在公有云建立集群验证submariner

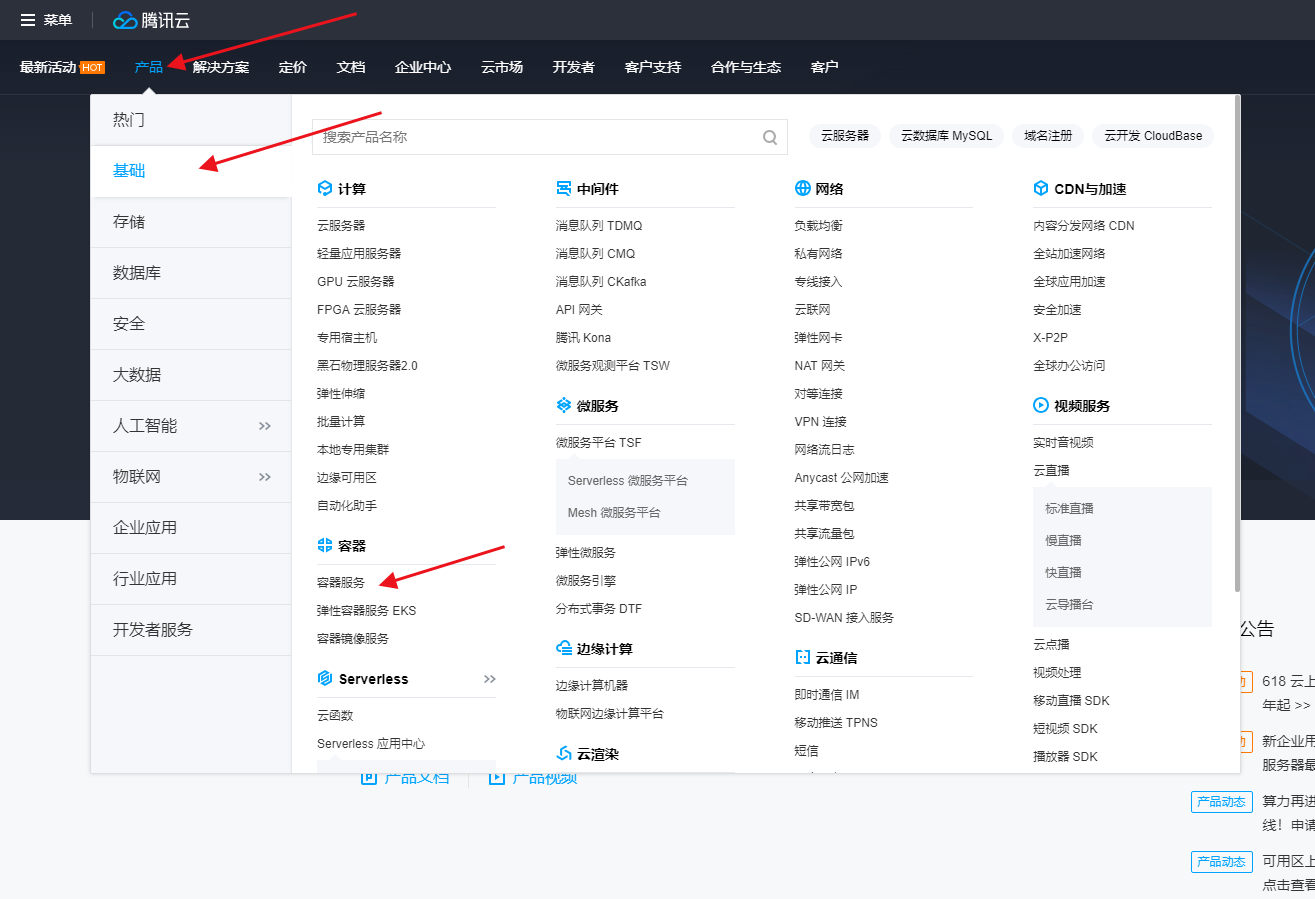
20210619

1. 在公有云建立集群

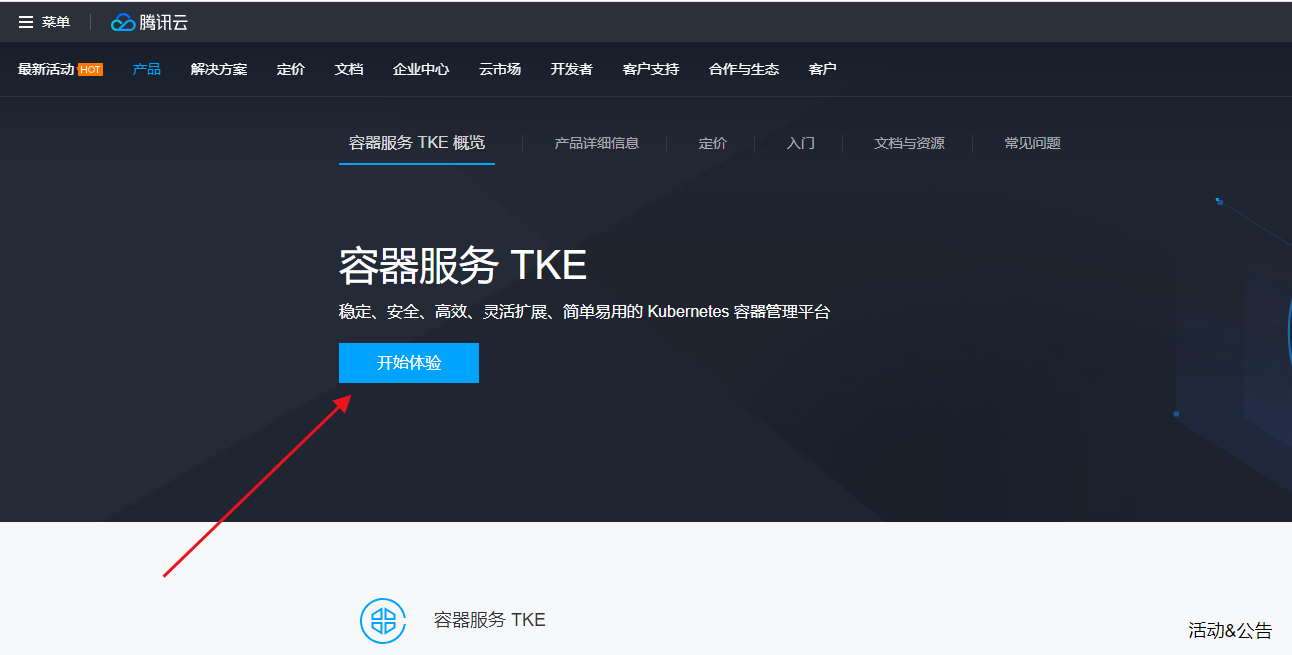
（1.1）首先登陆腾讯云官网（https://cloud.tencent.com/），点击“云服务器”



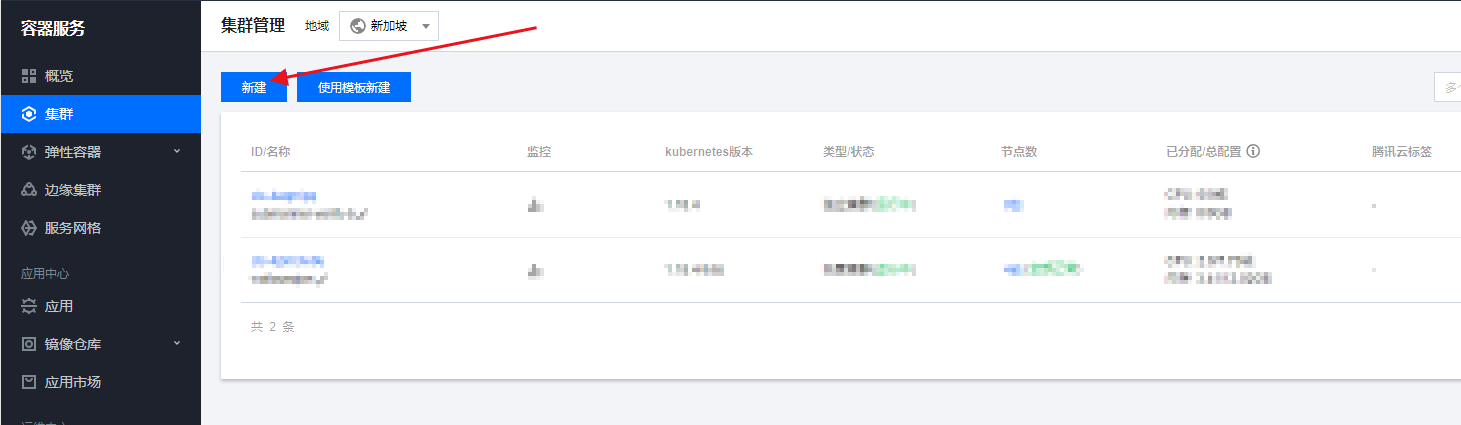
（1.2）点进去之后，依次点击“产品”，“基础”，“容器服务”



（1.3）在“容器服务”的页面，点击“开始体验”



（1.4）进去之后，点击“新建”



