



下载APP

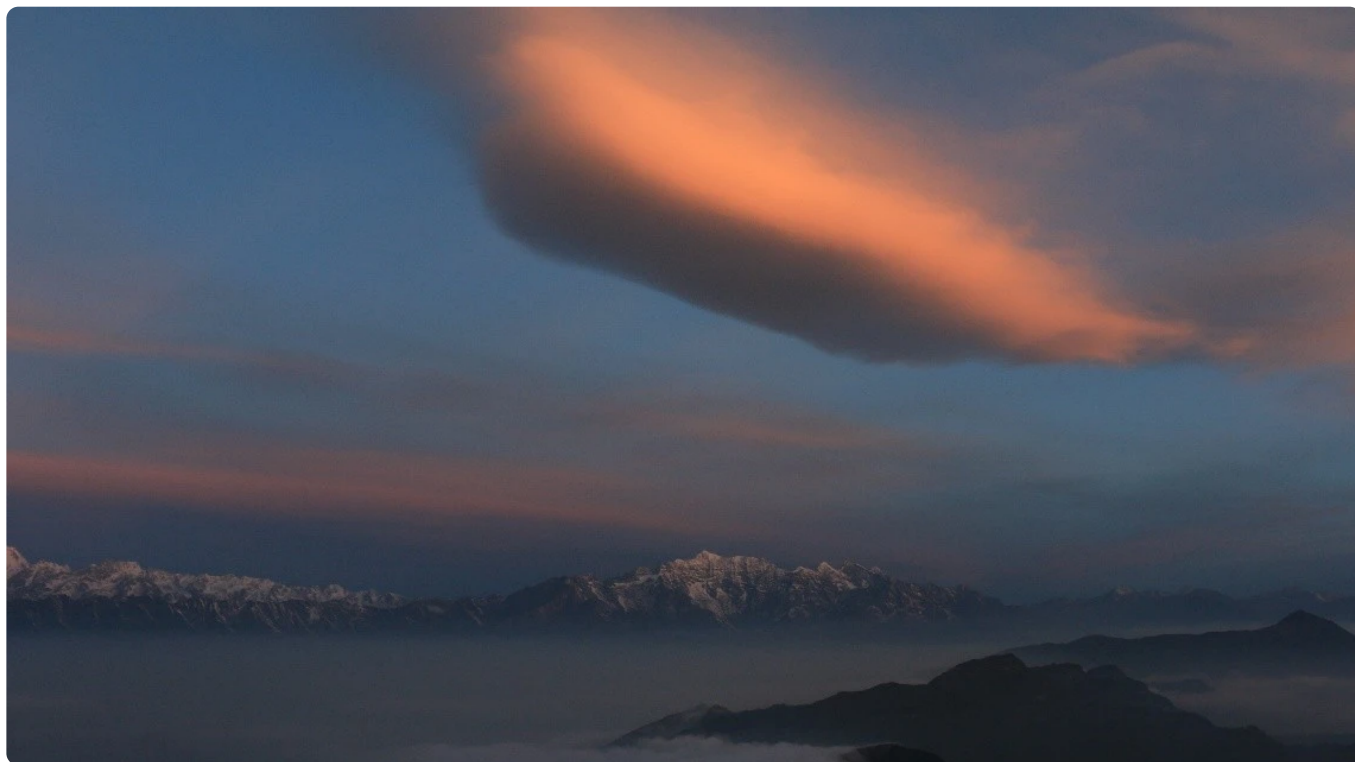


48 | 基于腾讯云 EKS 的容器化部署实战

2021-09-16 孔令飞

《Go 语言项目开发实战》

课程介绍 >



讲述：孔令飞

时长 15:44 大小 14.42M



你好，我是孔令飞。

在 [45 讲](#) 中，我介绍了一种基于 Kubernetes 的云原生架构设计方案。在云原生架构中，我们是通过 Docker + Kubernetes 来部署云原生应用的。那么这一讲，我就手把手教你如何在 Kubernetes 集群中部署好 IAM 应用。因为步骤比较多，所以希望你能跟着我完成每一个操作步骤。相信在实操的过程中，你也会学到更多的知识。

准备工作



在部署 IAM 应用之前，我们需要做以下准备工作：

1. 开通腾讯云容器服务镜像仓库。

2. 安装并配置 Docker。
3. 准备一个 Kubernetes 集群。

开通腾讯云容器服务镜像仓库

在 Kubernetes 集群中部署 IAM 应用，需要从镜像仓库下载指定的 IAM 镜像，所以首先需要有一个镜像仓库来托管 IAM 的镜像。我们可以选择将 IAM 镜像托管到 [DockerHub](#) 上，这也是 docker 运行时默认获取镜像的地址。

但因为 DockerHub 服务部署在国外，国内访问速度很慢。所以，我建议将 IAM 镜像托管在国内的镜像仓库中。这里我们可以选择腾讯云提供的镜像仓库服务，访问地址为 [容器镜像服务个人版](#)。

如果你已经有腾讯云的镜像仓库，可以忽略腾讯云镜像仓库开通步骤。

在开通腾讯云镜像仓库之前，你需要 [注册腾讯云账号](#)，并完成 [实名认证](#)。

开通腾讯云镜像仓库的具体步骤如下：

第一步，开通个人版镜像仓库。

1. 登录 [容器服务控制台](#)，选择左侧导航栏中的【镜像仓库】>【[个人版](#)】。
2. 根据以下提示，填写相关信息，并单击【**开通**】进行初始化。如下图所示：



用户名：默认是当前用户的账号 ID，是你登录到腾讯云 Docker 镜像仓库的身份，可在 [账号信息](#) 页面获取。

密码：是你登录到腾讯云 Docker 镜像仓库的凭证。

这里需要你记录用户名及密码，用于推送及拉取镜像。假如我们开通的镜像仓库，用户名为10000099xxxx，密码为iam59!z\$。

这里要注意，10000099xxxx**要替换成你镜像仓库的用户名。**

第二步，登录到腾讯云 Registry（镜像仓库）。

在我们开通完 Registry，就可以登录 Registry 了。可以通过以下命令来登录腾讯云 Registry：

```
1 $ docker login --username=[username] ccr.ccs.tencentyun.com
```

