K3s+rancher单节点

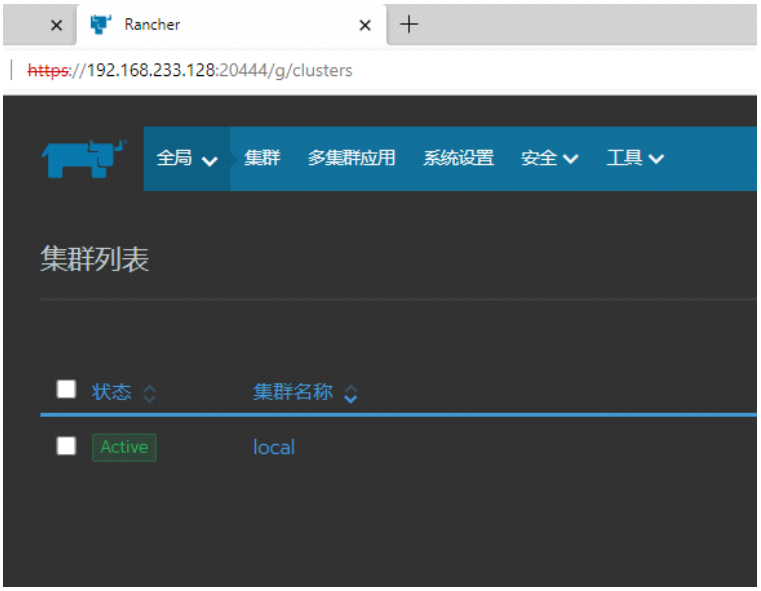
2022年2月15日 11:49

参考文档
https://docs.rancher.cn/docs/rancher2.5/installation/_index
https://docs.rancher.cn/docs/k3s/_index
https://www.jianshu.com/p/dd6199f32cfe
Ubuntu 2004
1，安装docker
curl https://releases.rancher.com/install-docker/20.10.sh | sh
2，安装rancher，内部80映射到外部2081端口，内部443映射到外部20444，使用-v参数实现内外的目录挂载，这里挂载/opt/rancher:/var/lib/rancher主要是实现服务器节点数据持久化
docker run -d --restart=unless-stopped -p 2081:80 -p 20444:443 --privileged -v /opt/rancher:/var/lib/rancher rancher/rancher:v2.5.12
3，查看运行状态
docker ps -a

```
root@ubuntu:/opt/rancher# docker run -d --restart=unless-stopped -p 2081:80 -p 20444:443 --privileged -v /
opt/rancher:/var/lib/rancher  rancher/rancher:v2.5.12
8f4bba49e0ab8862340c8ea013c604cef06f41c26e138372ae53037e433dd471
```

4，登录web的20444端口重置密码
https://192.168.233.128:20444/update-password

继续，登录，如果不是中文的话就退出重新登录
登录后的页面



5，安装k3s

由于我们安装的rancher是2.5.12版本的，再安装k3s之前首先要检查版本兼容性，兼容性列表可访问

https://www.suse.com/suse-rancher/support-matrix/all-supported-versions/rancher-v2-5-12/

通过兼容性列表可得知，2.5.12兼容最新的k3s版本为v1.20.14+k3s1

下面开始安装k3s的master端

export INSTALL_K3S_VERSION=v1.20.14+k3s1

export INSTALL_K3S_EXEC="--write-kubeconfig ~/.kube/config --write-kubeconfig-mode 644 --flannel-backend host-gw"

使用以上两个变量确定k3s的版本和相关参数，然后执行下面的安装

curl -sfL http://rancher-mirror.cnrancher.com/k3s/k3s-install.sh | INSTALL_K3S_MIRROR=cn sh -s - server

```
root@ubuntu:~# curl -sfL http://rancher-mirror.cnrancher.com/k3s/k3s-install.sh | INSTALL_K3S_MIRROR=cn sh
 -s - server
[INFO]  Using v1.20.14+k3s1 as release
[INFO]  Downloading hash http://rancher-mirror.cnrancher.com/k3s/v1.20.14-k3s1/sha256sum-amd64.txt
[INFO]  Downloading binary http://rancher-mirror.cnrancher.com/k3s/v1.20.14-k3s1/k3s
[INFO]  Verifying binary download
[INFO]  Installing k3s to /usr/local/bin/k3s
[INFO]  Creating /usr/local/bin/kubectl symlink to k3s
[INFO]  Creating /usr/local/bin/crictl symlink to k3s
[INFO]  Skipping /usr/local/bin/ctr symlink to k3s, command exists in PATH at /usr/bin/ctr
[INFO]  Creating killall script /usr/local/bin/k3s-killall.sh
[INFO]  Creating uninstall script /usr/local/bin/k3s-uninstall.sh
[INFO]  env: Creating environment file /etc/systemd/system/k3s.service.env
[INFO]  systemd: Creating service file /etc/systemd/system/k3s.service
[INFO]  systemd: Enabling k3s unit
[INFO]  systemd: Starting k3s
root@ubuntu:~#
```

