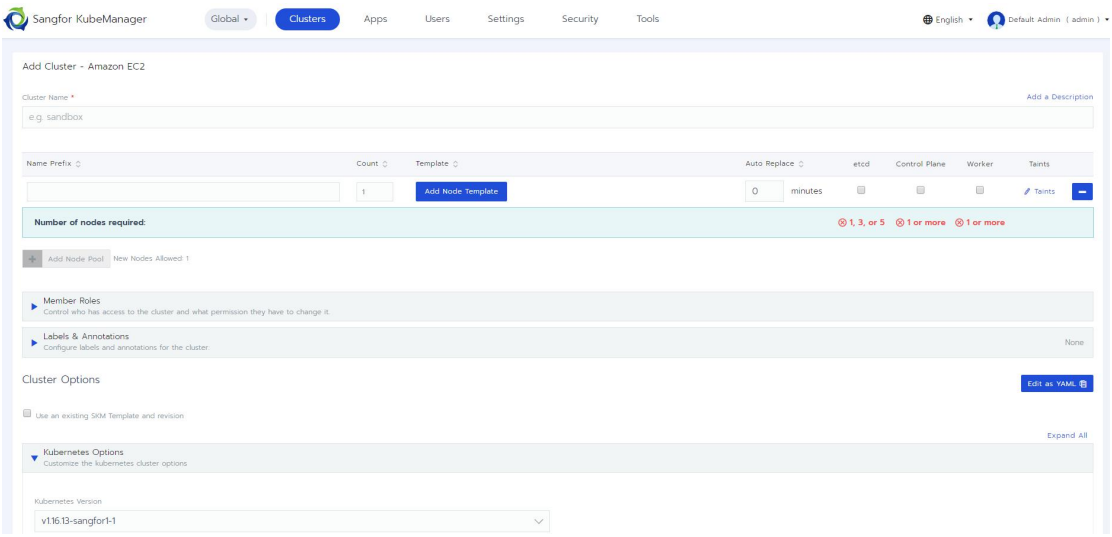
After configuring host credentials and cluster templates, we can create clusters of service providers and K8S services.

Add Node Drivers:



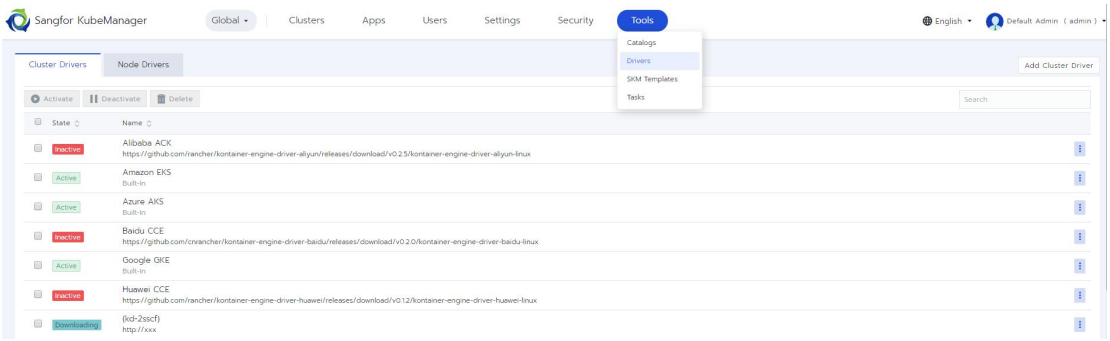
Create a cluster of service provider's virtual machine:



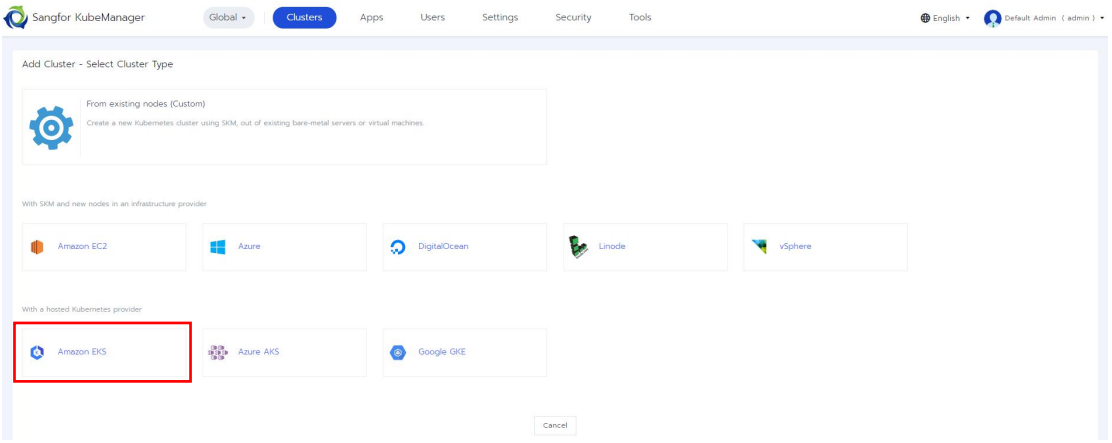


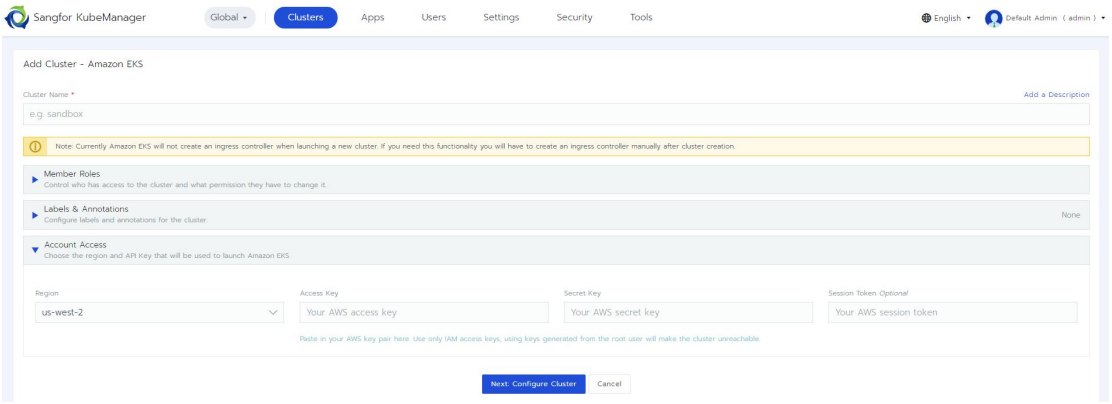
7.2 Cluster of "Kubernetes Hosting Service Providers"

Add Cluster Driver:



Create a cluster of K8S services:



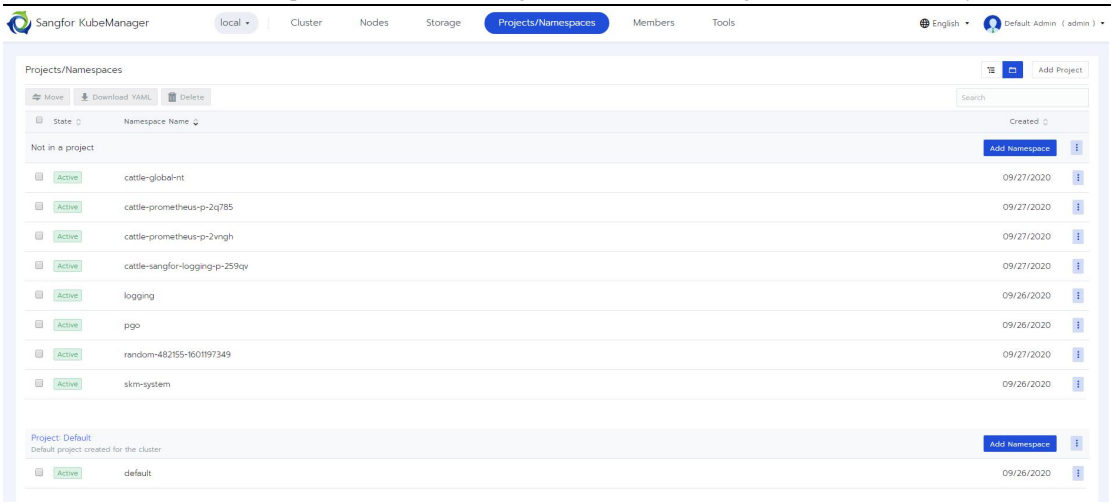


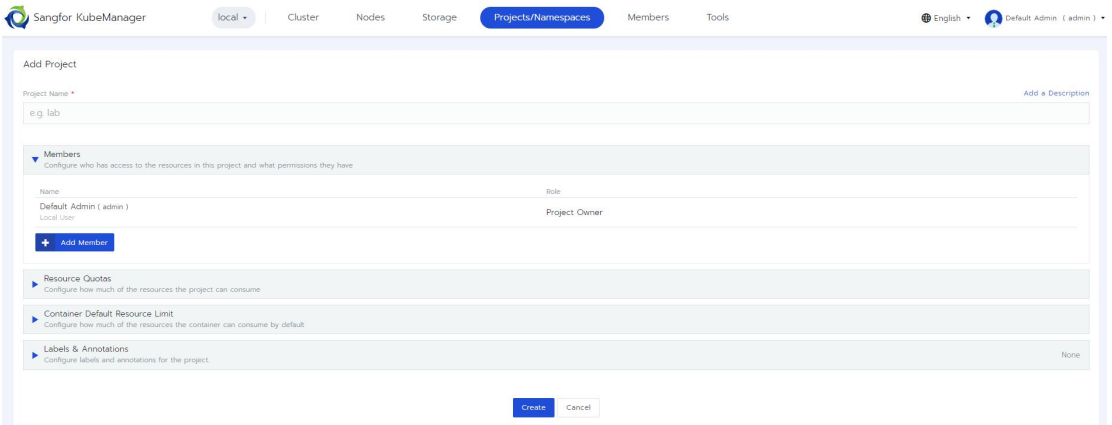
8 Project Configuration

For a project, the smallest system of the permission system, you can create or delete a project, grant permissions to a project, or restrict the use resources and quotas of a project.

8.1 Creation of Project

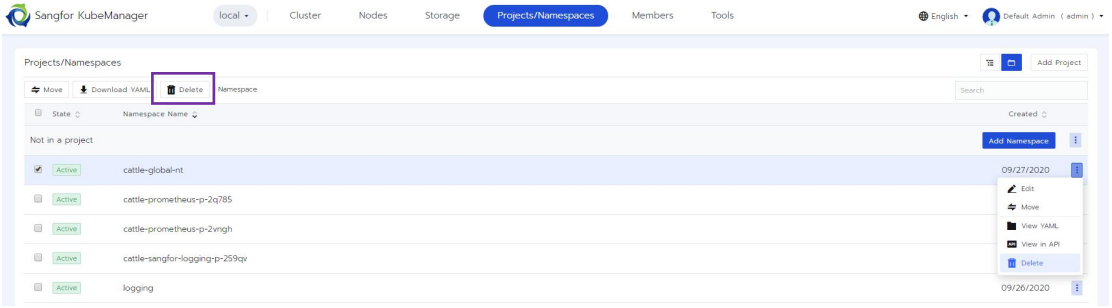
Select the corresponding cluster, and click **Projects/Namespaces**, to go to the project list page. Then, you can see all the projects and their namespaces. Click **Add Project** to create new project:

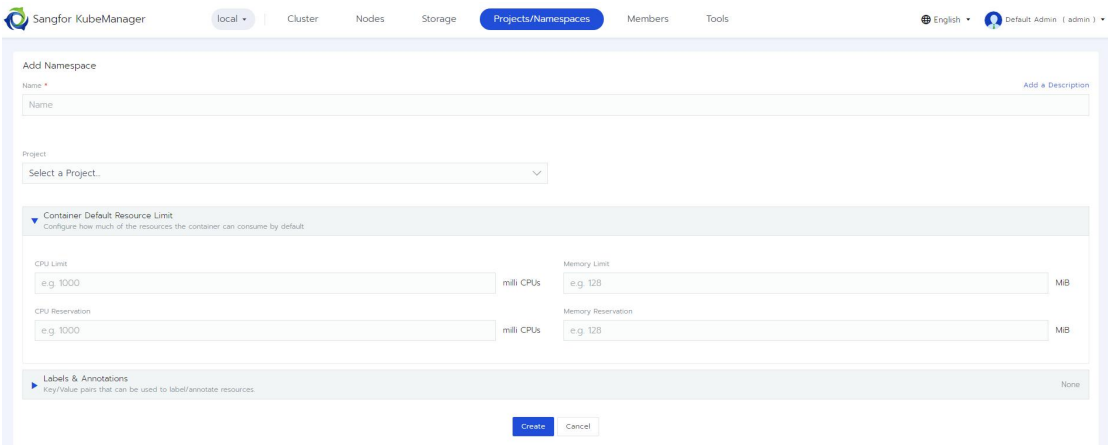




8.2 Namespace Management

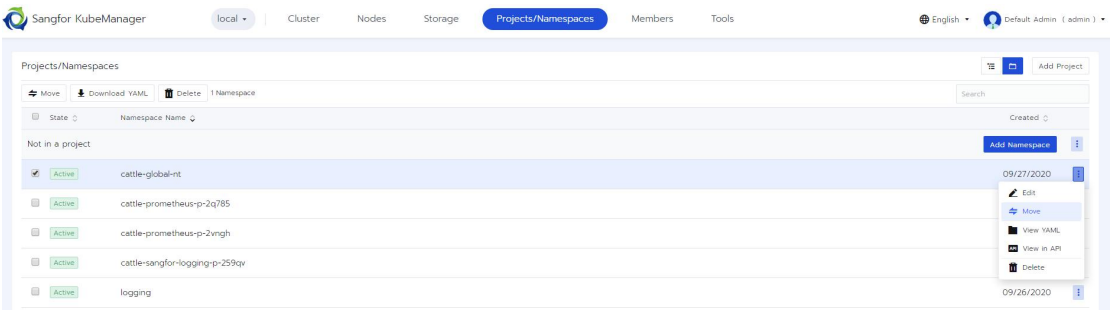
On the project list page of the cluster, you can create and delete a namespace.

