Breeze - Deploy a Production Ready Kubernetes Cluster with graphical interface

Project Breeze is an open source trusted solution allow you to create Kubernetes clusters on your internal, secure, cloud network with graphical user interface. （ https://github.com/wise2c-devops/breeze ）

Features

*** Easy to run**: Breeze combines all resources you need such as kubernetes components images, ansible playbooks for the deployment of kubernetes clusters into a single docker image (wise2c/playbook). It also works as a local yum repository server. You just need a linux server with docker and docker-compose installed to run Breeze.

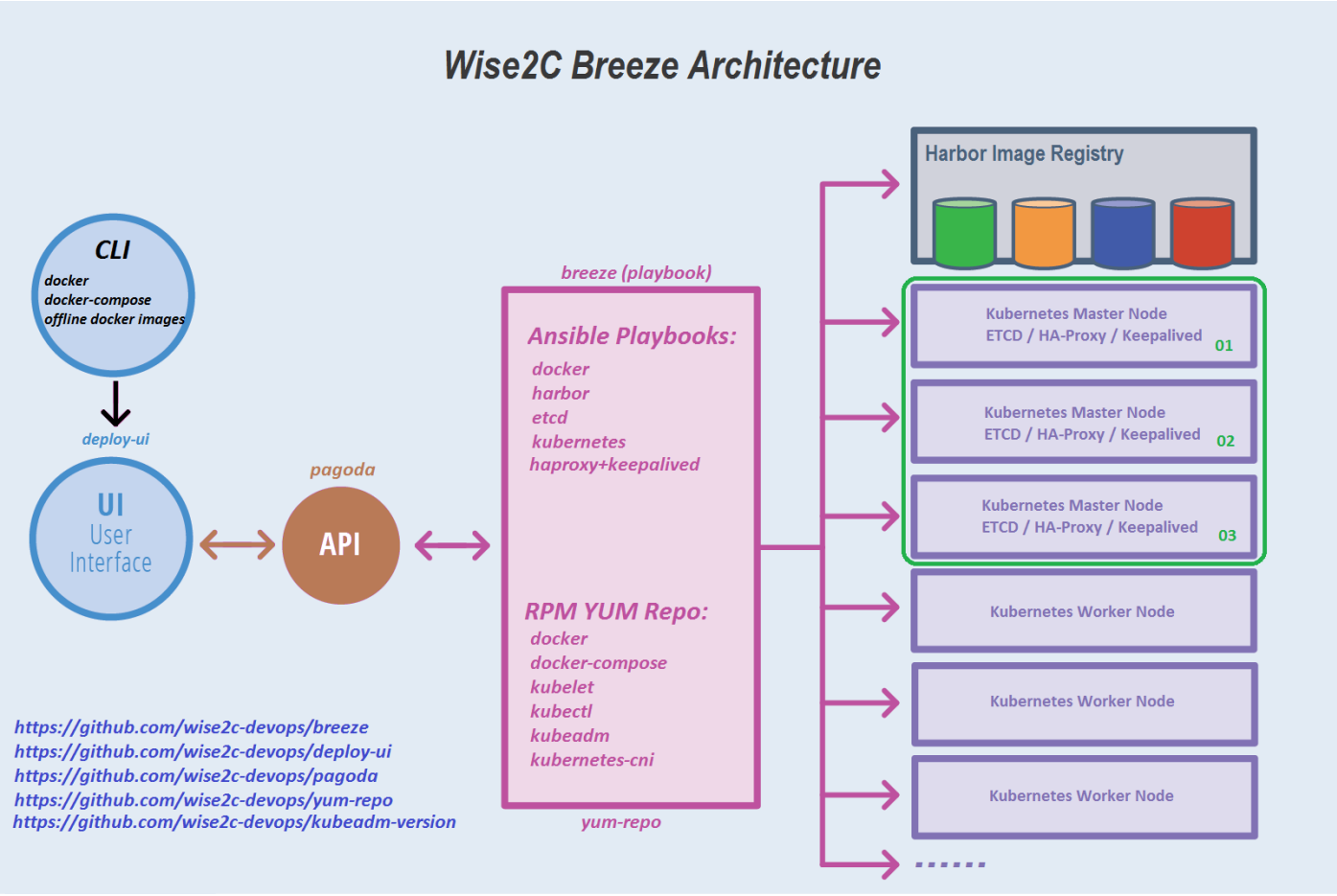
*** Simplified the process of kubernetes clusters deployment**: With a few simple commands, you can get Breeze running, and then finish all the other deployment processes by the graphical interface.

*** Support offline deployment**: After 4 images (playbook, yum-repo, pagoda, deploy-ui) have been loaded on the deploy server, kubernetes clusters can be setup without internet access. Breeze works as a yum repository server and deploys a local Harbor registry and uses kubeadm to setup kubernetes clusters. All docker images will be pulled from the local Harbor registry.

*** Support multi-cluster**: Breeze supports multiple kubernetes clusters deployment.

*** Support high available architecture**: With Breeze, you can setup kubernetes clusters with 3 master servers and 3 etcd servers combined with haproxy and keepalived. All worker nodes will use the virtual floating ip address to communicate with the master servers.

Architecture



You just need a linux server with docker and docker-compose installed to run Breeze.

For offline deployment, just download those 4 images listed in the file docker-compose.yml.

Below is the server list in our test environment:

Hostname	IP Address	Role	OS	Components
deploy	192.168.9.10	Breeze Deploy	CentOS 7.5 x64	docker / docker-compose / Breeze
k8s01	192.168.9.11	K8S Master	CentOS 7.5 x64	K8S Master / etcd / HAProxy / Keepalived
k8s02	192.168.9.12	K8S Master	CentOS 7.5 x64	K8S Master / etcd / HAProxy / Keepalived
k8s03	192.168.9.13	K8S Master	CentOS 7.5 x64	K8S Master / etcd / HAProxy / Keepalived
k8s04	192.168.9.14	K8S Minion Node	CentOS 7.5 x64	K8S Worker
registry	192.168.9.20	Harbor	CentOS 7.5 x64	Harbor 1.6.2
	192.168.9.30	VIP		HA virtual IP address

Steps:

1. Prepare the deploy server（deploy / 192.168.9.10）

(1) Install CentOS 7.5 with Minimal mode and execute commands as below:

```
setenforce 0
sed --follow-symlinks -i "s/SELINUX=enforcing/SELINUX=disabled/g" /etc/selinux/config
firewall-cmd --set-default-zone=trusted
firewall-cmd --complete-reload
```

(2) Install docker-compose

```
curl -L https://github.com/docker/compose/releases/download/1.21.2/docker-compose-$(uname -s)-$(uname -m) -o /usr/local/bin/docker-compose

chmod +x /usr/local/bin/docker-compose
```

(3) Install docker

```
yum install docker
```

(4) ssh login to other servers without password

a) ssh keygen:

```
ssh-keygen
```

b) execute the ssh-copy-id command:

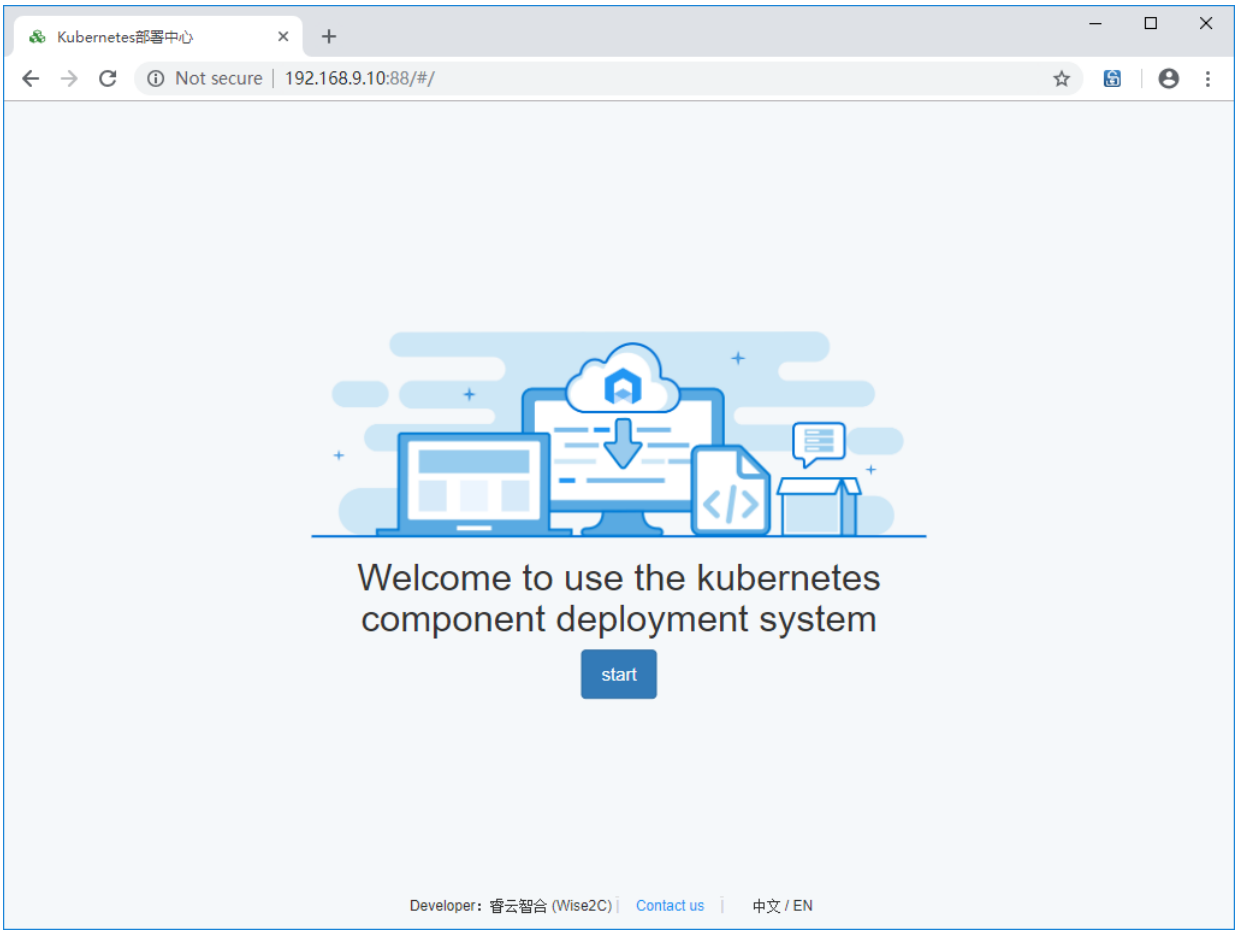
```
ssh-copy-id 192.168.9.11
ssh-copy-id 192.168.9.12
ssh-copy-id 192.168.9.13
ssh-copy-id 192.168.9.14
ssh-copy-id 192.168.9.20
```

2. Get the compose file (e.g. for Kubernetes v1.12.3)

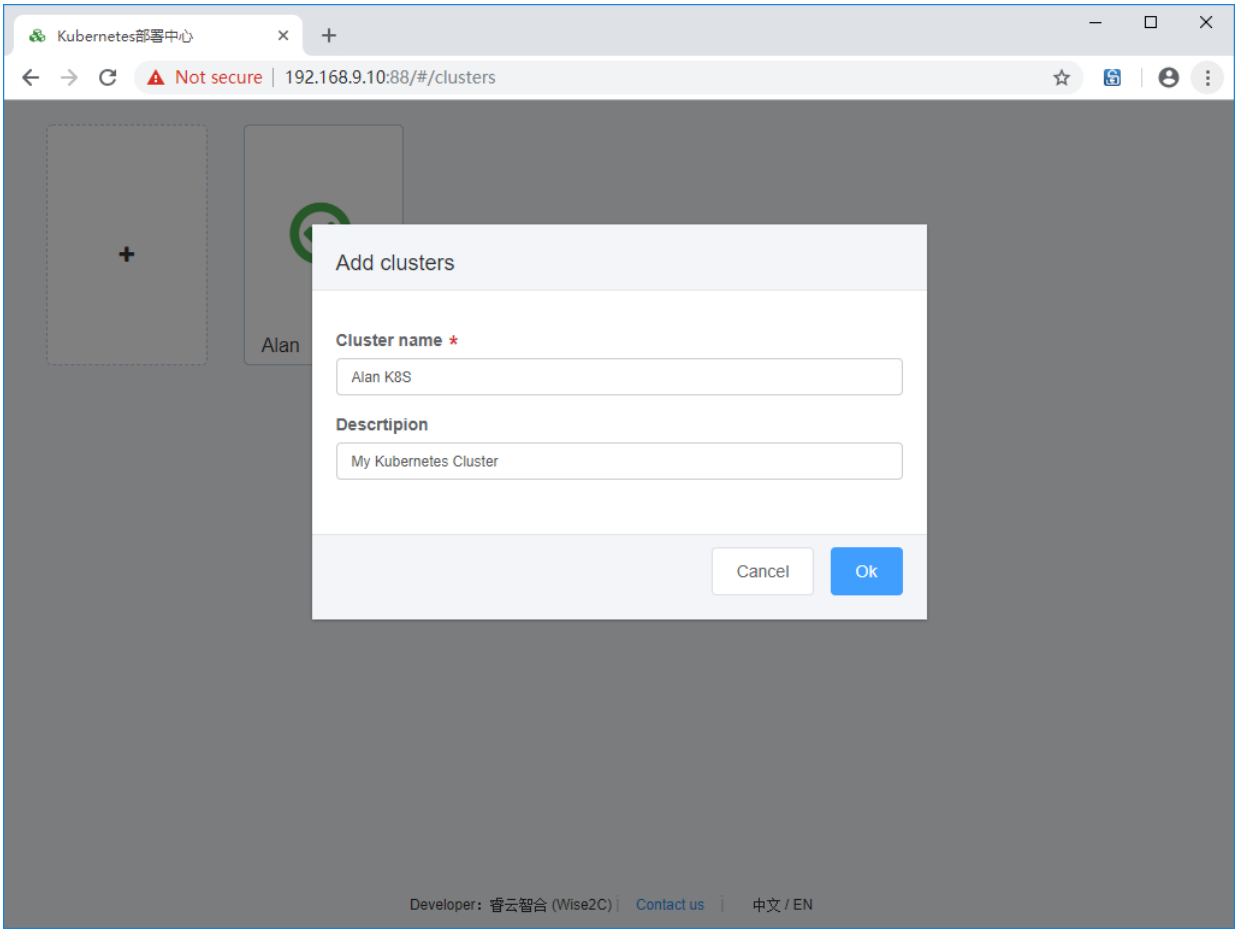
```
curl -L https://raw.githubusercontent.com/wise2ck8s/breeze/v1.12.3/docker-compose.yml -o docker-compose.yml
docker-compose up -d
```