（1.5）配置集群，具体参数和注意事项在截图中给出



（1.6）然后在容器子网中选择一个子网id，这个子网id未来是和pod CIDR相关的，所以这个也要记下来，例如这里我选的是subnet-fz0l7lba

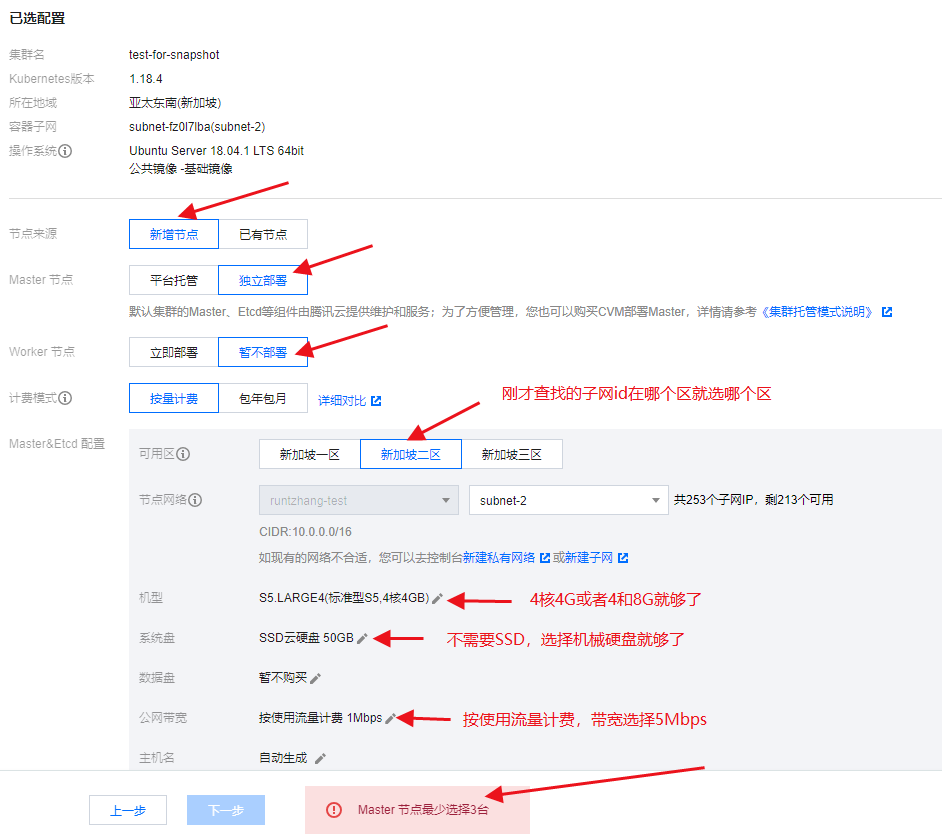


（1.7）这时先别急着点“下一步”，先找pod CIDR，新打开一个标签页，输入网址<https://console.cloud.tencent.com/vpc/subnet>

然后先选择云服务器所在的区域（例如我的是新加坡），然后在搜索框中查找子网id，然后回到刚才的新建服务器集群的页面，点击下一步



（1.8）这里按照截图走就好了，然后点击“下一步”



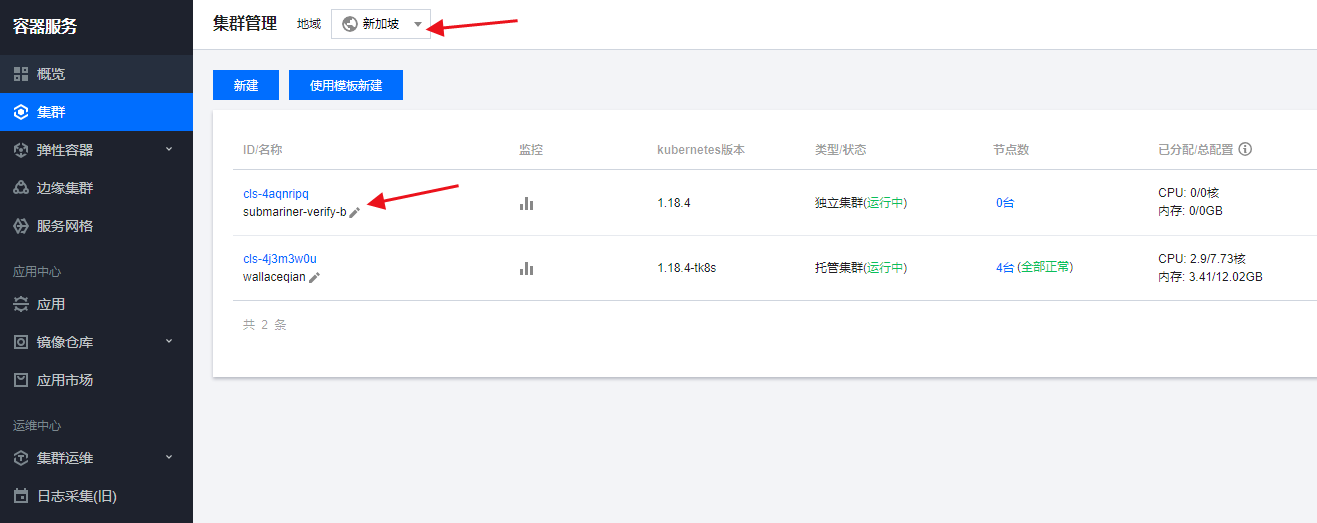
（1.9）这一步还比较简单，设置密码



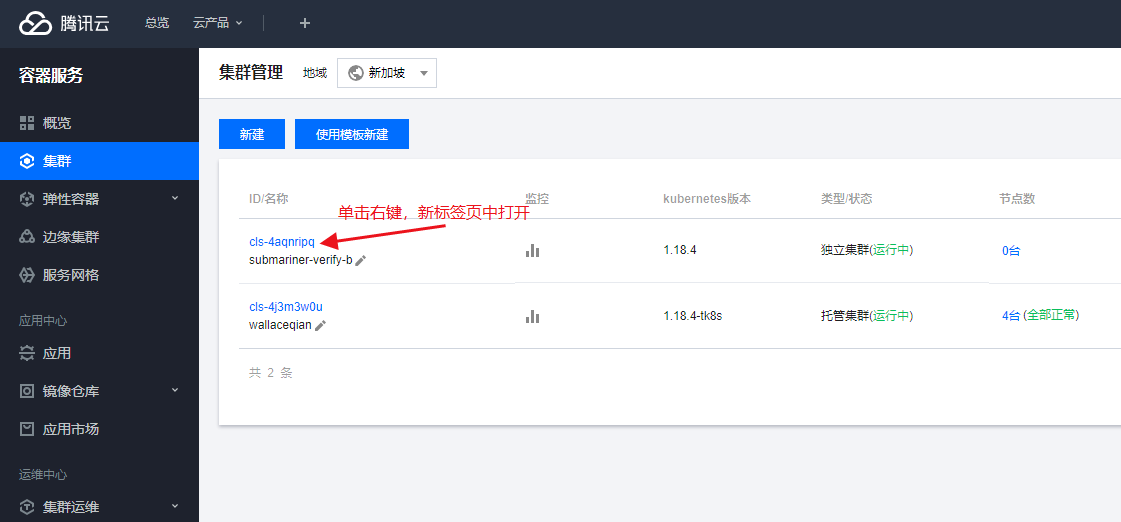
（1.10）这一步会让你选择集群的一些组件，不选也可以，点击下一步，然后信息确认，开始配置集群



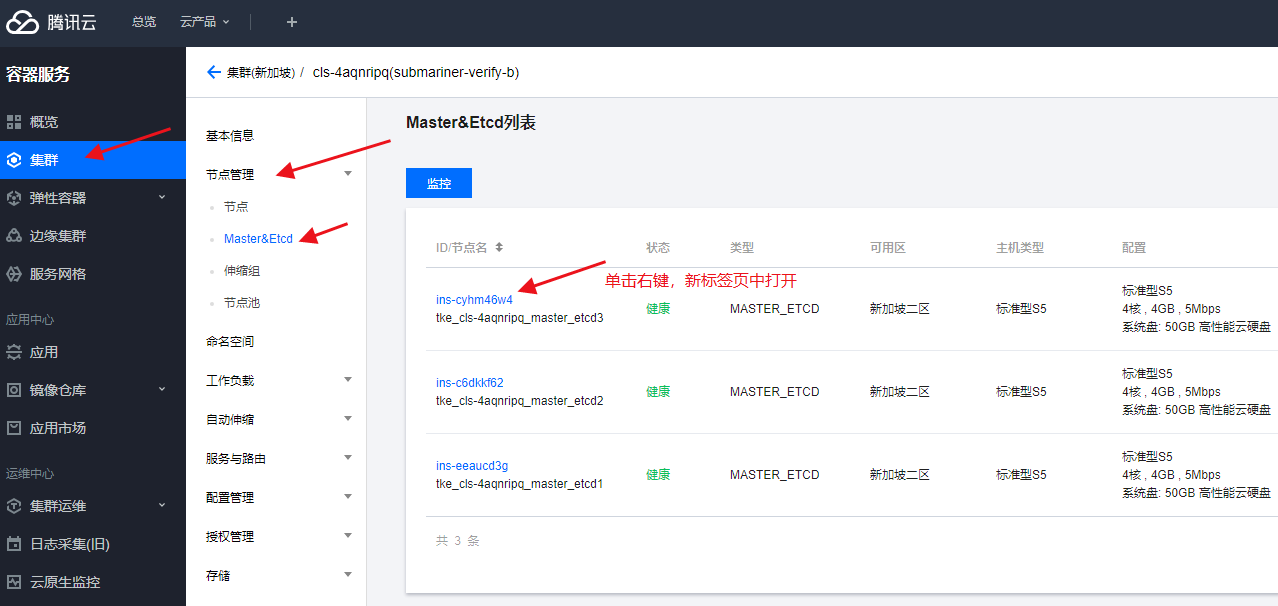
（1.11）然后等上3-5分钟，集群就搭建好了，因为创建集群的时候会有一个集群名字，根据这个名字就可以找到你的集群