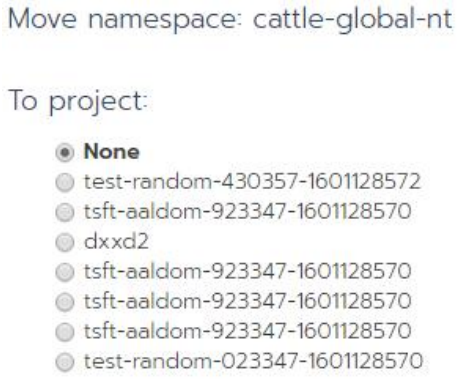




You can limit a namespace by quota.

Similarly, you can also manage namespaces in a project. For non-system default namespaces, you can move them from one project to another:



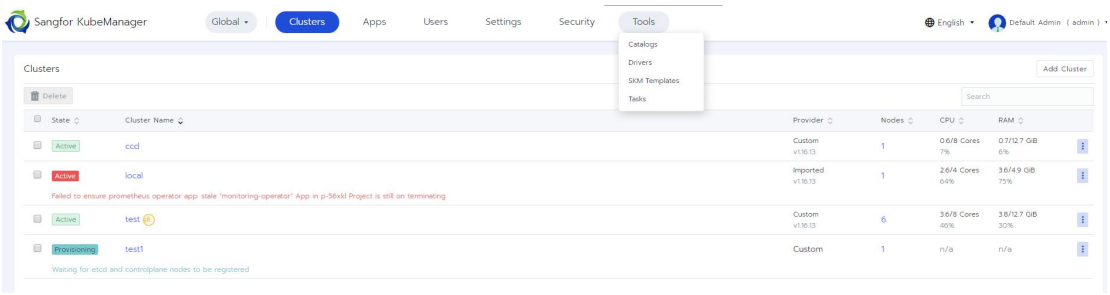


9 System Tools

Most of the auxiliary functions of the platform are set in **Tools**, through which you can set the corresponding functions at **Global**, **Cluster** and **Project** levels.

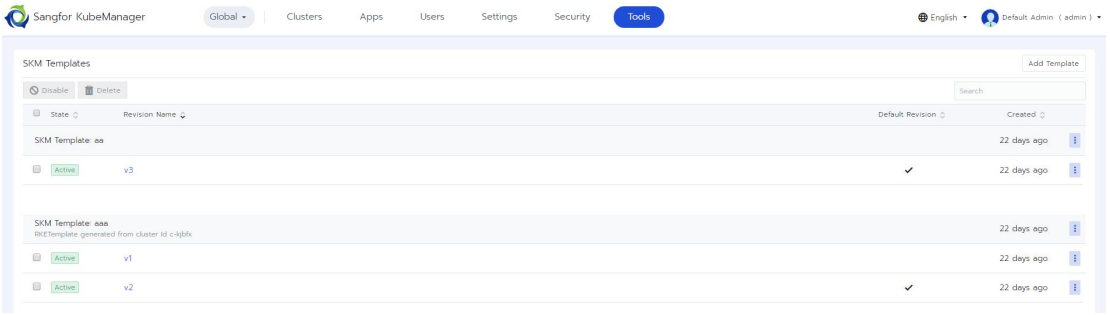
9.1 Global Settings

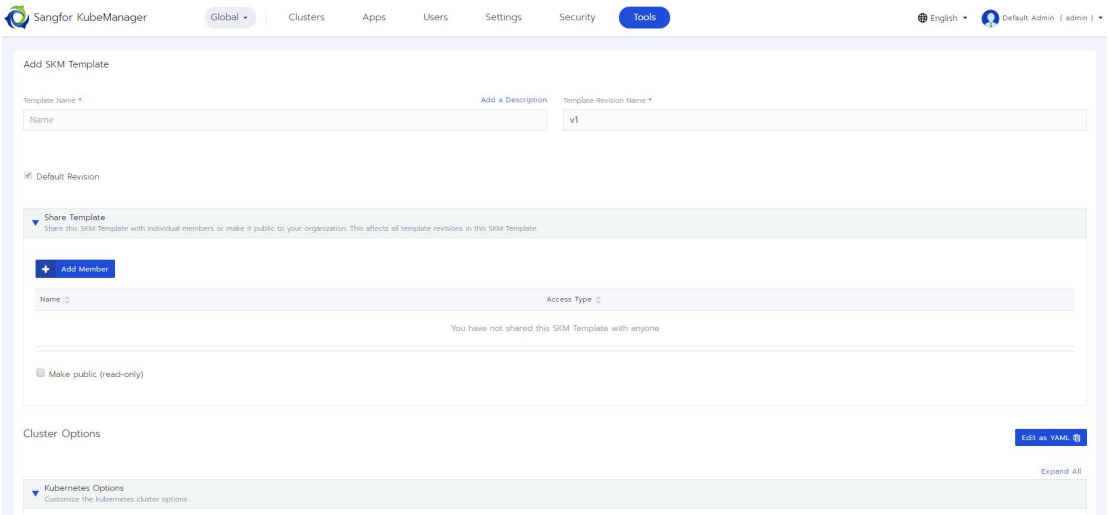
In the **Global** mode, there are four options, **Store settings**, **Driver management**, **SKM cluster template**, and **Operation audit**. Among them, Store settings and Driver management have been introduced before. We mainly introduce **SKM cluster management** and **Operation audit** in this section.