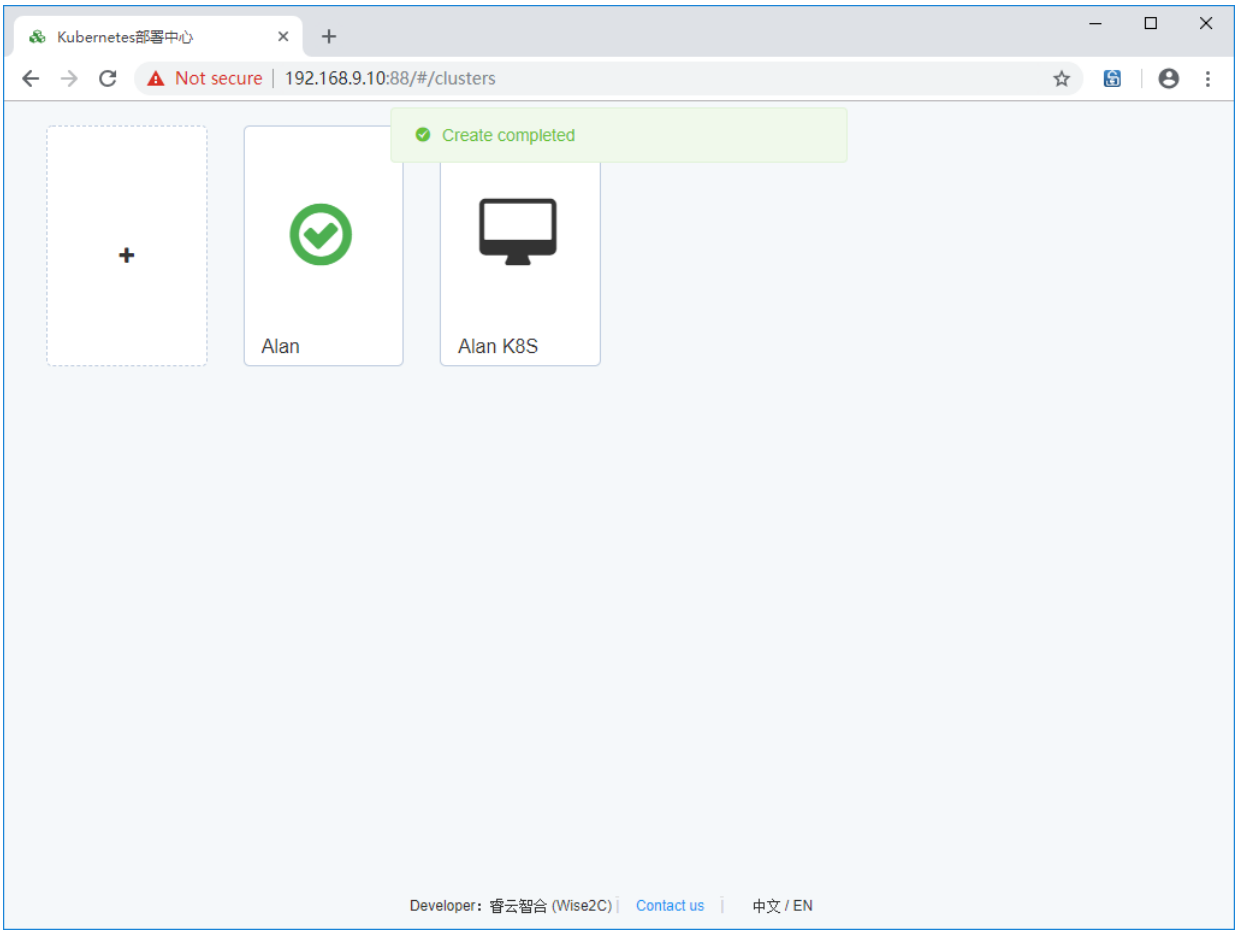
3. Access the Breeze web portal:

http://192.168.9.10:88

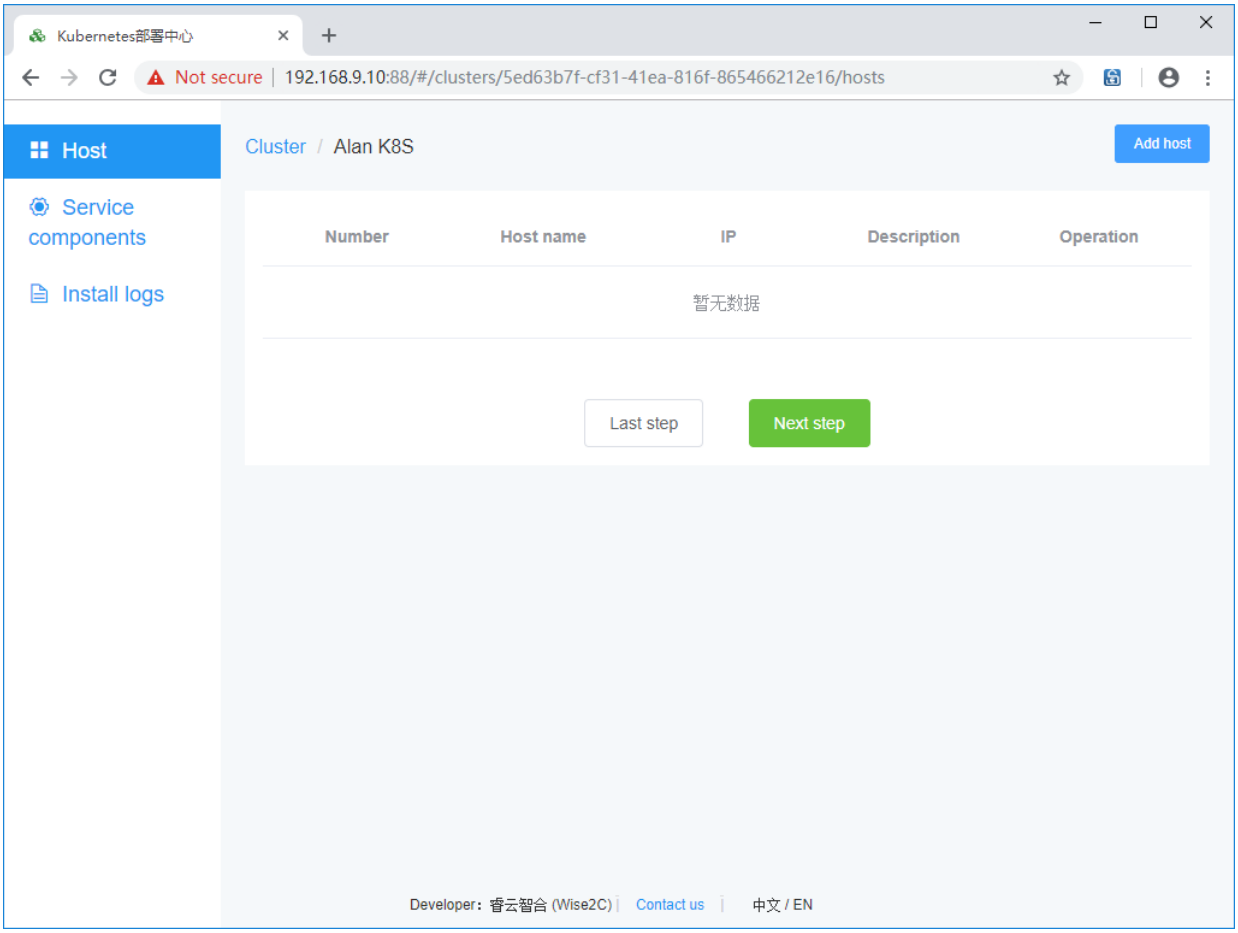


（1）Click “start” button and then click “+”