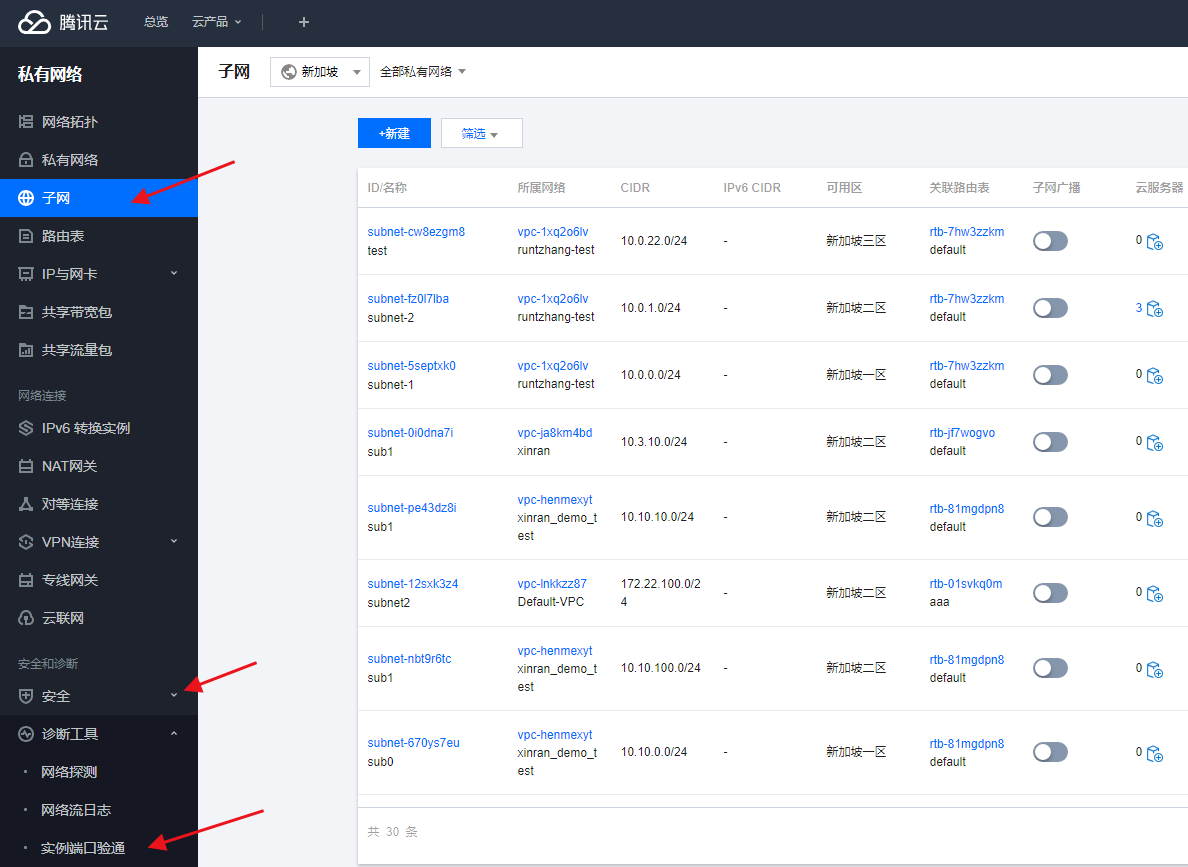
（1.12）现在需要进入集群，找到对应的master节点，开放它的全部端口。还是在上一步的页面，然后单击右键，选择新标签页中打开，进入集群

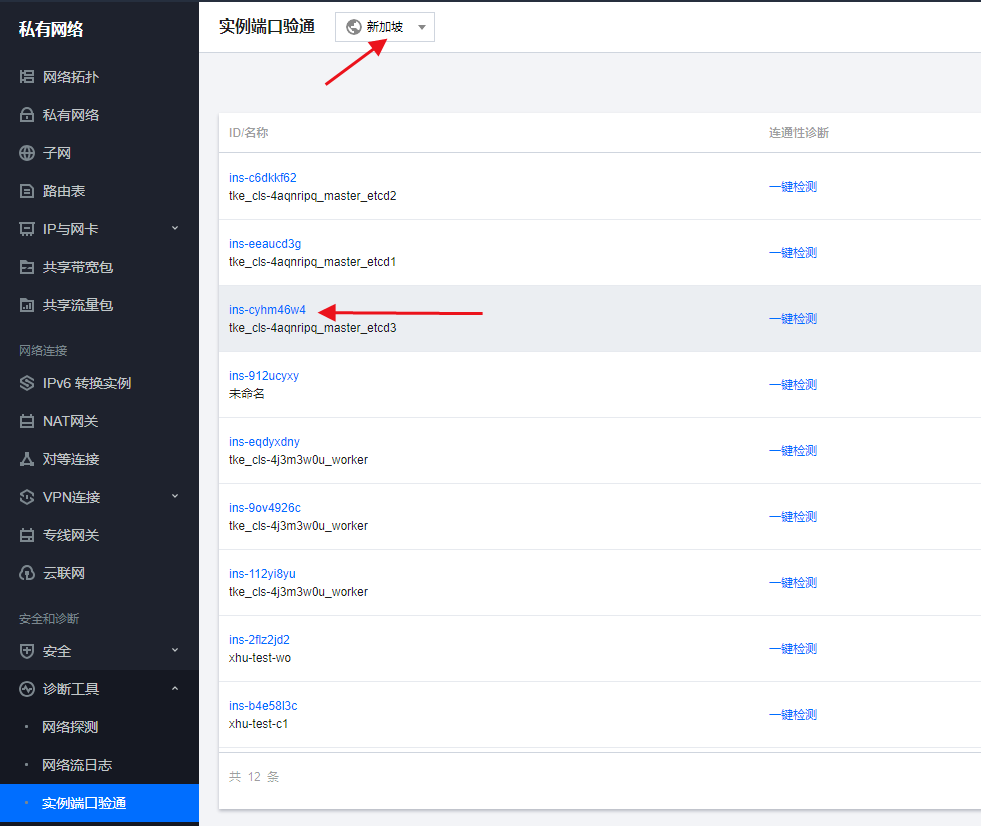


（1.13）点击“集群”🡪“节点管理”🡪“Master&Etcd”选择一个master节点，把鼠标放在蓝色的超链接上会显示“跳转到CVM实例详情页”，单击右键，在新标签页中打开（直接单击左键打开的页面是不对的）

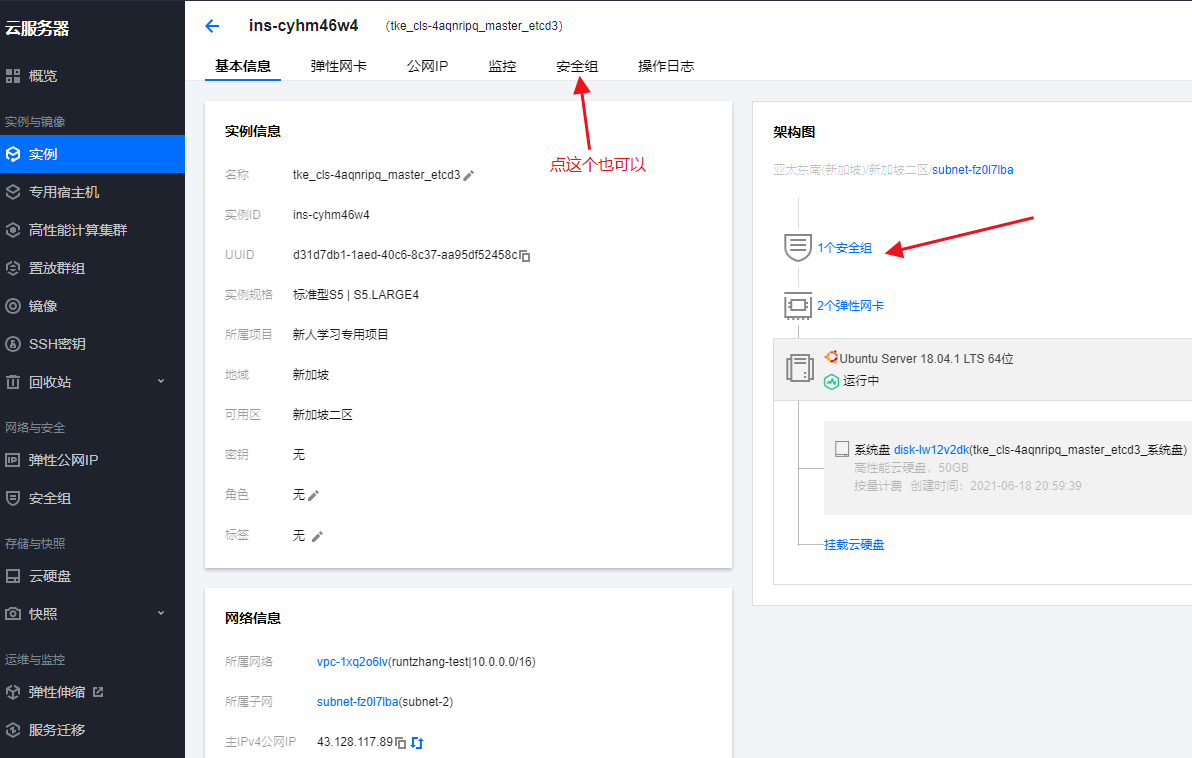


（1.14）如果打开的页面是空的，可以找到刚才（1.7）时查找pod CIDR的页面，点击“安全”🡪“实力端口验通”，然后选择区域新加坡，可能还可以直接找到你想要的master节点，如果没找到，在搜索栏搜索master节点的id也可以，找到之后，单击右键，打开新标签页