复制代码

这里的 username 是腾讯云账号 ID，开通时已注册，可在 [账号信息](#) 页面获取。docker 命令会在后面安装。

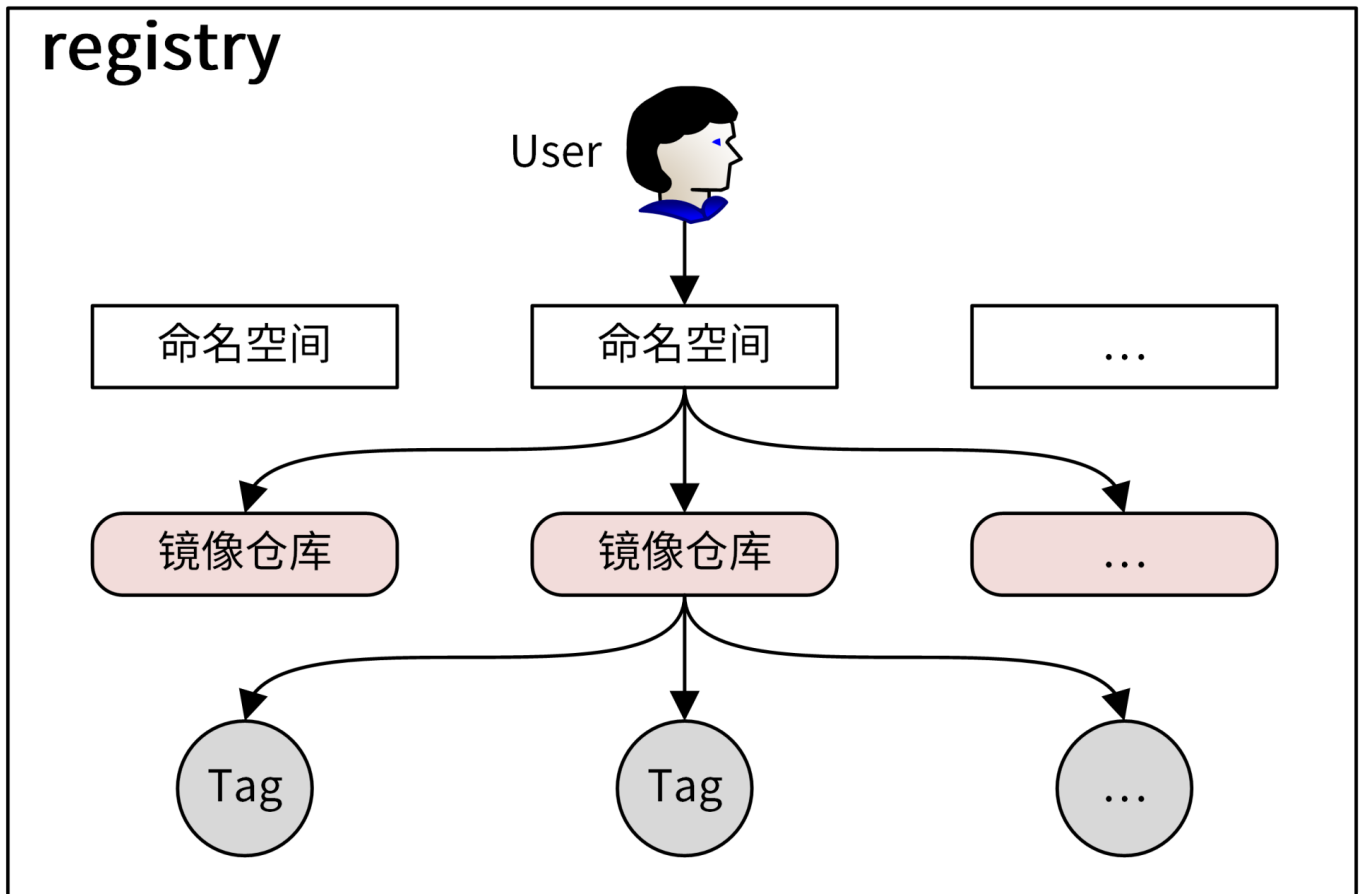
第三步，新建镜像仓库命名空间。

如果想使用镜像仓库，那么你需要首先创建一个用来创建镜像的命名空间。上一步，我们开通了镜像仓库，就可以在“命名空间”页签新建命名空间了，如下图所示：



上图中，我们创建了一个叫marmotedu的命名空间。

这里，镜像仓库服务、命名空间、镜像仓库、标签这几个概念你可能弄不清楚。接下来，我详细介绍下四者的关系，关系如下图所示：



先来看下我们使用镜像仓库的格式：<镜像仓库服务地址>/<命名空间>/<镜像仓库>:<标签>，例如`ccr.ccs.tencentyun.com/marmotedu/iam-apiserver-amd64:v1.1.0`。

如果想使用一个 Docker 镜像，我们首先需要开通一个镜像仓库服务（Registry），镜像仓库服务都会对外提供一个固定的地址供你访问。在 Registry 中，我们（User）可以创建一个或多个命名空间（Namespace），命名空间也可以简单理解为镜像仓库逻辑上的一个分组。

接下来，就可以在 Namespace 中创建一个或多个镜像仓库，例如`iam-apiserver-amd64`、`iam-authz-server-amd64`、`iam-pump-amd64`等。针对每一个镜像仓库，又可以创建多个标签（Tag），例如`v1.0.1`、`v1.0.2`等。


<镜像仓库>:<标签>又称为镜像。镜像又分为私有镜像和公有镜像，公有镜像可供所有能访问 Registry 的用户下载使用，私有镜像只提供给通过授权的用户使用。

安装 Docker

开通完镜像仓库之后，我们还需要安装 Docker，用来构建和测试 Docker 镜像。下面我来讲解下具体的安装步骤。

第一步，安装 Docker 前置条件检查。

需要确保 CentOS 系统启用了centos-extras yum 源，默认情况下已经启用，检查方式如下：

 复制代码

```
1 $ cat /etc/yum.repos.d/CentOS-Extras.repo
2 # Qcloud-Extras.repo
3
4 [extras]
5 name=Qcloud-$releasever - Extras
6 baseurl=http://mirrors.tencentyun.com/centos/$releasever/extras/$basearch/os/
7 gpgcheck=1
8 enabled=1
9 gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-Qcloud-8
```