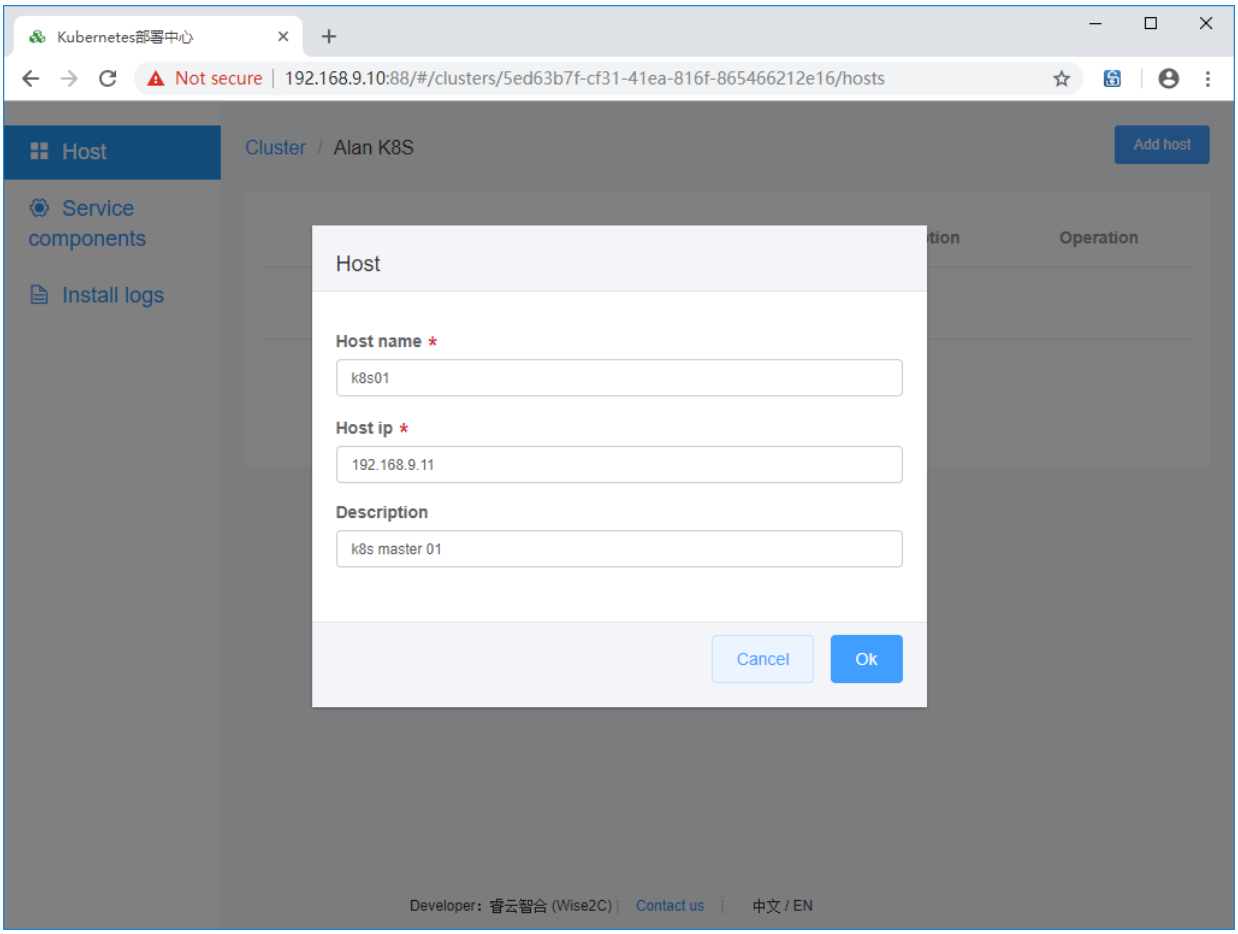




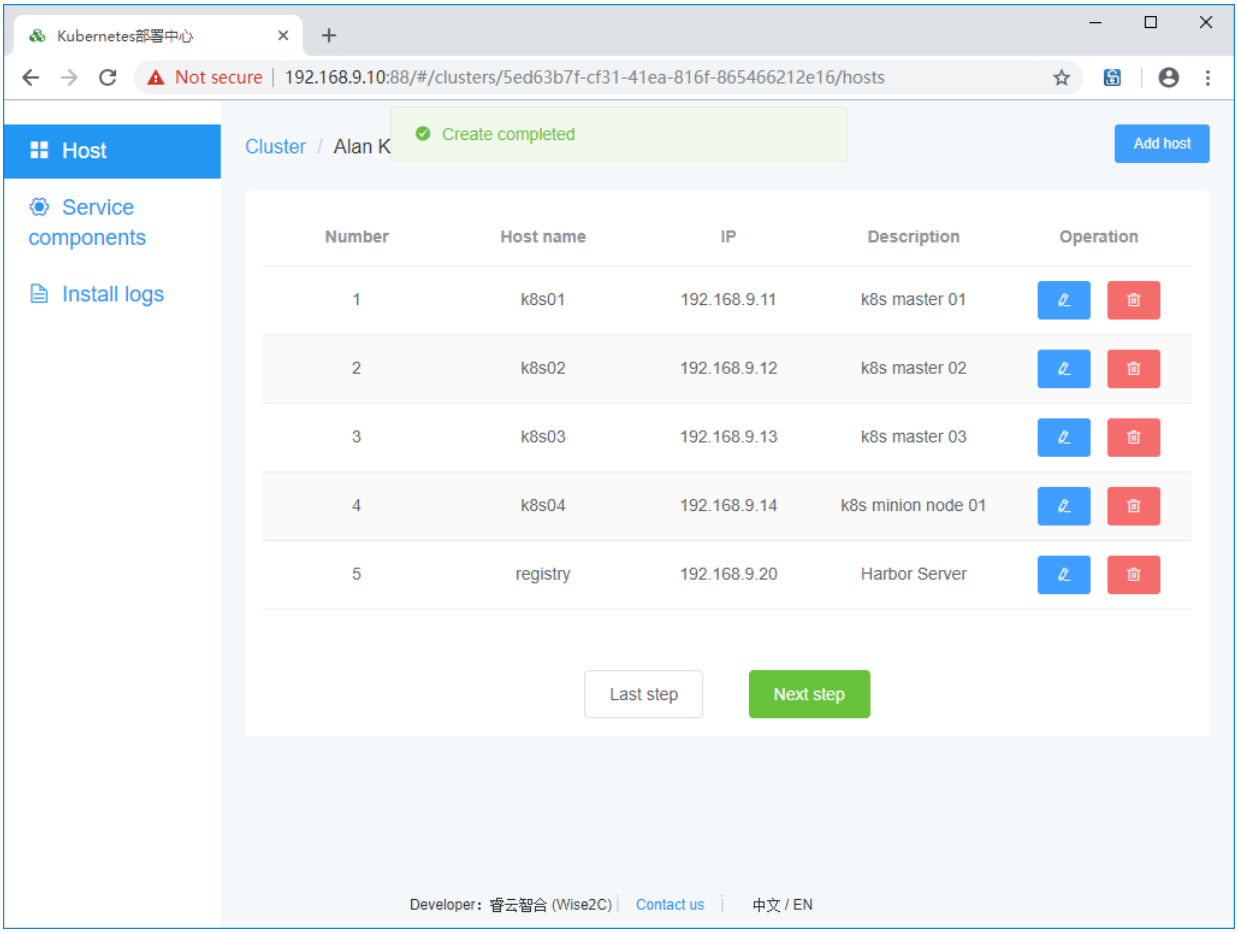
（2）Click this cluster icon to add servers:



Click the “Add host” button.

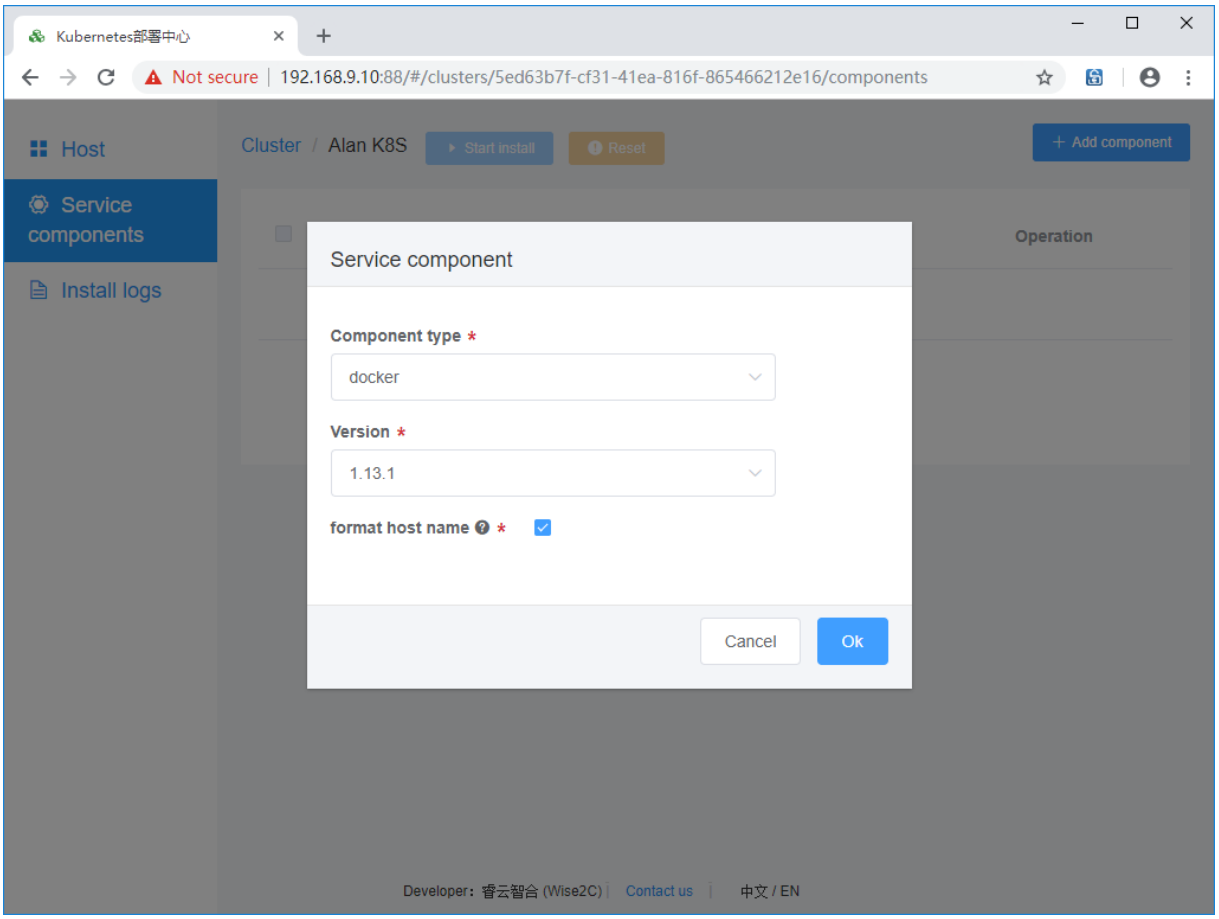


Repeatedly adding five servers to the entire cluster in turn:

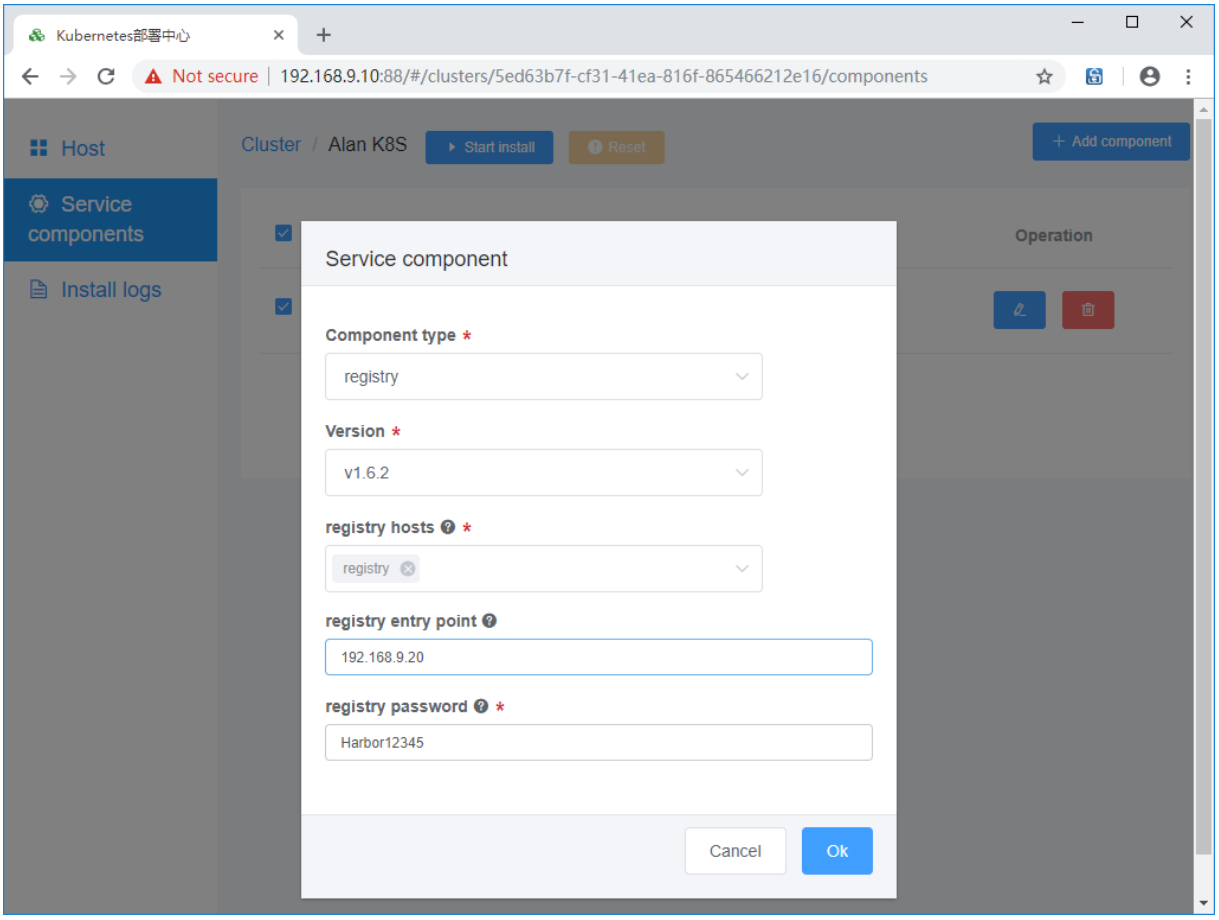


Click Next step for Service Component Definition

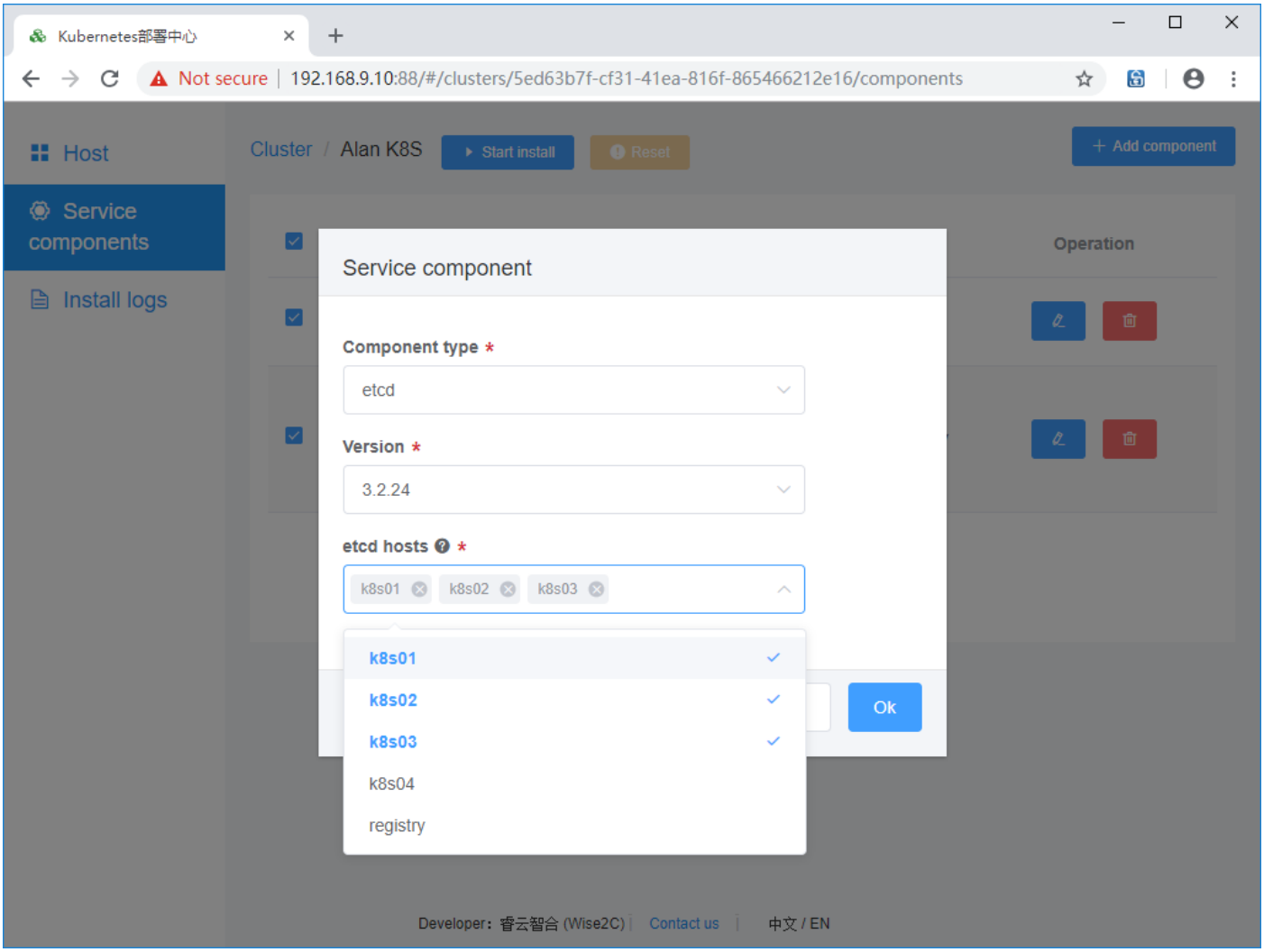
(3) Click on the "Add Components" button in the upper right corner to add service components and select docker, because all hosts need to be installed, so there is no need to select a server.



Adding harbor registry components. Note: registry entry point is the entry point for the Harbor server. You can use domain name or IP address.