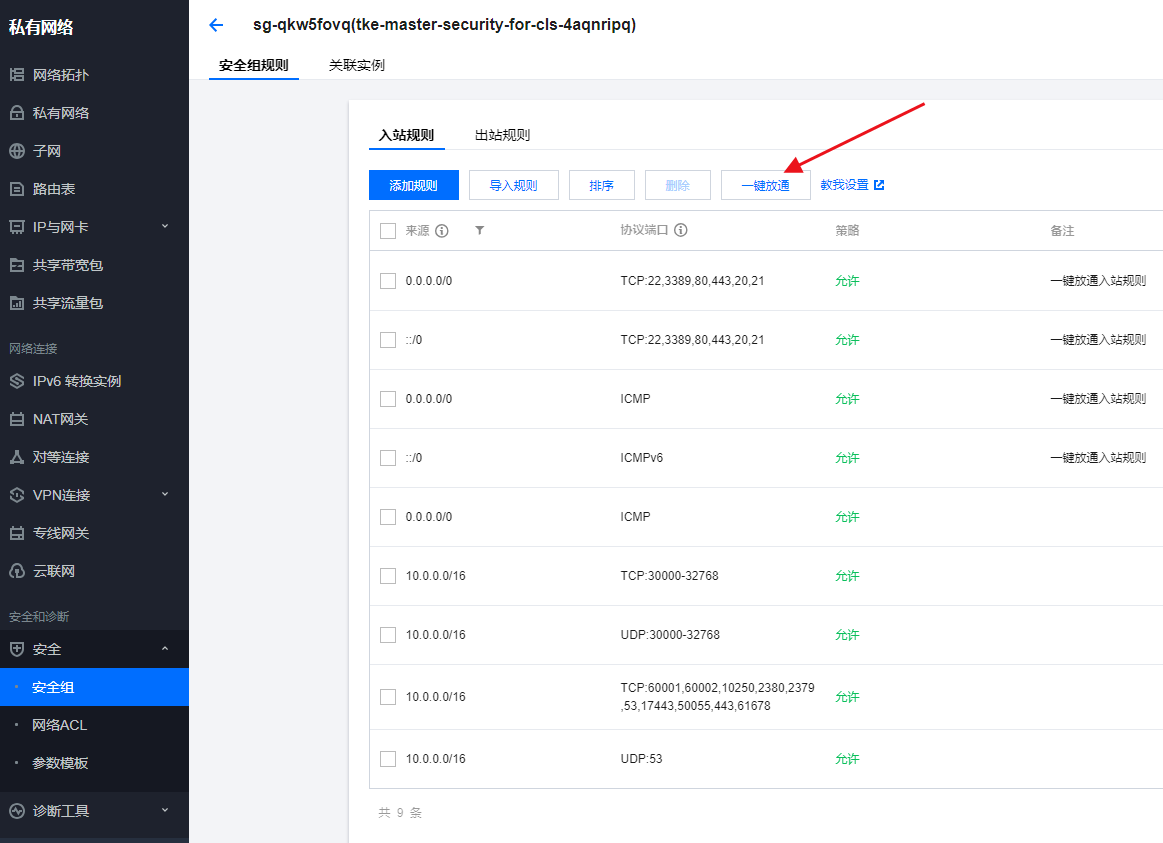


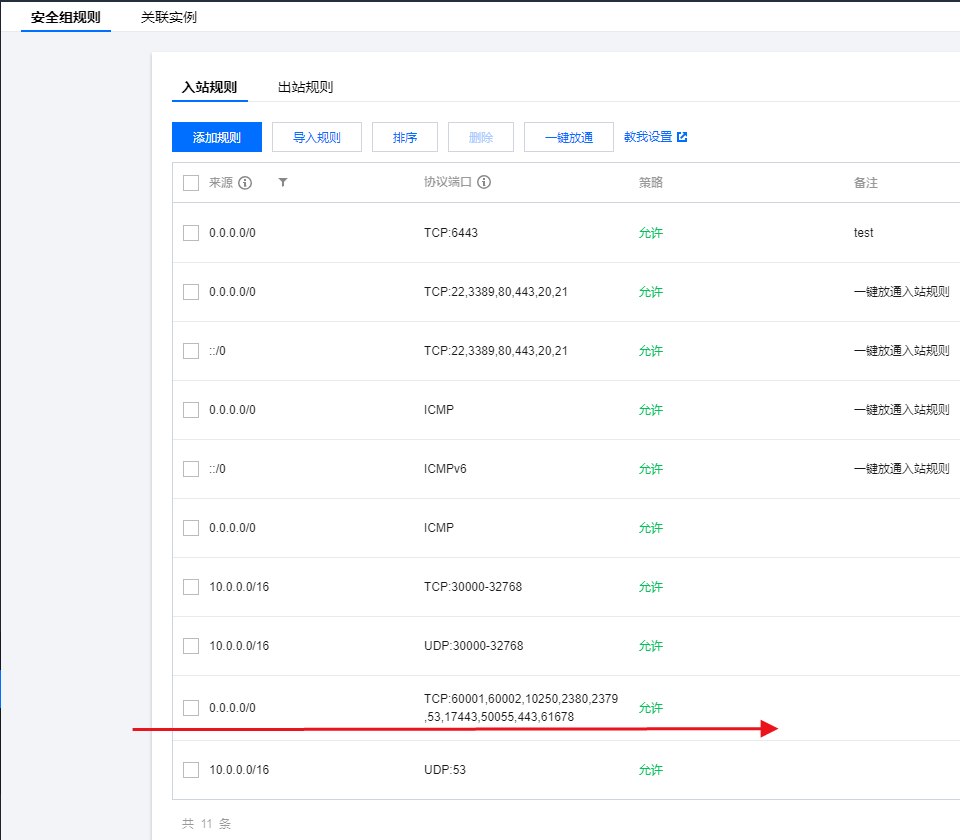


（1.15）进去之后点击安全组

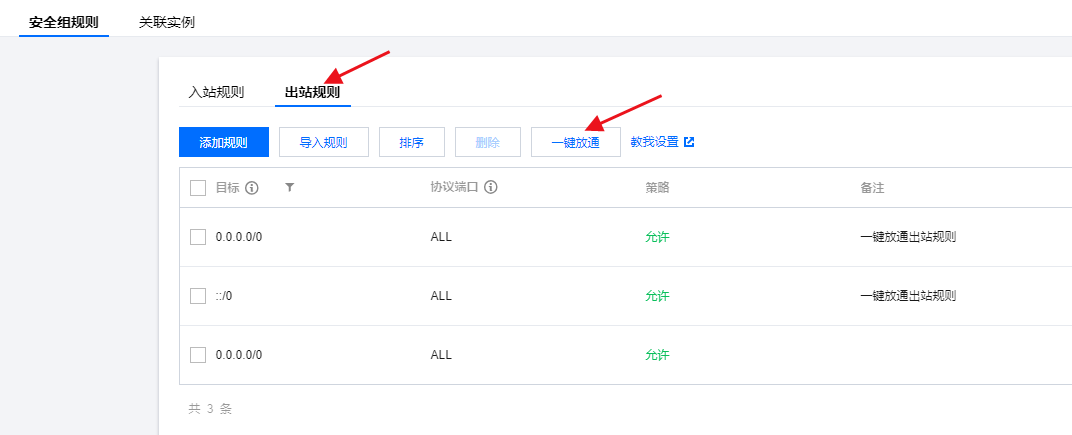


（1.15.1）对于入站规则，点击“一键放通”，确定，然后还要再修改一条入站规则





出站规则也要修改



（1.15.2）如果觉得（1.15.1）比较麻烦的话，可以从采用这种办法，直接放通全部端口



cluster-a和cluster-b一共2个cluster，6个master，都要按照刚才的步骤设置安全组策略

同样地，再搭建一个集群，选择不同的区域（比如刚才选择的是新加坡，这次选择香港），然后开放master节点的所有端口，记录总结两个集群的配置信息

|  |  |  |  |
| --- | --- | --- | --- |
| ubuntu 18.04; VPC CIDR: 10.0.0.0/16; Moscow; POD CIDR: 10.0.0.0/24; SERVICE CIDR: 10.45.0.0/24; iptables | | | |
| ubuntu 18.04; VPC CIDR: 10.0.0.0/16; Singapore; POD CIDR: 10.0.1.0/24; SERVICE CIDR: 10.145.0.0/24; iptables | | | |
| kubernetes version | pod/pod connectivity across cluster | service discovery | note |
| 1.18.4 | ok | ok | 独立集群部署；subctl v0.9.0 |