如果/etc/yum.repos.d/CentOS-Extras.repo文件存在，且文件中extras部分的enabled配置项值为1，说明已经启用了centos-extras yum 源。如果/etc/yum.repos.d/CentOS-Extras.repo 文件不存在，或者enabled 不为 1，则需要创建/etc/yum.repos.d/CentOS-Extras.repo 文件，并将上述内容复制进去。

第二步，安装 docker。

Docker 官方文档 [🔗 Install Docker Engine on CentOS](#)提供了 3 种安装方法：


通过 Yum 源安装。

通过 RPM 包安装

通过脚本安装。

这里，我们选择最简单的安装方式：**通过 Yum 源安装**。它具体又分为下面 3 个步骤。


1. 安装 docker。

 复制代码

```
1 $ sudo yum install -y yum-utils # 1. 安装 `yum-utils` 包, 该包提供了 `yum-config-  
2 $ sudo yum-config-manager --add-repo https://download.docker.com/linux/centos/  
3 $ sudo yum-config-manager --enable docker-ce-nightly docker-ce-test # 3. 启用 `  
4 $ sudo yum install -y docker-ce docker-ce-cli containerd.io # 4. 安装最新版本的
```


2. 启动 docker。

可以通过以下命令来启动 docker：

 复制代码

```
1 $ sudo systemctl start docker
```

docker 的配置文件是 `/etc/docker/daemon.json`，这个配置文件默认是没有的，需要我们手动创建：

 复制代码

```
1 $ sudo tee /etc/docker/daemon.json << EOF  
2 {  
3     "bip": "172.16.0.1/24",  
4     "registry-mirrors": [],  
5     "graph": "/data/lib/docker"  
6 }  
7 EOF
```


这里，我来解释下常用的配置参数。

`registry-mirrors`：仓库地址，可以根据需要修改为指定的地址。

`graph`：镜像、容器的存储路径，默认是 `/var/lib/docker`。如果你的 `/` 目录存储空间满足不了需求，需要设置 `graph` 为更大的目录。

`bip`：指定容器的 IP 网段。

配置完成后，需要重启 Docker：

 复制代码

```
1 $ sudo systemctl restart docker
```

3. 测试 Docker 是否安装成功。

[复制代码](#)

```
1 $ sudo docker run hello-world
2 Unable to find image 'hello-world:latest' locally
3 latest: Pulling from library/hello-world
4 b8dfde127a29: Pull complete
5 Digest: sha256:0fe98d7debd9049c50b597ef1f85b7c1e8cc81f59c8d623fcb2250e8bec85b3
6 Status: Downloaded newer image for hello-world:latest
7 ...
8 Hello from Docker!
9 This message shows that your installation appears to be working correctly.
10 ....
```

`docker run hello-world`命令会下载hello-world镜像，并启动容器，打印安装成功提示信息后退出。

这里注意，如果你通过 Yum 源安装失败，可以尝试 Docker 官方文档 [Install Docker Engine on CentOS](#)提供的其他方式安装。

第三步，安装后配置。

安装成功后，我们还需要做一些其他配置。主要有两个，一个是配置 docker，使其可通过 non-root 用户使用，另一个是配置 docker 开机启动。

1. 使用non-root用户操作 Docker

我们在 Linux 系统上操作，为了安全，需要以普通用户的身份登录系统并执行操作。所以，我们需要配置 docker，使它可以被non-root用户使用。具体配置方法如下：

[复制代码](#)

```
1 $ sudo groupadd docker # 1. 创建`docker`用户组
2 $ sudo usermod -aG docker $USER # 2. 将当前用户添加到`docker`用户组下
3 $ newgrp docker # 3. 重新加载组成员身份
4 $ docker run hello-world # 4. 确认能够以普通用户使用docker
```

如果在执行 `sudo groupadd docker` 时报 `groupadd: group 'docker' already exists` 错误，说明 `docker` 组已经存在了，可以忽略这个报错。

如果你在将用户添加到 `docker` 组之前，使用 `sudo` 运行过 `docker` 命令，你可能会看到以下错误：

[复制代码](#)

```
1 WARNING: Error loading config file: /home/user/.docker/config.json -
2 stat /home/user/.docker/config.json: permission denied
```

这个错误，我们可以通过删除 `~/.docker/` 目录来解决，或者通过以下命令更改 `~/.docker/` 目录的所有者和权限：

[复制代码](#)

```
1 $ sudo chown "$USER":"$USER" /home/"$USER"/.docker -R
2 $ sudo chmod g+rxw "$HOME/.docker" -R
```

2. 配置 docker 开机启动

大部分 Linux 发行版（RHEL、CentOS、Fedora、Debian、Ubuntu 16.04 及更高版本）使用 `systemd` 来管理服务，包括指定开启时启动的服务。在 Debian 和 Ubuntu 上，Docker 默认配置为开机启动。

在其他系统，我们需要手动配置 Docker 开机启动，配置方式如下（分别需要配置 `docker` 和 `containerd` 服务）：

要在引导时为其他发行版自动启动 Docker 和 Containerd，你可以使用以下命令：

[复制代码](#)

```
1 $ sudo systemctl enable docker.service # 设置 docker 开机启动
2 $ sudo systemctl enable containerd.service # 设置 containerd 开机启动
```

如果要禁止 `docker`、`containerd` 开启启动，可以用这个命令：

复制代码

```
1 $ sudo systemctl disable docker.service # 禁止 docker 开机启动
2 $ sudo systemctl disable containerd.service # 禁止 containerd 开机启动
```

准备一个 Kubernetes 集群

安装完 Docker 之后，我们还需要一个 Kubernetes 集群，来调度 docker 容器。安装 Kubernetes 集群极其复杂，这里选择一种最简单的方式来准备一个 Kubernetes 集群：购买一个腾讯云弹性集群（EKS）。