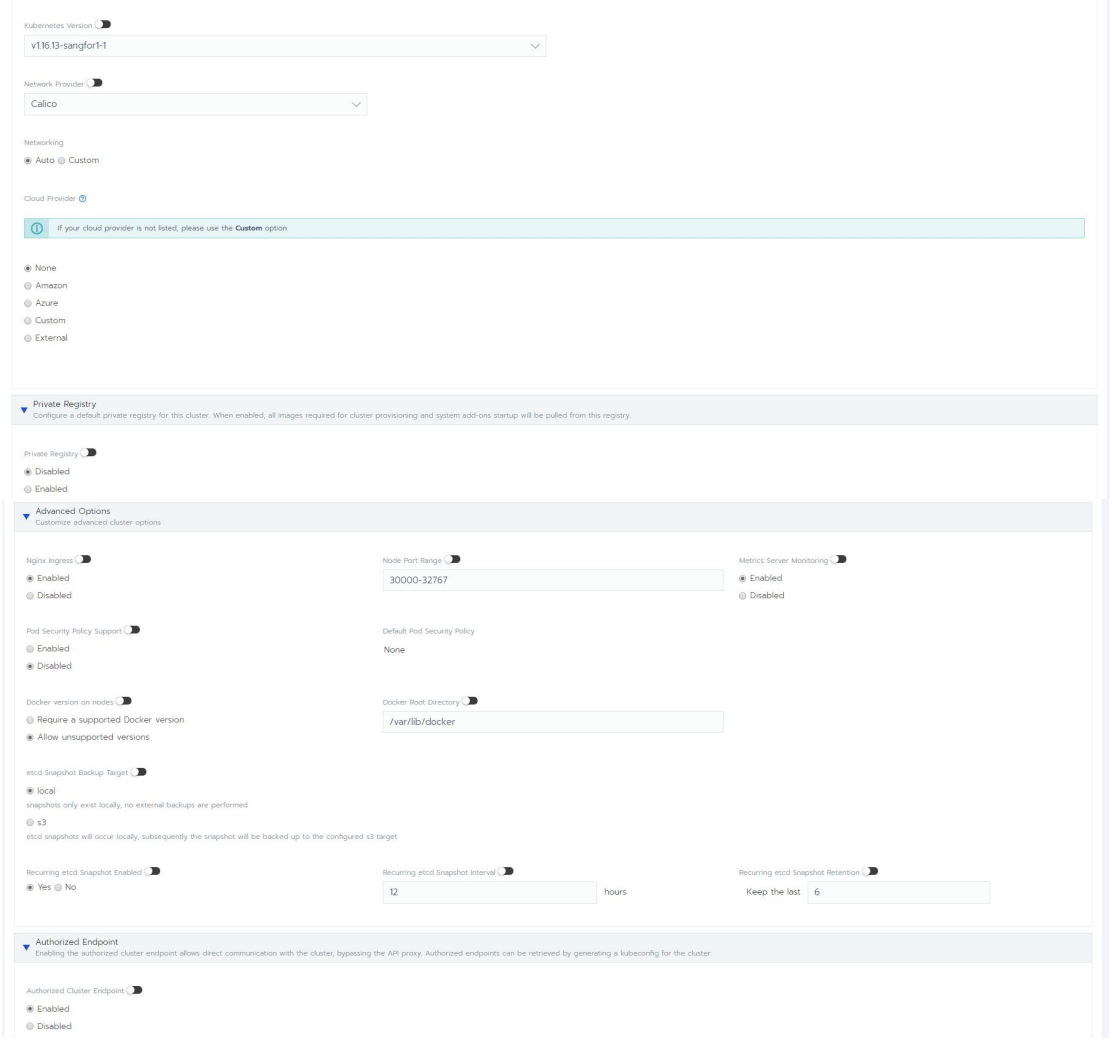


The cluster template is used to set the cluster by applying the previous configuration in this template, and quickly create a cluster.

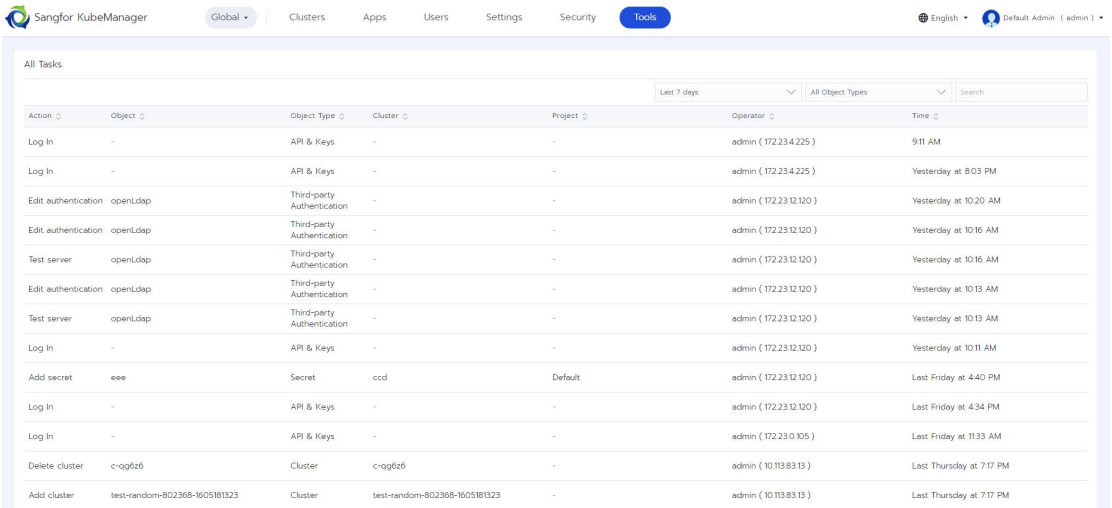
SKM cluster list page:





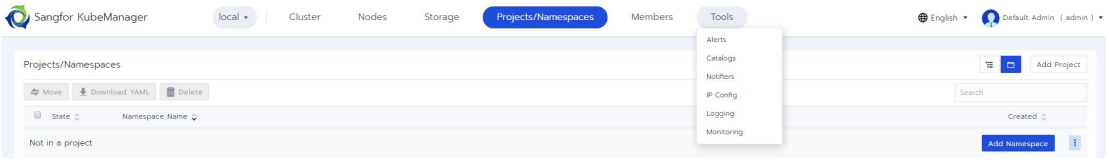


Operation audit: You can trace all the operation history based on the operation records of all users of this platform. Operation audit supports searching and sorting.



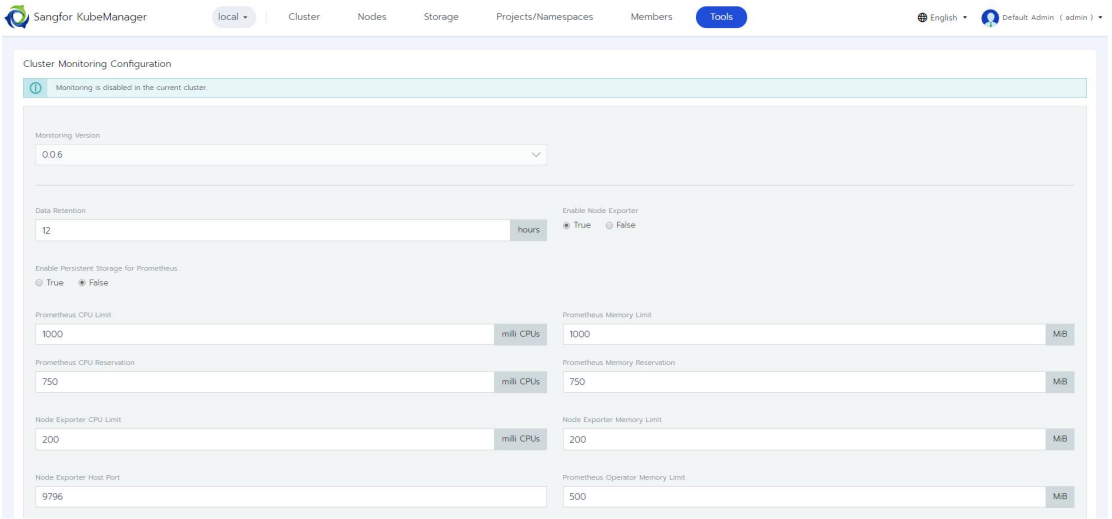
9.2 Cluster Settings

The **Tools** menu in the **Cluster** page mainly includes **Alarm**, **Notification**, **Monitoring**, **Log**, **Store settings**, **Network exit**, etc.