复习CIDR（无类别域间路由，Classless Inter-Domain Routing）

一个IP地址包含两部分：标识网络的前缀和紧接着的在这个网络内的主机地址。以前，IP地址的分配把IP地址的32位按每8位为一段分开，所以前缀必须为8，16或者24位。这导致不能充分使用IP地址同时在路由不方便，因为大量的需要单独路由的小型网络（C类网络）因在地域上分得很开而很难进行聚合路由，于是给路由设备增加了很多负担。CIDR是基于可变长子网掩码（VLSM)来进行任意长度的前缀的分配，遵从CIDR规则的地址有一个后缀说明前缀的位数，例如 192.168.0.0/20。

1. 部署submariner前的检查

（2.1）公有云已经做好的

安装docker（公有云默认安装）

conntrack（公有云默认安装）

禁用selinux

firewall（公有云默认没装）

（2.2）需要自己做的

（2.2.1）配置安全组策略

（2.2.2）以root用户登录

sudo passwd root

1q2w3e4!Q@W#E$

su

vi /etc/ssh/sshd\_config

PermitRootLogin yes

（2.2.3）禁用swap

swapoff -a

sed -i 's/.\*swap.\*/#&/' /etc/fstab

（2.2.4）hostname修改为小写（这是submariner的bug，现在已经被修复了，但是还是要修改hostname）

vi /etc/hostname

vm-XXXXX

vi /etc/cloud/cloud.cfg

preserve\_hostname: true

（2.2.5）安装subctl

curl -Ls https://get.submariner.io | VERSION=0.9.0 bash curl -Ls https://get.submariner.io | bash

export PATH=$PATH:~/.local/bin

echo export PATH=\$PATH:~/.local/bin >> ~/.profile

（2.2.6）ssh连通cluster-b

ssh-keygen

ssh-copy-id <ip address>

（2.2.7）calicoctl不安装

（2.2.8）安装yq（选装）

BINARY=yq\_linux\_amd64

VERSION=v4.8.0

wget https://github.com/mikefarah/yq/releases/download/${VERSION}/${BINARY} -O /usr/bin/yq && chmod +x /usr/bin/yq

yq --help

（2.2.9）安装dos2linux（选装）

（2.3）卸载已有的submariner，如果本来就没有安装submariner和nginx，直接跳转（2.5）

（2.3.1）卸载已有的submariner（官网教程https://submariner.io/operations/cleanup/）

# 删除broker-info.subm

rm -rf broker-info.subm

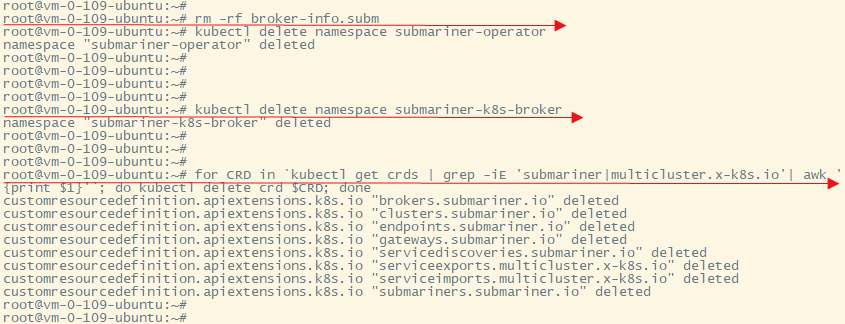
（2.3.2）# 对每一个加入submariner的节点，删除指定的namespace（每一个delete语句大概要等30秒），对于submariner的broker所在的节点还需要多执行一条命令删除broker

kubectl delete namespace submariner-operator

kubectl delete namespace submariner-k8s-broker

（2.3.3）# 删除submariner的 CRDs

for CRD in `kubectl get crds | grep -iE 'submariner|multicluster.x-k8s.io'| awk '{print $1}'`; do kubectl delete crd $CRD; done



（2.3.4）# 对每一个加入submariner的节点，删除submariner的ClusterRoles和ClusterRoleBinding

roles="submariner-operator submariner-operator-globalnet submariner-lighthouse submariner-networkplugin-syncer"

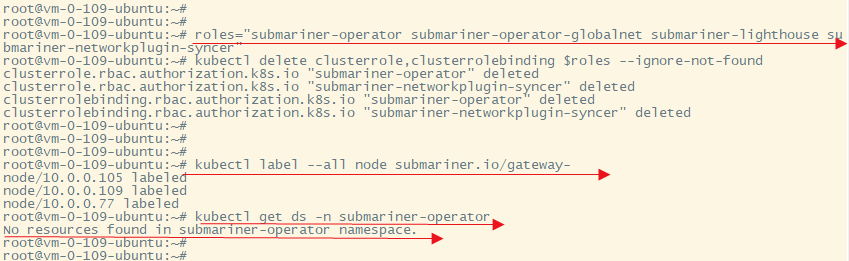
kubectl delete clusterrole,clusterrolebinding $roles --ignore-not-found

（2.3.5）# 删除submariner的网关标签（命令最后的减号表示去除标签）

kubectl label --all node submariner.io/gateway-

# 查看是否删除成功，如果提示说指定的资源not found那么就是成功了（ds表示daemonset，是在Kubernetes 集群里运行一个daemon pod，它管理全部机器上的pod 副本，​负责对它们进行更新和删除）

kubectl get ds -n submariner-operator



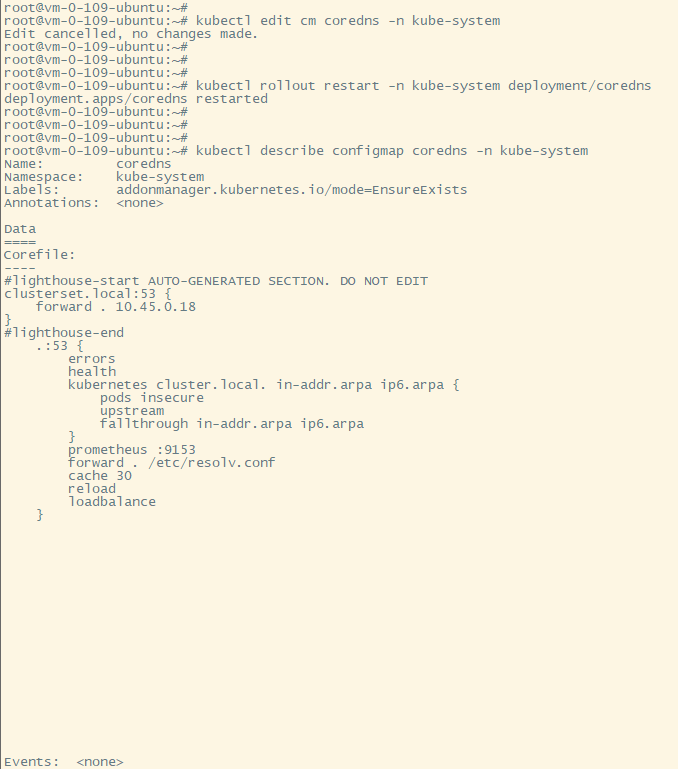
（2.3.6）# 手动编辑coredns的Corefile configmap然后删除lighthouse条目（两条命令选一个）

kubectl edit cm coredns -n kube-system

kubectl rollout restart -n kube-system deployment/coredns

# 检查dns-default config map的Corefile文件，验证lighthouse条目被删除

kubectl describe configmap coredns -n kube-system



（2.3.7）# 删除submariner在iptables中的链

iptables --flush SUBMARINER-INPUT

iptables -D INPUT $(iptables -L INPUT --line-numbers | grep SUBMARINER-INPUT | awk '{print $1}')

iptables --delete-chain SUBMARINER-INPUT

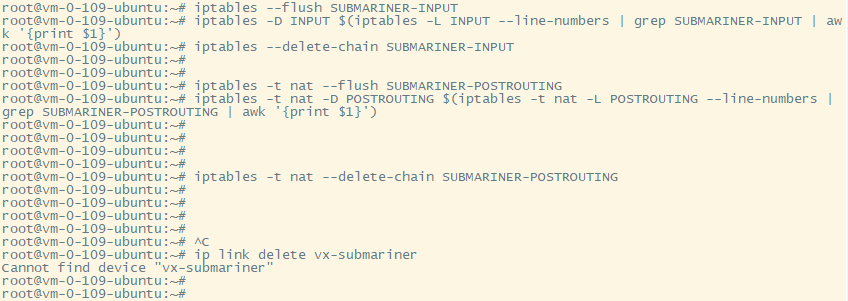
iptables -t nat --flush SUBMARINER-POSTROUTING

iptables -t nat -D POSTROUTING $(iptables -t nat -L POSTROUTING --line-numbers | grep SUBMARINER-POSTROUTING | awk '{print $1}')