如果你想自己搭建 Kubernetes 集群，这里建议你购买 3 台腾讯云 CVM 机器，并参照 [follow-me-install-kubernetes-cluster](#) 教程来一步步搭建 Kubernetes 集群，CVM 机器建议的最小配置如下：

配置	OS	用途
1核4G	CentOS Linux release 8.2.2004 (Core)	Kubernetes Master、Worker节点
1核4G	CentOS Linux release 8.2.2004 (Core)	Kubernetes Master、Worker节点
1核4G	CentOS Linux release 8.2.2004 (Core)	Kubernetes Master、Worker节点

EKS 简介

我先简单介绍一下 EKS 是什么。EKS（Elastic Kubernetes Service）即腾讯云弹性容器服务，是腾讯云容器服务推出的无须用户购买节点即可部署工作负载的服务模式。它完全兼容原生的 Kubernetes，支持使用原生方式创建及管理资源，按照容器真实的资源使用量计费。弹性容器服务 EKS 还扩展支持腾讯云的存储及网络等产品，同时确保用户容器的安全隔离，开箱即用。

EKS 费用

那它是如何收费呢？EKS 是全托管的 Serverless Kubernetes 服务，不会收取托管的 Master、etcd 等资源费用。弹性集群内运行的工作负载采用后付费的按量计费模式，费用根据实际配置的资源量按使用时间计算。也就是说：**Kubernetes 集群本身是免费的，只有运行工作负载，消耗节点资源时收费。**

EKS 有 3 种计费模式：预留券、按量计费、竞价模式，这里我建议选择**按量计费**。按量计费，支持按秒计费，按小时结算，随时购买随时释放，从专栏学习的角度来说，费用是最低的。EKS 会根据工作负载申请的 CPU、内存数值以及工作负载的运行时间来核算费用，具体定价，你可以参考：[🔗 定价|弹性容器服务](#)。

这里我通过例子来说明一下费用问题，IAM 应用会部署 4 个 Deployment，每个 Deployment 一个副本：

iam-apiserver：IAM REST API 服务，提供用户、密钥、策略资源的 CURD 功能的 API 接口。


iam-authz-server：IAM 资源授权服务，对外提供资源授权接口。

iam-pump：IAM 数据清洗服务，从 Redis 中获取授权日志，处理后保存在 MongoDB 中。

iamctl：IAM 应用的测试服务，登陆 iamctl Pod 可以执行 iamctl 命令和 smoke 测试脚本，完成对 IAM 应用的运维和测试。

上述 4 个 Deployment 中的 **Pod 配置均为 0.25 核、512Mi 内存。**

这里，我们根据 EKS 的费用计算公式 $\text{费用} = \text{相关计费项配置} \times \text{资源单位时间价格} \times \text{运行时间}$ 计算 IAM 部署一天的费用：

 复制代码

```
1 总费用 = (4 × 1) × (0.25 × 0.12 + 0.5 × 0.05) × 24 = 4.8 元
```

也就是按最低配置部署 IAM 应用，运行一天的费用是 4.8 元（一瓶水的钱，就能学到如何将 IAM 应用部署在 Kubernetes 平台上，很值！）。你可能想这个计算公式里每个数值都代表什么呢？我来解释一下，其中：

- (4 x 1) : Kubernetes Pod 总个数 (一共是 4 个 Deployment , 每个 Pod 1 个副本) 。
- 0.25 x 0.12 : 连续运行 1 小时的 CPU 配置费用。
- 0.5 x 0.05 : 连续运行 1 小时的内存配置费用。
- 24 : 24 小时 , 也即一天。

这里需要注意 , 为了帮助你节省费用 , 上述配置都是最低配置。在实际生产环境中 , 建议的配置如下 :

组件	配置	副本数
iam-apiserver	4核8G	2
iam-authz-server	8核16G	2
iam-pump	4核8G	1
iamctl	0.5核1G	1

因为 iam-pump 组件是有状态的 , 并且目前没有实现抢占机制 , 所以副本数需要设置为 1。

另外 , Intel 按量计费的配置费用见下图 :

在这里有个很重要的事情提醒你 : **学完本节课 , 销毁这些 Deployment , 避免被继续扣费。**

申请 EKS 集群

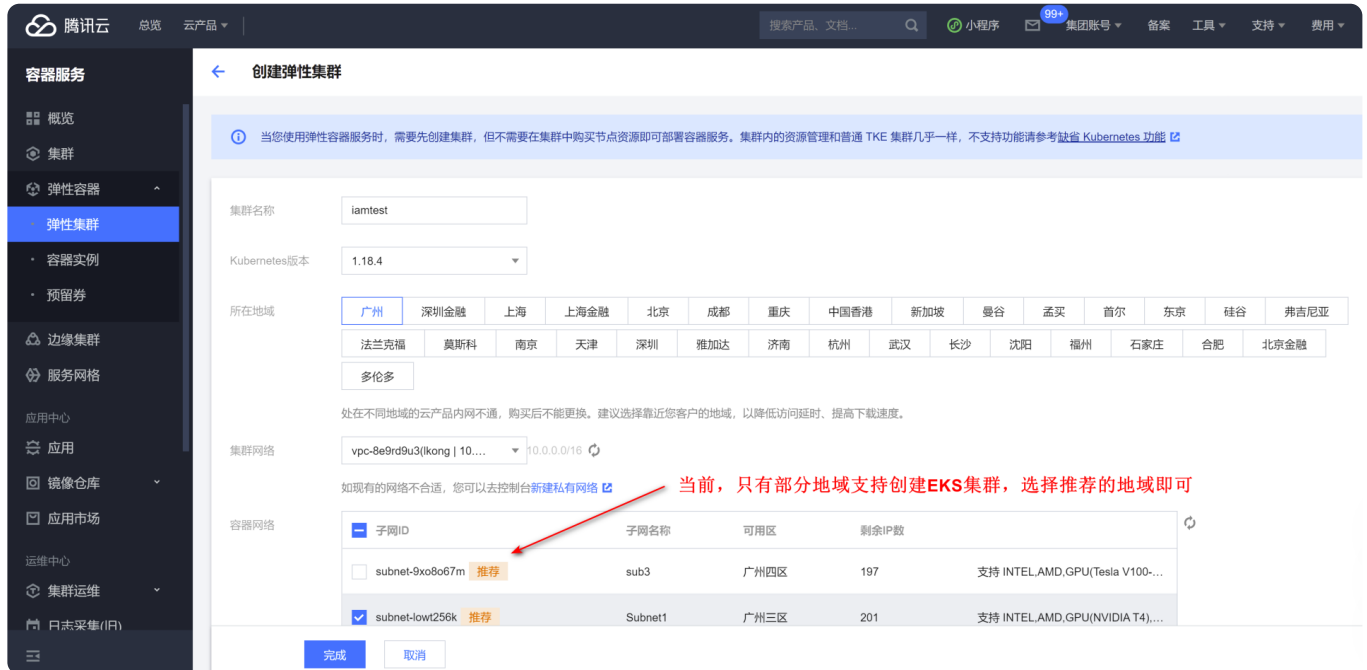
了解了 EKS 以及费用相关的问题 , 接下来我们看看如何申请 EKS 集群。你可以通过以下 5 步来申请 EKS 集群。在正式申请前 , 请先确保腾讯云账户有大于 **10 元**的账户余额 , 否则在创建和使用 EKS 集群的过程中可能会因为费用不足而报错。