上述安装脚本curl -sfL http://rancher-mirror.cnrancher.com/k3s/k3s-install.sh | INSTALL_K3S_MIRROR=cn sh -s -是官方标准的安装脚本，

server不知道含义是什么意思，

--write-kubeconfig　　#将管理客户端的 kubeconfig 写入这个文件

--write-kubeconfig-mode　#使用这种模式写入 kubeconfig。允许写入 kubeconfig 文件的选项对于允许将 K3s 集群导入 Rancher 很有用。示例值为 644。

--flannel-backend　#使用主机网关承载k3s集群网络

6，在rancher上导入集群

点击导入按钮，输入名称直接点确定，弹出如下画面

查看k3s的状态

k3s kubectl get nodes

```
root@ubuntu:~# k3s kubectl get nodes
NAME     STATUS   ROLES                  AGE   VERSION
ubuntu   Ready    control-plane,master   27m   v1.22.6+k3s1
```

直接执行刚才web上面复制的命令

根据介绍，我这边是没有考虑证书问题，所以复制最后一段命令在终端上执行，命令带kubectl apply -f -

curl --insecure -sfL https://192.168.233.128:20444/v3/import/txjdtwx9fwp7qhkvm4hccdtlx4dwp6p4v5n22tx7wq9nllgnl6nzbn_c-xh2c2.yaml | kubectl apply -f -

```
root@ubuntu:~# curl --insecure -sfL https://192.168.1.111:20444/v3/import/w6n8hscfl9c7bl7tp9dqwgh7h4rzc2bq
nhc2grvx8v5l5zlhkzpgzw_c-m9w56.yaml | kubectl apply -f -
clusterrole.rbac.authorization.k8s.io/proxy-clusterrole-kubeapiserver created
clusterrolebinding.rbac.authorization.k8s.io/proxy-role-binding-kubernetes-master created
namespace/cattle-system created
serviceaccount/cattle created
clusterrolebinding.rbac.authorization.k8s.io/cattle-admin-binding created
secret/cattle-credentials-1108e43 created
clusterrole.rbac.authorization.k8s.io/cattle-admin created
deployment.apps/cattle-cluster-agent created
root@ubuntu:~#
```

查看master端的token

token位置：/var/lib/rancher/k3s/server/node-token

cat /var/lib/rancher/k3s/server/node-token

```
root@ubuntu:~# cat /var/lib/rancher/k3s/server/node-token
K10c1e507fa566cd4bc43a2aece5d29dfb605bb015a00f99e0b1c2b438e80fd88a1::server:de30b178d17b4297049ced9dce2aa5
75
```

添加临时变量

```
export INSTALL_K3S_VERSION=v1.20.14+k3s1
export K3S_TOKEN="K10c1e507fa566cd4bc43a2aece5d29dfb605bb015a00f99e0b1c2b438e80fd88a1::server:de30b178d17b4297049ced9dce2aa57 5"
export K3S_URL="https://192.168.1.111:6443"
export K3S_NODE_NAME="agent-2"
```

INSTALL_K3S_VERSION #定义k3s版本

K3S_TOKEN #定义master节点的token

K3S_URL #定义master节点的对接url

K3S_NODE_NAME #定义本节点的节点名称

curl -sfL http://rancher-mirror.cnrancher.com/k3s/k3s-install.sh | INSTALL_K3S_MIRROR=cn K3S_URL=${K3S_URL} K3S_TOKEN=${K3S_TOKEN} sh -

定义docker私有镜像

在所有的k3s-agent的服务器上创建/etc/rancher/k3s/registries.yaml文件

vim /etc/rancher/k3s/registries.yaml

文件内容

```
mirrors:
  docker.io:
    endpoint:
      - "https://mycustomreg.com:5000"
configs:
  "mycustomreg:5000":
```

```
    auth:
      username: xxxxxx # 这是私有镜像仓库的用户名
      password: xxxxxx # 这是私有镜像仓库的密码
    tls:
      cert_file: # 镜像仓库中使用的cert文件的路径。
      key_file:  # 镜像仓库中使用的key文件的路径。
      ca_file:   # 镜像仓库中使用的ca文件的路径。
```

实测可以不使用验证，跳过tls，

不使用验证跳过tls的配置文件内容为

```
mirrors:
  192.168.1.112:
    endpoint:
     - "192.168.1.112"
configs:
  192.168.1.112:
    auths: {}
    tls:
     insecure_skip_verify: true
```