iptables -t nat --delete-chain SUBMARINER-POSTROUTING

# 删除vx-submariner接口

ip link delete vx-submariner



（2.4）卸载nginx

# 先查看当前的node节点（cluster-b）安装了哪些pod

kubectl get po -o wide

# 删除已经安装的nginx的这个pod

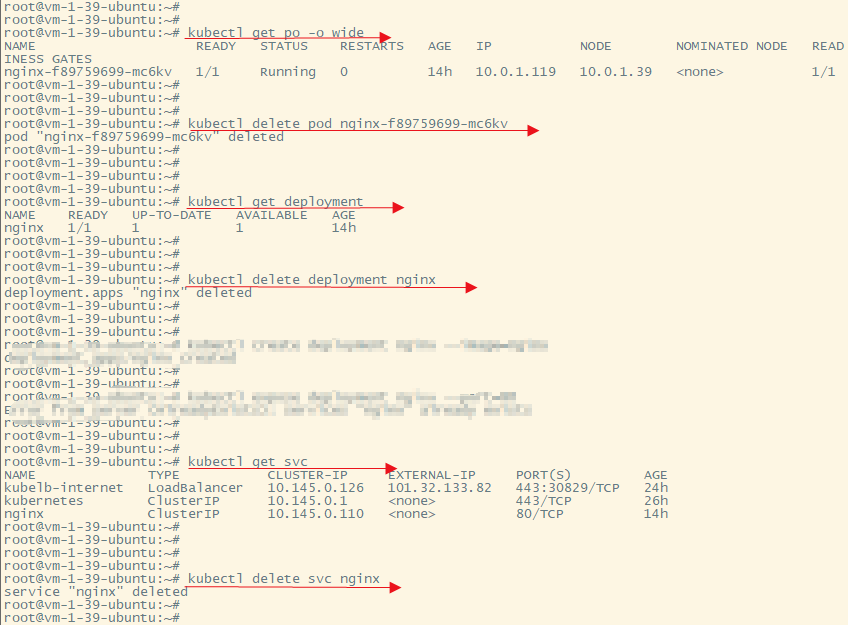
kubectl delete pod <nginx的pod名字>

# 删除部署nginx的deployment

kubectl delete deployment nginx

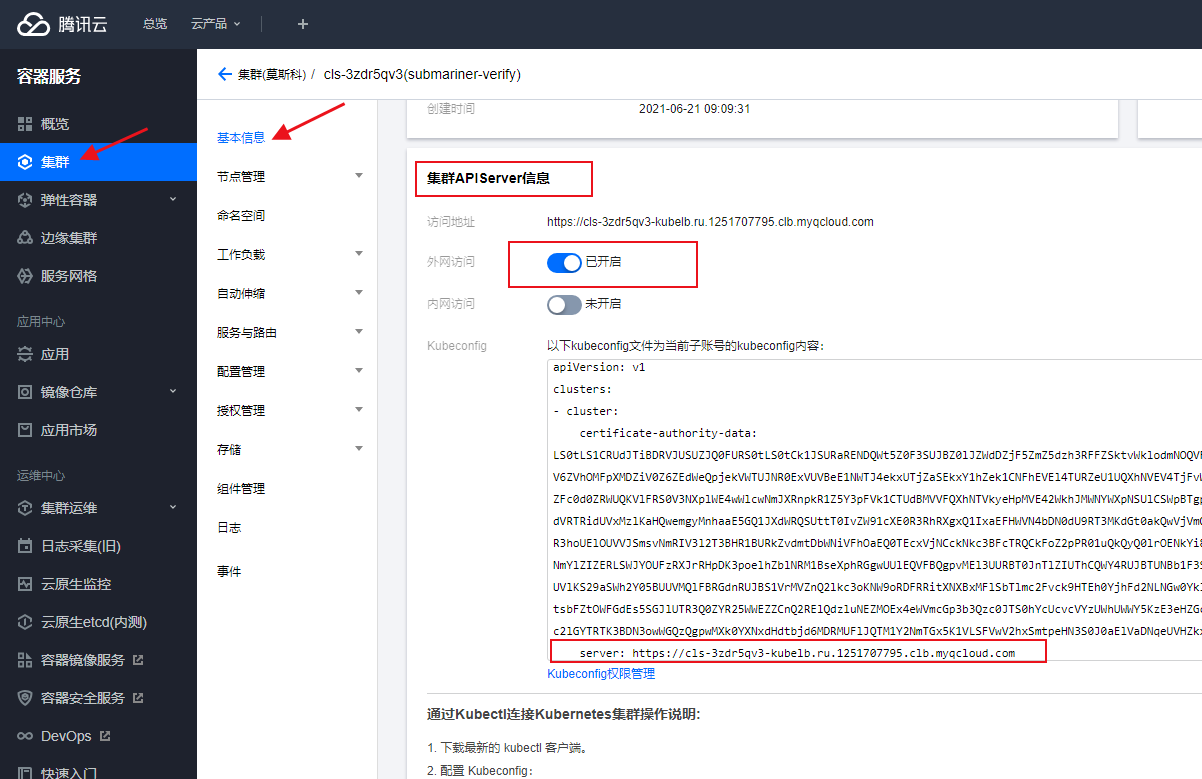
# 删除nginx的service

kubectl delete svc nginx



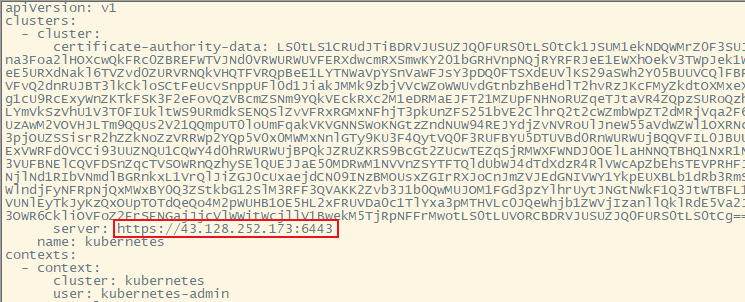
（2.5）修改cluster-a 中的master01的配置文件（\*\*\*这一步很重要），首先需要开启集群的外网访问（两个cluster都开启外网访问），集群 🡪 基本信息，然后往下翻，找到集群APIServer信息（如果没有找到“集群”页面，参看（1.11）-（1.13）），然后选择开启外网访问（需要等待大概30秒）。

然后找到server: <https://XXXX> 这个字段，复制下这个URL



回到secure CRT，修改配置文件，把框起来的这个字段替换成刚才那个URL，保存并退出

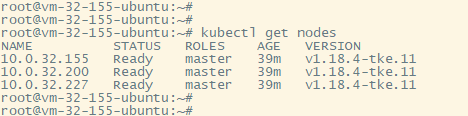
vi $HOME/.kube/config



三、在cluster-a部署submariner

# 首先查看集群的状态，可以看到，cluster-a有三个master，master的名字就是各自的内网ip

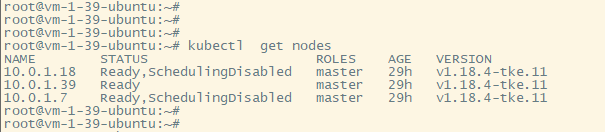
（3.1）kubectl get nodes



（3.2）# 把当前节点保留，然后执行cordon命令禁用其他两个节点

kubectl cordon <node name>

# 如果有节点被scheduling disabled ，那么将会是这样，有这些标记的节点不会被部署任何pod



或者在控制台也可以看到，有两个节点的状态是“已封锁”



# 如果要解除节点的scheduling disabled状态，cordon表示封禁，uncordon表示解除封禁

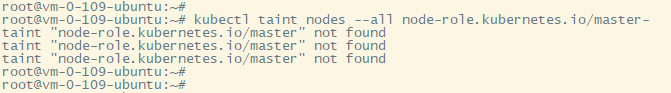
kubectl uncordon <.节点的名字>

（3.3）# 把当前节点都标记为submariner的网关未来部署的节点

kubectl label node <node name> submariner.io/gateway=true

（3.4）# 默认情况下，出于安全原因，集群不会在control-plane节点上调度pod。如果非要在control-plane节点上调度pod，例如为单机Kubernetes集群，取消master节点的污点标记，（参见网页链接 https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/create-cluster-kubeadm/）如果你得到的是下面这种结果，说明你已经执行过这个语句了

kubectl taint nodes --all node-role.kubernetes.io/master-



（3.5）# 部署submariner

subctl deploy-broker

# 向cluster-b节点发送. subm文件

scp broker-info.subm 43.128.117.89:/root

# cluster-a以cluster-a的名字加入自己的submariner，这里还会让你输入cluster-a的pod CIDR

subctl join broker-info.subm --clusterid cluster-a --natt=true



如果出现cluster-a的deploy命令执行成功，而cluster-a继续执行join命令却失败，大概率是步骤（2.5）没有做好，会去检查（2.5）中的config文件。

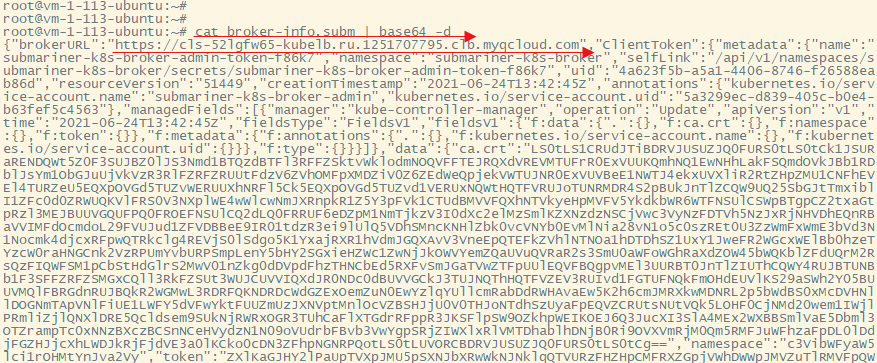
如果真的是（2.5）没做好导致的话，直接对当前node回滚然后再重新deploy并不能解决问题，原因未知。可行的解决方案是，立即cordon当前node，uncordon之前禁用的某个节点，然后用这个节点重新开始deploy