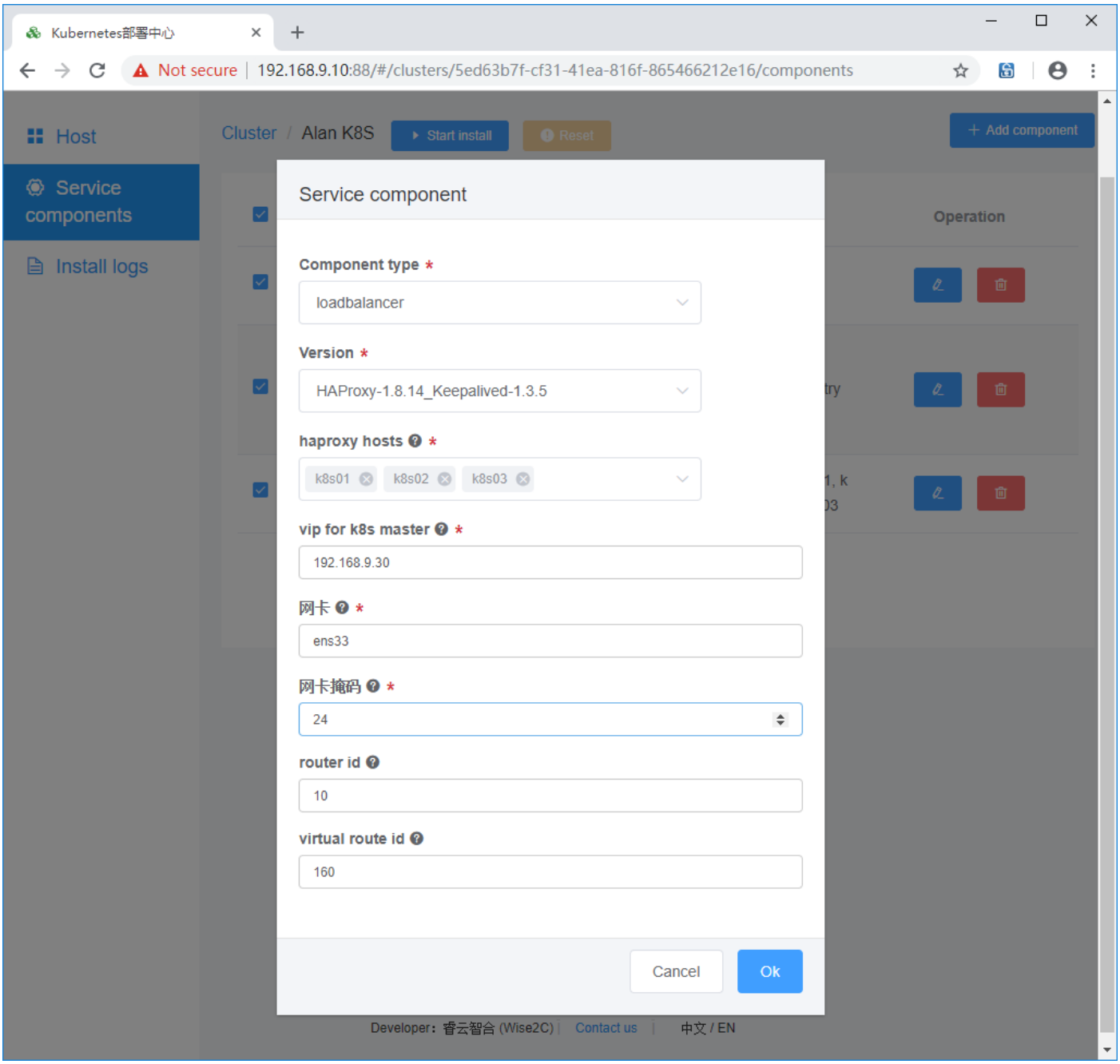


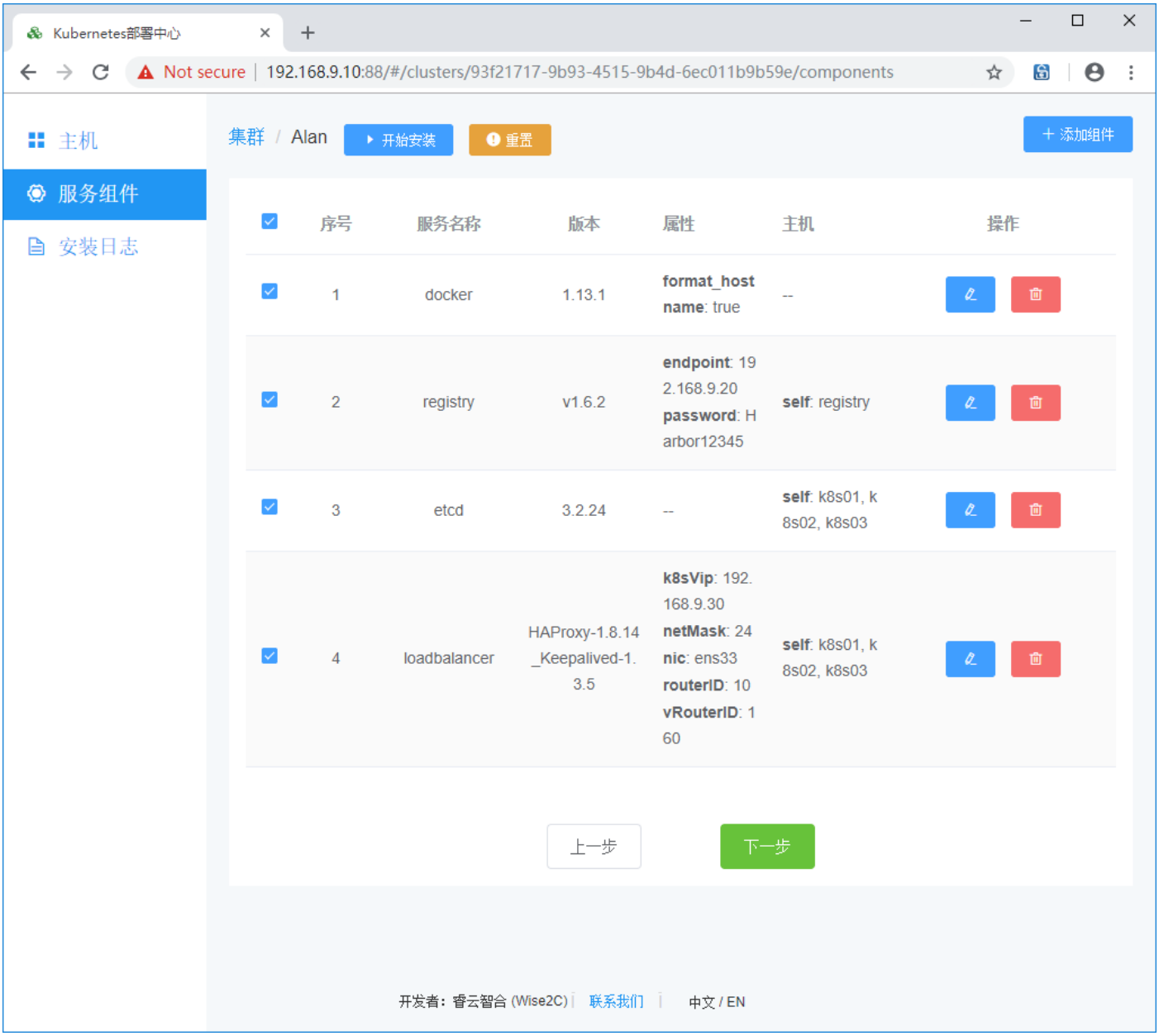
Then add etcd servers:



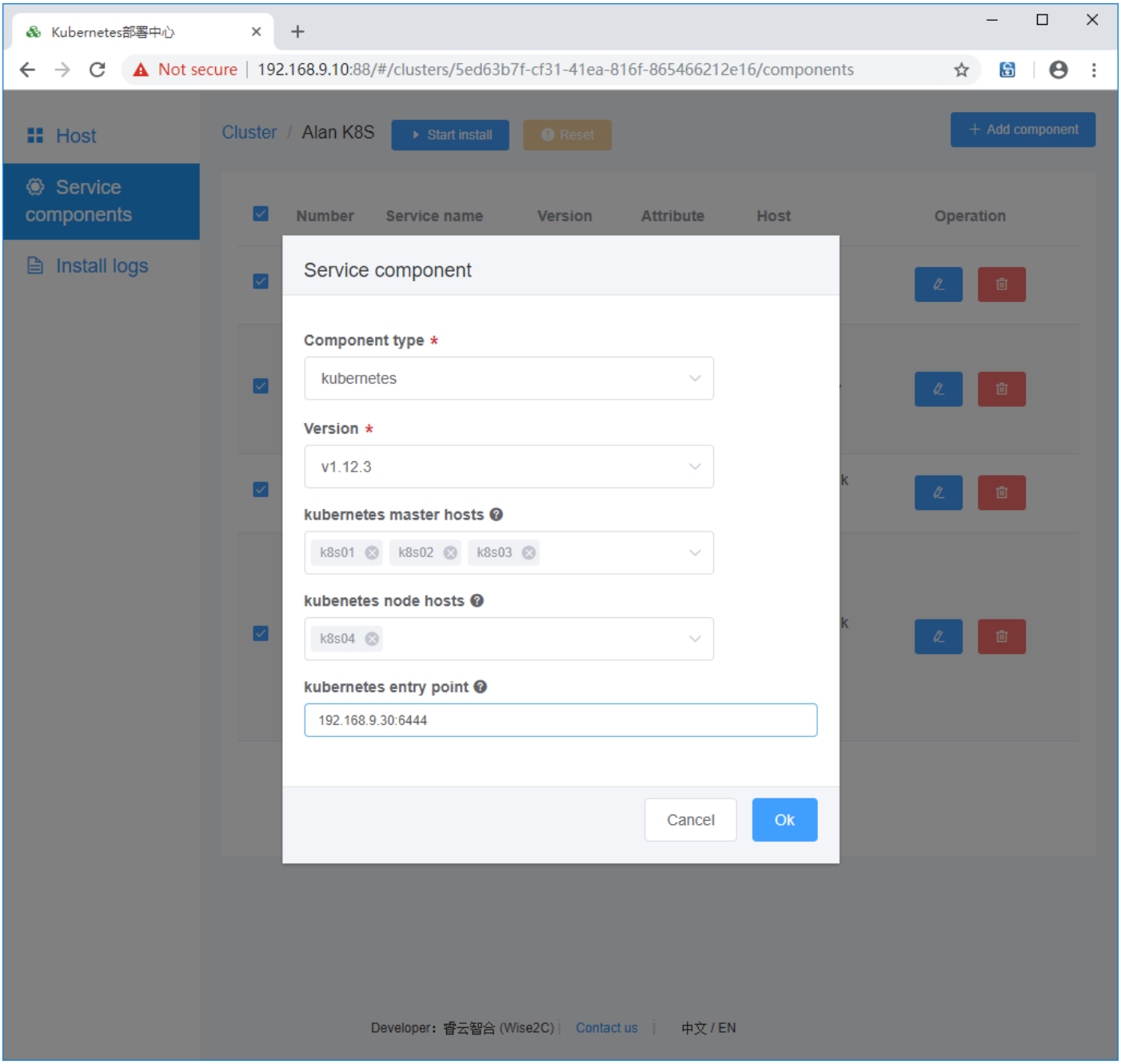
Next is the HA components（haproxy+keepalived）：

vip for k8s master: Virtual floating IP address for master servers.
NIC name: network interface name in the linux OS. Please ensure all the interfaces have the same name.
Network mask: input the network mask bit number
router id: configuration in keepalived, do not use same value for multiple clusters
virtual router id: configuration in keepalived, do not use same value for multiple clusters

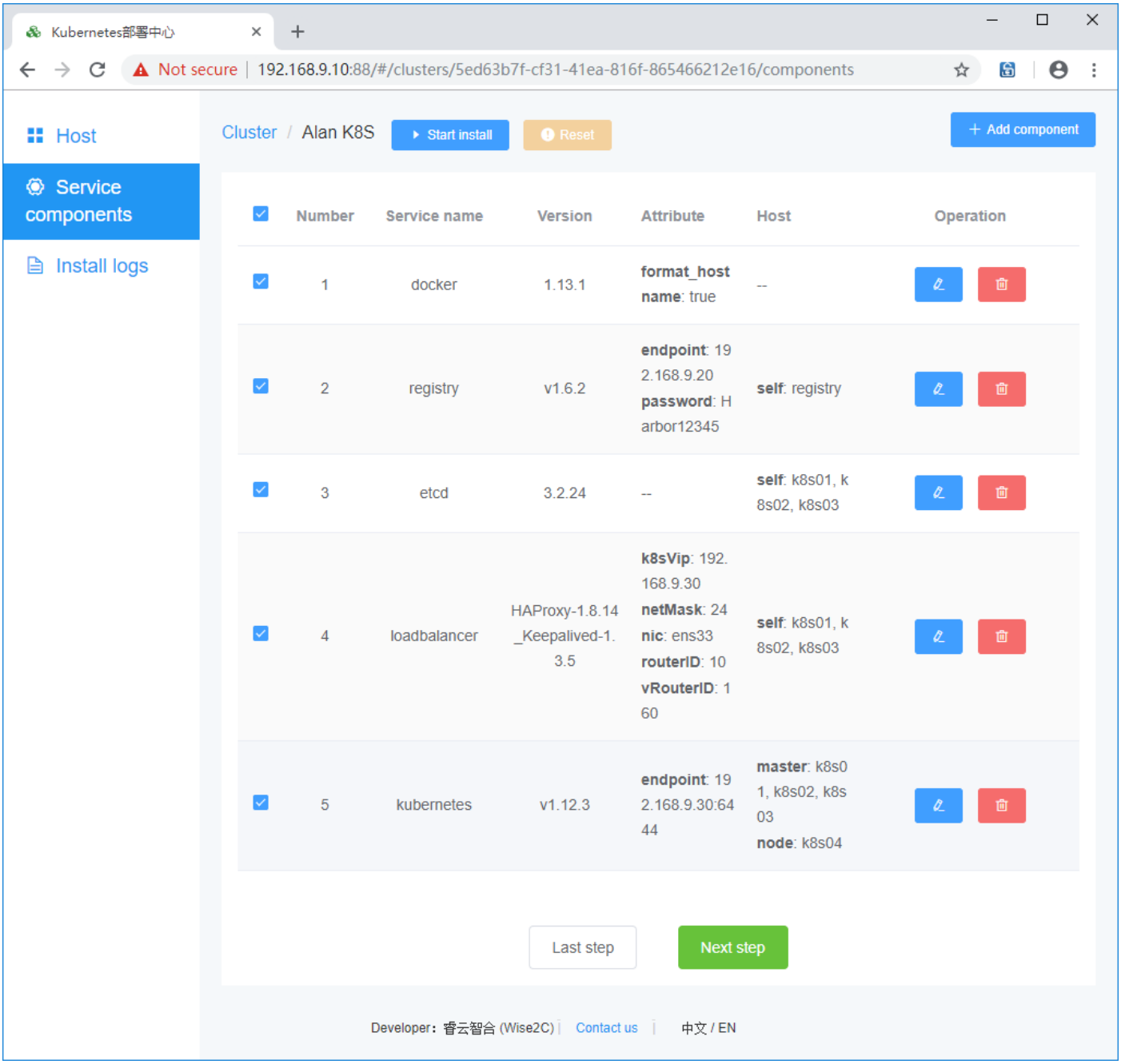



Kubernetes部署中心
Not secure
192.168.9.10:88/#/clusters/93f21717-9b93-4515-9b4d-6ec011b9b59e/components
主机
服务组件
安装日志
集群 / Alan
开始安装
重置
+ 添加组件
序号
服务名称
版本
属性
主机
操作
1
docker
1.13.1
format_host
name: true
--
2
registry
v1.6.2
endpoint: 19
2.168.9.20
password: H
arbor12345
self: registry
3
etcd
3.2.24
--
self: k8s01, k
8s02, k8s03
4
loadbalancer
HAProxy-1.8.14
_Keepalived-1.
3.5
k8sVip: 192.
168.9.30
netMask: 24
nic: ens33
routerID: 10
vRouterID: 1
60
self: k8s01, k
8s02, k8s03
上一步
下一步
开发者：睿云智合 (Wise2C)
联系我们
中文 / EN