1. 创建腾讯云弹性集群

具体步骤如下：

首先，登录容器服务控制台，选择左侧导航栏中的【[弹性集群](#)】。

然后，在页面上方选择需创建弹性集群的地域，并单击【新建】。在“创建弹性集群”页面，根据以下提示设置集群信息。如下图所示：



页面中各选择项的意思，我来给你解释一下：

集群名称：创建的弹性集群名称，不超过 60 个字符。

Kubernetes 版本：弹性集群支持 1.12 以上的多个 Kubernetes 版本选择，建议选择最新的版本。

所在地域：建议你根据所在地理位置选择靠近的地域，可降低访问延迟，提高下载速度。

集群网络：已创建的弹性集群 VPC 网络，你可以选择私有网络中的子网用于弹性集群的容器网络，详情请见 [私有网络（VPC）](#)。

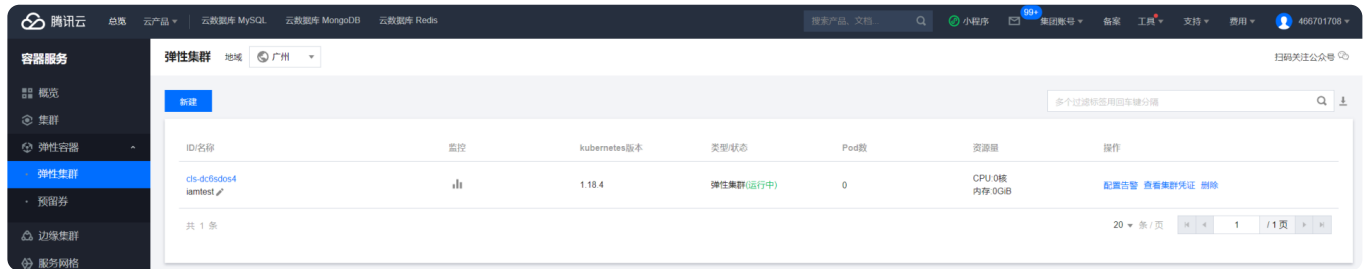
容器网络：为集群内容器分配在容器网络地址范围内的 IP 地址。弹性集群的 Pod 会直接占用 VPC 子网 IP，请尽量选择 IP 数量充足且与其他产品使用无冲突的子网。

Service CIDR：集群的 ClusterIP Service 默认分配在所选 VPC 子网中，请尽量选择 IP 数量充足且与其他产品使用无冲突的子网。

集群描述：创建集群的相关信息，该信息将显示在“集群信息”页面。

设置完成后，单击【完成】即可开始创建，可在“弹性集群”列表页面查看集群的创建进度。

等待弹性集群创建完成，创建完成后的弹性集群页面如下图所示：



我们创建的弹性集群 ID 为 **cls-dc6sdos4**。

2. 开启外网访问

如果想访问 EKS 集群，需要先开启 EKS 的外网访问能力，开启方法如下：

登录容器服务控制台 -> 选择左侧导航栏中的【[弹性集群](#)】 -> 进入 **cls-dc6sdos4** 集群的详情页中 -> 选择【基本信息】 -> 点击【外网访问】按钮。如下图所示：

这里要注意，开启外网访问时，为了安全，需要设置允许访问 kube-apiserver 的 IP 段。为了避免不必要的错误，外网访问地址我们设置为 `0.0.0.0/0`。如下图所示：

外网访问设置



开启外网访问, 会将集群apiserver暴露公网, 请谨慎操作, 需配置来源授权, 默认全拒绝, 可配置放通单个IP或CIDR, 强烈不建议配置0.0.0.0/0放通全部来源

外网访问地址

0.0.0.0/0

保存

取消

注意, 只有测试时才可这么设置为 0.0.0.0/0, 如果是生产环境, 建议严格限制可以访问 kube-apiserver 的来源 IP。

3. 安装 kubectl 命令行工具

如果要访问 EKS (标准的 Kubernetes 集群), 比较高效的方式是通过 Kubernetes 提供的命令行工具 kubectl 来访问。所以, 还需要安装 kubectl 工具。

安装方式如下:

复制代码

```
1 $ curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/s
2 $ mkdir -p $HOME/bin
3 $ mv kubectl $HOME/bin
4 $ chmod +x $HOME/bin/kubectl
```

具体可参考 [安装和设置 kubectl](#)。

你可以通过以下命令来配置 kubectl 的 bash 自动补全:

复制代码

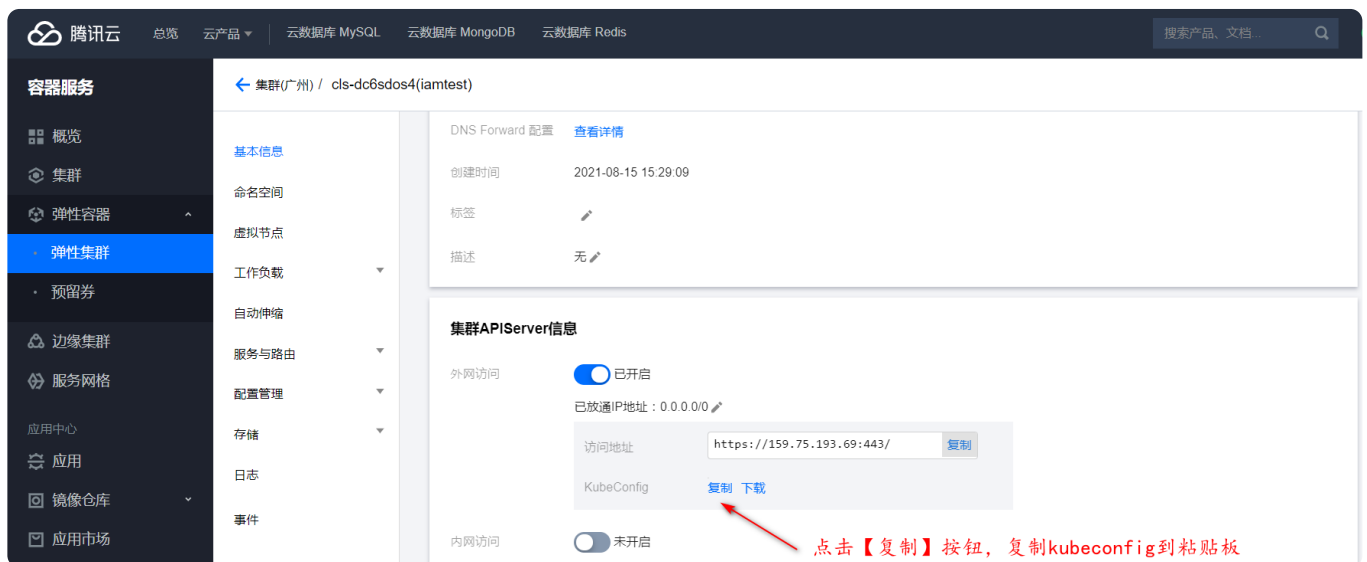
```
1 $ kubectl completion bash > $HOME/.kube-completion.bash
2 $ echo 'source $HOME/.kube-completion.bash' >> ~/.bashrc
3 $ bash
```

4. 下载并安装 kubeconfig

安装完 kubectl 工具之后，需要配置 kubectl 所读取的配置文件。

这里注意，在上一步，我们开启了外网访问，开启后 EKS 会生成一个 kubeconfig 配置（kubeconfig 即为 kubectl 的配置文件）。我们可以从页面下载并安装。

在弹性集群的基本信息页面，点击【复制】按钮，复制 kubeconfig 文件内容，如下图所示：



复制后，将粘贴板的内容保存在 `$HOME/.kube/config` 文件中。需要先执行 `mkdir -p $HOME/.kube` 创建 `.kube` 目录，再将粘贴版中的内容写到 `config` 文件中。

你可以通过以下命令，来测试 kubectl 工具是否成功安装和配置：

```
1 $ kubectl get nodes
2 NAME                                STATUS    ROLES    AGE    VERSION
3 eklet-subnet-lowt256k               Ready    <none>    2d1h   v2.5.21
```

复制代码

如果输出了 Kubernetes 的 eklet 节点，并且节点状态为 **Ready**，说明 Kubernetes 集群运行正常，并且 kubectl 安装和配置正确。

5. EKS 集群开通集群内服务访问外网能力

因为 IAM 应用中的数据库：MariaDB、Redis、MongoDB 可能需要通过外网访问，所以还需要开通 EKS 中 Pod 访问外网的能力。

EKS 支持通过配置 [NAT 网关](#) 和 [路由表](#) 来实现集群内服务访问外网。具体开启步骤，需要你查看腾讯云官方文档：[通过 NAT 网关访问外网](#)。

在开通过程中有以下两点需要你注意：

在**创建指向 NAT 网关的路由表**步骤中，目的端要选择：0.0.0.0/0。

在**关联子网至路由表**步骤中，只关联创建 EKS 集群时选择的子网。

如果你的数据库需要通过外网访问，这里一定要确保 EKS 集群成功开通集群内服务访问外网能力，否则部署 IAM 应用时会因为访问不了数据库而失败。

安装 IAM 应用

上面，我们开通了镜像仓库、安装了 Docker 引擎、安装和配置了 Kubernetes 集群，那么接下来，我们就来看下如何将 IAM 应用部署到 Kubernetes 集群中。

假设 IAM 项目仓库根目录路径为 `$IAM_ROOT`，具体安装步骤如下：

1. 配置 `scripts/install/environment.sh`

`scripts/install/environment.sh` 文件中包含了各类自定义配置。你可能需要配置跟数据库相关的配置（当然，也可以都使用默认值）：

MariaDB 配置：`environment.sh` 文件中以 `MARIADB_` 开头的变量。

Redis 配置：`environment.sh` 文件中以 `REDIS_` 开头的变量。

MongoDB 配置：`environment.sh` 文件中以 `MONGO_` 开头的变量。

其他配置，使用默认值。

2. 创建 IAM 应用的配置文件

[复制代码](#)

```
1 $ cd ${IAM_ROOT}
2 $ make gen.defaultconfigs # 生成iam-apiserver、iam-authz-server、iam-pump、iamctl
3 $ make gen.ca # 生成 CA 证书
```

上述命令会将 IAM 的配置文件存放在这个`\${IAM_ROOT}/_output/configs/`目录下。

3. 创建 IAM 命名空间

我们将 IAM 应用涉及到的各类资源，都创建在iam命名空间中。将 IAM 资源创建在独立的命名空间中，不仅方便维护，还可以有效避免影响其他 Kubernetes 资源。