四、在cluster-b部署submariner

（4.1）检查cluster-b是否需要删除submariner和nginx，如果需要删除，查看（2.3）和（2.4）

（4.2）# 查看broker-info.subm （base64是一种编码方式，-d表示decode），主要是查看subm文件有没有cluster-a的URL

cat broker-info.subm | base64 -d



（4.3）执行cordon命令禁用其他两个节点

kubectl get nodes

kubectl cordon <node name>

（4.4）# 把当前节点都标记为submariner的网关未来部署的节点

kubectl label node <node name> submariner.io/gateway=true

# 去除master节点的taint标记

kubectl taint nodes --all node-role.kubernetes.io/master-

（4.5）加入broker

subctl join broker-info.subm --clusterid cluster-b --natt=true

（4.6）安装nginx

# 先创建一个deployment对象，对象的名字叫做nginx，然后这个deployment对象运行nginx镜像

kubectl create deployment nginx --image=nginx

# 为deployment对象的nginx创建service，并通过Service的80端口转发至容器的80端口上。

kubectl expose deployment nginx --port=80

#使用subctl创建ServiceExport对象，使service可见，使得submariner中的其他集群可以发现这个service（执行这条指令可能会卡住大概10秒）

subctl export service --namespace default nginx

# 获取当前运行的namespace所有pods的信息（比如获取nginx的IP），包括pod运行在哪个节点上

kubectl get po -o wide

五、回到cluster-a，验证submariner能否正常工作

# 查看当前submariner的gateway是否安装在多个节点上

kubectl get ds -n submariner-operator

# 获取当前运行的所有cluster的信息（cluster-a上执行该指令），看看其他的cluster有没有加入进来

kubectl get clusters.submariner.io -A -w

# 如果你的cluster-a没安装tmp-shell（执行这条指令可能会卡住大概10秒）

kubectl -n default run tmp-shell --rm -i \

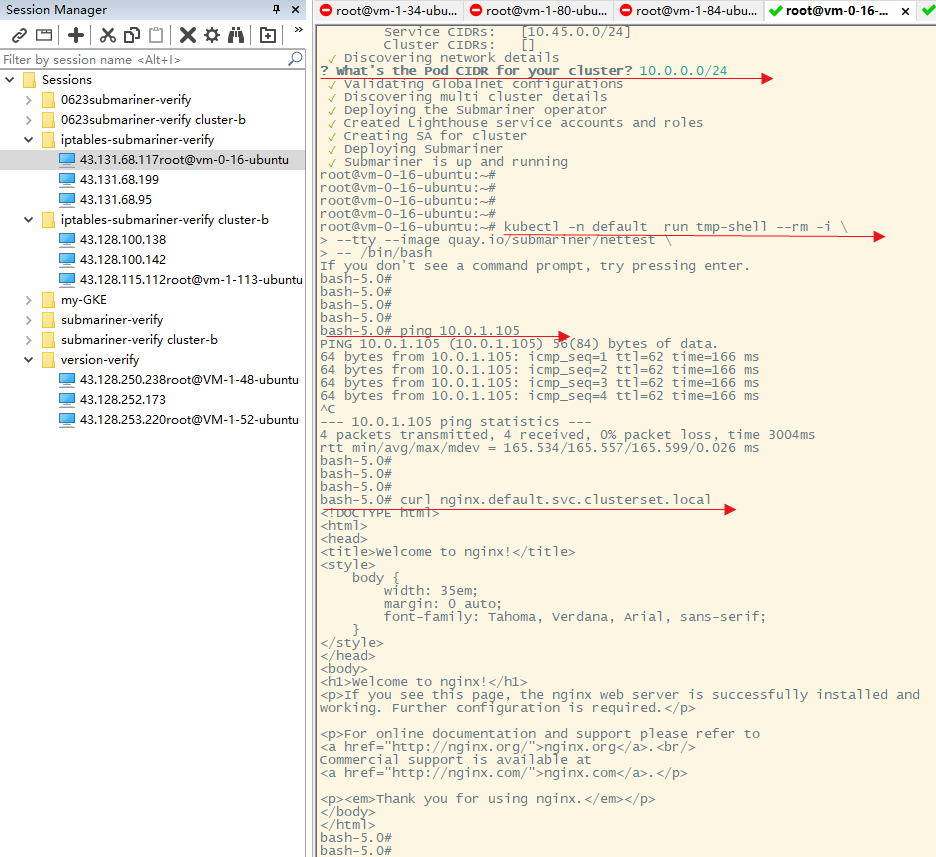
--tty --image quay.io/submariner/nettest \

-- /bin/bash

# 在cluster-a上检查能否连接上cluster-b的nginx这个pod

ping <cluster-b显示的nginx的IP>

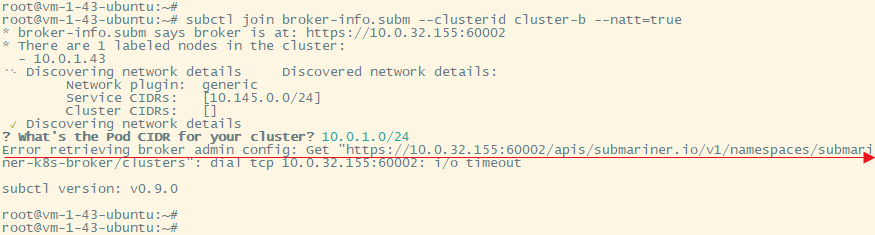
curl nginx.default.svc.clusterset.local



六、可能遇到的问题

（6.1）cluster-a能正确部署submariner，但是cluster-b无法加入cluster-a的submariner

检查步骤（2.5）



（6.2）secure CRT窗口的命令行终端字体变得百扁长，看起来很奇怪

打开secure CRT 🡪 选项 🡪 会话选项 🡪 终端 🡪 外观 🡪 字体

字体选择Lucia Console 字体大小选择10，保存确定

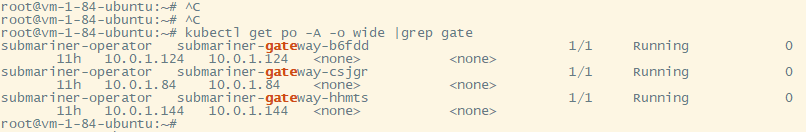
重新打开一个连接终端，看看字体是否修改成功，如果修改不成功，关闭secure CRT再打开就好了

（6.3）cluster-a正确部署submariner，cluster-a正确加入submariner的broker，cluster-b正确加入submariner的broker，但是cluster-a执行命令ping cluster-b上的nginx的pod执行失败

gateway的选举问题，检查submariner的gateway的日志文件

# debug 查看gateway的pod数目

kubectl get po -A -o wide | grep gateway



这里出现了3个gateway，一般情况下出现多个gateway会出现gateway选举，结果是只有1个gateway处于active状态，其余的gateway处于standby状态，如果你的ping请求传给standby状态的gateway，自然就ping不通了。所以在（3.2）中执行cordon命令禁用其他两个节点