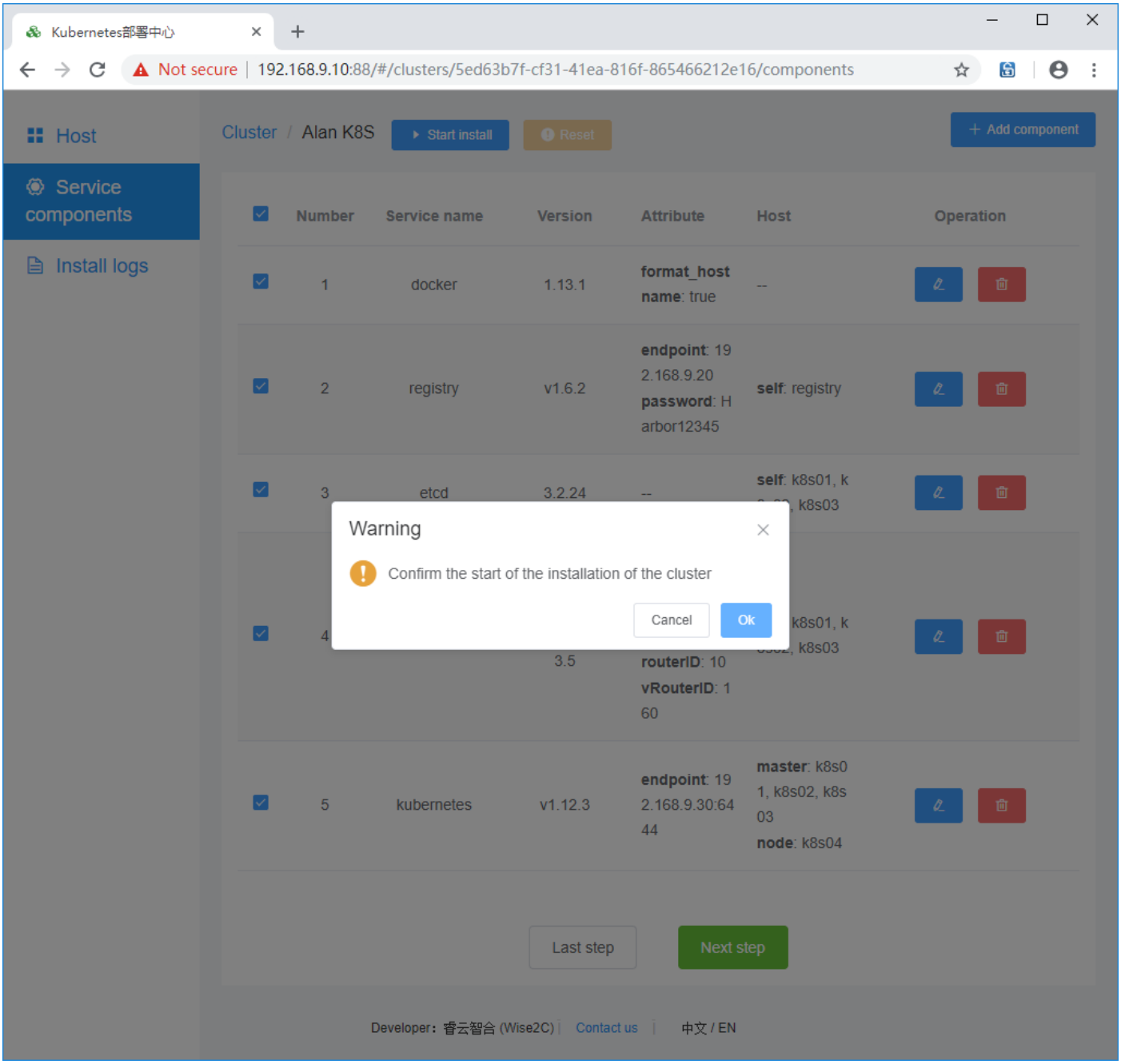
In the end, add the k8s components: master nodes and worker nodes:



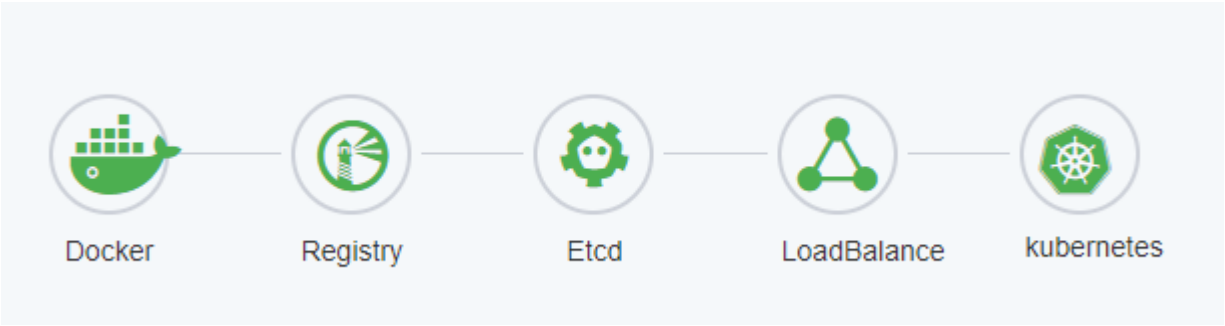
Set the correct values for kubernetes entry point: VIP:6444



4. Click Next step to start deploy:



Patiently waiting for all components icon to turn green which means the deployment is finished without problem.



Verify:

```
[root@k8s01 ~]# kubectl get cs
NAME                 STATUS    MESSAGE             ERROR
controller-manager   Healthy   ok
scheduler            Healthy   ok
etcd-0               Healthy   {"health": "true"}
etcd-1               Healthy   {"health": "true"}
etcd-2               Healthy   {"health": "true"}
[root@k8s01 ~]#
[root@k8s01 ~]# kubectl -n kube-system get pods
NAME                                    READY   STATUS    RESTARTS   AGE
coredns-64fcf865cf-gnvtt                1/1     Running   2          47h
coredns-64fcf865cf-x9r7q                1/1     Running   2          47h
heapster-7c4668c895-rxcwp               1/1     Running   2          47h
kube-apiserver-k8s01                    1/1     Running   2          47h
kube-apiserver-k8s02                    1/1     Running   2          47h
kube-apiserver-k8s03                    1/1     Running   2          47h
kube-controller-manager-k8s01           1/1     Running   2          47h
kube-controller-manager-k8s02           1/1     Running   2          47h
kube-controller-manager-k8s03           1/1     Running   2          47h
kube-flannel-ds-2vn5m                   1/1     Running   2          47h
kube-flannel-ds-k5wt6                   1/1     Running   2          47h
kube-flannel-ds-mzbmx                   1/1     Running   3          47h
kube-flannel-ds-wzdt7                   1/1     Running   3          47h
kube-proxy-c6jwq                        1/1     Running   2          47h
kube-proxy-fslhg                        1/1     Running   2          47h
kube-proxy-rtsmz                        1/1     Running   2          47h
kube-proxy-xbcd9                        1/1     Running   2          47h
kube-scheduler-k8s01                    1/1     Running   2          47h
kube-scheduler-k8s02                    1/1     Running   2          47h
kube-scheduler-k8s03                    1/1     Running   2          47h
kubernetes-dashboard-5db77f9dfb-swspm   1/1     Running   2          47h
monitoring-grafana-5565445645-c66p8     1/1     Running   2          47h
monitoring-influxdb-7bd68f7d84-52wjr    1/1     Running   2          47h
[root@k8s01 ~]#
[root@k8s01 ~]# kubectl get nodes -o wide
NAME    STATUS   ROLES    AGE   VERSION   INTERNAL-IP    EXTERNAL-IP   OS-IMAGE                KERNEL-VERSION          CONTAINER-RUNTIME
k8s01   Ready    master   47h   v1.12.3   192.168.9.11   <none>        CentOS Linux 7 (Core)   3.10.0-862.el7.x86_64   docker://1.13.1
k8s02   Ready    master   47h   v1.12.3   192.168.9.12   <none>        CentOS Linux 7 (Core)   3.10.0-862.el7.x86_64   docker://1.13.1
k8s03   Ready    master   47h   v1.12.3   192.168.9.13   <none>        CentOS Linux 7 (Core)   3.10.0-862.el7.x86_64   docker://1.13.1
k8s04   Ready    <none>   47h   v1.12.3   192.168.9.14   <none>        CentOS Linux 7 (Core)   3.10.0-862.el7.x86_64   docker://1.13.1
[root@k8s01 ~]#
```

Alan Peng