[复制代码](#)

```
1 $ kubectl create namespace iam
```

4. 将 IAM 各服务的配置文件，以 ConfigMap 资源的形式保存在 Kubernetes 集群中

[复制代码](#)

```
1 $ kubectl -n iam create configmap iam --from-file=${IAM_ROOT}/_output/configs/
2 $ kubectl -n iam get configmap iam
3 NAME      DATA      AGE
4 iam       4          13s
```


执行`kubectl -n iam get configmap iam`命令，可以成功获取创建的iam configmap。

如果你觉得每次执行kubectl命令都要指定`-n iam`选项很繁琐，你可以使用以下命令，将kubectl上下文环境中的命名空间指定为iam。设置后，执行kubectl命令，默认在iam命名空间下执行：

[复制代码](#)

```
1 $ kubectl config set-context `kubectl config current-context` --namespace=iam
```

5. 将 IAM 各服务使用的证书文件，以 ConfigMap 资源的形式创建在 Kubernetes 集群中

 复制代码

```
1 $ kubectl -n iam create configmap iam-cert --from-file=${IAM_ROOT}/_output/cer
2 $ kubectl -n iam get configmap iam-cert
3 NAME      DATA      AGE
4 iam-cert  14         12s
```

执行 `kubectl -n iam get configmap iam-cert` 命令，可以成功获取创建的 `iam-cert` configmap。

6. 创建镜像仓库访问密钥

在准备阶段，我们开通了腾讯云镜像仓库服务（访问地址为 `ccr.ccs.tencentyun.com`），并创建了用户 `10000099xxxx`，其密码为 `iam59!z$`。


接下来，我们就可以创建 `docker-registry secret`。Kubernetes 在下载 Docker 镜像时，需要 `docker-registry secret` 来进行认证。创建命令如下：

 复制代码

```
1 $ kubectl -n iam create secret docker-registry ccr-registry --docker-server=cc
```


7. 创建 Docker 镜像，并 Push 到镜像仓库

将镜像 Push 到 CCR 镜像仓库，需要确保你已经登录到腾讯云 CCR 镜像仓库，如果没登录，可以执行以下命令来登录：

 复制代码

```
1 $ docker login --username=[username] ccr.ccs.tencentyun.com
```

执行 `make push` 命令构建镜像，并将镜像 Push 到 CCR 镜像仓库：

 复制代码

```
1 $ make push REGISTRY_PREFIX=ccr.ccs.tencentyun.com/marmotedu VERSION=v1.1.0
```

上述命令，会构建 iam-apiserver-amd64、iam-authz-server-amd64、iam-pump-amd64、iamctl-amd64 四个镜像，并将这些镜像 Push 到腾讯云镜像仓库的 marmotedu 命名空间下。

构建的镜像如下：

[复制代码](#)

```
1 $ docker images|grep marmotedu
2 ccr.ccs.tencentyun.com/marmotedu/iam-pump-amd64          v1.1.0    e078d340e3f
3 ccr.ccs.tencentyun.com/marmotedu/iam-apiserver-amd64     v1.1.0    5e90b67cc94
4 ccr.ccs.tencentyun.com/marmotedu/iam-authz-server-amd64  v1.1.0    6796b02be68
5 ccr.ccs.tencentyun.com/marmotedu/iamctl-amd64            v1.1.0    320a77d525e
```

8. 修改 \${IAM_ROOT}/deployments/iam.yaml 配置

这里请你注意，如果在上一个步骤中，你构建的镜像 tag 不是 v1.1.0，那么你需要修改 \${IAM_ROOT}/deployments/iam.yaml 文件，并将 iam-apiserver-amd64、iam-authz-server-amd64、iam-pump-amd64、iamctl-amd64 镜像的 tag 修改成你构建镜像时指定的 tag。

9. 部署 IAM 应用

[复制代码](#)

```
1 $ kubectl -n iam apply -f ${IAM_ROOT}/deployments/iam.yaml
```

执行上述命令，会在 iam 命名空间下，创建一系列 Kubernetes 资源，可以使用以下命令，来获取这些资源的状态：

[复制代码](#)

```
1 $ kubectl -n iam get all
2 NAME                                READY   STATUS    RESTARTS   AGE
3 pod/iam-apiserver-d8dc48596-wkhp1   1/1     Running   0           94m
4 pod/iam-authz-server-6bc899c747-fbpbk 1/1     Running   0           94m
5 pod/iam-pump-7dcbfd4f59-2w9vk        1/1     Running   0           94m
6 pod/iamctl-6fc46b8ccb-gs62l         1/1     Running   1           98m
```

7	NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)
8	service/iam-apiserver	ClusterIP	192.168.0.174	<none>	8443/TCP,
9	service/iam-authz-server	ClusterIP	192.168.0.76	<none>	9443/TCP,
10	service/iam-pump	ClusterIP	192.168.0.155	<none>	7070/TCP
11					
12	NAME	READY	UP-TO-DATE	AVAILABLE	AGE
13	deployment.apps/iam-apiserver	1/1	1	1	101m
14	deployment.apps/iam-authz-server	1/1	1	1	101m
15	deployment.apps/iam-pump	1/1	1	1	101m
16	deployment.apps/iamctl	1/1	1	1	101m
17					
18	NAME	DESIRED	CURRENT	READY	AGE
19	replicaset.apps/iam-apiserver-d8dc48596	1	1	1	101m
20	replicaset.apps/iam-authz-server-6bc899c747	1	1	1	101m
21	replicaset.apps/iam-pump-7dcbfd4f59	1	1	1	101m
22	replicaset.apps/iamctl-6fc46b8ccb	1	1	1	101m
23					

我们看到pod/iam-apiserver-d8dc48596-wkhpl、pod/iam-authz-server-6bc899c747-fbpbk、pod/iam-pump-7dcbfd4f59-2w9vk、pod/iamctl-6fc46b8ccb-gs62l 4 个 Pod 都处在Running状态，说明服务都成功启动。

测试 IAM 应用

我们在iam命令空间下创建了一个测试 Deployment iamctl。你可以登陆iamctl Deployment 所创建出来的 Pod，执行一些运维操作和冒烟测试。登陆命令如下：

```
1 $ kubectl -n iam exec -it `kubectl -n iam get pods -l app=iamctl | awk '/iamctl`
```