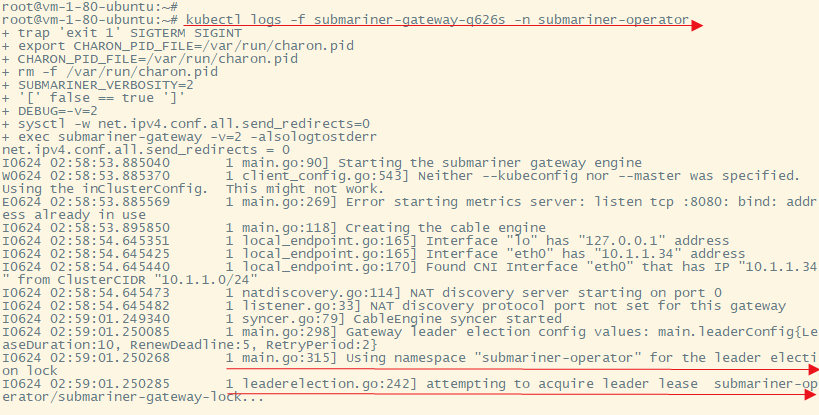


这样就是只有一个gateway，并且这个gateway一定处于活跃状态

# 查看日志

kubectl logs -f <pod的名字> -n <namespace的名字>

kubectl logs -f submariner-gateway-2rgz8 -n submariner-operator



出现这种十几二十行的日志说明这个gateway处于standby状态，如果gateway处于active状态那么它的日志将会非常的长

kubectl delete po submariner-gateway-fbwf8 -n submariner-operator

在公有云建立ipvs集群验证submariner

在第一步新建集群的时候选择高级，然后在里面选择ipvs

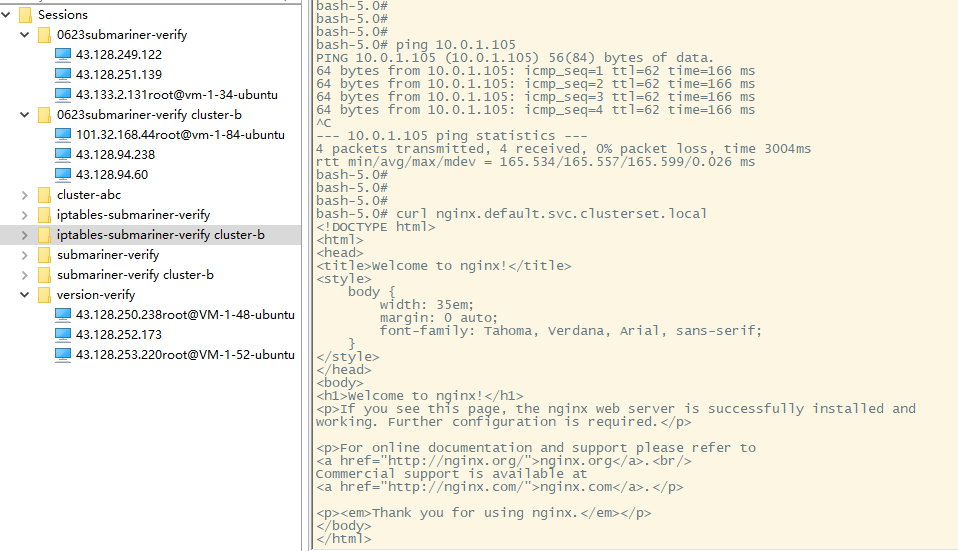


剩下的步骤ipvs和iptables步骤差不多

|  |  |  |  |
| --- | --- | --- | --- |
| ubuntu 18.04; VPC CIDR: 10.1.0.0/16; Tokyo; POD CIDR: 10.1.1.0/24; SERVICE CIDR: 10.55.0.0/24; ipvs | | | |
| ubuntu 18.04; VPC CIDR: 10.0.0.0/16; Singapore; POD CIDR: 10.0.1.0/24; SERVICE CIDR: 10.155.0.0/24; ipvs | | | |
| kubernetes version | pod/pod connectivity across cluster | service discovery | note |
| 1.18.4 | ok | ok | VPC-CNI; subctl v0.9.0 |

20210624复现iptables

|  |  |  |  |
| --- | --- | --- | --- |
| ubuntu 18.04; VPC CIDR: 10.0.0.0/16; Moscow; POD CIDR: 10.0.0.0/24; SERVICE CIDR: 10.45.0.0/24; iptables | | | |
| ubuntu 18.04; VPC CIDR: 10.0.0.0/16; Singapore; POD CIDR: 10.0.1.0/24; SERVICE CIDR: 10.145.0.0/24; iptables | | | |
| kubernetes version | pod/pod connectivity across cluster | service discovery | note |
| 1.18.4 | ok | ok | 独立集群部署；subctl v0.9.0 |



kubectl get po kube-apiserver-10.0.0.144 -n kube-system -o yaml

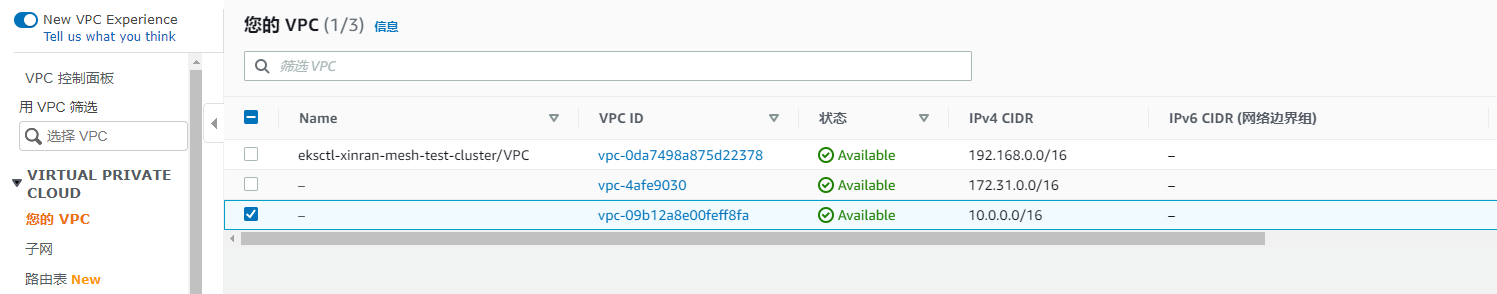
kubectl --kubeconfig /opt/config get po kube-proxy-gke-cluster-1-default-pool-d2a4431c-68l8 -n kube-system -o yaml

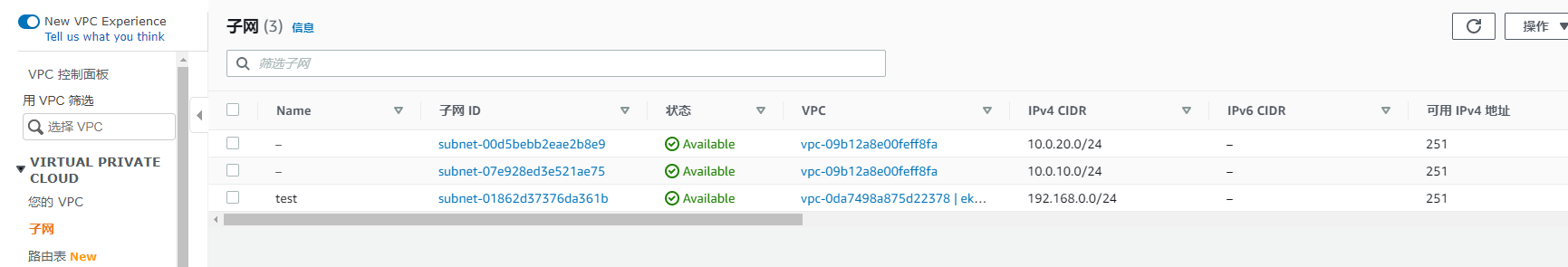
可能遇到的问题

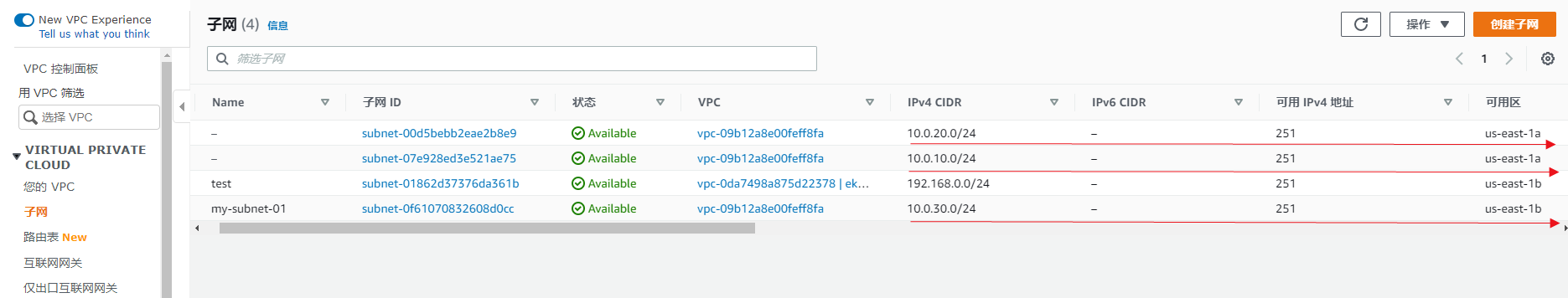
1. cordon other nodes (gateway election) 禁用其他节点

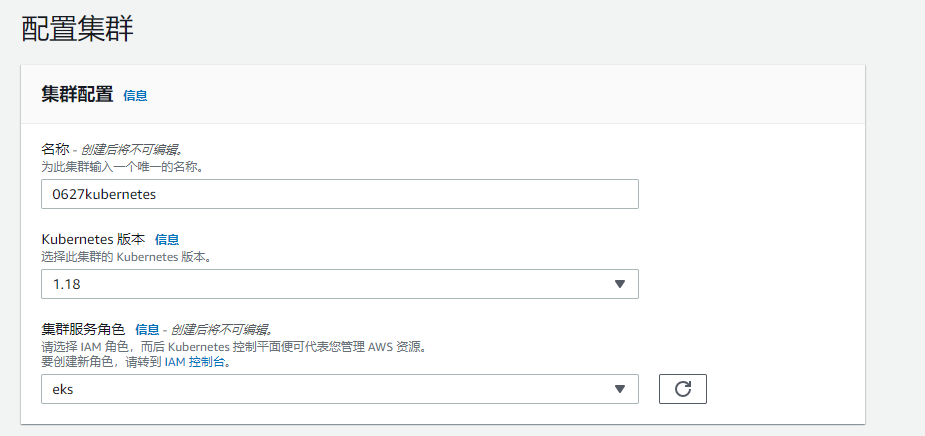
2. 8080 conflict on tke 端口冲突导致日志报错，中控台资源监控失效

3. todo













SSH 密钥对

eksctl-eksc1-nodegroup-ng-1802ec22-c4:b2:40:73:a3:da:07:f8:b1:fb:d0:79:2d:21:27:c1