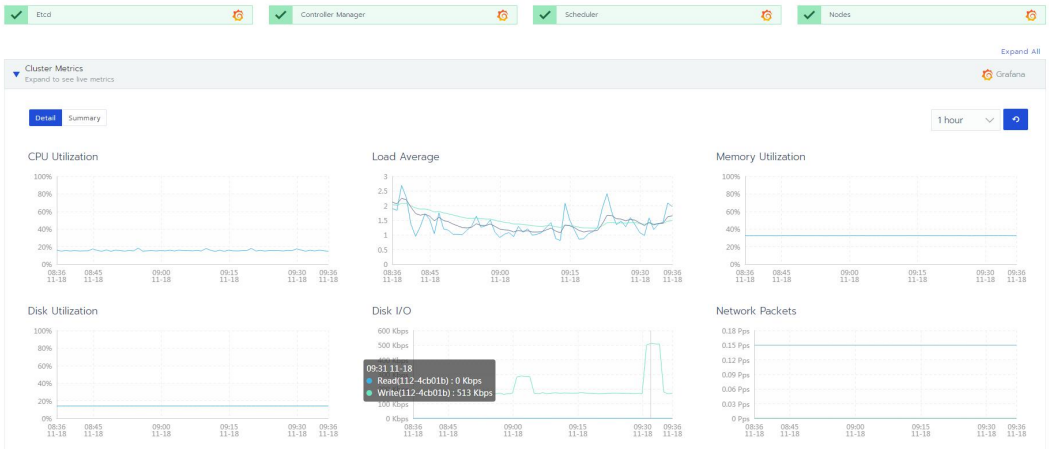
Store settings has been described above, so it will not be further described.

Cluster monitoring:



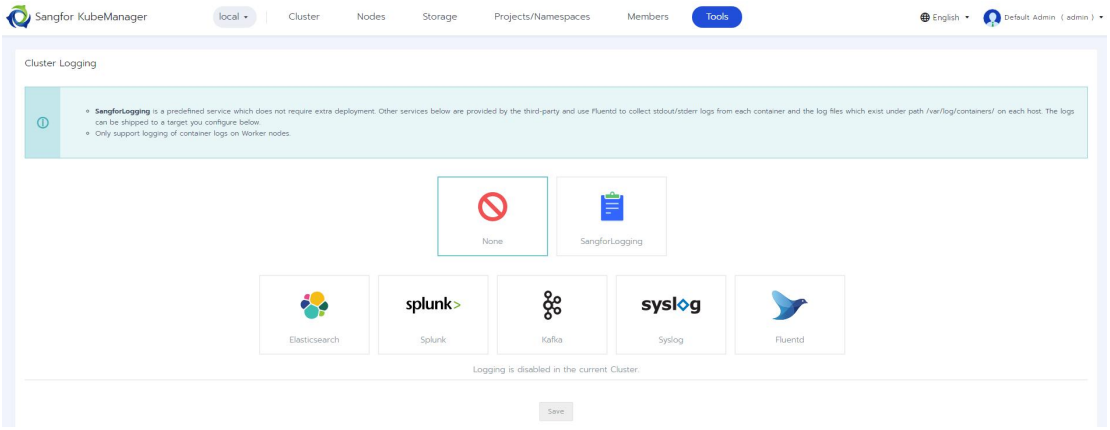
Cluster monitoring allows to use persistent storage and local storage, add node selectors, set tolerance, and support the persistent storage for grafana. Once Cluster monitoring is enabled, you can monitor resources in the cluster.

The cluster page after Cluster monitoring is enabled:

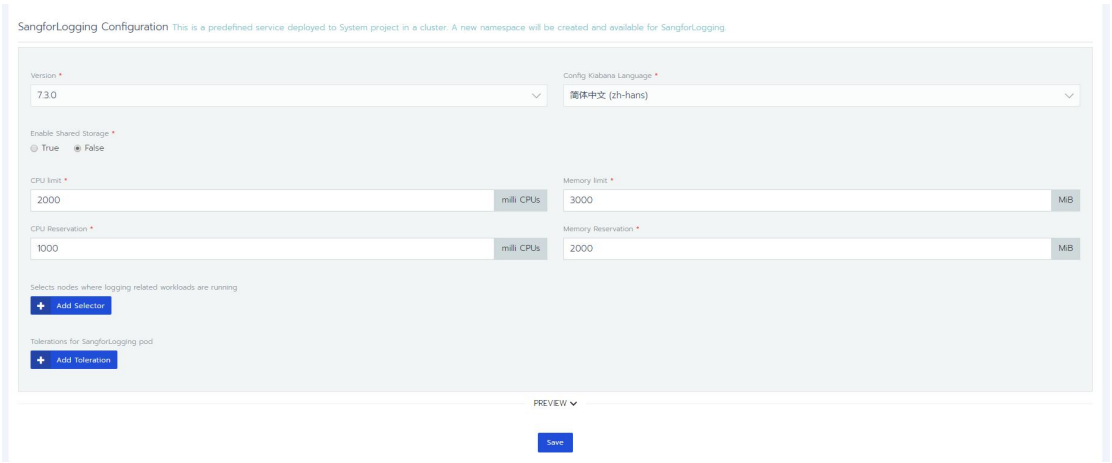


Just like monitoring, you can collect logs through Sangfor Logging or other log collection

methods:

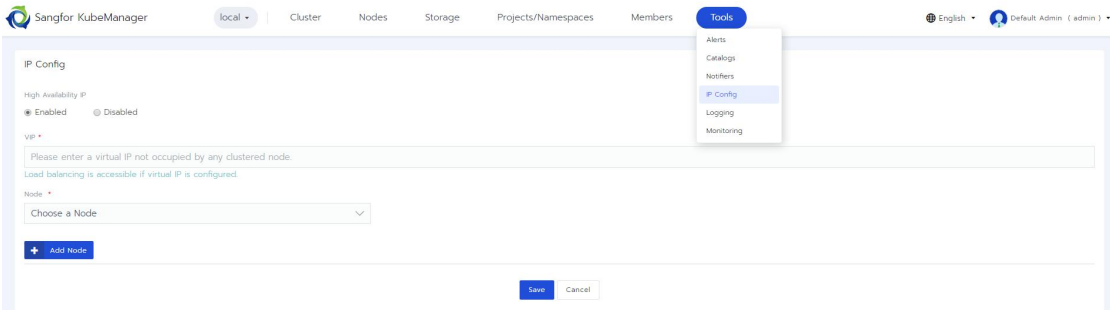


Sangfor Logging allows to configure the persistent storage and local storage:

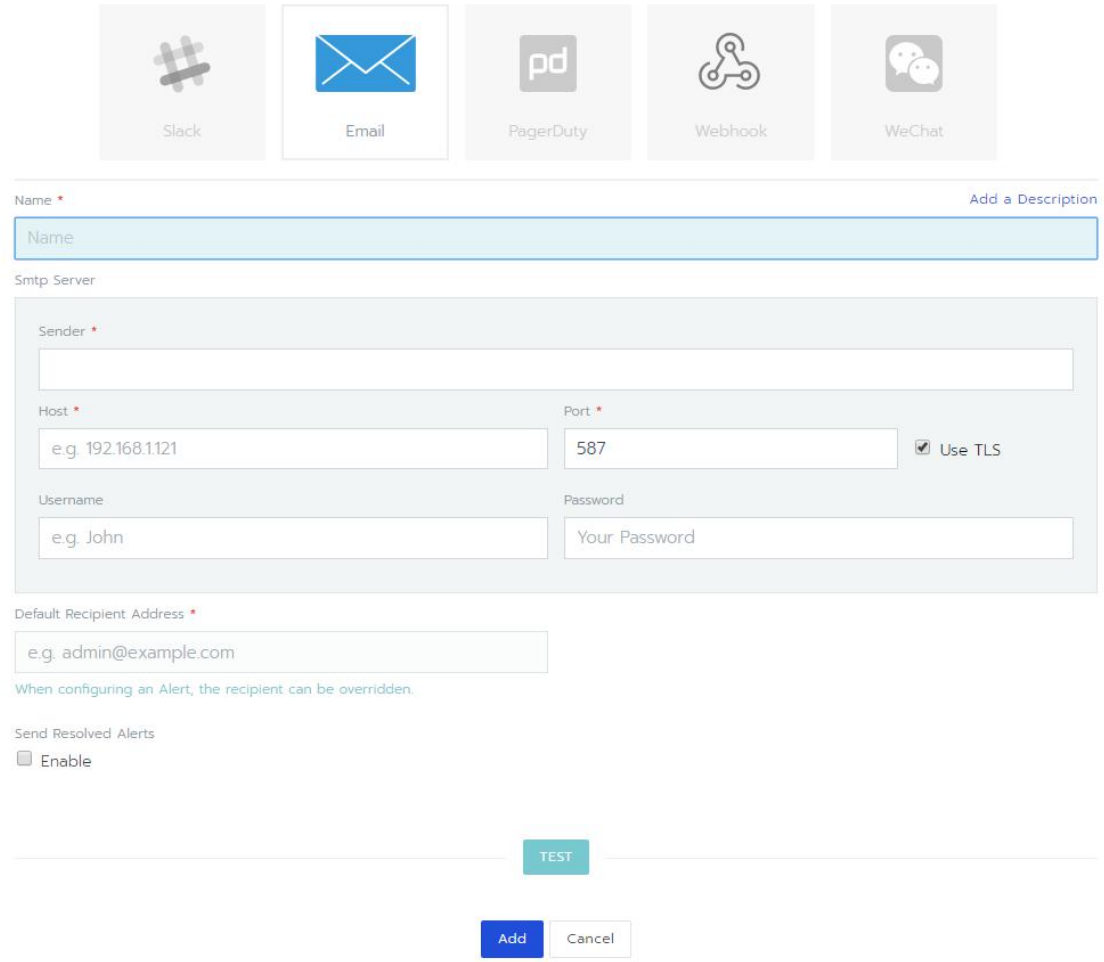


If local storage is selected, 3 duplicates will be created, and high availability will be guaranteed through ES.

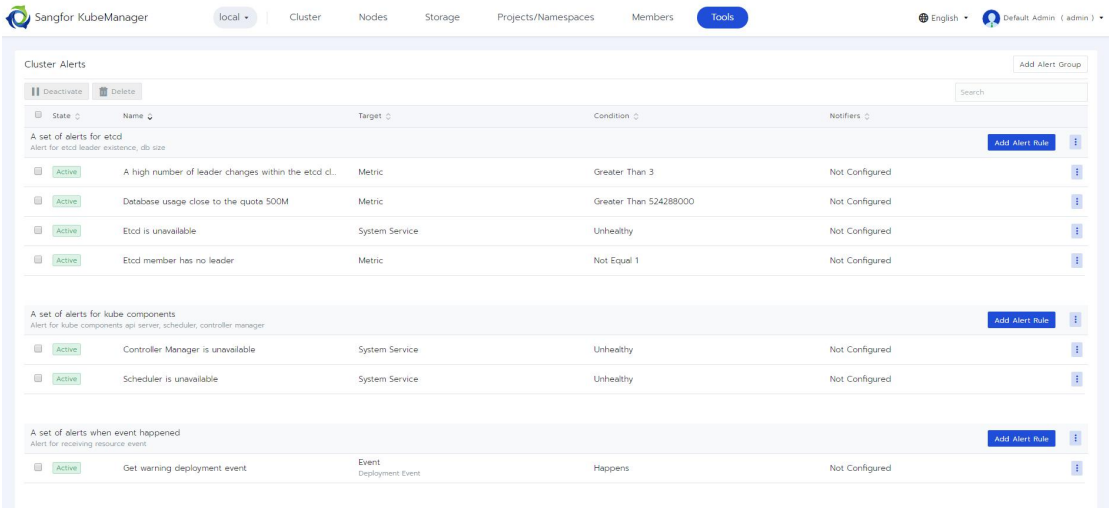
The network exit aims to ensure the high availability of service release. Two nodes are selected to ensure the high availability of the network in the form of vip, and guarantee that the IP for service release can ensure the high availability at different nodes:



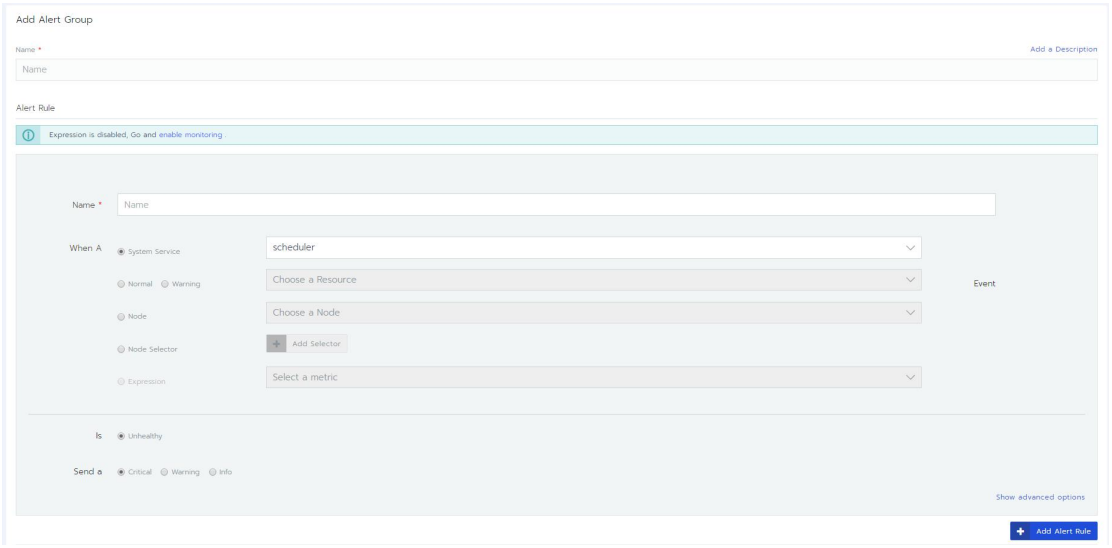
The platform supports the alarm via Slack, E-mail, PagerDuty, Webhook, Enterprise WeChat and other means:



The alarm function aims to alarm the notification group in the notification according to the set rules:



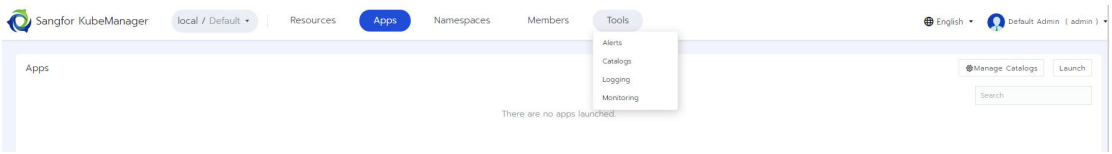
Add Alert Group:



The alarm group mainly consists of alarm rules and notification objects.

9.3 Project Settings

The Tools menu on the project page is the same as that of the cluster page. The only difference is that it applies to the project only. So it will not be further described here.

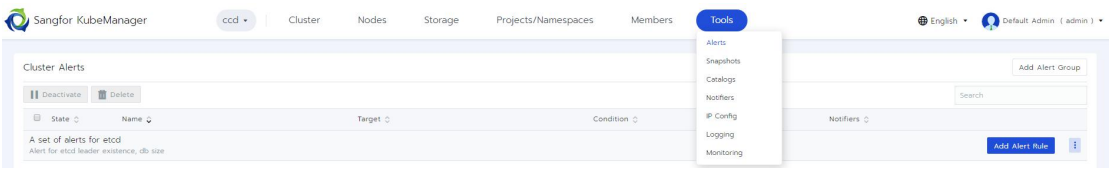


10 Other Configurations

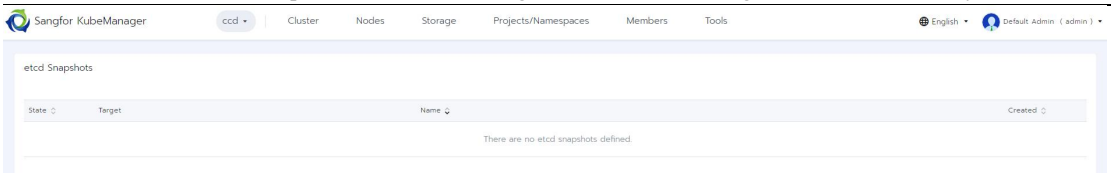
KubeManager also supports some other management functions.

10.1 Backup and Recovery

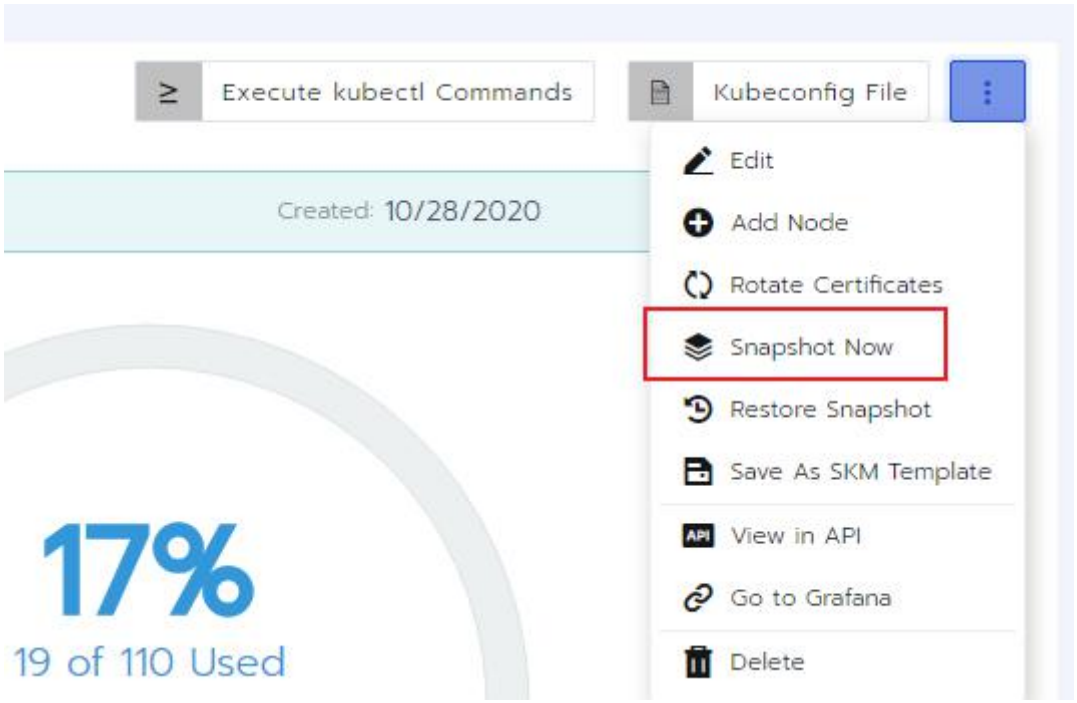
There are two backup methods for cluster. One is local backup, and the other is to backup to S3 storage.