 复制代码

登陆到iamctl-xxxxxxxxxx-xxxxx Pod 中后，就可以执行运维操作和冒烟测试了。

1. 运维操作

在 iamctl 容器中，你可以使用 iamctl 工具提供的各类功能，iamctl 以子命令的方式对外提供功能。命令执行效果见下图：

```
[root@iamctl-6fc46b8ccb-f4v6w install]# iamctl user list
NAME      NICKNAME  EMAIL                PHONE      CREATED                UPDATED
foo1234   foo1234   aa@qq.com            2021-06-27 21:18:00  2021-06-27 21:18:01
admin     admin     admin@foxmail.com    1812884xxxx 2021-05-27 18:01:40  2021-05-06 05:13:14
[root@iamctl-6fc46b8ccb-f4v6w install]# iamctl secret create foo
secret/foo created
[root@iamctl-6fc46b8ccb-f4v6w install]# iamctl secret list
NAME      SECRETID  SECRETKEY             EXPIRES      CREATED
foo       gCtDwsqevwFX2BIaFuYy79lt4GfLYWnpFfF  1xfyIzguVXHC3H4UB5Chj8bpqBlpBIAz  2021-08-21 23:34:56  2021-08-15 23:34:56
[root@iamctl-6fc46b8ccb-f4v6w install]# iamctl secret delete foo
secret/foo deleted
[root@iamctl-6fc46b8ccb-f4v6w install]# iamctl secret list
NAME      SECRETID  SECRETKEY             EXPIRES      CREATED
```

2. 冒烟测试

 复制代码

```
1 # cd /opt/iam/scripts/install
2 # ./test.sh iam::test::smoke
```


如果 `./test.sh iam::test::smoke` 命令，打印的输出中，最后一行为 `congratulations, smoke test passed!` 字符串，说明 IAM 应用安装成功。如下图所示：

```
{"metadata":{"id":97,"instanceID":"policy-jzx4nq","name":"policy0","createdAt":"2021-08-15T23:36:10+08:00","updatedAt":"2021-08-15T23:36:10.979+08:00"},"username":"admin","policy":{"id":"","description":"One policy to rule them all(modified).","subjects":["users:\u003cpeter|ken\u003e","users:maria","groups:admins"],"effect":"allow","resources":["resources:articles:\u003c.*\u003e","resources:printer"],"actions":["delete","\u003ccreate|update\u003e"],"conditions":{"remoteIPAddress":{"type":"CIDRCondition","options":{"cidr":"192.168.0.1/16"}}},"meta":null}}
null
congratulations, /v1/policy test passed!
congratulations, iam-apiserver test passed!
null
{"metadata":{"id":98,"instanceID":"policy-3xpp31","name":"authzpolicy","createdAt":"2021-08-15T23:36:11.344+08:00","updatedAt":"2021-08-15T23:36:11.359+08:00"},"username":"admin","policy":{"id":"","description":"One policy to rule them all.","subjects":["users:\u003cpeter|ken\u003e","users:maria","groups:admins"],"effect":"allow","resources":["resources:articles:\u003c.*\u003e","resources:printer"],"actions":["delete","\u003ccreate|update\u003e"],"conditions":{"remoteIPAddress":{"type":"CIDRCondition","options":{"cidr":"192.168.0.1/16"}}},"meta":null}}
null
congratulations, /v1/authz test passed!
congratulations, iam-authz-server test passed!
congratulations, iam-pump test passed!
congratulations, iamctl test passed!
congratulations, smoke test passed!
```

销毁 EKS 集群及其资源

好了，到这里，你已经成功在 EKS 集群中部署了 IAM 应用，EKS 的使命也就完成了。接下来，为避免账户被持续扣费，需要删除 EKS 内的资源和集群。

1. 删除 EKS 内创建的 IAM 资源

 复制代码

```
1 $ kubectl delete namespace iam
```

因为删除 Namespace 会删除 Namespace 下的所有资源，所以上述命令执行时间会久点。

2. 删除 EKS 集群

首先，登录容器服务控制台，选择左侧导航栏中的【[弹性集群](#)】。

然后，选择创建的 EKS 集群：cls-dc6sdos4，点击最右侧的【删除】按钮，删除 EKS 集群。如下图所示：



总结

云原生架构设计中，需要将 IAM 应用部署到 Kubernetes 集群中。所以，首先需要你准备一个 Kubernetes 集群。你可以自己购买腾讯云 CVM 机器搭建 Kubernetes 集群，但这种方式费用高、操作复杂。所以，我建议你直接申请一个 EKS 集群来部署 IAM 应用。

EKS 集群是一个标准的 Kubernetes 集群，可以快速申请，并免运维。EKS 集群只收取实际的资源使用费用。在专栏学习过程中，部署 IAM 应用期间产生的资源使用费用其实是真的很低的，所以推荐使用这种方式来部署 IAM 应用。

有了 Kubernetes 集群，就可以直接通过以下命令来部署整个 IAM 应用：

```
1 $ kubectl -n iam apply -f ${IAM_ROOT}/deployments/iam.yaml
```

复制代码

应用部署起来之后，我们可以登陆到 iamctl-xxxxxxxxxx-xxxxxPod，并执行以下命令来测试整个 IAM 应用是否被成功部署：

```
1 # cd /opt/iam/scripts/install
2 # ./test.sh iam::test::smoke
```

课后练习

1. 思考下，如何将 MariaDB、MongoDB、Redis 实现容器化？
2. 思考下，如何更相信 IAM 应用中的 iam-apiserver 服务，试着更新这个服务。

欢迎你在留言区与我交流讨论，我们下一讲见。

分享给需要的人，Ta订阅后你可得 **24 元现金奖励**

 赞 1

 提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 47 | 如何编写Kubernetes资源定义文件？

下一篇 特别放送 | 给你一份清晰、可直接套用的Go编码规范

更多学习推荐

175 道 Go 工程师 大厂常考面试题

限量免费领取 



精选留言

写留言

由作者筛选后的优质留言将会公开显示，欢迎踊跃留言。