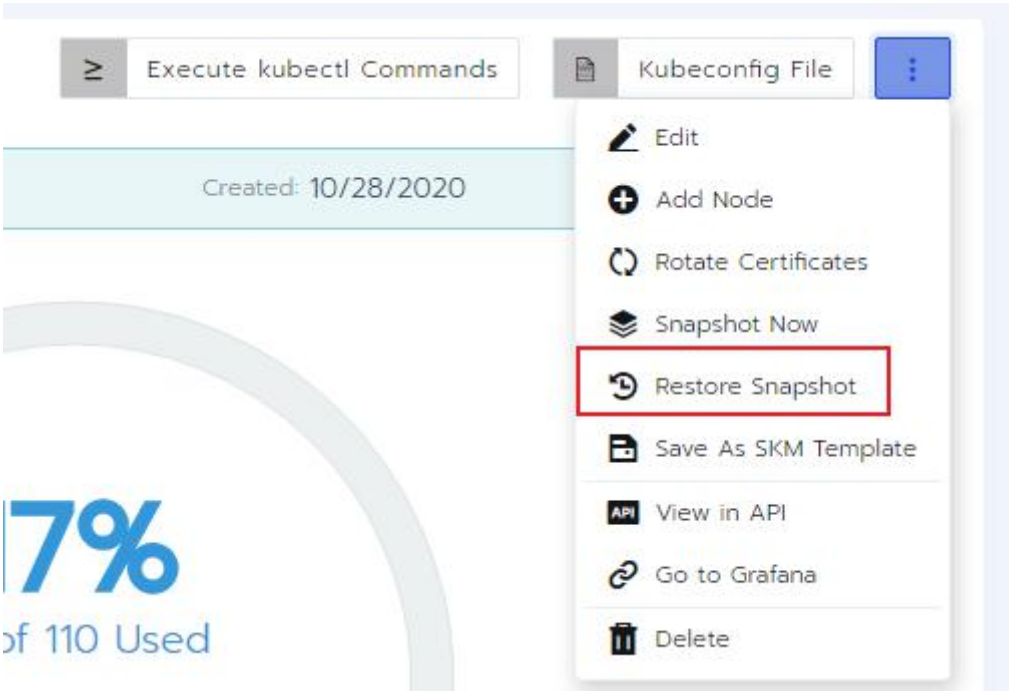
Set the backup mode and policy in the cluster configuration:

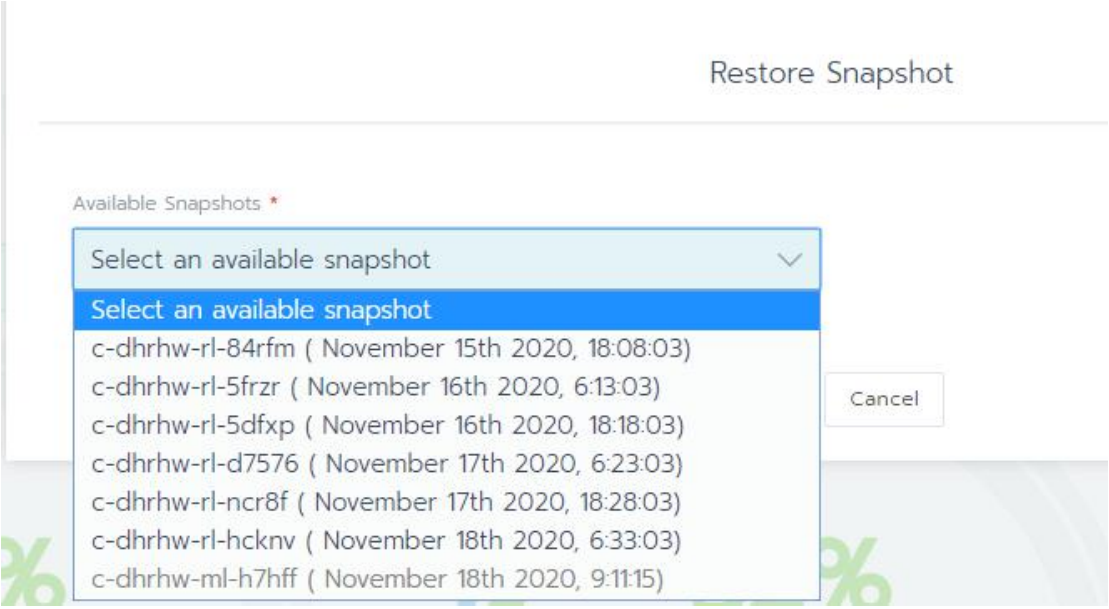


Make a manual backup on the cluster page:



Recover it on the cluster page:

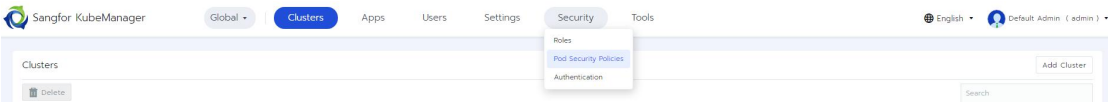




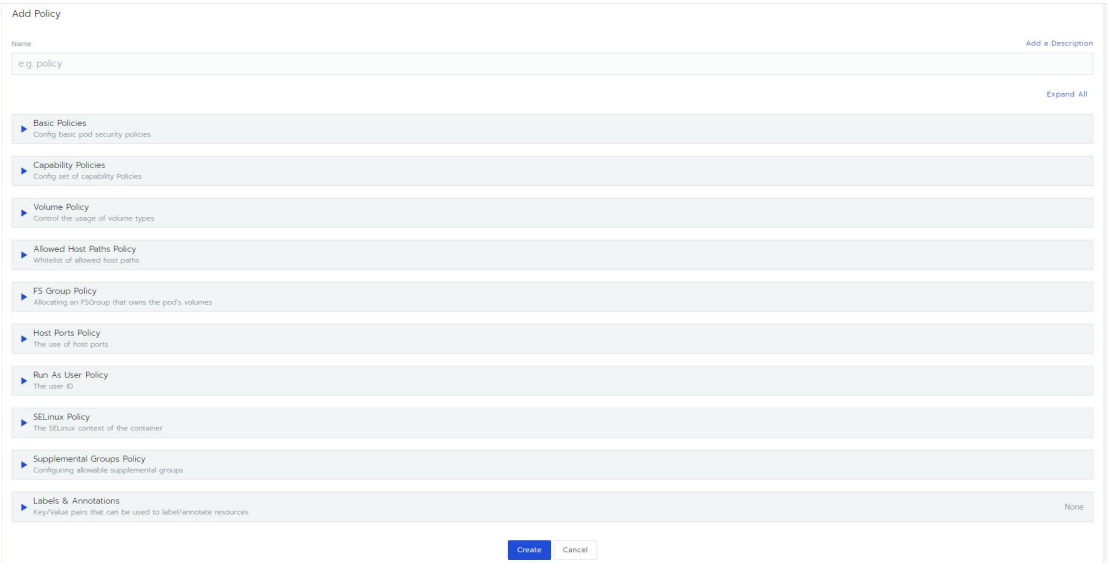
10.2 Security

There are two main types of KubeManager security, namely the image security that has been described in the registry part and the container security. Container security is mainly guaranteed by PSP security policy.

Security policy list:



Add security policy:



When creating a cluster, you can enable the security policy and select the security policy to be

enabled:

10.3 Precautions

KubeManager provides a sound multi-cluster management function and a user-friendly interface. During use, please ensure the network access and sufficient system resources, and guarantee that the registry could be connected to the node network.